Themen für die mündliche RDP 2019 als PDF-Download

Themenpool Matura 2019 - 8BI

- 1) Zahlensysteme
- 2) Schaltalgebra
- 3) Audiobearbeitung
- 4) Bild- und Videobearbeitung
- 5) Betriebssysteme Windows und Linux
- 6) Hardware - Grundlagen, Vertiefung und Entwicklung
- 7) Textverarbeitung – MS Word
- 8) Tabellenkalkulation – MS Excel
- 9) Datenbanksysteme - ER-Modell, Relationenmodell, SQL
- 10) Satzsystem LaTeX
- 11) HTML - Grundlagen und Vertiefung - CSS
- 12) PHP – Grundlagen und Vertiefung
- 13) PHP/MySQL Datenbanken
- 14) Grundlagen des Programmierens in C++
- 15) Grundlagen des visuellen (objektorientierten) Programmierens
- 16) Strukturen – Verkettete Listen, binäre Bäume, Rekursionen
- 17) Contentmanagementsysteme
- 18) Netzwerke - Filius
- mündliche RDP - Umfeldfragen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura

Last update: **2019/05/14 16:04**

1) Zahlensysteme in der Informatik

- Kurzüberblick - Geschichtliche Entwicklung des Zählens
- Skriptum zur Einführung
- Informationseinheiten in der Informatik

- 1.1) Dualsystem
- 1.2) Hexadezimalsystem
- 1.3) Oktalsystem
- 1.4) Umrechnungen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:01

Last update: **2019/05/14 16:06**

1.1) Dualsystem

Das **duale Zahlensystem - auch Dualsystem oder Binärsystem** genannt - besteht aus **2 Ziffern**, gekennzeichnet durch 0 und 1. Man benötigt dieses Zahlensystem in der Informatik, da sich mit technischen Bauteilen sehr leicht die Zustände AN und AUS erzeugen lassen können. Diese Zahlen können entsprechend unserem „normalen“ Dezimalsystem verwendet werden. Man kann sie addieren, subtrahieren, multiplizieren und dividieren. Da sie sich also kaum vom „normalen“ Rechnen unterscheiden, eignen sie sich hervorragend, um in der EDV eingesetzt zu werden.

Zählen im Dualsystem

Auch hier beginnen wir mit 0 und zählen dann 1. Leider haben wir nur 2 Zahlen, also gehen uns hier die Zahlen schnell aus. Wir machen es jetzt aber genau wie im Dezimalsystem und nehmen eine Stelle dazu. Nach 0 und 1 kommen dann also 10 und 11. Wieder reichen die Stellen nicht! Also noch eine dazu: 100, 101, 110, 111, usw.

Zählen von 0 bis 15 im Dezimal- und Dualsystem

Dezimal	Dual
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Eine andere Schreibweise

Man kann Zahlen auch anhand ihrer Basis darstellen. Im Dezimalsystem haben wir 10 Zahlen zur Verfügung, von 0 bis 9. Mit 2 Stellen können wir also 10 * 10 = 100 Zahlen darstellen. 100 Zahlen? Aber 100 hat doch drei Stellen! Dieser Einwand stimmt. Da wir jedoch mit der Zahl 0 beginnen, ist 0 die 1. Zahl, 1 die 2. Zahl, … 98 die 99. Zahl und 99 die 100. Zahl.

Mit 3 Stellen können wir 10 * 10 * 10 = 1000 Zahlen darstellen. Jede Stelle entspricht einer 10-er Potenz.

An einem einfachen Beispiel versuche ich diesen Sachverhalt zu erklären.

Wir nehmen dazu die Zahl 372 und schreiben sie als kleine Rechnung auf:

$372 = 3*100 + 7*10 + 2*1$.

Das kann man jetzt noch anders darstellen als:

$3*10^2 + 7*10^1 + 2*10^0$.

Auf diese Weise kann man jetzt alle anderen Zahlen auch darstellen:

$6574 = 6*10^3 + 5*10^2 + 7*10^1 + 4*10^0$
$12032 = 1*10^4 + 2*10^3 + 0*10^2 + 3*10^1 + 2*10^0$

Verwendung der Potenzschreibweise bei binären Zahlen

Wendet man die Potenzschreibweise bei binären Zahlen an, so muss man eine andere Basis wählen. Es gibt ja nur 2 verschiedene Ziffern, 0 und 1. Also nehmen wir als Basis 2.

Die Zahl 1011 schreibt sich dann als

$1*2^3 + 0*2^2 + 1*2^1 + 1*2^0$

Die Umrechnung von dezimalen in binäre Zahlen

Bei der Umrechnung der Dezimalzahlen verwenden wir die „Division mit Rest“ aus der Grundschule. Wir teilen die Zahl solange durch 2, bis als Ergebnis 0 herauskommt und merken uns dabei den Rest. Als Beispiel sollen die Zahlen 13 und 14 dienen.

13 / 2 = 6 Rest 1
6 / 2 = 3 Rest 0
3 / 2 = 1 Rest 1
1 / 2 = 0 Rest 1

Die Reste von unten nach oben aneinander gereiht ergeben dann die Dualzahl 1101.

14 / 2 = 7 Rest 0
7 / 2 = 3 Rest 1
3 / 2 = 1 Rest 1
1 / 2 = 0 Rest 1

Hieraus ergibt sich dann die Dualzahl 1110.

Addition von Dualzahlen

Einige erinnern sich vielleicht noch an die Addition von Zahlen wie wir sie in der Grundschule gelernt haben. Wir schreiben die Zahlen untereinander und addieren sie Stelle für Stelle. Dabei beginnen wir mit der letzten Stelle und arbeiten uns langsam nach vorne durch. Die gleiche Vorgehensweise

benutzen wir nun auch wenn wir Dualzahlen addieren wollen.

Die Addition funktioniert wie bei der Addition von Dezimalzahlen:

```
 0111
+0100
 ====
 1011
```

Addition mit Überlauf:

```
 1111
+0100
 ====
10011
```

Es gelten die Regeln 0+0=0, 1+0=1, 0+1=1, 1+1=0 Übertrag 1. Im Prinzip also nichts neues. Addiert man im Dezimalsystem 2 Zahlen so kommt bei 5+5 auch 0 Übertrag 1 heraus. Der Übertrag wird bei beiden System jeweils voran gestellt. Also gilt im Dualsystem 1+1=10.

Vorsicht Überlauf!

Die Addition ist im Prinzip problemlos. Es gibt allerdings einen Haken an der Sache: Die Addition funktioniert nur innerhalb eines bestimmten Wertebereiches. Woran liegt das? In der Realität können wir beliebig grosse Zahlen darstellen. Das geht leider nicht in der Informatik. Wir haben nur einen begrenzten Raum bzw. Speicherplatz zur Verfügung. Wir müssen aus diesem Grund den Speicherbereich einschränken (siehe was ist ein Byte), in unserem Beispiel nehmen wir als Speicherplatz ein Byte. Normalerweise verwendet man zur Addition von ganzen Zahlen, einen deutlich größeren Wertebereich, aber um einen überschaubaren Rahmen zu haben, begrenzen wir uns absichtlich auf ein Byte. Unsere größte darstellbare Zahl ist die 11111111, also dezimal 255. Was passiert nun, wenn wir eine 00000001, also 1, addieren? Das verrückte ist: es kommt 00000000, also 0 heraus. Da bekanntlich 1+1=0 Übertrag 1 gibt, bekommt man als Ergebnis in Wirklichkeit nicht 00000000 sondern 00000000 Übertrag 1. Dieser letzte Übertrag kann jedoch nicht mehr gespeichert werden und wird deshalb einfach ersatzlos gestrichen. Es ergibt sich daraus jedoch auch eine gewisse Logik. Durch meinen beschränkten Wertebereich komme ich irgendwann an meine obere Grenze. Bei der Addition von 1 fängt dann jedoch der Wertebereich wieder von vorne an, ich bin jetzt an der unteren Grenze, man durchläuft dann wieder den Bereich bis zur oberen Grenze, usw. Wenn wir also immer und immer wieder 1 addieren zählen wir unendlich oft von 0 bis 255, dann wieder 0 bis 255 usw.

Subktraktion von Dualzahlen

Drei Schritte zu Subtraktion

Hier kommen wir mit unserer normalen Schulmathematik nicht mehr weiter. Bevor wir uns mit dem komplizierten „Warum ist das denn so?“ beschäftigen, merken wir uns erst einmal den Mechanismus. Die Subtraktion von binären Zahlen wird durch die **Addition des Zweierkomplementes**

durchgeführt. Zur Erklärung beginnen wir im ersten Schritt mit dem **Einerkomplement**, dann schauen wir im zweiten Schritt was das **Zweierkomplement** ist und dann kommen wir im letzten Schritt zur **Subtraktion**.

Das Einerkomplement

Was ist das Komplement von Dualzahlen? Man bildet das sogenannte Einerkomplement, indem man jede Zahl durch ihr Gegenteil ersetzt, also die 0 durch die 1 und die 1 durch die 0.

01011010 wird zu 10100101 11101101 wird zu 00010010

Das Zweierkomplement

Das Zweierkomplement entspricht dem Einerkomplement, nur wird zusätzlich noch 00000001 addiert.

01011010 wird im Einerkomplement zu 10100101 im Zweierkomplement zu 10100110 11101101 wird im Einerkomplement zu 00010010 im Zweierkomplement zu 00010011

Die Subtraktion von Dualzahlen

Der Satz lautet: Die Subtraktion von 2 Zahlen erfolgt durch die Addition des Zweierkomplementes. Als konkretes Beispiel nehmen wir dazu die Rechnung 14-9=5.

!!WICHTIG:!!
Die restlichen Ziffern müssen immer mit 0 aufgefüllt werden.

9 ist im Dualsystem 00001001.

Das Einerkomplement zu 00001001 ist 11110110.

Das Zweierkomplement 11110111.

Dies addieren wir nun zu 14 also 00001110.

```
 00001110
+11110111
 ========
 00000101
```

Auch hier wäre die richtige Zahl eigentlich 00000101 Übertrag 1, da wir den Übertrag jedoch nicht speichern können, bleiben wir bei 00000101 was ja der Dezimalzahl 5 entspricht.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:01:01_01

Last update: **2019/05/14 16:07**

1.2) Hexadezimalsystem

Besonders **wichtig** ist in der **Informatik und Digitaltechnik** neben dem Binärsystem auch das **Hexadezimalsystem (Sedezimalsystem)**. Das Hexadezimalsystem verwendet die **Basis 16**, d.h. es gibt **16 verschiedene Ziffern**, **0 bis 9** und zusätzlich die Buchstaben **A bis F** (sog. Zahlzeichen; können auch als klein geschrieben werden: a-f).

Mit dem Hexadezimalsystem können auf einfachere und kürzere Weise Binärzahlen notiert werden. Mit einer 4-stelligen Binärzahl (auch als Halbbyte oder Nibble bezeichnet) lassen sich 16 (24 = 16) verschiedene Zahlen darstellen, und zwar 0 bis 15 (die Null zählt mit!). Da das Hexadezimalsystem die Basis 16 (= 24) verwendet, reicht eine (!) Hexadezimalzahl aus, um vier Bits (Binärziffern) darzustellen. Mit zwei Hexadezimalzahlen kann ein Byte (8 Bits) angeschrieben werden.

Gegenüberstellung Hexadezimal-, Binär- und Dezimalsystem

Hex	Binär	Dezimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Um eindeutig darauf hinzuweisen, dass es sich um eine Hexadezimalzahl handelt, kann ebenso wie in anderen Zahlensystemen die Basis tiefgestellt dazu geschrieben werden, z.B. $\mathbf{3F_{16}}$ (= 63_{10} dezimal) oder $\mathbf{93_{16}}$ (= 147_{10} dezimal). Es sind aber auch andere Schreibweisen üblich:

a) Vorangestelltes **0x (Prefix), z.B. 0x93**. Diese Notation wird in Programmiersprachen mit C-ähnlicher-Syntax verwendet.

b) Nachgestelltes **h** (Postfix), z.B. **93h**. Letztere Schreibweise ist besonders in der Technik gebräuchlich.

Umrechnung vom Dezimal- ins Hexadezimalsystem

Die Umrechnung funktioniert ähnlich der Umrechnung von Dezimal- zu Binärzahlen (s.o.). Nun muss aber, statt durch 2, durch 16 dividiert werden. Die Reste werden genauso von rechts nach links angeschrieben und geben, wenn das Ergebnis der Ganzzahlendivision 0 ist, das Endergebnis.

Beispiel: Die Dezimalzahl **304_{10}** soll in eine Hexadezimalzahl umgewandelt werden.

```
304 / 16 = 19 => 0 Rest
 19 / 16 =  1 => 3 Rest
  1 / 16 =  0 => 1 Rest
```

Für das Endergebnis werden jetzt die Reste **von unten nach oben gelesen**.
Somit ergibt sich ein Endergebnis von 130_{16}, das entspricht der Dezimalzahl **304_{10}**.

Umrechnung vom Hexadezimal- ins Dezimalsystem

Die Umrechnung vom Hexadezimal- ins Dezimalsystem kann genauso wie oben von Binär→Dezimal demonstriert, erfolgen. Die einzelnen Ziffern werden mit dem jeweiligen Stellenwert (16^n, wobei n = 0, 1, 2, ...) multipliziert und die jeweiligen Ergebnisse aufsummiert. Das folgende Beispiel demonstriert dies anhand der Hexadezimalzahl 130_{16}:

```
0 * 16^0  =    0
3 * 16^1  =   48
1 * 16^2  =  256
---------------
          =  304
```

Als Ergebnis erhalten wir 304 dezimal, womit die Probe - zur vorigen Rechnung in die umgekehrte Richtung - erfolgreich war. 304_{10} entspricht 130_{16}. Diese Antwort hätte in der Praxis natürlich auch ein wissenschaftlicher Taschenrechner geliefert. 😃 Es reicht dazu sogar der **Windows-Rechner** (den Sie nur auf die wissenschaftliche Ansicht umstellen müssen) oder unter Linux Programme wie z.B. KCalc.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:01:01_02

Last update: **2019/05/14 16:07**

1.3) Oktalsystem

Das Oktalsystem, auch Achtersystem genannt, verwendet die **Basis 8 (acht)**. Um Zahlen darzustellen, stehen die **Ziffern 0 bis 7** zur Verfügung. Die Bedeutung in der Informatik/Digitaltechnik ergibt sich dadurch, dass sich mit einer **Oktalzahl drei Bits** darstellen lassen. 23 ist 8, somit lassen sich mit 3 Bits 8 verschiedene Möglichkeiten darstellen. Eine Oktalzahl reicht, um diese Information wiederzugeben.

Das Oktalsystem wird hier insbesondere deshalb erwähnt, weil in vielen Programmiersprachen Zahlen auch in Oktalform angegeben werden können. Meist, z.B. in PHP, wird dazu eine 0 (Null) vorangestellt, z.B. 077 für 77_8 (= 63_{10}).

Umrechnungen erfolgen genauso wie beim Hexadezimalsystem gezeigt. Um die Oktalzahl auszurechnen, die einer best. Dezimalzahl entspricht, dividieren Sie die Dezimalzahl fortlaufend durch 8 und schreiben die Reste von rechts nach links an. In umgekehrter Richtung - von Oktal nach Dezimal - multiplizieren Sie die einzelnen Ziffern mit dem Stellenwert (8^n für n = 0, 1, 2, ...) und addieren die Teilergebnisse.

Umrechnung vom Dezimal- ins Oktalsystem

Die Umrechnung funktioniert ähnlich der Umrechnung von Dezimal- zu Binärzahlen (s.o.). Nun muss aber, statt durch 2, durch 8 dividiert werden. Die Reste werden genauso von rechts nach links angeschrieben und geben, wenn das Ergebnis der Ganzzahlendivision 0 ist, das Endergebnis.

Beispiel: Die Dezimalzahl **304_{10}** soll in eine Hexadezimalzahl umgewandelt werden.

```
304 / 8 = 38 => 0 Rest
 38 / 8 =  4 => 6 Rest
  4 / 8 =  0 => 4 Rest
```

Für das Endergebnis werden jetzt die Reste **von unten nach oben gelesen**.
Somit ergibt sich ein Endergebnis von 460_8, das entspricht der Dezimalzahl **304_{10}**.

Umrechnung vom Oktal- ins Dezimalsystem

Die Umrechnung vom Oktal- ins Dezimalsystem kann genauso wie oben von Binär→Dezimal demonstriert, erfolgen. Die einzelnen Ziffern werden mit dem jeweiligen Stellenwert (8^n, wobei n = 0, 1, 2, ...) multipliziert und die jeweiligen Ergebnisse aufsummiert. Das folgende Beispiel demonstriert dies anhand der Oktalzahl 460_8:

```
0 * 8^0  =    0
6 * 8^1  =   48
4 * 8^2  =  256
---------------
         =  304
```

Als Ergebnis erhalten wir 304 dezimal, womit die Probe - zur vorigen Rechnung in die umgekehrte Richtung - erfolgreich war. 304_{10} entspricht 460_{8}. Diese Antwort hätte in der Praxis natürlich auch ein wissenschaftlicher Taschenrechner geliefert. Es reicht dazu sogar der **Windows-Rechner** (den Sie nur auf die wissenschaftliche Ansicht umstellen müssen) oder unter Linux Programme wie z.B. KCalc.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:01:01_03

Last update: **2019/05/14 16:07**

1.4) Umrechnungen

Prinzipiell kann man jede Zahl von einem Zahlensystem in ein anderes Zahlensystem umrechnen. Dazu muss man normalerweise die Zahl immer zuerst in das Dezimalsystem und dann in das Ziel-Zahlensystem umrechnen. Ausnahmen gibt es bei Umrechnungen vom Binärsystem in ein anderes Zahlensystem, hier braucht man nicht unbedingt das Dezimalsystem.

Binärsystem ⇔ Oktalsystem

Binärsystem ⇒ Oktalsystem

Wir wissen ja bereits, dass eine Ziffer im Oktalsystem die Ziffern 0-7 annehmen kann und dass man 3 Bits im Dualsystem braucht um 8 verschiedene Zahlenwerte darzustellen.

⇒ Immer 3 Stellen im Binärsystem ergeben eine Ziffer/Stelle im Oktalsystem

Beispiel

Umwandlung der Zahl 010110011111_2 ins Oktalsystem:

```
010 110 011 111
 ⇓   ⇓   ⇓   ⇓
 2   6   3   7
```

Die Zahl 010110011111_2 entspricht somit der Zahl 2637_8.

Oktalsystem ⇒ Binärsystem

Selbiges gilt auch für die Umrechnung vom Oktalsystem in das Binärsystem.

⇒ Eine Stelle im Oktalsystem entspricht 3 Stellen im Binärsytem

Beispiel

Umwandlung der Zahl 672_8 ins Binärsystem:

```
 6   7   2
 ⇓   ⇓   ⇓
110 111 010
```

Die Zahl 672_8 entspricht somit der Zahl 110111010_2.

Binärsystem ⇔ Hexadezimalsystem

Binärsystem ⇒ Hexadezimalsystem

Wir wissen ja bereits, dass eine Ziffer im Hexadezimalsystem die Werte 0-15 annehmen kann und dass man 4 Bits im Dualsystem braucht um 16 verschiedene Zahlenwerte darzustellen.

⇒ Immer 4 Stellen im Binärsystem ergeben eine Ziffer/Stelle im Oktalsystem

Beispiel

Umwandlung der Zahl 010110011111_2 ins Hexadezimalsystem:

```
0101 1001 1111
 ⇓    ⇓    ⇓
 5    9    F
```

Die Zahl 010110011111_2 entspricht somit der Zahl $59F_{16}$.

Hexadezimalsystem ⇒ Binärsystem

Selbiges gilt auch für die Umrechnung vom Hexadezimalsystem in das Binärsystem.

⇒ Eine Stelle im Hexadezimalsystem entspricht 4 Stellen im Binärsytem

Beispiel

Umwandlung der Zahl 672_{16} ins Binärsystem:

```
 6    7    2
 ⇓    ⇓    ⇓
0110 0111 0010
```

Die Zahl 672_{16} entspricht somit der Zahl 11001110010_2.

Oktalsystem ⇒ Hexadezimalsystem

Will man vom Oktalsystem ins Hexadezimalsystem umrechnen, rechnet man am besten zuerst ins Binärsystem und dann ins Hexadezimalsystem um.

Hexadezimalsystem ⇒ Oktalsystem

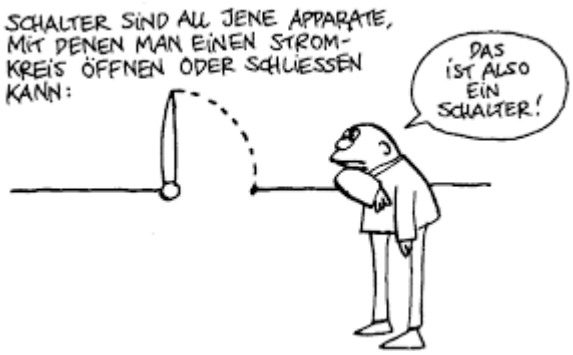
Will man vom Hexadezimalsystem ins Oktalsystem umrechnen, rechnet man am besten zuerst ins Binärsystem und dann ins Oktalsystem um.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:01:01_04

Last update: **2019/06/12 08:35**

2) Schaltalgebra



- 2.1) Grundlagen
- 2.2) Grundschaltungen
- 2.3) Zusammengesetzte Schaltungen
- 2.4) Gesetze der Schaltalgebra
- 2.5) Digitale Rechenschaltungen
- 2.6) Übungen
- 2.7) Übungen
- 2.8) Übungen

Simulator für logische Schaltungen

https://elektroniktutor.de/swfdatei/gatter.swf

Das Adobe Flash Plugin wird benötigt, um diesen Inhalt anzuzeigen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:02

Last update: **2019/05/14 16:07**

2.1) Grundlagen

In der Digitaltechnik sind die Eingangsvariablen so miteinander zu verknüpfen, dass die Ausgangsvariable einen definierten Zustand annimmt, um damit einen Prozess zu steuern. So soll ein Fahrstuhl nur dann fahren, wenn eine Zieletage gewählt wurde, in der er zurzeit nicht steht, seine Tür vollständig geschlossen ist und von ihr nichts eingeklemmt wurde und die Kabine nicht überlastet ist. Man kann durch Kombinieren der beschriebenen digitalen Gatter und Ausprobieren bei einfacheren Aufgaben die eine oder andere Lösung finden. Das eigentliche Ziel ist es, eine wirtschaftliche Digitalschaltung zu entwickeln, die mit einem Minimum gleicher Gattertypen zum Ziel führt.

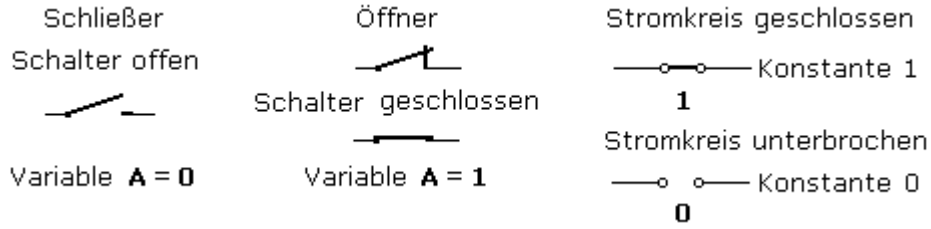
Der englische Mathematiker Georg Boole entwickelte eine Mengenalgebra, die in angepasster Weise als Boolesche Schaltalgebra bei der Problemlösung hilfreich ist. In der Schaltalgebra gibt es Variablen und Konstanten. Die binäre Digitaltechnik kommt mit zwei definierten logischen Zuständen, der 0 und 1 aus.

Konstante

In der Schaltalgebra gibt es nur die zwei konstanten Größen 0 und 1. In der elektronischen Schaltung entspricht eine dauerhaft geschaltete Leitung der Konstanten mit dem Zustand 1. Die dauerhafte Unterbrechung eines Stromkreises steht für die Konstante mit dem Wert 0.

Variable

Veränderbare, schaltbare Eingangsgrößen und davon abhängige veränderliche Ausgangswerte werden als Variable bezeichnet. In der binären Digitaltechnik nehmen sie entweder den Zustand 0 oder 1 an. In elektronischen Schaltungen können sie mit einem Schalter verglichen werden. Der geöffnete Schalter entspricht einer Variablen mit dem Wert 0. Bei geschlossenem Schalter nimmt die Variable den Wert 1 an.



Wir wissen, dass alle Grundrechnungsarten auf eine Addition zurückgeführt werden können. Deshalb ist das Ziel dieses Kapitels eine Maschine simulieren zu können, die addieren kann.

Informationseinheiten

BIT

Die Ziffern 0 und 1 werden physikalischen Zuständen zugeordnet:

```
0   Strom fließt nicht
1   Strom fließt
```

0 und 1 werden in der Informatik mit Bit bezeichnet. Ein Bit stellt die kleinste Informationseinheit dar (nur 2 Zustände – 0 und 1).

1 Bit ⇒ 2^1 ⇒ 2 Möglichkeiten
2 Bit ⇒ 2^2 ⇒ 4 Möglichkeiten
3 Bit ⇒ 2^3 ⇒ 8 Möglichkeiten
8 Bit = 1 Byte ⇒ 2^8 ⇒ 256 Möglichkeiten

BYTES

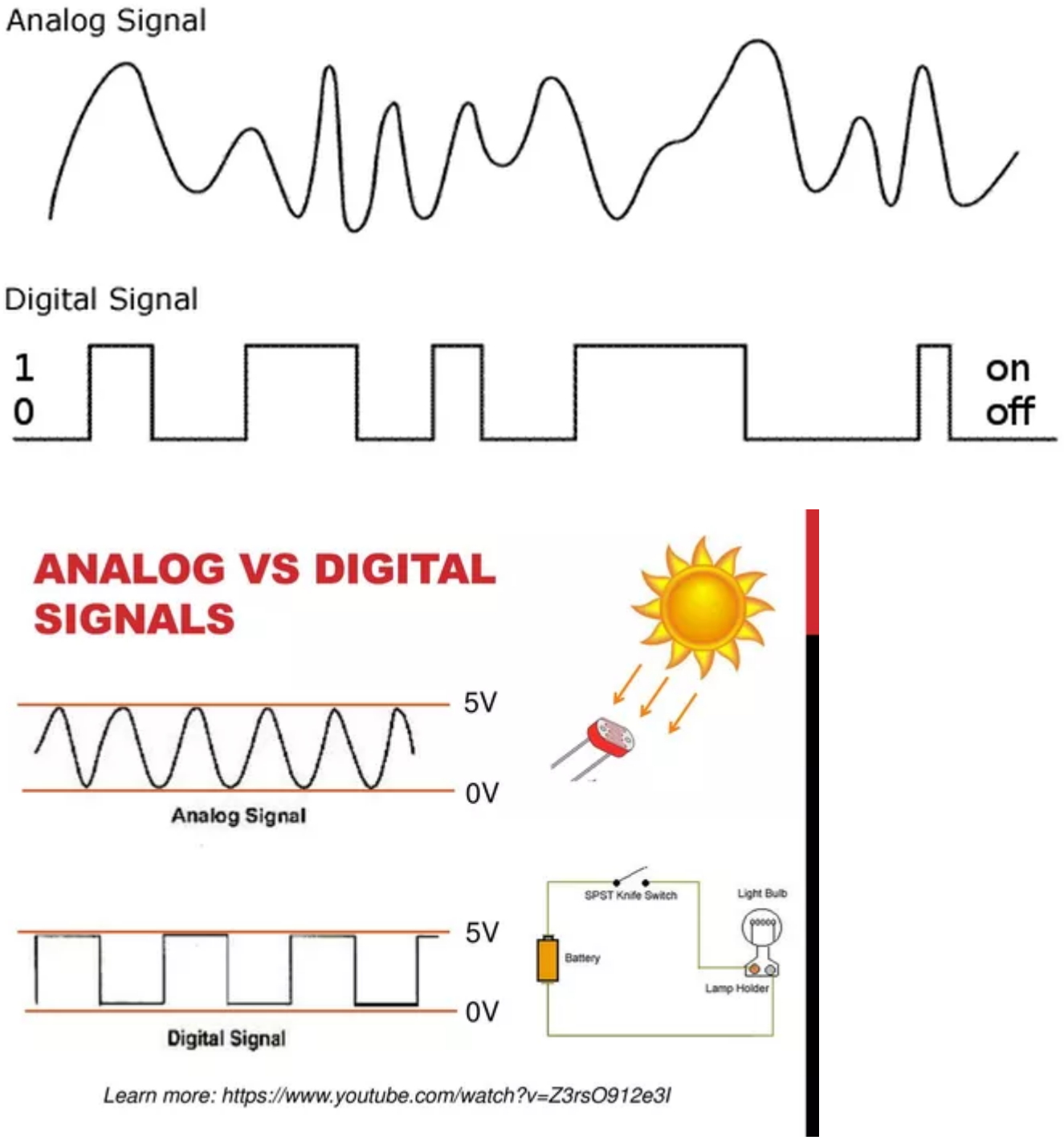
8 Bits sind ein Byte!

1 KB (Kilobyte) = 2^{10} = 1024 Byte
1 MB (Megabyte) = 2^{10} KB = 2^{10} x 2^{10} Byte = 2^{20} Byte = 1048576 Byte
1 GB (Gigabyte) = 2^{30} Byte
1 TB (Terabyte) = 2^{40} Byte

Darstellung von 0 und 1

Ein Computer verarbeitet systembedingt Digitalsignale, da nur ausgewertet werden kann, ob eine Spannung anliegt oder nicht bzw. innerhalb eines Definitionsbereichs einen Wert über- oder unterschreitet. Diese zwei Zustände werden auch häufig bezeichnet als:

HIGH und **LOW**
TRUE und **FALSE**
AN und **AUS**
1 und **0**



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:02:2_01

Last update: **2019/06/12 09:31**

2.2) Grundschaltungen

UND-Verknüpfung (=Serienschaltung)

Bei der UND-Verknüpfung führt der Ausgang das Signal 1, wenn an beiden Eingängen (a und b) das Signal 1 anliegt.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)	Zeichen (Mathematik)
UND	AND	Konjunktion	X=A∧B	X=A*B

Schaltwerttabelle

a	b	a∧b
0	0	0
0	1	0
1	0	0
1	1	1

Schaltfunktion

f(a,b)=a∧b

Schaltsymbol

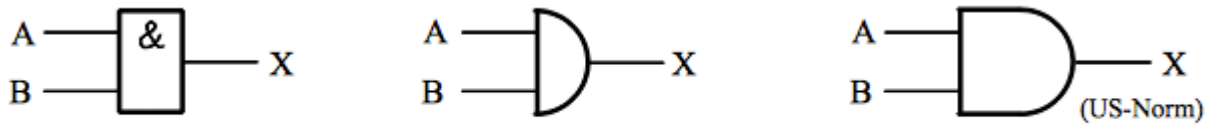


Bild 5-5: Schaltzeichen für UND-Gatter

elektronische Schaltung

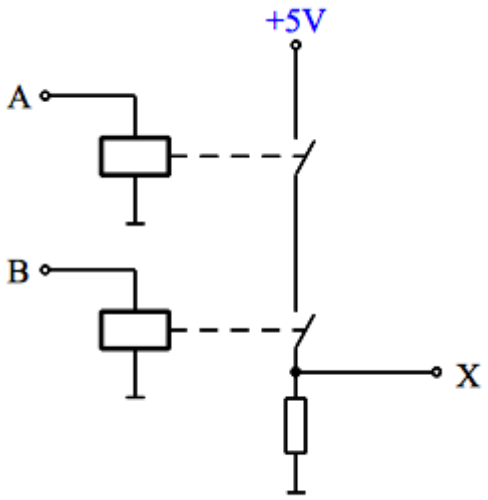


Bild 5-4: UND-Verknüpfung mit Relais

ODER-Verknüpfung (=Parallelschaltung)

Bei der ODER-Verknüpfung führt der Ausgang das Signal 1, wenn an einem der beiden Eingänge das Signal 1 anliegt.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)	Zeichen (Mathematik)
ODER	OR	Disjunktion	X=A∨B	X=A+B

Schaltwerttabelle

a	b	avb
0	0	0
0	1	1
1	0	1
1	1	1

Schaltfunktion

f(a,b)=a∨b

Schaltsymbol

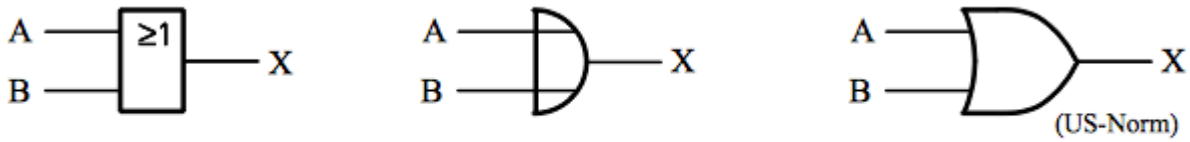


Bild 5-8: Schaltzeichen für ODER-Gatter

elektronische Schaltung

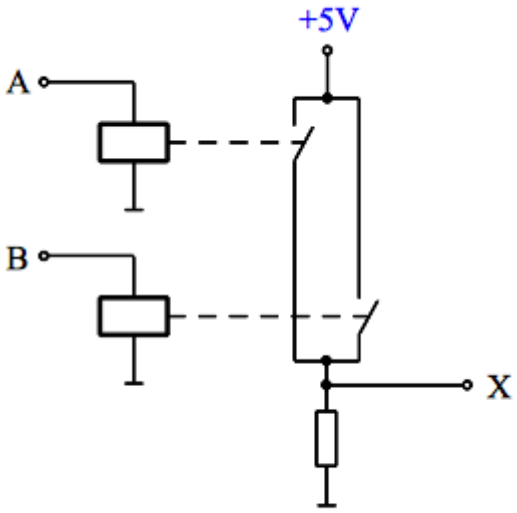


Bild 5-7: ODER-Verknüpfung mit Relais

NICHT-Verknüpfung (=NOT-Schaltung)

Bei einer NICHT-Verknüpfung wird der Ausgang logisch 1, wenn der Eingang logisch 0 ist, und umgekehrt.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)	Zeichen (Mathematik)
NICHT	NOT	Negation	X=¬A	X=Ā

Schaltwerttabelle

a	¬a
0	1
1	0

Schaltfunktion

f(a)=¬a

Schaltsymbol

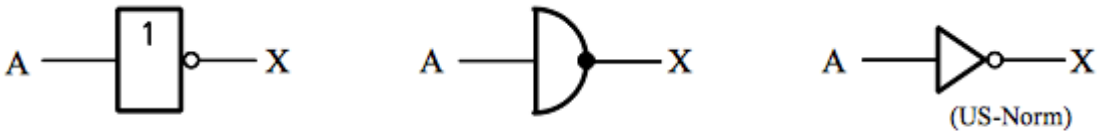


Bild 5-11: Schaltzeichen für NICHT-Gatter

elektronische Schaltung

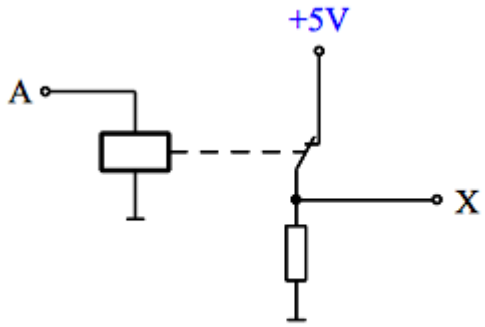


Bild 5-10: NICHT-Verknüpfung mit Relais

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:02:2_02

Last update: **2019/06/16 08:07**

2.3) Zusammengesetzte Schaltungen

Die **Grundverknüpfungen** werden in der Regel miteinander **kombiniert**, sodass die Logik der Schaltung verändert wird. Beispielsweise kann man eine UND- mit einer NICHT-Verknüpfung kombinieren und erhält dadurch eine sogenannte NAND-Verknüpfung (NICHT-UND). Da solch zusammengesetzte Schaltfunktionen sehr häufig verwendet werden, haben sie **eigene Schaltzeichen** bekommen. Mit den folgenden Schaltfunktionen werden die Grundverknüpfungen erweitert.

- NAND
- NOR
- XOR
- XNOR

NAND-Verknüpfung

Eine Kombination aus einem **UND-Gatter mit nachfolgendem NICHT-Gatter ergibt ein NAND-Gatter**. Die Ausgangsvariable Z des UND-Gatters wird durch das NICHT-Gatter negiert und erzeugt die Ausgangsvariable X des NAND-Glieds. Viele logische Funktionen lassen sich durch den Einsatz von NAND-Gattern lösen. Oft steigt dadurch die Anzahl der notwendigen Gatter, mit dem wirtschaftlichen Vorteil nur einen Gattertyp zu verwenden.

Der Ausgangszustand eines NAND-Glieds ergibt 1, wenn nicht alle Eingangszustände 1 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
NEGIERTES UND	NOT AND	Negative Konjunktion	$Z = \overline{A \wedge B}$

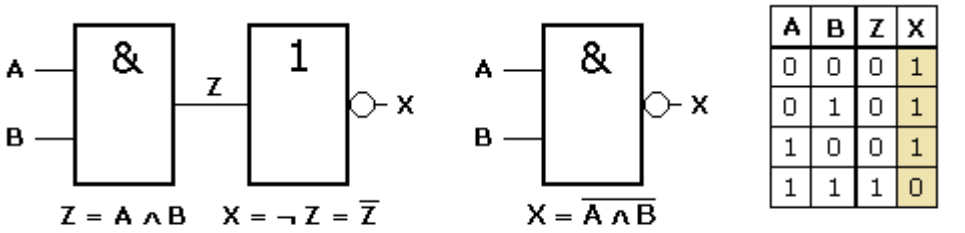
Schaltwerttabelle

a	b	a⊼b
0	0	1
0	1	1
1	0	1
1	1	0

Schaltfunktion

f(a,b)=a⊼b

Schaltsymbol



A	B	Z	X
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Wahrheitstabelle

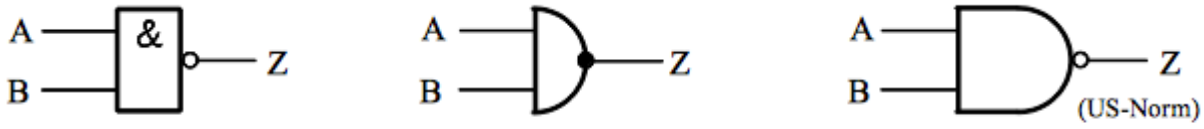


Bild 5-13: Schaltzeichen für NAND-Gatter

NOR-Verknüpfung

Eine Kombination aus einem **ODER-Gatter mit nachfolgendem NICHT-Gatter ergibt ein NOR-Gatter**. Die Ausgangsvariable Z des ODER-Gatters wird durch das NICHT-Gatter negiert und erzeugt die Ausgangsvariable X des NOR-Glieds. NOR-Glieder haben in logischen Schaltungen die gleiche wichtige Bedeutung wie NAND-Glieder.

Der Ausgangszustand eines NOR-Glieds ergibt 1, wenn alle Eingangszustände 0 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
NEGIERTES ODER	NOT OR	Negative Disjunktion	$Z = \overline{A \vee B}$

Schaltwerttabelle

a	b	a⊽b
0	0	1
0	1	0
1	0	0
1	1	0

Schaltfunktion

f(a,b)=a⊽b

Schaltsymbol

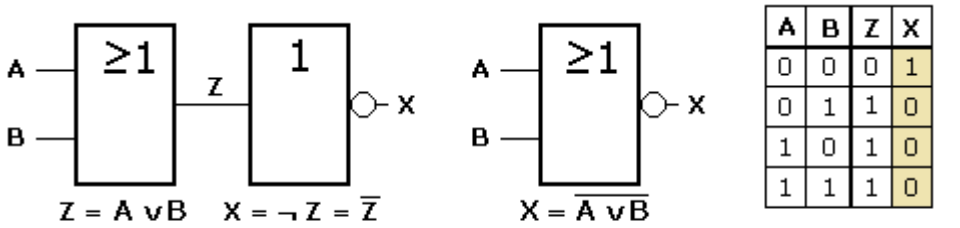




Bild 5-15: Schaltzeichen für NOR-Gatter

XOR-Verknüpfung

Die logische Verknüpfung des XOR-Gatters mit zwei Eingangsvariablen kann mit einem **'entweder - oder'** umschrieben werden. Die **Ausgangsvariable wird immer dann 'true' liefern, wenn die Eingangsvariablen unterschiedliche Zustände haben**. Die Wahrheitstabelle des XOR-Gatters entspricht dem ODER-Gatter mit dem Ausschluss gleicher Eingangszustände, also exklusiv der Äquivalenz. Dieses Verhalten wird als Antivalenz bezeichnet.

Der Ausgangszustand eines XOR-Glieds ergibt 1, wenn eine ungerade Zahl der Eingangszustände 1 und alle anderen Eingangszustände 0 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
Exklusives ODER (Antivalenz)	Exclusiv OR	Exklusive Disjunktion	Z=a⊻b

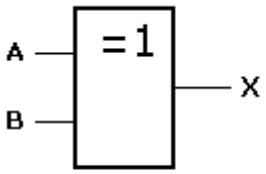
Schaltwerttabelle

a	b	a⊻b
0	0	0
0	1	1
1	0	1
1	1	0

Schaltfunktion

f(a,b)=a⊻b

Schaltsymbol



Exklusiv ODER
XOR - Antivalenz

$X = (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$
$X = A \underline{\vee} B$
$X = A \oplus B$

Funktionsgleichung

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Wahrheitstabelle

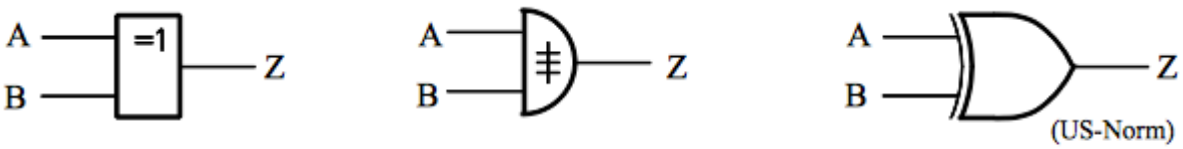


Bild 5-21: Schaltzeichen für ANTIVALENZ-Gatter

XNOR-Verknüpfung

Die Bezeichnung **Äquivalenz bedeutet Gleichwertigkeit**. Beim XNOR-Gatter mit zwei Eingangsvariablen ist der **Zustand der Ausgangsvariable 1, wenn beide Eingangsvariablen den gleichen Zustand 0 oder 1 haben**. Die Funktion kann durch eine Schaltung mit vier NOR-Gattern erreicht werden. Es sind zwei unterschiedliche Schaltzeichen zu finden.

Der Ausgangszustand eines XNOR-Glieds ergibt 1, wenn eine gerade Zahl der Eingangzustände 1 und alle anderen Eingangszustände 0 aufweisen oder alle 0 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
Exklusives NICHT ODER (Äquivalenz)	Exclusiv NOT OR	Biimplikation	Z=a≡b

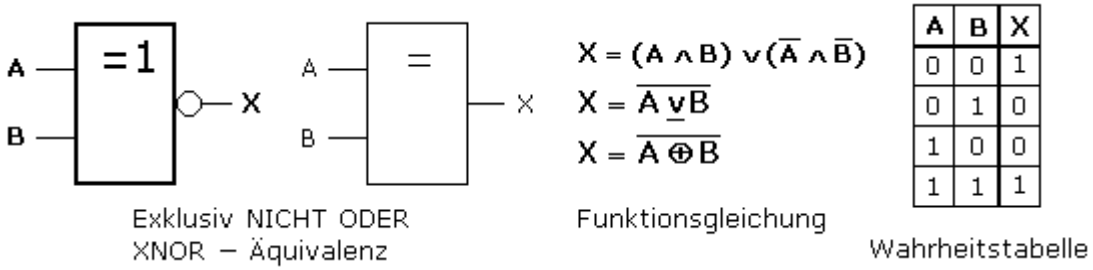
Schaltwerttabelle

a	b	a≡b
0	0	1
0	1	0
1	0	0
1	1	1

Schaltfunktion

f(a,b)=a≡b

Schaltsymbol



Exklusiv NICHT ODER
XNOR – Äquivalenz

Funktionsgleichung

Wahrheitstabelle

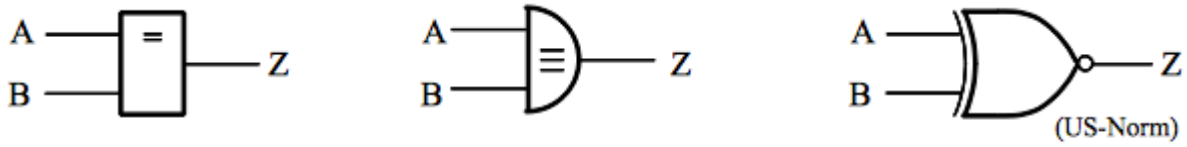


Bild 5-19: Schaltzeichen für ÄQUIVALENZ-Gatter

xor_nand.swf

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:02:2_03

Last update: **2019/05/14 16:08**

2.4) Gesetze der Schaltalgebra

Vorrang- und Klammerregel

Wie in der Algebra ist auch in der Digitaltechnik auf eine bestimmte Reihenfolge der Operationen zu beachten. Die Negation sollte vor der Konjunktion (UND) und diese vor der Disjunktion (ODER) ausgeführt werden. Das ist vergleichbar mit der Aussage, dass die Multiplikation Vorrang vor der Addition hat.

Bei mehreren Variablen und unterschiedlichen Verknüpfungen kann eine Klammersetzung notwendig sein. Beachtet man die Vorrangregel, kann man auf viele Klammern verzichten. Ein Setzen zeigt sogleich eindeutig, welche der Variablen wie zu verknüpfen ist. Bei ODER sollte immer geklammert werden, ebenfalls beim Anwenden der De Morganschen Gesetze auf NAND- und NOR-Verknüpfungen.

$X = A \vee B \wedge C$

nach der Vorrangregel gilt

$\boxed{X = A \vee (B \wedge C)}$

festgelegte Abfolge der Verknüpfungen

$\boxed{Z = (A \vee B) \wedge C}$

A	B	C	B∧C	A∨(B∧C)	A∨B	(A∨B)∧C
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	1	0
0	1	1	1	1	1	1
1	0	0	0	1	1	0
1	0	1	0	1	1	1
1	1	0	0	1	1	0
1	1	1	1	1	1	1

$A \vee (B \wedge C) \neq (A \vee B) \wedge C$

Bei drei Eingangsvariablen und binären Zuständen gibt es 2^3 = 8 Eingangskombinationen. Die Wahrheitstabelle zeigt bei definierter Klammersetzung oder Beachtung der Vorrangregel UND vor ODER unterschiedliche Ergebnisse.

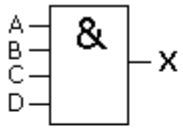
Kommutativgesetz

Das Kommutativ- oder Vertauschungsgesetz besagt, dass bei der UND sowie ODER Verknüpfung auch bei beliebiger Vertauschung der Reihenfolge der Eingangsvariablen das Ergebnis gleich bleibt.

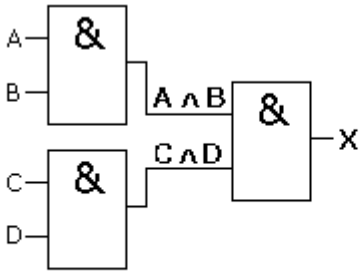
UND	$X = A \wedge B \wedge C = C \wedge A \wedge B = B \wedge C \wedge A$
ODER	$X = A \vee B \vee C = C \vee A \vee B = B \vee C \vee A$

Assoziativgesetz

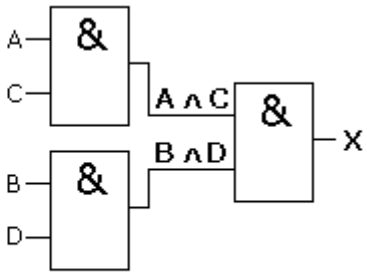
Das Assoziativ- oder Verbindungsgesetz besagt, dass bei einer UND- beziehungsweise ODER-Verknüpfung mit mehr als zwei Schaltvariablen die Verknüpfung auch stufenweise nacheinander in beliebiger Reihenfolge erfolgen kann.



A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1



A∧B	C∧D	X
0	0	0
0	1	0
1	0	0
1	1	1



A∧C	B∧D	X
0	0	0
0	1	0
1	0	0
1	1	1

$$X = A \wedge B \wedge C \wedge D = (A \wedge B) \wedge (C \wedge D) = (A \wedge C) \wedge (B \wedge D)$$

Beispiele zum Assoziativgesetz der UND-Verknüpfung

Anstelle des UND-Gatters mit vier Eingangsvariablen lassen sich zwei UND-Gatter mit zwei Eingängen verwenden. Die Variablen A, B, C, D werden einzeln beliebig mit den Eingängen verbunden. Die Ausgangsvariable der beiden UND-Gatter kann den Wert 0 oder 1 annehmen. Beide Ausgangsvariablen sind nochmals mit einem UND-Gatter zu verknüpfen, wobei sich wieder vier Eingangskombinationen ergeben. Nur wenn alle Eingangsvariablen 1 sind, wird der Ausgangszustand ebenfalls 1.

Das Assoziativgesetz gilt gleichermaßen für die ODER-Verknüpfung. Die Klammersetzung ist nicht notwendig und soll nur verschiedene Verteilungen besser erkennbar machen.

Distributivgesetz

Das Distributiv- oder Verteilungsgesetz wird zur Vereinfachung von Verknüpfungsgleichung angewendet. Es ist vergleichbar mit dem Ausmultiplizieren und Ausklammern von Variablen der normalen Algebra. Da die logische UND-Verknüpfung der algebraischen Multiplikation entspricht, während die ODER-Verknüpfung mit der Addition vergleichbar ist, gibt es zwei unterschiedliche Distributivgesetze.

Konjunktives Distributivgesetz

Eine Variable wird mit UND verknüpft und auf den Folgeausdruck verteilt. Die Vorgehensweise entspricht dem aus der Algebra bekannten Ausmultiplizieren eines Klammerausdrucks mit einem Faktor. Im umgekehrten Fall kann eine Variable, die mit mehreren anderen Variablen verknüpft ist, ausgeklammert werden. Das bedeutet für den Schaltungsaufbau eine Einsparung an Gattern.

$$X = A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$$

⇨ Ausmultiplizieren
⇦ Ausklammern

$X = A \wedge (A \vee B)$
$X = (A \wedge A) \vee (A \wedge B)$
$X = A \vee (A \wedge B) = A$
$X = A$

		UND	ODER	
A	B	A∧B	A	X
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	1	1

Beim Ausklammern wird die Variable mit ihrem Verknüpfungszeichen, hier UND, vor die Klammer gesetzt. Die in der Klammer stehenden Variablen werden mit dem zuvor zwischen den Klammern stehenden ODER verknüpft.

Disjunktives Distributivgesetz

Eine Variable kann durch ein vergleichbares Ausmultiplizieren auf andere Ausdrücke verteilt werden oder bei mehrfachem Auftreten ausgeklammert werden. Beim Ausklammern wird das ODER Verknüpfungszeichen mit der Variablen vor die Klammer geschrieben. disjunktives Distributivgesetz

$$X = A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$$

⇨ Ausmultiplizieren
⇦ Ausklammern

$X = A \vee (A \wedge B)$
$X = (A \vee A) \wedge (A \vee B)$
$X = A \wedge (A \vee B)$
$X = A$

		ODER	UND	
A	B	A∨B	A	X
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

De Morgansche Gesetze

Die boolesche Algebra wurde vom englischen Mathematiker De Morgan weiter entwickelt. Für die Schaltalgebra gibt es zwei De Morgansche Gesetze. Sie machen Aussagen zur Negation einer Verknüpfung und der Umkehr von Verknüpfungszeichen. Mit den De Morganschen Gesetzen lassen sich bei der Entwicklung von Digitalschaltungen Gatter gegeneinander austauschen, Schaltungen verkleinern oder mit nur einem Gattertyp verwirklichen. Das erste Gesetz ist für die NAND-Verknüpfung und das zweite Gesetz entsprechend für die NOR-Verknüpfung definiert.

$$X = \overline{A \wedge B} = \overline{A} \vee \overline{B}$$

1. De Morgansche Gesetz

			ODER		
A	B	$\overline{A \wedge B}$	$\overline{A}$	$\overline{B}$	X
0	0	1	1	1	1
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

$$X = \overline{A \vee B} = \overline{A} \wedge \overline{B}$$

2. De Morgansche Gesetz

			UND		
A	B	$\overline{A \vee B}$	$\overline{A}$	$\overline{B}$	X
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

Ist eine Verknüpfung insgesamt negiert, so ist ihre Ausgangsvariable negiert. Die Negation kann auf die Einzelglieder aufgeteilt werden, wobei aus der Grundverknüpfung UND ein ODER beziehungsweise aus ODER ein UND wird. Eine doppelte Negation hebt sich auf. Getrennte Negationsstriche über

Variablen bedeuten, dass die Eingangsvariablen negiert sind.

Die oben zum 1. De Morganschen Gesetz gezeigte Verknüpfung kann schaltungstechnisch mit zwei NICHT- und einem ODER-Gatter verwirklicht werden. Wie zu erkennen ist, folgt das gleiche Ergebnis mit nur einem NAND-Gatter. Für das 2. De Morgansche Gesetz gilt die entsprechende Aussage. Zwei NICHT- und ein UND-Gatter reduzieren sich auf den Einsatz eines NOR-Gatters. Mit der doppelten Negation und den De Morganschen Gesetzen kann bei einer gegebenen Funktionsgleichung auf das Bestehen einer derartige Vereinfachung geprüft werden. Treten dabei mehrfache Negationen auf, so lassen sie sich von innen nach außen auflösen.

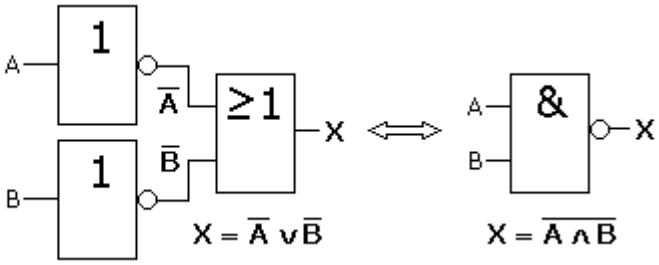
$X = \overline{A} \vee \overline{B}$

$\overline{\overline{X}} = \overline{\overline{A} \vee \overline{B}}$ $\quad \overline{v} \rightarrow \wedge$

$\overline{\overline{X}} = \overline{\overline{A} \wedge \overline{B}}$ $\quad \overline{\overline{A}} = A$

$X = \overline{A \wedge B}$

$\boxed{X = \overline{A \wedge B} = \overline{A} \vee \overline{B}}$



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:02:2_04

Last update: **2019/05/14 16:08**

2.5) Digitale Rechenschaltungen

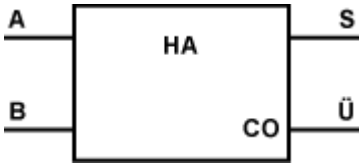
Aus logischen Verknüpfungen lassen sich digitale Schaltungen zusammenbauen, mit denen man Rechenvorgänge durchführen kann. Das heißt, diese Schaltungen haben zwischen ihren Eingängen eine Kombination aus logischen Verknüpfungen, die einem Rechenvorgang entspricht. In der Digitaltechnik kennt man Rechenschaltungen hauptsächlich für das duale Zahlensystem und den BCD-Code. Im Prinzip kann für jedes Zahlensystem eine Rechenschaltung aufgebaut werden.

Einige Rechenschaltungen der Digitaltechnik

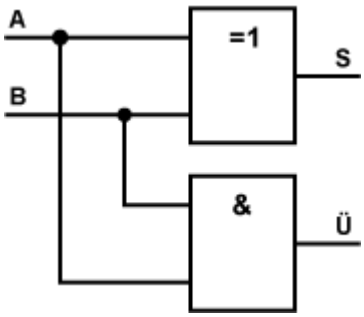
- Addiererschaltungen
- Subtrahiererschaltungen
- Addier-Subtrahier-Werke
- Multiplikationsschaltungen
- Arithmetisch-logische Einheit (ALU)

Stellvertretend für alle Rechenschaltungen dienen die folgenden Ausführungen zum Halbaddierer und dem Volladdierer.

Halbaddierer



Ein Halbaddierer ist die einfachste Rechenschaltung und kann zwei einstellige Dualziffern addieren. Der Eingang A des Halbaddierers ist der Summand A, der Eingang B ist der Summand B. Die Schaltung hat zwei Ausgänge. Der Ausgang S als Summenausgang (2^0) und der Ausgang Ü als Übertrag (2^1) für die nächsthöhere Stelle im dualen Zahlensystem.



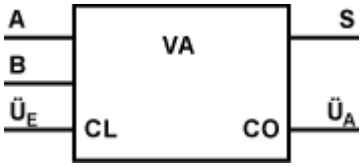
B	A	Ü	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Der Halbaddierer ist eine Schaltung aus den Verknüpfungsgliedern XOR und UND. Das XOR ist die Addier-Verknüpfung. Das UND stellt fest, ob ein Übertrag für die nächsthöhere Stelle vorgenommen

werden muss. Aus der Tabelle ist ersichtlich, dass das Ergebnis aus der Spalte Summe (S) einer Exklusiv-ODER-Verknüpfung (Antivalenz, XOR) entspricht. Das Ergebnis der Spalte Übertrag (Ü) entspricht einer UND-Verknüpfung. Die so entstandene Schaltung wird als Halbaddierer bezeichnet. Sie ist in der Lage zwei 1-Bit Summanden (einstellig) zu addieren.

Volladdierer

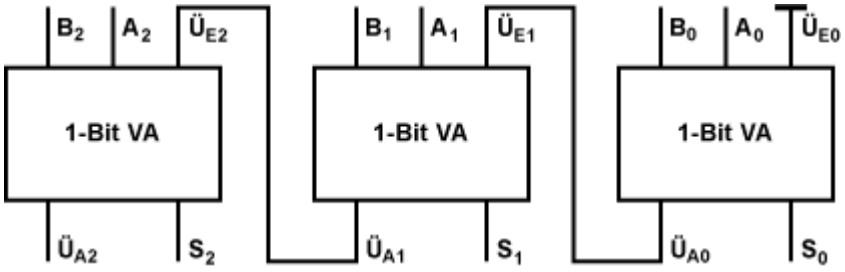
Um **mehrstellige Dualzahlen** addieren zu können benötigt man Schaltungen die auch einen Übertrag einer niederwertigen Stelle berücksichtigen. Man spricht vom **Übertragseingang ÜE**. Die Schaltung bezeichnet man als **Volladdierer (VA)**. Ein Volladdierer kann drei Dualzahlen addieren. Bzw. zwei Dualzahlen addieren und den Übertrag aus einer niederwertigen Stelle berücksichtigen.



Folgende Wahrheitstabelle ergibt sich:

$Ü_E$	B	A	$Ü_A$	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

3-Bit-Volladdierer



Folgende Schaltung zeigt einen 3-Bit-Volladdierer (VA), der aus drei 1-Bit-Volladdierer (VA) realisiert wurde. Damit lassen sich zwei dreistellige Dualzahlen addieren. Der Eingang $Ü_{E0}$ liegt an 0 V (Masse), weil in der niederwertigsten Stelle kein Übertrag berücksichtigt werden muss.

Beispiel

```
 010 = A
+111 = B
```

```
---
1001 = S
```

0. Stelle

$Ü_{E0}$	A_0	B_0	$Ü_{A0}$	S_0
0	0	1	0	1

1. Stelle

$Ü_{E1}$	A_1	B_1	$Ü_{A1}$	S_1
0	1	1	1	0

2. Stelle

$Ü_{E2}$	A_2	B_2	$Ü_{A2}$	S_2
1	0	1	1	0

Ergebnis

$Ü_{A2}$ S_2 S_1 S_0

```
1  0  0  1
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:02:2_05

Last update: **2019/05/14 16:08**

2.6) Übungen

1) Um ihren Swimmingpool schneller füllen zu können, hat Fam. Huber zwei Wasserleitungen (beide verfügen über einen Absperrhahn B und C verlegt, die direkt in den Pool führen. Diese Leitungen zweigen beide von einer Hauptleitung ab, die ebenfalls über einen Absperrhahn A verfügt. Demnach kann Wasser nur in den Pool fließen, wenn der Hauptabsperrhahn A und zumindest einer der anderen Absperrhähne offen sind.
Stelle die Situation mittels einer Wahrheitstabelle und einer Schaltfunktion (LogikSim, …) dar.

2) Frau Müller sagt zu ihrem Mann: „Den ganzen Sonntag hockst du auf dem Sofa, trinkst Bier und stopfst dir Kartoffelchips in den Schlund! Du solltest mal joggen!
Ihr Gatte erwidert: „Ich verspreche dir für nächsten Sonntag Folgendes:

- Wenn ich jogge, werde ich sogar auf Bier oder auf Chips verzichten.
- Wenn ich keine Chips esse, trinke ich kein Bier oder ich jogge nicht.
- Wenn ich aber jogge, brauche ich unbedingt hinterher ein Bier!
- Allerdings: Auch falls ich nicht jogge, werde ich auf jeden Fall Bier oder Chips zu mir nehmen.“

Falls Herr Müller seine Versprechen einhält, wie könnten seine Aktivitäten am nächsten Sonntag aussehen? Erstelle einen schaltalgebraischen Ausdruck und die Wahrheitstabelle zu diesen Aussagen.

3) Zeichne für folgende schaltalgebraischen Ausdrücke die zugehörige Schaltfunktion bzw. erstelle die jeweilige Wahrheitstabelle!

- $f(A,B) = (\neg A \vee \neg B)$
- $f(A,B,C) = (\neg A \wedge \neg B) \wedge C$
- $f(A,B) = (\neg A \vee \neg B) \vee (\neg A \wedge B)$

4) Überprüfe das Gesetz von De Morgan mit Hilfe einer Leitwerttabelle und stelle die Schaltung im Digitalsimulator dar.

5) Überprüfe das Distributivgesetz mittels einer Leitwerttabelle und stelle die Schaltung im Digitalsimulator dar.

6) Vereinfache die Schaltfunktion $f(a,b)= a \wedge (\neg a \vee b)$ und baue die Schaltung im Digitalsimulator auf.

7) Vereinfache die Schaltfunktion $f(a,b)= (a \vee b) \wedge (a \vee \neg b) \wedge (\neg a \vee b)$

8) Vereinfache die Schaltfunktion $f(a,b)= (a \wedge b) \vee (a \wedge \neg b) \vee (\neg a \wedge b)$

9) Zur automatischen Brandbekämpfung wurden drei Sensoren in einem Raum angebracht. Melden mindestens zwei der drei Sensoren eine Rauchentwicklung, so schaltet sich die Sprinkleranlage ein. Entwirf eine Leitwerttabelle, eine Schaltfunktion und stelle diese mittels Digitalsimulator dar!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:02:2_07

Last update: **2019/05/14 16:08**

3) Audiobearbeitung

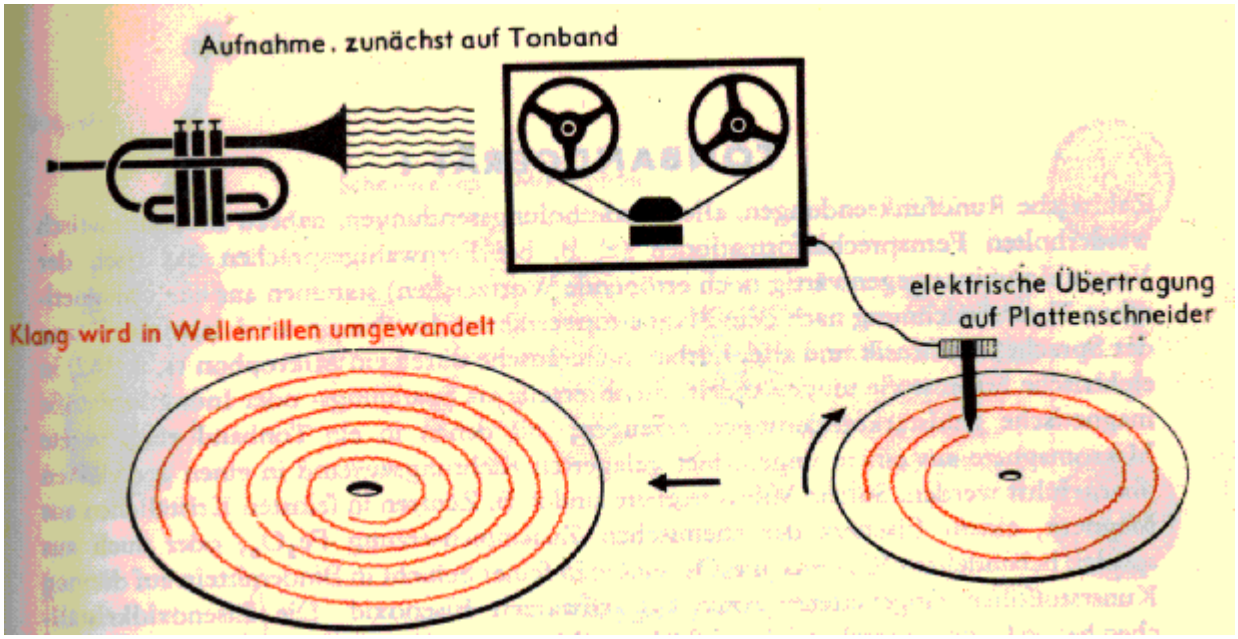
3.1) Theorie

Schallwellen

Schallwellen sind Druckschwankungen in einem Übertragungsmedium (z.B.Luft, Wasser), die von uns wahrgenommen werden können.
In der Analogtechnik wird der Schalldruck durch eine dazu proportionale elektrische Spannung repräsentiert.

- ein Mikrofon wandelt den Schalldruck in die elektrische Spannung um
- diese Spannung kann in andere anolge Darstellungsformen überführt werden
  - Magnetisierung (auf Tonband)
  - Ausleuchtung (auf die Tonspur beim Film)
  - mechanische Auslenkungen (zur Erstellung einer Schallplatte
- am Ende übersetzt ein Lautsprecher die Spannung wieder in einen dazu analogen Wechseldruck





Ton

Ein Ton ist eine Schwingung, die durch periodische Regelmäßigkeit eine definierbare Grundfrequenz - die Tonhöhe - hörbar werden lässt. Der Klang eines Tons hängt von den Anteilen weiterer Schwingungen innerhalb der Grundschwingung ab. Ein Geräusch - etwa einer Rassel - hat dagegen keine periodische Grundschwingung, die Luftdruckschwankungen verlaufen so unregelmäßig, dass

keine Tonhöhe definierbar ist. Die Übergänge zwischen Ton und Geräusch sind fließend, denkt man daran, dass sich Konzertpauken sehr wohl stimmen lassen, obwohl sie als perkussive Instrumente bereits sehr diffuse Klangspektren erzeugen.

Sound

Sound wird heute oft im 'Multimedia-Deutsch' für alles, was mit dem Audiobereich zu tun hat verwendet. Zum Teil geschieht das sicher aus Unkenntnis der Begriffe aus Akustik und Musik. Der Begriff **Sound** hat jedoch auch Vorteile, da er problematische Fragen der traditionellen Begriffe und Wissenschaftsdisziplinen ausblendet (Was ist Musik?!).

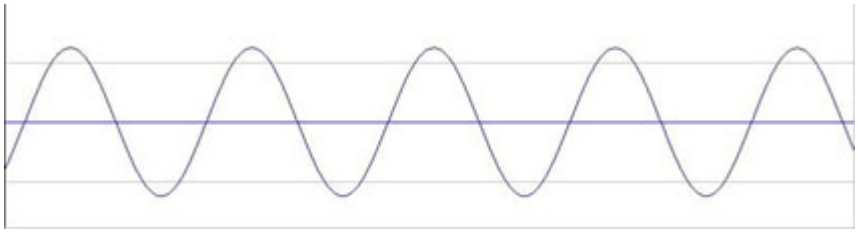
Sprache

Sprache ist danach ein geräuschhaftes=nichtperiodisches Klangereignis, das durch seine definierten Frequenzgemische sprachliche Lautmuster enthält. Im Gesang werden diese Lautmuster einer periodischen Schwingung aufmoduliert. Für gute Sprachverständlichkeit reicht ein Frequenzbereich bis 8 kHz vollkommen aus.

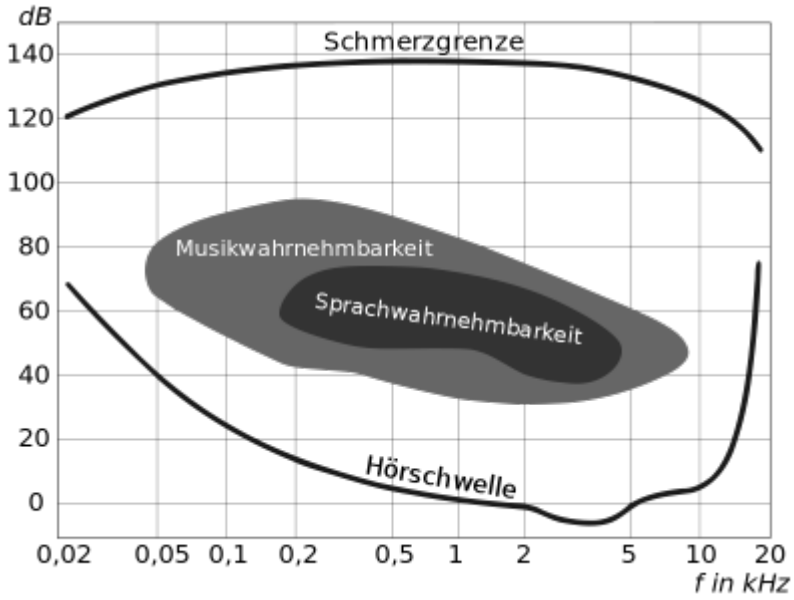
Beispiele für Schwingungen im Audibereich

Sinuswelle

Das Urbild einer periodischen Schwingung, sie besitzt keine weiteren Frequenzanteile.



Hörbereich



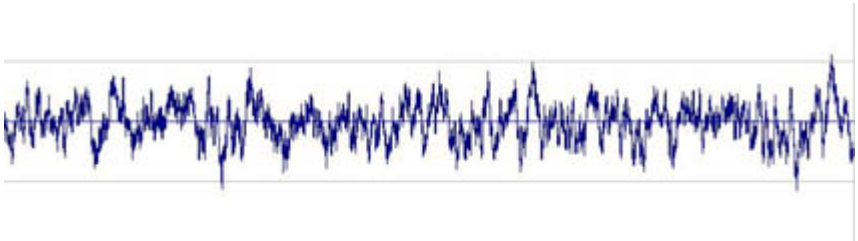
Sinuswellenton in verschiedenen Frequenzen → Teste dein Gehör!



Sinuswelle in verschiedenen Frequenzen (20Hz bis 20kHz)

Rauschen

Schwingungschaos = das Gegenteil einer periodischen Schwingung.



Digitalisierung von Audio

Grundsätzlich sind Daten im Audiobereich zeitkritisch, d.h. sie stellen Anforderungen an Hard- und Software, die historisch gesehen atypisch für einen 'Rechner' sind. Bis heute sind damit

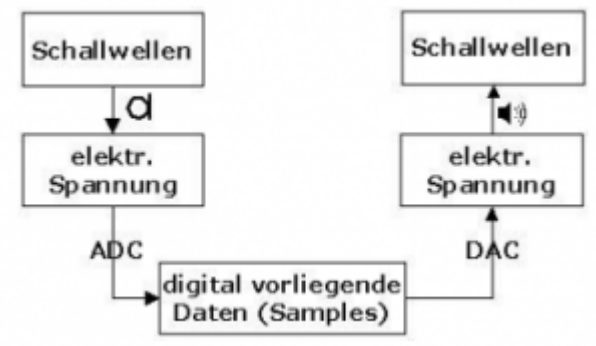
Geschwindigkeits-, Bandbreiten- und Speicherprobleme verbunden, die von technischen und ökonomischen Rahmenbedingungen abhängen.

Um den Sound am Computer bearbeiten zu können, müssen die analogen elektrischen Signale in digitale Daten umgewandelt werden.

Damit Audiodaten aufgenommen und wiedergegeben werden können, werden **A**(nalog)/**D**(igital)-Wandler (**AD-Converter**) bzw. **D**/**A**-Wandler (**DA-Converter**) benötigt. Diese werden i.d.Regel von der Soundkarte zur Verfügung gestellt.

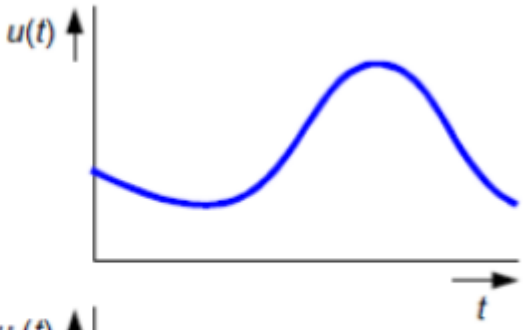
- hierbei wird die analoge Spannung in einem regelmäßigen Zeittakt (**Samplingfrequenz**) gemessen.
- Diese Messwerte sind je nach ausgewählter **Bitrate** aufgelöst
- heraus kommt eine zahlencodierte Nachbildung des analogen Signals als eine **Abfolge von diskreten Schritten**

Beim Abspielen dieser Meßwerte wandelt ein Digital-AnalogConverter (DAC) den Zahlencode wieder in eine Spannung um, die vom Lautsprecher wieder in Schalldruck umgesetzt wird.



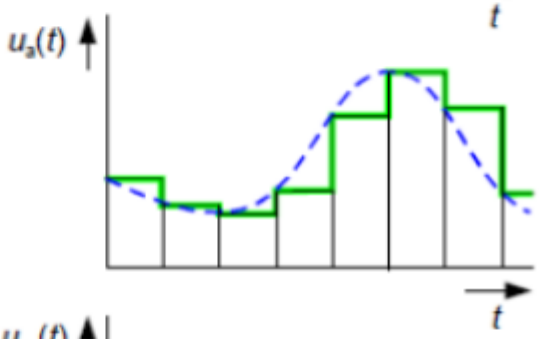
Beim Computer befinden sich die beiden Converter in der Soundkarte oder sind schon direkt im Mainboard integriert.

Kontinuierliches Signal:



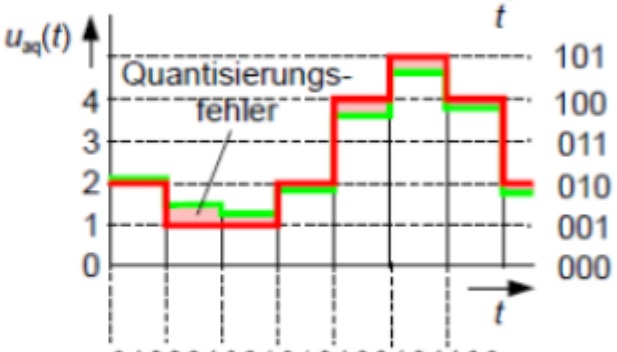
Abgetastetes Signal:

(S&H 100%)



Quantisiertes Signal:

(lineare Quantizierung)



Digitalsignal:

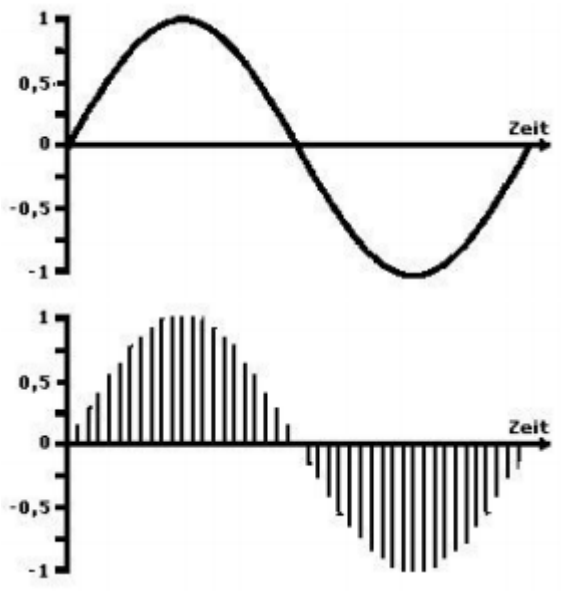


Kanäle

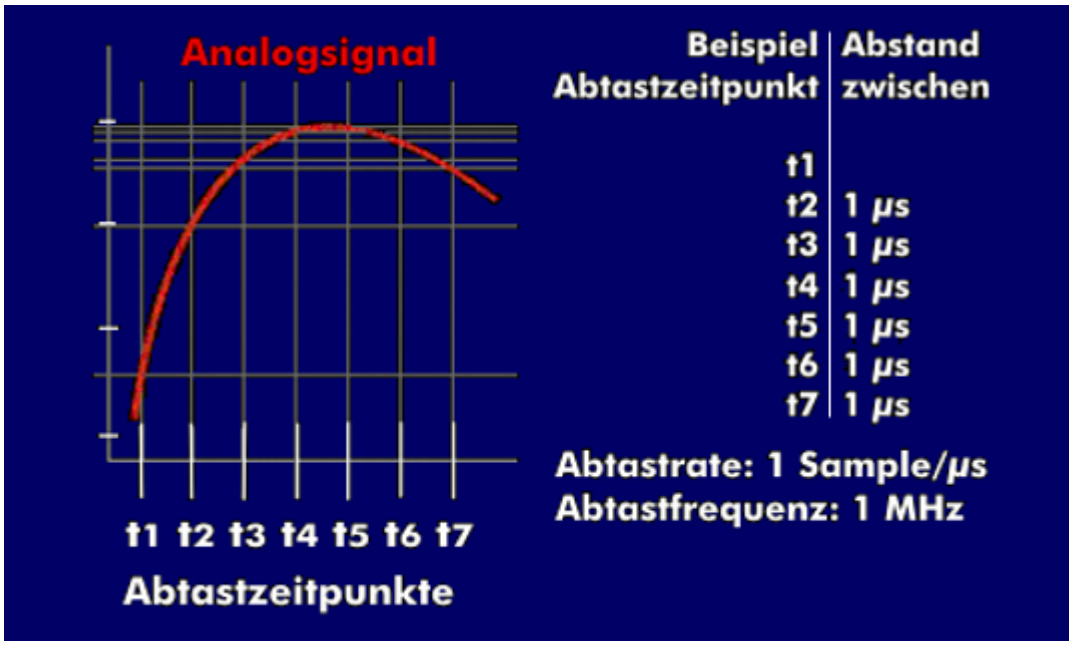
Eine Audiodatei kann mehrere Kanäle haben. Verbreitet sind Mono (ein Kanal), Stereo (2 Kanäle) und 5.1 bzw. 7.1 (Surround). Ein Kanal liefert jeweils die Informationen für einen der Lautsprecher und ist ein eigenständiges Audiosignal. Das heißt wir können eine Stereo-Datei aufteilen und in zwei Mono Dateien speichern.

Samplingrate/Samplingfrequenz

Bei der Digitalisierung werden in **regelmäßigen zeitlichen Abständen Messproben (Samples)** des analogen Ausgangssignals entnommen. Die **Abtastrate** wird **pro Sekunde** angegeben.



z.B. :Für die Audio-CD wird mit 44.100 Hz (also 44100 mal in der Sekunde) gemessen.



Aliasing

Beim Digitalisieren können Frequenzen entstehen, die im Original nicht vorhanden sind. Diese Artefakte oder 'Aliasfrequenzen' des Digitalisierungsvorgangs sind Ergebnisse einer **unzureichenden Abtastfrequenz**. Die **Abtastrate** muss **mindestens das Doppelte der zu digitalisierenden Frequenz** (siehe Abtasttheorem nach Nyquist/Shannon) betragen. Je höher die Abtastrate, desto besser ist die Audioqualität nach der Digitalisierung.

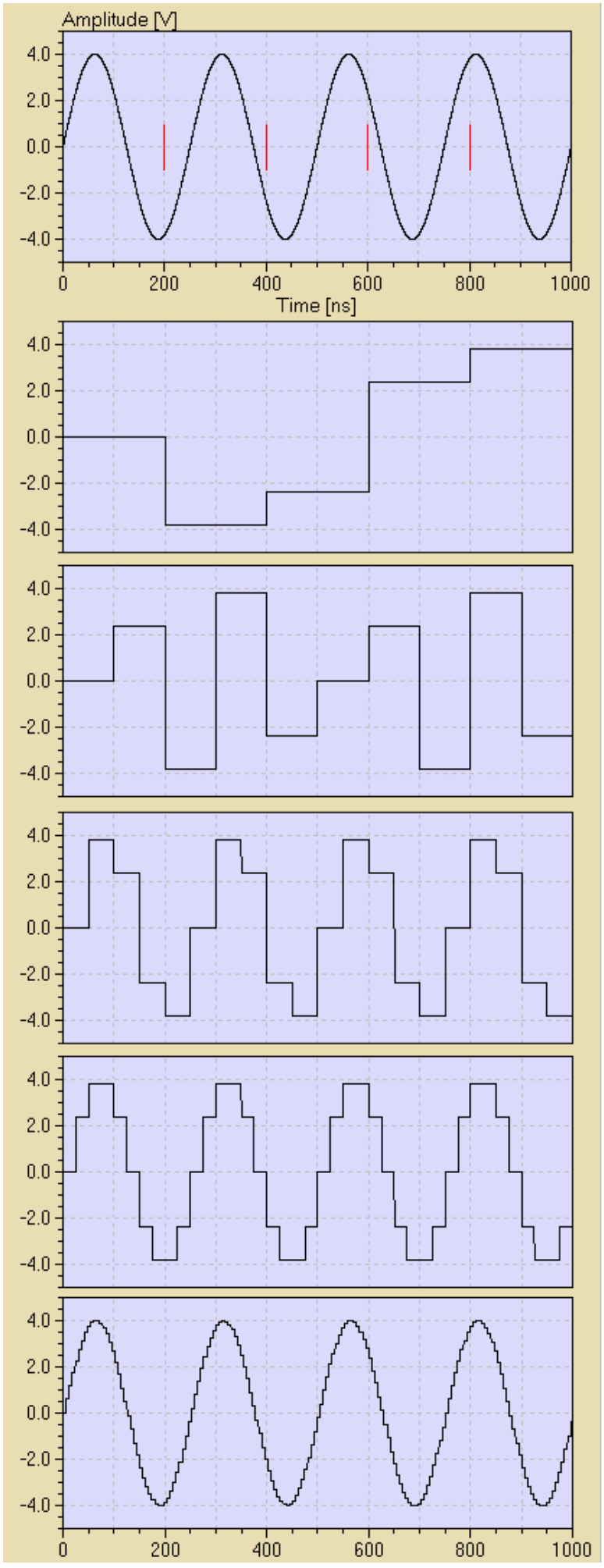
Beispiel

Das folgende Originalsignal zeigt in der Skizze eine Sinuswelle mit der Schwingungsdauer von T=250ns.

Das heißt die Frequenz (`1/s = Hz`) des Orignalsignals ist:

`1/250ns` ⇒ `1/250*10^9 s` ⇒ `1/250*10^9 s` ⇒ `4000000 *1 /s` ⇒ `4000000 Hz` ⇒ `4 MHz`

D.h. Um ein 4-MHz-Signal korrekt wiedergeben zu können ist eine minimale Abtastrate von 8 MHz notwendig. Wird diese Grenzfrequenz unterschritten, entstehen Artefakte, die mit dem ursprünglichen Signal nichts mehr zu tun haben.


Amplitude [V]
4.0
2.0
0.0
-2.0
-4.0
0
200
400
600
800
1000
Time [ns]
Original-signal
Abtastrate 5 MHz
Abtastrate 10 MHz
Abtastrate 20 MHz
Abtastrate 40 MHz
Abtastrate 160 MHz

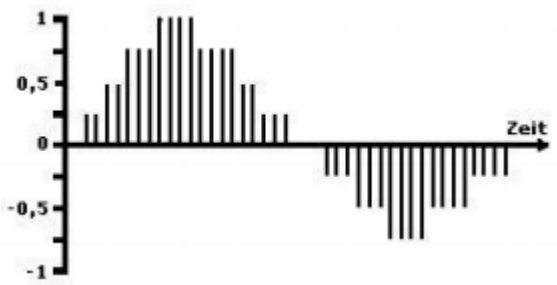
Bittiefe

Für die jeweiligen Samples stehen nur eine begrenzte Anzahl an Messwerten zur Verfügung. Die für das Digitalisieren gewählte Bittiefe bestimmt diese Menge, die sich über den Messbereich verteilt.

Mit der **Bittiefe ist die Länge der Binärzahlen** gemeint, aus denen die digitale Audiodatei letztendlich besteht. Binärzahlen sind Zahlen, die nur aus Nullen und Einsen bestehen. Ein Computer rechnet in seinem Inneren nur mit diesen Zahlen.

- 1 Bit –> 2^1 = 2 Zustände (0 oder 1)
- 2 Bit –> 2^2 = 4 Zustände (0 bis 3)
- 3 Bit –> 2^3 = 8 Zustände (0 bis 7)
- …

Abtastung mit Bittiefe = 3



Bitrate

Die Bitrate (kBit/s) wird gern mit der Auflösung verwechselt. Sie steht für die „Bandbreite“ der Audiodatei, also welche Datenmenge in einer Sekunde verarbeitet wird. Für unkomprimierte Formate wie WAV und AIFF berechnet man die Bitrate ganz einfach, indem man die drei Werte von oben multipliziert:

Bitrate = Kanäle x Samplingrate x Bittiefe

Beispiel

Eine Stereo WAV-Datei in CD-Qualität hat eine Samplingrate von 44,1kHz & Bittiefe von 16Bit.

```
2 Kanäle x 16 Bit x 44,1 kHz = 1411,2 kBit/s
```

unkomprimierte Audioformate

Für unkomprimierte Audiosignale gibt es für den Computer die Formate **WAV(Windows)**, **AIFF (Macintosh)**, **AU (Linux)** und **PCM**.

Diese Formate unterscheiden sich nicht wesentlich. Sie können Audio prinzipiell in jeder Samplingfrequenz und Bittiefe speichern. Die Wahl richtet sich dabei nach der gewünschten Wiedergabequalität, der Downloadgeschwindigkeit und der Speicherplatzverfügbarkeit.

Üblich sind folgende Kombinationen:

Qualität	Bittiefe	Samplingfrequenz Hz	Anzahl Kanäle	Downloadrate kbit/s	Speicherbedarf MByte/min
Telefon	8	8000	1 (Mono)	62,5	0,5
Radio	16	22050	1 (Mono)	352,8	2,8
CD	16	44100	2 (Stereo)	1378,1	11
DVD	24	48000	6 (5.1-Sound)	6750,0	54

komprimierte Audioformate

Für die meisten Anwendungen ist der Speicherverbrauch unkomprimierter Formate zu hoch (z.B. für Downloads aus dem Internet, als Soundtrack für Videos auf CD oder DVD)

Für solche Fälle gibt es Formate, welche die Datenmenge reduzieren, indem sie nicht hörbare Frequenzen eliminieren.

Diese Verfahren verschlechtern die Qualität mit jedem Kompressions- und Dekompressionsvorgang und sind daher für die Zwischenspeicherung nicht zu empfehlen. Es sollte daher beim Editieren möglichst lange mit der unkomprimierten Datei gearbeitet werden, bis man das Endresultat schließlich für die finale Anwendung passend komprimiert.

Beispiele für die Anwendung vom Komprimierungs-Codecs:

Anwendung	Qualität	Codecs	Original kbit/s	Komprimiert kbit/s	Kompressionsfaktor
Internet Echtzeit-Streaming	Telefon bis Radio	mp3, mpeg4, wma, RealAudio, ogg, vorbis	689-1378	30-60	ca. 11-45
Internet Download	quasi-CD	mp3, wma, ogg, vorbis	1378	128	10,7
DVD	quasi-DVD	AC3, mpeg2	6750	448 (AC3)	15(AC3)

Streamingdienste

Streaming Audio ist eine Variante des Streaming Media, bei der Audiodaten kontinuierlich über ein Rechnernetz übertragen werden. In seiner einfachsten Form kann man sich Streaming Audio als Hörfunk im Internet vorstellen.

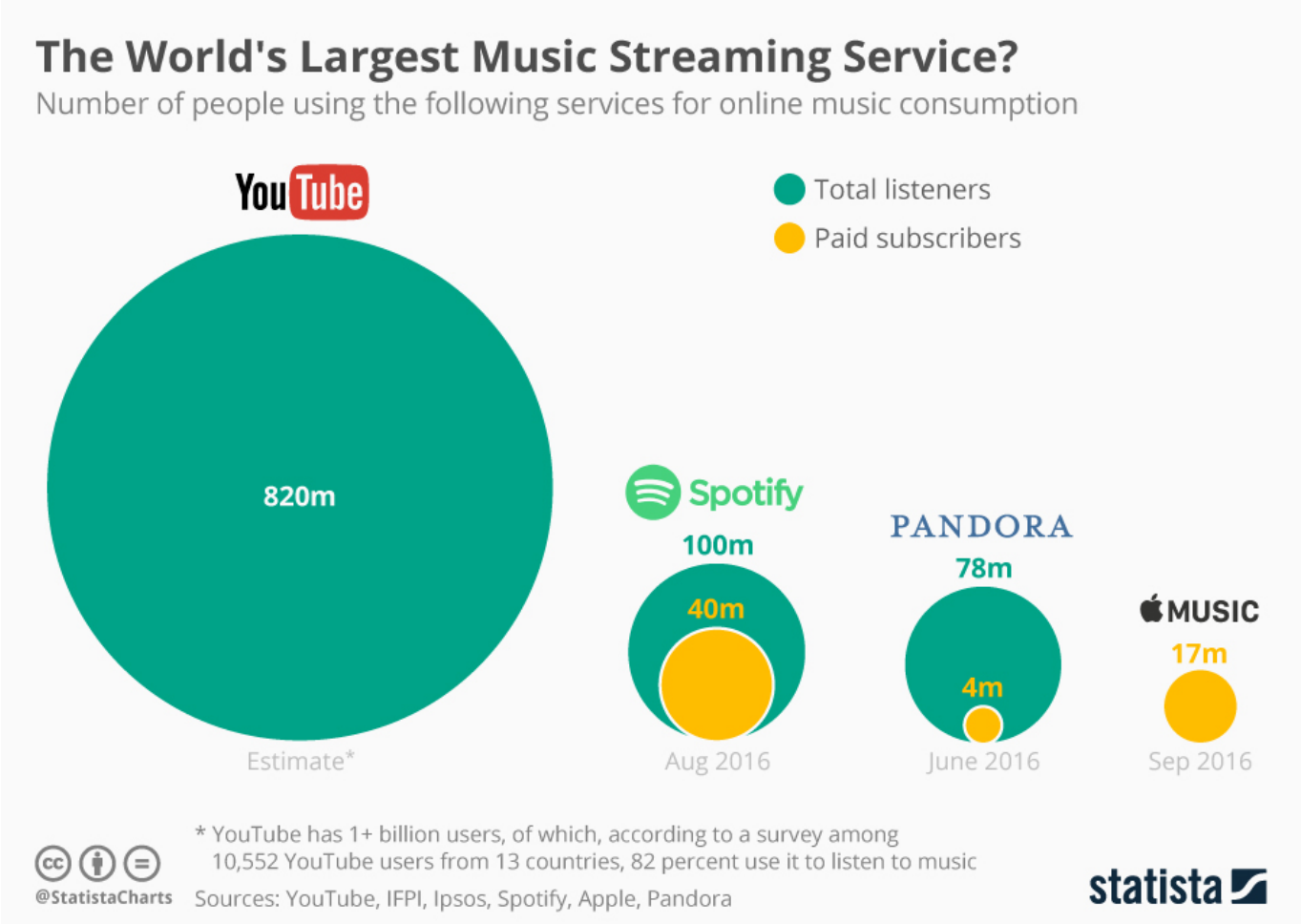
Musik-Streaming beschreibt die Option, jederzeit stationär oder mobil auf viele Millionen Musikstücke zugreifen zu können. Die Songs und Alben lagern in der Cloud und können via Internetverbindung bei verschiedenen Anbietern auf Anfrage abgerufen werden. Das Prinzip des Musikhörens auf Abfrage per Flatrate liegt voll im Trend. Eindeutige Indizien liefern stetig steigende Nutzerzahlen der Abo-Dienste. Die wichtigsten Anbieter der Branche verzeichnen ein konstantes bis atemberaubendes Wachstum. Ein Ende der Erfolgsgeschichte ist vorerst nicht abzusehen.

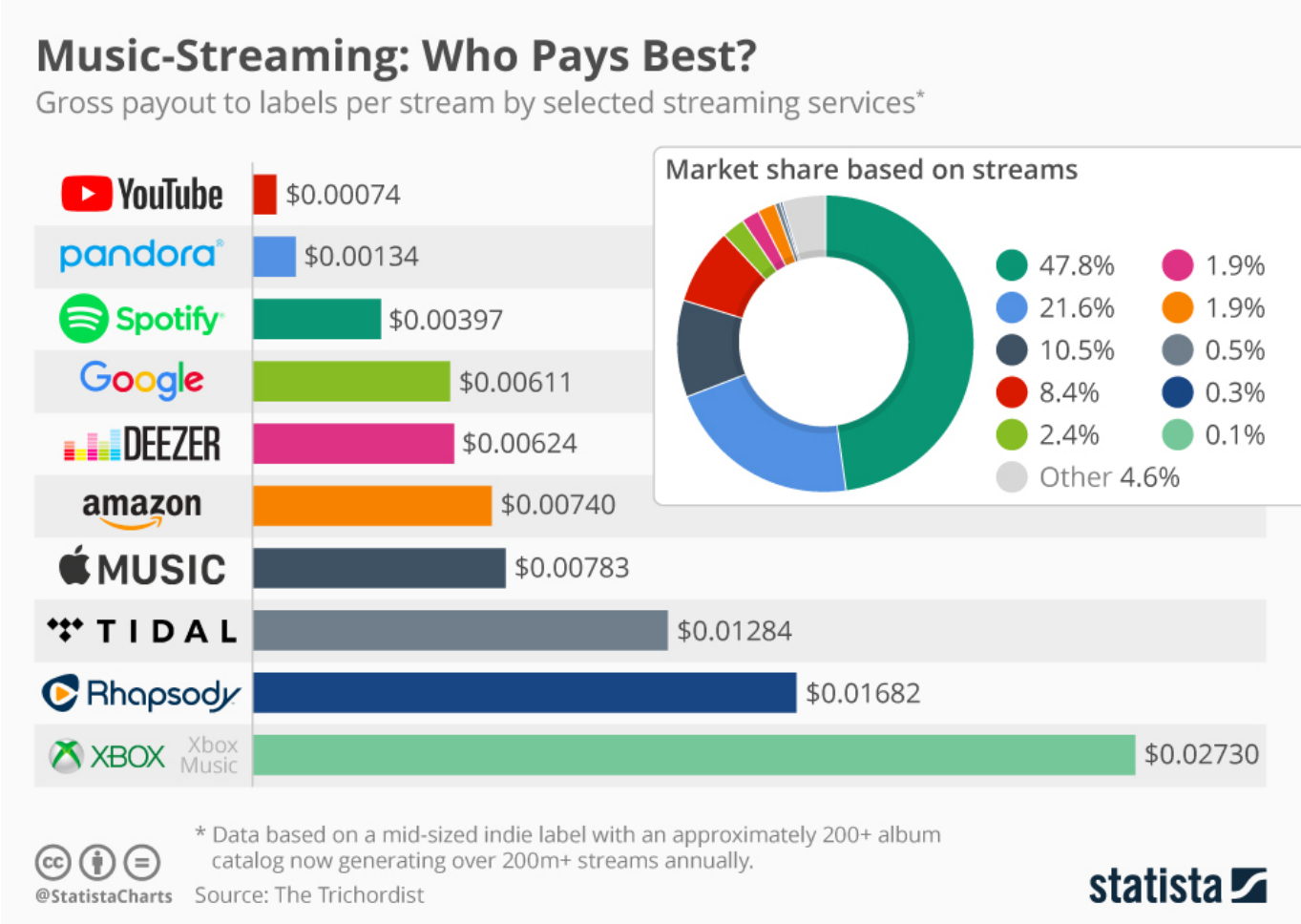
Spotify nennt diesbezüglich mittlerweile Zahlen weit jenseits der 100 Millionen-Marke. Auch Apple Music ist auf dem Vormarsch und spricht von knapp 30 Millionen zahlenden Nutzern. Immer mehr Menschen setzen auf Musik-Streaming und lassen die CD oder auch Schallplatte im Regal verstauben. Doch die Konkurrenz schläft nicht. Der Markt hält zahlreiche weitere Angebote bereit. So buhlen unter anderem Google mit Play Music, Amazon mit Prime Music und Music Unlimited, Napster, Deezer, Tidal oder auch SoundCloud Go um die Gunst der Hörerschaft.

Problemfall bis 2016: Youtube

Nachdem der digitale Musikkonsum nun den Großteil des Marktes ausmacht, versucht die Branche, hier die Einnahmen zu steigern. Die Musikindustrie beklagt hier vor allem einen sogenannten Value Gap.

Wie die Industrie in ihrem globalen Jahresreport darlegt, würde zwar ein Großteil der gestreamten Musik über Googles Videoportal YouTube konsumiert. Von dort käme aber nur ein winziger Teil der Einnahmen. „Es muss endlich geregelt werden, dass sie am Markt Lizenzen erwerben müssen“, sagt BVMI-Chef Florian Drücke.



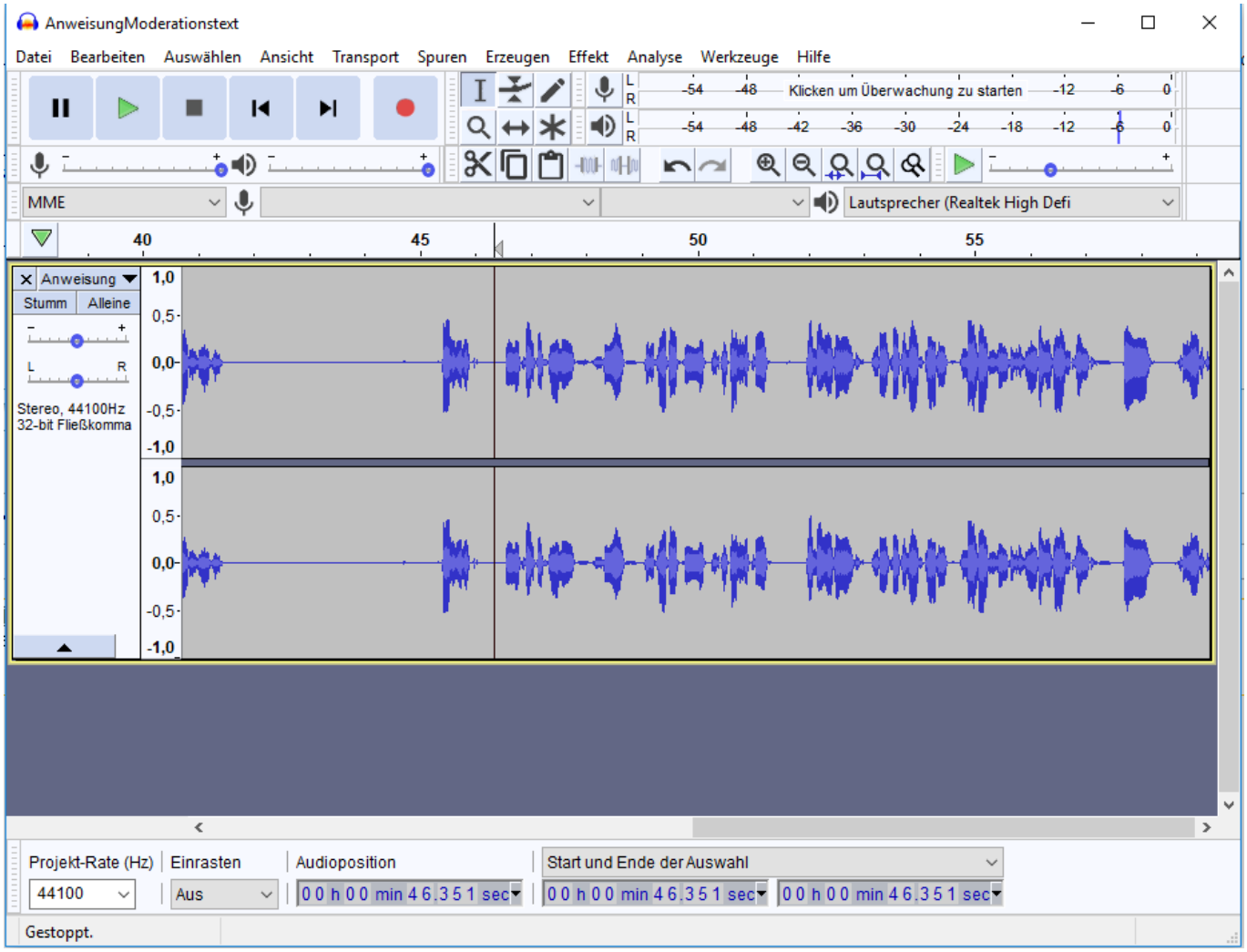


3.2) Praxis mit Audacity

Die Audiobearbeitung gehört zu den Grundlagen der Musikproduktion und wird auch »Audio Editing« genannt. Die Bearbeitung von Audiodateien findet in der Regel vor dem Abmischen statt. Allerdings spricht man auch von Audiobearbeitung, wenn man nur Knackser aus MP3 Dateien entfernen oder MP3s schneiden möchte.

Ein mögliches Programm für Audio Editing ist:

- Programm Audacity
- Eine Anleitung auf YouTube
- Eine Reihe kurzer Filme zur Einführung in Audacity **-> mit Microsoft Edge öffnen**
- Shortcuts Audacity
- Übungen



Destruktive vs. Non-destruktive Audiobearbeitung

Sehr wichtig für deine Audiodateien ist die Unterscheidung zwischen **destruktiver** und **non-destruktiver Audiobearbeitung**. Als **destruktiv** wird die Bearbeitung von Audiodateien dann bezeichnet, wenn die **Änderung an der Originaldatei** selbst durchgeführt und festgeschrieben werden (Bsp. Audacity). Ist die Änderung erst einmal gemacht, lässt sie sich **nicht mehr rückgängig** machen.

Bei der **non-destruktiven Audiobearbeitung** werden die Änderungen **nicht fest in die Datei geschrieben** (Bsp. Reaper). Das Programm zur Bearbeitung merkt sich in diesem Fall nur, wie die Datei zu klingen hat und **berechnet die Änderungen in Echtzeit beim Abhören**. Möchtest Du die **Änderungen festschreiben**, so musst Du die Datei **exportieren (bouncen)** – in eine neue Datei. Dein **Original bleibt jederzeit unangetastet** und Du kannst schnell zum Ursprung zurückkehren.

Vorteile non-destruktiv

- Das Original ist jederzeit sicher
- Du kannst immer wieder von vorne beginnen …
- und ganze neue Effekte ausprobieren
- Fehler lassen sich leicht ausmerzen

Vorteile destruktiv

Selbst wenn die Audiorechner heute eine Menge Leistung bereithalten – sie ist endlich. Und spätestens hier kommt die destruktive Audiobearbeitung ins Spiel. Schreibst Du die Effekte in die Datei fest (destruktiv), wird keine zusätzlcihe Rechenleistung für die Berechnung der Effekte mehr benötigt. Diese Audiodatei wird von allen Geräten und Musikprogrammen in exakt demselben Zustand abgespielt.

Normalisieren von Audio

Als **Normalisierung** wird in der Audiotechnik der Vorgang bezeichnet, die Amplitude analogen oder digitalen Audiomaterials so zu vergrößern oder zu verkleinern, dass sie innerhalb eines vorgegebenen Bereichs liegt. Es gibt im Wesentlichen zwei Gründe, warum normalisiert wird – zum einen technische Rahmenbedingungen bei der Übertragung bzw. Speicherung des Audiomaterials, zum anderen die Anpassung der Lautheit (vom Menschen empfundene Lautstärke) des Materials.

Die Funktion „Normalisieren“ ist zum Beispiel dann sinnvoll, wenn **mehrere Musikstücke hintereinander** abgespielt werden sollen. Mit der Normalisierung der Audiodaten kann eine gleichmäßigere Lautstärke über das gesamte Album hinweg erreicht werden bzw. verhindert werden, dass einzelne Stücke so leise im Vergleich sind, dass der Hörer die Lautstärke an der Stereoanlage verändern muss.

Effekt: Fade-In & Fade-Out (Einblenden & Ausblenden)

Fade-Ins und Fade-Outs werden dazu genutzt, das Audiosignal **sanft ein- bzw. auszublenden**. Das kennst Du beispielsweise aus dem Radio, wo ein Song reinkommt oder wieder rausgefahren wird, während die Moderation startet.

Wenn bei einer Aufnahme die Welle nicht mit einem Nullpunkt beginnt, führt dies zu einem mehr oder minder leisen Knackser. Auch an den Übergängen zweier Audioclips kann es zu einem solchen Knacksen kommen. Eine sehr kurze Einblendung, ein Fade-In zu Beginn und eine Ausblendung, ein Fade-Out am Ende sorgen hier für Nullpunkte und damit knackserfreie Übergänge.

Viele Musik- und Audiobearbeitungsprogramme machen das inzwischen automatisch am Beginn und am Ende einer Audiodatei. Die Handbücher weisen dieses Feature als »automatic fade«, »Auto-Fade«, »automatische Ein-/Ausblendung« oder dergleichen aus.

Tempo ändern

Das Tempo, auch Zeitmaß, gibt in der Musik an, wie schnell ein Stück zu spielen ist, bestimmt also die absolute Dauer der Notenwerte.

Tonhöhe ändern

Die Tonhöhe wird in der Praxis mit der **Frequenz (Schwingungsanzahl pro Zeitspanne) eines**

hörbaren Tones gleichgesetzt, genauer: mit der **Grundfrequenz**, also der Frequenz des tiefsten Teiltons in dem betreffenden Klang. Die Tonhöhe ist neben der Tondauer, der Lautheit und der Klangfarbe eine wichtige Eigenschaft musikalischer Töne und der Vokale gesprochener Sprache.

Lautstärke ändern

Die Lautstärke ist die physikalisch messbare Amplitude oder Stärke des Schalls (z. B. als Schalldruck bzw. als Schalldruckpegel).

Audioformate

Die Vielzahl der digitalen Audio ist für den Nutzer manchmal verwirrend. Hier findest du eine Übersicht der gängigsten Audioformate und deren Dateiendungen inklusive einer Kurzbeschreibung.

.WAV - wave form audio

Das WAV Dateiformat dient der digitalen Speicherung von Audiodaten. In den allermeisten Fällen enthält eine WAV Datei dabei unkompimierte Rohdaten. Der Verzicht auf eine Komprimierung bedeutet folglich, dass die Auiodatei in höchster Qualität gespeichert ist. Allerdings erfordert dies einen enormem Speicherbedarf. Zwei Minuten Musik können schnell um die 20 Megabyte (MB) Platz belegen.

.MP3 - MPEG-1 Audio Layer 3

Von einer Expertengruppe entwickeltes Audioformat mit standardisierten Komprimierungsverfahren. Die Vorteile von MP3 Audiodateien sind die vergleichsweise kleinen Dateien bei dennoch sehr guter Qualität. In der Praxis bedeutet dies, dass Sie den Inhalt einer kompletten Musik CD ohne wahrnehmbare Qualitätsverluste auf die Größe von 10 bis 20 Megabyte (MB) reduzieren können.

.WMA - Windows Media Audio

Ein von Mirosoft © entwickeltes Audioformat für Windows Betriebssysteme. Hohe Kompression mit guter Qualität. Das bedeutet relativ kleine Dateien und dennoch Musik- oder Sounddateien mit guter Qualität möglich.

.AAC - Advanced Audio Coding

Ein ebenfalls von der MPEG (Moving Picture Experts Group) entwickeltes Audiodatenkompressionsverfahren. AAC gilt als qualitativ verbesserter Nachfolger von MP3. Die Vorteile von AAC Audiodateien sind die vergleichsweise kleinen Dateien bei dennoch exzellenter Qualität. In der Praxis bedeutet dies, dass Sie den Inhalt einer kompletten Musik CD ohne wahrnehmbare Qualitätsverluste auf die Größe von 10 bis 20 Megabyte (MB) reduzieren können.

OGG - Ogg Vorbis genannt

Ein freier (patentfreier) Codec zur verlustbehafteten Audiodatenkompression, der von der Xiph.Org Foundation als Alternative zum weit verbreiteten MP3 Format entwickelt wurde.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:03

Last update: **2019/05/20 18:48**

Shortcuts in Audacity

Leertaste	Abspielen - Play/Stop
F1	Auswahlwerkzeug
F2	Hüllkurvenwerkzeug - Lautstärke verstellen
F3	Zeichenwerkzeug - einzelne Samples verändern
F4	Zoomwerkzeug / ev. mit Shift
Strg+Mausrad	Zoomen
F5	Bewegen von Auswahlbereichen

Strg+d	Duplizieren des Auswahlbereichs auf neue Spur
Strg+x	Ausschneiden
Strg+Alt+x	Ausschneiden, Spur rückt nicht nach
Strg+v	Einfügen
Strg+t	Nur Auswahlbereich übrig lassen
Entf	Entfernen
Strg+Alt+k	Löschen, Spur rückt nicht nach
Strg+l	in Stille umwandeln
Strg+i	Trennen

Rauschentfernung

- Effekt - Rauschentferneung
  - zuerst Rauschprofil ermitteln (Teil, wo Rauschen auftritt)
  - Rauschentfernung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:03:shortcuts_audacity

Last update: **2019/05/10 13:13**

Übungen

- Übung Sprechpuppe
- Übung Audioschnitt

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:03:uebungen

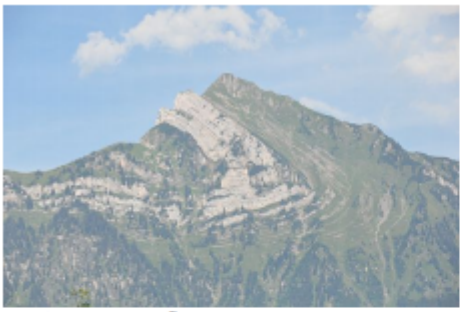
Last update: **2019/05/10 13:14**

4) Digitale Bildbearbeitung

4.1) Theorie

Die Bildbearbeitung dient zur Veränderung von digitalen Bildern. Oft wird die Bildbearbeitung dazu angewandt, um Fehler (z.B. Rote Augen, zu viel/wenig Helligkeit, …) zu beseitigen, die beim Fotografieren entstehen. Bildbearbeitung wird auch kreativ angewandt, um Fotos zu verändern oder neu zusammenzusetzen.

Die folgenden Bilder zeigen einige Möglichkeiten der Bildbearbeitung:



wenig Kontrast, flau



Kontrast erhöht und nachgeschärft



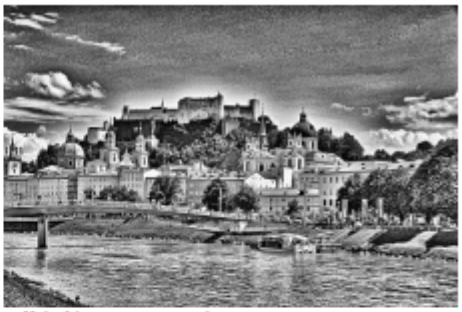
Altes Foto: ausgebleicht mit Schäden



Schäden retuschiert



Original



Effektfilter angewendet



Himmel durch eine Grafik ersetzt



Lichteffekt, Beschriftung eingefügt

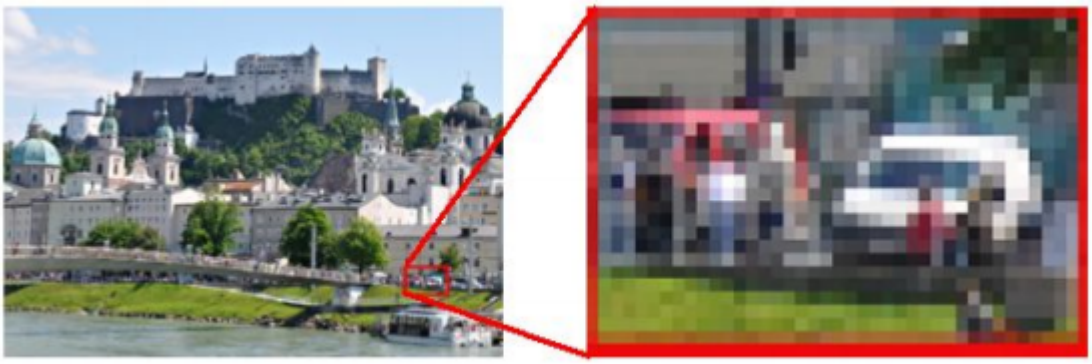
Rastergrafik (Pixelgrafik oder Bitmaps) vs. Vektorgrafik

Man unterscheidet grundsätzlich zwischen diesen beiden Arten von Grafiken.

Rastergrafiken / Pixelgrafiken

bestehen aus Bildpunkten (Pixel), denen Farbinformation zugeordnet sind. Die Größe der Grafik hängt von der Anzahl und Größe dieser Farbpunkte ab. Bei der Bearbeitung von Pixelgrafiken werden die einzelnen Bildpunkte bearbeitet. Rastergrafiken werden für Digitalfotos und eingescannte Bilder

verwendet. Rastergrafiken können nicht beliebig vergrößert werden: je stärker die Vergrößerung, desto mehr erkennt man die Pixelstrukturen (es wird pixelig).



Pixelstrukturen werden im vergrößerten Bildausschnitt sichtbar

Dateiformate für Rastergrafiken

JPG	Grafikformat zur Speicherung von Bildern mit verlustbehafteter Kompression. Als Dateinamenerweiterung wird meistens **jpg**, seltener **jpeg** oder **jpe** verwendet. Die meisten Fotos im Internet werden im Format JPG gespeichert.
PNG	Grafikformat mit verlustfreier (lossless) Kompression. PNG unterstützt neben unterschiedlichen Farbtiefen auch Transparenz per Alphakanal. PNG ist das meistverwendete verlustfreie Grafikformat im Internet.
GIF	Grafikformat mit verlustfreier Kompression für Bilder mit bis zu 256 Farben. GIF-Bilder können auch Transparenz darstellen. GIF-Bilder können aus mehreren Einzelbildern bestehen, die als Animation abgespielt werden.
TIF	Professionelles Grafikformat für Austausch von hochwertigen Bilddateien. TIFF speichert mit verlustfreier Kompression und erlaubt hohe Farbtiefen.
BMP	Unkompromiertes Grafikformat mit hoher Farbtiefe. Aufgrund des hohen Speicherbedarfs nicht für das Internet geeignet.

Vektorgrafiken

Eine Vektorgrafik ist eine Computergrafik, die aus Objekten wie Linien, Kreisen, Polygonen oder Kuven zusammengesetzt ist. Um beispielsweise das Bild eines Kreises zu speichern, benötigt eine Vektorgrafik mindestens zwei Werte:

- die Lage des Kreismittelpunkts und
- den Kreisdurchmesser

Neben der Form und Position der Objekte werden auch die Farbe, Strichstärke, diverse Füllmuster und weitere, das Aussehen bestimmende Daten angegeben.





Vektorgrafiken können ohne Qualitätsverlust vergrößert werden. Viele Vektorgrafiken findet man auf https://www.freepik.com/free-vectors/graphics

Anwendungen

Die Stärke von Vektorgrafiken liegt bei Darstellungen, die als Zusammenstellung von **grafischen Primitiven** befriedigend beschrieben werden können, zum Beispiel **Diagramme** oder **Firmenlogos**. Sie sind **nicht geeignet für gescannte Bilder** und **Digitalfotos**, die naturgemäß als Rastergrafik erfasst werden und nicht verlustfrei umgewandelt werden können. Ebenfalls an die Grenzen stoßen Vektorformate bei komplexen gerenderten Bildern, die ebenfalls direkt als Rastergrafik berechnet werden. Allerdings spezialisieren sich immer mehr Firmen auf die Vektorisierung von Rastergrafiken. Dies ist vor allem von Interesse für **großflächige Bildwerbung**, **Fahrzeugbeschriftung** oder wenn die Vektorisierung als grafischer Effekt genutzt wird.

Grafikformate für Vektorgrafiken:

SVG	Scalable Vector Graphics (SVG, engl. „skalierbare Vektorgrafik“) ist die vom World Wide Web Consortium (W3C) empfohlene Spezifikation zur Beschreibung zweidimensionaler Vektorgrafiken. Aktuelle Webbrowser können SVG-Grafiken darstellen. SVG-Dateien bestehen aus Text und können mit einem Texteditor bearbeitet werden.
EPS	Das Vektorformat Enculapsed PostScript (EPS) wurde von Adobe entwickelt. Es wird von zahlreichen Programmen unterstützt und im professionellen Bereich häufig als Austauschformat für Vektorgrafik (und Bitmaps) verwendet.

Komprimierung von Bildern

Nicht komprimierte Bilder benötigen zur zur Speicherung sehr viel Speicherplatz. Um den Speicherbedarf zu verringern, werden Bilddateien komprimiert. Man unterscheidet zwischen der verlustfreien und der verlustbehafteten Kompression:

Verlustfreie (lossless) Kompression

Bei der verlustfreien Kompression geht keine Information verloren. Die Daten werden nur anders als vorher organisiert, indem bestimmte Redundanzen (Wiederholungen) erkannt und zusammengefasst werden. Aus den komprimierten Daten kann das Originalbild zu 100% wieder rekonstruiert werden. Dateiformate für verlustfreie Kompression sind **GIF, PNG, TIF und JPEG 2000**.

Verlustbehaftete Kompression

Bei der verlustbehafteten Kompression wird versucht, den Informationsverlust unmerklich oder wenigstens erträglich zu halten. Bei der Komprimierung werden Bildinformationen entfernt, die für das Auge kaum wahrnehmbar sind. Je stärker das Bild komprimiert wird, desto deutlicher werden Bildfehler erkennbar. Das am häufigsten eingesetzte Dateiformat für verlustbehaftete Kompression ist **JPEG**

Kompressionsverfahren im Vergleich

Die verbreiteten Formate weisen Vor- und Nachteile auf und eignen sich daher für den jeweiligen Zweck mehr oder weniger gut. Computergrafiken, die harte Kanten enthalten – etwa einfache Grafiken oder Screenshots – können durch verlustfreie Verfahren meist besser komprimiert werden. So zeigt das folgende Beispiel, dass bei gleicher Dateigröße JPG deutliche Fehler (Artefakte) aufweist.



PNG Dateigröße 4,5 KB
Es sind keine Bildfehler vorhanden



JPG Dateigröße 4,5 KB
Bildfehler an den Kanten sichtbar

Das JPG-Format eignet sich besonders gut zur Speicherung von Fotos, wenn es auf geringen Speicherbedarf ankommt. Verlustfreie Formate wie PNG eignen sich in der Regel nicht für Fotografien, da sie erheblich größere Dateien produzieren.

Farbmodelle

Licht besteht aus elektromagnetischen Wellen. Die Wellenlänge bzw. die Frequenz der Lichtwelle bestimmt die Farbe des Lichts. Der Mensch hat zur Farberkennung drei verschiedene Farbrezeptoren, die Zapfen, die unterschiedliche Empfindlichkeit für Wellenlängen aufweisen. Dadurch kann der Mensch verschiedene Farben unterscheiden. Man unterscheidet verschiedene Farbmodelle:

RGB-Farbmodell

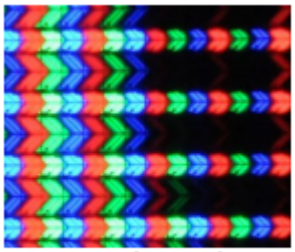
Im RGB-Farbmodell (RGB: **R**ot, **G**rün, **B**lau) werden Farben durch Mischen der drei Grundfarben Rot, Grün und Blau dargestellt. Der RGB-Farbraum ist ein additives Farbmodell: Farben werden durch ausgesendetes Licht dargestellt. Je nach Anteil von rotem, grünem und blauem Licht werden verschiedene Farben wiedergegeben.
Man kann sich die Farbkreise im Bild rechts als drei Scheinwerfer vorstellen, die eine rote, eine grüne und eine blaue Kreisfläche an die Wand projizieren. Dort wo alle drei Farben übereinander liegen, ergibt sich Weiß. Die Addition von Rot und Grün ergibt die Farbe Gelb etc.
Pro Farbe stehen 256 Helligkeitsstufen zur Verfügung. Ist jede Farbe auf maximale Helligkeit (255) eingestellt, so ergibt sich die Farbe Weiß, bei einer Einstellung von 0 erhält man Schwarz.

Daher hat das RGB-Modell eine Farbtiefe von 24 Bit (8 Bit pro Kanal) und kann 16,7 Mio. Farben darstellen. Die restlichen Farben werden durch Farbaddition der Grundfarben dargestellt; d.h. alle Farben zusammen ergeben nicht Schwarz, sondern heben sich auf und ergeben Weiß. Bildschirme gehen immer von einem schwarzen Hintergrund aus, auf dem durch Aktivieren von Elektronenstrahlen rote, blaue und grüne Punkte zum Leuchten gebracht werden. Für jeden Bildpunkt stehen drei verschiedenfarbige Pixel zur Verfügung, deren Leuchtstärke je nach erwünschter Farbe geändert wird.

Die **additive Farbmischung** ist Grundlage für jene **Farbwiedergabeverfahren**, die auf dem RGB-Farbraum beruhen. Anwendungen bestehen im Besonderen bei **Bildschirmen**, wie beim **Farbfernsehen**. Die **Digitalfotografie** beruht ebenfalls auf dieser Form.



RGB-Farbmodell

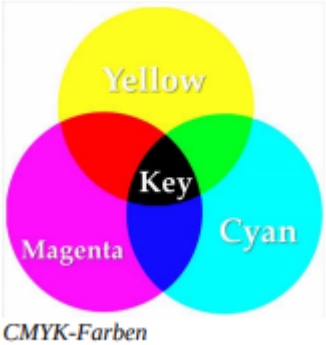


LCD-Fernseher: stark vergrößerter Ausschnitt

CMYK – Farbmodell

CMYK steht für die Farben **C**yan, **M**agenta, **Y**ellow und Schwarz (**K**ey). Diese Farben stehen für Druckfarben und ihr Anteil wird in Prozent von 0 bis 100 % angegeben.
*Das CMYK-Farbmodell ist ein **subtraktives** Farbmodell: die im Bild rechts gezeigten Farbscheiben kann man sich als bunte Filterscheiben vor einem weißen beleuchteten Hintergrund vorstellen.*
Die Mischung der Farben Cyan, Magenta, Yellow ergibt nur theoretisch Schwarz, in der Praxis wird im Druckprozess Schwarz (**K**ey) hinzugefügt. Dabei wird zusätzlich noch ein besserer Kontrast erzielt.

Alle Farbdrucker haben mindestens CMY-Farben und Schwarz. Jeder Druckvorgang eines Bildes im RGB-Farbmodell erfordert vor dem Ausdruck eine Umrechnung in das CMYK-Farbmodell.

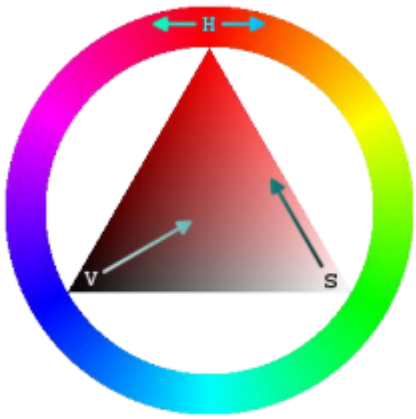


CMYK-Farben

HSV – Farbmodell

Im HSV-Farbraum bestimmt man eine Farbe mit Hilfe des Farbwerts (englisch **h**ue), der Farbsättigung (**s**aturation) und des Helligkeitswerts (**v**alue).

- **Farbwert** als Farbwinkel H auf dem Farbkreis (etwa 0° für Rot, 120° für Grün, 240° für Blau)
- **Sättigung** S in Prozent (0 % = Neutralgrau, 50 % = wenig gesättigte Farbe, 100 % = gesättigte, reine Farbe) oder in einem Intervall von Null bis Eins
- **Hellwert** V als Prozentwert (0 % = keine Helligkeit, 100 % = volle Helligkeit), oder in einem Intervall von Null bis Eins, auch Dunkelstufe genannt.



Ein HSV-Farbwähler

Graufen - Farbmodell

Ein Graustufenbild hat keine Farben. Es besteht aus verschiedenen Grauwerten in Abstufungen von 0 (Schwarz) bis 255 (Weiß).

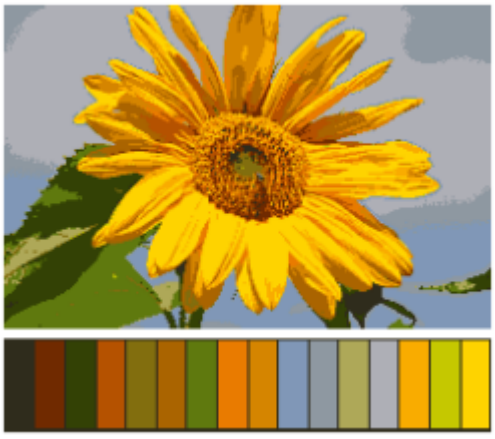
Farbtiefe

Bilder bestehen aus Bildpunkten, denen Farben zugeordnet sind. Die Farbtiefe gibt an, wie viele Farbabstufungen möglich sind und wird in Bit angegeben. Je größer die Anzahl der Bits ist, desto mehr Farbabstufungen sind möglich.

Bild	**Farbtiefe**	**Anzahl Farbabstufungen**
Schwarz-Weiß-Bild ohne Grauabstufungen	1 Bit	2 Farben: Schwarz und Weiß
Graustufenbild	8 Bit	2^8 = 256 Farben
GIF-Bild (indizierte Farben)	8 Bit	2^8 = 256 Farben
Farbbilder (JPG)	24 Bit = 3 Farben je 8 Bit	2^{24} = 16,7 Mill. Farben
Farbbilder mit Transparenz (TIF)	32 Bit = 3 Farben + Transparenz je 8 Bit	2^{32} = 4,3 Mrd. Abstufungen
CMYK	32 Bit = 3 Farben + K-Wert je 8 Bit	2^{32} = 4,3 Mrd. Farben

Farbpaletten – indizierte Farben

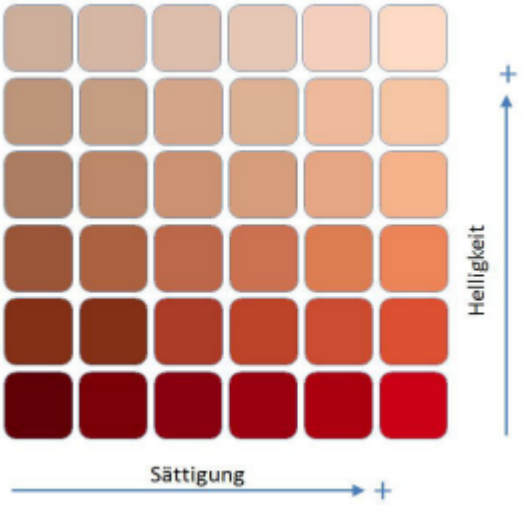
Im GIF-Format kann man pro Grafik aus den 16,7 Mio. möglichen Farben des RGB-Farbmodells maximal 256 Farben wählen und abspeichern. Bei einer solchen Farbauswahl spricht man von einer Farbpalette. Jede der 256 Farben der Farbpalette bzw. Farbtabelle kann beliebige RGB-Werte haben. Zu jedem Bildpixel wird also nur die Nummer der dazugehörigen Farbe aus der Farbpalette gespeichert. Dadurch wird deutlich Speicherplatz gespart.



Farbton

Der Farbton bezeichnet in der Farbenlehre die Eigenschaft, nach der man Farbempfindungen nach beispielsweise **rot**, **gelb** oder **grün** unterscheidet. Eine Farbe desselben Farbtons kann entweder in der **Farbsättigung** variieren, wie graublau oder in der **Helligkeit**, beispielsweise rosa.





Farbsättigung

Die Farbsättigung gibt die Stärke bzw. Intensität der Farbe an: Je bunter und kräftiger uns die Farben eines Bildes erscheinen, desto höher ist die Farbsättigung. Flaue Farben haben eine geringe Farbsättigung. Bilder, die entsättigt wurden, haben nur noch Grautöne.



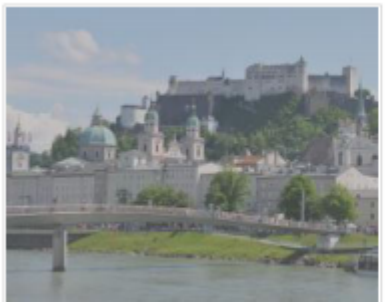
Farbbalance

Ein Bild mit guter Farbbalance zeigt keinen Farbstich. Die Beurteilung der Farbbalance eines Bildes ist subjektiv und hängt von Geschmack und Urteilsvermögen ab. Im Bild rechts ist deutlich ein Farbstich zu sehen – hier wurde die Farbbalance korrigiert.



Kontrast und Dynamik

Der **Kontrast** bezeichnet den Unterschied zwischen hellen und dunklen **Bereichen** eines Bildes. Der **Kontrastumfang** oder die **Dynamik** beschreiben den **Intensitätsunterschied** zwischen dem **hellsten** und **dunkelsten Punkt** eines Bildes.







Helligkeit (Luminanz)

Die **Luminanz** ist eine fotometrische Größe aus der Videotechnik, die als Maß für die Helligkeit von Bildpunkten verwendet wird. Jede Farbe im RGB-Farbmodell kann 256 Helligkeitsstufen annehmen. Verändert man bei einem Bild die Helligkeit, so wird bei allen Pixeln die Helligkeitsstufe erhöht bzw. verringert. Bei einem Foto führt die Änderung der Helligkeit meist zu keinem befriedigenden Ergebnis: schwarze Bereiche werden heller bzw. weiße Stellen werden hellgrau. Damit schwarze Bereiche schwarz und weiße weiß bleiben, muss man eine **Gammakorrektur** durchführen!


verringerte Helligkeit


normale Helligkeit


erhöhte Helligkeit

Gamma

Bei einer Gammakorrektur bleibt Schwarz auch schwarz und Weiß bleibt weiß – nur die mittleren Farbtöne werden verändert.

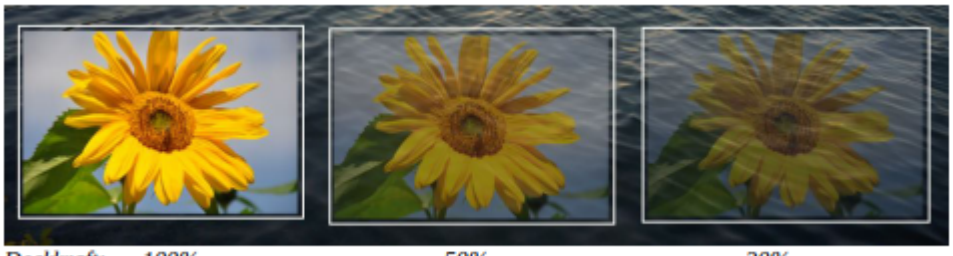

Gammakorrektur: dunkler


Ursprungsbild


Gammakorrektur: aufgehellt

Um die „Helligkeit“ eines Fotos zu ändern ist so gut wie immer eine Gammakorrektur der richtige Weg!

Transparenz


Deckkraft: 100% 50% 30%

Die Transparenz ist die Durchsichtigkeit bzw. Lichtdurchlässigkeit einer Fläche. In der Bildbearbeitung werden oft Bildebenen verwendet. Durch transparente Flächen scheinen darunter liegende Bildflächen durch. Die **Deckkraft** bestimmt, wie durchsichtig die Fläche ist. Sie geht von 100 % (keine Transparenz) bis 0 % (komplette Transparenz) Dateiformate zur Speicherung von Transparenz sind beispielsweise **PSD** (Photoshop), **XCF** (Gimp), **PNG** und **TIF** (stufenlose Transparenz) und **GIF** (nur eine transparente „Farbe“). **JPG** kennt keine Transparenz

Gefahren der Bildbearbeitung

Digitale Fotos finden sich heute fast immer als Zugabe oder Beleg zu Nachrichten-Meldungen, die im Internet verbreitet werden. In manchen Fällen wollen sie nur auf das Thema aufmerksam machen,

haben aber keinen Bezug zum aktuellen Geschehen. Sind solche Abbildungen entsprechend gekennzeichnet, ist dagegen nichts einzuwenden.

Problematisch sind allerdings Fotos, die entweder bewusst in einem falschen Kontext eingesetzt oder die manipuliert werden, um eine beabsichtigte Wirkung zu verstärken. In diesem Fall haben Fotos eine tragende Rolle im Umfeld von „Fake News“; sie können deren Wirkung verstärken oder sogar als scheinbarer Beleg für die ver- oder gefälschte Nachricht dienen.

Die Multimedia-Forensik kennt eine Reihe von Methoden, die hier bei der Aufdeckung helfen können. Dabei werden die bekannten Schutzziele Authentizität und Integrität betrachtet. Hinsichtlich der Authentizität sind die Quelle des Fotos, aber auch Ort und Zeitpunkt des Erstellens wichtig. Bei der Integrität wird geprüft, ob Veränderungen am Foto selbst vorgenommen wurden.

Von der AfD manipuliertes Foto bei der Klimademo in Potsdam

Authentizität

Eine Kamera hinterlässt auf einem mit ihr gemachten Foto Spuren, die sich forensisch auswerten lassen. Das sind zum einen unvermeidliche Fehler im Chip der Kamera, mit dem das Foto aufgenommen wird, zum anderen aber auch Eigenschaften der Kamera-Software beim Erzeugen der Fotodatei. Eine ebenfalls bei der Beurteilung eines Fotos wichtige Frage ist, ob dieses Foto bereits in Nachrichten oder auf andere Weise verwendet wurde und sich somit ein Mindestalter bestimmen lässt. Zu guter Letzt kann auch von Interesse sein, in welchem Kontext das Foto bereits verwendet wurde, da ein Sprung im Kontext auf einen Bruch der Authentizität hinweist.

Quantisierungstabellen

JPEG ist das übliche Format für Fotos im Internet. Es ist verlustbehaftet und führt zur Datenreduktion eine Quantisierung von Bildinhalten durch. Diese Quantisierung wird durch Quantisierungstabellen gesteuert. Verschiedene Kameras und Softwarelösungen verwenden individuelle Tabellen, die aus einer JPEG-Datei abgeleitet werden.

Eine der einfachsten Möglichkeiten, die Authentizität einer Bilddatei zu überprüfen, ist es daher, die tatsächliche Quantisierungstabelle mit derjenigen der behaupteten Quelle zu vergleichen. Beispielsweise könnte behauptet werden, ein Foto stammt direkt aus einer Kamera, tatsächlich passt die Tabelle aber zu der JPEG-Funktion eines Bildbearbeitungsprogramms.

In der Praxis ist diese Authentizitätsprüfung allerdings fehleranfällig: Zum einen werden kaum Fotos direkt aus der Kamera verwendet, sondern durchlaufen erst eine Bildanpassung in einer Software. Die Exif-Daten in den JPEG-Dateien, die die Kamera angibt, werden aber von der Software einfach kopiert. So entsteht eine JPEG-Datei, bei der die angebliche und die tatsächliche letzte Quantisierungsursache nicht übereinstimmen. Anderseits kann die Tabelle auch leicht kopiert oder übernommen werden, um so Angriffe zu verwischen.

Ballistik

Eine zuverlässige Strategie, die Herkunft einer Bilddatei nachzuweisen, ist die sogenannte Kamera-Ballistik. Hier werden Fehler in den bilderzeugenden Chips der Kamera erkannt und so ein

individueller Fingerabdruck der Kamera erstellt, der sich auf jedem Foto findet, das mit der Kamera gemacht wird.

Allerdings erfordert das Erzeugen eines solchen Fingerabdrucks eine Sammlung von Fotos, die nachweislich mit einer Kamera gemacht wurden. Rund 100 Bilder sind hierzu erforderlich. Dann ist es allerdings möglich, ein neues Foto dahingehend zu bewerten, ob es mit dieser Kamera erstellt wurde. Nutzt ein Journalist also immer dieselbe Kamera, kann ein angeblich neues Foto von ihm so überprüft werden.

Allerdings besteht auch hier Manipulationsgefahr: Auch ein Angreifer könnte den Fingerabdruck berechnen und diesen dann auf seinen eigenen Bildern fälschen, um so den Journalisten als Quelle vorzutäuschen.

Robuste Hashverfahren

Es reicht häufig, alte Bilder als scheinbaren Beleg für eine Falschmeldung beizustellen, um deren Glaubwürdigkeit zu erhöhen. Robuste Hashverfahren erlauben es, dasselbe Bild wiederzuerkennen, auch wenn es in verschiedenen Formaten gespeichert oder auch skaliert wurde. Idealerweise sammelt hier eine zentrale Stelle Fotos, die in Pressemitteilungen verwendet wurden und prüft neue Mitteilungen dann gegen die Datenbank ab. Die robusten Hashverfahren erlauben das in einer sehr effizienten Weise.

Fortgeschrittene Lösungen sind auch resistent gegen Ausschnittsbildung. Sie erkennen also das ursprüngliche Bild auch dann, wenn in einer Falschmeldung nur ein Teil des Originals verwendet wurde.

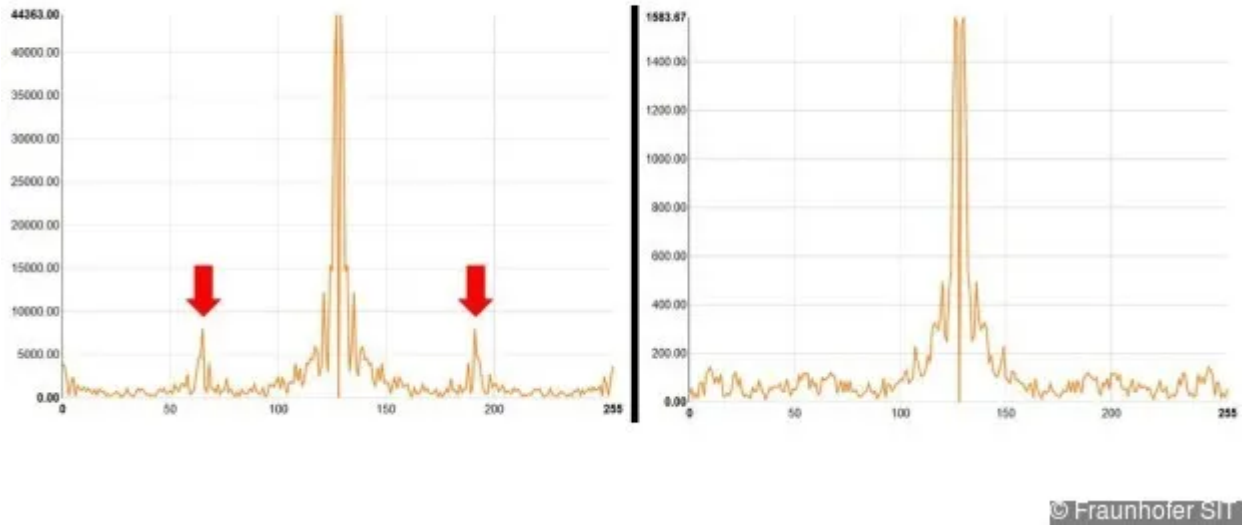
Integrität

Digitale Fotos lassen sich einfach und auf vielfältige Weise verändern. Während einige dieser Änderungen nur die Wirkung des Fotos verbessern sollen, also beispielsweise Helligkeit oder Farbe optimieren, greifen andere bewusst in die Aussage des Fotos ein. Objekte werden aus anderen Bildern hinzugefügt oder unerwünschte Bildteile werden verdeckt. Diese Änderungen hinterlassen Spuren, die man mit forensischen Verfahren entdecken kann.

Mehrfache Kompression

Eine der einfachsten Möglichkeiten, einen ersten Anhaltspunkt für Bildmanipulationen zu erhalten, ist es, eine mehrfache Kompression eines Fotos nachzuweisen. Da ein JPEG-Algorithmus immer eine Quantisierung der Farbkanäle durchführt und diese Quantisierung je nach Qualität unterschiedlich ausfällt, entstehen hierbei in einem Histogramm der JPEG-Koeffizienten deutliche Spuren.

Dabei gilt aber ebenso wie bei den verwendeten Quantisierungstabellen, dass eine mehrfache Kompression auch ohne Manipulation entstehen kann, beispielsweise wenn ein Foto aus einer Kamera (1. Kompression) mit einem Bildbearbeitungsprogramm aufgehellt und dann wieder gespeichert wurde (2. Kompression).



© Fraunhofer SIT

Die Betrachtung von Histogrammen der DCT-Koeffizienten zeigt eine mehrfache Kompression. Links treten zwei verdächtige Häufungen auf, rechts im Vergleich ein unverdächtiges Histogramm. Die Y-Achse zeigt die Häufigkeit, die X-Achse die Werte

Kopierstempel

Die einfachste Art, ein Foto mit einer entsprechenden Software zu manipulieren, ist es, unerwünschte Elemente mit anderen Bereichen des Fotos zu ersetzen oder zu überdecken. Der Vorteil hierbei ist, dass viele Eigenschaften wie Beleuchtung und Farbpalette identisch bleiben. Eine solche Änderung hinterlässt allerdings eine verräterische Spur: Ein identischer Teil des Fotos kommt jetzt zweimal im Bild vor.

Dies lässt sich mit sehr einfachen Algorithmen erkennen. Theoretisch könnte man einfach alle Pixel miteinander vergleichen, identische markieren und Häufungen von Nachbarschaften kenntlich machen. Dies führt aber zu sehr langsamen und auch ungenauen Verfahren. Daher werden heute zuerst übergeordnete Merkmale des Bildes extrahiert und sortiert. Treten untypische Übereinstimmungen auf, werden diese weiterverfolgt, indem die identischen Bereiche maximiert werden. So lassen sich identische Bereiche effizient auffinden.



© Fraunhofer SIT

Ein bekanntes Beispielfoto, in dem Bildelemente an andere Stellen kopiert wurden, um die Aussage zu manipulieren. Rechts das Ergebnis des Detektors, zwei Bereiche kommen doppelt vor

Fremde oder geänderte Bildelemente

Ein allgemein anwendbarer Ansatz, Manipulationen in einem Bild aufzudecken, ist es, statistische Unregelmäßigkeiten in spezifischen Bereichen zu finden. So führt eine JPEG-Kompression in einem Bild immer zu schwachen Veränderungen im Vergleich zum Ausgangsbild. Wie stark diese Änderungen sind, kommt allerdings darauf an, ob das Bild eine vergleichbare Kompression bereits erfahren hat.

Werden nun Objekte aus einem anderen Bild in ein Bild hineinkopiert, so haben diese eine andere Historie und verhalten sich bei einer Kompression anders. Die Änderungen lassen sich aufzeigen. Ebenso gilt dies, wenn Bereiche weichgezeichnet oder ausgeschnitten, skaliert und an anderer Stelle wieder eingefügt werden.



© Fraunhofer SIT

Abweichungen im Verhalten eines Fotos können auf Manipulationen hinweisen. Links das manipulierte Bild, in der Mitte die ermittelten Abweichungen bei der Kompression, rechts dann die erkannte Position der Manipulation

4.2) Praxis mit Adobe Photoshop

- Skriptum Photoshop
- 4.2.1) Grundeinstellungen
- 4.2.2) Ebenen
- 4.2.3) Auswahlwerkzeuge
- 4.2.4) Bildqualität optimieren

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

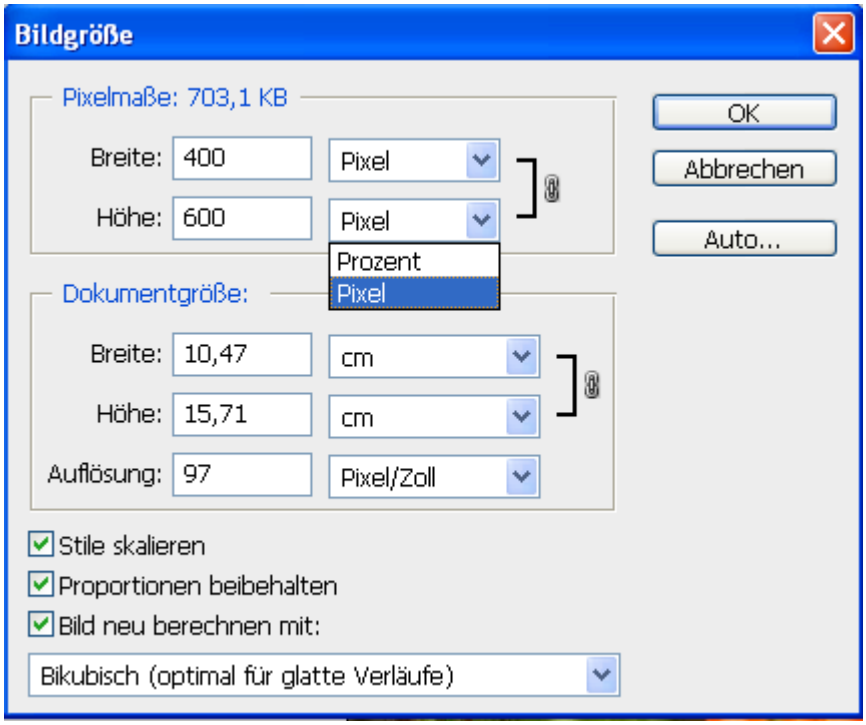
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:04

Last update: **2019/05/20 20:06**

4.2.1) Grundeinstellungen

Auflösung, Bildgröße und Farbtiefe ändern

Um die Bildgröße, Auflösung oder Farbtiefe zu ändern, wählt man den Menüpunkt `Bild - Bildgröße`.



Im Dialogfeld `Pixelmaße` wird die Größe des Bildes in Pixel angezeigt. Dort kann man auch die gewünschte Größe in Pixel angeben. Darüber hinaus kann man das Bild mit Hilfe des Listenfeldes `Prozent` auch prozentual vergrößern oder verkleinern.

Ist das Kontrollfeld `Proportionen erhalten` aktiv, so wird bei der Eingabe des einen Wertes, der zweiter Wert automatisch ermittelt.

Auflösung

Mit der Auflösung wird festgelegt, wie viel Pixel pro Zentimeter auf einem Ausgabemedium wiedergegeben werden. Da viele Bildbearbeitungen aus den USA stammen, wird für die Defintion der Auflösung überweise das amerikanische Längenmaß Inch (Zoll=2,54 cm) verwendet.

Wie bereits erwähnt, ist die Auflösung abhängig vom Ausgabemedium. Es gilt das Prinzip: *So viel wie nötig, so wenig wie möglich* , da sich die Wahl der Auflösung deutlich auf die Dateigröße auswirkt. Im Dialogbereich `Dokumentgröße` bzw. `Dateigröße` kann die Bildgröße in absoluten Werten (cm, mm, pt,...) festgelegt werden.
Weiters kann dort auch die Auflösung festgelegt werden.
Eine zu hohe Auflösung führt zu einer großen Dateigröße, eine zu geringe Auflösung sorgt für einen unschönen Mosaik-Effekt.

Bilder beschneiden

Das Beschneiden von Bildern kann beispielsweise auf zwei Arten erfolgen:

- Auswahlwerkzeug + Menüpunkt: `Bild - Freistellen`
- Freistellungswerkzeug

Bild beschneiden mit dem Menüpunkt BILD - FREISTELLEN

- Wähle in der Toolbox ein Auswahlwerkzeug (z.B. Rechtecke)
- Zeichne einen Auswahlrahmen um den Bereich, der erhalten werden soll.
- Wähle `BILD - FREISTELLEN`

Bild beschneiden mit dem Freistellungswerkzeug

Mit dem Freistellungswerkzeug können Bilder beschnitten werden, indem man einen Rahmen aufzieht. Um einen quadratischen Rahmen zu erhalten muss man die `Umschalt-Taste` gedrückt halten.

Ist ein Aufwahlbereich festgelegt, kann dieser nachträglich noch vergrößert, verschoben oder auch gedreht werden. Um das Bild endgültig freizustellen klickt man doppelt innerhalb des ausgewählten Bereichs oder aktiviert in der Optionsleiste das Symbol ✔.

Bild auf eine bestimmte Größe zuschneiden

Für bestimmten Anwendungen, beispielsweise fürs Web, ist es manchmal notwendig, eine Bild auf eine bestimmte Größe zuzuschneiden.

Das empfiehlt sich die Anwendung von Varianten 1 (Menüpunkt `Bild-Freistellen`).



Stellt man in der Optionsleiste die Art der Auswahl auf `Feste Größe` (s. Abb. oben), so können die gewünschten Werte für Breite und Höhe in cm oder Pixel (px) eingegeben werden. Der Auswahlrahmen nimmt dann automatisch die vorgegebene Größe an.

Aufgabe 1

Öffne die Datei „birgit.png“. Das Bild soll auf die Größe von 350 px mal 450 px zugeschnitten werden! Das Gesicht der Person soll erhalten werden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:04:04_01

Last update: **2019/05/14 20:22**

4.2.2) Ebenen

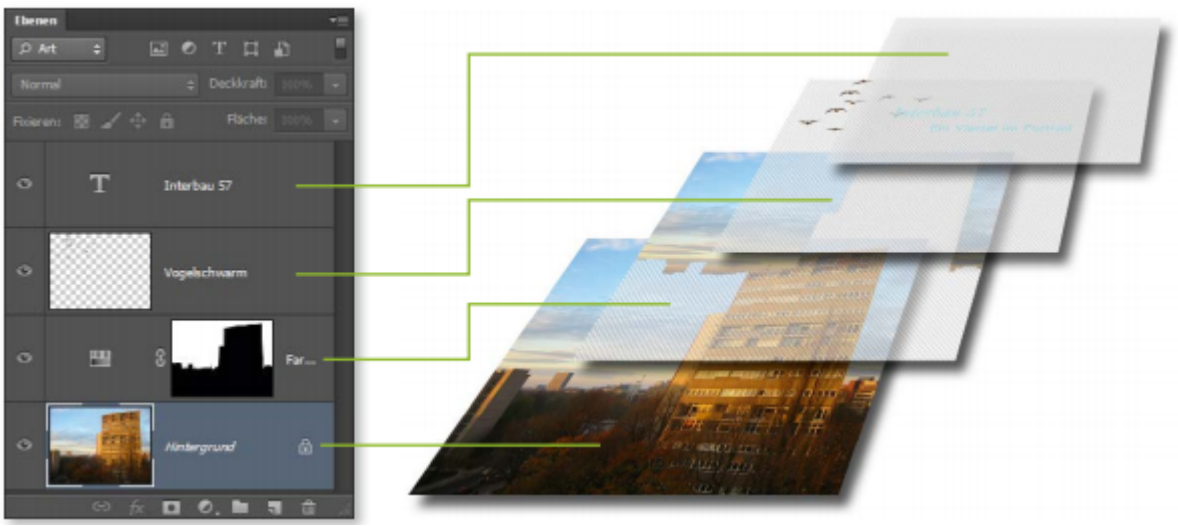
Jedes Bild, das Sie in Photoshop öffnen oder neu anlegen, besteht aus mindestens einer Ebene. Weitere Ebenen können Sie in fast unbegrenzter Zahl hinzufügen. Die Grenze des Machbaren wird somit weniger vom Programm als von der Kapazität Ihres Rechners bestimmt, denn die Größe einer Datei und die Rechenzeit für einzelne Arbeitsschritte steigen rapide an, je mehr Ebenen vorhanden sind.

Vorteile

- Ebenen ermöglichen das separate Bearbeiten, Verschieben, Kopieren, Verändern und Korrigieren einzelner Bildteile
- ein einfaches Anbringen von Änderungen auch bei komplexen Kompositionen
- das Herstellen von Bildvarianten
- kreatives Experimentieren.

Funktionsweise

Stellen Sie sich Ebenen wie übereinandergeschichtete Folien vor. Jede der Folien ist ganz oder teilweise mit Pixeln gefüllt, und auch die Deckkraft von Pixeln auf einer Ebene lässt sich stufenlos ändern. Ebenen mit so verringerter Deckkraft oder nur teilweise mit Bildpixeln gefüllte Ebenen (im Beispielbild die Ebene »Vogelschwarm«) lassen die Inhalte darunterliegender Ebenen erkennen (im Beispiel die Hintergrundebene). Die Reihenfolge der Ebenen im Ebenen-Bedienfeld entspricht der Schichtung der Ebenen im Bild und ist für das Aussehen des Gesamtbildes maßgeblich. Zudem können Sie festlegen, ob und wie die Pixel übereinanderliegender Ebenen miteinander verrechnet werden .



Ebenentransparenz und Ebenendeckkraft

Einem reinen Bild sehen Sie die Ebenen, aus denen es zusammengesetzt ist, nicht an – und auch nicht immer die Deckkraft der Bildpixel. Erst ein Blick in das Ebenen-Bedienfeld offenbart die Ebenenstruktur.

Ebenentransparenz

In der digitalen Fotografie ist Transparenz die Funktionalität, die transparente Bereiche in einem Bild oder einer Bildebene unterstützt. Bestimmte Bildformate bieten keine Unterstützung für die Transparenz. Die Deckkraft ist die Fähigkeit eines Materials bzw. Objekts, Licht nicht durchzulassen. Sie können die Deckkraft von Ebenen, Filtern und Effekten ändern, um mehr oder weniger des jeweils darunter liegenden Bildes durchscheinen zu lassen.

Ebenendeckkraft

Es ist auch möglich, die Deckkraft von Bildpixeln einer Ebene herabzusetzen. Liegt unter der deckkraftreduzierten Ebene eine weitere Bildebene, wird diese sichtbar. Liegt unterhalb der Ebene, deren Deckkraft gesenkt wurde, keine weitere Ebene, ist wieder das grau-weiße »Schachbrett« (=Transparenz) zu sehen.



◂ Abbildung
Hier liegen die Buddha-Figur und ein hellblauer Hintergrund auf zwei getrennten Ebenen. Die Ebene »Blauer Hintergrund« ist in der Deckkraft ❶ auf 70 % reduziert. Das Schachbrettmuster ist daher ein wenig zu sehen.

Ebenenarten

In Photoshop arbeiten Sie mit verschiedenen Ebenenarten, die sich hinsichtlich möglicher Inhalte, Bearbeitungsmöglichkeiten und Einsatzzwecke voneinander unterscheiden.

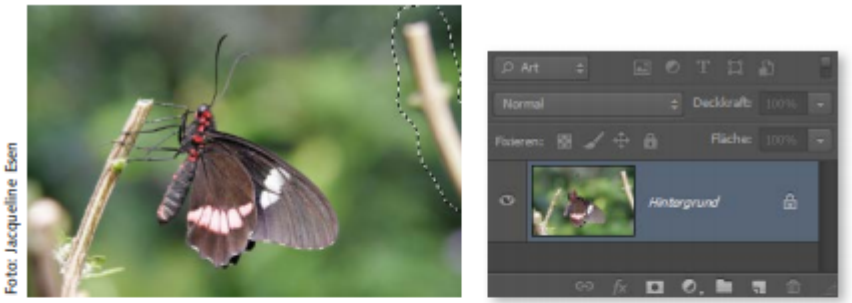
Bildebenen

»Normale« Bildebenen sind der mit Abstand am häufigsten genutzte Ebenentyp. Bildebenen enthalten Pixelinformationen oder Transparenz und lassen sich mit allen Funktionen und Werkzeugen bearbeiten.

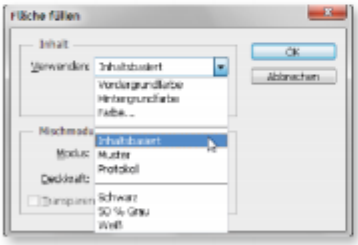
Hintergrundebenen

Die Hintergrundebene ist immer die unterste Ebene einer Datei. Sie erkennen sie auch am kursiv geschriebenen Ebenentitel Hintergrund. Pro Bild kann es nur eine Hintergrundebene geben. Hintergrundebenen unterscheiden sich in einigen Details von normalen Bildebenen: Sie können nicht transparent sein, und nicht alle Arbeitstechniken sind auf sie anwendbar. So können Sie Deckkraft

und Mischmodus von Hintergrundebenen nicht verändern, können keine Ebenenstile zuweisen, und auch beim Löschen von Pixeln gibt es eine Besonderheit (siehe unten). Außerdem lassen sich Hintergrundebenen nicht transformieren. Gedacht sind sie wohl als eine Art »Mal-Leinwand«. Es ist jedoch auch möglich, Bilder ganz ohne Hintergrundebene, ausschließlich mit anderen Ebenenarten, zu erstellen. Allerdings haben einige andere Anwendungen Schwierigkeiten, Dateien ohne reguläre Hintergrundebene zu verarbeiten



▲ **Abbildung**
Das Dokument besteht aus einer Hintergrundebene, der ausgewählte Bereich soll gelöscht werden (etwa durch Betätigen der Entf-Taste).



▲ **Abbildung**
Unter VERWENDEN steht auch die Option INHALTSBASIERT zur Verfügung. Ist sie aktiv …



▲ **Abbildung**
… berechnet Photoshop neue Pixel für den weggeschnittenen Bereich, die zur Umgebung passen. Je nach Motiv funktioniert das mehr oder weniger gut.

Textebenen

Textebenen erkennen Sie an dem großen »T« in der Ebenenminiatur. In ersten Abbildung ist die oberste Ebene eine Textebene (»Interbau 57«).

Wenn Sie Text in ein Bild einfügen, besteht diese Schrift aus Pixeln – bei vergrößerter Bildansicht werden die typischen zackigen Pixelkanten sichtbar. Tatsächlich besteht Text in Photoshop jedoch aus mathematisch definierten Formen (Vektoren!), die die einzelnen Zeichen einer Schrift beschreiben. Dadurch sind Schriften verlustfrei skalierbar und ergeben trotz der »pixeligen« Bildschirmdarstellung ein scharfes Bild im Druck – vorausgesetzt, Dateiformat und Drucker stimmen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:04:04_02

Last update: **2019/05/14 20:22**

4.2.3) Auswahlwerkzeuge

Rechteck und Ellipse

Der Einsatz des Auswahlwerkzeugs „Rechteck und Ellipse“ wurde bereits im vorangehenden Kapitel Einstellung der Bildgröße und Bildabmessung erläutert.

Eine andere Möglichkeit um die Optik ausgeschnittener Bildteile zu verbessern, ist die Verwendung „weicher Kanten“. Im Optionsfeld oder mit Hilfe des Kontextmenüs kann im Feld `weiche Kante` ein Wert eingegeben werden. Dabei werden die Pixel des Kantenbereichs abgestuft und erscheinen weicher bzw. etwas verschwommen. Empfehlenswert für die Weiterverarbeitung ist es, diese Bildabschnitte anschließend auf einen transparenten Hintergrund einzufügen.

Aufgabe

- Hole dir die Datei Blume von Moodle!
- Du möchtest die „rote Blume“ auf das Titelblatt einer Präsentation einfügen und benötigst deshalb nur einen Bildausschnitt.
- Markiere das Bild mit dem Auswahlwerkzeug „Ellipse“ und gib bei „weiche Kante“ den Wert 20px ein.
- Kopiere den Bildauschnitt in eine neue Datei mit transparenten Hintergrund und speichere das Bild unter „Blume_weicheKante.png“.
- Jetzt wiederhole den Vorgang, wobei du den Wert weiche Kante zurück auf 0 px setzt („Blume_harteKante.png)“!
- Öffne eine leere Powerpoint-Präsentation und formatiere dieser mit einer passende Hintergrundfarbe!
- Anschließend füge beide Bilder ein und vergleiche das Ergebnis!

Lasso-Werkzeuge

Das Lasso-Werkzeug

Ein Werkzeug zur Auswahl von unregelmäßigen und schlecht konturierten Bildbereichen ist das Lasso-Werkzeug (bzw. Freihand-Lasso). Dabei werden die auszuwählenden Bildbereiche freihändig mit dem Lasso umkreist.

Aufgabe

Öffne die Datei „Apfel“ und stelle den Apfel mit Hilfe des „Lasso-Werkzeuges“ frei!

Das Polygon-Lasso-Werkzeug

Der große Vorteil des Polygon-Lassos gegenüber dem Standard-Lasso ist, dass während der Erstellung der Auswahl die Maustaste nicht permanent gedrückt werden muss. Stattdessen wird die Auswahl anhand eines Linienzugs definiert, der Punkt für Punkt erstellt wird.

Tipps

- Während der Erstellung eines Linienzugs kann der zuletzt gesetzte Punkt mit der ENTF-Taste gelöscht werden.
- Die gesamte Arbeit mit dem Poygon-Lasso kann mit Hilfe der ESC-Taste abgebrochen werden.
- Besonders empfehlenswert ist es, vor der Arbeit mit dem Lasso in das Bild zu zoomen, um eine möglichst genaue Auswahl treffen zu können.

Besonders bei der Arbeit mit dem Lasso, erscheint der erzeugte Linienzug oft sehr eckig. Um dies zu verbessern, sollte man bei der Arbeit mit dem Polygon-Lasso möglichst viele Punkte bei z.B. Rundungen setzen.

Aufgabe

Öffne die Datei „Gabel.psd“. Ziehe mit Hilfe des Polygon-Lassos einen Linienzug rund um die Gabel und kopiere die Gabel anschließend in eine neue Datei.

Magnetisches Lasso-Werkzeug

Das dritte Lasso-Werkzeug ist das magnetische Lasso. Dieses Werkzeug reagiert magnetisch auf im Bild vorhandene Kanten und sollte zur Erstellung von Auswahlen rund um gut konturierte Kanten innerhalbe eines Bildes verwendet werden.

Der erste Klick mit dem magnetischen Lasso sollte genau die gewünschten Kanten treffen, danach genügt es, den Mauszeiger in etwa entlang der Kante zu bewegen. Es werden währenddessen automatisch Punkte auf der von Photoshop erkannten Kante erzeugt. Die Auswahl wird durch einen Doppelklick oder durch einen Klick auf den zuerste erzeugten Punkt beendet.

Die Werkzeugoptionen des magnetischen lassos bieten neben den auch bei anderen Lasso-Werkzeugen vorhandenen Optionen noch drei zusätzliche Punkte:

- Die `Breite` bestimmt den Bereich rund um die aktuelle Mausposition, der für die Kantenerkennung verwendet wird.
- Mit dem `Kantenkontrast` wird der für eine Kante notwendige Tonwertunterschied festgelegt.
- Die `Frequenz` legt fest, wie häufig automatische Punkte gesetzt werden.

Aufgabe

Öffne die Datei „Apfel.*“ und stelle den Apfel mit Hilfe des magnetischen Lasso-Werkzeugs frei!

Zauberstab

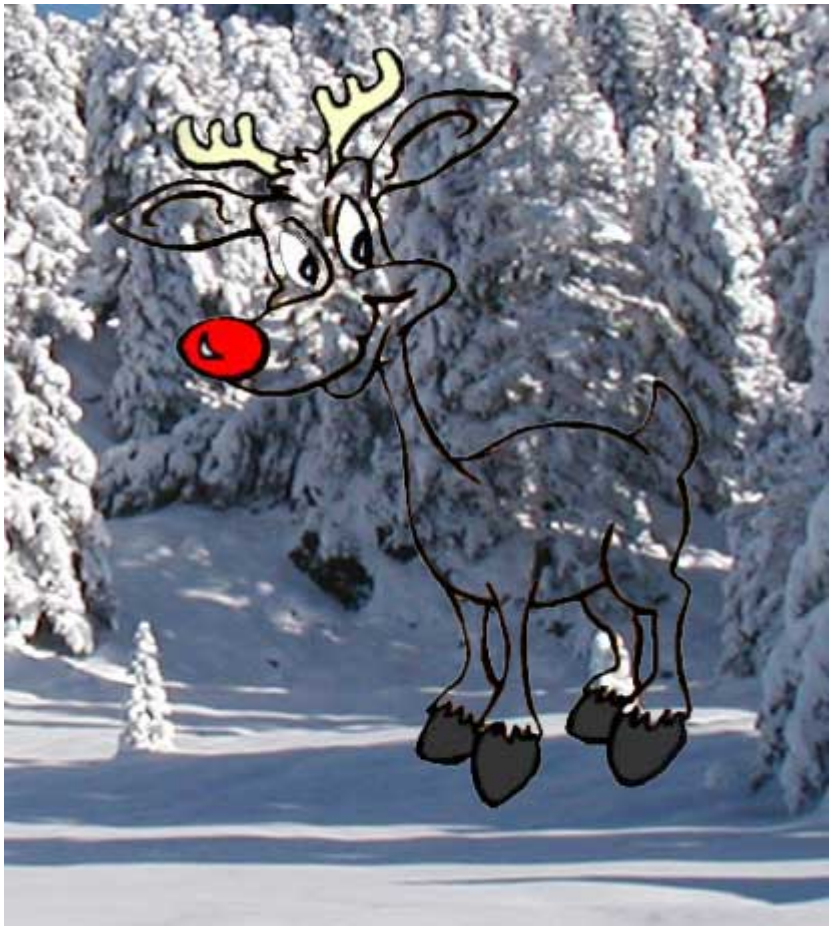
Beim Klick mit dem Zauberstab im Bild merkt sich Photoshop die Farbe des angeklickten Pixels. Die Farbe aller an den Bildpunkt angrenzenden Pixel wird mit der gemerkten Farbe verglichen. Ähneln sich die Farben, so werden die jeweiligen Pixel zur Auswahl hinzugezogen. Die Toleranz für diesen Farbvergleich ist über die Werkzeugoptionen einstellbar. Je größer die Toleranz, desto mehr Pixel werden entfernt.

Aufgabe 1

- Öffne die Datei „Zürich.psd“ und entferne den Hintergrund (Himmel) mit Hilfe des Zauberstabs!

Aufgabe 2

- Hol die von Moodle die Dateien „Winterlandschaft“ und „Rentier“!
- Versuche mit Hilfe des Zauberstabs das folgende Bild nachzugestalten.
- nützliche Hilfsmittel: Auswahl - Auswahl umkehren!



Kombinieren von Auswahlwerkzeugen

Da jedes Auswahlwerkzeug seine Vorzüge und Einschränkungen hat, ist es oft nützlich mehrere Werkezug miteinander zu kombinieren. Die Vorgehensweise wird anhand eines Beispiels erläutert.

Aufgabe

Öffne die Datei „Lampe“! Das Ziel der Aufgabe ist es, den Lampenschirm freizustellen!

1. **Wahl des Werkeugs:** Der Lampenschirm hat eine ähnliche Farbe wie der Hintergrund. Aufgrund des mangelnden Farbunterschieds ist an dieser Stelle das magnetische Lasso ungeeignet. Die einfache Form jedoch spricht für den Einsatzt des **Polygon-Lassos**. Der Sockel der Lampe enthält sehr viele Rundungen, weshalb an dieser Stelle, das Polygon-Lasso ungeeignet ist und sich an dieser Stelle das **magnetische Lasso** deutlich besser eignet.
2. **Auswählen des Lampenschirms** mit Hilfe des Polygon-Lassos! Achtung bei der Oberkante des Schirms! Hier müssen aufgrund der leichten Rundung mehrere Punkte gesetzt werden.
3. Aktiviere in der Werkzeug-Palette das mangetische Lasso. Bevor du mit der Auswahl beginnst, muss in der Werkzeugoption der Auswahlmodus auf `Der Auswahl hinzufügen` gestellt werden (ICON-links). Der Kantenkontrast des mangetischen Lassos soll auf 7 px gesetzt werden.
4. **Auswählen des Sockels:** Bevor du mit der Arbeit beginnst, vergrößere die Ansicht auf ca. 200 %. Fahre nun mit dem magnetischen Lasso den Sockel entlang.
5. Nach dem Schließen der Sockelauswahl werden automatisch beide Auswahlen addiert. Nun sollte der Lampenschrim vollständig ausgewählt sein.
6. Wähle anschließend in der Menüleiste den Punkt `Filter - Kunstfilter` und teste verschiedene Optionen an deiner Auswahl.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:04:04_03

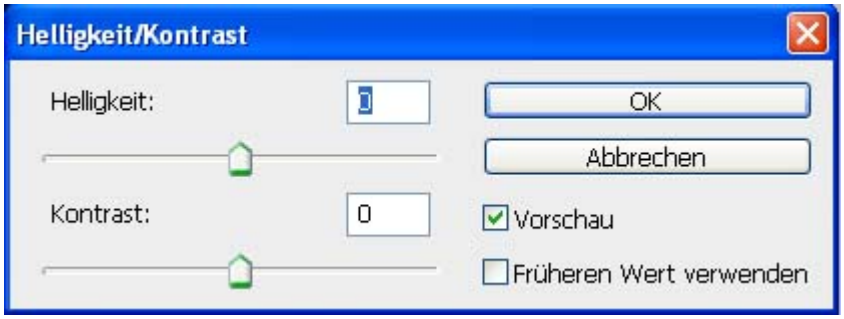
Last update: **2019/05/14 20:22**

4.2.4) Bildqualität optimieren

Helligkeit und Kontrast

Mittles dieser Funktion kann die Balance zwischen Lichtpartien und Schattenbereichen angepasst werden, um Bilder heller oder dunkler zu gestalten. Eine Erhöhung der Helligkeit und des Konstrastes führt in der Regel zu einer deutlichen Verbesserung der Bildqualität.

`Bild - Anpassen - Helligkeit/Kontrast`



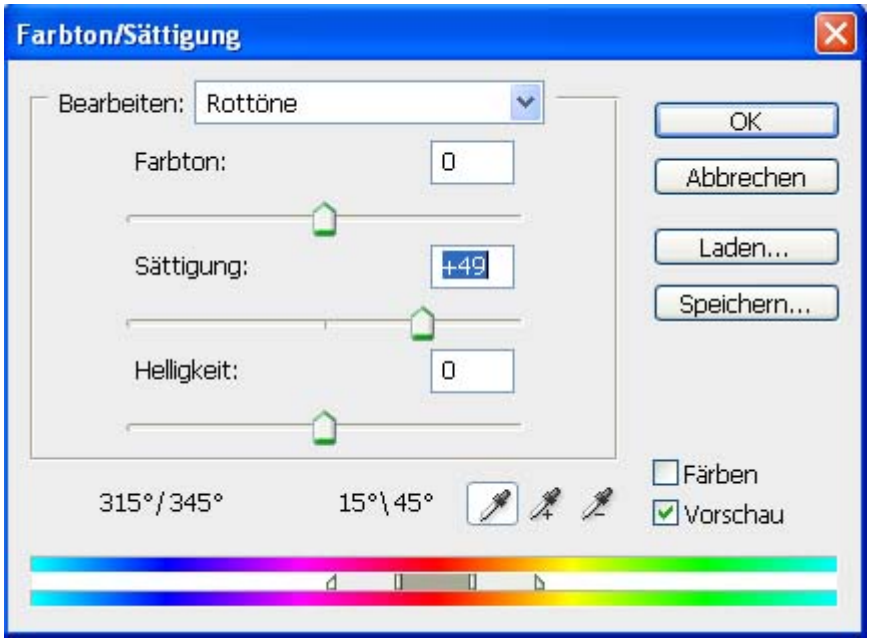
Aufgabe

Öffne das Bild „Schönbrunn.jpg“ und versuche mit Hilfe dieser Option die Qualität des Bildes zu optimieren.

Farbton und Sättigung

Mit dem Menüpunkt `Bild - Anpassen - Farbton/Sättigung` können die Farbtöne eines Bildes ersetzt bzw. deren Sättigung verringert und erhöht werden.

Grundsätzlich wirken sich die Änderungen auf das gesamte Bild aus. Möchte man jedoch nur gezielt einzelne Bereiche bearbeiten, so muss mit Hilfe eines geeigneten Werkzeuges zuvor eine Auswahlbereich festgelegt werden.



Wählt man die Checkbox `Färben`, so kann ein Bild komplett in eine neue Farbe getaucht werden.

Besonders hilfreich ist auch die Möglichkeit, einen Farbtonbereich zu definieren, auf welchen sich die Farbveränderung beschränken soll. Dazu muss im Listenfeld der entsprechende Farbbereich ausgewählt werden.

Aufgabe

Öffne das Bild „Hundertwasser1.jpg“.
Dieses Bild wurde an einem bewölkten Tag erstellt, weshalb die Farben wenig gesättigt erscheinen. Wähle im Listenfeld die Rottöne aus und verändere die Sättigung mit Hilfe des Schieberegler. Anschließend verbessere auch die Rot-, Gelb- und Blautöne des Bildes!

Staub und Kratzer aus einem Bild entfernen

Störungsfilter

Mit dem Hilfe der Option `Filter - Störungsfilter (Rauschfilter) - Staub und Kratzer` können Fehler in einem Bild in beschränktem Maße reduziert werden.

Mit dem Regler *Radius* legt man fest, wie weit das Programm nach abweichenden Pixeln suchen soll. Mit dem Regler *Schwellenwert* legt man fest, wie unterschiedlich die Werte der Pixel sein müssen, damit sie beseitigt oder verändert werden.

Kopierstempel

Das Kopierstempelwerkzeug aus der Werkzeugpalette ermöglicht das Ersetzen von Bildbereichen durch andere Bereiche. Deshalb muss zuerst ein Referenzpunkt an der Stelle im Bild gesetzt werden, von der eine Kopie erzeugt werden soll. Der Referenzpunkt wird durch einen Kreis mit einem Kreuz

dargestellt. Der Referenzpunkt wird mit gedrückter `ALT-Taste` und einem Mausklick an die gewünschte Stelle festgelegt.

Bevor jedoch mit der Arbeit begonnen wird, ist es notwendig, in der Optionsleiste entsprechende Einstellungen vorzunehmen.



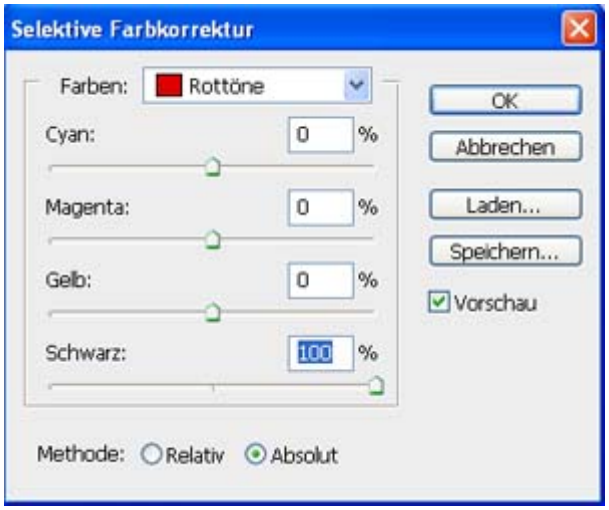
Neben den verschiedensten Optionen sollte hier auch eine geeignete Stärke der Pinselspitze ausgewählt werden.
Bei der Arbeit mit diesem Werkzeug sollte auf alle Fälle die Ansicht des Bildes starkt vergrößert werden.

Aufgabe

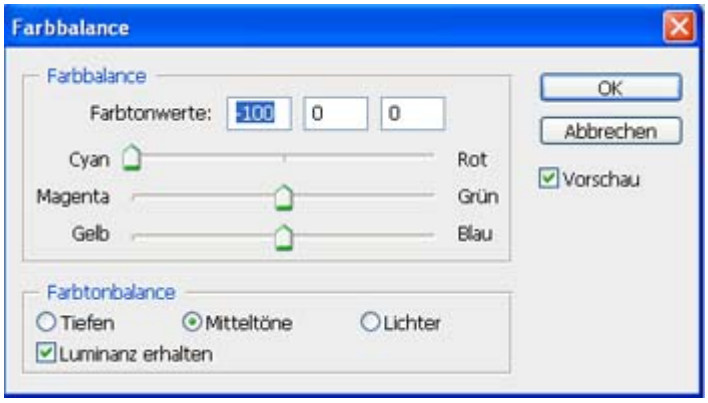
Öffne die Datei „Kratzer_kind“ und verbessere die Qualtät des sehr alten Bildes mit Hilfe des Kopierstempels. Entferne insbesondere die größeren Kratzer.

Entfernen roter Augen

- Öffne die Datei „Auge.jpg“
- Wähle mit Hilfe eines Auswahlwerkzeugs (z.B. Zauberstab) den gesamte rote Stelle aus!
- Wähle `Bild - Anpassungen - Selektive Farbkorrektur`: Farben „Rottöne“, Methode „Absolut



- Setze „Schwarz“ auf 100 %!
- Um die restlichen Rotstiche zu entfernen wähle anschließend `Bild - Anpassungen - Farbbalance` und schiebe den obersten Farbregler richtung Cyan.



Diese Methode hat den Vorteil, dass Blitzreflexe im Auge erhalten bleiben und dadurch natürlich wirken.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:04:04_04

Last update: **2019/05/14 20:22**

5) Betriebssysteme

Ein Betriebssystem, auch **OS** (**operating system**) genannt, ist eine **Zusammenstellung von Computerprogrammen**, die die **Systemressourcen eines Computers wie Arbeitsspeicher, Festplatten, Ein- und Ausgabegeräte verwaltet** und diese **Anwendungsprogrammen zur Verfügung stellt**. Das Betriebssystem bildet dadurch die **Schnittstelle zwischen den Hardware-Komponenten und der Anwendungssoftware des Benutzers**. Betriebssysteme bestehen in der Regel aus einem **Kernel (deutsch: Kern)**, der die **Hardware des Computers verwaltet**, sowie speziellen **Programmen, die beim Start unterschiedliche Aufgaben** übernehmen. Zu diesen Aufgaben gehört unter anderem das Laden von **Gerätetreibern**. Betriebssysteme finden sich in fast allen Arten von Computern: Als Echtzeitbetriebssysteme auf Prozessrechnern und Eingebetteten Systemen, auf Personal Computern, Tabletcomputern, Smartphones und auf größeren Mehrprozessorsystemen wie z. B. Servern und Großrechnern.

- 5.1) Einführung
- 5.2) Zusammenhang Hardware - Betriebssystem
- 5.3) Aufgaben, Ziele und Eigenschaften von Betriebssystemen
- 5.4) Betriebssystemarten
- 5.5) Entwicklung des Betriebssystems
- 5.6) Formatierung von Datenträgern
- 5.7) Planung eines Systems
- 5.8) Übersicht über wichtige Betriebssysteme
- 5.9) Windows
- 5.10) Linux

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05

Last update: **2019/05/14 16:02**

5.1) Einführung Betriebssysteme

Wiederholung Hardware

Wortherkunft

„Hardware“ kommt ursprünglich aus dem Englischen und bedeutet übersetzt Eisenwaren.

Hardware vs. Software

- Hardware = sind alle greifbaren/sichtbaren Elemente eines PCs
- Software = Programme und Daten, also nicht greifbare Elemente eines PCs

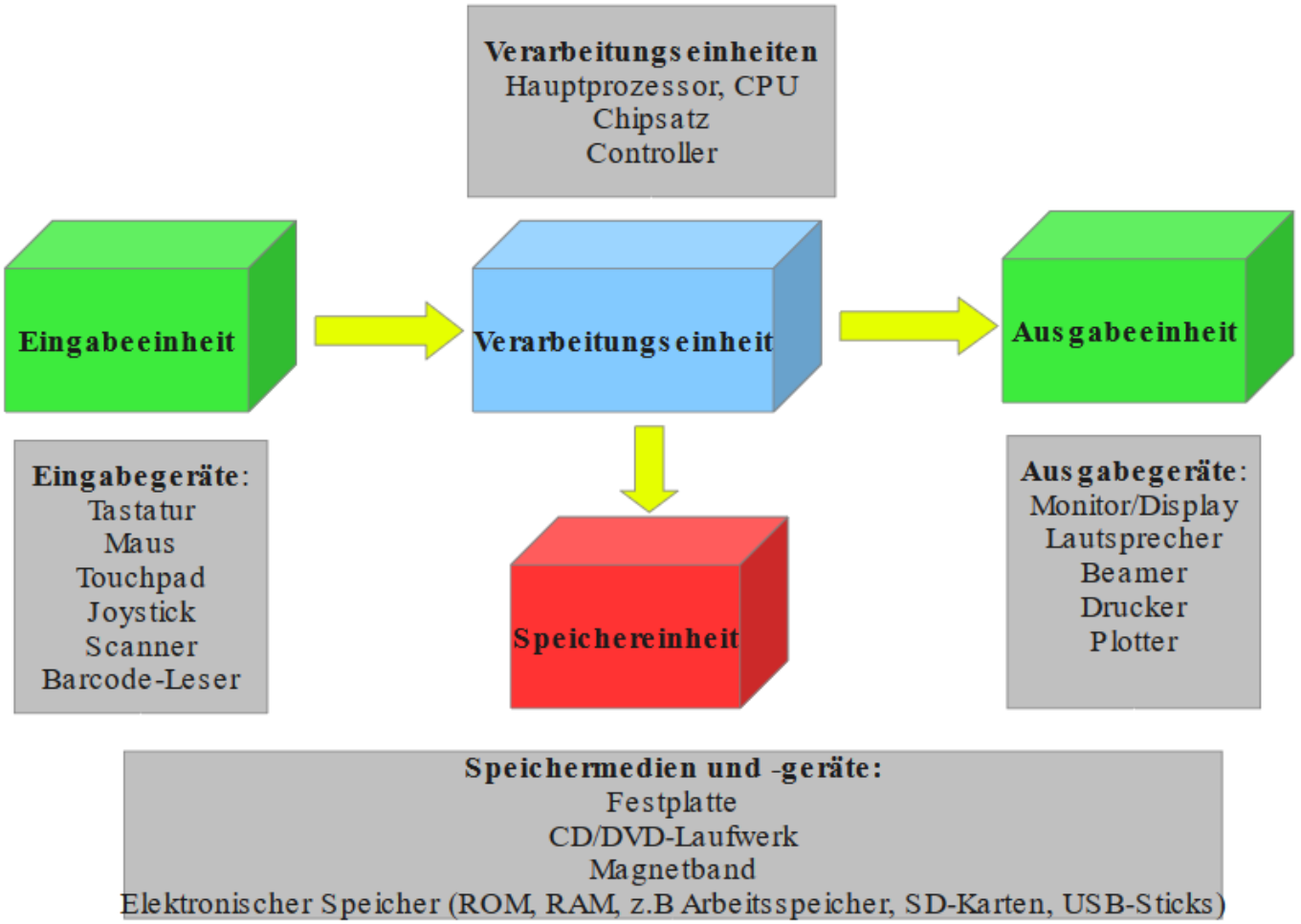
Komponenten

- **Grundbestandteile**
  - Motherboard/Mainboard - Hauptplatine
  - Central Processing Unit (CPU) - zentrale Recheneinheit (Prozessor)
  - Random Access Memory (RAM) - Arbeitsspeicher
- **Massenspeicher**
  - Festplatte (SSD, SATA)
  - Laufwerke (CD, DVD, Band,..)
- **Erweiterungskarten** (optional)
  - Grafikkarte
  - Soundkarte
  - Netzwerkkarte
- **Peripheriegeräte**
  - Eingabegeräte
    - Maus & Tastatur
    - Scanner
  - Ausgabegeräte
    - Bildschirm
    - Drucker

EVA/IPO Prinzip

Das **EVA**-Prinzip ist ein Grundprinzip für die Datenverarbeitung und beschreibt die Reihenfolge in der Daten verarbeitet werden.

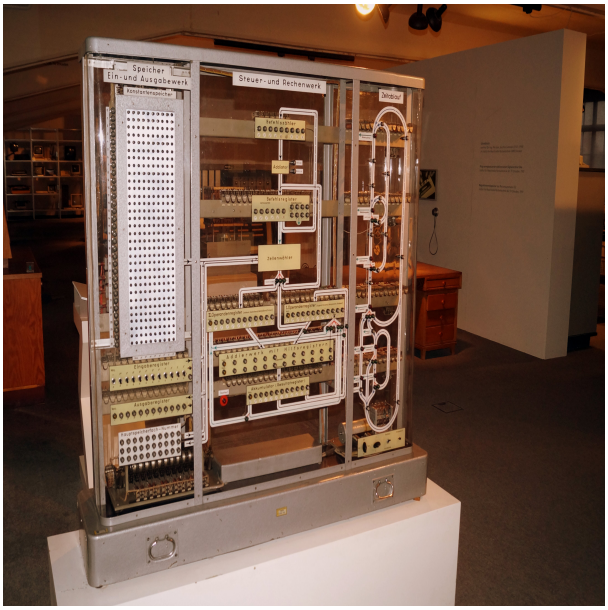
Eingabe (**I**nput) - **V**erarbeitung (**P**rocess) - **A**usgabe (**O**utput)



EVA Prinzip

Von-Neumann-Architektur (VNA)

Die **V**on-**N**eumann-**A**rchitektur ist ein Modell für Computer und bietet die Grundlage für alle heutigen Computer. Das Modell wurde von Johann von Neumann im Jahr 1945 entwickelt. Heute ist Johann von Neumann unter seinem amerikanischen Namen John von Neumann bekannt.

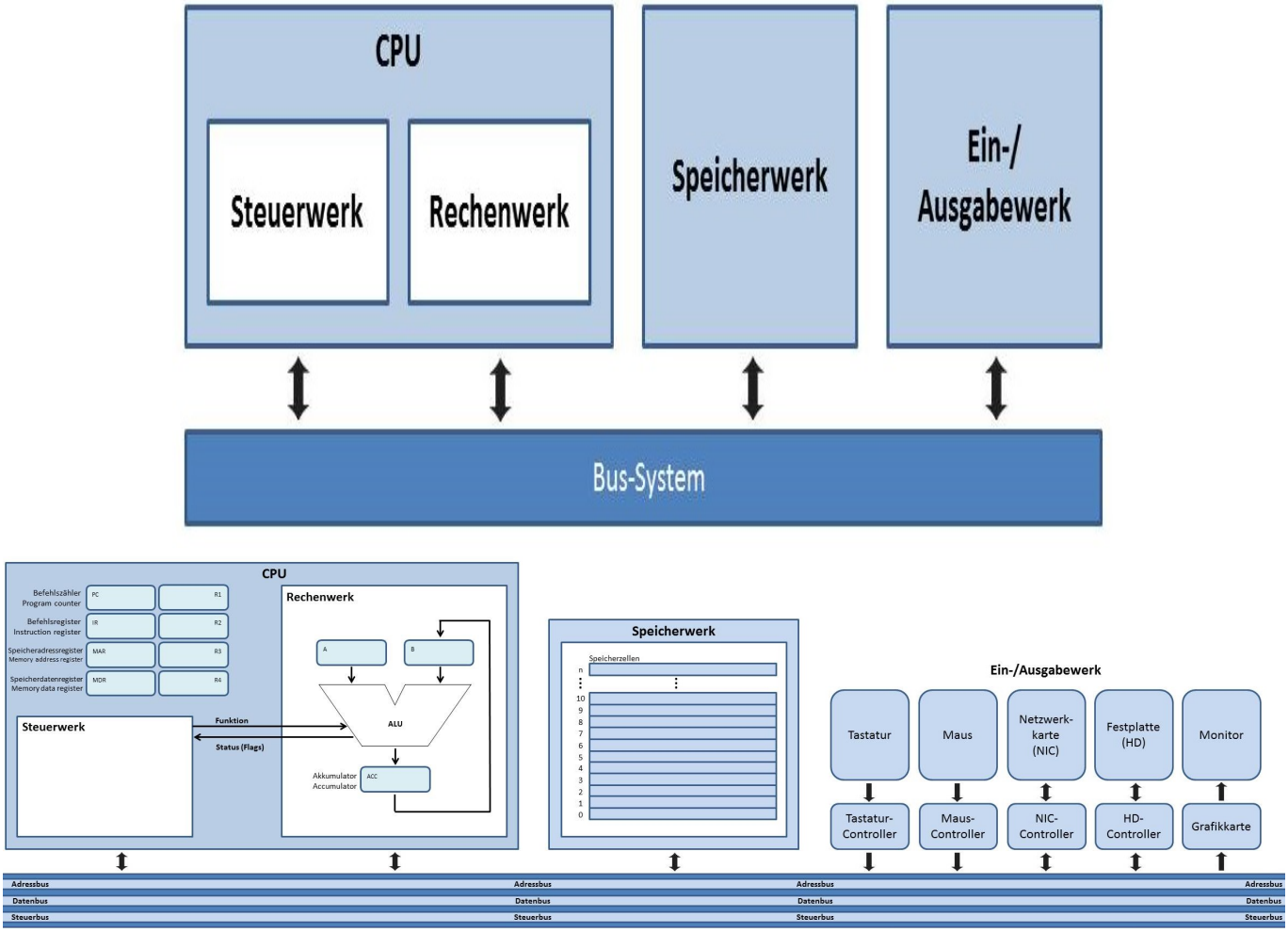


Er revolutionierte die bisherigen Computer, da durch seine Architektur verschiedene Programme auf derselben Hardware laufen konnten. Seine Struktur hat es erstmals ermöglicht, dass mehrere Probleme von ein und derselben Hardware gelöst werden können.

Komponenten

Ein Von-Neumann-Rechner beruht auf folgenden Komponenten, die bis heute in Computern verwendet werden:

- ALU (Arithmetic Logic Unit) – Rechenwerk, selten auch Zentraleinheit oder Prozessor genannt, führt Rechenoperationen und logische Verknüpfungen durch. (Die Begriffe Zentraleinheit und Prozessor werden im Allgemeinen in anderer Bedeutung verwendet.)
- Control Unit – Steuerwerk oder Leitwerk, interpretiert die Anweisungen eines Programms und verschaltet dementsprechend Datenquelle, -senke und notwendige ALU-Komponenten; das Steuerwerk regelt auch die Befehlsabfolge.
- Bussystem: Dient zur Kommunikation zwischen den einzelnen Komponenten (Steuerbus, Adressbus, Datenbus)
- Memory – Speicherwerk speichert sowohl Programme als auch Daten, welche für das Rechenwerk zugänglich sind.
- I/O Unit – Eingabe-/Ausgabewerk steuert die Ein- und Ausgabe von Daten, zum Anwender (Tastatur, Bildschirm) oder zu anderen Systemen (Schnittstellen).



Register

- **Befehlszähler (Program Counter - PC)**

Enthält die Speicheradresse vom Speicherwerk des aktuellen Befehls. (Startadresse = 0x0000). Wird nach jedem Befehl um 1 erhöht.

- **Befehlsregister (Instruction Register - IR)**

Speichert den vom Speicherwerk zurückbekommenen Befehl.

- **Speicheradressregister (Memory Address Register - MAR)**

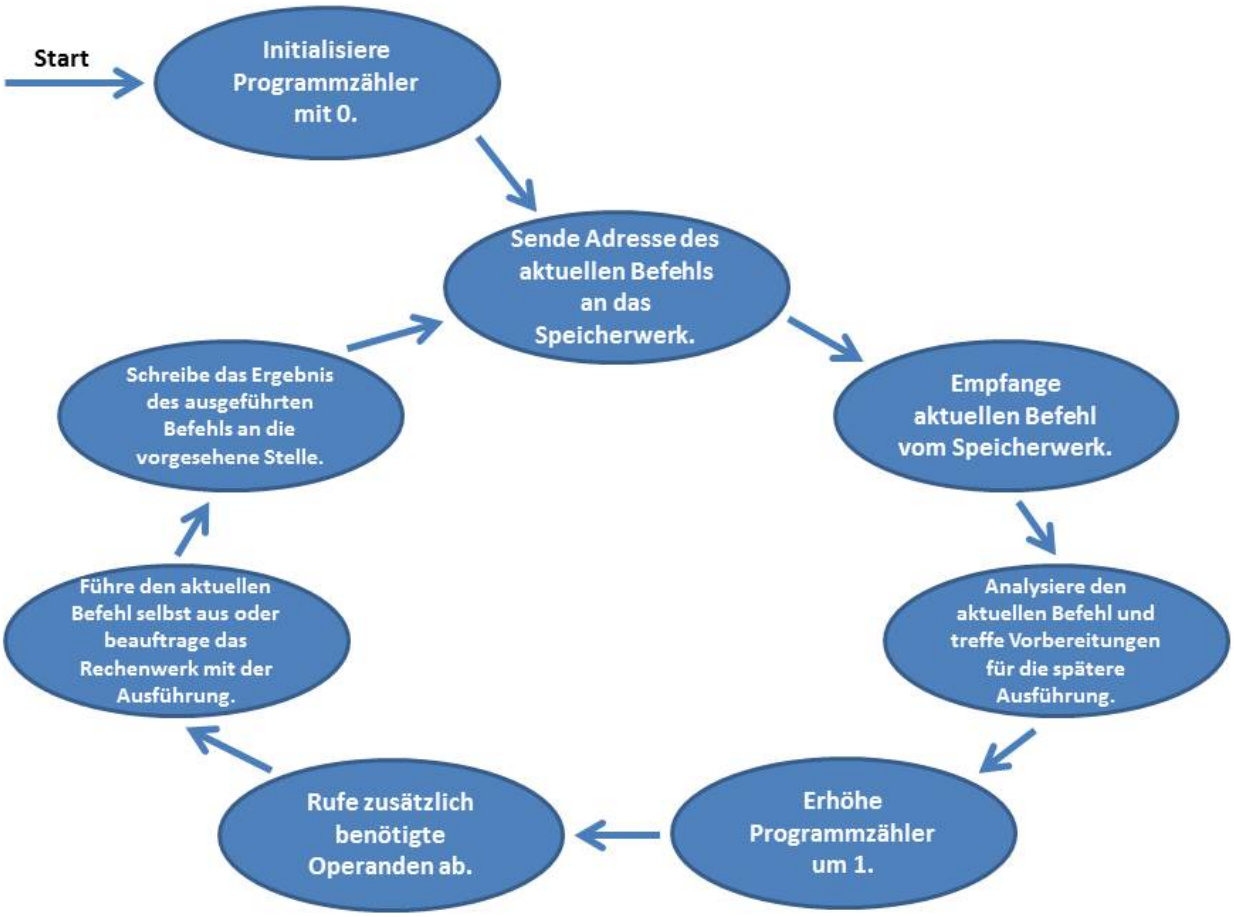
Ausschließlich für die Kommunikation zwischen Steuerwerk und Rechenwerk. Im MAR legt das Steuerwerk jeweils die Adresse ab, welche im Speicherwerk angesprochen werden soll.

- **Speicherdatenregister (Memory Data Register - MDR)**

Ausschließlich für die Kommunikation zwischen Steuerwerk und Rechenwerk. Bei einem Lesezugriff auf die Speicherzelle wird der vom Speicherwerk über den Datenbus bereitgestellte Wert im MDR abgelegt und kann von hier aus weiter verarbeitet werden. Bei einem Schreibzugriff muss sich im MDR der zu schreibende Wert befinden, so dass er über den Datenbus an das Speicherwerk übermittelt werden kann.

Prozesszyklus

1. Initialisiere das Befehlszählerregister (Program Counter- PC) mit 0 (Start)
2. Sende Adresse des aktuellen Befehls zum Speicherwerk
3. Empfange aktuellen Befehl vom Speicherwerk und speichere diesen in das Befehlsregister (Instruction Register - IR)
4. Analysiere aktuellen Befehl und treffe Vorbereitungen für die spätere Ausführung (Welcher Befehl und was ist dazu notwendig?)
5. Erhöhe den Befehlszähler (PC) um 1
6. Rufe zusätzlich benötigte Operanden ab (z.B.: Befehl ADD OP1 OP2)
7. Führe den Befehl selbst (Steuerwerk) aus oder beauftrage das Rechenwerk für die Ausführung
8. Schreibe das Ergebnis des ausgeführten Befehls an die vorgesehene Stelle





Arbeitsweise des Steuer- und

Rechenwerks



Arbeitsweise des Speicherwerks



Befehlszähler und Befehlsregister im Zusammenspiel mit dem Bus-System



Arbeitsweise des Steuerwerks

Die 7 Prinzipien der Von-Neumann-Architektur

- Rechner besteht aus fünf Funktionseinheiten
- Struktur des Rechners ist unabhängig vom zu bearbeitenden Problem. Zur Lösung eines Problems muss Programm im Speicher abgelegt werden.
- Programme, Daten und Ergebnisse werden im selben Speicher abgelegt.
- Der Speicher ist in fortlaufenden nummerierten Zellen unterteilt. Über die Adresse einer Speicherzelle kann auf den Inhalt zugegriffen werden.
- Aufeinanderfolgende Befehle eines Programms werden in aufeinanderfolgende Speicherzellen abgelegt.
- Durch Sprungbefehle kann von der gespeicherten Befehlsreihenfolge abgewichen werden.
- Es gibt zumindest
    - arithmetische Befehle (Addition, Subtraktion, Multiplikation)
    - logische Befehle (EQUAL, NOR, AND, OR)
    - Transportbefehle, z.B. von Speicher zu Rechenwerk und für Ein- und Ausgabe
- Alle Daten (Befehle, Adressen, usw.) werden binär kodiert

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_01

Last update: **2019/05/14 15:43**

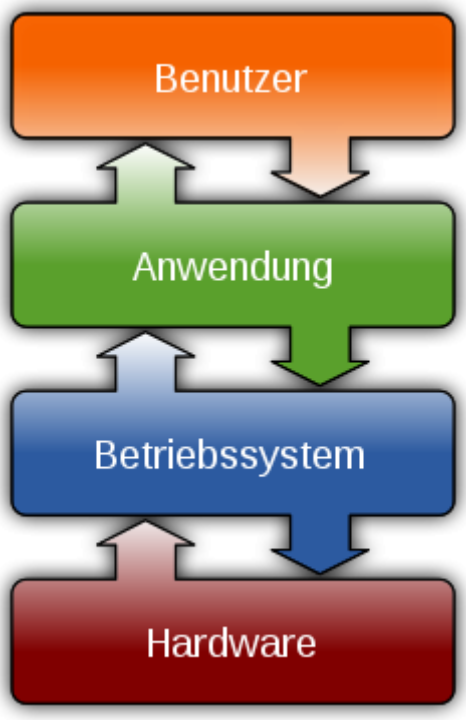
5.2) Zusammenhang Hardware - Betriebssysteme

Grundlegendes

Ein Rechner besteht zunächst aus den sichtbaren und greifbaren Komponenten der Hardware. Damit aber mit dem Rechner auch gearbeitet werden kann, benötigt er noch geeignete Programme. Nach dem Von-Neumannschen Prinzipien ist ein Rechner als Universalrechner konstruiert. Das heißt die Erledigung unterschiedlicher Aufgaben wird ausschließlich über Programme (auch Software genannt) gesteuert.

Da sich aber nicht jeder Softwareentwickler um die Details zur Veraltung der einzelnen Hardwarekomponenten (z.B.: Steuerungsinformationen für Plattenspeicher, Speicherverwaltung, Prozessverwaltung etc.) kümmern sollte, erscheint es sinnvoll diese Komponenten zentral und damit allgemein verfügbar bereitzustellen. Andernfalls würde jeder Entwickler viel Zeit verschwenden.

Somit entwickelte man eine Ebene (=Betriebssystemebene) auf der verschiedenste Verwaltungsprogramme bereits laufen und allen Anwendern bzw. Anwendungsprogrammen eine vereinfachte Sicht auf die Hardware zur Verfügung stellen.



Folgendes gilt, dass das Betriebssystem quasi der Mittler (=Übersetzer) zwischen Hardware und Benutzer ist. Dadurch kann die Komplexität der darunterliegenden Systemarchitektur verborgen werden und dem Anwender wird eine einfache und verständliche Schnittstelle (Interface) angeboten. Er kann sich somit voll und ganz auf die Implementierung seiner Funktion konzentrieren und muss sich nicht mit komplexen Maschinenbefehlen herumschlagen.

Des Weiteren verwaltet das Betriebssystem die Ressourcen des Rechners, also alle physikalischen

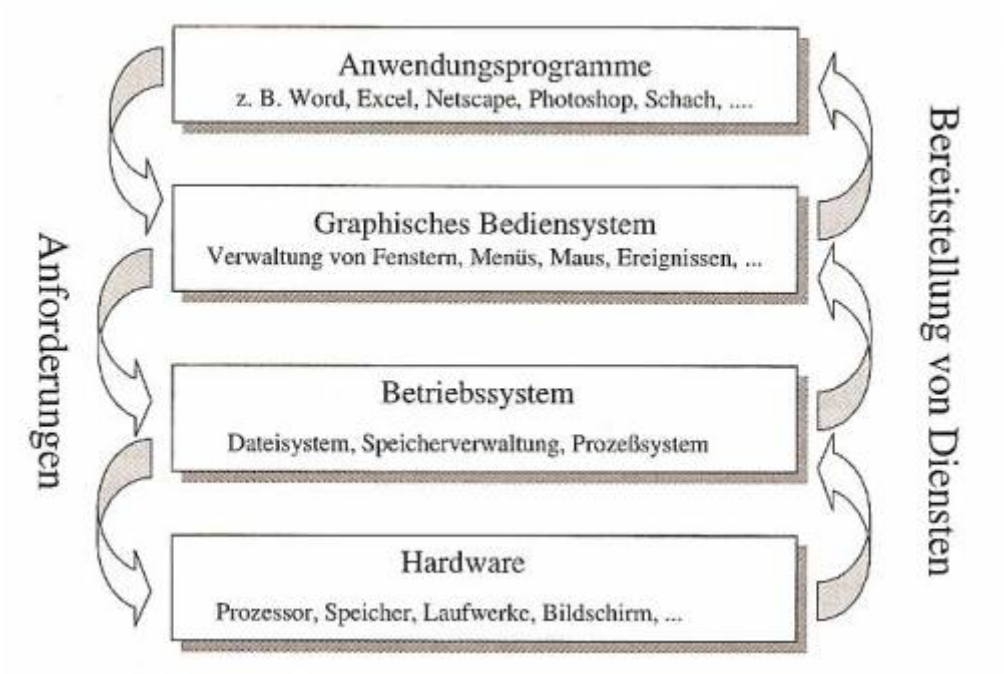
Geräte (Prozessoren, Speicher, Grafikkarte,..). Es teilt also die Ressourcen des Gesamtsystems gerecht an die konkurrierenden Anwenderprogramme auf. Ohne einer solchen Aufteilung wäre eine sinnvolle Benützung des PCs nicht möglich.

Als normaler Computeranwender befinden wir uns auf der **Benutzerebene**, wo der Rechner als Arbeitsgerät auch für technische Laien dienen muss.

Zwischen dem vom Anwender intuitiv zu bedienenden Rechner und den Fähigkeiten der Hardware klafft eine riesige Lücke, die vom

- Betriebssystem (Datei-, Prozess- und Speicherverwaltung) und dem
- grafischen Bediensystem (Menüs, Fenster, Maus)

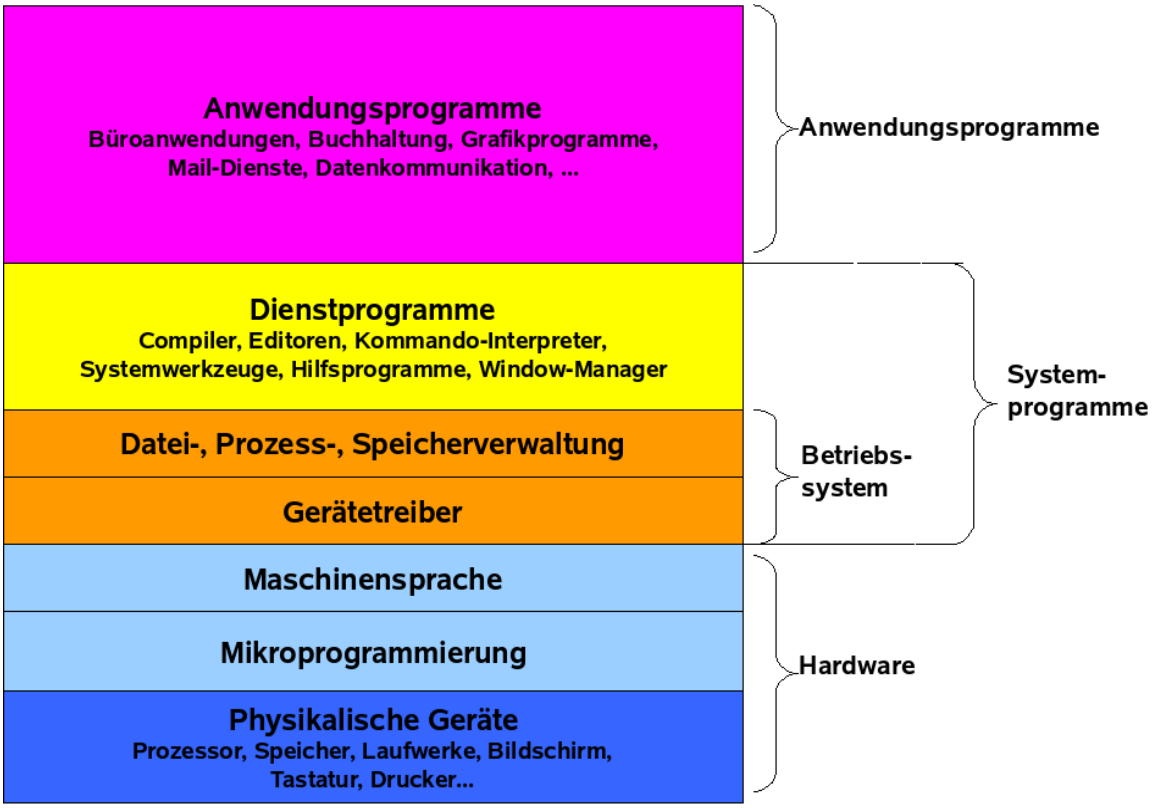
überbrückt wird.



Jede Schicht fordert von der niedrigeren Schicht Dienste an. Diese wiederum benötigt zur Erfüllung der Anforderungen selber Dienste von der nächsten Schicht. Auf diese Weise setzen sich die Anforderungen in die tiefen Schichten fort, bis die Hardware zu geeingeten Aktionen veranlasst wird.

Anders betrachtet, bietet jede Schicht der jeweils höherliegenden Schicht Dienste an.

Schichten eines Computersystems



Physikalische Geräte

Die unterste Schichte enthält die physikalischen Geräte. Sie besteht aus integrierten Schaltungen, Drähten, Stromversorgung usw.

Mikroprogrammierung/Mikrocode

Aufgabe des Mikrocodes ist die direkte Steuerung der physikalischen Geräte. In Microcode entworfene Programme werden von einem Interpreter in entsprechende Steueranweisungen übersetzt und an die Geräte übermittelt. Die Instruktionen der Maschinensprache (z.B.: ADD, MOVE, JUMP,...) werden in einzelne kleine Microcodeprogramme übersetzt.

Maschinensprache

Die Menge von Instruktionen die das Mikroprogramm ausführen kann wird als Maschinensprache bezeichnet. Maschinensprache besteht aus zahlreichen elementaren Maschinenbefehlen die im Prozessor in Microcode zu einem Programm übersetzt und ausgeführt werden. Wichtig ist, dass jeder Maschinenbefehl durch ein festgelegtes Mikroprogramm implementiert bzw. fest verdrahtet ist.

Da die Maschinensprache hardwarespezifisch ist, wird sie noch der Hardware zugeordnet.

Betriebssystem

Das Betriebssystem vermindert die Komplexität der Hardware und deren Verwaltung um dem Programmierer eine angemessene Menge an Instruktionen zur Verfügung zu stellen, mit denen er effizienter arbeiten kann.

Systemprogramme / Dienstprogramme

Das sind vom Betriebssystemhersteller mitgelieferte Programme wie z.B.: Window-Manager, Compiler für das Kompilieren von C-Programmen, Editoren, Systemwerkzeuge wie z.B.: Taskmanager, Explorer ...

Anwenderprogramme

Die oberste Schicht stellt den Benutzern des Systems Anwendersoftware zur Verfügung. Diese Programme setzen entweder auf den Dienstprogrammen oder unmittelbar auf dem Betriebssystem auf. Beispiele dafür sind:

- Browser
- Microsoft Office
- ....

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_02

Last update: **2019/05/14 15:43**

5.3) Aufgaben, Ziele & Eigenschaften von Betriebssystemen

Hauptaufgaben

1) Abstraktion (Vereinfachung) der Hardware --> abstrakte Programmierschnittstelle

2) Verwaltung von Systemressourcen (CPU, Speicher, Netzwerk, Drucker, Platten..)

- **Prozessverwaltung**
- **Speicherverwaltung**
- **Dateiverwaltung**

Der Rechner mit seinen Peripheriegeräten stellt eine Fülle von Ressourcen zu Verfügung, wie CPU, Hauptspeicher, Plattenspeicher, externe Geräte. Die Verwaltung dieser Ressourcen ist Aufgabe des Betriebssystems.

Der Aufruf eines Programms führt oft zu vielen gleichzeitig und unabhängig voneinander ablaufenden Teilprogrammen, auch **Prozesse** genannt.

Ein Prozess ist also ein eigenständiges Programm mit eigenem Speicherbereich, der vor dem Zugriff durch andere Prozesse geschützt ist. Allerdings ist es erlaubt, dass verschiedene Prozesse untereinander *kommunizieren* , also Daten untereinander austauschen.

Dabei hat das Betriebssystem die Aufgabe, die *Kommunikation* zwischen diesen Prozessen zu verwalten, damit sich die gleichzeitig aktiven Prozesse nicht untereinander beeinträchtigen oder gar zerstören. Das Betriebssystem verwaltet unter anderem also gleichzeitig aktive Prozesse, so dass einerseits keiner benachteiligt wird, andererseits aber kritische Prozesse mit Priorität behandelt werden.

Selbstverständlich können Prozesse aber nicht wirklich gleichzeitig ablaufen. Das Betriebssystem erzeugt aber eine scheinbare Parallelität dadurch, dass jeder Prozess immer wieder eine kurze Zeitspanne (wenige Millisekunden) an die Reihe kommt, dann unterbrochen wird, während andere Prozesse bedient werden. Nach kurzer Zeit ist der unterbrochene Prozess wieder an der Reihe und setzt seine Arbeit fort. Diesen Vorgang nennt man **Multitasking**.

Ähnlich verhält es sich mit der Verwaltung des Hauptspeichers, in dem nicht nur der Programmcode, sondern auch die Daten der vielen Prozesse gespeichert werden. Neuen Prozessen muss freier Hauptspeicher zugeteilt werden, der Speicher beendeter Prozesse muss wiederverwendet werden.

Die dritte wichtige Aufgabe ist die **Dateiverwaltung**. Damit ein Benutzer sich nicht darum kümmern muss, in welchen Sektoren auf welchen Spuren noch Platz ist um den gerade geschriebenen Text zu speichern, stellt das Betreibssystem das Konzept „Datei“ als Behälter für Daten aller Art zur Verfügung. Die Übersetzung von Dateien und ihren Namen in bestimmte Spuren, Sektoren und Köpfe der Festplatte nimmt das *Dateisystem* als Bestandteil des Betriebssystem vor.

Eigenschaften von Betriebssystemen

- Betriebssystem muss **änderbar** sein (durch neue Hardware, Software bzw. bei Sicherheitslücken)
- **modular** und **klar strukturiert** aufgebaut
- gut **dokumentiert** –> Schnittstellenbeschreibung!!

Ziele von Betriebssystemen

- **Bequemlichkeit** - aus Sicht des Benutzers
- **Effizienz** - Nutzung der Ressourcen
- **Entwicklungsfähigkeit** - Neue Dienste

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_03

Last update: **2019/05/14 15:44**

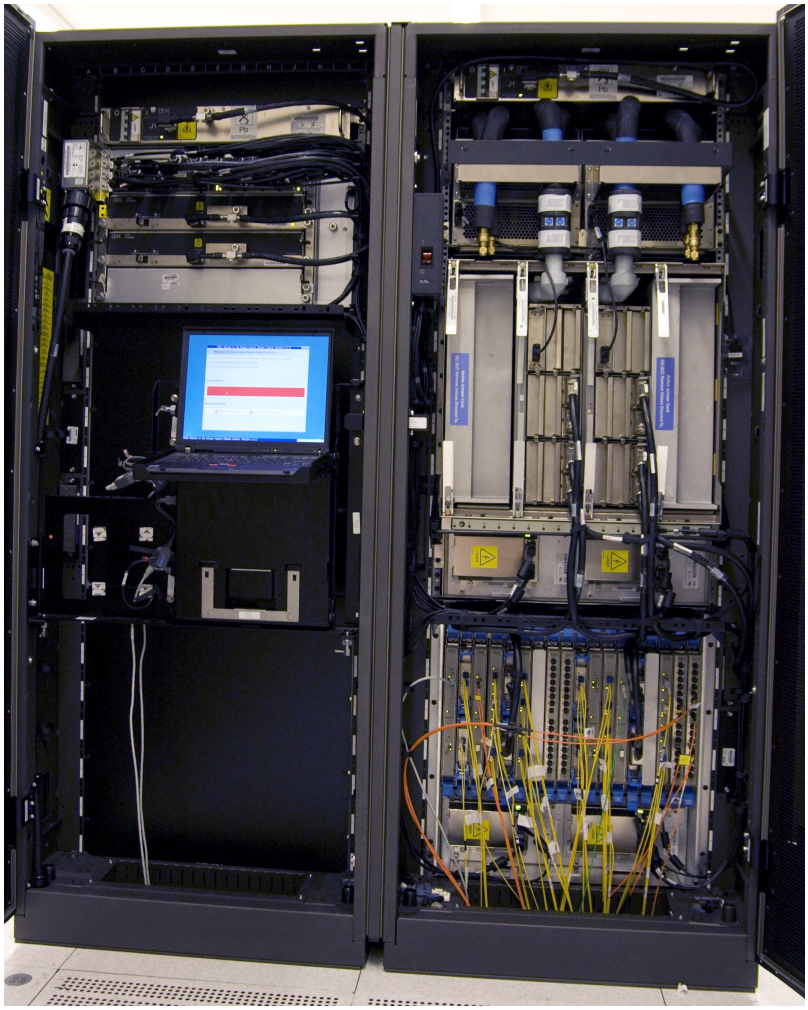
5.4) Betriebssystemarten

Man unterscheidet mehrere Arten von Betriebssystemen, die im Laufe der Zeit entstanden sind.

Mainframe-Betriebssysteme

- Betriebssysteme für Großrechner
- Einsatz: Webserver, Datenbankserver, E-Commerce, Business-to-Business (B2B)
- Viele Prozesse gleichzeitig mit hohem Bedarf an schneller Eingabe/Ausgabe
- Beispiele sind MVS, OS/390, z/OS 2.1 –> zumeist IBM-Großrechner





Server-Betriebssysteme

- Betriebssysteme die auf Servern laufen
- Bieten verschiedene Dienste im Netzwerk an (Dateidienste, Webdienste,...)
- Dienste stehen vielen Benutzern im Netz zur Verfügung z.B.: Netzlaufwerk
- Aktuelle Beispiele sind Windows Server 2016 bzw. Red Hat Enterprise Linux 7.2

PC(Desktop)-Betriebssysteme

- Betriebssystem für Personalcomputer
- Meist nur 1 oder wenige Benutzer (über Netzwerk)
- Einsatz: Programmierung, Textverarbeitung, Spiele, Internetzugriff
- Mehrere Programme pro Benutzer → quasi-parallel
- Aufteilung der Prozesse auf vorhandene Kerne
- Zuteilung der Systemressourcen
- Beispiele: Linux, Windows, Mac OS X

Echtzeit-Betriebssysteme

- Einhaltung von harten Zeitbedingungen

- Einsatz: Steuerung von maschinellen Fertigungsanlagen, Steuerung von Ampeln, Robotorsteuerung…
- Aktion in einem fest vorgegebenen Zeitintervall

Eingebettete Systeme / Embedded Systems

- = Computer die man nicht unbedingt gleich sieht
- Einsatz: Fernseher, Handy, Auto,…
- Meist Echtzeitanforderungen
- Wenig Ressourcen zur Verfügung (geringer Stromverbrauch, wenig Arbeitsspeicher)
- Beispiele: iOS, Android, Windows Phone

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_04

Last update: **2019/05/14 15:45**

5.5) Historische Entwicklung von Betriebssystemen

Arbeitsauftrag

Recherchiere im Internet hinsichtlich den historischen Entwicklungen von Betriebssystemen.

Beschreibe und erkläre die folgenden Entwicklungsstadien (= 4 Generationen) und gib die ungefähre Zeitspanne an:

1. Generation - Serielle Systeme
2. Generation - Einfache Stapelvearbeitungssysteme
3. Generation - ICS, Multiprogramming, Timesharing
4. Generation - Personal Computer

Das Ergebnis soll ein Word-Dokument (gegliedert in den 4 Generationen) im Umfang von 2-4 Seiten exklusive Titelblatt (Download unter

Titelblatt

- vor der Abgabe bitte den Namen im Titelblatt und den Dateinamen auf Betriebssysteme_NACHNAME.docx ändern) sein (d.h. mindestens 3 Seiten).

Letzter Abgabetermin ist der 21.02.2019 @ 23:59.

Die bis dahin abgegebenen Arbeitsaufträge werden bewertet und zählen zur Teilnote Praktische Arbeiten (PA)!!

Nicht abgegebene Arbeitsaufträge werden negativ beurteilt.

Musterlösung unter

Entwicklung Betriebssysteme

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_05

Last update: **2019/05/14 15:48**

5.6) Formatierung von Datenträgern

5.6.1) Stufen der Formatierung

Wenn man von Formatierung spricht, unterscheidet man dabei mehrere Stufen:

1. **Low-Level-Formatierung** = physikalische Einteilung eines Speichermediums
2. **Partitionierung** = logische Einteilung des Speichermediums in zusammenhängende Strukturen
3. **High-Level-Formatierung** = die logische Einteilung der Partitionsstruktur mit einem Dateisystem durch eine Software (meist durch das Betriebssystem).

Soweit durch die Beschaffenheit, industrielle Standards oder durch spezielle Verwendung die physikalische Einteilung des Mediums als Norm feststeht, ist eine separate Low-Level-Formatierung nicht erforderlich. In diesen Fällen können beide Vorgänge gleichzeitig vorgenommen werden, so zum Beispiel bei Disketten, CD-ROM, CD-RW oder DVD-ROM/RW.

5.6.1.1) Physikalische Formatierung (Low-Level-Formatierung)

Damit eine Festplatte logisch formatiert werden kann, muss sie physikalisch formatiert sein.

Die physikalische Formatierung eines Festplattenlaufwerks (auch als Zwei-Stufen-Formatierung bezeichnet) wird in der Regel vom Hersteller durchgeführt.

Durch die physikalische Formatierung wird eine Festplatte in ihre physikalischen Grundbausteine unterteilt: **Spuren**, **Sektoren** und **Zylinder**. Durch diese Elemente wird die Art und Weise vorgegeben, mit der Daten physikalisch auf der Festplatte aufgezeichnet und von dort gelesen werden können.

Spuren

Die **Spuren** sind konzentrische Kreispfade, die auf jede Scheibenseite geschrieben werden, wie auf einer Schallplatte oder einer CD. Die Spuren werden mit einer Nummer identifiziert, die mit Spur 0 am äußeren Rand einsetzt.

Zylinder

Der Spurensatz, der auf allen Seiten der Platte im gleichen Abstand von der Mitte angelegt ist, wird als “**Zylinder**” bezeichnet. Hardware und Software arbeiten häufig mit diesen Zylindern.

Sektoren

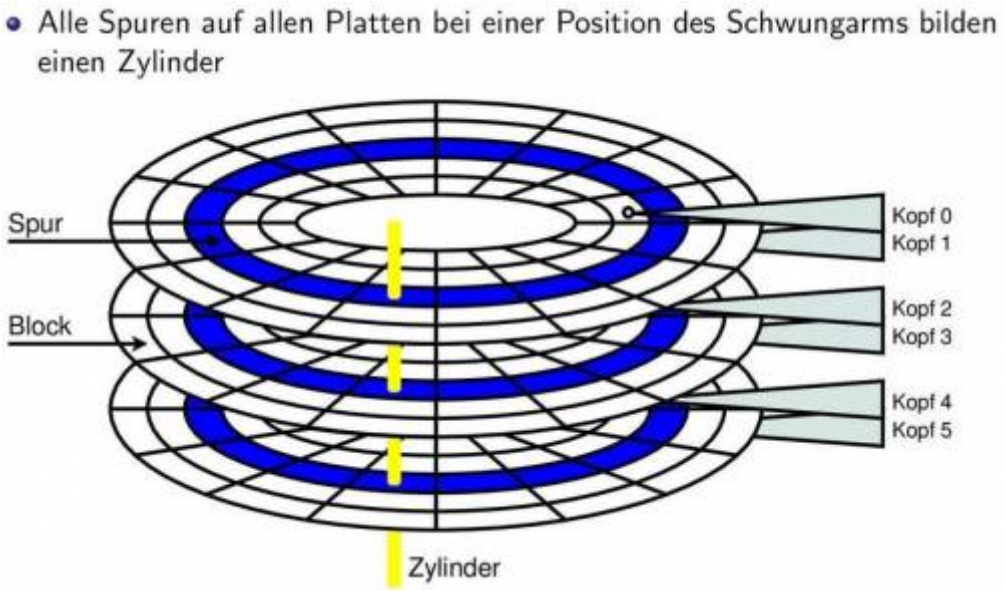
Die Spuren sind in Bereiche, sogenannte "**Sektoren**" oder „**Blöcke**" unterteilt, in denen eine festgeschriebene Datenmenge gespeichert wird. Die Sektoren werden in der Regel für eine Speicherkapazität von 512 Byte (ein Byte besteht aus 8 Bit) formatiert.

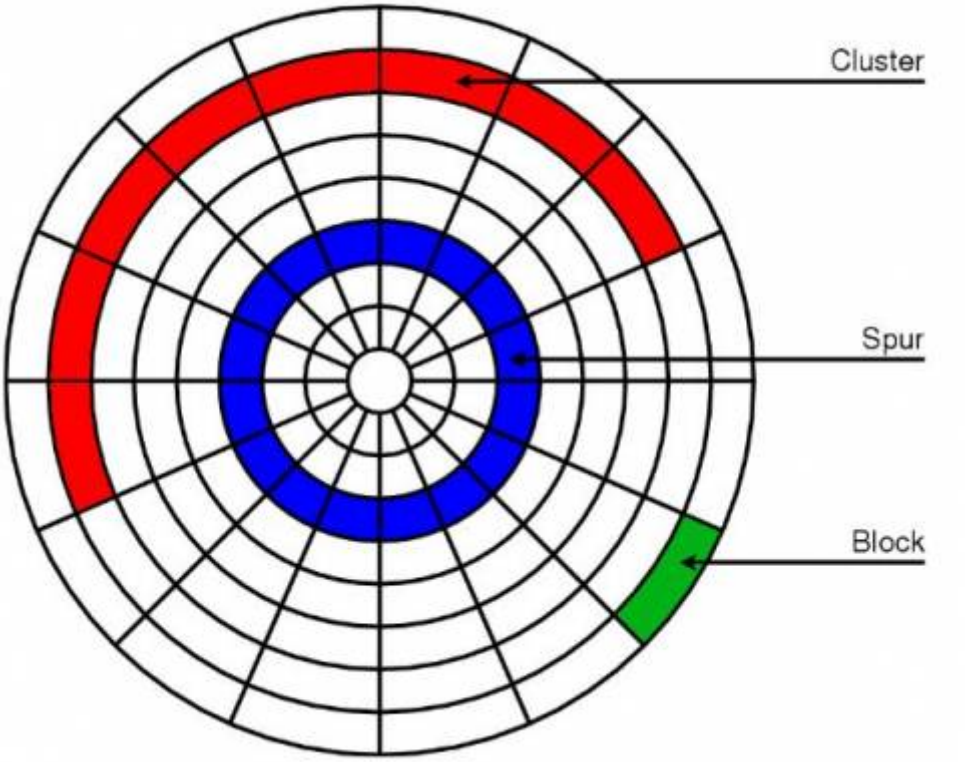
Fehlerhafte Sektoren

Nachdem ein Festplattenlaufwerk physikalisch formatiert wurde, kommt es gelegentlich vor, dass die magnetischen Eigenschaften der Eisenoxydschicht an einigen Stellen auf der Festplatte nach und nach an Wirksamkeit verlieren. Dies hat zur Folge, dass es für die Schreib-/Leseköpfe der Festplatte in zunehmenden Maße schwieriger wird, ein Bit-Muster auf der Festplatte zu speichern, das später von der Festplatte gelesen werden kann.

Wenn dieser Prozess einsetzt, bezeichnet man die Sektoren, in denen Daten nicht mehr zuverlässig gespeichert werden können, als "fehlerhafte Sektoren". Glücklicherweise ist die Qualität moderner Festplatten so hoch, dass solche "fehlerhaften Sektoren" nur sehr selten auftreten.

Darüber hinaus sind die modernen Computer normalerweise in der Lage, einen fehlerhaften Sektor zu identifizieren und ihn als solchen zu kennzeichnen, damit er nicht mehr genutzt wird; es wird dann ein alternativer Sektor benutzt.





Adressierung der Daten auf einer (magnetischen) Festplatte

CHS - Adressierung

Bei Festplatten mit einer Kapazit< 8 GB werden die einzelnen Sektoren mit der sogenannten "**Zylinder-Kopf-Sektor-Adressierung(Cylinder-Head-Sector-Adressierung)**" durchgeführt.

Jeder Sektor lässt sich mit **CHS** klar lokalisieren und adressieren.

Einschränkungen durch das BIOS

- Zylinder = 10 Bits
- Kopf = 4 Bits –> später 8 Bits (erweitertes CHS)
- Sektor/Spur = 8 Bits

Maximaler Speicherplatz früher: 1.024 Zylinder * 16 Köpfe * 63 Sektoren/Spur * 512 Byte/Sektor = 528.482.304 Byte = 504 MB

Maximaler Speicherplatz heute: 1.024 Zylinder * 255 Köpfe * 63 Sektoren/Spur * 512 Byte/Sektor = 8.422.686.720 Byte = 7,844 GB

- Größe einer Spur berechnen?
- Größe einer Scheibe berechnen?

LBA- Adressierung

Seit der Einführung der logischen Blockadressierung (**Logical Block Addressing (LBA)**) 1995 werden alle Sektoren auf der Festplatte von 0 beginnend durchnummeriert. Durch LBA wurde das

BIOS - und CHS-Korsett umgangen und es kann damit jede derzeit gängige Festplatte voll adressiert werden. Die Zahlen zu Zylinder, Köpfen und Sektoren auf der Festplatte haben heute nichts mehr mit der tatächlichen Lage der Sektoren auf der Festplatte zu tun. Es handelt sich um eine logische Plattengeometrie, die nur aus Kompatibilitätsgründen existiert

5.6.1.2) Partitionierung von Festplatten

Eine Partition ist ein zusammenhängender Bereich einer Festplatte, der in der Regel ein Dateisystem enthält.
Wenn man einen PC oder ein Notebook mit vorinstalliertem Windows kaufen, enthält die Festplatte zumeist zwei Partitionen: eine winzige Partition mit Windows-Boot-Dateien und eine zweite Partition, die den Rest der Festplatte füllt und Windows enthält. Unter Umständen kann es auch weitere Partitionen geben, die beispielsweise ein Recovery-System enthalten.

Um mehrere Betriebssysteme (Windows, Linux etc.) auf einem Rechner zu installieren, benötigt man mehrere Partitionen. Für jedes Betriebssystem ist mindestens eine Partition erforderlich; für Linux sind sogar mehrere Partitionen sinnvoll.

Master-Boot-Record (MBR)

Der Master-Boot-Record ist der **erste Sektor** auf der Festplatte. Er beinhaltet zum einen den „**Bootstrap**“. Dies ist ein Programm, das von dem BIOS aufgerufen wird, um das eigentliche Betriebssystem zu laden. Zum anderen enthält dieser Sektor auch eine Beschreibung, ob / wie die Festplatte in unterschiedliche Bereiche (Partitionen) unterteilt ist. Diese Beschreibung erfolgt in der sogenannten „Partitionstabelle“. Sie enthält für jede Partition einen Eintrag. Dieser besteht aus der Lage der Partition auf der Festplatte und dem „Typ“ dieser Partition.
Die Master-Boot-Record Partitionstabelle ist eine Tabelle, welche die Unterteilung der Festplatte in Partitionen beschreibt. Sie ist seit der Verbreitung von Festplatten im Jahr 1980 fest definiert und wird von allen Betriebssystemen zwingend vorausgesetzt.

Aus historischen Gründen kann diese Partitionstabelle nur vier Einträge (primäre Partitionen oder erweiterte Partitionen) aufnehmen. Das Format dieses Master-Boot-Records (Bootstrap / Partitionstabelle) ist fest definiert und wird von allen Betriebssystemen zwingend vorausgesetzt.

Partitionsarten

Es gibt im wesentlichen zwei Partitionsarten: **Primärpartitionen** und **erweiterte Partitionen**. Darüber hinaus lassen sich erweiterte Partitionen noch weiter in **logische Partitionen** unterteilen. Sie können bis zu vier Hauptpartitionen auf dem Festplattenlaufwerk definieren, wovon eine als erweiterte Partition definiert werden kann, d.h. maximal vier Primärpartitionen oder drei Primärpartitionen und eine erweiterte Partition.

a) Primärpartitionen

In einer Primärpartition können Sie jedes beliebige Betriebssystem sowie Datendateien wie z.B. Programm- und Benutzerdateien speichern. Eine Primärpartition ist logisch formatiert, damit sich ein Dateisystem darauf definieren lässt, das mit dem auf der Partition installierten Betriebssystem kompatibel ist.

Wenn Sie mehrere Betriebssysteme auf Ihrem Festplattenlaufwerk installieren müssen, ist es in der Regel notwendig, mehrere Primärpartitionen zu erstellen, da die meisten Betriebssysteme (vor allem Windows) nur von einer Primärpartition gebootet werden können. Pro Festplatte ist es möglich 4 primäre Partitionen einzurichten, ohne den Bootsektor der Festplatte anzupassen.

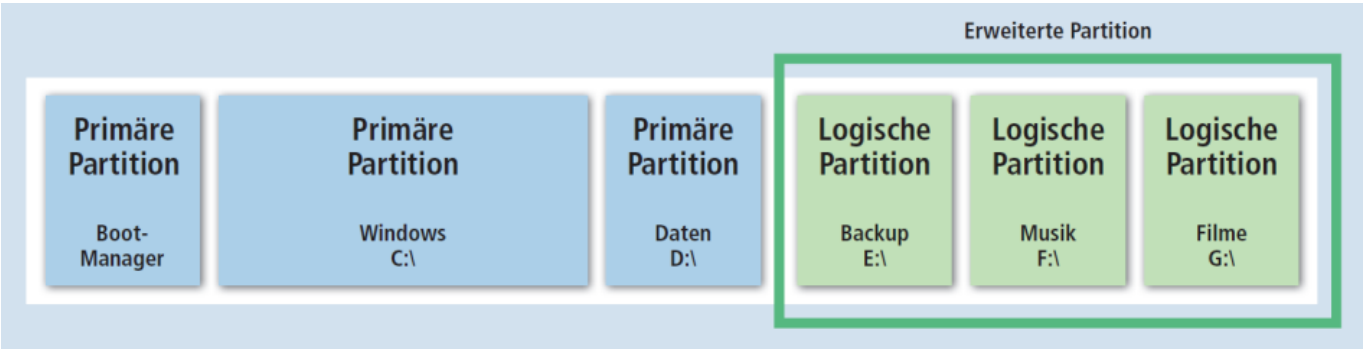
b) Erweiterte Partitionen

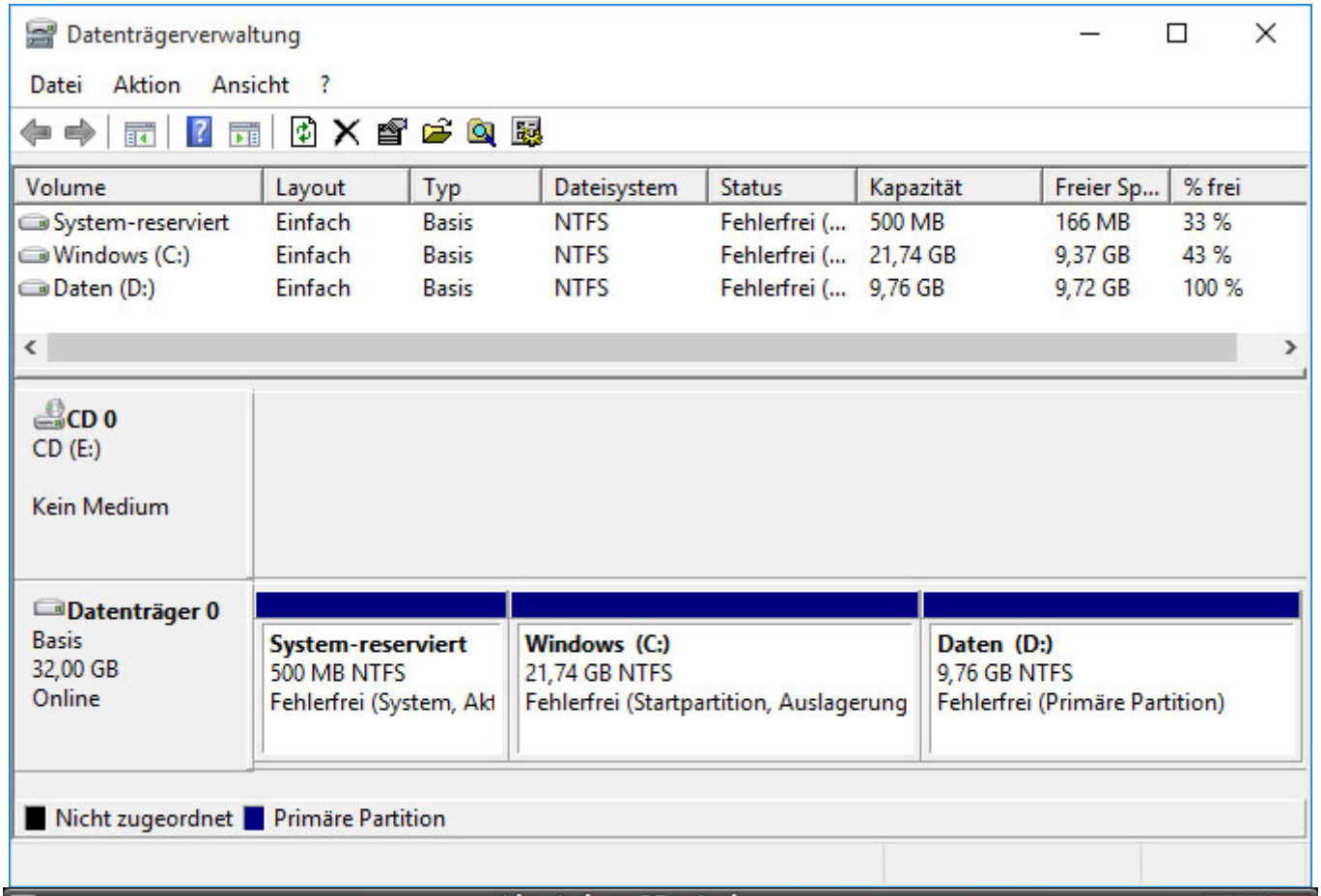
Die erweiterte Partition wurde entwickelt, um die willkürliche 4-Partitionen-Begrenzung zu umgehen. Es handelt sich im Prinzip um einen Bereich, in dem Sie den Datenträgerspeicher weiter physikalisch unterteilen können, indem Sie eine unbegrenzte Anzahl logischer Partitionen definieren (weitere physikalische Unterteilungen des Datenträgerspeichers).

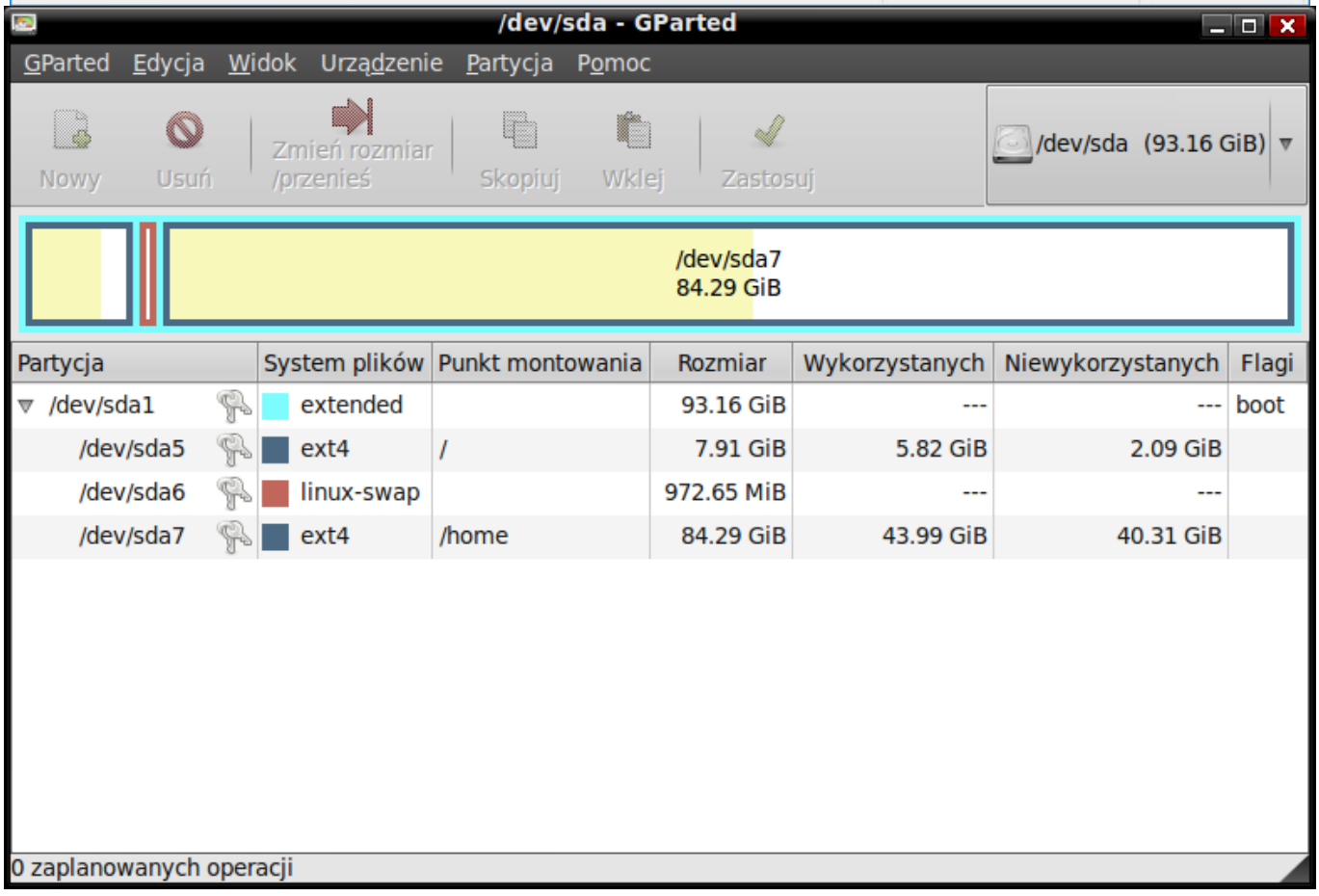
Eine erweiterte Partition dient nur indirekt der Datenspeicherung. Der Benutzer muss in der erweiterten Partition logische Partitionen definieren, in denen die Daten gespeichert werden. Logische Partitionen müssen logisch formatiert sein; auf jeder logischen Partition kann ein anderes Dateisystem definiert sein. Nach der logischen Formatierung gilt jede logische Partition als separater Datenträger.

c) Logische Partitionen

Logische Partitionen können nur innerhalb einer erweiterten Partition definiert werden und sind nur für die Speicherung von Datendateien und Betriebssystemen vorgesehen, die von einer logischen Partition gebootet werden können (z.B. Linux und Windows NT). Betriebssysteme, die von einer logischen Partition gebootet werden können, sollten gewöhnlich in einer logischen Partition installiert werden; damit können Primärpartitionen für andere Zwecke freigehalten werden.







Frage: Wie finde ich die aktuelle Partitionierung von meinem PC?

Gründe für Partitionen

Das Problem früher waren die Festplatten-Controller die nicht in der Lage waren, einen größeren Adressbereich anzusprechen. Und, der technische Fortschritt und die höheren Kapazitäten von Festplatten wurden schneller eingeführt als neue und bessere Dateisysteme. Vor allem unter Windows-Betriebssystemen war das FAT-Dateisystem (File Allocation Table) lange führend. FAT ermöglichte durch die Zusammenführung mehrerer Blöcke zu einer logischen Ansprecheinheit (Cluster), um die Adressierungsbeschränkung zu umgehen. Es hatte den Nachteil, dass es Festplatten nur bis zu einer bestimmten Kapazität verwalten und die Dateien nicht besonders platzsparend speichern konnte.

Beschränkungen der Partitionsgröße vom Dateisystem:

- FAT16 ⇒ max. Partitionsgröße = 2 GByte
- Nachfolger FAT32 ⇒ 2 TByte
- Über 32 GByte verwendet man üblicherweise das Dateisystem NTFS.

Heutige Gründe für Partitionierungen:

- Installation mehrerer Betriebssysteme
- Einrichten verschiedener Dateisysteme
- Reservierung für eine Windows- oder Linux-Auslagerungsdatei (SWAP-Partition)
- Installationsdateien
- sehr großen Speicher verwalten
- Trennung von Programmen und Daten

5.6.1.3) Logische Formatierung (High-Level-Formatierung)

Ein physikalisch formatiertes Festplattenlaufwerk muss noch logisch formatiert werden. Durch die logische Formatierung wird ein **Dateisystem** auf der Festplatte definiert. Erst ein Dateisystem gestattet es einem Betriebssystem wie z.B. DOS, OS/2, Windows 95/98, Windows NT/Windows 2000/WindowsXP/Windows7/Windows8 oder Linux), den verfügbaren Speicher für das Speichern und Laden von Dateien zu nutzen.

Vor der logischen Formatierung kann eine Festplatte in **Partitionen** unterteilt werden. Auf jeder Partition kann ein Dateisystem (logisches Format) definiert werden. Eine Festplattenpartition, die bereits logisch formatiert ist, wird als Datenträger (Volume) bezeichnet. Als Teil des Formatierungsvorgangs werden Sie durch das Formatierungsprogramm aufgefordert, der Partition einen Namen zuzuweisen, die sogenannte “Datenträgerbezeichnung”. Mit diesem Namen können Sie den Datenträger (die Partition) später identifizieren.

Normal- und Schnellformatierung

Es gibt zwei unterschiedliche Methoden, den Datenträger high-level zu formatieren:

Die Schnell- und die Normalformatierung.

Sie unterscheiden sich in folgenden Punkten:

Normalformatierung – Es wird eine Suche nach fehlerhaften Sektoren durchgeführt. Diese Suche nimmt den Großteil der Zeit in Anspruch. Anschließend erfolgt das Schreiben der Dateisystem-Metadaten und somit die Löschung der vorhandenen Dateien.

Schnellformatierung – Es werden zwar die Dateien aus dem Inhaltsverzeichnis des Datenträgers beziehungsweise der betreffenden Partition entfernt, die Suche nach fehlerhaften Sektoren wird jedoch ausgelassen.

Bei einer High-Level-Formatierung sind in dem neuen Dateisystem die alten Daten nicht verfügbar, da sie nicht mehr durch entsprechende Verweise im Dateisystem referenziert werden. Sie werden jedoch nicht notwendigerweise gelöscht. Meistens verbleiben sie rein physikalisch auf der Festplatte, bis sie mit neuen Daten überschrieben werden. Solange die verwendeten Datenblöcke nicht erneut beschrieben werden, ist mit entsprechender Software eine weitgehende Wiederherstellung noch möglich, gestaltet sich jedoch schwerer, als wenn die Dateien einfach nur gelöscht würden. Nicht selten können daher via Software nur die einzelnen Dateien, nicht aber die Ordnerstruktur wiederhergestellt werden.

Dateisysteme

Jeder Computer benötigt ein funktionsfähiges Betriebssystem, auch wenn es lediglich aus den DOS- oder Windows 98-Bootdateien besteht. Verschiedene Betriebssysteme verwenden unterschiedliche Dateisysteme, um auf Datenträger zuzugreifen und die Speicherorte von Daten zu ermitteln. Das Dateisystem bestimmt hierbei die Art der Verwaltung und des Zugriffs auf Dateien auf einem Datenträger.

Die Methoden und Datenstrukturen, die ein Betriebssystem verwendet, um Speicherorte von Dateien auf einer Partition zu ermitteln, werden also als das **Dateisystem** definiert. Bei den hier besprochenen Dateisystemen handelt es sich um FAT, FAT32, HPFS, NTFS, ReFS, Linux Ext2 und Linux ReiserFS.

Windows

FAT16

Das FAT16-Dateisystem ist das ursprünglich von MS-DOS für die Dateiverwaltung verwendete System. Das FAT (File Allocation Table) ist eine Datenstruktur, die MS-DOS beim Formatieren eines Datenträgers auf demselben erstellt. FAT16-Partitionen können bis zu 2 GB groß sein. Die Größe einer Partition bestimmt die Größe der Cluster, in denen Daten gespeichert werden. Bei einer Partitionsgröße von über einem GB kann dies zu einem erheblichen Speicherverlust führen. FAT16-Partitionen werden bei den meisten Versionen von DOS und Windows 95 eingesetzt und von den folgenden Betriebssystemen unterstützt: Windows 95, Windows 98, Windows NT, Windows 2000, DOS (inklusive MS-DOS, PC-DOS und DR-DOS), OS/2, Linux und viele andere Betriebssysteme.

FAT32

FAT32 ist ein Dateisystem, das von Microsoft für Windows 95, Version B (OSR2), entwickelt wurde. Es verfügt über die Fähigkeit, 32-Bit-Clusteradressen zu verwalten und unterstützt Partitionen von bis zu 8 GB bei einer Clustergröße von nur 4 KB. Partitionen von bis zu 16 GB können bei einer Clustergröße von 8 KB verwaltet werden, was den unnötigen Verbrauch von Festplattenspeicher aufgrund ineffizienter Clusternutzung deutlich reduzieren kann. FAT32-Partitionen kommen ausschließlich unter Windows 95B oder höher, Windows 98, Windows 2000 und unter neueren Versionen von Linux zum Einsatz.

NTFS

NTFS (New Technology File System) wurde von Microsoft speziell für die Verwendung mit Windows NT, Windows 2000 und Windows XP entwickelt und unterstützt die verbesserten Sicherheitsmerkmale, die über dieses Betriebssystem zur Verfügung stehen. NTFS unterstützt vollständige Zugriffskontrolle, Dateisystemwiederherstellung und besonders große Datenträger. NTFS-Partitionen werden ausschließlich unter Windows NT (Workstation und Server) sowie unter Windows 2000, Windows XP, Windows 7 und Windows 8 eingesetzt.

ReFS

ReFS (Resilient File System) ist ein Dateisystem von Microsoft, wobei engl. „resilient“ für „elastisch“, „belastbar“ oder gar „unverwüstlich“ steht. Es wurde mit den Betriebssystemen Windows 8 und Windows Server 2012 als Ergänzung zum NTFS-Dateisystem eingeführt.

Linux

Ext2

Das Ext2-Dateisystem war bis ca. 2002 das Standardsystem für Linux. Ist außerordentlich stabil und sicher, aber leider muss nach einem Systemabsturz (z.B. Stromausfall) eine sehr zeitaufwändige Überprüfung des gesamten Dateisystems durchgeführt werden.

Ext3

Das Ext3-Dateisystem ist z.Z. das Standardsystem für Linux. Weitgehend kompatibel zu Ext2, hat aber den Vorteil das es über eine **Journaling-Funktion** verfügt. D.h. nach einem Stromausfall entfällt eine langwierige Überprüfung des gesamten Dateisystems.

Ext4

Ext4 (Fourth Extended File System): weiterentwickelte Variante von Ext3, unter anderem mit

erweiterten Limits.

ReiserFS

Das ReiserFS ist eine Weiterentwicklung des Ext2-Dateisystem ist das Standardsystem für Linux (Verbesserte Dateisystemwiederherstellung nach Unterbrechungen im laufenden Betrieb). Linux ReiserFS-Partitionen werden ausschließlich unter Linux-Installationen verwendet.

MAC

HFS

- HFS

Weiterführende Informationen

Festplatten Fakten

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_06

Last update: **2019/05/14 15:49**

5.7) Planung eines Systems

Man sollte sich vor der Installation der Betriebssysteme (oder eines Bootmanagers) einige Gedanken über den Aufbau des Systems machen: Hierbei ist am wichtigsten, sich zuerst überlegen, welche Betriebssysteme eingesetzt werden sollen.
Als nächstes ist abzuklären, wie viel Festplattenplatz für welches Betriebssystem benötigt wird(Größe der Festplatte). Hierzu macht der Hersteller des Betriebssystem bereits einen Vorschlag, der als Minimum eingehalten werde sollte.
Unter Umständen kann es sinnvoll sein, einen Teil der Festplatte (eine Partition) ausschließlich dazu zu verwenden, Daten unter allen, oder zumindest unter mehreren Betriebssystemen, zur Verfügung zu stellen. Am besten lässt man einen Teil der Festplatte unbenutzt (falls diese groß genug ist), damit dieser später für Erweiterungen genutzt werden kann.

Besonderheiten und Probleme der einzelnen Betriebssysteme

Probleme mit DOS / Windows 95/98/ME

Bei der Verwendung von FAT 16 durfte die Partition nicht größer als 2 GB sein.
Von der 2. Festplatte kann nur gebootet werden, wenn auf der 1. Festplatte keine primäre Partition sichtbar ist.

Probleme mit Windows NT/2000/XP

- Windows 2000/XP: Das Setup-Programm von Windows 2000/XP überschreibt während der

Installation den Master-Boot-Record.

- Unter Windows 2000/XP kann eine einmal angelegte Partition nicht mehr verkleinert werden.

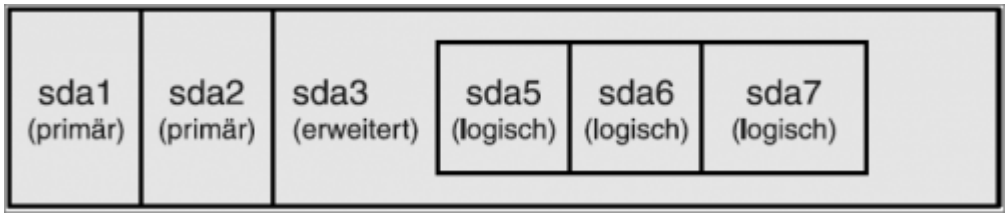
Windows 7, Windows 8

- ab Windows 7 können Partitionen auch verkleinert werden:
    - Datenträgerverwaltung - Rechtsklick auf die Partition - Volume verkleinern

Linux

Linux kann auf jeder Festplatte maximal 15 Partitionen ansprechen, davon maximal 11 logische Partitionen.
Unter Linux erfolgt der interne Zugriff auf Festplatten bzw. deren Partitionen über so genannte Device-Dateien: Die Festplatten erhalten der Reihe nach die Bezeichnungen /dev/sda, /dev/sdb, /dev/sdc etc.

Um eine einzelne Partition und nicht die ganze Festplatte anzusprechen, wird der Name um die Partitionsnummer ergänzt. Die Zahlen 1 bis 4 sind für primäre und erweiterte Partitionen reserviert. Logische Partitionen beginnen mit der Nummer 5 – auch dann, wenn es weniger als vier primäre oder erweiterte Partitionen gibt. Die folgende Abbildung veranschaulicht die Nummerierung: Auf der Festplatte gibt es zwei primäre Partitionen und eine erweiterte Partition, die drei logische Partitionen enthält.



Die maximale Partitionsgröße beträgt 2 TByte. Da es mittlerweile Festplatten mit mehr als 2 TByte Speichervolumen gibt, ist eine sinnvolle Nutzung von Festplatten mit mehr als 2 TByte nur noch mit GPT-Partitionstabellen möglich.

Anzahl und Größe der Linux-Partitionen

Leider gibt es auf die Frage, wie viele Linux-Partitionen ein System haben sollte, keine allgemein gültige Antwort.

Die **Systempartition** ist die einzige Partition, die man unbedingt benötigt. Sie nimmt das Linux-System mit all seinen Programmen auf. Diese Partition bekommt immer den Namen /. Dabei handelt es sich genau genommen um den Punkt, an dem die Partition in das Dateisystem eingebunden wird (den mount-Punkt).
Eine vernünftige Größe für die Installation und den Betrieb einer gängigen Distribution liegt bei 10 bis 20 GByte.

Mit einer **Datenpartition** trennt man den Speicherort für die Systemdateien und für die eigenen Dateien. Das hat einen wesentlichen Vorteil: Man kann später problemlos eine neue Distribution in die Systempartition installieren, ohne die davon getrennte Datenpartition mit Ihren eigenen Daten zu gefährden. Bei der Datenpartition wird **/home** als Name bzw. mount-Punkt verwendet, weswegen oft auch von einer Home-Partition die Rede ist. Die Größe hängt von den eigenen Bedürfnissen ab.

Die **Swap-Partition** ist das Gegenstück zur Auslagerungsdatei von Windows: Wenn Linux zu wenig RAM hat, lagert es Teile des gerade nicht benötigten RAM-Inhalts dorthin aus. Im Gegensatz zu den anderen Partitionen bekommt die Swap-Partition keinen Namen (keinen mount-Punkt). Die Größe sollte das ein- bis zweifache des RAMs betragen.

Mehrere Betriebssysteme verwalten

Es gibt verschiedene Möglichkeiten, mehrere Betriebssysteme zu verwalten:

- Mit Hilfe von Bootmanagementprogrammen, wie beispielsweise BootMagic von PowerQuest, Bootstar von Star-Tools, Acronis von SWSoft oder GRUB von SuSE-Linux
- Mit Hilfe einer von einem Betriebssystem verwalteten doppelten Bootkonfiguration, wie

beispielsweise der Boot Loader von Windows NT oder GRUB (bzw. mit dem älteren) LILO unter Linux.
- Manuell, indem Sie ein Betriebssystem als „aktiv“ einstellen. Nehmen Sie diese Einstellung über ein Dienstprogramm wie PQBoot vor, oder bearbeiten Sie den Masterbootdatensatz von Hand (nicht zu empfehlen!).

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_07

Last update: **2019/05/14 15:49**

5.8) Verschiedene Betriebssystem

Windows-Betriebssysteme

- MS-DOS / DOS
- Windows 3.11
- Windows 95
- Windows 98
- Windows NT
- Windows 2000
- Windows XP
- Windows Vista
- Windows 7
- Windows 8
- Windows 10

Linux-Betriebssysteme

- OpenSuSE Linux
  - Open Suse für Profis
- SuSE Linux

Ubuntu

- Ubuntu
- www.ubuntu.com
  - Ubuntu Austria
  - Ubuntu Server
- Kubuntu
- Edubuntu
- Xubuntu
- Gobuntu
- Knoppix
- RedHat Linux
- Debian Linux
- Xubuntu von USB-Stick: Pendrive Linux

Macintosh-Betriebssysteme

- Leopard
- Snow Leopard

Betriebssystem-Virtualisierung

- Virtualisierung

Online-Betriebssysteme

- Ein Online-Desktop von eyeOS

Portable-Betriebssysteme

- PC am Datenstick
- Linux Advanced

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_08

Last update: **2019/05/14 15:50**

5.9) Windows

Microsoft Windows bzw. Windows ist **ursprünglich** eine von Microsoft entwickelte **grafische Benutzeroberfläche**, aus der **später** eine Reihe von eigenständigen **Betriebssystemen** entstanden sind.

Das ursprüngliche Windows war eine **grafische Erweiterung des Betriebssystems MS-DOS**, wie beispielsweise auch GEM oder PC/GEOS. Dieses System wurde ab Windows 95 um einen überarbeiteten Kernel, die 32-Bit-API Win32 und Internetfähigkeit erweitert und mit den Versionen Windows 98, 98 SE sowie ME weitergeführt. Für diese Betriebssystemfamilie hat sich die Sammelbezeichnung Windows 9x etabliert. Parallel dazu wurde unter der Leitung von David N. Cutler seit 1988 das auf den Konzepten des Betriebssystems VMS basierende Windows NT entwickelt. Seit Windows XP vertreibt Microsoft für den Desktop nur noch Nachfolger von Windows NT, da Windows 9x aufgrund technischer Schwächen aufgegeben wurde. Seitdem heißt das Betriebssystem als Ganzes nur noch Windows (englisch für „Fenster“). Der Name rührt daher, dass die Benutzeroberflächen von Anwendungen auf dem Bildschirm als rechteckige Fenster dargestellt werden.

Windows-Betriebssysteme sind vor allem auf **Personal Computern und Servern verbreitet**; daneben existieren Varianten für Geräte wie Smartphones oder PDAs sowie für spezielle Embedded Devices wie etwa vollelektronische Messgeräte und Einzelhandels-Kassensysteme oder für die Anwendung in Kraftfahrzeugen. Kevin Turner, der Chief Operating Officer von Microsoft, nannte auf der Worldwide Partner Conference 2014 einen Gesamt-Marktanteil von 14 Prozent für alle Windows-Varianten.

- Kommandozeile
- 5.9.1) MS-DOS
- 5.9.2) Benutzerverwaltung
- 5.9.3) Dateimanagement
- 5.9.4) Sicherung & Wiederherstellung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_09

Last update: **2019/05/14 15:53**

5.9.1) Das Betriebssystem MSDOS

MSDOS (DOS = Disk Operating System) ist ein Betriebssystem, das von der Firma Microsoft in den USA entwickelt wurde und früher zu den am meisten verbreiteten Betriebssystemen zählte. Die Entwicklungsgeschichte reicht sehr lange zurück, die erste Version gab es 1981.

Mittlerweile ist MSDOS weitgehend duch die neuen Windows-Betriebssysteme abgelöst worden - trotzdem wird DOS für bestimmte Aufgaben auch heute noch verwendet, wie z.B. zur Erstellung von Batch-Dateien. Die Kenntnisse über DOS könnten auch insbesondere bei einem Crash nützlich sein, wenn Windows z.B. nicht mehr startet oder man ein Uralt-DOS-PC-Spiel noch mal spielen will.

Arbeiten mit DOS auf der Systemkonsole

Speicher zuvor folgenden Ordner auf deinem Verzeichnis: dos-befehle.zip

Start der Systemkonsole

- Start - Ausführen: cmd
- Start - Programme - Zubehör - Eingabeaufforderung

- Vollbildmodus: ALT + ENTER
- Beenden: exit

Einige wichtige DOS-Befehle

Allgemeine Befehle

Befehl	**Beschreibung**
help	zeigt eine Liste aller MS-DOS-Befehle und eine kurze Beschreibung
help *Befehl*	man erhält Informationen zu einem bestimmten Befehl
dir	zeigt Inhaltsverzeichnis des aktuellen Verzeichnisses an
dir i:	zeigt den Inhalt des Laufwerks i
dir *.exe	Stern ist eine „Wild Card“. Er steht für eine beliebige Zeichenkombination, es werden somit alle .EXE-Dateien ausgegeben
dir m??er.doc	Fragezeichen ist ebenfalls eine „Wild Card“, jedoch für ein einzelnes Zeichen
whoami	Gibt den Usernamen des ausführenden Benutzers aus
hostname	Gibt den Rechnernamen aus

Befehle für Verzeichnisse

Die folgende Befehle sind zum Anlegen, Wechseln und Löschen neuer Verzeichnisse. Das Verzeichnis, welches im aktuellen Eingabebereitschaftszeichen aufscheint, wird meist als **aktuelles Verzeichnis** oder **Arbeitsverzeichnis** bezeichnet.

Befehl	**Beschreibung**
cd	change directory, Bedeutung von Syntax abhängig
cd \..	geht zum root (oberste Verzeichnisebene)
cd \programme	wechselt in das Verzeichnis Programme ausgehend vom root
cd programme	wechselt - ausgehend vom derzeitigen Arbeitsverzeichnis - in das Unterverzeichnis `programme`
cd .	Zeiger auf aktuelles Verzeichnis
cd ..	Zeiger auf übergeordnetes Verzeichnis
md	make directory - **erzeugt Unterverzeichnis**
rd	remove directory - **löscht Unterverzeichnis**
tree	zeigt die Verzeichnisstruktur auf dem aktuellen Laufwerk an

Befehle für Dateien

Wichtiger Operationen mit Dateien sind das Kopieren, das Löschen und das Umbenennen.

Befehl	**Beschreibung**
copy	benötigt zwei Parameter: 1. Quelldatei, die kopiert werden soll, 2. Zieldatei
copy x.dat x.old	kopiert die Datei x.dat auf x.old (dabei wird x.dat nicht gelöscht!). Existiert bereits eine Datei x.old, so wird diese gelöscht und mit den Daten von x.dat gefüllt.
copy *.* a:	kopiert alle Dateien im Arbeitsverzeichnis nach A:
copy adam.dat a:	kopiert adam.dat **nach** a:adam.dat
copy *.* c:\xy\z	kopiert alle Dateien des aktuellen Ordners auf die Festplatte c: ins Unterverzeichnis Z des Verzeichnisses xy
del x.dat	löscht die Datei `x.dat` im aktuellen Verzeichnis
del *.*	löscht alle Dateien im aktuellen Verzeichnis
ren x.dat y.dat	rename, benennt x.dat in y.dat um
move muster.doc windows	verschiebt `muster.doc` ind das Verzeichnis `windows`
type *Dateiname*	zeigt den Inhalt der angegebenen Datei am Bildschirm an
edit *Dateiname*	ruft einen Editor auf, um den Inhalt einer Datei verändern zu können
print *Dateiname*	gibt den Inhalt einer Datei auf dem Drucker aus
xcopy /E	Kopiert ganze Verzeichnisse
rmdir /S	Löscht Verzeichnisse trotz Inhalt

DOS-Übung

Speichere den Ordner „DOS“ dos.rar in dein Verzeichnis, bevor du mit der Übung beginnst!

1. Erzeuge ein Verzeichnis `uebungen` in deinem homedirectory.
2. Wechsle in dieses Verzeichnis.
3. Erstelle eine Datei `test1.txt` und `test2.txt` mit beliebigen Inhalt.
4. Kopiere die Datei `test1.txt` in die Datei `test21.txt`.
5. Erstelle ein Verzeichnis `aufgabe`.
6. Benenne `test2.txt` um in `test12.txt`.
7. Kopiere die Dateien test1?.txt in das Verzeichnis `aufgabe`. Welche Dateien wurden kopiert?
8. Kopiere dir alle Dateien aus dem Ordner`DOS` in das Verzeichnis `aufgabe`. Wie viele Dateien

befinden sich nun in diesem Verzeichnsi?
9. Lasse dir alle Dateien mit der Endung `*.txt` ausgeben.
10. Lösche alle Dateien mit der Endung `*.dat`.
11. Benenne alle Dateien mit der Endung `*.bak` um in `*.txt`.
12. Lasse dir alle Dateien sortiert nach der Dateigröße anzeigen. Finde den Befehl selber heraus.
13. Gib den Inhalt der Datei `hallo.txt` aus.
14. Lasse dir alle Dateien anzeigen, bei denen im Dateinamen die Zahl 1 vorkommt.
15. Lösche diese Dateien.
16. Verschiebe alle Dateien aus dem Verzeichnis Aufgabe in das Verzeichnis uebungen.
17. Lösche das Verzeichnis Aufgabe.
18. Gib den Befehl `attrib +h *.*` ein.
19. Lasse dir alle Dateien anzeigen. Was ist passiert?
20. Mache dir über den Befehl `attrib` schlau und gib einen Befehl ein, sodass alle Dateien wieder angezeigt werden.

BATCH (.bat) Files

Befehl	**Beschreibung**
echo	Gibt Text aus
@echo off	Schaltet die Autobefehlsanzeige aus
title	Definiert den Titel des BATCH-Files
set VAR=Text	Speichert Text in die Variable VAR
set /P VAR=	Fordert den User auf eine Zeichenfolge einzugeben
set/A C=%A%+%B%	/A gibt an, dass die Zeichenfolge rechts vom = ein numerischer Ausdruck ist, der ausgewertet wird. in C wird das Ergebnis von A+B gespeichert
echo %VAR%	Ausgabe einer Variable → Text
echo %USERNAME%	Gibt den Usernamen des ausführenden Benutzers aus
REM	Definiert einen Kommentar

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_09:05_09_01

Last update: **2019/05/14 15:53**

5.9.2) Benutzerverwaltung

Microsoft Windows ist ein Multi-User Betriebssystem. Was ist ein Benutzer? Was ist eine Gruppe? Welche Rolle spielt die Sicherheitsrichtlinie?

BENUTZER

SID

Jeder Benutzer wird durch eine Nummer, einer SID (Security Identifier) eindeutig gekennzeichnet und erhält ein eigenes Profil unter c:\users\%username%.

Mithilfe von SIDs ist es u.a. möglich Benutzer umzubenennen, ohne Veränderung seiner Zugriffsrechte. Selbst wenn das Administratorkonto umbenannt wird, so hat dieser immer die SID mit der Endung 500. Welcher User welche SID verwendet kann in der CMD oder mit PowerShell eingesehen werden:

Befehl für die PowerShell:

```
PS C:\> get-wmiobject win32_useraccount


AccountType : 512
Caption     : SERVER01\Administrator
Domain      : SERVER01
SID         : S-1-5-21-140281148-68646805-4244902722-500
FullName    :
Name        : Administrator

AccountType : 512
Caption     : SERVER01\Gast
Domain      : SERVER01
SID         : S-1-5-21-140281148-68646805-4244902722-501
FullName    :
Name        : Gast
```

Befehl für die CMD:

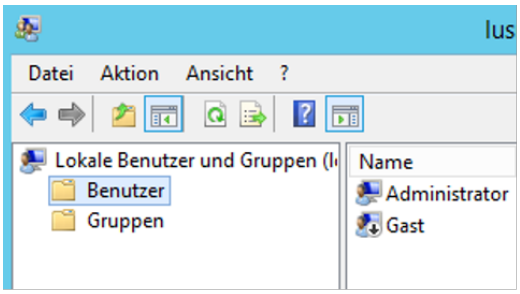
```
wmic useraccount get name,sid
```

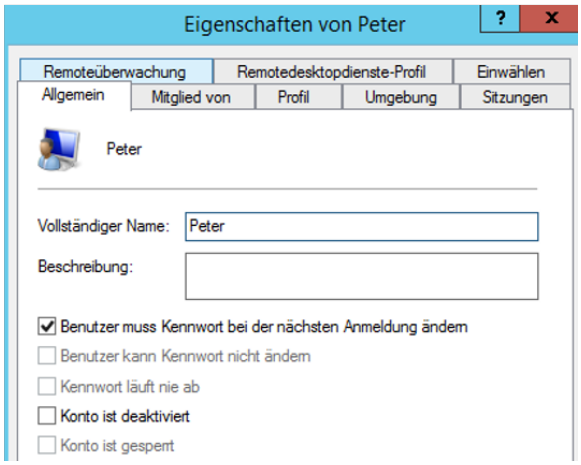
Kennwörter/Passwörter

Kennwörter werden in der SAM (Security Account Manager) Datenbank gespeichert. Diese Datenbank kann im laufenden Betrieb (mit Bordmitteln) nicht eingesehen werden. Die Kennwörter werden als Hashwert gespeichert, d.h. nicht im Klartext.

Anlegen von Lokalen Benutzern

Dies ist grafisch mit lusrmgr.msc oder mithilfe des Befehls net user möglich.

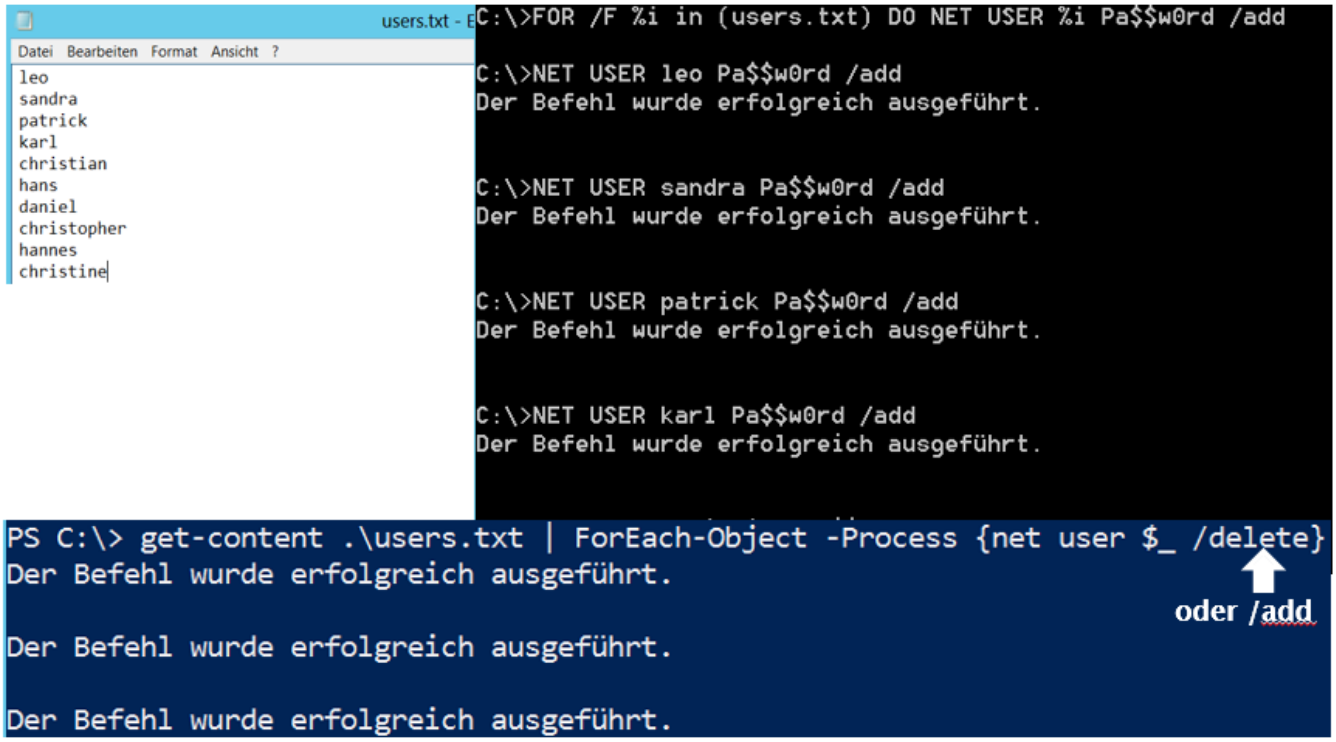





```
net user Max 'Pa$$w0rd' /add
```

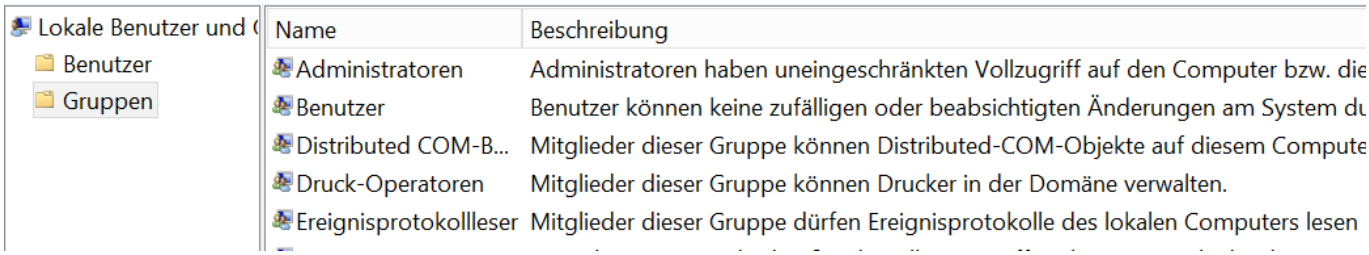
Massenimport von Usern

Hier am Beispiel einer Textdatei users.txt mit einer Auflistung von Vornamen:



GRUPPEN

Bei einer Gruppe handelt es sich um eine Sammlung von Benutzer- und Computerkonten, Kontakten sowie weiteren Gruppen, die als einzelne Einheit verwaltet werden können. Gruppen findet man ebenfalls in lusrmgr.msc bzw. auch mit net localgroup.

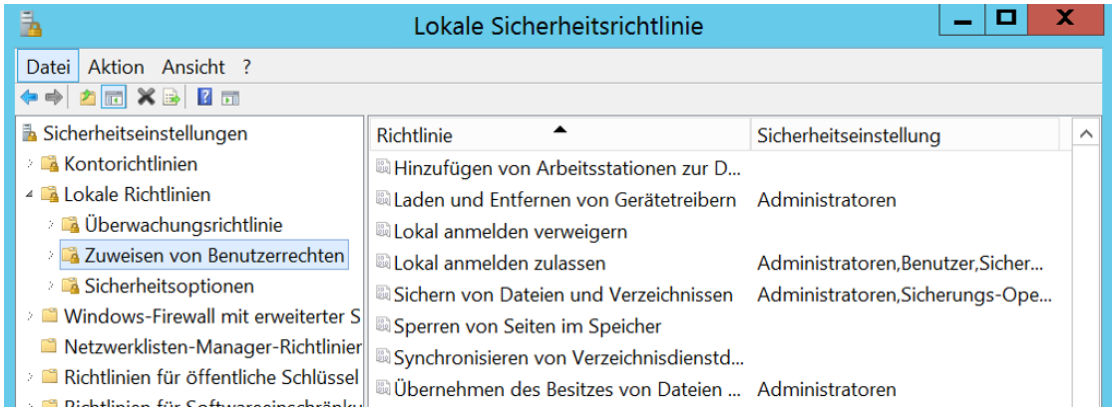


Hier wird der Benutzer admin01 der Gruppe Administratoren hinzugefügt:

```
net localgroup administratoren admin01 /add
```

SICHERHEITSRICHTLINIEN

Wozu das Ganze? Gruppen, Benutzer? Nun, es ist wichtig zu unterscheiden, wer, was, wann, wo am System tun darf oder nicht. Und dies regelt die Sicherheitsrichtlinie, welche unter secpol.msc zu finden ist.



Fügen Sie einen Benutzer in die Gruppe der Administratoren hinzu, so darf dieser laut der Liste Gerätetreiber installieren oder auch Dateien sichern. Würde es keine Gruppen(-richtlinien) geben, so müsste man bei jedem einzelnen Benutzer jedes einzelne Recht hinzufügen oder entfernen. Mithilfe von Gruppen braucht man dies nur einmal für die Gruppe und kann später alle gewünschten Benutzer in diese Gruppe hinzufügen. Somit spart man sehr viel Zeit als IT-Administrator.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_09:05_09_02

Last update: **2019/05/14 15:54**

5.9.3) Dateimanagement

Eine Hauptaufgabe des Betriebssystems ist die Verwaltung der Daten. Dabei werden Daten in sogenannten **Dateien** zusammengefasst, die auf Festplatten und anderen Speichermedien abgespeichert werden können. Eine Datei lässt sich mit dem Inhalt einer Karteikarte vergleichen.

Mehrere Dateien („Karteikarten“) werden in einem Verzeichnis (directory) abgelegt. Ein Verzeichnis ist vergleichbar mit einer Schachtel, in der verschiedene Karteikarten untergebracht werden können. Selbstverständlich ist es auch möglich, in einer Schachtel mehrere kleinere Schachteln unterzubringen (Unterverzeichnisse - subdirectory) usw.

Klarerweise entsteht dadurch ein baumartiges System, weshalb man dieses auch als „Tree“ bezeichnet. Die größte Schachtel, die alle anderen enthält bezeichnet man als **Hauptverzeichnis** oder **Root** (Wurzel).

Dateiformate

Die Art und Weise, wie Informationen innerhalb einer Datei gespeichert werden, und wie diese zu interpretieren sind, wird als Dateiformat bezeichnet. Um den Inhalt einer Datei wieder benutzen oder weiterverwenden zu können, muss bekannt sein, auf welche Weise und in welcher Reihenfolge die Informationen in der Datei abgelegt wurden.

Für unterschiedliche Inhalte und Einsatzzwecke gibt es zahlreiche Methoden zur Speicherung in Dateien, die unter dem Oberbegriff **Dateiformate** zusammengefasst werden. Um also z.B. den Inhalt eines einfachen Briefes in einer Datei abzulegen, wird ein auf Textspeicherung ausgerichtetes Format verwendet, während bei der Speicherung eines Bildes ein entsprechendes Grafikformat verwendet wird.

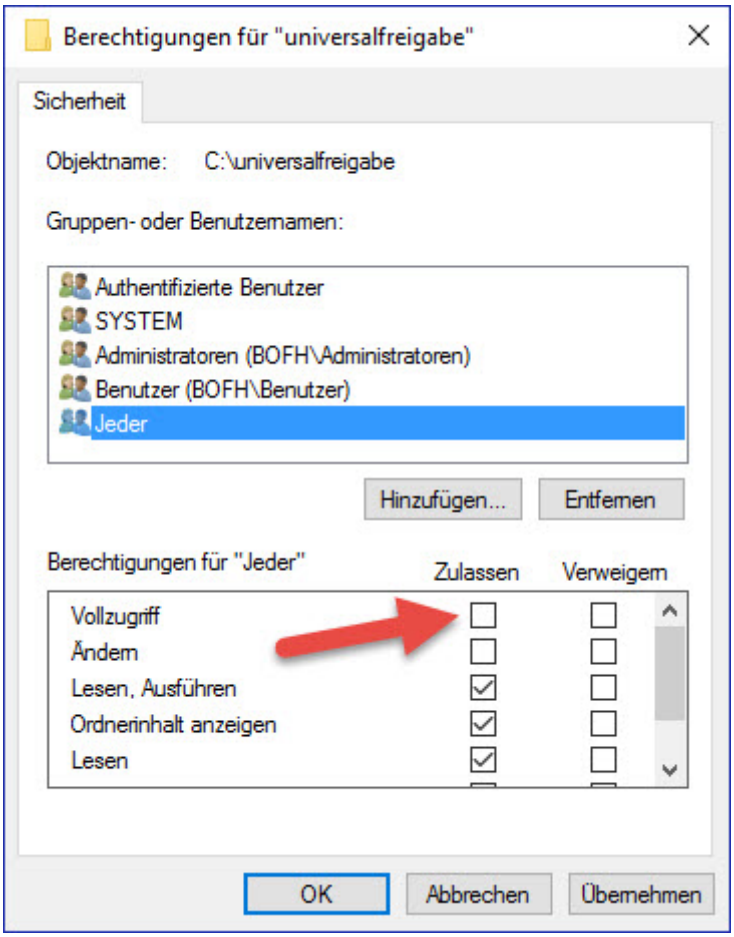
Vor allem bei Microsoft-Betriebssystemen hat es sich eingebürgert, einen Hinweis auf das verwendete Dateiformat durch einen Zusatz am jeweiligen Namen der Datei anzubringen. Diesen Hinweis nennt man **Dateiendung** und wird durch einen **Punkt** vom Dateinamen getrennt.

Übersicht über einige Standarderweiterungen

für ausführbare Dateien	
EXE	ausführbare Datei
BAT	Batch-File
programmspezifische Erweiterungen	
DOC	formatiertes Dokument
TXT	Textdatei ohne Sonderzeichen
SYS	System(Programm)datei
DBF	Datenbankfile
…	…

Dateirechte

Da nicht jeder Benutzer immer alle Rechte auf einem Rechner haben darf/soll/muss, gibt es im Dateisystem (z.B. NTFS) die Rechte für die jeweiligen Benutzer bzw. Gruppen zu definieren. Diese Dateirechte sind die Sicherheitseinstellungen, die man auf einer NTFS-Partition einem Dateiobjekt vergibt. Diese werden üblicherweise im Kontextmenü des Objektes unter Eigenschaften > Sicherheitseinstellungen vergeben:



Es gibt die folgenden Rechte in der Registerkarte Sicherheit bei einem Dateiobjekt

Berechtigung	**Rechte**
Vollzugriff / Full Control	Alle
Ändern / Modify	Alle, außer Berechtigungen ändern und Besitzerrechte übernehmen
Lesen & Ausführen / Read & Execute	Datei öffnen und lesen, ausführbare Dateien und Batchdateien starten
Ordnerinhalt auflisten / List Folder Contents	Nur bei Ordner: Lesen und Lesen & Ausführen
Lesen / Read	Datei öffnen und lesen
Schreiben / Read	Datei ändern oder neu erzeugen
Spezielle Berechtigungen	Siehe unten

Kopieren / Verschieben von Dateien

Wenn eine Datei kopiert wird, wird sie am Zielort neu erstellt und erbt daher die Berechtigungen des Ordners, in den sie kopiert wurde. Beim Verschieben innerhalb einer Partition behält die Datei Ihre Ursprungsrechte.

Will man Dateien verschieben und Berechtigungen des Zielordners annehmen lassen, kopiert man also am Besten und löscht dann die Ursprungsdaten.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_09:05_09_03

Last update: **2019/05/14 15:55**

5.9.4) Sichern und Wiederherstellen in Windows

Ein Windows 10 Backup könnt ihr mit integrierten Backup-Tools im Handumdrehen erstellen, eure persönlichen Daten durch eine Sicherung schützen und ein komplettes Windows-Systemabbild mit Bordmitteln erstellen.

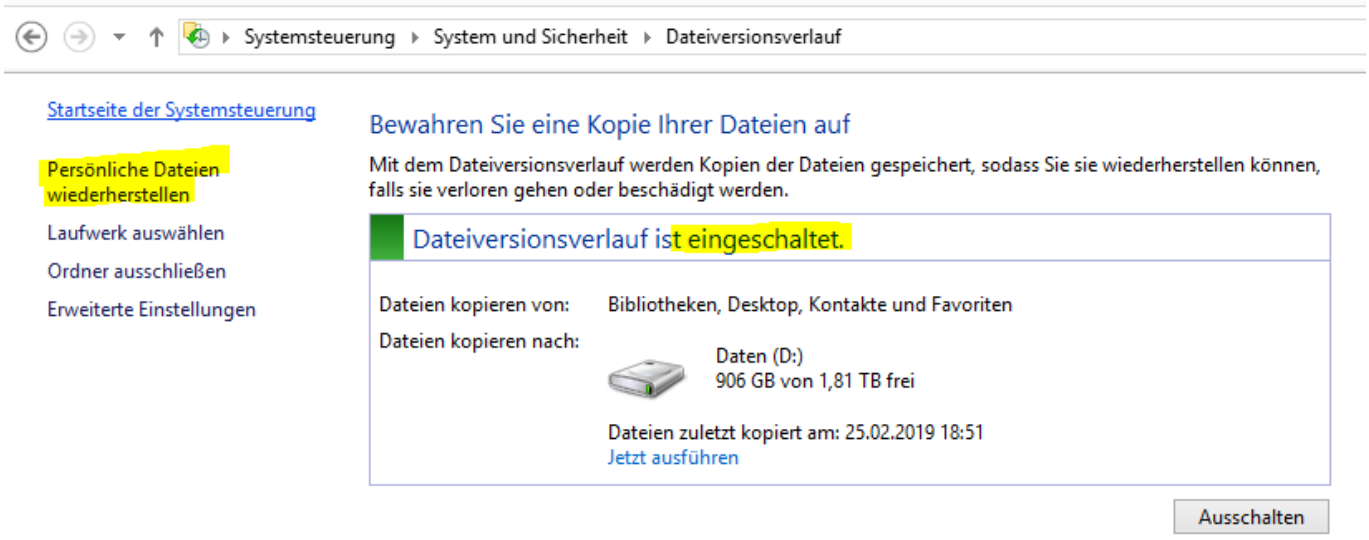
Mit den Windows 10-Bordmitteln könnt ihr bei einem Defekt der Festplatte nicht nur einzelne Daten, sondern euer komplettes Windows-System wiederherstellen. Außerdem könnt ihr mit den Windows-Systemprogrammen eure Daten automatisch sichern.

Hier sind die einzelnen Varianten beschrieben:

1) Dateiversionsverlauf

Mit dem Dateiversionsverlauf werden Kopien der Dateien gespeichert, sodass sie wiederhergestellt werden können, falls sie verloren gehen oder beschädigt werden.

- Systemsteuerung - System und Sicherheit - Dateiversionsverlauf
- Wenn der Dateiversionsverlauf eingeschaltet ist, kann auf ältere Dateibestände zurückgegriffen werden.



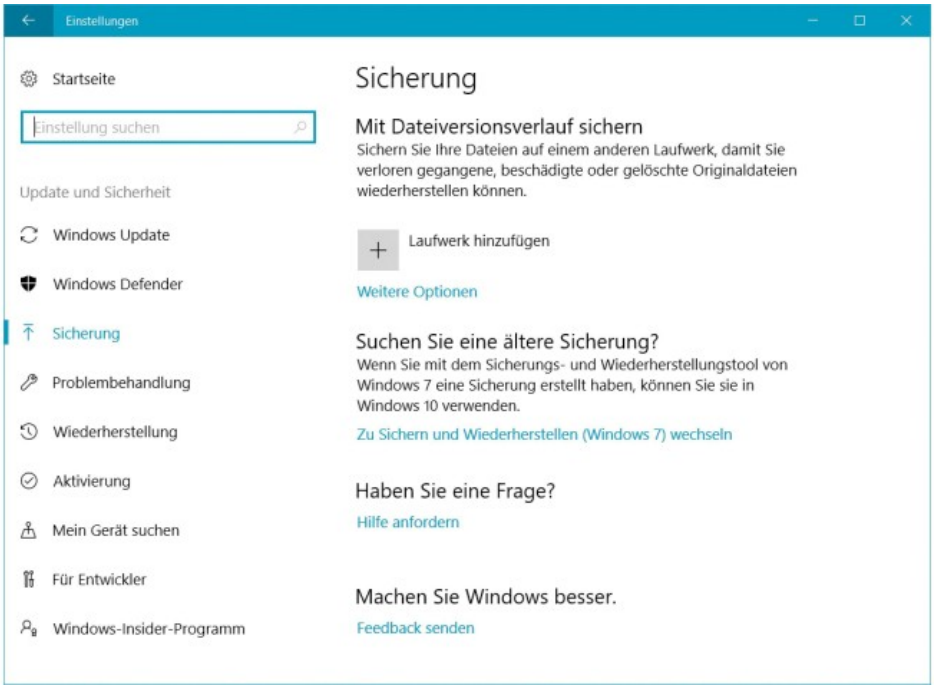
2) Backup erstellen und einzelne Dateien sichern

Mit den folgenden Schritten könnt ihr einzelne Ordner und Dateien, etwa Bilder, Videos und Dokumente, automatisch sichern und einzelne Dateien im Fehlerfall wiederherstellen:

- Öffnet unten links das vierkachelige Windows-Startmenü und klickt auf den Reiter „Einstellungen“.
- Wählt im neuen Fenster das Feld „Update und Sicherheit“ und klickt anschließend auf die

Kategorie „Sicherung“.

- Klickt auf „Laufwerk hinzufügen“ und Windows listet euch alle verfügbaren Datenträger aus. Wählt die Festplatte oder den USB-Stick aus, auf dem eure Daten gesichert werden sollen.
- Klickt danach auf „Weitere Optionen“, wo ihr über „Ordner hinzufügen“ weitere Ordner und Daten dem Backup hinzufügen könnt. Über das Drop-Down-Menü bei „Meine Daten sichern“ könnt ihr außerdem auswählen, ob und wie oft eure Daten automatisch gesichert werden sollen.
- Schießt die Datensicherung mit einem Klick auf „Jetzt sichern“ ab.

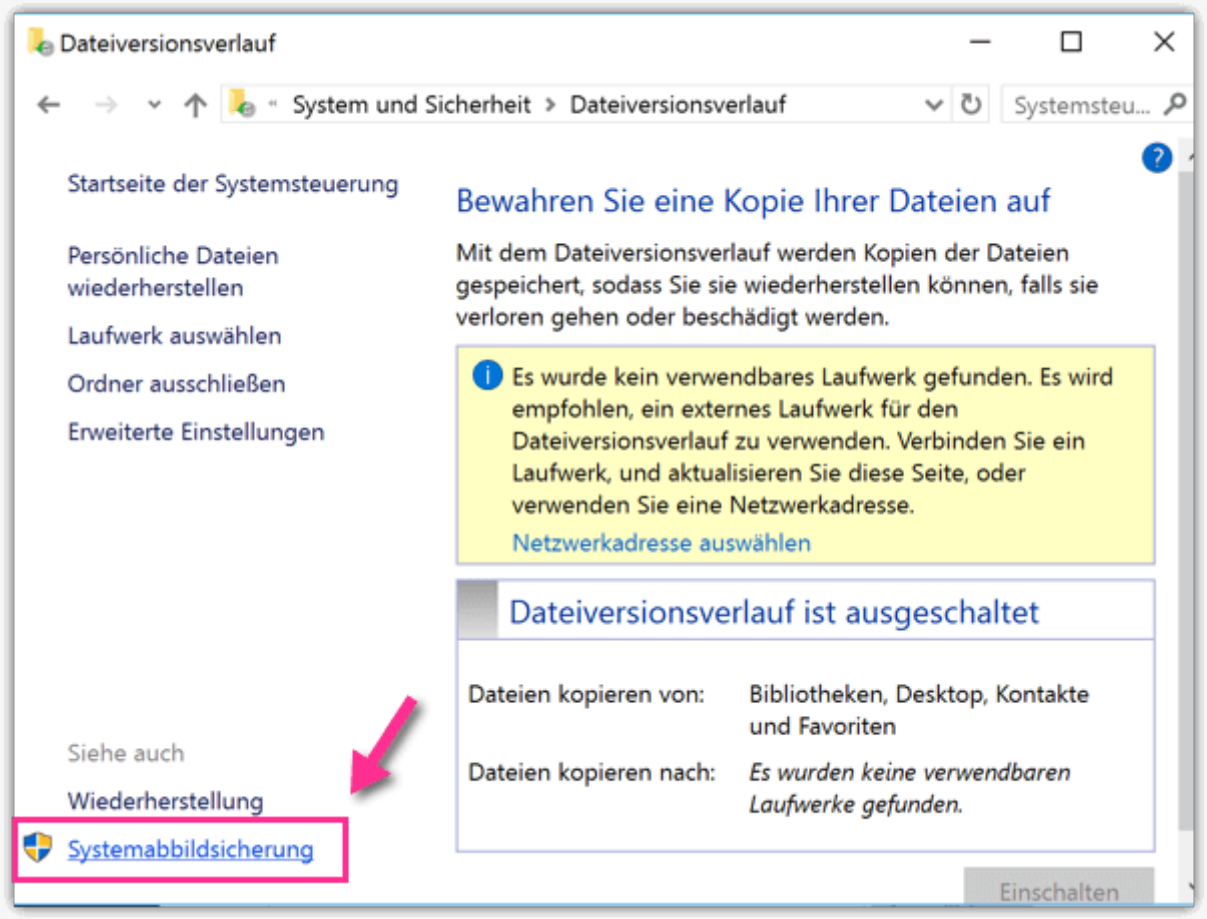


3) Komplettes Systemabbild mit Bordmitteln erstellen

Wollt ihr aber euer komplettes System absichern und ein Windows 10-Systemabbild erstellen, so sind folgende Schritte notwendig. Damit kann man nicht nur einzelne Dateien sondern auch das komplette Betriebssystem im Fehlerfall wiederherstellen.

Achtung!!: Speichert das Backup nie auf das Systemlaufwerk, denn im Falle eines Festplattendefekts hilft das beste Backup nichts!! Als Speicherort eignet sich am besten eine externe Festplatte oder ein Serverlaufwerk mit genügende Speicherplatz.

- Navigiert, wie oben beschrieben, erneut in das „Weitere Optionen“-Menü.
- Scrollt in den Sicherungsoptionen bis an das Ende der Seite und klickt auf „Siehe erweiterte Einstellungen“.
- Wählt im neu geöffneten Fenster unten links „Systemabbildsicherung“ und anschließend „Systemabbild erstellen“ aus.
- Wählt die Festplatte aus, auf dem das Systemabbild erstellt werden soll und bestätigt mit „Weiter“.
- Windows zeigt euch an, welche Laufwerke mit dem Backup gesichert werden. Schließt den Vorgang mit „Sicherung starten“ ab



4) Alternative Backup-Methoden

Neben den Windows-Bordmitteln könnt ihr für eure Datensicherung auch auf verschiedene Software-Lösungen zurückgreifen.

- Acronis
- Duplicati

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_09:05_09_04

Last update: **2019/05/14 15:55**

5.10) LINUX

Linux ist ein freies Multiplattform-Mehrbenutzer-Betriebssystem, das den Linux-Kernel enthält. Im praktischen Einsatz werden meist sogenannte Linux-Distributionen genutzt, in denen der Linux-Kernel und verschiedene Software zu einem fertigen Paket zusammengestellt sind.

Das wichtigste gleich vorweg, in Linux ist alles eine Datei –> Everything is a file.

Everything is a file beschreibt eine der definierenden Eigenschaften von Unix und seinen Abkömmlingen, demnach Ein-/Ausgabe-Ressourcen wie Dateien, Verzeichnisse, Geräte (z. B. Festplatten, Tastaturen, Drucker) und sogar Interprozess- und Netzwerk-Verbindungen als einfache Byteströme via Dateisystem verfügbar sind

- 5.10.1) Aufbau des Betriebssystems
- 5.10.2) Benutzer
- 5.10.3) Dateimanagement
- 5.10.4) Dateirechte
- 5.10.5) Inodes
- 5.10.6) Distributionen und Desktops
- 5.10.7) Konsole/Bash/Terminal
- 5.10.8) Shell Scripts

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10

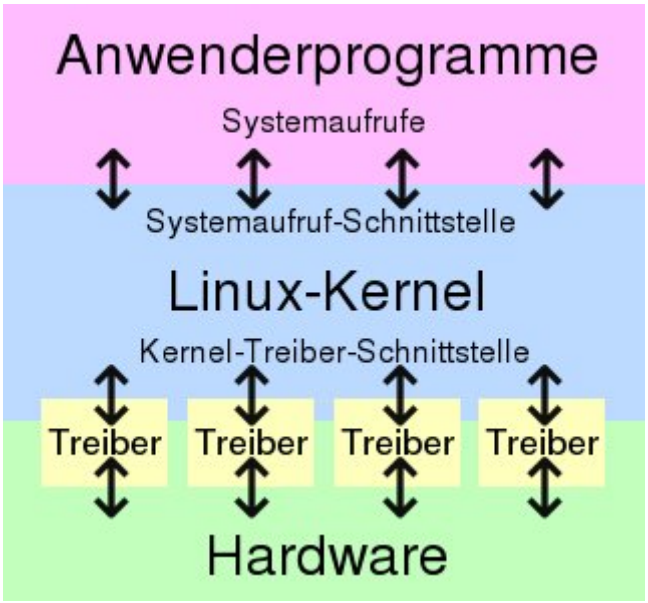
Last update: **2019/05/14 15:59**

5.10.1) Aufbau des Betriebssystems

Das zentrale Kernstück des Betriebssystems, der Linux-Kernel (meist nur Kernel genannt) bildet eine Trennschicht zwischen Hardware und Anwenderprogrammen. Das heißt, wenn ein Programm auf ein Stück Hardware zugreifen will, so kann es niemals direkt darauf zugreifen, sondern nur über das Betriebssystem.

Dazu bedient sich das Programm der **Systemaufrufe**. Über den Systemaufruf teilt das Anwenderprogramm dem Betriebssystem mit, dass es etwas zu tun gibt. Will etwa ein Programm eine Zeile Text auf dem Bildschirm ausgeben, so wird ein Systemaufruf gestartet, dem der Text übergeben wird. Das Betriebssystem erst schreibt ihn auf den Bildschirm.

Auf der anderen Seite muss das Betriebssystem die Möglichkeit haben, mit den einzelnen Hardware-Komponenten zu sprechen. Mittels seiner **Treiberschnittstelle** spricht es spezielle Geräte-Treiber an. Erst die Treiber kommunizieren dann direkt mit den Geräten.



Zu den **Anwenderprogrammen** zählen alle von uns gestarteten Programme (Videoplayer, Webbrowser ...), wie auch die grafische Oberfäche des Betriebssystems, das Desktop-Environment. Letzteres ist nicht ein Programm, sondern eine Sammlung von Programmen, die zusammen die gewohnten Funktionalitäten beisteuern.

Ein ganz spezielles Anwenderprogramm ist die Shell - die „Benutzeroberfläche“. Es existieren viele verschiedene Shells - wir werden hier mit der Bash (Bourne again shell) arbeiten. Diese ist die Standardshell auf Linuxsystemen. Alle Shells stellen dem Benutzer eine Kommandozeile zur Verfügung, mit der Befehle eingegeben werden können, die direkt als Systemaufrufe an das Betriebssystem weitergeleitet werden.

Linux ist ein **Multitasking-Betriebssystem**: das heißt, es können mehrere Prozesse - so nennt man Programme, sobald sie in den Speicher geladen sind und laufen - gleichzeitig laufen. Das bedingt, dass das System die verfügbare Rechenzeit des Prozessors in kleine Zeitscheiben aufteilt (im Millisekundenbereich), die dann den jeweiligen Prozessen zur Verfügung stehen. Diese Aufgabe übernimmt eine übergeordnete Instanz - der Scheduler. Dieser verwaltet die Zuteilung der Zeitscheiben an die verschiedenen Prozesse.

Daher kann kein Prozess die ganze Rechenleistung für sich beanspruchen und auch ein „hängender“ Prozess kann nicht das ganze System lahmlegen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_01

Last update: **2019/05/14 15:58**

5.10.2) Benutzer

Linux ist auch ein **Multiuser**-System, das heißt, es können mehrere Benutzer an verschieden Terminals auf dem selben Rechner arbeiten. Dazu ist es natürlich notwendig, dass jeder Benutzer eindeutig identifiziert ist. Die User (engl., Benutzer) werden zwar mit ihren Namen verwaltet, intern arbeitet ein Unix-System aber mit Usernummern. Jeder Benutzer hat also eine Nummer welche UserID oder kurz UID genannt wird. Jeder Benutzer ist auch Mitglied mindestens einer Gruppe. Es kann beliebig viele Gruppen in einem System geben und auch sie haben intern Nummern (GroupID oder GID). Im Prinzip sind Gruppen nur eine Möglichkeit, noch detailiertere Einstellungsmöglichkeiten zu haben, wer was darf.

Eine spezielle Rolle hat der Benutzer mit der UserID 0 - er ist Root (engl., Wurzel). Root steht außerhalb aller Sicherheitseinrichtungen des Systems - kurz - er darf alles. Er kann mit einem Befehl das ganze System zerstören, er kann die Arbeit von Wochen und Monaten löschen usw. Aus diesem Grund meldet sich auch der Systemverwalter im Normalfall als normaler Benutzer an - zum Root-Benutzer wird er nur dann, wenn er Systemverwaltungsarbeiten abwickelt, die diese Identität benötigen.

Benutzertypen

root

Der Benutzer root ist mit allen Rechten ausgestattet, die ihm die Administration (bei Unachtsamkeit natürlich auch die Beschädigung!) des Systems erlauben. Diesem auch als Superuser bezeichneten Benutzer ist immer die UID 0 zugeordnet.

Systembenutzer

Je nach System kann eine Vielzahl von Prozessen und Diensten erwünscht sein, die bereits beim Hochfahren des Systems verfügbar sein sollen. Nicht jeder dieser Prozesse benötigt jedoch die volle Rechteausstattung des Superusers. Man möchte natürlich so wenige Prozesse wie nur möglich unter einer root Kennung starten, da die weitreichenden Rechte solcher Prozesse unnötige Möglichkeiten für Missbrauch und Beschädigung des Systems liefern.

Ein Systembenutzerkonto ist in diesem Sinne ein Benutzerkonto, das jedoch (nahezu) ausschließlich zur Ausführung von Programmen unter einer speziellen Benutzerkennung verwendet wird. Kein menschlicher Benutzer meldet sich normalerweise unter einem solchen Konto an.

Standardbenutzerkonto

Dies ist das normale Benutzerkonto, unter welchem jeder üblicherweise arbeiten sollte.

Die zentralen Benutzerdateien

Die Dateien zur Benutzerverwaltung finden Sie unter Linux im Verzeichnis /etc. Es handelt sich dabei um die Dateien **/etc/passwd**, **/etc/shadow** und **/etc/group**.

/etc/passwd

Die Datei /etc/passwd ist die zentrale Benutzerdatenbank.

Mit cat /etc/passwd können Sie einen Blick in diese zentrale Benutzerdatei werfen. Hier werden alle Benutzer des Systems aufgelistet. Zu beachten ist, dass alle Benutzertypen eingetragen sind, also sowohl der Superuser root als auch die Standard- und Systembenutzer.

Ein Benutzerkonto in der Datei /etc/passwd hat generell folgende Syntax:

```
Benutzername : Passwort : UID : GID : Info : Heimatverzeichnis : Shell
```

Spalte	**Erklärung**
Benutzername	Dies ist der Benutzername in druckbare Zeichen, meistens in Kleinbuchstaben.
Passwort	Hier steht verschlüsselt das Passwort des Benutzers (bei alten Systemen). Meist finden Sie dort ein x. Dies bedeutet, dass das Passwort verschlüsselt in der Datei /etc/shadow steht. Es ist auch möglich, den Eintrag leer zu lassen. Dann erfolgt die Anmeldung ohne Passwortabfrage (in der Datei /etc/shadow muss dann an Stelle des verschlüsselten Passwortes ein * stehen).
UID	Die Benutzer-ID des Benutzers. Die Zahl hier sollte größer als 100 sein, weil die Zahlen unter 100 für Systembenutzer vorgesehen sind. Weiterhin muss die Zahl aus technischen Gründen kleiner als 64000 sein.
GID	Die Gruppen-ID des Benutzers. Auch hier muss die Zahl wie bei der UID kleiner als 64000 sein.
Info	Hier kann weitere Information vermerkt werden, wie z.B. der vollständige Name des Benutzers und persönliche Angaben (Telefonnummer, Abteilung, Gruppenzugehörigkeit u.ä.).
Heimatverzeichnis	Das Heimatverzeichnis des Benutzers bzw. das Startverzeichnis nach dem Login.
Shell	Die Shell, die nach der Anmeldung gestartet werden soll. Bleibt dieses Feld frei, dann wird die Standardshell /bin/sh gestartet.

Hier ein Beispiel für einen Systembenutzer:

```
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
```

Der Benutzer heißt uucp, das Passwort ist in der Datei /etc/shadow gespeichert (x), die UID ist 10, die GID 14, als erläuternde Bezeichnung trägt der Benutzer den Namen „Unix-to-Unix CoPy system“, das Startverzeichnis nach der Anmeldung ist /etc/uucp, und die vorgeschlagene Shell ist die bash.

An dieser Stelle sei nochmals darauf hingewiesen, dass die meisten Linux-Distributionen komfortable Werkzeuge zur Benutzerverwaltung mitliefern und es auch eine Reihe von Befehlen gibt, die für die Benutzerverwaltung verwendet werden können

/etc/shadow

Bei früheren Versionen von Linux speicherte man die die Passwörter direkt in die passwd-Datei. Allerdings war dies durch einen sogenannten Wörterbuchangriff und der beispielsweise mit Hilfe des Programmes crypt möglich, diese Passwörter in vielen Fällen zu entschlüsseln und auszulesen. Deshalb hat man die Datei /etc/shadow eingeführt, in der die Angaben über die Passwörter durch ein spezielles System besser geschützt werden.

Der Eintrag in diese Datei erfolgt nach einem ähnlichen Schema, wie in der Datei /etc/passwd:

```
Benutzername : Passwort : DOC : MinD : MaxD : Warn : Exp : Dis : Res
```

Benutzername	**Dies ist der Benutzername in druckbaren Zeichen, meistens in Kleinbuchstaben.**
Passwort	Hier steht verschlüsselt das Passwort des Benutzers. Wenn hier ein * oder ! steht, dann bedeutet dies, dass kein Passwort vorhanden bzw. eingetragen ist.
DOC	Day of last change: der Tag, an dem das Passwort zuletzt geändert wurde. Besonderheit hier: Der Tag wird als Integer-Zahl in Tagen seit dem 1.1.1970 angegeben.
MinD	Minimale Anzahl der Tage, die das Passwort gültig ist.
MaxD	Maximale Anzahl der Tage, die das Passwort gültig ist.
Warn	Die Anzahl der Tage vor Ablauf der Lebensdauer, ab der vor dem Verfall des Passwortes zu warnen ist.
Exp	Hier wird festgelegt, wieviele Tage das Passwort trotz Ablauf der MaxD noch gültig ist.
Dis	Bis zu diesem Tag (auch hier wird ab dem 1.1.1970 gezählt) ist das Benutzerkonto gesperrt
Res	Reserve, dieses Feld hat momentan keine Bedeutung.

Es folgt wieder ein Beispiel:

```
selflinux:/heSIGnYDr6MI:11995:1:99999:14:::
```

Der Benutzer heißt selflinux, das Passwort lautet verschlüsselt „/heSIGnYDr6MI“. Es wurde zuletzt geändert, als 11995 Tage seit dem 1.1.1970 vergangen waren. Das Passwort ist minimal einen Tag gültig, maximal 99999 Tage (was man als immer deuten kann - 99999 Tage sind ca. 274 Jahre). Es soll ab 14 Tage vor Ablauf des Passwortes gewarnt werden. Die anderen Werte sind vom Administrator nicht definiert und bleiben daher leer.

/etc/group

In dieser Datei finden Sie die Benutzergruppen und ihre Mitglieder. In der Datei /etc/passwd wird mit der GID eigentlich schon eine Standardgruppe für den Benutzer festgelegt. In der /etc/group können Sie weitere Gruppenzugehörigkeiten definieren. Das hat in der Praxis vor allem in Netzwerken eine große Bedeutung, weil Sie so in der Lage sind, z.B. Gruppen für Projekte oder Verwaltungseinheiten zu bilden. Für diese Gruppen kann man dann entsprechend die Zugriffsrechte einstellen. Dies hat dann wiederum den Vorteil, dass man die Daten gegen eine unbefugte Benutzung absichern kann.

Der Eintrag einer Gruppe in die Datei sieht so aus:

```
Gruppenname : Passwort : GID : Benutzernamen (Mitgliederliste)
```

Gruppenname	**Der Name der Gruppe in druckbare Zeichen, auch hier meistens Kleinbuchstaben.**
Passwort	Die Besonderheit hier ist folgende: Wenn das Passwort eingerichtet ist, können auch Nichtmitglieder der Gruppe Zugang zu den Daten der Gruppe erhalten, wenn ihnen das Passwort bekannt ist. Ein x sagt hier aus, das das Passwort in /etc/gshadow abgelegt ist. Der Eintrag kann auch entfallen, dann ist die Gruppe nicht durch ein Passwort geschützt. In diesem Fall kann jedoch auch kein Benutzer in die Gruppe wechseln, der nicht in diese Gruppe eingetragen ist.
GID	Gruppen-ID der Gruppe
Benutzernamen	hier werden die Mitglieder der Gruppe eingetragen. Diese sind durch ein einfaches Komma getrennt.

Für einen korrekten Eintrag in die /etc/group reicht eigentlich der Gruppenname und die GID aus. Damit ist die Gruppe dem System bekannt gemacht. Die Felder für das Passwort und die Benutzernamen können frei bleiben.

Soll der Benutzer nur in seiner Standardgruppe bleiben, ist kein Eintrag in die /etc/group notwendig. Hier reicht der Eintrag in die /etc/passwd völlig aus, weil dort die Standardgruppe schon mit angegeben wird. Nur wenn der Benutzer in weiteren bzw. mehreren Gruppen Mitglied sein soll, muss dies in die /etc/group-Datei eingetragen werden. Für Passwörter gilt das oben in der Tabelle Gesagte.

Hier sehen Sie ein Beispiel für einen Eintrag:

```
dialout:x:16:root,tatiana,steuer,selflinux
```

Sie sehen eine Gruppe mit der GID „16“ und den Namen dialout. (Zur Information: dialout erlaubt es normalen Benutzern einen ppp-Verbindungsaufbau zu starten, normalerweise hat nur root dieses Recht). Das x bedeutet hier, dass das Passwort in /etc/shadow abgelegt ist. Da in /etc/gshadow hier bei Passwort ein * steht, ist also kein Passwort für die Gruppe vorhanden (Das bedeutet wiederum, das nur die eingetragenen Mitglieder Zugang zu dieser Gruppe haben). Mitglieder der Gruppe sind: root, tatiana, steuer, selflinux.

Benutzerklassen: user, group und others

Aus der Sicht des Systems existieren drei Benutzerklassen, wenn entschieden werden soll, ob die Berechtigung für einen Dateizugriff existiert oder nicht. Soll beispielsweise eine Datei gelöscht werden, so muss das System ermitteln, ob der Benutzer, welcher die Datei löschen möchte, das erforderliche Recht besitzt:

user@linux ~$ rm testdatei rm: Entfernen (unlink) von „testdatei“ nicht möglich: Keine Berechtigung

In diesem Fall wurde dem rm Kommando der beabsichtigte löschende Zugriff auf die Datei verwehrt - der ausführende Benutzer hatte nicht das Recht, die Datei zu löschen. Um diese Entscheidung zu treffen, verwendet das System das Konzept der Benutzerklassen. Drei Benutzerklassen werden unterschieden: user, group und others. Jede dieser Benutzerklassen ist wiederum in ein Lese-, Schreib- und Ausführrecht unterteilt. Diese werden im Folgenden als Berechtigungsklassen bezeichnet. Somit ergibt sich folgende Körnung für die einfachen Zugriffsrechte einer Datei:

Recht	Beschreibung
user-read	Leserecht für Dateieigentümer
user-write	Schreibrecht für Dateieigentümer
user-execute	Ausführrecht für Dateieigentümer
group-read	Leserecht für Gruppe des Dateieigentümers
group-write	Schreibrecht für Gruppe des Dateieigentümers
group-execute	Ausführrecht für Gruppe des Dateieigentümers
other-read	Leserecht für alle anderen Benutzer
other-write	Schreibrecht für alle anderen Benutzer
other-execute	Ausführrecht für alle anderen Benutzer

Benutzerklassen sind also eng mit der Eigentümerschaft von Dateien verbunden. Jede Datei und jedes Verzeichnis ist sowohl einem Benutzer (einer UID) als auch einer Gruppe (einer GID) zugeordnet. UID und GID gehören zur elementaren Verwaltungsinformation von Dateien und Verzeichnissen und werden in der sogenannten Inode gespeichert.

Beim Zugriff auf eine Datei werden nun UID und GID des zugreifenden Prozesses mit UID und GID der Datei verglichen. Ist other-read gesetzt, darf jeder Benutzer lesend zugreifen und ein weiterer Vergleich erübrigt sich. Ist lediglich group-read gesetzt, muss der Zugreifende mindestens der Gruppe des Dateieigentümers angehören, d.h. eine identische GID aufweisen. Ist ausschließlich user-read gesetzt, so darf nur der Eigentümer selbst die Datei lesen. root ist von dieser Einschränkung freilich ausgenommen. („Ich bin root, ich darf das!“). Von sehr speziellen Ausnahmen abgesehen, die sich außerhalb der hier besprochenen Rechteklassen bewegen, ist root in seinen Aktionen in keinerlei Weise eingeschränkt.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_02

Last update: **2019/05/14 15:58**

5.10.3) Dateimanagement

Nachdem unter Linux das Prinzip **Everything is a file** gilt, werden hier die Besonderheiten von Dateien beschrieben.

Dateien

Datei- und Verzeichnisnamen können bis zu 255 Zeichen lang sein. Dabei wird in jedem Fall zwischen Groß- und Kleinschreibung unterschieden. Die Dateinamen

- DATEI
- datei
- Datei

bezeichnen drei unterschiedliche Dateien. Ein Dateiname darf beliebig viele Punkte enthalten, also zum Beispiel auch Datei.Teil.1.txt. Ein Punkt gilt als normales Zeichen in einem Dateinamen. Dateien, die mit Punkt beginnen, gelten als versteckt und werden normalerweise nicht angezeigt - zum Beispiel .datei. Das Zeichen zum Trennen von Verzeichnis- und Dateinamen ist der Slash („/“) statt dem Backslash („\“) bei Windows.

Es gibt verschiedene Dateiarten: (in Klammer die offizielle Darstellung, wie sie symbolisiert werden)

- Normale Dateien (-)
- Verzeichnisse (d)
- Symbolische Links (l)
- Blockorientierte Geräte (b)
- Zeichenorientierte Geräte (c)
- Named Pipes (p)

Wir sehen hier schon, dass auch Verszeichnisse bloß eine bestimmte Dateiart sind. Eine spezielle nämlich, in der andere Dateien aufgelistet sind. Mit einem Dateibrowser (von Windows kennen wir „Explorer“, bei Apple den „Finder“) sehen wir uns immer nur genau diese Verzeichnisse an, sofern wir nicht mittels verschiedener Plugins die Dateien selbst auswerten und Textdokumente, Bilder anzeigen oder Videos und Musik wiedergeben.

In einem Unix-Dateisystem hat jede einzelne Datei jeweils einen Eigentümer und eine Gruppenzugehörigkeit. Neben diesen beiden Angaben besitzt jede Datei noch einen Satz Attribute, die bestimmen, wer die Datei wie benutzen darf. Diese Attribute werden dargestellt als „rwx“. Dabei steht r für lesen (read), w für schreiben (write) und x für ausführen (execute).

Das Dateisystem

Das Dateisystem ist die Ablageorganisation auf einem Datenträger eines Computers. Um die Funktionsweise zu verstehen, betrachten wir einen Datenträger, die Festplatte, näher:

Die Festplatte besteht aus mehreren Scheiben mit einer magnetisierbaren Oberfläche, auf die die

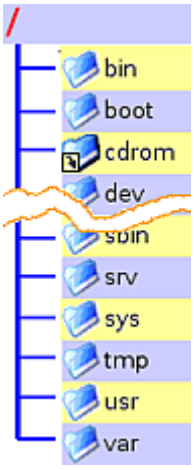
Schreibköpfe unsere Daten als Einsen (ein) und Nullen (aus) abspeichern. Um diese aber vernünftig adressieren zu können, benutzen wir Dateisysteme. Ein solches teilt die Festplatte (eigentlich die „Partition“, denn die Festplatte wird häufig in mehrere Partitionen aufgeteilt, die dann unabhängig formatiert werden können) in kleine Einheiten, die „Blöcke“, welche aus Performancegründen häufig noch zu „Clustern“ zusammengefasst werden.

Der Block (oder Cluster) ist dann die kleinste Einheit, in die eine Datei geschrieben wird, jede Datei benötigt dadurch immer diesen Speicherplatz (oder ein vielfaches) auf der Festplatte.

Von Windows kennen wir NTFS und FAT32, bzw. Apple-Benutzer werden schon von HFS+ gehört haben. Unter Linux werden meist ext2, ext3 oder ext4 (Second, Third bzw. Fourth Extented File System) verwendet. „ext3“ unterscheidet sich von „ext2“ nur dadurch, dass zusätzlich ein „Journal“ geschrieben wird, welches bei Systemabstürzen eine zuverlässige Wiederherstellung möglich macht. „ext4“ ist eine performantere Weiterentwicklung von „ext3“ und heute Standard. Daneben gibt es gelegentlich noch ReiserFS, XFS oder JFS, aber die Wahl des Dateisystems bestimmt tatsächlich immer das Abwägen zwischen höherer Sicherheit und schnellerer Schreibgeschwindigkeit - mit oder ohne Journal.

Verzeichnisstruktur

Das Dateisystem beginnt mit einem Wurzelverzeichnis (auch Rootverzeichnis genannt - /). Es enthält im Regelfall keine Dateien, sondern nur die folgenden Verzeichnisse (Ubuntu):



/bin

Von: binaries (Programme); muss bei Systemstart vorhanden sein; enthält für Linux unverzichtbare Programme; diese Programme können im Gegensatz zu /sbin von allen Benutzern ausgeführt werden; /bin darf keine Unterverzeichnisse enthalten.

/boot

Muss bei Systemstart vorhanden sein; Enthält zum Booten benötigte Dateien.

/dev

Von devices (Geräte); muss bei Systemstart vorhanden sein; enthält alle Gerätedateien, über die die Hardware im Betrieb angesprochen wird

/etc

Von: et cetera („alles übrige“), später auch: editable text configuration (änderbare Text Konfiguration); muss bei Systemstart vorhanden sein; enthält Konfigurations- und Informationsdateien des Basissystems.

- /etc/init.d: dort liegen alle Start- und Stopskripte
- /etc/opt: Verzeichnisse und Konfigurationsdateien für Programme in /opt
- /etc/network: Verzeichnisse und Konfigurationsdateien des Netzwerkes
- ….

/home

Von: home-directory (Heimatverzeichnis); enthält pro Benutzer ein Unterverzeichnis; jedes Verzeichnis wird nach dem Anmeldenamen benannt

/lib

Von: libraries (Bibliotheken); muss bei Systemstart vorhanden sein; enthält unverzichtbare Bibliotheken fürs Booten und die dynamisch gelinkten Programme des Basissystems;

/lost+found

(verloren und gefunden); Dateien und Dateifragmente, die beim Versuch, ein defektes Dateisystem zu reparieren, übrig geblieben sind.

/media

Für (Speicher-)Medien. Enthält Unterverzeichnisse, welche als mount- oder Einhängepunkte für transportable Medien wie z.B. externe Festplatten, USB-Sticks, CD-ROMs, DVDs und andere Datenträger dienen. Ubuntu legt hier auch die Einhängepunkte für Partitionen an. Unterverzeichnisse sind u.a.:

- /media/floppy: Einhängepunkt für Disketten
- /media/cdrom0: Einhängepunkt für CD-ROMs

/mnt

Von: mount (eingehängt); normalerweise leer; kann für temporär eingehängte Partitionen verwendet werden. Für Datenträger, die hier eingehängt werden, wird im Gegensatz zu /media kein Link auf dem Desktop angelegt

/opt

Von: optional; ist für die manuelle Installation von Programmen gedacht, die ihre eigenen Bibliotheken mitbringen und nicht zur Distribution gehören;

/proc

Von: processes (laufende Programme); muss bei Systemstart vorhanden sein; enthält Schnittstellen zum aktuell geladenen Kernel und seinen Prozeduren; Dateien lassen sich mittels cat auslesen; Beispiele: version (Kernelversion), swaps (Swapspeicherinformationen), cpuinfo, interrupts, usw.;

/root

Ist das Homeverzeichnis des Superusers (root). Der Grund, wieso sich das /root-Verzeichnis im Wurzelverzeichnis und nicht im Verzeichnis /home befindet, ist, dass das Homeverzeichnis von Root immer erreichbar sein muss, selbst wenn die Home-Partition aus irgendeinem Grund (Rettungs-Modus, Wartungsarbeiten) mal nicht eingehängt ist.

/sbin

Von: system binaries (Systemprogramme); muss bei Systemstart vorhanden sein; enthält alle Programme für essentielle Aufgaben der Systemverwaltung; Programme können nur vom Systemadministrator (root) oder mit Superuserrechten ausgeführt werden

/srv

Von: services (Dienste); Verzeichnisstruktur noch nicht genau spezifiziert; soll Daten der Dienste enthalten; unter Ubuntu in der Regel leer

/sys

Von: system; im FHS noch nicht spezifiziert; erst ab Kernel 2.6. im Verzeichnisbaum enthalten; besteht ebenso wie /proc hauptsächlich aus Kernelschnittstellen

/tmp

Von: temporary (temporär); enthält temporäre Dateien von Programmen; Verzeichnis soll laut FHS beim Booten geleert werden.

/usr

Von: user (siehe: Herkunft); enthält die meisten Systemtools, Bibliotheken und installierten Programme; der Name ist historisch bedingt - früher, als es /home noch nicht gab, befanden sich hier auch die Benutzerverzeichnisse;

- /usr/bin : Anwenderprogramme; Hier liegen die Desktopumgebungen und die dazu gehörigen Programme, aber auch im Nachhinein über die Paketverwaltung installierte Programme, wie Audacity

/var

Von variable (variabel); enthält nur Verzeichnisse; Dateien in den Verzeichnissen werden von den Programmen je nach Bedarf geändert (im Gegensatz zu /etc); Beispiele: Log-Dateien, Spielstände, Druckerwarteschlange

- /var/log : Alle Log-Dateien der Systemprogramme; Beispiele: Xorg.0.log (Log-Datei des XServer), kern.log (Logdatei des Kernels), dmesg (letzte Kernelmeldungen), messages (Systemmeldungen); Siehe auch Logdateien

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_03

Last update: **2019/05/14 15:59**

5.10.4) Dateirechte

UNIX-Systeme wie Linux verwalten ihre Dateien in einem virtuellen Dateisystem (VFS, Virtual File System). Dieses ordnet jeder Datei über eindeutig identifizierbare Inodes unter anderem folgende Eigenschaften zu:

- Dateityp (einfache Datei, Verzeichnis, Link, …)
- Zugriffsrechte (Eigentümer-, Gruppen- und sonstige Rechte)
- Größe
- Zeitstempel
- Verweis auf Dateiinhalt

Jedes unter Linux gängige UNIX-Dateisystem (z.B. ext2/3/4, ReiserFS, xfs usw.) unterstützt diese Rechte. Gar nicht umgesetzt werden die Rechte hingegen auf VFAT-Dateisystemen; dort können Dateirechte lediglich beim Einhängen simuliert werden. Partitionen mit dem Windows-Dateisystem NTFS werden zwar in Linux standardmäßig ähnlich wie VFAT-Partitionen behandelt; mit den Mount-Optionen permissions und acl lässt sich aber auch auf NTFS-Partitionen eine echte Rechteverwaltung wie bei UNIX-Dateisystemen einrichten. Siehe hierzu Windows-Partitionen einbinden sowie NTFS-3G.

Rechte in symbolischer Darstellung

Im Terminal lassen sich die Rechte mit dem Befehl ls -l anzeigen. Im Folgenden sind als Beispiel die Dateirechte des Verzeichnisses /var/mail/ dargestellt

```
 ls -l /var/mail/
drwxrwsr-x 2 root mail 4096 Apr 23  2012 /var/mail/
```

Für die Darstellung der Rechte sind die markierten Teile der Ausgabe relevant:

- Der erste Buchstabe (d) kennzeichnet den Dateityp.
- Danach folgen die Zugriffsrechte (rwxrwsr-x).
- Eigentümer der Datei
- Gruppe

Wie auch in anderen Betriebssystemen kann man verschiedene Rechte für Eigentümer (Owner) und Gruppe (Group) vergeben. Neben Eigentümer und Gruppe gibt es noch eine weitere, allgemeine Gruppe. Diese Gruppe nennt sich andere (Others).

Darstellungsarten

Neben der symbolischen Darstellung (z.B. rwxrwxr-x) gibt es auch noch eine oktale Darstellung (nach dem Oktalsystem). Die Grundrechte (Lesen, Schreiben, Ausführen) und Kombinationen daraus werden hierbei durch eine einzelne Ziffer repräsentiert und dem Eigentümer, der Gruppe und allen anderen zugeordnet. Je nach Anwendung wird dabei von unterschiedlichen Grundwerten ausgegangen und entweder Rechte gegeben oder entzogen. Bei chmod wird beispielsweise von der Grundeinstellung „keine Rechte“ (000) ausgegangen und Rechte gegeben, wohingegen bei umask von „alle Rechte

vorhanden“ (777) ausgegangen und Rechte entzogen werden. Entsprechend sind die Werte je nach Anwendung anders.

Mögliche Werte für:

Recht(e)	**chmod (octal)**	**umask (octal)**	**Symbolisch**	**Binäre Entsprechung**
Lesen, schreiben und ausführen	7	0	rwx	111
Lesen und Schreiben	6	1	rw-	110
Lesen und Ausführen	5	2	r-x	101
Nur lesen	4	3	r–	100
Schreiben und Ausführen	3	4	-wx	011
Nur Schreiben	2	5	-w-	010
Nur Ausführen	1	6	–x	001
Keine Rechte	0	7	—	000

Hier ein paar Beispiele:

- rwxrwxrwx entspricht 0777 (chmod) oder 0000 (umask): Jeder darf lesen, schreiben und ausführen.
- rwxr-xr-x entspricht 0755 (chmod) oder 0022 (umask): Jeder darf lesen und ausführen, aber nur der Dateibesitzer darf diese Datei (oder das Verzeichnis) auch verändern.

Die nachfolgenden Erklärungen beziehen sich vor allem auf Dateien vom Typ File (ohne Kennbuchstaben) und „Ordner“ (Directory, Kennbuchstabe d).

Nach dem Dateityp kommen drei Zeichengruppen zu je drei Zeichen. Diese kennzeichnen die Zugriffsrechte für die Datei bzw. das Verzeichnis. Hat der Benutzer/Gruppe/andere ein Recht, so wird der Buchstabe dafür angezeigt; ansonsten wird ein - dafür angezeigt.

In obigen Beispiel erscheint nach dem Dateityp dann die Zeichenfolge rwxrwsr-x. Wenn man diese in drei Dreiergruppen aufteilt, erhält man diese Gruppen:

- rwx: Rechte des Eigentümers
- rws: Rechte der Gruppe
- r-x: Recht von allen anderen (others)

Die folgende Tabelle erklärt die Bedeutung der einzelnen Buchstaben, Diese stehen immer in der gleichen Reihenfolge:

Symbole für Zugriffsrechte	
Zeichen	**Bedeutung**
r	Lesen (read) Erlaubt lesenden Zugriff auf die Datei. Bei einem Verzeichnis können damit die Namen der enthaltenen Dateien und Ordner abgerufen werden (nicht jedoch deren weitere Daten wie z.B. Berechtigungen, Besitzer, Änderungszeitpunkt, Dateiinhalt etc.).
w	Schreiben (write) Erlaubt schreibenden Zugriff auf eine Datei. Für ein Verzeichnis gesetzt, können Dateien oder Unterverzeichnisse angelegt oder gelöscht werden, sowie die Eigenschaften der enthaltenen Dateien/Verzeichnisse verändert werden.
x	Ausführen (execute) Erlaubt das Ausführen einer Datei, wie das Starten eines Programms. Bei einem Verzeichnis ermöglicht dieses Recht, in diesen Ordner zu wechseln und weitere Attribute zu den enthaltenen Dateien abzurufen (sofern man die Dateinamen kennt ist dies unabhängig vom Leserecht auf diesen Ordner). Statt x kann auch ein Sonderrecht angeführt sein.

Sonderrechte

Die oben gezeigten Dateirechte kann man als Basisrechte bezeichnen. Für besondere Anwendungen gibt es zusätzlich noch besondere Dateirechte. Der Einsatz dieser ist nur dann ratsam, wenn man genau weiß, was man tut, da dies unter Umständen zu Sicherheitsproblemen führen kann.

Sonderrechte		
Zeichen	**Bedeutung**	**Beschreibung**
s	Set-UID-Recht (SUID-Bit)	Das Set-UID-Recht („Set User ID“ bzw. „Setze Benutzerkennung“) sorgt bei einer Datei mit Ausführungsrechten dafür, dass dieses Programm immer mit den Rechten des Dateibesitzers läuft. Bei Ordnern ist dieses Bit ohne Bedeutung.
s (S)	Set-GID-Recht (SGID-Bit)	Das Set-GID-Recht („Set Group ID“ bzw. „Setze Gruppenkennung“) sorgt bei einer Datei mit Ausführungsrechten dafür, dass dieses Programm immer mit den Rechten der Dateigruppe läuft. Bei einem Ordner sorgt es dafür, dass die Gruppe an Unterordner und Dateien vererbt wird, die in diesem Ordner neu erstellt werden.
t (T)	Sticky-Bit	Das Sticky-Bit („Klebrig“) hat auf modernen Systemen nur noch eine einzige Funktion: Wird es auf einen Ordner angewandt, so können darin erstellte Dateien oder Verzeichnisse nur vom Dateibesitzer gelöscht oder umbenannt werden. Verwendet wird dies z.B. für /tmp.

Die Symbole für die Sonderrechte erscheinen an der dritten Stelle der Zugriffsrechte, die normalerweise dem Zeichen x (für executable) vorbehalten ist, und ersetzen ggf. dieses. Die Set-UID/GID-Rechte werden anstelle des x für den Besitzer bzw. die Gruppe angezeigt, das Sticky-Bit anstelle des x für andere. Wenn das entsprechende Ausführrecht gesetzt ist, wird ein Kleinbuchstabe verwendet, ansonsten ein Großbuchstabe.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_04

Last update: **2019/06/16 10:00**

5.10.5) INODES

Ein Inode (englisch index node, gesprochen „eye-node“) ist die grundlegende Datenstruktur zur Verwaltung von Dateisystemen mit unixartigen Betriebssystemen. Jeder Inode wird innerhalb einer Partition eindeutig durch seine Inode-Nummer identifiziert. Jeder Namenseintrag in einem Verzeichnis verweist auf genau einen Inode. Dieser enthält die Metadaten der Datei und verweist auf die Daten der Datei beziehungsweise die Dateiliste des Verzeichnisses.

Die Anwendungssoftware unterscheidet beim Lesen oder Schreiben von Daten nicht mehr zwischen Gerätetreibern und regulären Dateien. Durch das Inode-Konzept gilt bei den Unixvarianten alles als Datei („On UNIX systems it is reasonably safe to say that everything is a file: …“). Dadurch unterscheiden sich solche Betriebssysteme in der Verwaltung ihres Datenspeichers von anderen Systemen wie Microsoft Windows mit NTFS, aber auch von VMS oder MVS.

Grundsätzliches

Speichert man eine Datei auf einem Computer ab, so muss nicht nur der Dateiinhalt (Nutzdaten) gespeichert werden, sondern auch Metadaten, wie zum Beispiel der Zeitpunkt der Dateierstellung oder der Besitzer der Datei. Gleichzeitig muss der Computer einem Dateinamen – inklusive Dateipfad – die entsprechenden Nutzdaten und Metadaten effizient zuordnen können. Die Spezifikation, wie diese Daten organisiert und auf einem Datenträger gespeichert werden, nennt man Dateisystem. Dabei gibt es abhängig von Einsatzbereich und Betriebssystem unterschiedliche Dateisysteme. Umgesetzt wird die Dateisystemspezifikation von einem Treiber, der wahlweise als ein Kernel-Modul des Betriebssystemkern (Kernel) oder seltener als gewöhnliches Programm im Userspace umgesetzt sein kann.



Dateisysteme unixoider Betriebssysteme – wie Linux und macOS – verwenden sogenannte Inodes. Diese enthalten die Metadaten sowie Verweise darauf, wo Nutzdaten gespeichert sind. An einem speziellen Ort des Dateisystems, dem Superblock, wird wiederum die Größe, Anzahl und Lage der Inodes gespeichert. Die Inodes sind durchnummeriert und an einem Stück auf dem Datenträger gespeichert. Das Wurzelverzeichnis eines Dateisystems besitzt eine feste Inodenummer. Unterordner sind „gewöhnliche“ Dateien, welche eine Liste der darin enthaltenen Dateien mit der Zuordnung der dazugehörenden Inodenummern als Nutzdaten enthalten.

Soll also beispielsweise die Datei /bin/ls geöffnet werden, so läuft dies, vereinfacht, wie folgt ab:

- Der Dateisystemtreiber liest den Superblock aus, dadurch erfährt er die Startposition der Inodes und deren Länge, somit kann nun jeder beliebige Inode gefunden und gelesen werden.
- Nun wird der Inode des Wurzelverzeichnisses geöffnet. Da dies ein Ordner ist, befindet sich darin ein Verweis auf die Speicherstelle der Liste aller darin enthaltenen Dateien mitsamt ihren Inodenummern. Darin wird das Verzeichnis bin gesucht.
- Nun kann der Inode des bin-Verzeichnisses gelesen werden und analog zum letzten Schritt der

Inode der Datei ls gefunden werden.
- Da es sich bei der Datei ls nicht um ein Verzeichnis, sondern um eine reguläre Datei handelt, enthält ihr Inode nun einen Verweis auf die Speicherstelle der gewünschten Daten.

Aufbau

Jedem einzelnen von einem Schrägstrich / (slash) begrenzten Namen ist genau ein Inode zugeordnet. Dieser speichert folgende Metainformationen zur Datei, aber nicht den eigentlichen Namen:

- Die Art der Datei (reguläre Datei, Verzeichnis, Symbolischer Link, …), siehe unten;
- die nummerische Kennung des Eigentümers (UID, user id) und der Gruppe (GID, group id);
- die Zugriffsrechte für den Eigentümer (user), die Gruppe (group) und alle anderen (others);
- Die klassische Benutzer- und Rechteverwaltung geschieht mit den Programmen chown (change owner), chgrp (change group) und chmod (change mode). Durch Access Control Lists (ACL) wird eine feinere Rechtevergabe ermöglicht.
- verschiedene Zeitpunkte der Datei: Erstellung, Zugriff (access time, atime) und letzte Änderung (modification time, mtime);
- die Zeit der letzten Status-Änderung des Inodes (status, ctime);
- die Größe der Datei;
- den Linkzähler (siehe unten);
- einen oder mehrere Verweise auf die Blöcke, in denen die eigentlichen Daten gespeichert sind.

Reguläre Dateien

Reguläre Dateien (engl. regular files) sind sowohl Anwenderdaten als auch ausführbare Programme. Letztere sind durch das executable-Recht gekennzeichnet und werden beim Aufruf durch das System in einem eigenen Prozess gestartet. Als „ausführbar“ gelten nicht nur kompilierte Programme, sondern auch Skripte, bei denen der Shebang den zu verwendenden Interpreter angibt. Bei „dünnbesetzten Dateien“, sogenannten sparse files, unterscheidet sich die logische Größe vom durch die Datenblöcke tatsächlich belegten Festplattenplatz.

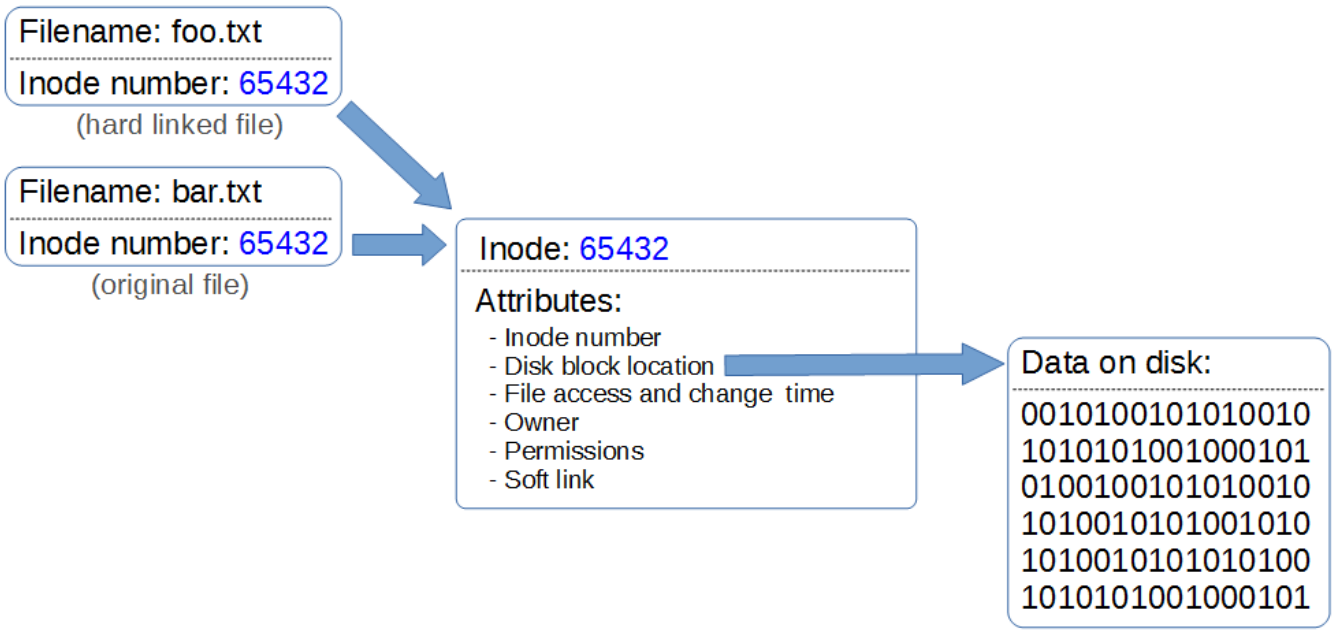
Verzeichnisse

Verzeichnisse sind Dateien, deren „Dateiinhalt“ aus einer tabellarischen Liste der darin enthaltenen Dateien besteht. Die Tabelle enthält dabei eine Spalte mit den Dateinamen und eine Spalte mit den zugehörigen Inodenummern. Bei manchen Dateisystemen umfasst die Tabelle noch weitere Informationen, so speichert ext4 darin auch den Dateityp aller enthaltenen Dateien ab, so dass dieser beim Auflisten eines Verzeichnisinhalts nicht aus den Inodes aller Dateien ausgelesen werden muss. Für jedes Verzeichnis existieren immer die Einträge . und .. als Verweise auf das aktuelle bzw. übergeordnete Verzeichnis.

Harte Links (hard links)

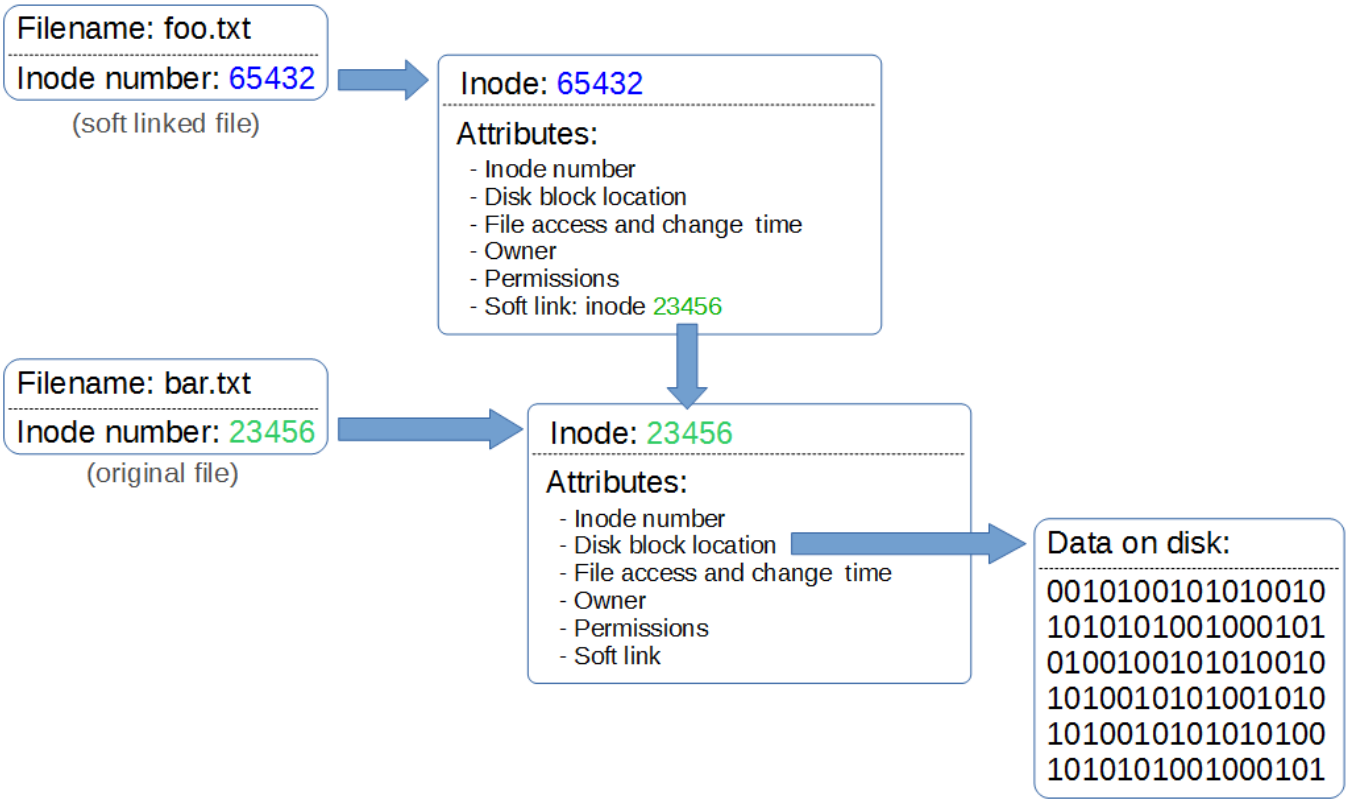
Bei Harten Links hingegen handelt es sich nicht um spezielle Dateien. Von einem Harten Link spricht man, wenn auf einen Inode mehrfach von verschiedenen Ordnern oder verschiedenen Dateinamen verwiesen wird. Alle Verweise auf den Inode sind gleichwertig, es gibt also kein Original. Im Inode gibt

der Linkzähler an, wie viele Dateiennamen auf diesen verweisen, er steht nach dem Anlegen einer Datei also bei 1 und wird erhöht, sobald weitere Hardlinks für diese Datei erstellt werden. Bei einem Verzeichnis beträgt er zwei mehr, als Unterordner darin enthalten sind, da neben dem Eintrag im Ordner darüber und dem Eintrag '.' im Ordner noch die Einträge '..' in allen Unterordnern darauf verweisen. Wird eine Datei gelöscht, so wird ihr Eintrag aus dem übergeordneten Verzeichnis entfernt und der Linkzähler um eins reduziert. Beträgt der Linkzähler dann 0, wird gegebenenfalls abgewartet, bis die Datei von keinem Programm mehr geöffnet ist, und erst anschließend der Speicherplatz freigegeben.



Symbolische Links (symbolic links, soft links, symlinks)

Bei symbolischen Links handelt es sich um spezielle Dateien, die anstelle von Daten einen Dateipfad enthalten, auf den der Link verweist. Je nach Dateisystem und Länge des Dateipfads wird der Link entweder direkt im Inode gespeichert oder in einem Datenblock, auf welchen der Inode verweist.



Praxis

Die Inodenummer einer Datei lässt sich mittels des Befehls ls -li Dateiname anzeigen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_05

Last update: **2019/05/14 15:59**

5.10.6) LINUX-DISTRIBUTIONEN

Wie schon besprochen besteht ein Linux-Betriebssystem aus dem Linux-Kernel und einer großen Anzahl verschiedener Anwenderprogramme. Tatsächlich gibt es ein Projekt, das eine Anleitung bietet, wie man aus den Kernelquellen und selbst selektierten Programmen ein komplettes, maßgeschneidertes Betriebssystem bauen kann. Das Projekt nennt sich „Linux From Scratch“ oder „LFS“ (www.linuxfromscratch.org) und ich kann jedem, der etwas Zeit übrig hat, nur empfehlen, dies selbst einmal zu probieren. Man erhält eine ultrakompaktes, ultraschnelles Betriebssystem und kann sagen, „das ist mein eigenes reinrassiges Linux-System“. Aber was kommt dann?

Man braucht schon einige Zeit bis alle Programme, die so benötigt werden, kompiliert und konfiguriert sind und dann müssen diese auch noch laufend aktualisiert werden. Sicherheitsupdates müssen selbst organisiert und kompiliert werden. Mit all der Administrationsarbeit kommt man zu sonst nichts mehr.

Um das zu vermeiden, gibt es Linux-Distributionen. Diese bieten nicht nur einen fertigen Satz von notwendigen Programmen, sondern auch die regelmäßige Versorgung mit Updates an. Die aus meiner Sicht wichtigsten Distributionen sollen hier aufgeführt werden:

Debian



„Debian“ (www.debian.org) ist eine nicht-kommerzielle Distribution und das „Debian-Projekt“ ist nach der „Debian-Verfassung“ geregelt, die eine demokratische Organisationsstruktur vorsieht. Darüberhinaus ist das Projekt über den „Debian Social Contract“ zu völlig freier Software verpflichtet. Mit einigen 1000 Mitarbeitern ist Debian der Gigant unter den Linux-Systemen.

Da bei Debian eine „stabile Version“ immer eine wirklich stabile Version ist, sind die Entwicklungszeiten relativ lang und böse Zungen behaupten auch, dass die stabile Version schon bei Erscheinen veraltet ist.

Allerdings bietet Debian auch immer schon die zukünftigen Versionen an und so gibt es mehrere Zweige, aus denen man sich bedienen kann:

stable - wirklich stabile Version, die auch für den kommerziellen Serverbetrieb geeignet ist!

testing - die zukünftige stable-Version. Ab einem gewissen Entwicklungsstand wird die Distribution „eingefroren“ (engl. „frozen“) - d.h. es werden keine neueren Versionen von Programmen mehr aufgenommen, sondern nur noch an der Fehlerbeseitigung bei den vorhandenen gearbeitet. Dies entspricht ungefähr dem Zustand, bei dem andere Distributoren ihre „stabilen“ Versionen veröffentlichen. Da ich schon mehrmals eine testing-Version ab dem Anfangsstadium benutzt habe, glaube ich sogar sagen zu können, dass testing nie so instabil ist, wie manche andere Distribution im „ausgereiften“ Zustand. Für den Desktopbetrieb kann ich ein eingefrorenes testing jedenfalls empfehlen.

unstable - ist der erste Anlaufpunkt für neue Versionen von Paketen und Programmen, bevor sie in testing integriert werden. Man installiert sich mit unstable das neueste vom neuen, muss aber wissen,

dass das nicht immer stabil ist.

experimental - ist kein vollständiger Zweig, denn es dient nur dazu, Programme und deren Funktionen zu testen, die sonst das ganze System gefährden würden. Es enthält immer nur die gerade getesteten, bzw. die von diesen benötigten Programmpakete.

Diese Zweige haben auch immer Codenamen und sind, einer Vorliebe der frühen Entwickler folgend, immer nach Figuren aus dem Film „Toy Story“ benannt. So heißt im Moment die stable-Version „Jessie“ und „Stretch“ ist testing. unstable ist immer „Sid“, der Junge von nebenan, der die Spielsachen zerstört, aber es lässt sich auch als Abkürzung für „still in development“ (noch in Entwicklung) deuten.

Und wer einen dieser Zweige installiert hat, kann auf eine unüberschaubare Vielzahl an Programmen zurückgreifen, auf Wunsch (und auf eigene Gefahr) auch aus den anderen Zweigen. Für Anfänger ist es wohl nur bedingt zu empfehlen, obwohl sich in den letzten Jahren sehr viel in Sachen Benutzerführung getan hat. Auch steht ein deutschsprachiges Forum (debianforum.de) zur Verfügung, wo man Hilfe bekommt und wo auch dumme Fragen gestellt werden dürfen. Für ambitionierte Linux-EinsteigerInnen, die sich auch mit den Möglichkeiten ihres Betriebssystems auseinandersetzen wollen, könnte es sogar die beste Distribution sein.

Fedora

Das nicht-kommerzielle „Fedora“ (fedoraproject.org) ist der Nachfolger des traditionsreichen, kommerziellen „Red Hat Linux“, welches nicht mehr selbständig weiterentwickelt wird. Statt dessen verkauft die Firma Red Hat, das auf Fedora basierende „Red Hat Enterprise Linux“.

Fedora ist eine sehr innovative Distribution und vor allem in den USA sehr beliebt. Es werden nur völlig unter freier Lizenz stehende Inhalte akzeptiert, weshalb nach der Installation zum Beispiel keine MP3-unterstützenden Programme zu finden sind. Für Anfänger gibt es bessere Distributionen.

Gentoo Linux



Das nicht-kommerzielle „Gentoo Linux“ (www.gentoo.de) ist eine quellbasierte Linux-Metadistribution - das heißt, alle Programme, inklusive des Kernels, werden selbst kompiliert. Das klingt sehr anstrengend, ist es aber gar nicht so, da die Distribution geeignete

Werkzeuge zur Verfügung stellt, mit denen dies einfachst möglich gelingt. Auch sorgt eine große, sehr aktive „Community“ bei jedem Problem für Rat und Hilfe. Dennoch ist sie für Linux-Neulinge wohl nicht empfehlenswert.

OpenSUSE



„openSUSE“ (www.opensuse.org) ist die zweitbeliebteste Distribution am Heim-PC. Die nicht-kommerzielle Variante der von Novell aufgekauften kommerziellen SUSE-Distribution (heute „SUSE Linux Enterprise“) glänzt mit einem universellen Konfigurationswerkzeug. Sie gilt als anfängerfreundlich. Die neueste Versionsnummer 42.1 bezieht sich übrigens auf die Antwort auf die Frage „nach dem Leben, dem Universum und dem ganzen Rest“ aus Douglas Adam's „Per Anhalter durch die Galaxis“. Schon 1996 hatte die Version 4.2 diesen Bezug.

Slackware



„Slackware“ (www.slackware.com) ist die älteste noch heute existierende Distribution. Sie verzichtet aus Prinzip auf grafische Einrichtungswerkzeuge und ist daher eher nur für fortgeschrittene BenutzerInnen geeignet.

Bisher nicht vorgekommen sind Distributionen, die nur Abwandlungen anderer Distributionen sind und häufig auch deren Quellen benutzen. Vor allem von Debian gibt es unzählige davon. Sie werden als „Derivate“, oder oft auch, etwas abfällig, als „Klone“ bezeichnet. Ein solcher „Debian-Klon“ hat allerdings Geschichte geschrieben:

Ubuntu

Das von der Firma des Gründers gesponserte kostenlose Betriebssystem soll nach dem Willen der Entwickler ein einfach zu installierendes und leicht zu bedienendes Betriebssystem mit aufeinander abgestimmter Software sein. Es bedient sich dazu aus den Quellen von Debian unstable und hat das Ziel, nach der enormen Popularität zu schließen, eindeutig geschafft.

Tatsächlich kann Ubuntu von der Live-CD mit wenigen Mausklicks problemlos auf die Festplatte installiert werden und dann erwartet den Benutzer ein weitgehend komplettes Betriebssystem mit vielen Multimedia-Programmen. Releases erscheinen mit schöner halbjährlicher Regelmäßigkeit und der Upgrade auf diese lässt sich ebenfalls auf Mausklick bewerkstelligen. Alle 2 Jahre gibt es eine „Versionen mit verlägerter Unterstützung“ (Long Term Support oder kurz LTS), die dann deutlich stabiler als die kürzer unterstützten ist. Für EinsteigerInnen ist Ubuntu (www.ubuntu.com) bestens geeignet.

Linux Mint



Diese Distribution war ursprünglich ein Ubuntu-Klon, aber neuerdings ist Linux Mint auch als LMDE - Linux Mint Debian Edition - erhältlich. Den Entwicklern ist es wichtig , die bestmögliche Integration von Programmen zu bieten, die bei Benutzern beliebt, aber eben nicht quelloffene freie Software sind. Die anderen Distribution, inklusive Ubuntu, bieten zwar auch die Installation von „non-free“-Paketen an, aber in einem eigenen Zweig und erst nach der Basisinstallation.

Für absolute Stabilität setzt Linux Mint immer auf LTS (Ubuntu) oder stable (Debian) Versionen auf, aber die integrierten Programme erhalten auch zwischenzeitig Versionsupgrades. Als Vorbild wird die Benutzerfreundlichkeit und Stabilität von Apple's OS X genannt. Auch Linux Mint (www.linuxmint.com) ist für EinsteigerInnen bestens geeignet.

GRAFISCHE OBERFLÄCHEN (GUIs, Desktops)

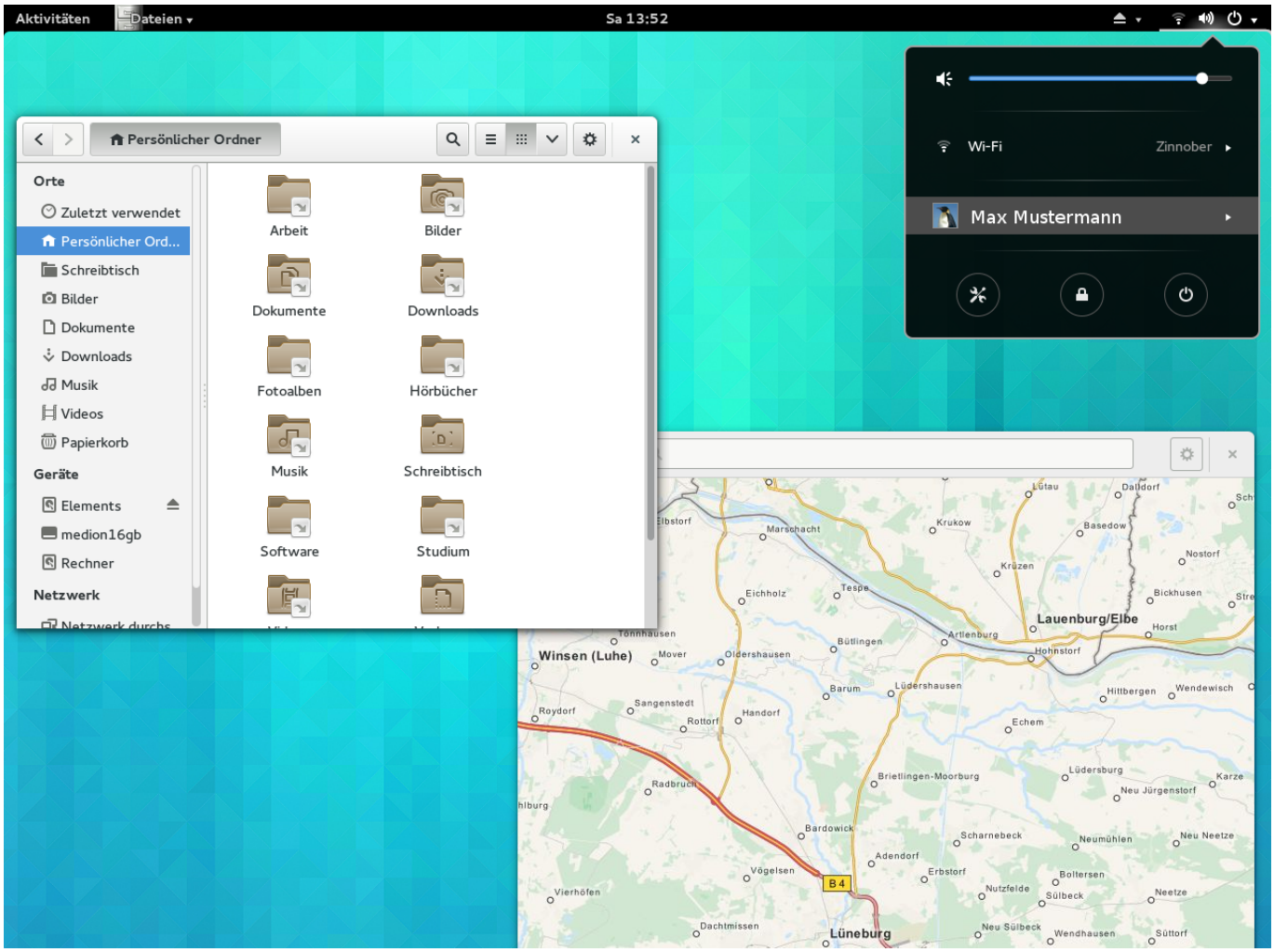
Anders als bei Microsoft und Apple gibt es bei Linux-Distributionen keinen festgelegten Desktop. Alle

Distributionen bieten die Möglichkeit den Desktop zu ändern und zunehmend kann schon bei der Installation der gewünschte Desktop festgelegt werden. Wenngleich für Anfänger im allgemeinen der Standard-Desktop der jeweiligen Distribution sicher eine gute Wahl ist, möchte ich abschließend auch noch kurz einige Desktops vorstellen. Gnome und KDE sind die Platzhirsche auf Linux-Bildschirmen.

Gnome

Gnome (Eigenschreibweise GNOME) [gnoʊm][3] ist eine Desktop-Umgebung für Unix- und Unix-ähnliche Systeme mit einer grafischen Benutzeroberfläche und einer Sammlung von Programmen für den täglichen Gebrauch. Gnome wird unter den freien Lizenzen GPL und LGPL veröffentlicht und ist Teil des GNU-Projekts.

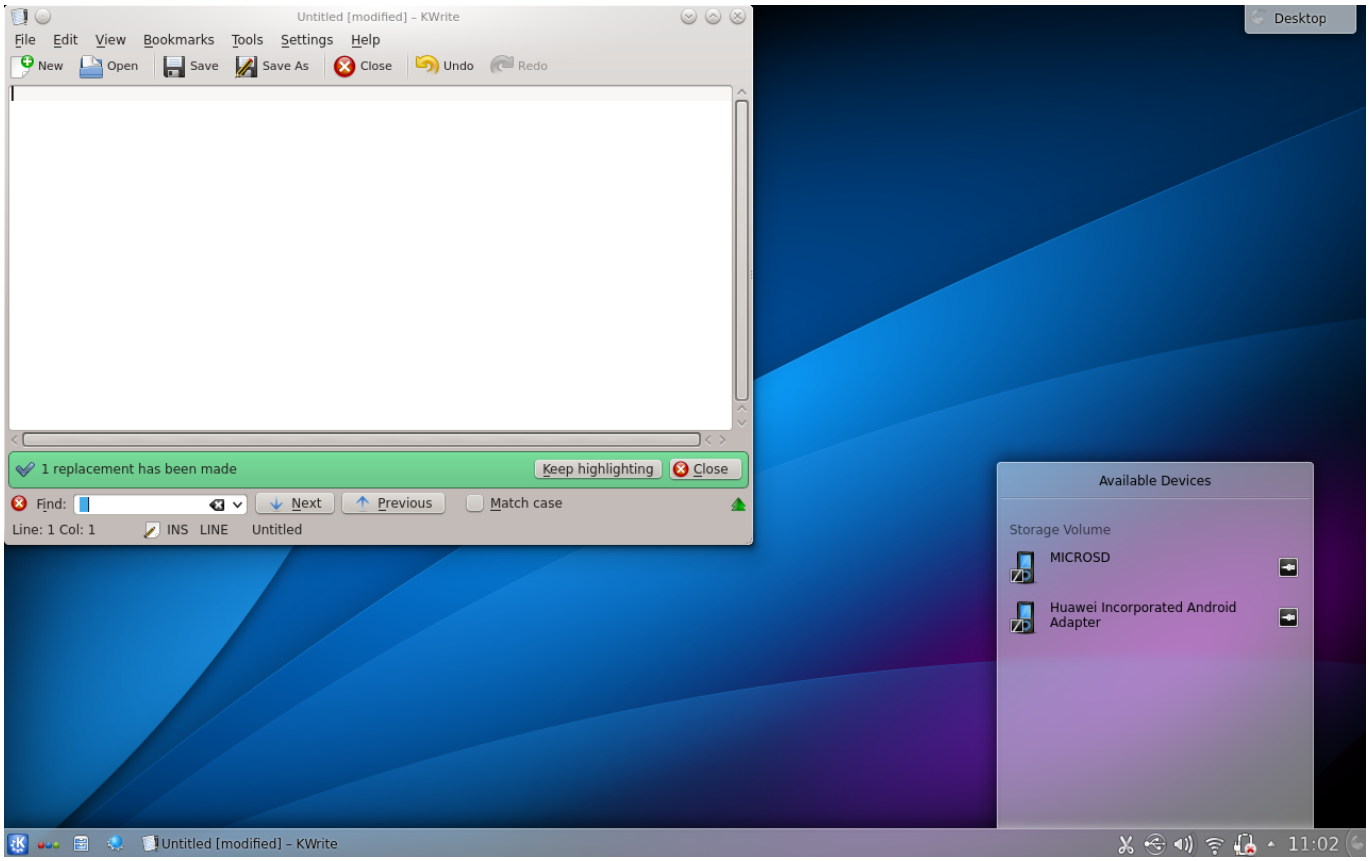
Gnome ist unter anderem der Standard-Desktop von Fedora und Ubuntu. Einige Komponenten von Gnome wurden nach Windows und MacOS portiert, etwa Evolution oder GStreamer, werden jedoch teilweise, wie im Falle von Evolution, nicht mehr länger gepflegt.



KDE

KDE ist eine Community, die sich der Entwicklung freier Software verschrieben hat. Eines der bekanntesten Projekte ist die Desktop-Umgebung KDE Plasma 5 (früher K Desktop Environment,

abgekürzt KDE).



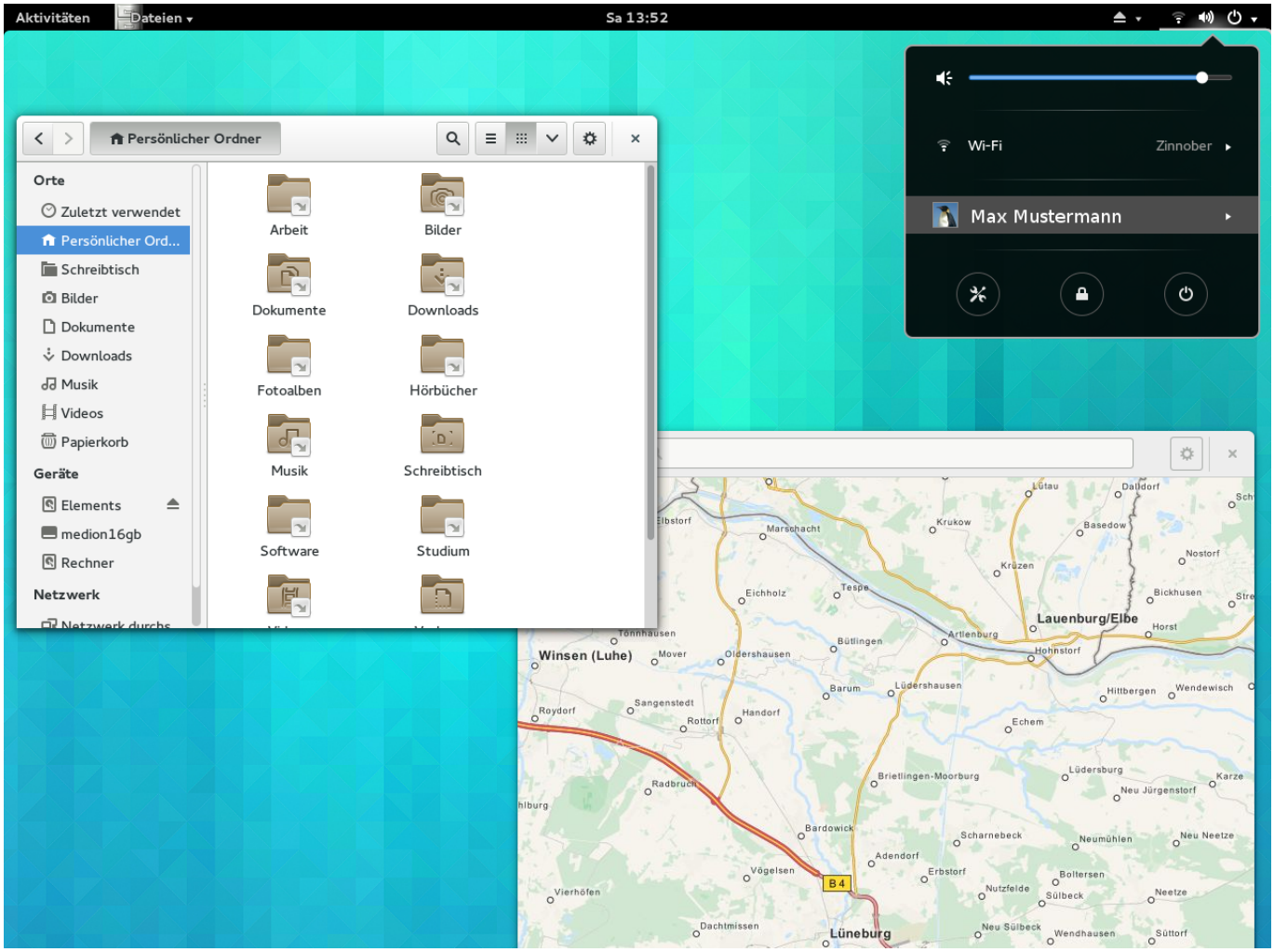
Unity

Unity ist eine durch das Unternehmen Canonical entwickelte Desktop-Umgebung für Linux-Betriebssysteme, die besonders sparsam mit Bildschirmplatz umgehen soll.



XFCE

Während erstgenannte Desktops mit Features protzen gehen die EntwicklerInnen bei XFCE einen anderen Weg. XFCE möchte einen schlanken, performanten Desktop bieten, der auf „unnötige Spielereien“ verzichtet und auch auf leistungsschwachen und zum Teil für andere Aufgaben (Multimedia) bestimmten Systemen resourcenschonend läuft. Viele heutige Distributionen bieten XFCE als vorkonfigurierte Alternative zum Standarddesktop an.



MATE und Cinnamon

Der umstrittene Upgrade von Gnome2 auf Gnome3 führte zur Geburt von MATE. Dieser Desktop ist eine direkte Fortentwicklung von Gnome2. Cinnamon dagegen ist aus den Quellen von Gnome3 entstanden, aber deutlich performanter als dieser. Was beide gemeinsam haben - sie sind die Standarddesktops von „Linux Mint“ und nur als Ubuntu-Editionen von Linux Mint gibt es auch KDE und XFCE vorkonfiguriert.

Es können auch mehrere Desktops gleichzeitig installiert werden. Bei der Anmeldung kann dann zwischen den Desktops gewechselt werden. So kann jeder seinen Lieblingsdesktop herausfinden.



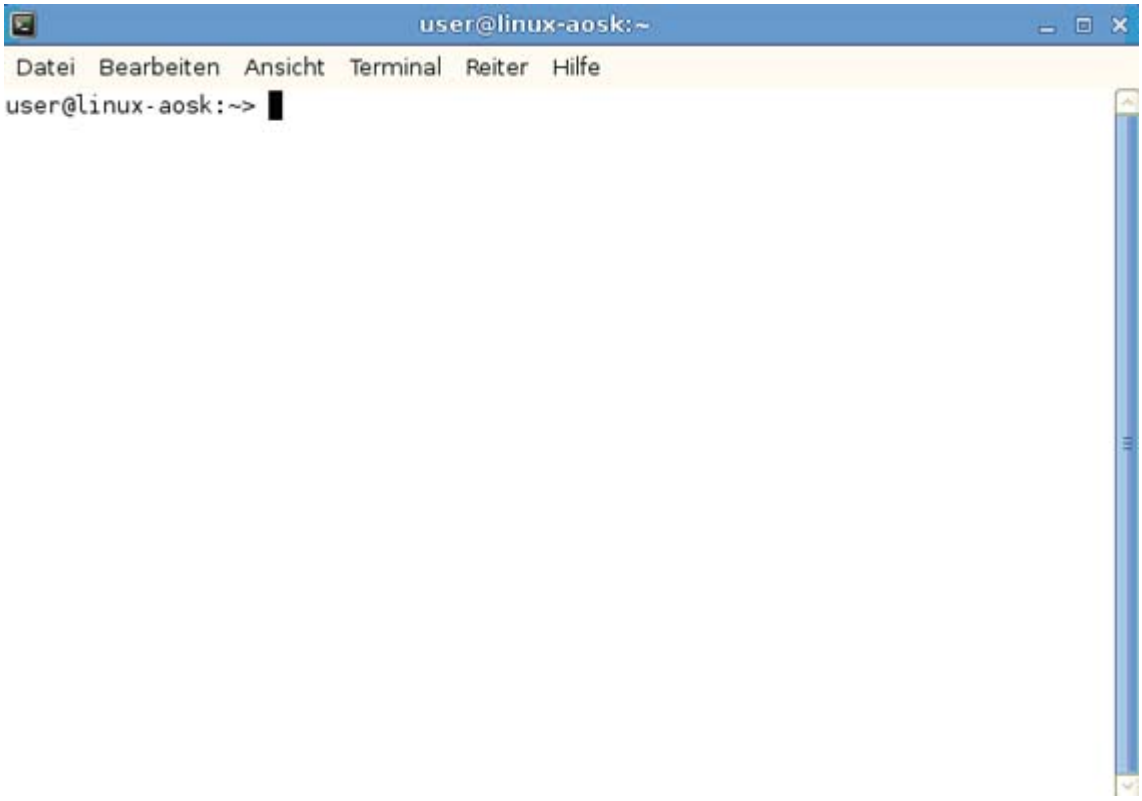
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_06

Last update: **2019/05/14 16:00**

5.10.7) Arbeiten auf der Konsole

Gnome Terminal findet man unter *Weiter Anwendungen* .



prompt *user@linux-aosk:~→* hat folgende Bedeutung:

- *user* : Benutzername
- *linux-aosk* : Rechnername
- ~ : Homeverzeichnis

Kommandos zur Bearbeitung von Dateien

Obwohl unter KDE und Gnome moderne Dateimanager zur Verfügung stehen, verwenden erfahrene Linux-Anwender oft noch immer diverse, text-orientierte Kommandos.

Kommando	**Beschreibung**	**Kommando in DOS**
Hilfe		
man *Befehl*	Hilfe zum Kommando	
Befehl - - help	Hilfe zum Kommando	
Als Root		
su	wechselt als Root (Passwort eingeben)	
sudo	einen Befehl als Root ausführen	
Verzeichnisbaum		
cd	wechselt das aktuelle Verzeichnis	
cd /	wechselt ins root-Verzeichnis	

Kommando	Beschreibung	Kommando in DOS
Hilfe		
ls	zeigt alle Dateien des aktuellen Verzeichnisse an	dir
ls -l	zeigt eine detaillierte Liste	
ls -a	zeigt versteckte Dateien an	
mkdir	erzeugt ein neues Verzeichnis	md
rmdir	löscht Verzeichnisse	rd
pwd	zeigt aktuellen Pfad an	
Joker		
*	steht für eine beliebige Anzahl von beliebigen Zeichen	
?	steht für ein beliebiges Zeichen	
Dateien		
mv quelle ziel	verschiebt Dateien bzw. ändert ihren Namen	move
cp quelle ziel	kopiert Dateien	copy
cp ordner ziel -r	kopiert gesamten ordner inkl. aller unterordner an ziel	
cat	zeigt Dateiinhalt an	type
less	öffnet Anzeigeprogramm	
more	zeigt Dateiinhalt seitenweise an	
touch Dateiname	erstellt leere Datei	
mcedit Dateiname	öffnet Datei in einem Editor zur Bearbeitung	edit
vim Dateiname	öffnet Datei mit dem Editor VIM zur Bearbeitung	
vimtutor	Tutorial zum Erlernen vom Editor VIM	
rm	löscht Dateien	del
rm unterordner -r	löscht gesamten Unterordner inkl. aller Dateien	
find -name dateinamen	sucht Dateien nach Namen	
Packen und Komprimieren von Verzeichnissen und Dateien		
tar	vereint mehrere Dateien (und Verzeichnisse) in einer Datei	
tar -t	Inhalt eines Archivs anzeigen	
tar -x	Dateien aus Archiv holen	
tar -c	neues Archiv erzugen	
tar -f	um Namen des Archiv anzugeben	
tar -xvjf	entzippen	

Weitere Befehle

- http://www.admintalk.de/konsolenbefehle.php
- http://www.shellbefehle.de/befehle/

Übung 1

1. Erstelle in deinem Home-Directory einen Ordner uebungen.
2. Speichere das File uebung1.tar in diesen Ordner!
3. Entpacke das Archiv mit Hilfe von `tar -xvjf uebung1.tar`. Welche Verzeicnisse und Dateien befinden sich nun in deinem Home-Directory?
4. Gib den Befehl `./hallo` ein.

5. Finde die Datei `ichbinhier`.
6. Wechsle in das Verzeichnis, in dem sich die Datei befindet.
7. Erstelle ein Verzeichnis mit dem Namen `backup` in deinem Homedirectory.
8. Kopiere das gesamte Verzeichnis `uebung1` in das Verzeichnis `backup`.
9. Lösche das Verzeichnis `uebung1`.
10. Erstelle ein Verzeichnis mit dem Namen `aufgabe` und wechsle hinein.
11. Erstelle drei leere Dateien `datei1` bis `datei3`.
12. Öffne mit einem Editor `datei1` und gib drei Zeilen Text ein. Speicher ab!
13. Lasse dir die Datei mit einem entsprechendem Kommando ausgeben!
14. Gib den Befehl `tac datei1` ein. Was passiert?
15. Wechsle in die grafische Oberfläche!
16. Orientiere dich an der Oberfläche!
17. Versuche den Bildschirmhintergrund umzustellen.
18. Öffne ein Konsolenfenster. Lösche darin den gesamten Ordner `uebungen` inklusive Unterverzeichnis.

Übung 2

Grundkurs 1-3 & 7, Für Experten 4-6

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_07

Last update: **2019/05/14 16:00**

5.10.8) Shell Scripts sh (Bourne Shell)

Shell-Scripts (Kommandoprozeduren) sind unter Unix das Analogon zu Batch-Dateien (Stapeldateien) von MS-DOS, sie sind jedoch wesentlich leistungsfähiger. Ihre Syntax hängt allerdings von der verwendeten „Shell“ ab. In diesem Artikel werden Bourne Shell (sh) Scripts betrachtet. Sie gelten im allgemeinen als zuverlässiger als csh-Scripts.

Ein Shell-Script ist eine Textdatei, in der Kommandos gespeichert sind. Es stellt selbst ein Kommando dar und kann wie ein Systemkommando auch mit Parametern aufgerufen werden. Shell-Scripts dienen der Arbeitserleichterung und (nach ausreichenden Tests) der Erhöhung der Zuverlässigkeit, da sie gestatten, häufig gebrauchte Sequenzen von Kommandos zusammenzufassen.

Aufruf

Bourne Shell Scripts lassen sich generell wie folgt aufrufen:

```
sh myscript
```

Im allgemeinen ist es aber praktischer, die Datei als ausführbar anzumelden (execute permission), mittels

```
chmod +x myscript
```

Das Shell-Script läßt sich dann wie ein „normales“ (binäres) Kommando aufrufen:

```
myscript
```

Vorausgesetzt ist hierbei allerdings, daß das Betriebssystem das Script mit der richtigen Shell abarbeitet. Moderne Unix-Systeme prüfen zu diesem Zweck die erste Zeile. Für die sh sollte sie folgenden Inhalt haben:

```
#!/bin/sh
```

Obwohl das Doppelkreuz normalerweise einen Kommentar einleitet, der vom System nicht weiter beachtet wird, erkennt es hier, daß die „sh“ (mit absoluter Pfadangabe: /bin/sh) eingesetzt werden soll.

Ein Beispiel

```
#!/bin/sh
# Einfaches Beispiel
echo Hallo, Welt!
echo Datum, Uhrzeit und Arbeitsverzeichnis:
date
pwd
```

```
echo Uebergabe-Parameter: $*
```

Das vorstehende einfache Script enthält im wesentlichen normale Unix-Kommandos. Abgesehen von der ersten Zeile liegt die einzige Besonderheit im Platzhalter „$*“, der für alle Kommandozeilen-Parameter steht.

Testen eines Shell-Scripts

```
sh -n myscript
```

Syntax-Test (die Kommandos werden gelesen und geprüft, aber nicht ausgeführt)

```
sh -v myscript
```

Ausgabe der Shell-Kommandos in der gelesenen Form

```
sh -x myscript
```

Ausgabe der Shell-Kommandos nach Durchführung aller Ersetzungen, also in der Form, wie sie ausgeführt werden

Kommandozeilen-Parameter

```
$0
```

Name der Kommandoprozedur, die gerade ausgeführt wird

```
$#
```

Anzahl der Parameter

```
$1
```

erster Parameter

```
$2
```

zweiter Parameter

```
$3
```

dritter ... Parameter

```
$*
```

steht für alle Kommandozeilen-Parameter ($1 $2 $3 ...)

```
$@
```

wie $* ($1 $2 $3 ...)

```
$$
```

Prozeßnummer der Shell (nützlich, um eindeutige Namen für temporäre Dateien zu vergeben)

```
$-
```

steht für die aktuellen Shell-Optionen

```
$?
```

gibt den Return-Code des zuletzt ausgeführten Kommandos an (0 bei erfolgreicher Ausführung)

```
$!
```

Prozessnummer des zuletzt ausgeführten Hintergrund-Prozesses

Beispiel

```
#!/bin/sh
# Variablen
echo Uebergabeparameter: $*
echo user ist: $USER
echo shell ist: $SHELL
echo Parameter 1 ist: $1
echo Prozedurname ist: $0
echo Prozessnummer ist: $$
echo Anzahl der Parameter ist: $#
a=17.89        # ohne Luecken am = Zeichen
echo a ist $a
```

Prozesssteuerung

Bedingte Ausführung: if

```
if [ bedingung ]
  then kommandos1
  else kommandos2
fi
```

Anmerkungen: „fi“ ist ein rückwärts geschriebenes „if“, es bedeutet „end if“ (diese Schreibweise ist eine besondere Eigenheit der Bourne Shell). Die „bedingung“ entspricht der Syntax von test, siehe auch weiter unten. Im if-Konstrukt kann der „else“-Zweig entfallen, andererseits ist eine Erweiterung

durch einen oder mehrere „else if“-Zweige möglich, die hier „elif“ heißen:

```
if [ bedingung1 ]
    then kommandos1
  elif [ bedingung2 ]
    then kommandos2
  else kommandos3
fi
```

Die Formulierung

```
if [ bedingung ]
```

ist äquivalent zu

```
if test bedingung
```

Alternativ ist es möglich, den Erfolg eines Kommandos zu prüfen:

```
if kommando
```

(beispielsweise liefert das Kommando „true“ stets „wahr“, das Kommando „false“ hingegen „unwahr“)

Wichtige Vergleichsoperationen (test)

Hinweis: Es ist unbedingt notwendig, daß alle Operatoren von Leerzeichen umgeben sind, sonst werden sie von der Shell nicht erkannt! (Das gilt auch für die Klammern.)

Zeichenketten

```
"s1" = "s2"
```

wahr, wenn die Zeichenketten gleich sind

```
"s1" != "s2"
```

wahr, wenn die Zeichenketten ungleich sind

```
-z "s1"
```

wahr, wenn die Zeichenkette leer ist (Länge gleich Null)

```
-n "s1"
```

wahr, wenn die Zeichenkette nicht leer ist (Länge größer als Null)

(Ganze) Zahlen

```
n1 -eq n2
```

wahr, wenn die Zahlen gleich sind

```
n1 -ne n2
```

wahr, wenn die Zahlen ungleich sind

```
n1 -gt n2
```

wahr, wenn die Zahl n1 größer ist als n2

```
n1 -ge n2
```

wahr, wenn die Zahl n1 größer oder gleich n2 ist

```
n1 -lt n2
```

wahr, wenn die Zahl n1 kleiner ist als n2

```
n1 -le n2
```

wahr, wenn die Zahl n1 kleiner oder gleich n2 ist

Sonstiges

```
!
```

Negation

```
-a
```

logisches „und“

```
-o
```

logisches „oder“ (nichtexklusiv; -a hat eine höhere Priorität)

```
\( ... \)
```

Runde Klammern dienen zur Gruppierung. Man beachte, daß sie durch einen vorangestellten Backslash, \, geschützt werden müssen.

```
-f filename
```

wahr, wenn die Datei existiert. (Weitere Optionen findet man in der man page zu test)

Beispiel:

```
#!/bin/sh
# Interaktive Eingabe, if-Abfrage
echo Hallo, user, alles in Ordnung?
echo Ihre Antwort, n/j:
read answer
echo Ihre Antwort war: $answer
# if [ "$answer" = "j" ]
if [ "$answer" != "n" ]
  then echo ja
  else echo nein
fi
```

Mehrfachentscheidung: case

```
case var in
  muster1) kommandos1 ;;
  muster2) kommandos2 ;;
  *) default-kommandos ;;
esac
```

Anmerkungen: „esac“ ist ein rückwärts geschriebenes „case“, es bedeutet „end case“. Die einzelnen Fälle werden durch die Angabe eines Musters festgelegt, „muster)“, und durch ein doppeltes Semikolon abgeschlossen. (Ein einfaches Semikolon dient als Trennzeichen für Kommandos, die auf derselben Zeile stehen.) Das Muster „*)“ wirkt als „default“, es deckt alle verbleibenden Fälle ab; die Verwendung des default-Zweiges ist optional.

Beispiel:

```
#!/bin/sh
# Interaktive Eingabe, Mehrfachentscheidung (case)
echo Alles in Ordnung?
echo Ihre Antwort:
read answer
echo Ihre Antwort war: $answer
case $answer in
  j*|J*|y*|Y*) echo jawohl ;;
  n*|N*) echo nein, ueberhaupt nicht! ;;
  *) echo das war wohl nichts ;;
esac
```

Schleife: for

```
for i in par1 par2 par3 ...
  do kommandos
done
```

Anmerkungen: Die for-Schleife in der Bourne Shell unterscheidet sich von der for-Schleife in üblichen Programmiersprachen dadurch, daß nicht automatisch eine Laufzahl erzeugt wird. Der Schleifenvariablen werden sukzessive die Parameter zugewiesen, die hinter „in“ stehen. (Die Angabe „for i in $*“ kann durch „for i“ abgekürzt werden.)

Beispiel:

```
#!/bin/sh
# Schleifen: for
echo Uebergabeparameter: $*
# for i
for i in $*
  do echo Hier steht: $i
done
```

Schleife: while und until

```
while [ bedingung ]
  do kommandos
done
until [ bedingung ]
  do kommandos
done
```

Anmerkung: Bei „while“ erfolgt die Prüfung der Bedingung vor der Abarbeitung der Schleife, bei „until“ erst danach. (Anstelle von „[bedingung]“ oder „test bedingung“ kann allgemein ein Kommando stehen, dessen Return-Code geprüft wird; vgl. die Ausführungen zu „if“.)

Beispiel:

```
#!/bin/sh
# Schleifen: while
# mit Erzeugung einer Laufzahl
i=1
while [ $i -le 5 ]
do
  echo $i
  i=`expr $i + 1`
done
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:05:05_10:05_10_08

Last update: **2019/05/14 16:00**

6) Hardware

Hardware ist der Oberbegriff für die **physischen Komponenten (die elektronischen und mechanischen Bestandteile)** eines datenverarbeitenden Systems, als **Gegenteil zu Software (den Programmen und Daten)**

Wortherkunft

„Hardware“ kommt ursprünglich aus dem Englischen und bedeutet übersetzt Eisenwaren.

Hardware vs. Software

- Hardware = sind alle greifbaren/sichtbaren Elemente eines PCs
- Software = Programme und Daten, also nicht greifbare Elemente eines PCs

Überblick

- 6.1) EVA-Prinzip
- 6.2) Von-Neumann-Architektur
- 6.3) HW-Komponenten
- 6.4) Schnittstellen
- 6.5) Fragen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

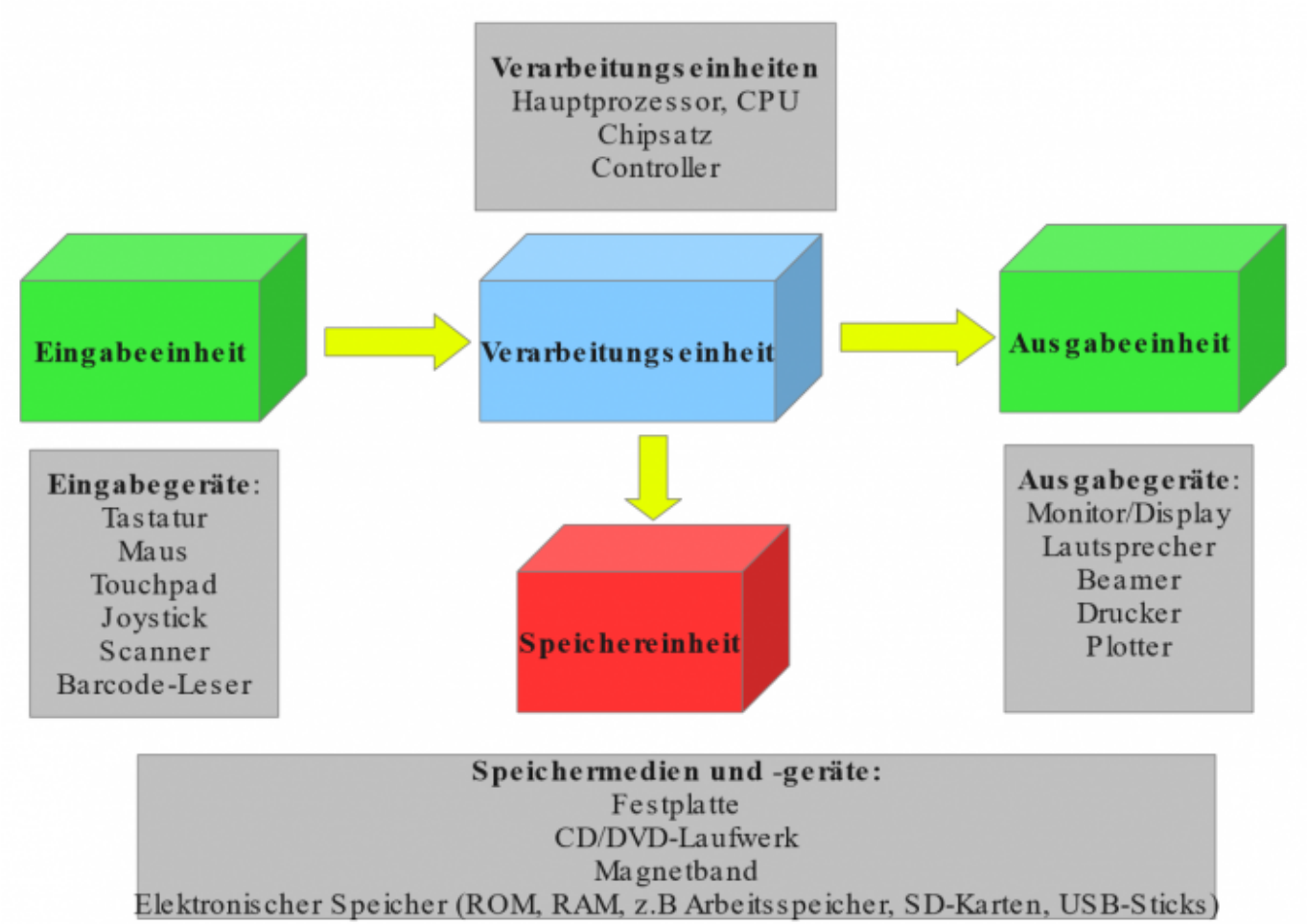
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06

Last update: **2019/05/14 21:53**

6.1) EVA/IPO Prinzip

Das **EVA**-Prinzip ist ein Grundprinzip für die Datenverarbeitung und beschreibt die Reihenfolge in der Daten verarbeitet werden.

Eingabe (**I**nput) - **V**erarbeitung (**P**rocess) - **A**usgabe (**O**utput)



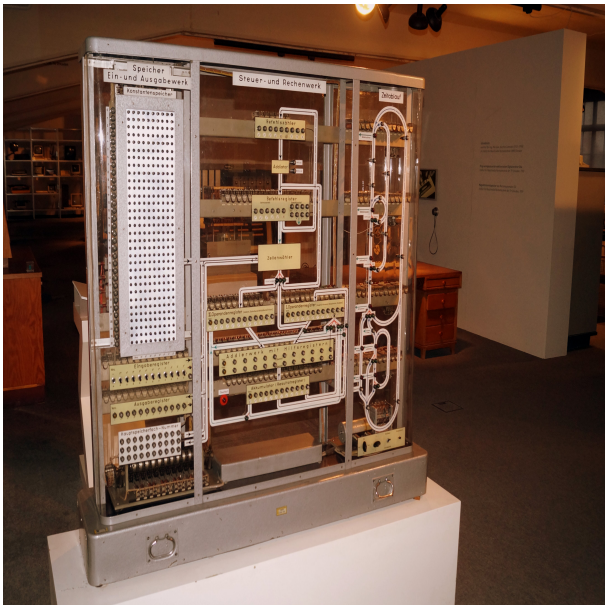
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_01

Last update: **2019/05/14 20:24**

6.2) Von-Neumann-Architektur (VNA)

Die **V**on-**N**eumann-**A**rchitektur ist ein Modell für Computer und bietet die Grundlage für alle heutigen Computer. Das Modell wurde von Johann von Neumann im Jahr 1945 entwickelt. Heute ist Johann von Neumann unter seinem amerikanischen Namen John von Neumann bekannt.

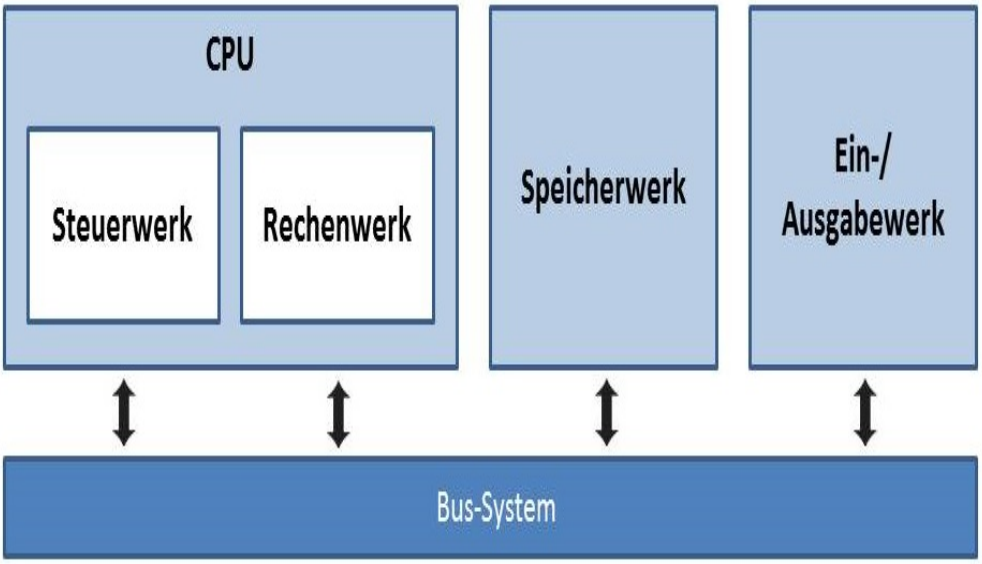


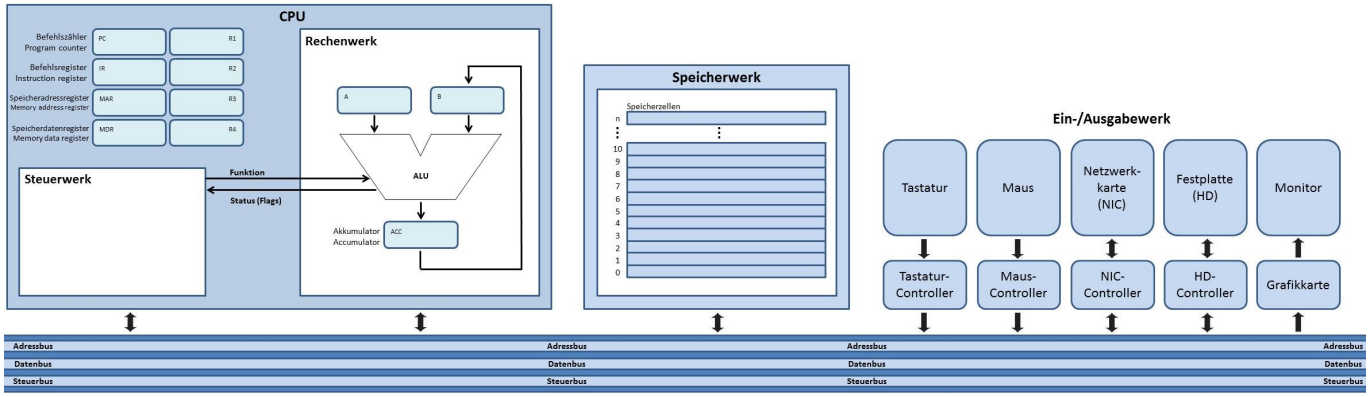
Er revolutionierte die bisherigen Computer, da durch seine Architektur verschiedene Programme auf derselben Hardware laufen konnten.

Komponenten

Ein Von-Neumann-Rechner beruht auf folgenden Komponenten, die bis heute in Computern verwendet werden:

- ALU (Arithmetic Logic Unit) – Rechenwerk, selten auch Zentraleinheit oder Prozessor genannt, führt Rechenoperationen und logische Verknüpfungen durch. (Die Begriffe Zentraleinheit und Prozessor werden im Allgemeinen in anderer Bedeutung verwendet.)
- Control Unit – Steuerwerk oder Leitwerk, interpretiert die Anweisungen eines Programms und verschaltet dementsprechend Datenquelle, -senke und notwendige ALU-Komponenten; das Steuerwerk regelt auch die Befehlsabfolge.
- Bussystem: Dient zur Kommunikation zwischen den einzelnen Komponenten (Steuerbus, Adressbus, Datenbus)
- Memory – Speicherwerk speichert sowohl Programme als auch Daten, welche für das Rechenwerk zugänglich sind.
- I/O Unit – Eingabe-/Ausgabewerk steuert die Ein- und Ausgabe von Daten, zum Anwender (Tastatur, Bildschirm) oder zu anderen Systemen (Schnittstellen).





Register

- **Befehlszähler (Program Counter - PC)**

Enthält die Speicheradresse vom Speicherwerk des aktuellen Befehls. (Startadresse = 0x0000). Wird nach jedem Befehl um 1 erhöht.

- **Befehlsregister (Instruction Register - IR)**

Speichert den vom Speicherwerk zurückbekommenen Befehl.

- **Speicheradressregister (Memory Address Register - MAR)**

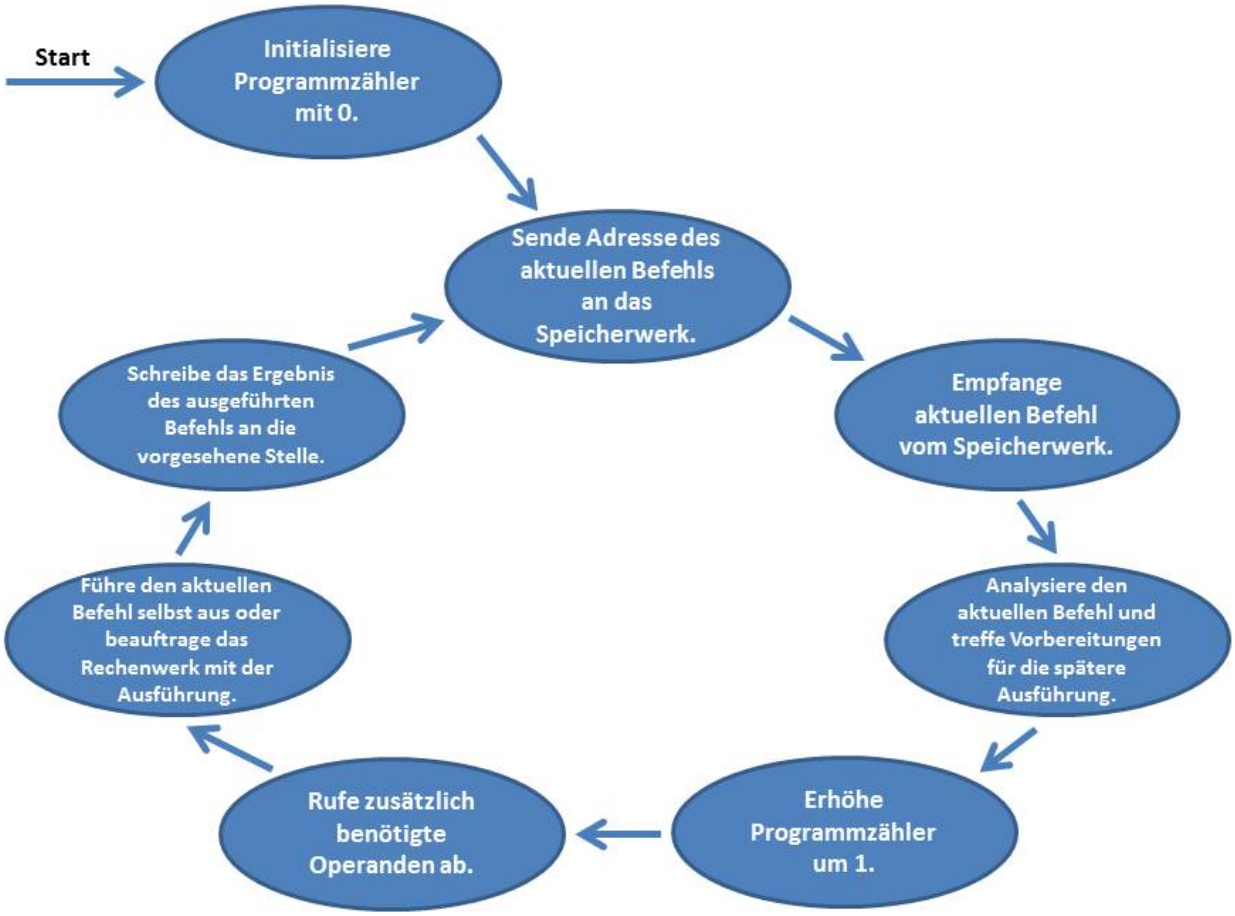
Ausschließlich für die Kommunikation zwischen Steuerwerk und Rechenwerk. Im MAR legt das Steuerwerk jeweils die Adresse ab, welche im Speicherwerk angesprochen werden soll.

- **Speicherdatenregister (Memory Data Register - MDR)**

Ausschließlich für die Kommunikation zwischen Steuerwerk und Rechenwerk. Bei einem Lesezugriff auf die Speicherzelle wird der vom Speicherwerk über den Datenbus bereitgestellte Wert im MDR abgelegt und kann von hier aus weiter verarbeitet werden. Bei einem Schreibzugriff muss sich im MDR der zu schreibende Wert befinden, so dass er über den Datenbus an das Speicherwerk übermittelt werden kann.

Prozesszyklus

1. Initialisiere das Befehlszählerregister (Program Counter- PC) mit 0 (Start)
2. Sende Adresse des aktuellen Befehls zum Speicherwerk
3. Empfange aktuellen Befehl vom Speicherwerk und speichere diesen in das Befehlsregister (Instruction Register - IR)
4. Analysiere aktuellen Befehl und treffe Vorbereitungen für die spätere Ausführung (Welcher Befehl und was ist dazu notwendig?)
5. Erhöhe den Befehlszähler (PC) um 1
6. Rufe zusätzlich benötigte Operanden ab (z.B.: Befehl ADD OP1 OP2)
7. Führe den Befehl selbst (Steuerwerk) aus oder beauftrage das Rechenwerk für die Ausführung
8. Schreibe das Ergebnis des ausgeführten Befehls an die vorgesehene Stelle





Arbeitsweise des Steuer- und Rechenwerks



Arbeitsweise des Speicherwerks



Befehlszähler und Befehlsregister im Zusammenspiel mit dem Bus-System



Arbeitsweise des Steuerwerks

Die 7 Prinzipien der Von-Neumann-Architektur

- Rechner besteht aus fünf Funktionseinheiten
- Struktur des Rechners ist unabhängig vom zu bearbeitenden Problem. Zur Lösung eines Problems muss Programm im Speicher abgelegt werden.
- Programme, Daten und Ergebnisse werden im selben Speicher abgelegt.
- Der Speicher ist in fortlaufenden nummerierten Zellen unterteilt. Über die Adresse einer Speicherzelle kann auf den Inhalt zugegriffen werden.
- Aufeinanderfolgende Befehle eines Programms werden in aufeinanderfolgende Speicherzellen abgelegt.
- Durch Sprungbefehle kann von der gespeicherten Befehlsreihenfolge abgewichen werden.
- Es gibt zumindest
    - arithmetische Befehle (Addition, Subtraktion, Multiplikation)
    - logische Befehle (EQUAL, NOR, AND, OR)
    - Transportbefehle, z.B. von Speicher zu Rechenwerk und für Ein- und Ausgabe
- Alle Daten (Befehle, Adressen, usw.) werden binär kodiert

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_02

Last update: **2019/05/14 20:23**

6.3) Hardware-Komponenten

Grundbestandteile

- Stromversorgung
- 6.3.1) Motherboard/Mainboard - Hauptplatine
- 6.3.2) Central Processing Unit (CPU) - zentrale Recheneinheit (Prozessor)
- 6.3.3) Random Access Memory (RAM) - Arbeitsspeicher









Massenspeicher

- 6.3.4) Datenspeicher




SAMSUNG
Solid State Drive
SAMSUNG
Solid State Drive


ASUS
QuieTrack



Erweiterungskarten

- 6.3.5) Erweiterungskarten









Peripheriegeräte

Eingabegeräte

- 6.3.6) Eingabegeräte

Ausgabegeräte

- 6.3.7) Ausgabegeräte









From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

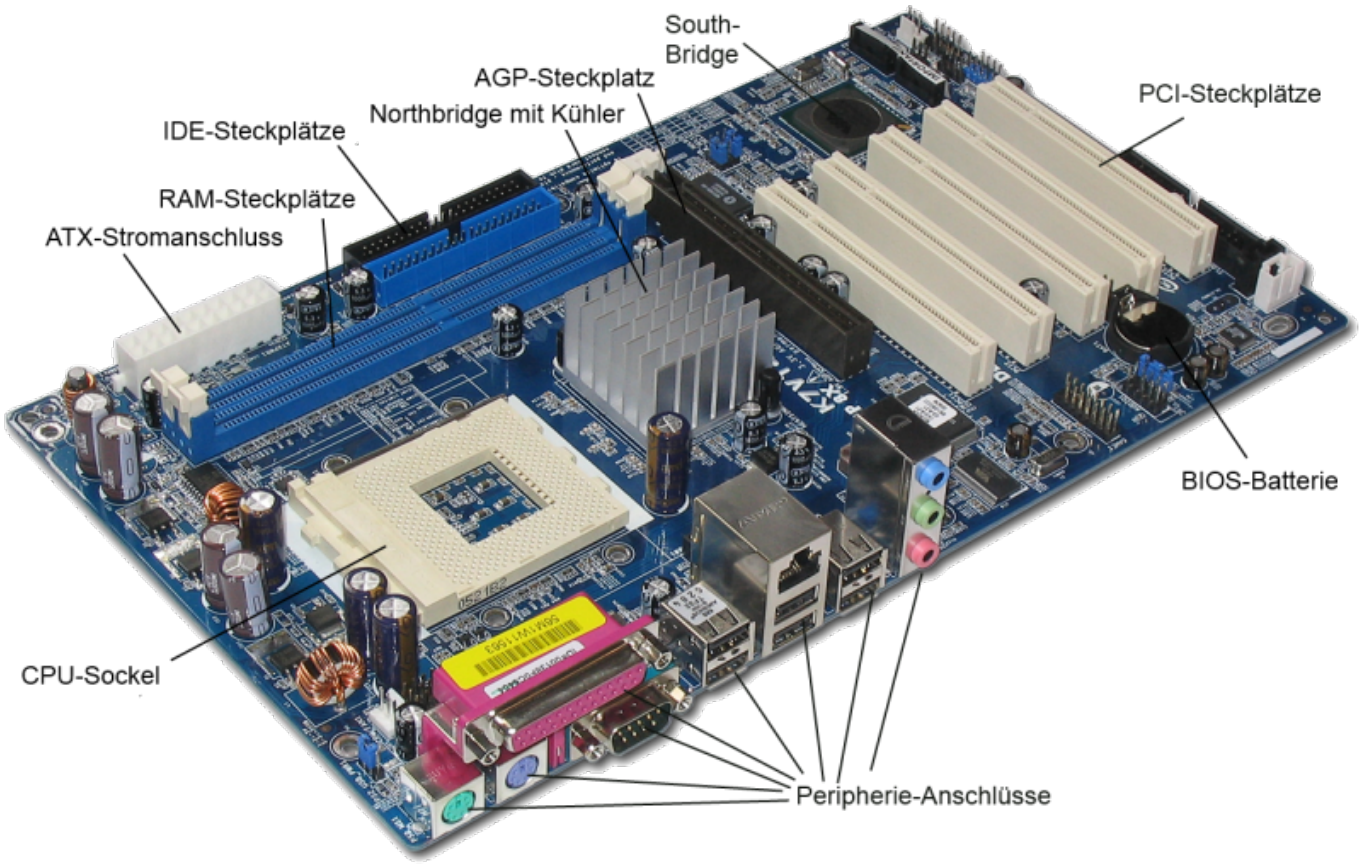
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03

Last update: **2019/05/14 20:49**

6.3.1) Motherboard / Mainboard / Hauptplatine

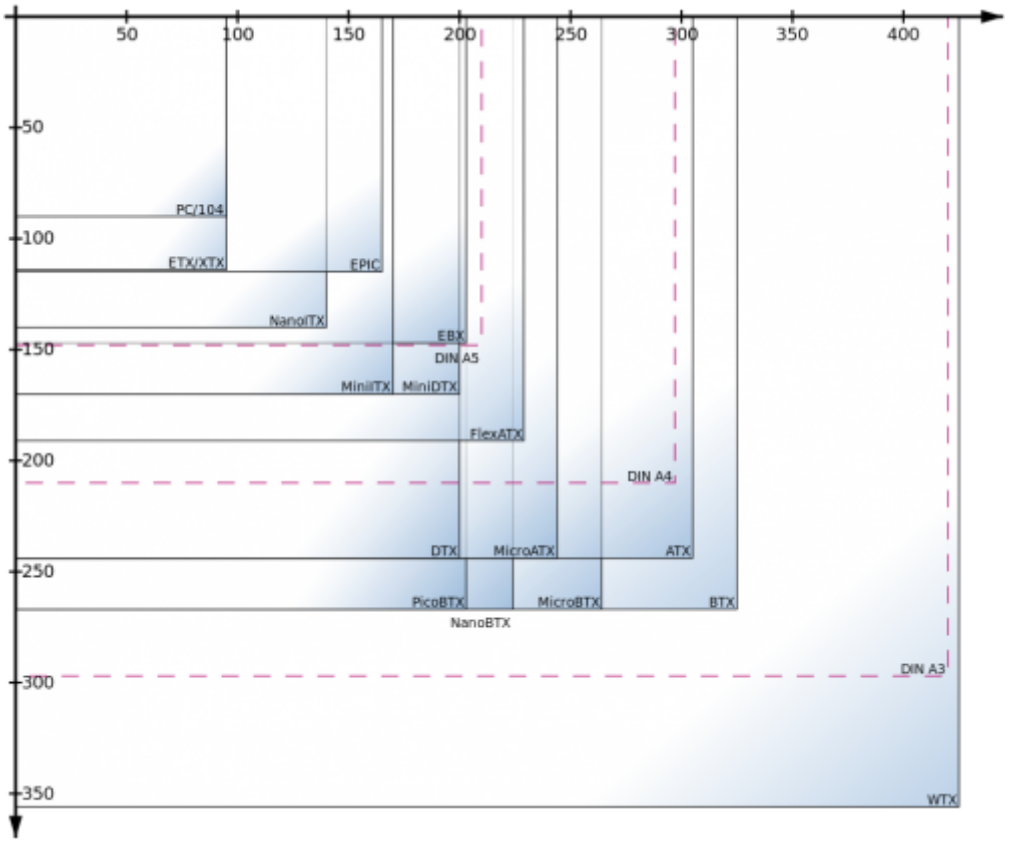
Das Motherboard ist der **Grundbaustein eines Computers**. Das Motherboard ist die Platine, auf der alle Systemkomponenten eines Computers eine **physikalische und logische Verbindung** erhalten. Die wichtigsten und einige leistungsbeeinflussenden Bauteile sind fest auf dieser Platine miteinander verbunden.

Die **Ausstattung des Motherboards bestimmt die System-Leistung, Erweiterbarkeit und Zukunftsfähigkeit eines Computersystems**. Die meisten Motherboards sind auf eine bestimmte Anwendung mit einem bestimmten Prozessor zugeschnitten. Man kann also nicht jeden beliebigen Prozessor auf jedem Motherboard verwenden. Der **Einsatz eines Prozessors hängt vom Motherboard bzw. vom Prozessorsockel und dem Chipsatz ab**.



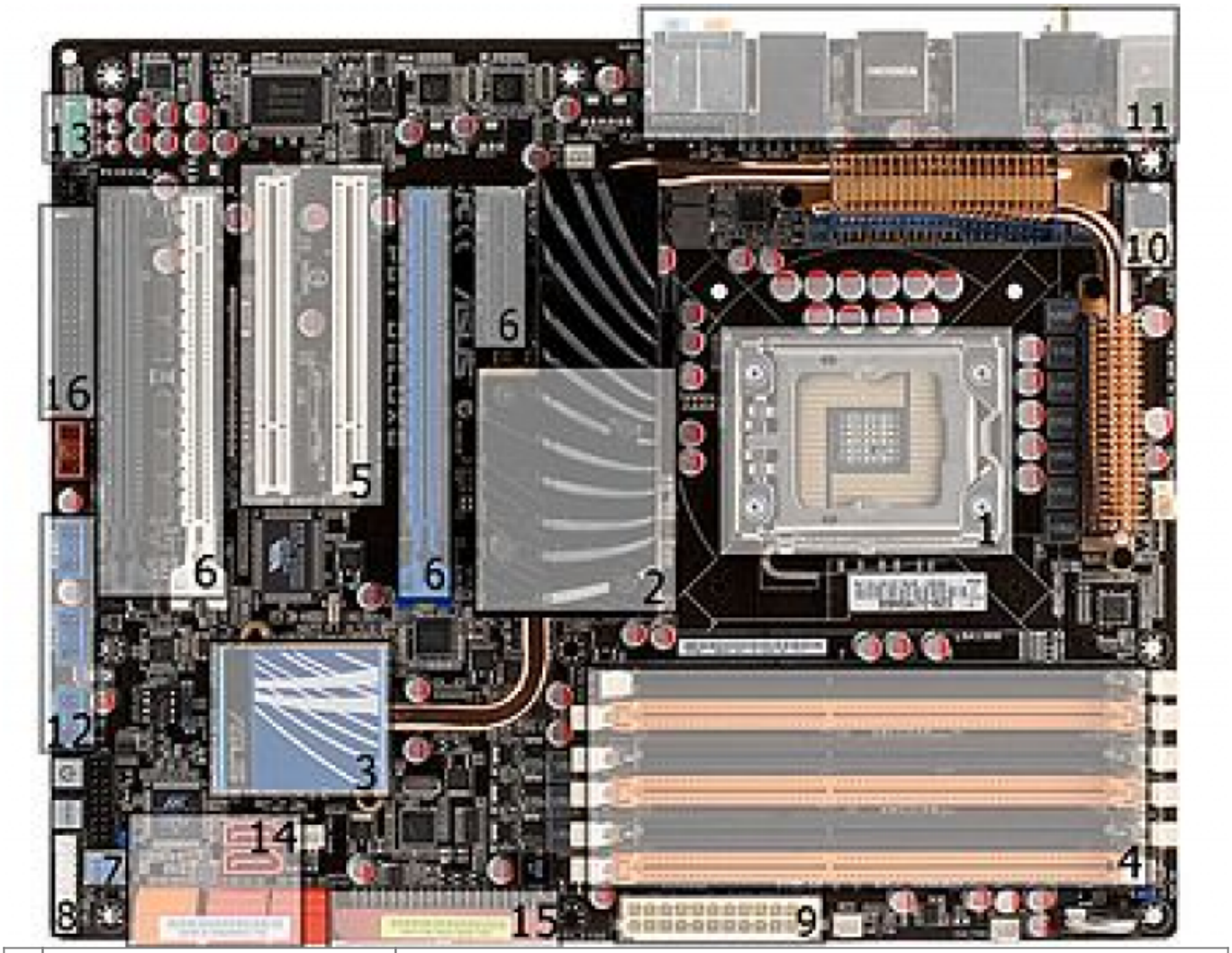
Bauformen

Das Format der Hauptplatinen wird nach dem **Formfaktor** unterschieden. Seit 1995 ist das **ATX-Format aktuell**. Es löste das AT-Format ab und brachte diverse Umstellungen an Gehäusen und Netzteilen mit sich. Es existieren diverse Variationen von ATX und AT, um auch kompaktere Geräte ohne proprietäre Formate bestücken zu können, etwa Baby AT oder µATX. Mini-ITX, Flex-ATX und Micro-ATX passen in ATX-Gehäuse. **Steckplätze, Schrauben** und das **Fenster für I/O-Shield** befinden sich an einheitlichen oder **derselben Position**.



Komponenten eines Motherboards

Auf einem Motherboard können je nach Bauform, Ausstattung und Integrationsdichte unterschiedliche Systemkomponenten zu finden sein.



Nr	Komponente	Beschreibung
1	**Prozessorsockel**	Der Prozessor, der auch als Hauptprozessor bezeichnet wird, hat auf dem Motherboard einen eigenen Steckplatz bzw. Sockel (z.B.: LGA775 oder LGA1155).
2	**Chipsatz - Northbridge**	Der Chipsatz ist das Bindeglied zwischen den einzelnen Systemkomponenten eines Computers. Egal was in einem Computer passiert, der Chipsatz hat immer damit zu tun. Er sorgt dafür, dass alle Komponenten über unterschiedliche Schnittstellen miteinander kommunizieren können. Dabei werden unterschiedliche Spannungspegel, Taktfrequenzen und Protokolle berücksichtigt und untereinander umgewandelt.
3	**Chipsatz - Southbridge**	
4	**RAM-Steckplatz**	Steckplätze für den RAM (z.B.: hier für DDR3)
5	**PCI-Steckplatz**	für Erweiterungskarten (z.B.: Audiokarte)
6	**PCI-Express-Steckplatz**	2 PCIe-x16 Slots (Grafikkarte) & 1 PCIe-x1-Slot (Erweiterungskarten)
7	**Jumper**	Kurzschlussstecker zur Aktivierung und Deaktivierung von Einstellungen (z.B. Übertakten)
8	**Anschlüsse Frontblende**	Hier werden die Schalter (Power, Reset, Lampen für FP) von vorne an das Mainboard angebunden
9	**24-poliger ATX-Connector**	Haupstromversorgung für das Mainboard
10	**8-poliger-ATX-Connector**	Anschluss für die Stromversorgung der CPU
11	**Externe Anschlüsse**	Anschlüsse ragen aus dem Gehäuse nach hinten hinaus und dienen als Anschluss für die Peripheriegeräte (Maus, Tastatur,..)
12	**Interne USB-Anschlüsse**	USB-Anschlüsse für die Vorderseite des Gehäuses

Nr	Komponente	Beschreibung
13	**AAFP-Soundanschluss**	An diesem Anschluss werden die Sound Ein- und Ausgänge der Front des Gehäuses angeschlossen
14	**Serial-ATA-Anschlüsse**	Ermöglicht die Verbindung von Festplatten oder optischen Laufwerken mit der Hauptplatine
15	**IDE-Anschluss**	Ermöglicht die Verbindung von Festplatten oder optischen Laufwerken mit der Hauptplatine
16	**Floppy-Disk-Stecker**	Anschluss des Diskettenlaufwerks

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_01

Last update: **2019/05/14 20:28**

6.3.2) CPU - Central Processing Unit

Der Prozessor, Hauptprozessor oder die CPU ist heutzutage das **Herzstück eines jeden elektronischen Geräts**. Er wird in Smartphones, Taschenrechnern und in Computern, für die er eigentlich erfunden wurde, eingesetzt. Eine Welt ohne diese Rechengenies ist undenkbar.
Die bekanntesten Prozessoren stammen von den Unternehmen **Intel und AMD**.



In einem **Computersystem** kann es **mehrere Prozessoren** geben. Wenn man vom Prozessor spricht, dann ist damit in der Regel immer der Hauptprozessor gemeint. Wegen seiner **zentralen Stellung** wird die Bezeichnung **„Zentrale Verarbeitungseinheit“** verwendet.
Der Hauptprozessor ist die **Funktionseinheit** in einem Computer, der **die eigentliche Verarbeitungsleistung erbringt**. Der Hauptprozessor ist für die **Informationsverarbeitung und die Steuerung der Verarbeitungsabläufe zuständig**. Dazu holt sich der Prozessor **aus dem Speicher nacheinander die Befehle** und **veranlasst die Informationsverarbeitung**.
Neben dem Hauptprozessor gibt es noch **weitere Prozessoren**, die den Hauptprozessor von der Arbeit entlasten. Der **Grafikprozessor (GPU)** ist ein solcher Prozessor.

Prozessor-Hersteller

Es gibt 2 marktführende Prozessore Hersteller, **AMD** und **INTEL**.
Seit jeher gilt der Prozessor-Hersteller AMD als ewiger Zweiter im Kampf um die Marktanteile im PC-Prozessor-Markt. Dazu kommt, dass AMD lange Zeit von dem am stärksten wachsenden Markt, den Notebooks, abgekoppelt war. Mit wenigen Ausnahmen, waren die meisten AMD-Prozessoren in Low-Cost-PCs für Privatkunden verbaut. In den Bereichen, in denen richtig Geld verdient werden konnte, war Intel immer besser unterwegs.

Das Problem von AMD und INTEL ist, dass die Prozessoren von den beiden Herstellern jeweils komplett andere Sockel gebrauchen und somit ein Umstieg von einem Hersteller zum anderen auch meistens mit einem neuen Motherboard-Kauf einhergeht.

Mikroprozessor

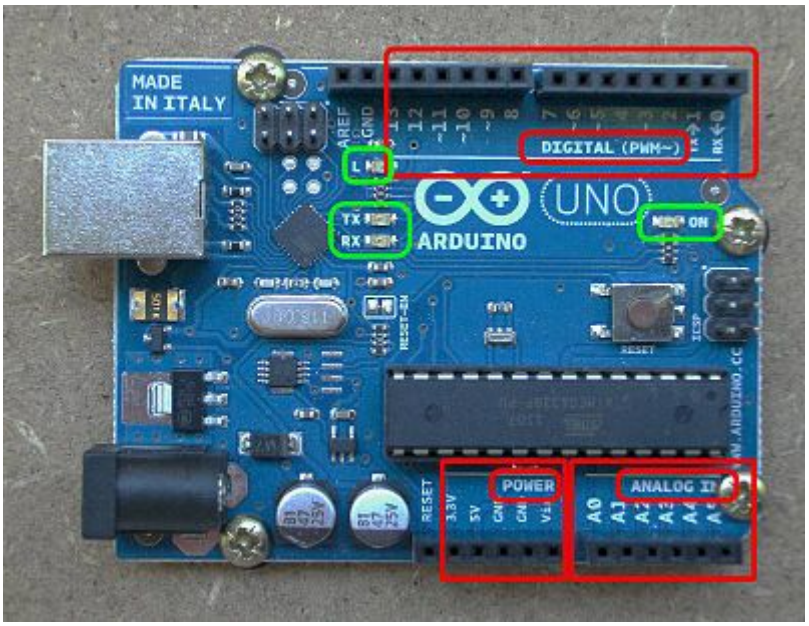
Der Mikroprozessor ist ein Prozessor, der vollständig in einem einzigen Schaltkreis untergebracht ist. Der Prozessor im Personal Computer ist ein solcher Mikroprozessor.



⇒ Ein **Mikroprozessor ist die Vereinigung von Rechenwerk und Steuerwerk eines Computers mit den dazugehörigen Daten-, Adress- und Steuerleitungen auf einem Chip**

Mikrocontroller

Ein Mikrocontroller ist ein **Prozessor**, der über **zusätzliche digitale Ein- und Ausgänge** verfügt und für Steuerungsaufgaben vorgesehen ist. Er wird bereits **als vollständiger Computer** angesehen, der im Embedded-Bereich eingeordnet wird.



Moderne Prozessortechnik

Die **Entwicklung der Prozessoren** verlief lange Zeit nach einem scheinbar einfachen Gesetz. Man **optimierte den internen Aufbau**, **verkleinerte die Strukturen**, **senkte die Spannung**, **erhöhte die Taktfrequenz** oder verbesserte den Herstellungsprozess.
Schon war die **nächste Prozessor-Generation** geboren. Doch diesem Spiel sind enge Grenzen gesetzt. So muss mit **zunehmender Chipgröße das Taktsignal immer längere Wege** zurücklegen. Damit der Zeitunterschied der Taktflanken im akzeptablen Bereich liegt, muss der **Takttreiber immer leistungsstärker** werden. Dadurch **erhöht er die Verlustleistung** des Prozessors. Deshalb wird die **Chipfläche regelmäßig verkleinert**. Dabei werden zwischen den Schaltelementen **immer dünnere Verbindungen** eingesetzt. Dadurch **steigt der Widerstand der Verbindungen** und die Signale bewegen sich immer langsamer. Das führt unter Umständen dazu, dass die Signallaufzeit unter der Verarbeitungszeit der Gatter liegt. Die **physikalischen Grenzen zeigten sehr schnell**, dass **insbesondere die Taktfrequenz nicht unendlich weit gesteigert werden konnte**. Die Taktfrequenz bestimmt unter anderem die entstehende Verlustleistung und damit **die Lebensdauer des Prozessors**. Deshalb wurden sehr bald andere leistungssteigernde Techniken entwickelt.

Während vor Jahren die Geschwindigkeit eines Prozessors immens wichtig war, ist heute ein **ausgewogenes System aus Prozessor, Arbeitsspeicher und Chipsatz** das Maß **für einen schnellen Computer**. Immer weniger Anwendungen benötigen die volle Rechenleistung eines aktuellen Prozessors. Aus diesem Grund bieten die heutigen Prozessoren viel mehr als nur reine Rechengeschwindigkeit. Sie haben **mehrere Kerne, nutzen Befehlssatzerweiterungen, intelligente Zwischenspeicher, verfügen über Virtualisierungstechnik und Grafikfunktionen**.

Multi-Core-Prozessoren

Multi-Core bedeutet, dass **in einem Prozessor mehrere Prozessor-Kerne** eingebaut sind. Man bezeichnet diese Prozessoren als Multi-Core- oder Mehrkern-Prozessoren. Rein **äußerlich unterscheiden sich** Multi-Core-CPUs **nicht von Single-Core-CPUs**. Der Rechenkern ist bei Multi-Core-CPUs einfach mehrfach vorhanden. Innerhalb des **Betriebssystems** wird der **Multi-Core-Prozessor wie mehrere Recheneinheiten** behandelt.

Je nach Anzahl der Kerne gibt es abgewandelte Bezeichnungen, die darauf hindeuten, wie viele Kerne im Prozessor integriert sind.

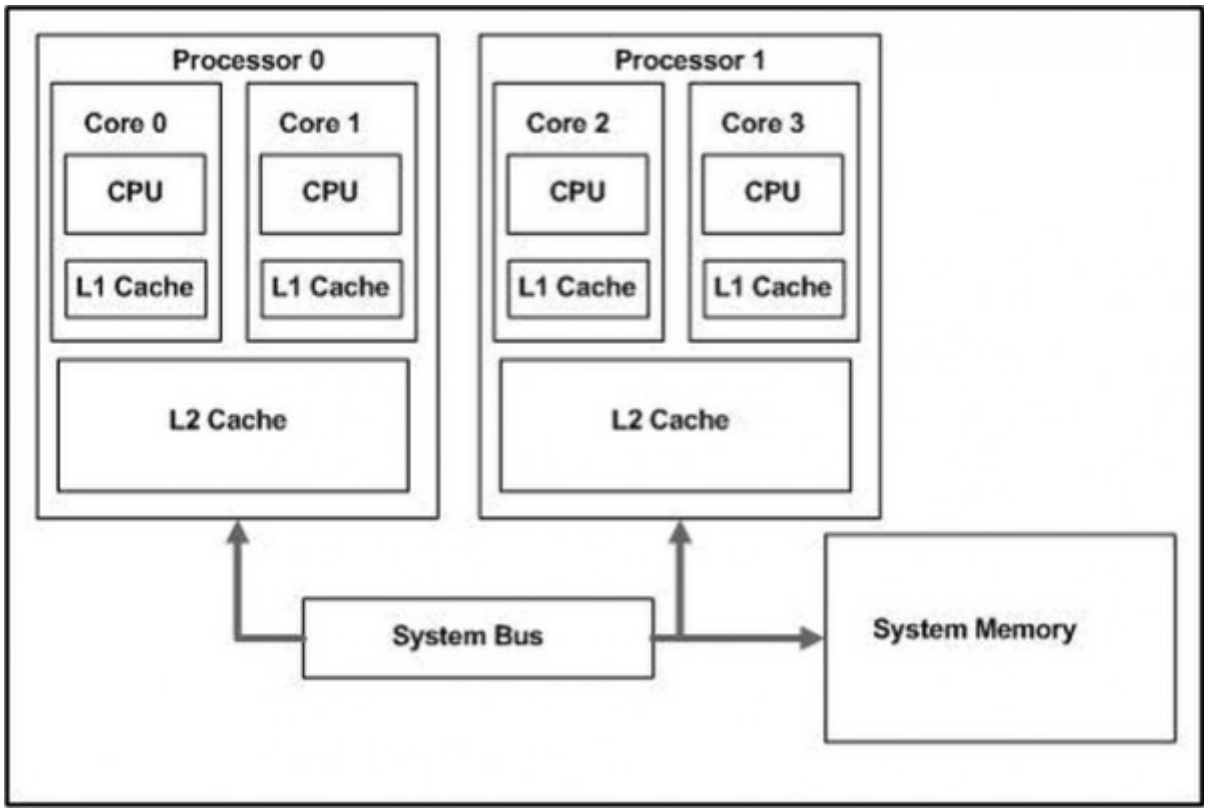
- Dual-Core (2)
- Triple-Core (3)
- Quad-Core (4)
- Hexa-Core (6)
- Octa-Core (8)
- ....

Unterschied Mehrprozessorsysteme vs. Mehrkernprozessor

Ein Mehrprozessorsystem ist ein Computersystem, das **mehr als einen Prozessorsockel** auf der

Multi-Prozessor-Hauptplatine besitzt, und in dem mehr als einer dieser Sockel auch bestückt ist.

Ein Mehrkernprozessor ist ein Prozessor mit mehreren Kernen.



In der obigen Abbildung ist erkennbar, dass dies 2 Prozessoren mit jeweils 2 Kernen sind. Also für das Betriebssystem insgesamt 4 Kerne zur Verfügung stehen!

Vom Takt-orientierten Prozessor zum Mehr-Kern-Prozessor

Um einen Prozessor schneller zu machen war die **Taktfrequenz lange Zeit der maßgebliche Faktor**, um **mehr Rechenleistung** aus einem Prozessor heraus zu bekommen. Leider hat die **Erhöhung der Taktfrequenz auch Nachteile**, die zu Folgeproblemen führen, die nur sehr schwer zu lösen sind.

- höhere Leistungsaufnahme
- höhere Wärmeentwicklung
- umfangreichere Kühlmaßnahmen
- lautere Computer durch aktive Kühlung

Schon eine **minimale Leistungssteigerung führt zu einem dramatisch höheren Energieverbrauch**. Die damit verbundene Leistungsaufnahme verhält sich proportional zur Taktfrequenz. Zusätzliche Transistoren und kleinere Halbleiterstrukturen erhöhen zusätzlich die Wärmeentwicklung. Die **Anforderungen an die Kühlung** sind mit den herkömmlichen Mitteln **nicht mehr zu leisten**. Gleichzeitig **verringert sich die Stabilität und Lebensdauer des Prozessors**.

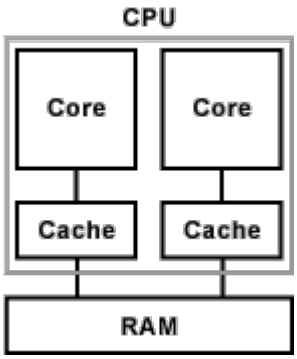
Eine **Alternative** zu immer höheren Taktraten sind **mehrere Rechenkerne**. Das bedeutet **mehr Leistung bei gleichzeitig geringerer Leistungsaufnahme**. Dabei werden die **einzelnen Kerne weit geringer getaktet**, als ein einziger Rechenkern. Insgesamt **steigt jedoch die**

Leistungsfähigkeit des Prozessors mit jedem weiteren Rechenkern.

Problematik

Grundsätzlich kann man die **Rechenleistung mehrerer Kerne nicht „addieren“**. Das würde voraussetzen, dass sich die vorliegenden Rechenaufgaben parallelisieren lassen. Die effektive Nutzung mehrerer Kerne erfordert es, dass die **Software so geschrieben ist, dass sie Daten parallel verarbeitet und so zum Beispiel mehrere Kerne gleichzeitig genutzt werden**. Das heißt, ein Problem bzw. die **Ausführung eines Programms** muss **in mehrere kleine Teilaufgaben zerlegt werden**, damit diese auf mehrere Kerne verteilt werden können. Doch leider sind die üblichen Anwendungen nicht auf parallele Ausführung ausgelegt und meist auch nicht notwendig.

Auswirkungen auf die Prozessor- und Computer-Architektur



In einer **Multi-Core-Architektur müssen** sich mehrere Rechenkerne die **vorhandenen Resourcen teilen**. Zum Beispiel **Arbeitsspeicher, Schnittstellen** usw. **Je mehr Kerne** ein Prozessor hat, **desto mehr Speicher und mehr Bandbreite zum Speicher ist erforderlich**. Aus diesem Grund ist der **Speicher-Controller nicht mehr im Chipsatz, sondern im Prozessor verankert**.

Arbeitsspeicher / Hauptspeicher

Gleichzeitig besteht das Problem, dass sich die **Rechenkerne darüber abstimmen** müssen, **wer gerade welche Daten im Cache** hält. In der Regel arbeiten die Rechenkerne mit einer **hierarchischen Cache-Struktur**. Dabei bekommt **jeder Kern einen eigenen L1- und L2-Cache**. Den **L3-Cache müssen sich die Kerne teilen**. Ein Cache-Kohärenz-Protokoll sorgt dafür, dass sich die Prozessorkerne bei der Nutzung des L3-Caches nicht in die Quere kommen.

Typische Anwendungen für Single-Core-Prozessoren

- Textverarbeitung
- Browser
- E-Mail
- Instant-Messaging
- Virenscanner
- MP3-Player

- einfache Bildbearbeitung (Schneiden, Skalieren, Farbkorrektur)

Typische Anwendungen für Multi-Core-Prozessoren

Gut parallelisierbare Anwendungen sind Schneiden von 4K-Videos, Computergrafiken mit Raytracing erzeugen oder aufwendige Softwareprojekte kompiliert. In diesen Fällen können im Prozessor nicht genug Kerne enthalten sein.

- CAD
- Simulation
- HD-Video
- Compiler
- 3D-Rendering
- professionelle Audio-Bearbeitung
- professionelle Bildbearbeitung
- Videoschnittprogramme

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_02

Last update: **2019/05/14 20:29**

6.3.4.1) Magnetische Massenspeicher

Der magnetische Datenspeicher besteht aus einem **Datenträger**, der aus einem **magnetisierbaren Material** besteht. Das kann auf **Bänder, Karten, Papier oder Platten** aufgebracht werden. Der Datenträger wird dann mittels einem **Lese-Schreib-Kopf gelesen und beschrieben**.
Typischerweise wird das Speichermedium im **Datenträger gedreht oder rotiert**, während der **bewegliche oder unbewegliche Lese-Schreib-Kopf über das Speichermedium geführt** wird.
Die Daten können auf dem Speichermedium sowohl **digital als auch analog** gespeichert werden.

- 6.3.4.1.1) Festplatten
- 6.3.4.1.2) Bandlaufwerke

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_01

Last update: **2019/05/14 20:33**

6.3.4.1.1) Festplatten - HDD (Hard Disk Drive)

Festplatten sind **Massenspeicher**, die auf einem **magnetischen Datenträger** beruhen, auf dem die **Daten fest gespeichert** werden und **auch ohne Energieversorgung gespeichert** bleiben. Festplatten werden typischerweise in einen Computer eingebaut. Auf Festplatten werden alle Daten und Anwendungen eines Computers gespeichert.

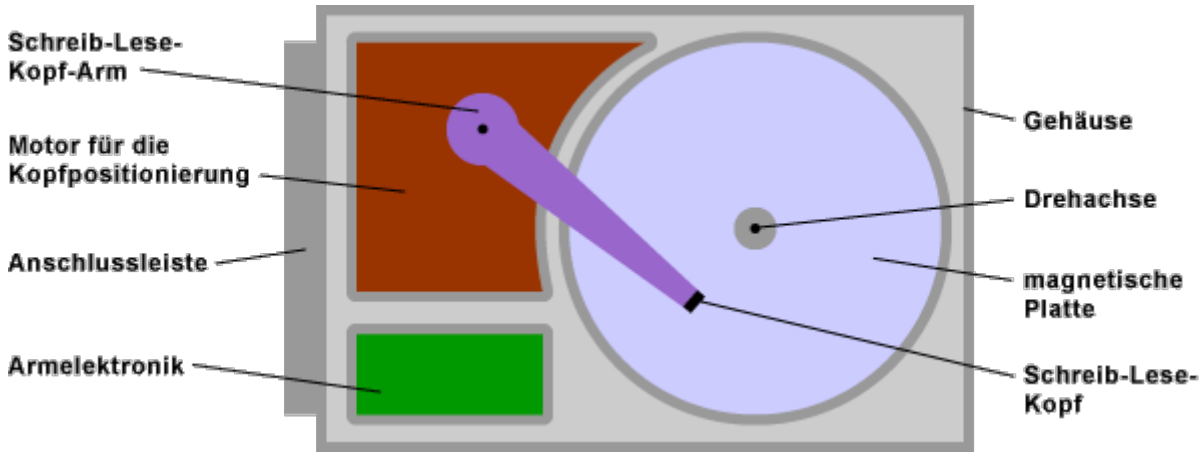




Der Begriff **Festplatte (engl. Harddisk bzw. Hard Density Disc, HDD)** kommt durch die Unterscheidung zur inzwischen veraltete Diskette (engl. Floppydisk, FDD), die als wechselbarer Datenträger lange Zeit vor der Festplatte verwendet wurde.
Festplatten **ersetzen den Festwertspeicher ROM und die Diskette**. Festplatten können **viel mehr Daten speichern als Disketten**. Im Gegensatz zum ROM kann man Daten von einer Festplatte nicht nur lesen, sondern auch darauf schreiben und jederzeit ändern. 1954 wurden Festplatten erstmals industriell eingesetzt.

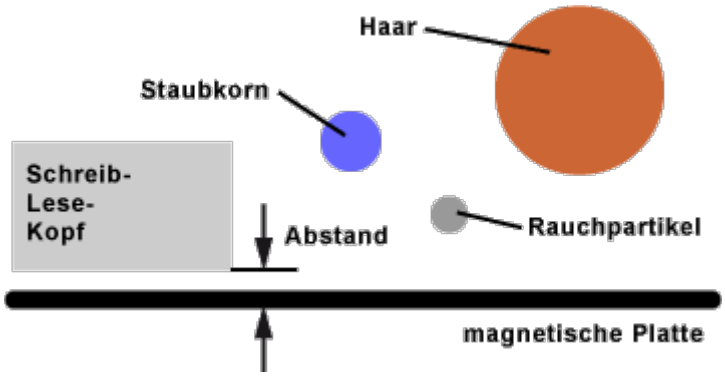
Festplatten unterscheiden sich in ihrer **Zuverlässigkeit, Geschwindigkeit, Speicherkapazität** und damit auch im Preis. Für unterschiedliche Anwendungsfälle haben die Festplatten-Hersteller **unterschiedliche Festplatten-Typen**. Im Vergleich zu anderen Datenspeichern haben Festplatten ein sehr gutes Preis-Leistungsverhältnis. Das bedeutet, **geringe Kosten pro Byte**. Momentan liegt die **größtmögliche Speicherkapazität bei 6 bis 8 TByte pro Einzellaufwerk**. Wobei es diese Festplatten nur für spezielle Anwendungen gibt. Bis **2020** soll die **Speicherkapazität auf 20 TByte** pro Einzellaufwerk steigen.

Aufbau einer Festplatte



In einem **geschlossenen Metallgehäuse** befinden sich alle Komponenten, die für das Funktionieren der Festplatte wichtig sind. Um das **Eindringen von Staub und Schmutz** in das Gehäuse zu

verhindern, sind die einzelnen Teile einer Festplatte in ein **nahezu luftdichtes Gehäuse** verschlossen. Als einziger Kontakt zum Computersystem dient eine **Anschlussleiste für eine Schnittstelle (IDE, SATA, SCSI, etc.)**, über die die Daten übertragen werden. Der eigentliche **Datenspeicher** einer Festplatte ist eine oder mehrere **Metallscheiben**, die mit einem **magnetisierbaren Material beschichtet** sind. Um die Speichermenge zu erhöhen liegen mehrere Scheiben übereinander. Die Scheiben sind um eine Drehachse mittels Halteklammern befestigt und dadurch voneinander getrennt. **Zwischen den Metallscheiben greifen die Schreib-Lese-Kopf-Arme** hinein. Auf diesen Armen befindet sich eine federnde Aufhängung. Auf dieser ist der Kopf befestigt, der zum Lesen und Schreiben der Daten dient.



Der **Abstand zwischen Kopf und Scheibe** ist **geringer, als ein Haar, Staub- oder Rauchpartikel**. Die **Berührung von Kopf und Scheibe führt zum Head-Crash**, der wiederum zum **Datenverlust** führt. Dabei wird der Datenträger zerstört, was die Festplatte unbrauchbar macht. Normalerweise können sich Kopf und Platte nicht berühren. Denn bei hohen Rotationsgeschwindigkeiten, bei der sich eine Festplatte dreht, bildet sich ein Luftpolster zwischen Kopf und Platte.

Geschwindigkeit einer Festplatte

Je schneller eine Festplatte ist, desto flüssiger laden die Daten und laufen die Programme. Besonders beim Start von Betriebssystem und Anwendungen spürt der Anwender eine schnelle Festplatte. Und nur mit einem **schnellen Massenspeicher speichert ein Computer große Datenmengen** mühelos. Folgende **vier Kriterien** machen eine Festplatte schnell:

1) Umdrehungsgeschwindigkeit

Die Umdrehungsgeschwindigkeit wird in Umdrehungen in der Minute (UPM, U/Min) angegeben. Je geringer die Drehzahl, desto länger dauert der Zugriff auf zufällig ausgewählte Sektoren. Üblich sind die Umdrehungsgeschwindigkeiten 10.000, 7.200 und 5.400 U/Min.

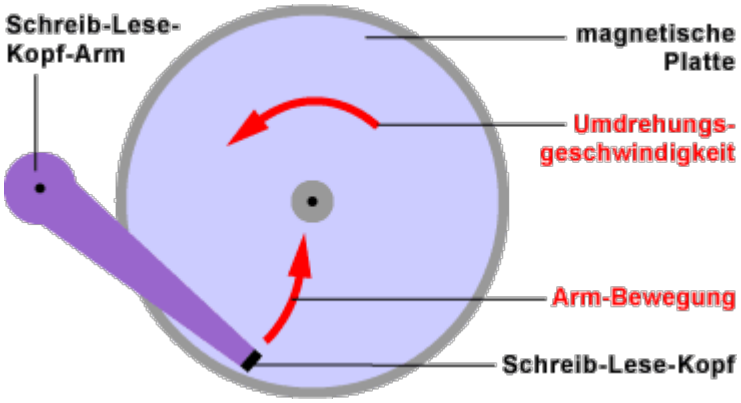
2) Anzahl der Datenscheiben

Je mehr Datenscheiben eine Festplatte hat, desto höher ist ihre Kapazität. Doch auch die Lese- und Schreibgeschwindigkeit steigt, wenn der Datenstrom über mehrere Lese- und Schreibköpfe summiert wird.

3) Datendichte auf den Datenscheiben

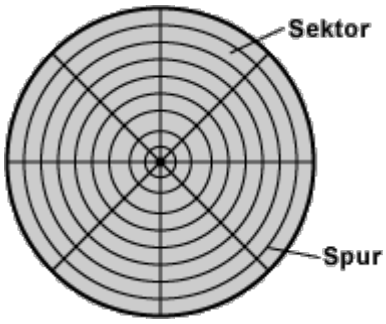
Je höher die Datendichte, desto mehr Bit ziehen pro Sekunde am Schreib-Lese-Kopf vorbei und können gelesen oder geschrieben werden.

4) Zugriffszeit / Access Time / Datenzugriffszeit



Die Zugriffszeit gibt an, wie lange es dauert, bis die Festplatte die gewünschten Daten auf ihren Datenschreiben gefunden hat und die ersten Bit liefert. Die Zugriffszeit ist die Summe aus Such- und Latenzzeit und wird hauptsächlich von der Umdrehungsgeschwindigkeit der Festplatte bestimmt. Je schneller sich die Platte dreht, desto geringer ist diese Zeit.

Organisation der Daten



Damit die Daten, die auf den magnetischen Platten abgelegt sind, wieder gefunden werden, ist es notwendig eine **Einteilung der Magnetscheiben** vorzunehmen. Als erster Schritt wird eine herstellerseitige Low-Level-Formatierung vorgenommen. Dazu werden auf den Scheiben **Spuren** angelegt. Es handeln sich dabei um **konzentrische Kreise**, die auf allen Magnetscheiben gleich sind. Die Spuren werden vom **äußeren Rand der Platte nach innen**, beginnend bei 0, **durchnummeriert**. Der Abstand der Spuren, die Spurdichte, bestimmt die Speichermenge. Diese Dichte wird in Spuren pro Zoll (Tracks per Inch, TPI) angegeben.
Die **Anordnung mehrerer Spuren** (durch **übereinander gelagerte Magnetscheiben**) nennt man **Zylinder**. Die **Spuren** werden wiederum in **kleinere Abschnitte** eingeteilt. Dieser Abschnitt nennt sich Sektor und entspricht einem **Kreisausschnitt**.

Spuren

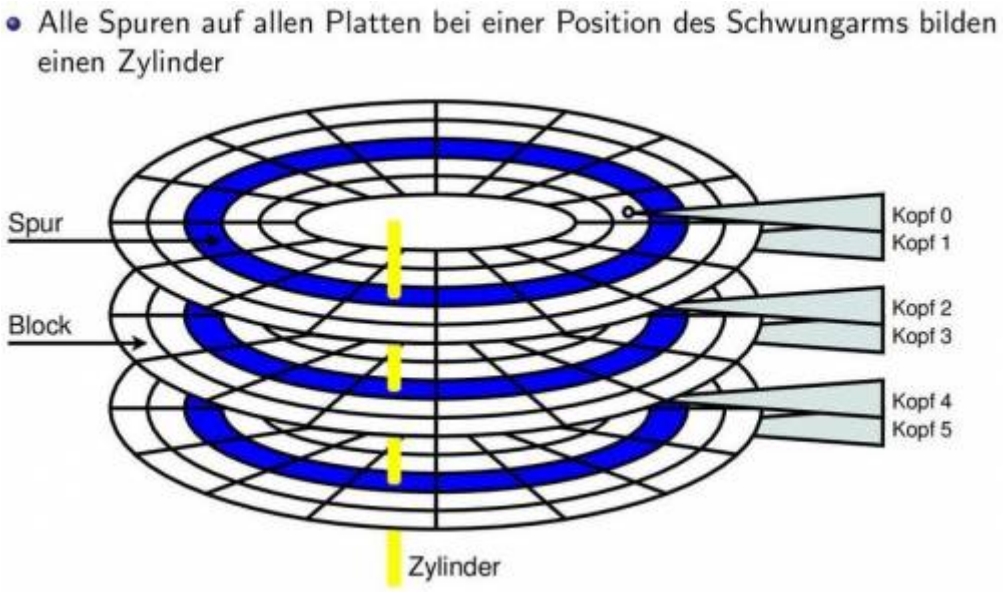
Die **Spuren** sind konzentrische Kreispfade, die auf jede Scheibenseite geschrieben werden, wie auf einer Schallplatte oder einer CD. Die Spuren werden mit einer Nummer identifiziert, die mit Spur 0 am äußeren Rand einsetzt.

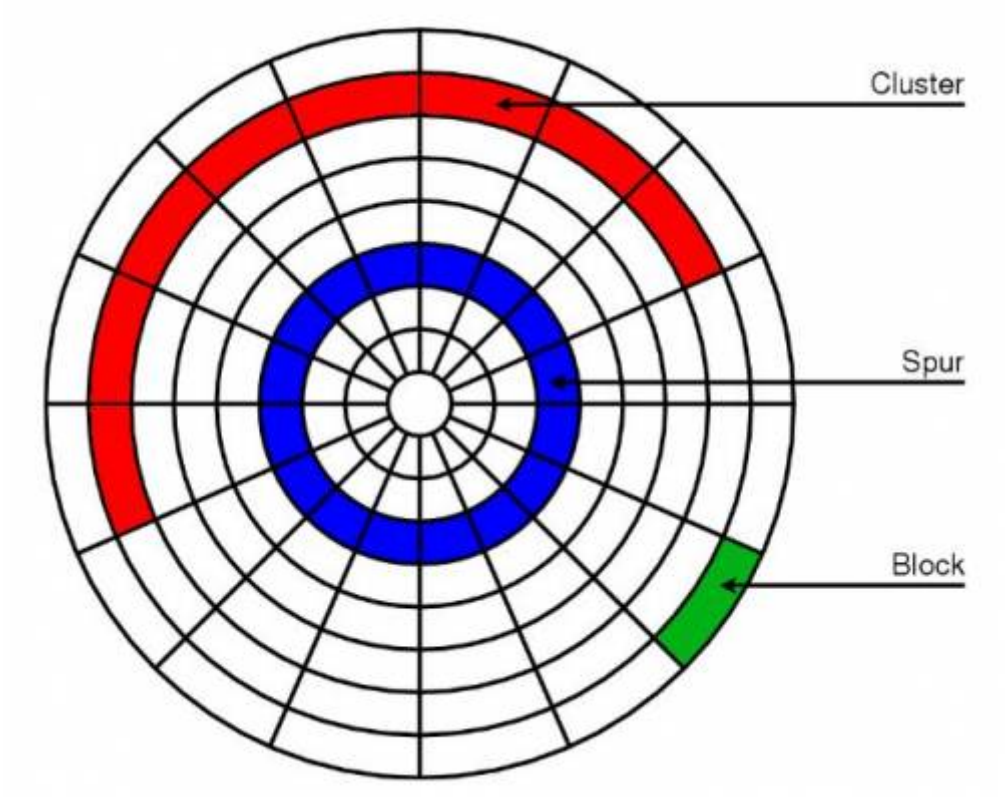
Zylinder

Der Spurensatz, der auf allen Seiten der Platte im gleichen Abstand von der Mitte angelegt ist, wird als "**Zylinder**" bezeichnet. Hardware und Software arbeiten häufig mit diesen Zylindern.

Sektoren

Die Spuren sind in Bereiche, sogenannte "**Sektoren**" oder „**Blöcke**" unterteilt, in denen eine festgeschriebene Datenmenge gespeichert wird. Die Sektoren werden in der Regel für eine Speicherkapazität von 512 Byte (ein Byte besteht aus 8 Bit) formatiert.





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_01:06_03_04_01_01

Last update: **2019/05/14 20:33**

4.3.4.1.2) Bandlaufwerke / Tapes

Streamer sind Bandlaufwerke zum Beschreiben und Lesen von Tapes, Magnetbändern oder einfach nur Bändern.

Die Bezeichnung Streamer wird vom Aufzeichnungsverfahren abgeleitet. Die Aufzeichnung findet linear statt, weshalb sie sich nur für das Sichern bzw. Speichern zusammenhängender Daten eignen. Zum Beispiel bei einer Datensicherung. Da Backups archiviert werden, ist die Datenmengen bis zu 10 mal größer als die eigentliche Datenmenge. Magnetbänder bzw. Tapes sind das einzige Speichermedium, das große Datenmenge vernünftig (Zeit, Aufwand, Kosten) sichern kann. Da es sich bei den Tapes um Wechselspeicher handelt, benötigt das gelagerte Speichermedium keinen Strom. Obwohl optische Speichermedien, wie CD, DVD und Blu-ray in der Computertechnik als Massenspeicher populär sind, hat das an der Bedeutung von Tapes und Bändern als externe Speichermedien im professionellen Bereich nichts geändert.





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_01:06_03_04_01_02

Last update: **2019/05/14 20:34**

4.3.4.2) Optischer Datenspeicher

Optische Datenspeicher **speichern Daten und Informationen optisch veränderter Form**. Bei der optischen Speicherung **nutzt** man die **Reflexions- und Beugungseigenschaften des Speichermediums** aus. Zum Lesen und Schreiben der Daten auf den Datenträger wird ein **Laserstrahl** verwendet. Die **Speicherform ist ausschließlich digital**.

- 6.3.4.2.1) CD-ROM
- 6.3.4.2.2) DVD
- 6.3.4.2.3) BD
- 6.3.4.2.4) UHD-BD

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_02

Last update: **2019/05/14 20:34**

6.3.4.2.1) CD-ROM

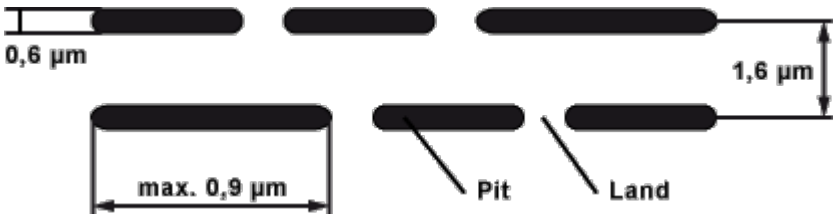
Die CD-ROM ist ein Massenspeicher mit **650 bis 879 MByte** Speicherkapazität. Als Vorlage diente die **Compact-Disc** (CD), die bereits Anfang der 80er Jahre als digitales Medium für Musik entwickelt wurde. Im aufkommenden Multimedia-Zeitalter wurde es nötig die umfangreichen Computerdaten sinnvoll zu speichern. Dazu reichte die **alte Diskette mit 1,4 MByte** nicht mehr aus.





Aufbau einer CD-ROM

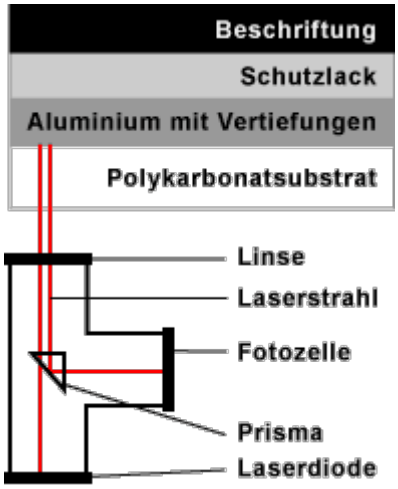
Der **Durchmesser** einer CD-ROM beträgt **12 cm**. Die CD besteht aus **Polycarbonat** mit einer Dicke von **1,2 mm**. Darin liegt eine **reflektierende Aluminiumschicht**. Sie ist **einseitig mit Daten beschrieben**.



Die Daten sind auf einer **spiralförmigen, von innen nach außen, führenden Spur in Vertiefungen gespeichert**. Diese **Vertiefungen werden Pits (Täler)** genannt. Sie sind 0,2 µm tief, 0,6 µm breit und maximal 0,9 µm lang. Der Abstand zwischen den Spuren beträgt 1,6 µm. Die **Zwischenräume zwischen den Pits werden Lands** genannt.

Die Daten auf einer CD-ROM werden in **gleich großen Sektoren** gespeichert. Diese Sektoren werden auf einer einzigen Spur, die spiralförmig angelegt ist, untergebracht.

Lesen der Daten



Die **Laserdiode erzeugt einen Laserstrahl**, der zusätzlich durch eine **Linse gebündelten** wird. Dieser Laserstrahl trifft auf die Unterseite der CD-ROM, **durchdringt die Schutzschicht** und **trifft auf die Aluminium-Schicht**. Dort wird er **reflektiert**. Das **Prisma leitet** den **Laserstrahl zur Fotozelle** weiter. Der Laserstrahl, der die Fotozelle trifft, erzeugt eine **geringe elektrische Spannung**.
Trifft der Laserstrahl auf einen **Übergang zwischen Pit und Land**, dann wird der **Laserstrahl abgelenkt**, was eine **andere elektrische Spannung** in der Fotozelle verursacht. Auf diese Weise entsteht eine **Reihe von Einsen und Nullen.**

Lesegeschwindigkeit

Eine Audio-CD wird mit einer konstanten Datenrate von 150 kByte/s ausgelesen. Dieser Datendurchsatz wird mit „1x“ bezeichnet. Laufwerke, die einen 52-fachen Datendurchsatz haben, erreichen 7800 kByte/s.

Umdrehungs-geschwindigkeit	max. Übertragungsrate	mittlere Zugriffszeit
1x	150 kByte/s	400 ms
2x	300 kByte/s	300 ms
4x	600 kByte/s	150 ms
8x	1200 kByte/s	100 ms
12x	1800 kByte/s	100 ms
16x	2400 kByte/s	90 ms
24x	3600 kByte/s	90 ms
32x	4800 kByte/s	85 ms
44x	6600 kByte/s	85 ms
52x	7800 kByte/s	-

Fehler

Jede CD-ROM erhält im Laufe der Zeit und durch unsachgemäße Behandlung fehlerhafte Stellen (Kratzer, Risse). Diese Stellen bringen ein CD-ROM-Laufwerk zum häufigen Wiederholen des Lesevorgangs mit reduzierter Geschwindigkeit. Arbeitet die Fehlerkorrektur gut, sind diese Vorgänge seltener. Wobei sich das positiv auf die Lesegeschwindigkeit auswirkt.

Formate

Format	Beschreibung
CD+G	Compact Disc and Graphics für Audio und Grafik mit einer sehr geringen Verbreitung.
CD+Midi	Compact Disc mit digitaler Schnittstelle für Musikinstrumente für Audio- und Midi-Informationen mit sehr geringer Verbreitung.
CD-DA	Compact Disc Digital Audio ist die klassische Musik-CD mit 74 Minuten Spieldauer.
CD-Extra	Enhanced Music Compact Disc ist die verbesserte Musik-CD. Nachfolger der CD-Plus für Audio und Daten.
CD-I	Compact Disc Interactive ist die Multimedia-CD mit der Möglichkeit in den Ablauf aktiv einzugreifen (interaktiv) für Audio, Video, Fotos und Spiele.
CD-MO	Magnetooptische Daten-CD mit geringer Verbreitung.
CD-R	Compact Disc Recordable mit der Möglichkeit einmal mit Audio oder Daten zu beschreiben.
CD-RW	Compact Disc Rewriteable mit der Möglichkeit mehrfach mit Audio oder Daten zu beschreiben.
CD-ROM	Compact Disc Read Only Memory ist ein Nurlesespeicher für Daten.
CD-ROM XA	CD-ROM eXtended Architecture ist ein Nurlesespeicher für Daten, Text, Grafik, Bild, Audio und Video.
Photo-CD	Foto-CD für die Ablage von digitalen Bildern in Echtfarben.
Video-CD	Video-CD für Filme mit max. 74 Minuten Spieldauer im Audio- und Videoformat MPEG. Videoqualität im VHS-Format.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

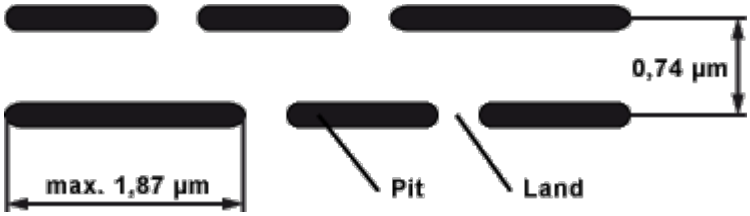
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_02:06_03_04_02_01

Last update: **2019/05/14 20:34**

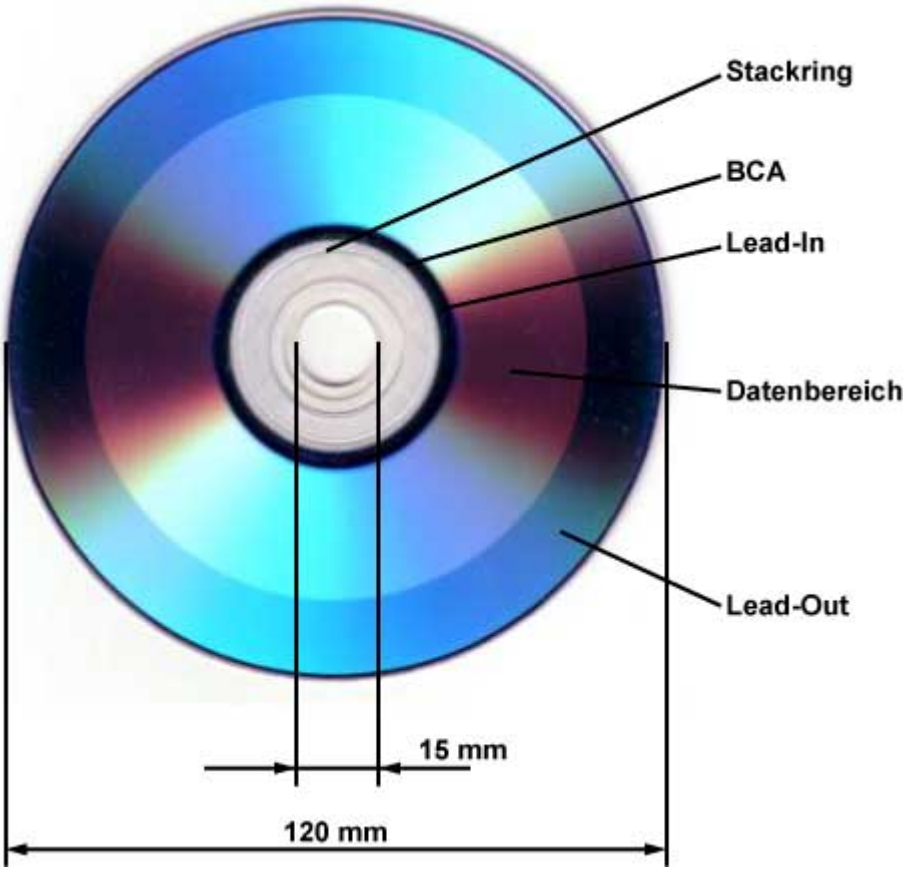
6.3.4.2.2) DVD - Digital Versatile Disc

Die DVD ist ein Massenspeicher, ähnlich wie die CD-ROM. Mit einer Speicherkapazität von **4,7 GByte (Single Layer) bzw. 8,5 GByte (Dual Layer)** konnten viel mehr Daten gespeichert werden, als auf einer CD-ROM. Parallel dazu wurden **beschreibbare und wiederbeschreibbare DVD-Varianten** entwickelt. Ursprünglich stand DVD für Digital Video Disc. Ein Großteil der Anwendungen liegen jedoch im Computerbereich.

Aufbau einer DVD



Die Daten auf einer DVD werden in **Erhöhungen (Pits) und Vertiefungen (Lands)** in der Reflektionsschicht gespeichert. Der **Übergang von einem Pit zu einem Land** bzw. umgekehrt wird als **logische 1** gewertet. Findet **kein Wechsel** statt, wird das als **logische 0** gewertet. Um mehr Daten auf ein gleich großes Medium wie die CD unterzubringen, werden die Pits und deren **Abstand zueinander verkürzt** und die Daten auf **vier übereinanderliegende Schichten** verteilt. Dadurch entsteht ein Speichermedium, das bis zu **17 GByte** speichern kann.
Um die kleineren Pits lesen zu können, ist die **Wellenlänge des roten Lasers auf 635 nm bzw. 650 nm** verkürzt. Zum Vergleich: die **CD-ROM** wird mit einem **infraroten Laser und einer Wellenlänge von 780 mm** gelesen.



Speicherkapazität

Die Speicherkapazitäten einer DVD sind eine **Vorgabe der Filmindustrie**. Diese geht von einer Standardlänge pro Film von 135 Minuten aus. Durch Addieren der Datenrate pro Sekunde kommt man mit Bild und Ton in mehreren Sprachen und Untertitel auf 4,69 MBit. In der Minute sind das 281,4 MBit. Bei 135 Minuten ergibt das eine Datenmenge von ca. 38 GBit, was umgerechnet etwa **4,7 GByte** entspricht. Das ist die Speicherkapazität einer **DVD-5 (Single Layer)**. Diese wird für Video-DVDs verwendet. Wenn die Speicherkapazität nicht ausreicht, dann greift man auf die **doppellagige DVD-9 (Dual-Layer)** zurück.

Typen

DVD-Typ	Seiten	Schichten	Speicherkapazität		Besonderheit
			binär	dezimal	
DVD-5	1	1	4,38 GibiByte	4,7 GByte	-
DVD-9	1	2	7,92 GibiByte	8,5 GByte	-
DVD-10	2	1	8,76 GibiByte	9,4 GByte	-
DVD-14	2	1/2	12,3 GibiByte	13,2 GByte	-
DVD-18	2	2	15,84 GibiByte	17 GByte	-
DVD-plus	2	1	4,28 GibiByte	4,7 GByte	DVD-5/CD-ROM
DVD-1	1	1		1,5 GByte	8 cm Durchmesser
DVD-2	1	2		2,7 GByte	8 cm Durchmesser
DVD-3	2	1		2,9 GByte	8 cm Durchmesser
DVD-4	2	2		5,3 GByte	8 cm Durchmesser

Formate

Formate	Beschreibung
DVD-Video	Mehrstündige Videos mit hochqualitativen, interaktiven Videosequenzen mit mehreren Soundspuren und Untertiteln. Die Video-Daten werden mit MPEG2 komprimiert.
DVD-Audio	Musik mit 24-Bit-Auflösung und 96 kHz Sampling-Frequenz.
DVD-ROM	Speichermedium für Computer, ähnlich der CD-ROM.
DVD-R	Einmalbeschreibbare DVD wie die CD-R.
DVD-R DL	Einmalbeschreibbare DVD mit zwei Aufnahmeschichten, wie bei einer DVD-9
DVD-RW	Wiederbeschreibbare DVD des DVD-Forums, ähnlich der CD-RW.
DVD+R	Einmalbeschreibbare DVD und Konkurrenz zur DVD-R.
DVD+R DL	Einmalbeschreibbare DVD mit zwei Aufnahmeschichten, wie bei einer DVD-9
DVD+RW	Wiederbeschreibbare DVD und Konkurrenz zur DVD-RW.
DVD-RAM	Wiederbeschreibbare DVD ähnlich wie DVD-RW und DVD+RW.
Recordable CSS	Filme mit dem „DVD Download“-Siegel lassen sich auf spezielle „DVD-R for CSS Managed Recording“ brennen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_02:06_03_04_02_02

Last update: **2019/05/14 20:34**

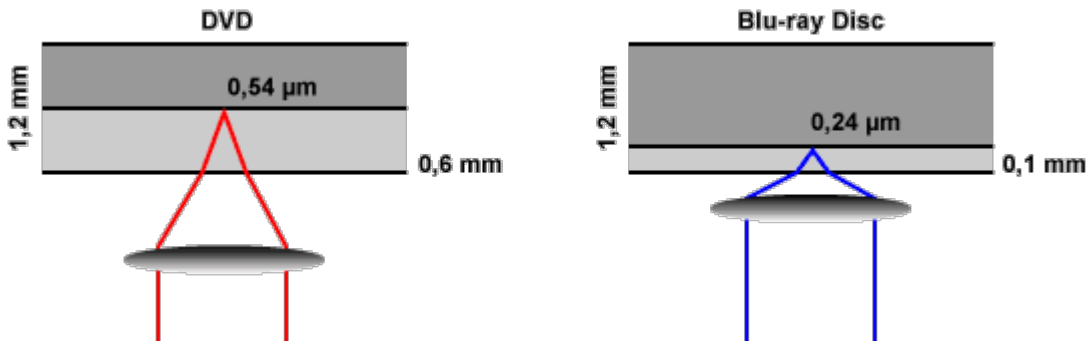
6.3.4.2.3) BD - Blu-ray Disc

Die Blu-ray Disc (BD) wurde für die **Speicherung von hochauflösenden Videodaten (HDTV)** entwickelt. Die Speicherung von **mehr als zwei Stunden Audio- und Videodaten** ist das maßgebliche Kriterium bei der Entwicklung gewesen. Die Speicherkapazität einer **DVD reicht für diese Datenmenge nicht aus**.

Die **Bezeichnung** Blu-ray Disc ist von der **Farbe des Lasers abgeleitet**. Weil sich eine Farbe nicht als Warenzeichen schützen lässt, hat man einfach das „e“ aus dem Wort „Blue“ weggelassen. Deshalb findet man ab und zu fälschlicherweise die Bezeichnung „Blue-ray Disc“.

Aufbau der Blu-ray Disc

Mit Hilfe eines blauen Lasers mit einer **Wellenlänge von 405 nm können 25 GByte** auf einer einlagigen, **einseitig beschreibbaren Blu-ray Disc** gespeichert werden. **Zweilagige Blu-ray Discs** haben sogar die **doppelte Kapazität (50 GByte)**. **Denkbar wären Medien mit 8 Schichten, also 200 GByte**. Allerdings wäre die Produktion dieser Medien sehr schwierig.



Um auf dem gleichen Medium, wie die CD bzw. DVD, eine höhere Speicherkapazität möglich zu machen, wurde die **Struktur auf der Blu-ray Disc verkleinert**. Dazu wird ein **Laser im blauvioletten Bereich** verwendet und die Wellenlänge auf **405 nm** abgesenkt. Blaues Licht lässt sich feiner fokussieren und ermöglicht so eine **höhere Datendichte auf der Scheibe**. Dadurch muss aber auch der **Abstand zwischen Scheibe und Schreib-/Leseoptik verringert** werden.

Formate

Format	Speicherkapazität
BD-ROM SL	25 GByte
BD-ROM DL	50 GByte
BD-R (Recordable)	25 GByte
BD-RE SL (Rewriteable)	25 GByte
BD-RE DL (Rewriteable)	50 GByte
BDXL	100 und 128 GByte (40 MByte/s)

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_02:06_03_04_02_03

Last update: **2019/05/14 20:35**

6.3.4.2.4) UHD-BD - Ultra-HD Blu-ray Disc



Die Ultra-HD Blu-ray Disc, kurz UHD-BD, ist ein optisches Speichermedium, dass auf der BDXL-Disc der Blu-ray Disc Association basiert. Allerdings sind UHD-BD und BDXL nicht vollständig identisch, weshalb nicht alle BDXL-Brenner UHD-BDs abspielen können. Wie gewohnt hat die Scheibe einen **Durchmesser von 12 cm**. Eine UHD-BD hat **zwei Schichten auf die zusammen 66 GByte** passen. Auf einer zweiten Variante mit **drei Schichten finden rund 100 GByte** Platz.

Die Ultra-HD Blu-ray dient **zum Speichern von 4K-Filmen und Abspielen auf TV-Geräten mit Ultra-HD-Auflösung (3.840 × 2.160 Pixel)**. Die entsprechenden UHD-BD-Hüllen sind schwarz und mit einem silbernen Logo versehen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_02:06_03_04_02_04

Last update: **2019/05/14 20:35**

4.3.4.3) Elektronischer Datenspeicher (Halbleiterspeicher)

Halbleiterspeicher dient sowohl der **zeitlich begrenzten** als auch der **unbegrenzten Aufbewahrung von Daten, Zuständen und Programmen** in **Form von digitalen Signalen**.

Elektronische Datenspeicher fassen **Halbleiterbauelemente, vorwiegend Transistoren und Kondensatoren, zu integrierten Schaltkreisen zusammen**, um Daten und Informationen zu speichern. Dazu werden **spezielle nicht-leitende Materialien**, wie **zum Beispiel Silizium, gezielt verunreinigt**, um sie **unter bestimmten Bedingungen in einen leitenden oder nicht-leitenden Zustand zu versetzen**. Die Zustände leitend und nicht-leitend können dabei die beiden binären Zuständen „0“ und „1“ abbilden.

- 6.3.4.3.1 ROM - Read Only Memory
- 6.3.4.3.2 RAM - Random Access Memory
- 6.3.4.3.3 Flash-Speicher

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03

Last update: **2019/05/14 20:35**

4.3.4.3.1 ROM - Read Only Memory

Das ROM ist ein **digitaler Festwertspeicher bzw. permanenter Halbleiterspeicher**, in dem **Daten dauerhaft und unveränderlich gespeichert werden**. Der **Speicherinhalt kann also NICHT neu programmiert oder beschrieben werden**. Typischerweise enthalten permanente Halbleiterspeicher **Betriebssysteme, Anwendungsprogramme und Firmware** an denen **während des Betriebs keine Änderungen** vorgenommen werden müssen. In der Regel werden ROM- und PROM-Bausteine nicht fest die Platinen gelötet, sondern werden in Sockel gesteckt und sind somit austauschbar.
Die Herstellung von ROM-Bausteinen war schon immer **relativ teuer**. Wenn überhaupt **lohnt sich das nur bei Massenprodukten**.

Heute dienen **Flash-Speicher als ROM- und PROM-Ersatz**. Vor allem deshalb, weil sich der **Speicherinhalt von Flash-Memory jederzeit**, auch während des Betriebs, **überschreiben** lässt. Speicher in der Form „Read Only Memory (ROM)“ sollte man nur noch als eine Form eines Datenspeichers sehen, der in der Praxis so nicht mehr vorkommt. Höchstens nur noch dort, wo man keine Änderung des Speicherinhalts wünscht. In der Praxis ist es jedoch wichtig, dass ein Halbleiterspeicher jederzeit veränderbar ist.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03:06_03_04_03_01

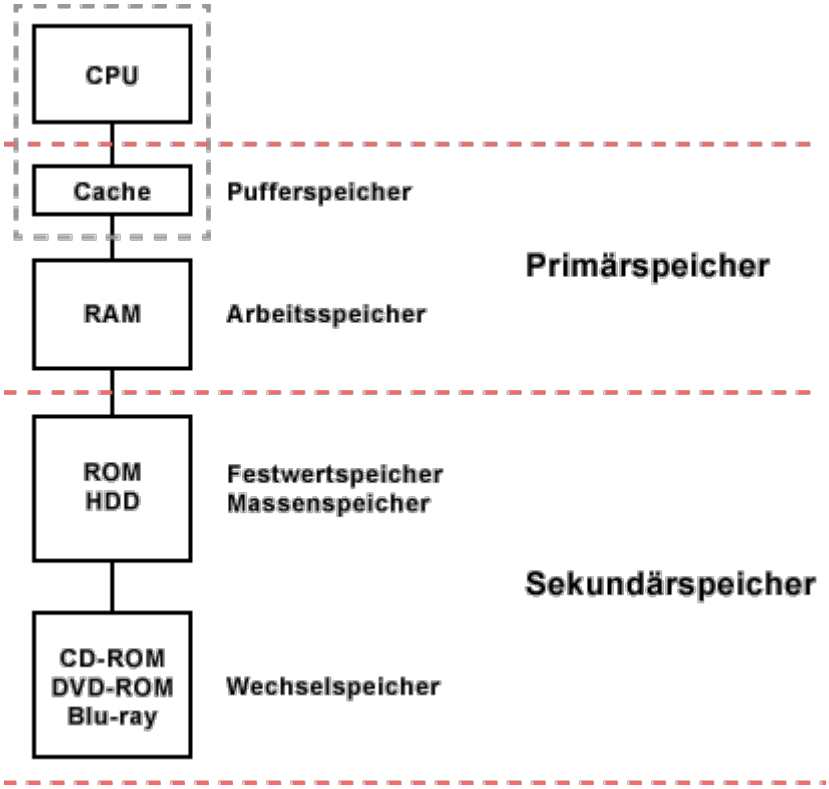
Last update: **2019/05/14 20:35**

6.3.4.3.2) RAM (Random Access Memory)

RAM bezeichnet einen **Speichertyp dessen Speicherzellen über ihre Speicheradressen direkt angesprochen** werden können. In diesem Zusammenhang wird auch von **„wahlfrei“** gesprochen, was sich auf **„random“** bezieht und **nichts mit „Zufall“ oder „zufälligem Zugriff“ zu tun** hat. In diesem Zusammenhang spricht man von **„Speicher mit wahlfreiem Zugriff“** oder „Direktzugriffsspeicher“. Auf **andere Speicherarten (zum Beispiel Flash) kann man nur blockweise zugreifen**. **RAM erlaubt den Zugriff auf jede einzelne Speicherzelle**. Bei ROM (Read-Only-Memory, Nur-Lese-Speicher) funktioniert das genauso. Bei RAM funktioniert es lesend, wie auch schreibend. **Doch wird die Stromversorgung abgeschaltet gehen die Daten im RAM verloren**.



RAM wird in **Computer- und Mikrocontroller-Systemen als Arbeitsspeicher** eingesetzt. Weil in der Regel nur RAM als Arbeitsspeicher verwendet wird, wird **RAM gerne als Abkürzung für Arbeitsspeicher** verwendet. In diesem **Arbeitsspeicher werden Programme und Daten von externen Speicherträgern und Festplatten geladen**. Zur schnellen Verarbeitung kann der Prozessor darauf zugreifen, verarbeiten und danach wieder in den Arbeitsspeicher schreiben.



Prinzipiell **unterscheidet** man zwischen **statischem RAM und dynamischem RAM**. SRAM und

DRAM sind flüchtige Halbleiterspeicher. Sie verlieren nach dem Ausschalten ihren Speicherinhalt.

- **SRAM - Static Random Access Memory** (Statisches RAM)
- **DRAM - Dynamic Random Access Memory** (Dynamisches RAM)

SRAM - Static Random Access Memory

SRAM ist ein statischer Halbleiterspeicher, was bedeutet, dass der **Speicherinhalt mittels Flip-Flops gespeichert wird und so nach dem Abruf des Speicherinhaltes erhalten bleibt**. Dadurch ist der **Stromverbrauch sehr hoch**, was aber zu einem **schnellen Arbeiten innerhalb des Speichers** führt. Aufgrund seines **hohen Preises** und des **großen Stromverbrauchs** wird SRAM **nur als Cache- oder Pufferspeicher** mit geringen Kapazitäten verwendet.

- Speicherung erfolgt in Flip-Flops
- sehr schnell
- kein Refresh nötig
- hoher Stromverbrauch
- Einsatz als L1-, L2- und L3-Cache

DRAM - Dynamic Random Access Memory

DRAM ist der **einfachste und billigste Speicher**. Im Computer-Bereich ist **DRAM als Arbeitsspeicher bzw. Hauptspeicher der bevorzugte Halbleiterspeicher**, den es in verschiedensten Varianten und Weiterentwicklungen gibt. Heute ist der SDRAM der am weitesten verbreitete Halbleiterspeicher in der Computertechnik. Im Gegensatz zum SRAM muss der **Speicherinhalt beim DRAM zyklisch aufgefrischt werden (Refresh)**. Dies ist normalerweise in Abständen von **einigen zig Millisekunden** erforderlich. Das Auffrischen des Speichers erfolgt zeilenweise.

- Kondensator als Speicherelement
- Speicherhaltung durch Refresh der Speicherzellen
- langsam
- geringer Stromverbrauch
- Einsatz als Arbeitsspeicher oder Hauptspeicher

Eine DRAM-Speicherzelle besteht aus einem Transistor und einem Kondensator (1T1C), der das eigentliche Speicherelement ist. In einer DRAM-Speicherzelle wird ein Bit durch die Ladung des Kondensators gespeichert. Die Messung der Spannung am Speicherkondensator (Lesen) und dessen anschließende Aufladung (Schreiben) benötigen eine gewisse Zeit.

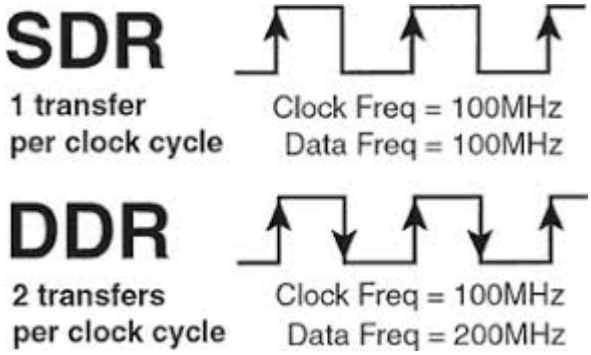
SDRAM - Synchronous DRAM

SDRAM ist der am **häufigsten verwendete Arbeitsspeicher** bzw. Hauptspeicher **in Computersysteme**n. SDRAM hat die Eigenschaft, dass er seine **Schreib- und Lesezugriffe am Systemtakt orientiert**. Das bedeutet, er arbeitet **synchron mit dem Speicherbus**. Woraus die Bezeichnung „Synchronous DRAM“ abgeleitet wird. Im Gegensatz dazu arbeitet normales DRAM asynchron. Die synchrone Arbeitsweise vereinfacht und beschleunigt die Ansteuerung des Speichers. SDRAM kann programmiert und so die Art des Zugriffs gesteuert werden. Auf diese Weise lässt sich

SDRAM an jede beliebige Anwendung anpassen.

DDR-SDRAM - Double Data Rate SDRAM (DDR1 / DDR2 / DDR3 / DDR4)

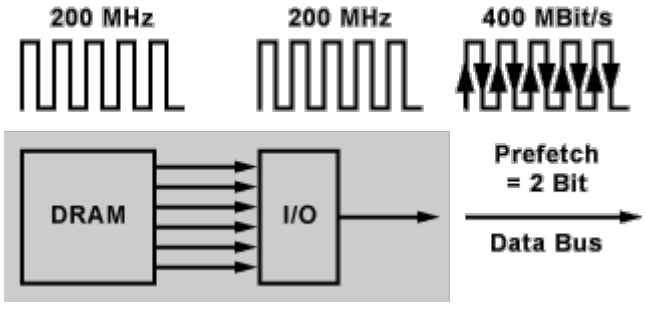
DDR-SDRAM und die Varianten DDR2, DDR3 und DDR4 sind gängige Halbleiterspeicher für Computersysteme. **DDR-SDRAM wird als Arbeitsspeicher bzw. Hauptspeicher** verwendet. Dieser Halbleiterspeicher wird nicht nur in Computern, sondern auch in Kraftfahrzeugen, Netzwerken, Kommunikationstechnik, medizinischen Apparaten und in der Unterhaltungselektronik eingesetzt.



DDR-SDRAM entspricht dem normalen SDRAM, jedoch mit einer kleinen Modifikation: Bei der Übertragung der Daten wird **nicht nur die ansteigende Flanke, sondern auch die abfallende Flanke des Taktsignals zur Datenübertragung genutzt**. In der Praxis entspricht das einer **Taktverdopplung**. Rein rechnerisch entsteht so eine **Verdopplung der Übertragungsrate**.

Double Data Rate bedeutet, dass **pro Übertragungszyklus (eine Taktrate) zweimal bzw. die doppelte Menge an Daten übertragen** wird. Damit die beteiligten Komponenten des Speicherbusses synchron arbeiten orientieren sie sich am Speichertakt und nutzen durch die DDR-Technik sowohl die steigende als auch die fallende Taktflanke. Normalerweise würde man nur die steigende Taktflanke nehmen. Ein Speicherbus, der mit 100 MHz arbeitet würde bei DDR rein rechnerisch mit 200 MHz arbeiten. Soviel zur grundsätzlichen Funktionsweise von Double Data Rate (DDR).

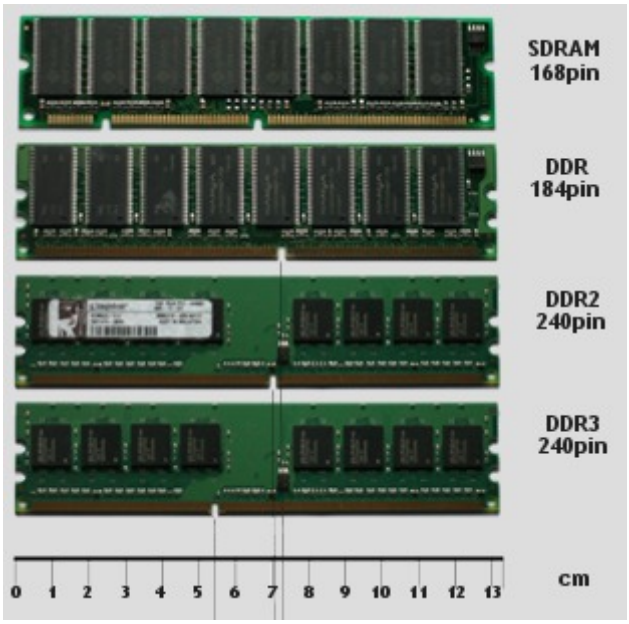
Eine weitere Erhöhung der Übertragungsrate wäre dann nur noch durch die Erhöhung der Taktrate möglich. Doch Vorsicht, man **unterscheidet zwischen dem internen Speicher-Takt in den Speicherchips und dem externen Speicherbus-Takt**. Das bedeutet, wenn der Speicherbus-Takt mit 100 MHz und durch DDR rechnerisch mit 200 MHz arbeitet, bedeutet das nur, dass sich die DDR-Technik als Taktverdopplung von 100 auf 200 MHz auswirkt. Denn intern arbeitet der Speicher nur mit 100 MHz. Double Data Rate bezieht sich hier nur auf den Speicherbus, nicht auf die Speicherchips.

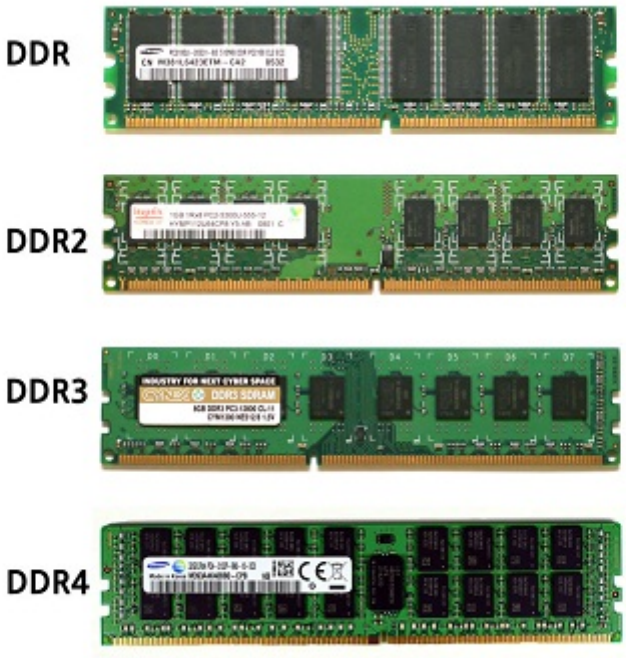


Das bedeutet, die Verdopplung der Übertragungsrate auf dem Speicherbus ist nur wenig sinnvoll,

wenn der eigentliche Speicher die Daten in dieser Geschwindigkeit nicht liefern kann. Die Frage ist, wie schafft man es, dass die Verdopplung der externen Datentransferrate auch wirklich genutzt werden kann. Das erreicht man nur durch **Prefetching** innerhalb des Speichers. Dazu werden **einfach zwei (DDR) oder mehr (DDR2, DDR3) Datenbits auf einmal ausgelesen**. Die Zugriffsbeschleunigung durch **Prefetching funktioniert** aber **nur dann**, wenn der **Speicher-Controller hintereinander liegende Adressbereiche aus der gleichen Speicherfeldzeile anfordert** oder wenn die **angeforderten Daten auf unterschiedlichen Speicherbänken** liegen.

DDR SDRAM standard	Release year	Bus clock (MHz)	Internal memory clock (MHz)	Prefetch (min burst)	Transfer rate (MT/s)	Voltage (V)	DIMM pins	SO-DIMM pins	MicroDIMM pins
DDR1	2000	100–200	100–200	2n	200–400	2.5/2.6	184	200	172
DDR2	2003	200–533.33	100–266.67	4n	400–1066.67	1.8	240	200	214
DDR3	2007	400–1066.67	100–266.67	8n	800–2133.33	1.5/1.35	240	204	214
DDR4	2014	800–1600	200-400	8n	1600–3200	1.2/1.05	288	256	—





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03:06_03_04_03_02

Last update: **2019/05/14 20:36**

6.3.4.3.3) Flash-Memory / Flash-Speicher



Flash-Speicher bzw. Flash-Memory **kombiniert die Vorteile von Halbleiterspeicher und Festplatten**. Wie jeder andere Halbleiterspeicher kommt Flash-Speicher **ohne bewegliche Teile** aus. Und die **Daten bleiben** wie bei einer Festplatte auch **nach dem Abschalten der Energieversorgung erhalten**.

Computer, deren Speicher rein auf Flash-Speicher basieren, sind der Traum eines jeden Software-Entwicklers und Anwenders. Der Computer müsste nie mehr minutenlang beim Starten booten, sondern wäre innerhalb weniger Sekunden sofort betriebsbereit. Genauso schnell wäre er auch ausgeschaltet.

Vorteile von Flash-Speicher

- Die gespeicherten Daten bleiben auch bei fehlender Versorgungsspannung erhalten. Auf eine Erhaltungsladung kann verzichtet werden. Somit ist auch der Energieverbrauch und die Wärmeentwicklung geringer.
- Wegen fehlender beweglicher Teile ist Flash geräuschlos, unempfindlich gegen Erschütterungen und magnetische Felder.
- Im Vergleich zu Festplatten haben Flash-Speicher eine sehr kurze Zugriffszeit. Lese- und Schreibgeschwindigkeit sind über den gesamten Speicherbereich weitestgehend konstant.
- Die erreichbare Speichergröße ist durch die einfache und platzsparende Anordnung der Speicherzellen nach oben offen.

Nachteile von Flash-Speicher

- begrenzte Schreib- bzw. Löschvorgänge
- begrenzte Speicherkapazität
- hoher Preis

Anwendungen von Flash-Speicher

Flash-Speicherchips sind allgegenwärtig. Sie stecken in vielen Geräte des täglichen Gebrauchs.

- 6.3.4.3.3.1) USB-Stick
- 6.3.4.3.3.2) Speicherkarten
- 6.3.4.3.3.3) SSD - Solid State Drive
- 6.3.4.3.3.4) SSHD - Solid State Hybrid Drives
- Handy
- MP3-Player

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

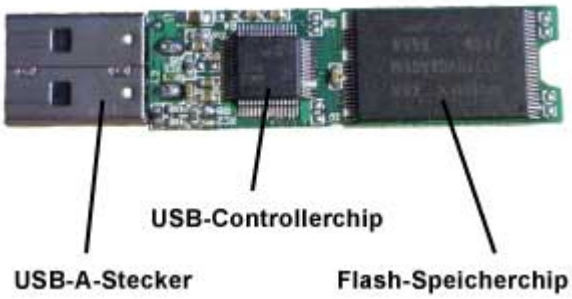
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03:06_03_04_03_03

Last update: **2019/05/14 20:36**

6.3.4.3.3.1 USB-Stick



Ein USB-Speicherstick bzw. USB-Stick ist ein **Wechselspeicher**, der direkt an einen **USB-Port eines Computers steckbar** ist. Der Datenspeicher auf einem USB-Stick besteht aus Flash-Halbleiterspeicher.



USB-Sticks zeichnen sich durch eine **hohe Speicherkapazität und Zugriffsgeschwindigkeit** aus. Dadurch sind sie eine Alternative zu Disketten, wiederbeschreibbaren CDs oder DVDs. Weil die USB-Schnittstelle eine **Standard-Schnittstelle** ist, die jeder Computer hat, eignen sich **USB-Sticks als Wechselspeicher**. Der Zugriff ist so einfach, wie bei einer Festplatte. Der USB-Stick ersetzt damit auch die beliebten beschreibbaren CDs und DVDs, wenn es um den Transport und die Archivierung großer Dateimengen geht.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03:06_03_04_03_03:06_03_04_03_03_01

Last update: **2019/05/14 20:36**

6.3.4.3.3.2) Speicherkarten



Speicherkarten sind **kleine Datenspeicher, die als Wechselspeicher in mobilen Geräten** eingesetzt werden. Speicherkarten verwenden Flash-Speicher als Speichertechnik. Die Flash-Speicherzellen behalten die Daten auch ohne Stromversorgung. Die **kleine und leichte Bauweise** erlaubt den Einsatz in kleinen mobilen Geräten. Auch der **Stromverbrauch hält sich in Grenzen**. Das ist **besonders bei Akku-betriebenen Geräten wichtig**. Zum Lesen und Beschreiben der Speicherkarten gibt es Lesegeräte in den unterschiedlichsten Ausführungen. Egal ob als Gehäuse-Einbaukit, als Karten-Adapter für USB oder fest eingebaut in Notebooks.

In der Regel werden Speicherkarten für **Digitalkameras und Handys** verwendet. Für Handys gibt es auch ganz kleine Speicherkarten, die nur so groß wie ein Fingernagel sind. Noch vor ein paar Jahren war die Auswahl an Speicherkarten sehr groß. Man musste zwischen CompactFlash, Memory Stick, Secure Digital, xD und MultiMediaCard wählen.

Übersicht

Bezeichnung	Abkürzung	Breite	Länge	Höhe	Kapazität	Bemerkung
CompactFlash I	CF	42,8 mm	34,4 mm	3,4 mm	bis 32 GByte	
CompactFlash II	CF	42,8 mm	34,4 mm	5 mm	bis 32 GByte	
Secure Digital High Capacity	SDHC	24 mm	32 mm	2,1 mm	bis 32 GByte	
SDXC	SDXC	24 mm	32 mm	2,1 mm	bis 2 TByte	
miniSD	miniSD	20 mm	21,5 mm	1,4 mm	bis 2 GByte	veraltet
miniSDHC	miniSDHC	20 mm	21,5 mm	1,4 mm	bis 8 GByte	veraltet
microSD	microSD	11 mm	15 mm	1 mm	bis 2 GByte	veraltet
microSDHC	microSDHC	11 mm	15 mm	1 mm	bis 16 GByte	
microSDXC	microSDXC	11 mm	15 mm	1 mm	bis 64 GByte	

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03:06_03_04_03_03:06_03_04_03_03_02

Last update: **2019/05/14 20:37**

6.3.4.3.3.3) SSD - Solid State Drive

Ein **Solid State Drive, kurz SSD, ist ein Massenspeicher**, vergleichbar mit einer Festplatte. Im **Gegensatz zur Festplatte hat eine SSD keine beweglichen Teile**. Bei der SSD ist das Speichermedium ein Flash-Speicher (z. B. NAND-Flash).
Flash-Speicher arbeiten vollkommen **geräuschlos**, hat **kurze Zugriffszeiten** und **schont den Akku, weil keine mechanischen Teile durch Motoren bewegt werden** müssen.
Die **SSD** wird **wie** eine **herkömmliche Festplatte angesprochen**. Um einen bestimmten Sektor zu lesen, muss eine Festplatte die Köpfe auf die richtige Spur bewegen. Diese Bewegung unterliegt einer gewissen mechanische Trägheit, die überwunden werden muss. Dann vergeht noch eine geringe Zeit, bis der gewünschte Sektor am Lesekopf vorbeidreht. Dieser ganze Zeitverlust fällt bei der SSD weg. SSDs zeichnen sich also durch eine **sehr hohe Lese- und Schreibgeschwindigkeit**, sowie einen **geringeren Energieverbrauch** aus.



Vorteile von SSD (Solid State Drive)

- hohe Transferraten
- kurze Zugriffszeiten, vor allem beim Lesen
- niedrige Leistungsaufnahme
- geräuschloser Betrieb

SSD-Schnittstellen

Wegen immer schnellerer Flash-Speicher und -Controller nimmt die **Geschwindigkeit von SSDs unaufhörlich** zu. Da SSDs als Festplatten-Ersatz dienen, ist die **SATA- bzw. SAS-Schnittstelle als Massenspeicher-Schnittstelle** hier maßgeblich im Einsatz. Im Vergleich zur Weiterentwicklung von SSDs bleibt die **Weiterentwicklung von SATA leider zurück**. Während es schon **SSDs** gibt, die **Daten mit 2 GByte/s** schaufeln können, hängt **SATA 6G bei 600 MByte/s** bzw. **SAS bei 1,2 GByte/s** fest. In der **Praxis können SATA-6G-Desktop-Festplatten** Daten linear mit **ca. 180 MByte/s lesen**. Sehr **schnelle Server-Festplatten erreichen ca. 250 MByte/s**. Mit der bisherigen Übertragungstechnik und den dazugehörigen Steckverbindern ist es leider nicht möglich, die Datenrate von SATA zu steigern.

SSD oder Festplatte?

Obwohl die SSD in den letzten Jahren sich immer mehr durchgesetzt hat, hat die klassische Festplatte immer noch ihre Daseinsberechtigung. Trotz der typischen Vorteile, wie hohe Geschwindigkeit bei

nicht-sequenziellen Zugriffen und niedrigen Energieverbrauch ist eine SSD nicht zwangsläufig der bessere Datenspeicher. Mit einem Blick auf **Speicherkapazität, Preis und Zuverlässigkeit spricht mehr für die herkömmliche Festplatte**.

Hohe Performance erreichen SSDs bisher **nur mit rechenstarken Controllern**, die ausgefeilte Wear-Leveling-Algorithmen beherrschen, viele Flash-Chips parallel anbinden und SDRAM-Cache nutzen. Eine **Zwischenlösung** zwischen Festplatten und SSDs sind **Hybrid-Festplatten (SSHD, Solid-State Hybrid Drives)**, die neben dem Plattenspeicher auch einen Chip-Speicher haben, der das Booten des Betriebssystems und den Start von Anwendungen beschleunigen kann.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03:06_03_04_03_03:06_03_04_03_03_03

Last update: **2019/05/14 20:37**

6.3.4.3.3.4) SSHD - Solid-State Hybrid Drives (Hybrid-Festplatten)

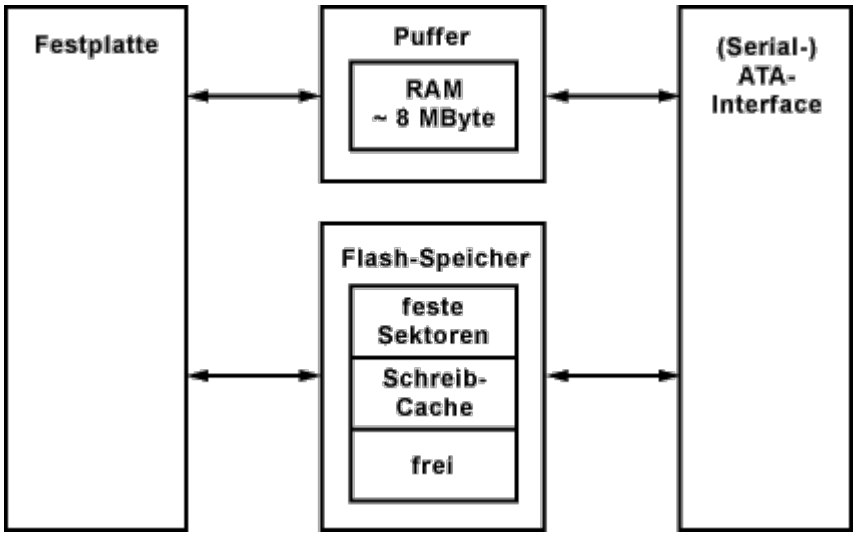


Solid-State Hybrid Drives, kurz SSHDs, sind **Hybrid-Festplatten**, die aus einer **herkömmlichen Festplatte mit einem zusätzlichen Flash-Speicher** bestehen. Der **Flash-Speicher dient als Datenpuffer für Schreib- und Lesezugriffe** und soll für **kürzere Zugriffszeiten** sorgen. Im Flash-Speicher speichert der Festplatten-Controller **häufig von den Magnetscheiben gelesene Daten** zusätzlich ab, so dass diese von dort mit SSD-Tempo gelesen werden können. Gleichzeitig werden Schreibvorgänge mit geringem Datenumfang in den Flash-Speicher und später am Stück auf die Magnetscheiben geschrieben. Das spart Strom, weil die Magnetscheiben in der Festplatte seltener anlaufen müssen.

Zusammenfassen kann man sagen, dass dieses Verfahren die **Leistungsaufnahme von Festplatten reduziert und die Lese- und Schreibgeschwindigkeit erhöht**. Hierbei gilt, **je größer der Flash-Speicher, desto besser funktioniert es**. Denkbar wäre, dass ein Computer vom Flash-Cache aus bootet oder aus dem Ruhezustand aufwacht. So könnte ein Computer in weniger als einer Sekunde betriebsbereit sein.

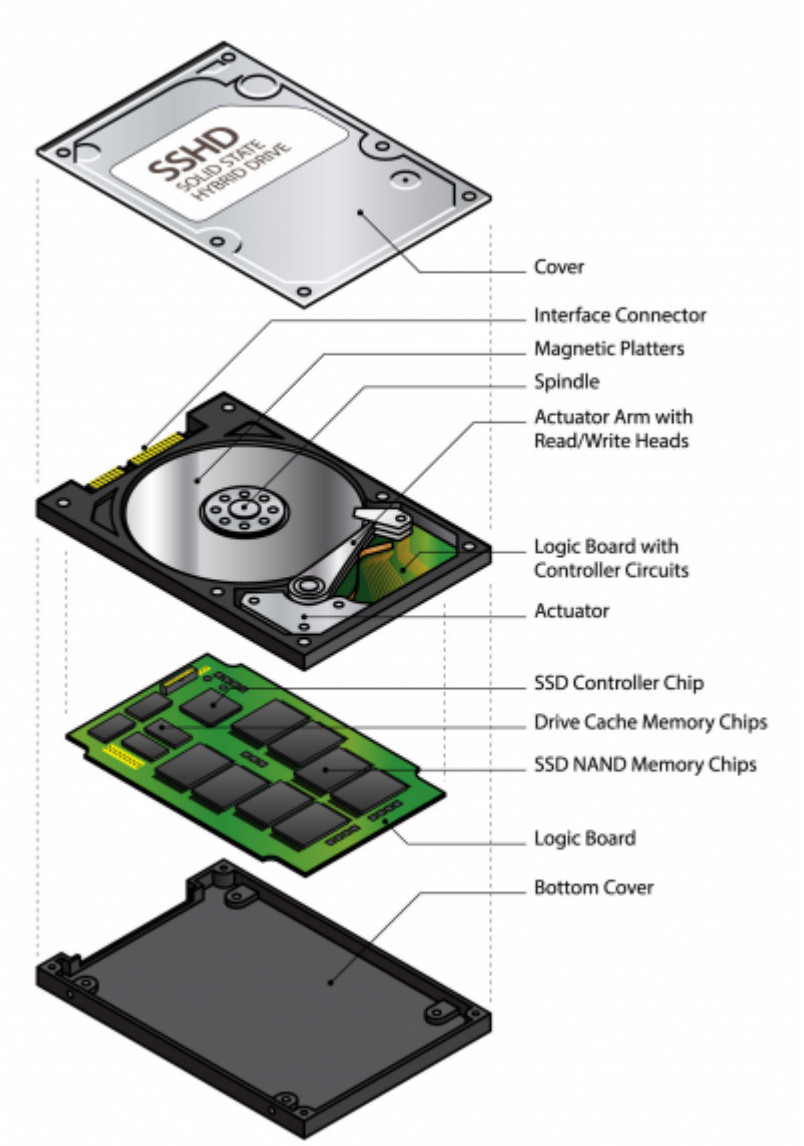
Beim Lesen und Schreiben arbeiten Hybrid-Festplatten rund **20% schneller**, wodurch die **Systemleistung aber nur um ca. 10%** steigt. Hinweis: Geschwindigkeitssteigerungen von unter 20% bemerkt der Anwender in der Regel kaum. **In der Regel lohnt sich der Einsatz von Hybrid-Festplatten nicht**.

Aufbau und Funktionsweise einer SSHD



Ein Teil des **Flash-Speichers** wird zum **Puffern von Schreibzugriffen** verwendet. Anstatt die Daten gleich auf die Hybrid-Festplatte zu schreiben, werden die **Daten im Flash-Speicher abgelegt**. Der

Speicher wird so lange beschrieben, **bis er voll ist**. In dieser Zeit bleibt das Laufwerk im Stromsparmodus. Erst dann, wenn der Flash-Speicher voll ist, wird das Laufwerk aufgeweckt und der **Inhalt des Flash-Speichers auf den Magnetspeicher** übertragen.
Auf diese Weise **beschleunigt der Flash-Speicher die Schreibzugriffe**. Vor allem die nichtlinearen Zugriffe, bei denen der Schreib-/Lese-Kopf der Festplatte sich mehrmals über die Platte bewegt. Das ist sehr zeitintensiv. Beim Schreiben auf den Flash-Speicher fallen die mechanischen Vorgänge erst mal weg. Auch beim Lesevorgang können Geschwindigkeitsvorteile entstehen.
Interessant sind **Hybrid-Festplatten vor allem in Notebooks**. Hier kommt es auf **geringe Leistungsaufnahme und schnelle Startzeiten** an. Es gibt ein spezielles ATA-Kommando, mit dem eine Hybrid-Festplatte angewiesen werden kann den Flash-Speicher zum Stromsparen zu verwenden. Dabei wird das Laufwerk möglichst oft schlafen gelegt und alle Schreibzugriffe erst mal im Flash-Speicher abgelegt.





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:06_03_04_03:06_03_04_03_03:06_03_04_03_03_04

Last update: **2019/05/14 20:37**

6.3.4) Datenspeicher

Ein Datenspeicher dient zum **Speichern von Daten bzw. Informationen**. **Als Datenspeicher dienen Datenträger**. Der Begriff Speichermedium wird auch als Synonym für einen konkreten Datenträger verwendet. Bei der Datenspeicherung geht es darum, Daten jeglicher Art auf Datenträger zu schreiben bzw. speichern und von dort auch wieder zu lesen.

Es wird unterschieden zwischen:

- 6.3.4.1) Magnetische Datenspeicher
- 6.3.4.2) Optischer Datenspeicher
- 6.3.4.3) Elektronischer Datenspeicher

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_04:start

Last update: **2019/05/14 20:32**

6.3.5) Erweiterungskarten

Erweiterungskarten sind, ganz grob ausgedruckt, nichts weiter als Teile der Hardware, die benötigt wird, um den Computer ohne Austausch von ganzen Komponenten (z.B. Mainboard) um mehr Leistung oder Funktionen zu erweitern.

Gängige Erweiterungskarten sind:

- 6.3.5.1) Grafikkarte
- 6.3.5.2) Soundkarte
- 6.3.5.3) Netzwerkkarte

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_05

Last update: **2019/05/14 20:45**

6.3.5.1) Grafikkarte

Die Grafikkarte **berechnet visuelle Daten** und **gibt sie über eine Schnittstelle aus**, an der ein Bildschirm angeschlossen ist. Auf dem Bildschirm wird dann die **Benutzeroberfläche von Betriebssystem und Anwendungsprogrammen dargestellt**.



Grafikkarten bzw. deren Funktionen **bestimmen maßgeblich die Systemleistung und Eigenschaften** eines Computers. Wobei die Leistung der Grafikkarte in der **Hauptsache für Computerspiele** und **umfangreiche Grafikdarstellungen** relevant ist. In **komplexen Spielszenen berechnet die Grafikkarte detaillierte Objekte und die richtige Beleuchtung**. Während der **Hauptprozessor allgemeine Berechnungen** erledigt und zum Beispiel das **Verhalten von Spielfiguren** berechnet, übernimmt die **Grafikkarte die Berechnung** beispielsweise von **physikalischen Phänomenen, Explosionen und einstürzenden Gebäuden**. Wenn die Bildschirmdarstellung bei diesen Bewegungen oder Verschiebungen ruckelt, dann ist die Grafikkarte, insbesondere der Grafikprozessor (GPU), bei der aktuell gewählten Auflösung und Detailtiefe nicht leistungsfähig genug.

Zunehmend wird die **Grafikverarbeitung** von der Grafikkarte **in den Hauptprozessor** verlagert. Das bedeutet jedoch nicht, dass die Zeit der Grafikkarten vorbei ist. Es ist eher so, dass die Grafikfunktionen, die bereits „onboard“ im Chipsatz integriert sind, gleich in den Prozessor zu integrieren. Konsequenterweise muss die Grafikausgabe dann auch vom Prozessor aus erfolgen.

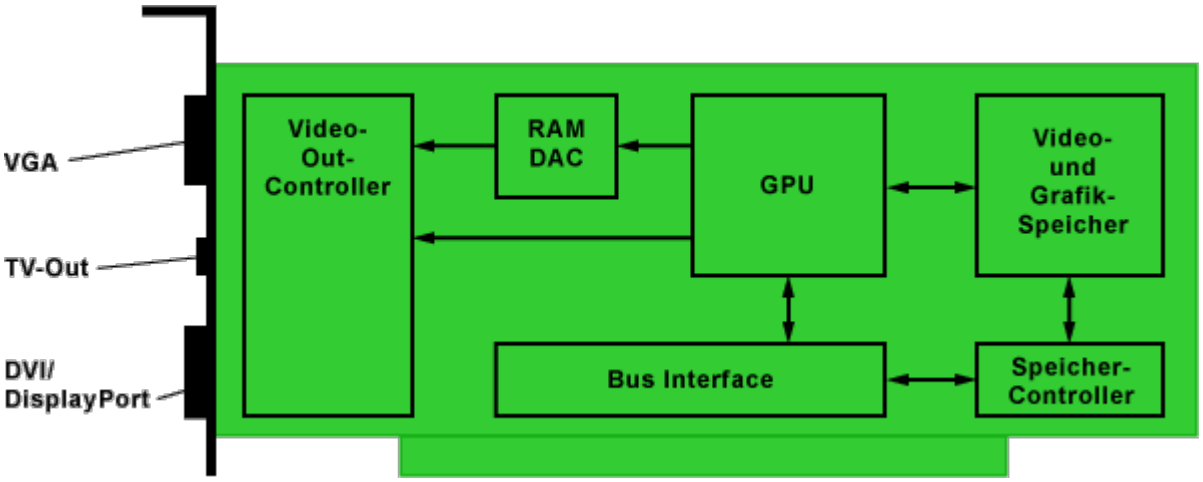
Anders als man denkt **profitieren Bildbearbeitungsprogramme** vor allem **von einer schnellen CPU** und **viel Arbeitsspeicher**. Die **Grafikkarte** bzw. der Grafikprozessor **verbessert** in der Hauptsache die **Geschwindigkeit der Bildschirmdarstellung**. Zum Beispiel beim **Verschieben von Fenstern und Bildelementen**, stufenloses **Zoomen und Drehen der Zeichenfläche**. Nur in bestimmten Fällen unterstützt die Grafikkarte das Computersystem direkt bei der Verarbeitung von Bild- und Videodaten. Zum Beispiel beim **Abspielen und Transcodieren von Videos**.



Aufbau einer Grafikkarte

Eine Grafikkarte besteht aus folgenden mechanischen und elektronischen Teilen:

- Abdeckung
- Lüfter
- Kühlkörper
- Grafikprozessor
- Speicher
- Platine
- Slotblech
- weitere Abdeckung



Aufgrund des **hohen Datenaufkommens zwischen Prozessor und Grafikkarte**, werden Grafikkarten mit einem **eigenen Prozessor** ausgestattet. Er soll den **Hauptprozessor** mit parallel laufenden Rechenoperationen **entlasten**. Das Herz der Grafikkarte ist der **Grafikprozessor (GPU)**, der die **wesentlichen Leistungsmerkmale einer Grafikkarte bestimmt**. Es gibt auch Grafikkarten, die zwei GPUs haben und die aufgrund der hohen Verlustleistung mit einem Kühlkörper oder sogar Lüfter gekühlt werden müssen.
Um die enormen Datenmengen verarbeiten zu können haben **Grafikkarten** einen **eigenen Arbeitsspeicher** und eine **spezielle Speicheranbindung mit hohen Taktraten**. Der **Grafikspeicher** kann durchaus die **Größe des normalen Arbeitsspeichers** haben.

Schnittstellen für Grafikkarten

Anfangs wurden Grafikkarten **über den internen Systembus** betrieben. Mit steigender Auflösung, Forderung nach Multimedia und 3D-Darstellung wurden die **Datenmengen immer größer**, die zur Grafikkarte transportiert werden mussten. **Um die steigenden Datenmengen zwischen Prozessor, Arbeitsspeicher und Grafikkarte bewältigen** zu können, hat Intel den **AGP-Steckplatz** für Grafikkarten eingeführt. Dieser Steckplatz war ausschließlich für Grafikkarten gedacht. Ein paar Jahre später wurde der **AGP** zusammen mit dem PCI **durch den PCI Express (PCIe) ersetzt**. Die heutigen Grafikkarten sind PEG-Grafikkarten (PCIe).

- PCI-X / PCI
- AGP - Accelerated Graphics Port
- PCI Express / PCIe
- PEG - PCI Express for Graphics

Anschlüsse für Bildschirm/Monitor

Ist die Darstellung schlecht, dann wird fast immer der Bildschirm dafür verantwortlich gemacht. Wird der Bildschirm analog, also über den VGA an die Grafikkarte angeschlossen, dann kann die **Signalqualität des VGA für das schlechte Bild verantwortlich** sein. Obwohl man annehmen könnte, dass das heute kein Problem mehr sein sollte, weisen manche Grafikkarten eine jämmerliche Signalqualität auf. Insbesondere die VGA-Anschlüsse haben diese Probleme. **Vorzugsweise sollte man den DVI- oder DisplayPort-Anschluss verwenden**.

VGA



DVI



DisplayPort



Obwohl mit **DVI** eine **digitale Schnittstelle für qualitativ hochwertige Bild- und Video-Signale** verfügbar ist macht die Einführung neuer Schnittstellen durchaus Sinn. Denn **DVI ist für hohe Auflösungen nicht geeignet**. Der **DisplayPort soll DVI** (Digital Visual Interface) im PC **und HDMI (High Definition Multimedia Interface)** in der Unterhaltungselektronik **ersetzen**.
Der **DisplayPort** ist elektrisch **kompatibel** zu den bestehenden Digitaleingängen **DVI und HDMI**. Er unterstützt auch den HDCP-Kopierschutzmechanismus und beherrscht zusätzlich ein eigenes Protokoll namens DPCP (Display Port Content Protection).

Auswahl einer Grafikkarte

Grafikkarten **unterscheidet** man im Hinblick auf ihre **3D- bzw. Spiele-Tauglichkeit, ihre Performance und Funktionsvielfalt**. Die Grafikkarte ist **nach den Anforderungen des Einsatzzwecks auszuwählen**. Man unterscheidet Grafikkarten nach ihrem Preis in **Einsteiger-, Mittelklasse-, Performance- und High-End-Modellen**. Je nach Preis unterscheiden sich Grafikkarten in Ihrer 3D-Leistung und Ausstattung.
Typische Leistungsmerkmale sind

- Taktfrequenz
- Shader-Anzahl
- Speichertakt
- Speicherbandbreite.

Bei einer Einsteiger-Grafikkarte ist die Spieletauglichkeit eingeschränkt, die Performance schwach und die Funktionsvielfalt begrenzt. Einfache Büro- und Internet-Anwendungen kommen mit einer günstigen Grafikkarte oder Onboard-Lösung aus. Bei einer High-End-Grafikkarte ist die Spieletauglichkeit unschlagbar, die Performance enorm und die Funktionsvielfalt lässt keine Wünsche offen. Insbesondere (Action-)Spiele erfordern zwangsläufig eine leistungsfähige Grafikkarte.

	Einsteiger	Mittelklasse	Performance	High-End
Preis	bis 75 EUR	bis 150 EUR	bis 300 EUR	bis 1000 EUR
Anzahl Shader	bis 160	bis 800	bis 2000	bis 4000
Speichergröße	512 MByte	1 GByte	2 GByte	4 GByte
Speichertyp	DDR2, DDR3, GDDR5	DDR3, GDDR5	GDDR5	GDDR5
typische 3D-Leistungsaufnahme	bis 50 Watt	bis 90 Watt	bis 170 Watt	bis 300 Watt oder mehr
Netzteil	min. 300 Watt	min. 350 Watt	min. 450 Watt	min. 500 Watt oder mehr

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_05:06_03_05_01

Last update: **2019/05/14 20:46**

6.3.5.2) Soundkarte



Die Soundfunktion, in Form einer Soundkarte oder im Chipsatz integriert, ist ein fast unersetzlicher Teil eines Computers. Erst mit der Möglichkeit Sprache und Musik wiederzugeben und aufzunehmen, wird ein Computer zum Multimedia-Computer. In den **Anfangszeiten der Soundkarte waren Erweiterungskarten für den ISA-Bus** üblich. Diese Karten mussten per Jumper konfiguriert werden (Interrupt, DMA, Speicher) und gaben den Ton mit einer Auflösung von 8 Bit wieder. Das entsprach 256 möglichen Zuständen. Mehr als Piepsen war den Lautsprechern nicht zu entlocken. **Später** kamen **Soundkarten mit einer Auflösung von 16 Bit**. Daraus resultierten 65.536 Klangabstufungen. Das **entspricht Hifi-Qualität**. Da der ISA-Bus am Aussterben war, haben führende Soundkarten-Hersteller auf PCI-Karten (mit Plug & Play) umgestellt. Neben der automatischen Konfiguration durch das BIOS und Betriebssystem waren auch höhere Datenraten zwischen Soundkarte und Prozessor möglich. **Inzwischen ist es üblich, dass Soundfunktionen auf dem Motherboard integriert sind (onboard)**. Anfangs hat die Klangqualität sehr zu wünschen übrig gelassen. Heute ist die **Klangqualität vollkommen ausreichend und genügt auch anspruchsvollen Ohren**. Fast jedes Motherboard bietet heute einen integrierten Sound-Chip, der eine ausreichend gute Klangqualität liefert.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_05:06_03_05_02

Last update: **2019/05/14 20:46**

6.3.5.3) Netzwerkkarte (NIC - Network Interface Card)

Eine Netzwerkkarte ermöglicht es, auf ein Netzwerk zuzugreifen und arbeitet auf der Bitübertragungsschicht (Schicht 1) des OSI-Schichtenmodells. Jede **Netzwerkkarte hat eine Hardware-Adresse (Format: XX-XX-XX-XX-XX-XX), die es auf der Welt nur einmal gibt**. Anhand dieser Adresse lässt sich eine Station auf der Bitübertragungsschicht adressieren.



An einer Netzwerkkarte ist nicht nur die **RJ45-Buchse herausgeführt**, sondern meist **auch zwei LEDs**, die den **Status der Verbindung anzeigen**. Üblich sind RJ45-Buchsen, die zwei integrierte Status-LEDs in den **Farben Grün und Orange haben**.
Die **grüne LED zeigt an, dass eine hardwareseitige Verbindung besteht**. Dazu muss der Computer nicht eingeschaltet sein (bei Formfaktor ATX).
Die **orangene LED zeigt den Status der Übertragung** an. Wenn diese LED blinkt oder flackert, dann werden gerade Daten übertragen.
Bei manchen Netzwerkkarten sind diese Funktionen etwas anders. Wenn die grüne LED flackert, dann werden gerade Daten übertragen. Ansonsten ist sie ständig grün, wenn eine Verbindung besteht. Leuchtet die orangene LED (manchmal ist sie auch gelb), dann besteht eine 100-MBit-Verbindung.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_05:06_03_05_03

Last update: **2019/05/14 20:46**

6.3.6) Eingabegeräte

Als Eingabegeräte werden **alle Geräte** bezeichnet, **über die einem Computer Informationen zugeführt werden** können, sodass Interaktion mit den Computerprogrammen möglich ist.

Einige Eingabegeräte sind hier aufgelistet:

- Maus
- Tastatur
- Touchscreen
- Grafiktablett
- Mikro
- Webcam
- Scanner
- Joystic
- PC-Controller










Y
X
B
A


Why?

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_06

Last update: **2019/05/14 20:49**

6.3.7) Ausgabegeräte

Als Ausgabegeräte werden in der Computertechnik **alle Geräte** bezeichnet, **die das Ergebnis einer Operation oder eines Programms der Außenwelt zugänglich machen**.

Dies sind insbesondere:

- Bildschirm/Beamer zur flüchtigen sichtbaren Ausgabe,
- Drucker/Plotter zur permanenten sichtbaren Ausgabe,
- Lautsprecher/Headset zur hörbaren Ausgabe,
- Braillezeile zur fühlbaren Ausgabe.

In Verbindung mit einem Eingabegerät ist Interaktion mit einem Computerprogramm möglich.











From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_03:06_03_07

Last update: **2019/05/14 20:49**

6.4) Schnittstellen (PC)

Eine Schnittstelle **verbindet Systeme, die unterschiedliche physikalische, elektrische und mechanische Eigenschaften** besitzen. Die **Definition oder Spezifikation einer Schnittstelle enthält gemeinsame Eigenschaften**. Dazu gehört auch ein **Protokoll** für die **Kommunikation** und den **Datenaustausch**. Schnittstellen befinden sich überall dort, wo unterschiedliche Systeme miteinander verbunden werden müssen. Die Schnittstellen bilden den **Übergang von einem System in ein anderes System**. Dieser Übergang kann zur Kommunikation oder dem Datenaustausch verwendet werden.

Ein Computer hat **interne** Schnittstellen, die sich **im Computer-Gehäuse** befinden und **externe** Schnittstellen, die **aus dem Computer-Gehäuse herausgeführt** sind.

- 6.4.1) Interne Schnittstellen
- 6.4.2) Externe Schnittstellen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_04

Last update: **2019/05/14 20:50**

6.4.1) Interne Schnittstellen

Interne Schnittstellen verbinden Systeme innerhalb eines Computers. Diese Schnittstellen werden meist auf dem **Motherboard als Sockel oder Slot** herausgeführt. Dort werden dann Erweiterungskarten direkt oder interne Laufwerke über Kabel angeschlossen. Einige andere Schnittstellen werden nicht herausgeführt, sondern befinden sich auf der Hauptplatine zwischen den einzelnen Controllern.

- 6.4.1.1) Interne Schnittstellen für Erweiterungskarten
- 6.4.1.2) Interne Schnittstellen für Massenspeicher

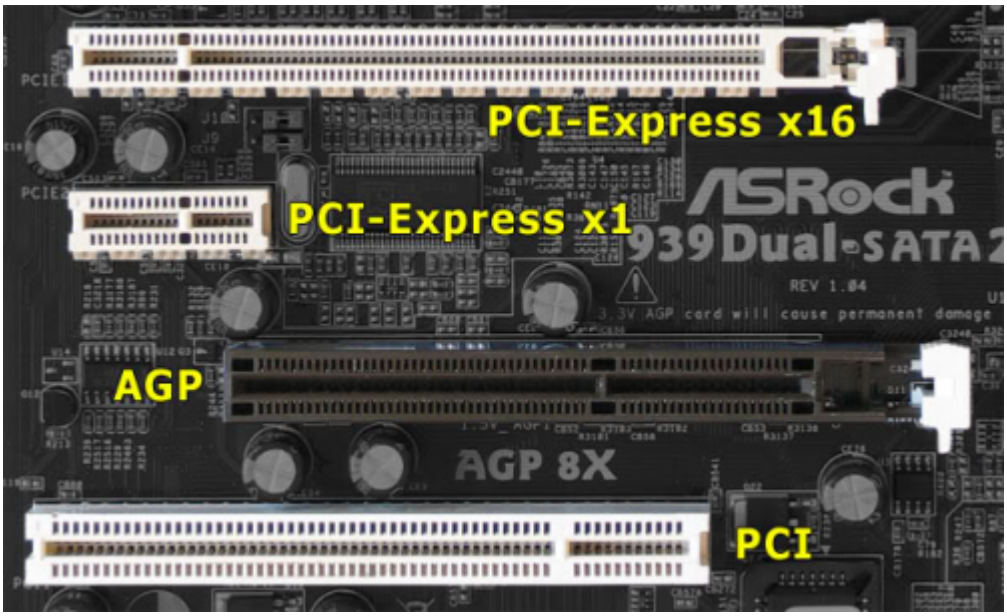
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_04:06_04_01

Last update: **2019/05/14 20:50**

6.4.1.1) Interne Schnittstellen für Erweiterungskarten

AGP (Accelerated Graphics Port)



Der braun gefärbte AGP- Steckplatz wurde als **Schnittstelle für Grafikkarten** entwickelt und ist **doppelt so schnell, wie der PCI-Bus**. Mit der letzten Spezifikation „AGB 3.0“ kann eine **maximale Taktrate von 2133 MHz** erreicht werden. Aufgrund der **Nachfrage nach höher getakteten Grafikkarten** - vor allem für die Spieleindustrie - wurde dieser Slot **von der PCI Express Schnittstelle abgelöst.**

PCI - Peripheral Component Interconnect

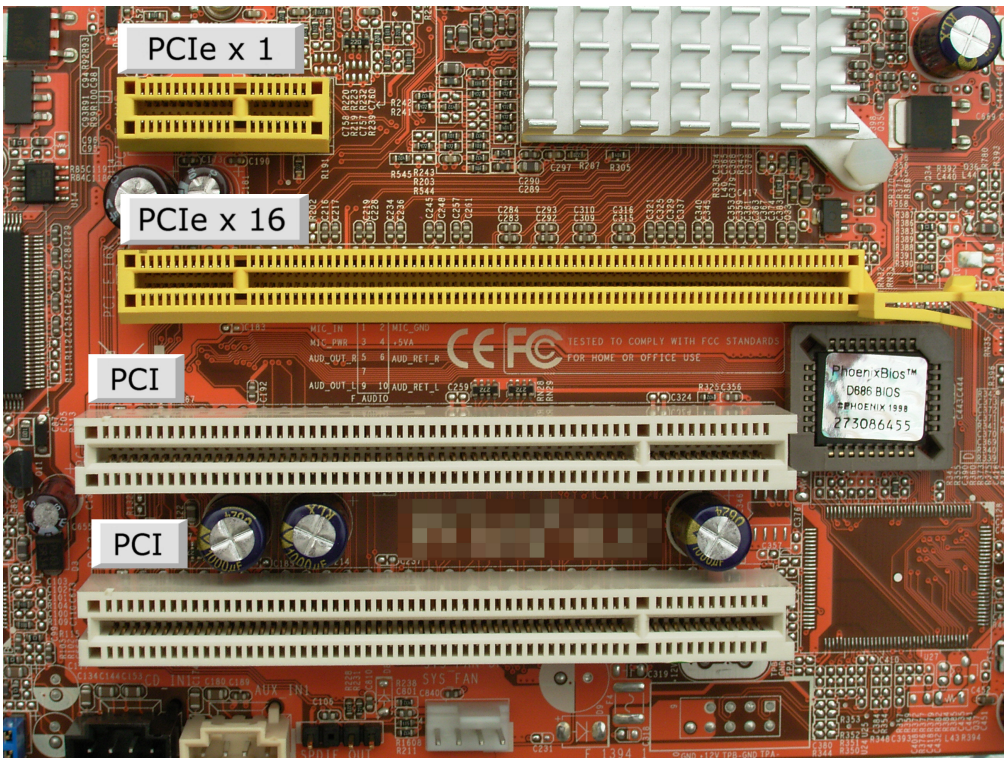
Der **PCI-Bus ist Industriestandard** und war viele Jahre fester Bestandteil von IBM-kompatiblen PCs, Macs von Apple und Alpha-Workstations von Digital. Das von Intel im Jahr 1993 entwickelte Bus-System ist bis ins Detail normiert, so dass andere Computerhersteller den PCI-Bus nachbauen können.

Der PCI-Bus kann **32 oder 64 Signalleitungen** haben. Der PCI-Bus mit 32 Bit teilen sich Adress- und Datenbus die Signalleitungen. Die Signalleitungen werden im Multiplex-Betrieb genutzt. Das bedeutet, mit einem Takt wird zuerst die Adresse und in einem zweiten Takt das Datenwort gesendet. So werden 32 Signalleitungen eingespart. In Servern kommt der PCI-Bus mit 64 Bit zum Einsatz. dort stehen jeweils 32 Adress- und Datenleitungen zur Verfügung.

Sie sind an ihrer weißen Färbung zu erkennen. Eine Erweiterung des Computers mittels Soundkarten, Netzwerkkarten, TV-Karten o.ä. ist über diese Slots möglich.

Spezifikation	Bus-Breite	Taktfrequenz	Datentransferrate	Signalspannung	Geräte pro Bus	Einführung
PCI 2.0	32 Bit	8 bis 33 MHz	0,12 GByte/s	5 V	6	1993
PCI 2.3	32 Bit	33 MHz	0,133 GByte/s	5V	6	2002
PCI 2.3	64 Bit	33 MHz	0,266 GByte/s	5V	6	2002
PCI 2.3	32 Bit	66 MHz	0,266 GByte/s	3,3V	3	2002
PCI 2.3	64 Bit	66 MHz	0,533 GByte/s	3,3V	3	2002
PCI-X 1.0	64 Bit	66 MHz	0,533 GByte/s	3,3V	4	1999
PCI-X 1.0	64 Bit	100 MHz	0,800 GByte/s	3,3V	2	1999
PCI-X 1.0	64 Bit	133 MHz	1,066 GByte/s	3,3V	1	1999
PCI-X 266 (2.0)	64 Bit	133 DDR	2,133 GByte/s	1,5V	1	2003
PCI-X 533 (2.0)	64 Bit	133 QDR	4,266 GByte/s	1,5V	1	2003

PCI-E(xpress)



PCI Express (PCIe) ist eine schnelle interne Schnittstelle für Erweiterungskarten in Computer-Systemen. Mit der **Einführung von PCIe im Jahr 2004 wurde dem AGP als Grafikkarten-Schnittstelle ein Ende gesetzt** und auch der **PCI als internes Computer-Bussystem abgelöst**.

Die **Übertragungsgeschwindigkeit** bei PCIe **orientiert sich an der Version und der Anzahl der Links bzw. Lanes**. Je höher die Version und je mehr Links, desto höher die Bandbreite und desto höher ist die Übertragungsgeschwindigkeit. Die Bandbreite gibt dabei an, wie viel Kapazität für die Datenübertragung theoretisch bzw. maximal zur Verfügung steht. Die tatsächliche Datenrate liegt jedoch darunter.

PCIe	Bandbreite pro Link		PCIe x1	PCIe x4	PCIe x8	PCIe x16	Kodierung/Balast	Verfügbar seit
1.0	2,5 GT/s	2,5 GBit/s	250 MByte/s	1 GByte/s	2 GByte/s	4 GByte/s	8b10b / 20%	2004
2.0	5 GT/s	5 GBit/s	500 MByte/s	2 GByte/s	4 GByte/s	8 GByte/s	8b10b / 20%	2008
3.0	8 GT/s	10 GBit/s	0,9846 GByte/s	3,938 GByte/s	7,877 GByte/s	15,754 GByte/s	128b/130b / <2%	2011
4.0	16 GT/s	20 GBit/s	1,969 GByte/s	7,877 GByte/s	15,754 GByte/s	31,508 GByte/s	128b/130b / <2%	2017
5.0	32 GT/s		3,9 GByte/s	15,8 GByte/s	31,5 GByte/s	63 GByte/s	128b/130b / <2%	?

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_04:06_04_01:06_04_01_01

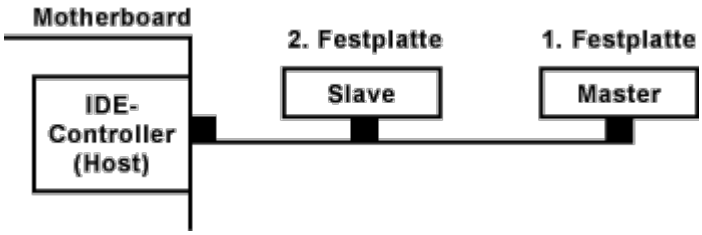
Last update: **2019/05/14 20:50**

6.4.1.2) Interne Schnittstellen für Massenspeicher

IDE - Integrated Drive Electronics



IDE bedeutet offiziell **Integrated Device Elektronics**. Bei IDE handelt sich um eine **alte Festplatten-Schnittstelle**, bei der die Steuerungselektronik bzw. der Controller in das Festplattengehäuse integriert ist. Nachfolger von IDE ist Enhanced IDE (EIDE), was auch als ATA (PATA) bezeichnet wird, eine deutlich höhere Übertragungsgeschwindigkeit hat und den Anschluss von CD-ROM- und DVD-Laufwerken ermöglicht. IDE-Controller war Anfangs eine ISA-Steckkarte für den AT-Bus. Er war jedoch eher ein sogenannter Host-Adapter, der nur die notwendigen Systembussignale mit Pufferung zur Festplattenelektronik weiterleitete. Das **40-adrige IDE-Flachbandkabel** stellte praktisch die **Verlängerung des Systembusses** dar. Später wurde der IDE-Controller fest auf dem Motherboard integriert.



Das **IDE-Flachbandkabel hat drei Steckerleisten**. Die eine ist für den **Hostanschluss** auf dem IDE-Controller. Die anderen beiden Steckerleisten sind für das **Master- und Slave-Endgerät**. **Pro IDE-Controller** lassen sich **zwei Festplatten** betreiben. Weil die eigentliche Steuerung auf den Festplatten sitzt, muss die eine Festplatte, am besten die schnellste, als Master und die andere als Slave konfiguriert werden. Dazu müssen Jumper oder Dip-Schalter gesetzt werden. Die Master-Slave-Konfiguration sorgt dafür, dass beim Systemstart der Master die höhere Priorität hat. Bevor er Funktionsbereitschaft an das Bios meldet, wartet er auf die Bestätigung der Slave-Festplatte. Beide **Festplatten (Laufwerke) arbeiten unabhängig voneinander**. Sie belegen aber die gleichen Adressen im Computersystem.

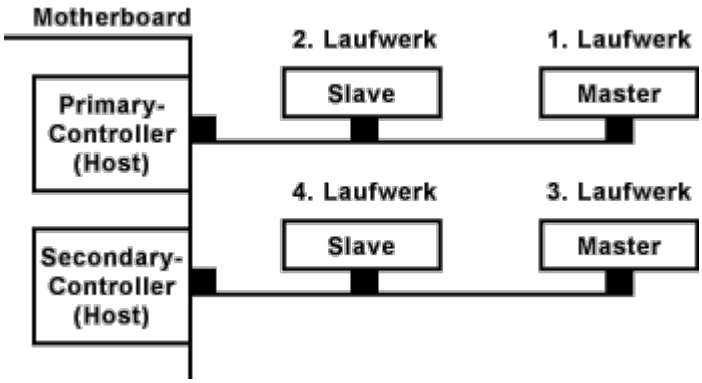
P-ATA / Ultra-ATA / EIDE





EIDE bzw. ATA sind **alte Schnittstellen für den Anschluss von Festplatten und Wechselspeicher-Laufwerken**, wie CD-ROM, DVD oder Streamer in einem Computer. **EIDE wurde von SATA abgelöst**.

Die **EIDE-Schnittstelle (Enhandced Integrated Drive Electronics)** ist eine **Weiterentwicklung des IDE-Standards**. Die EIDE-Schnittstelle bezeichnet man auch als **ATA-Schnittstelle**. ATA steht für **Advanced Technology (AT) Attachment**. Die Bezeichnung EIDE wird nur noch selten verwendet. Mit dem Aufkommen von **Serial ATA (S-ATA)** wurde die Bezeichnung **P-ATA** immer gebräuchlicher. Wobei das **P für parallel** steht.



S-ATA / Serial-ATA





Serial-ATA, kurz SATA oder S-ATA, ist eine **Schnittstelle zum Anschluss von Massenspeichern, wie Festplatten und Wechselspeicher-Laufwerken**. Schnittstellen für Massenspeicher waren **ursprünglich immer Bussysteme mit parallel geführten Signalleitungen** in Leiterbahnen und Anschlusskabeln. Mit **zunehmender Übertragungsgeschwindigkeit** ergaben sich **technische Schwierigkeiten**, die für die Übertragungsrate eine obere Grenze setzten. So blieb auch die ATA (EIDE)-Schnittstelle nicht davon verschont, dass sie auf eine **seriellen Betriebsart umgestellt** wurde.

Im **Jahr 2000** setzten sich mehrere Firmen aus dem IT-Sektor zusammen, um eine **Spezifikation über Serial-ATA (Serielles ATA) zu erstellen**. Im Jahr 2001 wurde die erste Version von Serial-ATA

vorgestellt. Anfang 2003 waren bereits die ersten Controller und Festplatten erhältlich. Bis zur vollständigen Marktdurchdringung hat es noch bis zum Jahr 2004 gedauert. Mit **150 MByte/s hat SATA** direkt an die parallele EIDE-Schnittstelle (P-ATA) mit 133 MByte/s angeknüpft. Die **SATA-Schnittstelle unterstützt 1,5 GBit/s** bei einer **Nettodatenrate von ca. 150 MByte/s**. **Festplatten mit 10.000 Umdrehungen** in der Minute (U/m) liefern rund **75 MByte/s** an Daten. Mit **SATA 6G** erreichen herkömmliche Festplatten **fast 500 MByte/s (Schreibgeschwindigkeit)**.



Zwar wurde SATA mit SATA-II und SATA 6G noch zwei mal auf maximal 600 MByte/s beschleunigt. Für **Datenspeicher mit Flash-Memory (SSD, Solid State Drive)** ist das aber **nicht schnell genug**. Allerdings gibt es SSDs, die Daten mit weit über 1.000 MByte/s lesen und schreiben können. Dafür bedarf es auch einer Schnittstelle, die diese Datenmenge bewältigen kann. SATA kann das nicht leisten. Deshalb wird **SATA durch SATA Express (SATAe) oder PCI Express (PCIe) als Massenspeicher-Schnittstelle abgelöst**.

Schnittstelle	Bezeichnung	Transferrate		Reichweite	Geräteanzahl	Einführung
Serial-ATA	SATA	1,5 GBit/s	150 MByte/s	1 m	4	2003
Serial-ATA-2	SATA-II	3 GBit/s	300 MByte/s	1 m	16	2005
Serial-ATA-3	SATA 6G / SATA-600	6 GBit/s	600 MByte/s	1 m	16	2007
Serial Attached SCSI (SAS)	SAS	3 GBit/s	300 MByte/s	1 m	16.384	2004
Serial Attached SCSI 2 (SAS 2)	SAS 6G	6 GBit/s	600 MByte/s	1 m	16.384	2007
Serial Attached SCSI 3 (SAS 3)	SAS 12G	12 GBit/s	1.200 MByte/s	1 m	16.384	2010

SATAe / Serial-ATA Express

SATA Express ist der **offizielle Nachfolger von SATA 6G** und nutzt einen zu SATA 6G abwärtskompatiblen Steckverbinder. Dieser Steckverbinder bündelt **zwei SATA-Ports** die **wahlweise für die SATA-6G- oder PCIe-Übertragung** genutzt werden können. Denn **SATA Express beherrscht sowohl SATA-6G, als auch PCIe**. Die SATAe-SSD hat dabei die Form einer herkömmlichen Festplatte. Mit PCIe 2.0 bringt SATA Express aber nur 1 GByte/s und mit PCIe 3.0 wären es auch „nur“ 2 GByte/s. Das ist für viele Anwendungen viel zu wenig. **SATA Express** scheint

schon vor seiner breiten Nutzung veraltet.

Es ist davon auszugehen, dass es sich bei SATAe nur um Übergangslösungen handelt. Mittelfristig wird sich für **SSDs der PCI Express als Standardschnittstelle** durchsetzen.



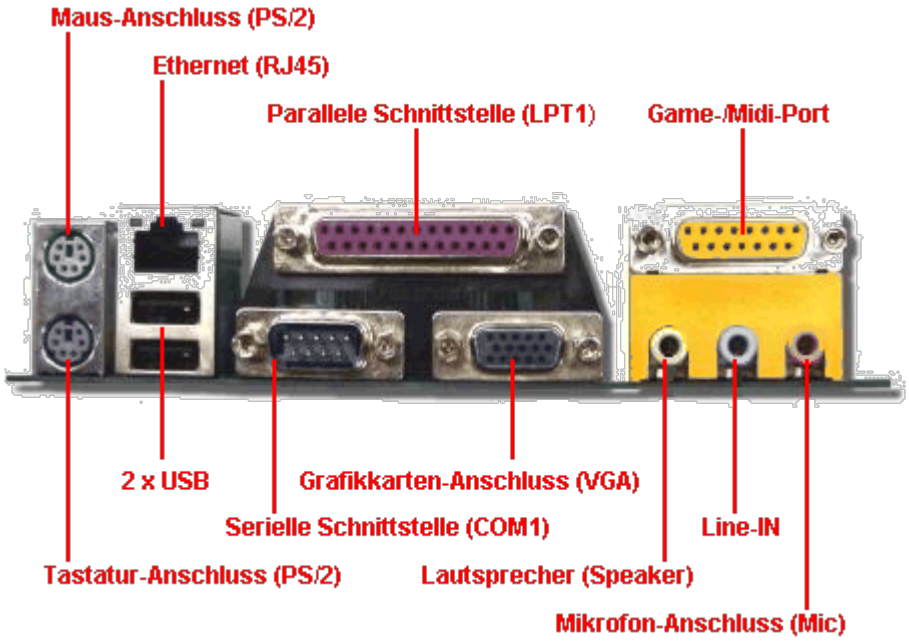
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_04:06_04_01:06_04_01_02

Last update: **2019/05/14 20:50**

6.4.2) Externe Schnittstellen

Externe Schnittstellen werden aus dem Computer-Gehäuse herausgeführt. Sie **verbinden Systeme oder Peripherie-Geräte** mit dem Computer. Die Verbindung wird mit einer **Kombination aus Stecker und Buchse** realisiert.



Serielle Schnittstelle



An jeder seriellen Schnittstelle kann nur ein weiteres Gerät angeschlossen werden. Die Bits werden nacheinander (seriell) über eine einzige Leitung übertragen, deshalb entsteht hier nur ein sehr geringer Kostenaufwand, aber auch eine niedrige Übertragungsrate. Klassische Endgeräte einer seriellen Schnittstelle sind Maus und Modem aber auch zahlreiche technische Einrichtungen, wie zB. Messgeräte, haben diese Schnittstelle, um mit Computern verbunden werden zu können.

Parallele Schnittstelle



Diese Schnittstelle wurde ursprünglich vom Drucker-Hersteller Centronics (daher oft Centronics-Schnittstelle genannt) entwickelt und war eine der ersten Schnittstellen für Drucker. Hier werden bereits 8 Bit gleichzeitig über jeweils eine eigene Leitung übertragen was im Vergleich zur seriellen Schnittstelle zu einer höheren Übertragungsrate führt. Je nach der Qualität des Kabels kann problemlos eine Leitungslänge von 5m für eine fehlerlose Datenübertragung genutzt werden.

USB (Universal Serial Bus)



Der USB ist eine universelle, externe Schnittstelle für alle Peripheriegeräte, die an einem Computer angeschlossen werden. Egal ob Tastatur, Maus, Modem, Drucker, Mikrofon, Lautsprecher, Kamera oder Scanner. Mit dem USB sind die Anwender unabhängig von der Anzahl der verfügbaren Schnittstellen und Steckplätze für Erweiterungskarten. Der USB lässt sich durch Steckkarten oder USB-Hubs fast beliebig erweitern.
Die Identifikation der Geräte wird vom USB-Hostadapter im Computer durchgeführt, der auch das Laden der Treiber und die Grundkonfiguration vornimmt. Zusätzlich verbessert sich durch Hot-Plugging, das Hinzufügen und Entfernen von Peripherie-Geräten im laufenden Betrieb, die Bedienerfreundlichkeit.

Der USB erfüllt folgende Anforderungen:



- eine einheitliche Schnittstelle für alle Peripherie-Geräte
- eine mechanisch stabile und einfache Steckverbindung
- kleine platzsparende Stecker und Buchsen

Zusätzlich weisen alle USB-Spezifikationen folgende Eigenschaften auf:

- billig
- abwärtskompatibel

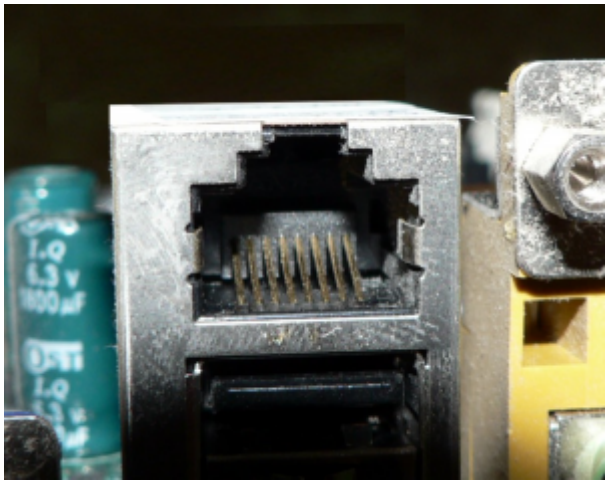
USB-Version	USB 1.0/1.1		USB 2.0	USB 3.0	USB 3.1	USB 3.2
	Low Speed	Full Speed	High Speed	Super-Speed	Super-Speed-Plus	
Symbolrate	1,875 MBi/ts	15 MBit/s	600 MBit/s	5 GBit/s	10 GBit/s	20 GBit/s
Datenrate (brutto)	1,5 MBit/s	12 MBit/s	480 MBit/s	4 GBit/s	-	-
Datenrate (theoretisch)	188 kByte/s	1,5 MByte/s	60 MByte/s	600 MByte/s	1.200 MByte/s	
Datenrate (netto)	ca. 150 kByte/s	ca. 1 MByte/s	ca. 36 bis 44 MByte/s	ca. 480 MByte/s	ca. 800 MByte/s	ca. 1.600 MByte/s
Interface	UHCI/OHCI	UHCI/OHCI	EHCI	xHCI	-	-
Leitungslänge	5 m	5m	5m	3 m	1 m	1 m
Anwendungen	Maus Tastatur	Audio	Video, Speichermedien			

FireWire



Firewire wurde ursprünglich von Apple entwickelt und wurde dann aus urheberrechtlichen Gründen von Sony in i.LINK umbenannt. Eine weitere Bezeichnung für diese Schnittstelle ist der vom Standardisierungsgremium IEEE festgelegte Name IEEE 1394. Die besondere Fähigkeit von FireWire ist der schnelle Datenaustausch zwischen externem Gerät und Computer, mit einer Übertragungsrate von bis zu 800 MBit/s. Durch diese hohe Geschwindigkeit eignet sich diese Schnittstelle besonders gut zur Übertragung von Bildern und Videos und zum Anschluss von Videokameras, Festplatten und DVD-Brennern. Im Gegensatz zu USB 2.0 wird die theoretische Datenübertragungsrate nahezu erreicht.

Ethernet (RJ45 - Netzwerk)





Dient zum Senden und Empfangen von Daten über das Netzwerk.

PS/2





PS/2 ist eine **Computer-Schnittstelle bzw. ein Anschluss für Tastaturen oder Computer-Mäuse**. Die Abkürzung PS/2 stammt ursprünglich aus dem Jahr 1987, als IBM einen Personal Computer mit der Bezeichnung PS/2 auf den Markt brachte. Dieser PC hatte erstmals diesen Schnittstellen-Typ, der für die Tastatur verwendet wurde. Die Computer-Maus wurde an einer seriellen Schnittstelle angeschlossen. Später bekamen die PCs eine zweite PS/2-Schnittstelle für die Maus dazu.

PS/2 ist im Prinzip eine **serielle Schnittstelle**, die trotz der Universal-Schnittstelle USB nicht auszusterben scheint. Der Grund warum viele Motherboards immer noch PS/2-Schnittstellen für Maus und Tastatur haben liegt in der Nachfrage begründet.

Der **USB überträgt** von der Tastatur zum Rechner immer **sechs Befehle gleichzeitig**. Für einen schnellen Tipper ist das zu wenig. **PS/2 überträgt die Tastaturanschläge kontinuierlich an den Rechner**. Dadurch ist die Latenz niedriger. Die meisten Tastaturschreiber merken diesen Unterschied natürlich nicht. Auch aus Sicherheitsgründen kann PS/2 interessant sein. Während man einen **USB-Port** für alle möglichen Geräte **„missbrauchen“** kann, eignet sich der **PS/2 eben nur für den Anschluss einer Maus oder Tastatur**. Computer, bei denen USB-Ports elektrische oder mechanisch

blockiert sind, sind weniger anfällig für lokale Manipulationen durch „böse“ USB-Endgeräte. Leider kann man eine Tastatur am PS/2 im **laufenden Betrieb nicht ausstecken, ohne Abstürze oder ein stehendes System zu provozieren**. Hier ist USB einfach flexibler.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_04:06_04_02

Last update: **2019/05/14 20:51**

6.5) Fragen

Eingabemedien

1. Nenne 4 Eingabegeräte!

Tastatur

1. Wie erhältst du das Eurozeichen?
2. Wozu dient die NumLock-Taste?
3. Was versteht man unter Scancode?

Rechner

Das Mainboard

1. Nenne 5 Komponenten des Motherboards!
2. Welche Mainboard-Formate kennst du?
3. Wozu dient der AGP-Slot? Wie heißt seine Weiterentwicklung?
4. Was versteht man unter dem Chipsatz?
5. Wie heißt die Schnittstelle(n) für Erweiterungskarten?
6. Wozu dient der Sockel? Welche verschiedene Sockelarten kennst du?

Der Prozessor (CPU - Central Processing Unit)

1. Was versteht man unter einem Mikroprozessor?
2. Was versteht man unter einem Mikrocontroller?
3. Wozu dienen die Register?
4. Welche Aufgabe hat die ALU?
5. Was ist der Unterschied zwischen Mehrkernprozessoren und Mehrprozessorsystemen?
6. Wie misst man die Geschwindigkeit von CPUs und was versteht man darunter?
7. Nenne 2 Hersteller von CPUs!
8. Nenne 4 typische Anwendungen für Single-Core CPUs!
9. Nenne 4 typische Anwendungen für Mehrkernprozessoren!
10. Warum gibt es Mehrkernprozessoren überhaupt?
11. Was ist die Problematik bei Mehrkernprozessoren bzgl. Leistungssteigerung?
12. Wie heißt der schnelle Zwischenspeicher zwischen CPU und RAM und was bedeuten dabei die verschiedenen Levels?

Speichermedien

Speicherkategorien

1. Nenne die drei Arten von Speichermedien und jeweils 2 Beispiele dafür!
2. Gib an, welche Speicherart am schnellsten und welche am langsamsten ist!

Magnetische Speicher

1. Wie lautet die Abkürzung für eine magnetische Festplatte und für was steht die Abkürzung!
2. Wie ist eine magnetische Festplatte aufgebaut?
3. Wie groß war die Kapazität einer 3 1/2- Zoll-Diskette?
4. Wieso sollte man Festplatten während des Betriebs keinen Stößen aussetzen bzw. warum sollte kein Staubkorn in das Innere einer magnetischen Festplatte gelangen?
5. Welche 4 Kriterien sind entscheidend für die Geschwindigkeit einer magnetischen Festplatte (HDD)?
6. Wie sind die Daten auf einer magnetischen Festplatte organisiert? Erkläre dabei die Begriffe Spuren, Zylinder und Sektoren (=Blöcke)!

Elektronische Speicher

1. Welche zwei elektronische Bauteile sind in einem elektronischen Speicher am häufigsten verbaut?
2. Wofür stehen die Abkürzung ROM?
3. Wofür stehen die Abkürzung RAM?
4. Was ist der Unterschied zwischen RAM und ROM?
5. Welche 2 Arten von RAM gibt es?
6. Welche 4 Arten von SDRAM gibt es?
7. Warum kommt es bei einem DDR-SDRAM zu einer verdopplung der Taktrate im Vergleich zum SDRAM?
8. Wie hoch sind die aktuellen Taktraten von den neuesten DDR4-SDRAM-Speicher?
9. Welche RAM-Bausteine sind heutzutage bei Computern in Verwendung?
10. Warum und worauf muss ich beim Motherboardkauf auf den RAM-Speicher Rücksicht nehmen?
11. Nenne 2 Primärspeicher und 2 Sekundärspeicher!
12. Welche Flash-Speicher kennst du?
13. Nenne 4 Vorteile und 2 Nachteile von Flash-Speicher!
14. Nenne 4 Anwendungen von Flashspeicher!
15. Warum sind SSDs schneller als HDDs?
16. Was ist eine SSHD und wie ist diese aufgebaut?
17. Welche Speicherkartenarten kennst du? Welche wird heutzutage in Handys verwendet?

Optische Speicher

1. Erkläre wofür die Abkürzungen CD, DVD, BD und UHD-BD stehen!
2. Wie viele MB kann eine CD speichern?
3. Wie viele GB kann eine DVD speichern?
4. Wie viele GB kann eine BD speichern?
5. Wie viele GB kann eine UHD-BD speichern?
6. Wieso kann eine DVD bei gleicher Größe mehr Daten fassen?
7. Wie bezeichnet man die Erhebungen und Vertiefungen auf einer CD bzw. DVD?

8. Wozu werden BD zumeist verwendet?

Schnittstellen

1. Was ist der Unterschied zwischen internen und externen Schnittstellen?
2. Welche interne Schnittstellen gibt es für Massenspeicher? Was sind die Unterschiede zwischen den einzelnen Schnittstellen?
3. Welche Geräte werden/wurden an die parallele Schnittstelle angeschlossen?
4. Was ist der große Vorteil von USB?
5. Für welche Geräte dient die externe DVI-Schnittstelle als Verbindung zum Computer?
6. Was ist der Unterschied zur externen VGA-Schnittstelle?

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:06:06_05

Last update: **2019/05/14 21:54**

7) Textverarbeitung & Zeichencodierung

7.1) Theorie

Textverarbeitung ist die Erstellung und Bearbeitung von schriftlichen Texten mit Hilfe von Programmen/Applikationen.

Bis weit ins 19. Jahrhundert war es üblich, Texte (sofern sie nicht gedruckt wurden) mit der Hand zu schreiben. Nachdem im 19. Jahrhundert die ersten praktisch verwendbaren Schreibmaschinen auf den Markt kamen, begann sich dies jedoch allmählich zu ändern. Nach und nach setzte sie sich gegenüber der Handschrift durch, ihre **Vorteile gegenüber der Handschrift**:

- **Klareres Schriftbild**
- **Normierte Zeichengrößen**
- **Schnelleres Schreiben**

Vor allem im Bürobereich fanden Schreibmaschinen deshalb große Verbreitung. Dies hatte **enormen Einfluss auf die Arbeitswelt**: Die bis dahin oft gebräuchlichen Stehpulte wurden durch Schreibtische abgelöst, **das Schreiben von einer stehenden zu einer sitzenden Tätigkeit**. Dieser Prozess lief parallel zu anderen Entwicklungen in der Mechanisierung der Büroarbeit (Einführung von Rechenmaschinen, Hollerithmaschine und Telefon).

Mit der zunehmenden Verbreitung von PCs in den 1990er Jahren verschwanden Schreibmaschinen jedoch vom Markt.
PCs waren **universell einsetzbar** und **nicht nur auf Textverarbeitung** beschränkt, bei zunehmend günstigerem Preis-Leistungs-Verhältnis zugunsten des PCs. Die Fähigkeiten von Textverarbeitungen erforderten zunehmend vollwertige Computer, vor allem nach dem Aufkommen grafischer Benutzeroberflächen. Seit **Einführung des Personal Computers** hat sich der **Bereich der Textverarbeitung rasant entwickelt**. Das heutige Ergebnis sind Programme, bei denen der reine Textverarbeitungsteil programmiertechnisch wohl noch den geringsten Aufwand erforderte. Vielmehr wurden sie angereichert mit Funktionen, die zuvor einzeln von anderen Programmen und oft auch anderen Herstellern bezogen werden mussten. **Heutige Textverarbeitungsprogramme** integrieren die reine Texterfassung mit den **Möglichkeiten der Grafikeinbindung, Tabellenerstellung, Formelgenerierung oder der Präsentationsgrafik**.

Die Einführung der grafischen Benutzeroberflächen wie die des Apple Macintosh 1984 und später Windows ermöglichte erst den Schritt in diese Richtung, verhalf sie doch den Programmen, auf Techniken wie dem **Datenaustausch über eine zentrale Zwischenablage**, dem dynamischen Datenaustausch oder dem Objekt-verbinden-und-einbetten zurückzugreifen. Die Möglichkeit, den Text gemäß dem Prinzip **WYSIWYG (engl. What You See Is What You Get)** bereits auf dem Bildschirm so präsentiert zu bekommen, wie er auch später ausgedruckt erscheint (Druckdarstellung), war dabei von großer Bedeutung.

WYSIWYG

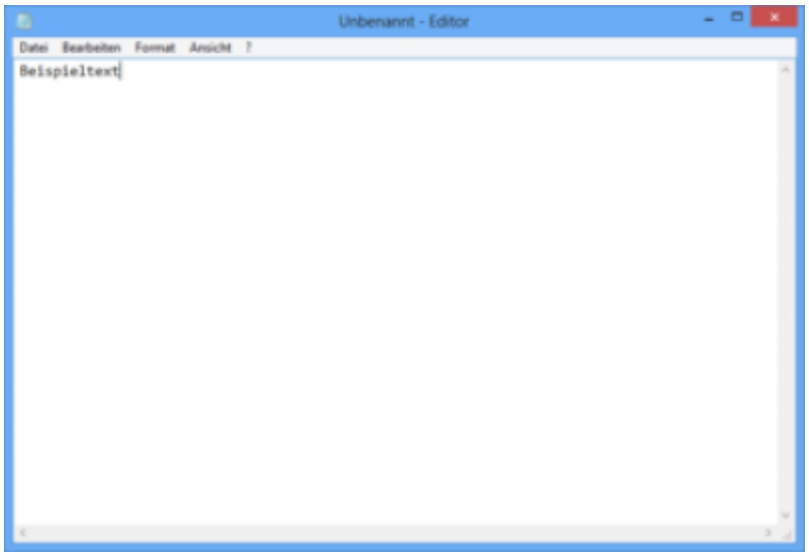
WYSIWYG ist das Akronym für den Grundgedanken ***What You See Is What You Get*** – auch als **Echtzeitdarstellung** bekannt (beziehungsweise, bezogen auf die Bildschirmdarstellung,

Echtbilddarstellung). Bei echtem WYSIWYG wird ein Dokument während der Bearbeitung am Bildschirm genauso angezeigt, wie es bei der Ausgabe über ein anderes Gerät, z. B. einen Drucker, aussieht. Der Begriff wurde zuerst Anfang der 1980er im Zusammenhang mit Computer-Drucksatz-Systemen (Desktoppublishing) und Textverarbeitungsprogrammen verwendet.

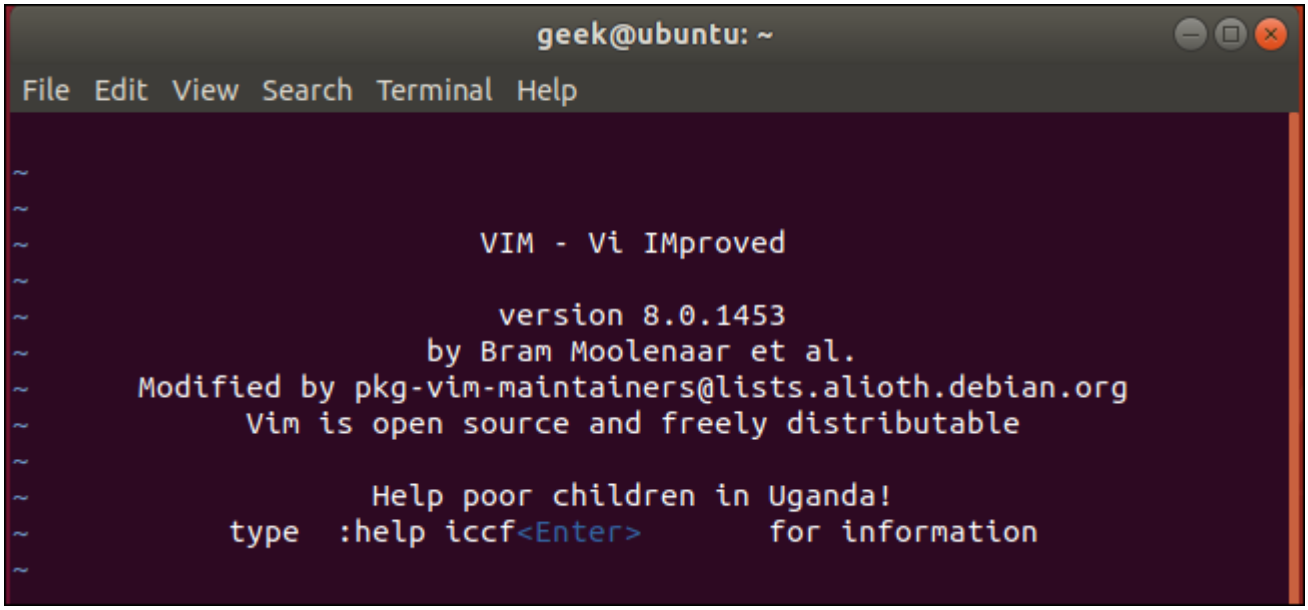
Texteditoren und ASCII-Editoren

Die ersten **Textverarbeitungsprogramme**, treffender bezeichnet als „Texterfassungsprogramme“, waren **einfache Texteditoren**, die sich aus dem **Zeichenvorrat nichtgenormter Zeichensätze** bedienten. Somit waren nicht nur die **Editoren an den zugehörigen Computer (bzw. das zugehörige Betriebssystem) gebunden**, sondern auch die darauf erstellten Texte, die bei Bedarf erst für andere Computersysteme in andere Zeichensätze konvertiert werden mussten.

Erst mit der Einführung **genormter Zeichensätze (z.B. ASCII)** kann man von **ASCII-Editoren** sprechen, die sich aus dem Zeichenvorrat der **standardisierten ASCII-Tabelle** bedienen. Damit wurden auch die Editorprogramme selbst grundlegend **kompatibel zu gleichartiger Hardware unter den meisten gängigen Betriebssystemen**.
Der Befehlsumfang einfacher ASCII-Editoren ging in der Regel über das Speichern, Laden und Drucken nicht hinaus.
Ein Beispiel für einen reinen ASCII-Editor ist der Editor unter Microsoft Windows.



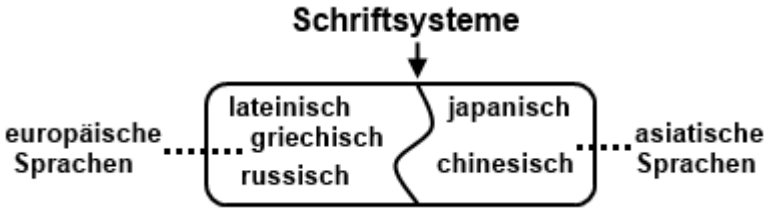
Erweiterte Programme bieten jedoch neben Funktionen wie Textmarkierung oder das Ausschneiden, Kopieren und Versetzen von Textabschnitten an: **Syntaxhervorhebung und Makroprogrammierung**. Besonders **komplexe Editoren** wie **Emacs und Vim** unter Linux kommen einer modernen Textverarbeitung unter den Einschränkungen eines textorientierten Betriebssystems bereits sehr nahe und berücksichtigen die Bedürfnisse von Systemadministratoren oder Programmierern, etwa durch **Verknüpfung mit Compilern** und anderer **externer Software**.



Zeichencodierung

Da in der frühen Phase der **Computertechnologie** die ersten Computersysteme in den **westlichen Ländern** entwickelt wurden, vor allem in den USA, **konzentrierten** sich die **Entwickler von Textverarbeitungsprogrammen** zunächst auf die Kodierung von Texten, die auf dem **lateinischen Schriftsystem** basierten.

- Eine **Schrift** besteht aus einer **bestimmten Anzahl von Schriftzeichen**.
- Diese **Anzahl unterscheidet sich stark**, je nach verwendetem **Schriftsystem**



Während Schriftsysteme **europäischer Sprachen**, wie z.B. lateinisch, griechisch und russisch aus einer relativ geringen Anzahl von **Schriftzeichen** bestehen, enthalten die **Schriftsysteme asiatischer Sprachen**, wie z.B. chinesisch und japanisch eine **wesentlich größere Anzahl** von Schriftzeichen.

Bits, Bytes und Buchstaben

Computersysteme kennen im Prinzip **keine Buchstaben** oder **Zeichen**, sondern nur **Bits und Bytes**.

⇒ daher muss bei der Textverarbeitung auf Computersystemen die **Schriftzeichen** erst in Bits (0 und 1) **umgewandelt („kodiert“)** werden! Dies macht der Computer nach einer **Zuordnungstabelle**, in der verzeichnet ist, welche Zahl für welchen Buchstaben steht. Diese Tabelle ist in beiden Richtungen eindeutig, aus einem Buchstaben wird also exakt eine Zahl und aus einer Zahl wird umgekehrt wieder exakt dieser Buchstabe!

⇒ Die bezeichnet man als **„ZEICHENCODIERUNG“.**

Dazu wird jedem Schriftzeichen eine Zahlenreihe aus Nullen und Einsen zugeordnet.

Beispiel:

- die Zahlenreihe 1000001 ⇒ hat den Dezimalwert 65 und steht für den großen Buchstaben „A“
- die Zahlenreihe 1100001 ⇒ hat den Dezimalwert 97 undsteht für den kleinen Buchstaben „a“
- die Zahlenreihe 1000010 ⇒ hat den Dezimalwert 66 undsteht für den großen Buchstaben „B“
- die Zahlenreihe 1100010 ⇒ hat den Dezimalwert 98 undsteht für den kleinen Buchstaben „b“, usw.

ASCII

Grundsätzlich ist die Zuordnung der Zahl zum Buchstaben vollkommen willkürlich wählbar, und daher entwickelten sich in der **Anfangszeit der Computer in Amerika mehrere unterschiedliche Zuordnungsvorschriften**. Von diesen haben sich zwei heute gehalten:

- der American Standard Code for Information Interchange (ASCII)
- Extended Binary Coded Decimals Interchange Code (EBCDIC)



ASCII-TABELLE

Dezimal	Hexadezimal	Binär	Oktal	ASCII-Character
0	0	0	0	[NULL]
1	1	1	1	[START OF HEADING]
2	2	10	2	[START OF TEXT]
3	3	11	3	[END OF TEXT]
4	4	100	4	[END OF TRANSMISSION]
5	5	101	5	[ENQUIRY]
6	6	110	6	[ACKNOWLEDGE]
7	7	111	7	[BELL]
8	8	1000	10	[BACKSPACE]
9	9	1001	11	[HORIZONTAL TAB]
10	A	1010	12	[LINE FEED]
11	B	1011	13	[VERTICAL TAB]
12	C	1100	14	[FORM FEED]
13	D	1101	15	[CARRIAGE RETURN]
14	E	1110	16	[SHIFT OUT]
15	F	1111	17	[SHIFT IN]
16	10	10000	20	[DATA LINK ESCAPE]
17	11	10001	21	[DEVICE CONTROL 1]
18	12	10010	22	[DEVICE CONTROL 2]
19	13	10011	23	[DEVICE CONTROL 3]
20	14	10100	24	[DEVICE CONTROL 4]
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]
22	16	10110	26	[SYNCHRONOUS IDLE]
23	17	10111	27	[ENG OF TRANS. BLOCK]
24	18	11000	30	[CANCEL]
25	19	11001	31	[END OF MEDIUM]
26	1A	11010	32	[SUBSTITUTE]
27	1B	11011	33	[ESCAPE]
28	1C	11100	34	[FILE SEPARATOR]
29	1D	11101	35	[GROUP SEPARATOR]
30	1E	11110	36	[RECORD SEPARATOR]
31	1F	11111	37	[UNIT SEPARATOR]
32	20	100000	40	[SPACE]
33	21	100001	41	!
34	22	100010	42	"
35	23	100011	43	#
36	24	100100	44	$
37	25	100101	45	%
38	26	100110	46	&
39	27	100111	47	'
40	28	101000	50	(
41	29	101001	51	)
42	2A	101010	52	*
43	2B	101011	53	+
44	2C	101100	54	,
45	2D	101101	55	-
46	2E	101110	56	.
47	2F	101111	57	/
48	30	110000	60	0
49	31	110001	61	1
50	32	110010	62	2
51	33	110011	63	3
52	34	110100	64	4
53	35	110101	65	5
54	36	110110	66	6
55	37	110111	67	7
56	38	111000	70	8
57	39	111001	71	9
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	A
66	42	1000010	102	B
67	43	1000011	103	C
68	44	1000100	104	D
69	45	1000101	105	E
70	46	1000110	106	F
71	47	1000111	107	G
72	48	1001000	110	H
73	49	1001001	111	I
74	4A	1001010	112	J
75	4B	1001011	113	K
76	4C	1001100	114	L
77	4D	1001101	115	M
78	4E	1001110	116	N
79	4F	1001111	117	O
80	50	1010000	120	P
81	51	1010001	121	Q
82	52	1010010	122	R
83	53	1010011	123	S
84	54	1010100	124	T
85	55	1010101	125	U
86	56	1010110	126	V
87	57	1010111	127	W
88	58	1011000	130	X
89	59	1011001	131	Y
90	5A	1011010	132	Z
91	5B	1011011	133	[
92	5C	1011100	134	\
93	5D	1011101	135	]
94	5E	1011110	136	^
95	5F	1011111	137	_
96	60	1100000	140	`
97	61	1100001	141	a
98	62	1100010	142	b
99	63	1100011	143	c
100	64	1100100	144	d
101	65	1100101	145	e
102	66	1100110	146	f
103	67	1100111	147	g
104	68	1101000	150	h
105	69	1101001	151	i
106	6A	1101010	152	j
107	6B	1101011	153	k
108	6C	1101100	154	l
109	6D	1101101	155	m
110	6E	1101110	156	n
111	6F	1101111	157	o
112	70	1110000	160	p
113	71	1110001	161	q
114	72	1110010	162	r
115	73	1110011	163	s
116	74	1110100	164	t
117	75	1110101	165	u
118	76	1110110	166	v
119	77	1110111	167	w
120	78	1111000	170	x
121	79	1111001	171	y
122	7A	1111010	172	z
123	7B	1111011	173	{
124	7C	1111100	174	|
125	7D	1111101	175	}
126	7E	1111110	176	~
127	7F	1111111	177	[DEL]

Drückt man also eine Taste der Tastatur, erkennt der Computer, welche Taste gedrückt wurde, und ordnet ihr aufgrund des konfigurierten Tastaturlayouts den Bytewert des Buchstabens zu, der sich dahinter verbirgt. Bei der Ausgabe des Buchstabens auf dem Bildschirm wird der Bytewert umgewandelt in die Anweisung, wie der entsprechenden Buchstabe zu malen ist, und er erscheint dann auf dem Bildschirm. Wie der Buchstabe konkret aussieht, entscheidet dabei die **Malanweisung der Schriftart**. **Ändert man die Schriftart**, wird eine **neue Malanweisung** verwendet, aber die **Bytewerte der Zeichen bleiben identisch**.

Diese betagten Codes wie ASCII und EBCDIC haben jedoch eine Schwäche: Mit ihnen lassen sich, da sie einem Buchstaben ein Byte zuordnen, **maximal 256 verschiedene Zeichen** darstellen. Dieser Zeichenvorrat **reicht gerade für das lateinische Alphabet** und **einige zusätzliche Zeichen**. Als die Codes in den **1960ern** entstanden, war das für die englisch sprechenden Amerikaner **ausreichend**. Als ASCII entstand, kam man – zumindest in der Datenverarbeitung im englischen Sprachraum – mit 128 verschiedenen Zeichen aus, es reichten daher **7 Bit für die Codierung des Zeichensatzes**. Das **8. Bit** eines Bytes wurde **nicht benutzt** oder für verschiedene andere Zwecke benutzt!

Die erste, nicht hundertprozentig befriedigende Lösung für dieses Problem bestand darin, für andere Alphabete neue Zuordnungen zu erstellen. Dies bedeutete bei der interkulturellen Kommunikation aber einen kleinen Mehraufwand. Wollte beispielsweise ein Russe einem Amerikaner einen russischen Text zukommen lassen, so reichte es nicht, einfach die Textdatei weiterzugeben. Der Absender musste zusätzlich die für das kyrillische Alphabet verwendete Codierung angeben. Mit der amerikanischen Codierung würde der Computer des Empfängers die Bytes als lateinische Buchstaben interpretieren und somit einen sinnlosen Buchstabensalat produzieren.

ISO-8859-Familie

Im **westeuropäischen Sprachraum** verbreiteten sich insbesondere **zwei Codierungen aus der sogenannten ISO-8859-Familie**:

- ISO-8859-1 und
- ISO-8859-15.

Den Codierungen der ISO-8859-Familie ist gemeinsam, dass sie nur 256 verschiedene Zeichen codieren können, da ein Zeichen immer mit einem Byte codiert wird. Jede dieser Codetabellen versucht, möglichst alle Zeichen möglichst vieler Schriften zu speichern. Sie **verwenden alle die ASCII-Codetabelle für die Zeichen 0 bis 127** und definieren **zusätzlich die Werte 160 bis 255**. Sofern in diesem Bereich das gleiche Zeichen in verschiedenen Schriften vorkommt, überlappen sie sich zumeist.

Upper part of ISO 8859-1 character set

160 NBSP	161 ¡	162 ¢	163 £	164 ¤	165 ¥	166 ¦	167 §	168 ¨	169 ©	170 ª	171 «
172 ¬	173 SHY	174 ®	175 ¯	176 °	177 ±	178 ²	179 ³	180 ´	181 µ	182 ¶	183 ·
184 ¸	185 ¹	186 º	187 »	188 ¼	189 ½	190 ¾	191 ¿	192 À	193 Á	194 Â	195 Ã
196 Ä	197 Å	198 Æ	199 Ç	200 È	201 É	202 Ê	203 Ë	204 Ì	205 Í	206 Î	207 Ï
208 Ð	209 Ñ	210 Ò	211 Ó	212 Ô	213 Õ	214 Ö	215 ×	216 Ø	217 Ù	218 Ú	219 Û
220 Ü	221 Ý	222 Þ	223 ß	224 à	225 á	226 â	227 ã	228 ä	229 å	230 æ	231 ç
232 è	233 é	234 ê	235 ë	236 ì	237 í	238 î	239 ï	240 ð	241 ñ	242 ò	243 ó
244 ô	245 õ	246 ö	247 ÷	248 ø	249 ù	250 ú	251 û	252 ü	253 ý	254 þ	255 ÿ

Der universale Code: Unicode

Mit **Unicode** werden so ziemlich **alle Zeichencodierungsprobleme** dieser Welt **gelöst**. Und weil Webseiten potentiell mit allen Schriften dieser Welt genutzt werden, ist für HTML 4.0 und XML (und damit auch XHTML) festgelegt, dass grundsätzlich alle in Unicode definierten Zeichen vorkommen dürfen.

Mit dieser Festlegung kann man also jedes der mittlerweile über **100.000 Unicode-Zeichen** in seinem Text verwenden. Aber Unicode regelt nicht nur die Codierung aller Schriftzeichen dieser Welt, sondern kennt zu jedem Zeichen dutzende definierte Eigenschaften. Ein Zeichen besitzt mindestens folgende Informationen:

- der eindeutige Unicode-Name des Zeichens
- den zugeordneten Codepoint (die „Nummer“ des Zeichens)
- eine generelle Kategorisierung (Buchstabe, Zahl, Symbol, Interpunktion, …)
- weitere grundsätzliche Charakteristiken (Whitespace, Bindestrich, alphabetisch, ideographisch, Nicht-Zeichen, veraltet, …)
- anzeigerelevante Informationen (Schreibrichtung, Form, Spiegelung, Breite)
- Groß-/KleinschreibungZahlenwert und -typ bei Zahlzeichen
- Trennfunktionen (zwischen Wörtern, Zeilen, Sätzen,…)
- usw.

Die meisten Eigenschaften eines Zeichens sind nur für die interne Behandlung durch den Computer relevant. Beispielsweise werden arabische Schriftzeichen automatisch korrekt von rechts nach links geschrieben, oder der Zeilenumbruch nutzt die Stellen, an denen eine Wortgrenze z. B. durch ein normales Leerzeichen existiert. Dieser kleine Exkurs soll aber verdeutlichen, dass die Bytewerte der Buchstaben für den Computer viel mehr bedeuten können, als nur den Buchstaben an sich.

Der oben erwähnte Codepoint eines Unicode-Zeichens ist nur **eine abstrakte Nummer**. Die Schreibweise dieser Nummer im Unicode-Standard erfolgt in **hexadezimalen Zahlen mit vorangestelltem „U+“**. Der Codepoint legt noch keinerlei computerkompatible Darstellung fest, dies ist Aufgabe des Codierschemas. Da die Unicode-Codepoints von U+0000 bis U+10FFFF (hexadezimale Zahlendarstellung), mit einer beabsichtigten Lücke zwischen U+D7FF und U+E000, reichen, sind für eine vollständige Codierung des gesamten Codepoint-Bereichs als Binärzahl mindestens 3 Byte erforderlich.

Folgende Speichervarianten von Unicode sind heute gängig:

- **UTF-8** (8 Bits ⇒ 1 Byte)
- **UTF-16** (16 Bits ⇒ 2 Byte)
- **UTF-32** (32 Bits ⇒ 3 Byte)

Praxistipp

Empfehlung: Verwenden Sie, wo immer es geht, UTF-8! Es hat viele Vorteile, bei der Wahl der codierbaren Zeichen nicht eingeschränkt zu sein. Selbst wenn ein Webauftritt garantiert nur für eine einzige Sprache (z. B. Deutsch) gestaltet werden muss, bietet nur Unicode die komplette Freiheit, wirklich alle Zeichen – insbesondere Interpunktionszeichen wie „typografische Anführungszeichen“ oder eben das Eurozeichen – beliebig verwenden zu können. Eingestreute Fremdwörter in einer anderen Sprache sind genauso problemlos möglich wie die Verwendung eher exotischerer Zeichen wie den Brüchen ⅓, ⅔, ⅛, ⅜, ⅝, ⅞.

Viele Programme können schon seit langer Zeit mit UTF-8 umgehen: Server, Browser, Datenbanken, Programmiersprachen, Editoren. Der einzige Grund, auf UTF-8 zu verzichten, ist die Existenz von alten Systemen oder Daten, die nicht in UTF-8 gespeichert sind und bei denen es zu aufwendig wäre, sie nach UTF-8 zu konvertieren.

Begrifflichkeiten

Für die Zeichen-Problematik gibt es eine Reihe Begrifflichkeiten, die allerdings oftmals falsch angewendet werden. Nachfolgend der Versuch einer Aufklärung:

Zeichensatz

Ein **Zeichensatz (character set, kurz charset)** ist die **Gesamtheit der zur Verfügung stehenden Zeichen**. Ein Zeichensatz ist ein eher abstraktes Gebilde, das nur die Zeichen selbst und eine Reihenfolge beschreibt, **nicht jedoch eine konkrete Abbildung auf Byte-Werte**. Das ist Aufgabe der **Zeichencodierung**.

Beispielsweise ist **Unicode ein Zeichensatz**, **UTF-8** hingegen ist eine **Zeichencodierung**.

Codepoint

Zeichen in einem Zeichensatz (z. B. ASCII oder Unicode) werden in einer **bestimmten Reihenfolge in einem Coderaum (code space)** angeordnet. Die Position eines Zeichens ist der Codepoint (zu Deutsch etwa **„Codeposition“**).
Das heißt die Codeposition ist der Zahlenwert eines Zeichens im Coderaum. \
Bsp.:
A hat die Codeposition 65 im Coderaum ASCII.

Ein **Zeichensatz mit Codepoints** ist ein **codierter Zeichensatz (coded character set)**.

Schriftart

Für die grafische Darstellung von Zeichen wird eine Schriftart verwendet, die in der Regel eine in sich **konsistente Gestaltung** aufweist, beispielsweise für die **Strich-Dicke** oder **Verzierungen**. Arial, Times New Roman oder Courier sind Beispiele für Schriftarten.
Schriftart ist im engeren Sinne nach eher ein **Synonym zu Schriftschnitt**, also bspw. **Helvetica 24pt fett**. Da Schriftarten wie Arial, Helvetica, Times Roman usw. oft aus mehreren Schriftschnitten für Breiten (schmal, breit, ...), Strichstärken (leicht, normal, fett, ...) und Zeichenlagen (normal, kursiv, ...) bestehen, verwendet man auch den Begriff **Schriftfamilie**, der nicht genau von **Schriftart abzugrenzen** ist.



Kriterien für Schriftarten

Die Kriterien für die Verwendung einer bestimmten Schriftart können sein:

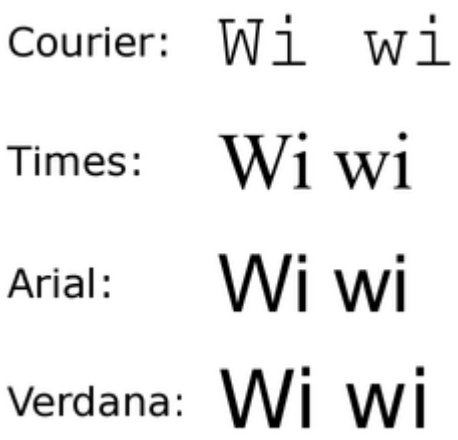
- **Verfügbarkeit**: Die Windows-Schriftarten wie Arial zum Beispiel sind gerade deshalb weit verbreitet, weil sie auf jedem Windows-System verfügbar sind und entsprechend formatierte Dokumente auch unter mehreren Benutzern leicht ausgetauscht werden können
- **Emotionale Wirkung**: Ein Text kann die unterschiedlichsten Zwecke erfüllen. Entsprechend seiner Aufgabe wird er in einer dazu passenden Schrift gesetzt. Im Verkehrsbereich finden vorwiegend Grotesken, wie die Univers, Anwendung, in belletristischer Literatur wird gerne auf Renaissance-Antiqua wie die Garamond zurückgegriffen, und eine spielerischere Schriftart ist die Comic Sans MS, die auch als „Kinder-Schriftart“ benutzt wird.
- **Lesbarkeit**: Sie ist wichtig, um ein Buch entspannt lesen zu können oder weiter entfernte Straßenschilder zu entziffern. Das Auge des Lesers muss möglichst leicht die einzelnen Buchstaben eines Textes voneinander trennen und erkennen können. Einfluss auf die Lesbarkeit einer Schrift hat neben der Gestaltung der Wort- und Zeichenzwischenräume auch die Gestalt der Zeichen selbst.

Schriftgestalt

Serifen

Serifen sind **kleine Endstriche eines Buchstabens**, umgangssprachlich auch „**Füßchen**“ genannt. Sie bilden eine horizontale Linie, an der sich das **Auge des Lesers orientieren** kann. Daher eignen sich Serifenschriften, insbesondere die Antiqua, **besonders für gedruckten Fließtext (Bücher,**

Artikel). Bei Postern, Plakaten, Schildern usw. kommt es dagegen darauf an, auch auf größere Distanz einzelne Wörter zu entziffern. Hier werden wegen ihrer größeren Klarheit Schriften ohne Serifen eingesetzt.



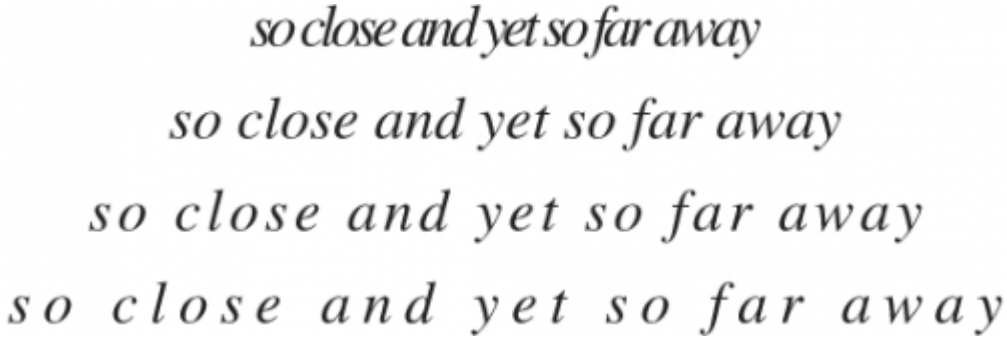
Höhe

Neben der Tatsache, dass es zwei parallele Alphabete von Groß- und Kleinbuchstaben gibt, unterscheiden sich seine Buchstaben noch in weiteren Punkten. Zum Beispiel betrachtet man die Höhe der einzelnen Buchstaben. Zunächst einmal allen gemeinsam ist, dass sie eine Grundlinie teilen. Die **Grundlinie ist der Boden der Zeile** selbst, der bei Serifenschriften durch ihre Füße angedeutet wird. Auf dieser Grundlinie „stehen“ alle Buchstaben und gehen unterschiedlich weit über bzw. unter diese hinaus.



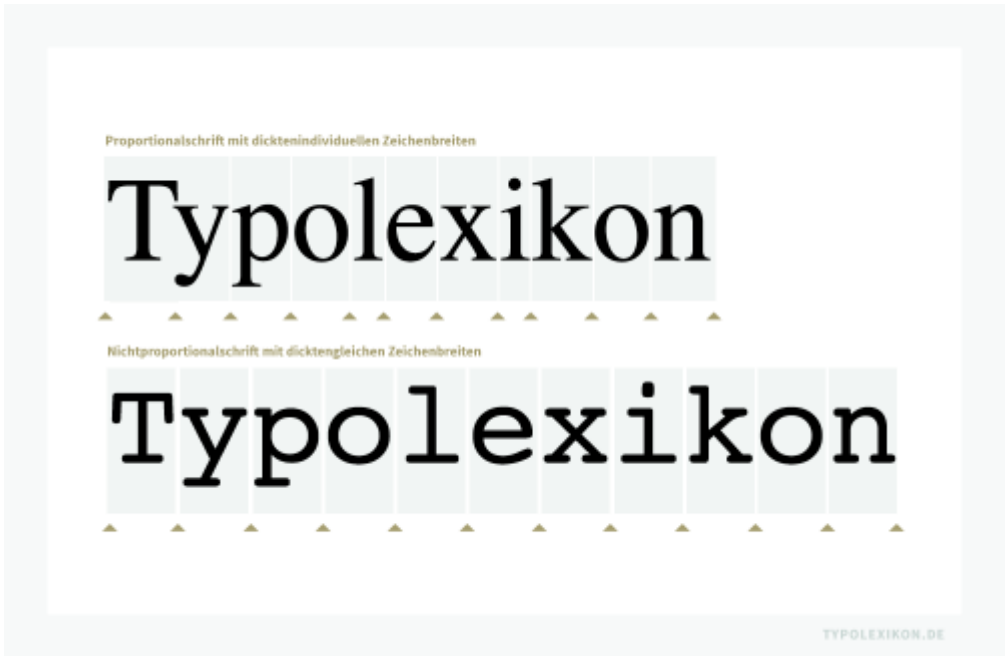
Laufweite

Die **Laufweite** einer Schrift beschreibt, wie groß die **horizontale Ausdehnung eines geschriebenen Textes** ist. Sie entsteht durch die Breite der einzelnen Buchstaben sowie den Abstand, den sie zueinander haben. Die Laufweite spielt **beim Schriftsetzen in Büchern, Zeitungen, Magazinen** usw. eine große Rolle, da sie bestimmt, wie „**ökonomisch**“ eine Schrift ist, das heißt, wie **platzsparend** sie ist. Um wertvollen Platz zu sparen, gibt es gerade für diesen Zweck entworfene Schmal-Schnitte einer Schrift (englisch „condensed“). Wesentlich seltener werden extra breite Schnitte verwendet (englisch „extended“), die meisten hiervon sind Grotesken (Sans Serifs). Der **Verwendungszweck von breiten Schnitten** ist weniger ökonomisch als **grafisch (für Überschriften, Plakate, Logos, Corporate Design usw.)**.



Proportionen

Im **Normalfall** sind die einzelnen **Zeichen einer Schriftart unterschiedlich breit**, ein „w“ nimmt also mehr Platz ein als ein „i“. Solche Schriftarten werden **proportional** genannt. Verbreitete Mitglieder dieser Gruppe sind zum Beispiel **Times**, **Arial** oder **Lucida**. Um die **Konstruktion von Schreibmaschinen und Computern zu vereinfachen**, kamen **nichtproportionale**, sogenannte **dicktengleiche Schriftarten zum Einsatz**, bei denen alle Zeichen eine identische Breite aufweisen. Die wohl bekanntesten dieser Schriften sind **Courier** und **Lucida Console**.



Glyphe

Der Begriff **Glyphe** wird gelegentlich für die **konkrete Darstellung eines Zeichens** verwendet. Die Schriftart bestimmt, welches Zeichen mit welcher Glyphe dargestellt wird.



Font

Ein **Font** ist die **Aufbereitung einer Schriftart für den Einsatz mit einem Computer**, also eine **Datei**, die eine **Schriftart beschreibt**.

Tastatur-Layout

Ein **Tastatur-Layout** oder eine **Tastaturbelegung** (auch Tastenlayout oder Tastenbelegung) **ordnet** einer Taste auf der **Tastatur einen Codepoint** zu.

Beispielsweise produziert die auf einer deutschen Tastatur mit „z“ beschriftete Taste mit einer deutschen Tastenbelegung den Unicode-Codepoint U+007A, also (erwartungsgemäß) ein „z“. Ändert man die Tastenbelegung auf US-amerikanisch, so erzeugt dieselbe Taste den Codepoint U+0079, also ein „y“. Mit einer russischen Belegung produzieren praktisch sämtliche Tasten andere Resultate, nämlich die Codepoints für kyrillische statt lateinischer Buchstaben, also z.B. „н“ (U+043D) statt „z“.

Textverarbeitungsprograme

Textverarbeitungsprogramme bieten im Gegensatz zu reinen Texteditoren in der Regel **erweiterte Layout- und Formatierungsfunktionen** an. Neben der Textüberarbeitung erhöhen **Rechtschreibprüfung, Indexerstellung, Such- und Ersetzfunktionen** den Nutzen für den Anwender. **Formatvorlagen** vereinfachen zudem eine einheitliche Gestaltung der zu veröffentlichenden Dateien, Textbausteine die Einbindung von wiederkehrenden Inhalten.

Zeichenorientierte Textverarbeitung

Die Kategorie der im Funktionsumfang erweiterten zeichenorientierten Textprogramme (**C**haracter **U**sed **I**nterface) verwendet ebenfalls, wie die ASCII-Editoren auch, den normierten und **beschränkten ASCII-Zeichensatz** als Grundlage. Die fest definierten Zeichensätze werden sowohl für die Bildschirmdarstellung als auch für den Ausdruck verwendet. Der **Abstand der Zeichen untereinander ist fest vorgegeben**, wie auch die Zeichen selbst statisch sind. Diese **„Statik“** bedeutet, dass alle **Zeichen des Zeichenvorrats vorgeformt** und **fertig zur Verfügung** stehen. Hier liegt der **große Vorteil der CUI-Programme** begründet: **die Arbeitsgeschwindigkeit**.
Da alle **Zeichen in Größe und Form fertig** vorliegen, bedarf es **keiner ständigen**

Neuberechnung von deren Bildschirmdarstellung. Die **Hardware-Anforderungen** beim Einsatz von CUI-Programmen sind entsprechend **gering**. Der dadurch erkaufte Nachteil ist jedoch die heute gewünschte Darstellungsqualität, die fortwährende Ansicht als Seitenvorschau.

Grafisch orientierte Textverarbeitung

Die grafisch orientierten Programme basieren auf dem **GUI, dem Graphical User Interface**. Alle **Zeichen** sind in **Form und Größe variabel**. Am Bildschirm können also **Formatierungen und Schriftgrößen exakt so angezeigt werden**, wie sie formatiert wurden und später auch auf dem **Ausdruck** erscheinen. Ermöglicht wird dies durch die **getrennte Handhabung von Schriftzeichen für den späteren Ausdruck** zum einen und **Bildschirmschriften** zum anderen. Der Preis hierfür ist jedoch ein **enormer Rechenaufwand** für die ständige **Aktualisierung und Neuberechnung der Bildschirmanzeige**. **Wichtig** geworden sind deshalb neben der reinen **Prozessorleistung** des Rechners Komponenten wie die **Grafikkarte** und der Bildschirmspeicher. Das Druckergebnis soll also exakt dem entsprechen, was auf dem Bildschirm zu sehen ist **(WYSIWYG, What You See Is What You Get)**. Oder anders ausgedrückt: **Alles, was und wie man es auf dem Bildschirm sieht, soll auch genauso ausgedruckt werden.**

Dateiformate

Jedes **Textverarbeitungsprogramm** hat bis jetzt noch **sein eigenes Dateiformat** für die Speicherung von Dokumenten. Eine gewisse **Standardisierung** ist durch das **Rich Text Format** erreicht worden, mit dem aber die **Einheitlichkeit des Layouts auf verschiedenen Rechnern nicht gewährleistet** ist.

Eines der am weitesten verbreiteten Dokumentenformate zum Austausch von Dokumenten, die nicht zur weiteren Bearbeitung vorgesehen sind, ist **PDF**. Dieses wird als **ergänzendes Standardformat** von immer mehr Textverarbeitungsprogrammen oder **als separater virtueller Druckertreiber als Exportfunktion** (d. h. zum Speichern) angeboten, kann unter den meisten gängigen Betriebssystemen (darunter auch einige ältere Versionen) eingelesen werden und **behält dabei das Aussehen**.
Nachteil: Man kann das **PDF-Dokument nicht mehr bearbeiten**.

Klassifikation von Textverarbeitungsprogrammen

Textverarbeitungsprogramme können nach unterschiedlichen Kriterien voneinander unterschieden und klassifiziert werden:

- Komplexität (vom einfachen Texteditor bis zur hochkomplexen Office-Software)
- Verwendungszweck (Textverarbeitungsprogramme für Briefe, Verfassen von Quellcode z.B. notepad++)
- Plattformunterstützung (Linux,Windows,iOS,Android,...)
- Lizenzierung (Freeware oder Commercial)

Kritik

Das Verfassen am Bildschirm könne dazu führen, dass man einen **geringen Überblick** über den Text habe.
Texte würden **nicht mehr selbst erstellt und durchdrungen (und auch kaum noch gelesen)**, sondern mittels **Kopieren und Einfügen** zusammengesetzt. Dieses Phänomen zeige sich besonders bei **ergoogelten Plagiaten**

7.2) Praxis mit MS Word

- Grundlagen
- Kopf und Fußzeile
- Formatvorlagen
- Tabulatoren
- Serienbrief

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:07

Last update: **2019/05/14 20:51**

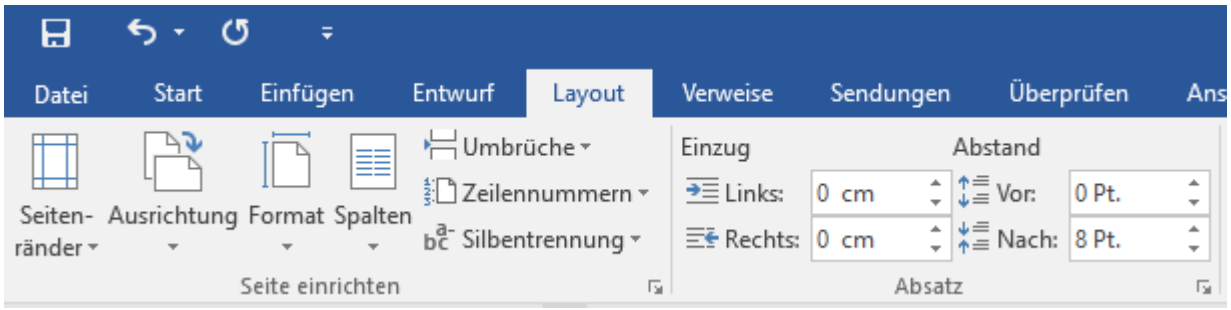
Textverarbeitung - Grundlagen

Layout

Seite einrichten

Mit dem Seitenformat definiert man eine Formatierung, die (meist) für das ganze Dokument gilt.

- Papierformat (im Menü Datei → Seite einrichten → Layout zu finden)
- Hoch- und Querformat (im Menü Datei → Seite einrichten → Seitenränder zu finden)
- Randeinstellungen (im Menü Datei → Seite einrichten findest du die wichtigsten Einstellungen)



Wichtige Tastenkombinationen

- **STRG + C** ... markierten Text kopieren
- **STRG + V** ... markierten Text einfügen
- **STRG + X** ... markierten Text ausschneiden

- **STRG + Mausrade** ... Einstellen der angezeigten Seitenfläche
- **Maus auf Blattkante → Symbol mit zwei Pfeilen** ... Leerfläche aus- bzw. einblenden

- **verkehrtes Pi-Symbol** ... Sichtbarmachen von Steuerzeichen (zB. Absatzende, Umbrüche,...)

Markierungen

Prinzipiell mit der Maus möglich, aber oftmals sind Shortcuts ganz nützlich!

- Markiere ein ganzes Dokument ⇒ **STRG+A**
- Markiere ein ganzes Wort ⇒ Doppelklick mit der Maus auf das Wort oder **STRG+SHIFT+Pfeiltaste rechts** bzw. **Pfeiltaste links**
- Markiere eine ganze Zeile ⇒ Mausklick links auf den Rand auf Höhe der gewünschten Zeile **STRG+SHIFT+ENDE**
- Markiere mehrere Zeilen untereinander ⇒ Mausklick links auf den Rand auf Höhe der gewünschten Zeile und nach unten oder oben ziehen **STRG+SHIFT+Pfeiltaste oben** oder **Pfeiltaste unten**

- Markiere mehrere einzelne Zeilen ⇒ Bei gedrückter **STRG + ein Mausklick links** auf die gewünschten Zeilen
- Markiere von der aktuellen Position alles bis zum Seitenanfang ⇒ **STRG+SHIFT+POS1**
- Markiere von der aktuellen Position alles bis zum Seitenende⇒ **STRG+SHIFT+ENDE**

Umbrüche

Enter ... Absatzende
Strg + Enter ... Seitenumbruch
Shift + Enter ... Zeilenumbruch
Einfügen→Manueller Umbruch ... Menü zum Einfügen verschiedener Umbrüche

Zeichenformate

Um die Schrift verändern zu können gibt es drei Möglichkeiten:

- Auf der **Symbolleiste FORMAT** findest du die gängigsten Optionen
- Im Menü **FORMAT → Zeichen** findest du,
- genauso wie mit **RECHTER MAUSTASTE → Schriftart**, eine größere Auswahl um deinen Text zu formatieren

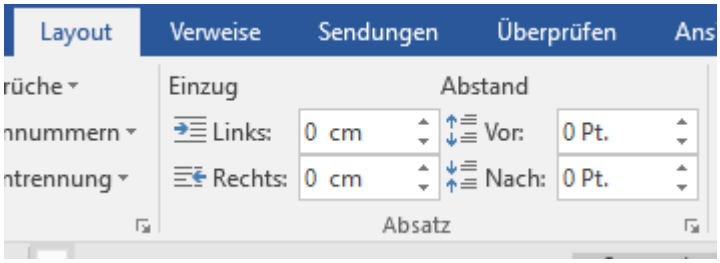
!! Der zuändernde Text muss dabei immer markiert sein !!

Textausrichtung und Absatzgestaltung

Die gewünschte Textausrichtung **links - rechts - zentriert - Blocksatz** kann mittels

- Symbolleiste FORMAT,
- im Menü FORMAT → Absatz oder
- mit RECHTER MAUSTASTE → Absatz eingestellt werden.

!! Der zuändernde Text muss dabei immer markiert sein !!

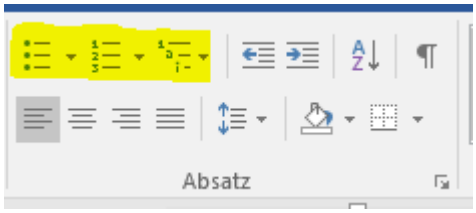


Aufzählung - Nummerierung

Ein Text kann durch Aufzählungszeichen und/oder Nummerierungen gegliedert werden.
Je nach Gliederungswunsch, kann:

- in der Symbolleiste FORMAT → Aufzählung / Nummerierung oder
- mit RECHTER MAUSTASTE → Aufzählung / Nummerierung eine Gliederung vorgenommen werden.

Dabei ist sowohl das Symbol bzw. die Nummerierung, sowie der Einzug frei wählbar.
Auch benutzerdefinierte Gliederung ist möglich!
!! Der zuändernde Text muss dabei immer markiert sein !!



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:07:07_01

Last update: **2019/05/10 14:32**

Kopf- und Fußzeile

Seitenzahl

Bei mehrseitigen Dokumenten möchte man manchmal unterschiedliche Kopf- und Fußzeilen für gerade und ungerade Seiten erstellen.
Unter `Kopfzeile bearbeiten`, bzw. `Fußzeile bearbeiten` gibt es die Möglichkeit, „Gerade und ungerade Seiten untersch.“ auszuwählen.
Weiters ist es möglich, die erste Kopf- bzw. Fußzeile anders zu gestalten.

Abschnittswechsel

Mithilfe von Abschnitten wird ein Dokument unterteilt. Die einzelnen Abschnitte eines Dokumentes nummeriert WORD fortlaufend. Abschnitte sind notwendig, wenn man innerhalb eines Dokumentes folgende Merkmale ändern will:

- Seitenränder, Seitenzahlen, Papierformat und Ausrichtung,
- Spaltenanzahl,
- Kopf- und Fußzeilen, sowie Fußnotennummerierung.

Mittels `Seitenlayout - Umbrüche - Nächste Seite` fügt man einen Abschnittsumbruch ein. Allerdings ist der neue Abschnitt mit dem vorigen verknüpft! Möchte man die Kopf- und Fußzeilen im neuen Abschnitt anders formatieren, so muss man im neuen Abschnitt die Verknüpfung aufheben unter: `Kopf- oder Fußzeile bearbeiten - Mit voriger verknüpfen`. Es ist darauf zu achten, dass `Mit voriger verknüpfen` **nicht** markiert ist!

Beispielsweise möchte man mit der Seitennummerierung erst ab Seite 4 beginnen. In diesem Fall muss zwischen Seite drei und Seite 4 ein Abschnittsumbruch eingefügt werden, sowie die Verknüpfung aufgehoben werden. Erst dann kann man die Kopf- und Fußzeilen der beiden Abschnitte unabhängig voneinander formatieren.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:07:07_02

Last update: **2019/05/10 14:33**

Formatvorlagen

Anwenden von Formatvorlagen

Formatvorlagen verleihen Ihrem Dokument ein einheitliches, professionelles Aussehen.

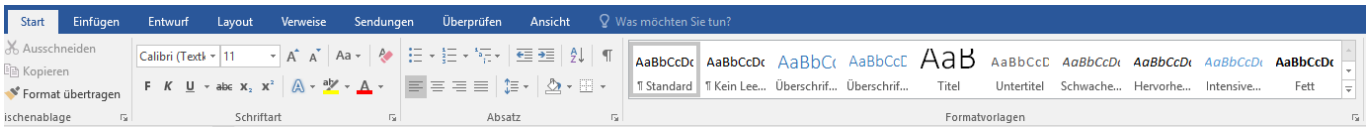
- **Markieren Sie den Text, den Sie formatieren möchten.**

Tipp: Wenn Sie den Cursor in einen Absatz setzen, wird die Formatvorlage auf den ganzen Absatz angewendet. Wenn Sie eine bestimmte Textpassage markieren, wird nur diese formatiert.

- **Zeigen Sie auf der Registerkarte Start auf eine Formatvorlage, um sich anzuschauen, wie der Text mit dieser Formatvorlage aussehen wird. Anwenden von Formatvorlagen-Optionen auf der Word-Multifunktionsleiste**

Tipp: Wenn die gewünschte Formatvorlage nicht angezeigt wird, klicken Sie auf die Schaltfläche Weitere Schaltfläche „Weitere“ , um den Katalog zu erweitern.

- **Wählen Sie eine Formatvorlage aus.**



Nachdem Sie in ihrem Textdokument die Überschriften mit einer oder mehreren Formatvorlagen formatiert haben, können Sie daraus automatisch ein Inhaltsverzeichnis generieren.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:07:07_03

Last update: **2019/05/10 14:35**

Tabstopps

Mithilfe von Tabstopps können Sie einfach zu formatierende Dokumente erstellen. Mit den Optionen für das Dokumentlayout in Word können Sie beispielsweise ein Inhaltsverzeichnis oder einen Index erstellen, ohne einen einzigen Tabstopp setzen zu müssen.

Word stellt zudem vordefinierte Tabellen, Kopf- und Fußzeilen, Deckblätter und weitere Seitenlayoutoptionen bereit. Mit diesen Optionen erübrigt sich das Hinzufügen von Tabstopps.

Grundlegendes

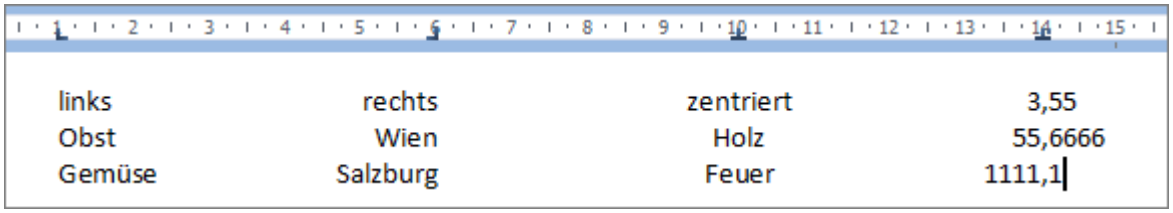
Ganz gleich, welche Version von Word Sie verwenden, das Verfahren zum Setzen (Hinzufügen) von Tabstopps auf dem horizontalen Lineal ist immer dasselbe. Standardmäßig befinden sich keine Tabstopps auf dem Lineal, wenn Sie ein neues leeres Dokument öffnen. Es gibt jedoch eine standardmäßige Tabstoppweite, die auf jeweils 1,25 cm festgelegt ist.

Bevor Sie mit dem Setzen (Hinzufügen) von Tabstopps beginnen, möchten Sie sich möglicherweise mit der Tabstoppauswahl und den verfügbaren Tabstopps vertraut machen.



Die Tabstoppauswahl ist das kleine Feld am äußerst linken Rand des Lineals. Sie können auf die Tabstoppauswahl klicken, um jeden der im Folgenden beschriebenen Tabstopps auszuprobieren.

Tabstopp links	Der Text wird rechts des Tabstopps ausgerichtet.
Tabstopp zentriert	Der Text wird während der Eingabe zentriert.
Tabstopp rechts	Der Text wird rechtsbündig ausgerichtet und nach links erweitert.
Tabstopp dezimal	Zahlen werden links des Dezimaltrennzeichens ausgerichtet. Sie können den **Tabstopp dezimal** nicht verwenden, um Zahlen um ein anderes Zeichen wie einen Bindestrich oder ein kaufmännisches Und-Zeichen auszurichten.
Leiste-Tabstopp	Text kann nicht mit einem **Leiste-Tabstopp** positioniert werden. Stattdessen wird eine vertikale Leiste an der Tabstopp-Position eingefügt.
Erstzeileneinzug	Die Option **Erstzeileneinzug** befindet sich ganz oben im Lineal und ermöglicht Ihnen, den Text der ersten Zeilen eines Absatzes einzuziehen.
Hängender Einzug	Die Option **Hängender Einzug** befindet sich unten im Lineal und ermöglicht Ihnen, die zweite Zeile des Absatzes und alle nachfolgenden Zeilen einzuziehen.



links	rechts	zentriert	3,55
Obst	Wien	Holz	55,6666
Gemüse	Salzburg	Feuer	1111,1

Setzen von Tabstopps auf dem Lineal

Eine schnelle Methode zum Setzen eines Tabstopps in Ihrem Dokument besteht darin, einen Tabstopp auszuwählen und im Lineal an die Position zu klicken, an der der Tabstopp erscheinen soll.



Klicken Sie am äußerst linken Rand des Lineals in der Tabstoppauswahl auf den gewünschten Tabstopp, und klicken Sie dann im Lineal auf die Position, an der der Tabstopp gesetzt werden soll.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:07:07_04

Last update: **2019/05/10 14:35**

Serienbriefe

Ein Serienbrief ist ein Schreiben mit gleich bleibendem Text, welcher automatisch mit unterschiedlichen Anschriften bzw. Anreden versehen wird.
Elemente:
Datenquelle ←——> Hauptdokument
Datenquelle: Sammlung von Adressen und personenbezogenen Angaben (Anrede, Geschlecht, Geburtsdatum etc.) (Excel, Word, Access-Datei)
Hauptdokument: allgemein gehaltener Text, der mit den Daten gefüllt wird.

1. Datenquelle erstellen (Excel-Tabelle, etc.)
2. Word: Sendungen – Seriendruck starten
    - Empfänger auswählen – Vorhandene Liste…
    - Brief schreiben – Seriendruckfelder nach Bedarf einfügen.
    - Regeln – Wenn-Dann-Sonst…
    - Alt+F9 → Formelansicht (erweitertes Bearbeiten)
    - Strg+F9 → geschwungene Klammern einfügen!

- Serienbriefe Übungen 1
- Serienbriefe Übungen 2

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:07:07_05

Last update: **2019/05/10 14:39**

8) Tabellenkalkulation

8.1) Theorie

Eine Tabellenkalkulation ist eine Software für die interaktive Eingabe und Verarbeitung von numerischen und alphanumerischen Daten in Form einer Tabelle. Vielfach erlaubt sie zusätzlich die grafische Darstellung der Ergebnisse in verschiedenen Anzeigeformen.

Das Bildschirmfenster der Software ist dabei in Zeilen und Spalten eingeteilt. Je nach Programm bzw. Bedienungskonzept heißt dieser Bereich zum Beispiel Arbeitsblatt, Worksheet oder Spreadsheet. Jede Zelle der Tabelle kann eine Konstante (Zahl, Text, Datum, Uhrzeit …) oder eine Formel enthalten. Für die Formeln stehen meist zahlreiche Bibliotheksfunktionen zur Verfügung. Die Formeln können Werte aus anderen Zellen benutzen.

Bei Änderung der referenzierten Zellen einer Formel aktualisiert die Software den erst angezeigten Wert der Formelzelle normalerweise automatisch, ggf. aber auch nur auf Anforderung. Werden Formeln eines Tabellenfeldes an andere Stellen kopiert, muss zwischen absolutem und relativem Zellbezug unterschieden werden. Formelzellen können auf andere Formelzellen verweisen. Mit diesem Prinzip können komplizierte Rechengänge mit vielen verknüpften Teil-Ergebnissen übersichtlich dargestellt werden.

Wozu?

Die Erfindung der elektronischen Tabellen zusammen mit Textverarbeitungs-Software und Datenbanken war zweifellos ein wesentlicher Faktor im Menschen den Wert von Mikrocomputern in den Anfangsjahren nach Apple und IBM begann Vermarktung Personalcomputer zu überzeugen. Seit dieser Zeit haben die stetig wachsende Vielseitigkeit und weiteren Anwendungen der Tabellenkalkulations-Software es in ein Produkt gemacht, die scheint fast unentbehrlich für Unternehmen und Privatanwender. Tabellen sind jetzt ein Standardbestandteil der Office-Suite-Pakete.

Funktionen

Kalkulationstabellen können mehr als einfache arithmetische Berechnungen durchführen. Eine Tabellenkalkulation übersetzen komplizierte Daten und Berichte in eine Kombination von Zahlen und Grafiken. Moderne Versionen enthalten eine umfangreiche Liste von finanzielle Rechner, z. B. Interesse Berechnungen, Kreditberechnung, auch Berechnungen für Schatzwechsel Preise. Statistische Funktionen von üblichen Berechnungen (Chi-Quadrat, Pearson-Koeffizient der Korrelation und Standardabweichung) auf abstruse Funktionen wie die hypergeometrische Verteilung und die Poisson-Verteilung Rückgabewerte benötigten mit keinen Schmerz. Es gibt mindestens 100 dieser Formeln in zeitgenössischen Tabellen enthalten.

Tabellen können als einfachen Datenbanken funktionieren. Durch das Einfügen von Daten und Anzahl in verschiedenen Spalten, können die Ergebnisse sortieren, gesucht oder gefiltert. Was-wäre-wenn-Analyse läuft, wenn Daten aus Zellen in einem Arbeitsblatt. Es gibt auch mathematische und trigonometrische Funktionen. Ein Benutzer kann eine vereinfachte Mailing-Liste in einer

Kalkulationstabelle erstellen, durch die Eingabe von Namen und Adressen in einzelnen Spalten. Es gibt sogar Funktionen der Textverarbeitung bietet Kontrolle über Schriftarten, Fett oder kursiv-Schrift, Größe, Farbe und Seitenformatierung.

Bei Änderung der referenzierten Zellen einer Formel aktualisiert die Software den erst angezeigten Wert der Formelzelle normalerweise automatisch, ggf. aber auch nur auf Anforderung. Werden Formeln eines Tabellenfeldes an andere Stellen kopiert, muss zwischen absolutem und relativem Zellbezug unterschieden werden. Formelzellen können auf andere Formelzellen verweisen. Mit diesem Prinzip können komplizierte Rechengänge mit vielen verknüpften Teil-Ergebnissen übersichtlich dargestellt werden.

Verschiedene Arten des Einsatzes von Tabellenkalkulationsprogrammen können sein:

- Automatisches Ausführen und Aktualisieren von Berechnungen
- Datenspeicherung & Datenverwaltung
- Automatisches erstellen und aktualisieren von Diagrammen
- Modellierung
- Programmierung (Makros, Visual Basic)

Einsatzmöglichkeiten

Vor allem aus dem kaufmännischen Bereich ist sie nicht mehr wegzudenken, wie ein Auszug aus den Anwendungsfeldern in der mittleren Spalte zeigt. Aber auch im wissenschaftlichen Bereich hat Tabellenkalkulation ihren Platz, insbesondere dann, wenn ein spezifischer Funktionsvorrat für Standardaufgaben aus den jeweiligen Bereichen angeboten wird (zB Statistikfunktionen, finanzmathematische Funktionen)

privat	**betrieblich**	**wissenschaftlich**
Haushaltsbudget	Kalkulationen	Statistik
Angebotsvergleich	einfache Kostenrechnung	Charts
Kostenvergleich	Reisekostenabrechnung	mathematische Modelle
Kreditkostenberechnung	Statisitk	kaufmännische Anwendungen
uvm…	Charts	volkswirtschaftliche Modelle
	Investitutionsrechnung	…uvm
	einfache Fakturierung	
	ABC-Analyse	
	Budgetierung	

Einschränkungen

Jedoch ist auch ein Tabellenkalkulationsprogramm keine „Eierlegende Wollmichsau“ und man stößt auch hier an Grenzen:

- Datenverwaltung wird schnell aufwendig ⇒ Lösung: Datenbanksystem verwenden
- Beschränkt in der Grösse ⇒ Lösung: Datenbanksystem verwenden
- Zusammenhänge sind nicht sichtbar ⇒ Lösung: Gute Dokumentation
- Gefahr von Nebenwirkungen (side effects) ist gross ⇒ Lösung: Sorgfältig arbeiten

8.2) Praxis

- Einführung
  - Übungen

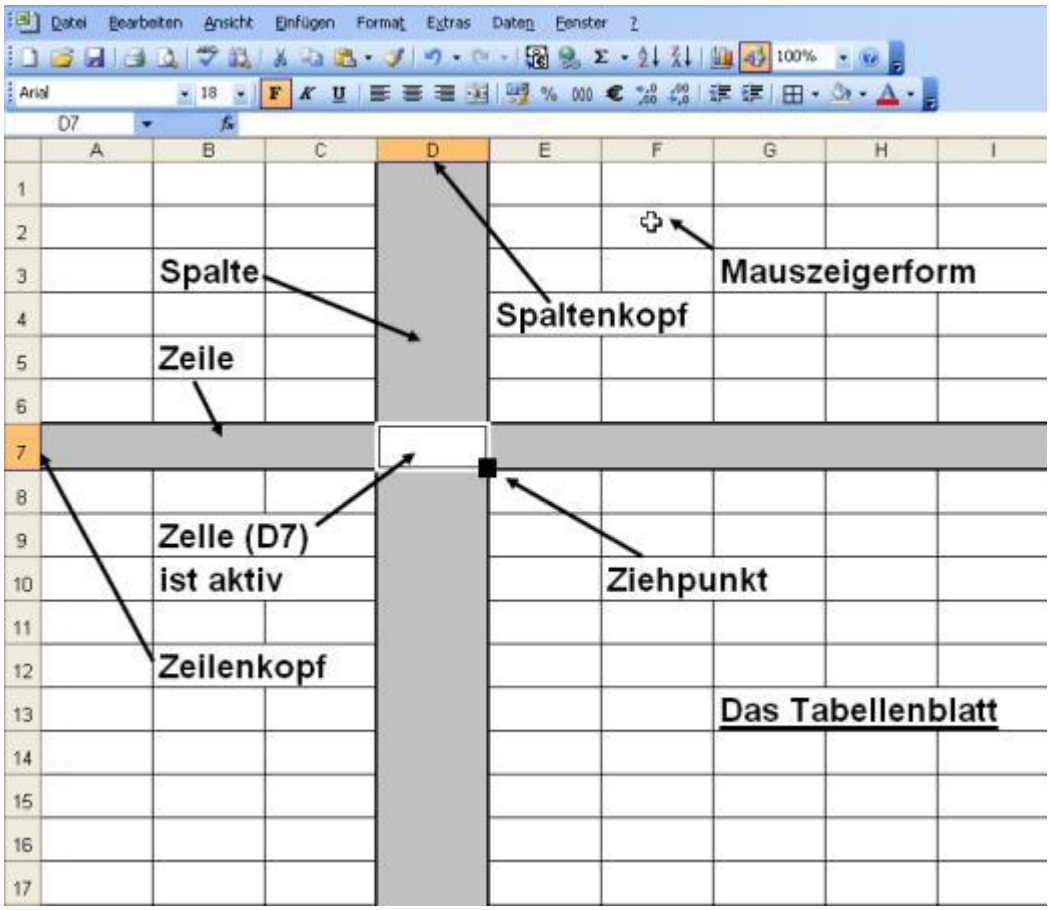
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:08

Last update: **2019/05/14 20:51**

Tabellenkalkulation - Einführung

Wichtige Begriffe



Zellenbezüge

- relativer Zellenbezug: A4; B12; KL77; AX500; ...
- absoluter Zellenbezug: \$A\$4; \$B\$12; \$KL\$77; \$AX\$500; ...
  - mittels 'F4' in Zelle hinzufügen.
- gemischter Zellenbezug: \$A4; B\$12; \$KL77; AX\$500; ...

Grundrechnungsarten

- Addition: =Zelle+Zelle
- Subtraktion: =Zelle-Zelle
- Multiplikation: =Zelle*Zelle
- Division: =Zelle/Zelle

Funktionen

SUMME

Die SUMME Funktion ist eine der meistgebräuchlichsten Funktionen in der Tabellenkalkulation, durch die gegebene Werte aufsummiert werden können. Dabei ist das Symbol ∑ hilfreich.

```
=SUMME(BEREICHSANFANG:BEREICHSENDE)

Beispiele:
  =SUMME(A1:A44)
  =SUMME(B2:D20)
```

MITTELWERT

Mit dieser Funktion wird der Mittelwert des gewünschten Bereiches ausgegeben.

```
=MITTELWERT(BEREICHSANFANG:BEREICHSENDE)

Beispiele:
  =MITTELWERT(A1:A44)
  =MITTELWERT(B2:D20)
```

MIN / MAX

Während mit MIN() der geringste Wert eines Bereiches ausgegeben wird, sucht MAX() nach dem größten Wert.

```
=MIN(BEREICHSANFANG:BEREICHSENDE)
=MAX(BEREICHSANFANG:BEREICHSENDE)

Beispiele:
  =MIN(A1:A44)
  =MAX(B2:D20)
```

ANZAHL / ANZAHL2

Beide Funktionen berechen, wie viele Werte in einem Bereich enthalten sind. Mit ANZAHL2() werden jedoch nur jene ausgegeben, die auch tatsächlich Werte enthalten.

```
=ANZAHL(BEREICHSANFANG:BEREICHSENDE)
=ANZAHL2(BEREICHSANFANG:BEREICHSENDE)

Beispiele:
  =ANZAHL(A1:A44)
```

```
=ANZAHL2(B2:D20)
```

WENN

Hierbei können Ausgaben an Bedingungen geknüpft werden. WENN das eintritt, DANN soll das passieren, SONST das andere. Deshalb wird sie auch als „Wenn-Dann-Sonst“ Funktion bezeichnet. Diese Funktion kommt einer IF Schleife in C++, oder anderen Programmiersprachen sehr nahe.

```
=WENN(BEDINGUNG;DANN;SONST)

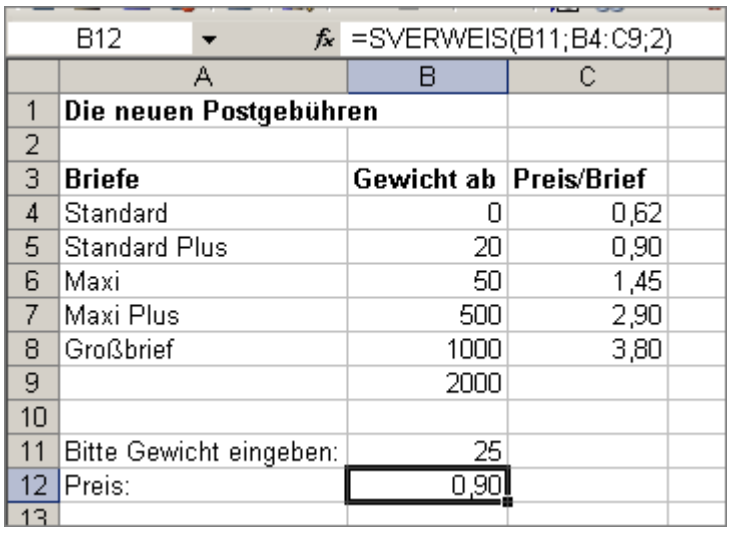
Beispiel:
Wenn die Höhe des Hauses kleiner als 20 Meter ist,
soll die Zelle HAUS ausgeben, sonst HOCHHAUS!
=WENN(B4<10;"HAUS";"HOCHHAUS")
```

SVERWEIS

Der SVERWEIS sucht in einem angegebenen Zellbereich nach einem passenden Wert und holt diesen in ein bestimmtes Feld.

```
=SVERWEIS(SUCHWERT;BEREICHSTABELLE;ZIELSPALTE)

Beispiel:
Suche in einer Posttariftabelle den richtigen Preis heraus
=SVERWEIS(B11;B5:C9;2)
```

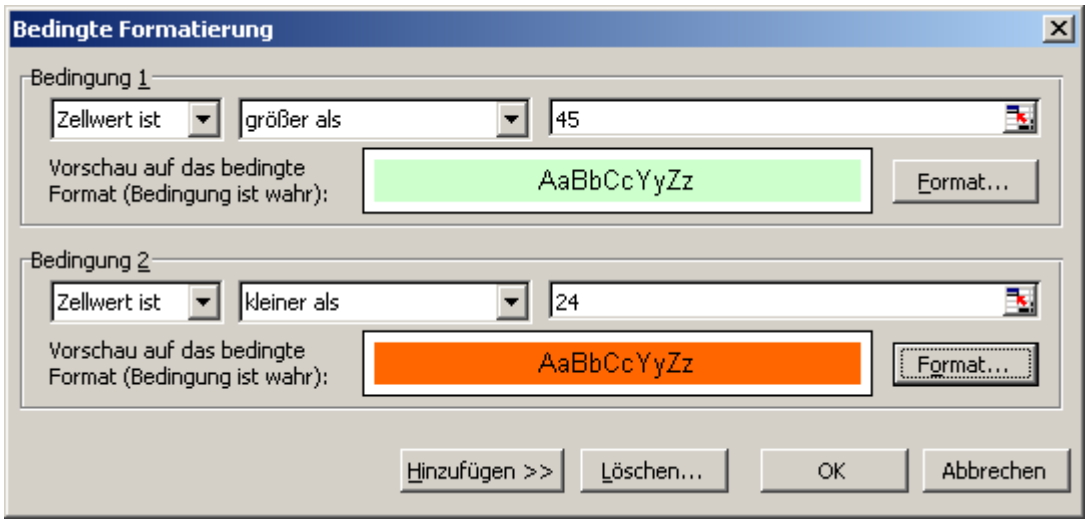


	A	B	C
1	**Die neuen Postgebühren**		
2			
3	**Briefe**	**Gewicht ab**	**Preis/Brief**
4	Standard	0	0,62
5	Standard Plus	20	0,90
6	Maxi	50	1,45
7	Maxi Plus	500	2,90
8	Großbrief	1000	3,80
9		2000	
10			
11	Bitte Gewicht eingeben:	25	
12	Preis:	0,90	
13			

Grundfunktionen

Bedingte Formatierung

Das Format einer Zelle lässt sich durch bedingte Formatierung je nach dem Wert in der Zelle verändern:



Sortieren

Mit Hilfe des Sortierens können die Tabelleninhalte in eine gewünschte Reihenfolge gebracht werden. Dabei darf nur derjenige Bereich makiert werden, der auch tatsächliche sortiert werden möchte! Danach auf:

```
DATEN - Sortieren - Sortieren nach
und das gewünschte Suchkriterium auswählen
```

Filtern

Mit dieser Funktion können Inhalte einer Tabelle herausgefiltert werden.
Erste Zeile eines Bereiches makieren und dann:

```
DATEN - Filter - AutoFilter
```

Inhalte einfügen

Mit dieser Funktion kann das Einfügen eines Textes zielführend beeinflusst werden. Einen Bereich makieren - kopieren - und an gewünschter Stelle:

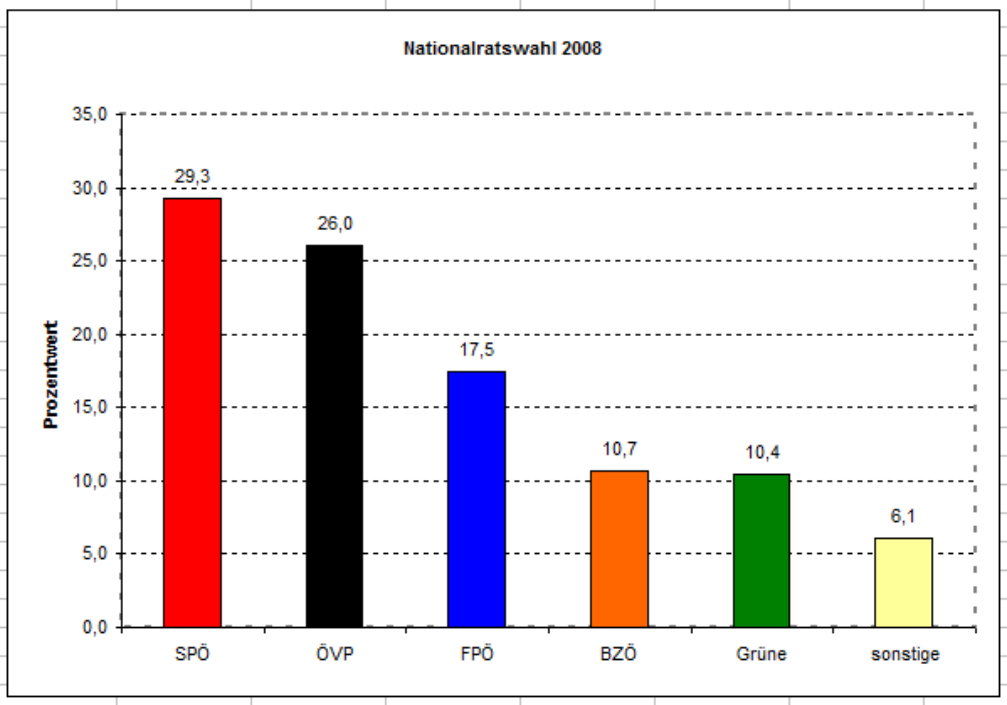
```
BEARBEITEN - Inhalte einfügen
und dort kann die gewünschte Einfügeoption gewählt werden
```

Diagramm

Erstellung eines Diagramms:

- Tabelle erstellen (auf die Wahl von Zelle und Spalte sollte geachtete werden)
- EINFÜGEN - Diagramm
- einen passenden und für den Zweck geeigneten Diagrammtyp auswählen
- den Datenbereich festlegen
- Einstellungen für den Titel, der Achsen, der Legende, ... vornehmen
- Ort zum Einfügen des Diagramms festlegen - FERTIG STELLEN
- Jetzt kann das Diagramm noch formatiert (Schriftgröße, Hintergrund, ...) werden, indem das zuändernde Objekt makiert wird und RECHTE MAUSTASTE gedrückt wird.

	A	B
1		
2	Parteien	Prozentwert
3	SPÖ	29,3
4	ÖVP	26,0
5	FPÖ	17,5
6	BZÖ	10,7
7	Grüne	10,4
8	sonstige	6,1



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:08:08_01

Last update: **2019/05/12 06:44**

Übungsaufgaben

1. Klassenübersicht

Gestalte ein Excelworksheet für einen Lehrer, der sich in Informatik nicht auskennt.
Der Lehrer hat Schularbeiten, Mitarbeitskontrollen und Hausübungen.
Schularbeiten haben immer 48 Punkte, Mitarbeitskontrollen können unterschiedliche Gesamtpunkte haben, Hausübungen haben 0 bis 5 Punkte.
Jeder dieser drei Bereiche wird auf einem eigenen Tabellenblatt dargestellt.
Der Mittelwert jeweils aller Schularbeiten, Mitarbeitskontrollen und Hausübungen wird im 48-Punkte-Schema ausgegeben.
Auf einem eigenen Tabellenblatt wird die Semesternote ermittelt. Sie berechnet sich wie folgt: 40% Schularbeiten, 30% Mitarbeitskontrollen und 30% Hausübungen.
Verwende die bedingte Formatierung: Eine negative Note soll einen roten Hintergrund erhalten, ein „Sehr gut“ soll grün hinterlegt werden.
Stelle die Notenübersicht mittels Balkendiagramm dar!

2. Excelübung 01

- excel_uebung01.rtf

3. Excelübung 02

- excel_uebung02.rtf

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:08:08_01:08_01_01

Last update: **2019/05/12 06:45**

2) DATENBANKEN

- 9.1) Allgemeines
- 9.2) Datenmodellierung
- 9.3) Entity Relationship Modell (Konzeptionelles Modell)
  - 9.3.1) Übungen
- 9.4) Relationenmodell (Logisches Modell)
  - 9.4.1) Übungen
- 9.5) Umsetzung ER-Modell --> Relationenmodell
  - 9.5.1) Übungen
- 9.6) Normalformen
- 9.7) SQL (Physisches Modell)
- 9.8) SQL Anbindung mit PHP
  - 9.8.1) Übung
  - SQL-ISLAND GAME - Schaffst du es den Piloten zu befreien?
  - SOLOLEARN - Play with SQL

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09

Last update: **2019/05/14 20:55**

9.1) Allgemeines

9.1.1) Definitionen

9.1.1.1) Datenbanksystem:

Ein Datenbanksystem ist ein computergestütztes System bestehend aus einer Datenbasis zur Beschreibung eines Ausschnitts der realen Welt sowie Programmen zum geregelten Zugriff auf die Datenbasis.

9.1.1.2) Datenbankverwaltungssystem (DBMS-data base managment system)

Ist jener Teil der Software die zwischen den eigentlichen Daten und den Benutzern der Daten liegt und alle Anfragen der Benutzer verarbeitet. Sie stellt jene Einrichtung zur Verfügung die notwendig sind um neue Daten anzulegen, zu löschen, abzufragen und zu verändern

9.1.2) Motivation

Die ersten Computer unterstützten Informationssysteme, wurden in Form von Einzellösungen, d.h. durch einzelne Anwendungsprogramme mit privaten Dateien realisiert. Diese Programme verwendeten unmittelbar das zugrunde liegende Dateisystem auf den jeweiligen Rechner. Gleichartige Daten wurden in separaten Dateien gespeichert, die selbst wieder aus einzelnen Datensätzen bestanden.

Produktion
Angestellte
Teile

Verkauf
Kunden
Teile

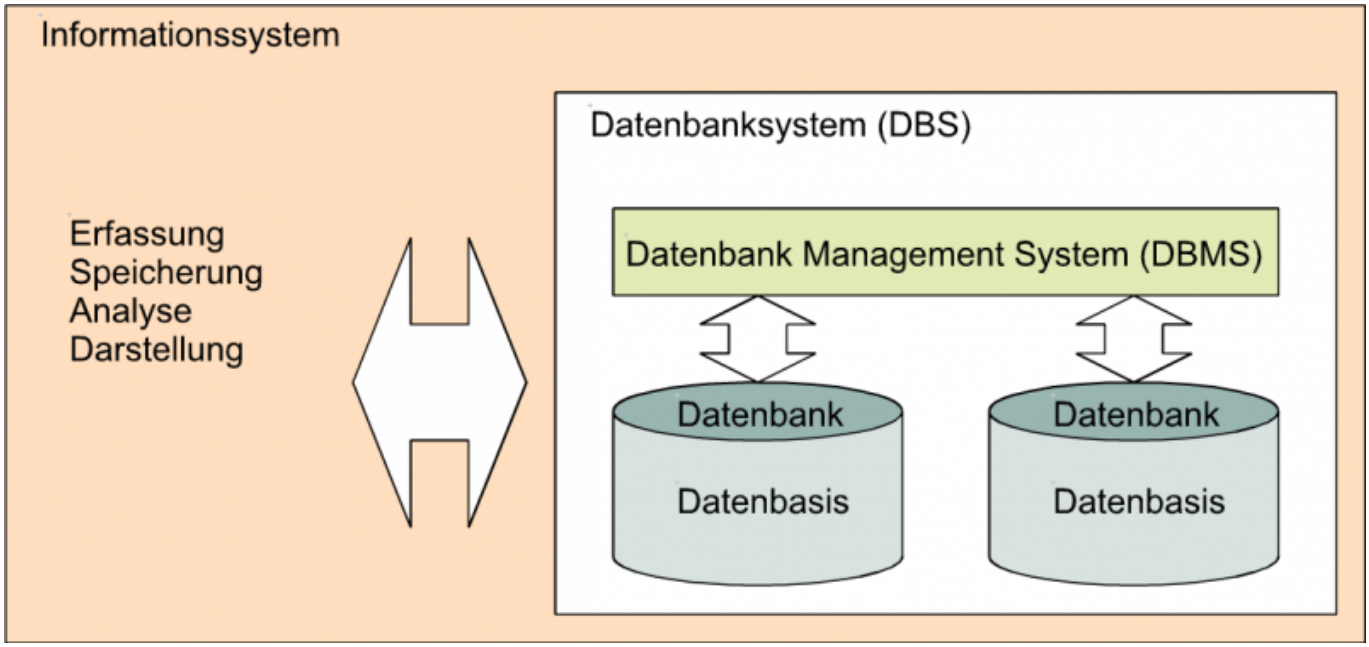
Fakturierung
Kunden
Teile

Diese Systeme waren schwer wartbar da mehrfach verwendete Daten auch mehrfach gespeichert wurden, deshalb entstand die integrierte Datenverarbeitung bei der Dateien in mehreren Anwendungsprogrammen verwendet werden. Doch auch die separate Abspeicherung von teilweise in Beziehung stehenden Daten würde zu schwerwiegenden Problemen führen:

- **Redundanz** - Dieselben Informationen werden doppelt gespeichert.
- **Inkonsistenz** - Dieselben Informationen werden in unterschiedlichen Versionen gespeichert.
- **Integritätsverletzung** - Die Einhaltung komplexer Intigritätsbedingungen fällt schwer.
- **Verknüpfungseinschränkung** - Logisch verwandte Daten sind schwer zu verknüpfen wenn sie

in isolierten Dateien liegen.
- **Mehrbenutzerprobleme** - Gleichzeitiges Editieren der Datei führt zur Anomalie (lost update).
- **Verlust von Daten** - Außer einem kompletten Backup ist kein Recovery-Mechanismus vorhanden.
- **Sicherheitsprobleme** - Abgestufte Zugriffsrechte können nicht implementiert werden.
- **Hohe Entwicklungskosten** - Für jedes Anwendungsprogramm müssen die Fragen zur Dateiverwaltung erneut gelöst werden.

Heute werden Informationssysteme meist mit Hilfe von Datenbanksystemen realisiert.



Einerseits ermöglicht diese Trennung zwischen Anwendungsprogrammen und Daten die sogenannte **physische Datenunabhängigkeit**, d.h. Programme sind von den konkreten Speicher- und Zugriffsmethoden unabhängig. Andererseits stellen Datenbanksysteme ein Datenmodell, also eine Sprache zur Beschreibung von Datenstrukturen, zur Verfügung, die es ermöglicht einzelne Programme auf speziellen logischen Darstellungen der gespeicherten Datenbank arbeiten zu lassen (logische Unabhängigkeit).

9.1.3) Funktionalität von Datenbanksystemen

9.1.3.1) Persistente Datenhaltung

Ein DBMS muss Mechanismen zur Verfügung stellen, die eine **persistente Speicherung von Daten** garantieren, d.h. dass die Daten in der DB über die Ausführungszeit von Programmen hinaus erhalten bleiben. Die Daten werden üblicherweise auf einem Hintergrundspeicher (Sekundärspeicher - HDD) persistent gehalten. Nachdem Programme nur auf Daten im Hauptspeicher (Primärspeicher) direkt zugreifen können werden Ausschnitte der Datenbank zeitweise auch in einem Teil des Hauptspeichers, dem Datenbankpuffer, verwaltet.
Nachdem ein **Plattenzugriff sehr viel länger dauert als ein Hauptspeicherzugriff**, sind spezielle Techniken notwendig um unnötige Plattenzugriffe zu vermeiden.
Spezielle Puffersatzstrategien werden verwendet um bei Platzmangel im Datenbankpuffer zu entscheiden welche Blöcke wieder auf die Platte ausgelagert werden (z.B. **least recently used, least frequently used**). Aus Effizienzgründen gruppieren sogenannten Clustertechniken Datensätze

so, dass jene Datensätze, auf die oft gemeinsam zugegriffen wird, physisch benachbart gespeichert werden. Weiters werden verschiedene Indextechniken verwendet um Daten auf einem Hintergrundspeicher rasch zu finden.

9.1.3.2) Recovery

DBMS unterstützen Änderungen in der Datenbank durch Transaktionen. Eine Transaktion ist eine Folge von Aktionen (Lese- und Schreibzugriffe auf Daten in der DB),die eine Datenbank von einem konsistenten Zustand in einen anderen konsistenten Zustand überführt. Die Recoveryeinheit eines **DBMS gewährleistet die Atomarität und die Dauerhaftigkeit (Persistenz) von Transaktionen** trotz eventuell bei der Transaktion aufgetretenen Hard- oder Softwarefehlern.

Atomarität bedeutet, dass **entweder alle Aktionen** einer Transaktion ausgeführt werden **oder keine**.

Dauerhaftigkeit bedeutet dass **alle Effekte** einer einmal erfolgten Transaktion **trotz aufgetretenen Fehlern erhalten bleiben**.

Bsp. zu Atomarität: Überweisung eines Geldbetrages von einem Konto auf ein Sparbuch

```
- Kontostand lesen
- Kontostand schreiben
- Sparbuch lesen
- Sparbuch schreiben
```

Angenommen während der Überweisung tritt nach der Abbuchung aber noch vor der Aufbuchung ein Systemabsturz ein, so möchten die Kunden davon ausgehen können, dass sich nach einem Wideranlauf der gesamte Geldbetrag noch auf dem Konto befindet.

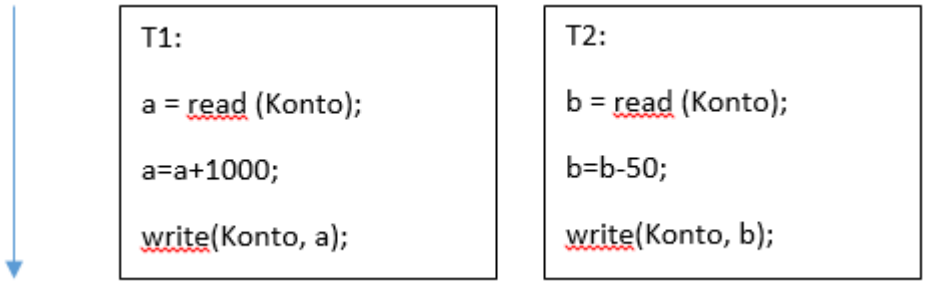
Bsp. zu Dauerhaftigkeit:
Angenommen wir zahlen einen Millionengewinn im Lotto auf unser Konto ein, dann stellt die Dauerhaftigkeit von Transaktionen sicher, dass der Gewinn auch nach einem Systemneustart immer noch auf dem Konto liegt. Für den Wiederanlauf verwenden die meisten DBS ein Log-Protokoll. In diesem Log-Protokoll werden der Start, das Ende und der Abbruch von Transaktionen verzeichnet, sowie die von Transaktionen durchgeführten Modifikationen von Datensätzen (Einfügen, Löschen und Ändern von Datensätzen). Zu jeder Änderung wird der alte Datensatz (before image) und der neue Datensatz (after image) im Log-Protokoll verzeichnet. Beim Wiederanlauf werden alle nicht beendeten Transaktionen unter Verwendung der before-images zurückgesetzt und alle bereits erfolgreich abgeschlossenen Transaktionen nachgeholt.

9.1.3.3) Concurrency Control

Die Concurrency Control - Einheit eines DBMS ermöglicht mehreren Benutzern eine Datenbank gemeinsam zur selben Zeit zu nutzen ohne ihre Konsistenz zu gefährden. Das traditionell verwendete **Korrektheitskriterium für parallele (oder verzahnte) Ausführung von Transaktionen im Mehrbenutzerbetrieb ist die Serialisierbarkeit**.
Die Serialisierbarkeit garantiert folgende Eigenschaft:
Das Ergebnis der beliebigen Parallelausführung mehrerer Transaktionen entspricht dem

Ergebnis irgendeiner Hintereinander-Ausführung dieser Transaktion.

Bsp.: Angenommen wir wollen unsere Telefonrechnung (50€) bezahlen. Wir gehen zur Bank, wo die Abbuchung vom Konto durchgeführt wird. Diese Abbuchung wird nun gleichzeitig mit der Gehaltsbuchung (1000€) auf das Konto durchgeführt.



Angenommen der aktuelle Kontostand beträgt 2000€. Nachdem beide Transaktionen abgeschlossen wurden, ist der Kontostand auf 1950€. Eine Hintereinanderausführung hätte aber 2950€ ergeben, d.h. diese verzahnte Ausführung ist nicht serialisierbar und daher nicht korrekt.

Um Serialisierbarkeit von Transaktionen zu gewährleisten verwenden die meisten DBMS Sperrverfahren.

Dabei legt eine Transaktion auf Datenobjekte die sie schreiben oder lesen soll, eine **Sperre**. Besitzt eine Transaktion auf einem Datenobjekt eine Sperre und fordert eine andere Transaktion für dieses Datenobjekt ebenfalls eine Sperre an, so wird diese Sperre nur dann gewährt, wenn die neu angeforderte Sperre mit der bereits bestehenden Sperre verträglich ist. Ist sie es nicht, so muss die neue Transaktion **auf die Freigabe der bestehenden Sperre warten**.

Meist werden 2 Typen von Sperren verwendet:

- Geteilte Lese-Sperren
- Exklusive Schreib-Sperren

Lesesperren verschiedener Transaktionen für dasselbe Datenobjekt sind miteinander verträglich, eine Schreibsperre ist mit keiner Sperre anderer Transaktionen verträglich.

Das Sperren von Datenobjekten ist alleine jedoch nicht ausreichend um die Serialisierbarkeit paralleler Transaktionen zu gewährleisten. Es muss darüber hinaus ein **Sperrprotokoll** eingehalten werden. Das am meisten gebräuchliche Sperrprotokoll ist das **2-Phasen-Sperrverfahren**. Eine Transaktion erfüllt das 2-Phasen-Sperrverfahren, wenn es nach der 1. Freigabe einer Sperre keine neue Sperre mehr anfordert. Weiters garantiert die Concurrency Control-Einheit eines DBMS die Isolation von Transaktionen. **Isolation** bedeutet, dass **Effekte nach einer Transaktion erst nach ihrem erfolgreichem Abschluss für andere Transaktionen sichtbar** werden.

9.1.3.4) ACID - Atomicity Consistency Isolation Durability (zu d. Dt. AKID - Atomarität, Konstistenz, Isolation, Dauerhaftigkeit)

Die Eigenschaften des Transaktionskonzeptes werden unter der Abkürzung **ACID** zusammengefasst:

- **Atomicity:** Eine Transaktion stellt eine nicht weiter zerlegbare Einheit dar, mit dem Prinzip:

„ALLES oder NICHTS“

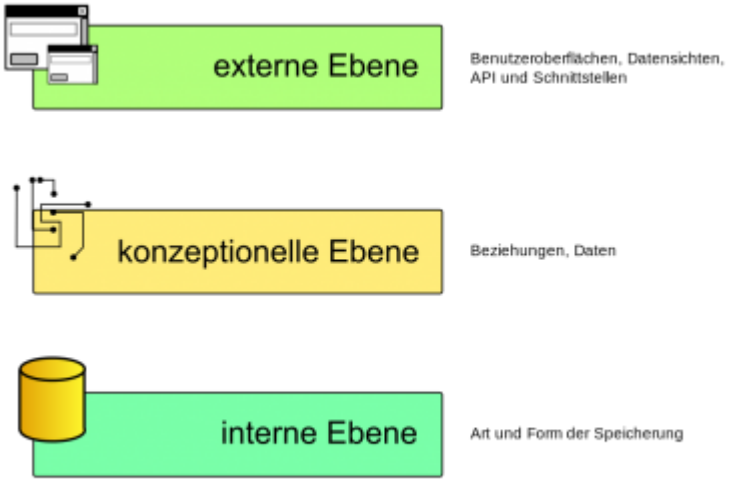
- **Consistency:** Nach Abschluss der Transaktion liegt wieder ein konsistenter Zustand vor, während der Transaktion sind inkonsistente Zustände erlaubt
- **Isolation:** Nebenläufig ausgeführte Transaktionen dürfen sich nicht beeinflussen, d.h. jede Transaktion hat den Effekt, den sie verursacht hätte, als ob sie allein im System gewesen wäre.
- **Durability:** Die Wirkung einer erfolgreich abgeschlossenen Transaktion bleibt dauerhaft in der Datenbank, auch nach einem späteren Systemfehler.

9.1.3.5) Datenschutz

DBMS bieten die Möglichkeit für einzelne Benutzer oder Benutzergruppen den Zugriff auf Ausschnitte der Datenbank zu beschränken. Dabei kann hinsichtlich der Art des Zugriffs zwischen Lese-, Änderungs-, Einfüge- und Löschzugriffe unterschieden werden.

9.1.4) Architektur eines Datenbanksystems

Moderne Datenbanksysteme unterstützen alle die **ANSI-SPARC-Architektur**:



Die Architektur wurde 1975 vom Standards Planning and Requirements Committee (SPARC) des American National Standards Institute (ANSI) entwickelt und hat das Ziel, den Benutzer einer Datenbank vor nachteiligen Auswirkungen von Änderungen in der Datenbankstruktur zu schützen.

Die drei Ebenen sind:

- **Die externe Ebene**, die den Benutzern und Anwendungen individuelle Benutzersichten bereitstellt. Beispiele: Formulare, Masken-Layouts, Listen, Schnittstellen.
- **Die konzeptionelle Ebene**, in der beschrieben wird, welche Daten in der Datenbank gespeichert sind, sowie deren Beziehungen zueinander. Designziel ist hier eine vollständige und redundanzfreie Darstellung aller zu speichernden Informationen. Hier findet die Normalisierung des relationalen Datenbankschemas statt.
- **Die interne Ebene (auch physische Ebene)**, die die physische Sicht der Datenbank im Computer darstellt. In ihr wird beschrieben, wie und wo die Daten in der Datenbank gespeichert werden. Designziel ist hier ein effizienter Zugriff auf die gespeicherten Informationen. Das wird

meistens nur durch eine bewusst in Kauf genommene Redundanz erreicht (z. B. im Index werden die gleichen Daten gespeichert, die auch schon in der Tabelle gespeichert sind).

Die Vorteile des Drei-Ebenen-Modells liegen in der

- **physischen Datenunabhängigkeit**, da die interne von der konzeptionellen und externen Ebene getrennt ist. Physische Änderungen, z. B. des Speichermediums oder des Datenbankprodukts, wirken sich nicht auf die konzeptionelle oder externe Ebene aus.
- **logischen Datenunabhängigkeit**, da die konzeptionelle und die externe Ebene getrennt sind. Dies bedeutet, dass Änderungen an der Datenbankstruktur (konzeptionelle Ebene) keine Auswirkungen auf die externe Ebene, also die Masken-Layouts, Listen und Schnittstellen haben.

Allgemein kann also von einer **höheren Robustheit gegenüber Änderungen** gesprochen werden.

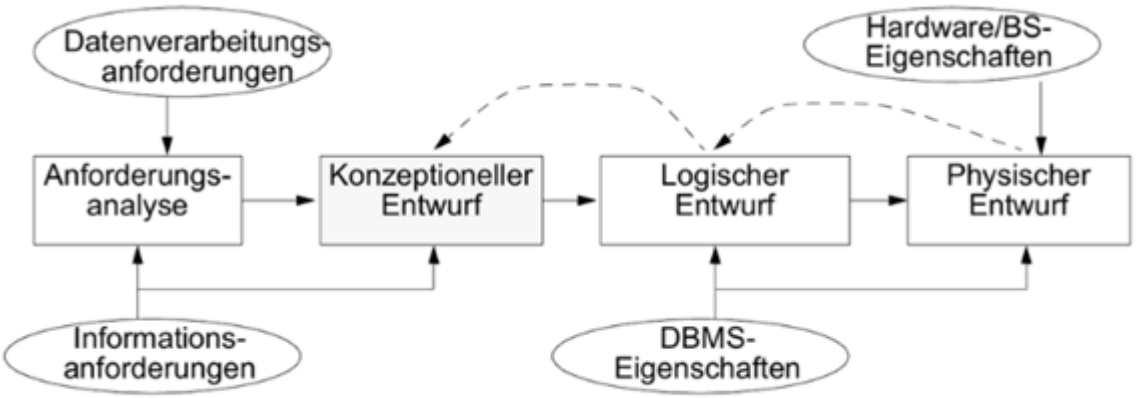
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

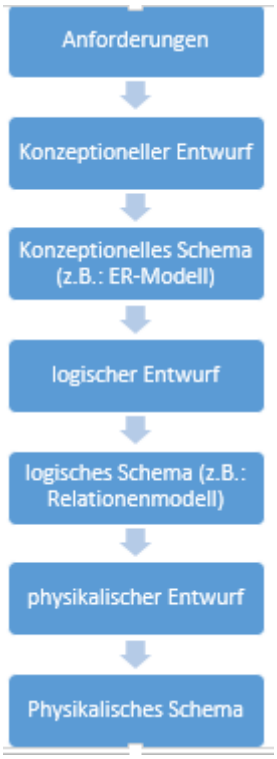
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_01

Last update: **2019/05/22 16:11**

9.2) Datenmodellierung

Im Datenbankentwurf werden verschiedene Datenbankmodelle verwendet:





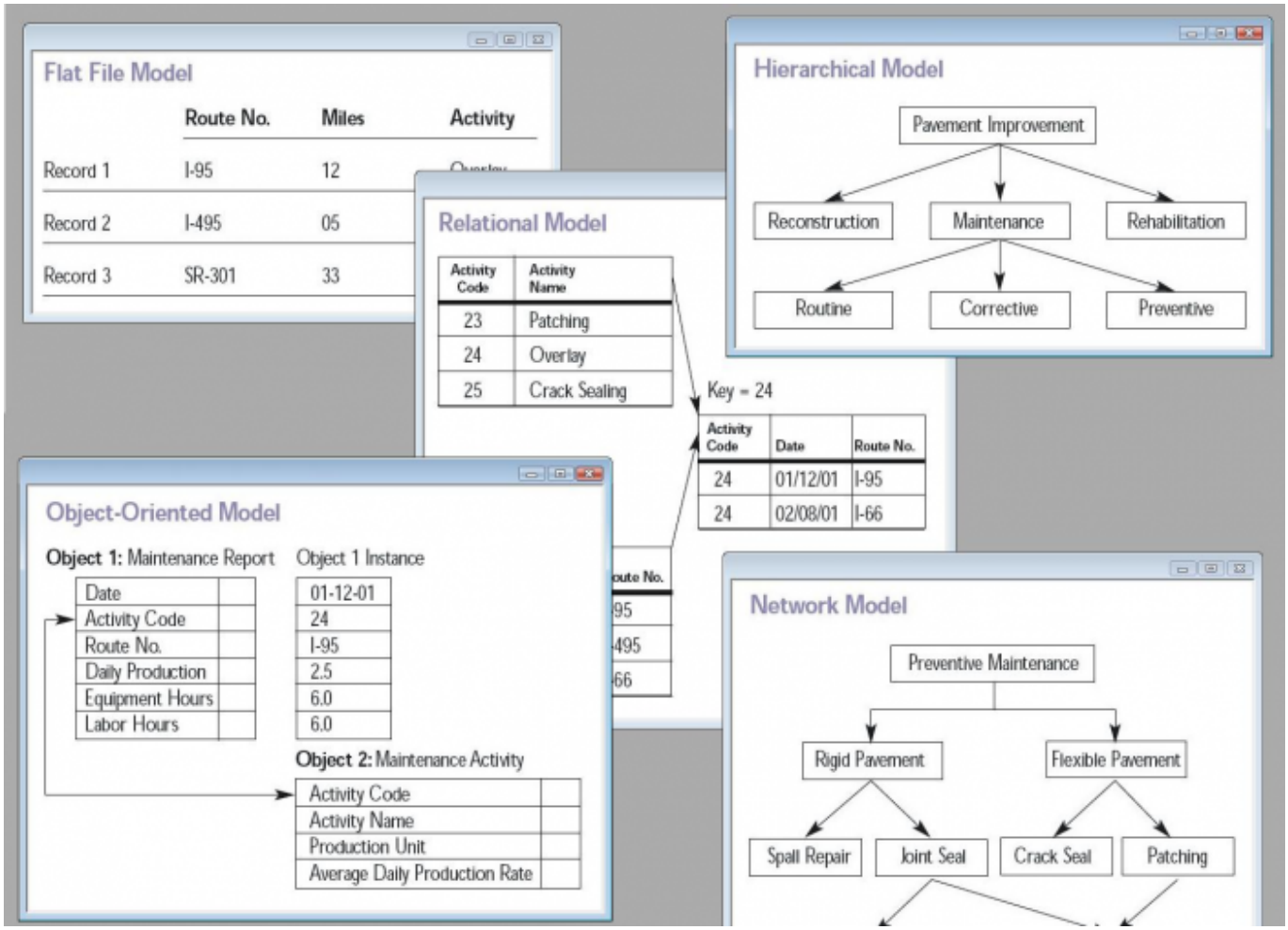
Konzeptionelle Datenmodelle (z.B.: ER-Modell) stehen **problemnahe Modellierungskonzepte** für die ersten Schritte des Datenbankentwurfs zur Verfügung und dienen zur **Kommunikation zwischen Endbenutzern und Datenbankentwicklern**.

Physische Modelle stellen **maschinennahe Konzepte** zur Verfügung und dienen zur **Beschreibung der Organisation von Daten in Dateien** sowie zur Beschreibung von **Zugriffsstrukturen** die ein rasches Einfügen, Suchen und Ändern von Daten ermöglichen.

Logische Datenmodelle dienen der **Überbrückung zwischen konzeptionellen und physischen Datenmodellen**. Sie werden oft auch als Implementierungsmodelle bezeichnet, das logische Datenmodell steht dem Entwickler zur Definition eines Datenbankschemas zur Verfügung und wird weitgehend automatisch vom DBMS in ein physisches Datenbankschema übersetzt.

Zur Formulierung des logischen Schemas stehen je nach zugrunde liegendem DBS folgende Möglichkeiten zur Wahl:

- Hierarchisches Modell
- Netzwerkmodell
- Relationales Modell
- Objektorientiertes Modell



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_02

Last update: **2019/05/22 17:26**

9.3) ENTITY-RELATIONSHIP-MODELL

Das ER-Modell wurde im Jahr 1976 von Chen eingeführt. In der Sicht des ER-Modells besteht die Welt aus Entities (Objekttypen) und Relationships (Beziehungen), zwischen den Objekten. Das ER-Modell wurde von Teorey im Jahr 1986 erweitert zum Extended ER-Modell (EER).

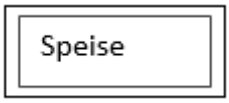
9.3.1) Grundlegende Objekte

Objekte bzw. Entity-Instanzen repräsentieren unterscheidbare Objekte des Problembereichs. Objekte mit den selben Eigenschaften werden zu Entities zusammen gefasst.

- **Entity**



- **Weak Entity**

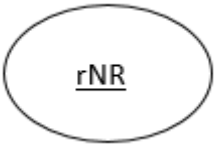


Beziehungen zwischen Objekten haben keine physische oder konzeptionelle Existenz sondern ergeben sich aufgrund der vorhandenen Objekte.



Attribute sind Eigenschaften der Entities. Es wird unterschieden zwischen:

- **identifizierende Attribute = (Schlüssel, Key)** ⇒ sind jene Attribute, die die Objekte einer Entity eindeutig kennzeichnen.



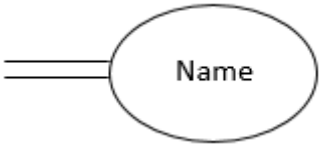
- **beschreibende Attribute** ⇒ sind jene Attribute, die ein Objekt nicht eindeutig kennzeichnen.



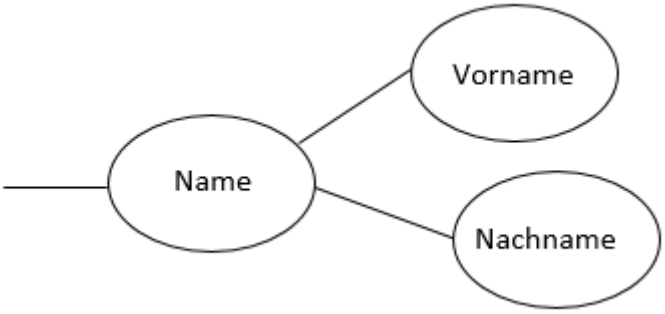
Schlüssel (Keys) können aus **einem oder mehreren Attributen** bestehen. Jede Entität kann mehrere verschiedene Schlüssel haben, wir stellen im ER-Modell aber immer **einen ausgezeichneten Schlüssel als Primärschlüssel (=Primary Key)** dar. Entitäten deren Objekte, nur mit Hilfe von anderen Entitäten identifiziert werden können, nennt man Weak Entities. Sie haben keine Attribute, die alleine den Schlüssel bilden können, obwohl sie dazu beisteuern können.

Weiters können Attribute mehrwertig sein oder komplexe Werte annehmen.

- **mehrwertige Attribute**



- **komplexe Attribute**

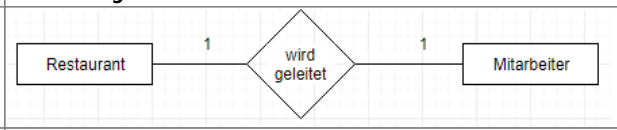
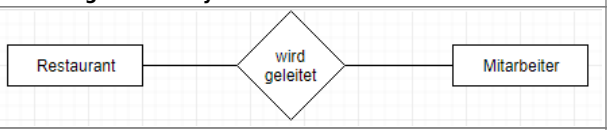
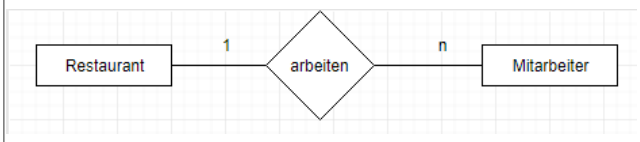
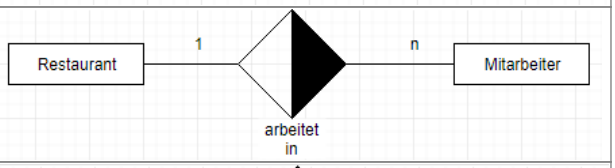
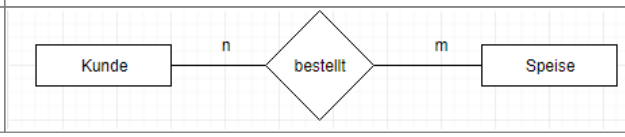
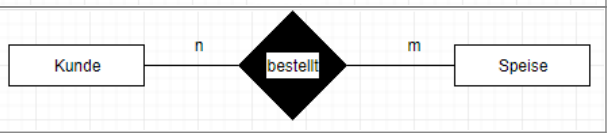


9.3.1.1) Komplexität von Beziehungen

Die Komplexität einer Beziehung zwischen Entities beschreibt mit vielen Instanzen, der jeweils anderen Entität jede Instanz einer Entität in Beziehung stehen kann. Bei einer Beziehung zwischen 2 Entitäten A und B gibt es folgende Möglichkeiten:

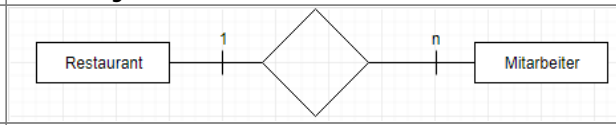
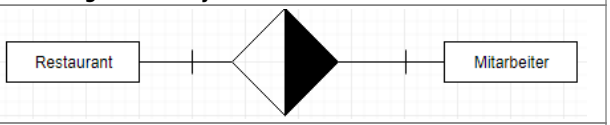
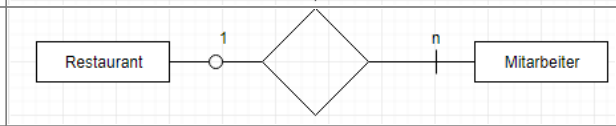
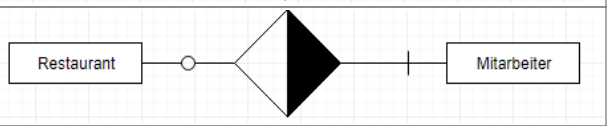
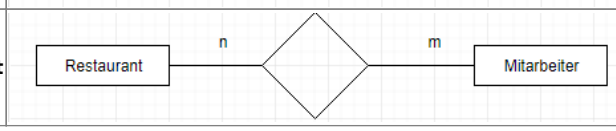
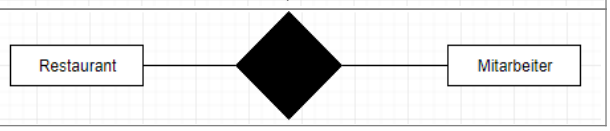
- **1:1** ⇒ Jedes Objekt von A gehört genau zu einem Objekt von B und umgekehrt
- **1:n** ⇒ Jedes Objekt von A gehört zu einem oder mehreren Objekten von B, aber jedes Objekt von B gehört genau zu einem Objekt von A.
- **n:m** ⇒ Jedes Objekt von A gehört zu einem oder mehreren Objekten von B und umgekehrt.

Die Darstellung einer Komplexität von Beziehung nach Chen geschieht durch Anbringung von Ziffern an die jeweiligen Entitäten. Dabei wird nur zwischen den Werten 1 und viele (n,m) unterschieden. Die genaue Anzahl der zugeordneten Objekte, d.h. die Kardinalität wird selten verwendet, da diese sich bei jeder Instanz einer Beziehung unterscheiden kann. Teory verwendet nicht mehr die Werte neben den Entitäten sondern färbt die Hälften der Beziehungen ein, die zu den n-Entitäten gedreht sind.

Konzept	Darstellung nach Chen	Darstellung nach Teory
1:1		
1:n		
n:m		

9.3.1.2) Existenz einer Entität in einer Beziehung

Die Kopmplexität einer Beziehung gibt an mit wie vielen Instanzen der anderen Entität in Beziehung stehen kann. Dabei haben wir immer von einer oder von vielen Instanzen gesprochen. Das Konzept der Existenz einer Entität gibt uns die Möglichkeit auszudrücken, ob es immer mindestens eine Instanz geben muss oder eine Instanz geben kann.

Konzept	Darstellung nach Chen	Darstellung nach Teory
zwingend		
optional		
unbekannt		

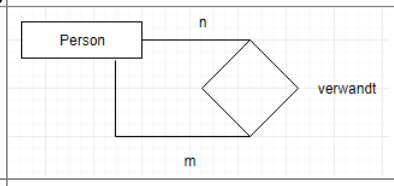
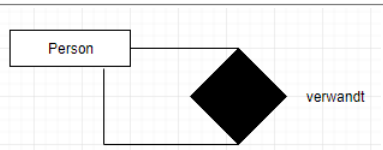
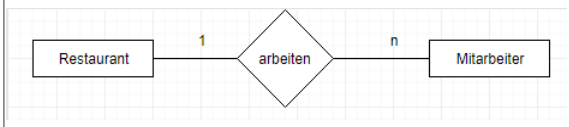
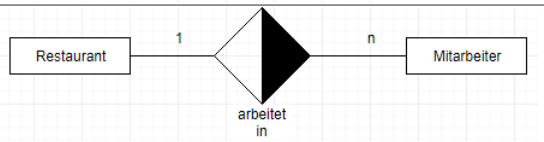
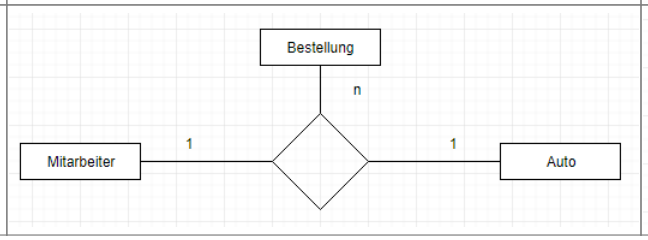
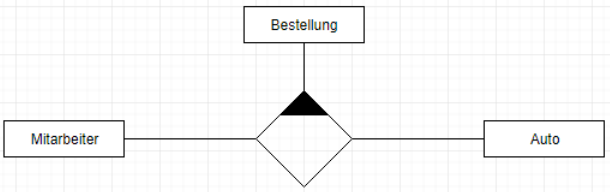
Bsp.:
zwingend ⇒ Ein Restaurant hat hat zumindest einen oder mehrere Mitarbeiter und jeder Mitarbeiter ist genau einem Restaurant zugeordnet.
optional ⇒ Ein Restaurant hat zwingend einen Geschäftsführer. Manche Mitarbeiter sind Geschäftsführer.

9.3.1.3) Grad einer Beziehung

Der **Grad einer Beziehung** ist die **Anzahl der Entitäten**, die in der **Beziehung miteinander verbunden** sind. **Unäre und binäre Beziehungen** kommen in der realen Welt **am häufigsten** vor.

Unäre Beziehungen (Beziehung einer Entität mit sich selbst) heißen auch binär rekursive Beziehungen.

Ternäre Beziehungen (Beziehungen zwischen 3 Entitäten) werden benötigt wenn binäre Beziehungen einen Sachverhalt nicht ausreichend beschreiben. Kann jedoch eine ternäre Beziehung mit 2 oder 3 binären Beziehungen ausgedrückt werden, so ist diese Darstellung vorzuziehen.

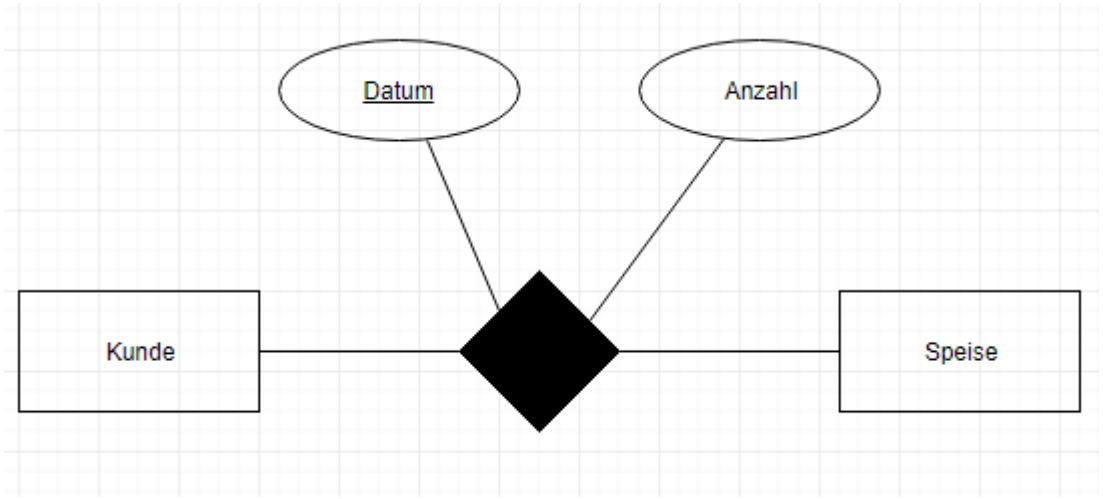
Grad der Beziehung	Darstellung nach Chen	Darstellung nach Teory
unär		
binär		
ternär		

Bsp.:
Für eine konkrete Pizzabestellung wird für die Lieferfirma ein Auto und ein Mitarbeiter abgestellt. Daher folgende ternäre Beziehung:

1. Ein Mitarbeiter verwendet für eine Bestellung ein Auto
9. Ein Auto wird bei einer Bestellung von einem Mitarbeiter gefahren.
3. Ein Mitarbeiter kann mit einem Auto mehrere Bestellungen erledigen

9.3.1.4) Attribute einer Beziehung

Attribute können wie wir bisher gesehen haben zu Entitäten hinzugefügt werden. Es gibt aber auch die Möglichkeit Attribute zu Beziehungen hinzuzufügen.
So könnte man die Attribute Anzahl und Datum zu Beziehung zwischen Kunde und Speise schreiben. Dadurch trägt man den Umstand Rechnung, dass ein Kunde eine Speise mehrmals bestellen kann. Würde man Attribut Anzahl zum Kunden hinzufügen würde man ein mehrwertiges Attribut erhalten und zudem die Information verlieren, welche Speise der Kunde zu welchem Datum bestellt hat.



Attribute werden üblicherweise nur zu n:m Beziehungen hinzugefügt, da im Falle von 1:1 oder 1:n zumindest auf einer der beiden Seiten der Beziehung ein einziges Objekt steht und damit die Mehrdeutigkeit kein Problem darstellt.

Wenn Attribute zu n:m Beziehungen geschrieben werden, so muss man überlegen, ob man nicht an Stellt von n:m Beziehungen eine Weak-Entität modelliert.
Denn jede n:m Beziehung entspricht einer Weak-Entität die durch eine 1:n Beziehung mit den beiden anderen Entitäten verbunden ist.

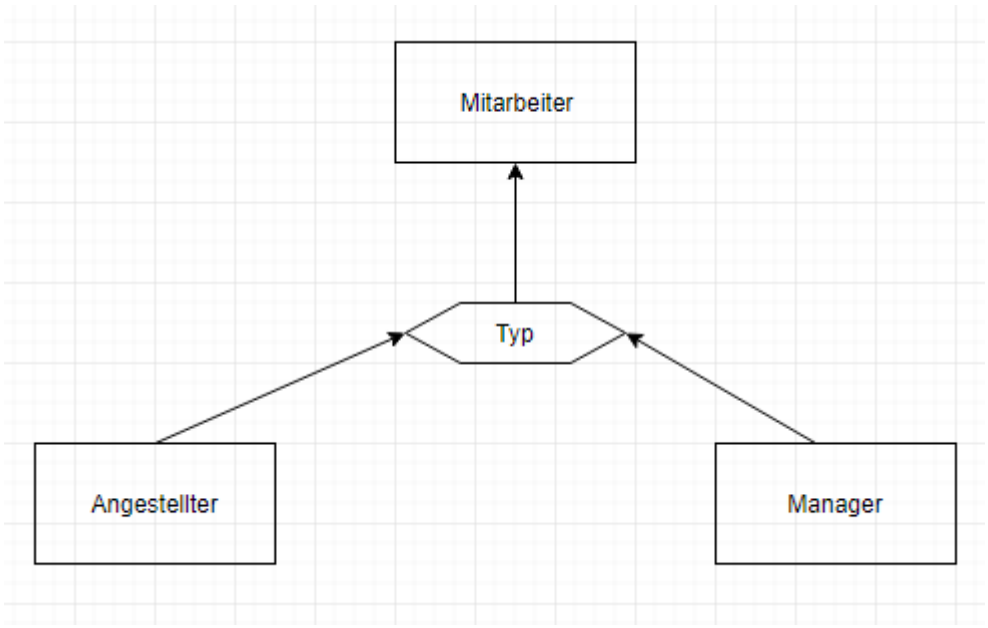
9.3.2) Erweiterte ER-Konstrukte: Die Generalisierung

Bei der Modellierung der Entities Mitarbeiter und Kunden wird man feststellen, dass sie viele gemeinsame Attribute besitzen. Diese können zu einer Enitity Person verallgemeinert werden. Mitarbeiter und Kunden werden jeweils mit einer 1:1 Beziehung zu Person verbunden. Die Attribute der Person werden entlang der Hierarchie vererbt und spezielle Attribute wie die SVNR der Mitarbeiter nur bei den speziellen Entities gespeichert.

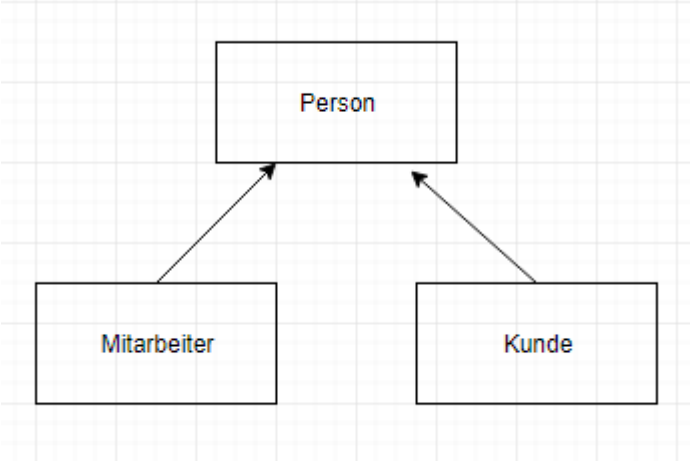
Die Generalisierung gibt an, dass mehrere Entities (Subtyp-Entity) mit bestehenden gemeinsame Attributen zu einer Entity auf einer höheren Ebene (Supertyp-Entity) generalisiert werden können.

Die Disjunktheit der Generalisierbarbeit beschreibt ob die einzelnen Subtyp-Entities **disjunkt** oder **überlappend** sind.

- **disjunkte Spezialisierung**: Entität kann zu maximal einem Untertyp gehören (Mitarbeiter ist entweder Angestellter oder Manager)



- **überlappende Spezialisierung**: Entität kann zu einem oder mehreren Untertypen gehören (Person kann ein Mitarbeiter oder/und eine Kunde sein)



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_03

Last update: **2019/05/14 20:54**

9.3.1) Übungen

9.3.1.1) Universität

Ein Institut hat einen eindeutige Nummer, einen Namen und eine Adresse. Ein Lektor identifiziert sich anhand seiner Sozialversicherungsnummer, und hat einen Namen. Er ist genau einem Institut zugeordnet, ein Institut kann keine oder mehrere Lektoren haben. Weiters gibt es Lehrveranstaltungen, wobei diese eine eindeutige Nummer haben und einen Titel. Ein Lektor kann mehrere Lehrveranstaltungen leiten, eine Lehrveranstaltung kann von mehreren Lektoren geleitet werden, hat aber mindestens einen Leiter.

9.3.1.2) Familienstammbaum

Jemand will seine Vorfahren (und sonstige Verwandte) in einer Datenbank speichern. Jede Person wird durch ihren Vornamen, Geburtstag und Geburtsort identifziert. Des weiteren wird ihr Zuname gespeichert. Bei verstorbenen Personen wird auch der Sterbetag und -ort in der Datenbank verwaltet. Bei den Personen wird zwischen Männern und Frauen unterschieden. Für jede Person wird auch der Vater und die Mutter gespeichert, falls diese bekannt sind.
Personen können andere Personen adoptieren. Dabei wird der Tag der Adoption gespeichert.
Des weiteren sollen alle Ehen mit dem Tag der Hochzeit und, falls die Ehe schief gehen sollte, dem Tag der Scheidung vermerkt werden. Paare können nur einmal heiraten.

9.3.1.3) Friedhof

Für einen Friedhof wird zur Arbeitserleichterung der Verwaltung eine Datenbank erstellt. Unterstreichen Sie je Relation einen Schlüssel. Verwenden Sie nur die vorgegebenen Attributnamen. (Diese sind nur bei ihrer jeweils ersten Erwähnung angeführt.)

Auf dem Friedhof gibt es viele Gräber. Jedes Grab hat eine eindeutige Nummer (GNR), eine Lagebeschreibung (LAGE), einen Besitzer (BESITZER) und eine maximale Sarganzahl (MAXSARG). In einem Grab können sich mehrere Verstorbene befinden. Särge haben eine eindeutige Bestellnummer (BNR) und einen Hersteller (HERSTELLER).
Weiters gibt es Verstorbene, von denen die eindeutige Totenscheinnummer (TNR), das Sterbedatum (SDATUM), das Geburtsdatum (GDATUM), der Vorname (VORNAME) und der Nachname (NACHNAME) bekannt sind. Es ist auch bekannt, in welchem Sarg der Verstorbene liegt. Man kann die Position (POSITION) von Verstorbenen relativ zu einem anderen Verstorbenen im selben Grab angeben (z. B. kann ein Verstorbener rechts, links, unter . . . einem anderen Verstorbenen liegen).

Es gibt 3 Friedhofsgärtner mit eindeutiger Sozialversicherungsnummer (SVNR), und Vor- und Nachnamen (VORNAME, NACHNAME), die für die Betreuung der Gräber zuständig sind. Es ist bekannt, welcher von den Gärtnern für ein bestimmtes Grab verantwortlich ist.

Man kann bei der Friedhofsgärtnerei verschiedene Dienstleistungen bestellen (z. B. Pflanzen setzen, Kerzen zu Allerheiligen, . . .). Jede Dienstleistung wird durch eine Nummer (DNR), eine Beschreibung

(BESCHREIBUNG) und einen Preis (PREIS) beschrieben. Bestellungen beziehen sich immer auf Gräber und Dienstleistungen, es wird das Datum (DATUM) angegeben und die Person (PERSON), die die anfallende Rechnung bezahlt. Für die Statistik wird mitprotokolliert, wann (DATUM) welcher Gärtner welche Dienstleistung bei einem Grab durchgeführt hat und wie viele Stunden (STUNDEN) er dafür gebraucht hat.

9.3.1.4) Politik

Um den Überblick über das Kommen und Gehen der politischen Akteure zu behalten bittet Sie ein Freund um eine Datenbank. Zeichnen Sie aufgrund der vorliegenden Informationen ein ER-Diagramm. Achtung!! Beachten Sie, dass der unten beschriebene Sachverhalt stark vereinfacht ist und nicht notwendigerweise mit der Realitat ubereinstimmt. Modellieren Sie bitte auf jeden Fall den angegebenen Sachverhalt!

Zu jeder Person wird ihr Vorname (VNAME), Nachname (NNAME) sowie eine besondere Eigenschaft (EIGENSCHAFT) gespeichert. Es kann keine zwei Personen mit dem selben Namen (gleicher Vorname und gleicher Nachname) geben.
Parteien besitzen eine eindeutige Farbe (FARBE), und darüber hinaus ein (nicht notwendiger Weise eindeutiges) Kürzel (KRZL).
Jede Legislaturperiode ist durch ihren Beginn (VON) gemeinsam mit ihrem Ende (BIS) identifizierbar.
Jeder Aufgabenbereich der Regierung hat eine eindeutige Bezeichnung (BEZ). Außerdem gibt es eine Beschreibung (BESCHREIBUNG) zu jedem Aufgabenbereich.
Es wird vermerkt welche Person in welcher Legislaturperiode welche Aufgaben übernimmt. Außerdem wird gespeichert welcher Aufgabenbereich in welcher Legislaturperiode in welchem Ministerium angesiedelt war. Ministerien haben einen eindeutigen Namen (NAME) und ein Budget für Werbung (WBUDGET). In jedem Ministerium muss mindestens ein Aufgabenbereich angesiedelt sein (in irgendeiner Legislaturperiode). Außerdem gibt es in jeder Legislaturperiode mindestens drei Aufgabenbereiche.
Es soll außerdem vermerkt werden wie viele Stimmen jede Partei in den verschiedenen Legislaturperioden hatte.
Jeder Parteieintritt erhält eine innerhalb der entsprechenden Partei eindeutige Nummer (NR), es wird das Datum des Eintritts (DATUM) gespeichert, sowie welche Person eingetreten ist (bei jedem Parteieintritt tritt genau eine Person einer Partei bei).

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_03_01

Last update: **2019/05/14 20:54**

9.4 Relationenmodell

Das Relationenmodell wurde 1970 von Codd entwickelt. Es **repräsentiert** die **Daten einer Datenbank als eine Menge von Relationen**. Eine **Relation** können wir uns dabei **als Tabelle vorstellen**, der **Zeilen Objekte oder Beziehungen zwischen Objekten beschreiben**. Der **Tabellenkopf** heißt **Relationenschema**, die einzelnen **Spalten** werden als **Attribute** bezeichnet, die einzelnen **Zeilen als Tupel**.

Durch das Relationenmodell kann das konzeptionelle Schema hinsichtlich

- **Semantik der Daten**
- **Integrität der Daten** (Qualität & Zuverlässigkeit der Daten ⇒ Primary Key, Foreign Key)
- **Redundanzen** (Normalformen)

überprüft und verbessert werden.

9.4.1 Formalisierung

Eine **Relationenschema R** ist eine **endliche Menge von Attributnamen** $\{A_1, A_2, ... A_n\}$. Zu jedem Attributnamen A_1 gibt es eine Menge D_i, $1 \leq i \leq n$, den **Wertebereich (=domain)** von A_i, der auch mit $Dom(A_i)$ bezeichnet wird.

Eine **Relation r(R)** auf einem **Relationenschema R** ist eine endliche Menge von Abbildungen $\{t_1,....,t_m\}$ von R nach D. Die Abbildungen werden **Tupel** genannt.

Bsp.: Relationenschemata für RESTAURANT und SPEISE

```
Restaurant = {rnr,name,adresse,haube,typ}
Speise     = {name,preis,rnr}
```

Bsp.: Domänen für das Relationenschemata RESTAURANT:

```
Dom(rnr)   = Menge aller Integer
Dom(name)  = Menge aller Namen
Dom(haube) = {k|0<=k<=4}
```

Bsp.: Die Tabelle Restaurant hat mehrere Tupel. Eines von ihnen ist t mit:

```
t(rnr)       = 3
t(name)      = "Green Cottage"
t(haube,typ) = 2, "chinesisch"
t            = 3, "Green Cottage", "Kettenbrückengasse 3, 1050 Wien", 2,
"chinesisch"
```

Ein Schlüssel einer Relation r(R) ist eine Teilmenge K von R, sodass für zwei verschiedene

Tupel t_1 und t_2 aus r(R) immer $t_1(K) != t_2(K)$ gilt und keine echte Teilmenge K' von K diese Eigenschaft hat.

Im Allgemeinen wird dabei ein Schlüssel als Primärschlüssel ausgezeichnet und in den Relationenschemata durch Unterstreichen der Schlüsselattribute gekennzeichnet.
Bsp.:

```
Restaurant = {__rnr__, name, adresse, haube, typ}
Speise     = {__name__, preis, rnr}
```

9.4.2 Operationen auf Relationen

Um Operationen auf Relationen durchführen zu können, wurde die relationale Algebra eingeführt.

9.4.9.1 Mengenoperationen

Zu den Mengenoperationen gehören:

- Durchschnitt („∩“)
- Vereinigung („∪“)
- Differenz („-“)

von Relationen, die über der gleichen Attributmenge mit derselben Anordnung (identische Reihenfolge der Attribute) definiert sind:

Bsp.: Es sind 2 Relationen r und s gegeben:

r

A	**B**	**C**
a1	b1	c1
a1	b2	c1
a2	b1	c2

s

A	**B**	**C**
a1	b2	c1
a2	b2	c1
a2	b2	c2

Gesucht sind:

- Durchschnitt (r ∩ s)
- Vereinigung (r ∪ s)
- Differenz (r - s)

- Differenz (s - r)

Lösung:

- Durchschnitt (r ∩ s)

A	B	C
a1	b2	c1

- Vereinigung (r ∪ s)

A	B	C
a1	b1	c1
a1	b2	c1
a2	b1	c2
a2	b2	c1
a2	b2	c2

- Differenz (r - s)

A	B	C
a1	b1	c1
a2	b1	c2

- Differenz (s - r)

A	B	C
a2	b2	c1
a2	b2	c2

9.4.9.2 Die Selektion ("σ")

Bei der Selektion werden Zeilen ausgewählt, die einem bestimmten Kriterium entsprechen:

$ σ_{A=a}(r) = \{ t∈r|t(A)=a \} $

Beispiele:
a) $ σ_{A=a1}(r) = ? $

A	B	C
a1	b1	c1
a1	b2	c1

b) $ σ_{A=a1}(s) = ? $

A	B	C
a1	b2	c1

Noch allgemeiner wird die Selektion, wenn wir erlauben, wohldefinierte Operatoren auf den Attributwerten auszuführen, z.B.: arithmetische Operationen auf Zahlenwerte und logische Verknüpfungen von Attributen durch „und“ („∧“), „oder“ („∨“) und Negationen („¬“).

c) $ σ_{(B=b1)∧¬(A=a1)}(s)=? $

d) $ σ_{(haube > 1) ∧ ¬(typ=österreichisch ∨ typ=international)}(Restaurant)=? $

Bsp: Relation Restaurant

rnr	name	adresse	haube	typ
1	Schnitzelhaus	Amstetten	1	österreichisch
2	Brauhof	Amstetten	3	international
3	Pizza Hollywood	Amstetten	2	italienisch

Lösung ⇒ Folgender Tupel:

3	Pizza Hollywood	Amstetten	2	italienisch

9.4.9.3 Die Projektion ("π")

Bei der Projektion werden gewisse Spalten einer Tabelle ausgewählt. Man projiziert nach einer Teilmenge der Attribute:

$π_X(r) = \{t(X)|t∈r\} für X ⊆ R $

Beispiele:

e) $ π_{A,B}(r)=? $

f) $ π_{name,adresse,haube}(Restaurant)= ?? $

name	adresse	haube
Schnitzelhaus	Amstetten	1

name	adresse	haube
Brauhof	Amstetten	3
Pizza Hollywood	Amstetten	2

9.4.9.4 Der Verbund

9.4.9.4.1 Der natürliche Verbund

Der Verbundoperator verknüpft zwei Relationen über ihre gemeinsamen Attribute:

$ r ⋈ s = \{|\{R,S\}| ∃ t_r ∈ r \text{ und } ∃ t_s ∈ s:t_r =t(R) \text{ und } t_s=t(S) \} $

Der Verbundoperator ist kommutativ.

Beispiel:
Relation r

A	B
a1	b1
a2	b1
a3	b2

Relation s

B	C
b2	c1
b2	c2
b1	c3
b3	c4

Relation r ⋈ s

A	B	C
a1	b1	c3
a2	b1	c3
a3	b2	c1
a3	b2	c2

Aufgabe:

```
Restaurant  Speise
```

Lösung:

rnr	name	adresse	haube	typ	preis

Erklärung:
Der natürliche Verbund verknüpft beide Relationen über die gemeinsamen Attribute und gibt daher jene Tupel aus, bei denen sowohl der Name **name** des Restaurants und der Name **name** der Speise als auch die Restaurantnummer **rnr** des Restaurants und jene der Speise gleich sind. In unserem Fall ist das die leere Menge!

Projektionseigenschaften des Verbundoperators

Seien R und S zwei Relationenschema, $ q=r⋈s $ und $ r'=π_R(q) $, dann gilt $ r'⊆r $. Joinen wir also eine Relation r mit einer anderen und projizieren dann nach den ursprünglichen Attributen von r, so können unter Umständen Tupel verloren gehen.

Beispiel:

Relation r

A	**B**
a	b
a	b'

Relation s

B	**C**
b	c

Relation r⋈s=q

A	**B**	**C**
a	b	c

Relation $ π_{AB}(q)=r' $

A	**B**
a	b

9.4.9.4.2 Das Kartesische Produkt

Falls $R ∩ S = \{\}$,die beiden Relationenschemata also kein gemeinsames Produkt haben, so liefert die Verknüpfung r ⋈ s das Kartesische Produkt, geschrieben als r x s.

Beispiel:
Relation r

A	**B**
a1	b1

A	B
a2	b1

Relation s

C	D
c1	d1
c2	d1
c2	d2

Relation r x s = r ⋈ s

A	B	C	D
a1	b1	c1	d1
a1	b1	c2	d1
a1	b1	c2	d2
a2	b1	c1	d1
a2	b1	c2	d1
a2	b1	c2	d2

Wie wir oben gesehen haben, ist das Kartesische Produkt nur für den Fall definiert, dass $R ∩ S=\{\}$. Möchte man das Kartesische Produkt von Relationen bilden, die gemeinsame Attribute haben, so müssen diese in einer der Relationen umbenannt werden. Ist etwa R = {A,B,C} und S = {A,B,D}, so benennen wir die Attribute von S um, so dass S={A',B',D} oder kennzeichnen sie durch Voranstellen des Relationennamens, also S={S.A,S.B,D}.

9.4.9.4.3 Weitere Verbundarten

Weitere Verbundarten sind:

- der Gleichverbund (equi-join)
- der Theta-Verbund (theta-join)
- der Semi-Verbund (semi-join)
- der Äußere Verbund (outer-join)

9.4.9.5 Division

Möchte man die Relation r durch s dividieren, so muss die Attributmenge von s eine Teilmenge der Attributmenge von r sein. Das Ergebnis hat die Differenz der Attributmengen als Attribute und wählt jene Tupel aus r aus, die eingeschränkt auf die Differenz der Attribute R-S für alle Tupel aus s denselben Wert haben.

Beispiel:
Relation R

A	B	C	D
a	b	c	d
a	b	e	f
b	c	e	f
e	d	c	d
a	b	d	e
e	d	e	f
a	d	e	f

Relation S

C	D
c	d
e	f

Relation R % S

A	B
a	b
e	d

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_04

Last update: **2019/05/22 17:38**

9.4.1) Übungen zum Relationenmodell

Übung 1

Relation Restaurant			
rnr	**RName**	**Adr**	**Haube**
1	McDonalds	Amstetten	0
2	McDonalds	Ybbs	0
3	Hollywood	Amstetten	1
4	Casa Venezia	Amstetten	2
5	grill.Bar	Amstetten	2
6	Schinakel	Grein	1

Relation Speise		
rnr	**SName**	**Preis**
1	Hamburger	1
2	Hamburger	1
2	Cheeseburger	1,30
3	Hawaii	8,6
4	Hawaii	9
5	Salat	5
1	Salat	1,80

Abfragen in relationaler Algebra:

a) alle Restaurants, die in Amstetten zu finden sind

b) alle Restaurants von Amstetten mit mindestens 2 Hauben

c) Namen aller Restaurants

d) Namen aller Restaurants die in Amstetten zu finden sind

e) alle Restaurants, die auch Speisen anbieten

f) Namen der Restaurants, die Speisen anbieten

g) Namen aller Restaurants, die einen Salat anbieten

h) Namen und Preise aller Speisen samt Name des Restaurants, von Restaurants, die einen Salat anbieten

Übung 2

Gegeben sind einige Relationen und Abfragen. Formulieren Sie die Abfragen mittels Relationaler Algebra und berechnen Sie auch das Ergebnis der Abfragen.

Die Relation **Rechner** beschreibt die Rechner eines Institutes. **RNr** ist eine eindeutige Bezeichnung für den Rechner, **StudAss** ist der eindeutige Name des Studienassistenten, der den Rechner (und die darauf installierten Programme) wartet, **Speicher** gibt die Größe der Festplatte des Rechners an und **Leist** ist ein Maß für die Leistungsfähigkeit des Rechners, wobei 1 die schlechteste und 5 die beste Leistung ist.

Relation Rechner			
RNr	**StudAss**	**Leist**	**Speicher**
R1	Huber	1	100
R2	Brunner	3	80
R3	Brunner	3	400
R4	Vogt	2	120
R5	Huber	2	500

In **Programm** sind die Programme, die das Institut besitzt gespeichert. **PNr** ist eine eindeutige Nummer des Programms, **PName** ist sein Name, **Bereich** gibt an, um was für eine Art von Programm es sich dabei handelt und **MinLeist** bezeichnet die Leistungsfähigkeit, die ein Rechner mindestens besitzen muss, damit das Programm auf ihm laufen kann. Hat ein Programm also die **MinLeist** 4, so kann man ihn nur in einem Rechner mit **Leist** 4 oder 5 einsetzen, nicht aber einem mit **Leist** 1,2 oder 3.

Relation Programm			
PNr	**PName**	**MinLeist**	**Bereich**
P1	DrawIt	1	Grafik
P2	AskIt	3	Datenbank
P3	WriteIt	1	Text
P4	ConnectIt	2	Internet
P5	PaintIt	2	Grafik
P6	StoreIt	3	Datenbank

Die Relation **Assistent** beschreibt die Assistenten, die an dem Institut arbeiten. (Diese sind von den Studienassistenten verschieden) Dabei ist **ANr** eine eindeutige Nummer für den Assistenten, **AName** ist dessen Name, **StudAss** ist der eindeutige Name des Studienassistenten, der den Assistenten bei seiner Tätigkeit unterstützt. **Gehalt** bezeichnet die Gehaltsstufe des Assistenten.

Relation Assistent			
ANr	**AName**	**StudAss**	**Gehalt**
A1	Novak	Brunner	3

Relation Assistent			
ANr	**AName**	**StudAss**	**Gehalt**
A2	Dvorak	Vogt	1
A3	Husak	Vogt	1
A4	Pfeiffer	Brunner	2

In der Relation **Installation** ist verzeichnet, welche Programme auf welchen Rechnern installiert sind. **RNr** und **PNr** geben den entsprechenden Rechner und das Programm an, **Platz** gibt die Größe der Installation auf der Festplatte an und **Code** ist ein Code, den nur die Studienassistenten verstehen.

Relation Installation			
RNr	**_PNr_**	**Platz**	**Code**
R1	P1	500	X
R1	P3	300	z
R2	P6	300	X
R2	P2	200	pp
R3	P1	400	c
R3	P2	100	tt
R3	P3	500	c
R3	P4	200	pp
R3	P5	200	z
R3	P6	100	t
R4	P5	1000	T
R5	P1	200	p
R5	P5	100	ccc

In der Relation **Benutzung** wird vermerkt, wie lange die Assistenten die verschiedenen Programme benutzen. **ANr** verweist auf den Assistenten, **PNr** auf das Programm und **Stund** gibt an wieviele Stunden am Tag das Programm vom Assistenten pro Tag höchstens benötigt wird.

Relation Benutzung		
ANr	**_PNr_**	**Stund**
A1	P1	5
A1	P2	3
A2	P1	6
A2	P4	2
A2	P5	5
A3	P1	7
A3	P3	3
A4	P1	1
A4	P4	4

Abfragen:

a) Geben Sie den Namen der Assistenten aus, die ein Programm möglicherweise länger als 5 Stunden benutzen.

b) Wie heißen die Programme, die von Huber gewartet werden?

c) Wie heißen die Programme, die von allen Assistenten benutzt werden?

d) Auf welchem Rechner sind dieselben Programme installiert, wie auf dem Rechner R1?

e) Von welchen Studienassistenten wird das Programm WriteIt nicht gewartet?

f) Welche Paare von Assistenten werden vom gleichem Studienassistenten betreut? (Dabei soll jedes Paar nur einmal ausgegeben werden)

g) Welche Studienassistenten betreuen sowohl Assistenten, als auch Rechner?

h) Welche Paare von Rechner haben dieselbe RLeistung?

i) Welche Programme (gesucht sind die Namen) sind auf allen Rechnern installiert?

j) Welche Programme laufen auf einem Rechner, der genaue die minimale Leistungsfähigkeit für das Programm besitzt? (Gesucht sind die Paare aus Rechnernummer und Programmnummer)

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

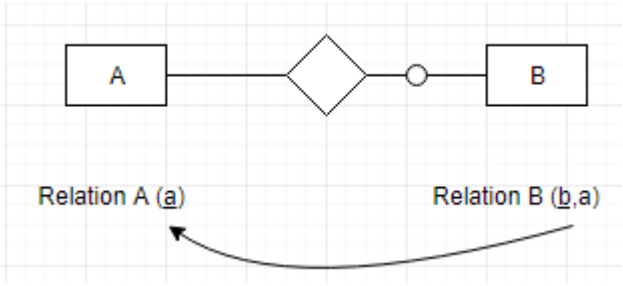
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_04_01

Last update: **2019/05/14 20:55**

9.5 Umsetzung des ER-Modells in ein Relationenmodell

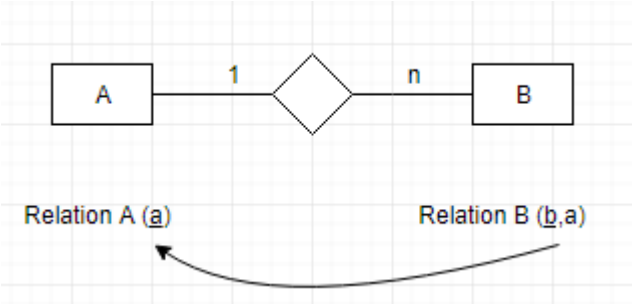
9.5.1 1-zu-1 Beziehung

Bei einer 1-zu-1 Beziehung zwischen Entitäten wird die Beziehung aufgelöst, in dem der Schlüssel der einen Entität zur zweiten Entität hinzukommt. Welche Richtung hier verwendet wird, ist dem Designer überlassen. Wenn allerdings, eine optionale Beziehung besteht, wird der Schlüssel auf der optionalen Seite gespeichert.



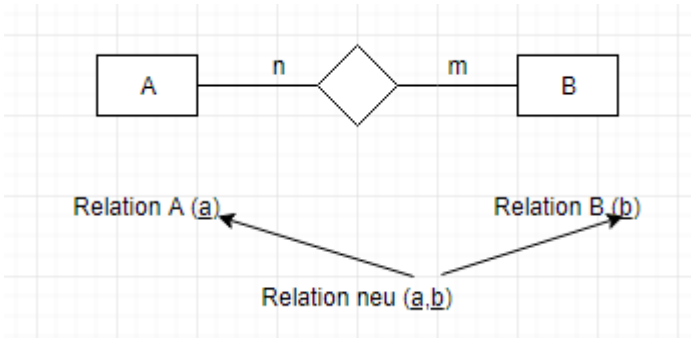
9.5.2 1-zu-n Beziehung

Im Falle einer 1-zu-n Beziehung schreiben wir den Schlüssel auf der 1-Seite in die Relation die der Entität auf der n-Seite entspricht.



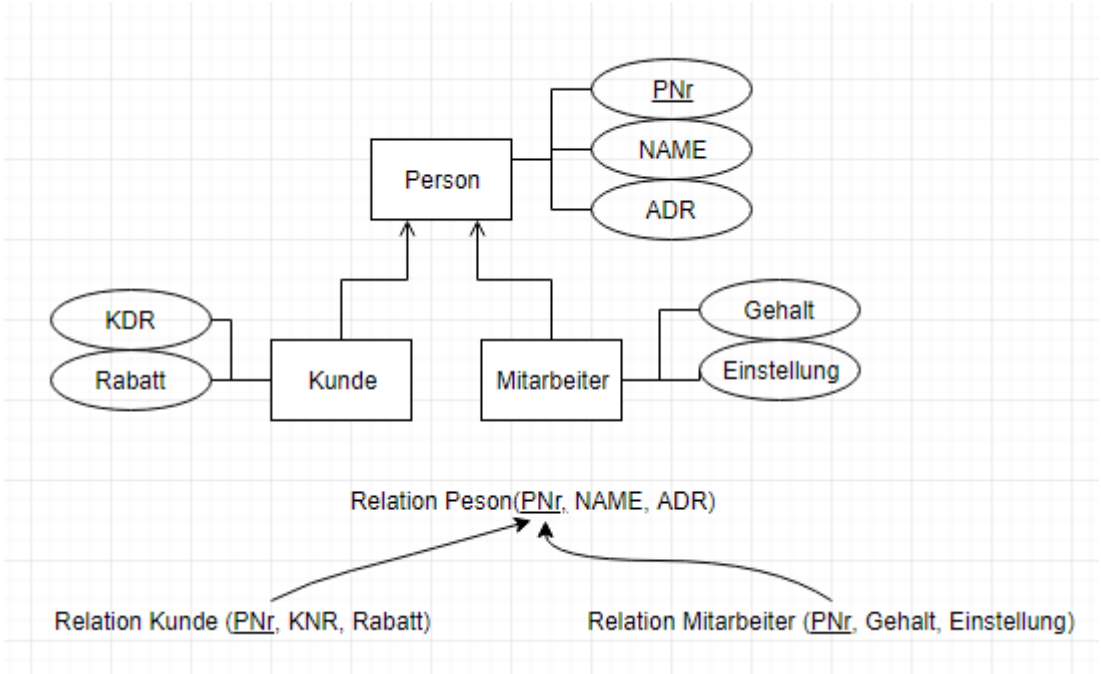
9.5.3 n-zu-m Beziehung

Bei einer n-zu-m Beziehung führen wir eine neue Relation ein, die die Schlüssel beider Entitäten als Schlüssel besitzt.



9.5.4 Generalisierung

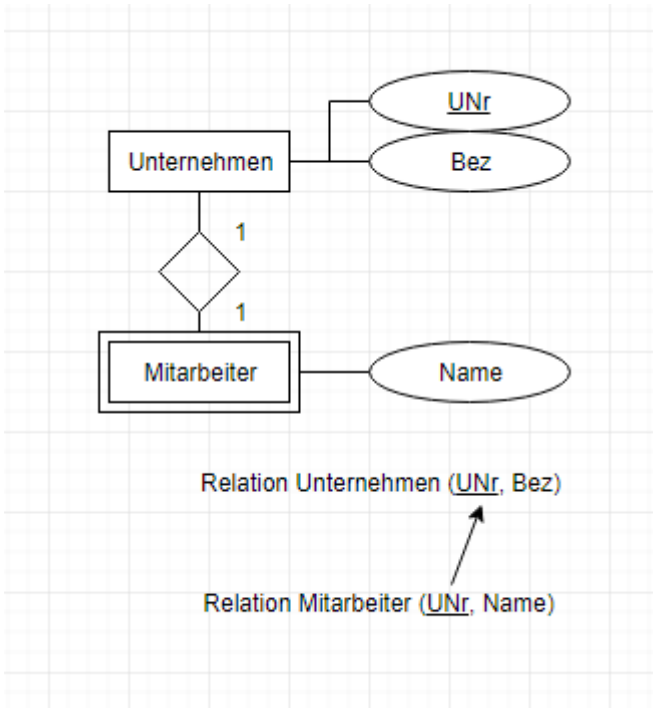
9.5.4.1 nicht disjunkte Entitäten



9.5.4.1 disjunkte Entitäten

9.5.5 Weak-Entities

Bei schwachen Entitäten, bei denen die eigenen Attribute nicht ausreichen um ein Tupel eindeutig zu identifizieren, müssen die Schlüsselattribute der damit verbundenen Entitäten zum Schlüssel hinzugenommen werden.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

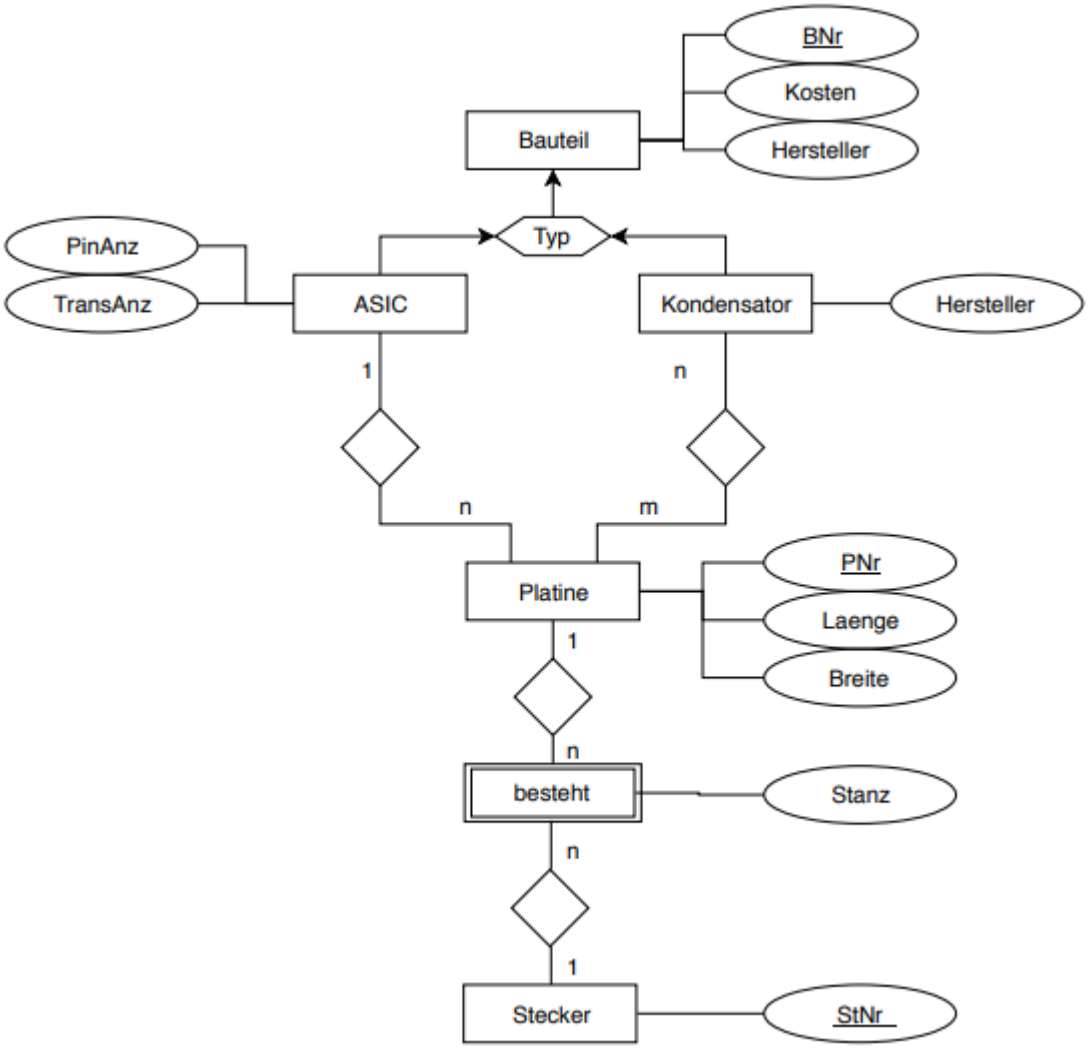
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_05

Last update: **2019/05/14 20:56**

9.5.1) Übungen zur Umsetzung von ER-Modell => Relationenmodell

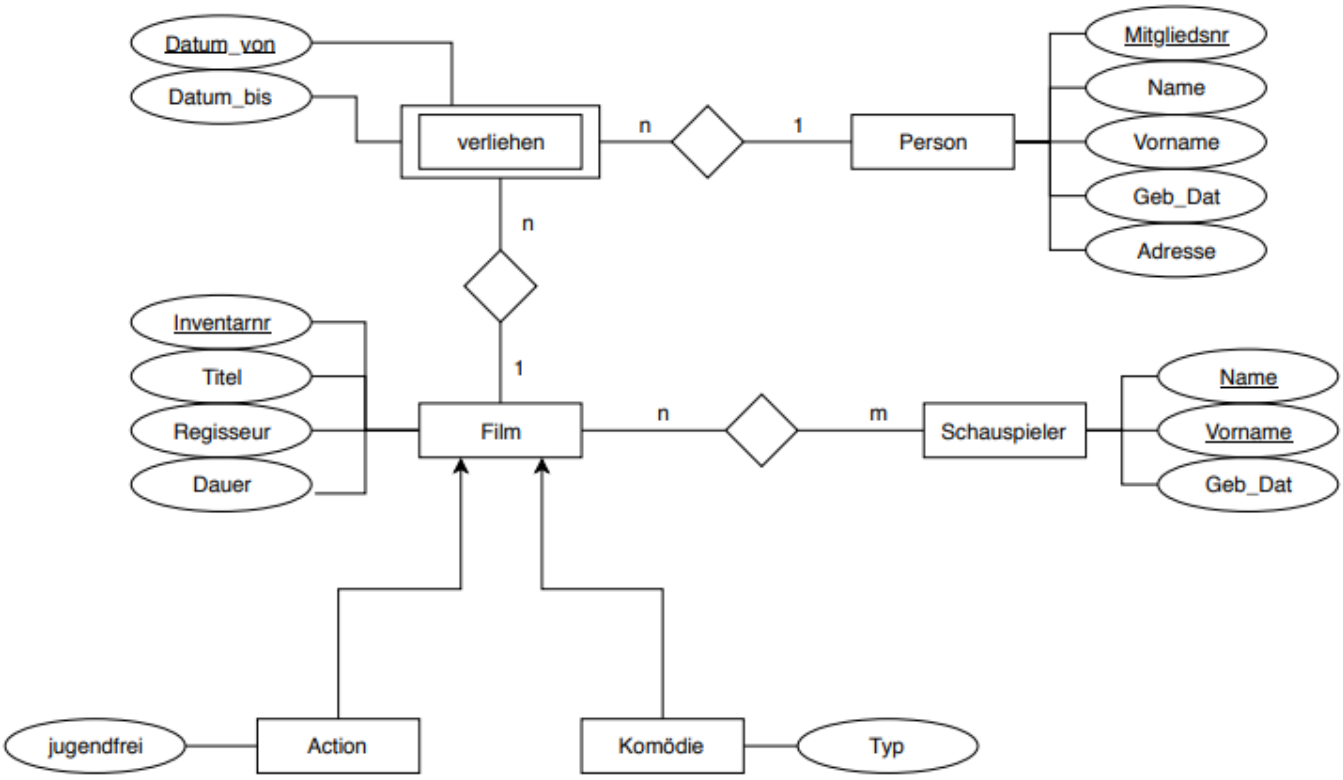
Übung 1

Gegeben ist folgendes ER-Diagramm zur Verwaltung von Elektronikbauteilen. Entwickeln Sie daraus die Relationen der Datenbank.



Übung 2

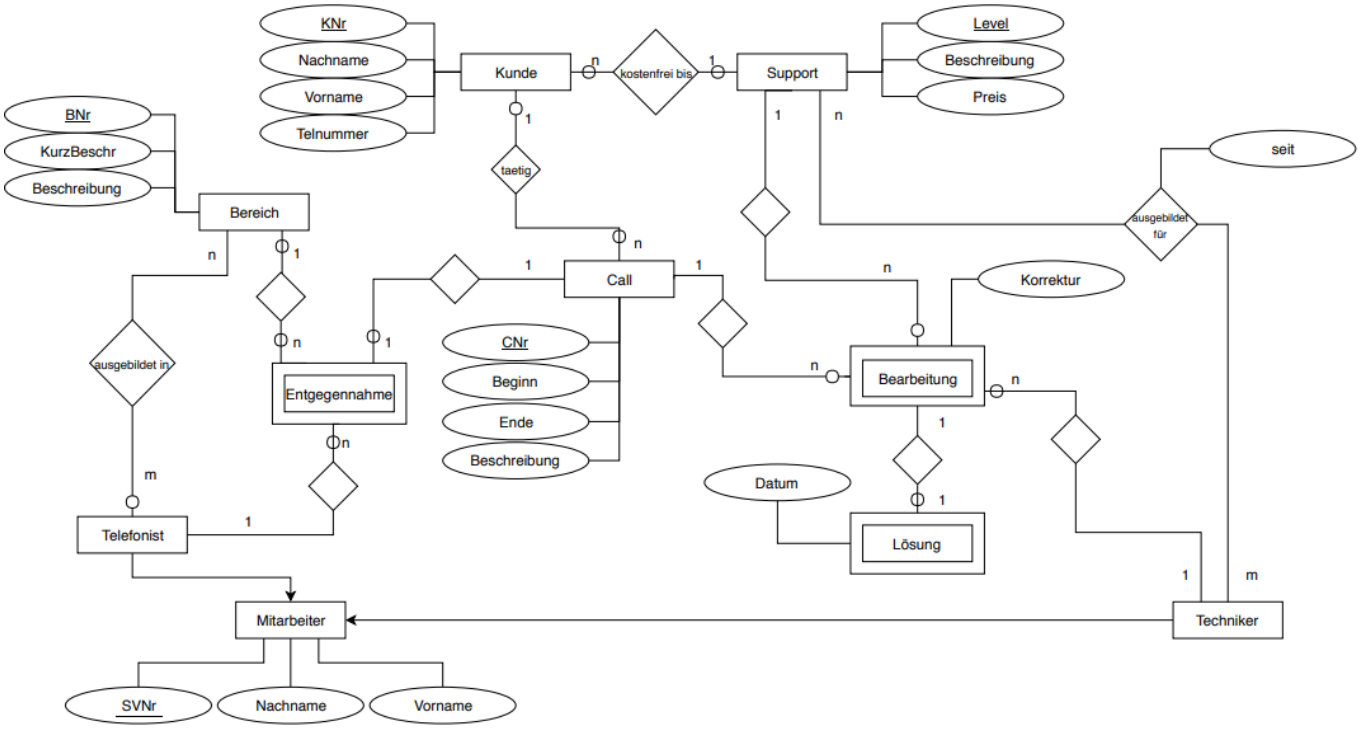
Gegeben ist folgendes ER-Diagramm zur Verwaltung eines Videoverleihs. Entwickeln Sie daraus die Relationen der Datenbank.



Übung 3

Gegeben ist folgendes ER-Diagramm zur Verwaltung eines Videoverleihs. Entwickeln Sie daraus die Relationen der Datenbank.

Das Call Center System erlaubt es bestehenden oder zukünftigen Kunden sich über ein menügesteuertes Interface (Tastenkombinationen am Telefon, Handy, etc.) im Support Center einzuwählen. Das Menü ist dabei in mehrere Bereiche (Produktauskunft, Rechnungsinfo, Technische Probleme, etc.) gegliedert. Eingehende Calls werden entweder von Telefonisten entgegengenommen, welche in allgemeinen Bereichen ausgebildet sind, oder von Technikern bearbeitet, sofern es sich um technische Anfragen handelt. Technische Anfragen werden im Allgemeinen nur für bestehende Kunden bearbeitet. Die mit den Anfragen betrauten Techniker sind für bestimmte Support Levels ausgebildet. Support Levels sind kostenpflichtig, aber Stammkunden bzw. mehr zahlende Kunden können bestimmte Levels gratis in Anspruch nehmen. Technische Probleme können von verschiedenen Technikern auf verschiedene Levels bearbeitet und einer Lösung zugeführt werden.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_05_01

Last update: **2019/05/14 20:56**

9.6) Normalformen

Die Kriterien eines guten Datenbankentwurfs sind einerseits möglichst geringe **Redundanz** und andererseits von **Einfügen, Lösch- und Änderungsanomalien** abhängig. Um Redundanzen und Anomalien zu vermeiden, führen wir den Begriff **Normalform** ein, der uns hilft unerwünschte Eigenschaften der Datenbank zu vermeiden.

1. Normalform (1NF)

Ein Relationenschema R ist in 1NF wenn der Wertebereich aller Attribute von R atomar sind. D.h. wenn keine mehrwertigen Attribute vorkommen.

Beispiel - CD Lieder

_CD_ID_	Name	Titelliste
4711	All That You Can Leave Behind (U2)	Beautiful Day, Walk On, Kite, Wild Honey
4712	Tattou You (Rolling Stones)	Start Me Up, Hang Fire, Slave

- Das Feld **Name** beinhaltet Album und Interpret
- Das Feld **Titelliste** enthält eine Aufzählung aller Titel

Fehler/Problem

- Zur Sortierung nach Interpret muss das Feld **Name** in Album und Interpret aufgeteilt werden
- Den Titel können (mit einfachen Mitteln) nur alle gleichzeitig als Titelliste oder gar nicht dargestellt werden

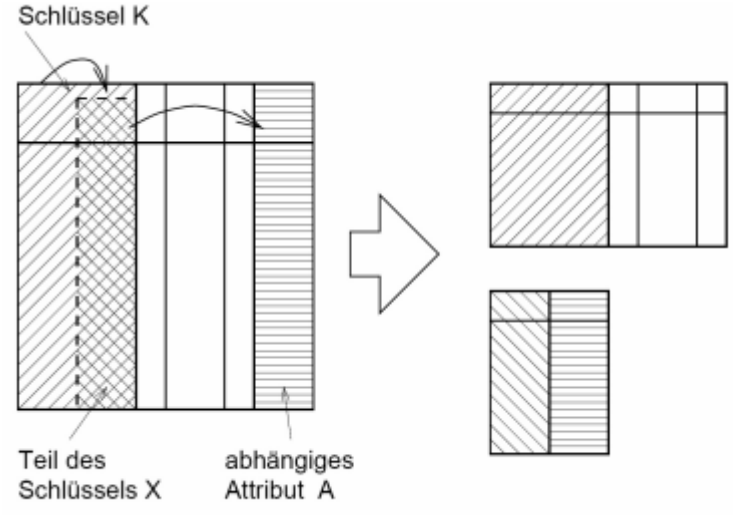
Lösung 1NF

_CD_ID_	Album	Interpret	Titelliste
4711	All That You Can Leave Behind	U2	Beautiful Day
4711	All That You Can Leave Behind	U2	Walk On
4711	All That You Can Leave Behind	U2	Kite
4711	All That You Can Leave Behind	U2	Wild Honey
4712	Tattou You	Rolling Stones	Start Me Up
4712	Tattou You	Rolling Stones	Hang Fire
4712	Tattou You	Rolling Stones	Slave

2. Normalform (2NF)

Ein Attribut heißt Prim in einer Relation, wenn es mindestens in einem Schlüssel von R enthalten ist bzw. nicht prim, wenn es in keinem Schlüssel enthalten ist.
Ein Relationenschema R ist in zweiter Normalform, wenn es in 1NF und jedes nicht prime Attribut voll funktional von jedem Schlüssel von R abhängig ist, d.h. nur vom ganzen

Schlüssel und nicht nur von einem Teil abhängig ist.



Beispiel

Gegeben ist folgende Tabelle:

_CD_ID_	Album	Interpret	_Track_	Titel
4711	All That You Can Leave Behind	U2	1	Beautiful Day
4711	All That You Can Leave Behind	U2	2	Walk On
4711	All That You Can Leave Behind	U2	3	Kite
4712	Tattou You	Rolling Stones	1	Start Me Up
4712	Tattou You	Rolling Stones	2	Hang Fire

Problem

z.B.: Durch ein Update des Albums kommt es zu einer Dateninkonsistenz

_CD_ID_	Album	Interpret	_Track_	Titel
4711	All That You Can Leave Behind	U2	1	Beautiful Day
4711	All That You Can Leave Behind	U2	2	Walk On
4711	All That You Can Leave Behind	U2	3	Kite
4712	**Sticky Fingers**	Rolling Stones	1	Start Me Up
4712	Tattou You	Rolling Stones	2	Hang Fire

Lösung 2NF

_CD_ID_	Album	Interpret
4711	All That You Can Leave Behind	U2
4712	Tattou You	Rolling Stones

_CD_ID_	_Track_	Titel
4711	1	All That You Can Leave Behind
4711	2	Walk On
4711	3	Kite

_CD_ID_	_Track_	Titel
4712	1	Start Me Up
4712	2	Hang Fire

3. Normalform (3NF)

Ein Relationenschema R ist in 1NF ist genau dann in 3NF, wenn kein nicht primes Attribut von einem Schlüssel in R transitiv abhängt.

Transitivitätsregel K→X, X→A : K→A

Beispiel

Gegeben sei eine Tabelle mit folgenden Feldern:

_CD_ID_	Album	Interpret	Gründungsjahr
4711	All That You Can Leave Behind	U2	1976
4712	Sticky Fingers	Rolling Stones	1962
4713	Tattou You	Rolling Stones	1962

Offensichtlich lässt sich der **Interpret** einer CD aus der **CD_ID** bestimmen. Das **Gründungsjahr** der Band hängt dagegen vom **Interpret** und damit nur **!!transitiv!!** von der **CD_ID** ab.

Problem

Auch hier besteht eine Datenredundanz, wodurch Inkonsistenzen beim Ändern auftreten können.

_CD_ID_	Album	Interpret	Gründungsjahr
4711	All That You Can Leave Behind	U2	1976
4712	Sticky Fingers	Rolling Stones	**1972**
4713	Tattou You	Rolling Stones	1962

Lösung

_CD_ID_	Album	Interpret
4711	All That You Can Leave Behind	U2
4712	Sticky Fingers	Rolling Stones
4713	Tattou You	Rolling Stones

Interpret	Gründungsjahr
U2	1976
Rolling Stones	1962

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_06

Last update: **2019/05/14 20:57**

9.7) SQL - Structured Query Language

SQL (**Structured Query Language**) hat sich als Standardabfragesprache für relationale Datenbanken etabliert. SQL stellt die Schnittstelle zwischen der relationalen Datenbank und dem Anwendungsprogramm dar. Es lassen sich damit alle Operationen der Relationenalgebra aus Kapitel 4 realisieren. Die Sprachelement von SQL lassen sich in mehrere Kategorien unterteilen, die allerdings im Standard nicht festgeschrieben sind.

Allgemeines

Erstellen von Datenbanken

```
CREATE DATABASE DBName;
```

Anzeigen von Datenbanken

```
SHOW DATABASES;
```

Selektieren von Datenbanken

```
USE DBName;
```

Anzeigen von Tabellen in einer Datenbank

```
SHOW TABLES;
```

Anzeigen von Spalten in einer Tabelle

```
SHOW COLUMNS FROM tabellenname;
```

Löschen von Datenbanken

```
DROP DATABASE DBName;
```

9.7.1) DDL (Data Definition Language)

- Anweisungen zur Anlage und Verwaltung von Datenbankschemata
- Anweisungen zur Definition von Relationen einschließlich der Konsistenzbedingungen
- Anweisungen zur Anlage von Datensichten (Views)

- 9.7.1) DDL
    - 9.7.1.1) DDL - Übungen

9.7.2) DQL (Data Query Language)

- Abfrage von Daten
- 9.7.2) Data Query Language
    - 9.7.2.1) DQL- Übungen

9.7.3) DML (Data Manipulation Language)

- Eingabe von Daten in eine vorhandene Tabelle
- Änderung von Daten in einer Tabelle
- Löschung von Daten in einer Tabelle
- 9.7.3) DML
    - 9.7.3.1) DML - Übungen

9.7.4) DCL (Data Control Language)

- Anlegen von Benutzern
- Vergabe von Zugriffsrechten

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_07

Last update: **2019/05/14 21:01**

9.7.1) DDL - Data Definition Language

9.7.1.1) Datentypen

In SQL stehen folgende Datentypen zur Auswahl:

- exakt numerisch
  - INTEGER
  - SMALLINT
  - NUMERIC (p,q)
  - DECIMAL (p,q)
- angenähert numerisch
  - REAL
  - DOUBLE
  - FLOAT
- Zeichenketten
  - CHAR
  - VARCHAR
- Bitketten
  - BLOB (Binary Large Objects)
- Datum und Uhrzeit
  - DATE
  - TIME
  - TIMESTAMP
- Logischer Datentyp
  - BOOLEAN

9.7.1.2) ERZEUGEN von Relationenschemata

Mit dem Befehl „**CREATE TABLE *relationenname* (...)**“ wird ein Relationenschema definiert. Zu jedem Attribut wird ein Typ angegeben. Optional können eine oder mehrere Integritätsbedingungen angegeben werden. Mögliche Integritätsbedingungen sind:

- die Angabe, dass das Attribut einen Primärschlüssel darstellt (**PRIMARY KEY**)
- dass das Attribut keinen Nullwert annehmen darf (**NOT NULL**)
- dass ein Attribut eindeutig sein muss (**UNIQUE**) - diese Bedingung gilt automatisch für Primärschlüsselattribute
- oder eine durch die Klausel „**CHECK (*bedingung*)**“ formulierte Wertebereichseinschränkung

Weiters kann für Attribute durch die Klausel „**DEFAULT *wert***“ ein Vorgabewert angegeben werden.

Beispiel

Es soll eine leere Relation mit dem Namen **Restaurant** erstellt werden. Es wird festgelegt, dass das Attribut **rnr** ein Primärschlüssel ist, dass die Attribute **name** und **adresse** keine Nullwerte annehmen

dürfen, und dass **haube** nur die Werte (0-4) annehmen darf. Für den Typ eines Restaurants ist als Defaultwert „österreichisch“ festzulegen.

```
CREATE TABLE restaurant
(
    rnr INT NOT NULL,
    name VARCHAR(100) NOT NULL,
    adresse    VARCHAR(100) NOT NULL,
    haube INT CHECK (0<=haube AND haube <=4),
           /*oder CHECK (haube IN (0,1,2,3,4)),*/
           /*oder CHECK (haube BETWEEN 0 AND 4),*/
    typ VARCHAR(100) DEFAULT 'österreichisch',
    CONSTRAINT PK_Restaurant PRIMARY KEY(rnr)
);
```

Eine spezielle Integritätsbedingung (zu Attributen oder zu Relationen) ist die mittels einer **REFERENCES-Klausel** angegebene Fremdschlüsselbedingung **FOREIGN KEY**, die eine Abhängigkeit repräsentiert. Zu jeder Fremdschlüsselbedingung kann durch **„ON UPDATE *action*„** und **„ON DELETE *action*„** angegeben werden, wie auf Verletzungen durch Änderung oder Löschen des referenzierten Schlüsselwertes reagiert werden soll. Mögliche Aktionen sind:

- die Änderung bzw. das Löschen zu verhindern (**NO ACTION**)
- fortzusetzen (**CASCADE**) oder
- den Wert des referenzierenden Attributes auf einen Nullwert (**SET NULL**) bzw. den Defaultwert des Attributs (**SET DEFAULT**) zu setzen

Beispiel

Der nächste SQL-Befehl erzeugt das Relationenschema **Speise**. Die angeführten Integritätsbedingungen legen u.a. fest, dass die Attribute **rnr** und **name** gemeinsam den Primärschlüssel der Relation **Speise** bilden und dass das Attribut **rnr** der Relation **Speise** ein Fremdschlüssel ist, der sich auf das Schlüsselattribut **rnr** der Relation **Restaurant** bezieht (**Speise[rnr] ⊆ Restaurant[rnr]**). Die Angabe **ON UPDATE CASCADE** legt fest, dass eine Änderung der Nummer eines Restaurants bei den entsprechenden Speisen mitgezogen wird, und die Angabe **ON DELETE NO ACTION** legt fest, dass ein Restaurant nicht gelöscht werden darf, solange noch Speisen für dieses Restaurant vorhanden sind.

```
CREATE TABLE Speise
(
    rnr INTEGER,
    name VARCHAR(150),
    CONSTRAINT PK_Speise PRIMARY KEY (name),
    CONSTRAINT FK_Speise FOREIGN KEY (rnr) REFERENCES restaurant(rnr) ON
UPDATE CASCADE ON DELETE NO ACTION
);
```

9.7.1.3) ÄNDERN von Relationenschemata

Die Änderung der Struktur einer Tabelle wird mit dem Befehl „**ALTER TABLE *relationenname…***“ durchgeführt.

Beispiel - Hinzufügen eines Attributes KALORIEN zur Relation SPEISE

```
ALTER TABLE Speise ADD Kalorien INT;
```

Beispiel - Hinzufügen einer CHECK-Klausel für Kalorien

```
ALTER TABLE Speise ADD CONSTRAINT ck CHECK (Kalorien>=0);
```

Beispiel - Entfernen eines Attributes

```
ALTER TABLE Restaurant DROP Adresse;
```

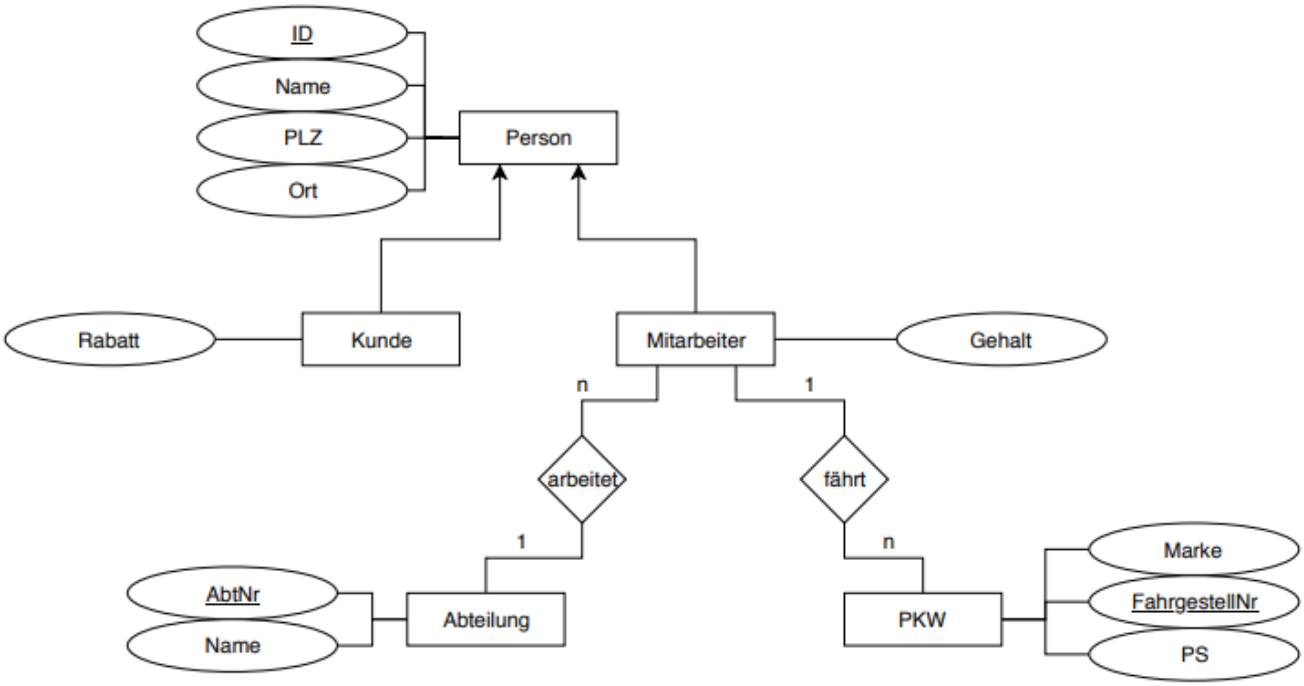
9.7.1.4) ENTFERNEN von Relationenschemata

Mit dem Befehl „**DROP TABLE *relationenname***“ wird eine Relation wieder gelöscht.

```
DROP TABLE Restaurant;
```

9.7.1.5) Übungsbeispiel

Gegeben ist folgendes ER-Modell:



Gesucht sind die SQL-Statements zur Erstellung der Relationen

```
CREATE TABLE Orte (
    Plz INT NOT NULL,
    Ort VARCHAR(100),
    CONSTRAINT PK_Plz PRIMARY KEY(Plz)
);

CREATE TABLE Person (
    ID INT NOT NULL,
    Name VARCHAR(100),
    Plz INT NOT NULL,
    CONSTRAINT FK_Orte FOREIGN KEY (Plz) REFERENCES Orte(Plz),
    CONSTRAINT PK_ID PRIMARY KEY (ID)
);

CREATE TABLE Kunde (
    ID INT NOT NULL,
    Rabatte INT NOT NULL,
    CONSTRAINT PK_ID PRIMARY KEY (ID),
    CONSTRAINT FK_ID FOREIGN KEY (ID) REFERENCES Person(ID) ON DELETE
CASCADE
);

CREATE TABLE Abteilung(
    AbtNr INT NOT NULL,
    Name VARCHAR(140),
    CONSTRAINT PK_AbtNr PRIMARY KEY(AbtNr)
);

CREATE TABLE Mitarbeiter(
    ID INT NOT NULL,
```

```
    Gehalt INT NOT NULL,
    AbtNr INT NOT NULL,
    CONSTRAINT PK_AbtNr PRIMARY KEY(ID),
    CONSTRAINT FK_AbtNr FOREIGN KEY(AbtNr) REFERENCES Abteilung(ID),
    CONSTRAINT FK_ID FOREIGN KEY(ID) REFERENCES PERSON(ID)
);

CREATE TABLE PKW(
    Marke VARCHAR(100),
    PS INT NOT NULL,
    FahrgestellNr INT NOT NULL,
    ID INT NOT NULL,
    CONSTRAINT PK_fgsNr PRIMARY KEY (FahrgestellNr),
    CONSTRAINT FK_ID FOREIGN KEY (ID) REFERENCES Mitarbeiter(ID)
);
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_07:09_07_01

Last update: **2019/05/14 20:59**

9.7.1.1) DDL-Übungen

Aufgabe 1 (am PC)

Gegeben sind folgende Relationen:

Relation Rechner			
RNr	StudAss	Leist	Speicher
R1	Huber	1	100
R2	Brunner	3	80
R3	Brunner	3	400
R4	Vogt	2	120
R5	Huber	2	500

Relation Programm			
PNr	PName	MinLeist	Bereich
P1	DrawIt	1	Grafik
P2	AskIt	3	Datenbank
P3	WriteIt	1	Text
P4	ConnectIt	2	Internet
P5	PaintIt	2	Grafik
P6	StoreIt	3	Datenbank

Relation Installation			
RNr	PNr	Platz	Code
R1	P1	500	X
R1	P3	300	z
R2	P6	300	X
R2	P2	200	pp
R3	P1	400	c
R3	P2	100	tt
R3	P3	500	c
R3	P4	200	pp
R3	P5	200	z
R3	P6	100	t
R4	P5	1000	T
R5	P1	200	p
R5	P5	100	ccc

Relation Benutzung		
ANr	PNr	Stund
A1	P1	5
A1	P2	3
A2	P1	6
A2	P4	2
A2	P5	5
A3	P1	7
A3	P3	3
A4	P1	1
A4	P4	4

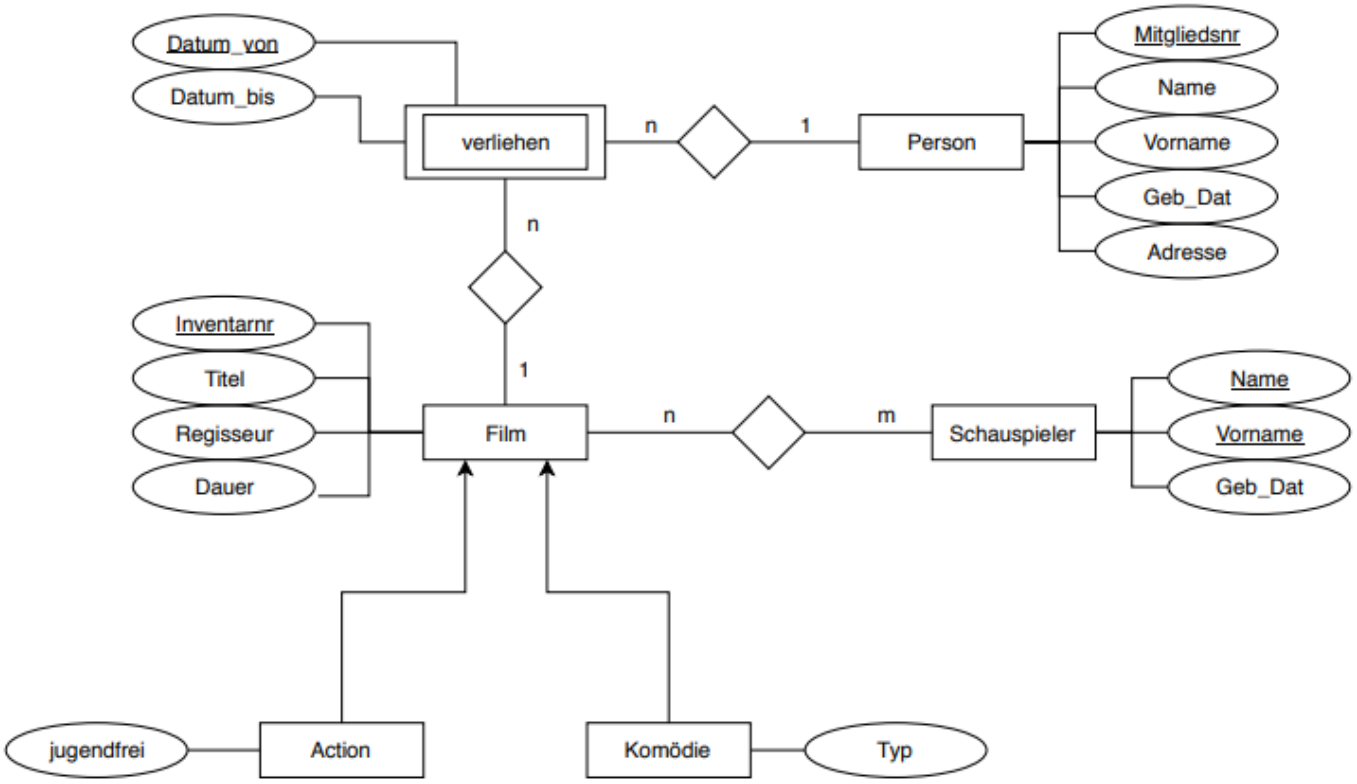
Relation Assistent			
ANr	AName	StudAss	Gehalt
A1	Novak	Brunner	3
A2	Dvorak	Vogt	1
A3	Husak	Vogt	1
A4	Pfeiffer	Brunner	2

Gesucht sind die SQL-Statements zur Erstellung der Relationen! Erstellen sie die Relationen mithilfe der SQL-Statements in einer MySQL-Datenbank namens „Rechnerverwaltung“(XAMPP)! Zum Erstellen der Datenbank verwendet folgenden Befehl:

```
CREATE DATABASE Rechnerverwaltung;
```

Aufgabe 2 (am PC)

Gegeben ist folgendes ER-Modell:



Gesucht sind die SQL-Statements zur Erstellung der dafür notwendigen Relationen! Erstellen sie die Relationen mithilfe der SQL-Statements in einer MySQL-Datenbank namens „Filmverwaltung“(XAMPP)!

Aufgabe 3 (am PC)

Gegeben sind folgende Relationen:

Relation Winzer					
wnr	**name**	**strasse**	**plz**	**ort**	**telefon**
1	Lackner Tinnacher	Steinbach 8	4567	Gamlitz	1234567
2	Weingut Prager	Weissenkirchen 48	3610	Weissenkirchen	1234567
3	Weingut Emmerich Knoll	Unterloiben 10	3601	Unterloiben	12344457
4	Weingut F.X. Pichler	Unterloiben 27	3601	Unterloiben	11122233
5	Weingut Spätlese	Weintalstrasse 23	1136	Wien	
6	Freie Weingarten Wachau	Kremstalstrasse 23	3600	Krems	2304002
7	Stiftskellerei				

Relation Winzer

wnr	name	strasse	plz	ort	telefon
8	Weingut Biegler	Wienerstrasse 88	3502	Gumpoldskirchen	54564565

Relation Wein

nr	bezeichnung	sorte	jahrgang	preis	anzahl	*wnr*	position
1	Riesling Kellerberg	Riesling	1999	28.00	24	4	1
2	Loibenberg	Gr. Veltliner	2000	19.00	36	4	2
3	Ried Kreutles	Gr. Veltliner	2000	19.00	15	3	4
4	Riesling Smaragd	Riesling	2000	21.00	30	2	5
5	Grauburgunder	Grauburgunder	2003	16.00	72	1	6
6	Morillon	Chardonnay	2003	9.00	55	1	7
7	Riesling Federspiel	Riesling	2003	9.90	80	6	3
8	Chardonnay	Chardonnay	2003	9.00	16	8	8

Relation Keller

knr	reihe	regal	fach
1	1	1	1
2	1	1	2
3	1	1	3
4	1	2	1
5	1	2	2
6	2	1	1
7	2	1	2
8	2	2	1

Relation Protokoll

pnr	*nr*	pDatum	verwendung	anzahl
1	1	2003-01-12	Geschenk an Herrn Berger	12
2	3	2003-07-10	Eigenbedarf	2
3	1	2003-07-23	Eigenbedarf	4
4	6	2003-08-14	Geschenk (Frau Kunz)	6
5	1	2003-08-27	Glasbruch	1
6	4	2003-11-03	Korkgeruch	1
7	6	2003-11-03	Eigenbedarf	3

Gesucht sind die SQL-Statements zur Erstellung der dafür notwendigen Relationen! Erstellen sie die Relationen mithilfe der SQL-Statements in einer MySQL-Datenbank namens „Winzerverwaltung“(XAMPP)!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_07:09_07_01:09_07_01_01

Last update: **2019/05/14 20:59**

9.7.2) DQL - Data Query Language

9.7.2.1) Grundkonstruktion einer SQL-Abfrage

Grundkonstruktion einer SQL-Abfrage ist von der Form

„**SELECT** *attributListe* **FROM** *relation*“

Beispiel

```
SELECT name,adresse FROM restaurant;
```

entspricht
$ π_{name,adresse}(restaurant) $

```
SELECT * FROM restaurant;
/*liefert die gesamte Tabelle (d.h. mit allen Attributen)
```

SQL-Abfrage mit Verknüpfung mehrerer Relationen:

„**SELECT** *attributListe* **FROM** *relationen* **WHERE** *bedingung*“

Eine solche SQL-Abfrage wird ausgewertet, indem zunächst das **kartesische Produkt** der in der **FROM**-Klausel angegebenen Relationen gebildet wird. Daraus werden dann jene Tupel ausgewählt, die die in der **WHERE**-Klausel angegebenen Bedingungen erfüllen. Zudem wird auf die in der **SELECT**-Klausel angegebenen Attribute dieser Tupel projiziert.

Beispiel

Wir wollen wissen welche Speisen in welchem Restaurant angeboten werden.

```
SELECT * FROM Speise, Restaurant WHERE Restaurant.rnr = Speise.rnr;
```

Die Bedingunng in der WHERE-Klausel kann sich aus mehreren, durch die logischen Operatoren AND, OR und NOT verbundenen Teilbedingungen zusammensetzen.

Beispiel

```
    SELECT rnr, name, haube, typ
    FROM restaurant
```

```
    WHERE haube > 1 AND NOT (typ='österreichisch' OR typ='international');
```

In der SELECT-Klausel können auch Ausdrücke, die aus Attributen berechnet werden, stehen. Durch das Schlüsselwort **AS** kann einem so berechneten Ausdruck ein Attributname zugewiesen werden.

Beispiel

Wir wollen alle Paare von Restaurants auswählen, die sich um mehr als zwei Hauben unterscheiden und die Differenz der Hauben ausgeben.

```
    SELECT r1.name, r2.name, (r2.haube-r1.haube) AS differenz
    FROM restaurant r1, restaurant r2
    WHERE r1.haube +2 < r2.haube;
```

In SQL werden Duplikate, d.h. Ergebnistupel mit identischen Werten, nicht automatisch eliminiert. Um Duplikate zu eliminieren, muss der Liste von ausgewählten Attribute das Schlüsselwort **DISTINCT** vorangestellt werden.
Durch die Angabe einer ORDER BY-Klausel mit den Schlüsselworten ASC oder DESC kann die Ausgabe aufsteigend bzw. absteigend nach bestimmten Attributen sortiert werden. Der Defaultwert ist ASC.

Beispiel

Wir wollen eine alphabetisch sortierte Liste aller Speisen von Restaurants mit mehr als einer Haube.

```
    SELECT DISTINCT s.name
    FROM Restaurant r, Speise r
    WHERE (r.rnr = s.rnr) AND (r.haube>1)
    ORDER BY s.name, s.preis;
```

Übungsbeispiele

Gegeben sind folgende Relationen

Person (id, name, plz)
Orte (plz, ort)
Mitarbeiter (id, gehalt, abtnr)
Kunde (id, rabatt)
Abteilung (abtnr, name)
PKW (marke, ps, id, fahrgestellnr)

Aufgaben

Entwickeln Sie SQL-Anweisungen für folgende Abfragen

1. Geben Sie die Namen aller Personen aus Amstetten aus

2. Geben Sie die Namen aller Mitarbeiter aus Amstetten aus
3. Gesucht sind die Namen aller Mitarbeiter die auch Kunden sind
4. Namen aller Mitarbeiter die PKWs mit mehr als 1000 PS fahren
5. Gesucht sind die Namen jener Orte, in denen Mitarbeiter wohnen, die in einer Abteilung „Buchhaltung“ arbeiten

Lösungen

```
1)
SELECT name
FROM Person p, Orte o
WHERE o.ort="Amstetten" AND p.plz=o=plz;


2)
SELECT name
FROM Person p, Orte o, Mitarbeiter m
WHERE m.id=p.id AND o.ort="Amstetten" AND p.plz=o.plz;


3)
SELECT name
FROM Person p, Orte o, Mitarbeiter m
WHERE m.id=p.id AND m.id=k.id;

4)
SELECT name
FROM Person p, Mitarbeiter m, PKW
WHERE p.id=m.id AND PKW.id=p.id AND PKW.ps>100;

5)
SELECT ort
FROM Orte o, Mitarbeiter m, Person p, Abteilung a
WHERE o.plz=p.plz AND p.id=m.id AND m.abtnr=a.abtnr AND p.plz=o.plz;
```

9.7.2.2) Die WHERE-Klausel

Neben den Boolschen Funktionen (AND,OR,NOT) sind folgende weitere Angaben zulässig:

IN-Operator

Anstatt eine Reihe von OR-Verknüpfungen durchzuführen, kann auch der IN-Operator verwendet werden.

Bsp:

```
SELECT * FROM ... WHERE typ IN ('österreichisch', 'international');
```

BETWEEN-Operator

Bsp.:

```
SELECT * FROM ... WHERE preis BETWEEN 100 AND 150;

//ist dasselbe wie

SELECT * FROM ... WHERE preis >= 100 AND preis<=150;
```

IS NULL-Operator

Bsp.: Es sollen alle Personen gesucht werden, die beim Attribut TelNr den NULL-Wert eingetragen haben.

```
SELECT * FROM ... WHERE TelNr IS NULL;
```

LIKE-Operator

Der Like-Operator ist ein Vergleichsoperator, der Datenwerte einer Spalte mit einem vorgegebenen Muster vergleicht.

Allgemeine Form:

AttributName [NOT] LIKE 'muster';

Dabei haben in 'muster' zwei Zeichen eine besondere Bedeutung:

- % .... steht für 0 bis beliebig viele Zeichen
- _ .... steht für genau ein beliebiges Zeichen

Bsp.: Alle Speisen deren Name mit 'M' beginnt

```
SELECT Name FROM Speise WHERE Name LIKE 'M%';
```

Bsp.: Alle Restaurants, welche an 2. Stelle im Namen ein 'o' haben

```
SELECT Name FROM Restaurant WHERE Name LIKE '_o%';
```

9.7.2.3) Mengenoperationen

SQL stellt die Mengenoperationen **UNION** (Vereinigung), **INTERSECT** (Durchschnitt) und **EXCEPT** (Mengendifferenz) für typkompatible Relationen (d.h. Relationen mit der gleichen Anzahl von jeweils typkompatiblen Attributen) zur Verfügung.

Beispiel: Wir wollen eine Liste der Namen aller Restaurants und der Namen aller Speisen

```
(SELECT Name FROM Restaurant)
UNION
(SELECT Name FROM Speise)
```

Mit typkompatiblen Relationen sind dabei nicht die Relationen **Restaurant** oder **Speise** gemeint, sondern die Ergebnisse der beiden **SELECT** Statements.

9.7.2.4) Aggregatfunktionen und Gruppierung

Aggregatfunktionen

Es gibt folgende Aggregatfunktionen in SQL:

- MIN
- MAX
- SUM
- AVG
- COUNT

Bsp: Gesucht ist der Preis des teuersten Burgers:

```
SELECT MAX(Preis)
FROM Speise
WHERE Name LIKE '%Burger%';
```

Bsp.: Wie viele Speisen werden im Brauhof angeboten?

```
SELECT COUNT (*)   //COUNT (*) zählt alle Tupel
FROM Restaurant r, Speise s,
WHERE r.rnr = s.rnr AND r.name="Brauhof";
```

Gruppierung

Die Select-Abfrage kann um eine Klause „**GROUP BY** *attributListe*“ und eine Klausel „**HAVING** *bedingung*“ ergänzt werden, um Gruppen von Tupeln zu bilden und auf diese Gruppen Aggregatfunktionen anzuwenden.
Dabei werden diese Tupel nach gemeinsamen Werten in den in der **GROUP BY**-Klausel angeführten

Attributen gruppiert. Von diesen Gruppen werden jene eliminiert, die die in der **HAVING**-Klausel angegebene Bedingung nicht erfüllen.

Anschließend werden etwaige in der **SELECT**-Klausel angegebenen Aggregatfunktionen auf jede der verbliebenen Gruppen angewendet. Diese Aggregatfunktionen (außer **COUNT**) können auch auf arithmetische Ausdrücke über Attributen angewandt werden.

Beispiel

Relation Restaurant			
rnr	**RName**	**Adr**	**Haube**
1	McDonalds	Amstetten	0
2	McDonalds	Ybbs	0
3	Hollywood	Amstetten	1
4	Casa Venezia	Amstetten	2
5	grill.Bar	Amstetten	2
6	Schinakel	Grein	1

Frage: Wie viele Restaurants haben die gleiche Haubenanzahl?

```
SELECT Haube, COUNT (*) AS AnzahlRestaurants
FROM Restaurant
GROUP BY Haube;
```

Ergebnis:

Haube	**AnzahlRestaurants**
0	2
1	2
2	2

Frage: Wie viele Restaurants in Amstetten haben die gleiche Haubenanzahl?

```
SELECT Haube, COUNT (*) AS AnzahlRestaurants
FROM Restaurant
WHERE Adr='Amstetten'
GROUP BY Haube;
```

Ergebnis:

Haube	**AnzahlRestaurants**
0	1
1	1
2	2

Beispiel

Gesucht ist eine SQL-Abfrage, die alle Restaurants (rnr und name) und die Anzahl der jeweils gebotenen Speisen ausgibt:

Relation Restaurant			
rnr	**RName**	**Adr**	**Haube**
1	McDonalds	Amstetten	0
2	McDonalds	Ybbs	0
3	Hollywood	Amstetten	1
4	Casa Venezia	Amstetten	2
5	grill.Bar	Amstetten	2
6	Schinakel	Grein	1

Relation Speise		
rnr	**SName**	**Preis**
1	Hamburger	1
2	Hamburger	1
2	Cheeseburger	1,30
3	Hawaii	8,6
4	Hawaii	9
5	Salat	5
1	Salat	1,80

```
SELECT r.rnr, r.name, COUNT(*) AS Anzahl
FROM Restaurant r, Speise s
WHERE r.rnr=s.rnr
GROUP BY r.rnr, r.name;
```

1. Schritt: JOIN

1	McDonalds	Amstetten	0	Hamburger	1
1	McDonalds	Amstetten	0	Salat	1,80
2	McDonalds	Ybbs	0	Hamburger	1
2	McDonalds	Ybbs	0	Cheeseburger	1,30
3	Hollywood	Amstetten	1	Hawaii	8,6
4	Casa Venezia	Amstetten	2	Hawaii	9
5	grill.Bar	Amstetten	2	Salat	5

2. Schritt: GROUP BY

1	McDonalds	Amstetten	0	Hamburger	1
1	McDonalds	Amstetten	0	Salat	1,80
2	McDonalds	Ybbs	0	Hamburger	1
2	McDonalds	Ybbs	0	Cheeseburger	1,30
3	Hollywood	Amstetten	1	Hawaii	8,6
4	Casa Venezia	Amstetten	2	Hawaii	9
5	grill.Bar	Amstetten	2	Salat	5

3. Schritt: COUNT (*) AS Anzahl (pro Gruppe)

rnr	RName	Anzahl
1	McDonalds	2
2	McDonalds	2
3	Hollywood	1
4	Casa Venezia	1
5	grill.Bar	1

Beispiel

Gesucht sind alle Restaurants (rnr, name), welche mehr als 5 verschiedene (Tipp: Schlüsselwort DISTINCT) Speisen anbieten!

Relation Restaurant			
rnr	**RName**	**Adr**	**Haube**
1	McDonalds	Amstetten	0
2	McDonalds	Ybbs	0
3	Hollywood	Amstetten	1
4	Casa Venezia	Amstetten	2
5	grill.Bar	Amstetten	2
6	Schinakel	Grein	1

Relation Speise		
rnr	**SName**	**Preis**
1	Hamburger	1
2	Hamburger	1
2	Cheeseburger	1,30
3	Hawaii	8,6
4	Hawaii	9
5	Salat	5
1	Salat	1,80

Beispiel

Gesucht ist eine SQL-Abfrage, die die rnr und die Anzahl der Haube jener Restaurants angibt, deren teuerste Speise mehr als 8€ kostet. Weiters werden jeweils die Anzahl der Speisen als auch deren Durchschnittspreis ausgegeben.

Relation Restaurant			
rnr	**RName**	**Adr**	**Haube**
1	McDonalds	Amstetten	0
2	McDonalds	Ybbs	0
3	Hollywood	Amstetten	1
4	Casa Venezia	Amstetten	2
5	grill.Bar	Amstetten	2
6	Schinakel	Grein	1

Relation Speise		
rnr	**SName**	**Preis**
1	Hamburger	1
2	Hamburger	1
2	Cheeseburger	1,30
3	Hawaii	8,6
4	Hawaii	9
5	Salat	5
1	Salat	1,80
4	Hühnerstreifensalat	8,20

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_07:09_07_02

Last update: **2019/05/14 21:00**

9.7.2.1) DQL - ÜBUNGEN

Aufgabe 1 (am PC)

Gegeben sind folgende Relationen:

Relation Rechner

RNr	StudAss	Leist	Speicher
R1	Huber	1	100
R2	Brunner	3	80
R3	Brunner	3	400
R4	Vogt	2	120
R5	Huber	2	500

Relation Programm

PNr	PName	MinLeist	Bereich
P1	DrawIt	1	Grafik
P2	AskIt	3	Datenbank
P3	WriteIt	1	Text
P4	ConnectIt	2	Internet
P5	PaintIt	2	Grafik
P6	StoreIt	3	Datenbank

Relation Installation

RNr	PNr	Platz	Code
R1	P1	500	X
R1	P3	300	z
R2	P6	300	X
R2	P2	200	pp
R3	P1	400	c
R3	P2	100	tt
R3	P3	500	c
R3	P4	200	pp
R3	P5	200	z
R3	P6	100	t
R4	P5	1000	T
R5	P1	200	p
R5	P5	100	ccc

Relation Benutzung

ANr	PNr	Stund
A1	P1	5
A1	P2	3
A2	P1	6
A2	P4	2
A2	P5	5
A3	P1	7
A3	P3	3
A4	P1	1
A4	P4	4

Relation Assistent

ANr	AName	StudAss	Gehalt
A1	Novak	Brunner	3
A2	Dvorak	Vogt	1
A3	Husak	Vogt	1
A4	Pfeiffer	Brunner	2

Welches Ergebnis liefern die folgenden SQL-Abfragen? Überprüfen Sie ihre DQL-Statements anhand der in 2.7.1 erstellten Datenbank namens **„Rechnerverwaltung“** (Sind Fehler enthalten, so sind diese zu markieren)

a)

```
SELECT I.RNr
FROM Installationen I, Programm P
WHERE I.RNr=P.RNr AND Bereich='Grafik';
```

b)

```
SELECT StudAss, SUM(Gehalt) AS Ges
FROM Assistent
GROUP BY StudAss;
```

c)

```
SELECT StudAss, SUM(Gehalt)
FROM Assistent
GROUP BY StudAss
HAVING SUM(Gehalt)>4;
```

d)

```
SELECT SUM(Speicher)
FROM Rechner
WHERE Leist=3;
```

Aufgabe 2

In einem Weinkeller werden viele Weinflaschen (WEIN) gelagert. Jede Weinflasche wurde von einem Winzer (WINZER) befüllt und kann anhand der Datenbank leicht in den Regalreihen (KELLER) des Kellers gefunden werden. Es kann dabei angenommen werden, dass immer ausreichend Platz in den Regalen des Kellers vorhanden ist. Wird eine Flasche aus dem Keller entfernt, wird ein entsprechender Eintrag (PROTOKOLL) vermerkt. Wenn der Eigentümer des Kellers eine Flasche selbst trinkt, wird im Protokoll als Verwendung 'Eigenbedarf' eingetragen.

Relation Winzer

wnr	name	strasse	plz	ort	telefon
1	Lackner Tinnacher	Steinbach 8	4567	Gamlitz	1234567
2	Weingut Prager	Weissenkirchen 48	3610	Weissenkirchen	1234567
3	Weingut Emmerich Knoll	Unterloiben 10	3601	Unterloiben	12344457
4	Weingut F.X. Pichler	Unterloiben 27	3601	Unterloiben	11122233
5	Weingut Spätlese	Weintalstrasse 23	1136	Wien	
6	Freie Weingarten Wachau	Kremstalstrasse 23	3600	Krems	2304002
7	Stiftskellerei				
8	Weingut Biegler	Wienerstrasse 88	3502	Gumpoldskirchen	54564565

Relation Wein

nr	bezeichnung	sorte	jahrgang	preis	anzahl	wnr	position
1	Riesling Kellerberg	Riesling	1999	28.00	24	4	1
2	Loibenberg	Gr. Veltliner	2000	19.00	36	4	2
3	Ried Kreutles	Gr. Veltliner	2000	19.00	15	3	4

Relation Wein							
nr	**bezeichnung**	**sorte**	**jahrgang**	**preis**	**anzahl**	**wnr**	**position**
4	Riesling Smaragd	Riesling	2000	21.00	30	2	5
5	Grauburgunder	Grauburgunder	2003	16.00	72	1	6
6	Morillon	Chardonnay	2003	9.00	55	1	7
7	Riesling Federspiel	Riesling	2003	9.90	80	6	3
8	Chardonnay	Chardonnay	2003	9.00	16	8	8

Relation Keller			
knr	**reihe**	**regal**	**fach**
1	1	1	1
2	1	1	2
3	1	1	3
4	1	2	1
5	1	2	2
6	2	1	1
7	2	1	2
8	2	2	1

Relation Protokoll				
pnr	**nr**	**pDatum**	**verwendung**	**anzahl**
1	1	2003-01-12	Geschenk an Herrn Berger	12
2	3	2003-07-10	Eigenbedarf	2
3	1	2003-07-23	Eigenbedarf	4
4	6	2003-08-14	Geschenk (Frau Kunz)	6
5	1	2003-08-27	Glasbruch	1
6	4	2003-11-03	Korkgeruch	1
7	6	2003-11-03	Eigenbedarf	3

Folgende SQL-Abfragen sind gesucht! Überprüfen Sie ihre DQL-Statements anhand der in 2.7.1 erstellten Datenbank namens **„Winzerverwaltung“**

a) Geben Sie für die Sorte ’Riesling’ die Namen aller Winzer sowie die Flaschenanzahl aus. Sortieren Sie dabei nach der Flaschenanzahl absteigend.

name	**anzahl**
Freie Weingarten Wachau	80
Weingut Prager	30
Weingut F.X. Pichler	24

b) Ermitteln Sie für jeden Winzer den durchschnittlichen Flaschenpreis und die Gesamtanzahl der Flaschen im Keller. Berücksichtigen Sie dabei nur Winzer, von denen bekannt ist, aus welchem Ort sie kommen. Sortieren Sie die Liste nach dem Preis absteigend.

name	**durchschnittspreis**	**gesamtanzahl**
Weingut F.X. Pichler	23.50	60
Weingut Prager	21.00	30
Weingut Emmerich Knoll	19.00	15
Lackner Tinnacher	12.50	127
Freie Weingarten Wachau	9.90	80

name	durchschnittspreis	gesamtanzahl
Weingut Biegler	9.00	16

c) Geben Sie eine Liste aller Weinbezeichnungen sowie den Namen des erzeugenden Winzers aus, von denen im Jahr 2003 keine Flasche getrunken worden ist (Verwendung in Tabelle Protokoll ='Eigenbedarf'). Sie brauchen dabei nur Winzer berücksichtigen, von denen mindestens eine Flasche im Keller vorhanden ist.

nr	bezeichnung	name
2	Loibenberg	Weingut F.X. Pichler
4	Riesling Smaragd	Weingut Prager
5	Grauburgunder	Lackner Tinnacher
7	Riesling Federspiel	Freie Weingarten Wachau
8	Chardonnay	Weingut Biegler

d) Suchen Sie die Winzer, von denen der Kellereigentümer die meisten Flaschen getrunken (Verwendung in Tabelle Protokoll = 'Eigenbedarf') hat. Geben Sie jeweils den Namen des Winzers sowie die Gesamtkosten des von diesem Winzer konsumierten Weines aus.

name	anzahl	kosten
Weingut F.X. Pichler	4	112.00

e) Geben Sie fur jeden Winzer aus,

- wie viele gunstige (Preis ≤ 10 Euro, Preisklasse niedrig),
- wie viele im Mittelfeld (10 Euro bis 20 Euro, Preisklasse mittel) und
- wie viele teure (Preis > 20 Euro, Preisklasse gehoben) Weinflaschen im Keller liegen.

name	anzahl	preisklasse
Freie Weingarten Wachau	80	niedrig
Lackner Tinnacher	55	niedrig
Lackner Tinnacher	72	mittel
Weingut Biegler	16	niedrig
Weingut Emmerich Knoll	15	mittel
Weingut F.X. Pichler	24	gehoben
Weingut F.X. Pichler	36	mittel
Weingut Prager	30	gehoben

f) Erstellen Sie eine Liste, die angibt, wie viele Flaschen jedes in der Datenbank gespeicherten Winzers sich im Keller befinden. Sortieren Sie dabei nach der Flaschenanzahl absteigend.

name	anzahl
Lackner Tinnacher	127
Freie Weingarten Wachau	80
Weingut F.X. Pichler	60
Weingut Prager	30
Weingut Biegler	16
Weingut Emmerich Knoll	15
Weingut Spätlese	

name	anzahl
Stiftskellerei	

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_07:09_07_02:09_07_02_01

Last update: **2019/05/14 21:01**

9.7.3) DML (Data Manipulation Language)

Unsere Tabellen sind angelegt und warten darauf, mit Daten gefüttert bzw. durchsucht zu werden. Lernen Sie im Folgenden, wie das gemacht wird.
Die Data Manipulation Language umfasst insgesamt 3 Arten von Statements:

9.7.3.1) Einfügen von Datensätzen

Bevor Sie mit einer Datenbank arbeiten, sollten zunächst einmal Daten in der Datenbank vorhanden sein.
Um Daten in eine Datenbank einzufügen, benutzen wir den Befehl INSERT. Dessen Aufbau ist recht einfach:

```
INSERT INTO <Tabelle> [(Spalte1, Spalte2,...)] VALUES (<wert1>,
<wert2>,...);
```

Als Beispiel wollen wir einen Datensatz in die Tabelle Administratoren einfügen:

```
INSERT INTO Administratoren (UserID, USER, Passwort) VALUES
(1,'Meister','geheim');
```

Die Zuordnung, welcher Wert in welche Spalte eingefügt wird, geschieht über die jeweiligen Positionen innerhalb der Klammern;
⇒ der erste Wert „Meister“ wird in die erste Spalte „User“ eingefügt.

Wenn wir einen neuen Datensatz anlegen, müssen wir nicht alle Spalten sofort füllen. Die Tabelle Administratoren hat eigentlich drei Spalten. Damit das aber funktioniert und sich das DBMS nicht beschwert, darf die Spalte nicht als **NOT NULL** definiert sein. Oder, wenn sie es ist, muss sie einen Defaultwert aufweisen.

9.7.3.2) Ändern von Datensätzen

Einen Datensatz können wir mit UPDATE ändern - Vorsicht, es ist nicht möglich, damit einen Datensatz einzufügen.

Hier die vollständige Syntax:

```
UPDATE Relation
   SET <Spalte1>=<Wert> [, <Spalte2>=<Wert>, ...]
     [WHERE <Bedingung>]
```

Die WHERE-Bedingung ist optional, Sie haben dieselben Möglichkeiten zum Formulieren wie bei der SELECT-Anweisung. Geben Sie keine Bedingung an, wirkt sich ein Update auf alle aufgezählten Spalten aus.

Wollen wir die Spalte User aller Datensätze auf einen neuen Wert setzen, schreiben wir:

```
UPDATE Administratoren SET Passwort='noch geheimer';
```

Wollen wir nur einen einzelnen Datensatz ändern bzw. nur einige, so müssen wir eine entsprechende WHERE-Klausel formulieren.

```
UPDATE Administratoren
   SET Passwort='ganz geheim'
   WHERE UserID='1';
```

Sie sollten für das Update eines einzelnen Datensatzes unbedingt dessen Primärschlüssel als Bedingung nehmen, wie in diesem Beispiel. Die Bedingung über die Spalte User zu formulieren, würde nur sicher sein, wenn wir beim Einfügen des Datensatzes darauf achten, in dieser Spalte keine Werte doppelt zu haben.

9.7.3.3) Löschen von Datensätzen

Was wir zum Abschluss des Abschnitts über das UPDATE gesagt haben, gilt auch für die DELETE-Anweisung. Benutzen Sie zum Löschen mit DELETE immer eine WHERE-Klausel mit Bezug auf den Primärschlüssel, außer Sie wollen wirklich alle Datensätze einer Tabelle löschen.

Die Syntax zum Löschen eines oder mehrerer Datensätze lautet:

```
DELETE FROM Relation
   [WHERE <Bedingung>]
```

Auch hier ist die WHERE-Klausel optional, wird keine angegeben, werden alle Datensätze gelöscht - die Tabelle ist danach leer.

Um eine Eintrag aus der Administrator-Tabelle zu löschen, schreiben wir:

```
DELETE FROM Administratoren WHERE UserID=1;
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_07:09_07_03

Last update: **2019/05/14 21:01**

9.7.3.1) DML-Übungen

Aufgabe 1

- a) Fügen Sie zur Relation Kunde einige Datensätze ein.(Es soll dabei einen Kunden mit einer ID=1 geben und auch einige, die einen negativen Kontostand haben)
- b) Erstellen Sie eine neue Relation Kunde Mahnung mit den gleichen Attributen wie die Relation Kunde. Fügen Sie anschließend jene Tupel der Relation Kunde in die Relation Kunde Mahnung ein, die einen negativen Kontostand besitzen.
- c) Der Weinhändler möchte jene Kunden, die ihm kein Geld schulden (Kontostand≥0), belohnen und schreibt ihnen einen Betrag von 10 Euro auf ihrem Kontostand gut.
- d) Der Kunde mit der KundenID=1 hat auf die Mahnung des Weinhändlers bezahlt, hat jedoch versichert, dass er nie wieder Kunde bei ihm sein wird. Löschen Sie diesen Kunden aus der Relation Kunde.
- e) Löschen Sie alle Tupel der Relation Kunde Mahnung und danach die gesamte Relation.

Aufgabe 2 (am PC)

Gegeben sind folgende Relationen:

Relation Rechner			
RNr	StudAss	Leist	Speicher
R1	Huber	1	100
R2	Brunner	3	80
R3	Brunner	3	400
R4	Vogt	2	120
R5	Huber	2	500

Relation Programm			
PNr	PName	MinLeist	Bereich
P1	DrawIt	1	Grafik
P2	AskIt	3	Datenbank
P3	WriteIt	1	Text
P4	ConnectIt	2	Internet
P5	PaintIt	2	Grafik
P6	StoreIt	3	Datenbank

Relation Installation			
RNr	PNr	Platz	Code
R1	P1	500	X
R1	P3	300	z
R2	P6	300	X
R2	P2	200	pp
R3	P1	400	c
R3	P2	100	tt
R3	P3	500	c
R3	P4	200	pp
R3	P5	200	z
R3	P6	100	t
R4	P5	1000	T
R5	P1	200	p
R5	P5	100	ccc

Relation Benutzung		
ANr	PNr	Stund
A1	P1	5
A1	P2	3
A2	P1	6
A2	P4	2
A2	P5	5
A3	P1	7
A3	P3	3
A4	P1	1
A4	P4	4

Relation Assistent			
ANr	AName	StudAss	Gehalt
A1	Novak	Brunner	3
A2	Dvorak	Vogt	1
A3	Husak	Vogt	1
A4	Pfeiffer	Brunner	2

Befüllen Sie die in 2.7.1) bereits angelegten Relationen in ihrer MySQL-Datenbank namens **„Filmverwaltung“** mithilfe von DML-Statements.

Aufgabe 3 (am PC)

Gegeben sind folgende Relationen:

Relation Winzer					
wnr	**name**	**strasse**	**plz**	**ort**	**telefon**
1	Lackner Tinnacher	Steinbach 8	4567	Gamlitz	1234567
2	Weingut Prager	Weissenkirchen 48	3610	Weissenkirchen	1234567

Relation Winzer

wnr	name	strasse	plz	ort	telefon
3	Weingut Emmerich Knoll	Unterloiben 10	3601	Unterloiben	12344457
4	Weingut F.X. Pichler	Unterloiben 27	3601	Unterloiben	11122233
5	Weingut Spätlese	Weintalstrasse 23	1136	Wien	
6	Freie Weingarten Wachau	Kremstalstrasse 23	3600	Krems	2304002
7	Stiftskellerei				
8	Weingut Biegler	Wienerstrasse 88	3502	Gumpoldskirchen	54564565

Relation Wein

nr	bezeichnung	sorte	jahrgang	preis	anzahl	wnr	position
1	Riesling Kellerberg	Riesling	1999	28.00	24	4	1
2	Loibenberg	Gr. Veltliner	2000	19.00	36	4	2
3	Ried Kreutles	Gr. Veltliner	2000	19.00	15	3	4
4	Riesling Smaragd	Riesling	2000	21.00	30	2	5
5	Grauburgunder	Grauburgunder	2003	16.00	72	1	6
6	Morillon	Chardonnay	2003	9.00	55	1	7
7	Riesling Federspiel	Riesling	2003	9.90	80	6	3
8	Chardonnay	Chardonnay	2003	9.00	16	8	8

Relation Keller

knr	reihe	regal	fach
1	1	1	1
2	1	1	2
3	1	1	3
4	1	2	1
5	1	2	2
6	2	1	1
7	2	1	2
8	2	2	1

Relation Protokoll

pnr	*nr*	pDatum	verwendung	anzahl
1	1	2003-01-12	Geschenk an Herrn Berger	12
2	3	2003-07-10	Eigenbedarf	2
3	1	2003-07-23	Eigenbedarf	4
4	6	2003-08-14	Geschenk (Frau Kunz)	6
5	1	2003-08-27	Glasbruch	1
6	4	2003-11-03	Korkgeruch	1
7	6	2003-11-03	Eigenbedarf	3

Befüllen Sie die in 2.7.1) bereits angelegten Relationen in ihrer MySQL-Datenbank namens **„Winzerverwaltung“** mithilfe von DML-Statements.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_07:09_07_03:09_07_03_01

Last update: **2019/05/14 21:02**

9.8) SQL-Datenbankzugriff mit PHP

Um auf eine MySQL-Datenbank zuzugreifen, muss zuerst eine Verbindung (***mysqli_connect***) hergestellt werden. Dazu ist der **Host**, auf dem der MySQL-Server läuft, der **Benutzername** sowie das **Kennwort** anzugeben. Anschließend muss noch eine Datenbank auf dem DBS ausgewählt werden (***mysqli_select_db***). Danach können die SQL-Anweisungen folgen (***mysqli_query***). Im Falle einer SQL-Abfrage liefert diese Funktion alle Tupel zurück, welche der Reihe nach mit ***mysqli_fetch_array*** abgearbeitet werden können.

Im Folgenden soll als Beispiel ein Gästebuch realisiert werden. Dazu wurde eine Tabelle **meldung** erstellt, welche die einzelnen Nachrichten aufnehmen soll:

```
CREATE TABLE meldung (
  id INT NOT NULL AUTO_INCREMENT,
  datum DATETIME DEFAULT NULL,
  name VARCHAR(200) DEFAULT NULL,
  eintrag TEXT,
  CONSTRAINT PK_id PRIMARY KEY (id)
);
```

Zusätzlich zu den Daten **datum, name** und **eintrag** wurde ein Attribut **id** eingeführt, welches als Schlüssel dient.

Das folgende Skript **show.php** liest alle Einträge der Datenbank aus und gibt sie in Tabellenform aus:

```
<?php
    $DBHost = "127.0.0.1";
    $DBUser = "root";
    $DBPasswd = "";
    $DBName = "db";

    //Verbindung zu DB-Server herstellen
    $con=mysqli_connect($DBHost, $DBUser, $DBPasswd, $DBName)
      OR die("Konnte DB-Server nicht erreichen!");
    mysqli_select_db($con,$DBName);
?>
<html>
    <head>
        <title>Alle Meldungen</title>
    </head>
    <body>
    <?php
        $erg=mysqli_query($con, "SELECT datum, name, eintrag FROM meldung
ORDER BY datum DESC");
        echo mysqli_error($con);

        while($row=mysqli_fetch_array($erg))
        {
            echo "<table bgcolor=\"#dddddd\" border=\"0\" width=\"600\">
```

```
\n";
            echo "<tr><td>Datum: </td><td>".$row["datum"]."</td></tr> \n";
            echo "<tr><td>Name:</td><td>" .$row ["name"]. "</td></tr> \n";
            echo "<tr><td valign=\"top\">Eintrag:</td> \n";
            echo "<td>".nl2br(htmlentities($row["eintrag"])."</td></tr>
\n";
            echo "</table>\n<br>\n";
        }
    ?>
    <hr><a href="insert.php">neuen Eintrag hinzufügen</a>
    </body>
</html>
```

Datum:	2018-11-15 20:19:20
Name:	Prof. Andreas Lahmer
Eintrag:	Hier sehen sie die SQL-Anbindung mittels PHP in Form eines Gästebuchs!

---

neuen Eintrag hinzufügen

Um neue Einträge für das Gästebuch zu erstellen, existiert ein weiteres Skript insert.php:

```
<?php
    $DBHost = "127.0.0.1";
    $DBUser = "root";
    $DBPasswd = "";
    $DBName = "db";

    //Verbindung zu DB-Server herstellen
    $con=mysqli_connect($DBHost, $DBUser, $DBPasswd)
      OR die("Konnte DB-Server nicht erreichen!");
    mysqli_select_db($con,$DBName);
?>
<html>
    <head>
        <title>Neuer Eintrag in unser Gästebuch</title>
    </head>
    <body>
        <?php

            if(isset($_GET["submit"]))
            {
                $submit = $_GET["submit"];
                $name = $_GET["name"];
                $eintrag = $_GET["eintrag"];

                //Der Submit-Button wurde gedrückt --> die Werte müssen
überprüft werden
                //und bei Gültigkeit in die DB eingefügt werden
```

```
                $DatenOK=1;         //wir gehen prinzipiell von der
Gültigkeit der Daten aus
                $error="";         //es gab noch keine Fehlermeldung bis hier
hier

                if($name=="")    //Kein Name eingegeben
                {
                    $DatenOK=0;
                    $error.="Es muss ein Name eingegeben werden!<br>\n";
                }
                if($eintrag=="")    //Kein Kommentar eingegeben
                {
                    $DatenOK=0;
                    $error.="Ein Eintrag ohne Kommentar macht nicht viel
Sinn!<br>\n";
                }
                if($DatenOK)        //Daten OK -> also in DB eintragen
                {
                                        $timestamp=date("Y-m-d h:i:s",
time());
                    mysqli_query($con,"INSERT INTO eintraege (datum, name,
eintrag) VALUES (\"$timestamp\", \"$name\", \"$eintrag\" );");
                    echo mysqli_error($con);
                    echo "<b>Daten wurden eingetragen.<b>";
                }
                else
                {
                    echo "<h2>Fehler: </h2>\n"; //Fehlermeldung
                    echo $error;
                }
            }

            //Formular
        ?>



        <form action="insert.php" method="GET">
            Name: <input type="text" name="name" size="30" maxlength="200"
value=""><br>
            Text: <br><textarea rows="10" cols="50" wrap="virtual"
name="eintrag"></textarea>
            <br><input type="submit" name="submit" value="Absenden">
        </form>
        <a href="show.php"> Alle Einträge anzeigen</a>
    </body>
</html>
```

Name:
Text:

Absenden

Alle Einträge anzeigen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_08

Last update: **2019/05/14 21:02**

Aufgabe 1

Erstellen Sie eine php-Seite welche dem Benutzer ein Formular (siehe Abbildung) präsentiert. In diesem Formular kann der Benutzer auswählen, welche der 6 Aufgaben (a bis f) aus der DQL-Übung angezeigt werden sollen.



Ergebnis der SQL-Abfrage

name	durchschnittspreis	gesamtanzahl
Weingut F.X.Pichler	23.50	60
Weingut Prager	21.00	30
Weingut Emmerich Knoll	19.00	15
Lackner Tinnacher	12.50	127
Freie Weingärten Wachau	9.90	80
Weingut Biegler	9.00	16

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:09:09_08:2_08_01

Last update: **2019/05/14 21:02**

10) Das Satzsystem La-TeX

- LaTeX - Was ist das?
- Rund um LaTeX
- Installationsanleitung
- Einführungen und Dokumentationen
- Literaturverzeichnis mit BibTeX
- Generierung von Literatureinträgen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10

Last update: **2019/05/14 21:08**

10.0.1) LaTeX

LaTeX ist ein **Textsatzsystem**. Bei LaTeX verfasst man ein Eingabedokument in reinem Text in einem Text-Editor. Dabei schreibt man inhaltliche Fließtexte und spezielle LaTeX-Befehle. Daraus wird ein formatiertes Ausgabedokument (beispielsweise PDF) erzeugt.

Grundprinzipien

Kein WYSIWYG

Im Gegensatz zu anderen Textverarbeitungsprogrammen, die nach dem **What-you-see-is-what-you-get-Prinzip** funktionieren, arbeitet der Autor mit Textdateien, in denen er innerhalb eines Textes anders zu formatierende Passagen oder Überschriften mit Befehlen textuell auszeichnet. Das Beispiel unten zeigt den Quellcode eines einfachen LaTeX-Dokuments. Bevor das LaTeX-System den Text entsprechend setzen kann, muss es den **Quellcode verarbeiten**. Das dabei von LaTeX generierte Layout gilt als **sehr sauber**, sein **Formelsatz als sehr ausgereift**. Außerdem ist die Ausgabe u. a. nach PDF, HTML und PostScript möglich. LaTeX eignet sich insbesondere für **umfangreiche Arbeiten** wie **Diplomarbeiten** und **Dissertationen**, die oftmals **strengen typographischen Ansprüchen genügen** müssen. Insbesondere in der **Mathematik und den Naturwissenschaften** erleichtert LaTeX das Anfertigen von Schriftstücken durch seine **komfortablen Möglichkeiten der Formelsetzung** gegenüber üblichen Textverarbeitungssystemen. Das Verfahren von LaTeX wird auch mit **WYSIWYAF (What you see is what you asked for.)** umschrieben.

Das schrittweise Arbeiten erfordert vordergründig im Vergleich zu herkömmlichen Textverarbeitungen einerseits eine **längere Einarbeitungszeit,** andererseits kann das **Aussehen des Resultats genau festgelegt** werden. Die längere Einarbeitungszeit kann sich jedoch, insbesondere bei Folgeprojekten mit vergleichbarem Umfang oder ähnlichen Erfordernissen, lohnen. Inzwischen gibt es auch **grafische Editoren**, die mit LaTeX arbeiten können und **WYSIWYG** oder **WYSIWYM (What you see is what you mean.)** bieten und **ungeübten Usern den Einstieg deutlich erleichtern** können. Beispiele für LaTeX-Entwicklungsumgebungen sind im Abschnitt Entwicklungsumgebungen aufgelistet.

Logisches Markup

Bei der Benutzung von LaTeX wird ein sogenanntes **logisches Markup** verwendet. Soll beispielsweise in einem Dokument eine Überschrift erstellt werden, so wird der Text **nicht wie in TeX rein optisch hervorgehoben** (etwa durch Fettdruck mit größerer Schrift, also: \font\meinfont=cmb10 at 24pt \meinfont Einleitung), sondern die **Überschrift wird als solche gekennzeichnet** (z. B. mittels \section{Einleitung}). In den Klassen- oder sty-Dateien wird **global festgelegt**, wie eine **derartige Abschnittsüberschrift** zu gestalten ist: „das Ganze fett setzen; mit einer Nummer davor, die hochzuzählen ist; den Eintrag in das Inhaltsverzeichnis vorbereiten“ usw. Dadurch erhalten alle diese Textstellen eine **einheitliche Formatierung**. Außerdem wird es dadurch möglich, automatisch aus allen Überschriften im Dokument mit dem Befehl \tableofcontents ein Inhaltsverzeichnis zu generieren.

Rechnerunabhängigkeit

Wie TeX selbst ist LaTeX **weitestgehend rechnerunabhängig** verwendbar. Das bedeutet, dass es für die meisten Betriebssysteme analog zu TeX auch für LaTeX lauffähige, produktiv einsetzbare LaTeX-Installationen gibt. Zu diesen Betriebssystemen gehören zum Beispiel Microsoft Windows von der Version 3.x bis zur aktuellen Version 10, Apple macOS sowie diverse Linux-Distributionen. Unter der Voraussetzung, dass alle verwendeten Zusatzpakete (siehe unten) in geeigneten Versionen installiert sind, besteht der Vorteil der Verwendung von LaTeX darin, dass das **Ergebnis unabhängig** von der **verwendeten Rechnerplattform und dem verwendeten Drucker** in den **beiden Ausgabeformaten DVI und PDF** hinsichtlich der Schriftarten und -größen sowie der Zeilen- und Seitenumbrüche im Druck stets exakt gleich ist. LaTeX ist hierbei **nicht auf die Schriftarten** des jeweiligen **Betriebssystems angewiesen**. Jene Betriebssystem-Schriftarten sind häufig für die Anzeige am Monitor optimiert. LaTeX enthält eine Reihe eigener Schriftarten, die ihrerseits für den Druck optimiert sind.

Technik, Alternativen, Entwicklung, Lizenz

LaTeX bildet ein „Format“ in einem TeX-spezifischen Sinne. Es besteht im Kern aus einer Datei latex.ltx von Makrodefinitionen, die in ein Speicherabbild latex.fmt nach Ausführung der Definitionen umgewandelt wird. Diese Datei wird für jeden Dokumenterzeugungslauf zuerst von der „Engine“ (einem Binärprogramm wie TeX oder pdfTeX) eingelesen. Andere bedeutende Formate sind plain TeX und ConTeXt. Im Vergleich zu diesen existieren für LaTeX besonders viele Zusatzpakete – weitere Makrosammlungen – die auf den Makros in latex.ltx aufbauen und die zunächst mit der Engine und den Makros in latex.ltx gegebenen Darstellungsmöglichkeiten beträchtlich erweitern. Sie sind zu einem großen Teil von LaTeX-Anwendern zunächst für spezielle eigene Bedürfnisse (Veröffentlichungen, Lehrmaterial, Abschlussarbeiten) entwickelt und dann der Allgemeinheit mit meist freier Lizenz zur Verfügung gestellt worden.[10] Speziell für diesen Zweck wurde die LaTeX Project Public License entwickelt, die auch für latex.ltx und weitere Pakete der Hauptentwickler von LaTeX gilt. Viele solcher Zusatzpakete sind auch im Auftrag von Verlagen, Fachzeitschriften, Fachgesellschaften und Hochschulen (für Abschlussarbeiten) für die Umsetzung ihrer Gestaltungsrichtlinien entwickelt worden.

Wofür verwende ich das?

Für LaTeX gibt es viele Anwendungsbereiche. Um einige zu nennen:

- Wissenschaftliche und vorwissenschaftliche Arbeiten
- Berichte, Artikel und Aufsätze
- Präsentationen, Lebensläufe, Briefe
- Romane, Symphonien, Memoiren
- Und noch viel mehr!

Im Gegensatz zu klassischen Textverarbeitungsprogrammen ergeben sich andere Arbeitsmuster. Das gestalterische Layout und das Niederschreiben der Inhalte sind voneinander getrennt. Üblicherweise ist während der Textbearbeitung das zu erwartende Ergebnis nicht kontinuierlich sichtbar. Stattdessen wird wiederholt (bei Bedarf) ein aktualisiertes Ausgabedokument erzeugt.

Was ist so toll an LaTeX?

- Die **Qualität der erzeugten Dokumente** ist sehr hoch.
- LaTeX unterstützt von Grund auf klar und einheitlich formatierte Dokumente, eine Voraussetzung für **wissenschaftlich orientierte Arbeiten**. Es verleitet im Gegensatz zu anderen Textverarbeitungsprogrammen weniger dazu, unglücklich gestaltete Layouts und unstrukturierte Dokumente zu verfassen.
- Für **mathematische Formeln** ist LaTeX überragend gut.
- Eingabedateien werden in **reinem Text** verfasst. Dieses Dateiformat ist unverwüstlich. Es kann in beliebigen Text-Editoren bearbeitet werden. Inkompatibilitäten gibt es quasi nicht.
- LaTeX ist **erweiterbar und flexibel**. Dokumente können aus Einzel-Modulen erstellt werden. Das eignet sich besonders gut für **verteilte Arbeiten** (Gruppenarbeiten).
- LaTeX ist robust und stabil - unabhängig von der Länge des Dokuments.
- Ausgabeformate beinhalten Standards wie PDF, HTML, PostScript und DVI – Formate die sich für Druck und digitale Verteilung eignen.

Und obendrauf:

- **LaTeX ist freie Software!** Frei bedeutet: „Open Source“, frei benutzbar und nebenbei gratis.
- LaTeX ist für **viele Computerplattformen** verfügbar (Windows, Unix, Linux, Mac OS X, …).
- LaTeX fordert nur milde Systemanforderungen.
- LaTeX ist sehr **gut dokumentiert**. Es gibt unzählige Anleitungen für beinahe jeden denkbaren Anwendungsfall, von den eigenen Memoiren bis hin zum Fernschachspiel.

LaTeX vs. Microsoft Word

Worin besteht nun der Unterschied zwischen Latex und Office Werkzeugen? Nun, in Latex konzentriert man sich lediglich auf den Inhalt des Textes und nicht auf die Formatierung. Die Formatierung übernimmt Latex. Deshalb wirkt eine Latexbedienoberfläche auch oft sehr einfach, denn man gibt hauptsächlich den Text seines Werkes ein. Damit Latex den Text richtig formatieren kann, muss es natürlich ein paar Hinweise bekommen. Latex kann nicht erraten, ob es sich bei einer Überschrift auch wirklich um eine Überschrift handelt oder ob es ein normaler Satz ist, bei dem lediglich das abschließende Satzzeichen vergessen wurde. Deshalb muss man alles kennzeichnen, was nicht normaler Text ist. Dazu gehört z. B.:

- Überschriften, z. B. von Kapiteln oder Abschnitten
- hervorgehobene Worte, z. B. in Fettdruck
- wenn man z. B. ein Bild, eine Tabelle oder eine mathematische Formel einfügen möchte
- und natürlich gibt es noch viele Ausnahmen

Man führt also keine Formatierung durch. Das bedeutet auch, dass man die Kapitel nicht nummeriert. Dies tut Latex automatisch. Aus den Überschriften kann natürlich auch ein Inhaltsverzeichnis automatisch erzeugt werden. Man kann komfortabel Querverweise in den Text einfügen. Eine ganz große Stärke von Latex ist weiterhin die Verwaltung von Literaturverzeichnissen. Man gibt lediglich an einer zentralen Stelle die Daten der Quelle (also z. B. eines Buches) ein und fügt dann an den entsprechenden Stellen im Text lediglich einen Querverweis auf das Werk ein. Latex erstellt dann automatisch das komplette Literaturverzeichnis (nach den entsprechenden DIN Normen). Weiterhin

kann man relativ komfortabel ein Glossar oder ein Stichwortverzeichnis erstellen. Die Verwaltung von Fußnoten ist ebenfalls kinderleicht. Weiterhin kann man sich für seine Grafiken und Tabellen ein entsprechendes Abbildungs- bzw. Tabellenverzeichnis automatisch erstellen lassen.

Latex bietet somit sehr viel Funktionalität. Natürlich muss man auch etwas dafür tun, nämlich sich mit Latex beschäftigen. Ich glaube, dies ist das größte Hindernis für Latex. Viele Computernutzer sind an die Grundfunktionen von Office Paketen gewöhnt und sind lieber bereit sich mit diesen durch viele Seiten Text zu quälen, als sich die Zeit zu nehmen und ein besseres Werkzeug (für lange Texte) kennen zu lernen. Man kann sicher davon ausgehen, dass man die in die Latexerlernung investierte Zeit sehr schnell durch höhere Produktivität wieder herausholen kann. Andererseits macht es keinen Sinn zum Schreiben eines Briefes erst Latex zu lernen.

Was brauche ich dafür?

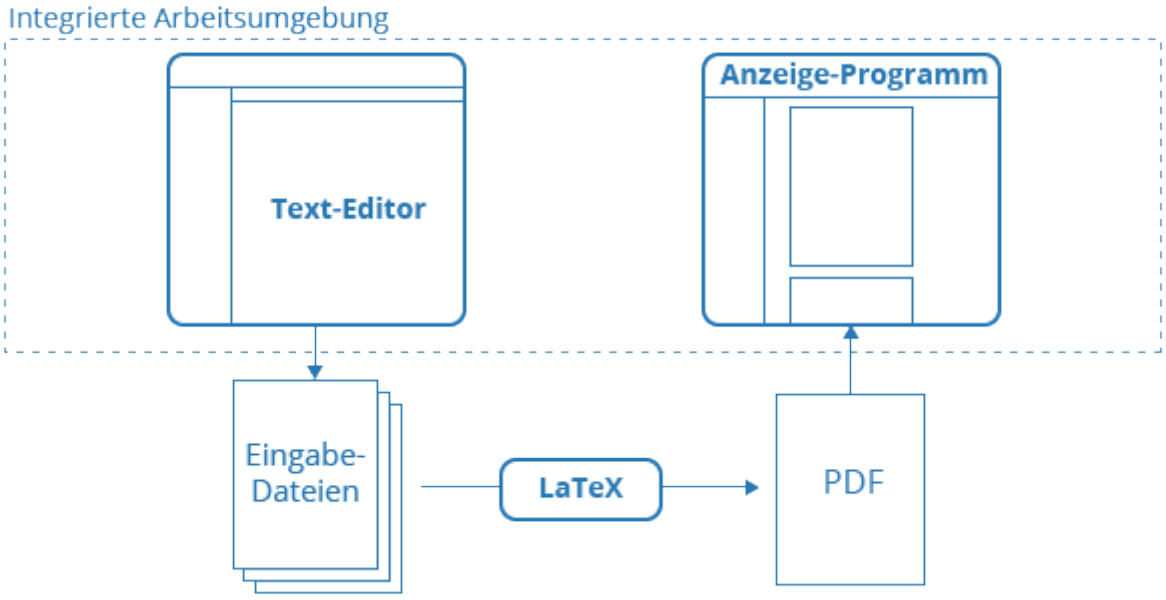
Bislang haben wir einfach nur von **LaTeX** gesprochen. LaTeX an sich ist eigentlich ein Kommandozeilen-Programm, dessen einzige Funktion darin besteht, Eingabedateien in ein Ausgabeformat wie PDF zu überführen. Damit lassen sich aber keine Eingabedateien erstellen oder Ergebnisse anzeigen.

Für die grundlegende Arbeit mit LaTeX benötigt man üblicherweise drei Komponenten:

- **LaTeX-Distribution**: ein geschnürtes Paket aus allen notwendigen Grundelementen, wie die LaTeX-Kommandozeilen-Tools, Hilfsprogramme und Erweiterungspakete
- **Text-Editor**: ein Programm zum Erstellen der Eingabedateien (bzw. der Inhalte)
- **Anzeige-Programm**: zur Betrachtung der Ergebnisse

Für Anfänger empfehlen wir die Benutzung einer für LaTeX optimierten, **integrierten Arbeitsumgebung**: sie vereint die Funktionen von Text-Editor und Anzeige-Programm. Fortgeschrittene Benutzer bevorzugen oft spezielle Text-Editoren. Deren Konfiguration und das Erlernen der Bedienung ist aber gerade anfangs vergleichsweise aufwändig.

Das Zusammenspiel schematisch:



Nun geht's zur Praxis:

LaTeX Einführung

Einführungsbeispiel in LaTeX

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10:10_00:10_00_01

Last update: **2019/05/22 18:13**

10.0.2) Geschichte von LaTeX

- Donald E. Knuth
- Donald E. Knuth - Homepage
- CTAN - Comprehensive TeX Archive Network

Distributionen

- TeX Die Grundversion
- LaTeX Eine Makrosammlung für TeX von Leslie Lamport
  - MiKTeX Eine LaTeX-Distribution für Windows
  - Offizielle MikTeX-Site
  - teTeX Eine LaTeX-Distribution für Linux
  - TeXLive Das am meisten verbreitete System
  - ConTeXt
  - ConTeXt-Site

Arbeitsumgebungen

Name	Linux	Windows
TeXlipse (Eclipse)	ja	ja
KBibTeX	ja	nein
TeXmaker	ja	ja
Kile	ja	nein
LyX	ja	ja
TeXnicCenter	nein	ja
WinEdt	nein	ja
TeXWorks	ja	ja

- Editoren in Ubuntu
- Ein Header mit Anpassungen für verschiedene Distributionen und Zeichensätze

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10:10_00:10_00_02

Last update: **2019/05/14 21:10**

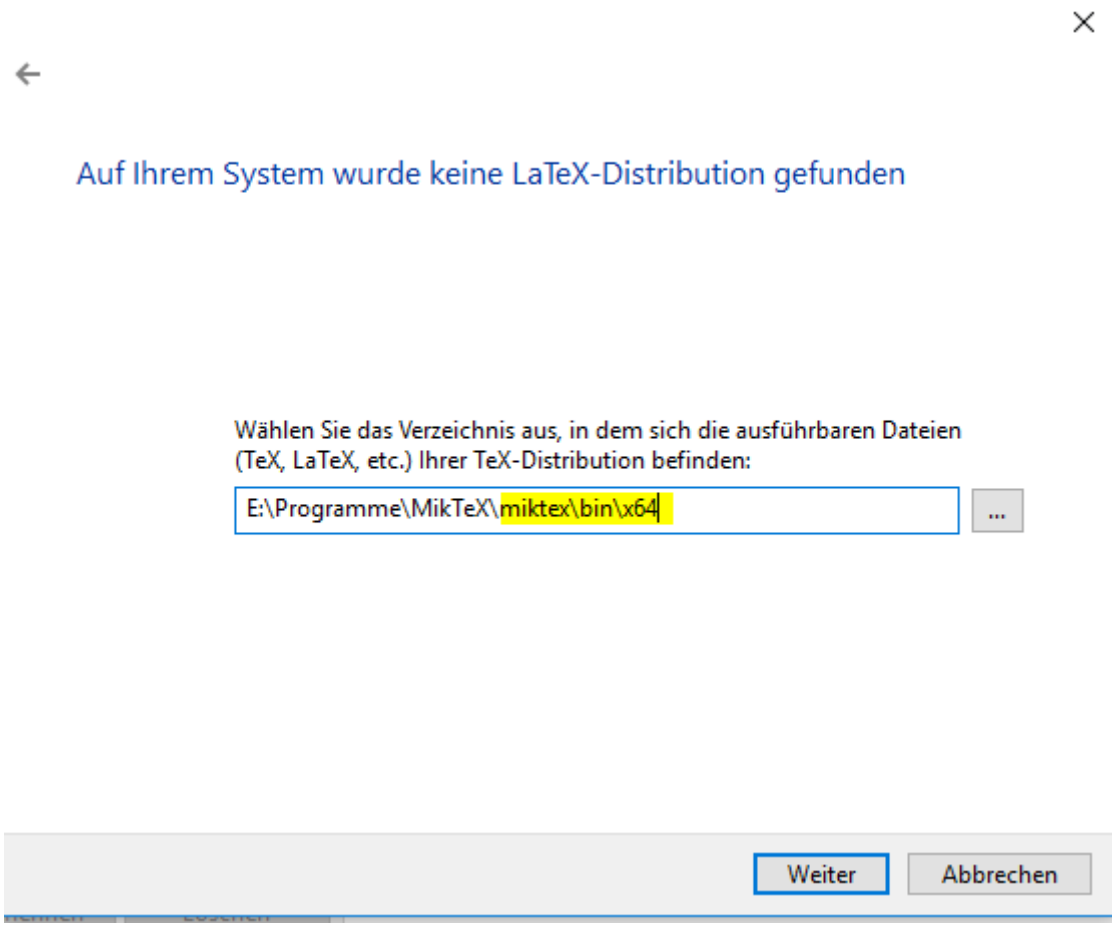
10.0.3) Installation einer LaTeX-Arbeitsumgebung

- MikTeX TeX/LaTeX-Paket herunterladen und installieren (alles einfach bestätigen)
- Sumatra Sumatra-PDF-Viewer herunterladen und installieren
- TeXnicCenter Entwicklungsumgebung/-editor herunterladen und installieren (alles einfach bestätigen)

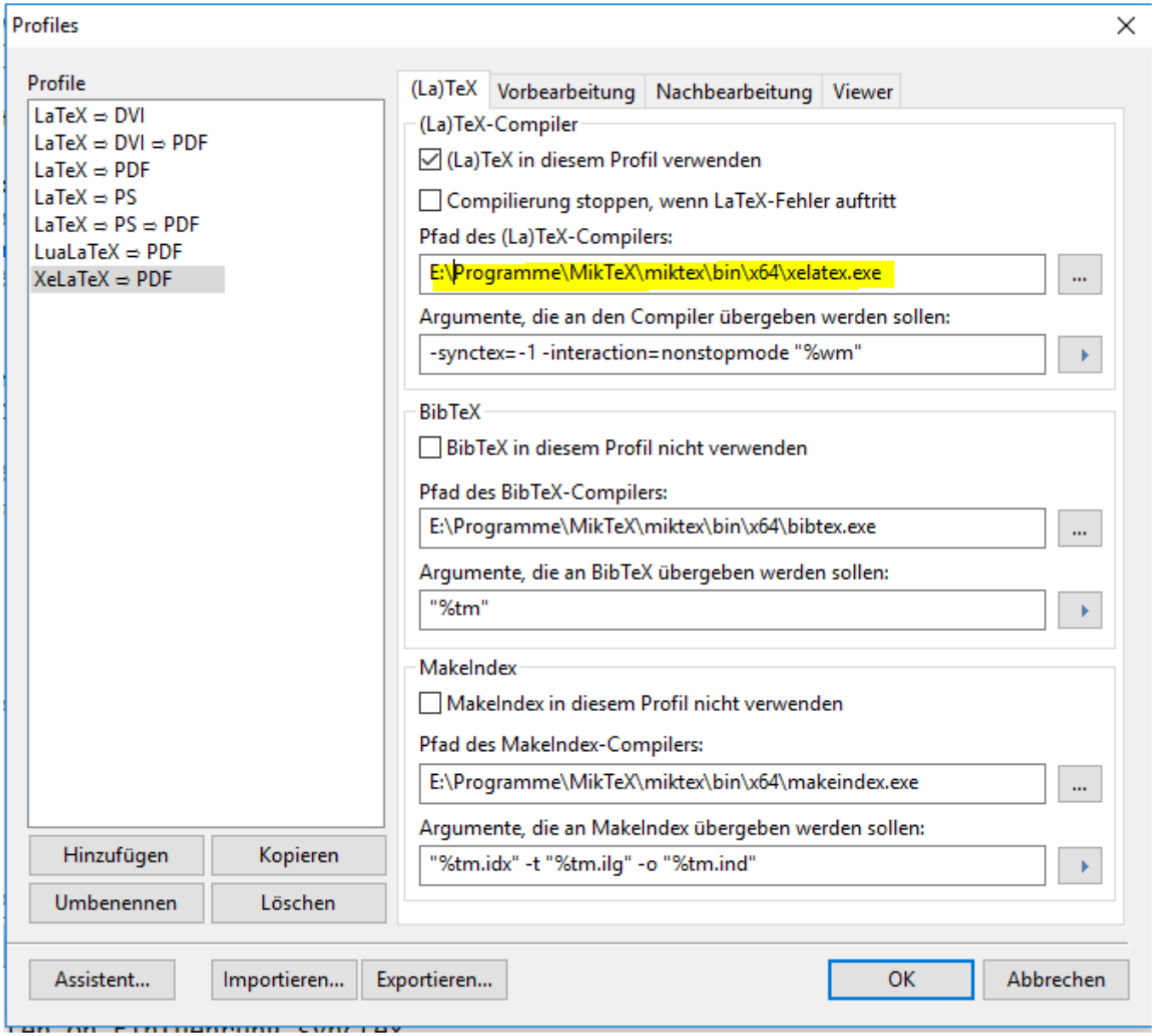
Hinweise:

- Beim ersten Start von Texniccenter erscheint ein Konfigurationsassistent. Bitte konfigurieren Sie diesen wie folgt (**natürlich mit ihren eigenen Pfadangaben!!!**):

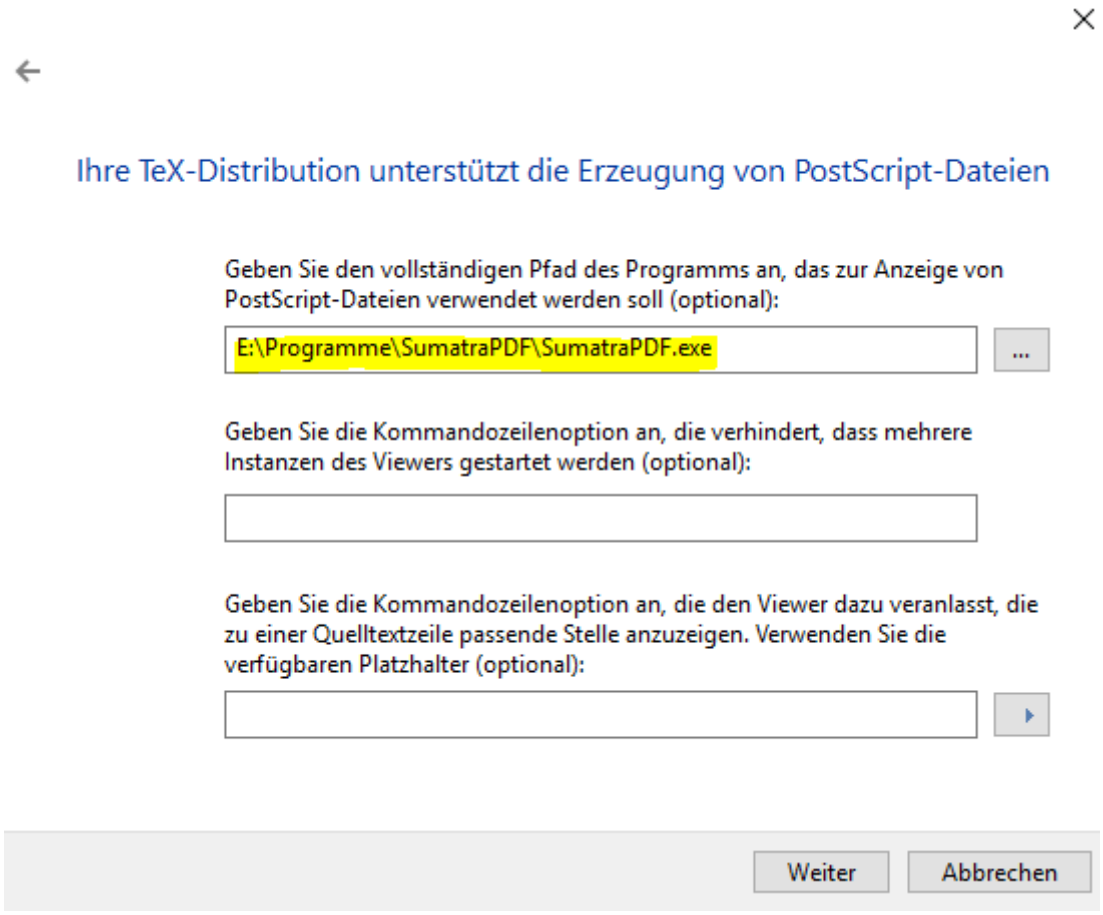
Auswahl der LaTeX Distribution



Angabe des LaTeX-Pfades



Auswahl des PDF-Viewers



- Mit der folgenden Übung kann dann z.B. begonnen werden. Dabei einfach die Projektdatei (Uebung1.tcp) oder die TeX-Datei (Uebung1.tex) im TeXnicCenter laden, als Compiler XeLaTeX⇒PDF (letzter Eintrag) einstellen und Strg-Shift-F5 oder Strg-F5 übersetzen und betrachten. Im Sumatra-PDF-Viewer (der schneller als der Acrobat-PDF-Reader ist) sollte unter „Einstellungen→Optionen“ ganz unten („Befehlszeile für die Inverssuche„) als Editor TeXnicCenter ausgewählt werden. Dann kann man im PDF an eine beliebige Stelle klicken und TeXnicCenter geht dann im Quelltext an die selbe Stelle. Achtung: bei den ersten Compiliervorgängen (mit Strg-Shift-F5 oder Strg-F5) wird meist einiges nachinstalliert, also bitte geduldig sein, das kann etwas dauern.
- Oft gibt es Probleme mit der Anzeige von Umlauten. Mit folgenden Einstellungen (=Präambel) ist man anfangs gut beraten:

```
%Kommentar
%Angabe des Papierformates
\documentclass[a4paper, 11pt]{article}

%%%%%%%%%%%%%%%%%%%%%%%%%% Präambel %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%deutsche Silbentrennung
\usepackage[ngerman]{babel}
%Umlaute
\usepackage[T1]{fontenc}
%Bilder
\usepackage{graphicx}
%kein Einzug bei neuem Absatz
setlength{parindent}{0cm}

%%%%%%%%%%%%%%%%%%%%%%%%%% Metadaten %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
\title{Unser erstes Textdokument über \\ Soziale Netzwerke in \LaTeX}
\author{Andreas Lahmer \\ Matthias Haslauer \thanks{in Kooperation mit Josef Lechner}}
\date{17. Jänner 2018}


%%%%%%%%%%%%%%%%%%%%%%%% Dokument %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Beginn des Dokuments
\begin{document}
...
%Ende des Dokuments
\end{document}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10:10_00:10_00_03

Last update: **2019/05/14 21:11**

10.0.4) Einführungen und Dokumentationen

Einführungen

Empfehlenswerte Einführungen in LaTeX

- Wikibook: LaTeX-Kompendium
- Hubert Partl u.a.: LateX 2ε - Kurzbeschreibung. *Die Einführung* (in LaTeX 2ε), abrufbar (inclusive aller Quelltexte) vom CTAN.
- Manuela Jürgens: LaTeX — Eine Einführung und ein bisschen mehr…. Hrsg. v. d. FernUniversität Hagen. Internet, 2000. (pdf, 1,2 MB) – Einführung in Text- und Formelsatz.
- Manuela Jürgens: LaTeX — Fortgeschrittene Anwendungen. Oder: Neues von den Hobbits …. Hrsg. v. d. FernUniversität Hagen. Internet, 1995. (pdf, 1,4 MB) – Beschreibung zusätzlicher Gestaltungsmöglichkeiten.
- LaTeX-Kurs TU Dresden

Empfehlenswerte Video-Anleitung

- TeX, Teil 1, TeXnicCenter, MikTeX, SumatraPDF 21:00, (Einführung von Dr.Jörn Loviscach)
- TeX, Teil 2, LaTeX-Anweisungen, Formeln, Bilder, Tabellen, Querverweise, Inhaltsverzeichnis 31:55, (Einführung von Dr.Jörn Loviscach)
- TeX, Teil 3, Literaturangaben mit BibTeX 10:29, (Einführung von Dr.Jörn Loviscach)

Kompendien

- Wikibook: LaTeX
- Befehlsübersicht

Spezielles

Ein sehr schönes Layout

- ClassicThesis

Symbole und schnelle Zeichenerkennung

- Übersicht über die unterschiedlichsten Symbole
- DeTexify

Formelsatz

- Mathematischer Text/Formeln in WikiPedia und in Content-Management-Systemen

- Mathematischer Text/Formeln in GeoGebra
- Mathematischer Text/Formeln in Word 2013

Von TeX zu HTML und eBook

- Converting LaTeX

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10:10_00:10_00_04

Last update: **2019/05/14 21:11**

10.0) Das Satzsystem La-TeX

- 10.0.1) LaTeX - Was ist das?
- 10.0.2) Rund um LaTeX
- 10.0.3) Installationsanleitung
- 10.0.4) Einführungen und Dokumentationen
- Literaturverzeichnis mit BibTeX
- Generierung von Literatureinträgen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10:10_00:start

Last update: **2019/05/14 21:11**

Fragen	Inhalte	Arbeitsaufträge

10.1) Das Satzsystem La-TeX

- Erläutere den Workflow bei La-TeX.
- Erkläre die Unterschiede zwischen TeX und LaTeX und gib einige LaTeX-Distributionen an.
- Erläutere die Textauszeichnung in La-TeX.
- Wie erstellt man Aufzählungen?
- Erläutere die Formelgenerierung in La-TeX.
- Erläutere die Einbettung von Abbildungen in La-TeX.
- Erläutere die Erstellung eines Inhaltsverzeichnisses in La-TeX.
- Erläutere die Erstellung eines Literaturverzeichnisses in La-TeX.
- Wie erstellt man mit La-TeX Tabellen?
- Wie bettet man Programmquelltexte in La-TeX ein?

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10:10_01

Last update: **2019/05/14 21:06**

10.2) Das Satzsystem La-TeX

Arbeitsauftrag 1

Erstelle mit dem Satzprogramm LaTeX einen Text zu einem beliebigen (mathematischen, physikalischen oder informatischen) Thema, der Folgendes umfasst

a) verschiedene Textauszeichnungen (fett, kursiv, Typwriter, Kapitälchen, usf)
b) verschiedene Schriftgrößen
c) Inhaltsverzeichnis
d) mind. 2 Abbildungen
e) mind. 2 Inline Formeln, mind. 2 abgesetzte Formeln
f) eine Tabelle mit mindestens drei Spalten und zwei Zeilen
g) einen Programmquelltext
h) mind. 2 Verweise auf andere Textstellen
i) Literaturverzeichnis (mit mind. 2 Literaturzitate, die im Text vorkommen).

Arbeitsauftrag 2

Untersuche, wie in WikiPedia oder DokuWiki Formeln erstellt werden können.
Was passiert, wenn aus WikiPedia eine Formel über die Zwischenablage in den La-TeX-Editor kopiert wird?
Gib einen kurze Bericht bzw. eine Demonstration.

Arbeitsauftrag 3

Untersuche, wie in GeoGebra Formeln erstellt werden können.
Gib einen kurze Bericht bzw. eine Demonstration.

Projekt Formatvorlage für VWA

Für das Erstellen einer VWA soll eine Formatvorlage erstellt werden. Zusätzlich soll in LaTeX eine Handbuch verfasst werden, das folgende Punkte beinhaltet

- Installation der LaTeX Arbeitsumgebung unter Windows (mit Sumatra-PDF als Viewer)
- Tipps für Anfänger
- Tipss zur Anpassung der Formatvorlage an die Bedürfnisse der AnwenderInnen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:10:10_02

Last update: **2019/05/14 21:07**

11) Webdesign Grundlagen

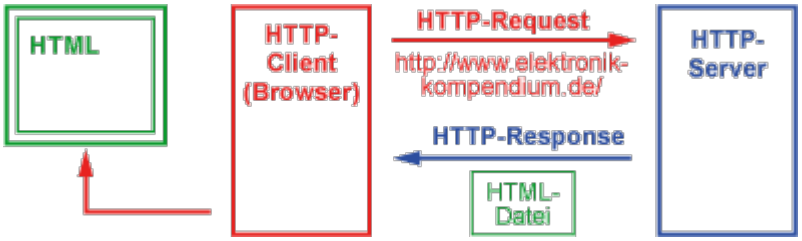
- 11.1) HTML
- 11.2) CSS
- 11.3) Javascript

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11

Last update: **2019/05/22 18:57**

11.1) HTML (Hyper Text Markup Language)



Aufbau des Grundgerüsts

Ein valides HTML5-Dokument, also eine Webseite, die dem vom W3C vorgegebenen Standard entspricht, besteht mindestens aus der Angabe des Dokumententyps und aus folgenden Elementen:

- **html**
- **head**
- **title**
- **body**

Als Grundgerüst bezeichnet man ein HTML-Dokument, das nur diese Elemente enthält. Es wird folgendermaßen notiert:

```
<!DOCTYPE html>
<html lang="de">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Titel der Seite | Name der Website</title>
  </head>
  <body>
    <!-- Sichtbarer Dokumentinhalt im body -->
  </body>
</html>
```

Doctype

Es gibt von HTML verschiedene Versionen (HTML 1 bis HTML 4.01 und HTML5). Eine Zeit lang wurden Webseiten nicht nur in HTML, sondern auch in XHTML geschrieben. Die Dokumenttyp-Deklaration am Anfang eines jeden HTML-Dokumentes informiert über die Art des Dokumentes.

In HTML5 können Sie die Dokumenttyp-Deklaration rigoros vereinfachen:

Beispiel: Angabe einer Dokumenttyp-Deklaration

```
<!DOCTYPE HTML>
```

Empfehlung: Erstellen Sie HTML5-Dokumente.

- Jeder Browser „versteht“ diesen Doctype.
- Sie können sowohl die neuen HTML5-Elemente nutzen als auch bestehenden HTML4-Code weiterverwenden.
- Der doctype ist kürzer und übersichtlicher

Das HTML-Element

Das html-Element umschließt das gesamte Dokument. Die Struktur eines HTML-Dokumentes kann wie ein Stammbaum dargestellt werden. Ausgangspunkt für diese Struktur ist immer das HTML-Element, weswegen man auch vom Wurzelelement (eng. root element) spricht.

Im html-Element muss mit dem lang-Attribut die Sprache des Seiteninhalts angegeben werden; das ist wichtig für die allgemeine Benutzbarkeit der Seite. (für deutschsprachige Seiten: <html lang=„de“>; für englischsprachige „en“)

Es wird unterteilt in

- head - Informationen über die Seite, die von Programmen wie z. B. Suchmaschinen oder Browsern ausgewertet werden
- body - die eigentlichen, vom Browser darzustellenden Informationen

Der head: unsichtbar, aber unverzichtbar

Das head-Element (deutsch: Kopf) ist in allen HTML-Standards ein Pflicht-Element und enthält Informationen über den im body notierten Text.

Zwingend ist mindestens die Angabe eines Namens, daher muss jedes valide HTML-Dokument einen Titel haben.

der Titel

Obwohl der title nicht auf der eigentlichen Webseite ausgegeben wird, erscheint er an vielen Stellen. Hier einige Beispiele:

- Browser zeigen ihn in der Registerkarte (Tab) an,
- Microsoft Windows blendet ihn in der Taskleiste als Tooltip ein,
- Suchmaschinen verwenden ihn in der Regel als Bezeichnung für einen Treffer
- Nutzern von Vorleseprogrammen wird der Titel beim Wechsel auf eine Seite als erstes Vorgelesen.

Dem Titel kommt also eine besondere Bedeutung zu und er sollte daher aussagekräftig sein. Um die einzelnen Seiten eindeutig unterscheiden zu können, empfiehlt es sich außerdem, einen Titel nicht für zwei oder mehr HTML-Dokumente innerhalb eines Webauftrittes zu verwenden. Vergeben Sie besser für jede Seite einen einmaligen Namen.

Beispiel: title -Element im head

```
<head>
  <title>aussagekräftiger Seitentitel</title>
</head>
```

Zeichencodierung

In HTML müssen Sie wie bisher die Zeichencodierung festlegen.[2] Am einfachsten erreichen Sie dies durch die Angabe:

Beispiel: Zeichencodierung

```
<meta charset="utf-8">
```

Beachten Sie: Wenn Sie für HTML5-Dokumente die Zeichencodierung durch <meta charset=„utf-8“> festlegen, müssen Sie darauf achten, dass sich diese Angabe vollständig innerhalb der ersten 1024 Bytes des Dokuments befindet. Des Weiteren muss das Dokument auch mit dieser Zeichencodierung vom Server ausgeliefert werden, dafür einfacherweise auch in derselben Zeichencodierung abgespeichert sein.

Viewportangabe

Für mobile Geräte ist wichtig, dass die Webseite responsiv ist:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

Angabe, dass der Viewport eingestellt wird, aber dennoch skaliert werden darf.

Der body: sichtbarer und strukturierter Aufbau

Was Sie in das body-Element schreiben, wird im Hauptbereich des Browsers dargestellt, der für die eigentliche Webseite reserviert ist (Viewport). In diesen Bereich kommen alle auf der Webseite sichtbaren Informationen.

- 11.1.1) Schichtenmodell einer Webseite
- 11.1.2) Navigation des Webauftritts
- 11.1.3) Bereiche definieren und positionieren
- 11.1.4) Sound und Videos einbinden
- 11.1.5) Document Object Model
- 11.1.6) HTML5

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01

Last update: **2019/05/22 19:04**

11.1.1) HTML und CSS und Scripts

Trennungen - Struktur, Präsentation, Verhalten

Teile und herrsche - diese Devise gilt auch Im Webdesign. Anders gesagt, beim Erstellen einer Webseile genügt uns eine einzige Formulierungsmögkeit nicht, um alle anfallenden Aufgaben abdecken zu können. Vielmehr greifen wir für unterschiedliche Teilbereiche auf entsprechend spezialisierte Sprachen zurück, die uns das jeweils geeignete Werkzeug an die Hand geben.

Lassen Sie uns diese Teilbereiche kurz umreißen. Es sind dies (meist auch in dieser Reihenfolge genannt) im Einzelnen:

Beschreibung der Struktur

Die Beschreibung der Struktur wird durch eine Sprache übernommen, die hierarchisch geordnete Informationen möglichst semantisch (ihrer Bedeutung entsprechend} darzustellen in der Lage ist, Im Webdesign geschieht dies durch HTML oder XHTML. Im Volksmund werden HTML und XHTML oft als „Programmiersprachen“ bezeichnet, was allerdings nicht zutrifft. Bei beiden handelt es sich vielmehr um Auszeichnungs- oder Markup-Sprachen.

Beschreibung der Präsentation

Die in HTML beschriebene Struktur muss optisch präsentiert werden, wobei die Darstellung gemäß des User Agents (Browser, Mobilgerät, Drucker) unterschiedlich erfolgen muss (eben darum darf die Definition der Darstellung nicht Teil de] Struktur sein). Für dieses Gebiet ist CSS die Sprache der Wahl.

Auch CSS ist keine Programmiersprache. Hier wird lediglich beschreibend festgelegt, welche Elemente aufweiche Weise dargestellt werden und in welchem Kontext welche der definierten Darsteilungsanweisungen einzusetzen ist. Man bezeichnet CSS daher auch ab Präsentationssprache. Da kein Prozess, sondern ein Ergebnis beschrieben wird, nennt man dies auch deklarative Sprache.

Beschreibung des Verhaltens

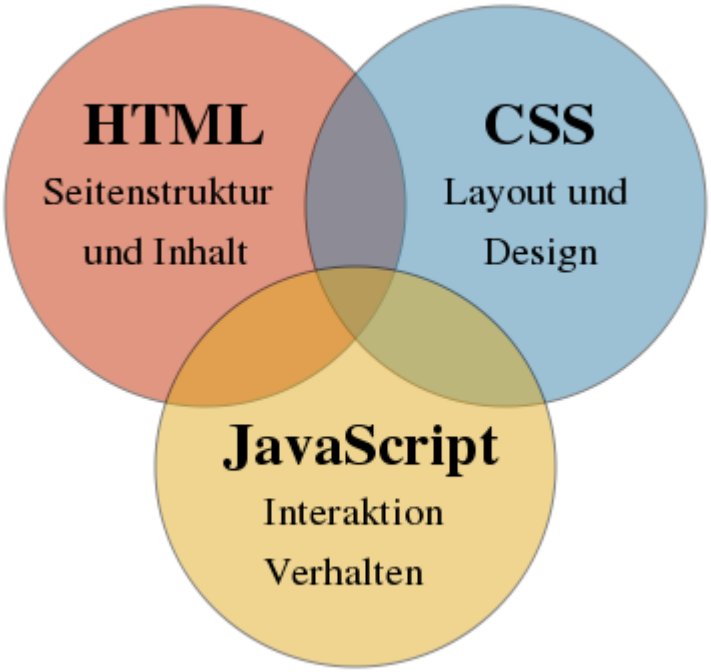
Interessant wird es, sobald eine Interaktion mit dem dargestellten Dokument möglich wird. Hier kommt JavaScript ins Spiel, mit dessen Hilfe Useraktionen erkannt und auf sie reagiert werden kann.

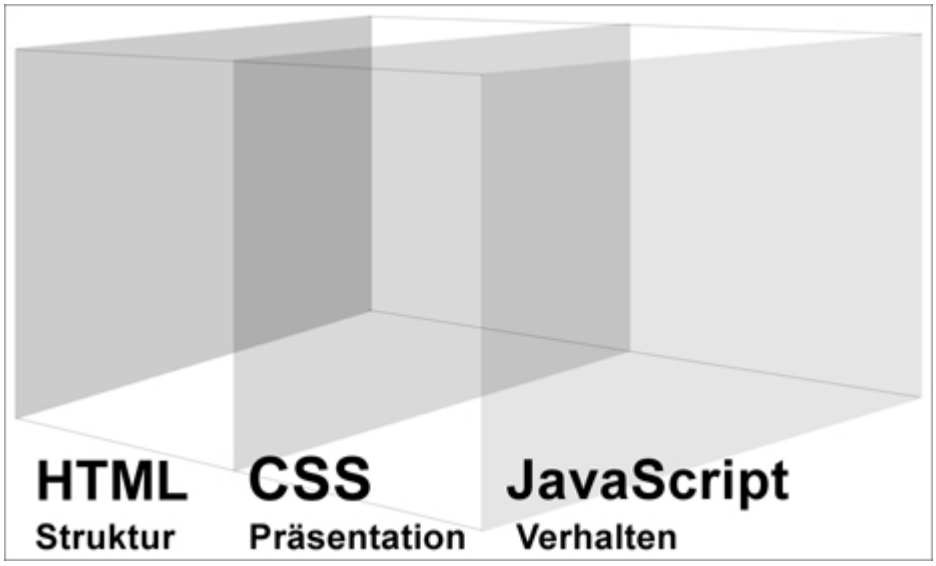
Bei JavaScript handelt es sich tatsächlich um eine ausgewachsene Programmiersprache. Mit seiner Hilfe kann man die eigentlich statischen Struktur- und Präsen-tationsaspekte dynamisch machen, also (auch nachträglich) sowohl das Aussehen als auch die Inhalte eines Dokuments verändern. Ein JavaScript Framework wie jQuery hilft Ihnen bei den dabei anfallenden, komplexen Programmieraufgaben.

Das Schichtenmodell

Da diese drei Aspekte aufeinander aufbauen (mit der Struktur als Basis), kann man sie auch als Schichtenniodell betrachten (siehe Abbildung). Hier wird deutlich, dass die HTML-Struktur die Basis bildet, auf der die darüberliegenden Schichten aufsetzen.

Tatsächlich liest der Browser auch zunächst diese Struktur ein, die er anschließend gemäß den CSS-Anweisungen »dekoriert;' und schließlich mit dem in Javascript beschriebenen Verhalten ausstattet. Setzen wir uns kurz mit den Grundlagen von HTML auseinander.





Die drei Ebenen - Struktur, Präsentation und Verhalten

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_01

Last update: **2019/05/22 18:26**

11.1.2) Navigation des Webauftritts

Die Navigation planen

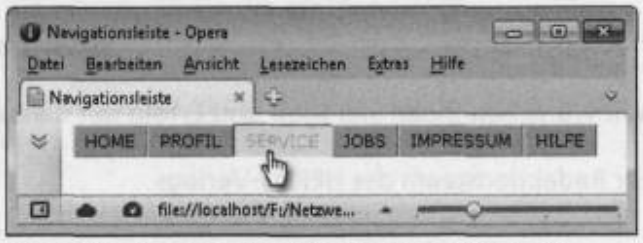
Navigationselemente, Navigationsleisten, Navigationsmenüs

Ein Webauftritt, auch Website oder Site genannt, besteht aus mehreren miteinander verknüpften HTML-Seiten und eventuell aus Ressourcen wie PDF-Dokumenten oder downloadbaren Dateien, z. B. ZIP-Archiven. Diese HTML-Seiten und Ressourcen werden mithilfe von Navigationselementen, z. B. Text-Hyperlinks und Buttons, miteinander vernetzt. Sobald der Nutzer auf ein Navigationselement klickt, steuert er die damit verknüpfte Webseite oder andere Ressource an.

Häufig werden die Navigationselemente (Text-Hyperlinks oder Buttons) zu übersichtlichen Navigationsleisten oder Navigationsmenüs zusammengefasst. Die meisten Webauftritte verfügen über eine oder mehrere solcher Navigationsleisten oder -menüs, mit deren Hilfe sich der Nutzer durch den Webauftritt bewegen kann.



Einstufige horizontale Navigationsleiste mit Text-Hyperlinks



Einstufige horizontale Navigationsleiste mit Buttons



Zweistufiges horizontales Navigationsmenü

Navigationselemente richtig platzieren

Eine richtige Platzierung dieser Gesamtnavigation, die aus Navigationsleisten oder -menüs und

zusätzlichen, im Textinhalt vorhandenen Hyperlinks bestehen kann, erleichtert dem Nutzer die Bewegung im Webauftritt.

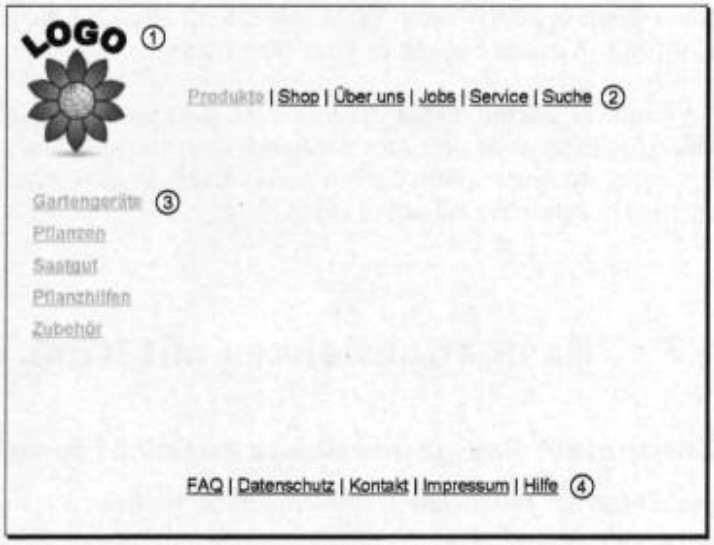
Im Lauf der Zeit haben sich bestimmte Konventionen für die Platzierung der Navigationselemente herausgebildet. Diese Konventionen sollten Sie beachten, damit sich der Nutzer schnell und problemlos in Ihrem Webauftritt zurechtfindet.

① In der linken oberen Ecke befindet sich ein Link zur Startseite. Meist ist er in Form des Firmen- oder Webauftritt-Logos gestaltet.

② Die wichtigsten Sprungverweise werden in Form einer Navigationsleiste oder eines Navigationsmenüs horizontal entlang des oberen Randes erwartet. Möglich ist auch eine vertikale Anordnung entlang des linken Rands.

③ Bei stark untergliederten Webseiten wird die horizontale Navigation häufig als Hauptnavigation zu den übergeordneten Themen des Webauftritts verwendet. Die vertikale Navigation dient in diesem Fall als ergänzende Unternavigation, die zu den verschiedenen Seiten des aktuellen Unterthemas führt (siehe nebenstehende Abbildung).

④ Mit einer weiteren Navigationsleiste am unteren Seitenrand kann der Nutzer ergänzende Seiten mit FAQ, Kontaktmöglichkeiten, Impressum usw. ansteuern.



Typische Anordnung von Navigationselementen

Der früher häufig genutzte rechte Seitenbereich wird heute nicht mehr für die Hauptnavigation verwendet, weil hier nicht gewährleistet ist, dass die Navigationselemente bei allen Browsereinstellungen und Bildschirmgrößen sichtbar sind.

Muss ein Besucher beim Besuch eines Webauftritts überlegen, wo er sich gerade befindet und wohin er gelangen kann, steigt seine Bereitschaft, den Webauftritt zu verlassen.

Navigationselemente gestalten

- **Symbole benennen und verwenden** Navigationselemente sollen möglichst sofort als solche erkennbar und allgemein verständlich sein. Versuchen Sie, eindeutige Hyperlinkbeschriftungen und/oder grafische Symbole zu finden, die möglichst jeder Nutzer versteht.
  Ein typisches Beispiel ist das Briefsymbol, das im Web zum Standard für einen E-Mail-Hyperlink geworden ist. 
- **Einheitlichkeit** Damit der Nutzer schon nach kurzer Zeit problemlos durch Ihren Webauftritt navigieren kann, sollten sich die wichtigsten Navigationselemente stets an derselben Stelle der Seiten befinden. Darüber hinaus sollten sie auch auf jeder Seite gleich gestaltet sein, um eine intuitive Orientierung zu ermöglichen. Eine solche konsistente Navigation nennt man persistente oder globale Navigation.
- **Barrierefreiheit** Bei der Planung der Navigationselemente sollten Sie auch berücksichtigen, auf welche Weise diese von den Suchmaschinen erfasst und von alternativen Ausgabegeräten - wie etwa Screenreadern für sehbehinderte Nutzer - dargestellt werden. Im Idealfall können Ihre

> Navigationselemente von allen Browsern, Suchmaschinen und alternativen Ausgabegeräten problemlos erfasst und ausgewertet werden. Man spricht dann von einer vollständigen Barrierefreiheit.

Mit Adobe Flash oder JavaScript realisierte Navigationsmenüs sorgen eventuell dafür, dass nur die Startseite Ihres Webauftritts indiziert wird. Manche Ausgabegeräte haben Schwierigkeiten mit der Auswertung solcher Navigationsmenüs. Ähnliches gilt für Imagemaps. Wenn Sie sich für Navigationselemente mit JavaScript, Java oder Flash entscheiden, sollten Sie deshalb für alternative Navigationsmöglichkeiten sorgen, die von allen Ausgabegeräten angezeigt und von den Suchmaschinen erfasst werden können.

Barrierefreiheit durch Trennung von Struktur und Darstellung

Navigationsleisten und -menüs aus Text- oder grafischen Hyperlinks ohne JavaScript sind barrierefrei. Deshalb konzentrieren wir uns in diesem Kapitel auf die Erstellung solcher Navigationselemente. Sie lernen, wie Sie ausschließlich in HTML und CSS erstellte Navigationsleisten und -menüs gestalten. Dabei verwenden Sie für die Strukturierung der Navigation reinen HTML-Code und für die attraktive Darstellung als Navigationsleiste oder -menü CSS-Formatierungen.

Diese strikte Trennung in der Vorgehensweise ist eine Vorbedingung für die problemlose Gestaltung barrierefreier Systeme: Entwickeln Sie zuerst die Inhalte und die Funktionalität bis zum Ende, testen Sie sie und beschäftigen Sie sich danach mit der Gestaltung.

Ein weiterer Vorteil dieser Lösungen ist, dass sie schnell und ohne großen Aufwand anpassbar sind, besonders wenn Sie externe Stylesheets verwenden. Wenn sich das Site-Design ändern soll, müssen Sie nur einige CSS-Definitionen ändern, um für ein neues Erscheinungsbild Ihrer Navigation zu sorgen, sie beispielsweise statt vertikal nun horizontal anzuzeigen.

Navigationsleisten mit HTML und CSS erstellen

Horizontale Navigationsleiste aus nicht formatierten Text-Hyperlinks erstellen

Die einfachste Möglichkeit, eine Navigationsleiste zu erstellen, ist die Aneinanderreihung von Text- oder grafischen Hyperlinks.

Beispiel: Text-Navigationsleiste ohne Formatierung

Quelltext

textnavigation.html

```
<!doctype html>
<html>
```

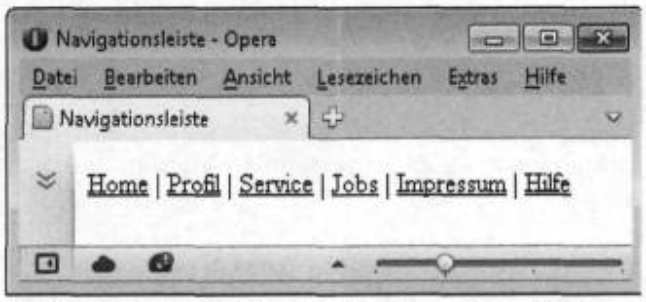
```
<head>
<meta charset="utf-8">
<title>Navigationsleiste</title>
</head>

<body>
<p><a href="home.html">Home</a> |
<a href="profil.html">Profil</a> |
<a href="service.html">Service</a> |
<a href="jobs.html">Jobs</a> |
<a href="impress.html">Impressum</a> |
<a href="hilfe.html">Hilfe</a></p>
</body>
</html>
```

Der folgende Code und die zugehörige Abbildung zeigen eine horizontale Navigationsleiste aus Text-Hyperlinks (<a> ...):

```
<p><a href="home.html">Home</a> |
<a href="profil.html">Profil</a> |
<a href="service.html">Service</a> |
<a href="jobs.html">Jobs</a> |
<a href="impress.html">Impressum</a> | <a
href="hilfe.html">Hilfe</a></p>
```

Diese einfache Codestruktur genügt als Grundlage für ausgefeiltere Darstellungen, die in reinem CSS ohne JavaScript o. Ä. gestaltet werden können:



Horizontale Navigationsleiste mit unformatierten Text-Hyperlinks

- Text-Navigationsleisten, die dynamisch auf die Mausaktionen des Benutzers (Zeigen, Klicken) reagieren;
- unterschiedliche Button-Navigationsleisten, wahlweise auch mit Reaktion auf die Mausaktionen des Benutzers.

Der Vorteil dieser Technik ist, dass solche Navigationselemente schnell geladen werden können und immer funktionieren - selbst in nicht CSS-fähigen oder Nur-Text-Browsern. Dort werden sie als Aneinanderreihung normaler Hyperlinks dargestellt.

Text-Navigationsleisten mit CSS gestalten

Die einfachste Möglichkeit, das Aussehen und die Reaktion der oben gezeigten Text-Navigationsleiste textnavigation.htmlzu verbessern, sind die Pseudoklassen `a:link, a:visited, a:hover` und `a:active`. Mit diesen können Sie das Aussehen von Text-Hyperlinks während bestimmter und nach bestimmten Mausaktionen des Anwenders ändern, also sogenannte Mouseover-Effekte erzeugen:

Pseudoklassen ermöglichen die CSS-Formatierung von HTML-Elementen, die sich nicht durch ein eindeutiges HTML-Tag ausdrücken lassen. Grundsätzlich geben Sie zuerst das jeweilige Element an, im Beispiel das Tag <a> für einen Hyperlink. Anschließend notieren Sie einen Doppelpunkt und eine

definierte Angabe wie etwa link für einen noch nicht besuchten Hyperlink. Das Aussehen der Pseudoklassen legen Sie wie gewohnt mit-hilfe von CSS-Eigenschaften fest.

a:link	Der Hyperlink im ursprünglichen Zustand, bevor der Anwender ihn angeklickt oder mit der Maus darauf gezeigt hat
a:visited	Der Hyperlink auf eine bereits besuchte Seite
a:hover	Der Hyperlink beim Hovern (wenn der Benutzer mit der Maus darauf zeigt)
a:active	Der Hyperlink auf die aktuelle Seite

Achten Sie darauf, bei der Definition der a-Pseudoklassen die in der Tabelle gezeigte Reihenfolge einzuhalten, damit sich die Angaben nicht gegenseitig außer Kraft setzen.

Beispiel: Text-Navigationsleiste mit Mouseover-Effekten

Quelltext

mouseover.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Menüleiste</title>
<style type="text/css">
<!--
a:link,a:visited,a:hover,a:active {
 font-family:Verdana, Arial, Helvetica, sans-serif;
 font-size:small;
 text-decoration:none;
}
a:link {
 color:#339;
}
a:visited {
 color:#909;
}
a:hover {
 color:#C00;
}
a:active {
 color:#999;
}
-->
</style>
</head>
<body>
<p><a href="home.html">Home</a> |
<a href="profil.html">Profil</a> |
<a href="service.html">Service</a> |
```

```
<a href="jobs.html">Jobs</a> |
<a href="impress.html">Impressum</a> | <a
href="hilfe.html">Hilfe</a></p>
</body>
</html>
```

Im folgenden Beispiel versehen Sie die zuvor erzeugte einfache Text-Navigationsleiste mit Mouseover-Effekten, sodass sie optisch auf die Aktionen des Benutzers reagiert.

- Öffnen Sie die Beispieldatei textnavigation.html.
- Geben Sie innerhalb des Tagpaars <head> … </head> den nachfolgenden Code ein.
  Zu Übungszwecken legen Sie die CSS-Stildefinitionen damit in einem eingebetteten Stylesheet fest.

In der Praxis würden Sie statt eines eingebetteten eher ein externes Stylesheet erstellen und im Seitenkopf der einzelnen HTML-Seiten darauf verweisen. Dann könnten Sie bei Bedarf alle Navigationsleisten der Webseiten, in denen auf das externe Stylesheet verwiesen wird, gleichzeitig neu formatieren. Sie müssten dazu nur die Stildefinitionen in der externen Stylesheet-Datei abändern.

```
<style type="text/css">
<!--
① a:link,a:visited,a:hover,a:active {
      font-family:Verdana, Arial, Helvetica, sans-serif;
      font-size:small;
② text-decoration:none;
}
③ a:link {
      color:#339;
}
a:visited {
      color:#909;
}
a:hover {
      color:#C00;
}
a:active {
      color:#999;
}
-->
</style>
```

① Zuerst definieren Sie alle Eigenschaften, die sämtlichen Hyperlink-Zuständen gemeinsam sind: die Schriftfamilie (`font-family`) und die Schriftgröße (`font-size`). Zu diesem Zweck fassen Sie alle vier Pseudoklassen für die einzelnen Zustände des `<a>`-Tags zu einer einzigen Stildefinition zusammen. Die einzelnen Pseudoklassen trennen Sie dabei durch ein Komma.

② Mit `text-decoration:none` sorgen Sie dafür, dass alle vier Zustände der Hyperlinks im Browser ohne Unterstreichung dargestellt werden.

③ Anschließend notieren Sie die vier Pseudoklassen noch einmal gesondert und geben jeweils die Merkmale an, die sich bei den verschiedenen Hyperlink-Zuständen ändern sollen, im Beispiel die Textfarbe (`color`).



Text-Navigationsleiste mit formatierten Hyperlinks

Button-Navigationsleisten mit CSS gestalten

Die einfache Text-Navigationsleiste im Beispiel textnavigation.html kann mit CSS so formatiert werden, dass die einzelnen Hyperlinks das Aussehen von dreidimensionalen Buttons annehmen. Auch dazu verwenden Sie die Pseudoklassen `a:link, a:visited, a:hover` und `a:active`.

Beispiel: Button-Navigationsleiste

Quelltext

3d-navigation.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Navigationsleiste</title>
<style type="text/css">
<!--
a:link, a:visited, a:hover, a:active {
 border-style:solid;
 border-width:thin;
 font-family:Verdana, Arial, Helvetica, sans-serif;
 font-size:small;
 padding:4px 8px;
 text-align:center;
 text-decoration:none;
 text-transform:uppercase;
}
a:link {
 background-color:#BDB76B;
 border-color:#F0E68C olive olive #F0E68C;
 color:#000;
}
a:visited {
 background-color:#BDB76B;
 border-color:#F0E68C olive olive #F0E68C;
 color:#8B008B;
```

```
}
a:hover {
 background-color:#F0E68C;
 border-color:olive #F0E68C #F0E68C olive;
 color:#FF8C00;
}
a:active {
 background-color:#BDB76B;
 border-color:#F0E68C olive olive #F0E68C;
 color:#F0E68C;
}
-->
</style>

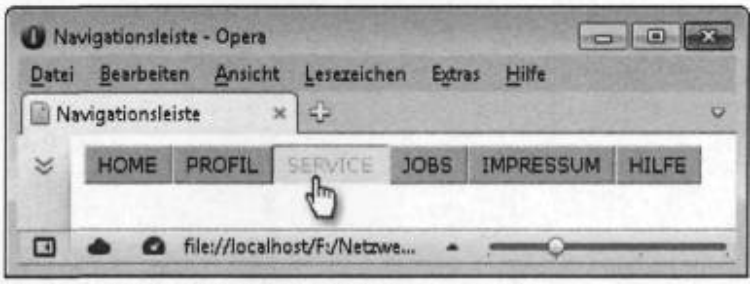
</head>
<body>
<p><a href="home.html">Home</a><a href="profil.html">Profil</a><a
href="service.html">Service</a><a href="jobs.html">Jobs</a><a
href="impress.html">Impressum</a><a href="hilfe.html">Hilfe</a></p>
</body>
</html>
```

Im folgenden Beispiel versehen Sie die verschiedenen Hyperlink-Zustände der einfachen Text-Navigationsleiste textnavigation.html mit unterschiedlichen Füllungen und Rahmenfarben, sodass sie wie grafische Buttons aussehen, die beim Anklicken eingedrückt erscheinen und ansonsten hervortreten.

- Öffnen Sie die Beispieldatei textnavigation.html.
- Löschen Sie die senkrechten Striche (|) zwischen den einzelnen Hyperlinks.
- Geben Sie innerhalb des Tagpaars <head> … </head> den nachfolgenden Code ein.

```
<style type="text/css">
<!--
① a:link, a:visited, a:hover, a:active {
      border-style:solid;
      border-width:thin;
      font-family:Verdana, Arial, Helvetica, sans-serif;
      font-size:small;
      padding:4px 8px;
      text-align:center;
      text-decoration:none;
      text-transform:uppercase;
}
② a:link {
      background-color:#BDB76B;
      border-color:#F0E68C olive olive #F0E68C;
      color:#000;
}
③ a:visited {
      background-color:#BDB76B;
      border-color:#F0E68C olive olive #F0E68C;
      color:#8B008B;
}
④ a:hover {
      background-color:#F0E68C;
      border-color:olive #F0E68C #F0E68C olive;
      color:#FF8C00;
}
⑤ a:active {
      background-color:#BDB76B;
      border-color:#F0E68C olive olive #F0E68C;
      color:#F0E68C;
}
-->
</style>
```

① Wie im vorigen Beispiel *mouseover.html* definieren Sie zuerst alle Eigenschaften, die sämtlichen Hyperlink-Zuständen gemeinsam sind, beispielsweise die Schriftmerkmale, die Rahmenart (`solid` = durchgezogen) und die Rahmenbreite (`thin`). Für einen entsprechenden Abstand zwischen Text und Buttonkanten sorgt die Eigenschaft `padding`.



Button-Navigationsleiste in HTML und CSS

② Jetzt folgen die Eigenschaften für den Hyperlink im Normalzustand. Mit `background-color` bestimmen Sie die Hintergrundfarbe des Buttons, mit `color` die Textfarbe. `border-color` definiert die Rahmenfarbe: Der linke und der obere Rahmen werden khaki (`#F0E68C`), der rechte und der untere Rahmen olivfarben. Durch die Helligkeitsunterschiede entsteht der dreidimensionale Eindruck.

③ Nach demselben Muster formatieren Sie den besuchten Hyperlink. Ändern Sie nur die Textfarbe (`color`).

④ Der Link beim Hovern erhält eine deutlich unterschiedliche Hintergrundfarbe. Ändern Sie auch die Textfarbe. Für die Rahmenfarbe (`border-color`) kehren Sie die Verhältnisse um: Der rechte und der untere Rahmen erhalten das hellere Khaki (`#F0E68C`), der linke und der obere Rahmen das dunklere Oliv.

⑤ Für den aktiven Link wählen Sie eine jeweils andere Hintergrund- und Textfarbe. Die Rahmenfarbe stellen Sie wieder ein wie beim Normalzustand und beim besuchten Hyperlink.

Button-Navigationsleisten mit CSS und Bilddateien gestalten

Bei Bedarf können Sie die zuvor gezeigte Technik zur Gestaltung von Button-Navigationsleisten mit CSS durch Bilder ergänzen. Dadurch ergeben sich weitere Gestaltungsmöglichkeiten, beispielsweise mit Farbverläufen oder Texturen.

Auch hier gilt: Kann das Bild aus irgendeinem Grund nicht geladen werden, schadet dies der Funktionalität der Navigationsleiste nicht.

Beispiel: Button-Navigationsleiste mit Hintergrundbild

Quelltext

3d-navigation-bild.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Navigationsleiste</title>
<style type="text/css">
<!--
a:link, a:visited, a:hover, a:active {
 border-style:solid;
 border-width:thin;
```

```
 font-family:Verdana, Arial, Helvetica, sans-serif;
 font-size:small;
 padding:4px 8px;
 text-align:center;
 text-decoration:none;
 text-transform:uppercase;
}
a:link {
 background-color:#BDB76B;
 border-color:#F0E68C olive olive #F0E68C;
 color:#000;
}
a:visited {
 background-color:#BDB76B;
 border-color:#F0E68C olive olive #F0E68C;
 color:#8B008B;
}
a:hover {
 background-color:#F0E68C;
 background-image:url(burst.jpg);
 border-color:olive #F0E68C #F0E68C olive;
 color:#FFFFFF;
}
a:active {
 background-color:#BDB76B;
 border-color:#F0E68C olive olive #F0E68C;
 color:#F0E68C;
}
-->
</style>

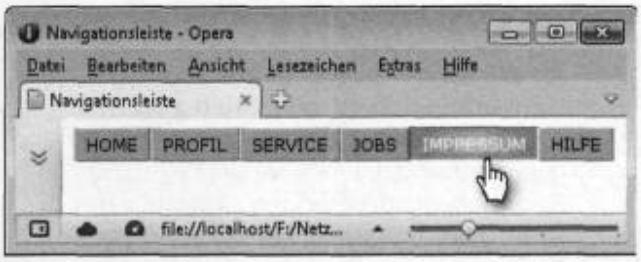
</head>
<body>
<p><a href="home.html">Home</a><a href="profil.html">Profil</a><a
href="service.html">Service</a><a href="jobs.html">Jobs</a><a
href="impress.html">Impressum</a><a href="hilfe.html">Hilfe</a></p>
</body>
</html>
```

Im folgenden Beispiel erhalten die als Buttons dargestellten Hyperlinks beim Hovern ein Hintergrundbild mit einem radialen Farbverlauf.

- Speichern Sie die Beispieldatei `3d-navigation.html` aus dem vorigen Beispiel und die Beispieldatei burst.jpg  in demselben Ordner.
- Öffnen Sie die Datei `3d-navigation.html`
- Ändern Sie die Definition der Pseudoklasse `a:hover` folgendermaßen ab:

```
a:hover {
      background-color:#F0E68C;
      background-image:url(burst.jpg);
      border-color:olive #F0E68C #F0E68C olive;
      color:#FFFFFF;
}
```

① Die Pseudoklasse `a:hover` wurde durch die Eigenschaft `background-image` um das Bild `burst.jpg` ergänzt.



Button-Navigationsleiste mit Hintergrundbild beim Hovern

Navigationsmenüs erstellen



Horizontales Navigationsmenü

Nicht nur Navigationsleisten, sondern auch mehrstufige Navigationsmenüs lassen sich in HTML strukturieren und mit CSS gestalten.

Durch diese Möglichkeit können Sie völlig auf JavaScript, Flash und sonstige Technologien verzichten, die häufig zur Erstellung von Navigationsmenüs verwendet werden. Sie stellen damit sicher, dass Ihre Navigationsmenüs sowohl von Suchmaschinen als auch von den alternativen Ausgabegeräten und von Nur-Text-Browsern problemlos ausgewertet werden können.

Die HTML-Struktur eines Navigationsmenüs erstellen

Als Grundlage für das Navigationsmenü bietet sich eine unsortierte HTML-Liste an.

Beispiel: HTML-Menüstruktur

Quelltext

menuestruktur.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Navigationsmenü</title>
</head>

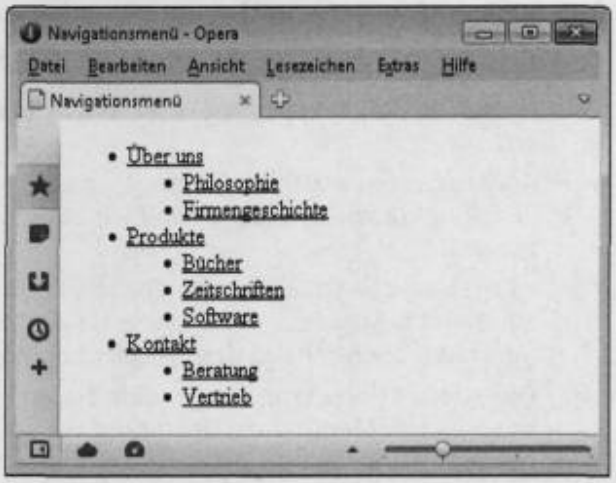
<body>
<ul>
  <li><a href="#">Über uns</a>
    <ul>
      <li><a href="philos.html">Philosophie</a></li>
      <li><a href="firmeng.html">Firmengeschichte</a></li>
    </ul>
  </li>
  <li><a href="#">Produkte</a>
    <ul>
      <li><a href="buecher.html">Bücher</a></li>
      <li><a href="zeitschr.html">Zeitschriften</a></li>
      <li><a href="software.html">Software</a></li>
    </ul>
  </li>
  <li><a href="#">Kontakt</a>
    <ul>
      <li><a href="beratung.html">Beratung</a></li>
      <li><a href="vertrieb.html">Vertrieb</a></li>
    </ul>
  </li>
</ul>
</body>
</html>
```

```
①  <ul>
     <li><a href="#">Über uns</a>
②      <ul>
         <li><a href="philos.html">Philosophie</a></li>
         <li><a href="firmeng.html">Firmengeschichte</a></li>
       </ul>
     </li>
③    <li><a href="#">Produkte</a>
       <ul>
         <li><a href="buecher.html">Bücher</a></li>
         <li><a href="zeitschr.html">Zeitschriften</a></li>
         <li><a href="software.html">Software</a></li>
       </ul>
     </li>
     <li><a href="#">Kontakt</a>
       <ul>
         <li><a href="beratung.html">Beratung</a></li>
         <li><a href="vertrieb.html">Vertrieb</a></li>
       </ul>
     </li>
   </ul>
```

① Geben Sie zunächst alle Hauptmenüpunkte als ungeordnete Liste ein.

② Diejenigen Punkte, die später im Untermenü erscheinen sollen, geben Sie als untergeordnete Liste ein.

③ Um eine vollständige Barrierefreiheit zu erzielen, sollten Sie auch die Hauptpunkte mit Hyperlinks versehen. Denn nur dann lässt sich das Navigationsmenü per Tab-Taste navigieren. Das ist wichtig für eine korrekte Ausgabe auf alternativen Ausgabegeräten. Da die Hauptpunkte keine echten Sprungverweise enthalten, behelfen Sie sich mit Null-Hyperlinks (`href="#"`), Hyperlinks, die zwar als solche erkannt werden, aber nirgendwohin führen.



Fertige Menüstruktur

Von der inhaltlichen Seite her ist das Navigationsmenü fertig. Es funktioniert in allen Browsern und mit sämtlichen alternativen Ausgabegeräten. Sie können sich jetzt mit der Anordnung der Menüpunkte und danach mit der Gestaltung beschäftigen.

Das Navigationsmenü mit CSS anordnen

Für die Anordnung der Menüpunkte verwenden Sie CSS.

Beispiel: CSS-Menüanordnung

Quelltext

menueanordnung.html

```
<!doctype html>
<html>
```

```
<head>
<meta charset="utf-8">
<title>Navigationsmenü</title>
<style type=text/css>
ul {
    font-family:Verdana, Arial, Helvetica, sans-serif;
    font-size:x-small;
    list-style:none;
    margin:0;
    padding:0;
}
li {
    float:left;
    position:relative;
    width:10em;
}
li ul {
    display:none;
    position:absolute;
    top:1.2em;
    left:0;
}
li:hover ul {
    display:block;
}
</style>

</head>

<body>
<ul>
  <li><a href="#">Über uns</a>
    <ul>
      <li><a href="philos.html">Philosophie</a></li>
      <li><a href="firmeng.html">Firmengeschichte</a></li>
    </ul>
  </li>
  <li><a href="#">Produkte</a>
    <ul>
      <li><a href="buecher.html">Bücher</a></li>
      <li><a href="zeitschr.html">Zeitschriften</a></li>
      <li><a href="software.html">Software</a></li>
    </ul>
  </li>
  <li><a href="#">Kontakt</a>
    <ul>
      <li><a href="beratung.html">Beratung</a></li>
      <li><a href="vertrieb.html">Vertrieb</a></li>
    </ul>
  </li>
</ul>
```

```
</body>
</html>
```

Zuerst entfernen Sie die Listenpunkte vor den einzelnen Elementen und definieren Schrift und Positionierung. Mit der Positionierung ordnen Sie die Liste in Menüform an. Erstellen Sie dazu im head-Bereich die folgende CSS-Regel:

```
<style type=text/css>
<!--
① ul {
②       font-family:Verdana, Arial, Helvetica, sans-serif;
        font-size:x-small;
③       list-style:none;
④       margin:0;
        padding:0;
}
⑤ li {
⑥       float:left;
        position:relative;
        width:10em;
}
⑦ li ul {
⑧       display:none;
        position:absolute;
⑨       top:1.2em;
        left:0;
}
⑩ li:hover ul {
        display:block;
}
-->
</style>
```

① Notieren Sie die Definition des Tags `ul` (die unsortierte Liste).

② Geben Sie die gewünschten Schriftformatierungen an.

③ Als `list-style` wählen Sie `none`, um die Aufzählungspunkte vor den einzelnen Elementen zu entfernen.

④ `margin` und `padding` weisen Sie jeweils den Wert `0` zu. Damit bestimmen Sie, dass das Navigationsmenü am linken oberen Rand des Fensters beginnt.

⑤ Diese formatierte Liste verwandeln Sie nun in ein horizontales Menü. Dazu definieren Sie das Tag `li` neu, das die einzelnen Listenpunkte erzeugt.

⑥ Notieren Sie `float:left` mit einer Breite von `10em` und einer relativen Position. Letzteres ist wichtig, damit die verschachtelten Elemente relativ zu den Hauptpunkten angeordnet werden.



Voll funktionsfähige Navigationsleiste

Ohne weitere Angaben werden HTML-Elemente im Browser in der Reihenfolge, wie sie im Quelltext auftauchen, angezeigt. Mit der CSS-Eigenschaft `float` können Sie Elemente jedoch von Text umfließen lassen. Mit `float:left` steht das Element links und wird rechts umflossen. Detaillierte

Informationen über `float` erhalten Sie in Abschnitt 2. Enthält Ihr Navigationsmenü längere Einträge, erhöhen Sie den Wert `10em` entsprechend.

⑦ Als Nächstes definieren Sie die Darstellung der Listenelemente der zweiten Ebene, also der zunächst unsichtbaren Menüelemente. Mit `li ul` erzeugen Sie eine Stildefinition, die nur für innerhalb eines `li`-Tags geschachtelte `ul`-Elemente gilt, also nur für die eingezogenen Unterpunkte.

⑧ Setzen Sie `display` auf `none`, um den initialen Status der eingezogenen Listenelemente auf unsichtbar zu setzen.

⑨ Positionieren Sie die Elemente anschließend absolut mit der Platzierung 1,2 em vom oberen und 0 von den übergeordneten Einträgen.

⑩ Mit der Pseudoklasse `li:hover` machen Sie das Navigationsmenü schließlich funktionsfähig. Mit `li:hover ul` sprechen Sie nur `ul`-Unterpunkte an, sobald sich die Maus über dem übergeordneten Hauptpunkt befindet. Mit `display:block` bestimmen Sie, dass die jeweiligen Unterpunkte beim Hovern über dem Hauptpunkt angezeigt werden.

Das Navigationsmenü mit CSS gestalten

Bisher gefällt das Navigationsmenü sicherlich nur absoluten Design-Puristen. Sie haben aber alle von CSS gebotenen Gestaltungsmöglichkeiten, sodass das fertige Navigationsmenü zum Design Ihres Webauftritts passt.

Beispiel: CSS-Menügestaltung

Quelltext

menuegestaltung.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Barrierefreies Menü</title>
<style type=text/css>
<!--
ul {
 font-family:Verdana, Arial, Helvetica, sans-serif;
 font-size:small;
 list-style:none;
 margin:0;
 padding:0;
}
li {
 background-color:#999;
 border-bottom-color:#F90;
 border-bottom-style:solid;
 border-bottom-width:.1em;
 border-right-color:#F90;
 border-right-style:solid;
 border-right-width:thick;
 float:left;
```

```
 padding:.5em;
 position:relative;
 width:13em;
}
li a {
 color:#FFF;
 text-decoration:none;
}
li a[href=#] {
 font-weight:700;
 text-transform:uppercase;
}
li ul {
 display:none;
 left:0;
 position:absolute;
 top:2.3em;
}
li ul a:hover {
 border-bottom-style:solid;
 border-bottom-width:thin;
 border-left-color:#FC0;
 color:#FC6;
}
li:hover ul {
 display:block;
}
-->
</style>
</head>
<body>
<ul>
  <li><a href="#">Über uns</a>
    <ul>
      <li><a href="philos.html">Philosophie</a></li>
      <li><a href="firmeng.html">Firmengeschichte</a></li>
    </ul>
  </li>
  <li><a href="#">Produkte</a>
    <ul>
      <li><a href="buecher.html">Bücher</a></li>
      <li><a href="zeitschr.html">Zeitschriften</a></li>
      <li><a href="software.html">Software</a></li>
    </ul>
  </li>
  <li><a href="#">Kontakt</a>
    <ul>
      <li><a href="beratung.html">Beratung</a></li>
      <li><a href="vertrieb.html">Vertrieb</a></li>
    </ul>
  </li>
```

```
</ul>
</body>
</html>
```

```
<style type=text/css>
<!--
ul {
     font-family:Verdana, Arial, Helvetica, sans-serif;
     font-size:small;
     list-style:none;
     margin:0;
     padding:0;
}
① li {
     background-color:#999;
     border-bottom-color:#F90;
     border-bottom-style:solid;
     border-bottom-width:.1em;
     border-right-color:#F90;
     border-right-style:solid;
     border-right-width:thick;
     float:left;
     padding:.5em;
     position:relative;
     width:13em;
}
② li a {
     color:#FFF;
     text-decoration:none;
}
③ li a[href=#] {
     font-weight:700;
     text-transform:uppercase;
}
li ul {
     display:none;
     left:0;
     position:absolute;
     top:2.3em;
}
li ul a:hover {
     border-bottom-style:solid;
④    border-bottom-width:thin;
     border-left-color:#FC0;
     color:#FC6;
}
li:hover ul {
     display:block;
}
-->
</style>
```

① Legen Sie im `li`-Tag die gewünschten Hintergrund- und Rahmenformatierungen fest, sodass die Listenelemente als rechteckige Kästchen dargestellt werden.

② Bei Bedarf legen Sie eine Textfarbe (`color`) für die innerhalb der Listenelemente aufgeführten Hyperlinks fest und entfernen mittels `text-decoration:none` die standardmäßige Hyperlink-Unterstreichung.



Mit CSS formatiertes Navigationsmenü

③ In diesem Beispiel wurden außerdem alle Null-Hyperlinks - also die übergeordneten Menüelemente - in Großbuchstaben und fett formatiert.

④ Beim Hovern werden die untergeordneten Listenelemente in einer anderen Farbe und mit den entsprechenden Rahmenattributen angezeigt.

Das horizontale Navigationsmenü mit CSS vertikal ausrichten

Ungefähr genauso häufig wie ein horizontales benötigen Sie ein vertikales Menü. Da in der vorliegenden Lösung Inhalt und Gestaltung strikt voneinander getrennt sind, lässt sich das horizontale mit wenigen Handgriffen in ein vertikales Navigationsmenü konvertieren.

Beispiel: Navigationsmenü mit CSS vertikal ausrichten

Quelltext

menue-vertikal.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Barrierefreies Navigationsmenü</title>
<style type="text/css">
<!--
ul {
 font-family: Verdana, Arial, Helvetica, sans-serif;
 font-size: small;
 padding: 0;
 margin: 0;
 list-style: none;
}
li {
 float: none;
 position: relative;
 width: 13em;
 background-color: #999999;
```

```
 padding: 0.5em;
 border-right-width: thick;
 border-right-color: #FF9900;
 border-right-style: solid;
 border-bottom-width: 0.1em;
 border-bottom-style: solid;
 border-bottom-color: #FF9900;
}
li a {
 color: #FFFFFF;
 text-decoration: none;
}
li a[href="#"] {
 text-transform: uppercase;
 font-weight: bold;
}
li ul {
 display: none;
 position: absolute;
 top: 0em;
 left: 14.4em;
}
li ul a:hover {
 color: #FFCC66;
 border-left-color: #FFCC00;
 border-bottom-style: solid;
 border-bottom-width: thin;
}
li.zeiger ul {
 display: block;
}
li:hover ul{
 display: block;
}

-->
</style>
</head>

<body>
  <ul id="navigation">
    <li><a href="#">&Uuml;ber uns</a>
      <ul>
        <li><a href="philosophie.htm">Philosophie</a></li>
        <li><a href="geschichte.htm">Firmengeschichte</a></li>
        <li><a href="niederlassungen.htm">Niederlassungen</a></li>
      </ul>
    </li>
    <li><a href="#">Produkte</a>
      <ul>
        <li><a href="landschaftsbau.htm">Landschaftsbau</a></li>
```

```
        <li><a href="reitplatzbau.htm">Reitplatzbau</a></li>
        <li><a href="strassenbau.htm">Stra&szlig;enbau</a></li>
      </ul>
    </li>
    <li><a href="#">Kontakt</a>
      <ul>
        <li><a href="beratung.htm">Beratung</a></li>
        <li><a href="vertrieb.htm">Vertrieb</a></li>
      </ul>
    </li>
  </ul>
</body>
</html>
```

```
<style type=text/css>
<!--
ul {
     /*Definition wie zuvor */
li {
     background-color:#999;
     border-bottom-color:#F90;
     border-bottom-style:solid;
     border-bottom-width:.1em;
     border-right-color:#F90;
     border-right-style:solid;
     border-right-width:thick;
①    float:none;
     padding:.5em;
     position:relative;
     width:13em;
}
li a {
     /*Definition wie zuvor */
}
li a[href=#] {
     /*Definition wie zuvor */
}
li ul {
     display:none;
②    left:14.4em;
     position:absolute;
③    top:0;
}
li ul a:hover {
     /*Definition wie zuvor */
}
li:hover ul {
     /*Definition wie zuvor */
}
-->
</style>
```

① Entfernen Sie den Float aus der `li`-Definition.
② In der `li ul`-Definition passen Sie `left` entsprechend der Breite von `li` zuzüglich aller eventuell definierten Rahmenbreiten an.
③ Setzen Sie `top` auf 0.



Vertikales Navigationsmenü

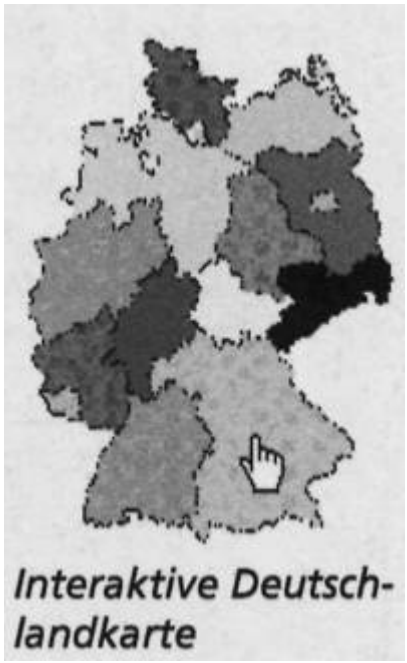
Imagemaps

Was sind Imagemaps?

Als Imagemaps werden Grafiken bezeichnet, in denen bestimmte Bereiche, sogenannte Hotspots, mit einem Hyperlink belegt sind. Diese Bereiche können rechteckig, rund oder polygonförmig (vieleckig) sein. Der Benutzer erkennt den Hyperlink am veränderten Mauszeiger, wenn er den Mauszeiger über das Bild bewegt.

Imagemaps verwenden

Oft werden Land- bzw. Regionalkarten als Imagemaps angeboten. Der Besucher klickt beispielsweise auf eine Region, und die damit verbundene Seite wird geöffnet.


Interaktive Deutschlandkarte

Früher waren Imagemaps als Navigationselemente äußerst beliebt. Man setzte auf grafisch aufwendig gestaltete Designs und machte sich noch nicht viele Gedanken über die Barrierefreiheit der Seiten. In der heutigen Zeit werden Imagemaps kritisch gesehen, denn sie können nicht von allen alternativen Ausgabegeräten interpretiert werden. Zwar können Sie die Hotspots für eine bessere Zugänglichkeit für die alternativen Ausgabegeräte mit Alternativtexten versehen, doch kommen Textbrowser wie Lynx mit einer herkömmlichen Imagemap nicht klar. Dies ist immer ein deutliches Anzeichen dafür, dass die Seite auch für einige alternative Ausgabegeräte nicht zugänglich sein wird. Aus diesem Grund sollten Sie Imagemaps möglichst nur als alternative Navigationsmöglichkeit zu einer barrierefreien Hauptnavigation verwenden.

Imagemaps erzeugen

Sie können Imagemaps aus beliebigen Webgrafiken erzeugen. Die als Imagemap dienende Grafik wird wie ein „normales“ Bild mit dem Tag `img` angegeben. Allerdings versehen Sie das Tag `img` mit dem Attribut `usemap`.

Beispiel: Imagemap mit runden Hotspots

Quelltext

imagemap.html

```
<!doctype html>
<html>
<head>
<link rel="alternate" type="application/rss+xml" href="rss-feed.xml"
title="RSS-Feed">
<link rel="stylesheet" type="text/css" href="../../src/selfhtml.css">
<link rel="shortcut icon" type="image/x-icon"
href="../../src/favicon.ico">
<link rel="author" title="Impressum"
href="../../editorial/impressum.htm">
<link rel="contents" title="Inhaltsverzeichnis"
href="../../navigation/inhalt.htm">
<link rel="index" title="Stichwortverzeichnis"
href="../../navigation/stichwort.htm">
<link rel="search" title="Suche"
href="../../navigation/suche/index.htm">
<link rel="help" title="Hilfe" href="../../editorial/index.htm">
<link rel="copyright" title="Urheberrecht"
href="../../editorial/copyright.htm">
<link rel="top" title="SELFHTML" href="../../index.htm">
<link rel="up" title="HTML-Kopfdaten" href="index.htm">
<link rel="next" title="Durchsuchbarkeit mit Server-Kommunikation"
href="durchsuchbarkeit.htm">
<link rel="prev" title="Adressbasis und Zielfensterbasis"
href="basis.htm">
<link rel="first" title="Titel einer HTML-Datei" href="titel.htm">
<link rel="last" title="Durchsuchbarkeit mit Server-Kommunikation"
```

```
href="durchsuchbarkeit.htm">
<meta charset="utf-8">
<title>Imagemap</title>
<style type="text/css">
.links {
    float: left;
    font-family: Arial, Helvetica, sans-serif;
    width: 200px;
}
</style>
</head>

<body>
<p>
<img src="schaubild.gif" alt="Motivation" width="640" height="587"
usemap="#Schaubild">
  <map name="Schaubild">
    <area shape="circle" coords="307,123,91" href="#"
alt="Technologie">
    <area shape="circle" coords="173,296,97" href="#"
alt="Organisation">
    <area shape="circle" coords="411,420,148" href="#">
  </map>
</p>
</body>
</html>
```



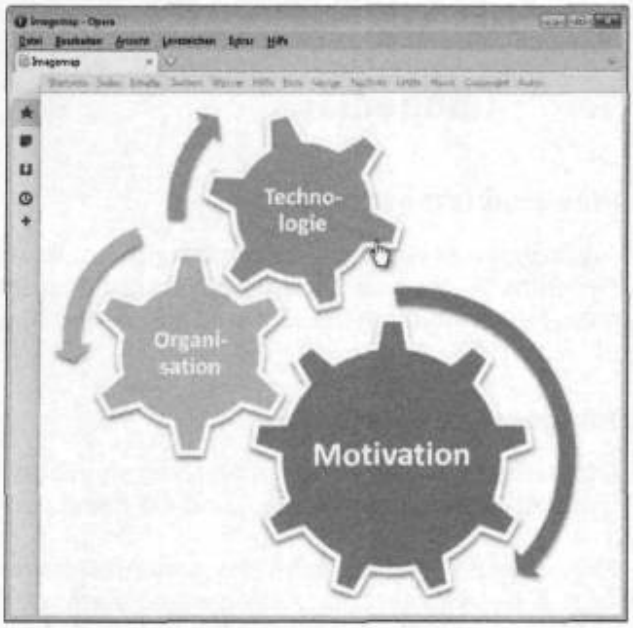
```
<p>
<img src="imagemap.gif" alt="Motivation" width="640" height="587"
usemap="#schaubild">
  <map name="schaubild">
    <area shape="circle" coords="307,123,91" href="tec.html" alt="Technologie">
    <area shape="circle" coords="173,296,97" href="org.html"
alt="Organisation">
    <area shape="circle" coords="412,420,148" href="mot.html">
  </map>
</p>
```

① Um eine Imagemap zu erstellen, verwenden Sie das Tag `img` mit dem Attribut `usemap`. Dieses Attribut benötigt als Wert eine URL-Adresse, wozu hier der Ankername `#schaubild` dient.

② Mit dem Tag `map` geben Sie das im Ankernamen referenzierte Element an, hier `name="schaubild"`.

Der Name darf keine Leerzeichen enthalten und sollte möglichst nur aus Buchstaben, Ziffern und Unterstrichen (_) bestehen.

③ Innerhalb des Tagpaars `<map> … </map>` geben Sie mithilfe von `area`-Tags die Hotspots an. Sie haben hierbei die in der folgenden Tabelle dargestellten Möglichkeiten. Die Koordinaten der Hotspots geben Sie mithilfe des Attributs `coords` in auf die Grafik bezogenen, durch Kommata getrennten Pixelwerten an. Außerdem gehört in das `area`-Tag auch das Verweisziel, das Sie wie bei Hyperlinks üblich mit dem Attribut `href` bestimmen.



Die Imagemap mit dem oben gezeigten Code

`area shape="rectangle"`	Definiert einen rechteckigen Hotspot. Geben Sie die Koordinaten `x1,y1,x2,y2` an. `x1`: linke obere Ecke, von links gemessen `y1`: linke obere Ecke, von oben gemessen `x2`: rechte untere Ecke, von links gemessen `y2`: rechte untere Ecke, von oben gemessen
`area shape="circle"`	Definiert einen kreisrunden Hotspot. Geben Sie Koordinaten `x,y,r` an. `x`: Mittelpunkt, von links gemessen `y`: Mittelpunkt, von oben gemessen `r`: Radius in Pixel
`area shape="polygon"`	Definiert einen vieleckigen (polygonförmigen) Hotspot mit beliebig vielen Ecken. Geben Sie die Koordinaten `x1,y1,x2,y2 … xn,yn` an. `x`: Ecke von links `y`: Ecke von oben

Die benötigten Koordinaten finden Sie am besten heraus, wenn Sie die Grafik in einem Programm wie etwa Adobe Photoshop öffnen. Wenn Sie mit der Maus über die Grafik fahren, zeigt das Info-Bedienfeld Ihnen die Koordinaten des gerade überfahrenen Bereichs an. Außer Photoshop sind auch Freeware-Programme wie etwa The Gimp geeignet.

Achten Sie darauf, auch bei `<area>`-Tags `<alt>`-Attribute zu verwenden, um die Benutzerfreundlichkeit zu verbessern. Falls die Grafik nicht dargestellt werden kann, werden dann stattdessen diese Texte angezeigt.

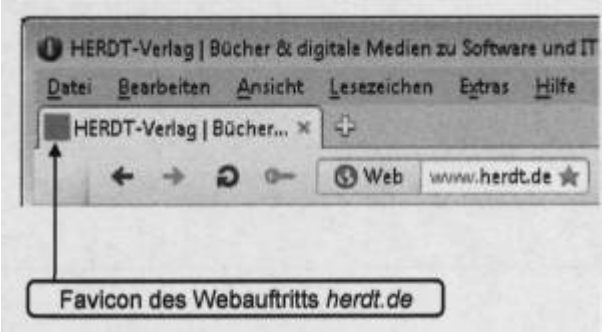
Logische Verlinkung

Hierarchische Beziehungen

Ein Webauftritt besteht gewöhnlich nicht einfach aus einer Anhäufung von Webseiten, die nichts miteinander zu tun haben. Vielmehr haben die meisten Seiten eine hierarchische Beziehung zueinander oder gehören zu einer Abfolge von Seiten (wie zum Beispiel bei einer Bedienungsanleitung). Zu vielen Sites gehören auch Stylesheet-Dateien, Favicons, Feeds usw.

Ein Feed oder Newsfeed ist ein über das Web angebotener Nachrichtenstrom, der mit einem Feedreader gelesen werden kann. Der Nutzer kann damit die Inhalte häufig aktualisierter Websites verfolgen, ohne den Webauftritt selbst besuchen zu müssen. Feeds liegen zumeist im RSS- oder Atom-Format vor.

Ein Favicon ist ein Symbol, das im Browser neben dem Dokumenttitel angezeigt wird.



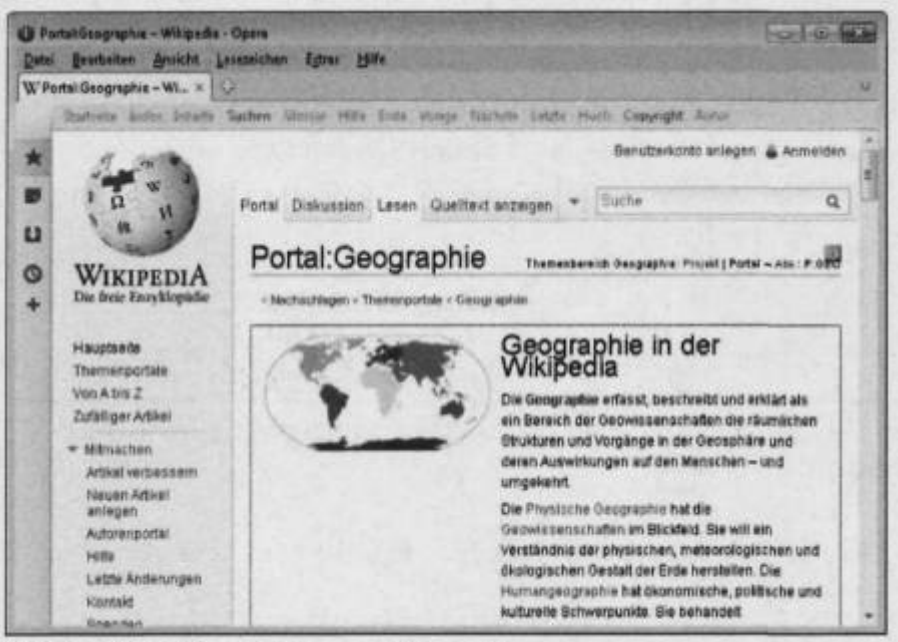
Im head-Bereich Ihrer Webseiten können Sie solche Beziehungen und die zugehörigen Ressourcen wie Stylesheets usw. notieren.

Eines der bekanntesten Einsatzgebiete für die logische Verlinkung ist `<link rel=„stylesheet“>` zum Einbinden eines externen Stylesheets.

Darstellung logischer Verlinkung

In manchen Browsern - zum Beispiel Opera, iCab, Netscape und Lynx - gibt es Schaltflächen, mit denen bestimmte logische Verlinkungen visualisiert und nutzbar gemacht werden. Geben Sie beispielsweise im Kopf Ihrer Seite eine logische Verlinkung zu einer Impressumsseite an und definieren dabei Copyright als Beziehung, wird die vordefinierte Copyright-Schaltfläche des Browsers aktiv und bringt den Nutzer nach einem Klick zu dieser Impressumsseite.

Im Kopfbereich der Wikipedia-Seite sind beispielsweise die Linktypen `link rel=„search“` sowie `link rel=„Copyright“` definiert und mit den URLs zu den entsprechenden Seiten versehen. Deshalb sind in der Opera-Menüleiste die beiden Schaltflächen Suchen und Copyright aktiviert.



Wikipedia-Seite im Browser "Opera"

In Opera zeigen Sie die Menüleiste für die logische Verlinkung mit dem Menübefehl Ansicht - Symbolleisten -Navigationsleiste an. Andere Arten logischer Verlinkung, wie etwa Feeds oder Favicons, werden in der Adresszeile bzw. im Browser-Tab angezeigt. Auch Suchdienste usw. haben es mit logischen Verlinkungen leichter, die Seitenstruktur Ihres Webauftritts zu erfassen. In Zukunft wird diese Möglichkeit sicherlich an Bedeutung gewinnen.

Syntax des link-Elements

`<link rel="Linktyp">`	`<link rel` definiert im Kopfbereich Ihrer Seite eine logische Beziehung. Danach geben Sie mit `rel` die Art der Beziehung an. Die in HTML möglichen Beziehungen werden weiter unten in diesem Abschnitt aufgeführt. `<link rel="author" href="impressum.htm" title="Impressum">` Mit dem Attribut `href` geben Sie die URL der zugehörigen Ressource an, mit dem Attribut `title` optional eine Beschriftung. Weitere Attribute: ✔ `media` bestimmt das Ausgabemedium. ✔ `target` bestimmt das Zielfenster für das Verknüpfungsziel. ✔ `hreflang` gibt die Sprache des Verknüpfungsziels an. ✔ `charset` gibt die Codierung des Verknüpfungsziels an.

Linktypen für logische Verlinkung

Praxisrelevante Linktypen

Momentan sind nur vier Linktypen für die Praxis von größerer Bedeutung:

`<link rel="alternate">`	`rel="alternate"` dient zum Verlinken einer alternativen Darstellungsweise mit demselben Inhalt (beispielsweise ein zugehöriger RSS-Feed innerhalb eines Blogs): `<link rel="alternate" type="application/rss+xml" href="rss-feed.xml" title="RSS-Feed">` Weil es sich bei Feeds um XML-Dokumente handelt, wird im obigen Beispiel das `type`-Attribut mit dem MIME-Typ `application/rss+xml` verwendet. Die meisten Browser zeigen dann rechts außen in der URL-Adresszeile ein Feedsymbol ① an. Klicken Sie dieses an, können Sie den Feed anzeigen. 
`<link rel="icon">`	Damit ordnen Sie Ihrer Seite ein Favicon zu, das normalerweise im ICO-Format von Microsoft vorliegt: `<link rel="icon" type="image/x-icon" href="/favicon.ico" sizes="16x16 32x32">` Geben Sie im Attribut `type` den MIME-Typ an: `image/vnd.microsoft.icon`. Sie können aber auch - wie in diesem Beispiel - `image/x-icon` verwenden. Mit dem Attribut `sizes=` können Sie die Größe des Icons angeben. Das hier verwendete Icon ist in zwei Größen - 16 x 16 Pixel und 32 x 32 Pixel - darstellbar. Die beiden Größenangaben werden durch ein Leerzeichen getrennt.

`<link rel="pingback">`	Dieses Tag dient beispielsweise im Rahmen einer Blogsoftware der Angabe eines Pingback-Servers: `<link rel="pingback" href="http://www.beispiel.de/pingback/xmlrpc">` Mit der Pingback-Technik kann ein Webautor eine Benachrichtigung anfordern, wenn seine Dokumente oder Seiten verlinkt werden. Er erhält vom Pingback-Server eine Nachricht, wenn eine andere Website, die Pings versendet, einen Link auf die aktuelle Seite aufweist. Bei Bedarf kann die verlinkende Seite dann beispielsweise automatisch auf der aktuellen Seite angezeigt werden.
`<link rel="stylesheet">`	Mit diesem Tag wird eine externe Stylesheet-Datei verknüpft: `<link rel="stylesheet" type="text/css" href="sitestyle.css" title="Mein Sitestyle">` Als MIME-Typ geben Sie `type="text/css"` an. Denken Sie daran, im Sinne einer bestmöglichen Barrierefreiheit Ihrer Seiten auch das Attribut `title` hinzuzufügen. Dann kann der Nutzer in manchen Browsern zwischen Ihrem und einem eigenen Stylesheet umschalten.

Mit dem MIME-Typ (MIME = Multipurpose Internet Mail Extensions) werden die Struktur und der Aufbau von E-Mails festgelegt. Der MIME-Typ wird aber auch zur Deklaration von Inhalten in verschiedenen Internetprotokollen verwendet.

Weitere Linktypen

Außerdem gibt es noch weitere Linktypen für die logische Verlinkung, die in der Praxis aber eher selten verwendet werden. Viele davon aktivieren die entsprechende Schaltfläche im Browser, wenn dieser eine Menüleiste für die logische Verlinkung enthält.

Die folgende Tabelle bietet einen Überblick.

`<link rel="alternate">`	`rel="alternate"` dient zum Verlinken einer alternativen Darstellungsweise mit demselben Inhalt (beispielsweise ein zugehöriger RSS-Feed innerhalb eines Blogs): `<link rel="alternate" type="application/rss+xml" href="rss-feed.xml" title="RSS-Feed">` Weil es sich bei Feeds um XML-Dokumente handelt, wird im obigen Beispiel das `type`-Attribut mit dem MIME-Typ `application/rss+xml` verwendet. Die meisten Browser zeigen dann rechts außen in der URL-Adresszeile ein Feedsymbol ① an. Klicken Sie dieses an, können Sie den Feed anzeigen. 
`<link rel="icon">`	Damit ordnen Sie Ihrer Seite ein Favicon zu, das normalerweise im ICO-Format von Microsoft vorliegt: `<link rel="icon" type="image/x-icon" href="/favicon.ico" sizes="16x16 32x32">` Geben Sie im Attribut `type` den MIME-Typ an: `image/vnd.microsoft.icon`. Sie können aber auch - wie in diesem Beispiel - `image/x-icon` verwenden. Mit dem Attribut `sizes=` können Sie die Größe des Icons angeben. Das hier verwendete Icon ist in zwei Größen - 16 x 16 Pixel und 32 x 32 Pixel - darstellbar. Die beiden Größenangaben werden durch ein Leerzeichen getrennt.

Auf eine andere Adresse umleiten

Sie können von der aktuellen Seite automatisch auf eine andere Seite umleiten. Diese Folgeseite wird nach einer festgelegten Zeit geladen und angezeigt. Man spricht auch von Tunnel- oder Doorway-Seiten. Eine automatische Weiterleitung sollte nur mit gutem Grund angewendet werden, weil sie die Benutzerfreundlichkeit der Site verringert. Der Nutzer sollte kontrollieren können, was er im Browser darstellen und welche Seite er ansteuern möchte.

Achten Sie darauf, den Nutzer auf der Seite darauf hinzuweisen, in wie vielen Sekunden er weitergeleitet wird. Am besten bieten Sie zusätzlich einen direkten Hyperlink zum Weiterleitungsziel an. Für die Weiterleitung verwenden Sie das Attribut ref resh des meta-Tags im head-Bereich Ihrer Seite:

```
<meta http-equiv="Refresh" content="15;URL=../index-neu.html" />
```

Hier wird die Seite `index-neu.html` nach 15 Sekunden geladen.

Die meisten Suchmaschinen indizieren keine Seiten, deren ref resh-Wert kleiner als fünf Sekunden ist. In der Praxis wird diese Technik auch verwendet, um auf die aktuelle Seite weiterzuleiten, diese also zu aktualisieren. So stellen Sie sicher, dass häufig aktualisierte Informationen tatsächlich immer aktuell sind und nicht etwa aus dem Browser-Cache geholt werden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_02

Last update: **2019/05/14 21:18**

11.1.2.1) Fragen

- Was ist bei der Planung einer Navigation zu beachten?
- Wie werden Text- und grafische Navigationsleisten erstellt?
- Wie werden standardkonforme Navigationsmenüs erstellt?
- Wie werden Imagemaps erstellt?
- Wie werden logische Verlinkungen eingesetzt?
- Wie können Benutzer auf eine andere Adresse umleitet werden?

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_02:11_01_02_01

Last update: **2019/05/14 21:18**

11.1.2.2) Arbeitsaufträge

Arbeitsauftrag 1

Übungsdatei:	—
Ergebnisdatei:	uebung1.doc

Beantworten Sie schriftlich folgende Fragen:

1) Was versteht man unter Barrierefreiheit?

2) Wo sollte die Hauptnavigation eines Webauftritts angeordnet sein und warum?

3) Welche Nachteile bieten Imagemaps?

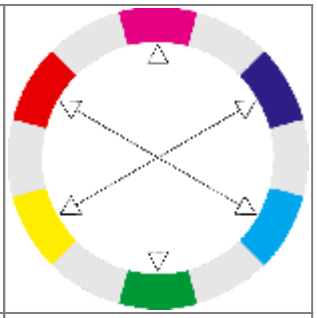
Arbeitsauftrag 2

Übungsdatei:	—
Ergebnisdatei:	uebung2.html

Erstellen Sie den folgenden grauen 3D-Button in reinem HTML und CSS. Beim Hovern soll sich die Farbe der Buttonfläche in ein helles Gelb ändern. Verwenden Sie die folgenden CSS-Eigenschaften: `text-decoration, padding, font-weight, border-width, border-style, border-color, color, background-color`.



Arbeitsauftrag 3

Übungsdatei:	
Ergebnisdatei:	uebung3.html

1) Fügen Sie das Bild farbkreis.gif in eine HTML-Seite ein.

2) Versehen Sie jede der sechs Farben mit einem polygonförmigen Hotspot.

3) Weisen Sie den Hotspots die Verweisziele `rothtml, magenta.html, blau.html, cyan.html, gruen.html` und `gelb.html` zu.

4) Statten Sie sie mit den entsprechenden Alternativtexten aus.

Arbeitsauftrag 4

Übungsdatei:	—
Ergebnisdatei:	uebung4.doc

Beantworten Sie schriftlich folgende Fragen:

1) Wie ordnen Sie Ihrer Seite ein Favicon zu?

2) Woran erkennt der Anwender, dass einer Seite mit `<link rel = „alternate“>` ein Feed zugeordnet wurde?

3) Welches HTML5-Tag nutzen Sie, um eine bestimmte Seite im Voraus zu laden, während die aktuelle Seite angezeigt wird?

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_02:11_01_02_02

Last update: **2019/05/14 21:18**

11.1.3) Bereiche definieren und positionieren

Zeitgemäßes Webseitenlayout

Nachteile von Tabellen als Layoutwerkzeug

Bis vor wenigen Jahren war es in der Welt des Webdesigns üblich, Seitenelemente mithilfe unsichtbarer Tabellen an bestimmten Stellen auf der Seite zu fixieren. In rein visueller Hinsicht sind Tabellen für das Seitenlayout auch recht gut geeignet, sie haben aber schwerwiegende Nachteile.

- Um ein stabiles Seitenlayout zu erzielen, mussten transparente GIF-Bilder verwendet werden, um die Breiten und Höhen der unsichtbaren Tabellen, und damit die Abstände der Seitenlayoutelemente, zu fixieren.
- Bilder müssen häufig im Bildbearbeitungsprogramm zerstückelt (in „Slices“ unterteilt) und in HTML wieder wie ein Puzzle zusammengesetzt werden.
- Für ältere Browser sind teilweise einander widersprechende Tabellenformatierungstags notwendig.
- Durch ständig wiederholte table- und td-Tags wird der Code unnötig aufgebläht.
- Tief greifende Umgestaltungen des Seitenlayouts sind schwierig und aufwendig.
- Struktur und Darstellung der Seite befinden sich komplett im HTML-Code.

Vorteile von div-Containern als Layoutwerkzeug

Aus diesem Grund hat das W3C die Empfehlung ausgesprochen, HTML-Tabellen nicht für Layoutzwecke zu verwenden, sondern ausschließlich für die Strukturierung von Tabellendaten. Für das Seitenlayout hingegen verwenden Sie am besten mit CSS formatierte div-Container und gegebenenfalls die neuen HTML5-Tags für Kopf- und Fußzeilen, Artikel usw. Daraus ergeben sich folgende Vorteile:

- vollständige Trennung von Struktur und Darstellung,
- bessere Kontrolle über das Seitenlayout,
- schnelle und einfache Aktualisierung oder Umgestaltung.

Diese Herangehensweise bedeutet auch, dass Sie verschiedene Stylesheets für verschiedene Zwecke spezifizieren können, ohne mehrere Versionen Ihrer Site verwalten zu müssen. Damit können Sie z. B. eine abgespeckte Version für PDAs, eine Hochkontrast- oder eine Nur-Text-Version für Sehbehinderte anbieten. Sogar eine spezielle Druckversion lässt sich ohne allzu großen Aufwand realisieren - Sie benötigen dazu nur ein Stylesheet mit geeigneten Farben, Seitenrändern usw.

Vorgehensweise beim Gestalten des Seitenlayouts mit div-Containern

Um ein Seitenlayout mit div-Containern aufzubauen, können Sie beispielsweise folgendermaßen vorgehen. Nach dieser Reihenfolge richtet sich auch der Aufbau dieses Kapitels.

Container erstellen	Zunächst fertigen Sie eine Skizze des gewünschten Seitenlayouts an. Dazu können Sie auch ein Blatt Papier verwenden. Erstellen Sie anschließend die benötigten `div`-Container. Für jeden Seitenbereich - etwa Hauptnavigation, Inhaltsbereich usw. - erstellen Sie einen eigenen `div`-Container.
Container formatieren	Weisen Sie den `div`-Containern per CSS die gewünschten Formatierungsmerkmale wie Hintergrundfarbe, Rahmen, Schriftmerkmale usw. zu. Dabei definieren Sie für jeden `div`-Container die jeweils benötigte Breite und eventuell eine Höhe.
Container positionieren	Positionieren Sie die einzelnen `div`-Container an der gewünschten Stelle der Seite.

div-Container erstellen

Bereiche definieren

Trennen Sie einzelne Gestaltungseinheiten bzw. Layout-Bereiche, die Sie an verschiedenen Stellen der Seite verteilen möchten, durch div-Elemente, auch div-Container genannt.

`div`	`div` leitet sich vom englischen "division" (Bereich) ab und bezeichnet einen gemeinsamen Bereich, in den Sie mehrere Inhalte aller Art einschließen können, beispielsweise Texte, Grafiken, Datentabellen usw. Wenn Sie einen `div`-Container nicht mit Cascading Style Sheets formatieren, beginnen alle in den Container eingeschlossenen Elemente in einer neuen Zeile.

Beispiel: Zwei div-Container ohne CSS-Formatierung

Quelltext

div-container.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Aktuelles</title>
</head>

<body>
<div>
  <ul>
    <li>Home</li>
    <li>Aktuelles</li>
    <li>Unser Angebot</li>
    <li>Kontakt</li>
    <li>Impressum</li>
```

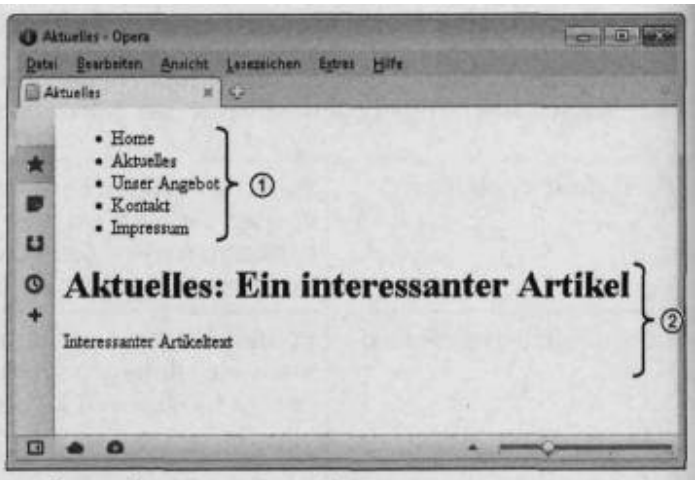
```
  </ul>
</div>
<div>
  <h1>Aktuelles: Ein interessanter Artikel</h1>
  Interessanter Artikeltext </div>
</body>
</html>
```

```
①  <div>
     <ul>
       <li>Home</li>
       <li>Aktuelles</li>
       <li>Unser Angebot</li>
       <li>Kontakt</li>
       <li>Impressum</li>
     </ul>
   </div>
②  <div>
     <h1>Aktuelles: Ein interessanter Artikel</h1>
     Interessanter Artikeltext
   </div>
```

Hier wurde ein `div`-Container für die Navigation ① und ein weiterer `div`-Container für den Seiteninhalt ② erstellt.

Da es keinerlei CSS-Formatierungen gibt, werden die beiden `div`-Container untereinander dargestellt, wie die nebenstehende Abbildung zeigt.

`div`-Bereiche können auch ineinander verschachtelt werden (vgl. Abschnitt 2.6).



Unformatierte `div`-Container

div-Container mit CSS formatieren

Die in Ihrem Dokument definierten div-Bereiche können Sie mit CSS gestalten.

Beispiel: Zwei div-Container mit CSS-Formatierung

Quelltext

div-container-formatiert.html

```
<!doctype html>
```

```
<html>
<head>
<meta charset="utf-8">
<title>Div formatiert</title>
<style type="text/css">
#Nav {
    font-family: Arial, Helvetica, sans-serif;
    font-size: small;
    background-color: #FFC;
    border: 2px dotted #600;
}
#Artikel {
    border: 3px double #600;
    padding: 3px;
}
</style>
</head>

<body>
<div id="Nav">
  <ul>
    <li>Home</li>
    <li>Aktuelles</li>
    <li>Unser Angebot</li>
    <li>Kontakt</li>
    <li>Impressum</li>
  </ul>
</div>
<div id="Artikel">
  <h1>Aktuelles: Ein interessanter Artikel</h1>
  <p>Interessanter Artikeltext</p>
</div>
</body>
</html>
```

- Öffnen Sie die Beispieldatei div-container.html.
- Definieren Sie innerhalb des Tagpaars <head> … </head> die folgenden CSS-Stildefinitionen, um die Seite mit einem eingebetteten Stylesheet zu versehen:

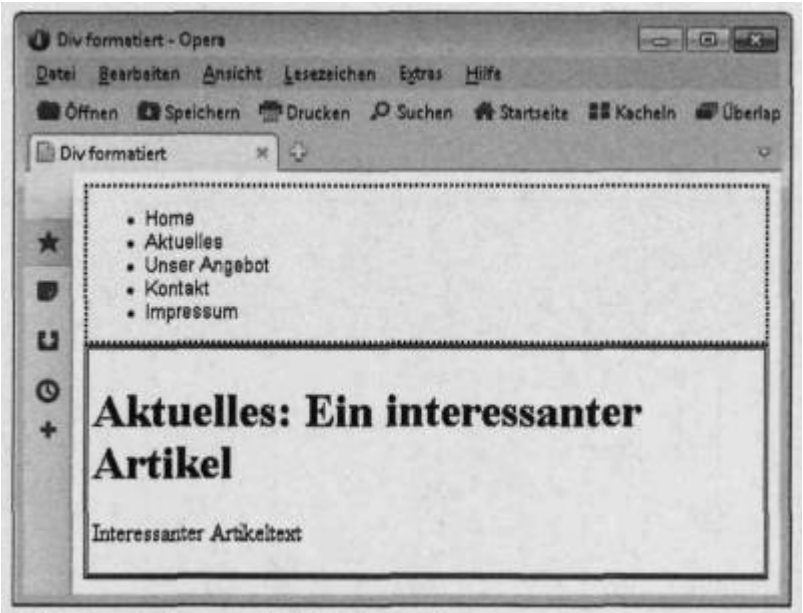
```
<style type=text/css>
①
#Nav {
     font-family:Arial, Helvetica, sans-serif;
     font-size:small;
     background-color:#FFC;
     border:2px dotted #600;
}
②
#Artikel {
     border:3px double #600;
     padding:3px;
}
</style>
```

(1) Erstellen Sie eine ID für die Navigation. Weisen Sie der ID die gewünschten CSS-Formatierungen zu.
(2) Erstellen Sie eine ID für den Seiteninhalt und weisen Sie der ID die gewünschten CSS-Formatierungen zu.

```
③ <div id="Nav">
    <ul>
      <li>Home</li>
      <li>Aktuelles</li>
      <li>Unser Angebot</li>
      <li>Kontakt</li>
      <li>Impressum</li>
    </ul>
  </div>
④ <div id="Artikel">
    <h1>Aktuelles: Ein interessanter Artikel</h1>
    <p>Interessanter Artikeltext</p>
  </div>
```

(3) Im body-Bereich des Dokuments weisen Sie dem ersten div-Container die ID Nav zu.
(4) Weisen Sie dem zweiten div-Container die ID Artikel zu.

Hier wurden für die Formatierung IDs verwendet, da die jeweiligen Elemente auf der Seite nur ein einziges Mal vorkommen. Wenn die Elemente auf der Seite mehrfach vorkommen, verwenden Sie stattdessen Klassen.



div-Container mit CSS-Formatierung

div-Containern eine Breite und eine Höhe zuweisen

Wenn Sie nichts anderes angeben, nehmen div-Container stets die gesamte Breite der Webseite oder des übergeordneten div-Containers ein. Dieses übergeordnete Element wird auch als Eltern-Element bezeichnet werden.

Damit Sie einen div-Container frei auf der Seite positionieren oder schweben lassen können (vgl. Abschnitt 2.4 und 2.5), müssen Sie per CSS auch seine Breite definieren.

width:Wert	Mit der CSS-Eigenschaft `width` können Sie die Breite des Elements festlegen. Für die Angabe des Werts haben Sie folgende Möglichkeiten: ✔ Zahlenangaben mit entsprechender Maßeinheit, ✔ prozentuale Angaben relativ zur Breite des Eltern-Elements, ✔ `auto` = automatische Festlegung.
height: Wert	Neben der Breite eines Elements können Sie auch dessen Höhe bestimmen. Die Wertangabe entspricht der für die Eigenschaft `width`.

div-Container schweben lassen

Der normale Elementfluss sieht vor, dass div-Container in der Reihenfolge, wie sie im Quelltext auftreten, aufeinanderfolgen. Das heißt, die Absätze werden untereinander dargestellt. Mit der CSS-Eigenschaft `float` können Sie div-Container jedoch von Text umfließen lassen.

float: Wert	Die folgenden Werte können Sie für `float` angeben: ✔ `left`: Element steht links und wird rechts umflossen. ✔ `right`: Element steht rechts und wird links umflossen. ✔ `none`: Element wird nicht umflossen. Die Eigenschaft `float` lässt sich nicht auf absolut positionierte Elemente (vgl. Abschnitt 2.6) anwenden.

Damit die Eigenschaft float eine sichtbare Auswirkung hat, müssen Sie für den div-Container eine Breite festlegen (z. B. 50%). Nur wenn neben dem Element Platz ist, können daneben weitere Inhalte stehen. Wie die Abbildung zeigt, befindet sich das umfließende Element (der durch das Tag p definierte Absatz) nicht neben dem Float, sondern dahinter. Lediglich die Inhalte des Absatzes werden verschoben: Der Float gibt den ihm zustehenden Platz komplett frei, die nachfolgenden Elemente rutschen entsprechend weit nach oben, die darin enthaltenen Texte und Bilder werden in den sichtbaren Bereich gedrückt.

Beispiel: Schwebender div-Container

Quelltext

div-container-schwebend.html

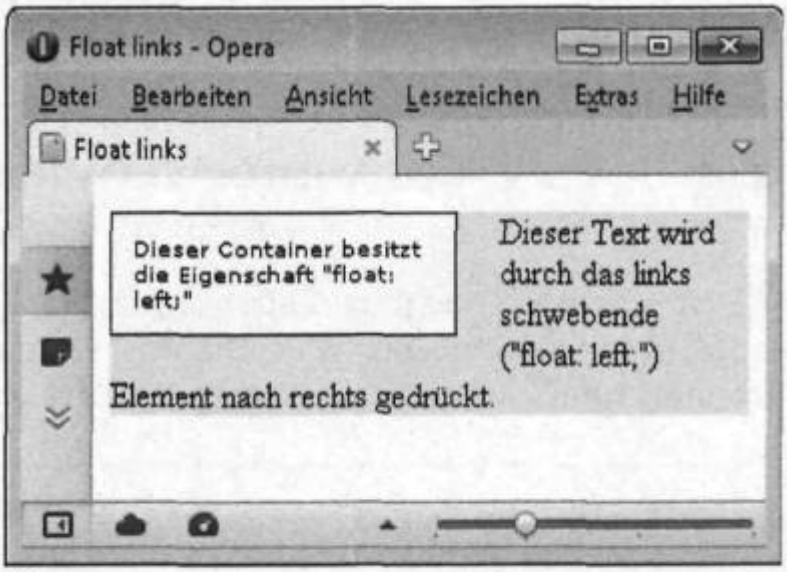
```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<style type=text/css>
<!--
#rahmen {
 width:15em;
 float:left;
```

```
 font-family:Verdana, Arial, Helvetica, sans-serif;
 font-size:x-small;
 background-color:#FFC;
 border:1px solid #036;
 margin-right:2em;
 padding-left:1em;
}
#absatz {
 background-color:#0FF;
}
-->
</style>
<title>Float links</title>
</head>
<body>
<div id="rahmen">
  <p>Dieser Container besitzt die Eigenschaft
&quot;float:left;&quot;</p>
</div>
<p id="absatz">Dieser Text wird durch das links schwebende
(&quot;float:left;&quot;) Element nach rechts gedrückt.</p>
</body>
</html>
```

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<style type=text/css>
<!--
① #rahmen {
        width:15em;
        float:left;
        font-family:Verdana, Arial, Helvetica, sans-serif;
        font-size:x-small;
        background-color:#FFC;
        border:1px solid #036;
        margin-right:2em;
        padding-left:1em;
  }
  #absatz {
        background-color:#0FF;
  }
  -->
  </style>
  <title>Float links</title>
  </head>
  <body>
② <div id="rahmen">
    <p>Dieser Container besitzt die Eigenschaft &quot;float:left;&quot;</p>
  </div>
③ <p id="absatz">Dieser Text wird durch das links schwebende
  (&quot;float:left;&quot;) Element nach rechts verschoben.</p>
  </body>
  </html>
```

Im Beispiel definieren Sie einen div-Container mit Textinhalt. Sie lassen ihn dann über dem Rest des Dokumentinhalts schweben.

(1) Erstellen Sie im head des Dokuments eine ID für Ihren div-Container. Weisen Sie der ID die gewünschten CSS-Formatierungen inklusive der CSS-Eigenschaft `float:left` zu. (2) Erstellen Sie im body des Dokuments einen div-Container, dem Sie die zuvor definierte ID zuweisen. (3) Geben Sie nach dem div-Container weitere Inhalte ein.



Über dem Seiteninhalt schwebender `div`-Container

div-Container positionieren

Für die Positionierung gibt es seit der Version CSS2 (Cascading Style Sheets Level 2) über die CSS-Eigenschaft Position vier verschiedene Möglichkeiten, ein Element zu positionieren. Sie können damit div-Elemente (und andere) unabhängig von ihrer Position im Quelltext an einer beliebigen Stelle des Browserfensters anzeigen.

`position: Wert`	Folgende Werte stehen zur Verfügung: ✔ `static`: keine spezielle Formatierung; Element positioniert sich im Textfluss ✔ `relative`: Abweichung von der Position, die das Element im Textfluss eingenommen hätte ✔ `absolute`: Positionierung des Elements mit Bezug auf das Element `body` oder den letzten Vorfahren mit einer relativen Positionierung ✔ `fixed`: wie `absolute`, aber beim Scrollen der Webseite bleibt das Element fixiert

Wo der div-Container genau auf der Seite platziert werden soll, geben Sie über die vier Startpositionen an. Sie können hier Zahlenwerte mit der entsprechenden Maßeinheit oder prozentuale Angaben zur Höhe und Breite des Eltern-Elements verwenden. Mit auto stellen Sie eine automatische Positionierung ein.

`left: Wert`	Abstand vom linken Rand des Containers
`top: Wert`	Abstand vom oberen Rand des Containers
`right: Wert`	Abstand vom rechten Rand des Containers
`bottom: Wert`	Abstand vom unteren Rand des Containers

Alle Dezimalstellen der Positionsangaben müssen mit einem Dezimalpunkt „.“ abgetrennt werden, beispielsweise `top:3.2cm` und `left:6.2px`. Haben Sie ein Element mit `position:static` oder keiner Positionsangabe zwischen die anderen Elemente fließen lassen, sind die Angaben der Eigenschaften `left, top, right` und `bottom` unwirksam.

Absolute Positionierung

Die absolute Positionierung bezieht sich auf ein Koordinatensystem, das über das HTML-Dokument gelegt wird. Der Nullpunkt liegt in der linken oberen Ecke. Von hier ab wird nach links (x-Achse) und nach unten (y-Achse) gemessen.

Beispiel: Absolut positionierter div-Container

Quelltext

div-container-absolut.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<style type="text/css">
<!--
#rahmen {
 position:absolute;
 width:250px;
 background-color:#FFC;
 border:1px solid #036;
 height:auto;
 left:10px;
 padding:5px;
 top:10px;
}
-->
</style>
<title>Absolute Positionierung</title>
</head>
<body>
<p>Dieser Text hat mit dem positionierten Rahmen nichts zu tun. Er wird
einfach hinter diesem dargestellt. </p>
<div id="rahmen">
  <p>Dieser Rahmen ist absolut positioniert. Er liegt über allen
anderen
    Seitenelementen.</p>
</div>
</body>
```

Der folgende Code zeigt ein absolut positioniertes Element. Dieses stellt innerhalb der Seite eine absolut unabhängige Einheit dar, die in der Grundeinstellung einfach über allen anderen Seitenelementen platziert wird. Das Element wird also vollständig aus dem Fluss des Dokuments herausgenommen. Die übrigen Seitenelemente verhalten sich, als sei das Element nicht vorhanden.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<style type="text/css">
<!--
#rahmen {
     position:absolute;
     width:250px;
     background-color:#FFC;
     border:1px solid #036;
     height:auto;
     left:10px;
     padding:5px;
     top:10px;
}
-->
</style>
<title>Absolute Positionierung</title>
</head>
<body>
<p>Dieser Text hat mit dem positionierten Rahmen nichts zu tun. Er wird einfach
hinter diesem dargestellt. </p>
<div id="rahmen">
  <p>Dieser Rahmen ist absolut positioniert. Er liegt über allen anderen
    Seitenelementen.</p>
</div>
</body>
```

(1) Erzeugen Sie eine CSS-Stildefinition für den zu positionierenden div-Container. Nehmen Sie in diese die absolute Positionierung mit auf (position:absolute).
(2) Geben Sie zur Verdeutlichung einen beliebigen <p>-Absatz ein.
(3) Erzeugen Sie einen div-Container, den Sie mit der zuvor erstellten ID formatieren.

Relative Positionierung

Die absolute Positionierung ist nur dann sinnvoll, wenn Sie akzeptieren, dass der Bezugspunkt des Elements der Seitenrand bzw. der letzte Vorfahre mit einer relativen Positionierung ist. Für den Aufbau eines differenzierten Layouts ist die absolute Positionierung deshalb nur bedingt geeignet, weil dabei Elemente nicht nur an der Seite, sondern auch aneinander ausgerichtet werden.

Hier bietet sich eine relative Positionierung an. Das relativ positionierte Element wird nicht am HTML-Dokument selbst, sondern an seiner Position im HTML-Dokument ausgerichtet. Die Position wird ab der tatsächlichen Platzierung des div-Containers gemessen.

Beispiel: Relativ positionierter div-Container

Quelltext

div-container-relativ.html

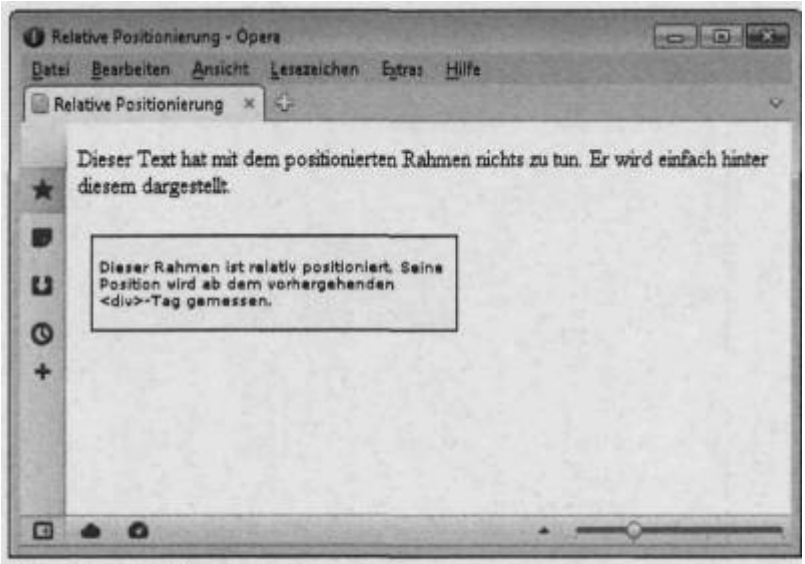
```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<style type="text/css">
<!--
#rahmen {
 position:relative;
 width:250px;
 background-color:#FFC;
 border:1px solid #036;
 height:auto;
 left:10px;
 padding:5px;
 top:10px;
}
-->
</style>
<title>Relative Positionierung</title>
</head>
<body>
<p>Dieser Text hat mit dem positionierten Rahmen nichts zu tun. Er wird
einfach hinter diesem dargestellt. </p>
<div id="rahmen">
  <p>Dieser Rahmen ist absolut positioniert. Er liegt über allen
anderen
    Seitenelementen.</p>
</div>
</body>
```

Anhand eines praktischen Beispiels lässt sich dies schnell erfassen.

- Öffnen Sie die Beispieldatei div-container-absolut.html.
- Ändern Sie die Position in der Stildefinition für den Rahmen im oben stehenden Beispiel ab.

```
#rahmen  {
   position:relative;
   /*   ...   */
}
```

Nun befindet sich der Ausgangspunkt für die Berechnung der Position des Rahmens nicht mehr in der linken oberen Dokumentecke, sondern unter dem vorhergehenden p-Absatz, weil der div-Container unter diesem Absatz eingefügt wurde.



Relative Positionierung

Seitenlayouts mit positionierten div-Containern aufbauen

Mit den vorgestellten Methoden zur Positionierung können Sie Ihre Seitenlayouts aufbauen. Nachfolgend sehen Sie drei Beispiele.

Beispiel: Seitenlayout mit absoluter und relativer Positionierung

Quelltext

div-layout.html

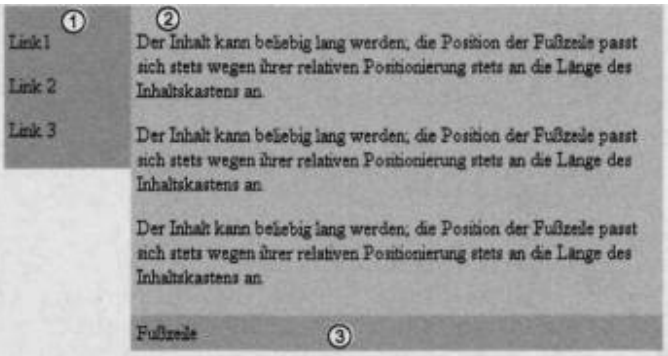
```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>DIV-Layout</title></head>
<style type="text/css">
<!--
body { margin: 0 }
#navigation {
 position: absolute;
 padding: 5px;
 border: 1px solid #999999;
 height: auto;
 width: 20%;
 left: 5px;
 top: 9px;
 bottom: auto;
 background-color: #33CCFF;
}
```

```
#inhalt {
 position: relative;
 padding: 5px;
 border: 1px solid #999999;
 height: auto;
 width: 78%;
 left: 20%;
 top: 9px;
 background-color: #FFCC99;
}
#fusszeile {
 position: relative;
 padding: 5px;
 border: 1px solid #999999;
 height: 20px;
 width: 78%;
 left: 20%;
 top: 9px;
 background-color: #CCCCCC;
}
-->
</style>
</head>

<body>
<div id="navigation">
  <p>Link 1 </p>
  <p>Link 2</p>
  <p>Link 3</p>
</div>
<div id="inhalt">
  <p>Der Inhalt kann beliebig lang werden; die Position der Fußzeile
passt sich stets wegen ihrer relativen Positionierung stets an die
Länge des Inhaltskastens an.</p>
  <p>Der Inhalt kann beliebig lang werden; die Position der Fußzeile
passt sich stets wegen ihrer relativen Positionierung stets an die
Länge des Inhaltskastens an.</p>
  <p>Der Inhalt kann beliebig lang werden; die Position der Fußzeile
passt sich stets wegen ihrer relativen Positionierung stets an die
Länge des Inhaltskastens an.</p>
</div>
<div id="fusszeile"> Fußzeile </div>
</body>
</html>
```

Mithilfe von `div`-Containern soll das nebenstehend abgebildete Seitenlayout entstehen.

① Links soll eine senkrecht angeordnete Navigationsleiste zu sehen sein.
② Rechts daneben soll sich der Inhaltsteil der Seite befinden,
③ darunter eine Fußzeile.



Geplantes Seitenlayout

- Die Navigationsleiste kann absolut positioniert werden, weil sie sich immer an derselben Stelle des Dokuments befindet.
- Der Inhaltsteil hingegen muss eine relative Positionierung erhalten. Seine Länge kann unterschiedlich sein, und die Position der Fußzeile hängt von dieser Länge ab.
- Die Fußleiste muss ebenfalls relativ positioniert sein, weil sie unterhalb des Inhaltsteils mit seiner variablen Länge liegen soll.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>DIV-Layout</title></head>
<style type="text/css">
<!--
<!-- body {
① margin:0;
}
② #navigation {
      bottom:auto;
      height:auto;
      left:5px;
      padding:5px;
      position:absolute;
      top:9px;
      width:20%;
}

③ #inhalt {
      height:auto;
      left:20%;
      padding:5px;
      position:relative;
      top:9px;
      width:78%;
}

#fusszeile {
      height:20px;
      left:20%;
      padding:5px;
      position:relative;
      top:9px;
      width:78%;
}
-->
</style>
<title>Seitenlayout CSS</title>
</head>
```

```
<body>
<div id="navigation">
  <p>Link 1</p>
  <p>Link 2</p>
  <p>Link 3</p>
</div>
<div id="inhalt">
  <p>Der Inhalt kann beliebig lang werden; die Position der Fußzeile passt sich
stets wegen ihrer relativen Positionierung stets an die Länge des
Inhaltskastens an.</p>
  <p>Der Inhalt kann beliebig lang werden; die Position der Fußzeile passt sich
stets wegen ihrer relativen Positionierung stets an die Länge des
Inhaltskastens an.</p>
  <p>Der Inhalt kann beliebig lang werden; die Position der Fußzeile passt sich
stets wegen ihrer relativen Positionierung stets an die Länge des
Inhaltskastens an.</p>
</div>
<div id="fusszeile"> Fußzeile </div>
</body>
</html>
```

④

Im abgebildeten Code fehlen die Formatierungseinstellungen für Rahmenfarbe und -stärke sowie Hintergrundfarben. Die Beispieldatei div-layout.html enthält den vollständigen Code.

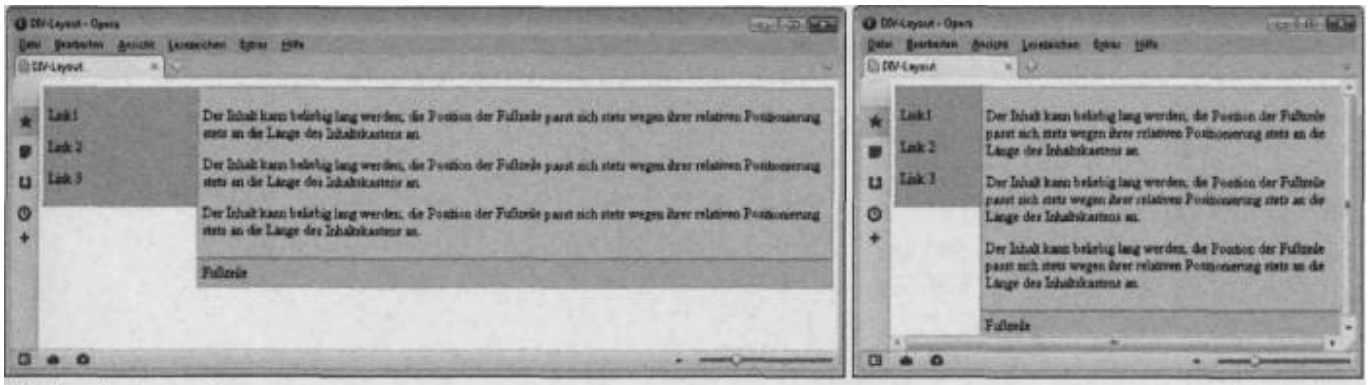
(1) Damit die Container für Inhalt und Navigation auf der y-Achse an derselben Stelle beginnen, versehen Sie das body-Tag mit der Eigenschaft `margin:0`.

Über die Eigenschaft margin wird der Rand des Browsers definiert. Dieser wird von der relativen Positionierung berücksichtigt, von der absoluten hingegen nicht. Da Sie im Folgenden sowohl relative als auch absolute Positionsangaben verwenden, ist diese Definition erforderlich.

(2) Definieren Sie die Eigenschaften der Navigation. Verwenden Sie dabei eine absolute Positionierung, so-dass die Navigation komplett aus der Abhängigkeit der Elemente von der Seitenposition herausgenommen wird.
(3) Definieren Sie die Eigenschaften für Inhalt und Fußzeile. Verwenden Sie für beide Elemente eine relative Positionierung.
(4) Erzeugen Sie jetzt die drei div-Container im Dokument.

Notieren Sie zuerst den absolut positionierten Container für die Navigation und erst danach die beiden relativ positionierten Container für Inhalt und Fußzeile, damit Sie eine korrekte Darstellung erhalten.

Wie die folgenden Abbildungen zeigen, handelt es sich um ein fließendes Layout, das sich einem breiteren oder schmaleren Browserfenster anpasst.



Fließendes Layout

Beispiel: Textspalten mit verschachtelten div-Containern

Quelltext

textbox.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Spaltensatz</title>
<style type="text/css">
<!--
#cont {
 background-color:#FFFFCC;
 border:3px double #006699;
 height:300px;
 left:0px;
 position:relative;
 padding:5%;
 top:0px;
 width:400px;
}
#Textspalte1, #Textspalte2 {
 background-color: #FFFFFF;
 border: 1px solid #333333;
 height: 85%;
 position: absolute;
 padding: 2%;
 top: 5%;
 width: 38%;
}
#Textspalte1 {
 left: 5%;
}
#Textspalte2 {
 right: 5%;
}
-->
</style>
</head>
<body>
<div id="cont"> Kopfzeile
  <div id="Textspalte1">Dies ist die linke Textspalte. </div>
  <div id="Textspalte2">Dies ist die rechte Textspalte.</div>
</div>
</body>
</html>
```

Wie erwähnt, lassen sich div-Container auch ineinander verschachteln. Sie können diese Technik verwenden, wenn Sie bestimmte Elemente wie etwa Textspalten oder Bilder samt Beschriftung zusammenhalten möchten. So fallen nachträgliche Layoutänderungen leichter.

Der folgende Code erzeugt eine zweispaltige Textbox:

```
<style type="text/css">
<!--
#cont {
     height:200px;
     left:50px;
②    position:relative;
     padding:5%;
     top:50px;
     width:200px;
}
#Textspalte1, #Textspalte2 {
     height:85%;
④    position:absolute;
     padding:2%;
     top:5%;
     width:38%;
}
#Textspalte1 {
     left:5%;
}
#Textspalte2 {
     right:5%;
}
-->
</style>
</head>
<body>
① <div id="cont"> Kopfzeile
③    <div id="Textspalte1">Dies ist die linke Textspalte. </div>
     <div id="Textspalte2">Dies ist die rechte Textspalte.</div>
</div>
</body>
</html>
```

Im abgebildeten Code fehlen die Formatierungseinstellungen für Farben, Rahmeneinstellungen usw. Die Beispieldatei textbox.html enthält den vollständigen Code. Erzeugen Sie zunächst einen div-Container (1) mit relativer Positionierung (2).

Platzieren Sie innerhalb dieses Containers einen oder mehrere weitere div-Container (3) mit absoluter Positionierung (4).

Die Positionierung der eingefügten Elemente wird nicht mehr vom Koordinatensystem der Seite, sondern ab der Position des übergeordneten Elements gemessen.

Die beiden inneren Textcontainer sind ausschließlich mit Prozentangaben formatiert. Dadurch lässt sich nicht nur die Position, sondern auch die Größe des äußeren Containers beliebig ändern - die inneren Container werden entsprechend angepasst:



Relative Positionierung ab der tatsächlichen Platzierung des `div`-Containers

```
#cont {
      background-color:#FFFFCC;
      border:3px double #006699;
      height:300px;
      left:0px;
      position:relative;
      padding:5%;
      top:0px;
      width:400px;
}
```



Neuformatierung des äußeren Containers mit automatischer Anpassung der inneren Container

Beispiel: Ein Layout im Browserfenster zentrieren

Quelltext

zentriert.html

```
... do be done.
```

Mit der absoluten Positionierung lässt sich ein div-Container zentriert auf die Seite setzen. Unabhängig davon, wie klein oder groß das Browserfenster ist, wird die zentrierte Ausrichtung immer beibehalten.

Innerhalb dieses div-Containers können Sie dann beliebige Elemente, auch das gesamte Seitenlayout, unterbringen. Das Ergebnis ist ein vollständig zentriertes Seitenlayout.

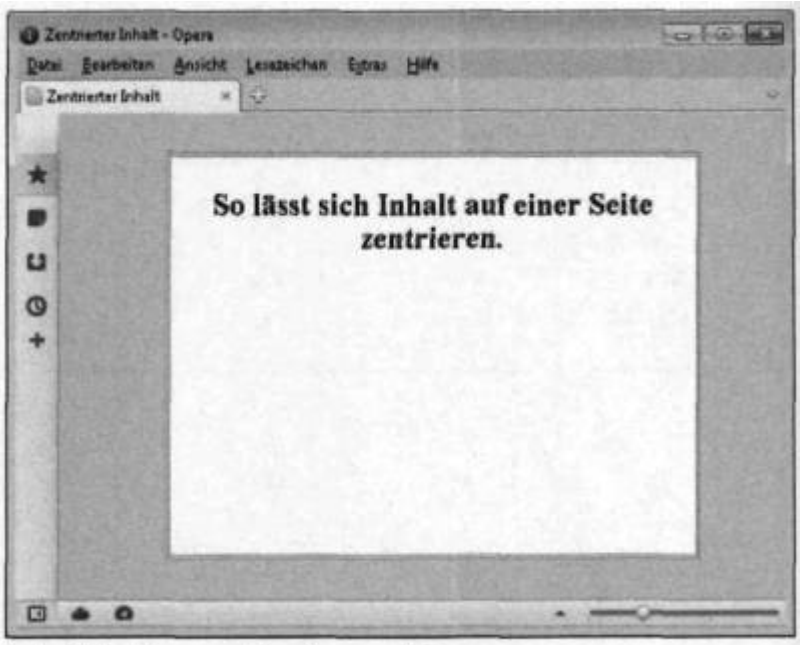
```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<style type="text/css">
<!--
#zentrierterInhalt {
     height:300px;
①    left:50%;
②    margin-left:-200px;
③    margin-top:-150px;
     padding:5px;
     position:absolute;
     top:50%;
     width:400px;
}

body {
     margin:50px;
}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Zentrierter Inhalt</title>
</head>

<body>
<div id="zentrierterInhalt">
  <h2 align="center">So lässt sich Inhalt auf einer Seite zentrieren. </h2>
</div>
</body>
</html>
```

Dazu müssen Sie lediglich die linke obere Ecke des div-Bereichs im Mittelpunkt der Bildschirmdarstellung anzeigen (1) und dann um die Hälfte der Breite des div-Bereichs nach links (2) und um die Hälfte der Höhe des Bereichs nach oben (3) setzen.

Im abgebildeten Code fehlen die Formatierungseinstellungen für Farben, Rahmeneinstellungen usw. Die Beispieldatei `zentriert.html` enthält den vollständigen Code.



Inhalt im Browserfenster zentrieren

Bereiche mit semantischen Tags definieren

HTML5 sieht die Verwendung des div-Elements nur dann vor, wenn es kein anderes Element gibt, mit dem Sie den gewünschten Bereich exakter auszeichnen können. Es gibt semantische HTMLS-Elemente für Kopf-und Fußzeilen, Artikel, Seitenleisten und Abschnitte. Diese Tags funktionieren grundsätzlich wie div-Container, sorgen jedoch - richtig eingesetzt - für ein semantisch korrektes Dokument. Zum Zeitpunkt der Erstellung des Buchs wurden die im Folgenden erläuterten Elemente nicht von allen Browsern unterstützt. Trotzdem ist es wichtig, dass Sie sich bereits mit den neuen HTML5-Tags vertraut machen.

`header`	Dient zur Definition eines Kopfbereichs, wie etwa das Firmenlogo, die waagerechte Hauptnavigation, der Seitentitel etc. Setzen Sie den Kopfbereich zwischen das Tagpaar `<header> … </header>`. Anders als bei `div`-Tags können Sie in das `header`-Tag keine weiteren `header` verschachteln (dasselbe gilt auch für das nachfolgend aufgeführte `footer`-Element).
`footer`	Dient zur Definition eines Fußbereichs, beispielsweise für die Zusatznavigation usw. Setzen Sie den Fußbereich zwischen das Tagpaar `<footer> … </footer>`.
`article`	Dient zur Definition eines Artikels, beispielsweise in einem Magazin oder Blog. Innerhalb des Tagpaars `<article> … </article>` können Sie sämtliche verfügbaren Block- und Inline-Elemente verschachteln.
`section`	Dient zur Definition eines Abschnitts, besonders in umfangreichen und stark strukturierten HTML-Elementen. Innerhalb des Tagpaars `<section> … </section>` können sämtliche verfügbaren Block- und Inline-Elemente verschachtelt werden, beispielsweise auch `article`-Elemente.
`nav`	Dient zur Definition eines Navigationsbereichs. Innerhalb des Tagpaars `<nav> … </nav>` können sämtliche verfügbaren Block- und Inline-Elemente verschachtelt werden.
`aside`	Dient zur Definition einer Seitenleiste. Innerhalb des Tagpaars `<aside> … </aside>` können sämtliche verfügbaren Block- und Inline-Elemente verschachtelt werden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_03

Last update: **2019/05/14 21:20**

11.1.3.2) Bereiche definieren und positionieren

Arbeitsauftrag 1: Fragen zu div-Containern

Übungsdatei:	—
Ergebnisdatei:	uebung1.doc

1. Nennen Sie die Vorteile der Layoutgestaltung mit div-Containern gegenüber der Layoutgestaltung mit Tabellen.
2. Nennen Sie die Unterschiede zwischen einer statischen, relativen, absoluten und fixierten Positionierung.
3. Welches HTML5-Tag können Sie verwenden, um eine Fußzeile zu definieren? Welches Tag können Sie alternativ verwenden, um eine möglichst breite Browserunterstützung zu erzielen?

Arbeitsauftrag 1: Bereiche positionieren

Übungsdatei:	—
Ergebnisdateien:	uebung2.html

1. Stylesheets ermöglichen ein fast perfektes Zeitungslayout. Versuchen Sie, die nachfolgende Abbildung mithilfe der gelernten Möglichkeiten der Stylesheets nachzubilden. Es geht nur um die Platzierung der Textcontainer; die Textgestaltung können Sie nach Belieben vornehmen.



In drei Spalten angeordneter Text, dargestellt in Opera

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_03:11_01_03_02

Last update: **2019/05/14 21:21**

11.1.4) Sound und Videos einbinden

Multimedia im Web

Im Web werden in immer stärkerem Maß multimediale Technologien wie Videos oder Musik eingesetzt, um die Attraktivität der Seiten zu erhöhen.

Vor HTML5: Plug-ins

Bis HTML4 ist Hypertext nicht multimediafähig. Das bedeutet, dass der Internetbrowser Unterstützung benötigt, um Sound- oder Videodateien anzeigen zu können. Dies kann mit Plug-ins realisiert werden. Plug-ins sind an den Browser angekoppelte oder von ihm abhängige Zusatzdateien mit den benötigten Programmfunktionen. Sie ermöglichen dem Browser, fremde Dateiformate auf der Webseite anzuzeigen, auch wenn der Browser diese Dateiformate nicht direkt unterstützt.

Voraussetzung hierfür ist, dass der Anwender das entsprechende Plug-in auf seinem Computer installiert hat.

Die wichtigsten Plug-ins sind das Animationsformat Adobe Flash, der Dokumentanzeiger Acrobat Reader für die Anzeige von PDF-Dateien, die Audio-Plug-ins für die Formate MP3, MIDI, WAVE, Real Audio sowie die Plug-ins zum Anschauen von Videos in Formaten wie etwa AVI, MPEG und QuickTime.

Viele Plug-ins, zum Beispiel für Sound, digitales Video sowie der Flash-Player, werden bereits mit den Browsern ausgeliefert, andere muss der Benutzer nachträglich von der Website des Herstellers herunterladen und installieren. Sie sind oft kostenlos.

Plug-ins einbinden

Plug-ins werden mit dem obj ect-Element in die Webseite eingebunden. Sobald Sie eine Webseite mit einer `object`-Anweisung aufrufen, ermittelt der Browser im Hintergrund zunächst, welches Plug-in für die Mediendateien installiert ist, und lädt dieses anschließend:

Sie binden Multimedia-Objekte auf Ihrer Webseite über das HTML-Element object ein.

Das Attribut `data` des `object`-Elements enthält die Referenz zu der entsprechenden Objektdatei und muss in Anführungszeichen angegeben werden. Dabei spielt es keine Rolle, welches Format das Objekt hat. Dieses legen Sie über das Attribut type und das entsprechende MIME-Format fest. Die Attribute width und height bestimmen die Größe der anzuzeigenden Datei im Browser. Diese Attribute müssen Sie angeben, da ansonsten das Objekt nicht angezeigt werden kann. Mit der Angabe standby können Sie einen alternativen Text angeben, der angezeigt wird, bis das Objekt vollständig geladen ist.

Für einige eingebundene Elemente werden über das optionale Tag `param` ein oder mehrere Parameter übergeben. Dabei werden der Name (`name`) und der Wert (`value`) des gewünschten

Parameters angegeben:

```
<object data="clock.avi" type="video/x-msvideo" width="100" height="100"
        standby="Lade AVI"></object>
<object data="universal.html" type="text/html" width="400" height="200"></object>
```

In der ersten Zeile binden Sie ein Video in die Webseite ein, das in einer Größe von 100 x 100 Pixeln dargestellt wird. Auch Webseiten lassen sich damit einbinden, wie die zweite Zeile zeigt. Die genaue Größe legen Sie über die beiden Attribute width und height fest.

HTML5: Sound und Videos direkt einbinden

In HTML5 können Sie multimediale Elemente wie Audio- und Videodateien einbinden, ohne dazu wie bisher Plug-ins wie QuickTime oder Flash zu benötigen.

Sie verwenden dazu die neuen HTML5-Elemente <audio> und <video>. Dies hat gegenüber der zuvor erläuterten Lösung mit der Verwendung des object-Elements eine Reihe von Vorteilen:

Vereinfachte Einbindung	Die meisten Plug-ins sind kostenlos, ihre korrekte Einbindung bedeutet jedoch einen gewissen Entwicklungsaufwand. Mit den HTML5-Elementen audio und video ist dieser Vorgang standardisiert und deutlich vereinfacht worden und es sind keine Plug-ins notwendig.
Keine Probleme mit Nutzungsrechten und Werbung	Zwar ist es kein Problem, ein Video auf YouTube oder Vimeo zu laden und den Code in die eigene Website zu integrieren, jedoch gehen die Nutzungsrechte dann normalerweise an den Plattformbetreiber und es wird Werbung in die Videos eingeblendet. Durch die Elemente audio und video ist auch dieses Problem behoben.

Beachten Sie, dass die <audio>- und <video>-Elemente nur von den folgenden Browsern unterstützt werden:

Internet Explorer ab Version 9	Opera ab Version 10.5
Firefox ab Version 3.5	iPhone ab Version 1
Safari ab Version 3	Android ab Version 2
Google Chrome ab Version 3	

Damit Nutzer mit älteren Browsern Ihre Multimedia-Dateien abspielen können, ist eine Lösung (Fallback) notwendig, die bei der ausführlichen Erläuterung der Elemente audio und video behandelt wird (vgl. Abschnitt „Sound in HTML einbinden“).

Ein zusätzliches Problem ist, dass nicht alle Multimediaformate von jedem Browser ohne zusätzliche Software abgespielt werden können. So ist beispielsweise das weitverbreitete Audioformat MP3 patentgeschützt, sodass es nicht in jeden Browser implementiert werden kann (vgl. folgender Abschnitt).

Multimediaformate kennenlernen

Codecs

Digitale Multimedia-Dateien sind normalerweise mehrere Megabyte groß. Deshalb könnten sie ohne Kompression (auch Komprimierung genannt) nicht über das Internet gestreamt werden. Dazu werden verschiedene sogenannte Codecs verwendet.

Codec ist ein Kunstwort aus Coder/Decoder. Es handelt sich dabei um ein Verfahren zur Komprimierung während der Aufzeichnung und der Dekomprimierung während des Abspielens von Mediendateien. Durch die Kompression wird die Audio-A/ideodatei kleiner. Dabei werden redundante Informationen sowohl zwischen den einzelnen Frames als auch in den Frames selbst entfernt.

Dadurch kommt es stets zu einem mehr oder minder ausgeprägten Daten- und damit Qualitätsverlust. Die Komprimierung ist stets ein Kompromiss zwischen möglichst geringer Dateigröße und möglichst hoher Qualität der Mediendatei.

Außerdem sind viele Codecs durch Patente geschützt, die zwar nicht den Endanwender betreffen, wohl aber Softwarehersteller, die diese Codecs nutzen möchten. Patentgeschützte Codecs bieten momentan eine deutlich höhere Kompressionsrate als patentfreie Codecs. Die Browser Mozilla und Opera setzen auf patentfreie Codecs mit geringerer Kompressionsrate. Die Browser von Microsoft, Google und Apple verwenden patentgeschützte Codecs mit besserer Kompressionsrate.

Deshalb sollten Sie Multimedia-Dateien in HTML5-Webseiten in mehreren Formaten anbieten. Im weiteren Verlauf dieses Kapitels erfahren Sie, wie das geht. Die derzeit wichtigsten Audioformate/Codecs für das Web sind:

.mp3 MPEG-1 Audio Layer 3 (MP3)	MP3 - kurz für MPEG-1 Audio Layer 3 - komprimiert Audiodateien verlustbehaftet. Die Kompression erfolgt dabei so, dass die Wahrnehmung je nach Kompressionsstufe nicht oder nur gering beeinträchtigt wird. Die Kompressionsrate ist hoch, die resultierenden Dateien sind klein, sodass MP3 sich schnell zum wichtigsten Audioformat für das Web entwickelt hat.
.ogg OGG/Vorbis	Das Container-Format OGG eignet sich für Multimedia-Inhalte aller Art - Video, Audio und Text. Es können unterschiedliche Audio- und Video-Codecs enthalten sein. Üblicherweise wird für OGG-Audio der patentfreie Vorbis-Codec verwendet. Ähnlich wie MP3 komprimiert auch Vorbis verlustbehaftet. Der Nachteil gegenüber MP3 ist, dass die resultierenden Dateien größer sind.

Die folgende Tabelle zeigt Ihnen die derzeit wichtigsten Videoformate/Codecs für das Web:

.mp4 MPEG-4/H.264	Der ISO-Standard MPEG-4 ist aus dem QuickTime-Format von Apple hervorgegangen. Es kann unter anderem mehrere Video- und Audiospuren speichern. Für das MPEG-4-Format wird der durch Patente geschützte Video-Codec H.264 verwendet.
.ogv OGV/Theora	Der patentfreie Video-Codec Theora ist der übliche Codec für OGG-basierte Videos. Die Kompressionsrate ist weniger hoch als die des H.264-Codecs.

Multimedia-Dateien encodieren

Bevor Sie Ihre eigenen Multimedia-Dateien in HTML5 einbinden, müssen Sie sie also encodieren. Für

.mp3 und .mp4 können Sie dazu beispielsweise den Adobe Media Encoder o.Ä. verwenden. Für das OGG-Format gibt es verschiedene frei verfügbare Encoder, die Sie sich aus dem Internet herunterladen können, beispielsweise Miro Video Converter.

Sound in HTML5 einbinden

Zum Einbinden von Audio in HTML5 verwenden Sie das <audio>-Element.

audio	Innerhalb des audio-Elements setzen Sie einen oder mehrere Verweise auf die Sounddatei oder mehrere alternative, in unterschiedlichen Dateiformaten gespeicherte Versionen der Sounddatei.

Folgende Attribute stehen für das audio-Element zur Verfügung:

src	Die URL der abzuspielenden Audiodatei
preload	Der Browser beginnt direkt beim Laden mit dem Download der Audiodatei.
autoplay	Der Inhalt wird bereits während der Übertragung aus dem Internet im Browser des Endnutzers abgespielt.
loop	Wird dieses Attribut angegeben, wird die Audiodatei in einer Endlosschleife abgespielt.
controls	Die Steuerelemente (Abspielen/Pausieren/Position/Lautstärke) werden auf der Webseite angezeigt.

Beispiel: Audio in Webseite einbetten

Quelltext

html5-audio1.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Kalimba</title>
</head>

<body>
<h1>Kalimba </h1>
<div>
<audio preload controls src="Kalimba.ogg">
   <source src="kalimba.mp3">
</audio>
</div>
</body>
```

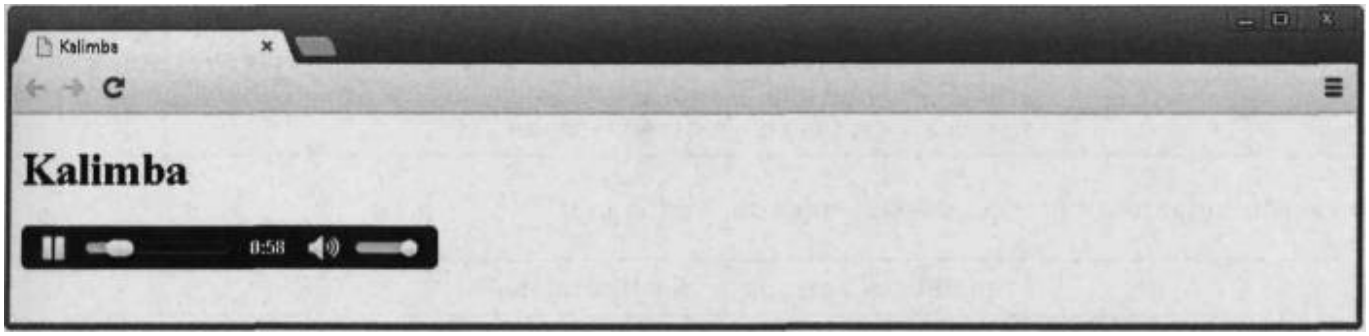
kalimba.mp3

```
<body>
<h1>Kalimba </h1>
<div>
① <audio preload controls src="Kalimba.mp3">
② <source src="Kalimba.ogg">
</audio>
</div>
</body>
```

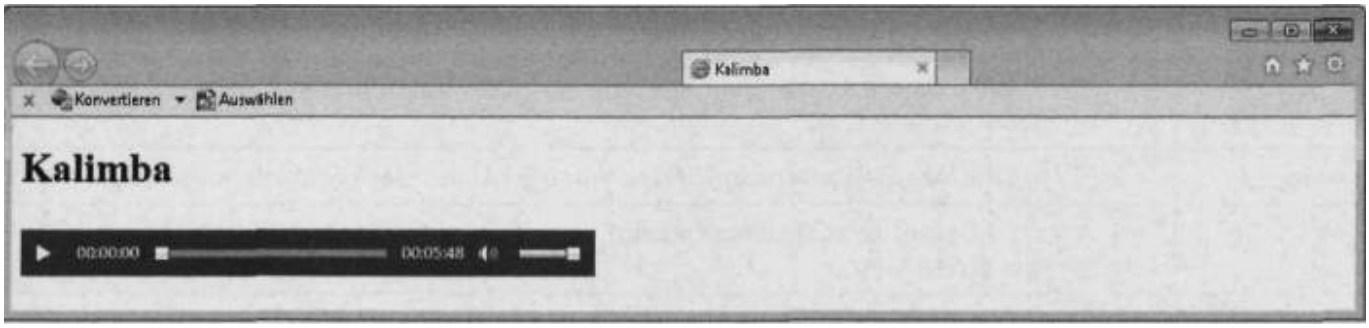
(1) Mit dem Tagpaar `<audio> … </audio>` binden Sie eine Audio-Ressource in die Webseite ein. Mithilfe des Attributs `src` dieses Elements geben Sie die URL der abzuspielenden Audiodatei an. Das Attribut `preload` sorgt dafür, dass der Browser direkt beim Laden mit dem Download der Audiodatei beginnt. Das Attribut `controls` sorgt dafür, dass ein Audio-Player auf der Webseite angezeigt wird. Mit diesem kann die Audiodatei abgespielt, pausiert und angehalten sowie ihre Lautstärke verändert werden.

(2) Mit dem source-Element innerhalb des `audio`-Elements können Sie den Namen einer alternativ abzuspielenden Audiodatei angeben, falls der Browser das Dateiformat der direkt im `audio`-Element angegebenen Datei nicht interpretieren kann. Sie können hier auch mehrere `source`-Elemente untereinander angeben.

Die Soundkonsole sieht in jedem Browser etwas anders aus.



Soundkonsole in Google Chrome



Soundkonsole im Microsoft Internet Explorer 9

Die direkt im audio-Element angegebene Sounddatei sollte ein Format haben, das der Browser auf jeden Fall erkennt. Zum Zeitpunkt der Erstellung des Buchs war unklar, welches Format sich durchsetzen wird. Sie können alternativ im Code z. B. zwei Formate gleichwertig nebeneinander angeben:

Beispiel: Audio in Webseite einbetten

Quelltext

html5-audio2.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Kalimba</title>
</head>
<body>
<h1>Kalimba </h1>
<div>
<audio preload controls>
   <source src="kalimba.mp3">
   <source src="kalimba.ogg">
</audio>
</div>
</body>
```

kalimba.mp3, kalimba.ogg

Das Element source kennt zwei Attribute:

type	Gibt ggf. den MIME-Typ der Audiodatei an, etwa audio/x-midi, audio/x-wav usw. Sie können das `type`-Attribut auch mit der Angabe des Codecs versehen, beispielsweise: `type="audio/ogg; codecs=vorbis"`
media	*Gibt an, für welche Medientypen die Audiodatei geeignet ist. Die Voreinstellung lautet:* `media="all"`

Diese Lösung funktioniert nur für Browser, die HTML5 unterstützen. Möchten Sie Ihre Multimedia-Dateien für ein möglichst breites Publikum zugänglich machen, können Sie auf JavaScript-Code zurückgreifen, der eine Lösung für ältere Browser bietet - diese spielen statt des unbekannten `audio`-Elements eine Flash-Datei ab.

Videos einbinden

Zum Einbinden eines Videos in HTML5 verwenden Sie das `video`-Element.

video	Innerhalb des `video`-Elements setzen Sie einen oder mehrere Verweise auf die Videodatei oder mehrere alternative, in unterschiedlichen Dateiformaten gespeicherte Versionen der Videodatei.

Folgende Attribute stehen für das `video`-Element zur Verfügung:

video	Innerhalb des video-Elements setzen Sie einen oder mehrere Verweise auf die Videodatei oder mehrere alternative, in unterschiedlichen Dateiformaten gespeicherte Versionen der Videodatei.

Beispiel: Video in Webseite einbetten

```
<h1>Diashow</h1>
<video src="diashow.mp4" width="640" height="480"
       controls preload poster="diashow.jpg">
  <source src="diashow.ogv" type="video/ogg"></source>
</video>
```

Quelltext

html5-video.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Diashow</title>
</head>
<body>
<h1>Diashow</h1>
<video src="diashow.ogv" width="436" height="320"
       controls preload poster="diashow.jpg">
  <source src="diashow.mp4" type="video/ogg"></source>
</video>
</body>
</html>
```

diashow.ogv, diashow.jpg, diashow.mp4

(1) Mit dem Tagpaar <video> … </video> binden Sie eine Video-Ressource in die Webseite ein. Mithilfe des Attributs src geben Sie die URL der abzuspielenden Videodatei an. Das Attribut preload sorgt dafür, dass der Browser direkt beim Laden der Seite mit dem Download der Videodatei beginnt. Das Attribut controls sorgt dafür, dass ein Video-Player auf der Webseite angezeigt wird. Mit diesem kann die Videodatei abgespielt, pausiert und angehalten sowie ihre Lautstärke verändert werden. Das Attribut poster verweist auf eine Grafik, die angezeigt wird, falls das Video nicht abgespielt werden kann.

(2) Mit dem source-Element innerhalb des video-Elements können Sie den Namen einer alternativ abzuspielenden Videodatei angeben, falls der Browser das Dateiformat der direkt im video-Element angegebenen Datei nicht kennt. Sie könnten hier auch mehrere source-Elemente untereinander angeben.



Video im Microsoft Internet Explorer 9

Die direkt im video-Element angegebene Videodatei sollte ein Format haben, das der Browser auf jeden Fall erkennt. Zum Zeitpunkt der Erstellung des Buchs war unklar, welches Format sich durchsetzen wird. Sie können alternativ im Code z. B. zwei Formate gleichwertig nebeneinander angeben:

Beispiel: Video in Webseite einbetten

```
<body>
<h1>Diashow</h1>
<div>
① <video width="640" height="480" controls preload poster="diashow.jpg">
② <source src="diashow.ogv" type="video/ogg"></source>
  <source src="diashow.mp4" type="video/mp4"></source>
</video>
</div>
</body>
```

Quelltext

html5-video2.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Diashow</title>
</head>
<body>
```

```
<h1>Diashow</h1>
<div>
<video width="436" height="320" controls preload poster="diashow.jpg">
   <source src="diashow.ogv" type="video/ogg"></source>
   <source src="diashow.mp4" type="video/mp4"></source>
</video>
</div>
</body>
```

diashow.jpg, diashow.ogv, diashow.mp4

Das Element `source` kennt zwei Attribute:

type	Gibt ggf. den MIME-Typ der Videodatei an, etwa video/mpeg, video/quicktime usw. Sie können das `type`-Attribut auch mit der Angabe des Codecs versehen, wenn Sie die genauen Codecs Ihrer Videodateien kennen, beispielsweise: `type="video/ogg; codecs=theora, vorbis"`
media	Gibt an, für welche Medientypen die Videodatei geeignet ist. Die Voreinstellung lautet: `Media="all"`

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_04

Last update: **2019/05/14 21:22**

11.1.4.2) Arbeitsaufträge

Übung 1: Sound einbinden

Übungsdateien:	Undertone.mp3, Undertone.ogg, Undertone.swf
Ergebnisdateien:	uebung1.html

(1) Binden Sie den Sound Undertone so in eine Webseite ein, dass er in den verschiedensten Browsern abgespielt werden kann.
(2) Fügen Sie eine Soundsteuerung ein, damit der Nutzer den Sound bei Bedarf abschalten kann.

Übung 2: Video einbinden

Übungsdateien:	Diashow.mp4, Diashow.ogv
Ergebnisdateien:	uebung2.html

(1) Binden Sie das Video Slideshow so in eine Webseite ein, dass es in allen möglichen Browsern abgespielt werden kann.
(2) Fügen Sie eine Grafik hinzu, die angezeigt wird, falls die Videodatei nicht abgespielt werden kann.
(3) Das Video soll bereits während der Übertragung abgespielt werden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_04:11_01_04_02

Last update: **2019/05/14 21:22**

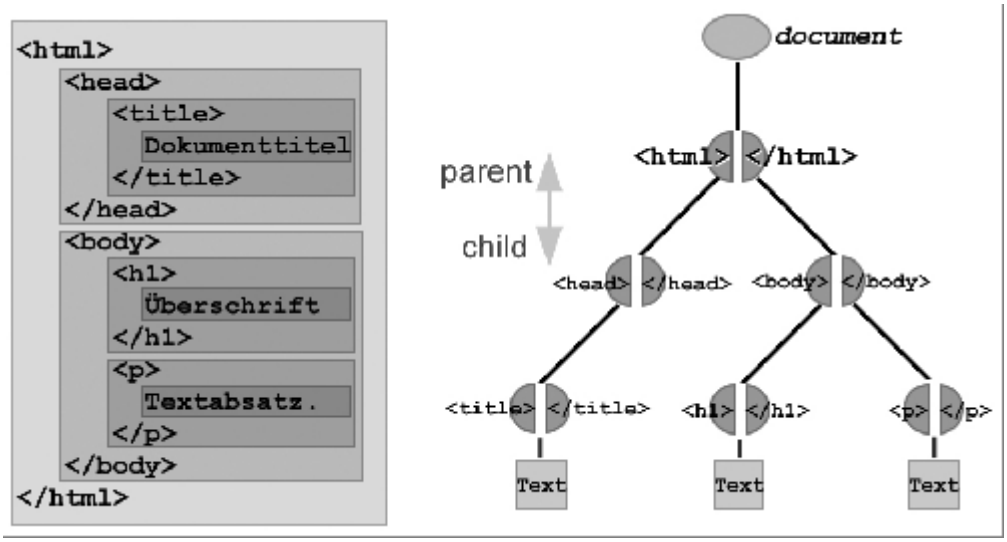
10.1.5) Die Synthese - das Document Object Model

Ein wichtiges Konzept im Zusammenhang mit dem Zugriff auf Webseiten ist das Document Object Model (.DOM). Das DOM stellt die Schnittstelle zwischen der Dokumentstruktur und dem Rest dar und dient Ihnen einerseits zur Verknüpfung von HTML und CSS, andererseits ist es auch das Interface, das Ihnen erlaubt, per Script auf Struktur und Präsentation des Dokuments einzuwirken.

Das Erstellen des DOM-Baums

Das DOM stellt eine Abstraktion des Dokuments dar, die der Browser beim Einlesen des HTML-Quelltexts im Arbeitsspeicher erzeugt. Eigentlich existiert es also rein virtuell. Stellen Sie es sich als Baumstruktur vor. und zwar in Form eines nach unten hängenden Baums, der an einem Punkt aufgehängt ist und sich ab dort verzweigt. Bezeichnen wir den Punkt, an dem der Baum ansetzt, als Dokumentknoten (`documentNode`). Dieser Dokumentknoten besitzt keine Entsprechung im Dokument, sondern stellt das »Dokument« an sich dar.

Einen solchen Dokumentknoten erzeugt der Browser beim Einlesen (Parsing) des Dokuments. Er hängt dann für jedes Element, das er in Quelltextreihenfolge antrifft, einen weiteren Knoten unten an diesen Dokumentknoten an. Das erste Element, das er antrifft, ist stets das Root-Element HTML (allgemein als `document`-Element bezeichnet). Ab hier spaltet sich der Baum in zwei Zweige, die den <head> und den <body> repräsentieren (siehe Abbildung).

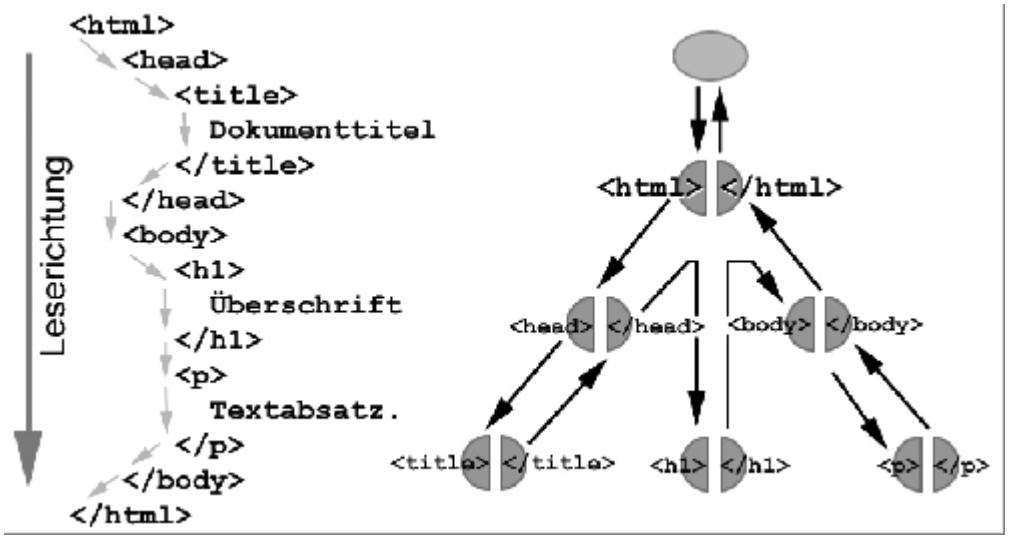


Der HTML-Quelltext und seine DOM-Repräsentanz

Der Baum besteht also aus Verbindungen und Knoten. Im Falle eines Elementknotens fallen in diesem Start- und Endmarke des Quelltexts zusammen. Textknoten bilden stets das Ende eines Zweiges (»leaf nodes«). Enthält ein Element einen Inhalt (gehen wir also hierarchisch in dessen Inneres), fächert sich der Baum weiter nach unten auf. Hierdurch entsteht für jeden Bestandteil dieses Inhalts ein weiterer Ast mit daran hängenden Knoten.

Hierbei gilt der oben liegende Knoten als »Elternknoten« (»parent«), die von ihm unmittelbar abstammenden Knoten als »Kindknoten« (»children«). Sowohl Elemente als auch Textknoten stehen

stets in einer Eltern-Kind-Beziehung, wobei jeder Knoten genau einen Elternknoten besitzt (niemals mehrere). Ein Element-knoten besitzt darüber hinaus weitere Eigenschaften, zu denen (salopp gesprochen) auch seine Attribute gehören. Der Browser baut auf diese Weise sukzessive ein komplettes Abbild der hierarchischen Struktur des Dokuments auf.



Das DOM ist also eine Abstraktion folgender Information:

- Wie ist der hierarchische Zusammenhang in der Dokumentstruktur?
- Welches Element ist an welcher Position der Hierarchie?
- Welche Eigenschaften hat es (z. B. Attribute, Inhalte, Nachbarelemente)?

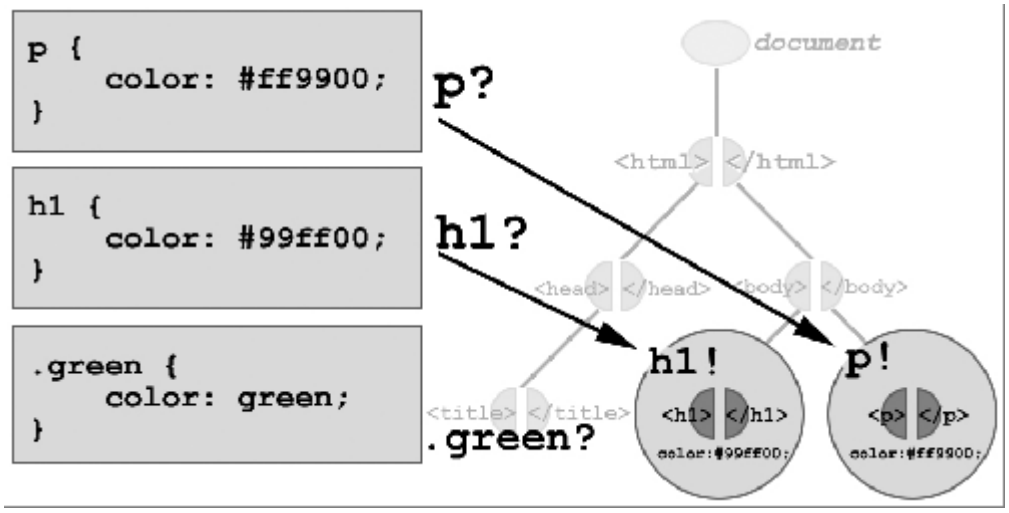
Hier soll kurz ein Vorteil des DOM-Konzepts gegenüber »seriellem« Quelltext erwähnt werden, der darauf beruht, dass ein Knoten gleichzeitig Start- und Endmarke eines HTML-Tags repräsentiert: Bei einer Manipulation des Baums (wir werden gleich sehen wie das geht), also der Entnahme oder dem Hinzufügen einzelner Knoten oder ganzer Zweige, wird stets mit vollständigen Strukturen gearbeitet. Ein Dokument behält so stets seine Wohtgeformtheit, also die Art von regelmäßiger Struktur, die in der XMT.-Datenverarbeitung gefordert ist.

Das »Schmücken« des DOM-Baums

Dieses Modell, in dem alle Elemente des Dokuments repräsentiert sind, ist noch eine Abbildung der »nackten« Struktur - durchaus wörtlich zu nehmen, denn bis jetzt hat der Browser die CSS-Informationen noch nicht eingebracht. Die nun folgende Phase wird als »decorating the tree« bezeichnet: Der Browser liest alle CSS-Informationen ein und löst dabei auftretende Konflikte und Unstimmigkeiten auf. Anschließend liegt das sogenannie Stylesheet vor: die vollständigen Präsentationsvorschriften, die auf das Dokument angewendet werden sollen.

Nun werden die Selektoren aller CSS-Regeln registriert und der Dokumentbaum anhand dieser Vergleichsmuster durchsucht. Jeder Elementknoten, auf den das Muster zutrifft, bekommt den Rcgclsatz zugewiesen. Auf diesem Weg werden alle Regeln durchgearbeitet. Regeln, für die keine Übereinstimmung mit dem Dokument gefunden wird, liegen brach (siehe folgende Abbildung).

Bekommt ein Element hierbei mehrfach Regeln zugewiesen, addieren sich diese. Ein dabei auftretender Konflikt wird nach Rang der Regel beigelegt. Sobald der Vorgang beendet ist, ist der Baum »dekorierte und der Browser geht daran, ihn im Viewport darzustellen.



Der DOM-Baum wird per Vergleichsmuster durchsucht

Im dekorierten Baum sind folgende Informationen zusätzlich in den einzelnen Elementknoten gespeichert:

- Welche Präsentationsvorschriften existieren für dieses Element?

Die Stilevorschriften können wir uns durchaus als »Eigenschaften« des Elements vorstellen, die beim Elementknoten gelagert werden. Wir werden gleich sehen, dass dies uns hilft, die CSS-Regeln eines Elements auszulesen und auch bei Bedarf zu ändern. In diesem Augenblick kommt wieder JavaScript ins Spiel.

Manipulation von DOM und CSS per JavaScript

Der dekorierte Baum stellt, wie wir jetzt wissen, ein Abbild des Dokuments im Arbeitsspeicher des Browsers dar, das der Darstellung des Dokuments im Viewport entspricht. Da es sich um eine rein virtuelle Sache handelt, ist dieses Abbild per Programmierung beliebig manipulierbar. Genau dies ist überhaupt die Aufgabe des DOM- eine Schnittstelle (API) zu bieten, die es ermöglicht, mittels einer Programmiersprache auf das Dokument einzuwirken.

In Zusammenhang mit JavaScript bietet das DOM eine Reihe von Schnittstellenfunktionen, die eine Brücke schlagen zwischen der Scripting-Umgebung und dem Dokumentbaum. Einige dieser Methoden sind dem Dokumentknoten (den wir hierfür praktisch erweise einlach als JavaScript-Objekt betrachten! unterstellt. Andere Methoden stehen auch direkt den Elementknoten zur Verfügung.

Diese DOM-Methoden sind nicht ausgesprochen zahlreich und zum Teil umständlich anzuwenden - ihre Anzahl hängt zudem von der Implementierung des DOM ab (»DOM-Level«), die der ausführende Browser jeweils unterstützt.

Dies sind ihre Aufgaben:

- Selektieren von Knoten
- Traversieren (Bewegung) innerhalb des Baums
- Wert (Inhalt) eines Knotens auslesen oder schreiben
- Attribute eines Elementknotens lesen oder schreiben
- Erzeugen von Element- und Attributknoten
- Einhängen von Knoten in den Baum

- Löschen von Knoten im Baum

DOM-Methode	Erläuterung
addEventListener()	Bindet Event Handler an DOM-Element.
removeEventListener()	Löst Event Handler an DOM-Element.
createAttribute()	Erzeugt einen Attributknoten.
createTextNode()	Erzeugt einen Textknoten.
createElement()	Erzeugt einen Elementknoten.
getAttribute()	Liest einen Attributwert.
setAttribute()	Schreibt einen Attributwert.
removeAttribute()	Entfernt einen Attributknoten.
appendChild()	Hängt Elementknoten am Ende des Inhalts des aktuellen Knotens ein.
insertBefore()	Hängt Elementknoten vor dem aktuellen Knoten ein.
removeChild()	Entfernt Kindknoten des aktuellen Knotens.
replaceChild()	Ersetzt Kindknoten des aktuellen Knotens.
getElementById()	Referenziert einen Elementknoten per ID.
getElementsByClassName()	Erstellt NodeList aus Elementknoten nach CSS-Klasse.
getElementsByTagName()	Erstellt NodeList aus Elementknoten nach Tag-Bezeichner.

Die wichtigsten DOM-Methoden zur DOM-Manipulation

Eine umfassende Onlinereferenz zu DOM (in englischer Sprache) findet man bei der Mozilla Foundation unter dieser Adresse

https://developer.mozilla.org/en/DOM

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_05

Last update: **2019/05/14 21:23**

11.1.6) HTML5

Mit der Auszeichnungs- bzw. Programmiersprache HTML (Hypertext Markup Language) können Sie mit Hilfe von Codes Webseiten erstellen. Die Organisation „World Wide Web Consortium“ (W3C) legt dabei fest, welche Vokabeln und welche Grammatik in diesen Codes verwendet werden müssen. 2014 erschien HTML5 als Nachfolger von der vierten HTML-Version und brachte eine Menge neuer Darstellungsmöglichkeiten und Funktionen mit sich.

Neuerungen unter HTML5

Die Auszeichnungssprache HTML4 wurde Ende der 90er Jahren entwickelt und 10 Jahre lang kaum überarbeitet. Dabei unterschied sich die Nutzung des Internets mittlerweile enorm von damals: Stand in den 90er Jahren noch die Darstellung von Inhalten im Vordergrund, geht es heutzutage vermehrt um Interaktion. Kommentarfunktionen, Multimedia-Elemente und Social-Media-Verknüpfungen waren zuvor nur umständlich über Plug-ins integrierbar - HTML5 vereinfacht die Integration solcher Funktionen. Im Folgenden haben wir die wichtigsten Neuerungen unter HTML5 zusammengefasst:

Strukturierende Elemente

Eine entscheidende Neuerung, die HTML5 mit sich brachte, ist die Einführung von strukturierenden Elementen wie Abschnitte (<section>), Navigationsleisten (<nav>), eigenständige Inhalte innerhalb eines Dokuments (<article>) oder Header (<header>). Diese strukturierenden Elemente sind besonders hilfreich, um Suchmaschinen das Crawlen zu erleichtern und damit ein gutes Ranking in den Suchergebnissen zu erzielen.

Multimedia-Inhalte

Mit HTML5 ist das Einbinden von Audio- und Videodateien ebenfalls ohne ein zusätzliches Plug-in möglich. Über die Elemente <audio> und <video> lassen sich die Dateien genauso einbauen wie Bilder.

Interaktive Elemente

Interaktive Elemente können mit HTML5 ebenfalls in eine Webseite eingebunden werden. Über <details> lassen sich beispielsweise ausklappbare Elemente integrieren. <summary> definiert dabei die dazugehörige Überschrift, die ständig angezeigt wird. Mit Hilfe von <menu> lassen sich auch Kontextmenüs erstellen.

Formular-Elemente

Im Vergleich zu HTML4 kann HTML5 erweiterte Formularelemente darstellen. So kann zum Beispiel dargestellt werden, inwieweit eine Aufgabe abgeschlossen ist (<progress>) oder es können

Vervollständigungsvorschläge gegeben werden (<datalist>).

Weitere Vorteile

Neben den spezifischen Neuerungen hat HTML5 weitere Vorteile gegenüber der Vorgänger-Version HTML4. Da der Code übersichtlicher und strukturierter aufgebaut ist als zuvor, ist die spätere Überarbeitung einfacher, der Code ist weniger anfällig für Fehler und die Kompatibilität der programmierten Seite in den unterschiedlichen Browsern steigt. Außerdem sorgt die gute Strukturierung für eine bessere Auffindbarkeit der Seite durch Suchmaschinen – Stichwort SEO.

Ein weiterer Vorteil von HTML5 ist die Möglichkeit, Web Apps zu erstellen, die auf Smartphones und Tablets besser den Inhalt darstellen können.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_01:11_01_06

Last update: **2019/05/22 18:52**

11.2) CSS

- 11.2.1) Grundlagen
- 11.2.2) Deklarationen
- 11.2.3) Vererbung und Kaskadierung
- 11.2.4) XHTML-Elemente gestalten
- 11.2.5) Weiteres
- 11.2.6) Webseiten gestalten mit CSS - Anwendungsbespiele
- 11.2.7) Übungen
- 11.2.8) Links

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02

Last update: **2019/05/14 21:27**

11.2.1) Grundlagen

- 11.2.1.1) Was ist CSS?, Geschichte und Vorteile von CSS
- 11.2.1.2) Einbindung in (X)HTML
- 11.2.1.3) Probleme
- 11.2.1.4) Einheiten
- 11.2.1.5) Block- und Inlineelemente
- 11.2.1.6) Exkurs: Dokumenttyp-Definitionen
- 11.2.1.7) Exkurs: Unterschiede zwischen HTML und XHTML

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01

Last update: **2019/05/14 21:30**

11.2.1.1) Cascading Styleseets



Was ist CSS?

CSS steht für Cascading Stylesheets, „kaskadierende Stilunterlagen“. CSS wird zum Designen eines (X)HTML-Layouts (Struktur) verwendet. CSS ist somit eine Möglichkeit, das Aussehen einer Website zu steuern, und bietet dazu unmengen an Attributen für die (X)HTML-Elemente.

Geschichtlich

1994 wurde der erste Entwurf von CSS veröffentlicht, 1996 wurde es schließlich vom vom W3C als Empfehlung freigegeben. (W3C: Word-Wide-Web-Consortium, es veröffentlicht Empfehlungen die als „De-faco-Standards“ fungieren. Tatsächlich Standards vorgeschrieben werden vom W3C jedoch nicht.)

Die neue Version CSS3 ist aktuell in Entwicklung, und wird viele neue Möglichkeiten zur Gestaltung von Websiten bieten. Die Spezifikation wird als *working draft* eingestuft. Viele Browser unterstützten in aktuellen Versionen aber schon viele Funktionen von CSS3.

Vorteile von CSS

- Trennung von Inhalt und Layout möglich
- Einfachere Wartung, Änderung in CSS-Datei gilt sofort für gesamte Website mit allen Unterseiten
- Suchmaschinenoptimierung einfacher, da die Crawler den CSS-Code nicht lesen und sich somit ganz auf den Inhalt der Website konzentrieren können
- Erleichterte Barrierefreiheit
- Deutlich mehr Möglichkeiten zur Formatierung als mit (X)HTML.

Beispiele

Beispiele für die Verwendung von CSS sind auf der Website http://www.csszengarden.com zu finden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01:11_02_01_01

Last update: **2019/05/14 21:30**

11.2.1.2) Einbindung in (X)HTML

CSS-Eigenschaften können im wesentlichen auf 3 Arten mit (X)HTML verwendet werden:

Externe Styles: Einbindung einer CSS-Datei im <head>-Bereich

Die CSS-Eigenschaften werden in eine externe Datei, meist mit der Endung „.css“ ausgelagert. Diese Datei wird dann im <head>-Bereich der (X)HTML-Seite mit einem <link>-Tag eingebunden.

```
<head>
<-- andere Head-Angaben -->
<link rel="stylesheet" type="text/css" href="style.css" />
</head>
```

Interne Styles: Definieren der CSS-Eigenschaften im <head>-Bereich

Bei dieser Methode werden die CSS-Eigenschaften nicht ausgelagert, sondern direkt im <head>-Bereich festgelegt. Dies geschieht innerhalb eines style-Tags.

```
<head>
<-- andere Head-Angaben -->
<style type="text/css">
/* <![CDATA[ */
h2 {
     background-color: #E6DACF;
     color: #806040;
}
p  {
     background-color:#CFDCE6;
     color: #406480;
}
/* ]]> */
</style>
</head>
```

Inline-Styles: CSS-Eigenschaften direkt im (X)HTML-Tag festlegen

Diese Methode eignet sich besonders, wenn es nur 1-2 CSS-Formatierung für die ganze Seite gibt. Dann wird die Eigenschaft direkt im Tag innerhalb eines style-Attributs festgelegt.

```
<span style="font-size: 10px; font-family: serif">Das ist ein Text.</span>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01:11_02_01_02

Last update: **2019/05/14 21:30**

11.2.1.3) Probleme

Unterschiedliche Implementierungen in Browsern

Die CSS-Codes werden vom Browser verarbeitet, somit hängt die Darstellung von der jeweiligen Implementierung ab. Häufig sind manche CSS-Eigenschaften in manchen Browsern nicht implementiert, obwohl sie schon lange Standard sind - andererseits werden aber auch Eigenschaften implementiert, die noch nicht standardisiert wurden. Auch kommt es immer wieder vor, dass die Browser die CSS-Codes unterschiedlich interpretieren, also sieht eine Website im schlimmsten Fall in jedem Browser anders aus. Ähnlich gestaltet sich dieses Problem auch bei Javascript und manchen HTML-Tags. Abhilfe gibt es mit Browser-Weichen (kommt später) und CSS-Hacks (kommt nicht).

Größere Probleme in dieser Hinsicht gab es in den letzten Jahren beim Internet Explorer, weil Microsoft häufig eigene Implementierungen verwendete. Seit einiger Zeit beteiligt sich Microsoft selbst am W3C und will ebenfalls deren Empfehlungen umsetzen. Mit dem in naher Zukunft erscheinenden Internet Explorer 9 gelingt das weitgehend, während es im IE8 und IE7 schon Fortschritte gab.

Browserspezifische Eigenschaften

Manchmal bauen Browserhersteller für ihren Browser spezifische Eigenschaften ein, ein Beispiel wäre die Eigenschaft **-moz-boarder-radius:**, mit dieser ist es möglich, dass Boxen mit abgerundeten Ecken dargestellt werden. Auch der Browser Safari hat hierzu eine eigene Eigenschaft, **-webkit-border-radius:**. Diese Eigenschaften werden in anderen Browsern dann natürlich ignoriert.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01:11_02_01_03

Last update: **2019/05/14 21:31**

11.2.1.4) Größenangaben

Für die Angabe von Schriftgrößen, Längen und Breiten, etc. gibt es in CSS verschiedene Einheiten.

relative Angaben

Zu bevorzugen, da die Ausgangsgröße vom Browser/Betriebssystem bestimmt werden kann, und somit die Einstellungen des Users nicht überschrieben werden.

```
font-size: 0.9em; /* Das 0.9-fache der Normalgröße */
font-size: 15px; /* Pixel/Bildpunkt */
font-size: 2ex; /* Angabe der Schriftgröße, relativ zum x der Normalgröße.
*/
font-size: 200%; /* doppelte Größe */
```

absolut

```
font-size: 10pt; /* Punkt (1 pt entspricht 1/72 Zoll) */

font-size: 1cm;
font-size: 12mm;
font-size: 1.5in; /* Zoll */
font-size: 12pc; /* pica (1 pc entspricht 12pt) */
```

Weitere Informationen

Eine vollständige Auflistung der Einheiten und weitere Informationen über diese findet man unter http://www.css4you.de/einheiten.html.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01:11_02_01_04

Last update: **2019/05/14 21:31**

11.2.1.5) Block- und Inlineelemente

(X)HTML-Elemente unterscheidet man in 2 Klassen: Block-Element und Inline-Elemente. Zwischen diesen beiden Typen gibt es Unterschiede in der Handhabung per CSS und den verfügbaren CSS-Eigenschaften. Die Unterschiede und Eigenschaften sind häufig auch von den Dokumenttyp-Definitionen abhängig.

Block-Elemente

- Starten in neuer Zeile und enden mit Zeilenumbruch
- Folglich ist es ohne Einsatz von CSS nicht möglich, mehrere Block-Elemente nebeneinander darzustellen
- Block-Elemente können Inline-Elemente und andere Block-Elemente beinhalten, manchmal auch nur Inline-Elemente (z.B. ... - Absatz), nur Block-Elemente oder keine weiteren Elemente (→ CSS4You.de Referenz!)
- Verwendung oft als Container, also übergeordnetes Element für andere Elemente

Wichtige CSS-Eigenschaften, die nur auf Block-Elemente angewendet werden können (Erklärung zur Anwendung kommt erst später, passt hier aber am besten):

Eigenschaft	**Erklärung**
clear	Umfließen beenden
text-align	Textausrichtung/Horizontale Ausichtung
text-indent	Texteinrückung

Inline-Elemente

- Werden in der aktuellen Zeile (im Textfluss) dargestellt (
 für Zeilenumbruch davor und danach notwendig)
- Inline-Elemente können nur andere Inline-Elemente beinhalten, manchmal auch keine weiteren Elemente (→ CSS4You Referenz!)
- Verwendungszweck sind für gewöhnlich Inhalte, die der Benutzer sieht, also Text, Bilder, Links, ...

Die **vertical-align:** (Vertikale AUsrichtung) - Eigenschaft kann nur auf Inline-Elemente (und Tabellenzellen) angewendet werden.

Bei vielen Elementen kann man die Darstellung als Block-Element oder Inline-Element erzwingen, auch wenn es sich nicht um ein solches ELement handelt. Dies geht mit der Eigenschaft display: - Informationen darüber hier.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01:11_02_01_05

Last update: **2019/05/14 21:31**

11.2.1.6) Dokumenttyp-Definitonen

Zwar nicht unbedingt im Zusammenhang mit CSS stehend, aber generell für die Entwicklung von (X)HTML-Seiten wichtig sind Dokumenttypen (DTDs).

Was sind Dokumenttypen?

Dokumenttypen beschreiben Regeln für ein XML- oder SGML-Dokument, also sowohl, wie diese aufgebaut sein sollen (vom Entwickler), als auch wie sie vom „übersetzenden“ Programm verstanden werden soll (im Falle von HTML/XHTML ist dies der Browser). Grundsätzlich sollte man immer eine DTD angeben, tut man das nicht, verwendet der Browser eine Standarddefinition.

Beispiele für DTDs

DTD von XHTML 1.0 Transitional:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

DTD von XHTML 1.0 Frameset:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

DTD von HTML 4.01 Transitional:

```
   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
            "http://www.w3.org/TR/html4/loose.dtd">
```

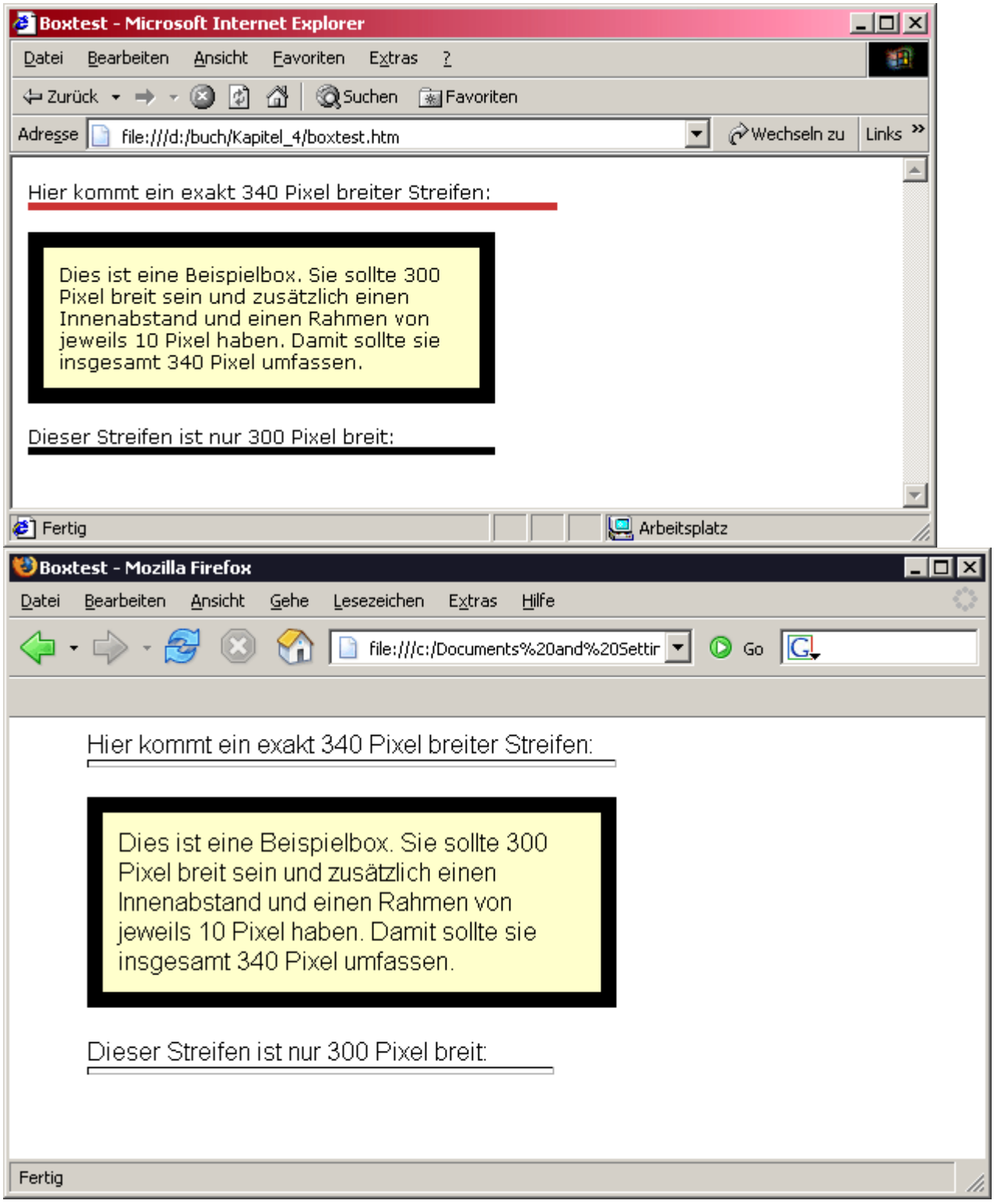
Wichtigkeit der Dokumenttyp-Angabe

Oft ist nicht ganz klar, wofür die Dokumenttyp-Angaben überhaupt notwendig sind, und es werden dann falsche oder garkeine angegeben. Diese sind jedoch sehr wichtig: Jeder Browser hat verschiedene Modis, mit denen er XHTML, HTML und CSS-Code interpretiert - und die DTDs entscheiden, welchen Modus der Browser verwendet!

- Gibt man einen anderen DTD an, als den, auf dem die eigene Webseite basiert (Beispiel: DTD-Angabe ist HTML4, man arbeitet aber mit XHTML) interpretiert der Browser den XHTML-Code als HTML4-Code, wenn es nun bei einem Element unterschiede zwischen XHTML und HTML4 gibt, zeigt der Browser das Element eventuell nicht wie gewünscht an.
- Gibt man garkeinen oder einen sehr alten (z.B. HTML2, HTML3) DTD an, fallen die Browser in den sogenannten Quirks-Mode zurück: Die Webseite wird dann mit veralteten Versionen der W3C-Empfehlungen bzw. nicht korrekt umgesetzten W3C-Empfehlungen angezeigt. Der Grund

für das vorhandensein eines solchen Quirks-Modus liegt darin, dass auch alte Webseiten noch korrekt angezeigt werden können.

- Der Internet Explorer stellte bis Version 6 das Boxmodell falsch dar (er berechnete die Außenabstände im Zusammenhang mit der Gesamtbreite/höhe eines Elements falsch), während das bei aktuellen Browserversionen und korrektem DTD kein Problem mehr darstellt, wird genau diese falsche Berechnungsweise im Quirks-Modus des Internet Explorers und auch anderer Browser verwendet.



(Quelle der Screenshots:
http://www.dma.ufg.ac.at/app/link/Grundlagen%3AInternet.Web.Multimedia/module/15459?step=all)

Darum: Immer den korrekten (X)HTML-DTD angeben!

Weitere Informationen

Weitere Informationen über DTDs und deren Aufbau, wenn man selbst DTDs erstellen will, gibt es bei Wikipedia.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01:11_02_01_06

Last update: **2019/05/14 21:32**

11.2.1.7) Exkurs: Unterschiede zwischen HTML und XHTML

Der Nachfolger von HTML 4.01 ist XHTML 1.0 (eXtensible HTML) - damit änderte sich die Basis von HTML grundlegend, auch wenn die Art und Weise, auf die es verwendet wird, nach wie vor sehr ähnlich ist. Die Basis von HTML ist SGML, die von XHTML ist XML.

Dokumenttypen

Um dem Browser klarzumachen, dass es sich um ein XHTML-Dokument handelt, müssen bestimmte Dokumenttypen verwendet werden. Man unterteilt XHTML in 3 verschiedene Typen: Strict, Transitional, Frameset - Im Normalfall verwendet man Transitional, Frameset nur, wenn man Frames verwendet. Strict hingegen macht häufig zu strikte Vorschriften und streicht auch sinnvolle Tags und Attribute (z.B. das Öffnen eines Links in einem neuen Fenster).

Die Dokumenttypangabe von XHTML Transitional ist:

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Namensraumangabe

Aufgrund des Umstiegs auf XML muss zusätzlich der XML-Namespace mit der verwendeten Sprache im <html>-Tag angegeben werden. Das sieht so aus:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
```

Kleinschreibung

Während es dem Entwickler in HTML 4.01 noch freigestellt war, „müssen“ (um ein valides Dokument zu erhalten - funktionieren wird es auch anders) in XHTML sämtliche Tags und Attribute kleingeschrieben werden.

End-Tag notwendig

XHTML verlangt, dass jeder Tag auch einen End-Tag hat. Während man bei HTML 4.01 in Listen bei den einzelnen Punkten oft nur <li>... geschrieben hat, also auf den schließenden Tag verzichtet hat, erfordert XHTML nun auch den End-Tag, also <li>....

Eine Element, welches keinen Inhalt zwischen den Tags hat, wie z.B. der <img>-Tag, kann auch direkt

mit einem Schrägstrich am Ende geschlossen werden: .

Dies gilt auch für „leere Elemente“, am wichtigsten ist hier <br />, welches nun den schließenden Tag hat.

Schachtelung

Die Elemente müssen in XHTML korrekt verschachtelt werden, d.h., ein Element, welches innerhalb eines anderen Elements geöffnet wurde, muss auch wieder innerhalb von diesem Element geschlossen werden.

- Falsch: ...
- Richtig: ...

Attribute müssen Werte haben

Attribute „müssen“ in XHTML einen Wert zugewiesen bekommen, die reine Angabe eines Attributs wie bei einer Checkbox das Attribut „checked“ ist nicht mehr erlaubt - stattdessen muss checked=„checked“ verwendet werden.

id- statt name-Attribut

Während man in HTML das name-Attribute verwendete, um ein Element eindeutig zu identifizieren (z.B. für die Verwendung mit Javascript), wird dieses in XHTML mit dem Attribut id ersetzt.

Einbinden von CSS/JS/Sonstigem im <head>-Bereich

CSS-Definitionen, JavaScript-Code, etc. müssen in XHTML in besondere „CDATA-Bereiche“ gesteckt werden, um sicherzugehen, dass der Browser den Inhalt nicht falsch versteht.

```
<style type="text/css">
/*<![CDATA[*/
CSS-CODE
/*]]*/
</style>
```

Ja, das ist kompliziert, und die Chance, sich dieses Konstrukt fehlerfrei zu merken ist eher gering. Darum empfiehlt es sich, es zu speichern und bei Bedarf einzufügen.

Konvertierung, Validierung

- Mithilfe des W3C-Tools „Tidy“ kann man HTML-Dokumente in XHTML konvertieren. Es steht zur

Verfügung unter http://www.w3.org/People/Raggett/tidy

- Mit dem XHTML-Validator unter http://validator.w3.org kann man überprüfen, ob man ein W3-Konformes XHTML-Dokument erstellt hat.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_01:11_02_01_07

Last update: **2019/05/14 21:32**

11.2.2) Deklarationen

- 11.2.2.1) Die verschiedenen Selektoren
- 11.2.2.2) Klassen und IDs
- 11.2.2.3) Kommentare
- 11.2.2.4) Verschachtelung
- 11.2.2.5) Kurzformen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_02

Last update: **2019/05/14 21:33**

11.2.2.1) Selektoren für HTML-Tags

Elementtypen

Um ein zentrales Format für alle HTML-Elemente eines bestimmten Typs zu definieren, wird als als sog. **Selektor** der Name des HTML-Elementtyps angegeben, und zwar ohne spitze Klammern. Im folgenden Beispiel werden `body` (Dokument), `h1` (Überschriften 1. Ordnung), `p` (Textabsätze) und `li` (Listeneinträge) auf diese Weise notiert.

Über den Universalselektor * hat man die Möglichkeit, Eigenschaften für alle Elemente zu definieren, wobei jedem Element natürlich nur die jeweils zulässigen Eigenschaften zugewiesen werden.

Wenn ein Format für mehrere HTML-Elementtypen definiert werden soll, werden alle gewünschten Elementtypen angegeben-

```
/* für einzelne Tags */
body { background-color: #FFFFCC;
       margin-left: 100px; }
h1 { font-size: 300%;
     color: #FF0000;
     font-style: italic;
     border-bottom: solid thin black; }

/* für mehrere Tags */
p,li { font-size: 110%;
       line-height: 140%;
       font-family: Helvetica,Arial,sans-serif;
       letter-spacing: 0.1em;
       word-spacing: 0.3em; }

/* für alle Tags */
* { color:blue; }
```

Nachfahren-Selektor

```
Selektor1 Selektor2 … {Eigenschaft: Wert}
```

```
<style type="text/css">
    p b { color: #a00; }
</style>
```

Kind-Selektor

```
Selektor1>Selektor2 {Eigenschaft: Wert}
```

```
<style type="text/css">
    p>b { background-color: #eee; }
</style>
```

spezielle Selektoren

Übersicht
Übersicht (zur Verknüpfung und Verschachtelung)

Wichtig sind nur die für die Links

```
a:link {color:#FF0000;}    /* unvisited link */
a:visited {color:#00FF00;} /* visited link */
a:hover {color:#FF00FF;}   /* mouse over link */
a:active {color:#0000FF;}  /* selected link */
```

(Die Reihenfolge der Deklaration ist wichtig!)

:active	aktiver Hyperlink
:after	Nach einem Element
:before	Vor einem Element
:first-child	Das erste „Kind“
:first-letter	Das erste Zeichen einer Zeile
:first-line	Die erste Zeile in einem Absatz
:focus	Fokusierte Formularelemente
:hover	mouse over
:link	Hyperlink
:visited	besuchter Link

Beispiel

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_02:11_02_02_01

Last update: **2019/05/14 21:34**

11.2.2.2) Klassen & IDs

Zur Verknüpfung von CSS mit (X)HTML werden primär sogenannte Klassen und IDs verwendent, womit einzelne und Gruppen von Elementen gestaltet werden können.

- **IDs**:
  - weist man einem einzigen Element im XHTML-Code zu, es darf nur einmal vorkommen und bestimmt ein Element somit eindeutig (was auch bei Javascript wichtig ist).
  - werden im XHTML-Code mit dem Attribut **id** angegeben.
  - im CSS-Code werden sie mit dem Rautezeichen (#) angesteuert.
  - Verwendung z.B. für ein DIV, in dem der Footer der Website liegt - dieser kommt nur einmalig vor
- **Klassen**:
  - werden bzw. können mehreren Elementen im XHTML-Code zugewiesen werden
  - werden im XHTML-Code mit dem Attribut **class** angegeben.
  - im CSS-Code werden sie mit einem Punkt (.) angesteuert
  - Verwendung z.B. für Überschriften, eine Überschrift des gleichen Ranges (z.B. h3) kann auf der selben Seite mehrmals vorkommen

Deklarierung

... in (X)HTML als Attribut

```
<ul id="menueliste">
<li class="menuepunkt">eins</li>
<li class="menuepunkt">zwei</li>
<li class="menuepunkt">drei</li>
<li class="menuepunkt">vier</li>
<li class="menuepunkt">...</li>
</ul>
```

... in CSS

```
#menueliste {
/* CSS-Eigenschaften für Elemente mit ID menueliste */
}

.menuepunkt {
/* CSS-Eigenschaften für Elemente mit Klasse menuepunkt*/
}
```

auch möglich:

```
ul#menueliste {
/* CSS-Eigenschaften */
```

```
}

li.menuepunkt {
/* CSS-Eigenschaften */
}
```

Beispiel

```
p.merksatz {
   font-style: italic;
   border: 4px solid #EE0000;
}


/* hätte man auch ohne ID-Tag ansteuern können, da h1 ohnehin nur einmal
vorkommt */
h1#ueberschrift {
   font-size: 0.9em;
   color: darkred;
}
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>TITEL</title>
   <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>

<h1 id="ueberschrift">Merksatz</h1>
<p>Text im Absatz ohne CSS-Formatierung</p>
<p class="merksatz">Das ist ein Merksatz, der per CSS als solcher Formatiert
wird. Man sollte ihn nie vergessen.</p>
</body>
</html>
```

DIV und SPAN

Die beiden HTML-Elemente und haben besondere Bedeutung für Stylesheets. Der Grund ist, dass sie selber eigenschaftslos sind. Der einzige Unterschied besteht darin, dass das div-Element eine neue Zeile im Textfluss erzwingt, während span innerhalb eines Textes verwendet werden kann und keinen neuen Absatz erzeugt. Unter Verwendung von div oder span ist es also möglich, beliebigen Textabschnitten Klassenstyles zuzuweisen.

Links

- Klassen/Klassen-Selektoren auf 4webmaster.de
- IDs/ID-Selektoren auf 4webmaster.de

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_02:11_02_02_02

Last update: **2019/05/14 21:34**

11.2.2.3) Kommentare

Kommentare sind in CSS zwischen

```
/* und */
```

möglich.

einzeilige Kommentare

```
p.left {
   background-color: gray;    /* Die Hintergrundfarbe ist grau */
}
```

mehrzeilige Kommentare

Das gleiche gilt für einen **mehrzeiligen Kommentar**:

```
p.left {
   /* Das wird ein mehrzeiliger Kommentar.
      Die Hintergrundfarbe wird jetzt als grau festgelegt. */
   background-color: gray;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_02:11_02_02_03

Last update: **2019/05/14 21:34**

11.2.2.4) Verschachtelung

Da der (X)HTML-Code eigentlich nie nur eine Ebene hat, ist es möglich die CSS-Definitionen entsprechend dem (X)HTML-Code zu verschachteln. Diese Verschachtelung ist beliebig weit nach „innen“ möglich. Dazu ein kleines Beispiel:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>TITEL</title>
   <link rel="stylesheet" type="text/css" href="style.css" />


</head>
<body>
<h1>Ein <span>teilweise kursiver</span>Header</h1>
<div id="beispiel">
   Das ist ein Text direkt im DIV-Element
   <p>Und dieser Text ist im P-Element innerhalb des DIV-Elements. <span>Und
ein SPAN-Element im P-Element im DIV-Element.</span></p>
</div>
<p>Ein P-Element außerhalb des DIV-Elements - die CSS-Eigenschaft von vorher
greift hier nicht.</p>
</body>
</html>
```

style.css:

```
h1 span {
   font-style: italic;
}

div#beispiel p {
   font-weight: bold;
}

div#beispiel p span {
   text-decoration: underline;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_02:11_02_02_04

Last update: **2019/05/14 21:34**

11.2.2.5) Kurzformen

Manche zusammengehörende Eigenschaften lassen sich als Kurzform als eine einzige Eigenschaft darstellen. Dies macht den CSS-Code kürzer und, wenn man die Kurzformen versteht, auch übersichtlicher. Man gibt die einzelnen Werte einer Kurzform mit Leerzeichen getrennt an, also:

- kurzformeigenschaft: wert1 wert2 wert3;

Dazwischen können Werte ausgelassen werden, die Reihenfolge darf jedoch nicht verändert werden. kurzformeigenschaft: wert1 wert3 wäre also ok - kurzformeigenschaft: wert3 wert2 wert1 aber nicht.

Wichtige Kurzformen

Kurzformbezeichnung	Werte	Beispiel
background	background-color background-image background-attachment background-repeat background-position	background: white url(bild.png) fixed no-repeat top left;
font	font-size line-height font-weight font-style font-variant font-family	font: italic small-caps bold 1em/1.5em verdana, sans-serif;
border	border-width border-color border-style	border: 3px black solid;
list-style	list-style-type list-style-image list-style-position	list-style: square outside;

Alle Kurzformen sind auch in der CSS4You-Referenz zu finden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_02:11_02_02_05

Last update: **2019/05/14 21:34**

11.2.3) Vererbung & Kaskadierung

- 11.2.3.1) Vererbung
- 11.2.3.2) Kaskadierung und Spezifität

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_03

Last update: **2019/05/14 21:36**

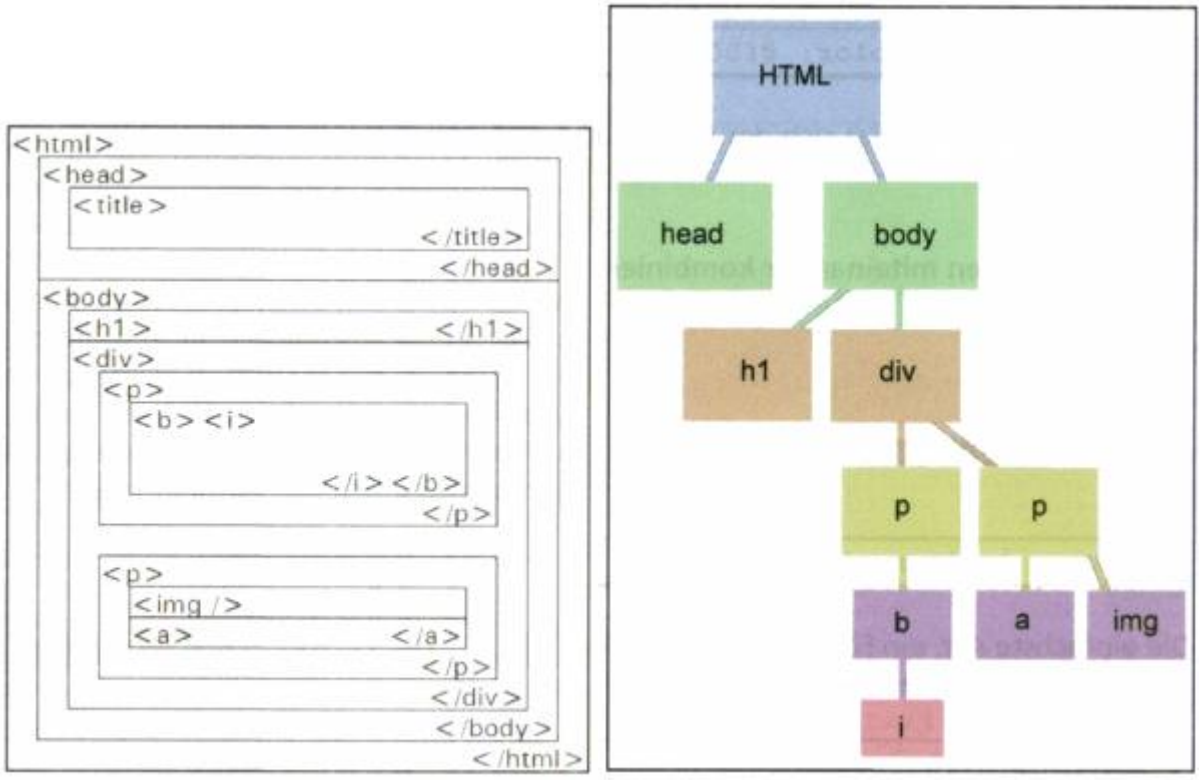
11.2.3.1) Vererbung

Vererbung heißt, dass Stileigenschaften von einem Element an ein beliebiges in ihm enthaltenes Element weitergegeben werden. Bei guter Planung kann die Vererbung Stilangaben sehr effizient machen.

Oder einfacher: Die Eigenschaften eines Elements werden von dem übergeordneten bestimmt, sofern sie nicht überschrieben werden.

Vererbung an untergeordnete Elemente

CSS-Eigenschaften werden teilweise an untergeordnete Elemente vererbt und teilweise nicht.



Eigenschaften wie die Schriftgröße, die Vorder- und Hintergrundfarben, font-weight, text-decoration, font-style, ... werden an die untergeordneten Elemente weitervererbt, d.h. von diesen übernommen, solange:

- das untergeordnete Element diese Eigenschaft nicht selbst für sich festlegt
- das untergeordnete Element für diese Eigenschaft keinen Standardwert vom Browser erhält (z.B. bei den h-Elementen die Größe und bei Links die Farbe und Unterstreichung)

Im 2. Fall kann man ein Element dazu zwingen, den Wert des übergeordneten Elements dennoch auf sich zu vererben und den Standardwert zu ignorieren, das geht indem man der Eigenschaft den Wert **inherit** zuweist. Ob dieser Wert zur verfügung steht, ist der CSS4You-Seite der einzelnen Eigenschaften zu übernehmen.

Beispiel: `font-size: inherit;`

Ein Beispiel für eine Eigenschaft, die nicht vererbt wird, ist der Rahmen (border) - was durchaus logisch und gut erscheint. Die Angabe, ob ein Element vererbbar ist oder nicht, findet man stets auf der jeweiligen Eigenschaftsinfoseite auf CSS4You.

kleines Anwendungsbeispiel:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>Anwendungsbeispiel Vererbung</title>

<style type="text/css">
/* <![CDATA[ */
div#box {
  font-size: 0.8em;
  color: white;
  background-color: black;
  font-family: Verdana; sans-serif;
}

a {
  color: inherit;
}
/* ]]> */
</style>
</head>
<body>

<div id="box">
<h1>Dieser Überschrift ist die font-size: Angabe des div-Elements egal, sie belässt es bei ihrer eigenen Größe, weil wir sie auch nicht mit inherit zur Vererbung zwingen.</h1>
<p>Dieser Absatz übernimmt die Eigenschaften des übergeordneten div-Elements anstandslos.</p>
<a href="#">Wie die Überschrift möchte dieser Link die Standardeigenschaft für color: (blau) gerne übernehmen. Mit inherit hindern wir ihn jedoch daran.</a>
<p style="background-color: darkred">Dieser Absatz legt die background-color-Eigenschaft selbst fest, darum wird sie auch nicht vererbt.</p>
<p>Alle gemeinsam übernehmen sie die Schriftart Verdana, weil kein Element diese selbst festlegen will!</p>
</div>
</body>
</html>
```

Relative Einheiten bei der Vererbung

Eine Besonderheit bei der Vererbung gibt es bei der Vererbung von relativen Größen. Dann bezieht sich nämlich die Größe des untergeordneten Elements relativ auf das Übergeordnete. Weißt man einem übergeordnetem Element also die Schriftgröße `font-size: 0.8em` zu, so bezieht sich eine Schriftgrößenangabe eines untergeordneten Elements auf diese `0.8em`.

Beispiel: Man weißt einem <p>-Element font-size: `0.8em` zu, einem untergeordnetem -Element ebenfalls `0.8em`. Dann ist die Schriftgröße des -Elements erneut um den Faktor 0.8 verringert.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>Beispiel Vererbung relativer Einheiten</title>

</head>
<body>

<p style="font-size: 0.8em;">Das ist ein Text im p-Element mit 0.8em<br />
   <span style="font-size: 0.8em;">Das ist ein Text im untergeordneten span-
Element, dem ebenfalls 0.8em Schriftgröße zugewiesen sind.</span>
</p>
</body>
</html>
```

Links

- CSS4You: Informationen über die Vererbung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_03:11_02_03_01

Last update: **2019/05/14 21:37**

11.2.3.2) Kaskadierung und Spezifität

Kaskadierung

Stoßen mehrere Stylesheets aufeinander (z.B. die Standardeigenschaften vom Browser für Elemente wie h1 und Links zum einen, ein externes Stylesheet der Website und zusätzlich noch eine Inline-Styleangabe) so stellt sich die Frage, welche Eigenschaften sich durchsetzen, wenn 2 oder mehrere Stylesheets das Aussehen desselben Elements bestimmen wollen. Dabei gilt, dass sich die Eigenschaften durchsetzen die am nächsten zum Element stehen.

Folgende Hierachie gibt es, umgesetzt wird immer die höchstmögliche Stufe in dieser Hierarchie:

- Default-Einstellungen des Browsers
- externe Style Sheets; werden mehrere externe Style Sheets mit einem Dokument verlinkt, haben im Konfliktfall die in der zuletzt eingelesenen CSS-Datei enthaltenen Anweisungen Vorrang;
- eingebettete (im Dokumentskopf definierte) Style Sheets;
- inline-Stilvorgaben.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Kaskadierung</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<style type="text/css">
/* <![CDATA[ */
h2 {
  font-weight: bold;
}

h3 {
  color: darkred;
}
/* ]]> */
</style>
</head>
<body>
<h1>Die wichtigste Überschrift, weil es keine weiteren Stilangaben gibt
werden die Standardeinstellungen des Browsers umgesetzt</h1>
<h2>Die zweitwichtigste Überschrift, hier gibt es eine interne Stilangabe,
diese wird auch verwendet.</h2>
<h3 style="font-color: black;">Die drittwichtigste Überschrift, hier gibt
zusätzlich zu einer internen Stilangabe auch eine Inline-Angabe, also setzt
sich diese durch.</h3>
<p class="wichtig">Beispielabsatz</p>
<p class="wichtig" style="color: white">Beispielabsatz</p>
```

```
</body>
</html>
```

Spezifität

Wie bereits bekannt, kann man mehrere (X)HTML-Elemente ineinander verschachteln und somit können durch die Vererbung einem Element verschiedene Werte für die gleiche CSS-Eigenschaft zugewiesen werden. Dabei gilt: Es setzt sich der Wert durch, der am spezifischten für das jeweilige Element ist.

Beispiel:

- Es gibt eine div-Box, die für font-size den Wert 12pt zugewiesen bekommt.
- Darin gibt es zwei p-Boxen (also Absätze),
    - eine davon weist font-size den Wert 10pt zu,
    - die andere setzt diese Eigenschaft nicht.
    - Dann gilt bei der ersten p-Box die spezifischte Eigenschaft, also 10pt.
    - Bei der zweiten p-Box gilt die allgemeine Eigenschaft der übergeordneten Box (12pt), weil es keine eigene Eigenschaft dazu gibt.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>TITEL</title>

<style type="text/css">
/* <![CDATA[ */
 div {
     font-size: 12pt;
 }
/* ]]> */
</style>
</head>
<body>
<div>
    <p style="font-size: 10pt;">Das ist der Absatz mit 10pt
Schriftgröße.</p>
    <p>Das ist der Absatz, der font-size nicht festlegt -> es gilt 12pt von
div</p>
</div>
</body>
</html>
```

Links

- CSS4You: Informationen über die Kaskadierung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_03:11_02_03_02

Last update: **2019/05/14 21:37**

11.2.4) XHTML-Elemente gestalten

- Schnellübersicht über alle CSS-Eigenschaften auf CSS4You.de (Mehr Infos mit Klick auf Eigenschaftsname)
- 11.2.4.1) Schriftformatierungen
- 11.2.4.2) Listen und Aufzählungen
- 11.2.4.3) Links mit Pseudoklassen gestalten
- 11.2.4.4) Boxen und Boxmodell
- 11.2.4.5) Farben, Hintergrundfarben und Hintergrundbilder
- 11.2.4.6) Tabellen
- 11.2.4.7) Positionierung von Elementen
- 11.2.4.8) Float- und Clear-Eigenschaft
- 11.2.4.9) Formulare
- 11.2.4.10) Übungen zur Elementgestaltung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04

Last update: **2019/05/14 21:39**

11.2.4.1) Schriftformatierungen

Mit CSS ist es ohne Probleme möglich, jede Art von Schrift- und Absatzformatierung umzusetzen.

Die Eigenschaften können grundsätzlich auf alle Elemente angewendet werden, die Text beinhalten, also auch auf Listen, Formulare, Tabellen, ...

Schriftart

Die Schriftart wird mit der Eigenschaft **font-family:** festgelegt, diese nimmt als Wert sowohl eindeutige Schriftarten (z.B. verdana) als auch Schriftfamilien (z.B. serif) an. Eine Schriftfamilie beinhaltet mehrere Schriftarten, von der der Browser die 1. beim Benutzer verfügbare auswählt. Man kann (und i.d.R. tut man das auch) auch mehrere Schriftarten bzw. -familien angeben. Der Browser nimmt dann standardmäßig die 1., wenn die Schriftart nicht installiert ist die 2., usw.

Beispiel: font-family: Verdana, Arial, sans-serif;

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<title>Generische Schriftarten</title>
</head>
<body>
<p style="font-family: sans-serif">sans-serif</p>
<p style="font-family: serif">serif</p>
<p style="font-family: cursive">cursive</p>
<p style="font-family: fantasy">fantasy</p>
<p style="font-family: monospace">monospace</p>
</body>
</html>
```

Eigene Schriftart mitliefern

Zuerst definitert man mit font-face eine eigene Schriftart, die man aus der Schriftdatei lädt. Diese Schriftart kann man dann anwenden.

```
@font-face {
  font-family: Raleway;    /* Dieser Name ist beliebig */
  src: url(Raleway-Thin.ttf);
}
```

```
<p style="font-family: Raleway">Raleway</p>   /* hier muss der Wert von
font-family bei der font-face-Deklarierung verwenet werden! */
```

Schriftfarbe

Die Schriftfarbe eines Elements wird (wie die Farbe aller Elemente) mit der CSS-Eigenschaften **color:** festgelegt.

Mögliche Werte sind wie bei HTML HEX-Codes (mit führender #) oder Farbnamen (green, darkgreen, ...)

Schriftgröße

Die Schriftgröße eines Elements wird mit der CSS-Eigenschaften **font-size:** festgelegt.

Diese Eigenschaft nimmt als Wert (wie alle Größen-/Längeneinheiten in CSS) verschiedene Einheiten an, z.B. px, em, cm, % → mehr Informationen zu diesen Einheiten.

Weitere CSS-Eigenschaften

Eigenschaft	wichtige Werte	Erklärung
font-style	italic, normal	Bestimmt die Darstellung der Schrift, im wesentlich kursiv oder normal
font-weight	bold, normal	Legt die „Schriftgewichtung“ fest
text-align	left, right, center, justify	Ausrichtung des Textes
text-decoration	underline, overline, line-through, blink, none	Ermöglicht Unter/Durch/Überstreichungen und blinkenden Text
text-shadow	Farbe, 3x Länge	Informationen s. hier
word-spacing	Längenangabe	Abstand zwischen den Wörtnern
letter-spacing	Längenangabe	Abstand zwischen den Buchstaben/Zeichen

Weitere Informationen auf CSS4You

Weitere Informationen zu diesen und weiteren CSS-Eigenschaften und den anwendbaren Werten gibt es unter http://www.css4you.de/Schrifteigenschaften/fontproperty.html (Schrifteigenschaften) und unter http://www.css4you.de/Texteigenschaften/textproperty.html (Texteigenschaften).

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_01

Last update: **2019/05/14 21:39**

11.2.4.2) Listen und Aufzählungen

Auch die bereits bekannten Listen kann man mit CSS einfach formatieren, häufig werden auch Menüs von Webseiten mit Listen gestaltet.

Zur Wiederholung eine ungeordnete Liste in (X)HTML:

```
<ul>
   <li>Ein Punkt</li>
   <li>Ein weiterer Punkt</li>
   <li>Ein weiterer Punkt</li>
</ul>
```

Listen werden in ul (ungeordnete Listen → Aufzählungszeichen) und ol (geordnete Listen → Zahlen oder Buchstaben zur Aufzählung) getrennt.

Wichtige CSS-Eigenschaften für Listen

Eigenschaft	erlaubte Werte	Erklärung	Anwendung möglich auf
list-style-type	none (ol+ul), disc, circle, square (bei ul), decimal, lower-alpha, upper-alpha, lower-roman, upper-roman, lower-greek, decimal-leading-zero, hebrew, armenian, georgian, cjk-ideographic, hiragana, katakana, hiragana-iroha, katakana-iroha (bei ol)	Bestimmt das Aufzählungszeichen (Standardmäßig disc - gefüllter Kreis)	ul, ol, li
list-style-image	none, url(), Inherit (vererbt vom übergeordneten Element)	Für die Verwendung einer Grafik anstatt eines Symbols von list-style-type	ul, ol, li
list-style-position	inside, outside	Bestimmt, ob Aufzählungszeichen innerhalb oder außerhalb des Textblocks stehen sollen (Standard: outside)	ul, ol, li

Weitere Informationen zu den einzelnen Eigenschaften gibt es unter http://www.css4you.de/listproperty.html.

Natürlichen können die „allgemeinen“ CSS-Eigenschaften für Text, Links, Bilder, Farben, etc. auch innerhalb von Listen angewendet werden.

Verschachtelung

Bekanntlicherweise können Listen verschachtelt werden. In X(HTML) sieht das folgendermaßen aus:

```
<ul>
   <li>Ein Punkt</li>
   <li>Ein weiterer Punkt
      <ul>
         <li>Unterpunkt
            <ul>
               <li>Unter-Unterpunkt</li>
            </ul>
         </li>
         <li>Unterpunkt</li>
      </ul>
   </li>
   <li>Ein weiterer Punkt</li>
</ul>
```

Zur Formatierung der Unterpunkte arbeitet man mit Nachfahrenselektoren, das sieht folgendermaßen aus (CSS-Code):

```
ul {
  list-style-type: disc;
}
ul ul
{
  list-style-type: circle;
}
ul ul ul
{
  list-style-type: square;
}
```

Würde man keine Nachfahrenselektoren deklarieren, würde das übergeordnete ul-Element die Eigenschaft an die untergeordneten vererben.

BSP: Menü mit Listen

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Listen inline</title>
<style type="text/css">
/* <![CDATA[ */
body {
   font-family: sans-serif;
   background-color: #FFF;
   color: #306;
}
ul {
```

```
   list-style-type: none;
   display: inline;
   padding-left: 0;
   margin-left: 0;
}
li {
   display: inline;
   padding: 3px 15px;
   border-left: 1px solid #000;
}

/* ]]> */
</style>
</head>
<body>

<ul>
  <li>Ein Punkt</li>
  <li>Ein Punkt</li>
  <li>Ein Punkt</li>
  <li>Ein Punkt</li>
</ul>

</body>
</html>
```

Links

CSS4You: Eigenschaften für Listen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_02

Last update: **2019/05/14 21:39**

11.2.4.3) Links mit Pseudoklassen gestalten

Für Links gibt es verschiedene Zustände. Wurde ein Link vom Besucher bereits aufgerufen, so hat er den Zustand „visited“, wird ein Link mit der Maus berührt, hat er den Zustand „hover“, usw. - Mit CSS kann man für jeden dieser Zustände bequem eine eigene Formatierung umsetzen, wie es auch bei vielen Websites gemacht wird.

Bei Pseudoelementen und Pseudoklassen notiert man zuerst das betroffene HTML-Element, im Beispiel das a-Element für Verweise. Dahinter folgt ein Doppelpunkt und dahinter eine erlaubte Angabe, im Beispiel etwa `link` (für noch nicht besuchte Verweisziele), `visited` (für bereits besuchte Verweisziele), `hover` (für Verweise, während der Anwender mit der Maus darüber fährt), `active` (für angeklickte Verweise) und `focus` (für z.B. mit der Tastatur ausgewählte Verweise).

Wichtig ist, dass dies keine frei wählbaren Namen sind, sondern feste Schlüsselwörter.

Zustandsname	**Erklärung**
link	Standard: Der Link wurde noch nicht aufgerufen, wird auch nicht mit der Maus berührt oder ähnliches
visited	Der Link wurde vom Besucher bereits aufgerufen, er ist im Verlauf des verwendeten Browsers
hover	Der Link wird mit der Maus gerade berührt
focus	Der Link hat aktuell den Fokus
active	Der Link wird gerade angeklickt, die Maustaste wurde aber noch nicht losgelassen

Diesen Zuständen kann mit Pseudoklassen eine eigene Formatierung zugeordnet werden. Dabei wird zur Auswahl in CSS einfach, wie bisher, das Element (und evtl. die Klasse/ID) angegeben, ergänzt mit **:zustand**

Praxisbeispiel

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Links</title>

<style type="text/css">
/* <![CDATA[ */
body {background-color: #333; }
a:link { color: red; }
a:visited { color: green; }
a:hover { color: blue; }
a:focus { color: white; }
a:active { color: yellow; }
/* ]]> */
</style>
</head>
```

```
<body>
<a href="#">Ein bedeutsamer Link</a> <br />
<a href="#">Ein bedeutsamer Link</a> <br />

</body>
</html>
```

Links

Pseudoelemente und -klassen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

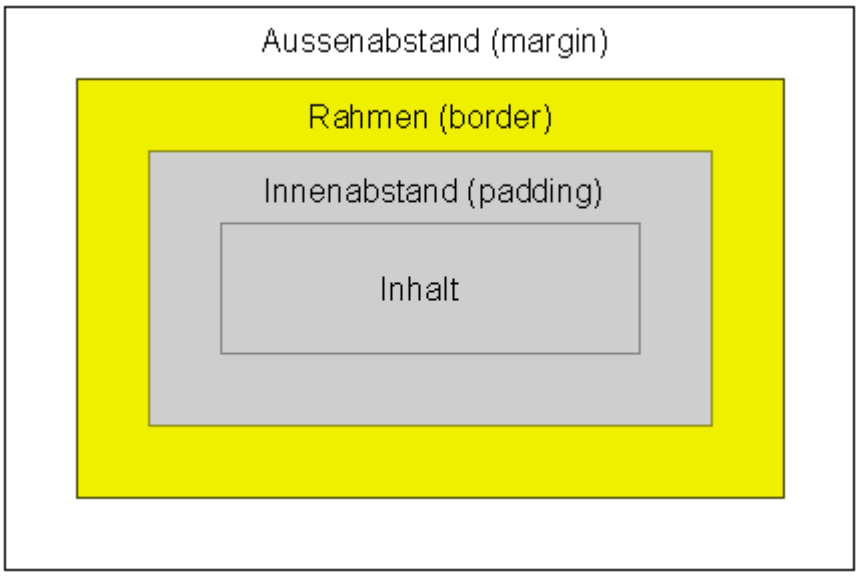
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_03

Last update: **2019/05/14 21:39**

11.2.4.4) Boxen und Boxmodell

Wie bereits erwähnt unterscheidet man Block-Elemente und Inline-Elemente. Block-Elemente verwenden zur Darstellung das Boxmodell. Eine Box (also ein Block-Element) hat immer einen Innenabstand (padding), der zwischen Inhalt und dem Rahmen ist, einen Rahmen (border) und einen Außenabstand (margin), der den Abstand rund um den Rahmen angibt.

Grafische Darstellung des Boxmodells:



(© css4you.de)

CSS-Eigenschaften

Eigenschaft	Wert	Erklärung
padding	Größenangabe	Innenabstand
margin	Größenangabe	Außenabstand
border-color	HEX-Code oder Farbbezeichnung	Farbe des Rahmens
border-width	Größenangabe	Breite des Rahmens
border-style	none, dotted, dashed, solid, double, groove, ridge, inset, outset	Style des Borders

Es ist auch möglich, eine Rahmen- oder Abstandseigenschaft nur einer Seite der Box zuzuweisen; weitere Informationen dazu und allgemein zu den Rahmen-Eigenschaften gibt es wie immer auf CSS4You: Eigenschaften für Rahmen, Eigenschaften für Innenabstand, Eigenschaften für Außenabstand

Umfangreiche Informationen zu Boxen und dem Boxmodell findet man unter http://www.css4you.de/wsboxmodell/index.html.

Beispiel

Die DIV-Box hat einen Innenabstand, Außenabstand und einen Rahmen. Innerhalb dieser DIV-Box gibt

es eine weitere DIV-Box die den Text beinhaltet. Die äußere DIV-Box muss ein Block-Element sein, die innere hätte auch ein Inline-Element sein können (z.B. span).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Boxmodell</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<style type="text/css">
/* <![CDATA[ */
body {
   background-color:#FFF5E6;
   color: #995C00;
   margin: 0;
   padding: 0;
   font: 100.01% sans-serif;
}
#box {
   width:20em;
   background-color:#FFC266;
   border: 3em solid #FF9900;
   padding: 4em;
   margin: 2em;
}
#inhalt {
   background-color:#FFF5E6;
   font-size: 3em;
}
/* ]]> */
</style>
</head>
<body>
<div id="box">
  <div id="inhalt">Inhaltsbereich</div>
</div>

</body>
</html>
```

Links

- CSS4You: Informationen über das Boxmodell
- CSS4You: Eigenschaften für Rahmen
- CSS4You: Eigenschaften für Außenabstand
- CSS4You: Eigenschaften für Innenabstand

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_04

Last update: **2019/05/14 21:40**

11.2.4.5) Farben, Hintergrundfarben und Hintergrundbilder

Farben

Jedem Element kann eine Farbe zugewiesen werden. Hier ist damit die Vordergrundfarbe, also z.B. die Farbe vom Text selbst gemeint. Die zuständige Eigenschaft ist **color:** und sie nimmt Farben im HEX-Format oder als Wort an.

Hintergrundfarben

Man kann mit CSS Hintergrundfarben für alle (X)HTML-Elemente vergeben, also sowohl eine Farbe für die gesamte Seite als auch nur für einen bestimmten Text, eine Tabellenzeile, etc.

Die Eigenschaft für die Hintergrundfarbe ist dabei immer **background-color:**, als Wert wird eine Farbe entweder im HEX-Format oder als Wort (green, ...) angenommen.

(X)HTML-Beispielcode anzeigen

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Hintergrundfarbe</title>
<link rel="stylesheet" type="text/css" href="style.css" />
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
<h1>Hintergrundfarbe</h1>
<p id="text">Das ist ein Absatz mit Text. Er hat eine eigene
Hintergrundfarbe. Das ist ein Absatz mit Text. Er hat eine eigene
Hintergrundfarbe. Das ist ein Absatz mit Text. Er hat eine eigene
Hintergrundfarbe. Das ist ein Absatz mit Text. Er hat eine eigene
Hintergrundfarbe. Das ist ein Absatz mit Text. Er hat eine eigene
Hintergrundfarbe. Das ist ein Absatz mit Text. Er hat eine eigene
Hintergrundfarbe.</p>

</body>
</html>
```

Dazu gehörenden CSS-Beispielcode anzeigen

```
body {
   background-color: #FFFF99;
}

h1 {
   font-size: 1.7em;
}

p#text {
   background-color: darkgreen;
   color: white;
}
```

Hintergrundbilder

Ebenso können für alle Elemente Hintergrundbilder vergeben werden. Die Eigenschaft dazu lautet **background-image:**, als Wert wird ein Link zu einem Bild angenommen: **url(./images/hintergrundbild.png)**

Zwei weitere wichtige Eigenschaften für Hintergrundbilder gibt es:

- **background-repeat:** Legt fest, ob und in welche Richtung das Hintergrundbild wiederholt werden soll. Mögliche Werte sind no-repeat, repeat (in alle Richtungen wiederholen), repeat-x (horizontal wiederholen), repeat-y (vertical wiederholen). Am sinnvollsten ist hier bei einem von oben nach unten gehenden Farbverlauf repeat-x; bei einem von links nach rechts gehendem Farbverlauf repeat-y; bei einem Hintergrundbild mit einem Muster repeat.
- **background-attachment:** Damit bestimmt man, ob das Hintergrundbild mit der Seite mitscrollen soll (Wert scroll) oder fixiert an seinem Platz stehen soll (Wert fixed)
- **backgtround-position:** Legt die Position des Hintergrundbildes fest, je nach der background-repeat Angabe. Genauere Informationen hier

Es gibt noch weitere Eigenschaften, um Hintergrundbilder noch genauer steuern zu können, diese sind unter http://www.css4you.de/backgroundproperty.html zu finden.

Hinweis: Einen Farbverlauf kann man sich entweder mit Bildbearbeitungprogrammen wie Photoshop oder mit Farbverlaufs-Generatoren im Internet (z.B. http://www.generateit.net/gradient) erstellen.

Beispiel:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>TITEL</title>

<style type="text/css">
/* <![CDATA[ */
```

```
body {
 background-image: url(farbverlauf.png);  /* alternative: muster */
 background-color: white;
 background-repeat: repeat-x;  /* alternativen: repeat, repeat-y, no-repeat
*/
 /*background-attachment: fixed  */  /* auskommentieren um Wirkung zu sehen!
*/
        /*background-position: 15px 25px */  /* auskommentieren um Wirkung
zu sehen! */
}

div#beispiel_div {
 height: 900px;
 background-color: white;
 border: black solid 3px;#
 border-radius: 10px;
 width: 400px;
 margin: 100px;
}

/* ]]> */
</style>
</head>
<body>
<div id="beispiel_div">Das ist der Inhalt der DIV-Box</div>
</body>
</html>
```

Farbverlaufs-PNG:

Links

CSS4You: Eigenschaften zu Hintergrundbildern

CSS4You: Verfügbare Farbnamen

CSS4You: Übersicht von websicheren Farben (in HEX-Codes)

CSS4You: Systemfarben

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_05

Last update: **2019/05/14 21:40**

11.2.4.6) Tabellen

Auch Tabellen können mithilfe von CSS grafisch aufgehübscht werden. Dazu werden die allgemeinen Eigenschaften, wie color, border, etc. auf Elemente der Tabelle, also <table>, <tr>, <td>, ... angewendet. Mit (X)HTML werden verschiedene neue Tags für Tabellen eingeführt, praktisch werden diese jedoch kaum genutzt. Deswegen beschränken wir uns auf den <caption>-Tag, der der Tabelle einen Titel gibt.

Beispiel

index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<title>Tabelle</title>
<link rel="stylesheet" type="text/css" href="style.css" />
</head>

<body>
<table cellspacing="0">
 <caption>Tabelle XY</caption>
  <tr><th class="ohne">&Uuml;bersicht</th> <th>Spalte 1</th> <th>Spalte
2</th></tr>
  <tr><th>Zeile 1</th><td>Erste Zelle</td><td>Zweite Zelle</td></tr>
  <tr class="gerade"><th>Zeile 2</th><td>Dritte Zelle</td><td>Vierte
Zelle</td></tr>
  <tr><th>Zeile 3</th><td>F&uuml;nfte Zelle</td><td>Sechste Zelle</td></tr>
  <tr class="gerade"><th>Zeile 4</th><td>Siebte Zelle</td><td>Achte
Zelle</td></tr>
  <tr><th>Zeile 5</th><td>Neunte Zelle</td><td>Zehnte Zelle</td></tr>
  <tr class="gerade"><th>Zeile 6</th><td>Elfte Zelle</td><td>Zw&ouml;lfte
Zelle</td></tr>
  <tr><th>Ergebnis</th><td>Erste Spalte</td><td>Zweite Spalte</td></tr>
</table>



</body>
</html>
```

style.css

```
body {
            background-color: #CFDCE6;
            font: 100.01% Verdana, sans-serif;
            color: #706348;
}
table {
            border-collapse: collapse;
}

td, th {
            padding: 10px;
            width: 15em;
            border: 1px solid #807459;
            font: 0.8em sans-serif;
}
th {
            font-weight: bold;
            letter-spacing: 0.5em;
}

caption {
            text-transform: uppercase;
            font-size: 1.5em;
            margin-bottom: 10px;
}


tr.gerade {
            background-color: #E6DFCF;
            color: #647D8F;
}
tr {
            background-color: white;
}
tr th {
            background: url(hg_table2.gif) white no-repeat 3px 3px;
}
tr.gerade th {
            background: url(hg_table1.gif) #E6DFCF no-repeat 3px 3px;
}
```

Grafiken



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_06

Last update: **2019/05/14 21:40**

11.2.4.7) Positionieren von Elementen

Es gibt verschiedene Möglichkeiten, Elemente per CSS an eine vorbestimmte Position zu setzen.

Statische Positionierung

Die statische Positionierung ist die häufigst-angwendete, und darum auch die Standardpositionierungsart. Die Elemente werden damit im normalen Textfluss dargestellt, wie bisher bei allen Beispielen.

Die CSS-Regel für die statische Positionierung ist **position: static** - weil es aber sowieso die Standardeinstellung ist, muss diese nicht angegeben werden.

(X)HTML/CSS-Beispielcode anzeigen

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>TITEL</title>

<style type="text/css">
/* <![CDATA[ */
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>statische Positionierung</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<style type="text/css">
/* <![CDATA[ */
div {
     width: 12em;
     color: #0000CC;
     border: dotted 1px;
     padding: 1em;
}
.box1 {
     background-color: #FFCC00;
     position:static;
}
.box2 {
     background-color: #FFFF99;
     position:static;
```

```
}
/* ]]> */
</style>
</head>
<body>
<div class="box1">Erste Box</div>
<div class="box2">Zweite Box</div>
</body>
</html>
```

Absolute Positionierung

Bei der absoluen Positionierung werden die Element aus dem Textfluss genommen, und an eine festgelegte Stelle gerückt. Dadurch können sich auch mehrere Elemente überlagern.

Angegeben wird wieder Positionierungstyp mit **position: absolute**, während die Position mit den Eigenschaften **left: ..**, **top: ..**, **right: ..** und **bottom: ..** festgelegt wird. Die Werte für diese Eigenschaften können beliebige Längenangaben mit allen bekannten Einheiten sein. Sie beziehen sich, wenn es kein übergeordnetes Element gibt, das selbst mit position: absolute; positioniert wird, auf das Browserfenster - ansonsten auf das übergeordnete Element (aber eben nur unter der **Bedingung**, das dieses übergeordnete Element selbst absolut positioniert ist!). Meist gibt man für left, top, right und bottom einen Wert mit der Einheit em an, weil sich das Design somit an die Schriftgröße es Benutzers anpasst.

- Die Eigenschaften stehen immer für Abstände, also top: legt den Abstand vom obersten Rand des Browserfensters bzw. des übergeordneten Elements fest.
- Man verwendet immer ENTWEDER top: .. ODER bottom: .. bzw. left: .. ODER right: ..

Beispiel 1: 2 DIVs ohne übergeordnetes Element

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Positionierung</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<style type="text/css">
/* <![CDATA[ */
div {
      width: 12em;
      color: #00C;
      border: dotted 1px;
      padding: 1em;
}
.box1 {
      background-color: #FC0;
      position:static;
}
.box2 {
```

```
      background-color: #FF9;
      position: absolute;
      left: 4em;
      top: 2em;
}
/* ]]> */
</style>
</head>
<body>

<div class="box1">Erste Box</div>
<div class="box2">Zweite Box</div>

</body>
</html>
```

Beispiel 2: 2 DIVs ohne übergeordnetes Element, 1 DIV das einem der beiden DIVs untergeordnet ist

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Positionierung</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<style type="text/css">
/* <![CDATA[ */
div   {
        width: 12em;
        color: #00C;
        border: dotted 1px;
        padding: 1em;
}
.box1 {
        background-color: #FC0;
        position:static;
}
.box2 {
        background-color: #FF9;
        position: absolute;
        left: 4em;
        top: 2em;
}
.box3 {
        background-color: #CCF;
        position: absolute;
        left: 4em;
        top: 2em;
}

/* ]]> */
```

```
</style>
</head>
<body>
<div class="box1">Erste Box
</div>
<div class="box2">Zweite Box
          <div class="box3">Dritte Box</div>
</div>


</body>
</html>
```

An Beispiel 2 sieht man: Wäre der Bezugspunkt immer das Browserfenster, und nicht das übergeordnete Element, müsste die 3. Box an der gleichen Position wie die 2. Box sein.

Beispiel 3: gleich wie BSP2, jedoch wird die übergeordnete Box nicht mehr mit position: absolute positioniert

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Positionierung</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<style type="text/css">
/* <![CDATA[ */

div   {
        width: 12em;
        color: #00C;
        border: dotted 1px;
        padding: 1em;
}
.box1 {
        background-color: #FC0;
        position: static;
}
.box2 {
        background-color: #FF9;
}
.box3 {
        background-color: #CCF;
        position: absolute;
        left: 4em;
        top: 2em;
}

/* ]]> */
</style>
```

```
</head>
<body>
<div class="box1">Erste Box
</div>
<div class="box2">Zweite Box
          <div class="box3">Dritte Box</div>
</div>


</body>
</html>
```

Beispiel 3 zeigt die Bedingung für die Beziehung zwischen übergeordnetem und untergeordnetem Element.

Relative Positionierung

Bei der relativen Positionierung wird ein Element relativ zu ihrem eigentlichen (unsprünglichen) Platz (im Textfluss) positoniert. Dabei verändert sich die Position der umgebenden Elemente nicht.

(X)HTML/CSS-Beispielcode anzeigen

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Positionierung</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<style type="text/css">
/* <![CDATA[ */

div   {
        width: 12em;
        color: #00C;
        border: dotted 1px;
        padding: 1em;
}
.box1 {
        background-color: #FC0;
        position: static;
}
.box2 {
        background-color: #FF9;
        position: relative;
        left: 4em;
        top: 2em;
}
```

```
/* ]]> */
</style>
</head>
<body>
<div class="box1">Erste Box</div>
<div class="box2">Zweite Box</div>
<div class="box3">Dritte Box</div>



</body>
</html>
```

Fixierte Positionieurng

Die fixierte Positionierung funktioniert wie die absolute, mit 2 Unterschieden:

- Die Positionsangaben beziehen sich IMMER auf das Browserfenster, nicht auf eventuelle übergeordnete Elemente
- Das Element scrollt mit den übrigen Elementen nicht mit, es bleibt immer an dieser einen Stelle.

(X)HTML/CSS-Beispielcode anzeigen

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Positionierung</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<style type="text/css">
/* <![CDATA[ */
body {
        font: 100.01% sans-serif;
        color: #00C;
        background-color: #fff;
}
div  {
        width: 12em;
        border: dotted 1px;
        padding:1em;
}
.box1 {
        background-color: #FFCC00;
        position: static;
}
.box2  {
        background-color: #FFFF99;
        position: fixed;
```

```
        left: 4em;
        top: 2em;
}
/* ]]> */
</style>
</head>
<body>
<div class="box1">Erste Box</div>
<div class="box2">Zweite Box</div>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
</body>
</html>
```

Überlappungen (z-Achse) und Durchsichtigkeit

Überlappungen

Wie bereits gesehen kann es bei manchen Positionierungstypen zu Überlappungen kommen. Will man festlegen, welche(s) Element(e) dabei vorne liegen, macht man das mit der Eigenschaft **z-index:** - je höher der Wert dieser Eigenschaft, desto weiter vorne erscheint das Element. Der Wert muss in keiner bestimmten Folge angegeben werden, sondern es können beispielsweise auch die Werte 5, 16, 33, 67 vorkommen. z-index funktioniert nur bei absolut positionierten Elementen.

(X)HTML/CSS-Beispielcode anzeigen

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
```

```
<title>z-index</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<style type="text/css">
/* <![CDATA[ */
div  {
       width: 12em;
       color: #00C;
       border: dotted 1px;
       padding: 1em;
}
.box1 {
       background-color: #FC0;
       position: absolute;
       left: 0;
       top: 1em;
       z-index: 3;
}
.box2 {
       background-color: #FF9;
       position: absolute;
       left: 4em;
       top: 2em;
       z-index: 1;
}
.box3 {
       background-color: #FFF;
       position: absolute;
       left: 6em;
       top: 3em;
       z-index: 2;
}
/* ]]> */
</style>
</head>
<body>
<div class="box1">Erste Box</div>
<div class="box2">Zweite Box</div>
<div class="box3">Dritte Box</div>
</body>
</html>
```

Durchsichtigkeit (Opacity)

Ebenfalls interessant ist die Eigenschaft **opacity:**, diese gibt den Durchsichtigkeitsgrad eines Elements an. Leider wird diese derzeit noch von vielen Browser unterschiedlich implementiert. Als Wert für opacity: gibt man einen Wert zwischen 0 und 1 an.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_07

Last update: **2019/05/14 21:40**

11.2.4.8) Float- und Clear-Eigenschaft

Neben den verschiedenen Positionierungsarten gibt es eine weitere Möglichkeit, die Darstellung von Webseiten zu verändern.

Float: Umfließen

Float setzt die für Block-Elemente geltende Regel, dass diese in einer neuen Zeile beginnen außer Kraft. Es bewirkt also, dass keine Veränderung des normalen Textflusses stattfindet.

Das geschieht, indem das betroffene Element von den anderen umflossen wird. Dem Element, das umflossen werden soll, weißt man die Regel float: left (für links umfließen) oder float: right (für rechts umfließen) zu. Der Standardwert ist float: none, also nicht umfließen.

Folgendes **Beispiel** zeigt ein Bild, das von einem Text im Absatz umflossen werden soll:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Float</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<style type="text/css">
/* <![CDATA[ */
* {
   padding: 0;
   margin: 0;
}
body {
   font: 100.01% sans-serif;
}
div {
   background-color: #EBE6E0;
   margin: 10px;
}
.rechts {
   float: right;
}
.links {
   float: left;
}



/* ]]> */
</style>
```

```
</head>
<body>
<div><img src="bild.jpg" width="100" height="75" alt="Katze" class="links"
/><p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo
consequat.</p></div>
<hr />
<div><img src="bild.jpg" width="100" height="75" alt="Katze" class="rechts"
/><p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo
consequat.</p></div>

</body>
</html>
```

Bild: 

Clear: Umfließen beenden

Das Problem ist nun, dass das Bild nicht nur von dem einen Absatz umflossen wird, sondern von allen folgenden Elementen. Das Umfließen muss also nach dem gewünschten Element wieder mit der Regel clear: left, clear: right oder clear: both stoppen. Die clear-Regel muss dem Element zugewiesen werden, die das umflossene Element als letztes umfließen soll.

Nun das gleiche **Beispiel** wie oben, nur mit der Anwendung von clear – Es muss nur der CSS-Code geändert werden:

```
<style type="text/css">
/* <![CDATA[ */
* {
   padding: 0;
   margin: 0;
}
body {
   font: 100.01% sans-serif;
}
.links {
   float: left;
   padding-right: 20px;
}
.rechts {
   float: right;
```

```
    padding-left: 20px;
}
div {
    background-color: #EBE6E0;
    margin: 10px;
}
hr {
    clear: left;
}
/* ]]> */
</style>
```

Das Problem mit den unterschiedlichen Höhen

Wie man im BSP sieht, gibt es noch ein Problem: Der Absatz wurde durch den DIV-Block grau hinterlegt, allerdings nicht so hoch wie das Bild, obwohl es im selben DIV liegt. Um das Problem zu beseitigen gibt es einen einfachen Trick: overflow: hidden – dem DIV muss man dazu jedoch eine fixe Breite zuweisen.

Beispiel vollendet - Es muss nur der CSS-Code geändert werden:

```
<style type="text/css">
/* <![CDATA[ */
* {
    padding: 0;
    margin: 0;
}
body {
    font: 100.01% sans-serif;
}
.links {
    float: left;
    padding-right: 20px;
}
.rechts {
    float: right;
    padding-left: 20px;
}
div {
    background-color: #EBE6E0;
    margin: 10px;
    overflow: hidden;
    width: 900px;
}
hr {
    clear: left;
}
/* ]]> */
</style>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_08

Last update: **2019/05/14 21:41**

11.2.4.9) Formulare

XHTML-Tags für Formulare

Formulare haben mit XHTML einige neue Tags bekommen, um sie genauer bzw. bequemer formatieren und ansteuern zu können.

Tag	wichtige Attribute	Bedeutung
<form>...</form>	action, method	Innerhalb der Tags befindet sich das Formular.
		action: Zieldatei des Formulars beim Absenden, **method:** Methode der Datenübermittlung (POST oder GET)
<fieldset>...</fieldset>	-	Fasst verschiedene Felder eines Formulars zu einem Block zusammen
<legend>...</legend>	-	Gibt dem <fieldset> eine Bezeichnung, die angezeigt wird
<label>...</label>	for	Darin wird zur besseren Verwendung, u.a. mit CSS, die für den Nutzer sichtbare Bezeichnung des Feldes geschrieben.
		for: Muss dem Attributwert „id“ des input-Felds entsprechen
<input />	type, name, id, size	Eingabeelement
		type: Feldtyp (z.B. text), **name:** eindeutige Bezeichnung für PHP, **id:** eindeutige Bezeichung für HTML/CSS/JS, **size:** Größe des Feldes

Gestaltung mit CSS

Durch die Verwendung der oben genannten Tags und ihrer Attribute hat man mit CSS sehr viele Möglichkeiten, Formulare zu gestalten. Es können alle allgemeinen, also nicht-spezifischen (wie z.B. list-style-type - außer natürlich es werden Listen im Formular verwendet) CSS-Eigenschaften angewandt werden.

Anzeigebeispiel - So sieht's aus: formular.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<title>legend und fieldset</title>
<link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>

<form action="#" method="post">
<fieldset><legend>Absenderangaben</legend>
<label for="vorname" class="text">Vorname:</label>
```

```
<input type="text" name="vorname" id="vorname" size="20" maxlength="40"
/><br />
<label for="nachname" class="text">Nachname:</label>
<input type="text" name="nachname" id="nachname" size="20" maxlength="40" />
</fieldset>
</form>

</body>
</html>
```

```
body {
    font-family: Verdana, Helvetica, sans-serif;
    font-size: 0.8em;
}


fieldset {
    border: 2px solid green;
    background-color: lightgreen;
    line-height: 1.8em;
    width: 22em;
    -moz-border-radius: 5px;
    -webkit-border-radius: 5px;
    -khtml-border-radius: 5px;
    border-radius: 5px;
}

legend {
    border: 1px solid black;
    font-weight: bold;
    background-color: darkgreen;
    color: white;
    line-height: 1.4em;
    -moz-border-radius: 3.5px;
    -webkit-border-radius: 3.5px;
    -khtml-border-radius: 3.5px;
    border-radius: 3.5px;
}

label {
    float: left;
    text-align: right;
    width: 7em;
    margin-right: 1em;
    white-space: pre;
}

br {
    clear: left;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_09

Last update: **2019/05/14 21:41**

11.2.4.10) Übungen zur Elementgestaltung

- 11.2.4.10.1) Übung Schriften
- 11.2.4.10.2) Übung Boxmodell
- 11.2.4.10.3) Übung Formulare
- 11.2.4.10.4) Tabellen
- 11.2.4.10.5) Positionierung
- 11.2.4.10.6) Übung float und clear

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_10

Last update: **2019/05/14 21:43**

11.2.4.10.1) Übung Schriften

Gegeben ist das Gedicht: der Erlkönig.

- Formatiere die erste Strophe in sans-serif, die zweite Strophe in serif, die dritte Strophe in cursive, die vierte Strophe in fantasy, die fünfte Strophe in monospace.
- Verwende für jede einzelne Strophe eine Farbe.
- Nun verwende 4 verschiedene Eigenschaften(deiner Wahl) für das Gedicht.

Wer reitet so spät durch Nacht und Wind?
Es ist der Vater mit seinem Kind.
Er hat den Knaben wohl in dem Arm,
Er faßt ihn sicher, er hält ihn warm.

Mein Sohn, was birgst du so bang dein Gesicht?
Siehst Vater, du den Erlkönig nicht!
Den Erlenkönig mit Kron' und Schweif?
Mein Sohn, es ist ein Nebelstreif.

Du liebes Kind, komm geh' mit mir!
Gar schöne Spiele, spiel ich mit dir,
Manch bunte Blumen sind an dem Strand,
Meine Mutter hat manch gülden Gewand.

Mein Vater, mein Vater, und hörest du nicht,
Was Erlenkönig mir leise verspricht?
Sei ruhig, bleibe ruhig, mein Kind,
In dürren Blättern säuselt der Wind.

Willst feiner Knabe du mit mir geh'n?
Meine Töchter sollen dich warten schön,
Meine Töchter führen den nächtlichen Reihn
Und wiegen und tanzen und singen dich ein

Wer reitet so spät durch Nacht und Wind?
Es ist der Vater mit seinem Kind.
Er hat den Knaben wohl in dem Arm,
Er faßt ihn sicher, er hält ihn warm.

Mein Sohn, was birgst du so bang dein Gesicht?
Siehst Vater, du den Erlkönig nicht!
Den Erlenkönig mit Kron' und Schweif?
Mein Sohn, es ist ein Nebelstreif.

Du liebes Kind, komm geh' mit mir!
Gar schöne Spiele, spiel ich mit dir,
Manch bunte Blumen sind an dem Strand,
Meine Mutter hat manch gülden Gewand.

Mein Vater, mein Vater, und hörest du nicht,
Was Erlenkönig mir leise verspricht?
Sei ruhig, bleibe ruhig, mein Kind,
In dürren Blättern säuselt der Wind.

Willst feiner Knabe du mit mir geh'n?
Meine Töchter sollen dich warten schön,
Meine Töchter führen den nächtlichen Reihn
Und wiegen und tanzen und singen dich ein

Vorschau:

Lösung:

```
.absatz1{font-family:sans-serif; color:#00FF00; font-weight:bold;}
.absatz2{font-family:serif; color:#0000FF; text-decoration:underline;}
.absatz3{font-family:cursive; color:#00FFFF; text-decoration:overline;}
.absatz4{font-family:fantasy; color:#008000;}
.absatz5{font-family:monospace;}


.text-shadow {text-shadow: #000000 3px 3px 5px;}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_10:11_02_04_10_01

Last update: **2019/05/14 21:43**

11.2.4.10.2) Übung Boxmodell



Erstelle eine Box mit den folgenden Eigenschaften:
-Länge: 1230px
-Höhe: 100px
-Hintergrundfarbe: hellblau
-Rahmen: Breite 3px / solide / rot gefärbt
-Textabstand zum Rahmen 20px
-Inhalt der Box:
„Gallia est omnis divisa in partes tres, quarum unam incolunt Belgae, aliam Aquitani, tertiam qui ipsorum lingua Celtae, nostra Galli appellantur. Hi omnes lingua, institutis, legibus inter se differunt. Gallos ab Aquitanis Garumna flumen, a Belgis Matrona et Sequana dividit.“
Hintergrundbild der Website:



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_10:11_02_04_10_02

Last update: **2019/05/14 21:43**

11.2.4.10.3) Übung Formulare

Erstelle ein Formular mittels fieldset nach folgendem Muster:

(Bild folgt)

Folgende Formatierungen sollen umgesetzt werden:

- Formular:
  - Hintergrundfarbe: #eeeeff;
  - gepunkteter Rahmen: 1 Pixel, Abstand: 20 Pixel, in Grün
- fieldset:
  - 4 Pixel starker durchgezogener Rahmen in Silber

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_10:11_02_04_10_03

Last update: **2019/05/14 21:45**

11.2.4.10.4) Tabellenübung

Erstelle eine Tabelle, welche wie folgt aussehen soll:



Parameter

- Die Überschrift soll im <table>-Tag mit einer Caption erstellt werden.
- Die Schriftfarbe und das permanente Großschreiben (CSS-Tag: text-transform) in der Überschrift soll in einem externen Stylesheet erledigt werden.
- Die Einträge „1. Einheiten“ und „2. Einheiten“ sollen mittels table-header (th) erledigt werden. Die Formatierung dafür soll im Stylesheet passieren. (Schriftfarbe, Buchstabenabstand, ...)
- Backgrounds für die „Zeile x.“ und „Eintrag x.“ sollen im CSS-Stylesheet mittels Gruppen, die im HTML-Code einzufügen sind, definiert werden. (.bgblue und .bgred)
- Die Abstände der Einheiten vom Tabellenrand können mit dem CSS-Tag „padding: ??px“ eingestellt werden.

Mögliche Realisierung der Übung

Der HTML-Code ist hier einzusehen:

```
<html>

<head>
<title>&Uuml;bung - Tabellen</title>
<meta name="author" content="user">
<link rel="stylesheet" type="text/css" href="03_tabellen.css">
</head>

<body text="#000000" bgcolor="#FFFFFF" link="#FF0000" alink="#FF0000"
vlink="#FF0000">
<table border="1">
  <caption>&Uuml;bungstabelle</caption>
    <tr><th>1. Einheiten</th>
      <td class="bgblue">Zeile 1</td>
      <td class="bgred">Eintrag 1</td>
    </tr>
    <tr><th>2. Einheiten</th>
      <td class="bgblue">Zeile 2</td>
      <td class="bgred">Eintrag 2</td>
```

```
    </tr>
</table>
</body>

</html>
```

Nützliche Links

Tabellenformatierung mittels CSS in der Wiki erklärt

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_10:11_02_04_10_04

Last update: **2019/05/14 21:45**

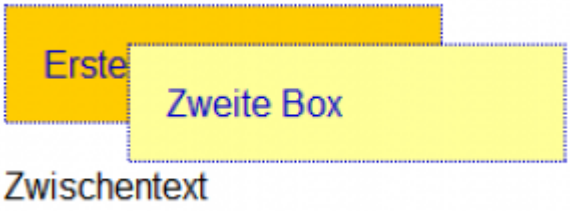
11.2.4.10.5) Positionierungsübung

Erstelle zwei Boxen, welche sich folgendermaßen reihen:



(1.Version)

Danach ändere die Datei so, dass sich die Elemente so überlappen:



(2. Version)

Parameter

- Die Boxen sollen im CSS-Stylesheet für den HTML-Tag definiert werden. (padding, border, width, color, ...)
- Für diese Boxen sollen zwei Klassen (.box1 und .box2) erstellt werden, in denen die Hintergrundfarbe und die Positionierung definiert werden.
- Die zweite Box soll bei der zweiten Version mittels „position:absolute;“ und den Einzügen „top: 2em;“ und „left: 4em;“ über die andere gelegt werden.

Mögliche Realisierung der Übung

Der HTML-Code mit CSS-Stylesheet ist hier einzusehen:

```
<html>

<head>
<title>Positionierungsübung</title>
<meta name="author" content="user">
<body text="#000000" bgcolor="#FFFFFF" link="#FF0000" alink="#FF0000"
vlink="#FF0000">
<style type="text/css">
```

```
body {
        font-family: Arial;
     }

div {
     width: 12em;
     color: #0000CC;
     border: dotted 1px;
     padding: 1em;
}

.box1 {
     background-color: #FFCC00;
     position:static;
}

.box2 {
     background-color: #FFFF99;
     position: absolute;
     left: 4em;
     top: 2em;
}
</style>
</head>

<body>
<div class="box1">Erste Box</div>
<br>Zwischentext<br><br>
<div class="box2">Zweite Box</div>
</body>
</html>
</body>

</html>
```

Nützliche Links

Positionierung mittels CSS in der Wiki erklärt

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_10:11_02_04_10_05

Last update: **2019/05/14 21:45**

11.2.4.10.6) Übung float und clear

Erstelle eine Seite mit einem Bild, das mithilfe von float und clear umflossen und dem durch overflow:hidden eine gewisse Breite zugewiesen wird.

Das Bild soll am linken Rand stehen und auf der rechten Seite umflossen werden.

So sähe es aus, wenn du den Code aus Float- und Clear-Eigenschaft kopiert hättest



Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.



KATZENBILDER!!!! Such cute!



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_04:11_02_04_10:11_02_04_10_06

Last update: **2019/05/14 21:45**

11.2.5) Weiteres

- 11.2.5.1) Browser-Weichen
- 11.2.5.1) Medienspezifische CSS-Dateien
- 11.2.5.1) Entwicklerwerkzeuge
- Layout ohne Tabellen inkl. Beispiele (CSS4You.de)

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_05

Last update: **2019/05/14 21:46**

11.2.5.1) Browser-Weichen

Wie schon bei den Problemen erwähnt, gibt es hin und wieder Unterschiede zwischen den verschiedenen Browserengines bei der Interpretierung von CSS- und (X)HTML-Code. Während es bei den aktuellen Browserversionen (Opera ab Version 10, Internet Explorer ab Version 8, Firefox ab Version 3) nicht mehr so häufig zu solchen Probleme kommt, muss man vor allem die Interpretierung in den älteren Versionen des Internet Explorers beachten. Sowohl IE6 als auch IE7 haben nach wie vor eine revelante Verbreitung.

Eine Möglichkeit dazu sind sogenannte Browser-Weichen, mit denen ein bestimmter Codeteil nur für oder ab einer bestimmten IE-Version interpretiert wird.

```
<!--[if IE 6]> <link href="content/css/ie6only.css" rel="stylesheet"
type="text/css" /> <![endif]-->
```

Die CSS-Datei ie6only.css wird nur in Internet Explorer Version 6 geladen.

```
<!--[if lt IE 8]> <link href="content/css/ieold.css" rel="stylesheet"
type="text/css" /> <![endif]-->
```

Die CSS-Datei ieold.css wird nur in IE-Versionen vor Version 8 eingebunden.

Für die Versionsangabe kann man (wie im letzten Beispiel) kleiner (lt), kleiner-oder-gleich (lte), größer (gt), größer-oder-gleich (gte) verwenden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_05:11_02_05_01

Last update: **2019/05/14 21:47**

11.2.5.2) Medienspezifische CSS-Dateien

Allgemeines

Es gibt die Möglichkeit, bestimmte CSS-Dateien nur für ein bestimmtes Ausgabemedium zu verwenden. Am wichtigsten ist hier wohl die Möglichkeit, für den Druck ein anderes Layout als für die normale Anzeige der Website zu verwenden.

Dazu bindet man die CSS-Datei folgendermaßen in die XHTML-Datei ein:

```
<link rel="stylesheet" type="text/css" media="{MEDIENTYP}"
href="{dateiname}.css">
```

Für {MEDIENTYP} kann man folgendes angeben:

Name	Bedeutung
all	umschließt alle Medientypen (Standard)
screen	normale Bildschirmausgabe
print	normaler Druck auf Papier
tty	nicht-grafische Ausgabemedien, z.B. Fernschreiber oder Textbrowser
projection	Projektionsgeräte wie Beamer
handheld	PDAs wie den Palm
tv	TV-ähnliche Ausgabemedien
aural	Sprachausgabe
braille	Braille-Zeilen zur Bildschirmausgabe für Blinde

Anwendung

Im Normalfall bindet man eine CSS-Datei für alle (also entwder media=„all“ oder keine media-Angabe) ein, und überschreibt dann mit weiteren medienspezifischen CSS-Dateien das Stylesheet für das betreffende Medium.

Beispiel:

XHTML-Datei

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
   <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
   <meta http-equiv="content-language" content="de" />
   <title>TITEL</title>
```

```
    <link rel="stylesheet" type="text/css" href="style_normal.css"
media="all" />  <!-- wird für alle eingebunden -->
    <link rel="stylesheet" type="text/css" href="style_druck.css"
media="print" /> <!-- überschreibt beim Druck manche der allg. Eigenschaften
-->
</head>
<body>
<div id="beispiel_div">
    Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo
ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis
parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec,
pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec
pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo,
rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede
mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper
nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu,
consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra
quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet.
Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur
ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus,
tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing
sem neque sed ipsum. Nam quam nunc, blandit vel, luctus pulvinar, hendrerit
id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut
libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros
faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec
sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue
velit cursus nunc, quis gravida magna mi a libero. Fusce vulputate eleifend
sapien. Vestibulum purus quam, scelerisque ut, mollis sed, nonummy id,
metus. Nullam accumsan lorem in dui. Cras ultricies mi eu turpis hendrerit
fringilla. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices
posuere cubilia Curae; In ac dui quis mi consectetuer lacinia. Nam pretium
turpis et arcu. Duis arcu tortor, suscipit eget, imperdiet nec, imperdiet
iaculis, ipsum. Sed aliquam ultrices mauris. Integer ante arcu, accumsan a,
consectetuer eget, posuere ut, mauris. Praesent adipiscing. Phasellus
ullamcorper ipsum rutrum nunc. Nunc nonummy metus. Vestibulum volutpat
pretium libero. Cras id dui. Aenean ut
</div>
</body>
</html>
```

CSS-Datei "style_normal.css"

```
body {
    font-family: Verdana, sans-serif;
    text-align: justify;
}

div#beispiel_div {
    height: 900px;
    width: 400px;
```

```
    background-color: white;
    border: black solid 3px;#
    border-radius: 10px;
    margin: 20px;
    padding: 10px;
}
```

CSS-Datei "style_druck.css"

```
body {
    font-family: Times New Roman, sans;
    /* text-align soll nicht überschrieben werden, also gibt man es hier
nicht an */
}

div#beispiel_div {
    height: 100%;
    width: 100%;
    background-color: white;
    border: black none 0;    /* keinen Rahmen anzeigen */
    margin: 10px;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_05:11_02_05_02

Last update: **2019/05/14 21:47**

11.2.5.3) Entwicklerwerkzeuge

Browser

Opera

Dragonfly

Firefox

Addons:

- Web Developer
- Firebug

online

W3Schools

Try it yourself Editor (CSS)

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_05:11_02_05_03

Last update: **2019/05/14 21:47**

11.2.6) Webseiten gestalten mit CSS - Anwendungsbespiele

- 11.2.6.1) 2 Beispiel-Menüs
- 11.2.6.2) festes einspaltiges Layout
- 11.2.6.3) flexibles zweispaltiges Layout

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_06

Last update: **2019/05/14 21:49**

11.2.6.1) 2 Beispiel-Menüs

einfaches vertikales Menü

Das ist ein ganz gewöhnliches vertikales Menü, wie es auf vielen Webseiten zum Einsatz kommt. Wie man es direkt in einer Website verwendet, also einbindet und positioniert, kommt bei den Layout-Beispielen. Das Beispiel ist zum besseren Verständnis ausführlich kommentiert.

XHTML-Code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<link rel="stylesheet" type="text/css" href="einfachesmenue.css">
<title>einfaches vertikales Menü</title>
</head>

<body>
<h1>einfaches vertikales Menü</h1>
<div id="menue">
    <ul id="navigation">
       <li><a href="#">Menüpunkt 1</a></li>
       <li><a href="#">Menüpunkt 2</a></li>
       <li><a href="#">Menüpunkt 3</a></li>
       <li><a href="#">Menüpunkt 4</a></li>
       <li><a href="#">Menüpunkt 5</a></li>
       <li><a href="#">Menüpunkt 6</a></li>
    </ul>
</div>
</body>
</html>
```

CSS-Code:

```
body {
    font-family: Verdana, sans-serif;
}


div#menue {
    border: 1px black solid;
    width: 140px;    /* DIV-Box wird durch nachfolgende padding-Angabe zu
180px! */
    padding: 0 20px;    /* oben kein Innenabstand, an den Seiten schon */
    border-radius: 10px;     /* DIV-Box soll runde Ecken haben */
}
```

```
div#menue ul {
    list-style-type: none;    /* Kein Aufzählungszeichen notwendig */
    margin-left: -41px;        /* Listenpunkte (=Menüpunkte) sollen nicht
eingerückt sein */
}

div#menue ul a{          /* man hätte hier auch #div menu ul li a oder
div#menu a oder sogar einfach nur a (jedoch nicht sinnvoll bei einer ganzen
Website) schreiben können */
    background-color: gray;
    color: white;
    text-decoration: none;
    border: 1px black solid;

/* Folgende display-Regel zwingt die Menü-Links dazu, die Eigenschaften
eines      Block-Elements anzunehmen. Ansonsten könnte man keine Außen- und
Innenabstände festlegen, genausowenig wie die Breite der a-Elemente und
somit des Rahmens um sie festlegen könnte */
    display: block;
    margin: 3px 0;    /* Außenabstand (somit Abstand zu anderen Menüpunkten)
nach oben und unten 2px, an die Seiten 0 */
    padding: 4px;    /* Innenabstand, also Abstand des Textes vom
Menüpunktsrahmen */
    width: 132px;    /* Fixe breite von 136px, wird durch padding-Angabe zu
140px!    */
    border-radius: 5px;        /* Menüpunkten haben runde Ecken */
}

div#menue ul a:hover {
    background-color: #acacac;
}

div#menue ul a:active {
    background-color: black;
}
```

horizontales Dropdown-Menü

Das folgende Beispiel zeigt ein mit CSS erstelltes Dropdown-Menü. Um volle Kompatibilität auch zu älteren Browsern zu haben, muss man zusätzlich Javascript verwenden. Für die aktuellen Browser wäre das aber nicht mehr nötig.

XHTML-Code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
```

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<link rel="stylesheet" type="text/css" href="dropdownmenue.css">
<title>Dropdown Menü</title>

<!-- Javascript für ältere Browser -->
<script type="text/javascript">
/* <![CDATA[ */
sfHover = function() {
 var sfEls =
document.getElementById("navigation").getElementsByTagName("li");
 for (var i=0; i<sfEls.length; i++) {
     sfEls[i].onmouseover=function() {
         this.className+=" sfhover";
     }
     sfEls[i].onmouseout=function() {
         this.className=this.className.replace(new RegExp(" sfhover\\b"),
"");
     }
 }
}
if (window.attachEvent) window.attachEvent("onload", sfHover);

/* ]]> */
</script>

</head>
<body>
<h1>Dropdown Menü</h1>
<ul id="navigation">
   <li><a href="#">Menüpunkt 1</a>
      <ul>
         <li><a href="#">Untermenüp. 1</a></li>
         <li><a href="#">Untermenüp. 2</a></li>
         <li><a href="#">Untermenüp. 3</a></li>
         <li><a href="#">Untermenüp. 4</a></li>
      </ul>
   </li>
   <li><a href="#">Menüpunkt 2</a>
      <ul>
         <li><a href="#">Untermenüp. 1</a></li>
         <li><a href="#">Untermenüp. 2</a></li>
         <li><a href="#">Untermenüp. 3</a></li>
      </ul>
   </li>
   <li><a href="#">Menüpunkt 3</a>
      <ul>
         <li><a href="#">Untermenüp. 1</a></li>
         <li><a href="#">Untermenüp. 2</a></li>
         <li><a href="#">Untermenüp. 3</a></li>
         <li><a href="#">Untermenüp. 4</a></li>
      </ul>
```

```
   </li>
   <li><a href="#">Menüpunkt 4</a>
      <ul>
         <li><a href="#">Untermenüp. 1</a></li>
         <li><a href="#">Untermenüp. 2</a></li>
      </ul>
   </li>
</ul>
<div id="inhalt"><p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam
erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea
rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum
dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed
diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat,
sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum.
Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet.</p>
<p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet
clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam
nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed
diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet
clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet.</p></div>
</body>
</html>
```

CSS-Code:

```
body {
    font: 100.01% sans-serif;
    background-color: #fff;
    color: #807573;
}

h1 {
    font-size: 1.2em;
}

#navigation, #navigation ul {
   margin: 0;
   padding: 0;
   list-style: none;
   font-weight: bold;
}
ul#navigation li a {
   background-color: #FFEBE6;
   color: black;
   text-decoration: none;
```

```
   display: block;
   width: 8em;
   padding-top: 5px;
}
#navigation li {
   float: left;
   width: 8em;
}
#navigation li ul {      /* steuert die untergeordnete Liste, also das
Untermenü an */
   position: absolute;
   width: 8em;
   left: -999em;    /* Untermenü wird weit außerhalb des Browserfensters
positioniert ... */
   background-color: #FFEBE6;

}
#navigation li:hover ul, #navigation li.sfhover ul {
   left: auto;  /* ... und bei Hover auf einen Hauptmenüpunkt zurückgeholt
*/
}

#navigation li li a:hover {    /* steuert einen Untermenüpunkt bei hover an
*/
   background-color: #BFB0AC;
}

#navigation li li a:active {
    background-color: white;
}

#inhalt {
    clear: left;
    width: 32em;
    padding-top: 5px;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_06:11_02_06_01

Last update: **2019/05/14 21:49**

11.2.6.2) festes einspaltiges Layout

Das folgende Beispiel zeigt ein mit CSS gestaltetes einspaltiges Layout.

XHTML-Code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<link rel="stylesheet" type="text/css"
href="festes_einspaltiges_layout.css">
<title>festes einspaltiges Layout</title>
</head>
<body>
<div id="container">
     <div id="titel">
          <h1>Kopfzeile mit Logo</h1>
     </div>
     <div id="navigation">
          <ul>
              <li><a href="#" class="erste">Men&uuml;punkt</a></li>
              <li><a href="#">Men&uuml;punkt</a></li>
              <li><a href="#">Men&uuml;punkt</a></li>
              <li><a href="#">Men&uuml;punkt</a></li>
              </ul>
     </div>
     <div id="inhalt">
          <h1>Lorem ipsum dolor</h1>
          <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed
diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. </p>

          <p>Duis autem vel eum iriure dolor in hendrerit in vulputate velit
esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero
et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril
delenit augue duis dolore te feugait nulla facilisi. </p>

          <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed
diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. </p>

          <p>Duis autem vel eum iriure dolor in hendrerit in vulputate velit
esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero
et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril
delenit augue duis dolore te feugait nulla facilisi. </p>
```

```
          <p>Duis autem vel eum iriure dolor in hendrerit in vulputate velit
esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero
et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril
delenit augue duis dolore te feugait nulla facilisi. </p>
     </div>
     <div id="fusszeile">
        <span>Platz für den Copyright-Vemerk, Versionsangabe, ...</span>
     </div>

</div>
</body>
</html>
```

CSS-Code

```
html, body     {
           margin:0;
           padding:0;
}
body {
           font-size: 100.01%;
           font-family: Arial, sans-serif;
           background-color: #E6DCCF;
           color: #E6DCCF;
           text-align: center;
}

#container {
           width: 40em;
           text-align: left;
           margin: 20px auto;
}

#titel {
           background-color: #5C73B8;
           border: 2px solid #604420;
           border-bottom: 1px solid #604420;
}

#fusszeile {
    color: black;
    background-color: white;
    border: 2px solid #604420;
    margin-top: -2px;
}

#navigation {
           background-color: #E6B873;
           color: #3054BF;
           border: 2px solid #604420;
           border-top:  1px solid #604420;
```

```
            border-bottom: 1px solid #E6B873 ;
            padding: 0.8em;
            text-align: center;
 }

#inhalt {
            background-color: #fff;
            color: #604420;
            border: 2px solid #604420 ;
}
#inhalt h1 {
            margin: 0 0 0 25px;
            padding-top: 1em;
}
#navigation a {
            color: #405080;
            text-decoration: none;
}
#navigation a:hover {
            color: black;
}


#navigation ul {
            list-style-type: none;
            display:inline;
            padding:0;
            margin: 0 0 0.5em 0;
}

#navigation li {
            display:inline;
}
#navigation li a {
            border-left: 2px solid #604420;
            padding: 3px 20px;
}

#navigation li a.erste {
            border: none;
}

h1 {
            font-size: 1.3em;
}
p  {
            margin: 25px;
}
#titel h1 {
            padding: 10px 0;
            text-align: center;
```

```
            margin: 0;
}
/*Diese Angabe ben&ouml;tigt der IE 5.0 */
* html #navigation ul a {
            width: 1em;
            w\idth: auto;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_06:11_02_06_02

Last update: **2019/05/14 21:49**

11.2.6.3) flexibles zweispaltiges Layout

Und nun ein flexibles (→ passt sich an Browser an) zweispaltiges Layout

XHTML-Code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<link rel="stylesheet" type="text/css"
href="flexibles_zweispaltiges_layout.css">
<title>Zweispalter mit Kopf- und Fu&szlig;zeile</title>

</head>
<body>
<div id="container">
<div id="kopf"><h1>Platz f&uuml;r ein Logo</h1></div>
<div id="navigation">
<h1>Navigation</h1>
<ul>
<li>Ein Men&uuml;punkt</li>
<li>Ein Men&uuml;punkt</li>
<li>Ein Men&uuml;punkt</li>
<li>Ein Men&uuml;punkt</li>
<li>Ein Men&uuml;punkt</li>
<li>Ein Men&uuml;punkt</li>
<li>Ein Men&uuml;punkt</li>
</ul>
</div>

<div id="inhalt">
<h1>Lorem ipsum dolor</h1>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. </p>

<p>Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse
molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero et
accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit
augue duis dolore te feugait nulla facilisi. </p>

<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. </p>
```

```
<p>Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse
molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero et
accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit
augue duis dolore te feugait nulla facilisi. </p>
<p>Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse
molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero et
accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit
augue duis dolore te feugait nulla facilisi. </p>
</div>

<div id="fuss"> Copyright-Angabe</div>
</div>
</body>
</html>
```

CSS-Code:

```
body, html, #navigation, #container, #inhalt, #fuss {
            margin:0;
            padding:0;
}
body {
            font: 100.01% sans-serif;
            background-color: #E6DCCF;
            text-align: center;
}
#container {
            width: 80%;  /* Webseite soll 80% des Browserfensters in der
Breite belegen */
            text-align: left;
            margin: 0 auto;
            background-color:#E6B873;
}
#kopf {
            background-color: #3054BF;
            color: #E6DCCF;
            padding: 10px;
            text-align: center;
}
#navigation {
            width: 14em;
            float: left;
            background-color:#E6B873;
            color: #3054BF;
}
#inhalt {
            background-color: #fff;
            color: #806640;
            margin-left: 14em;
            padding-bottom: 10px;
}
```

```
#fuss {
            background-color: #3054BF;
            color: #E6DCCF;
            padding: 10px;
            text-align: center;
            clear: left;
}
p {
            margin: 25px 25px 0 25px;
}
h1 {
            margin: 0 25px;
            padding:25px 0;
            font-size: 1.3em;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_06:11_02_06_03

Last update: **2019/05/14 21:49**

11.2.7) Übungen

- 11.2.7.1) Übung 1 - Einbinden
- 11.2.7.2) Übung 2 - Einfache Elemente
- 11.2.7.3) Wiederholungsübung zu "XHTML-Elemente gestalten": Anmeldung bei XY
- 11.2.7.4) Screenshot einer Website nachbauen
- Abschließendes CSS-Quiz

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_07

Last update: **2019/05/14 21:51**

11.2.7.1) Übung 1 - Einbinden von CSS

- Erstellen eine CSS-Datei, die die folgenden Formate definiert:
  - h1: Schriftgröße 18pt, Farbe dunkelgrau
  - h4: Schriftgröße 14pt, Schriftgewicht normal
  - p: Schriftgröße 12pt, Farbe blau, Hintergrund hellgrau
- Validiere die CSS-Datei mithilfe des CSS-Validation Service des W3C
- Erstelle ein HTML-Dokument, das eine Überschrift 1. Ordnung und zwei Absätze, denen jeweils eine Überschrift 4. Ordnung vorausgeht, enthält. Binden Sie die externe CSS-Datei ein.
- Wandle das HTML-Dokument mithilfe von Tidy online in ein gültiges XHTML-Dokument um (sofern du nicht bereits ein XHTML-Dokument erstellt hast)
- Validiere das XHTML-Dokument.

Links

- Try it Editor (W3schools)

HTML Validierung

- Markup Validation Service des W3C
- Web Design Group
- Validome

CSS Validierung

- CSSCheck der Web Design Group (WDG)
- CSS Validator des W3C

HTML -> XHTML Konvertierung

- Tidy Online

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_07:11_02_07_01

Last update: **2019/05/14 21:51**

11.2.7.2) Übung 2 - Einfache Elemente

Erzeuge eine neue XHTML-Datei (quiz.html) und erstelle das Grundgerüst einer XHTML-Seite:

XHTML-Code

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
  <head>
    <title> Quiz </title>
  </head>
  <body>
    <h2> Wer wird Auktionär </h2>
    <p>In diesem Quiz können Sie mit der richtigen Beantwortung von 15
Fragen
    zum Auktionär werden. Als Hilfestellung stehen Ihnen die bekannten drei
    Broker zur Verfügung:</p>
    <ul>
        <li>Publikumsbroker</li>
        <li>Anrufbroker</li>
        <li>Fifty/Fifty Broker</li>
    </ul>

    <p>Und hier kommt auch schon die erste Frage:</p>

    <div>Ein bekanntes Sprichwort lautet:
       <div>"Es ist noch kein Meister .."</div>
       <ol>
         <li>.. aus dem Bett gefallen.</li>
         <li>.. zur Ruhe gekommen.</li>
         <li>.. sitz' ich beim Schwager vorn.</li>
         <li>.. vom Himmel gefallen.</li>
       </ol>
     </div>
     <p>Herzlichen Glückwunsch, wollen Sie weitermachen ?</p>
  </body>
</html>
```

1. Lagere die CSS Angaben in eine externe Datei quiz.css aus und bette diese in dein HTML Dokument ein.
2. Modifiziere die Dokumenteigenschaften, indem du
   - die Hintergrundfarbe einstellst (background-color).
   - einen Rand links und rechts vorgibst (margin-left, margin-right).
   - die Schriftgrösse um 20% vergrößerst. (font-size).
   - die Schriftart wechselst (font-family).
3. Verändere die Überschrift, indem du
   - die Schriftfarbe wechselst (color).

- den Text zentrierst (text-align).
- die Schriftgrösse verdoppelst (font-size).
- die Schriftart wechselst (font-family).
- den Text unterstreichst (text-decoration).

4. Verändere die Quizfrage (Element), indem du
   - einen Rahmen zeichnest (border).
   - die Hintergrundfarbe einstellst (background-color).
   - den Rahmen zentrierst (text-align).
5. Verändere die Antwortliste (<ol> Element), indem du
   - Grossbuchstaben verwendest (list-style).
   - die Textfarbe wechselst.
   - den Text fett und kursiv setzt. (font-weight, font-style).
6. Verändere die Brokerliste (<ul> Element), indem du
   - den runden Listenpunkt in einen quadratischen Listenpunkt änderst (list-style-type).
   - ein eigenes kleines Icon verwendest (im Internet [*.gif, *.png] suchen und runterladen) (list-style-image).
7. Verändere die Dokumenteigenschaften, indem du
   - eine Hintergrundgrafik mit Farbverlauf verwendest.
   - Formatiere den Hintergrund so (background-attachment, etc.), wie es dir am besten gefällt.
8. Experimentiere weiter, indem du
   - CSS-Elemente einbaust, die du noch nicht verwendet hast...

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

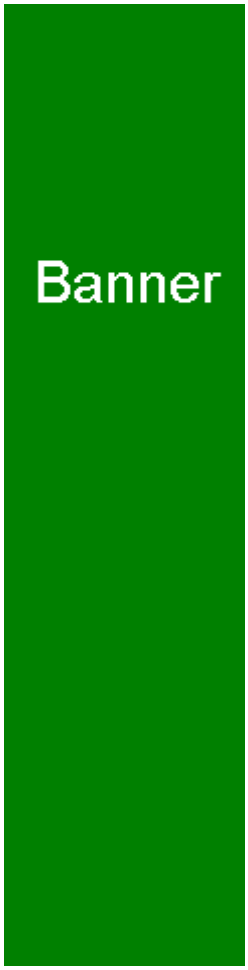
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_07:11_02_07_02

Last update: **2019/05/14 21:52**

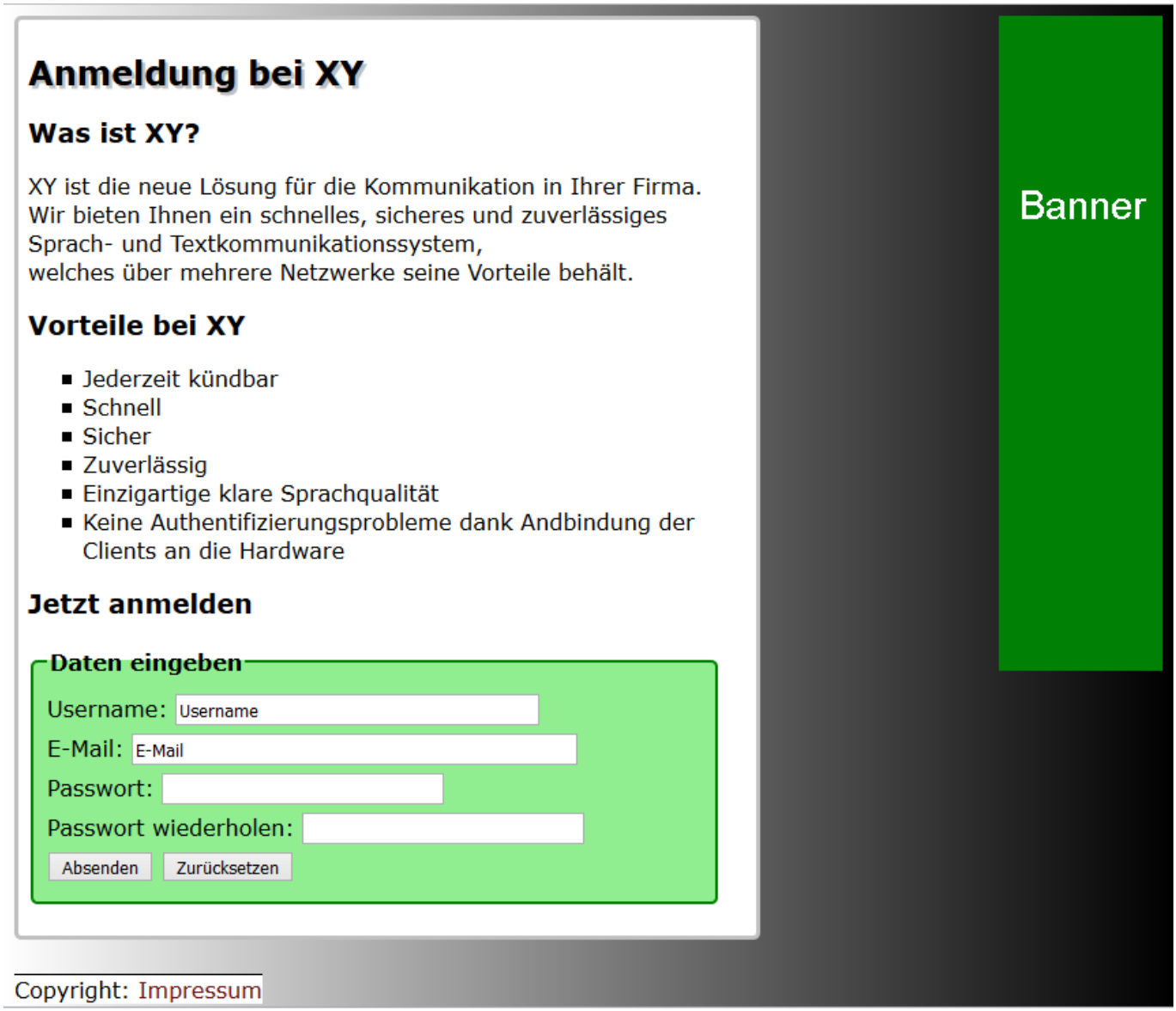
11.2.7.3) Wiederholungsübung zu "XHTML-Elemente gestalten": Anmeldung bei XY

Der Benutzer soll sich bei „XY“ anmelden können.

- *Die folgenden Angaben sind nur die grundsätzlichen Richtlinien, selbstverständlich können weitere CSS-Eigenschaften und XHTML-Elemente verwendet werden!*
- Erstelle ein XHTML-Dokument und eine externes Stylesheet „style.css“
  - Die Schriftart für das gesamte Dokument soll Verdana sein
  - Der Hintergrund der gesamten Seite soll ein Farbverlauf von schwarz auf weiß sein (Stichwort: `background-image` und `linear-gradient`)
- Für alle Links gilt:
  - Sie sollen nicht unterstrichen sein
  - Sie sollen dunkelrot sein
  - Sie sollen bei Berührung mit der Maus schwarz werden
  - Während sie angeklickt (aber noch nicht losgelassen) werden, sollen sie fett sein
- Es soll eine DIV-Box geben, die
  - 65% des verfügbaren Platzes in der Breite nutzt, jedoch eine passende minimale Breite (die entsprechende Eigenschaft findet sich auf css4you) hat
  - die Hintergrundfarbe weiß hat
  - Einen beliebigen (nicht angegeben) Platz in die Höhe nutzt
  - Einen schwarzen Rahmen mit einer Strichstärke von 3px mit abgerundeten Ecken (Abrundungsgrad beliebig) hat
  - Einen Innenabstand von 7px hat
- In dieser DIV-Box soll
  - eine h1-Überschrift (die mit CSS passend formatiert wird, u.a. Schriftgröße 1.5em + schattiert) „Anmeldung bei XY“
  - eine h2-Überschrift (die mit CSS passend formatiert wird, u.a. Schriftgröße 1.2em) „Was ist XY?“
  - darunter ein Absatz sein, in dem den Nutzer in Fließtext darüber informiert wird, wozu er sich anmeldet. (Blindtext möglich!)
  - unter diesem Absatz eine h2-Überschrift „Vorteile bei XY“
  - zu dieser soll es eine Liste (Aufzählungszeichen Quadrat oder kleine Grafik) geben, in dem dem Nutzer Vorteile die er durch eine Mitgliedschaft hat aufgelistet werden
  - darunter soll erneut eine h2-Überschrift „Jetzt anmelden“ sein
  - zu dieser soll es ein ansprechend gestaltetes Anmeldeformular (mit Verwendung der neuen XHTML-Tags fieldset) geben, legend soll sein „Daten eingeben“
- Als Footer soll es eine weitere DIV-Box geben, die am unteren Rand der Seite positioniert wird.
  - Hintergrund dieser Box soll weiß sein
  - Außerdem soll es auf den oberen Seite der Box einen schwarzen Rahmen geben
  - darin soll es einen Copyright-Vermerk und einen Link Impressum geben
- Für den Fall das der der Benutzer die Seite als Anmeldeformular ausdrucken will, soll es ein entsprechendes Druck-Layout geben (medienspezifische CSS-Datei)
- Validiere die XHTML und die CSS Datei und beseitige alle Fehler
- Rechts neben der Haupt-DIV-Box soll es einen 120×480 Pixel großen vertikalen Banner geben. (Verwende den grünen Banner von unten, Tipp: float, weiteres allumfassendes DIV)



Mögliche Lösung



HTML-Code

```
<html>

<head>
<title>Anmeldung bei XY</title>
<meta name="author" content="">
</head>

<body text="#000000" bgcolor="#FFFFFF" link="#FF0000" alink="#FF0000"
vlink="#FF0000">

<div>

<h1>Anmeldung bei XY</h1>
<h2>Was ist XY?</h2>
XY ist die neue L&ouml;sung f&uuml;r die Kommunikation in Ihrer Firma.<br>
Wir bieten Ihnen ein schnelles, sicheres und zuverl&auml;ssiges Sprach- und
Textkommunikationssystem,<br>
```

```
welches &uuml;ber mehrere Netzwerke seine Vorteile beh&auml;lt.<br>
<h2>Vorteile bei XY</h2>
<ul>
  <li>Jederzeit k&uuml;ndbar</li>
  <li>Schnell</li>
  <li>Sicher</li>
  <li>Zuverl&auml;ssig</li>
  <li>Einzigartige klare Sprachqualit&auml;t</li>
  <li>Keine Authentifizierungsprobleme dank Andbindung der Clients an die
Hardware</li>
</ul>
<h2>Jetzt anmelden</h2>

<form action="anmeldung.html" method="POST">
<fieldset><legend>Daten eingeben</legend>
Username: <input type="Text" name="username" value="Username" size="40"
maxlength="40"><br>
E-Mail: <input type="Text" name="email" value="E-Mail" size="50"
maxlength="50"><br>
Passwort: <input type="Password" name="passwort" size="30"
maxlength="30"><br>
Passwort wiederholen: <input type="Password" name="passwort1" size="30"
maxlength="30"><br>
<input type="Submit" name="absenden" value="Absenden"> <input
type="reset"><br>
</fieldset>
</form>
</div>

<div>
    Copyright: <a href=„#“>Impressum</a>
</div>

</body>

</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_07:11_02_07_03

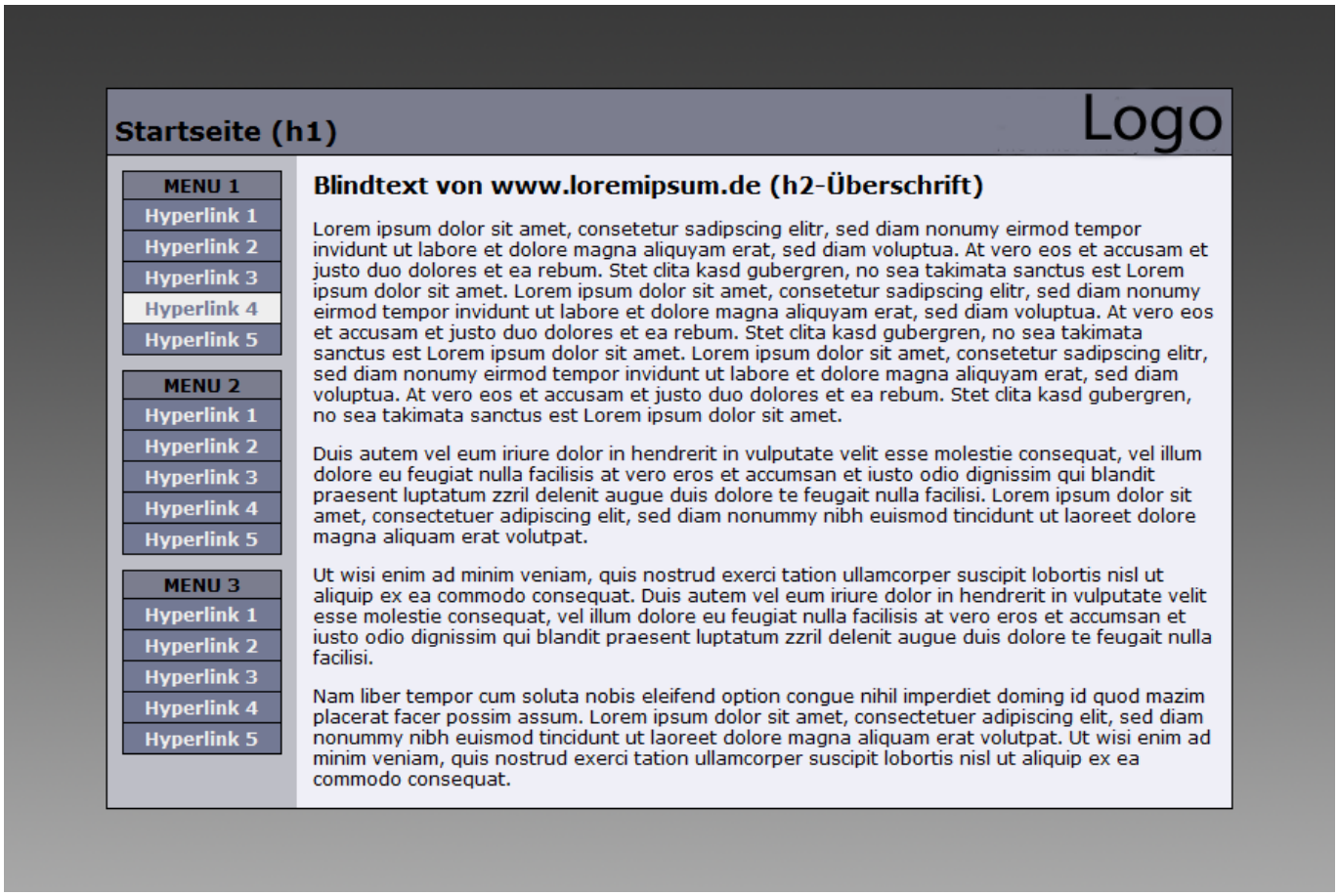
Last update: **2019/05/14 21:52**

11.2.7.4) Übung: Screenshot einer Website nachbauen

Baue folgende Webseite möglichst originalgetreu nach.

Hinweise:

- der hervorgehobene Menüpkt wird gerade mit der Maus berührt
- die Gesamtbreite soll immer 70% des verfügbaren Platzes ausmachen, der Menübereich soll immer gleich breit bleiben (128px)
- das „Logo“ rrechts oben soll eine Bilddatei sein!



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_07:11_02_07_04

Last update: **2019/05/14 21:52**

11.2.8) Links

- CSS4You: deutschsprachige CSS-Referenz mit vielen Informationen
- Übersicht über Neuerungen in CSS3 (webmasterpro.de)
- CSS auf 4webmaster.de
- Webseiten-Beispiele von Sebastian
- CSS-Workshop
- W3Schools: englischsprachige Referenz-, Übungs- und Lernplattform
- Buch: Jetzt lerne ich CSS
- Validator (W3):
  - CSS
  - Unicorn (HTML + CSS + Feed)
- browsershots.org
- CSS HOW-TOs, CSS3-Features und Templates

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_02:11_02_08

Last update: **2019/05/14 21:53**

Interaktion und Dynamik mit JavaScript

- 11.3.1) Einführung
- 11.3.2) Hintergründe
- 11.3.3) Einfache Befehle
- 11.3.4) Grundlegende Sprachelemente
- 11.3.5) Kontrollstrukturen
- 11.3.6) Funktionen
- 11.3.7) Objekte
- 11.3.8) Vordefinierte Objekte
- 11.3.9) Objektmodell
- 11.3.10) Zugriff auf HTML-Dokumente
- 11.3.11) Event-Handler
- 11.3.12) Unsere Scripts

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03

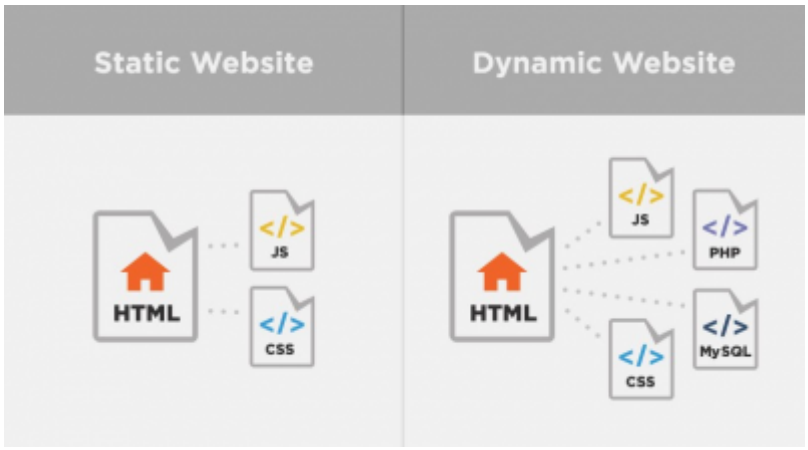
Last update: **2019/05/16 16:43**

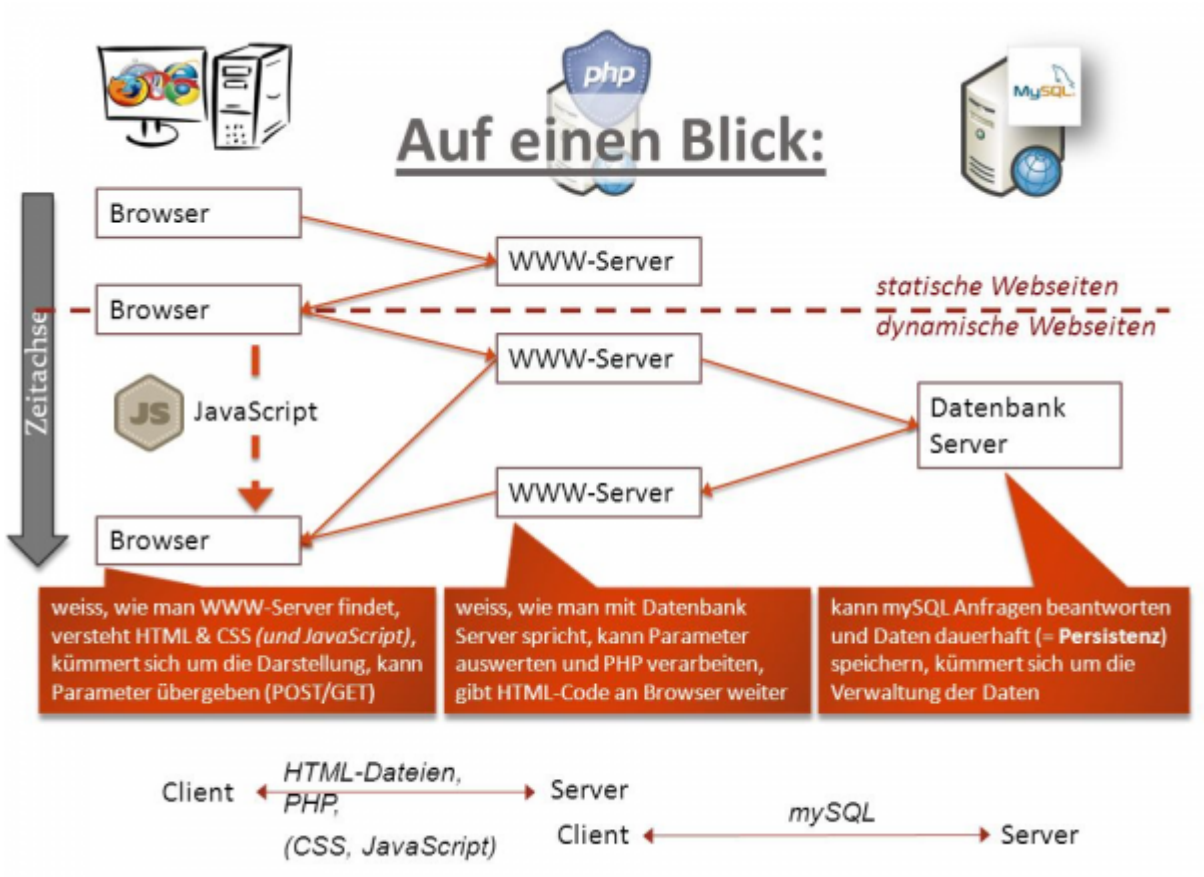
11.3.1) Einführung

Was ist unter JavaScript zu verstehen?

JavaScript ist eine Programmiersprache, die als Zusatztechnik in Webseiten eingebunden wird. Die JavaScript-Programme, auch Scripte genannt, werden vom Web-Browser interpretiert. Das heißt, sie werden in Prozessoranweisungen übersetzt und ausgeführt. Da JavaScripte auf dem Rechner des Websurfers ausgeführt werden, spricht man von einer clientseitigen Programmiersprache, um sie von Programmen abzugrenzen, die auf dem Web-Server laufen, wie etwa PHP- oder Perl-Scripte.

Mithilfe von Javascript werden statische Webseiten (HTML+CSS) zu **dynamische, mit dem Benutzer interagierende, Webseiten** ohne dass ein Webserver mit PHP benötigt wird.



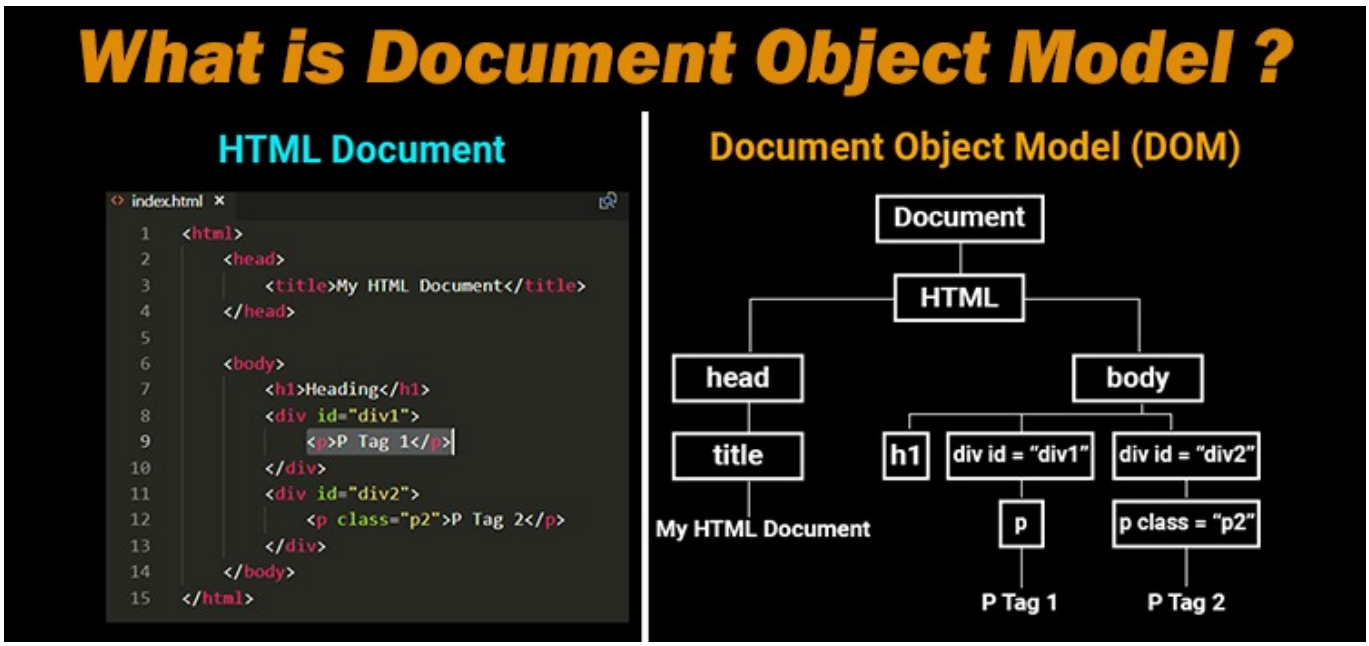


Was macht JavaScript und wie funktioniert JavaScript

JavaScripte haben Zugriff auf das Browserfenster und das darin angezeigte HTML-Dokument. Ihre wichtigste Aufgabe besteht darin, auf Benutzereingaben im Dokument zu reagieren (z.B. klickt der Benutzer auf ein Element oder gibt einen Text in ein Formularfeld ein). JavaScripte können daraufhin Änderungen im gegenwärtigen HTML-Dokument vornehmen. Diese Änderungen finden nur im Browser, genauer gesagt im Arbeitsspeicher des Rechners statt, während das Dokument auf dem Web-Server unangetastet bleibt.

JavaScripte können sowohl den Inhalt als auch die Darstellung des Dokuments ändern. Auf diese Weise kann ein Dokument interaktiv und »dynamisch« gestaltet werden. Das bedeutet, es kann auf Benutzereingaben reagieren und sich ändern, ohne dass ein neues Dokument vom Web-Server abgerufen werden muss. Beispielsweise können Zusatzinformationen eingeblendet werden, sobald der Anwender mit der Maus auf ein bestimmtes HTML-Element klickt.

Den Zugriff auf und das Verändern des Dokuments regelt das sogenannte Document Object Model (DOM) – dazu später mehr. Der überragende Teil der JavaScript-Programmierung besteht aus dem Umgang mit dem DOM. Wegen dieser zentralen Wichtigkeit wird auch von »DOM Scripting« gesprochen. Ein älterer Begriff lautet »Dynamisches HTML«, abgekürzt DHTML. Dies soll Ihnen verdeutlichen, dass sich JavaScript in erster Linie darum dreht, mit dem HTML-Dokument zu hantieren, im dem es ausgeführt wird.



Bedeutung von JavaScript für das Web

JavaScript ist aus dem heutigen Web nicht mehr wegzudenken und nimmt neben HTML und CSS eine wichtige Rolle ein. Beinahe jede Website verwendet kleinere oder größere JavaScripte.

Viele verbreitete Funktionen im Web lassen sich mit JavaScript vereinfachen und benutzerfreundlicher gestalten. JavaScript ermöglicht auch neue Funktionen. Das bedeutet allerdings nicht, dass Ihre Website JavaScript verwenden muss, um vollständig und erfolgreich zu sein. Viel JavaScript hilft nicht unbedingt viel. Ein bedachter JavaScript-Einsatz sorgt beispielsweise dafür, dass eine Website auch

ohne JavaScript gut benutzbar und zugänglich ist.

Die Sprache JavaScript existiert schon seit 1995 und wurde stetig fortentwickelt. Lange war JavaScript das »Schmuddelkind« unter den Webtechniken und seine Daseinsberechtigung wurde angezweifelt. Erst ab 2005 kam es zu einer Renaissance von JavaScript und die JavaScript-Nutzung explodierte unter dem Schlagwort Ajax. Zudem entstanden Theorien, die den vernünftigen JavaScript-Einsatz begründen.

JavaScript als vollwertige Programmiersprache

Für viele Webautoren ist JavaScript die erste Programmiersprache, mit der sie zu tun bekommen. HTML und CSS sind zwar auch künstliche Rechnersprachen, aber keine Programmiersprachen – es lassen sich damit keine Programme schreiben, die Anweisung für Anweisung abgearbeitet werden.

Was zuerst spitzfindig klingt, ist ein Unterschied ums Ganze: Mit HTML lassen sich Texte auszeichnen und Dokumente aufbauen. In HTML werden sogenannte Elemente und deren Tags genutzt, um Text zu strukturieren. Mit CSS werden diese Dokumente formatiert und gelayoutet. In CSS werden sogenannte Deklarationen verfasst, die HTML-Elemente ansprechen und ihnen Formatierungen zuweisen.

Eine Programmiersprache hingegen funktioniert grundlegend anders und erfordert eine eigentümliche Herangehensweise an Probleme. Die Grundlagen der Programmierung können an dieser Stelle nicht beschrieben werden, sie sind aber eine wesentliche Voraussetzung für den Umgang mit JavaScript.

Besonderheiten und Tücken von JavaScript

JavaScript ist eine Baustelle: Die Sprache hat sehr klein angefangen und wurde nicht entworfen, um den Anforderungen des heutigen Einsatzes zu genügen. Daher stecken der JavaScript-Kern und viele JavaScript-Techniken voller Stärken und Schwächen. Manches ist einsichtig und einfach, anderes verworren und macht Ihnen die Arbeit schwer.

Heutzutage beruhen große Webanwendungen auf umfangreichen und hochkomplexen JavaScripten. JavaScript gerät dadurch in das Interesse von Informatikern und Software-Entwicklern, die die Sprache erforschen, mit ihr experimentieren und Grenzen austesten. Empfehlenswerte Programmiertechniken sind dadurch erst nach und nach ans Licht gebracht worden – und es ist ein andauernder Prozess, den Sie verfolgen sollten und an dem Sie sich beteiligen können.

JavaScript nimmt unter den Programmiersprachen einen besonderen Platz ein. Wenn Sie bereits andere Programmiersprachen kennen, werden Sie sicher Gemeinsamkeiten erkennen und Ihr Wissen wiederverwenden können. Doch früher oder später werden Sie auf Eigentümlichkeiten von JavaScript stoßen, die Sie so in keiner anderen im Web verbreiteten Programmiersprache finden werden. Sobald Sie komplexere Scripte schreiben, sollten Sie mit diesen Eigentümlichkeiten und den daraus resultierenden Programmiertechniken Bekannschaft machen, denn Sie werden sie zu schätzen lernen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_01

Last update: **2019/05/16 14:48**

11.3.2) Hintergründe

JavaScript ist mittlerweile zu einer ernsthaften Programmierumgebung geworden, das heißt, man kann auch rein in JavaScript programmieren.

Node.js ist eine JavaScript aufbauende Plattform, mit der ein eigener Webserver geschrieben werden kann. Diesen kann man statt Apache oder parallel zu Apache verwenden.

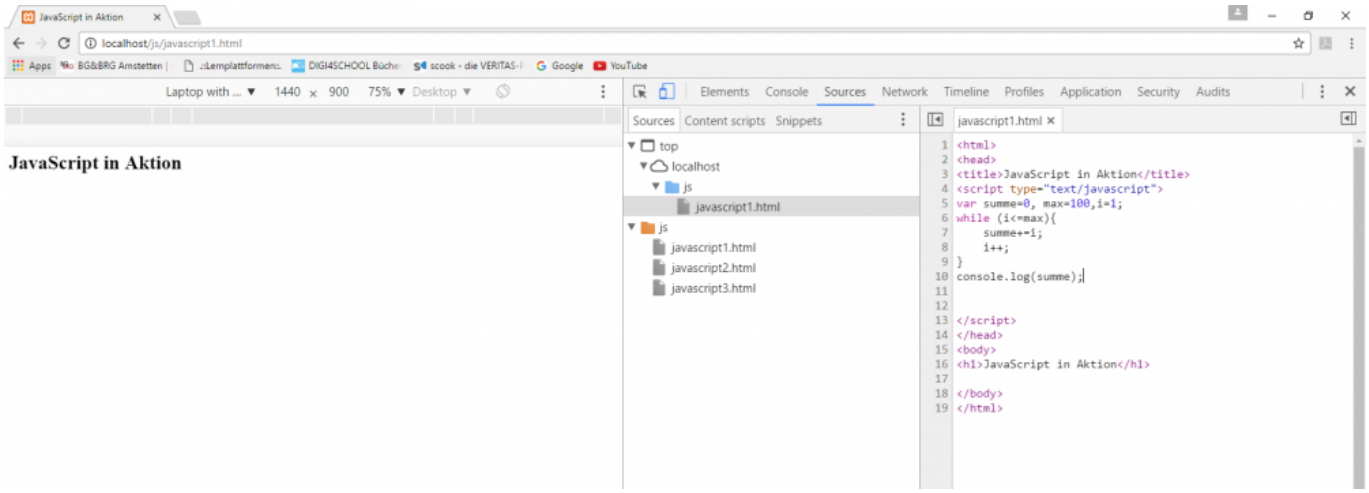
Die neuesten Entwicklungen laufen darauf hinaus, dass man JavaScript auch als Programmierung am Desktop verwendet. Dabei ist im wesentlichen jedes Programm ein eigenes Browserfenster. (Electron.io) Man startet lokal einen Browser, merkt aber nicht, dass es ein Browser ist. Man kann dabei eine Mischung aus lokaler Programmierung und Webseitenprogrammierung machen.

JavaScript dient dazu, im Browser Programme laufen zu lassen, man braucht keinen Server dazu. Man kann weiter Inhalte nachladen (AJAX, bzw. asynchrones JavaScript), ohne die Seite neu zu laden.

Auch wenn man für JavaScript eigentlich keinen eigenen Webserver braucht, ist es eine gute Idee, mit einem Webserver zu arbeiten, da man dadurch PHP und JavaScript gleichzeitig verwenden kann.

Es empfiehlt sich als Browser Google-Chrome zu verwenden. Mittels F12 kommt man Mittels Sources - Rechtsklick auf einen Ordner: „Add Folder to workspace“ - Zulassen - erhält man die Möglichkeit, den Ordner zu verbinden. Damit kann man im Browser z.B. eine bestehende Datei bearbeiten. Sobald man den Folder verbunden hat, wird alles mitprotokolliert, was in dem Ordner passiert (z.B. wenn man eine Datei umbenennt usw.)

Im Browser kann unter „Sources“ direkt das Programm geschrieben werden.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_02

Last update: **2019/05/16 14:48**

11.3.3) Einfache Befehle

prompt

Einlesen einer Variablen

```
max=prompt("Zahl, bis zu der summiert werden soll: ","200");
```

alert

Ausgabe

```
alert("Die Summe von 1 bis "+max+" ist: "+summe);
```

Diese aufgehenden Popup-Fenster der Befehle `prompt` bzw. `alert` können jedoch im Browser ausgeschaltet werden, daher sind sie für den Realbetrieb nicht wirklich sinnvoll!

document.write()

Das ist der allgemeine Ausgabebefehl:

```
document.write("Die Summe von 1 bis 100 ist "+summe);
```

Auslesen von Benutzereingaben

```
var max=document.getElementById("eingabe").value; // steht im <head>
...

<input type="text" id="eingabe">
```

mit `document.getElementById` kann man z.B. auf den Inhalt eines input-Feldes zugreifen

Typumwandlung

```
function summieren(){
  var max=Number(document.getElementById("eingabe").value); //Umwandlung in
Number

  if (isNaN(max)||max==0) {
     document.getElementById("ausgabe").innerHTML="?";
  }
  else{
     var summe=0,i=1;
```

```
      while (i<=max){
        summe+=i;
        i++;
      }
      document.getElementById("ausgabe").innerHTML=summe;
  }

}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_03

Last update: **2019/05/16 14:48**

11.3.4) Grundlegende Sprachelemente

JavaScript in HTML verwenden

JavaScript direkt einbinden

```
<script type="text/javascript">
  <!-- Hier werden die JavaScript-Anweisungen eingefügt -->
</script>
```

Beispiel: javascript-test.html

javascript-test.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>JavaScript-Test</title>
  <meta http-equiv="Content-script-type" content="text/javascript">
</head>
<body>
    <script type="text/javascript">
      document.write("Teste generell auf JavaScript ... Ok!<br>")
    </script>
    <noscript>
    Ihr Browser versteht kein JavaScript! Es kann nicht ausgeführt
werden!
    </noscript>
    <script language="javascript1.1">
      document.write("Teste auf JavaScript Version 1.1... Ok!<br>");
    </script>

    <script language="javascript1.2">
      document.write("Teste auf JavaScript Version 1.2... Ok!<br>");
    </script>
    <script language="javascript1.3">
      document.write("Teste auf JavaScript Version 1.3... Ok!<br>");
    </script>
    <script language="javascript1.4">
      document.write("Teste auf JavaScript Version 1.4... Ok!<br>");
    </script>
    <script language="javascript1.5">
      document.write("Teste auf JavaScript Version 1.5... Ok!<br>");
    </script>
    <script type="text/javascript; version=1.6">;
      document.write("Teste auf JavaScript Version 1.6... Ok!<br>");
```

```
    </script>
    <script  type="text/javascript; version=1.7">;
      document.write("Teste auf JavaScript Version 1.7... Ok!<br>");
    </script>

    <script type="text/javascript; version=1.8">;
      document.write("Teste auf JavaScript Version 1.8... Ok!<br>");
    </script>
</body>
</html>
```

JavaScript durch externe Datei einbinden

```
<head>
  <title>JavaScript-Test</title>
  <script type="text/javascript" src="javascript.js"></script>
</head>
```

Beispiel: javascript.js

```
document.write('Ich befinde mich in einer externen Datei.');
```

JavaScript innerhalb von HTML-Tags

```
<button type="button" onclick="document.write('Sie haben mich gedr&uuml;ckt');">
  Klicken Sie auf mich
</button>
```

```
<a href="javascript:history.back()">Zur&uuml;ck</a>
```

Ausführungsreihenfolge

JavaScript-Anweisungen werden in der Reihenfolge ausgeführt, in der sie in einem HTML-Dokument verwendet werden. Dadurch werden JavaScript-Anweisungen, die im Kopfbereich eines HTML-Dokuments eingebunden werden, vor JavaScript-Anweisungen im Rumpfbereich ausgeführt.

Ausgaben, die direkt Dokument ausgeführt werden, erscheinen beim Laden des HTML-Dokuments an der entsprechenden Stelle.

Beispiel: ausfuehrungsreihenfolge.html

ausfuehrungsreihenfolge.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
```

```
<html lang="de">
<head>
  <title>Ausführungsreihenfolge von JavaScript</title>
    <script type="text/javascript">
      document.write("Ausgabe aus dem Kopfbereich!<br>");

      function zeigeInfo()
      {
        document.write("Sie haben die Funktion zeigeInfo aufgerufen.");
      }
    </script>
</head>
<body>
  <noscript>
     Ihr Browser versteht kein JavaScript.
  </noscript>
  <script type="text/javascript">
    document.write("Ausgabe aus dem Rumpfbereich!<br>");
  </script>

  <br>
  <a href="javascript:zeigeInfo();">Information zeigen</a>
</body>
</html>
```

Allgemeine Notationsregeln

Anweisungen

```
document.write("Test");
```

Kommentare

```
// Ausgabe der Quadratzahlen von 1 bis 10
for(i = 1; i <= 10; i++)
  document.write(i * i, "<br>");
```

Reservierte Wörter

Reservierte Wörter, auch Schlüsselwörter genannt, sind feste, vorgegebene Wörter, die der Sprache JavaScript entstammen. Da jedes reservierte Wort eine feste, vorgegebene Bedeutung besitzt, dürfen diese nicht als Namen von Variablen verwendet werden. Die folgende Übersicht enthält alle reservierten Wörter von JavaScript.

abstract	debugger	final	instanceof	public	transient
boolean	default	finally	int	return	true
break	delete	float	interface	short	try
byte	do	for	long	static	typeof
case	double	function	native	super	undefined
catch	else	goto	new	switch	var
char	enum	if	null	synchronized	void
class	export	implements	package	this	volatile
const	extends	import	private	throw	while
continue	false	in	protected	throws	with

Bezeichner

Bezeichner benennen unter anderem Konstanten, Variablen und Funktionen in JavaScript-Programmen. Bei der Programmierung über einen Bezeichner auf diese Elemente zugegreifen werdeb. Ein Bezeichner wird vom Programmierer festgelegt.

```
var Kundennr = 123;         // Richtig, enthält nur Buchstaben
var $Kunden_Nr = 123;       // Richtig, enthält $, Buchstaben und den Unterstrich
var Kunden Nr = 123;        // Falsch, enthält Leerzeichen
var -Kundennr = 123;        // Falsch, beginnt mit einem Minuszeichen
var 0Kundennr = 123;        // Falsch, beginnt mit einer Zahl
var case = 123;             // Falsch, ist ein reserviertes Wort
```

Variablen

Variablen können während der Programmausführung unterschiedliche, veränderbare Werte annehmen, wie z. B. Zwischen- oder Endergebnisse aus Berechnungen. Für jede Variable wird ein Speicherplatz im Arbeitsspeicher Ihres Computers reserviert.

Beispiel für die Definition von Variablen

```
k = 10;                    // die Variable k ist vom Typ Zahl und besitzt den Wert 10
var k;                     // die Variable k ist noch undefiniert
var i, k = 10;             // die Variablen i und k sind vom Typ Zahl und besitzen
                           // den Wert 10
var k = 10, i = "Text";    // mehrere Definitionen durch Kommata trennen
```

Beispiel: variablen.html

variablen.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
  <head>
    <title>Der erste JavaScript-Test</title>
  </head>
  <body>
    <script type="text/javascript">
      // Wenn Sie die Kommentarzeichen entfernen, wird ein
Laufzeitfehler erzeugt und
      // die folgenden Ausgaben werden nicht mehr durchgeführt.

      // i;
      j = 10;
      var k;
      var l = 11;

      document.write("j hat den Wert: ", j, "<br>");
      document.write("k hat den Wert: ", k, "<br>");
      document.write("l hat den Wert: ", l, "<br>");
    </script>
  </body>
</html>
```

Wertzuweisungen

```
var summand1 = 1;
var summand2 = 8;
var summe = 0;
document.write(summe);    // Diese Anweisung löst jetzt keinen Fehler aus!
summe = summand1 + summand2;
```

Konstanten

Konstanten enthalten während der Programmausführung immer einen unveränderlichen Wert. Es

lässt sich also keine zweite Wertzuweisung an eine Konstante vornehmen, nachdem sie initialisiert wurde.

```
const MWST = 0.19;
```

Datentypen

Ganze Zahlen

Ganze Zahlen besitzen keine Nachkommastellen und können negative sowie positive Werte und den Wert null annehmen.

Dezimal	Hexadezimal
0	0x0
1	0x1
8	0x8
15	0xF
16	0x10
42	0x2A

Gleitkommazahlen

Gleitkommazahlen können Nachkommastellen besitzen. Das Kennzeichen ist entweder ein Dezimalpunkt, ein Exponent oder beides.

Fixkommazahl	Gleitkommazahl
1000.0	1.0E3
2000.4	20004.0E-1
-4000.0	-4e3
-0.004	-4e-3

```
document.write("Max: ", Number.MAX_Value);
document.write("Min: ", Number.MIN_Value);
```

```
k = 12345678901234567890;
document.write(k);  // liefert 12345678901234567000, da die Zahl zuerst in eine
                    // Gleitkommazahl konvertiert und danach gerundet wird !
```

Zeichenketten

Eine Zeichenkette (String) ist eine Folge von Zeichen, die in einfache oder doppelte Anführungszeichen eingeschlossen ist. Es ist jedoch zu beachten, dass gleiche Arten von Anführungszeichen verwendet werden.

Angabe in JavaScript	Ausgabe
"Dies ist eine Zeichenkette."	Dies ist eine Zeichenkette.
'Das ist auch eine Zeichenkette.'	Das ist auch eine Zeichenkette.
'Die Zahl "123" ist auch eine Zeichenkette.'	Die Zahl "123" ist auch eine Zeichenkette.
"'So funktioniert es auch', sagte er."	'So funktioniert es auch', sagte er.

Der wichtigste Unterschied zwischen Zeichenketten und numerischen Werten liegt in der Funktion der Operatoren. Die Elemente einer Zeichenkette können nicht arithmetisch miteinander verknüpft werden. Bei ganzen Zahlen und Gleitkommazahlen entspricht das Zeichen + der mathematischen Addition. Werden zwei Zeichenketten mit dem Zeichen + verknüpft, werden sie zu einer Zeichenkette zusammengefasst. In allen anderen mathematischen Operationen werden die Zeichenketten als Zahlenwerte interpretiert. Können Zeichenketten nicht in einen Zahlenwert umgewandelt werden, liefert die Berechnung den Wert `NaN` (not a number = keine Zahl) zurück.

```
var Zahl = 1 ;
var Zeichen = '1';
var Ergebnis  = Zahl + Zahl;          // ergibt 2;
Ergebnis = Zeichen + Zeichen;         // ergibt '11';
Ergebnis = Zeichen - Zeichen;         // ergibt 0;
Ergebnis = '1a' * Zahl;               // ergibt NaN
```

Steuerzeichen in Zeichenketten

Steuerzeichen	Bedeutung
\n	new line: Zeilenumbruch (neue Zeile)
\r	return: "Wagenrücklauf": Der Cursor steht in der nächsten Zeile wieder an Position 1.
\t	Tabulator
\f	form feed: Seitenvorschub
\b	backspace: ein Zeichen zurück
\"	doppeltes Anführungszeichen, auch innerhalb von doppelten Anführungszeichen, z. B. `alert("\"");`
\'	einfaches Anführungszeichen, auch innerhalb von einfachen Anführungszeichen, z. B. `alert('\'');`
\\	Backslash

Boolesche Werte (Wahrheitswerte)

Die booleschen Werte sind true (wahr) und false (falsch). Wahrheitswerte werden eingesetzt, wenn ein Wert nur zwei Zustände annehmen kann, z. B. Licht an oder Licht aus. Ausdrücke geben häufig einen booleschen Wert zurück, z. B. beim Vergleichen von Zahlen. Der Vergleich 2 > 3 liefert das Ergebnis false, weil die Zahl 2 nicht größer ist als 3.

Boolescher Wert	Bedeutung
`true`	"Wahr", die Bedingung ist erfüllt, z. B. 2 < 3.
`false`	"Falsch", die Bedingung ist nicht erfüllt, z. B. 2 > 3.

Objekte

Objekte sind Datenelemente, die Eigenschaften und objektgebundene Funktionen (Methoden) besitzen können. In JavaScript werden Ihnen vordefinierte Objekte angeboten, um auf verschiedene Elemente einer Webseite zuzugreifen, z. B. auf ein Eingabefeld eines Formulars. Jedes einzelne Objekt enthält Eigenschaften, die Sie auslesen können, z. B. die Adresse der geladenen Webseite oder den Namen des Fensters. Außerdem erlauben Ihnen die Objektmethoden das Ändern der Objekteigenschaften. So können Sie beispielsweise den Inhalt eines Eingabefeldes ändern.

Operatoren

Operatoren sind Zeichen, die verwendet werden, um Berechnungen durchzuführen oder Verknüpfungen und Vergleiche zwischen Variablen herzustellen.

Eine Operation arbeitet mit einem oder zwei Operanden. Entsprechend wird von unären oder binären Operatoren gesprochen. Ein Operator erzeugt immer einen Ergebniswert. Manche Operatoren können nur in Verbindung mit Variablen eines bestimmten Datentyps eingesetzt werden.

Operator	Datentyp
Arithmetischer Operator	Zahlen
Vergleichsoperator	Zahlen, Strings (Zeichenketten), boolesche Werte
Verknüpfungsoperator, Konkatenationsoperator	Strings (Zusammenfügen von Zeichenketten)
Logischer Operator	Boolesche Werte
Bit-Operator	Zahlen, boolesche Werte
Zuweisungsoperator	Alle

Arithmetische Operatoren

Arithmetische Operatoren dienen zur Berechnung eines Wertes. Dazu wird eine Operation auf einen oder mehrere Operanden angewendet.

Name	Operator	Syntax	Beispiel	Wert
Addition	+	Summand1 + Summand2	7 + 4	11
Subtraktion	-	Minuend - Subtrahend	7 - 4	3
Multiplikation	*	Faktor1 * Faktor2	7 * 4	28
Division	/	Dividend / Divisor	7 / 4	1.75

Name	Operator	Syntax	Beispiel	Wert
Modulo	%	Dividend % Divisor	7 % 4	3
Negation	-	-Operand	-(2 + 5)	-7
Inkrement	++	++Variable; Variable++	x = 10; ++x; y = 135; y++	11 136
Dekrement	--	--Variable; Variable--	x = 10; --x; y = 135; y--	9 134

Vergleichsoperatoren

Folgende Vergleichsoperatoren werden unterschieden:

Name	Operator	Syntax	Beispiel	Wert
gleich	==	Operand1 == Operand2	true == false 2 + 4 == 6	false true
ungleich	!=	Operand1 != Operand2	"Hund" != "HUND" 2 + 4 != 6	true false
strikte (Un-) Gleichheit	=== !==	Operand1 === Operand2 Operand1 !== Operand2	2 + 4 === 6 "Hund" !== "HUND"	true true
kleiner als	<	Operand1 < Operand2	4 < "5" "ADE" < "ABC"	true false

Unterschiede zwischen == und ===

Beim strikten Vergleichsoperator === müssen Typ und Wert zusammenpassen, beim Vergleichsoperator == genügt es, wenn der Wert gleich ist.

```
0 == false   // true
0 === false  // false, because they are of a different type
1 == "1"     // true, automatic type conversion for value only
1 === "1"    // false, because they are of a different type
null == undefined // true
null === undefined // false
'0' == false // true
```

```
'0' === false // false
```

Name	Operator	Syntax	Beispiel	Wert
größer als	>	Operand1 > Operand2	"abd" > "abc"	true
			"X" > 10	false
kleiner/gleich	<=	Operand1 <= Operand2	6 <= 7	true
			"abc" <= "abc"	true
größer/gleich	>=	Operand1 >= Operand2	"6" >= "6"	true
			"Buch" >= "buch"	false

Verknüpfungsoperator

Dieser Operator, der auch Konkatenationsoperator (engl.: concatenate = verknüpfen) genannt wird, verknüpft zwei Zeichenketten und liefert die zusammengesetzte Zeichenkette als Ergebniswert.

Name	Operator	Syntax	Beispiel	Wert
Verbinden	+	Operand1 + Operand2	"Zimmer" + "pflanze"	Zimmerpflanze
			x = 10; "x-Wert:" + x	x-Wert:10

Logische Operatoren

Im Gegensatz zu den Vergleichsoperatoren werden mit den logischen Operatoren die booleschen Wahrheitswerte true und false miteinander verknüpft. Der Ergebniswert eines logischen Ausdrucks besteht aus einem booleschen Wert.

Name	Operator	Syntax	Beispiel	Wert
UND	&&	Operand1 && Operand2	true && true	true
			false && true	false
			true && false	false
			false && false	false
ODER	||	Operand1 || Operand2	true || true	true
			false || true	true
			true || false	true
			false || false	false
NICHT	!	!Operand	!true	false
			!false	true

Bit-Operatoren

Bit-Operatoren werden auf Zahlen und boolesche Werte angewendet, die entsprechend ihrer binären Darstellung verknüpft werden (die Bits können z. B. für bestimmte Eigenschaften stehen). Die

booleschen Werte true und false entsprechen dabei den Werten 1 und 0.

Name	Operator	Syntax	Beispiel	Ergebnis
Bitweises NICHT	~	~Operand	~12	0000...00001100 12 1111...11110011 = -13
Bitweises ODER	|	Operand1 | Operand2	12 | 6	0000...00001100 12 0000...00000110 6 0000...00001110 = 14
Bitweises UND	&	Operand1 & Operand2	12 & 6	0000...00001100 12 0000...00000110 6 0000...00000100 = 4
Bitweises XODER (Exklusiv-ODER)	^	Operand1 ^ Operand2	12 ^ 6	0000...00001100 12 0000...00000110 6 0000...00001010 = 10
Arithmetische Linksverschiebung	<<	Operand << Anzahl	12 << 2	0000...00001100 12 0000...00110000 = 48

Name	Operator	Syntax	Beispiel	Ergebnis
Arithmetische Rechtsverschiebung	>>	Operand >> Anzahl	13 >> 2	0000...00001101 13 0000...00000011 = 3
			-13 >> 2	1111...11110011 -13 1111...11111100 = -4
Logische Rechtsverschiebung	>>>	Operand >>> Anzahl	13 >>> 2	0000...00001101 13 0000...00000011 = 3
			-13 >>> 2	1111...11110011 -13 0011...11111100 = 1073741820

Zuweisungsoperatoren

Operator	Wirkung	Beispiel	Ergebnis
=	Einfache Wertzuweisung	a = 5;	a = 5
+=	Addieren und Zuweisen	a = 5; a += 5;	5 + 5 a = 10
+=	Anfügen und Zuweisen	a = 'Ei'; a += 'dotter';	'Ei' + 'dotter' a = 'Eidotter'
-=	Subtrahieren und Zuweisen	a = 5; a -= 2;	5 - 2 a = 3
*=	Multiplizieren und Zuweisen	a = 5; a *= 2	5 * 2 a = 10
/=	Dividieren und Zuweisen	a = 20; a /= 4	20 / 4 a = 5

Operator	Wirkung	Beispiel	Ergebnis
%=	Modulo-Operation und Zuweisen	a = 5; a %= 3	5 % 3 a = 2
&=	Bitweises UND und Zuweisen	a = 5; a &= 3	0101 & 0011 a = 1
^=	Bitweises XODER und Zuweisen	a = 5; a ^= 3;	0101 ^ 0011 a = 6
|=	Bitweises ODER und Zuweisen	a = 5; a |= 3;	0101 | 0011 a = 7
<<=	Bitweises Linksverschieben und Zuweisen	a = 5; a <<= 2	0101 a = 20
>>=	Bitweises Rechtsverschieben und Zuweisen	a = 40; a >>= 2;	101000 a = 10
>>>=	Null-Fill-Rechtsverschiebung und Zuweisen	a = 100; a >>>= 4;	1100100 a = 6

Bedingungsoperator

Mithilfe des Bedingungsoperators können einige if-else-Anweisungen verkürzt dargestellt werden.

Operator	Wirkung	Beispiel	Ergebniswert
? :	Bedingung	Stunde > 12 ? 'P.M.' : 'A.M.'	Falls Stunde größer 12 ist, wird die Zeichenkette 'P.M.', ansonsten 'A.M.' geliefert

Typenkonvertierung und typeof-Operator

JavaScript ist keine streng typisierte Sprache. Der Interpreter wandelt Operanden automatisch in die entsprechenden Datentypen um, damit eine Operation richtig ausgeführt wird.

Beispiel	Ergebniswert	Erklärung
"Text plus Zahl " + 7	"Text plus Zahl 7"	Konvertierung der Zahl 7 in einen String
"Text plus Zahl " + 7 * 7	"Text plus Zahl 49"	"Punkt vor Strich", dann Konvertierung
7 + 7 + " Text plus Zahl"	"14 Text plus Zahl"	Abarbeitung der Reihe nach
"Text plus Zahl " + 7 + 7	"Text plus Zahl 77"	Abarbeitung der Reihe nach
"7" * 7	49	Konvertierung der Ziffer 7 in die Zahl 7

Beispiel	Rückgabewert als Typ
`var Variable = 5;` `Typ = typeof Variable;`	`number`
`var Variable = "Hallo";` `Typ = typeof Variable;`	`string`
`var Variable = true;` `Typ = typeof Variable;`	`boolean`
`Typ = typeof Variable1;`	`undefined`

Beispiel	Rückgabewert als Typ
`var Variable = new function("5+2");` `Typ = typeof Variable;`	`function`
`var Variable = [1, 2, 3, 4];` `Typ = typeof Variable;`	`object`

Rangfolge der Operatoren

Die Operatoren sind in ihrer Rangordnung festgelegt und werden dementsprechend nacheinander abgearbeitet:

Rangstufe (1=höchste Priorität)	Operatorzeichen	Operatorname	Operatortyp
1	( ) [ ]	Klammer	Gruppierung
2	! ++ -- - ~	NICHT Inkrement und Dekrement Negation NICHT	Logischer Operator Arithmetischer Operator Arithmetischer Operator Bit-Operator
3	* / %	Multiplikation Division Modulo-Operator	Arithmetische Operatoren
4	+	Aneinanderreihung	Konkatenationsoperator
5	+ -	Addition Subtraktion	Arithmetische Operatoren
6	<< >> >>>	Linksverschiebung Rechtsverschiebung Null-Fill-Rechtsverschiebung	Bit-Operatoren

Rangstufe (1=höchste Priorität)	Operatorzeichen	Operatorname	Operatortyp
7	< >> <= >=	Kleiner als Größer als Kleiner gleich Größer gleich	Vergleichsoperatoren
8	== === != !==	Werte sind gleich Werte sind ungleich	Vergleichsoperatoren
9	&	UND	Bit-Operator
10	^	ENTWEDER ODER	Bit-Operator
11	|	ODER	Bit-Operator
12	&&	UND	Logischer Operator
13	||	ODER	Logischer Operator
14	? :	Bedingung	Konditionaler Operator
15	= += -= *= /= %= &= ^= |= <<= >>= >>>=	Zuweisungsoperatoren	Zuweisungsoperatoren
16	, (Komma)	Aneinanderreihung	

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_04

Last update: **2019/05/16 14:49**

11.3.5) Kontrollstrukturen

Steuerung des Programmablaufs

Eine Anweisung ist ein Teil des Programms, das ausgeführt wird, um eine bestimmte Aktion durchzuführen. Sie wird durch ein Semikolon abgeschlossen. Mehrere Anweisungen werden in sogenannten Anweisungsblöcken zusammengefasst. In JavaScript werden die Anweisungen normalerweise von oben nach unten sequenziell abgearbeitet. Oft ist es jedoch erforderlich, dass Programmteile mehrmals oder gar nicht abgearbeitet werden sollen.

Um von der sequenziellen Abarbeitung des Programms abzuweichen, können wir bedingte Ausführungen und Schleifen einsetzen. Die Ausführung einer Anweisung ist dann von einer bestimmten Bedingung abhängig. Die Anweisungen können auch mehrfach ausgeführt werden.

Anweisungsblock

Eine Anweisung ist die kleinste ausführbare Einheit eines Programms. In JavaScript werden Anweisungen mit einem Semikolon (Strichpunkt) abgeschlossen. Dies ist jedoch optional.

Ein Skript kann es erforderlich machen, dass eine Folge von zwei und mehr Anweisungen ausgeführt werden muss. In diesem Fall werden die betreffenden Anweisungen in einen Anweisungsblock eingeschlossen.

Der Anweisungsblock wird durch geschweifte Klammern begonnen und beendet. Die Blöcke können auch verschachtelt sein, um z.B. eine bedingte Anweisung innerhalb einer Funktion zu beschreiben. Dies bedeutet, ein geklammerter Anweisungsblock kann wiederum einen geklammerten Anweisungsblock beinhalten.

```
{
  // Anweisung1;
  // Anweisung2;
  // Anweisung3;
}
```

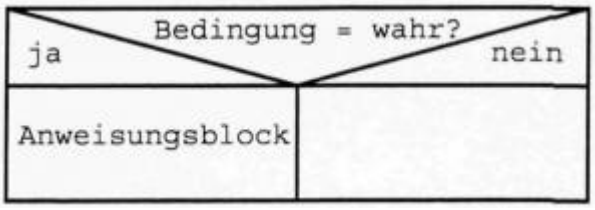
Je nachdem, wie viele alternativ abzuarbeitende Anweisungsblöcke zu einer Bedingung bzw. Bedingungskombination existieren, wird von einfacher, alternativer, mehrstufiger oder mehrseitiger Bedingung gesprochen. Einen Überblick über die verschiedenen Auswahlmöglichkeiten gibt Ihnen die folgende Abbildung.



Auswahl

Im einfachsten Fall soll ein Anweisungsblock nur dann ausgeführt werden, wenn eine Bedingung erfüllt ist. Eine Bedingung wird durch einen Ausdruck definiert, der als Ergebnis true oder f alse (wahr oder falsch) liefert. Die einfache Bedingung ist dadurch gekennzeichnet, dass nach einer Bedingungsabfrage eine oder mehrere Anweisungen (Anweisungsblock) ausgeführt werden oder nicht. Wenn die Bedingung erfüllt ist, dann werden die Anweisungen ausgeführt. Ist die Bedingung nicht erfüllt, werden sie nicht ausgeführt. In JavaScript wird dieses Konstrukt mit der if-Anweisung gebildet:

```
if (Bedingung)
{
  Anweisungsblock;
}
```



Das Schlüsselwort `if` leitet die einfache Bedingung ein. Die Bedingung selbst steht in runden Klammern. Als Bedingung ist ein logischer Ausdruck anzugeben.

Es wird geprüft, ob der Ausdruck den Wert `true` oder `false` zurückliefert, d.h., ob die nachfolgenden Anweisungen innerhalb der geschweiften Klammern ausgeführt werden sollen.

Wenn der Ausdruck den Wert `false` hat, werden alle Anweisungen innerhalb des Anweisungsblocks ignoriert.

Nach dem Ende des Anweisungsblocks ist die if-Anweisung beendet, und alle weiteren Anweisungen werden unabhängig von der Bedingung abgearbeitet.

Beispiel

```
var Angabe, Einheit;
Angabe = "Gewicht";
if (Angabe == "Gewicht")
{
 Einheit = "kg";
}
```

Wenn die Variable Angabe die Zeichenkette Gewicht enthält, wird der Variablen Einheit für die Angabe des Gewichts die Maßeinheit kg zugewiesen.

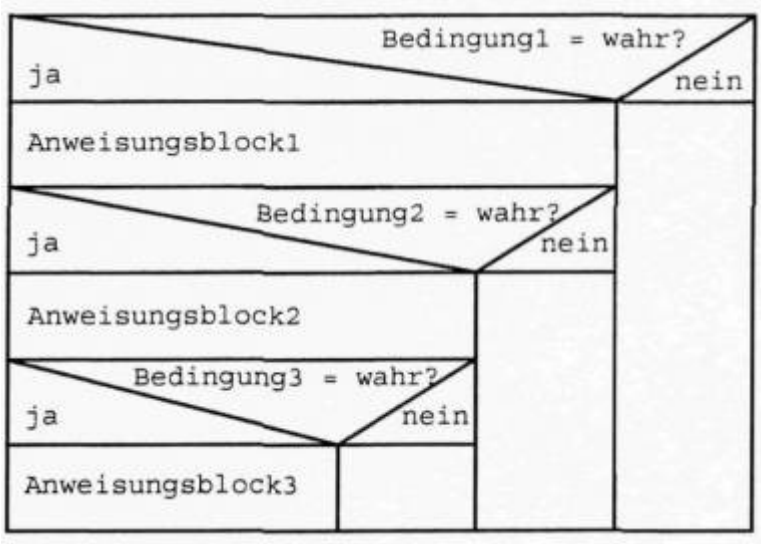
Soll in Abhängigkeit der Bedingung nur eine Anweisung ausgeführt werden, werden die geschweiften Klammern nicht benötigt:

```
if (Angabe == "Gewicht"; Einheit = "kg";
```

Verschachtelte if-Anweisungen

Die Ausführung eines Anweisungsblocks oder einer Anweisung kann von mehreren Bedingungen in mehreren Stufen abhängig sein. Hierzu werden die Bedingungsabfragen ineinander geschachtelt. Ist die erste Bedingung erfüllt, wird kontrolliert, ob auch die nächste Bedingung zutrifft. Trifft eine Bedingung nicht zu (f alse), wird die gesamte verschachtelte Auswahl verlassen.

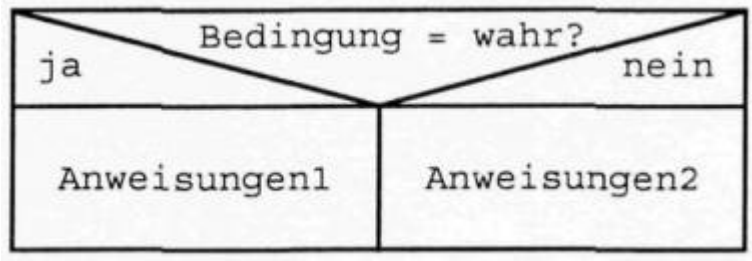
```
if (Bedingung1)
{                              // erste Stufe
  Anweisungsblock 1;
  if (Bedingung2)
  {                            // zweite Stufe
    Anweisungsblock 2;
    if (Bedingung3)
    {                          // dritte Stufe
      Anweisungsblock 3;
    }
  }
}
```



if-else-Anweisung

Die if-else-Anweisung beschreibt nicht nur, was passieren soll, wenn eine Bedingung erfüllt ist, sondern enthält auch die Anweisungen für den Fall, dass sie nicht erfüllt ist. Dazu wird nach dem Anweisungsblock der if-Anweisung ein weiterer Anweisungsblock hinzugefügt, der durch das Schlüsselwort eise eingeleitet wird.

```
if (Bedingung)
{
  Anweisungen1;
}
else
{
  Anweisungen2;
}
```



Beispiel

```
if (Angabe  ==  "Gewicht"){
 Einheit = "kg";
}
else {
 Einheit = "cm";
}
```

Wenn die Variable Angabe die Zeichenkette Gewicht enthält, wird der Variablen Einheit für die

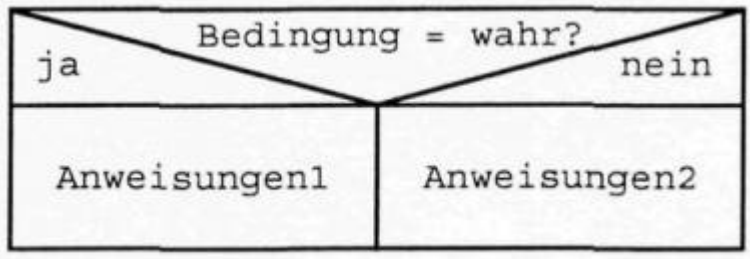
Angabe des Gewichts die Maßeinheit kg zugewiesen. Ansonsten erhält die Variable Einheit die Maßeinheit cm für die Angabe einer Länge.

Mehrstufige if-else-Anweisungen

Bei einer mehrstufigen Bedingung wird einer von mehreren Anweisungsblöcken in Abhängigkeit von verschiedenen Bedingungen ausgeführt. Wenn die erste Bedingung erfüllt ist, wird der erste Anweisungsblock ausgeführt. Ansonsten wird von mehreren Anweisungsblöcken derjenige durchgeführt, für den die Bedingung zutrifft. Wir können die mehrstufige Bedingung verwenden, wenn es mehr als zwei Auswahlmöglichkeiten gibt.

Die Einleitung der jeweils nächsten Bedingung erfolgt durch `else if`.

```
if (Bedingung)
{
  Anweisungen1;
}
else
{
  Anweisungen2;
}
```



Beispiel

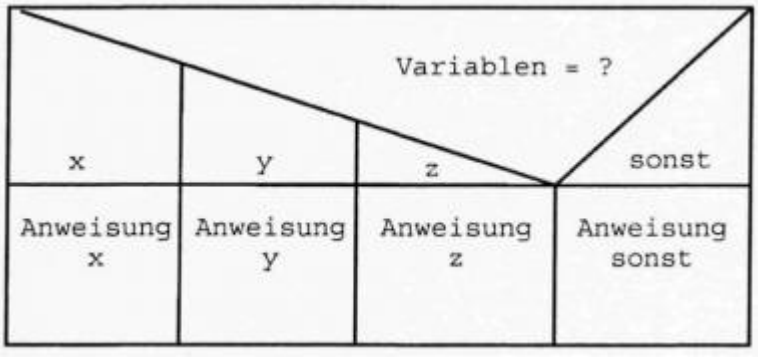
```
if   (Angabe == "Gewicht"){
  Einheit = "kg";
}
else if (Angabe == "Länge"){
  Einheit = "cm";
}
else
{
 Einheit = "s";
}
```

Wenn die Variable `Angabe` die Zeichenkette `Gewicht` enthält, wird der Variablen `Einheit` für die Angabe des Gewichts die Maßeinheit kg zugewiesen. Beinhaltet die Variable `Angabe` den Wert `Länge`, erhält die Variable `Einheit` den Wert `cm`. Treffen die vorherigen Bedingungen nicht zu, erhält die Variable `Einheit` die Maßeinheit `s` für die Zeitangabe.

Mehrseitige Bedingung

Bei einer Fallauswahl, die auch Selektion heißt, wird der Wert einer Variablen ausgewertet und in Abhängigkeit von diesem Wert eine Anweisung bzw. ein Anweisungsblock ausgeführt. Die Variable, deren Inhalt geprüft werden soll, wird auch als Selektor bezeichnet. Für diese Bedingung verwenden wir die Schlüsselwörter `switch` und `case`.

```
switch (Selektor)
{
  case Wert1: Anweisungsblock; break;
  case Wert2: Anweisungsblock; break;
  ...
  default:    Anweisungsblock; break;
}
```



Beispiel

```
switch (Angabe)
{
  case "Gewicht': Einheit = "kg"; break;
  case "Länge"   : Einheit = "cm"; break;
  case "Zeit"    : Einheit = "s";  break;
  default   : Einheit = "";      break;
}
```

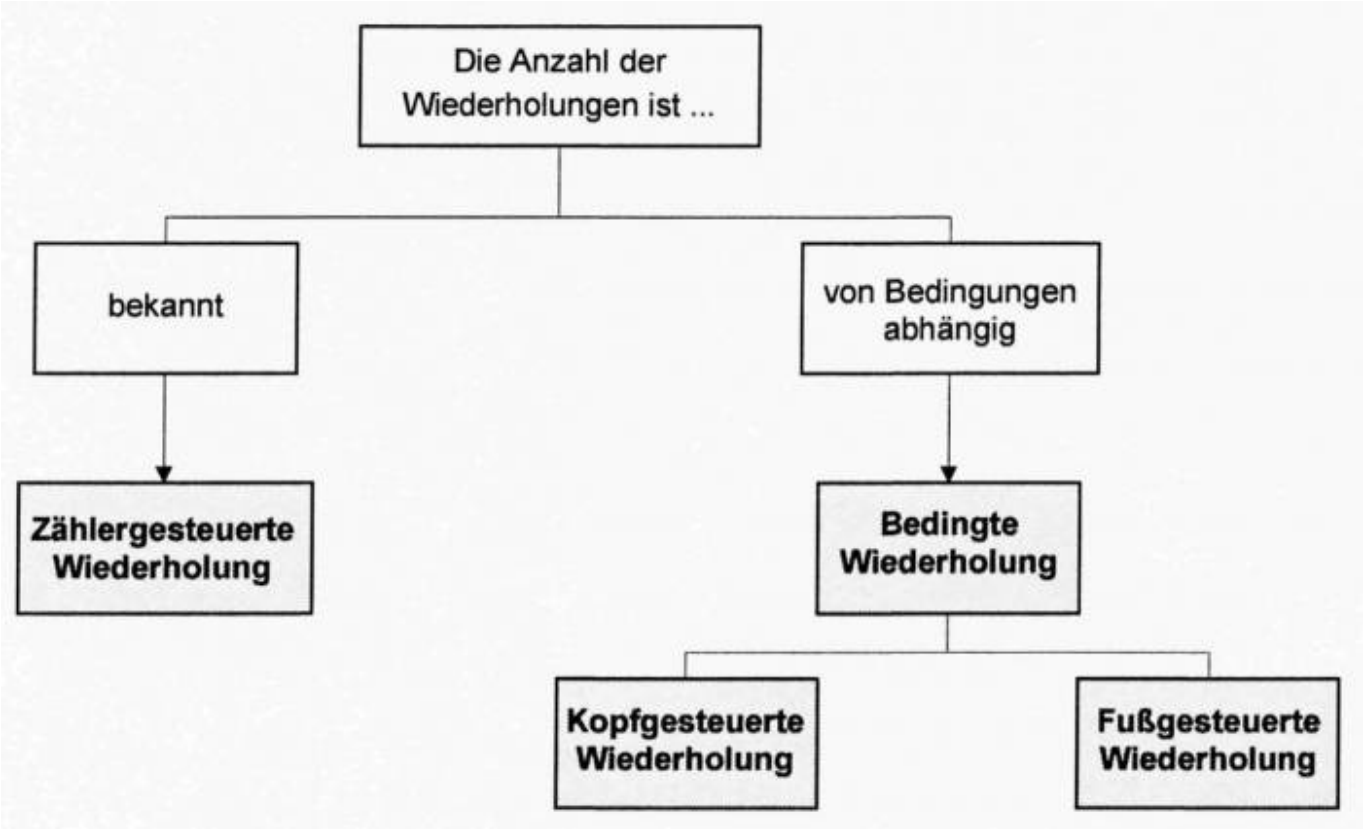
In Abhängigkeit vom Wert der Variablen `Angabe` werden die einzelnen Wertzuweisungen für die Variable `Einheit` durchgeführt. Hat die Variable `Angabe` den Wert `Gewicht`, erhält die Variable `Einheit` den Wert kg usw. Trifft keine der drei `case`-Abfragen zu, wird der Variablen `Einheit` über den `default`-Zweig eine leere Zeichenkette zugewiesen.

Wiederholung

Oftmals müssen die gleichen Anweisungen mehrmals in einer Schleife wiederholt werden. Meistens ist dabei auch nicht vorhersehbar, wie oft diese Schleife ausgeführt werden soll.

Eine Schleife besteht aus einer Schleifensteuerung und einem Schleifenkörper. Die

Schleifensteuerung gibt an, wie oft oder unter welcher Bedingung die Anweisungen abgearbeitet werden. Im Schleifenkörper sind die zu wiederholenden Anweisungen enthalten. Es gibt verschiedene Arten von Wiederholungen.



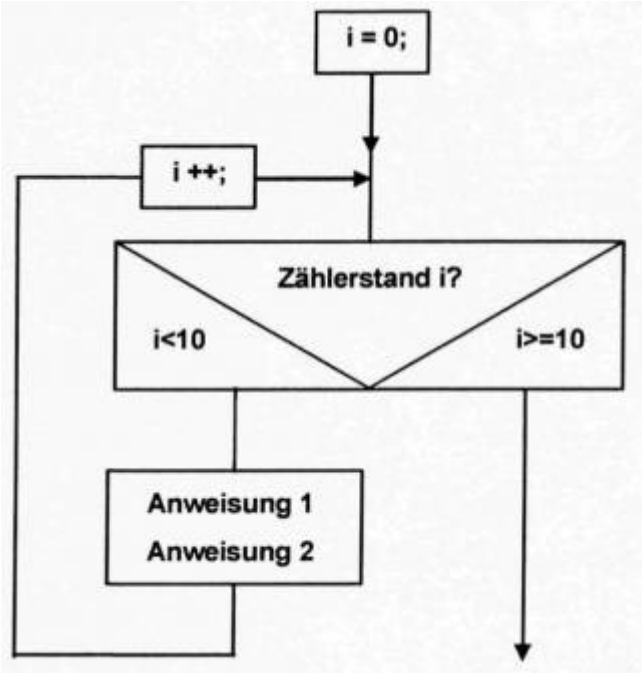
Die Wiederholungen werden entweder gezählt, wenn die Anzahl bekannt ist, oder in Abhängigkeit einer Bedingung ausgeführt. Welche Schleifenart verwendet werden sollte, hängt davon ab, wie oft die Schleife durchlaufen werden soll.

Zählergesteuerte Wiederholung

Eine zählergesteuerte Wiederholung realisieren wir über eine `for`-Schleife. Die Schleife wird mit dem Schlüsselwort `for` eingeleitet. Diese Einleitung wird von drei eingeklammerten Ausdrücken gefolgt. Der erste Ausdruck dient der Initialisierung von Variablen. Über den zweiten Ausdruck legen wir die Abbruchbedingung der Schleife fest. Im dritten Ausdruck legen wir die Schrittweite des Zählers fest.

Eine `for`-Schleife hat den folgenden Aufbau:

```
for(Initialisierung; Bedingung; Aktualisierung)
{
  Anweisung1;
  Anweisung2;
}
```



Beispiel for.html

Es wird der Wert die Zählvariable und somit die Anzahl der Schleifendurchläufe ausgegeben.

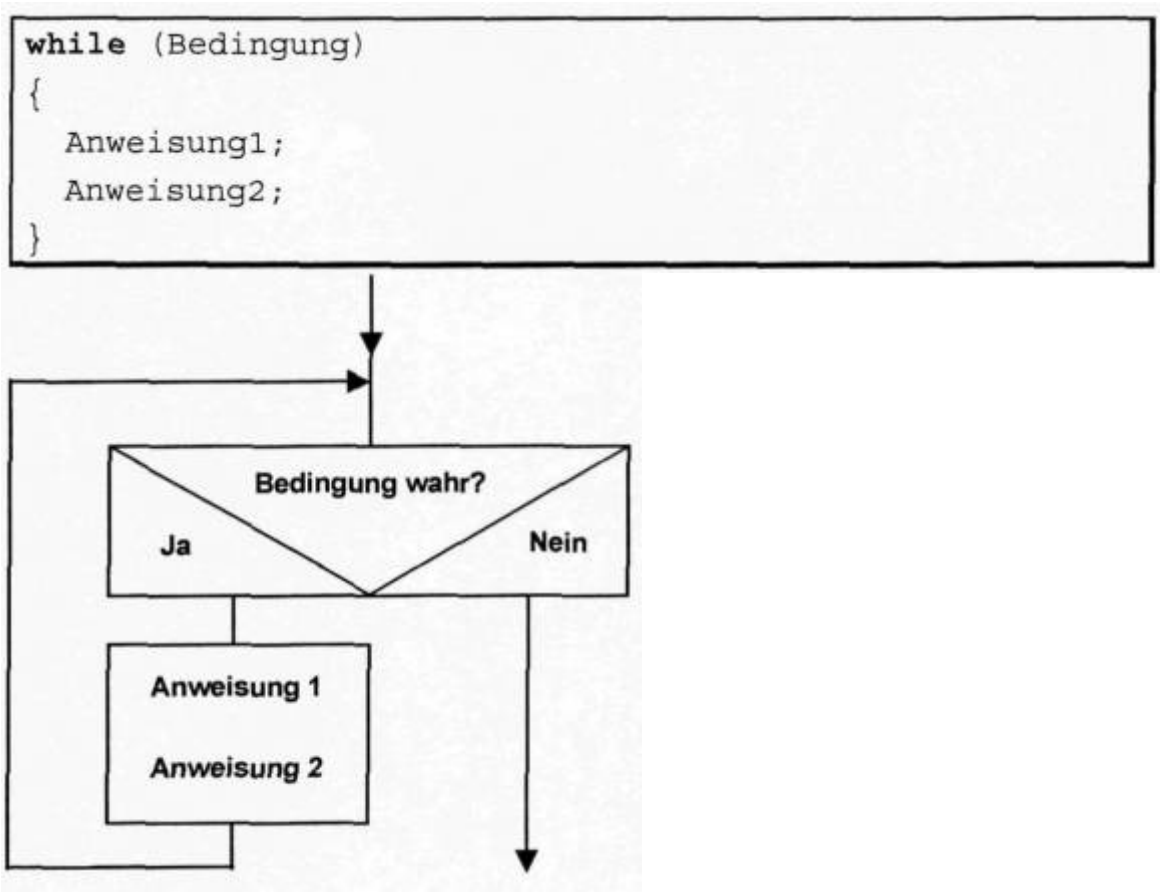
for.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>for-Schleifen</title>
</head>
<body>
  <h3>Die for-Schleife</h3>
  <script type="text/javascript">
    for(i = 0; i < 10; i++)
    {
      document.write("i: ", i, "; Durchlauf: ", i+1, "<br>");
    }
  </script>
</body>
</html>
```

Kopfgesteuerte Wiederholung

In der while-Anweisung wird die Ausführung von Anweisungen von der Gültigkeit eines beliebigen Ausdrucks abhängig gemacht. Der Ausdruck liefert einen booleschen Wert zurück. Ist der Wert true, werden die folgenden Anweisungen ausgeführt. Bei dem Rückgabewert f alse wird die Ausführung nach der while-Anweisung fortgesetzt. Der Ausdruck wird in jedem Durchlauf zu Beginn der while-

Anweisung ausgewertet.



Das Schlüsselwort `while` leitet die kopfgesteuerte Wiederholung ein. Es folgt in runden Klammern ein Ausdruck, der einen booleschen Wert liefert. Ist dieser true, werden die Anweisungen im Schleifenkörper ausgeführt. Nach dem Ausdruck folgt eine Anweisung oder ein ganzer Anweisungsblock.

Nachdem das Programm die Anweisungen im Schleifenkörper ausgeführt hat, wird der Ausdruck am Anfang der Schleife erneut geprüft. Falls dieser den Wert `false` zurückliefert, wird der Schleifenkörper übergangen und die erste Anweisung nach der Schleife durchgeführt. Bei Rückgabe von `true` wird der Schleifenkörper erneut ausgeführt.

Eine `while`-Schleife wird nicht in jedem Fall durchlaufen. Liefert der Ausdruck gleich zu Beginn den Wert `false`, wird keine Anweisung der Schleife ausgeführt.

Beispiel while.html

while.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>while-Schleifen</title>
</head>
<body>
  <h3>Die while-Schleife</h3>
  <script type="text/javascript">
```

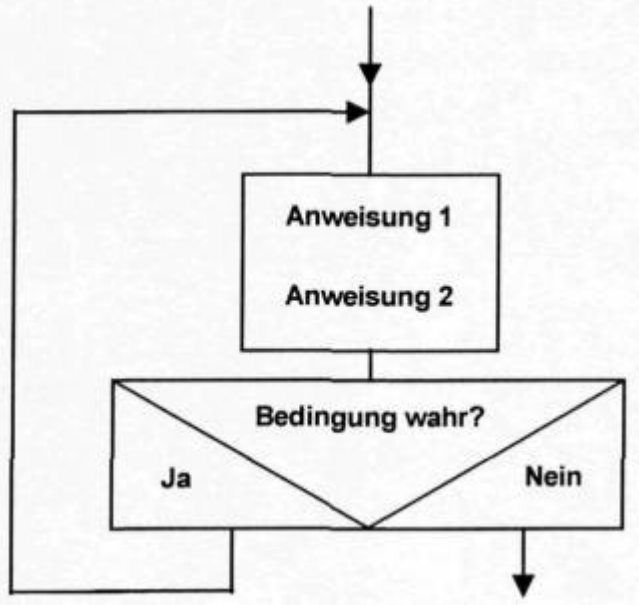
```
    var i = 0;
    var count = 0;
    while (i < 10)
    {
      document.write("i: ", i, "; Durchlauf: ", ++count, "<br>");
      i++;
    }

  </script>
</body>
</html>
```

Fußgesteuerte Wiederholung

Bei einer fußgesteuerten Schleife, auch do-while-Schleife genannt, wird der Anweisungsblock in jedem Fall mindestens einmal ausgeführt. Erst am Ende der Schleife wird getestet, ob der Durchlauf wiederholt wird oder nicht.

```
do
{
  Anweisung1;
  Anweisung2;
} while (Bedingung)
```



Der Bedingungsausdruck, der erfüllt werden soll, befindet sich am Ende der Schleife. Der Schleifenkörper wird ausgeführt, solange der Bedingungsausdruck am Ende der Schleife den Wert true liefert. Ist die Bedingung schon nach dem ersten Durchlauf nicht erfüllt, wird die Schleife dennoch mindestens einmal ausgeführt.

Beispiel dowhile.html

Im folgenden Beispiel wird die Ausgabeanweisung genau einmal ausgeführt, da der Ausdruck der do-while-Schleife bereits bei der ersten Auswertung den Wert `false` liefert.

dowhile.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>do-while-Schleife</title>
</head>
<body>
  <h3>Die do-while-Schleife</h3>
  <script type="text/javascript">
    var i = 5;
    var count = 0;
    do
    {
      document.write("i: ", i, "; Durchlauf: ", ++count, "<br>");
      i++;
    } while (i < 5)
  </script>
</body>
</html>
```

Scheifensteuerung

Die `break-` und `continue-`Anweisungen bieten die Möglichkeit, einen Schleifendurchlauf vorzeitig abzubrechen. Das Schlüsselwort `break` stoppt den aktuellen Schleifendurchlauf und beginnt mit der Anweisung, die der abgebrochenen Schleife unmittelbar folgt. Auch `continue` beendet einen aktuellen Schleifendurchlauf, beendet aber die Schleife nicht, sondern fährt mit dem nächsten Schleifendurchlauf fort.

Im Gegensatz zur `break`-Anweisung kann ein falsch platziertes continue zu Endlosschleifen führen. Deshalb sollten wir es mit Vorsicht benutzen.

Beispiel continue-break.html

Das Beispiel zeigt die Auswirkungen beider Schlüsselwörter. Zuerst wird mit continue vorzeitig ein neuer Schleifendurchlauf erzwungen. Im zweiten Teil des Beispiels wird mit break eine while-Schleife vorzeitig verlassen.

continue-break.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
```

```
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>continue und break</title>
</head>
<body>
  <h3>continue</h3>
  <script type="text/javascript">
    var count = 0;
    for (i = 0; i < 5; i++)
    {
      if (i == 2)
        continue;
      document.write("i: ", i, "; Ausgabe: ", ++count, "<br>");
    }
  </script>

  <h3>break</h3>

  <script type="text/javascript">
    var i = 0;
    var count = 0;
    while (i < 5)
    {
      if (i == 2) break;
      document.write("i: ", i, "; Ausgabe: ", ++count, "<br>");
      i++;
    }
  </script>
</body>
</html>
```

Beispiel

Um die Anzahl der möglichen Passworteingaben in einem Formularfeld zu begrenzen, können wir eine Schleife verwenden.

```
var Versuch = 0;
do {
  UserEingabe = Passwortabfrage();
  Versuch++;
  if (Versuch == 3) break;
} while (UserEingabe != Passwort)
```

Die `while`-Schleife wird so lange durchlaufen, bis das Passwort richtig eingegeben wurde. Die Abfrage if (Versuch == 3) break; bricht die Schleife jedoch ab, wenn drei Versuche erfolglos waren.

Achte bei der Arbeit mit Schleifen immer darauf, dass das Abbruchkriterium auch eintreten kann. Ansonsten wird der Schleifenkörper unendlich oft ausgeführt oder so lange, bis ein Fehler bei der Zuweisung entsteht. Dies kann z. B. der Fall sein, wenn wir zu einer Zahl einen konstanten Wert

addieren. Bei unendlicher Ausführung wird der Wertebereich dieser Zahl überschritten.

Benannte break- und continue-Anweisungen

JavaScript bietet mit benannten `break`- und `continue`-Anweisungen eine Besonderheit gegenüber anderen Sprachen. Durch das Versehen der Anweisungen mit einer Sprungmarke kann aus Schleifen (auch aus verschachtelten) herausgesprungen werden.

```
var j = 0;
label:
while(j == 0){
  for(j = 1; j < 10; j++)
     break label;
     // oder
     // continue label;
  }
}
//hier geht's weiter
```

Im obigen Beispiel wird während des ersten Durchlaufs der `for`-Anweisung zum Label `label` gesprungen. Da der Sprung über `break` erfolgte, wird die nächste Anweisung nach dem folgenden Block, also nach der `while`-Anweisung, ausgeführt.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_05

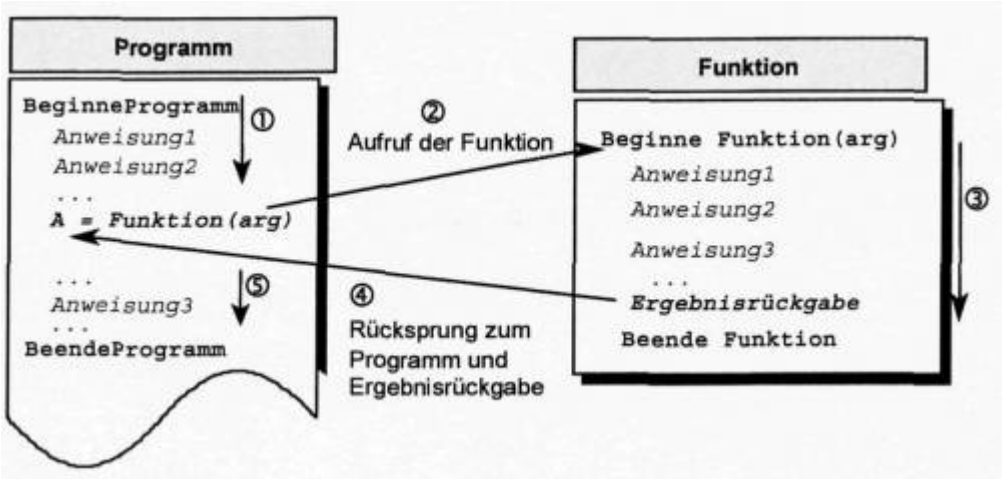
Last update: **2019/05/16 14:49**

11.3.6) Funktionen

Funktionen sind eigenständige Unterprogramme, die von Anweisungen direkt aus dem Skript oder anderen Unterprogrammen aufgerufen werden. Diese beinhalten zum Beispiel immer wiederkehrende Anweisungen. Anstatt wiederholt denselben Code anzugeben, wird er in eine Funktion ausgelagert und kann über den Funktionsaufruf beliebig oft aufgerufen werden.

Grundlagen zu Funktionen

```
var Resultat=0;
Resultat = Addiere(2,3)  // Resultat hat nach dem Aufruf der Funktion den Wert 5
```



Syntax einer Funktion

```
function  Funktionsname ([Parameter1, Parameter2, ...,   Parametern])
{ Anweisungen;
  return Wert;
}
```

Funktionsaufruf

Funktionen werden mit dem Namen aufgerufen, mit dem sie definiert sind. Wenn die Funktion aufgerufen wird, werden die internen Anweisungen abgearbeitet und Ergebnisse über das Schlüsselwort return dem Aufrufer übergeben.

```
Anweisung 1;
Anweisung 2;
Funktionsname();
Anweisung 3;
```

Nach der Funktionsdefinition kann die Funktion per Funktionsaufruf ausgeführt werden. Im Folgenden

wird zwischen Anweisung 2 und Anweisung 3 die Funktionen werden nur dann ausgeführt, wenn sie explizit aufgerufen werden.

Funktionen mit Parametern

Beim Aufruf der Funktion können Werte übergeben werden, mit denen in der Funktion gearbeitet werden soll. Diese Werte werden Parameter genannt. Es können beliebig viele Parameter beim Funktionsaufruf übergeben werden. Die Parameter werden bei der Definition und beim Aufruf durch Kommata getrennt in den runden Klammern angegeben. Die Übergabe der Parameter erfolgt immer als Wert, d. h., Änderungen der Parameterwerte innerhalb der Funktion haben keine Auswirkung auf den Wert des übergebenen Parameters außerhalb der Funktion. Übergeben Sie als Parameter Objekte, können Sie deren Werte in einer Funktion jedoch dauerhaft verändern.

```
function Funktionsname (Parameterl, Parameter2)
{
 // JavaScript - Anweisungen return Rueckgabewert;
 Anweisung 1;
 Anweisung 2;
 x = Funktionsname(Parameterl, Parameter2);
 Anweisung 3;
}
```

Kurzvariante

Statt:

```
function Funktionsname (x)
{
 return x*x;
}
```

kann man auch nur schreiben:

```
function Funktionsname (x) x*x;
```

Variable Paramaterliste

Über das Feld `arguments` man ebenfalls auf alle Parameter einer Funktion zugreifen. Der Zugriff erfolgt dabei allerdings über den Index des Parameters und nicht über den Parameternamen. Die Anzahl der übergebenen Parameter erhält man über `arguments.length`. Auf diese Weise können einer Funktion eine variable Parameterliste übergeben werden.

Beispiel: args_funktion.html

args_funktion.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
<title>Funktion mit unbestimmter Anzahl an Parametern</title>
</head>
<body>
<p>Die Funktion
<pre>function test(par1, par2) {

}</pre> wird mit zusätzlichen Parametern aufgerufen:
<pre>test(10, 11, 12, 13, 14, 15, 16);</pre>

<p> </p>
<p><b><u>Ergebnis:</u></b></p>

    <script type="text/javascript">

     function test(par1, par2)
     {
       document.write("Parameter 1 mit dem Wert: " + par1 + "<br>");
       document.write("Parameter 2 mit dem Wert: " + par2 +
"<br><br>");

       for(i = 2; i < arguments.length; i++)
         document.write("Zusätzlicher Parameter " + (i - 1) + " mit dem
Wert: " + arguments[i] + "<br>");
     }

     test(10, 11, 12, 13, 14, 15, 16);

    </script>
  </body>
</html>
```

Beispiel: funktion.html

funktion.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Funktionsaufrufe</title>
</head>
<body>
  <h3>Berechnen einer Kreisfläche</h3>
  <p>nach der Formel: <code>A = &pi; / 4 * d<sup>2</sup></code></p>

  <script type="text/javascript">
```

```
    function flaeche(d)
    {
      var PI = 3.141592653589793;
      var A  = (PI / 4) * d * d;
      return A;
    }

    var ergebnis = 0;
    var durchmesser = 10;
    ergebnis = flaeche(durchmesser);
    document.write("d=", durchmesser, "m;<br>A=", ergebnis,
"m<sup>2<\/sup><br>");

  </script>
</body>
</html>
```

Beispiel: funktion_schleife.html

funktion_schleife.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Funktionsaufrufe</title>
</head>
<body>
  <h3>Berechnen mehrerer Kreisflächen</h3>
  <p>nach der Formel: <code>A = &pi; / 4 * d<sup>2</sup></code></p>

  <script type="text/javascript">

    function flaeche(d)
    {
      return Math.PI / 4 * d * d;
    }
    for (durchmesser = 0; durchmesser <= 100; durchmesser +=5)
      document.write("d=", durchmesser, "m; A=", flaeche(durchmesser),
"m<sup>2<\/sup><br>");

  </script>
</body>
</html>
```

Weitere Möglichkeiten für die Definition von Funktionen

Funktionsdefinition von Bedingungen abhängig machen

Eine Funktionsdefinition kann auch von einer Bedingung abhängig gemacht werden. Auf diese Weise können Sie z. B. Funktionen mit gleichen Namen, aber unterschiedlichen Funktionalitäten definieren.

Beispiel: if_funktion.html

Die Wirkungsweise der Funktion `operation()` wird vom Inhalt der Variablen name abhängig gemacht. Auf diese Weise können Sie die Funktionalität eines Programms auf andere Weise vom Wert einer Variablen abhängig machen.

if_funktion.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
<title>Funktionsdefinition von Bedingung abhängig machen</title>
</head>
<body>
<p>Anhand des Variablenwerts <code>name="Addition"</code> soll
<br>die Funktion definiert werden, die zwei Zahlen addiert.
<p>Das Ergebnis sollte also lauten:
<pre>operation(10, 11) = 21;</pre>


<p> </p>
<p><b><u>Ergebnis:</u></b></p>

    <script type="text/javascript">

     name = "Addition";
     if(name == "Addition") {
       function operation(zahl1, zahl2) { return zahl1 + zahl2; }
     }
     else {
       function operation(zahl1, zahl2) { return zahl1 - zahl2; }
     }
     document.write(name, " -> operation(10, 11) = ", operation(10,
11));

    </script>
<hr>
<p>* erst ab JavaScript 1.5
  </body>
</html>
```

Funktionen in Ausdrücken definieren

Durch die Definition von Funktionen in Ausdrücken haben Sie eine weitere Möglichkeit, die Programmausführung dynamisch zu ändern. Statt eines Funktionsnamens wird die gesamte Funktionsdefinition im Ausdruck angegeben. Der einzige Unterschied zur üblichen Funktionsdefinition liegt im Weglassen des Funktionsnamens. Deshalb werden diese Funktionen auch als anonyme Funktionen bezeichnet.

Beispiel: ausdr_funktion.html

In der ersten Anweisung wird die Funktionsdefinition einer Variablen zugewiesen. Der Variablenname steht somit für den Funktionsnamen und kann wiederum in Ausdrücken verwendet werden. Im zweiten Anwendungsfall wird der Funktion TueEtwas () als Parameter eine Funktion übergeben. Der übergebenen Funktion wird in der return-Anweisung der Wert 10 als Parameter übergeben. Der Funktionsaufruf der Funktion TueEtwas () beinhaltet im ersten Fall die vollständige anonyme Funktionsdefinition. Im zweiten Fall wird die Variable quadrat übergeben, die wiederum auf eine Funktionsdefinition verweist.

ausdr_funktion.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
<title>Funktion in Ausdrücken definieren</title>
</head>
<body>
<pre>
     var quadrat = function (zahl) { return zahl * zahl; }
     document.write(quadrat(10));

     function TueEtwas(f)
     {
       return f(10);
     }

     document.write(TueEtwas(function(value) { return value * value;
}));
     document.write(TueEtwas(quadrat));
</pre>

<p> </p>
<p><b><u>Ergebnis:</u></b></p>

    <script type="text/javascript">

     var quadrat = function (zahl) { return zahl * zahl; }
     document.write(quadrat(10) + "<br>");

     function TueEtwas(f)
```

```
    {
      return f(10);
    }

    document.write(TueEtwas(function(value) { return value * value; })
+ "<br>");
    document.write(TueEtwas(quadrat));

  </script>

 </body>
</html>
```

Lokale und globale Variablen

Die Variable d aus den vorherigen Beispielen besitzt einen Gültigkeitsbereich, der auf die Funktion beschränkt ist, d. h., außerhalb der Funktion kann nicht auf den Wert von d zugegriffen werden. Solche Variablen werden als lokale Variablen bezeichnet.

Globale Variablen, auf die jederzeit zugegriffen werden kann, werden außerhalb von Funktionen deklariert.

Beispiel: funktion_lokal_global.html

funktion_lokal_global.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Lokale und globale Variablen</title>
</head>
<body>
  <h3>Lokale und globale Variablen</h3>
  <script type="text/javascript">

    var x = 10;
    var y = 10;
    var z = 1;

    function Produkt(x, y)
    {
      var Ergebnis = x * y;
      document.write("<br>Werte innerhalb der Funktion: x = " + x + ";
y = " + y + "; z = " + z + "<br>");

      document.write("Ergebnis in der Funktion: x * y = " + Ergebnis +
"<br>");
      return Ergebnis;
```

```
    }

    document.write("Werte außerhalb der Funktion: x = " + x + "; y = "
+ y + "; z = " + z + "<br>");

    var MeinProdukt = x * y * Produkt(3, 4);
    document.write("<br>Ergebnis außerhalb der Funktion: x(global) *
y(global) * (Ergebnis lokal) = " + MeinProdukt);

  </script>
</body>
</html>
```

Beachte: Viele Fehler in Skripts ergeben sich dadurch, dass globale und lokale Variablen durcheinander gebracht werden. Sie können beispielsweise den Wert einer globalen Variablen in einer Funktion verändern, obwohl Sie unter dem Namen eigentlich eine lokale Variable vermutet haben. Daher sollten zur besseren Übersicht alle lokalen Variablen einer Funktion mit einer var-Anweisung deklariert werden.

```
function MeineFunktion(x,y)
{
 var  LokaleVariable;
 ...        //   JavaScript   -  Anweisungen
}
```

Das Beispiel deklariert die Funktion `MeineFunktion()` mit den drei lokalen Variablen `x,y` und `LokaleVariable`. Generell zeugt es von gutem, strukturiertem Programmierstil, wenn Funktionen völlig ohne globale Variablen auskommen. Solche Funktionen sind auch in hohem Maße wiederverwendbar, da ihre Verarbeitungen nur von den Parametern abhängen, die ihnen beim Aufruf übergeben werden.

Vordefinierte Funktionen in JavaScript

Die folgenden Funktionen sind in JavaScript bereits vorhanden. Die Funktionsnamen sind reserviert und dürfen deswegen nicht für eigene Funktionsnamen verwendet werden.

`eval(Zeichenkette)`	Die übergebene Zeichenkette wird als JavaScript-Code ausgewertet und das Ergebnis der Ausführung zurückgegeben. Wenn Sie beispielsweise die Anweisung `document.write(eval("3 + 4 * 6"));` notieren, wird am Bildschirm das Ergebnis des Rechenausdrucks ausgegeben.
`encodeURI, decodeURI, encodeURIComponent decodeURIComponent`	Diese Funktionen dienen der Ersetzung von Zeichen in URLs bzw. in Zeichenketten. Der Aufruf von `decodeURI("t\"est")` ergibt z. B. das Ergebnis `t"est`.
`escape(Zeichenkette) unescape(Zeichenkette)`	Über diese Funktionen können Sie Zeichenketten konvertieren, die Steuerzeichen (ASCII-Wert von 0 bis 31) enthalten. Diese Funktionen sollten nicht mehr genutzt werden. Greifen Sie stattdessen auf die Funktionen `encodeXXX` und `decodeXXX` zurück.
`isNaN(Wert)`	Mit der Funktion `isNaN()` (*is not a number* = ist keine Zahl) können Sie testen, ob ein übergebener Wert eine ungültige Zahl ist. Ist der Wert eine Zeichenkette, wird `true`, ist er eine Zahl, wird `false` zurückgeliefert.
`isFinite(Wert)`	Es wird getestet, ob der übergebene Wert innerhalb des Zahlenbereichs liegt, den JavaScript verarbeiten kann. Liegt der Wert außerhalb oder ist der Wert eine Zeichenkette, wird `false` zurückgeliefert, andernfalls `true`.
`Number(Objekt) String(Objekt)`	Über diese Funktionen können Sie ein Objekt in eine Zahl oder eine Zeichenkette konvertieren.
`parseFloat(Zeichenkette)`	Wandelt eine Zeichenkette in eine Dezimalzahl um und gibt diese als numerischen Wert zurück. Diese Funktion gibt `NaN` (not a number = keine Zahl) zurück, wenn die Zeichenkette mit einem nicht numerischen Zeichen beginnt. Die Konvertierung der Zeichenkette in eine Zahl endet mit dem ersten ungültigen Zeichen oder dem Ende der Zeichenkette.
`parseInt(Zeichenkette)`	Wandelt eine Zeichenkette in eine ganze Zahl um und gibt diese als numerischen Wert zurück. Diese Funktion gibt `NaN` (Not a Number = keine Zahl) zurück, wenn die Zeichenkette mit einem nicht numerischen Zeichen beginnt. Die Konvertierung der Zeichenkette in eine Zahl endet mit dem ersten ungültigen Zeichen oder dem Ende der Zeichenkette.

Beispiel: parseFloat_parseInt.html

Die Anwendung der beiden Funktionen `parseFloat()` und `parseInt()` soll demonstriert werden. Als Eingabewerte werden Zeichenketten verwendet, die korrekte Zahlenwerte und gemischte Werte besitzen.

parseFloat_parseInt.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Existierende Funktionen in JavaScript</title>
</head>
<body>
  <h3>parseFloat() und parseInt()</h3>
  <script type="text/javascript">

    var a = "50";
```

```
    var b = "5.6abc";

    document.write('<p>a = "', a, '"; b = "', b, '";');
    document.write('<p>c = a + b');
    document.write('<br>c = ',a + b);

    document.write('<p>c = parseInt(a) + parseFloat(b)');
    document.write('<br>c = ',parseInt(a) + parseFloat(b));

    document.write('<p>c = parseFloat(a) + parseInt(b)');
    document.write('<br>c = ',parseFloat(a) + parseInt(b));

  </script>
</body>
</html>
```

Beachte: In HTML gibt es keine Eingabefelder, die speziell für Zahlen genutzt werden können. Die Eingaben werden immer als Zeichenkette betrachtet. Möchte man mit den Eingaben rechnen, so lässt sich mit isNaN z. B. überprüfen, ob die Eingabe als Zahl interpretiert werden kann. Danach habt man die Möglichkeit, die Zeichenkette über parseFloat oder parseint in eine Zahl des entsprechenden Typs umzuwandeln.

Debuggen von Funktionen

Es lässt sich nie ganz vermeiden, dass Fehler beim Erstellen und Ausführen eines JavaScripts auftreten. In diesem Abschnitt soll erläutert werden, wie Sie JavaScript-Fehler ausfindig machen und beseitigen. Die Art und Weise dieser Fehlersuche wird Debugging (deutsch: Entwanzen) genannt.

Beispiel: debug_variable.html

debug_variable.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Debugging</title>
</head>
<body>
  <script type="text/javascript">
    var +variable = true;
  </script>
</body>
</html>
```

In diesem Beispiel wird der Name der Variablen absichtlich falsch geschrieben. Ihr wird das Sonderzeichen [vorangesetzt, das nicht in Variablennamen genutzt werden darf.

Firefox Fehler-Konsole

Für das Debuggen stellt der Browser Firefox eine Fehler-Konsole zur Verfügung, mit der Fehler in der CSS-Formatierung als auch in JavaScript sichtbar werden können.

Wähle daszu das Menü EXTRAS → FEHLER-KONSOLE.

Es öffnet sich ein neues Fenster, in dem die Meldungen des JavaScript-Interpreters anzeigt werden. Im Fenster wird eine Meldung angezeigt, dass der Variablenname nicht angegeben wurde (Fehler: missing variable name). Außerdem wird der Programmcode mit der entsprechenden Zeilennnummer angezeigt. Zur besseren Lokalisierung des Fehlers wird die Stelle noch mit einem Pfeil markiert.

Die Konsole lässt auch Eingaben von JavaScript-Anweisungen zu, sodass man die Zuweisung der Variablen auch direkt im Eingabefeld angeben und mit der Schaltfläche „Evaluieren“ überprüfen kann.

Firebug

Mit Hilfe der Firefox-Addons „Firebug“ erhält man eine sehr komfortable Möglichkeit, Seiten auf ihre Struktur und ihre Korrektheit hin zu durchleuchten.

Typische Fehler

Zuweisung oder Vergleich? Das einfache Gleichheitszeichen ist der Zuweisungsoperator. Um zwei Werte zu vergleichen, muss jedoch das Gleichheitszeichen zweimal angegeben werden. Da keine Fehlermeldung angezeigt wird, tritt dieser Fehler recht häufig auf.

```
var Test = 0;
if (Test = 1)
document.write 'Test war in Ordnung!');
```

Die fehlerhafte Verwendung des Zuweisungsoperators erzeugt keine Fehlermeldung. Der Variablen Test wird der Wert l zugewiesen, d. h., es wird kein Vergleich, sondern eine Zuweisung durchgeführt. Die `if`-Anweisung wertet alle Werte ungleich null zu `true` aus, sodass die Anweisung ausgeführt wird. Richtig muss es heißen:

```
var Test = 0;
if (Test == 1)
document.write 'Test war in Ordnung!');
```

Groß oder klein? JavaScript ist `case-sensitive`, d.h., die Groß- und Kleinschreibung wird in allen Variablennamen und Bezeichnern unterschieden. Bei größeren Skripten mit vielen Funktionen kann es passieren, dass z. B. ein Funktionsaufruf falsch geschrieben wird.

```
function AusgabeImBrowser()
{
```

```
 // JavaScript-Anweisungen
}
document.write(ausgabeImBrowser()) ;
```

Die Browser werden diesen Aufruf als Fehler ausgeben, da die aufgerufene Funktion nicht definiert ist. Die Funktion wird in der Definition mit einem großen A geschrieben und hat somit einen anderen Bezeichner.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_06

Last update: **2019/05/16 14:49**

11.3.7) Objekte

Das Konzept des objektorientierten Programmierens hat sich wegen seiner Vorteile schon lange durchgesetzt. Es bildet reale und abstrakte Objekte auf Codestrukturen ab, die leichter verständlich sind und zusammengehörende Daten und Funktionen kapseln. Objekte bieten dem Programmierer wiederverwendbaren Code und erlauben eine bessere Lokalisierung von Fehlem. Objektorientiertes Programmieren erfordert jedoch ein gutes Verständnis der zugrunde liegenden Theorie.

Objekte bestehen aus Eigenschaften und Methoden. Die Eigenschaften sind eine Ansammlung von Werten, die das Objekt näher beschreiben. Methoden sind objekteigene Funktionen, welche die Eigenschaften oder andere Werte ändern. In JavaScript sind die folgenden Schritte zur Verwendung von Objekten notwendig:

- Erstellung der Objektdefinition durch eine Konstruktor-Funktion oder einen Objektinitialisierer
- Hinzufügen von Eigenschaften
- Hinzufügen von Methoden
- Erzeugen von Objekten
- Verwenden der Objekte

Grundlagen von Objekten

Erstellen von Objektdefinitionen über eine Konstruktor-Funktion

Eine Konstruktor-Funktion enthält in ihrem Anweisungsblock die Definition der Eigenschaften und Methoden eines Objekts. Mithilfe des Konstruktors können dann sogenannte Instanzen, d. h. Abbildungen des Objekts, im Speicher erzeugt werden. Der Name des Objekts ergibt sich aus dem Namen der Konstruktor-Funktion.

```
function Konstruktorname(Wertl,Wert2, ...)
{
 this.Eigenschaftl   =  Wertl;
 this.Eigenschaft2   =  Wert2;
}
```

Eine Instanz des Objekts wird mit new und dem Aufruf der Konstruktor-Funktion erzeugt, die dabei über-gebenen Parameter werden von der Konstruktor-Funktion zur Initialisierung der Eigenschaften verwendet.

```
var Instanz = new Konstruktorname(Wertl,Wert2, ...);
```

Zusätzlich können in der Konstruktor-Funktion Methoden zur Bearbeitung des Objekts deklariert werden. Dabei wird der Name der Methode einer meist gleichnamigen JavaScript-Funktion zugeordnet.

Erstellen der Objektdefinition über einen Objektinitialisierer

Während über eine Konstruktor-Funktion ein konkreter Objektname zur Verfügung steht, kann man mit einem Objektinitialisierer (seit JavaScript 1.2) anonyme Objekte definieren. Dazu wird einer Variablen eine Liste von Eigenschaft-Wert-Paaren, die durch einen Doppelpunkt getrennt sind, in geschweiften Klammern zugewiesen. Mehrere Paare werden durch Kommata getrennt. Man kann mit einem Objektinitialisierer auch Methoden mit entsprechenden Funktionen definieren.

Beispiel

Der Objektvariablen `ObjVar` werden die Eigenschaften `Eigenschaft1, Eigenschaft2` usw. zugewiesen, die bereits mit einem Wert vorbelegt sind.

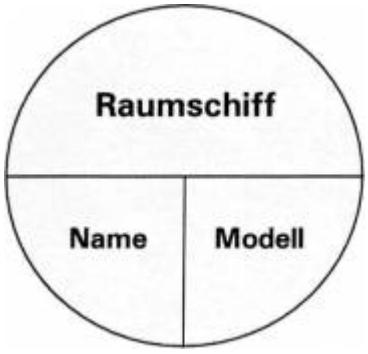
```
var ObjVar = {Eigenschaftl:Wertl,
              Eigenschaft2:Wert2,
              Methodel: new function() {}
};
document.write(ObjVar.Eigenschaftl);
document.write(ObjVar.Methodel);
```

Eigenschaften

Eigenschaften eines Objekts können in jeder Instanz unterschiedliche Werte haben. Instanzen sind somit Vertreter eines Objekts. So ist ein Raumschiff mit dem Namen U.S.S. Enterprise ein Vertreter aller Raumschiffe.

Es soll ein einfaches Objekt für Raumschiffe erstellt werden. Begonnen wird mit einer Konstruktor-Funktion, womit die zwei Eigenschaften Name und Modell für alle Instanzen des Objekts festgelegt werden.

```
function Raumschiff(Schiffsname, Schiffsmodell)
{
 this.Name = Schiffsname;
 this.Modell = Schiffsmodell;
}
```



Über das Schlüsselwort this beziehen Sie sich innerhalb der Konstruktor-Funktion immer auf das

aktuelle Objekt. Die Eigenschaft Name des aktuellen Objekts wird im Beispiel auf den Wert des ersten übergebenen Parameters (Schiffsname) gesetzt, die Eigenschaft Modell auf den Wert des zweiten Parameters (Schiffsmodell).

Mit der Anweisung new können Sie ab sofort Objekte über die Konstruktor-Funktion Raumschiff erzeugen.

```
var  Enterprise = new Raumschiff("U.S.S. Enterprise","NCC-1701");
|
```

Eine Variable mit Namen Enterprise wird deklariert, new erzeugt ein neues Objekt mit dem Konstruktor Raumschiff und übergibt diesem die zwei Werte U.S.S. Enterprise und NCC-1701. Der definierte Konstruktor erzeugt daraufhin das Objekt im Speicher und belegt die Eigenschaften mit den übergebenen Werten. Er liefert auch eine Referenz auf die Speicherstelle zurück, die in der Variablen Enterprise abgespeichert wird. Von nun an existiert das Raumschiffobjekt im Speicher und kann über den Variablennamen Enterprise angesprochen werden:

```
document.write(typeof(Enterprise));  //liefert den Typ - object
```

Um auf die Eigenschaften dieses Objekts zugreifen zu können, wird der Name des Objekts, gefolgt von einem Punkt und dem Namen der Eigenschaft, verwendet. Alternativ können Sie in eckigen Klammern den Namen der Eigenschaft angeben, der in doppelten Anführungszeichen eingeschlossen ist.

```
Objekt.Eigenschaft;
Objekt["Eigenschaft"];
```

In JavaScript wird die Punktnotation für den Zugriff auf Objekteigenschaften und -methoden genutzt. Sie wurde bereits in einigen Anwendungen verwendet, `document.write()` ist z.B. die Funktion `write()` des `document`-Objekts, die Ausgaben in ein HTML-Dokument durchführen kann.

Beispiel: raumschiff.html

Es werden die beiden Raumschiffe Enterprise und Voyager erzeugt. Danach wird auf deren Eigenschaften zugegriffen.

raumschiff.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekte, Eigenschaften und Methoden</title>
</head>
<body>
  <h3>Anlegen zweier Objekte vom Typ Raumschiff</h3>
  <script type="text/javascript">

    function Raumschiff(Schiffsname, Schiffsmodell)
    {
      this.Name = Schiffsname;
```

```
      this.Modell = Schiffsmodell;
    }

    var Enterprise = new Raumschiff("U.S.S. Enterprise", "NCC-1701");
    var Voyager    = new Raumschiff("U.S.S. Voyager", "NCC-74656");

    document.write("<p>", Enterprise.Name, ", Modell:  ",
Enterprise.Modell);
    document.write("<p>", Voyager.Name, ", Modell:  ", Voyager.Modell);

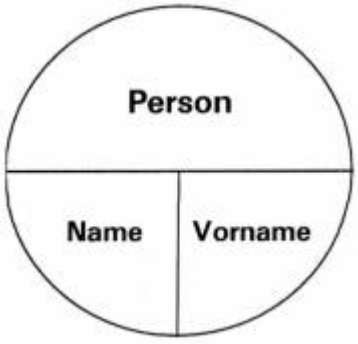
  </script>
</body>
</html>
```

Eigenschaften beschreiben Gemeinsamkeiten von Objekten. Die Werte von Eigenschaften (auch Attribute genannt) des Objekts können bei jeder Instanz unterschiedlich sein. Der erste Schritt, ein Objekt zu definieren, besteht darin, im Anweisungsblock der Konstruktor-Funktion die Eigenschaften zu deklarieren und zu initialisieren. Für jede Eigenschaft wird dazu eine eigene Variable angegeben. Die Anweisung zur Deklaration der Eigenschaft beginnt mit dem Schlüsselwort `this`, gefolgt von einem Punkt und dem Namen der Eigenschaft. Der Eigenschaft wird danach ein Wert zugewiesen, der z. B. über Parameter übergeben werden kann.

Für Eigenschaften gelten die gleichen Regeln wie für Variablen. Sie können alle einfachen Datentypen beinhalten, aber auch Objekte und Funktionen. Der Zugriff erfolgt im Gegensatz zu Variablen außerhalb des Objekts über den Instanz- und den Eigenschaftsnamen. Der Instanzname verweist auf das entsprechende Objekt, der Eigenschaftsname auf die gewünschte Eigenschaft des verwendeten Objekts.

Im übernächsten Beispiel wird dem Objekt Raumschiff die weitere Eigenschaft Kapitaen hinzugefügt. Der Unterschied zu den bisher verwendeten Eigenschaften liegt darin, dass diese Eigenschaft ebenfalls ein Objekt ist. Um einen Kapitän erzeugen zu können, wird daher zuerst ein Objekt benötigt.

```
function Person(PersonName, PersonVorname)
{
 this.Name = PersonName;
 this.Vorname = PersonVorname;
}
```



Über die Konstruktor-Funktion `Person` können Sie ein Objekt erzeugen, das die Eigenschaften `Name` und `Vorname` besitzt.

Beispiel: raumschiff_person.html

raumschiff_person.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekte, Eigenschaften und Methoden</title>
</head>
<body>
  <h3>Die Personen der Raumschiffe</h3>
  <script type="text/javascript">

    function Raumschiff(Schiffsname, Schiffsmodell)
    {
      this.Name = Schiffsname;
      this.Modell = Schiffsmodell;
    }

    function Person(PersonName, PersonVorname)
    {
      this.Name = PersonName;
      this.Vorname = PersonVorname;
    }

    var Enterprise = new Raumschiff("U.S.S. Enterprise", "NCC-1701");
    var kirk = new Person("Kirk", "James Tiberius");
    var spock = new Person("", "Spock");

    var Voyager = new Raumschiff("U.S.S. Voyager", "NCC-74656");
    var kathryn = new Person("Janeway", "Kathryn");

    document.write("<p>", Enterprise.Name, " ", Enterprise.Modell);
    document.write("<br>Person: ", kirk.Vorname, " ", kirk.Name);

    document.write("<p>",Voyager.Name, " ", Voyager.Modell);
    document.write("<br>Person: ",kathryn.Vorname, " ", kathryn.Name);

  </script>
</body>
</html>
```

Zu den definierten Eigenschaften lassen sich weitere hinzufügen. Sie könnten z. B. das Objekt `Person` um die Eigenschaften `Geschlecht` und `Lebensform` erweitern.

Nach der Definition des Objekts Person können Sie den jeweiligen Raumschiffen eine Person als Kapitän zuordnen. Dazu wird das Objekt Raumschiff um die Eigenschaft Kapitaen erweitert.

```
function  Raumschiff(Schiffsname,Schiffsmodell,Kapitaen)
```

```
{
  this.Name = Schiffsname;
  this.Modell = Schiffsmodell;
  this.Kapitaen  =  Kapitaen;
}
```

Die Konstruktor-Funktion `Raumschiff` besitzt jetzt eine Eigenschaft `Kapitaen`. Als Kapitän des Raumschiffs soll eine Person angegeben werden, die aus dem Objekt `Person` erzeugt wurde. Beim Aufruf wird als Argument für die Funktion `Raumschiff()` die Variable `Person` vom Typ `object` übergeben.

Damit ist es möglich, den Vornamen des Raumschiffkapitäns über ein Objekt `Raumschiff` und dessen Referenz `Kapitaen` auf das Objekt `Person` und somit deren Eigenschaft `Vorname` auszulesen, Z.B. `Enterprise.Kapitaen.Name`.

Durch dieses Prinzip können Sie Objekte beliebig tief verschachteln. So kann einer Person über die Eigenschaft Adresse z.B. ein Adressobjekt zugewiesen werden. Auf den Aufenthaltsort könnten Sie dann über Enterprise. Kapitaen.Adresse .Ort zugreifen.

Beispiel: raumschiff_kapitaen.html

raumschiff_kapitaen.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekte, Eigenschaften und Methoden</title>
</head>
<body>
  <h3>Die Kapit&auml;ne der Raumschiffe</h3>
  <script type="text/javascript">

    function Raumschiff(Schiffsname, Schiffsmodell, Kapitaen)
    {
      this.Name = Schiffsname;
      this.Modell = Schiffsmodell;
      this.Kapitaen = Kapitaen;
    }

    function Person(PersonName, PersonVorname, Geschlecht, Lebensform)
    {
      this.Name = PersonName;
      this.Vorname = PersonVorname;
      this.Geschlecht = Geschlecht;
      this.Lebensform = Lebensform;
    }

    var kirk = new Person("Kirk", "James Tiberius", "m&auml;nnlich",
"Mensch");
    var spock = new Person("", "Spock", "m&auml;nlich", "Mensch-
```

```
Vulcanier");
    var Enterprise = new Raumschiff("U.S.S. Enterprise", "NCC-1701",
kirk);

    var kathryn = new Person("Janeway", "Kathryn", "weiblich",
"Mensch");
    var Voyager = new Raumschiff("U.S.S. Voyager", "NCC-74656",
kathryn);

    document.write("<p>",Enterprise.Name, " ", Enterprise.Modell);
    document.write("<br>Kapit&auml;n: ", Enterprise.Kapitaen.Vorname, "
", Enterprise.Kapitaen.Name, ", ", Enterprise.Kapitaen.Geschlecht);
    document.write("<br>Lebensform: ", Enterprise.Kapitaen.Lebensform);

    document.write("<p>",Voyager.Name, " ", Voyager.Modell);
    document.write("<br>Kapit&auml;n: ", Voyager.Kapitaen.Vorname, " ",
Voyager.Kapitaen.Name, ", ", Voyager.Kapitaen.Geschlecht);
    document.write("<br>Lebensform: ", Voyager.Kapitaen.Lebensform);

  </script>
</body>
</html>
```

Das Objekt Raumschiff wird um die Eigenschaft Kapitaen erweitert. Dazu wird ein Person-Objekt definiert, das die Eigenschaften einer Person vereint. Die Eigenschaft Kapitaen ist vom Typ Person. Um auf eine Eigenschaft des Kapitäns eines Raumschiffs außerhalb eines Objekts zuzugreifen, verwenden Sie den Zugriffspfad `Raumschiff.Kapitaen.Eigenschaft`.

Werte überschreiben

Die Eigenschaft Kapitaen des Objekts Enterprise verweist auf dasselbe Objekt wie die Variable kirk. Ändern Sie eine Eigenschaft des Objekts über die Variable kirk, ändert sich ebenfalls der Wert der Eigenschaft Kapitaen im Objekt Enterprise. Die beiden folgenden Anweisungen führen dieselbe Operation aus:

```
kirk.Lebensform = "Nicht-Vulkanier";
Enterprise.Kapitaen.Lebensform =   "Nicht-Vulkanier";
```

Dynamisches Hinzufügen und Löschen von Eigenschaften und Methoden

Einer Objektinstanz können dynamisch Eigenschaften hinzugefügt oder daraus entfernt werden. Um einer Objektinstanz eine Eigenschaft hinzuzufügen, muss die Eigenschaft durch einen Punkt getrennt hinter dem Namen der Instanz angegeben werden und ihr ein Wert zugewiesen werden. Über den Operator `delete` kann man eine Eigenschaft wieder entfernen. Wird nach dem Entfernen auf die Eigenschaft zu gegriffen, wird der Wert `undefined` geliefert. Beachte, dass nur Eigenschaften gelöscht können, die nicht über das Schlüsselwort `var` definiert wurden.

Beispiel

```
function Kunde(name){
 this.name =  name;
}
kd = new Kunde("Meier");
kd.vorname = "Frank";  // neue Eigenschaft hinzufügen
...
delete kd.vorname;     // die Eigenschaft wieder entfernen
```

Methoden

Funktionen, die einem Objekt zugeordnet sind, werden als Methoden bezeichnet. Sie werden einem Objekt wie eine Eigenschaft zugewiesen.

```
function Methodenname(Wert1, Wert2, ...){
  this.Eigenschaft1 = irgendeinWert;
}

function Konstruktorname(Wert1, Wert2, ...){
  this.Eigenschaft1 = Wert1;
  this.Eigenschaft2 = Methodenname;
}
```

Über das Schlüsselwort function wird eine Funktion in JavaScript definiert. Wird der Funktionsname mit dem Schlüsselwort `this` zugewiesen, wird die Funktion als Methode des Objekts interpretiert. Innerhalb der Funktion können Sie mit dem Schlüsselwort this auf die Eigenschaften des aktuellen Objekts zugreifen. this enthält immer eine Referenz auf das aktuelle Objekt.

Beispiel: raumschiff_bewegung.html

Nun können wir dem Objekt `Raumschiff` jetzt eine Methode hinzufügen, die eine Fortbewegung des Raumschiffs erlaubt. In dem Beispiel wurde das Objekt `Person` zugunsten der Übersicht weggelassen. Deshalb muss auch die Eigenschaft `Raumschiff.Kapitaen` entfernt werden.

raumschiff_bewegung.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekte, Eigenschaften und Methoden</title>
</head>
<body>
  <h3>Und es bewegt sich doch</h3>
  <script type="text/javascript">
```

```
    function Raumschiff(Schiffsname, Schiffsmodell, Kapitaen)
    {
      this.Name = Schiffsname;
      this.Modell = Schiffsmodell;
      this.Kapitaen = Kapitaen;
      this.Entfernung = 0;
      this.entfernen = function VonErdeEntfernen(Lichtjahre)
      {
        this.Entfernung += Lichtjahre;
        document.write(this.Name + " => Aktuelle Reise: " + Lichtjahre
+ " Lichtjahre -> Erdentfernung: " + this.Entfernung + "
Lichtjahre<br>");
      }
    }

    var Enterprise = new Raumschiff("U.S.S. Enterprise", "NCC-1701");
    var Voyager = new Raumschiff("U.S.S. Voyager", "NCC-74656");

    document.write("<p>Beide Raumschiffe sind von der Erde
gestartet:<br>");
    Enterprise.entfernen(50);
    Voyager.entfernen(180);
    Enterprise.entfernen(200);
    Voyager.entfernen(-50);
    Enterprise.entfernen(-100);
    Enterprise.entfernen(-150);
    Voyager.entfernen(40);
    Voyager.entfernen(-170);

  </script>
</body>
</html>
```

Anweisungen und Operatoren für Objekte

if-Anweisung

Über die `if`-Anweisung können wir prüfen, ob ein Objekt eine bestimmte Eigenschaft besitzt.

Das Objekt Kunde wird mit der Eigenschaft name definiert. In den beiden folgenden if-Anweisungen wird geprüft, ob ein Objekt Kunde die Eigenschaften name und vorname besitzt.

```
function  Kunde(name){
   this.Name = name;
}
```

```
var kd = new Kunde('Meier');
if(kd.Name)
   document.write("Die Eigenschaft 'Name' im Objekt 'Kunde' existiert.");
if (!kd.Vorname)
document.write("Die  Eigenschaft 'Vorname' im Objekt 'Kunde' existiert
nicht.");
```

Wir haben jedoch keine Möglichkeit zu prüfen, ob ein Objekt mit einem bestimmten Namen existiert, d.h., ob eine Konstruktor-Funktion mit diesem Namen vorhanden ist.

with-Anweisung

Müssen wir hintereinander auf mehrere Eigenschaften und Methoden eines Objekts zugreifen, können wir den Zugriff darauf über die with-Anweisung verkürzen. Dazu wird nach dem Schlüsselwort with der Name der Objektinstanz in Klammern angefügt. Dadurch muss im folgenden Anweisungsblock nur der Name der Eigenschaft ohne den Objektnamen angegeben werden. Dabei wird für jeden Bezeichner zuerst nach einer Eigenschaft im Objekt gesucht, dann nach einem lokalen und globalen Bezeichner.

Beispiel

Durch die Verwendung der with-Anweisung müssen Sie innerhalb der geschweiften Klammern in den Methoden `document.write()` nicht mehr den Objektnamen kd angeben.

```
function Kunde(name,vorname){
  this.Name  =   name;
  this.Vorname  =  vorname;
}
var kd = new Kunde("Meier", "Hans");
with(kd) {
  document.write("Der Vorname lautet: " + Vorname + "<br>");
  document.write("Der Nachname lautet: " + Name + "<br>");
)
```

for-in-Anweisung

Um über alle Eigenschaften eines Objekts zu iterieren, nutzen Sie diese Anweisung. Dazu wird nach der f or-Anweisung in Klammern eine Laufvariable definiert. Dieser folgen das Schlüsselwort in und ein Objektname oder der Name einer Objektinstanz.

Beispiel

Die Laufvariable enthält in jedem Durchlauf den Namen der nächsten Eigenschaft. Außerdem können Sie über den Objektnamen und die gleichzeitige Angabe des Eigenschaftsnamens in eckigen Klammern auf den Wert der Eigenschaft zugreifen. Die folgende Anweisung gibt die Namen der Eigenschaften des Objekts Kunden und dessen Vorgabewerte aus. Sie können diese Anweisung auch

für eine Instanz des Objekts Kunden, z.B. kd, ausführen.

```
function Kunde(name,vorname,alter,geschlecht)
{
 this.Name = name;
 this.Vorname = vorname;
 this.Alter = alter;
 this.Geschlecht = geschlecht;
}
var kd = new Kunde('Meier','Hans',65,'männlich');
for(laufvariable in kd)
 document.write("Name: " + laufvariable + ", Wert: " + kd[laufvariable]);
```

in-Operator

Dieser Operator steht seit JavaScript 1.4 zur Verfügung und kann aus diesem Grund im Internet Explorer zurzeit nicht genutzt werden. Es wird bei dessen Verwendung jedoch keine Fehlermeldung erzeugt. Der `in`-Operator überprüft, ob eine Eigenschaft in einem Objekt definiert ist. Der Name der Eigenschaft wird dazu in doppelte Anführungszeichen gesetzt. Ist die Eigenschaft vorhanden, wird `true` geliefert, sonst `false`.

Beispiel

```
function Kunde(name,vorname,alter,geschlecht)
{
 this.Name    = name;
 this.Vorname = vorname;
 this.Alter   = alter;
 this.Geschlecht = geschlecht;
}
var kd = new Kunde('Meier','Hans',65,'männlich');
if("Name" in kd)
 document.write("Das Objekt 'Kunde' besitzt eine Eigenschaft 'Name'");
```

instanceof-Operator

Über diesen Operator können Sie prüfen, ob eine Objektinstanz vom Typ eines bestimmten Objekts ist. In diesem Fall wird als Ergebniswert true geliefert.

Beispiel

```
function Kunde () {}
var kd = new Kunde();
if(kd instanceof Kunde)
```

```
document.write("kd ist eine Instanz vom Objekt 'Kunde'");
```

Beispiel: anweisung+operator.html

anweisung+operator.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Anweisungen und Operatoren für Objekte</title>
</head>
<body>
  <h3>Kontrolle, ob Eigenschaften existieren</h3>
  <script type="text/javascript">
   function Kunde(name)
   {
     this.Name = name;
   }
   var kd = new Kunde('Meier');
   if(kd.Name)
     document.write("Die Eigenschaft 'Name' im Objekt 'Kunde'
existiert.<br>");
   if(!kd.Vorname)
    document.write("Die Eigenschaft 'Vorname' im Objekt 'Kunde'
existiert nicht.<br>");
  </script>

  <h3>Ausgabe der Eigenschaften mit "with"</h3>

  <script type="text/javascript">
   function Kunde(name, vorname)
   {
     this.Name = name;
     this.Vorname = vorname;
   }
   var kd = new Kunde('Meier', 'Hans');
   with (kd) {
     document.write("Der Vorname lautet: " + Vorname + "<br>");
     document.write("Der Nachname lautet: " + Name + "<br>");
   }
  </script>


  <h3>Objekteigenschaften auslesen mit "for... in..."</h3>
  <script type="text/javascript">
   function Kunde(name, vorname, alter, geschlecht)
   {
```

```
      this.Name     = name;
      this.Vorname = vorname;
      this.Alter    = alter;
      this.Geschlecht = geschlecht;
    }
    var kd = new Kunde('Meier', 'Hans', 65, 'männlich');
    for(laufvariable in kd)
      document.write("Eigenschaftsname: " + laufvariable + " Wert: " +
kd [laufvariable] + "<br>");

  </script>

  <h3>Objekteigenschaften suchen mit "in"</h3>
  <script type="text/javascript">
   function Kunde(name, vorname, alter, geschlecht)
   {
     this.Name     = name;
     this.Vorname = vorname;
     this.Alter    = alter;
     this.Geschlecht = geschlecht;
   }
   var kd = new Kunde('Meier', 'Hans', 65, 'männlich');
   if("Alter" in kd)
     document.write("Das Objekt 'Kunde' besitzt eine Eigenschaft
'Alter'.<br>");
   if("Haarschnitt" in kd)
     document.write("Das Objekt 'Kunde' besitzt eine Eigenschaft
'Haarschnitt'.<br>")
   else
     document.write("Das Objekt 'Kunde' besitzt nicht die Eigenschaft
'Haarschnitt'.<br>")
  </script>


  <h3>Objektinstanz kontrollieren mit "instanceof"</h3>
  <script type="text/javascript">
   function Kunde()
   {
   }
   var kd = new Kunde();
   if(kd instanceof Kunde)
     document.write("kd ist eine Instanz vom Objekt 'Kunde'.<br>");
  </script>

  </body>
</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_07

Last update: **2019/05/16 14:50**

11.3.8) Vordefinierte Objekte

Grundlagen zu vordefinierten Objekten

In JavaScript gibt es bereits vordefinierte Objekte, die häufig benötigte Methoden beinhalten. Mathematische Methoden sind in diesen Objekten genauso vorhanden wie Methoden zur Berechnung und Verwendung von Uhrzeiten oder Kalendertagen. Es gibt ein vordefiniertes Objekt zum Abspeichern von Variablenlisten. Zeichenketten werden grundsätzlich als `String`-Objekte abgelegt, die dem Programmierer auch Methoden zur Zeichenkettenverarbeitung zur Verfügung stellen. Auch Zahlen und boolesche Werte werden tatsächlich in Objekten abgelegt. Seit der Version 1.2 gibt es in JavaScript außerdem noch die Objekte `Screen` und RegExp. Ersteres kapselt die Eigenschaften des Bildschirms des Benutzers, Letzteres implementiert das Konzept der regulären Ausdrücke für JavaScript.

Objekt String für Zeichenketten

Das `string`-Objekt wurde bereits in den vorangegangenen Beispielen verwendet, da alle Zeichenkettenvariablen string-Objekte sind. Dies bedeutet, dass nicht nur der Inhalt der Zeichenkette durch den Variablennamen erreichbar ist, sondern auch die Eigenschaft `length` des `string`-Objekts (Länge der Zeichenkette). Außerdem besitzt das Objekt Methoden, um z. B. einen Teil der Zeichenkette zurückzuliefern.

```
x  =   String.Eigenschaft;
y  =   String.Methode(Übergabewert);
```

Zur Darstellung der verschiedenen Verarbeitungsmöglichkeiten soll als Beispiel die Variable zitat mit der Zeichenkette „Sein oder nicht sein,“ verwendet werden (`zitat = „Sein oder nicht sein,“`). Der Zugriff auf das erste Zeichen erfolgt im Folgenden über den Index 0. Auf das zweite Zeichen greifen Sie mit dem Index 1 zu usw.

Methode	Erläuterung	Ergebnis
`zitat.charAt(6);`	Mit `charAt(x)` können Sie das Zeichen auslesen, das sich an der x+1-ten Stelle befindet.	`d`
`zitat.charCodeAt(6);`	`charCodeAt(x)` liefert den ASCII-Code des x+1-ten Zeichens	`100`

`zitat.concat(" das ist hier die Frage.");`	Verbindet die Zeichenkette mit der in Klammern angegebenen Zeichenkette	`Sein oder nicht sein, das ist hier die Frage.`
`zitat.indexOf('c');`	Sucht nach der angegebenen Zeichenkette und gibt zurück, an der wievielten Stelle sie im String gefunden wurde. Wird das Zeichen nicht gefunden, wird `-1` zurückgegeben.	`12`
`zitat.lastIndexOf('e');`	Die Suche nach dem String startet am Ende der Zeichenkette und verläuft rückwärts.	`17`
`zitat.slice(5, 9);`	Extrahiert einen Teil der Zeichenkette. Die Parameter sind die Start- und Endposition in der Zeichenkette. Beim Weglassen des zweiten Parameters wird der Rest der Zeichenkette ausgelesen.	`oder`
`zitat.split(' ');`	Diese Methode teilt eine Zeichenkette in mehrere Teile. Die Trennung erfolgt an den Positionen des angegebenen Trennzeichens.	`('Sein','oder', 'nicht','sein...')`
`zitat.substr(5, 4);`	Diese Methode funktioniert ähnlich der Methode `slice()`. Es werden jedoch der Beginn der Teilzeichenkette und deren Länge angegeben.	`oder`
`zitat.substring(5, 9);`	Die Funktionsweise entspricht der Funktion der `slice()`-Methode.	`oder`
`zitat.toLowerCase();`	Der Rückgabewert ist die Zeichenkette, deren Buchstaben in Kleinbuchstaben umgewandelt wurden.	`sein oder nicht sein,`
`zitat.toUpperCase();`	Der Rückgabewert ist die Zeichenkette, deren Buchstaben in Großbuchstaben umgewandelt wurden.	`SEIN ODER NICHT SEIN,`

Beispiel: objekt_string1.html

objekt_string1.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt String</title>
<link type="text/css" rel=stylesheet href="style.css">
</head>
<body>
  <h2>Zeichenketten-Verarbeitung</h2>
  <script type="text/javascript">

    var zitat1="Sein oder nicht sein,";
    var zitat2=" das ist hier die Frage.";

    document.write(  "zitat1 = " + zitat1 + "<br>"
                   + "zitat2 = " + zitat2 + "<br>"
                   + "zitat1.length          = " + zitat1.length + "
```

```
Zeichen<br>"
                    + "zitat1.charAt(6)        = " + zitat1.charAt(6) +
"<br>"
                    + "zitat1.charCodeAt(6)    = " +
zitat1.charCodeAt(6) + "<br>"
                    + "zitat1.concat(zitat2)   = " +
zitat1.concat(zitat2) + "<br>"
                    + "zitat1.indexOf('c')     = " + zitat1.indexOf("c")
+ ". Stelle<br>"
                    + "zitat1.lastIndexOf('e') = " +
zitat1.lastIndexOf("e") + ". Stelle<br>"
                    + "zitat1.slice(5, 9)      = " + zitat1.slice(5, 9)
+ "<br>"
                    + "zitat1.split(' ')       = " + zitat1.split(" ") +
"<br>"
                    + "zitat1.substr(5, 4)     = " + zitat1.substr(5, 4)
+ "<br>"
                    + "zitat1.substring(5, 9)  = " + zitat1.substring(5,
9) + "<br>"
                    + "zitat1.toLowerCase()    = " +
zitat1.toLowerCase() + "<br>"
                    + "zitat1.toUpperCase()    = " +
zitat1.toUpperCase() + "<br>");
  </script>
</body>
</html>
```

Beispiel: objekt_string2.html

objekt_string2.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <link type="text/css" rel=stylesheet href="style.css">
  <title>Objekt String</title>
  <script type="text/javascript">
    var zitat1='Sein oder nicht sein,';
  </script>
</head>
<body>
  <h2>HTML-Methoden von String</h2>
  <p>Vorgabe: zitat1='Sein oder nicht sein,';</p>

  <table summary="Übersicht der HTML-Methoden des Objekts string"
border="1" width="100%">
    <tr>
      <td width="25%"><b>Methode</b></td>
```

```
    <td width="41%"><b>Beispiel</b></td>
    <td width="34%"><b>Ausgabe</b></td>
  </tr>
  <tr>

    <td>anchor(Anker)</td>
    <td>zitat1.anchor(&amp;quot;#anker&amp;quot;);</td>
    <td>
      <script type="text/javascript">

        document.write(zitat1.anchor("#anker"));

      </script>
    </td>
  </tr>

  <tr>
    <td>big()</td>
    <td>zitat1.big()</td>
    <td>
      <script type="text/javascript">

        document.write(zitat1.big());

      </script>
    </td>
  </tr>

  <tr>
    <td>blink()</td>
    <td>zitat1.blink()</td>
    <td>
      <script type="text/javascript">

        document.write(zitat1.blink());

      </script>
    </td>
  </tr>

  <tr>
    <td>bold()</td>
    <td>zitat1.bold()</td>
    <td>
      <script type="text/javascript">

        document.write(zitat1.bold());

      </script>
    </td>
  </tr>
```

```
<tr>
  <td>fixed()</td>
  <td>zitat1.fixed()</td>
  <td>
    <script type="text/javascript">

      document.write(zitat1.fixed());

    </script>
  </td>
</tr>

<tr>
  <td>fontcolor()</td>
  <td>zitat1.fontcolor(&amp;quot;red&amp;quot;)</td>
  <td>
    <script type="text/javascript">

      document.write(zitat1.fontcolor("red"));

    </script>
  </td>

</tr>
<tr>
  <td>fontsize()</td>
  <td>zitat1.fontsize(&amp;quot;+2&amp;quot;)</td>
  <td>
    <script type="text/javascript">

      document.write(zitat1.fontsize("+2"));

    </script>

  </td>
</tr>
<tr>
  <td>italics()</td>
  <td>zitat1.italics()</td>
  <td>
    <script type="text/javascript">

      document.write(zitat1.italics());

    </script>

  </td>
</tr>
<tr>
  <td>link(Hyperlink)</td>
```

```
      <td>zitat1.link(&amp;quot;http://www.mein.web&amp;quot;)</td>
      <td>
        <script type="text/javascript">

          document.write(zitat1.link("http://www.mein.web"));

        </script>

      </td>
    </tr>
    <tr>
      <td>small()</td>
      <td>zitat1.small()</td>
      <td>
        <script type="text/javascript">

          document.write(zitat1.small());

        </script>

      </td>
    </tr>
    <tr>
      <td>strike()</td>
      <td>zitat1.strike()</td>
      <td>
        <script type="text/javascript">

          document.write(zitat1.strike());

        </script>

      </td>
    </tr>
    <tr>
      <td>sub()</td>
      <td>zitat1.sub()</td>
      <td>
        <script type="text/javascript">

          document.write(zitat1.sub());

        </script>

      </td>
    </tr>
    <tr>
      <td>sup()</td>
      <td>zitat1.sup()</td>
      <td>
        <script type="text/javascript">
```

```
          document.write(zitat1.sup());

        </script>

      </td>
    </tr>
  </table>
</body>
</html>
```

Objekt Math für mathematische Berechnungen

Einige Aufgabenstellungen benötigen mathematische Funktionen oder Konstanten zu ihrer Lösung. JavaScript stellt Funktionen und Konstanten über das Math-Objekt zur Verfügung.

Mathematische Konstanten

Diese Konstanten sind fest in JavaScript implementiert. Die Werte lassen sich über den Namen der Konstanten als Eigenschaft des Objekts Math abrufen.

Beispiel: objekt_math_const.html

Im Beispiel werden die mathematischen Konstanten aufgelistet, die wir über die Angabe der entsprechenden Eigenschaft abrufen können.

objekt_math_const.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <link type="text/css" rel=stylesheet href="style.css">
  <title>Objekt Math</title>
</head>
<body>
  <h2>Mathematische Konstanten</h2>
  <table summary="Übersicht der mathematischen Konstanten" border="1"
width="100%">
    <tr>

      <td width="15%"><b>Eigenschaft</b></td>
      <td width="40%"><b>Bedeutung</b></td>
```

```
      <td width="20%"><b>Beispiel</b></td>
      <td width="25%"><b>Ausgabe</b></td>
    </tr>
    <tr>
      <td width="15%">E</td>

      <td width="40%">Eulersche Zahl</td>
      <td width="20%">Math.E</td>
      <td width="25%">
        <script type="text/javascript">

          document.write(Math.E);

        </script>
      </td>
    </tr>
    <tr>

      <td width="15%">LN2</td>
      <td width="40%">Natürlicher Logarithmus von 2</td>
      <td width="20%">Math.LN2</td>
      <td width="25%">
        <script type="text/javascript">

          document.write(Math.LN2);

        </script>
      </td>
    </tr>

    <tr>
      <td width="15%">LN10</td>
      <td width="40%">Natürlicher Logarithmus von 10</td>
      <td width="20%">Math.LN10</td>
      <td width="25%">
        <script type="text/javascript">

          document.write(Math.LN10);

        </script>
      </td>

    </tr>
    <tr>
      <td width="15%">LOG2E</td>
      <td width="40%">Logarithmus von e zur Basis 2</td>
      <td width="20%">Math.LOG2E</td>
      <td width="25%">
        <script type="text/javascript">

          document.write(Math.LOG2E);
```

```
      </script>

    </td>
  </tr>
  <tr>
    <td width="15%">LOG10E</td>
    <td width="40%">Logarithmus von e zur Basis 10</td>
    <td width="20%">Math.LOG10E</td>
    <td width="25%">

      <script type="text/javascript">

        document.write(Math.LOG10E);

      </script>
    </td>
  </tr>
  <tr>
    <td width="15%">PI</td>
    <td width="40%">Zahl &pi;</td>
    <td width="20%">Math.PI</td>

    <td width="25%">
      <script type="text/javascript">

        document.write(Math.PI);

      </script>
    </td>
  </tr>
  <tr>
    <td width="15%">SQRT1_2</td>
    <td width="40%">1 dividiert durch die Wurzel von 2</td>

    <td width="20%">Math.SQRT1_2</td>
    <td width="25%">
      <script type="text/javascript">

        document.write(Math.SQRT1_2);

      </script>
    </td>
  </tr>
  <tr>
    <td width="15%">SQRT2</td>

    <td width="40%">Wurzel aus 2</td>
    <td width="20%">Math.SQRT2</td>
    <td width="25%">
      <script type="text/javascript">
```

```
          document.write(Math.SQRT2);

        </script>
      </td>
    </tr>
  </table>

</body>
</html>
```

Mathematische Funktionen

Beispiel: objekt_math_methoden.html

Die Winkel beim Aufruf der Funktionen Sinus, Cosinus und Tangens sind im Bogenmaß (Einheit Radiant, rad) anzugeben.

objekt_math_methoden.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <link type="text/css" rel=stylesheet href="style.css">
  <title>Math-Objekt</title>
</head>
<body>
  <h2>Mathematische Funktionen</h2>
  <table summary="Übersicht der mathematischen Funktionen" border="1"
width="100%">
    <tr>

      <td width="15%"><b>Methode</b></td>
      <td width="30%"><b>Bedeutung</b></td>
      <td width="15%"><b>Wertebereich</b></td>
      <td width="20%"><b>Beispiel</b></td>
      <td width="20%"><b>Ausgabe</b></td>
    </tr>

    <tr>
      <td width="15%">abs(Zahl)</td>
      <td width="30%">Absolutwert</td>
      <td width="15%">Reale Zahlen</td>
      <td width="20%">Math.abs(-9)</td>
```

```
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.abs(-9));

        </script>

      </td>
    </tr>
    <tr>
      <td width="15%">acos(Zahl)</td>
      <td width="30%">Arcuscosinus</td>
      <td width="15%">[-1;1]</td>
      <td width="20%">Math.acos(-1)</td>

      <td width="20%">
        <script type="text/javascript">

            document.write(Math.acos(-1));

        </script>
      </td>
    </tr>
    <tr>
      <td width="15%">asin(Zahl)</td>
      <td width="30%">Arcussinus</td>

      <td width="15%">[-1;1]</td>
      <td width="20%">Math.asin(0.5)</td>
      <td width="20%">
        <script type="text/javascript">

          document.write(Math.asin(0.5));

        </script>
      </td>
    </tr>
    <tr>

      <td width="15%">atan(Zahl)</td>
      <td width="30%">Arcustangens</td>
      <td width="15%">[-1;1]</td>
      <td width="20%">Math.atan(-0.5)</td>
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.atan(-0.5));

        </script>

      </td>
```

```
    </tr>
    <tr>
      <td width="15%">ceil(Zahl)</td>
      <td width="30%">Aufrunden zur nächsten Ganzzahl</td>
      <td width="15%">Reale Zahlen</td>
      <td width="20%">Math.ceil(9.3)</td>

      <td width="20%">
        <script type="text/javascript">

            document.write(Math.ceil(9.3));

        </script>
      </td>
    </tr>
    <tr>
      <td width="15%">cos(Zahl)</td>
      <td width="30%">Cosinus</td>

      <td width="15%">Reale Zahlen (Rad)</td>
      <td width="20%">Math.cos(Math.PI)</td>
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.cos(Math.PI));

        </script>
      </td>
    </tr>
    <tr>

      <td width="15%">exp(Zahl)</td>
      <td width="30%"> Exponentialwert der Zahl auf Basis der
Eulerschen Konstanten
        E</td>
      <td width="15%">Reale Zahlen</td>
      <td width="20%">Math.exp(9.3)</td>
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.exp(9.3));

        </script>

      </td>
    </tr>
    <tr>
      <td width="15%">floor(Zahl)</td>
      <td width="30%">Abrunden zur nächsten Ganzzahl</td>
      <td width="15%">Reale Zahlen</td>
      <td width="20%">Math.floor(3.9)</td>
```

```
    <td width="20%">
      <script type="text/javascript">

          document.write(Math.floor(3.9));

      </script>
    </td>
  </tr>
  <tr>
    <td width="15%">log(Zahl)</td>
    <td width="30%">Natürlicher Logarithmus</td>

    <td width="15%">Reale Zahlen &gt; 0</td>
    <td width="20%">Math.log(3.9)</td>
    <td width="20%">
      <script type="text/javascript">

          document.write(Math.log(3.9));

      </script>
    </td>
  </tr>

  <tr>
    <td width="15%">max(Zahl1, Zahl2)</td>
    <td width="30%">Rückgabe der größeren Zahl</td>
    <td width="15%">Reale Zahlen</td>
    <td width="20%">Math.max(9,3)</td>
    <td width="20%">
      <script type="text/javascript">

          document.write(Math.max(9,3));

      </script>

    </td>
  </tr>
  <tr>
    <td width="15%">min(Zahl1, Zahl2)</td>
    <td width="30%">Rückgabe der kleineren Zahl</td>
    <td width="15%">Reale Zahlen</td>
    <td width="20%">Math.min(9,3)</td>

    <td width="20%">
      <script type="text/javascript">

          document.write(Math.min(9,3));

      </script>
    </td>
```

```
    </tr>
    <tr>
      <td width="15%">pow(Zahl1, Zahl2)</td>
      <td width="30%">Exponentialfunktion &quot;Zahl1 hoch
Zahl2&quot;</td>

      <td width="15%">Reale Zahlen</td>
      <td width="20%">Math.pow(9,3)</td>
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.pow(9,3));

        </script>
      </td>
    </tr>
    <tr>

      <td width="15%">random()</td>
      <td width="30%">Zufallszahl zwischen 0 und 1</td>
      <td width="15%"> </td>
      <td width="20%">Math.random()</td>
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.random());

        </script>
      </td>

    </tr>
    <tr>
      <td width="15%">round(Zahl)</td>
      <td width="30%">kaufmännisches Runden</td>
      <td width="15%">Reale Zahlen</td>
      <td width="20%">Math.round(3.5)</td>
      <td width="20%">

        <script type="text/javascript">

            document.write(Math.round(3.5));

        </script>
      </td>
    </tr>
    <tr>
      <td width="15%">sin(Zahl)</td>
      <td width="30%">Sinus</td>
      <td width="15%">Reale Zahlen (Rad)</td>

      <td width="20%">Math.sin(Math.PI/2)</td>
```

```
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.sin(Math.PI/2));

        </script>
      </td>
    </tr>
    <tr>
      <td width="15%">sqrt(Zahl)</td>

      <td width="30%">Quadratwurzel</td>
      <td width="15%">Reale Zahlen &gt; 0</td>
      <td width="20%">Math.sqrt(9)</td>
      <td width="20%">
        <script type="text/javascript">

            document.write(Math.sqrt(9));

        </script>
      </td>

    </tr>
    <tr>
      <td width="15%">tan(Zahl)</td>
      <td width="30%">Tangens</td>
      <td width="15%">Reale Zahlen (Rad)</td>
      <td width="20%">Math.tan(Math.PI/4)</td>
      <td width="20%">

        <script type="text/javascript">

            document.write(Math.tan(Math.PI/4));

        </script>
      </td>
    </tr>
  </table>
</body>
</html>
```

Beispiel: objekt_math_random.html

Die Methode `random()` liefert Zufallszahlen. Das folgende Beispiel beinhaltet die Funktion `RolleWuerfel()`, die mit der `random()`-Methode zufällige Zahlen zwischen 1 und 6 erzeugt:

objekt_math_random.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>W&uuml;rfel</title>
</head>
<body>
  <font size="+2">W&uuml;rfelergebnisse:<br>
  <script type="text/javascript">
    function RolleWuerfel()
    {
      var Zufallszahl = Math.random() * 6;
      return (Math.floor(Zufallszahl) + 1);
    }

    for(i = 0; i < 10; i++)
      document.write(RolleWuerfel() + " - ");

  </script>
  </font>

</body>
</html>
```

Die Werte ändern sich jedes Mal, sobald wir das HTML-Dokument über das entsprechende Menü im Browser aktualisieren. Es wird dadurch immer eine zufällige Zahlenfolge ausgegeben.

Objekt Number für Zahlen

So, wie Sie das `String`-Objekt für Zeichenketten einsetzen, wird das `Number`-Objekt für Zahlen verwendet. Alle Zahlen, unabhängig davon, ob es sich um Ganz- oder Gleitkommazahlen handelt, werden in `Number`-Objekten abgelegt. Das Objekt `Number` besitzt Eigenschaften für numerische Konstanten. Die Werte dieser Eigenschaften können nicht geändert werden.

Eigenschaft	Erläuterung	Wert
MAX_VALUE	MAX_VALUE beinhaltet die größte Zahl, die verarbeitet werden kann.	1.7976931348623157e+308
MIN_VALUE	MIN_VALUE beinhaltet die kleinste Zahl, die verarbeitet werden kann.	5e-324
NaN	Der Wert NaN (not a number) kann mit Ergebnissen von Operationen verglichen werden.	NaN
NEGATIVE_INFINITY	Der Wert einer Variablen ist -Infinity, wenn die Zahl kleiner als MIN_VALUE ist.	-Infinity
POSITIVE_INFINITY	Der Wert einer Variablen ist Infinity, wenn die Zahl größer als MAX_VALUE ist.	Infinity

Beispiel

In den folgenden Anweisungen wird geprüft, ob die Multiplikation zweier Werte den maximalen Wertebereich von JavaScript überschreitet. Solange der Wert der Multiplikation kleiner als der Maximalwert ist, könnte in der ersten Funktion eine weitere Berechnung durchgeführt werden. Die zweite Funktion gibt stattdessen eine Meldung aus, in der auf den Überlauf des Wertebereichs hingewiesen wird.

```
if (Zahl1 * Zahl2 <= Number.MAX VALUE)
  funktion1();
else
  funktion2();
```

Des Weiteren können wir überprüfen, ob eine Variable in einem Wertebereich liegt. In diesem Beispiel liegt der Wert für die Variable monat nicht im Wertebereich zwischen 1 und 12. Die Variable monat wird auf einen ungültigen Wert gesetzt, indem ihr über die Eigenschaft Number. NaN der Wert NaN zugewiesen wird.

```
var monat = 13
if (monat<1 || monat>12) {
  monat = Number.NaN
  alert("Der Monatswert muss zwischen 1 und 12 liegen.");
}
```

Mit den folgenden Methoden können wir die Ausgabe von Zahlen formatieren. In Klammern wird eine Zahl angegeben, die die Genauigkeit der Ausgabe festlegt. Ohne eine Parameterangabe wird die Standardeinstellung verwendet.

Methode	Erläuterung
`toExponential()`	Ab JavaScript 1.5; gibt eine Zahl in Exponentialschreibweise zurück: `77.1234 .toExponential() = 7.71234e+1`
`toFixed()`	Ab JavaScript 1.5; gibt eine Zahl als Dezimalzahl zurück: `10.1234 .toFixed(2) = 10.12`
`toPrecision()`	Ab JavaScript 1.5; gibt eine Zahl in Punkt-Notation zurück: `1250 .toPrecision(5) = 1250.0`
`toSource()`	Ab JavaScript 1.3; erzeugt ein neues Objekt auf Grundlage des angegebenen Objekts: `10.1234 .toSource() = (new Number(10.1234))`
`toString`	Liefert eine String-Repräsentation des gespeicherten Wertes: `document.write(10.1234 .toString());`
`valueOf`	Gibt den gespeicherten Wert zurück: `document.write(10.1234 .valueOf());`

Gibt man als Objekt den Wert als Literal an, so ist die Methode mit einem Leerzeichen Abstand anzugeben, z.B. `77 .toExponential()` oder `(77) . toExponential()` statt `77.toExponential()`.

Beispiel: objekt_number.html

Ein Beispiel mit einigen Methoden des Objekts `Number`.

objekt_number.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt Number</title>
</head>
<body>
  <h3>Ausgabe der Eigenschaften und Methoden</h3>
  <script type="text/javascript">

    document.write("<p>Number.MAX_VALUE: " , Number.MAX_VALUE);
    document.write("<br>Number.MIN_VALUE: " , Number.MIN_VALUE);
    document.write("<br>Number.NaN: " , Number.NaN);
    document.write("<br>Number.NEGATIVE_INFINITY: " ,
Number.NEGATIVE_INFINITY);
    document.write("<br>Number.POSITIVE_INFINITY: " ,
Number.POSITIVE_INFINITY);

    var num=77.1234;
    document.write("<hr>Wert von Variable num: ", num);
    document.write("<br>num.toExponential() = " + num.toExponential());
```

```
    document.write("<br>num.toExponential(4) = " +
num.toExponential(4));
    document.write("<br>num.toExponential(2) = " +
num.toExponential(2));
    document.write("<br>77.1234 .toExponential() = " + 77.1234
.toExponential())
    document.write("<br>77 .toExponential() = " + 77 .toExponential());

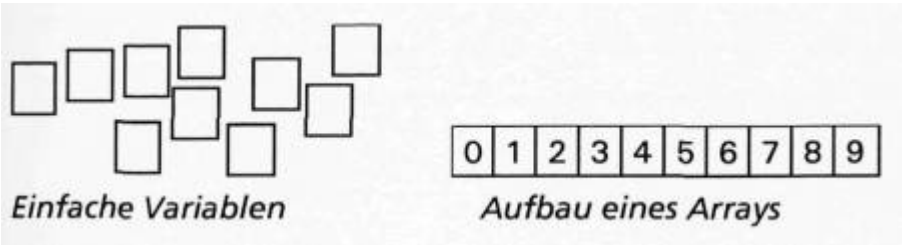
    var num=10.1234;
    document.write("<hr>Wert von Variable num: ", num);
    document.write("<br>num.toFixed() = " + num.toFixed());
    document.write("<br>num.toFixed(4) = " + num.toFixed(4));
    document.write("<br>10.1234 .toFixed(2) = " + 10.1234 .toFixed(2));

    var num=1250;
    document.write("<hr>Wert von Variable num: ", num);
    document.write("<br>num.toPrecision() = " + num.toPrecision());
    document.write("<br>num.toPrecision(4) = " + num.toPrecision(4));
    document.write("<br>num.toPrecision(2) = " + num.toPrecision(2));
    document.write("<br>num.toPrecision(1) = " + num.toPrecision(1));
    document.write("<br>1250 .toPrecision(2) = " + 1250
.toPrecision(2));
    document.write("<br>1250 .toPrecision(5) = " + 1250
.toPrecision(5));

  </script>
</body>
</html>
```

Objekt Array für Variablenlisten

Ein Array (Feld) ist ein Speicherbereich, in dem wir mehrere Variablen speichern können. Der Zugriff auf die Variablen erfolgt über einen Bezeichner und einen Index. Da JavaScript nicht typisiert ist, können in einem Array auch Werte verschiedenen Typs gespeichert werden. Um beispielsweise die Daten von 20 Personen zu speichern, benötigen wir normalerweise 20 Variablen. Mit dem `Array`-Objekt können wir diese in einem einzigen Array speichern.



Indizierte Felder

Im Gegensatz zu den anderen vordefinierten Objekten muss ein Array explizit erzeugt werden. Es gibt drei verschiedene Möglichkeiten, ein `Array`-Objekt zu erzeugen.

* Wir erzeugen ein leeres Feld mit einer unbestimmten Länge. Die Feldelemente werden dann bei der ersten Zuweisung eines Wertes erzeugt. Die Länge des Feldes ergibt sich aus der Größe des letzten Index. Auf diese Weise lässt sich ein Feld dynamisch anlegen und vergrößern.

```
var Feldname = new Array(); // leeres Feld
Feldname[1] = 100;
Feldname[4] = "Test";       // die Feldlänge ist 5, da der Index bei 0
beginnt
```

Sie teilen bereits beim Erzeugen des Objekts mit, wie viele Elemente es enthalten wird.

```
var name = new Array(4);    // das Feld wird 4 Elemente aufnehmen.
```

Wir können dem `Array`-Objekt bereits die gewünschten Werte zuweisen. Die einzelnen Werte werden mit Komma voneinander getrennt.

```
//  dem Feld werden vier Namen  zugewiesen
var name = ["Müller","Schmidt","Schulze","Lehmann"];

//  oder
var  name = new Array(4);
name[0] = "Müller";
name[1] = "Schmidt";
name[2] = "Schulze";
name[3] = "Lehmann";
```

Der Zugriff auf ein einzelnes Element des Feldes erfolgt über die Angabe eines Index, den Sie in eckigen Klammern angeben müssen. So erhalten Sie mit `name[0]` das erste Element des Feldes `name`, mit `name [1]` das zweite Element usw.

Assoziative Felder

Eine weitere Möglichkeit, Elemente von Feldern anzulegen und anzusprechen, sind assoziative Felder. Bei ihnen werden keine fortlaufenden Indizes zur Adressierung eines Eintrags im Array verwendet, sondern Zeichenketten.

```
var Stadt = new Array();
Stadt["Wohnort"] = "Frankfurt";
Stadt["Hauptstadt"] = "Berlin";
Stadt["Messestadt"] = "Leipzig";
```

Den Elementen können Sie dadurch eine beliebige Bezeichnung zuordnen, wobei der Zugriff auf die Feldelemente folgendermaßen erfolgt:

```
element = "Hauptstadt"; Stadtauswahl = Stadt[element];
```

oder über die Angabe der Zeichenkette:

```
Stadtauswahl = Stadt["Hauptstadt"];
```

Das Auslesen eines Elements wird direkt über dessen Bezeichnung realisiert. Dazu wird die Elementbezeichnung über eine Variable oder direkt in den eckigen Klammern des Feldes angegeben. In beiden Fällen der gezeigten Beispiele ist der Rückgabewert Berlin.

Methoden und Eigenschaften

Ein `Array`-Objekt besitzt nur die Eigenschaft `length`, womit die Anzahl der gespeicherten oder bei der In-stanziierung definierten Werte ausgelesen werden kann. Nach der Instanziierung `var Feldname = new Array(4);` liefert `Feldname.length` den Wert 4. Das letzte Element erhalten Sie direkt über die Angabe der Feldlänge, z.B. `name[name.length-1]`.

Ein Objekt vom Typ `Array()` besitzt neben der Eigenschaft length auch mehrere Methoden, um auf die Feldelemente zuzugreifen.

Methode	Erläuterung
`concat (Array1,Array2,..)`	Mit `concat()` können Sie dem ersten Feld die Inhalte weiterer Felder hinzufügen. Die neuen Inhalte werden hinten angehängt.

`join(Zeichen)`	Die einzelnen Elemente des Feldes werden durch das angegebene Zeichen miteinander verbunden und in einer Variablen abgelegt. Wird kein Verknüpfungszeichen angegeben, wird ein Komma verwendet.
	`a = new Array("Wind", "Regen", "Feuer")`
	`Var1 = a.join()  // Var1 = "Wind,Regen,Feuer"`
	`Var2 = a.join(" + ")  // Var2 = "Wind + Regen + Feuer"`

`pop()`	Liest das letzte Element im Feld aus und löscht es danach.
`push(Wert1, Wert2, ...)`	An das Ende des Feldes werden weitere Elemente angefügt. Zurückgegeben wird die neue Länge des Feldes.
`reverse()`	Kehrt die Reihenfolge der Elemente innerhalb des Feldes um. Das letzte Element wird zum ersten Element usw.
`shift()`	Das erste Element wird ausgelesen und gelöscht.
`slice(start, ende)`	Diese Methode gibt einen Teil des Feldes zurück, der mit dem Index `start` beginnt und mit dem Index `ende` endet. Das Ergebnis wird in einem neuen `Array`-Objekt zurückgegeben.
`splice(start, ende, Wert1, Wert2, ...)`	Die Elemente vom Index `start` bis `ende` werden durch die neuen Werte `Wert1, Wert2, ...` ersetzt. Als Rückgabewert erhalten Sie die Elemente von `start` bis `ende`.
`sort()`	Der Inhalt des Feldes wird alphabetisch sortiert. Enthalten die Elemente Zahlen, werden diese in Zeichenketten umgewandelt, sodass beispielsweise die Zahl 19 vor der Zahl 9 einsortiert wird. `name(1, 9, 19, 'Maria', 'Eva')` wird durch `name.sort();` sortiert zu `name(1, 19, 9, 'Eva', 'Maria')`.

Zusätzlich können wir in der Methode `sort()` eine Funktion hinterlegen, mit der Sie die

Sortierungslogik (d. h. den Vergleich zweier Elemente) selbst durchführen können.

Beispiel

```
function sortiere(a, b) {
  return a - b;
}
ArrayName.sort(sortiere);
```

Die Reihenfolge der Elemente wird gemäß der Sortierfunktion verändert. Die Sortierfunktion muss zwei Parameter entgegennehmen und eine Ganzzahl zurückliefern. Die Elemente des Arrays werden über diese Funktion verglichen und je nach Vorzeichen der Ganzzahl sortiert.

- `<0:` Der zweite Wert ist kleiner als der erste.
- `=0:` Beide Werte sind gleich.
- `>0:` Der erste Wert ist kleiner als der zweite.

Wird keine Funktion angegeben, werden die Elemente nach ihren ASCII-Codes sortiert (auch Zahlen). Die in der nachfolgenden Tabelle angegebenen Array-Methoden wurden ab JavaScript 1.6 implementiert.

Methode	Erläuterung
`indexOf (Elementname, [Index])`	Ab JavaScript 1.6 Mit `indexOf()` können Sie nach einem bestimmten Element innerhalb eines Feldes suchen. Neben dem zu suchenden Element können Sie optional den Startindex angeben. Wird das Element gefunden, wird der Indexwert zurückgeliefert, ansonsten `-1`.
`lastIndexOf (Elementname, [Index])`	Ab JavaScript 1.6 Wie `indexOf()`, nur wird hier mit der Suche von hinten begonnen.
`filter(Filterfunktion)`	Ab JavaScript 1.6 Mithilfe einer Filterfunktion können Sie aus einem Array ein neues Array bilden. Elemente, die dem Filter nicht genügen, werden nicht übernommen.
`forEach(Filterfunktion)`	Ab JavaScript 1.6 Für jedes Element wird eine Funktion aufgerufen. Das Array wird aber nicht verändert.
`every(Testfunktion)`	Ab JavaScript 1.6 Aufruf einer möglichen Funktion, um alle Elemente zu testen. Es wird `true` zurückgeliefert, wenn alle Elemente der Funktion genügen, ansonsten `false`.
`map(Funktion)`	Ab JavaScript 1.6 Generiert ein neues Array, das aus dem Resultat der aufgerufenen Funktion gebildet wird.
`some(Testfunktion)`	Ab JavaScript 1.6 Diese Methode liefert den Wert `true` zurück, wenn mindestens ein Feldelement der Testfunktion genügt.
`reduce(Funktion)`	Ab JavaScript 1.8 Jeweils zwei benachbarte Feldelemente, von links beginnend, werden einer Funktion übergeben, und ein entsprechender Zwischenwert wird gebildet. Als Endergebnis wird ein Wert zurückgeliefert (Reduzierung des Arrays auf einen Wert).
`reduceRight(Funktion)`	Ab JavaScript 1.8 Wie `reduce()`, nur werden hier die Elemente von rechts betrachtet.

Objekt Date für Zeitangaben

Auf die Systemzeit wird in JavaScript über das `Date`-Objekt zugegriffen. Instanzen des Objekts geben nicht nur das aktuelle Datum wieder, sondern sie lassen sich auch für beliebige Kalendertage und Uhrzeiten erzeugen und können so die verschiedensten Aufgaben erfüllen.

In JavaScript wird die Zeit, die seit dem 1. Januar 1970 um 0:00 Uhr vergangen ist, in Millisekunden gespeichert. Dies bedeutet, dass Sie kein Datum verarbeiten können, das vor diesem Zeitpunkt liegt.

Es gibt drei verschiedene Möglichkeiten, um ein Datumsobjekt zu initialisieren:

- Werden bei der Initialisierung keine Parameter angegeben, so wird das Objekt mit dem aktuellen Datum und der aktuellen Uhrzeit initialisiert.

```
var Variable = new Date();
```

- Wir können aber auch der Variablen ein bestimmtes Datum und eine Uhrzeit zuordnen. Dazu gibt man die entsprechenden Werte in Form einer Zeichenkette an.\\

```
var  Variable   =  new  Date("Monatsname_englisch Tag, Jahr
Stunde:Minute:Sekunde")
```

- Das Datum und die Uhrzeit lassen sich auch in numerischer Form angeben.

```
var  Variable = new Date(Jahr, Monat, Tag, Stunde, Minute, Sekunde);
```

Beispiel

```
var zeitl = new  Date();  // aktuelles Datum und Uhrzeit    wird übergeben
var zeit2 = new  Date("September 22, 2008 10:24:00');   // 22.09.2008 um
10:24 Uhr
var zeit3 = new Date(2008,8,22,10,24,00);              // 22.09.2008 um
10:24 Uhr
```

Bei der Initialisierung in numerischer Form ist zu beachten, dass die Monate im Wertebereich von 0 bis 11 liegen (0 = Januar, 1 = Februar ...11= Dezember). Außerdem können Sie die Parameter für die Angabe der Uhrzeit weglassen. Diese wird automatisch auf 0:00 Uhr gesetzt.

```
var zeit3 = new Date (2008,22);   // 22.09.2008 um 0:00 Uhr
```

Methoden des Date-Objekts

Nach der Instanziierung eines `Date`-Objekts ist das in ihm gespeicherte Datum über seine Methoden zugänglich. Folgende Methoden stehen dazu zur Verfügung:

Methode	Erläuterung	Mögliche Rückgabewerte
`getDate()`	Tag im Monat	1 bis 31
`getDay()`	Nummer des Wochentages	0 bis 6
`getHours()`	Stunde	0 bis 23
`getMinutes()`	Minuten	0 bis 59
`getMonth()`	Nummer des Monats	0 bis 11
`getSeconds()`	Sekunden	0 bis 59
`getTime()`	Millisekunden seit dem 1.1.1970, 0:00 Uhr	0 bis ...
`getTimezoneOffset()`	Abstand zwischen Lokalzeit und GMT in Minuten	-720 bis +720
`getFullYear()`	Vierstellige Jahreszahl	Jahresangabe

Über die folgenden Methoden können Sie das Datum in andere Formate umwandeln.

Methode	Erläuterung	Mögliche Rückgabewerte
toGMTString()	Aktuelle Greenwichzeit	Mon, 22 Sep 2008 10:24:53 GMT
toLocaleString()	Landestypische Zeitausgabe	Montag, 22. September 2008 10:24:53

Es existieren außerdem die beiden Methoden `parse()` und `utc()`. Mit den Methoden `parse(Datumszeichenkette)` und `utc(Jahr,Monat,Tag,Std.,Min.,Sek.)` können wir Datumsangaben in Millisekunden ausgeben (UTC=Universal Time Coordinate). Beide Methoden stehen bei selbst definierten Date-Objekten nicht zur Verfügung. Um diese Methoden zu verwenden, muss die Schreibweise `Date.parse(Parameter)` bzw. `Date.UTC(Parameter)` verwendet werden.

Verwenden Sie nicht mehr die bisherige Methode `getYear()`. Da diese Methode nicht Jahr-2000-kompatibel ist, liefert sie in den Mozilla-Browsern nur die Differenz zum Jahr 1900 zurück. Die neue Methode `getFullYear()` liefert hingegen in allen Browsern das korrekte Jahr zurück.

Datumsangaben verändern

Mit den folgenden Methoden ändern Sie die Datumsangaben. Diese Änderungen sind nur innerhalb des JavaScripts gültig, die Systemzeit Ihres Computers wird dadurch nicht verändert.

Methode	Erläuterung
setFullYear(Jahr)	Ändert das Jahr
setMonth(Monat)	Ändert den Monat
setDate(Tag)	Ändert den Tag des Monats
setHours(Stunde)	Ändert die Stunde
setMinutes(Minute)	Ändert die Minute
setSeconds(Sekunde)	Ändert die Sekunde
setTime(Millisekunden)	Umrechnung in eine Datumsangabe

Die Wertebereiche der Parameter entsprechen den Rückgabewerten der zugehörigen get-Methoden. Date-Objekte können zum Darstellen, Speichern und Berechnen von Zeitangaben verwendet werden.

Beispiel: objekt_date.html

Das folgende Beispiel enthält einige Methoden zum Auslesen und Manipulieren von Datum und Uhrzeit mit-hilfe von JavaScript. Weiterhin wird das Datum ermittelt und ausgegeben, das in 150 Tagen erreicht wird.

objekt_date.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt Date</title>
</head>
<body>
```

```
  <h3> Die aktuelle Uhrzeit</h3>
  <script type="text/javascript">
    var zeit = new Date(); // Variable mit aktuellem Datum und Uhrzeit
initialisieren
    document.write("Datum, Uhrzeit: " + zeit.toLocaleString() + "
Minuten<br>");
    document.write("Jahr: " + zeit.getFullYear() + "; ");
    document.write("Monat: " + zeit.getMonth() + "; "); /* Monate
0...11 */
    document.write("Tag: " + zeit.getDate() + "; ");
    document.write("Wochentag: " + zeit.getDay() + "; ");
    document.write("Stunden: " + zeit.getHours() + "; ");
    document.write("Minuten: " + zeit.getMinutes() + "; ");
    document.write("Sekunden: " + zeit.getSeconds() + "<br> ");
    document.write("Greenwichzeit: " + zeit.toGMTString() + "<br>");
    document.write("Lokale Ausgabe: " + zeit.toLocaleString() +
"<br>");
    document.write("Abweichung: " + zeit.getTimezoneOffset() + "
Minuten<br>");

    jahr  = zeit.getFullYear();
    monat = zeit.getMonth() + 1;
    tag   = zeit.getDate();

    document.write("Heute ist der " + tag + "." + monat + "." + jahr +
"<br>");

    dann = zeit.getTime() + (150*24*60*60*1000); // 150 Tage in
Millisekunden umrechnen

    /* Zeit auf neues Datum (+150 Tage) setzen */
    zeit.setTime(dann);
    document.write("In 150 Tagen ist " + zeit.toLocaleString() +
"<br>");

  </script>
</body>
</html>
```

Obwohl die Berechnung korrekt durchgeführt wurde, liegt die Uhrzeit in 150 Tagen eine Stunde hinter der aktuellen Uhrzeit. Dies liegt an der zwischendurch auftretenden Umstellung von der Sommer- auf die Winter-zeit.

Beispiel: objekt_date2.html

Im nächsten Beispiel soll das aktuelle Datum formatiert und ausgegeben werden. Im Zusammenhang mit dem `Date`-Objekt können Sie statt der bisherigen numerischen Tages- und Monatsausgabe in Verbindung mit einem `Array`-Objekt die entsprechenden Wochentags- und Monatsnamen zuweisen und ausgeben lassen.

objekt_date2.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt Date</title>
</head>
<body>
  <h3>Name des aktuellen Wochentags und Monats</h3>
  <script type="text/javascript">
    var zeit  = new Date();
    var Tag   = zeit.getDate();
    var Wtag  = zeit.getDay();
    var Monat = zeit.getMonth();
    var Jahr  = zeit.getFullYear();

    WochenTagName = new Array("Sonntag", "Montag", "Dienstag",
"Mittwoch", "Donnerstag", "Freitag", "Samstag");
    MonatsName = new Array("Januar", "Februar", "März", "April", "Mai",
"Juni", "Juli", "August", "September", "Oktober", "November",
"Dezember");

    document.write("Standardausgabe des Datums:<BR>Heute ist der " +
Wtag + ".Wochentag, " + Tag + "." + (Monat+1) + "." + Jahr +
".<BR><BR>");
    document.write("Ausgabe des Datums mit Hilfe zweier
Arrays:<BR>Heute ist " + WochenTagName[Wtag] + ", der " + Tag + ". " +
MonatsName[Monat] + " " + Jahr + ".");

  </script>
</body>
</html>
```

Objekt RegExp für reguläre Ausdrücke

Die regulären Ausdrücke, auch RegExpressions genannt, erlauben es, Zeichenketten zu vergleichen. Sie können verwendet werden, um zu überprüfen, ob z. B. die Eingabe einer E-Mail-Adresse in einem Formularfeld eine gültige Adresse ist oder ob eine Telefonnummer die korrekte Formatierung hat.

Im Internet finden Sie zahlreiche Informationen zu regulären Ausdrücken. Geben Sie beispielsweise in der Suchmaschine http://www.google.de die Anfrage „reguläre +Ausdrücke +Tutorial“ ein.

JavaScript besitzt seit der Version 1.2 das Objekt RegExp. Das Objekt dient dazu, Instanzen zu erzeugen, mit denen reguläre Ausdrücke verarbeitet werden können, und es dient der Überwachung der Ergebnisse solcher Operationen. Das statische Objekt RegExp speichert in seinen Eigenschaften

Informationen über die zuletzt durchgeführte Operation mit einem regulären Ausdruck. Auch Methoden des `string`-Objekts wie `match()`, `replace()` und `search()` verwenden implizit die regulären Ausdrücke.

Aufbau eines regulären Ausdrucks

Reguläre Ausdrücke sind Muster (Patterns), die auf eine Zeichenkette angewendet werden. Ein gewöhnliches Muster besteht aus drei Teilen, dem Pattern, den Flags und den Begrenzern(/).

```
/Pattern/Flag
```

Die Schrägstriche sind sogenannte Delimiter (Begrenzer), die den Anfang und das Ende des Musters kennzeichnen. Ein Delimiter darf jedes beliebige nicht alphanumerische Zeichen (außer dem Schrägstrich) sein. Das Pattern ist der Hauptteil, welches das Suchmuster beinhaltet. Der letzte Teil besteht aus den Flags, die das Suchmuster beeinflussen. Instanzen aus regulären Ausdrücken erzeugen

```
var Suchel =  /sel/
var Suche2 =  new RegExp("sel")
```

Der einfachste Weg ist es, einer Variablen einen regulären Ausdruck zuzuweisen. Die erste Instanziierung verwenden Sie, wenn das Suchmuster nicht verändert werden muss. Die zweite Variante sollten Sie anwenden, wenn das Suchmuster innerhalb des Skripts geändert wird. In beiden Fällen wird die erste Fundstelle gesucht, die die Zeichenkette sei enthält.

Flags

Einem Ausdruck können Sie die beiden Flags `g` und `i` anfügen.

- `g` bedeutet, dass die Suche nach dem Ausdruck global erfolgen soll. Nach dem ersten Treffer wird die Suche fortgesetzt, sodass mehrere Treffer gefunden werden können.
- `i` lässt bei der Suche die Groß- und Kleinschreibung unberücksichtigt.

```
var Suche = /sel/gi
```

Metazeichen

Für die Definition des Musters können Sie Metazeichen verwenden, die bestimmte Sonderfälle markieren. Folgende Metazeichen werden von JavaScript interpretiert:

Metazeichen	Findet ...	Beispiel
\b	eine Wortgrenze	/\bthe/ findet "thematisch", "theatralisch". /the\b/ findet "Agathe", "Mathe". /\bthe\b/ findet nur "the".
\B	eine Nicht-Wortgrenze, also nicht am Beginn oder Ende eines Wortes	/\Ber\B/i findet "Wert", aber nicht "Erde" oder "Heer".
\d	eine Ziffer von 0 bis 9	/\d\d/ findet eine zweistellige Ziffer wie "42".
\D	eine Nicht-Ziffer	/\D\D/ findet alles außer Zahlen.
\s	ein Leerzeichen, einen Tabulator, einen Zeilenumbruch	/E-Mail\sAdresse/ findet "E-Mail Adresse".
\S	ein Nicht-Leerzeichen	/E-Mail\SAdresse/ findet "E-Mail-Adresse".
\w	einen Buchstaben, eine Ziffer oder einen Unterstrich	/\w1\w/ findet "A1A", "_12", "Z1_".
\W	keinen Buchstaben, keine Ziffer und keinen Unterstrich	/\W1\W/ findet " 1%", aber nicht "11%".
.	jedes Zeichen außer einem Zeilenumbruch	/../ findet zwei zusammenhängende Zeichen, wie z. B. "Z3".
^	den Beginn einer Zeichenkette	/^Frau/ findet "Frau Klößer...", aber nicht "Ich bin Frau Klößer."
$	das Ende einer Zeichenkette	/sie\.$/ findet "Alle schätzten sie."
[…]	irgendein Zeichen, das in der Klammer aufgelistet ist	/W[eoa]rt/ findet "Wert", "Wort", "Wart", aber nicht "Wirt".
[^…]	keines der in Klammern angegebenen Zeichen	/W[^au]rt/ findet "Wirt", "Wert", "Wort", aber nicht "Wart" und "Wurt".
(...)	eine Zeichenkette und legt sie in der Variablen $1 ab	/(\d{1,},\d{2})€./ findet "Es kostet 49,90€." und legt "49,90€" in $1 ab.

Mit den nachfolgenden Metazeichen geben Sie die Häufigkeit des Suchbegriffs an.

Metazeichen	Findet ...	Beispiel
+	ein- oder mehrmals	/\d+/ findet "3", "5678".
?	kein- oder einmal	/\d?/ findet "", "3" oder "9", aber nicht "11".
*	kein-, ein- oder mehrmals	/\d*/ findet "" und alle zusammenhängenden Zahlen.
{n}	genau n-mal	/\d{3}/ findet dreistellige Zahlen, wie "345".
{n,}	n- oder mehrmals	/\d{2,}/ findet zwei- und mehrstellige Zahlen.
{n,m}	mindestens n-, maximal m-mal	/\d{3,5}/ findet die drei-, vier- und fünfstelligen Zahlen.

Wollen Sie nach den Zeichen . [ ] * ? { } ( ) ^ oder $ suchen, müssen diese mit einem Backslash \ eingeleitet werden, da sie ansonsten als Metazeichen interpretiert werden. Das gilt auch für den Backslash selbst. Nach ihm wird mit einem doppelten Backslash gesucht \\, z.B:

```
var Suche = /\D:\\Daten\\Grafiken\\Apfel\.GIF/i;
```

Dieses Suchmuster passt beispielweise auf die Zeichenkette `D:\DATEN\Grafiken\aPfel.gif`, da die Groß-und Kleinschreibung nicht beachtet wird.

Methoden

Damit Sie eine Suche durchführen können, verwenden Sie die Methoden des regulären Ausdrucks.

Methode	Erklärung
`exec(Zeichenkette)`	Über diese Methode führen Sie eine Suche nach dem angegebenen Suchmuster in der Zeichenkette durch. Zurückgeliefert wird ein Array, das die gefundenen Stellen beschreibt. Wird nichts gefunden, liefert `exec` den Wert `null` zurück. Die Eigenschaft `index` enthält die Position des Buchstabens, an dem das gefundene Muster beginnt. Die Eigenschaft `input` enthält die durchsuchte Zeichenkette. Der erste Feldinhalt enthält den Teilstring, der das Muster erfüllt hat. Die folgenden Felder enthalten die weiteren Suchergebnisse.
`test(Zeichenkette)`	Es wird ein boolescher Wert `true` oder `false` zurückgeliefert, je nachdem, ob die Suche mit dem vorgegebenen Suchmuster erfolgreich war oder nicht.

Beispiel: objekt_regexp.html

Es wird eine vorgegebene Zeichenkette nach dem Auftreten bestimmter Suchmuster verarbeitet.

objekt_regexp.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt RegExp</title>
</head>
<body>
  <h3>Methoden von RegExp()</h3>
  <script type="text/javascript">

    var Text = "Frau Klößer hat Bücher im Wert von 49,90€ bestellt und
muss diese innerhalb der nächsten 14 Tage bezahlen.";

    document.write("<b>Vorgabewert:<\/b><br>" + Text);

    Suche = /\d\d/;
    document.write("<hr>Suche nach der ersten zweistelligen Zahl<br>");
    document.write(Suche.test(Text) + "<br>");
    document.write(Suche.exec(Text));

    Suche = /\d{1,}\sTage/;
    document.write("<hr>Suche nach dem Wort \"Tage\" mit vorangesetzter
Zahl<br>");
```

```
    document.write(Suche.test(Text) + "<br>");
    document.write(Suche.exec(Text));

    Suche = /[MWG]ut/;
    document.write("<hr>Suche nach \"Mut\", \"Wut\" oder \"Gut\"<br>");
    document.write(Suche.test(Text) + "<br>");
    document.write(Suche.exec(Text));

    Suche = /(\d{1,},\d{2})€/;
    document.write("<hr>Suche nach einer Preisangabe mit zwei
Kommastellen<br>");
    document.write(Suche.test(Text) + "<br>");
    document.write(Suche.exec(Text));

  </script>
</body>
</html>
```

Die Instanz des regulären Ausdrucks besitzt weitere vier Eigenschaften, die bei der Verwendung der Flags gesetzt werden:

- `global` enthält `true` oder `false`, je nachdem, ob das Flag `g` gesetzt ist oder nicht.
- `ignoreCase` enthält `true` oder `false`, je nachdem, ob das Flag `i` gesetzt ist oder nicht.
- `lastindex` enthält den Index des ersten Buchstabens, bei dem der nächste Suchdurchlauf des Musters beginnt. In einer neuen Suche ist der Wert 0. Ändern Sie diesen Wert, um die Suche an einer anderen Position zu beginnen.
- `source` enthält das Suchmuster als String ohne die beiden Schrägstriche.

Eigenschaften von RegExp

Jedes Mal, wenn wir einen regulären Ausdruck anwenden, werden die Ergebnisse der Operation in den Eigenschaften des RegExp-Objekts abgelegt. Die Benennung der Eigenschaften erfolgt über die $-Variablen oder über den entsprechenden ausgeschriebenen Namen.

Eigenschaft	Erläuterung
`$_` oder `input`	Enthält den durchsuchten Original-String
`$&` oder `lastMatch`	Enthält den zuletzt gefundenen String
`$+` oder `lastParent`	Enthält die zuletzt gefundene Teilsuche
`$`` oder `leftContext`	Enthält den Teilstring links von der gefundenen Stelle
`$´` oder `rightContext`	Enthält den Teilstring rechts von der gefundenen Stelle
`$*` oder `multiline`	Enthält `true`, wenn die Suche über mehrere Zeilen stattgefunden hat
`$1, ... , $9`	Enthält die Ergebnisse der Teilsuchen 1 bis 9

Reguläre Ausdrücke im string-Objekt

Das `string`-Objekt besitzt Methoden zum Durchsuchen und Verändern von Zeichenketten. Die Suchmuster lassen sich über die regulären Ausdrücke festlegen.

Bei der Angabe des regulären Ausdrucks müssen Sie beachten, dass dieser im Gegensatz zum RegExp-Objekt als Parameter in Klammern angegeben wird.

Beispiel: objekt_regexp_multi.html

Es wird eine vorgegebene Zeichenkette nach dem Auftreten bestimmter Suchmuster verarbeitet.

objekt_regexp_multi.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt RegExp</title>
</head>
<body>
  <script type="text/javascript">

    var Text = "Frau Klößer hat mehrere Buchbestellungen im Wert von
49.90€, 149.90€, 65.35€ und 400.00€ aufgegeben.";

    document.write(Text + "<br><br><b>Die Bestellwerte im
einzeln:<\/b><br><hr>");

    SuchAusdruck = /\d*.\d*€/g;
    Bestellwerte = Text.match(SuchAusdruck);

    for(i = 0; i < Bestellwerte.length; i++)
      document.write(Bestellwerte[i] + "<br>");

  </script>
</body>
</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

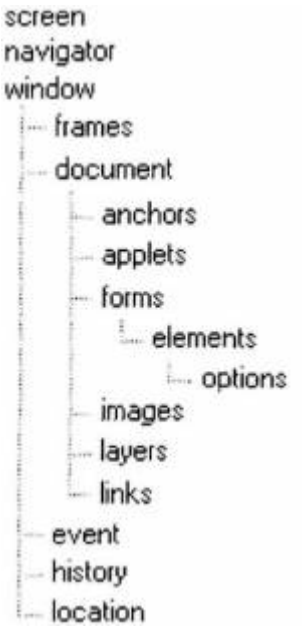
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_08

Last update: **2019/05/16 14:50**

11.3.9) Objektmodell

Hierarchie der JavaScript-Objekte

In JavaScript werden dem Benutzer neben den vordefinierten Objekten wie stringund Arrayweitere Objekte zur Verfügung gestellt, über die wir beispielsweise auf ein Element eines Formulars zugreifen und dieses ändern können. Diese JavaScript-Objekte sind in einer Hierarchie geordnet. Die hierarchiehöchsten Objekte sind das Fensterobjekt window, das Navigatorobjekt navigator und das Bildschirmobjekt screen.



Der Inhalt eines Fensters und somit das HTML-Dokument werden durch das document-Objekt repräsentiert. Dieses Objekt ist dem window-Objekt untergeordnet. Das HTML-Dokument enthält Elemente, wie zum Beispiel Grafiken, Verweise und Formulare. Für diese Elemente gibt es weitere Objekte, beispielsweise das Objekt `forms` für Formulare. Um auf die Elemente des Formulars (Eingabefelder, Auswahllisten oder Schaltflächen) zuzugreifen, existiert das Objekt `elements`, mit dem wir einzelne Felder und andere Elemente innerhalb eines Formulars ansprechen können.

Das `navigator`-Objekt und das `screen`-Objekt existieren unabhängig vom window-Objekt und geben Auskunft über den Browser, in dem das Skript läuft, bzw. über die Einstellungen des Computer-Bildschirms.

Aufgrund der großen Menge an Objekten werden wir anhand von ausgesuchten Beispielen im Folgenden die wichtigsten Objekte und deren Eigenschaften und Methoden erläutert.

Weitergehende Informationen zu allen JavaScript-Objekten finden sich hier.

Objekt navigator

Um die Eigenschaften eines Browsers auszulesen, wird das Objekt navigator angewendet.

Eigenschaft	Bedeutung
`userAgent`	Hiermit erhalten Sie die vollständige Browser-Bezeichnung, die dem Webserver übergeben wird.
`platform`	Dahinter verbirgt sich die verwendete Computer-Plattform.
`language`	Nur Firefox und Opera: Damit können Sie die Spracheinstellung des Client-Computers auslesen. Der Internet Explorer besitzt die Eigenschaften `systemLanguage` und `userLanguage`.
`plugins`	Nur Firefox und Opera: Diese Eigenschaft liefert ein Feld zurück, das alle installierten Plug-ins auflistet.
`mimeTypes`	Das Feld beinhaltet alle MIME-Typen, die vom Browser verarbeitet werden können.

Methode	Bedeutung
`javaEnabled()`	Diese Methode liefert den Wert `true`, wenn eine Java-Unterstützung vorhanden und eingeschaltet ist, sonst `false`.

Um zu ermitteln, welchen Browser der Besucher der Webseite verwendet, wird der Wert der Eigenschaft `appName` überprüft. Diese gibt den Namen des Browsers zurück. Eine weitere Spezifizierung kann mit der Nutzung von `appVersion` durchgeführt werden, da diese Eigenschaft die spezielle Versionsnummer des Browsers angibt.

Die Browser besitzen meist noch weitere individuelle Eigenschaften, die wir durch das Ausführen des folgenden Beispiels anzeigen können.

Beispiel: objekt_navigator.html

Dieses Beispiel liefert Angaben zu allen Eigenschaften des `navigator`-Objekts des verwendeten Browsers und den Rückgabewert der Methode `javaEnabled()`.

objekt_navigator.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Das Objekt navigator</title>
</head>
<body>
  <h3>Eigenschaften des navigator-Objekts</h3>
  <script type="text/javascript">
    for(eig in navigator)
      document.write("<b>" + eig + ":<\/b> " + navigator[eig] +
"<br>");
    document.write("<b>javaEnabled() :<\/b> " +
navigator.javaEnabled());
  </script>
</body>
</html>
```

Zu beachten ist, dass in den verschiedenen Browsern nicht alle Eigenschaften gleich verhanden sind und daher unterschiedliche Ergebnisse angezeigt werden.

Diese Informationen sind dann wichtig, wenn Verzweigungen für browserspezifische JavaScript-Anweisungen benötigt werden. Die weitverbreiteten Browser Internet Explorer und Mozilla Firefox weisen auch in den neuen Versionen Unterschiede in ihren Implementierungen der HTML- und JavaScript-Standards auf. Somit ist das Verzweigen in unterschiedliche Codesegmente nötig, um nicht eine Fehlermeldung in dem einen oder anderen Browser zu erhalten.

Beispiel: objekt_navigator_weiche.html

Ein typischer Einsatzzweck für das Objekt `navigator` ist die sogenannte Browserweiche, in der die verwendeten Browser abgefragt und speziell für die verschiedenen Browser angepasste Funktionen ausgeführt werden. Statt der hier angegebenen Funktion `alert()` können andere Funktionen ausgeführt werden.

objekt_navigator_weiche.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Browserweiche</title>
</head>
<body>
  <h3>Browserweiche</h3>
  <p>Dieses Skript dient zum Ermitteln des vom Nutzer verwendeten
Browsers.</p>
  <script type="text/javascript">

    if(parseInt(navigator.appVersion)>= 4)
    {
      if(navigator.appName == "Netscape")
      {
         alert("Sie verwenden den Mozilla Firefox.");
         document.write("Sie verwenden den Mozilla Firefox.");
      }
      else if(navigator.appName.indexOf("Internet Explorer") != -1)
      {
         alert("Sie verwenden den Internet Explorer.");
         document.write("Sie verwenden den Internet Explorer.");

      }
      else if(navigator.appName == "Opera")
      {
         alert("Sie verwenden den Opera-Browser.");
         document.write("Sie verwenden den Opera-Browser.");
      }
      else
      {
         alert("Sie verwenden einen anderen Browser.");
```

```
          document.write("Sie verwenden einen anderen Browser.");
        }
      }
      else
      {
        alert("Sie verwenden einen älteren Browser.");
      }
    </script>

</body>
</html>
```

Objekt window

Das Objekt `window` (Fenster) steht in der Objekthierarchie an oberster Stelle. Alle anderen Objekte, die das im Fenster geladene Dokument beschreiben, sind Eigenschaften des `window`-Objekts. Es erlaubt uns somit die Abfragen zu den einzelnen Dokumentfenstern und ermöglicht uns, neue Fenster zu öffnen sowie deren Eigenschaften (Name, Höhe, Breite usw.) festzulegen. Auch das Schließen von Fenstern ist hiermit möglich.



- Ist das Browserfenster in Frames unterteilt, befinden sich die Objekte der HTML-Dokumente in einer eigenen Hierarchie unterhalb eines frames-Objekts.
- Das document-Objekt enthält alle wesentlichen HTML-Elemente wie Links, Anker, Bilder und Formulare des betreffenden HTML-Dokuments.
- Das history-Objekt enthält die Liste der zuletzt geladenen URLs.
- Das location-Objekt enthält Informationen über die URL des aktuellen Dokuments.

Methoden des window-Objekts

Methoden des `window`-Objekts

Methode	Bedeutung
`alert()`	Es wird ein Meldungsfenster angezeigt.
`blur()`	Diese Methode deaktiviert das betreffende Browserfenster. Es wird in den Hintergrund verschoben.
`clearTimeout(TimeoutID)`	Die laufende Stoppuhr, die mit `TimeoutID = setTimeout()` gestartet wurde, wird angehalten.

Methode	Bedeutung
`close()`	Damit schließen Sie ein Browserfenster.
`confirm()`	Es wird ein Bestätigungsfenster angezeigt.
`focus()`	Hiermit aktivieren Sie das Browserfenster. Es wird in den Vordergrund gebracht.
`open("URL","Fenstername", "Optionen")`	Mit der Methode `open()` öffnen Sie ein neues Browserfenster.
`prompt()`	Zur Eingabe eines Textes wird ein Eingabefenster angezeigt.
`setTimeout("JavaScript-Ausdruck",Zeit)`	Nach der in Millisekunden angegebenen Zeit wird der JavaScript-Ausdruck ausgeführt. Als Rückgabewert der Methode wird eine `TimeoutID` geliefert, mit der sich die laufende Stoppuhr referenzieren lässt.

Eigenschaften des `window`-Objekts

Eigenschaft	Bedeutung
`defaultStatus`	Damit können Sie einen Text in der Statuszeile des Browsers als Standardtext festlegen.
`opener`	Die Eigenschaft `opener` enthält die Referenz auf das Fenster, welches das aktuelle Fenster geöffnet hat.
`self`	Die Eigenschaft `self` enthält immer die Referenz auf das aktuelle Objekt.
`status`	Hiermit ändern Sie den derzeit dargestellten Text der Statuszeile.

Standardmäßig müssen wir zum Aufruf einer Methode eines Objekts den Objektnamen voranstellen, z.B. `window.alert()`. Im Falle des `window`-Objekts können wir den Namen `window` weglassen. Dieser wird automatisch von JavaScript vorangesetzt.

Statusanzeige

Standardmäßig ist der Browser für die Textanzeige der Statuszeile verantwortlich. Nach dem Laden einer Webseite zeigt er z. B. den Text `Fertig` an oder beim Überfahren eines Hyperlinks mit der Maus die betreffende URL.

Standardmäßig können wir den Statusleistentext im Mozilla Firefox nicht per JavaScript ändern. Hierzu müssen wir in den erweiterten JavaScript-Einstellungen des Browsers das Überschreiben des Statusleistentextes explizit erlauben.

Beispiel: objekt_window_status.html

Diese Statusanzeige soll von Ihnen mit einem individuellen Text überschrieben werden. Zusätzlich soll beim Überfahren eines Links mit der Maus eine spezielle Anzeige erfolgen. Um die Statusanzeige anzusprechen, bietet das Objekt `window` die Eigenschaften `defaultstatus` (standardmäßige Statusanzeige) und `status` (aktuelle Statusanzeige).

objekt_window_status.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Statusanzeige beeinflussen</title>
```

```
  <script type="text/javascript">

    window.defaultStatus="Willkommen auf meiner Homepage";

  </script>
</head>
<body>
  <h3>Statusanzeige &auml;ndern</h3>
  F&uuml;hren Sie die Maus über den Link und beobachten Sie die
Statuszeile.<br>

  <a href="#" onMouseOver="window.status='Statuszeile wurde
überschrieben'; return true;">
  &Auml;ndern der Statuszeile</a>
</body>
</html>
```

Vorheriges Dokument aufrufen

Im Browser können wir mit der Schaltfläche Zurück zum vorherigen HTML-Dokument wechseln. Die entsprechende Methode nennt sich `back()`. Sie wird sehr oft in Zusammenhang mit einem Verweis angewendet. Diese Methode funktioniert nur, wenn sich vor der Webseite im Browserverlauf eine weitere Webseite befindet.

Beispiel: objekt_window_back.html

objekt_window_back.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <title>Objekt:window, Methode:back()</title>
</head>
<body>
  <a href="javascript:window.back()">Zur vorherigen Seite</a>
</body>
</html>
```

Fenster öffnen und schließen

Beim Browsen im Internet werden bei vielen Webseiten automatisch weitere Fenster, meist Werbeeinblendungen, geöffnet. Dazu wird die Methode `open()` verwendet.

```
window.open("URL","Fenstername",[Optionen])
```

Beispiel: objekt_window_open.html

Beim Laden der Webseite sollen zwei weitere Fenster per JavaScript geöffnet werden. Damit wir die geöffneten Fenster per JavaScript steuern können, werden die Referenzen auf die window-Objekte der neuen Fenster in Variablen gespeichert.

objekt_window_open.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt:window, Methode:open()</title>
</head>
<body>
  <h3>Das Hauptfenster</h3>
  <script type="text/javascript">

    Fenster1 = window.open("objekt_window_status.html", "FensterEins");
    Fenster2 = window.open("about:blank", "FensterZwei");

  </script>
</body>
</html>
```

Beim Öffnen eines neuen Fensters können wir weitere Optionen für die Anzeige des Fensters festlegen, z.B. ob im geöffneten Fenster die Statuszeile oder die Adresszeile angezeigt werden soll.

Option	Erklärung
menubar	Anzeige der Menüleiste ① (yes, no = Standard)
toolbar	Anzeige der Symbolleiste ② (yes, no = Standard)
locationbar	Anzeige der Adresszeile ③ (yes, no = Standard)
directories	Anzeige der persönlichen Symbolleiste ④ (yes, no = Standard)
status	Anzeige der Statuszeile ⑤ (yes, no = Standard)
resizable	Fenstergröße ⑥ veränderbar (yes = Standard, no)
scrollbars	Bildlaufleisten ⑦ einblenden (yes, no = Standard). Firefox und Opera blenden sie erst ein, wenn der Inhalt der Webseite entsprechend lang ist.
height	Festlegung der Höhe des Fensters in Pixeln
width	Festlegung der Breite des Fensters in Pixeln
top	Anzeigeposition von oben
left	Anzeigeposition von links

Beispiel

Wir öffnen ein Fenster, in dem die Anzeige aller Fensterelemente abgeschaltet ist. Die Höhe und die Breite des Fensters sollen dabei jeweils 300 Pixel betragen.

```
<script   type="text/JavaScript">
  Fenster =  window.open("","Individuell",
                         "menubar=no,locationbar=no,resizable=no,
                          status=no,height=300,width=300");
</script>
```

Zu beachten ist, dass die Fensteroptionen in einer Zeichenkette angegeben werden. Die einzelnen Optionen werden jeweils durch ein Komma voneinander getrennt. Ein Fenster können wir über die Methode `window.close()` schließen. Beispielsweise lässt sich diese Funktion als Link im HTML-Dokument darstellen.

```
<a href="JavaScript:window.close();">Aktuelles  Fenster  schließen</a>
//   oder
<a href="JavaScript:seif.close();">Aktuelles Fenster schließen</a>
```

- Mehr zum Öffnen und Schließen von Fenstern im Browser

Zeitgeber verwenden

Mit der Methode `setTimeout()` haben wir die Möglichkeit, eine Anweisung oder Funktion nach einer bestimmten Verzögerungszeit auszuführen. Dazu geben wir innerhalb der Klammern den entsprechenden JavaScript-Code an. Als zweite Angabe folgt die Verzögerungszeit in Millisekunden.

Beispiel: objekt_window_settimeout.html

Durch die Anweisung `setTimeout(„Zeitvorbei()“,5000)` wird nach einer Zeit von 5 Sekunden (5.000 Millisekunden) die Funktion `Zeitvorbei()` aufgerufen.

objekt_window_settimeout.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt:window, Methode:setTimeout()</title>
</head>
<body>
  <h3>Bitte warten, der Countdown läuft</h3>
  <script type="text/javascript">

    function Zeitvorbei()
    {
      window.alert("Die Zeit ist abgelaufen...");
      return;
    }
    window.setTimeout("Zeitvorbei()", 5000);
```

```
  </script>
</body>
</html>
```

Meldungsfenster

Meldungsfenster zeigen eine Information an, die der Benutzer mit einem Klick auf die Schaltfläche Ok bestätigen muss. Die Methode `alert()` (Alarm) öffnet ein Dialogfenster. Innerhalb der Anführungszeichen können wir eine Zeichenkette angeben, die als Meldung angezeigt werden soll. Möchten wir einen Zeilenumbruch einfügen, geben wir die Escapesequenz `\n` innerhalb des Textes an.

```
window.alert("Anzeige");
```

Beispiel: interaktion_alert.html

interaktion_alert.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Interaktion - Dialogfenster</title>
</head>
<body>
  <script type="text/javascript">

    alert("Dies ist kein Fehler, \nsondern nur ein Test!");

  </script>
</body>
</html>
```

Eingabefenster

In einem Eingabefenster kann der Benutzer einen Text eingeben, der danach einer Variablen zugewiesen wird und weiterverarbeitet werden kann. Die Methode `prompt()` (Eingabe) öffnet ein Eingabefenster. Als ersten Wert geben wir den Text an, der beschreibt, was für ein Wert eingegeben werden soll. Mit dem zweiten Parameter legen wir fest, welcher Vorgabewert im Eingabefeld angezeigt wird.

```
window.prompt("Anzeige", "Vorgabewert");
```

In einem Eingabefenster wird der Name des Benutzers eingegeben, der im darauf folgenden Meldungsfenster vom Programm wieder ausgegeben wird.

Beispiel: interaktion_prompt.html

interaktion_prompt.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Interaktion</title>
</head>
<body>
  <script type="text/javascript">
      name = prompt("Bitte geben Sie Ihren Namen an:", "");
      alert("Ihr Name ist '" + name + "'.");
  </script>
</body>
</html>
```

Bestätigungsfenster

Über dieses Dialogfenster können wir eine Frage anzeigen, die der Benutzer über die Schaltflächen Ok und Abbrechen beantworten kann. Die Methode `confirm()` (Bestätigung) zeigt das Dialogfenster an. Als Parameter geben wir die anzuzeigende Frage an.

```
window.confirm("Frage");
```

Beispiel: interaktion_confirm.html

In Abhängigkeit von der Beantwortung der Frage soll eine entsprechende Meldung angezeigt werden.

interaktion_confirm.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Interaktion</title>
</head>
<body>
  <script type="text/javascript">

      wahl = confirm("Sind Sie das erste Mal hier?");
```

```
      if(wahl == true)
        alert("Sie haben die Frage mit OK bestätigt!")
      else
        alert("Sie haben die Frage mit NEIN beantwortet!");

  </script>
</body>
</html>
```

Beispiel: interaktion_komplett.html

Das folgende Beispiel verwendet alle bisher vorgestellten Dialogfenster. Dabei soll der Name des Besuchers erfragt und auf der Webseite ausgegeben werden. Gibt er keinen Namen ein, wird ihm der Name Herr Namenlos zugewiesen.

interaktion_komplett.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Interaktion</title>
</head>
<body>
  <h3>Meine Homepage</h3>
  <script type="text/javascript">

      var name = "";
      janein = confirm("Möchten Sie mir Ihren Namen verraten?");
      if (janein == true)
        name = prompt("Wie ist Ihr Name?", "");

      if ((name == "") || (name == null))
      {
        alert("Name unbekannt.");
        name = "";
      }

      document.write("Hallo " + name + ",<br>willkommen auf meiner
Webseite.");
  </script>
</body>
</html>
```

Objekt screen

Nicht alle Eigenschaften des `screen`-Objekts sind sowohl im Internet Explorer als auch in Firefox

implementiert. Für die häufigste Anwendung des `screen`-Objekts, das Öffnen neuer Fenster an der richtigen Position und in der richtigen Größe, sind Eigenschaften in beiden Browsern vorhanden.

Eigenschaft(en)	Bedeutung
`height, width`	Über diese Eigenschaften lassen sich die Höhe und Breite der eingestellten Bildschirmauflösung in Pixeln bestimmen.
`availHeight, availWidth`	Die Ausmaße der wirklich zur Verfügung stehenden Fläche auf dem Bildschirm, ohne die Windows-Taskleiste oder die Macintosh-Menüleiste, erhalten Sie über diese Eigenschaften.
`colorDepth`	Die verwendete Farbtiefe wird in Bits pro Pixel ausgelesen.

Beispiel: objekt_screen.html

Es werden die ermittelten Bildschirmwerte ausgegeben.

objekt_screen.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt: screen</title>
</head>
<body>
  <h3>Eigenschaften des screen-Objekts</h3>
  <script type="text/javascript">
    document.write("<p><b>H&ouml;he & Breite des Bildschirms mit
screen.height & screen.width:<\/b><br>" + screen.height + " x " +
screen.width + "<br>");
    document.write("<b>Sichtbare H&ouml;he & Breite mit
screen.availHeight & screen.availWidth:<\/b><br>" + screen.availHeight
+ " x " + screen.availWidth + "<br>");
    document.write("<b>Verwendete Farbtiefe mit
screen.colorDepth:<\/b><br>" + screen.colorDepth + "Bit");
  </script>
</body>
</html>
```

Beispiel: objekt_screen2.html

Eine häufige Verwendung des `screen`-Objekts ist die Zentrierung eines Fensters in Abhängigkeit der vom Besucher verwendeten Bildschirmauflösung. Durch das Betätigen eines der Links wird ein leeres Fenster in der angegebenen Auflösung zentriert auf dem Bildschirm angezeigt.

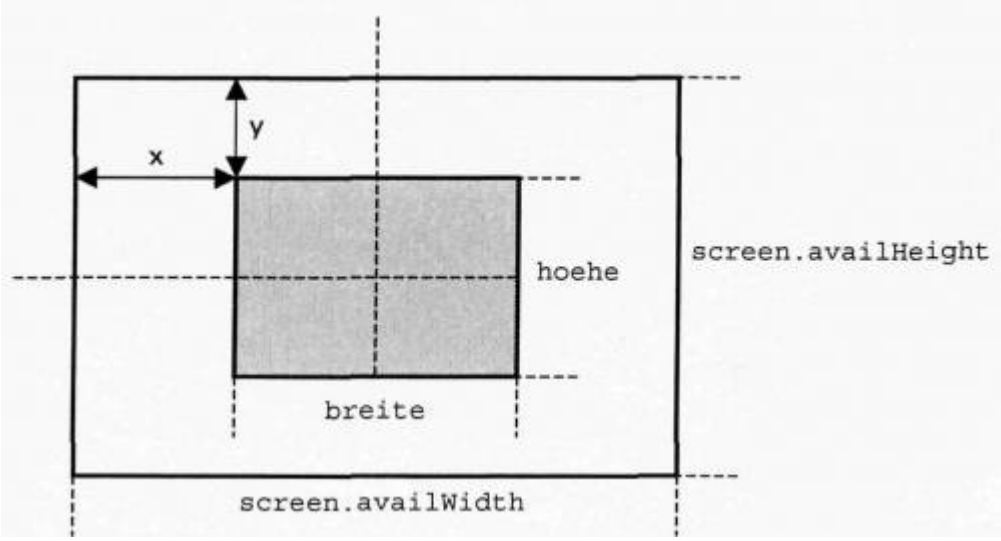
objekt_screen2.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
```

```
<head>
  <title>Objekt: screen</title>
  <script type="text/javascript">
    function openCentered(URL, breite, hoehe)
    {
      var PopupX = (screen.availWidth / 2) - (breite / 2);
      var PopupY = (screen.availHeight / 2) - (hoehe / 2);

      window.open(URL, "", "resizable=no, locationbar=no, width=" +
breite + ", height=" + hoehe + ", left=" + PopupX + ", top=" + PopupY);
    }
  </script>
</head>
<body>
  <h3>Fenster zentriert in folgenden Gr&ouml;&szlig;en &ouml;ffnen</h3>
  <p><a href="javascript:openCentered('about:blank', 300, 300);">300 x
300</a>

  <p><a href="javascript:openCentered('about:blank', 640, 480);">640 x
480</a>
  <p><a href="javascript:openCentered('about:blank', 800, 600);">800 x
600</a>
  <p><a href="javascript:openCentered('about:blank', 200, 600);">200 x
600</a>
</body>
</html>
```



Berechnung der Werte, um das Browserfenster zentriert darzustellen.

Objekt document

Das `document`-Objekt ist das komplexeste Unterobjekt des `window`-Objekts. Es enthält eine Vielzahl weiterer Eigenschaften, über die wir auf die Elemente eines HTML-Dokuments zugreifen. So enthält das `document`-Objekt in der Eigenschaft `forms` weitere Objektreferenzen, sobald sich ein HTML-

Formular `<form>…</form>` auf einer Webseite befindet. Weitere Eigenschaften sind `anchors`- (Verweisanker), `applets`- (Java-Applets), `images`- (Grafiken), `layers`- (Ebenen) und `links`-Objekte (Verweise).

In diesem Abschnitt werden die Eigenschaften und Methoden erläutert, die sich nicht auf HTML-Elemente beziehen.

Eigenschaften

Eigenschaft	Bedeutung
`bgColor`	Die Hintergrundfarbe der Webseite kann ausgelesen und verändert werden.
`fgColor`	Die Textfarbe kann ausgelesen und verändert werden.
`linkColor`	Die Farbe eines noch nicht besuchten Links im Text lässt sich auslesen und ändern.
`alinkColor`	Sie legt die Farbe eines aktiven Hyperlinks fest oder liest sie aus.
`vlinkColor`	Hiermit legen Sie die Farbe eines bereits besuchten Hyperlinks fest.
`title`	Mit der Eigenschaft `title` lässt sich der Dokumenttitel auslesen bzw. setzen.
`referrer`	Zurückgeliefert wird die URL des Dokuments, das per Hyperlink auf das aktuelle Dokument verwiesen hat. Der Referrer wird nur übermittelt, wenn sich die Webseiten auf einem Webserver befinden.
`lastModified`	Damit erhalten Sie das Datum, an dem das Dokument zuletzt verändert wurde.
`cookie`	Hiermit haben Sie Zugriff auf Cookies, die lokal abgespeichert werden und Parameter-Paare in Form von Zeichenketten enthalten können. Cookies können gelesen und verändert werden.
`URL`	Hiermit erhalten Sie die URL des geladenen Dokuments. `document.URL` ist ein Synonym für die Eigenschaft `document.location`, um bei der Anwendung eine Verwechslung mit `window.location` zu vermeiden. Denn `window.location` beinhaltet die vom Nutzer angeforderte URL, und `document.URL` beinhaltet die an ihn zurückgelieferte URL. Diese können sich durch Weiterleitungen (Redirects) voneinander unterscheiden.

Beispiel: objekt_document.html

Die meisten der Eigenschaften entsprechen den Tags in HTML. Über diese Eigenschaften werden im Beispiel die Werte der Tags einer Webseite ausgelesen.

objekt_document.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt: document</title>
</head>
<body>
  <h3>Eigenschaften des document-Objekts</h3>
  <script type="text/javascript">
    document.write("<b>Titel der Webseite:<\/b> mit document.title =" +
document.title + "<br>");
    document.write("<b>Adresse der Webseite:<\/b> mit document.location
```

```
= " + document.location + "<br>");
    document.write("<b>Adresse der Webseite:<\/b> mit document.URL = "
+ document.URL + "<br>");
    document.write("<b>Farbe des Hintergrunds:<\/b> mit
document.bgColor = " + document.bgColor + "<br>");
    document.write("<b>Farbe der Schrift:<\/b> mit document.fgColor = "
+ document.fgColor + "<br>");

    document.write("<b>Farbe der Verweise:<\/b> mit document.linkColor
= " + document.linkColor + "<br>");
    document.write("<b>Farbe der aktivierten Verweise:<\/b> mit
document.alinkColor = " + document.alinkColor + "<br>");
    document.write("<b>Farbe der besuchten Verweise:<\/b> mit
document.vlinkColor = " + document.vlinkColor + "<br>");
    document.write("<b>Seite wurde von Adresse aufgerufen:<\/b> mit
document.referrer = " + document.referrer + "<br>");
    document.write("<b>Letzte Änderung der Seite:<\/b> mit
document.lastModified = " + document.lastModified + "<br>");

    document.write("<b>anchors-Objekt:<\/b> mit document.anchors = " +
document.anchors + "<br>");
    document.write("<b>applets-Objekt:<\/b> mit document.applets = " +
document.applets + "<br>");
    document.write("<b>embeds-Objekt:<\/b> mit document.embeds = " +
document.embeds + "<br>");
    document.write("<b>forms-Objekt:<\/b> mit document.forms = " +
document.forms + "<br>");
    document.write("<b>images-Objekt:<\/b> mit document.images = " +
document.images + "<br>");
    document.write("<b>all-Objekt:<\/b> mit document.all = " +
document.all + " (nur IE)<br>");
    document.write("<b>links-Objekt:<\/b> mit document.links = " +
document.links + "<br>");
    document.write("<b>plugins-Objekt:<\/b> mit document.plugins = " +
document.plugins + "<br>");
  </script>
</body>
</html>
```

Methode	Bedeutung
`write(Text)`	Über diese Methode schreiben Sie Text in das HTML-Dokument. Dieser wird an der Stelle eingefügt, an der die Methode aufgerufen wird.
`writeln(Text)`	Diese Methode arbeitet wie die Methode `write`. Nach dem Text wird zusätzlich ein Zeilenumbruch eingefügt. Dieser Umbruch ist nur im Quelltext der Webseite sichtbar und entspricht nicht dem HTML-Tag ` `.
`open()`	Damit löschen Sie die aktuelle Webseite und öffnen das Dokument für neue Inhalte.
`close()`	Die Methode `close()` schließt das Dokument, sodass keine weiteren Inhalte eingefügt werden können.
`clear()`	Hiermit löschen Sie den Inhalt der aktuellen Webseite.
`getElementById()`	Damit können Sie auf ein HTML-Element mit einem im Attribut `id` festgelegten Namen zugreifen.
`getElementsByName()`	Wie `getElementById()`, nur dass hier das Attribut `name` ausgewertet wird.
`getElementsByTagName()`	Hiermit erhalten Sie ein Array der als Namen angegebenen HTML-Elemente, z. B. `h1`, `p`.

HTML-Elemente als Objekte von document

Der Zugriff auf die Elemente der Dokumenthierarchie erfolgt in der Regel über die im HTML-Quelltext vergebenen Namen. Zusätzlich werden die Objekte der Elemente in Feldern angelegt.

- `window.document.images[]` enthält alle Grafiken in der Reihenfolge, in der die <img>-Tags im HTML-Quelltext verwendet werden.
- `window.document.anchors[]` enthält alle Anker entsprechend den Tags <a name=…>.
- `window.document.links[]` enthält alle Links entsprechend den Tags <a href =…>.
- `window.document.forms[]` enthält alle Formulare, die durch <form>-Tags definiert wurden.

Auf die einzelnen Unterobjekte des Objekts `document` wird im nächsten Abschnitt näher eingegangen.

Objekt history

Eine weitere Eigenschaft des `window`-Objekts ist das `history`-Objekt (history = Geschichte). Laden wir eine Webseite in den Browser, wird die URL im Verlauf des Browsers gespeichert (im Firefox als Chronik bezeichnet). Dies können wir überprüfen, indem wir mit der Maus auf den nach unten weisenden Pfeil neben den Navigationsschaltflächen des Browsers klicken. Es wird eine Liste der zuletzt besuchten Webseiten angezeigt.

Mit diesem Objekt erhalten wir einen bedingten Zugriff auf diese gespeicherten URLs. Bedingt heißt, dass wir nur vorwärts und rückwärts auf die Einträge zugreifen können, jedoch nicht direkt. Dies dient dem Schutz der Privatsphare des Nutzers.

Eigenschaft	Bedeutung
length	Diese Eigenschaft gibt die Anzahl der Einträge in der History des Browsers wieder.

Methode	Bedeutung
back()	Die Methode lädt das zuvor besuchte Dokument.
forward()	Die Methode lädt das nächste Dokument innerhalb der History.
go(Schritte)	Bei der Methode go() können Sie eine bestimmte Anzahl an Seiten überspringen. Eine negative Zahlenangabe (go(-2)) lädt eine der vorherigen Seiten, eine positive Angabe (go(2)) lädt eine der nachfolgenden Seiten.

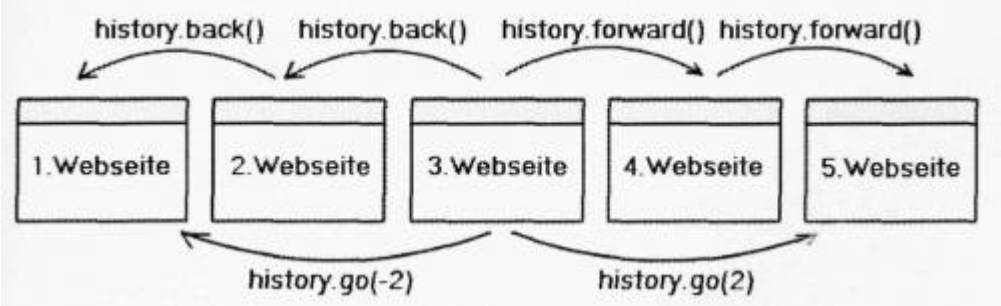
Wird `back()` beim ersten und `forward()` beim letzten Eintrag in der History aufgerufen, haben diese Methoden keine Auswirkung.

Beispiel: objekt_history.html

Auf einer Webseite fügen wir vier Hyperlinks ein, mit denen wir in der History des Browsers navigieren können. Wir müssen dazu im Browserfenster bereits einige Webseiten besucht haben, um diese Funktionalität zu nutzen.

objekt_history.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt: history</title>
</head>
<body>
  <h3>Eigenschaften des history-Objekts</h3>
  <p>&lt; <a href="javascript:history.back();">Eine Seite
zurückblättern</a> (<code>back()</code>)<br>
  <p>&gt; <a href="javascript:history.forward();">Eine Seite
vorwärtsblättern</a> (<code>forward()</code>)<br>
  <p>&lt;&lt; <a href="javascript:history.go(-2);">Zwei Seiten
zurückblättern</a> (<code>go(-2)</code>)<br>
  <p>&gt;&gt; <a href="javascript:history.go(2);">Zwei Seiten
vorwärtsblättern</a> (<code>go(2)</code>)
</body>
</html>
```



Mit den Methoden des `history`-Objekts können wir durch die History der besuchten Webseiten navigieren. Dies ist jedoch erst möglich, wenn wir bereits einige Webseiten mit dem Browser besucht haben und dann die betreffende Webseite im Browser laden. Zur Ausführung von `history.go(2)` müssen wir mindestens zwei weitere Webseiten besuchen und sich wieder zur Webseite des Beispiels in der `History` zurückbewegen.

Objekt location

Das `location`-Objekt (location = Ort) ist eine Eigenschaft des `window`-Objekts. Mit diesem Objekt können wir auf die URL des aktuell dargestellten Dokuments lesend und schreibend zugreifen. Wir können dabei die gesamte URL oder auch einzelne Teile davon auslesen und verarbeiten.

Eigenschaft	Bedeutung
`protocol`	Hiermit erhalten Sie das verwendete Internet-Protokoll, mit dem die Webseite aufgerufen wurde, z. B. `http:` oder `ftp:`.
`hostname`	Den Host- und Domain-Namen können Sie mit dieser Eigenschaft auslesen, z. B. `www.example.com`.
`port`	Die Port-Nummer des Servers wird über die Eigenschaft `port` zurückgeliefert.
`host`	Diese Eigenschaft entspricht den Angaben von `hostname:port`.
`pathname`	Die Pfadangabe des Dokuments wird über `pathname` ausgelesen.
`search`	Es wird die Zeichenkette der an die Webseite übergebenen Parameter geliefert, z. B. `?suche=test`.
`hash`	Falls in der URL ein Anker angegeben wurde, können Sie dies über diese Eigenschaft ermitteln, z. B. `#anker`.
`href`	Die vollständige URL wird ausgelesen.
Methode	**Bedeutung**
`reload()`	Die Methode lädt das Dokument erneut, wie beim Klick auf die Browser-Schaltfläche AKTUELLE SEITE NEU LADEN.
`replace()`	Lädt eine Seite und überschreibt den aktuellen History-Eintrag des Browsers.

Beispiel: objekt_location.html

In diesem Beispiel werden die Eigenschaften des `location`-Objekts in einer Tabelle ausgegeben. Damit die meisten Eigenschaften des Objekts location einen Wert zurückliefern, laden wir die Webseite über einen Webserver und rufen wir diese dort auf. Zusätzlich geben wir in der URL einen Anker sowie Parameter an.

```
<h3>Eigenschaften des  Iocation-0bjekts</h3>
<script type="text/JavaScript">
for(eig in location)
  document.write("<b>" + eig + ":</b>" + location[eig]+"<\/br>"); </script>
```

Beispiel: objekt_location_redirect.html

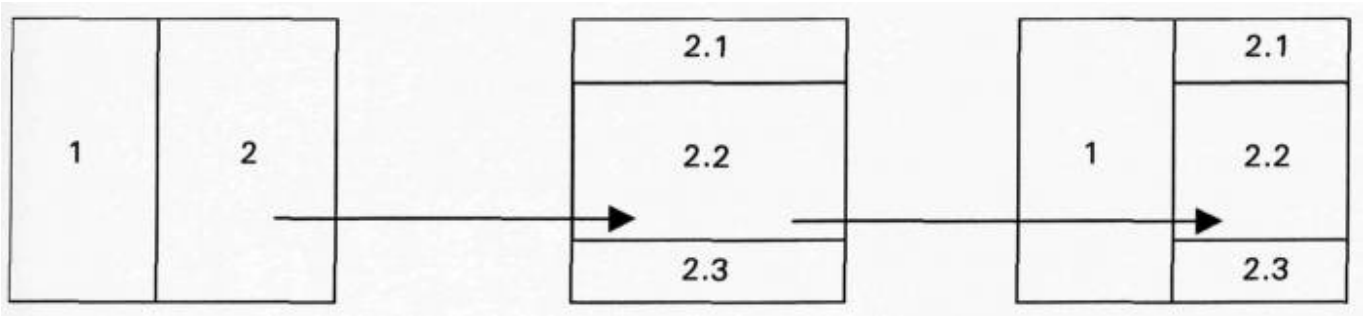
Durch das Setzen der Eigenschaft `href` kann der Browser über JavaScript veranlasst werden, eine neue Seite zu laden. Ein Anwendungsbeispiel für das Objekt `location` findet sich in einer URL-Weiterleitung. Beispielsweise können wir den Besuchern eine bestimmte Webseite anzeigen, wenn kein JavaScript aktiviert ist. Den Benutzern mit aktiviertem JavaScript bieten wir eine separate Webseite an, auf die automatisch weitergeleitet wird, z.B. `objekt_location_mit_js.html`.

```
<html>
<head>
  <title>Objekt: location</title>
</head>
<body onLoad="window.location.href='objekt_location_mit_js.html'">
  <h3>Bei Ihnen ist JavaScript deaktiviert.</h3>
</body>
</html>
```

Objekt frames

In einem Fenster können mehrere Frames enthalten sein. In jeden Frame kann wiederum ein HTML-Dokument geladen werden, in dem erneut Frames definiert sind.

Das `frames`-Objekt ist eine Eigenschaft des `window`-Objekts, das auch die Objekte `document, history, location` enthalten kann. Auf die einzelnen Unterframes können wird über das `frames`-Feld zugreifen.



Um einen Frame ansprechen zu können, geben wir entweder den Namen des Frames mit `window.FrameName` an oder verwenden wir einen Index. Zu beachten, dass der Zähler bei null beginnt, d. h., das erste Frame-Fenster sprechen wir mit `frames[0]` an, das zweite Frame-Fenster mit `frames[1]` usw. Beim Zählen gilt die Reihenfolge, in der die Frames im Frameset definiert sind.

Die Ermittlung der Eigenschaften eines Framesets soll Ihnen anhand eines Beispiels verdeutlicht werden.



Aufteilung eines Framesets in vier Frames

Beispiel: objekt_frames.html

Zuerst erstellen wir eine Webseite mit einem Frameset, das wiederum zwei Frames beinhaltet:

objekt_history.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
            "http://www.w3.org/TR/html4/frameset.dtd">
<html lang="de">
<head>
  <title>Objekt: frames</title>
</head>
<frameset cols="50%,*">
  <frame name="FLinks" src="flinks.html">   <!-- Frame 1 -->
  <frame name="FRechts" src="frechts.html"> <!-- Frame 2 -->
</frameset>
</html>
```

Beispiel: frechts.html

Wir erstellen den rechten Frame(2), der als Frameset für weitere drei Frames genutzt wird.

frechts.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
            "http://www.w3.org/TR/html4/frameset.dtd">
<head>
  <title>Objekt: frames</title>
</head>
<frameset rows="1,1,1">
  <frame name="FOben" src="foben.html">       <!-- Frame 2.1 -->
  <frame name="FZentrum" src="fzentrum.html"> <!-- Frame 2.2 -->
  <frame name="FUnten" src="funten.html">     <!-- Frame 2.3 -->
</frameset>
</html>
```

Beispiel: fzentrum.html, foben.html, funten.html, flinks.html

Nachfolgend wird der Code der HTML-Dokumente des rechten Frames näher betrachtet. Alle weiteren Frames `foben.html`, `funten.htm` und `flinks.html` enthalten die gleichen Anweisungen.

objekt_history.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt: frames</title>
  <script type="text/javascript">
      function ausgabe()
      {
        document.write("Frame 2.2<br>");
        document.write("Das &uuml;bergeordnete Dokument hei&szlig;t: "
+ parent.name + "<br>");
        document.write("Frames im &uuml;bergeordneten Dokument: " +
parent.length + "<br>");
        document.write("Diese lauten: ");
        for(i = 0; i < parent.frames.length; ++i)
           document.write(parent.frames[i].name + " ");
        document.write("<br>");
        document.write("Dieses Frame hei&szlig;t: " + self.name +
"<br>");
        document.write("Frames im Hauptdokument: " + top.length +
"<br>");
      }
  </script>
</head>
<body onLoad="ausgabe();">
  <noscript>Aktivieren Sie JavaScript, um dieses Beispiel anzuzeigen.
  </noscript>
</body>
</html>
```

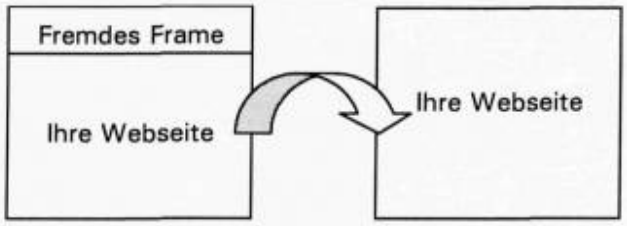
Durch das Auslesen der Frames-Eigenschaften können wir gezielt auf die Frames zugreifen und den Inhalt verändern, ohne die Namen der einzelnen Frames zu kennen.

Zwei weitere Beispiele sollen uns die Wirkungsweise des `frames`-Objekts sowie des bereits erklärten location-Objekts erläutern.

Beispielsweise kann ein anderer Autor von seiner Webseite auf unsere Webseite über einen Link verweisen. Verwendet die aufrufende Webseite Frames, kann es vorkommen, dass unsere Seite innerhalb seines Framesets aufgerufen wird. Ein Besucher kann dadurch den Eindruck gewinnen, dass unsere Webseite Bestandteil der fremden Homepage ist. Um dies zu verhindern, verwenden wir das `location`-Objekt, um das oberste Frameset zu überladen.

Seite aus einem Frameset befreien

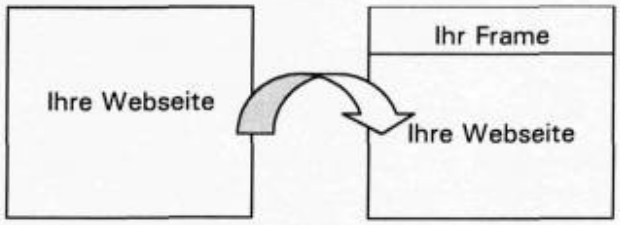
```
<script type="text/javascript">
  if(top.frames.length > 0)
    top.location.href = self.location;
</script>
```



Das Skript kontrolliert, ob sich unsere Webseite in einem Frameset befindet (`frames.length >0`). Ist dies der Fall, wird die Seite in das oberste Frameset geladen (`top.location.href`). Die Adresse Ihrer Webseite erhalten wir durch die Abfrage `self.location`. Ein weiteres Beispiel ist das Laden von Webseiten in einen Frame. Öffnen Besucher über den Link einer Suchmaschine eine Ihrer Webseiten, die eigentlich Teil eines Framesets ist, können wir das vollständige Frameset nachladen lassen.

Seite in ein Frameset laden

```
<script type="text/javascript">
  if(top.frames.length == 0)
    top.location.href = "frame.html";
</script>
```



Dieses Skript überprüft, im Gegensatz zum vorherigen Skript, ob sich die Seite nicht in einem Frame befindet (`frames.length == 0`). Trifft dies zu, wird die Webseite mit der Frame-Definition geladen, in diesem Beispiel die Webseite `frame.html`.

Objekt event

Eine Eigenschaft des `window`-Objekts ist seit der JavaScript-Version 1.2 das `event`-Objekt (event = Ereignis). Mit diesem Objekt können wir auf Ereignisse wie Mausklicks oder Tastatureingaben reagieren und JavaScript-Code ausführen. Beispielsweise können wir bei einem Mausklick ermitteln, ob die linke oder rechte Maustaste gedrückt wurde, oder bei einem Tastendruck die betätigte Taste ermitteln.

Anwendereereignisse können wir überwachen, indem wir in einem HTML-Tag eine Ereignisabfrage hinzufügen (`onClick, onMouseOver, onKeyPress` usw.).

Die Browser ermöglichen die Abfrage der betätigten (Maus-)Tasten. Allerdings stellt der Internet Explorer andere Konstanten zur Verfügung als die beiden Browser Firefox und Opera. So sind wir gezwungen, die entsprechenden Anweisungen für jeden Browser separat anzugeben.

Die folgende Tabelle listet die unterschiedlichen Eigenschaften des `event`-Objekts für die Browser Internet Explorer, Firefox und Opera auf. Gleichbedeutende Eigenschaften sind auf einer Zeile gegenübergestellt. Das heißt, dass beispielsweise die Explorer-Eigenschaften `altKey, ctrlKey, shiftKey` der Eigenschaft `modifiers` in Firefox bzw. Opera entsprechen.

Eigenschaften	Für den Internet Explorer:	Für den Firefox und Opera:
	altKey, ctrlKey, shiftKey	modifiers
	clientX, clientY	screenX, screenY
	keyCode	which, type
	offsetX, offsetY	layerX, layerY
	x,y	pageX, pageY
	button	

Firefox und Opera

Events können von `window-`, `frame-` und `document`-Objekten abgefangen werden. Wir legen die Funktion fest, die bei einem bestimmten Event ausgeführt werden soll, und weisen wir der entsprechenden Objekt-Methode zu.

Beispiel

Es wird der Klick mit der linken Maustaste in einen beliebigen Fensterbereich des Browsers abgefangen.

```
<script  type="text/javascript">
  function klick()
  {alert('Dies war ein einfacher Mausklick!');
  }
  window.onclick=klick;
</script>
```

Damit werden alle Klick-Ereignisse an die Funktion `klick()` übergeben. In der Funktion kann das Ereignis verarbeitet oder an das eigentliche Objekt, das angeklickt wurde, weitergegeben werden. Standardmäßig können wir im Browser Opera nicht die rechte Maustaste abfragen. Dies muss explizit in den Javascripteinstellungen zugelassen werden.

Die weiteren Events für Firefox und Opera lauten:

Event	Wird ausgelöst, wenn ...
click	ein Klick erfolgt
dblclick	ein Doppelklick erfolgt
dragdrop	per Drag & Drop etwas auf das Browserfenster gezogen wurde

Event	Wird ausgelöst, wenn ...
keydown, keypress, keyup	eine Taste betätigt, gedrückt gehalten bzw. wieder losgelassen wird
mousedown, mousemove, mouseup	eine Maustaste gedrückt, die Maus bewegt bzw. die Maustaste wieder losgelassen wird
move	das Browserfenster bewegt wird
resize	die Größe des Browserfensters verändert wird

Wird eine Event-Handler-Funktion aufgerufen, wird ihr immer ein `event`-Objekt übergeben, dessen Eigenschaften das eingetretene Ereignis beschreiben. Das `event`-Objekt kann folgende Eigenschaften besitzen:

Eigenschaft	Bedeutung
`data`	URL eines Drag-&-Drop-Ereignisses
`layerX, layerY`	Koordinaten des Ereignisses innerhalb eines Layers
`modifiers`	Eine Ganzzahl, die die gedrückten Tasten ALT, STRG und ⇧ UMSCHALTEN repräsentiert
`pageX, pageY`	Koordinaten des Ereignisses bezogen auf das Fenster oder den Frame
`screenX, screenY`	Koordinaten des Ereignisses bezogen auf den Bildschirm
`target`	Referenz auf das eigentliche Zielobjekt des Ereignisses
`type`	Zeichenkette mit dem Namen des Ereignisses, z. B. "click"
`which`	Ganzzahl, die bei Maus-Ereignissen den Button, bei Tasten den ASCII-Code der Taste repräsentiert

Internet Explorer

Der Internet Explorer benötigt keine spezielle Methode, um Ereignisse abzufangen. Jedes Ereignis wird entlang der Objekthierarchie abgeglichen. Das heißt beispielsweise, dass ein `click`-Ereignis auf einer Formular-Schaltfläche nach seinem Event-Handler automatisch an den `onClick`-Handler des Formulars und dann an den des Dokuments usw. weitergegeben wird. Praktisch jedes HTML-Element kann somit mit einem Event-Handler ausgestattet werden.

Das `event`-Objekt existiert im Internet Explorer nicht. Die Events sind im Internet Explorer eine Eigenschaft des window-Objekts. Die Eigenschaften von `window.event` werden bei jedem Ereignis überschrieben.

Eigenschaft	Bedeutung
`altKey, ctrlKey, shiftKey`	Wahrheitswert `true`, wenn die Tasten ALT, STRG bzw. ⇧ UMSCHALTEN betätigt wurden
`button`	Ganzzahl, die die gedrückte Maustaste repräsentiert
`cancelBubble`	Wahrheitswert `true`, wenn das Ereignis nicht weitergegeben werden soll
`clientX, clientY`	Koordinaten des Ereignisses bezogen auf den Dokumentbereich des Browsers
`fromElement`	Referenz auf das Objekt, von dem der Mauszeiger kam, als er ein MouseOver- oder MouseOut-Ereignis auslöste
`keyCode`	Enthält den ASCII-Code der betätigten Taste
`offsetX, offsetY`	Koordinaten des Ereignisses bezogen auf das auslösende Objekt oder Element
`returnValue`	Boolescher Rückgabewert des Ereignisses

Eigenschaft	Bedeutung
screenX, screenY	Koordinaten des Ereignisses bezogen auf den Bildschirm
srcElement	Referenz auf das eigentliche Zielobjekt des Ereignisses
toElement	Referenz auf das Objekt, auf das sich der Mauszeiger bewegte, als er ein MouseOver- oder MouseOut-Ereignis auslöste
type	Zeichenkette mit dem Namen des Ereignisses, z. B. "click"
x, y	Koordinaten des Ereignisses bezogen auf das body-Element eines positionierbaren Elements

Um z.B. das automatische Weitergeben des Ereignisses zu stoppen, muss der Event-Handler die Eigenschaft `cancelBubble` auf den Wahrheitswert `true` setzen:

```
<input type="button" onClick="klick(); window.event.cancelBubble=true" >
```

Dadurch hört das Weiterleiten des Ereignisses auf, und der Handler der Schaltfläche war der letzte, der durch das Ereignis aufgerufen wurde.

Maustasten überwachen

Beispiel: objekt_event.html

Zum Kennenlernen der `event`-Eigenschaften entwickeln wir ein Skript, das in den Browsern das Betätigen der Maustasten überwacht.

objekt_event.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>document: event</title>
  <script type="text/javascript">
      var message_lm="Ja, die linke Maustaste funktioniert!";
      var message_rm="Ja, die rechte Maustaste funktioniert!";

      function wache(e) {
        if (window.navigator.appName == "Microsoft Internet Explorer")
        {
          if(window.event.button == 2)
          {
            alert(message_rm);
            return false;
          }
          if(window.event.button == 1)
            alert(message_lm);
```

```
          return true;
        }

        if ((window.navigator.appName == "Netscape") ||
(window.navigator.appName == "Opera"))
        {
          if(e.which == 3)
          {
            alert(message_rm);
            return false;
          }
          if(e.which == 1)
            alert(message_lm);

          return true;
        }
      }

      document.onmousedown = wache;
  </script>
</head>
<body>
  <h3>&Uuml;berwachen der Maustasten</h3>
  <p>Drücken Sie eine Maustaste, um das Ereignis auszul&ouml;sen.

</body>
</html>
```

ASCII-Codes betätigter Tasten ausgeben

Beispiel: objekt_event2.html

In einem mehrzeiligen Eingabefeld werden die Tastatureingaben abgefangen, und der entsprechende ASCII-Code wird in der Statuszeile des Browsers ausgegeben.

objekt_event2.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>ASCII-Code ermitteln</title>
  <script type="text/javascript">
    var isNav, isIE = false;
    if (parseInt(navigator.appVersion) >= 4)
    {
      if ((navigator.appName == "Netscape") || (navigator.appName ==
```

```
"Opera"))
        isNav = true
      else
        isIE = true;
    }

    function ZeigeTaste(evt)
    {
      if (isNav) window.status = evt.which
      else       window.status = window.event.keyCode;
      return false;
    }
  </script>
</head>
<body>
  <h3>event: Tastaturcodes</h3>
  <p>Geben Sie etwas in das Textfeld ein, um den entsprechenden
Tastaturcode in der Statuszeile anzeigen zu lassen.</p>

  <form action="">
    <textarea cols="30" rows="4"
onKeyPress="ZeigeTaste(event)"></textarea>
  </form>
</body>
</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

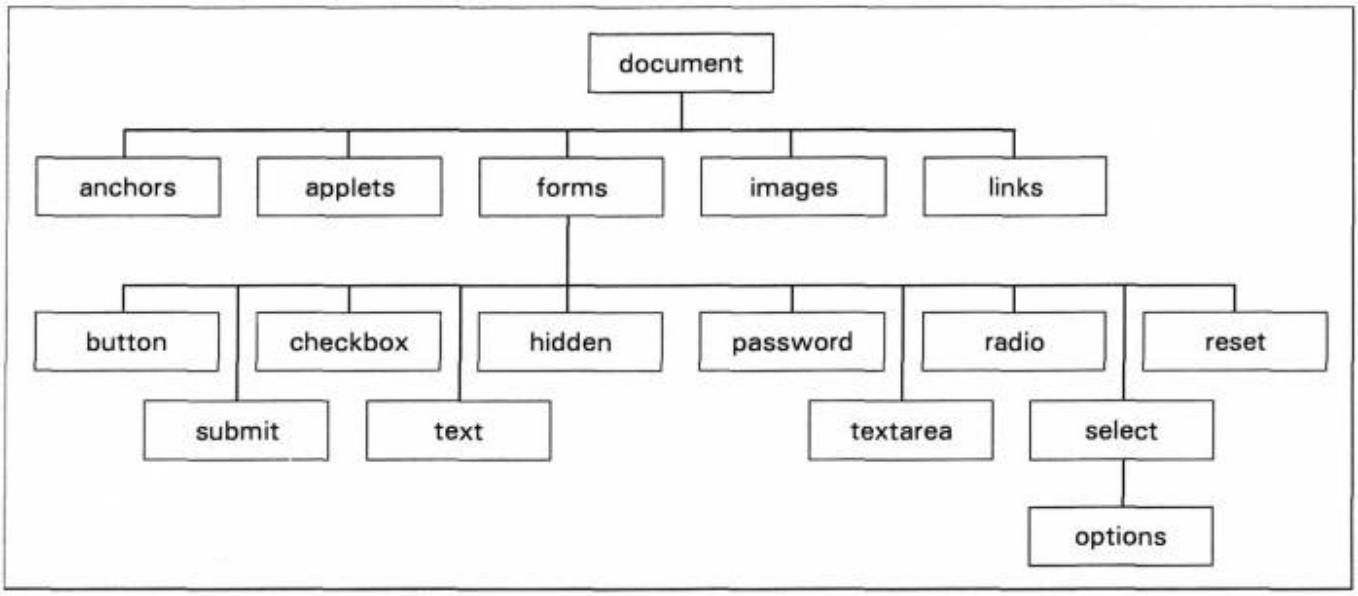
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_09

Last update: **2019/05/16 16:40**

11.3.10) Zugriff auf HTML-Dokumente

Grundlagen zum document-Objekt

Ein `document`-Objekt ist ein Element des `window`-Objekts. Da es den Zugriff auf die Elemente einer Webseite ermöglicht, besitzt es weitere Unterobjekte. So können Sie über das Unterobjekt `forms` auf Formulareigenschaften und -elemente zugreifen, sobald sich ein Formular `<form>…</form>` auf einer Webseite befindet.



Hierarchie des document-Objekts

Der Aufbau der Webseite bestimmt dabei das Vorhandensein von Instanzen eines Unterobjekts. Enthält beispielsweise ein HTML-Dokument kein Formular, besitzt das `document`-Objekt auch keine Objektreferenzen auf Formulare.

Unterobjekt anchors

Mit diesem Objekt haben wir Zugriff auf die Verweisanker einer Webseite. Über die folgende HTML-Anweisung wird z.B. ein Verweisanker definiert:

```
<a name="absatz2">&Überschrift zum 2.Absatz</a>
```

Eigenschaften

Eigenschaften	Beschreibung
`length`	Über die Eigenschaft `document.anchors.length` erhalten Sie die Anzahl der Verweisanker, die sich innerhalb einer Webseite befinden.
`name`	Erlaubt den Zugriff auf den Namen des Ankers
`text`	Erlaubt den Zugriff auf den Text des Ankers

Unterobjekt applets

Dieses Objekt erlaubt den Zugriff auf Java-Programme, die speziell für den Gebrauch im Browser erstellt wurden, sogenannte Java-Applets.

Eigenschaften und Methoden

Eigenschaften	Beschreibung
`code()`	Diese Methode liefert den Quelltext eines Applets zurück, sofern dieser vorliegt.
`length`	Über die Eigenschaft `document.applets.length` erhalten Sie die Anzahl der Applets, die innerhalb einer Webseite eingebunden sind.

Unterobjekt forms

Ein Formular wird z.B. auf einer Webseite verwendet, um Informationen zu erfassen und an einen Empfänger, z.B. ein Skript auf dem Webserver, zu übertragen. Wie die Daten eingegeben werden, ist von den Eingabeelementen im Dokument abhängig. In Zusammenhang mit den Formularen in einem HTML-Dokument besitzt das f orms-Objekt weitere Unterobjekte, die den Unterelementen des `<form>`-Tags in HTML entsprechen.

Formular

Ein Formular wird zur Eingabe von Daten verwendet, die nach dem Absenden verarbeitet werden sollen. Die Auswertung kann dabei auf unterschiedliche Weise erfolgen. Die einfachste Art ist es, die Daten per E-Mail an eine bestimmte Person zu senden. Oder sie werden an ein Programm, meist ein in den Programmiersprachen PHP oder Perl geschriebenes Skript, auf dem Server geschickt und dort ausgewertet. In diesem Fall müssen die Adresse des Programms (action), der zu übertragende Datentyp (encoding) und das zu verwendende Übertragungsverfahren (method) bekannt sein.

Den Aufbau des Formulars finden wir in den Eigenschaften und Methoden des `forms`-Objekts wieder.

Eigenschaft	Bedeutung
action	Hiermit erhalten Sie den Formularwert action, z. B. die URL des Skripts, das die Daten verarbeiten soll.
encoding	Die Kodierung der Daten ermitteln oder setzen Sie über diese Eigenschaft (z. B. "text/plain").
length	Um die Anzahl der im Formular befindlichen Elemente zu ermitteln, fragen Sie die Eigenschaft length ab.
method	Damit lesen Sie den Wert des method-Attributs aus.
name	Den Namen des Formulars erhalten Sie über die Abfrage der Eigenschaft name.
target	Das Zielfenster, in dem beispielsweise die Rückmeldung des Serverprogramms erscheinen soll, erhalten Sie mit target, das dem HTML-Attribut <form target=""> entspricht.
Methoden	**Bedeutung**
reset()	Damit werden alle Einträge im Formular auf den Anfangswert zurückgesetzt.
submit()	Hiermit werden die Daten des Formulars an das im Attribut action hinterlegte Ziel gesendet.

Beispiel: formular_auslesen.html

Es steht beispielsweise der folgende HTML-Quelltext als erstes Formular in einem HTML-Dokument.

```
<form name="Formular"  action="test.php"  method="post"
enctype="text/plain" target="_top"> </form>
```

Der JavaScript-Code zum Auslesen dieser Formulardaten lautet:

formular_auslesen.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Unterobjekt forms</title>
  <script type="text/javascript">
      function formauslesen()
      {
        var text = "";

        text ='name: ' +  document.forms[0].name + '\n'
            + 'action: ' + document.forms[0].action + '\n'
            + 'method: ' + document.forms[0].method + '\n'
            + 'encoding: ' + document.forms[0].encoding + '\n'
            + 'target: '+ document.forms[0].target + '\n'
            + 'length: '+ document.forms[0].length;

        alert(text);
      }
  </script>
```

```
</head>
<body onLoad="formauslesen();">
  <h3>Auslesen der Formulardaten</h3>
  <form name="Formular" action="test.php" method="post"
enctype="text/plain" target="_top">
  </form>

</body>
</html>
```

Wir können nicht nur die Eigenschaften des gesamten Formulars ermitteln. Die Funktionalität und Interaktivität eines Formulars verbirgt sich in den einzelnen Elementen, die in den vorhergehenden Abschnitten beschrieben wurden und im Array `document.FormName.elements[].Eigenschaft` des Formulars gespeichert sind. Mit der Angabe eines Index können wir ein bestimmtes Element des Formulars ansprechen, z.B. das vierte Element (mit dem Index 3) über `document.FormName.elements[3].Eigenschaft`. Kennen wir den Namen des Elements, können wir es direkt ansprechen und dessen Eigenschaften abfragen bzw. ändern: z.B. `document.FormName.ElementName.Eigenschaft= Wert`.

Eingabefelder

In HTML-Dokumenten kann in diesen Feldern Text ein- oder ausgegeben werden. Zur Verfügung stehen hierbei drei verschiedene Arten von Eingabefeldern: `<input type=„text“>` für einzeilige Felder, `<input type=„textarea“>` für mehrzeilige Felder und `<input type=„password“>` für nicht lesbare Passworteingaben.

Eigenschaft	Bedeutung
`defaultValue`	Wird im HTML-Quelltext für das Eingabefeld im Attribut `value` ein Wert vorgegeben, ist dieser über die Eigenschaft `defaultValue` erreichbar.
`form`	Damit haben Sie Zugriff auf den Namen des Formulars, zu dem das Element gehört.
`name`	Diese Eigenschaft erlaubt das Auslesen eines Feldnamens.
`type`	Es wird der Typ des Objekts zurückgeliefert (`text`, `textarea`, `password`).
`value`	Hiermit greifen Sie auf den Inhalt des Eingabefeldes zu.

Um die einzelnen Eingabefelder anzusprechen, existieren in JavaScript die jeweils entsprechenden Objekte `text, textarea` und `password`. Alle drei Objekte besitzen die folgenden Eigenschaften und Methoden:

Methoden	Bedeutung
`blur()`	Hiermit wird das Feld verlassen.
`focus()`	Diese Methode setzt den Mauscursor in das entsprechende Eingabefeld.
`select()`	Mit `select()` können Sie den Inhalt des Feldes auswählen.

Versteckte Felder

Innerhalb eines Formulars können wir mit der Angabe `<input type=„hidden“ …>` zusätzliche Informationen verschicken. Diese Felder werden nicht angezeigt. Mit JavaScript können wir folgende Eigenschaften des `hidden`-Elements abfragen.

Eigenschaft	Bedeutung
`form`	Damit haben Sie Zugriff auf den Namen des Formulars.
`name`	Den Namen des Objekts erhalten Sie mit der Abfrage von `name`.
`type`	Über diese Eigenschaft ermitteln Sie den Typ des Elements (`hidden`).
`value`	Hiermit lesen Sie den Wert des versteckten Feldes aus.

Kontroll- und Optionsfelder

Eine Auswahl können wir in einem HTML-Dokument durch Kontroll-(checkbox) oder Optionsfelder (radio buttons) ermöglichen. Der Unterschied in der Verwendung der beiden Felder besteht darin, dass über Kontrollfelder genau eine Einstellung vorgenomnn werden soll (ja/nein). Optionsfelder werden innerhalb einer Gruppe zusammen-gefasst, und es kann nur eine Option der Gruppe gewählt werden (z.B. eine Auswahl der Pizza: entweder groß, klein oder mittelgroß).

Eigenschaft	Bedeutung
`checked`	Mit der Eigenschaft `checked` erhalten Sie über die Werte `true` und `false` die Information, ob ein Kontroll- oder Optionsfeld markiert ist (`true` = ja, `false` = nein).
`defaultChecked`	Diese Eigenschaft liefert `true`, wenn das Element beim Laden der Webseite bereits aktiviert ist, sonst `false`.
`form`	Sie haben Zugriff auf den Namen des Formulars.
`length`	Damit können Sie die Anzahl der Optionsfelder innerhalb einer Optionsgruppe ermitteln.
`name`	Den Namen des Kontroll- bzw. Optionsfeldes erhalten Sie mit der Abfrage von `name`, der dem gleichnamigen HTML-Attribut entspricht.
`type`	Diese Eigenschaft liefert den Typ des Objekts zurück (`checkbox`, `radio`).
`value`	Hiermit lesen Sie den Übergabewert aus.
Methoden	**Bedeutung**
`blur()`	Hiermit wird das Feld verlassen.
`click()`	Die Funktionalität der Kontroll- und Optionsfelder können Sie mit der Methode `click()` erweitern, um z. B. bei einer Auswahl entsprechend reagieren zu können.
`focus()`	Diese Methode setzt den Mauscursor in das entsprechende Feld.

Auswahllisten

Auswahllisten ermöglichen eine Selektion von einer oder mehreren (multiple) Optionen. Diese Listen werden in HTML mit dem HTML-Tag <select> erzeugt und in JavaScript durch das gleichnamige Objekt select repräsentiert.

Eigenschaft	Bedeutung
`form`	Damit haben Sie Zugriff auf den Namen des Formulars.
`length`	Die Anzahl der in der Auswahlliste befindlichen Optionen erhalten Sie über die Abfrage der Eigenschaft `length`.
`name`	Enthält den Namen der Auswahlliste, der dem gleichnamigen HTML-Attribut entspricht
`options[]`	Hiermit können Sie auf die Einträge zugreifen.
`selectedIndex`	Diese Eigenschaft gibt den Index der ausgewählten Option zurück, bei `type="select-multiple"` wird der erste ausgewählte Eintrag zurückgegeben.
Eigenschaft	**Bedeutung**
`type`	Ob eine Mehrfachauswahl in der Auswahlliste möglich ist, ermitteln Sie über die Eigenschaft `type`. Die Rückgabe lautet entweder `"select-one"` für die einfache Auswahl oder `"select-multiple"` für die Mehrfachauswahl.
`value`	Damit lesen Sie den Übergabewert aus.
Methoden	**Bedeutung**
`blur()`	Hiermit wird die Auswahlliste verlassen.
`focus()`	Diese Methode setzt den Mauscursor in die Auswahlliste.

Das ''options''-Array

Die Einträge der Auswahlliste sind über das Feld options erreichbar. Der Index des Eintrags wird in eckigen Klammern angegeben. Die Listenelemente besitzen folgende Eigenschaften:

Eigenschaft	Bedeutung
`defaultSelected`	Den Wahrheitswert, ob eine Auswahl voreingestellt ist, erhalten Sie über die Eigenschaft `defaultSelected`.
`index`	Die Nummer des Eintrags wird über den Wert von `index` zurückgegeben.
`length`	Die Anzahl der Listenelemente lesen Sie hiermit aus.
`selected`	Die vom Benutzer vorgenommene Auswahl erhalten Sie über `selected`.
`text`	Den Text, der als Auswahl angezeigt wird, können Sie mithilfe von `text` ermitteln.
`value`	Wurde im HTML-Quelltext zu einer Auswahl noch ein zusätzlicher Wert angegeben, können Sie diesen über die Eigenschaft `value` ermitteln.

Beispiel: auswahlliste.html

Die Funktion durchlauf zeigt alle Eigenschaften eines `option`-Elements als Textausgabe in einem mehrzeiligen Textfeld an. Dazu werden über eine `for`-Schleife die Eigenschaften jedes Eintrags des Formularelements mit dem Namen `auswahl` ausgelesen und eine Zeichenkette erzeugt. Diese wird der Eigenschaft `value` des `textarea`-Elements zugewiesen und im Textfeld ausgegeben. Die Zeilenumbrüche werden durch die Escapesequenzen `\n` erzeugt.

auswahlliste.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Formulareigenschaften : Auswahlliste</title>
  <script type="text/javascript">
      function durchlauf()
      {
        var text = "";

        for (i = 0; i < document.formular.auswahl.length; i++)
        {
          text = text +
              'index: ' +  document.formular.auswahl.options[i].index +
'\n'
              + 'defaultSelected: ' +
document.formular.auswahl.options[i].defaultSelected + '\n'
              + 'selected: ' +
document.formular.auswahl.options[i].selected + '\n'
              + 'text: ' + document.formular.auswahl.options[i].text +
'\n'
              + 'value: '+ document.formular.auswahl.options[i].value +
'\n' +
              '==========================\n';
        }

        document.formular.ausgabe.value = text;
      }
  </script>
</head>
<body onLoad="durchlauf();">
  <h3>Anzeige der Eigenschaften der ausgewählten Elemente</h3>
  <form action="" name="formular">
    <select name="auswahl" size="4" multiple onChange="durchlauf();">

      <option value="Tier1">Der laute Hahn</option>
      <option value="Tier2" selected>Die sanftmütige Katze</option>
      <option value="Tier3">Der treue Hund</option>
      <option value="Tier4">Der störrische Esel</option>
    </select>
    <hr><br>
    <textarea cols="30" rows="25" name="ausgabe"></textarea>
```

```
  </form>
</body>
</html>
```

Die Auswahlliste `auswahl` befindet sich in einem HTML-Formular mit dem Namen `formular`. Die Funktion `durchlauf()` wird nach dem Laden der Webseite aufgerufen (`onLoad=„durchlauf()“`). Nach jeder geänderten Auswahl wird die Anzeige ebenfalls aktualisiert(`onChange`).

Schaltflächen

Eines der wichtigsten Elemente eines Formulars sind die Schaltflächen (buttons). Sie dienen zum Starten von Operationen, indem die entsprechende Schaltfläche betätigt wird.

Eigenschaften	`form, name, type, value`
Methoden	`click()`

Die Eigenschaften erlauben das Auslesen des Formularnamens (`form`), des Namens der Schaltfläche (`name`), des Typs (`type`) sowie der Beschriftung der Schaltfläche (`value`).

Schaltflächen können mit der Ereignisabfrage `onClick` auf das Anwählen mit der Maus oder auf das Drücken der Eingabetaste reagieren. Ansonsten bleiben diese Schaltflächen wirkungslos. Eine Ausnahme bilden die zwei Schaltflächen `Submit` und `Reset` zum Versenden und Zurücksetzen der eingegebenen Daten. Diese beiden standardmäßigen Schaltflächen dienen dazu, die eingegebenen Daten eines Formulars ohne Zuhilfenahme von JavaScript zu versenden oder zu löschen. Durch den Einbau einer Ereignisbehandlung für das Ereignis onSubmit wäre es beispielsweise möglich, vor dem Versenden der Daten diese auf Vollständigkeit zu prüfen bzw. beim Löschen eine Sicherheitsabfrage einzufügen.

Eingaben prüfen

Eingabefelder überprüfen

Die am meisten genutzte Funktion des f orms-Objekts ist die Überprüfung, ob alle Felder eines Formulars ausgefüllt wurden. Zu Beginn erstellen Sie ein Formular mit einem Eingabefeld und einer Schaltfläche submit zum Abschicken der Daten. Vor dem Abschicken der Formulardaten wird das Ereignis onSubmit ausgelöst und dadurch die Funktion testen ausgeführt.

Beispiel: form_tester.html

Zur Überprüfung der Eingabe wird das Ereignis onSubmit verwendet. Es ruft beim Betätigen der

Schaltfläche absenden die Funktion `testen()` auf. Um das Versenden der Daten bei einer fehlerhaften Eingabe zu verhindern, gibt `return` den Rückgabewert der Funktion zurück. Ist die Eingabe korrekt, liefert die Funktion den Wert `true` und die Aktion wird ausgeführt. Liefert die Funktion `testen()` jedoch den Wert `false`, werden die Formulardaten nicht abgesendet.

form_tester.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Eingabe testen</title>
  <script type="text/javascript">
      function testen()
      {
        if (document.Eingabe.Feld.value == "")
        {
          alert("Sie müssen etwas eingeben!")
          document.Eingabe.Feld.focus();
          return false;
        }
        else
        {
          alert("Vielen Dank!");
          return true;
        }
      }
  </script>
</head>
<body>
  <div align="center">
  <h3>Überprüfen der Eingabe</h3>
    <form action="" name="Eingabe" onSubmit="return testen();">

      <input type="Text" name="Feld">
      <input type="submit" name="absenden" value="Testen">
    </form>
  </div>
</body>
</html>
```

Das Beispiel können wir erweitern, indem wir z.B. den Mauscursor automatisch in das fehlerhafte Eingabefeld setzen. Die Methode zum Setzen des Cursors ist `focus()`. Durch die Angabe des entsprechenden Formularfeldes geben wir an, in welches Feld der Cursor gesetzt werden soll. In dem vorigen Beispiel könnten wir nach der Fehlermeldung die Methode `document.Eingabe.Feld.focus();` angeben. Bei umfangreichen Formularen mit vielen Eingabefeldern erleichtern wir somit dem Anwender die Suche nach dem fehlerhaften Eingabefeld.

Beispiel: auf_zahlen_testen.html

Mithilfe des Objekts `string` haben wir die Möglichkeit, die Daten eines Eingabefeldes auf bestimmte Zeichen zu kontrollieren. Das folgende Beispiel testet den Wert eines Eingabefeldes auf numerische Daten. Ist die Eingabe eine Zahl, wird der Wahrheitswert `true` als Resultat zurückgegeben. Besteht die Eingabe nicht aus einer Zahl, wird `false` zurückgeliefert. Wir können den Test auch mithilfe von regulären Ausdrücken realisieren.

auf_zahlen_testen.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Eingabe testen</title>
  <script type="text/javascript">
    function NumTest(eingabe)
    {
      var daten ="0123456789";
      for(i = 0; i < eingabe.length; i++) {
        if(daten.indexOf(eingabe.charAt(i)) < 0 )
        {
          alert(eingabe + " ist keine gültige nummerische Eingabe.");
          return false;
        }
      }
      return true;
    }
  </script>
</head>
<body>
  <div align="center">
  <h3>Testen auf nummerische Daten</h3>
  <p>Geben Sie bitte Zahlen in das Eingabefeld ein:</p>

    <form action="" name="formular" onSubmit="return
NumTest(eingabe.value);">
      <input type="text" name="eingabe">
      <input type="submit" name="absenden" value="Testen">
    </form>
  </div>
</body>
</html>
```

Abfrage eines Optionsfeldes

Der Test, ob mindestens ein Optionsfeld gewählt wurde, ist etwas umfangreicher.

Beispiel: form_tester_options.html

Die grundlegende Vorgehensweise bei dieser Abfrage ist, dass zuerst die Anzahl der Optionsfelder ermittelt werden muss. Danach prüfen wir über eine Zählschleife die Eigenschaft checked jedes Optionsfeldes. Wurde kein Element ausgewählt, wird eine Fehlermeldung angezeigt.

form_tester_options.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Test, ob ein Optionsfeld ausgewählt wurde</title>
  <script type="text/javascript">
      function optionstest()
      {
        var j = false;
        for(i = 0; i < document.RadioForm.Abfrage.length; i++)
          if(document.RadioForm.Abfrage[i].checked)
          {
            j = true;
            break;
          }

        if(!j)
        {
          alert("Sie müssen mindestens eine Option auswählen.")
          return false;
        }
        return true;
      }
  </script>
</head>
<body>
  <div align="center">
  <h3>Test, ob ein Optionsfeld ausgewählt wurde</h3>
  <form action="" name="RadioForm" onSubmit="return optionstest();">

    <input type="radio" value="1" name="Abfrage">Option 1<br>
    <input type="radio" value="2" name="Abfrage">Option 2<br>
    <input type="radio" value="3" name="Abfrage">Option 3<br>
    <input type="submit" name="x1" value="Testen">
  </form>
  </div>
</body>
</html>
```

Sicherheitsabfrage beim Zurücksetzen

Ein weiteres Beispiel zur Arbeit mit Formularen ist die Sicherheitsabfrage beim Zurücksetzen der eingegebenen Daten. Normalerweise werden alle Eingaben sofort gelöscht, sobald der Anwender die reset-Schaltfläche eines Formulars betätigt. Dies kann jedoch in einigen Fällen ungewollt sein. Die folgende Funktion ermittelt über ein Bestätigungsfenster, ob die Daten wirklich gelöscht werden sollen.

Beispiel: reset_abfrage.html

Die Funktion `warnung()` fängt das Zurücksetzen des Formulars ab und blendet ein Fenster zum Bestätigen des Vorgangs ein. Bestätigt der Anwender das Löschen der Daten, liefert die Funktion den Wert `true` zurück und das Zurücksetzen der Daten wird durchgeführt. Um den Löschvorgang abzubrechen, wird das Bestätigungsfenster über die Schaltfläche Abbrechen geschlossen und der Wert false wird zurückgeliefert. Die Daten bleiben erhalten. Diese Abfragefunktion beim Betätigen der `reset`-Schaltfläche mit der Ereignisabfrage `onReset()` im `<form>`-Tag aufgerufen.

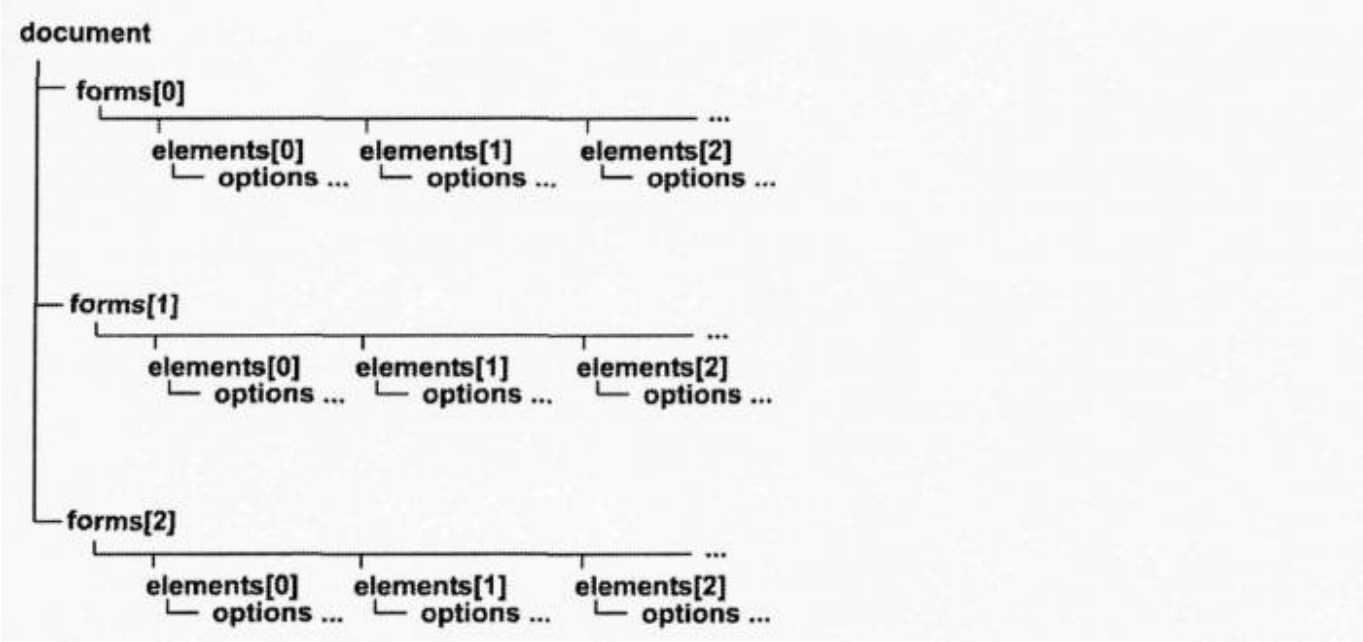
reset_abfrage.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Sicherheitsabfrage beim Zurücksetzen</title>
  <script type="text/javascript">
    function warnung()
    {
      return confirm("Wollen Sie wirklich alle Eingaben
zurücksetzen?");
    }
  </script>
</head>
<body>
  <div align="center">
  <h3>Sicherheitsabfrage beim Zurücksetzen</h3>
  <p>Geben Sie etwas in das Eingabefeld ein <br>und klicken Sie auf
"Zurücksetzen".</p>

    <form action="" name="formular" onReset="return warnung();">
      <textarea cols="20" rows="10" name="eingabe"></textarea>
      <br>
      <input type="submit" value="Senden">
      <input type="reset" value="Zurücksetzen">
    </form>
  </div>
</body>
</html>
```

Verschiedene Formulare

Auf einer Webseite können sich mehrere Formulare befinden. Damit jedes Element eines Formulars eindeutig angesprochen werden kann, werden die verschiedenen Formulare separat betrachtet.



Schematische Darstellung mehrerer Formulare und ihrer Elemente als Arrays

Beispiel: mehrere_formulare.html

Eine Webseite enthält zwei Formulare, die zur Eingabe von Personenangaben und Suchbegriffen dienen. Dem ersten Formular weisen wir über `onSubmit=„readForm1()“` die Ausführung der Funktion `readForm1()` zu. Nach dem Abschicken des zweiten Formulars soll die Funktion `readForm2()` ausgeführt werden.

mehrere_formulare.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Mehrere Formulare auslesen</title>
  <script type="text/javascript">
      function readForm1()
      {
        alert('Eingabefeld "' + document.Formular1.IhrName.name + '"
mit dem Wert "' + document.Formular1.IhrName.value + '"');
        alert('Optionsfeld "' + document.Formular1.Geschlecht[0].name +
'" mit dem Wert "' + document.Formular1.Geschlecht[0].value + '";
ausgewählt=' + document.Formular1.Geschlecht[0].checked);
        alert('Optionsfeld "' + document.Formular1.Geschlecht[1].name +
'" mit dem Wert "' + document.Formular1.Geschlecht[1].value + '";
ausgewählt=' + document.Formular1.Geschlecht[1].checked);
```

```
        return;
      }

      function readForm2()
      {
        alert('Eingabefeld "' + document.forms[1].elements[0].name + '"
mit Wert "' + document.forms[1].elements[0].value + '"');
        alert('Schaltfläche "' + document.forms[1].elements[1].name +
'" mit Wert "' + document.forms[1].elements[1].value + '"');
        return;
      }
  </script>
</head>
<body>
  <h3>Mehrere Formulare auslesen</h3>
  <h4>1.Formular</h4>

  <form action="" method="get" name="Formular1"
onSubmit="readForm1();">
    <input type="text" name="IhrName" value="Ihr Name">
    <p><input type="radio" name="Geschlecht" value="männlich">männlich
    <br><input type="radio" name="Geschlecht" value="weiblich">weiblich
    <p><input type="submit" name="Suchen" value="Absenden">
  </form>

  <hr>

  <h4>2.Formular</h4>
  <form action="" method="get" name="Formular2"
onSubmit="readForm2();">

    <input type="text" name="Suchfeld" value="Suchbegriff">
    <p><input type="submit" name="Suchanfrage" value="Suchen">
  </form>
</body>
</html>
```

Die einzelnen Elemente sind mit ihrer HTML-Definition als Objekte im Dokumentenmodell verankert. Ein Element kann jetzt auf verschiedene Weise referenziert werden. Im Folgenden werden die verschiedenen Möglichkeiten für das Auslesen des ersten Elements im ersten Formular aufgelistet. Alle Varianten greifen auf den Wert des Eingabefelds `IhrName` zu, das ein Element des ersten Formulars mit dem Namen `Formular1` ist.

Verschiedene Möglichkeiten, ein Formularelement anzusprechen:

```
document.forms[0].elements[0].value
document.forms[0].elements["IhrName"].value
document.forms[0].IhrName.value
document.forms["Formular1"].elements[0].value
document.forms["Formular1"].elements["IhrName"].value
```

```
document.forms["Formular1"].IhrName.value
document.Formular1.IhrName.value
```

Die Variante `document.Formular1.IhrName.value` mit der direkten Angabe der Objektnamen ist der am häufigsten verwendete Zugriff auf ein Formularelement.

Optionsfelder (radio buttons) nehmen eine Sonderstellung ein. Da mehrere Optionsfelder in einer Gruppe zusammengefasst werden können und somit den gleichen Namen haben, können sie nicht direkt über diesen Namen angesprochen werden. Sie können deshalb nur über den Index angesprochen werden, z.B. `document.Formular.elements[0]`, oder über das Namenfeld `document.Formular.Optionsname[0]`.

Unterobjekt images

Um die Eigenschaften von Grafiken auslesen und ändern zu können, nutzen Sie das Objekt `images`.

Eigenschaft	Bedeutung
`border`	Schaltet die Anzeige eines Rahmens an bzw. aus
`complete`	Hiermit können Sie kontrollieren, ob eine Grafik erfolgreich geladen wurde. Firefox wertet aus, ob generell das Laden beendet wurde. Mit Firefox kann daher nicht auf fehlerhafte Grafiken kontrolliert werden.
`height, width`	Über diese Eigenschaften erhalten Sie Zugriff auf die Höhe und Breite der Grafik.
`hspace, vspace`	Den horizontalen (`hspace`) und vertikalen (`vspace`) Abstand einer Grafik zum Text erhalten Sie über diese Eigenschaft.
`length`	Damit ermitteln Sie die Anzahl der im Dokument geladenen Grafiken.
`lowsrc, src`	Die Quellen des Vorschaubildes (`lowsrc`) und der eigentlichen Grafik (`src`) können Sie über diese Eigenschaften auslesen.
`name`	Die Bezeichnung einer Grafik erhalten Sie, wie bei den anderen Objekten auch, über die Eigenschaft `name`.

Der Zugriff auf die einzelnen Grafiken erfolgt über einen Index, den Sie hinter dem Objektnamen `images` in eckigen Klammern angeben, z.B. `document.images[0]`. Sie können eine Grafik auch direkt mit dem Namen ansprechen, den Sie ihr vorher mit dem Attribut name innerhalb des <img>-Tags gegeben haben, z.B. `document.BildName`.

Für Grafiken, die Sie nachträglich mit JavaScript anzeigen möchten, müssen Sie jedoch eigene Grafikobjekte erzeugen. Das ist beispielsweise notwendig, wenn Sie Grafiken durch andere Grafiken per JavaScript ersetzen wollen.

Objekterzeugung

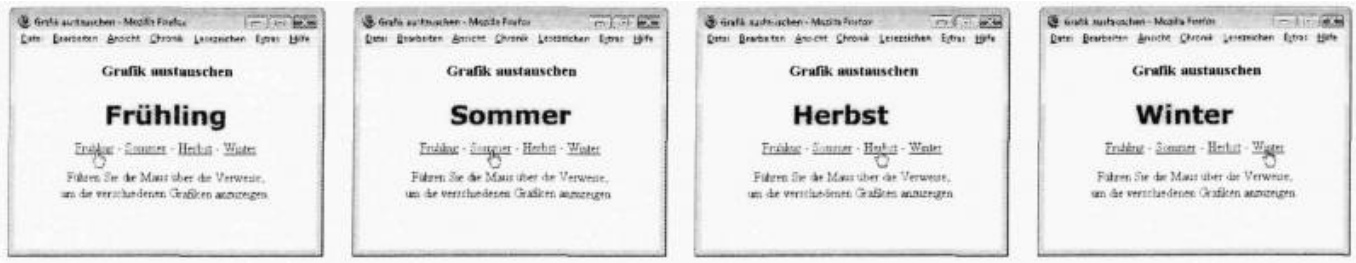
```
Bild1 = new Image();
Bild1.src = 'bild.jpg';
document.images[0].src = Bild1.src;
```

Zum Erzeugen eines Objekts vergeben Sie einen Variablennamen, den Sie mit `new Image()` initialisieren. Dem erzeugten Objekt können Sie jetzt über die Eigenschaft `src` die entsprechende

Grafikdatei zuweisen.

Ein sehr häufig genutzter Effekt im Internet ist der MouseOver-Effekt. Er wird deshalb sogenannt, weil sich eine Grafik ändert, sobald sich die Maus über einem Link oder einer Grafik befindet (`onMouseOver`).

Grafiken auswechseln



Anzeige verschiedener Grafiken beim Überfahren unterschiedlicher Verweise mit der Maus

Die angezeigte Grafik erhält die Bezeichnung bild, mit der wir über JavaScript angesprochen werden kann. Sobald sich die Maus über einem Link befindet, wird die Grafik durch das entsprechende Bild ersetzt. Wird der Link verlassen, soll eine leere Grafik angezeigt werden.

Beispiel: images_bildtausch.html

Zuerst erstellen Sie den HTML-Code mit einer unsichtbaren Grafik und den Hyperlinks. Die nicht sichtbare Grafik `images/blank.gif` dient als Platzhalter für die dynamisch erscheinende Grafik. Beim Überfahren der Hyperlinks mit der Maus wird statt der Platzhalter-Grafik eine andere Grafik angezeigt.

images_bildtausch.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Grafik austauschen</title>
  <script type="text/javascript">
      function zurueck()
      {
        document.bild.src = "images/blank.gif";
      }
  </script>
</head>
<body>
  <div align="center">
  <h3>Grafik austauschen</h3>
  <img src="images/blank.gif" name="bild" alt="" title=""><br>

  <a href="#" onMouseOver="document.bild.src='images/fruehling.gif';"
onMouseOut="zurueck();">Frühling</a> -
```

```
  <a href="#" onMouseOver="document.bild.src='images/sommer.gif';"
onMouseOut="zurueck();">Sommer</a> -
  <a href="#" onMouseOver="document.bild.src='images/herbst.gif';"
onMouseOut="zurueck();">Herbst</a> -
  <a href="#" onMouseOver="document.bild.src='images/winter.gif';"
onMouseOut="zurueck();">Winter</a>
  <p>Führen Sie die Maus über die Verweise, <br>um die verschiedenen
Grafiken anzuzeigen.</p>
  </div>

</body>
</html>
```

Beispiel: images_diashow.html

Mit diesem Beispiel realisieren wir eine Diashow, wobei die Reihenfolge der Anzeige mithilfe einer Navigation steuerbar ist.

Die Steuerung der Grafikanzeige realisieren wir hier über Formular-Schaltflächen. Wir können zu diesem Zweck auch Hyperlinks einsetzen. Beide unterstützen die Ereignisabfrage `onClick`, mit der der Grafikwechsel ausgelöst wird. Übergeben wird die Nummer der Schaltfläche, damit in der Funktion eine Unterscheidung für die anzuzeigende Grafik vorgenommen werden kann.

An das Ende der Webseite `images_diashow.html` fügen wir den folgenden JavaScript-Code ein:

```
<script type="text/JavaScript">
  wechsel(0);
</script>
```

Damit rufen wir die Funktion `wechsel()` auf, zeigen die erste Grafik an und füllen nach dem Laden der Webseite das Eingabefeld mit der Erläuterung zum ersten Bild.

images_diashow.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Diashow</title>
  <script type="text/javascript">
      var bilder = new Array("blau.jpg", "land.jpg", "winter.jpg",
"wueste.jpg");
      var bildlegende = new Array("Das blaue Meer.", "Alle Vöglein sind
schon da...", "Sonnenuntergang im Winter", "Fata Morgana");
      var count = 0;

      function wechsel(stelle)
      {
```

```
        if (stelle == 0)
          count = 0;
        if ((stelle == 1) && (count > 0))
          count--;
        if ((stelle == 2) && (count < bilder.length - 1))
          count++;
        if (stelle == 3)
          count = bilder.length - 1;

        document.bild.src = "images/" + bilder[count];
        document.legende.ausgabe.value = bildlegende[count];
      }
  </script>
</head>
<body>
  <div align="center">
    <h2>Dia-Show</h2>
    <form action="" name="legende">

      <input type="button" value="<<" onClick="wechsel(0);">
      <input type="button" value=" < " onClick="wechsel(1);">
      <input type="button" value=" > " onClick="wechsel(2);">
      <input type="button" value=">>" onClick="wechsel(3);"><br>
      <input type="Text" name="ausgabe" size="30" readonly>
      </form>
    <p><img src="images/blau.jpg" border="0" width="320" height="240"
name="bild" alt="Bildanzeige"></p>
    <script type="text/javascript">
        wechsel(0);
    </script>

  </div>
</body>
</html>
```

Unterobjekt links

Über das Objekt `links` haben Sie per JavaScript Zugriff auf die Verweise in einem HTML-Dokument. Der Zugriff erfolgt über deren Index. Über `links[i]` haben wir Zugriff auf die URL, auf die der Link verweist.

Eigenschaften	Beschreibung
length	Diese Eigenschaft liefert die Anzahl der Verweise, die sich innerhalb einer Webseite befinden.
name	Erlaubt den Zugriff auf den Namen des Verweises
target	Über diese Eigenschaft greifen Sie auf das Ziel des Verweises zu.
text	Erlaubt den Zugriff auf den Text des Verweises. Diese Eigenschaft wird vom Internet Explorer nicht interpretiert.

Beispiel: links.html

Zu Beginn werden im <body>-Tag zwei Links definiert. Im folgenden Skript werden die Informationen für jeden Link über Meldungsfenster ausgegeben.

links.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Objekt: links</title>
</head>
<body>
  <h3>Auslesen der hinterlegten Hyperlinks</h3>
  <p>Hier sind die Links: <a href="http://www.google.at/"
name="link1">Austrian Google</a> |
  <a href="form.html" target="_blank" name="link2">Zum Formular</a>

  <script type="text/javascript">
    for(i = 0; i < document.links.length; i++)
    {
      alert((i + 1) + ". Link:" + "\n" +
            "Name: " + document.links[i].name + "\n" +
            "Ziel: " + document.links[i].target + "\n" +
            "Verweis: " + document.links[i] + "\n" +
            "Text: " + document.links[i].text + "\n");
    }
  </script>
</body>
</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_10

Last update: **2019/05/16 16:42**

11.3.11) Event-Handler

Grundlagen zu Events

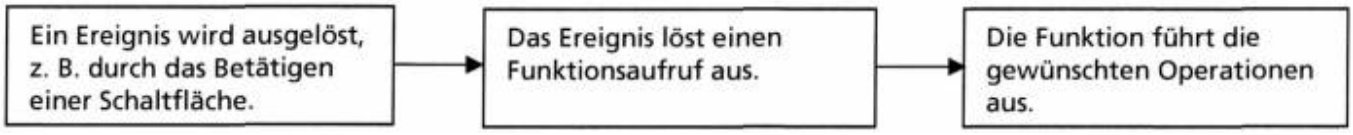
Eine Webseite interaktiv zu gestalten, bedeutet, auf die Aktivitäten des Benutzers zu reagieren. Ein typisches Beispiel ist das Auswerten eines HTML-Formulars. JavaScript ermöglicht dabei das Abfangen von fehlerhaften Eingaben, bevor der Formularinhalt an den Webserver geschickt wird. Dies hilft unnötigen Datentransfer im Internet zu vermeiden, die Server nicht durch Fehleingaben zu belasten und dem Benutzer Wartezeiten zu ersparen. Andere Aktivitäten sind z. B. vom Benutzer ausgeführte Mausbewegungen und -klicks. Diese können verschiedene Operationen auslösen, z. B. das Starten eines Countdowns, das Öffnen weiterer Fenster oder die Anzeige versteckter Grafiken.

Im Document-Object-Modell (DOM) besitzen viele HTML-Objekte neben Eigenschaften und Methoden auch Ereignisse (engl.: events), auf die der Browser mit der Ausführung von JavaScript-Anweisungen reagieren kann. Dazu wird ein sogenannter Event-Handler als Attribut im zugehörigen HTML-Tag definiert. Meist wird eine JavaScript-Funktion, die im Kopf des Dokuments definiert wurde, als Aktion des Event-Handlers genutzt. Browser, die JavaScript nicht unterstützen, ignorieren dieses Attribut.

Auf Ereignisse reagieren

JavaScript-Anweisungen werden oft sofort nach dem Laden einer Webseite ausgeführt. Die Anweisungen einer Funktion werden jedoch nicht sofort aufgerufen, sondern erst nach einem Aufruf der Funktion abgearbeitet. Der Anwender entscheidet, wann die Berechnung gestartet wird, beispielsweise wenn er eine Schaltfläche mit der Bezeichnung Berechnung starten betätigt.

JavaScript stellt uns zahlreiche Ereignisse zur Verfügung. Tritt ein bestimmtes Ereignis ein, wird die angegebene Funktion aufgerufen und die enthaltenen Anweisungen werden abgearbeitet.



Die am häufigsten verwendete Abfrage ist, ob mit der Maus auf ein bestimmtes Element, z.B. eine Schaltfläche, geklickt wurde.

onClick	Mit onClick (beim Klick) können Sie abfragen, ob ein Besucher mit der Maus auf ein bestimmtes Element klickt. Dieses Ereignis kann laut der HTML-4.0-Spezifikation in folgenden HTML-Elementen abgefragt werden: <a> <abbr> <acronym> <address> <area> <b> <big> <blockquote> <body> <button> <caption> <cite> <code> <col> <colgroup> <dd> <del> <dfn> <dir> <dl> <dt> <fieldset> <form> <h1> <h2> <h3> <h4> <h5> <h6> <hr> <i> <img> <input> <ins> <kbd> <label> <legend> <link> <map> <menu> <noframes> <noscript> <object> <optgroup> <option> <p> <pre> <q> <s> <samp> <select> <small> <strike> <sub> <sup> <table> <tbody> <td> <textarea> <tfoot> <th> <thead> <tr> <tt> <u> <var>

Beispiel: onClick.html

Im folgenden Beispiel wird die Funktion `meidung()` aufgerufen, sobald der Anwender die Schaltfläche Hier klicken betätigt.

onClick.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Ereignisse in JavaScript</title>
  <script type="text/javascript">
    function meldung()
    {
      alert("Sie haben auf die Schaltfläche geklickt. ");
    }
  </script>
</head>
<body>
  <h3>Auf das Ereignis <i>onClick</i> reagieren</h3>

  <form action="">
    <input type="button" value="Hier klicken" onClick="meldung()">
  </form>
</body>
</html>
```

Es gibt noch eine Vielzahl anderer Ereignisse, bei denen Sie eine bestimmte Funktion aufrufen können. Die Bezeichnungen der Ereignisse können beliebig groß- oder kleingeschrieben werden, da es sich nicht um JavaScript-Code handelt. Beachten Sie, dass die Browser diese Ereignisse nicht im gleichen Umfang in allen HTML-Tags unterstützen.

Ereignis	Wird ausgelöst ...	Nach HTML 4.0 erlaubt in ...
onAbort	bei Abbruch des Ladens einer Webseite	<img>
onBlur	beim Verlassen eines Elements	<a> <area> <button> <input> <label> <select> <textarea>
onChange	bei Änderungen von Angaben	<input> <select> <textarea>
onDblClick	beim doppelten Anklicken - wird kaum verwendet	In fast allen HTML-Tags
onError	im Fehlerfall (z. B. falsche Bildangabe)	<img>
onFocus	beim Aktivieren eines selektierbaren Elements	<a> <area> <button> <input> <label> <select> <textarea>
onKeyDown	beim Betätigen einer Taste	In fast allen HTML-Tags
onKeyPress	beim Betätigen einer Taste	In fast allen HTML-Tags
onKeyUp	nach dem Loslassen einer Taste	In fast allen HTML-Tags
onLoad	beim Laden einer Webseite	<frameset> <body>
onMouseDown	beim Betätigen der Maustaste	In fast allen HTML-Tags
onMouseOut	beim Verlassen eines Elements mit der Maus	In fast allen HTML-Tags
onMouseOver	beim Überfahren eines Elements mit der Maus	In fast allen HTML-Tags
onMouseUp	nach dem Loslassen der Maustaste	In fast allen HTML-Tags
onReset	beim Zurücksetzen eines Formulars	<form>
onSelect	beim Selektieren von Text in Eingabefeldern	<input> <textarea>
onSubmit	beim Absenden von Formulardaten	<form>
onUnload	beim Verlassen einer Webseite	<frameset> <body>

Eine Webseite laden und verlassen

Die Event-Handler onLoad und onUnload werden innerhalb des <body>- oder <frameset>-Tags definiert. Die Ereignisse werden ausgelöst, sobald die Webseite geladen oder verlassen wird.

Die Ereignisse eignen sich deshalb, um Besucher zu begrüßen oder beim Verlassen der Webseite eine Umfrageseite einzublenden. Das Ereignis onLoad wird ebenfalls in vielen Webseiten eingesetzt, um Werbefenster einzublenden.

Beispiel: onLoad_onUnload.html

Beim Laden wird ein Begrüßungsfenster angezeigt. Wenn Sie die Webseite über den enthaltenen Link verlassen, wird ebenfalls ein Meldungsfenster zur Verabschiedung geöffnet.

onLoad_onUnload.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
```

```
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Ereignisabfragen: onLoad und onUnload</title>
  <script type="text/javascript">
    function meldung1()
    {
      alert("Ereignis: OnLoad\nHerzlich willkommen auf meiner
Webseite!");
    }

    function meldung2()
    {
       alert("Ereignis: OnUnload\nSie wollen schon gehen? Vielen Dank
für Ihren Besuch...");
    }
  </script>
</head>
<body onLoad="meldung1()" onUnLoad="meldung2()">
  <h3>Mehrere Ereignisabfragen</h3>
  <p>Im &lt;body&gt; sind die Ereignisse OnLoad und OnUnload
definiert.<br>

  <a href="weiterleitung.html">Verlassen Sie die Seite</a>, um das
Ereignis OnUnLoad auszuführen.
</body>
</html>
```

Ereignisse bei Grafiken, Links und Ankern

Die folgenden Event-Handler können Sie in Bildern, Links und Ankern verwenden.

Event-Handler	HTML-Tag	Wird aktiviert, wenn ...
onAbort	<img>	der Benutzer das Laden einer Grafik abbricht
onClick	<a> <img>	der Benutzer mit der Maus auf einen Link klickt
onError	<img>	die Grafik nicht geladen werden konnte
onMouseOut	<a> <img>	die Maus die Grafik oder den Link-Verweis verlässt
onMouseOver	<a> <img>	die Maus sich über einer Grafik oder einem Link befindet

Die Event-Handler `onAbort` und `onError` werden in Verbindung mit Grafiken benutzt. Sie werden innerhalb des Tags `<img>` gesetzt und sorgen dafür, dass beim misslungenen Laden einer Grafikdatei dem Benutzer eine Meldung angezeigt wird. Das Ereignis `Abort` tritt ein, wenn der Benutzer das Laden einer Grafik unterbricht. Das Ereignis `onError` wird ausgelöst, wenn die im HTML-Dokument

angegebene Grafik vom Browser nicht geladen werden kann.

Der Browser Firefox setzt den Event-Handler onError nicht um und kann damit nicht auf das fehlerhafte Laden von Grafiken reagieren.

Der Event-Handler `onciick` kann für Links und Grafiken definiert werden und wird ausgelöst, wenn der Benutzer auf einen Link oder eine Grafik klickt.

Die Event-Handler `onMouseOut` und `onMouseOver` reagieren auf Ereignisse, die durch Bewegung mit der Maus ausgelöst werden. Ein `onMouseOver`-Ereignis tritt ein, sobald sich die Maus über dem festgelegten HTML-Element befindet. Beim Verlassen eines Elements tritt das `onMouseOut`-Ereignis ein.

Das Haupteinsatzgebiet der beiden Ereignisse `onMouseOut` und `onMouseOver` ist die gemeinsame Verwendung bei einem Rollover-Effekt. Rollover bedeutet, dass eine Grafik sich in dem Moment ändert, in dem die Maus über sie fährt. Verlässt die Maus die Grafik, kehrt die Grafik zur ursprünglichen Anzeige zurück. Diesen Effekt finden wir in den Navigationen vieler Webseiten.

Beispiel: onMouse_onError.html

Das Beispiel zeigt, wie Sie einen einfachen Rollover-Effekt erzeugen. Zunächst müssen Sie zwei gleichgroße Grafiken anfertigen, die dann gegeneinander ausgetauscht werden sollen.

onMouse_onError.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Ereignisabfragen: onMouseOver, onMouseOut, onError</title>
  <script type="text/javascript">
      var grafik = new Array;
      grafik[0] = new Image;
      grafik[0].src = "images/bild1.jpg";
      grafik[1] = new Image;
      grafik[1].src = "images/bild_2.jpg";

      function Bildwechsel(nummer)
      {
        document.images[0].src = grafik[nummer].src;
      }

      function Grafikfehler()
      {
        alert("Fehler! Die Grafik kann nicht geladen werden.");
      }
  </script>
</head>
<body>
  <h3>onMouseOver, onMouseOut, onError</h3>
```

```
  <a href="#" onMouseover="Bildwechsel(1)"
onMouseout="Bildwechsel(0)"><img
     src="images/bild1.jpg" width="100" height="75" border="0"
alt="Grafikwechsel" onError="Grafikfehler()"></a>
</body>
</html>
```

Ereignisse im Formular

Event-Handler	Tag	Ereignis wird ausgelöst, wenn ...
onBlur	<textarea></textarea> <select></select> <input type="text | password">	der Benutzer das Feld verlässt
onChange	<textarea></textarea> <select></select> <input type="text | password">	der Benutzer das Feld verlässt und eine Änderung am Inhalt vorgenommen hat
onClick	<input type="button | checkbox | radio | reset | submit">	der Benutzer das Element anklickt
onFocus	<textarea></textarea> <select></select> <input type="text | password">	der Benutzer das Feld aktiviert, z. B. durch eine Anwahl über die Tabulator-Taste
onReset	<form>	der Benutzer die Reset-Schaltfläche betätigt
onSelect	<textarea></textarea> <input type="text | password">	der Benutzer eine Textstelle im Eingabefeld markiert
onSubmit	<form>	der Benutzer die Submit-Schaltfläche betätigt

Beispiel: form_events.html

In diesem Beispiel wird ein Formular mit zwei Eingabefeldern für den Namen und den Vornamen erzeugt. Beim Auftreten der Ereignisse `onFocus`, `onBlur` und `onChange` wird eine Meldung in der Statusleiste oder ein Meldungsfenster angezeigt.

form_events.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Events im Formular</title>
```

```
</head>
<body>
  <form action="">
    <p>Vorname:<br>
    <input name="vorname" size="50" maxlength="70" type="text"
       onFocus = "window.status='Bitte geben Sie Ihren Vornamen ein.';"
       onBlur = "window.status='';"
       onChange = "alert(this.value + ' ist ja ein netter Name!');">
    <p>Nachname:<br>

    <input name="name" size="50" maxlength="70" type="text"
       onFocus = "window.status='Bitte geben Sie Ihren Nachnamen
ein.';"
       onBlur = "window.status='';"
       onChange = "alert(this.value + ' klingt aber
ungew&ouml;hnlich...');">
  </form>
</body>
</html>
```

Event-Handler ''onClick''

Die Eingabefelder eines Formulars lassen sich im Zusammenhang mit Event-Handlern auch für andere Zwecke nutzen. Beispielsweise können die Werte der Formularfelder nach dem Betätigen einer Schaltfläche als Eingabe einer Berechnung dienen, deren Ergebnis wiederum in einem anderen Formularfeld ausgegeben wird.

Beispiel: volumen_rechner.html

Dieses Beispiel zeigt Ihnen, wie wir nach der Eingabe eines Wertes und der Auswahl einer Maßeinheit die Umrechnungen in andere Maßeinheiten durchführen und die Werte ausgeben.

volumen_rechner.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Event: onClick</title>
  <script type="text/javascript">
      function umrechnen()
      {
        var menge = document.Volumen.eingabewert.value;
        switch (document.Volumen.einheit.options.selectedIndex)
        {
          case 0: tausender = 1; break;
          case 1: tausender = 1000; break;
```

```
          case 2: tausender = 1000 * 1000; break;
          case 3: tausender = 1000 * 1000 * 1000; break;
          case 4: tausender = 1000 * 1000; break
          case 5: tausender = 1000 * 1000 * 100; break
        }
        document.Volumen.cmm.value = menge * tausender;
        document.Volumen.ccm.value = menge * tausender / 1000;
        document.Volumen.cdm.value = menge * tausender / 1000 / 1000;
        document.Volumen.cm.value  = menge * tausender / 1000 / 1000 /
1000;
        document.Volumen.liter.value = menge * tausender / 1000 / 1000;
        document.Volumen.hektoliter.value = menge * tausender / 1000 /
1000 / 100;
      }
  </script>
</head>
<body>
  <h3>Volumenma&szlig;e berechnen</h3>
  <form action="" name="Volumen">

    <p><input type="text" name="eingabewert" size="6"
maxlength="5"><select name="einheit">
      <option value="cmm">mm&sup3;</option>
      <option value="ccm">cm&sup3;</option>
      <option value="cdm">dm&sup3;</option>
      <option value="cm">m&sup3;</option>
      <option value="l" selected>l (Liter)</option>

      <option value="hl">hl (Hektoliter)</option>
    </select>
    <p><input type="button" name="los" value="Umrechnen"
onClick="umrechnen();">
    <p><input type="text" name="cmm" readonly> mm&sup3;
    <br><input type="text" name="ccm" readonly> cm&sup3;
    <br><input type="text" name="cdm" readonly> dm&sup3;

    <br><input type="text" name="cm" readonly> m&sup3;
    <br><input type="text" name="liter" readonly> l (Liter)
    <br><input type="text" name="hektoliter" readonly> hl (Hektoliter)
  </form>
</body>
</html>
```

Formulareingaben testen

Eine wesentliche Anwendung von JavaScript ist es, Formulardaten auf Fehleingaben zu prüfen, bevor sie über das Internet versendet werden. Über den Event-Handler `onSubmit` können wir vor dem Absenden der Daten auf die Inhalte des Formulars zugreifen. Somit ist schon eine Kontrolle der eingegebenen Daten auf dem Clientrechner möglich, um bei Fehleingaben das Versenden des Formulars zu verhindern.

Beispiel: kontaktfomular.html

Das folgende umfangreichere Beispiel zur Prüfung der Formulareingaben zeigt ein Anmeldeformular, das Adressdaten des Benutzers aufnehmen und versenden soll. Das zugrunde liegende Skript gibt bei der Eingabe Hilfestellungen in der Statuszeile, führt vor dem Versenden eine Überprüfung der Formularfelder durch und fragt nach, ob die Felder beim Betätigen der Schaltfläche Zurücksetzen auf die Vorgabewerte gesetzt werden sollen.

Außerdem sollen einige Eingabefelder auf korrekte Eingaben überprüft werden. Zum Beispiel dürfen die Telefonnummer und die Postleitzahl keine Buchstaben beinhalten.

Zuerst erstellen wir die Webseite mit den einzelnen Formularelementen. Im zweiten Schritt werden im Kopfbereich des HTML-Dokuments die JavaScript-Funktionen zum Auswertendes Formulars eingefügt.

kontaktfomular.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
  <title>Formularauswertung</title>
  <script type="text/javascript">
      function testeZeichen (testString, erlaubteZeichen)
      {
        var allezeichenok = true;
        for (var i = 0; i < testString.length ; i++)
          if (erlaubteZeichen.indexOf(testString.charAt(i)) == -1)
             allezeichenok = false;
        return allezeichenok;
      }

      function testemailadresse(testString)
      {
        var suche = /^[\w\.\-]{2,}\@[äöüa-z0-9\-\.]{1,}\.[a-z]{2,4}$/i;
        return suche.test(testString);
      }

      function checkform(myform)
      {
        if (myform.name.value == "")
        {
          alert ("Bitte geben Sie Ihren Nachname an!");
```

```
          return false;
        }
        else if (myform.vorname.value == "")
        {
          alert ("Bitte geben Sie Ihren Vornamen an!");
          return false;
        }
        else if (myform.telefon.value == "")
        {
          alert ("Bitte geben sie Ihre Telefonnummer an!");
          return false;
        }
        else if (!testeZeichen(myform.telefon.value, "1234567890-/"))
        {
          alert ("Geben Sie für die Telefonnummer bitte nur Zahlen
ein!");
          return false;
        }
        else if (myform.anschrift.value == "")
        {
          alert("Bitte geben Sie Ihre Anschrift an!");
          return false;
        }
        else if (myform.plz.value == "")
        {
          alert("Bitte geben sie Ihre Postleitzahl an!");
          return false;
        }
        else if (!testeZeichen (myform.plz.value, "1234567890"))
        {
          alert ("Geben Sie für die Postleitzahl bitte nur Zahlen
ein!");
          return false;
        }
        else if (myform.plz.value.length != 5)
        {
          alert ("Fünf Stellen sollte die Postleitzahl schon haben!");
          return false;
        }
        else if (myform.ort.value == "")
        {
          alert ("Bitte geben Sie Ihren Wohnort an!");
          return false;
        }
        else if (myform.mail.value == "")
        {
          alert ("Bitte geben Sie Ihre E-Mail-Adresse an!");
          return false;
        }
        else if (!testemailadresse(myform.mail.value))
        {
```

```
          alert ("Die eingegebene E-Mail-Adresse ist nicht korrekt!");
          return false;
        }
        return confirm("Überprüfung abgeschlossen.\nMöchten Sie die
Daten jetzt absenden?");
      }
      function checkreset(myform)
      {
        return confirm("Wollen Sie die Eingaben wirklich auf die
Ausgangswerte zurücksetzen?");
      }

      function putstatus(mytext)
      {
        window.status = mytext;
      }

      function clearstatus()
      {
        window.status = "";
      }
  </script>
</head>
<body>
  <h3>Anmeldeformular</h3>
    <form name="anmeldeform" action="anmeldung.php" method="get"
onSubmit="return checkform(this)" onReset="return checkreset(this)">
      <table summary="Formularelemente anordnen" width="500"
cellspacing="0" cellpadding="5" border="0">

        <tr>
          <td width="100">Anrede</td>
          <td><input type="radio" name="anrede" value="Frau"
checked>Frau
              <input type="radio" name="anrede" value="Herr">Herr</td>
        </tr>
        <tr>
          <td>Nachname:</td>
          <td><input name="name" size="50" maxlength="70"
type="text"></td>

        </tr>
        <tr>
          <td>Vorname:</td>
          <td><input name="vorname" size="50" maxlength="70"
type="text"></td>
        </tr>
        <tr>
          <td>Telefonnummer:</td>
          <td><input name="telefon" size="15" type="text"
maxlength="15" onFocus="putstatus('Bitte geben Sie nur Zahlen ein.');"
```

```
onBlur="clearstatus();"></td>

        </tr>
        <tr>
          <td>Stra&szlig;e</td>
          <td><input name="anschrift" size="50" maxlength="70"
type="text" onFocus="putstatus('Bitte geben Sie die Stra&szlig;e und
Hausnummer ein.');" onBlur="clearstatus();"></td>
        </tr>
        <tr>
           <td>Postleitzahl</td>

           <td><input name="plz" size="5" maxlength="5" type="text"
onFocus="putstatus('Bitte geben Sie nur Zahlen ein.');"
onBlur="clearstatus();"></td>
        </tr>
        <tr>
          <td>Ort</td>
          <td><input name="ort" size="50" maxlength="70"
type="text"></td>
        </tr>
        <tr>
          <td>E-Mail-Adresse</td>

          <td><input name="mail" size="50" maxlength="50"></td>
        </tr>
      </table>
      <p>
        <input type="submit" value="Anmelden">
        <input type="reset" value="Zur&uuml;cksetzen">
      </p>
    </form>
</body>
</html>
```

javaScript:

Die Anweisung `javaScript:` ist im eigentlichen Sinne kein Event-Handler. Sie erlaubt Ihnen jedoch das Ausführen von einzelnen JavaScript-Befehlen über einen Hyperlink. Deshalb ist dieser Befehl auch nur in Verbindung mit dem HTML-Tag <a href =„“ > möglich.

`<a href="javascript:"></a>`	Nach der Anweisung `javascript:` müssen Sie einen JavaScript-Befehl angeben, der nach einem Klick auf den Verweis ausgeführt werden soll.

Beispiel

Nach dem Klick auf den Hyperlink wird eine Meldung ausgegeben.

```
<a href="JavaScript:alert('Sie haben diesen Dialog von einem Verweis aus geöffnet.')">Dialogfenster öffnen</a>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_11

Last update: **2019/05/16 16:42**

11.3.12) Unsere Scripts

Reines Javascript

javascript0.js

```
var summe = 0,
max = 100,
i = 1;
while (i <= max) {
  summe = summe + i;
  i++;
}
console.log(summe);
```

JavaScript in HTML

Ausgabe auf der Console

javascript1.html

```
<html>
<head>
<title>JavaScript in Aktion</title>
</head>
<body>
<h1>JavaScript in Aktion</h1>
<script type="text/javascript">
var summe=0,max=100,i=1;
while (i<=max){
    summe+=i;
    i++;
}
console.log(summe);
</script>
</body>
</html>
```

Ein- und Ausgabe mittels Prompt und Alert

javascript2.html

```
<html>
<head>
<title>JavaScript in Aktion</title>
</head>
<body>
<h1>JavaScript in Aktion</h1>
<script type="text/javascript">
var summe=0,max,i=1;
max=prompt("Zahl, bis zu der summiert werden soll: ","200");
while (i<=max){
    summe+=i;
    i++;
}
alert("Die Summe von 1 bis "+max+" ist: "+summe);
</script>
</body>
</html>
```

Ausgabe mittels document.write

javascript3.html

```
<html>
<head>
<title>JavaScript in Aktion</title>
</head>
<body>
<h1>JavaScript in Aktion</h1>
Die Summe von 1 bis 100 ist
<script type="text/javascript">
var summe=0,max=100,i=1;
while (i<=max){
    summe+=i;
    i++;
}
document.write("der Wert "+summe);
</script>
</body>
</html>
```

Interaktion mittels Zugriff auf HTML-Elemente

javascript4.html

```
<html>
```

```
<head>
<title>JavaScript in Aktion</title>
<script type="text/javascript">
function summieren(){
  var max=Number(document.getElementById("eingabe").value);
  alert(max);
  if (isNaN(max) || max==0) {
     document.getElementById("ausgabe").innerHTML="?";
  }
  else {
     var summe=0,i=1;
     while (i<=max){
         summe+=i;
         i++;
     }
     document.getElementById("ausgabe").innerHTML=summe;
  }
}
</script>
</head>
<body>
<h1>JavaScript in Aktion</h1>
<hr>
Die Summe von 1 bis <input type="text" id="eingabe" size="3"
maxlength="6" onkeyup="summieren()"></input> ist <span
id="ausgabe"></span>
</body>
</html>
```

button.html

```
<html>
<head>
<title>JavaScript in Aktion</title>
<script type="text/javascript">
summe=0;
function summiere(){
  summe+=parseInt(document.getElementById("eingabe").value);
  document.getElementById("ausgabe").innerHTML=summe;
}
</script>
</head>
<body>
<h1>JavaScript in Aktion</h1>
<hr>
Zahl <input type="text" id="eingabe" size="3"></input>
Summe: <span id="ausgabe"></span>
<br><br>
<button onclick="summiere()">Zahl addieren</button>
<br><br>
```

```
<a href="javascript:summiere()" >Zahl addieren</a>
</body>
</html>
```

Typkonvertierung

- Variablen
- Operatoren
- Konvertierungs-Operatoren

Eingabeabsicherung

javascript5.html

```
<html>
<head>
<title>JavaScript in Aktion</title>
<script type="text/javascript">
function summieren(){
  var max=Number(document.getElementById("eingabe").value);
  if (isNaN(max) || max==0) {
     document.getElementById("ausgabe").innerHTML="?";
  }
  else {
     var summe=0,i=1;
     while (i<=max){
         summe+=i;
         i++;
     }
     document.getElementById("ausgabe").innerHTML=summe;
  }
}
</script>
</head>
<body>
<h1>JavaScript in Aktion</h1>
<hr>
Die Summe von 1 bis <input type="text" id="eingabe" size="3"
maxlength="6" onkeyup="summieren()"></input> ist <span
id="ausgabe"></span>

</body>
</html>
```

Event mit Eventhandler behandeln

javascript6.html

```
<!doctype html>   //wir entwickeln für HTML 5
<html lang="de">
<head>
<meta charset="utf-8">     //damit der Browser weiß, dass unser File in
UTF-8 gespeichert ist, damit sollten Umlaute richtig angezeigt werden.
<title>JavaScript in Aktion</title>
</head>
<body>
<h1>JavaScript in Aktion</h1>
<hr>
Der Kreis mit Radius <input type="text" id="eingabe" size="3"></input>
besitzt die Fläche <span id="ausgabe"></span>

//Das ganze Script kann auch in eine eigene Datei gegeben werden.
<script>       //Es ist nicht mehr nötig, JavaScript anzugeben, da
JavaScript die Standardsprache für jeden Browser ist.
var eingabe=document.getElementById("eingabe");     //Eingabeobjekt
var ausgabe=document.getElementById("ausgabe");     //Ausgabeobjekt
eingabe.addEventListener('keyup', kreisflaeche);    //EventListener
wird an das Objekt 'eingabe' gebunden, wenn dort ein 'keyup' auftritt,
soll etwas gemacht werden
                                                    //bei
'kreisflaeche' sind keine runden Klammern nötig, da kreisflaeche selber
ein Objekt ist.
                                                    //EventListener
darf erst nach dem Element 'eingabe' aufgerufen werden!
function kreisflaeche() {
  var radius=parseFloat(eingabe.value);
  var flaeche=Math.PI*Math.pow(radius,2);
  flaeche=flaeche.toFixed(2);                //rundet auf 2 Stellen
  ausgabe.innerHTML=flaeche;
}
</script>

</body>
</html>
```

besser:

Event mit Eventhandler behandeln

javascript6a.html

```
<!doctype html>
<html lang="de">
<head>
<meta charset="utf-8">
<title>JavaScript in Aktion</title>
</head>
<body>
<h1>JavaScript in Aktion</h1>
<hr>
Der Kreis mit Radius <input type="text" id="eingabe" size="3"></input>
besitzt die Fläche <span id="ausgabe"></span>
</body>
<script scr="ourscript.js"></script>
</html>
```

ourscript.js

```
var eingabe=document.getElementById("eingabe");
var ausgabe=document.getElementById("ausgabe")
eingabe.addEventListener('keyup', kreisflaeche);

function kreisflaeche() {
  var radius=parseFloat(eingabe.value);
  var flaeche=Math.PI*Math.pow(radius,2);
  flaeche=flaeche.toFixed(2);
  ausgabe.innerHTML=flaeche;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:11:11_03:11_03_12

Last update: **2019/05/16 16:43**

12) PHP

12.1) Was ist PHP?

- 12.1.1) Was ist PHP
- 12.1.2) How to use PHP

12.2) Grundlegende Sprachelemente

- 12.2.1) Kommentare
- 12.2.2) Ausgabe
- 12.2.3) ÜBUNG 1

12.3) Variablen und Operatoren

- 12.3.1) Variablen und Operatoren

12.4) Interaktive Webseiten

- 12.4.1) Formulare
- 12.4.2) ÜBUNG 2

12.5) Anführungszeichen

- 12.5.1) Verwenden von Anführungszeichen

12.6) Kontrollstrukturen

- 12.6.1) Vergleichsoperatoren
- 12.6.2) if-Anweisung
- 12.6.3) switch-Anweisung
- 12.6.4) ÜBUNG 3-7

12.7) Schleifen

- 12.7.1) for-Schleife
- 12.7.2) while-Schleife
- 12.7.3) do-while-Schleife
- 12.7.4) foreach-Schleife

- 12.7.5) ÜBUNG 8-11

12.8) Felder

- 12.8.1) Grundlagen zu Felder
- 12.8.2) Indizierte Felder
- 12.8.3) Assoziative Felder
- 12.8.4) Mehrdimensionale Felder
- 12.8.5) Weitere Informationen zu Feldern
- 12.8.6) ÜBUNG 12-18
- 12.8.7) KONTROLLFRAGEN

12.9) Formulare

- 12.9.1) Hidden-Feld
- 12.9.2) Textarea
- 12.9.3) Checkbox
- 12.9.4) Radio-Button
- 12.9.5) Auswahllisten
- 12.9.6) ÜBUNG 20

12.10) Externe Dateien

- 12.10.1) Theorie: Nutzung von externen Dateien
- 12.10.2) Dateien öffnen,lesen und schließen
- 12.10.3) Aus Dateien lesen
- 12.10.4) In Dateien schreiben
- 12.10.5) ÜBUNG 21 Gästebuch
- 12.10.6) ÜBUNG 22 Besucher zählen
- 12.10.7) ÜBUNG 23-25

12.11) Datum und Zeit

- 12.112.1) Datum und Zeit
- 12.112.2) ÜBUNG 26

12.12) Funktionen

- 12.12.1) Funktionen erstellen und aufrufen
- 12.12.2) Funktionen verwenden
- 12.12.3) Gültigkeitsbereich von Variablen
- 12.12.4) PHP-Dateien einbinden

- 12.12.5) Andere Dateitypen einbinden
- 12.12.6) ÜBUNG 27-29

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12

Last update: **2019/05/16 17:11**

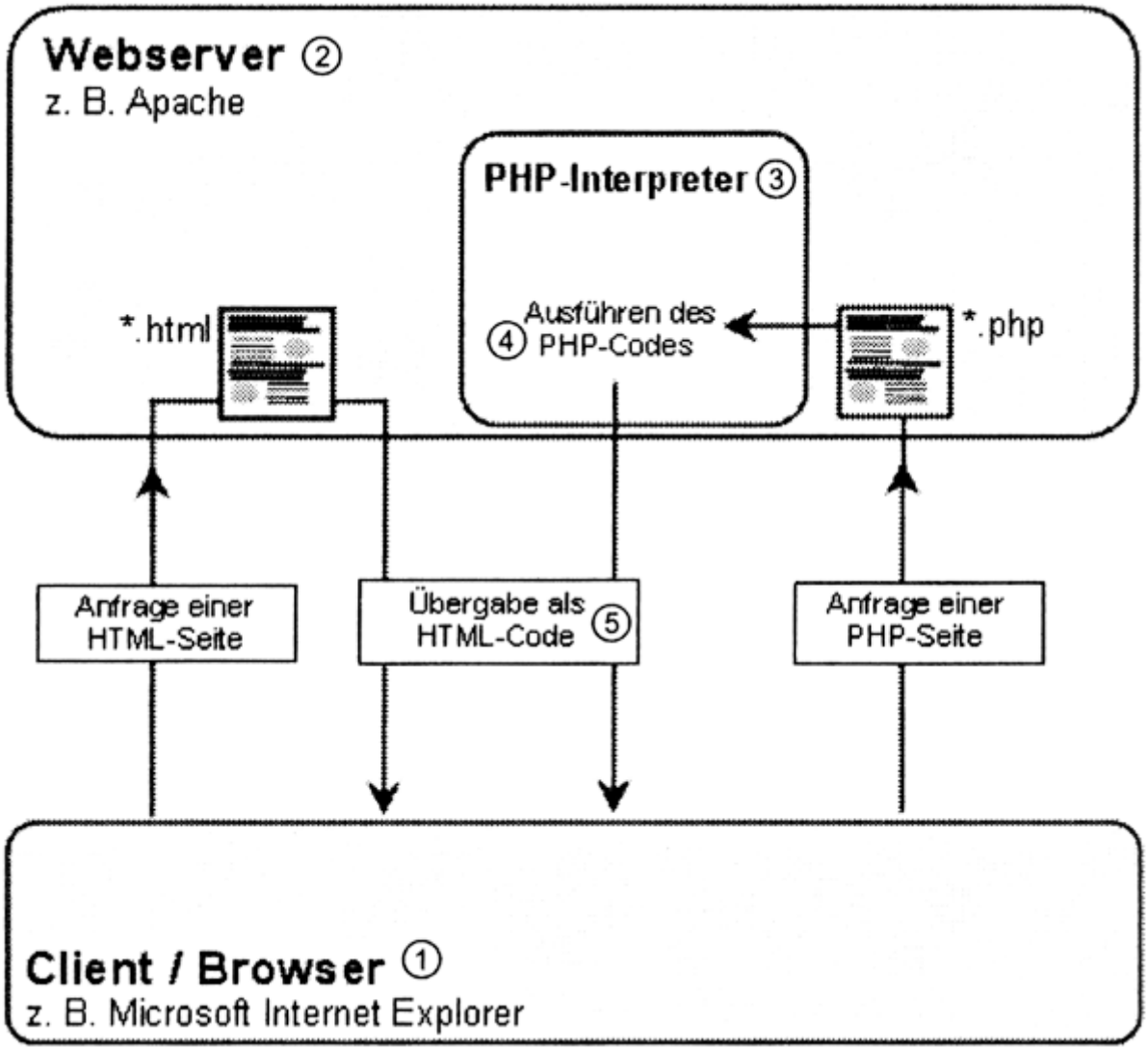
1.1.1) Was ist PHP?

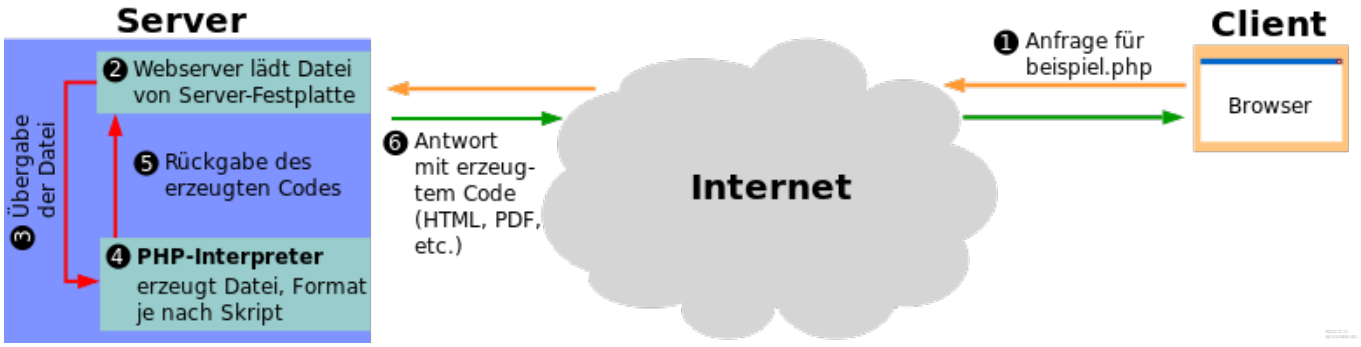
PHP (= PHP Hypertext Preprocessor) ... **Skriptsprache** für Internet, wird am **Webserver ausgeführt**.

PHP, die Kurzform von Personal HomePage oder auch PHP Hypertext Preprocessor, ist eine Skriptsprache, die speziell für den **Einsatz im Internet** entwickelt wurde. Die auszuführenden Anweisungen können **direkt in den HTML-Code integriert** werden.

Bei einer mit **HTML erstellten Webseite ist keine Dynamik** möglich. Mithilfe von **PHP** können auf einer Webseite **Interaktionen** eingebaut oder die Seite in irgendeiner Form, beispielsweise durch **Reaktionen auf Eingaben**, angepasst werden.

Öffnet der Betrachter in seinem Browser (1) eine solche Webseite über einen **PHP-fähigen Webserver** (2), werden die Anweisungen von PHP interpretiert (3), ausgeführt (4) und das Ergebnis als HTML-Code zurückgesendet (5). Ein klarer Vorteil dabei ist, dass der PHP-Quellcode dem Betrachter nicht zugänglich ist. Er sieht im Browser nur den zurückgelieferten HTML-Code.

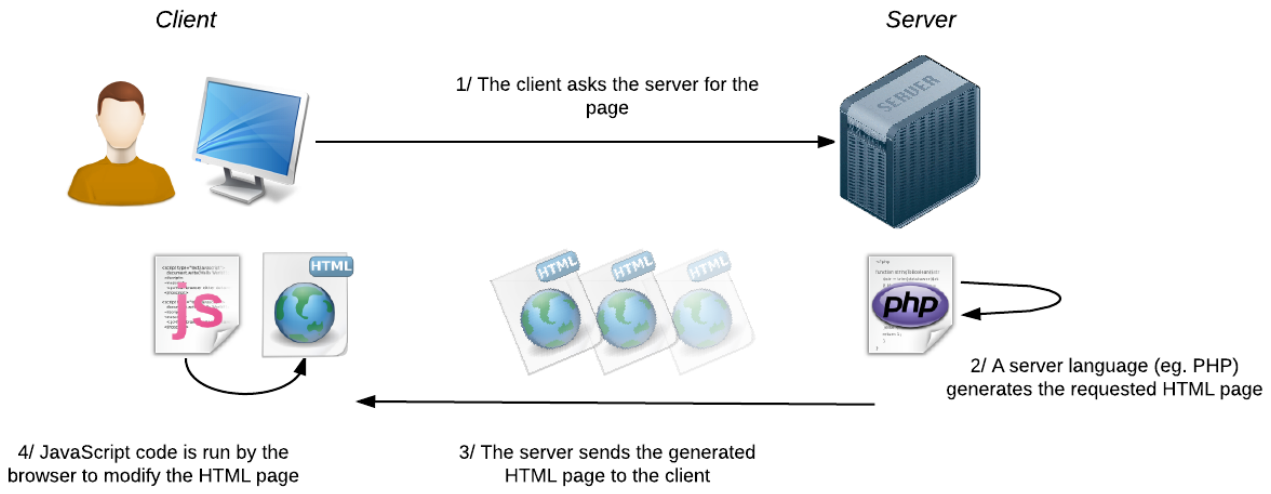


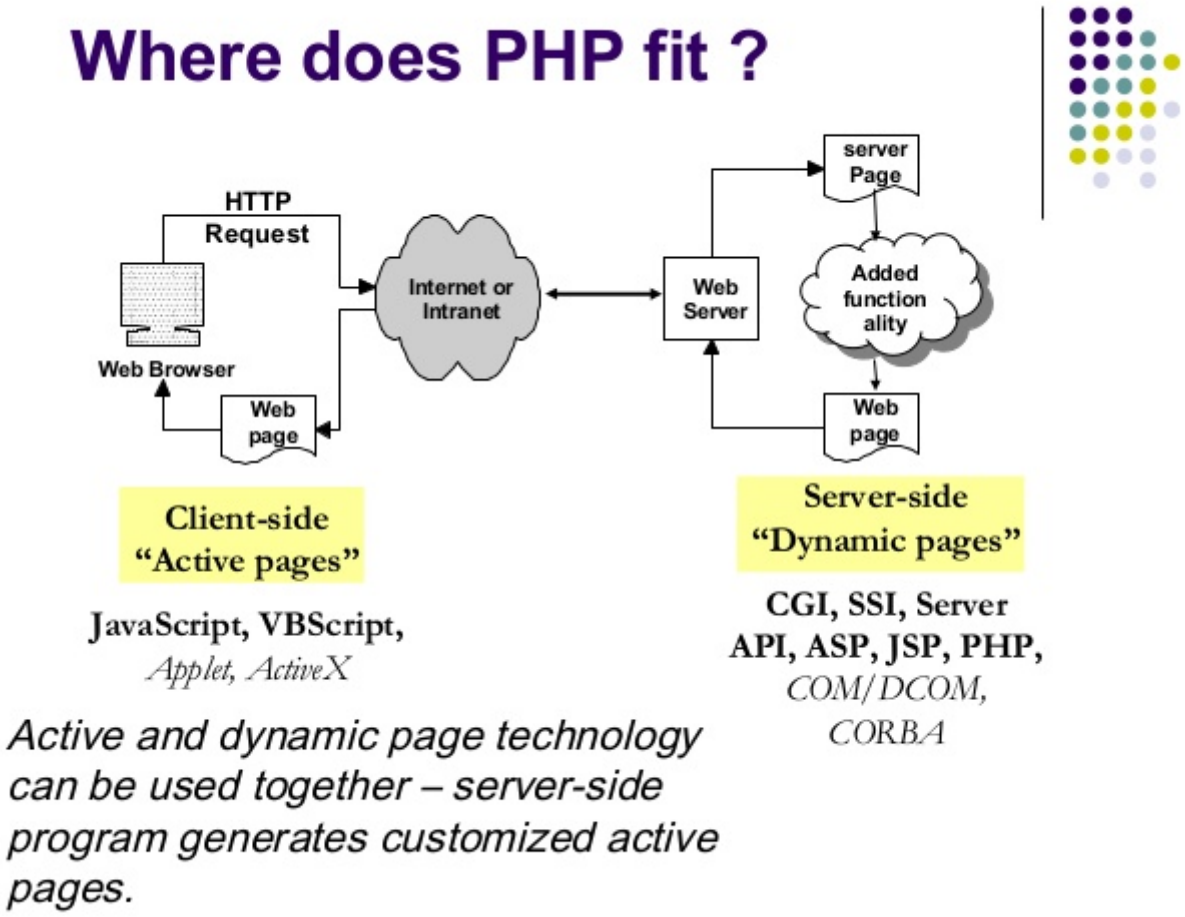


Um dynamische Webseiten zu erstellen und später auf Datenbanken zugreifen zu können, benötigt man neben PHP einen Webserver und einen Datenbankserver. Unter Windows kann man einen lokalen Webserver installieren. Als Entwicklungssoftware hat sich XAMPP sehr bewährt.

Unterschied zu Javascript

Im Gegensatz zu Javascript benötigt PHP einen Webserver, der eine Interaktion mit dem Benutzer ermöglicht und somit eine Webseite dynamisch erscheinen lässt.





Download von XAMPP

- http://www.apachefriends.org/de/xampp.html
- localhost greift auf den Ordner **…**`/htdocs` zu. In diesen Ordner müssen die html- bzw. php-Dateien gespeichert werden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_01

Last update: **2019/05/16 17:06**

1.1.2) How to use PHP

PHP-Befehle einfügen

PHP-Code wird direkt in den HTML-Code eingebettet. PHP-Blöcke können vollständig im Dokumentkopf <head> oder im Dokumentrumpf <body> stehen. Weiterhin hat man die Möglichkeit, innerhalb einer Datei mehrmals zwischen PHP- und HTML-Code zu wechseln.

Dateinamenerweiterungen mit PHP-Code

Damit der Webserver erkennt, dass es sich um eine Datei mit einem PHP-Aufruf handelt, werden die Dokumente mit der Dateinamenerweiterung `.php` oder `.php5` gespeichert.

XML-Schreibweise

Der auszuführende PHP-Code wird zwischen dem öffnenden Tag <?php und dem schließenden Tag ?> geschrieben

```
<?php
  hier steht ein (oder mehrere) PHP-Befehl(e)
?>
```

Diese Schreibweise ist am gebräuchlichsten. Alternativ dazu kann man auch einen Skript-Tag verwenden. Diese Schreibweise entspricht im Wesentlichen der JavaScript- bzw. CSS-Angabe:

```
<script language="php">
  hier steht ein (oder mehrere) PHP-Befehl(e)
</script>
```

PHP-Anweisungen voneinander trennen

Jede PHP-Anweisung wird duch einen Strichpunkt beendet.

```
<?php
  PHP-Code;
  PHP-Code;
?>
```

Es empfiehlt sich, nach einer Anweisung in einer neuen Zeile weiterzuschreiben.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_02

Last update: **2019/05/16 17:06**

1.2.1) Kommentare einfügen

- Kommentare werden nicht vom Interpreter ausgeführt.
- Kommentare dienen zur näheren Beschreibung des Quellcodes

Syntax und Bedeutung der Kommentare

```
<html>
<body>
<?php
   echo "Für den Anfang: "; //einzeiliger Kommentar am Ende eines PHP-Codes
   // einzeiliger KOmmentar

   /*
     mehrzeilige Kommentare sind ebenfalls möglich,
     um den Quellcode ausführlicher zu beschreiben
   */
   echo "Hallo Welt!";
?>
</body>
</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_03

Last update: **2019/05/16 17:06**

1.2.2) Daten im Browser ausgeben

Befehl ''echo''

Da PHP eine eingebettete Skriptsprache ist, die vom Server ausgeführt wird, bleibt der Quellcode im Gegensatz zu HTML für den Client unsichtbar. Zeichenketten und Daten werden über den Befehl `echo` auf dem Bildschirm ausgegeben.

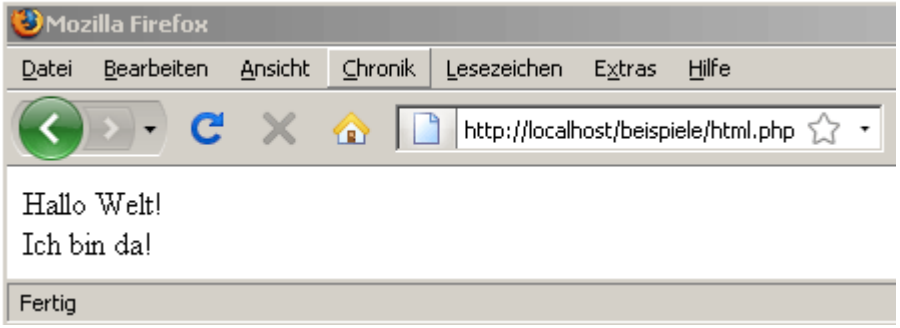
```
  echo "Hallo Welt";
```

Umgang mit HTML-Syntax bei der Ausgabe

Um Zeichenketten anzuzeigen, werden diese direkt an den Browser weitergegeben. Daher kann man innerhalb des Befehls `echo` auch jeden HTML-Tag (z.B. `<br>`) angeben. Dieser wird dann vom Browser als normaler HTML-Befehl interpretiert.

```
<html>
<body>
<?php
  echo "Hallo Welt! <br>"; //Der HTML-Tag <br> wird vom Browser korrekt interpretiert.
  echo "Ich bin da!";
?>
</body>
</html>
```

Screenshot Browser



Im Screenshot wird der HTML-Tag <br> richtig interpretiert. Der Browser fügt eine Leerzeile ein. Der Kommentar wird natürlich nicht angezeigt!

Quelltext HTML

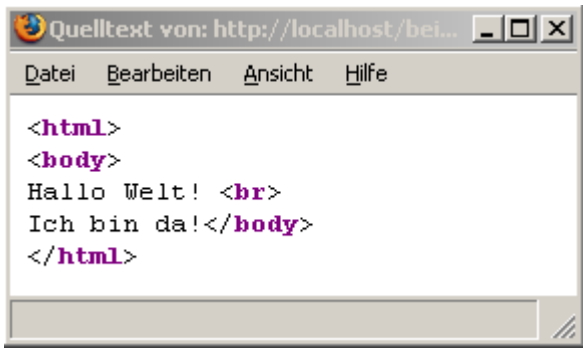


Wie man sieht, wird der PHP-Kommentar auch im HTML-Quelltext nicht angezeigt. Weiters erscheint der übersetzte HTML-Code in einer Zeile.

Der HTML-Quellcode sollte ebenfalls wie der PHP-Quellcode übersichtlich erscheinen. Mittels \n kann man in PHP erwirken, dass im HTML-Quelltext eine Leerzeilen eingefügt wird. \n hat keinen Einfluss auf das Erscheinungsbild im Browser.

```
<html>
<body>
<?php
  echo "Hallo Welt! <br>\n"; // \n bewirkt eine Leerzeile im HTML-Quellcode!
  echo "Ich bin da!";
?>
</body>
</html>
```

Das Ergebnis im HTML-Quelltext sieht wie folgt aus:



Sonderzeichen ausgeben

Wenn man Hochkommas oder einen Backslash ausgeben möchte, setzt man vor den Zeichen einen Backslash:

```
<?php
   echo "<p>Der \"Erfinder\" von PHP war Rasmus Lerdorf. </p>";
   echo "Das Programm wird installiert im Verzeichnis C:\\Programme";
?>
```

Zeilenumbruch im Quelltext

Möchte man im Quelltext einen Zeilenumbruch einfügen, verwendet man \n

```
<?php
   echo "Das ist ein Text<br>\n";
   echo "Das ist noch ein Text";
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_04

Last update: **2019/05/16 17:06**

1.2.3) Aufgabe 1

Schreibe ein einfaches PHP-Skript, z.B. eine einfache Webseite mit Informationen über dich (Name, Wohnort, Hobbies, ...)

Folgende Elemente sollen dabei vorkommen (eingebunden in PHP-Code):

- Kommentare
- Sonderzeichen, wie z.B. Anführungsstriche
- Farbformatierungen
- Listen
- Tabellen
- ...

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

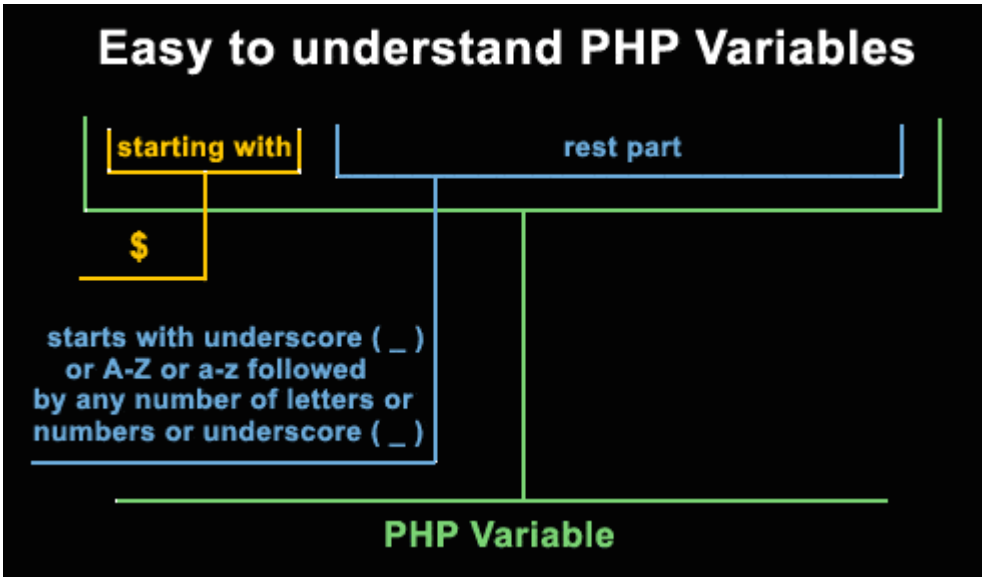
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_05

Last update: **2019/05/16 17:06**

1.3.1) Variablen und Operatoren

Variablen

Variablen müssen im Gegensatz zu vielen Programmiersprachen nicht extra deklariert werden. Sie werden mit einem vorangestellten Dollar- Zeichen gekennzeichnet und beim ersten Aufruf definiert.



Folgende Datentypen werden von PHP unterstützt:

Datentyp	**Bezeichnung**	**Beispiel**
Ganze Zahlen	**integer**	1911 oder -1911
Dezimalzahl	**double**	19.11 oder -19.11
Zeichenketten	**string**	„Kette von Zeichen“ oder 'Kette von Zeichen'
Felder (ein- oder mehrdimensional)	**array**	(„Frankfurt“, „Berlin“, „Zürich“)
Objekte	**object**	Verweis auf eine bestimmte Variable

Namensgebung bei Variablen

Bei der Vergabe von Namen für die Variablen gibt es folgende Regeln. Der Name einer Variablen

- muss mit dem Dollarzeichen beginnen und darf kein Leerzeichen enthalten
- darf nur aus Buchstaben und Ziffern bestehen, aber keine Umlaute oder „ß“ enthalten
- muss als zweites Zeichen einen Buchstaben enthalten
- darf keine Sonderzeichen außer dem Unterstrich „_“ enthalten
- kann Groß- oder Kleinbuchstaben enthalten, wobei zwischen Groß- und Kleinschreibung unterschieden wird (`$PrimZahl` ist nicht gleich `$primzahl`)
- darf nicht identisch sein mit einem so genannten reservierten Wort (`and, break, case, class, continue, default, do, else, elseif, empty, endfor, endif`)

Variablen und Operatoren für Zahlen

Numerische Datentypen

Die numerischen Datentypen werden in Ganzzahl-Datentypen `integer` und Fließkommazahl-Datentypen `double` unterteilt.

Beispiel: preis.php

```
<?php
$preis_apfel = 2.50;
$menge = 4;
$gesamtpreis = $preis_apfel * $menge;
echo $gesamtpreis;
?>
```

Arithmetische Operatoren

Operator	Name	Bedeutung
+	Addition	$a + $b ergibt die Summe
-	Subtraktion	$a - $b ergibt die Differenz
*	Multiplikation	$a * $b ergibt das Produkt
/	Division	$a / $b ergibt den Quotienten
%	Modulo	$a % $b ist der Rest der ganzzahligen Division von $a / $b
+ +	Prä-, bzw. Postinkrement	$a++ (++$a) erhöht die Variable a um 1 nach (vor) der weiteren Verwendung
- -	Prä-, bzw. Postdekrement	$a- - (- -$a) verringert die Variable a um 1 nach (vor) der weiteren Verwendung

Beispiel: berechnung.php

```
<?php
  $preis_apfel = 2.50;
  $menge1 = 4;
  $menge2 = 10;
  $menge3 = 15;
  $gesamtpreis = $preis_apfel * ($menge1 + $menge2 + $menge3);
  echo $gesamtpreis;
?>
```

Variablen und Operatoren für Zeichenketten

Zeichenkettenoperator

Mittels Zeichenkettenoperator „„.“ können mehrere Zeichenketten miteinander verknüpft werden:

```php
<?php
  $a = "Hallo ";
  $b = "Welt!";
  $c = $a.$b;
  echo $c;
?>
```

Ergebnis: Hallo Welt!

Typumwandlung

Der Datentyp einer Variablen wird durch die erste Wertzuweisung bestimmt.

Durch eine vorangestellte Typendeklaration kann dieser Datentyp geändert werden.

Typumwandlung in PHP funktioniert oft wie in C. Der Name des gewünschten Typs wird vor der umzuwandelnden Variablen in Klammern gesetzt, dies wird auch als Cast-Operation bezeichnet.

```php
<?php
      echo (int)$z;
      echo "<br>";
      var_dump($z);   //Gibt die Variable mit zugehörigen Datentyp aus!
      echo "<br>";
      echo $z;
?>
```

Der obige PHP-Code ergibt folgende Ausgabe:

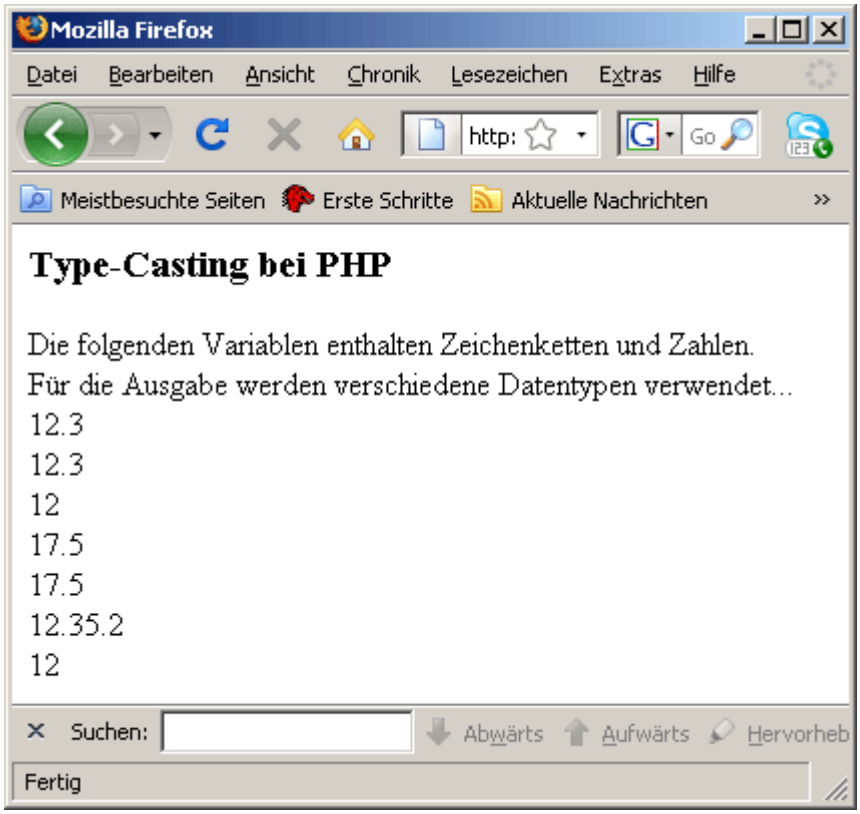
```
10
float(10.5)
10.5
```

Folgende Umwandlungen sind möglich:

- (int), (integer) – nach integer
- (bool), (boolean) – nach boolean
- (float), (double), (real) – nach float
- (string) – nach string
- (array) – nach array
- (object) – Wandlung zum Objekt

Anstatt eine Variable in einen String umzuwandeln, können Sie die Variable auch in doppelte Anführungszeichen einschließen.

Beachten Sie, dass Tabulatoren und Leerzeichen innerhalb der Klammern erlaubt sind. Deshalb sind die folgenden Beispiele identisch:



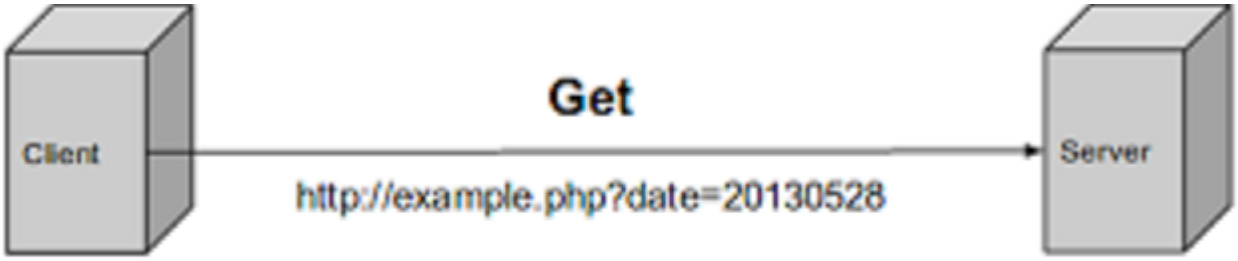
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_06

Last update: **2019/05/16 17:06**

1.4.1) Formulare

Mit PHP kann man **interaktive Webseiten** erstellen, bei denen **Benutzereingaben aus Formularen durch PHP ausgewertet** werden. Die **Darstellung der Formulare** wird mit **HTML-Code** umgesetzt. Die Daten aus dem Formular werden dabei **per HTTP** mit den **Methoden POST oder GET** an das **auswertende PHP-Programm** übermittelt.





Methoden der Datenübertragung

POST

- Angabe von `method=„POST“` im einleitenden HTML-Tag `<form>`
- Formulardaten werden direkt an die Webadresse versendet.
- Daten können nicht in der Adresszeile des Browsers manipuliert werden. Optionen sind nur über das Formular möglich.
- Daten werde nicht im Logfile des Servers gespeichert.
- Formulardaten sind nicht im Verlauf des Browsers sichtbar.
- Längere Texte oder Daten können in Formularen übermittelt werden, da keine Beschränkung der Größe besteht.

GET

- Angabe von `method=„GET“` im einleitenden HTML-Tag `<form>`
- Formulardaten werden in der URL übermittelt und werden dann durch den Server in einer speziellen Umgebungsvariablen zwischengespeichert.

- Daten sind in der Adresszeile des Browsers änderbar, ohne das Formular erneut ausfüllen zu müssen.
- Der Aufruf des Skripts mit Angabe der Daten kann als Favorit abgelegt werden.
- Daten sind auf ca. 2KB beschränkt.

Da bei der Methode GET die Daten über die URL an den Server weitergegeben werden, ist es möglich, die Daten zu manipulieren. Die sicherere und am häufigsten genutzte Methode ist daher die Methode POST.

Formulare auswerten mit PHP

Formulardaten eingeben

Zur Eingabe von Formulardaten, genügt eine HTML-Datei, in der ein Formular integriert ist:

formular_eingabe.html

formular_eingabe.html

```
<html>
<body>
<p>Bitte füllen Sie die nachfolgenden Eingabefelder aus:</p>
<form action="formular_auswertung.php" method="POST">
<pre>
<p>
Vorname:  <input type="Text" name="vorname"><br>
Nachname: <input type="Text" name="nachname"><br>
Wohnort:  <input type="Text" name="ort"><br>
</p>
<input type="Submit" value="Abschicken">
<input type="Reset" value="Zurücksetzen">
</pre>
</form>
</body>
</html>
```

Formulardaten übertragen

Klickt man auf den Button „Abschicken“, wird das in `action` angegebene Skript (`formular_auswertung.php`) gestartet.

Formulardaten auswerten

Die mit der Methode POST übermittelten Daten wird mit dem Befehl `$_POST[„Parameter“]` angesprochen. Der Name des Eingabefeldes im Formular (z.B. `name=„vorname“`) wird automatisch Element der Variablen `$_POST` (z.B. `$_POST[„vorname“]`) Der Eintrag des Eingabefeldes im Formular wird bei der Auswertung zum Wert der Variablen, z.B. `$_POST[„vorname“] = „Max“` .

formular_auswertung.php

formular_auswertung.php

```php
<?php
  echo "<p>Folgende Daten wurden übermittelt:</p>\n";
  echo "Vorname: " .$_POST["vorname"] ."<br>\n";
  echo "Nachname: " .$_POST["nachname"] ."<br>\n";
  echo "Wohnort: " .$_POST["ort"] ."<br>\n";
?>
```

Formular und Auswertung in einer Datei

Das Formular und die Auswertung können in einer Datei zusammengefasst werden. Mit einer if-Abfrage kann überprüft werden, ob der Submit-Button gedrückt wurde oder nicht.

pizzabestellung.php

pizzabestellung.php

```
Pizzabstellung<br>
Pizza Margarita zum günstigen Preis von 5.50 Euro bestellen! <br>
<pre>
<form action="pizzabestellung.php" method="POST">
  Name: <input type="Text" name="name" />
  Lieferadresse: <input type="Text" name="adresse" />
  Anzahl der Pizzen: <input type="Text" name="anzahl" /><br><br>
  <input type="Submit" name="schicken" value="Bestellung
abschicken"><br>
  <input type="reset" value="Zurücksetzen">
</form>
</pre>
<hr>
<?php
    if ($_POST["schicken"]=="Bestellung abschicken")
    {
    echo "<i>Für folgende Bestellung wird gedankt: </i><br><br>";
    echo "<b>Name:</b> ".$_POST["name"]."<br>";
    echo "<b>Lieferadresse:</b> ".$_POST["adresse"]."<br>";
    if ($_POST["anzahl"]<>0)
```

```
        {
    echo "Menge beträgt <b>".$_POST["anzahl"]." Stück</b> zum Preis von
<b>".$_POST["anzahl"]*5.50." Euro</b>";
    }
    }

?>
```

Überprüfen des mitgeschickten Arrays mittels POST

```
print_r($_POST);
var_dump($_POST);
```

`print_r()` Zeigt den Inhalt der Variablen in lesbarer Form an. `var_dump()` geht einen Schritt weiter, es können auch, durch Beistriche getrennt, mehrere Variablen angegeben werden. Die Informationen sind detaillierter. Sie enthalten auch Typ und Länge der Variablen.
`var_dump(Variable 1[, Variable 2,…]);` Das gesamte mittels POST übermittelte Array wird ausgegeben.

Beispiel Newsletter

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_07

Last update: **2019/05/16 17:06**

1.4.2) Aufgabe 2

Erstelle eine kleine Website für eine Apfelbestellung:

- bioladen.html
  - Eingabe von Name, Adresse, Menge (in kg) 1. Sorte Golden Delicious (1,20 pro kg), 2. Sorte: Idared (1,50 pro kg)
- bioladen-auswertung.php
  - Ausgabe von Name, Adresse, bestellter Menge und Gesamtpreis
  - ab insg. 10 kg bestellter Menge: keine Versandkosten, sonst 5,00 Euro.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_08

Last update: **2019/05/16 17:06**

1.5.1) Anführungszeichen

Doppelte oder einfache Anführungszeichen

```
$foo = 'Heute';

echo "$foo ist ein schöner Tag!"; // Ausgabe: Heute ist ein schöner Tag!
echo '$foo ist ein schöner Tag!'; // Ausgabe: $foo ist ein schöner Tag!
```

Wenn ein PHP-Programmierer einen Link aus Variablen erstellt, kann man häufig folgenden Code sehen:

```
echo '<a href="' . $link . '" id="' . $id .'" class="' . $class . '">' .
$linktext . '</a>';
```

oder

```
?>
<a href="<?php echo $link; ?>" id="<?php echo $id; ?>"
   class="<?php echo $class; ?>"<?php echo $linktext; ?></a>
<?php
```

oder

```
echo "<a href=\"$link\" id=\"$id\" class=\"$class\">$linktext</a>";
```

Abgesehen vom persönlichen Stil des Programmierers haben die Code-Beispiele eines gemeinsam: sie sind schlecht zu lesen. „Klar!“, werden jetzt die Programmierer sagen, „In HTML müssen die Attribute in doppelten Anführungszeichen stehen und in PHP muss man eine der oben gezeigten Methoden verwenden“. Sicher? Müssen Attribute wirklich in doppelten Anführungszeichen stehen? Die einfache Antwort ist: Nein! Ein Blick in die Spezifikation bringt Folgendes an den Tag:

[…]Standardmäßig verlangt SGML, dass alle Attributwerte entweder von doppelten Anführungszeichen (ASCII dezimal 34) oder einfachen Anführungszeichen (ASCII dezimal 39) begrenzt werden. Einfache Anführungszeichen können im Attributwert enthalten sein, wenn der Wert durch doppelte Anführungszeichen begrenzt ist und umgekehrt.[…]

Doppelte oder einfache Anführungszeichen. Das macht die Sache doch viel übersichtlicher:

```
echo "<a href='$link' id='$id' class='$class'>$linktext</a>";
```

```
echo 'Micro$oft'; // ergibt: Micro$oft
echo "Micro$oft"; // ergibt: Micro + Inhalt der Variable $oft
echo "Micro\$oft"; // ergibt: Micro$oft

echo 'c:\temp'; // ergibt: c:\temp
```

```
echo "c:\temp"; // ergibt: c: + Tabulator + emp
echo "c:\\temp"; // ergibt: c:\temp
echo 'c:\\temp'; // ergibt: c:\temp

echo 'Kein Hochkomma: \x27'; // ergibt: Kein Hochkomma: \x27
echo " Ein Hochkomma: \x27"; // ergibt: Ein Hochkomma: '
echo ' Ein Hochkomma: \''; // ergibt: Ein Hochkomma: '

echo "<input name='foo' value='$bar'>"; // gültiges HTML
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_09

Last update: **2019/05/16 17:06**

1.6.1) Vergleichsoperatoren

Eine Bedingung ist eine Möglichkeit, den Ablauf eines Skripts durch Entscheidungen zu beeinflussen. In einer Bedingung werden Ausdrücke verglichen. Das Ergebnis kann entweder „Ja“ (TRUE) oder „Nein“ (FALSE) sein.

Operator	Name	Bedeutung
==	Gleichheit	$a == $b ergibt TRUE, wenn $a und $b gleich sind.
===	Identisch	$a === $b ergibt TRUE, wenn $a und $b gleich und vom selben Datentyp sind.
!=	Ungleichheit	$a != $b ergibt TRUE, wenn $a und $b ungleich sind.
<	Kleiner	$a < $b ergibt TRUE, wenn $a kleiner $b ist.
>	Größer	$a > $b ergibt TRUE, wenn $a größer $b ist.
<=	Kleiner gleich	$a <= $b ergibt TRUE, wenn $a kleiner oder gleich $b ist.
>=	Größer gleich	$a >= $b ergibt TRUE, wenn $a größer oder gleich $b ist.

Verknüfung von Bedingungen

Operator	Name	Bedeutung
and bzw. &&	UND	$a and $b ergibt 1, wenn $a und $b 1 sind, ansonsten wird 0 zurückgegeben
or bzw. ||	ODER	$a or $b ergibt 1, wenn mindestens eine der beiden Variablen 1 ist.
xor	ausschließendes ODER	$a xor $b ergibt 1, wenn entweder $a oder $b 1 sind, aber nie beide gleichzeitig
!	NICHT	!$a ergibt die Umkehrung des Wahrheitswertes

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_10

Last update: **2019/05/16 17:06**

1.6.2) Die einfache if-Anweisung

Syntax

```
if (Bedingung)
{
  Anweisungen;
}
```

- Die Bedingung steht in runden Klammern.
- Die Anweisungen stehen in geschwungenen Klammern.
- Liefert die Bedingung TRUE zurück, werden die Anweisungen ausgeführt, ist die Bedingung FALSE, werden die Anweisungen ignoriert
- Wird nur eine Anweisung ausgeführt, können die geschwungenen Klammern entfallen.

Beispiel: if1.php

```
<?php
  $menge = 7;
  if ($menge > 5)
     echo "Sie haben mehr als 5 Kilo bestellt.";
?>
```

- Der Variable $menge wird der Wert 7 zugewiesen. Da $menge größer als 5 ist, wird die Anweisung ausgeführt.

Beispiel: if2.php

```
<?php
  $menge = 7;
  if ($menge > 5)
  {
    echo "Sie haben mehr als 5 Kilo bestellt.<br>\n";
    echo "Der Versand ist deswegen kostenfrei.";
  }
?>
```

Die if-Anweisung mit else Zweig

Syntax

```
if (Bedingung)
{
  Anweisungsblock1;
}
```

```
else
{
  Anweisungsblock2;
}
```

- Falls die Bedingung erfüllt ist, wird `Anweisungsblock1` ausgeführt, sonst `Anweisungsblock2`.

Beispiel: ifelse.php

```
<?php
$menge=4;
if($menge > 5)
{
   echo "Sie haben mehr als 5 Kilo bestellt.<br>\n";
   echo "Der Versand ist deswegen kostenfrei.";
}
else
{
  echo "Sie haben $menge Kilo bestellt.<br>\n";
  echo "Der Versand kostet pauschal 7,00 EUR";
}

?>
```

Verschachtelte if-Anweisungen

Syntax

```
if (Bedingung1)
{
    Anweisungsblock1;
    if (Bedingung2)
    {
        Anweisungsblock2;
    }
    else
    {
        Anweisungsblock3;
    }
}
else
{
   Anweisungsblock4;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_11

Last update: **2019/05/16 17:06**

1.6.3) Fallauswahl mit switch-Anweisung

Wenn eine Variable mit verschiedenen Werten verglichen werden soll, kann die Programmierung mit einer `if`-Anweisung sehr aufwändig sein. Als Alternative dazu kann man in diesem Fall die `switch`-Anweisung einsetzen:

Syntax

```
switch ($variable)
{
   case Wert1:
      Anweisungsblock1;
      break;
   case Wert2:
      Anweisungsblock2;
      break;
   default:
      Anweisungsblock3;
}
```

- Stimmt der Wert der Variable mit einem der angeführten Auswahlwerte überein, dann wird der Anweisungsblock unmittelbar bis zur Anweisung `break` ausgeführt. Die restlichen Blöcke werden nicht ausgeführt.
- Ist kein `break` am Ende eines `case`-Teils angegeben, so werden auch alle nachfolgenden `case`-Blöcke ausgeführt, bis eine `break`-Anweisung erfolgt.
- Stimmt der Wert der Variablen mit keinem der angegebene Werte überein, wird der Anweisungsblock nach der `default`-Anweisung durchgeführt.

Bsp: switch-case.php

```
<?php
$sorte = "Gala";
   echo "Der Preis für 1kg $sorte beträgt <br>\n";
   switch($sorte) //Preiszuordnung abhängig von Sorte
   {
    case "Jonagold":
         echo "1,50 EUR.";
         break;
    case "Delicious":
         echo "1,60 EUR.";
         break;
    case "Gala":
         echo "1,65 EUR.";
         break;
    case "Elstar":
         echo "2,00 EUR.";
         break;
```

```
    default:
        echo "Diese Sorte haben wir leider nicht im Angebot.";
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_12

Last update: **2019/05/16 17:06**

1.6.4) Aufgaben 3-7

Aufgabe 3

Apfelbestellung - Versandkosten

Erweitere die Aufgabe 2 folgendermaßen:

- Wenn mehr als 10 kg Äpfel bestellt werden, ist der Versand kostenfrei, ansonsten werden 7.00 Euro verrechnet.
- Ausgabe:
  - Rückmeldung, ob Versandkosten zu bezahlen sind oder nicht.
  - Gesamtpreis.

Aufgabe 4

Minimum - Maximum

- In einem Formular (zahlen_eingabe.html) sollen drei Zahlen eingegeben werden (mittels drei Input-Feldern)
- Als Rückmeldung (minmax_ausgabe.php) wird das Maximum sowie das Minimum der drei Zahlen ausgegeben.

Aufgabe 5

Kennwortabfrage mit if

Viele Webseiten arbeiten mit einer Anmeldung des Benutzers. Um Zugang zu weiteren Teilen der Webseite zu bekommen, muss sich der Benutzer einem „Loginverfahren“ unterziehen.

- Erstelle hierfür ein Formular (`kennwort.html`), in welchem der Benutzer ein Kennwort eingeben kann. Verwende dazu ein `<input>`-Feld, bei dem das Kennwort bei der Eingabe nicht sichtbar ist.
- Nach Klick auf den Button „Abschicken“ soll eine Rückmeldung (`kennwort_auswertung.php`) erfolgen, ob das Passwort richtig eingegeben wurde.
- Baue das Formular so aus, dass vor dem Passwort auch der Nickname eingegeben werden muss. Eine positive Rückmeldung (z.B. „Zutritt erfolgreich…“) wird nur dann gegeben, wenn Nickname **und** Passwort korrekt eingegeben wurden.

Aufgabe 6

Verschachtelte if-Anweisungen

Baue die Aufgabe 3 folgendermaßen aus:

- Wenn mehr als 20 kg Äpfel bestellt werden, gibt es zur Bestellung zusätzlich ein kleines Präsent.
- Ausgabe:
  - Rückmeldung, ob Versandkosten zu bezahlen sind oder nicht und ob es ein Präsent gibt.
  - Gesamtpreis.

Aufgabe 7

switch-Anweisung

Erweitere die Aufgabe 5 folgendermaßen aus:

- Bei der Bestellung soll die bestellte Apfelart in einem `<input>`-Feld eingegeben werden.

Apfelsorte	Preis
Jonagold	1,50 EUR
Delicious	1,60 EUR
Gala	1,65 EUR
Elstar	2,00 EUR

- Ausgabe
  - Welche Äpfelsorte bestellt wurde (inkl. Preisangabe), bei falscher Eingabe soll eine entsprechende Meldung erfolgen und kein Gesamtpreis berechnet werden.
  - Gesamtpreis je nach bestellter Sorte.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_13

Last update: **2019/05/16 17:06**

1.7.1) for-Schleife

In der for-Schleife wird genau angegeben, wie oft die Schlaufe durchlaufen werden soll.

Syntax

```
for (Initialisierung; Bedingung; Operation)
{
   Anweisungsblock;
}
```

Beispiel: Ausgabe der ersten n ungeraden Zahlen und die Summe dieser Folge

```
<?php
     echo "<h2>Ausgabe der ersten n ungeraden Zahlen und die Summe dieser
Folge</h2>\n";
     $n=10;
     $summe=0;
     for ($i=1;$i<=$n;$i++)
     {
        $ung=$i*2-1;
        $summe=$summe+$ung;
        echo $i.". ungerade Zahl: ".$ung."<br>\n";
     }
     echo "Die Summe der ersten $n ungeraden Zahlen beträgt: ".$summe;
?>
```

Beispiele für Operatoren in for-Schleifen

Bedingung in der for-Schleife	Werte von $i
for ($i=1; $i<=5, $i++)	1, 2, 3, 4, 5
for ($i=1; $i<5, $i++)	1, 2, 3, 4
for ($i=15; $i>=10, $i--)	15, 14, 13, 12, 11, 10
for ($i=15; $i>10, $i--)	15, 14, 13, 12, 11
for ($i=0; $i<=100, $i=$i+10)	10, 20, 30, 40, 50, 60, 70, 80, 90, 100
for ($i=1; $i<=12, $i=$i+1.2)	1, 2.4, 3.6, 4.8, 7.2, 8.4, 9.6, 10.8, 12

Geschachtelte for-Schleifen

Mit geschachtelten for-Schleifen können zweidimensionale Strukturen abgearbeitet werden.

Beispiel: Erstellen einer Tabelle mit 3 Zeilen und 4 Spalten mittels zweier for-Schleifen

```php
<?php
   echo "<table border='1'>\n";
   for ($i=1;$i<=3;$i++)
   {
       echo "<tr>\n";
       for ($j=1;$j<=4;$j++)
       {
           echo "<td> $i. Zeile, $j. Spalte </td>\n";
       }
       echo "</tr>\n";
   }
   echo "</table>\n";
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_14

Last update: **2019/05/16 17:06**

1.7.4) Foreach-Schleife

Die foreach-Schleife ermöglicht es, auf einfache Weise ein Array zu durchlaufen. foreach funktioniert nur in Verbindung mit Arrays. Wenn Sie versuchen, foreach mit einer Variablen eines anderen Datentyps oder einer nicht initialisierten Variablen zu benutzen, gibt PHP einen Fehler aus.

Definition

```
foreach (array_expression as $value) {
  Anweisung/en;
}
```

Mithilfe von foreach wird das Durchlaufen eines Arrays wesentlich vereinfacht. Gegenüber der while-Schleife mit list und each ist die foreach-Schleife syntaktisch deutlich im Vorteil.

Beispiel while

```
<?php
$zahlen = array (10, 20, 30, 40);
while (list(, $value) = each ($zahlen)) {
  echo "Wert: $value<br>\n";
}
?>
```

Beispiel foreach

```
<?php
foreach ($zahlen as $value) {
  echo "Wert: $value<br>\n";
}
?>
```

Das Array wird von Anfang bis Ende durchlaufen und bei jedem Schleifendurchlauf wird das aktuelle Element der Variablen array_expression zugewiesen. Jedes Arrayelement kann wiederum ein Array sein, dann könnte mit einer verschachtelten foreach-Schleife auch dieses Array ausgewertet werden.

Beispiel – mit assoziativem Array

```
$personen = array("Matthias","Caroline","Gülten");

foreach ($personen as $person) {
  echo "Name: $person<br>\n";
}
```

```
// Ausgabe:
// Name: Matthias
// Name: Caroline
// Name: Gülten
```

Typischerweise ist es erlaubt, einzelne Elemente eines solchen Arrays mit unterschiedlichen Datentypen zu belegen. Das können auch weitere Arrays sein.

Erweiterte Syntax

Sollten Sie Arrays mit Schlüssel-Wert-Paaren bauen, kann foreach mit einer erweiterten Syntax diese Paare direkt auslesen. Die grundlegende Syntax lautet:

Definition

```
foreach (array as $key => $value) {
  Anweisung/en;
}
```

Hier wird der Operator ⇒ eingesetzt, der schon bei der Konstruktion des Arrays verwendet wurde.

Beispiel

```
<?php
$person = array("Vorname" => "Caroline",
      "Nachname" => "Kannengiesser",
      "Alter" => 25,
      "Ort" => Berlin);
foreach ($person as $key => $val) {
  echo "Feld $key hat den Wert: $val<br>";
}
?>
```

Ausgabe

```
Feld Vorname hat den Wert: Caroline
Feld Nachname hat den Wert: Kannengiesser
Feld Alter hat den Wert: 25
Feld Ort hat den Wert: Berlin
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_14_1

Last update: **2019/05/16 17:06**

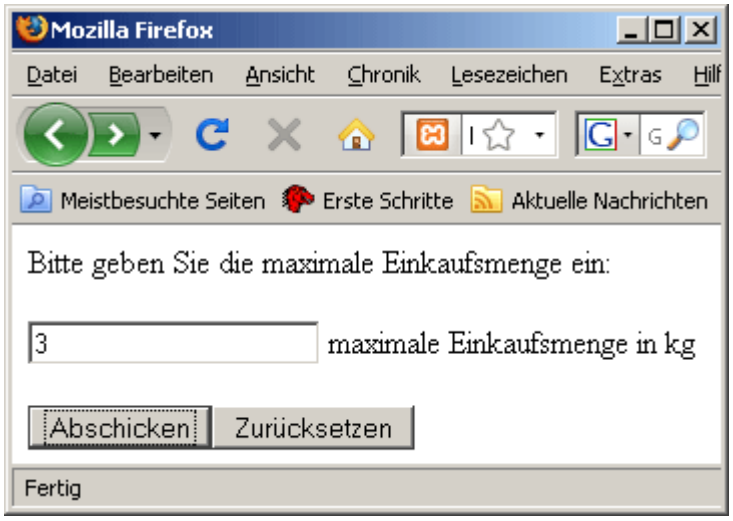
1.7.2) while-Schleife

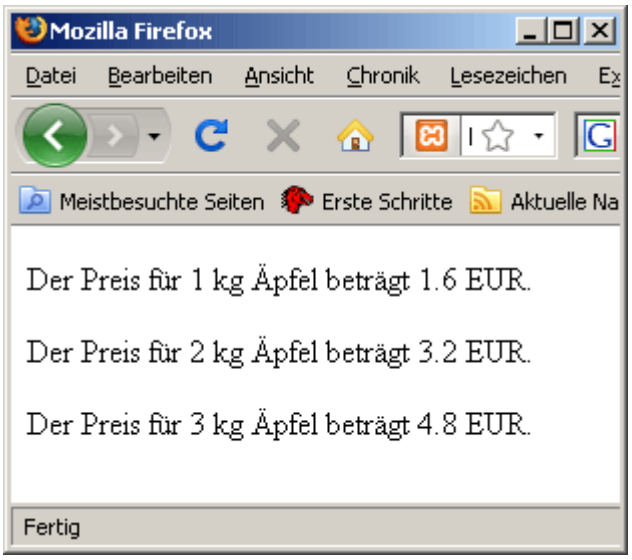
Syntax

```
while(Bedingung)
{
   Anweisungsblock;
}
```

Bsp: while.php

```
<?php
$menge = 1;
$preis = 1.6;
while ($menge <= $_POST["maxmenge"])
{
   $gesamtpreis = $preis * $menge;
   echo "<p>Der Preis für $menge kg Äpfel beträgt $gesamtpreis EUR.</p>\n";
   $menge++;
}
?>
```





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_15

Last update: **2019/05/16 17:06**

1.7.3) do-while-Schleife

Die do-while Schleife unterscheidet sich von der while-Schleife, dadurch, dass die Abbruchbedingung nicht am Anfang, sondern am Ende jedes Schleifendurchlaufs geprüft wird. Die Anweisung wird somit zumindest einmal durchlaufen.

Syntax

```
do
{
   Anweisungsblock;
}
while (Bedingung);
```

Bsp: dowhile.php

```
<?php
   $i=1;
   do
   {
     echo "<br>".$i*$i;
     $i++;
   }
   while ($i<=10);
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_16

Last update: **2019/05/16 17:06**

1.7.5) Aufgabe 8 bis 11

Aufgabe 8

Erstelle ein PHP-Skript, welches eine Folge von n Zahlen erzeugt, in der jede Zahl den doppelten Wert der vorhergehenden hat.

Aufgabe 9

Erstelle ein PHP-Skript, das das „Ein-mal-eins“ für die Faktoren zwischen 11 und 20 ausgibt („großes Ein-mal-eins“). Die einzelnen Rechnungen sollen dabei in einer Tabelle aufscheinen.

Aufgabe 10

Schreibe ein PHP-Skript, das die Fibonacci-Zahlen bis zu einer eingegebenen Höchstgrenze erzeugt und ausgibt.

Eine Fibonacci-Zahl wird als Summe der beiden vorhergehenden Fibonacci-Zahlen gebildet. Die erste und die zweite Fibonacci-Zahl sind gleich 1.

Beispiel: 1, 1, 2, 3, 5, 8, 13, ...

Aufgabe 11

Teiler einer Zahl

Erstelle ein HTML- und PHP-Skript, welches die Teiler von Zahlen ermittelt. Weiters soll zusätzlich bestimmt werden, ob es sich um eine Primzahl (Anzahl der Teiler: 2) handelt

a) Version 1: Es werden die Teiler der eingegebenen Zahl ermittelt:

Ausgabe z.B.: 27 hat die Teiler 1, 3, 9, 27

b) Version 2: Es werden die Teiler sämtlicher Zahlen bis zur eingegebenen Zahl ermittelt:

Ausgabe z.B.:

...

23 hat die Teiler 1, 23 (Primzahl)

24 hat die Teiler 1, 2, 3, 4, 6, 8, 12, 24

25 hat die Teiler 1, 5, 25

26 hat die Teiler 1, 2, 13, 26

27 hat die Teiler 1, 3, 9, 27

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_17

Last update: **2019/05/16 17:06**

1.8.1) Was sind Felder?

Felder, auch Arrays oder Feldvariablen genannt, sind in der Lage, mehrere Variablen unter einem gemeinsamen Bezeichner zu speichern. Beispielsweise erhalten Sie in einer Berechnung 20 Ergebnisse. Die Ausgabe der 20 Werte soll erst erfolgen, wenn alle Berechnungen abgeschlossen wurden. Sie könnten 20 Variablen (`$ergebnis1, $ergebnis2, $ergebnis3, $ergebnis4`, ...) definieren, um alle Ergebnisse der Berechnungen zu speichern. Einfacher und schneller kann die Aufgabe gelöst werden, wenn die Ergebnisse der Berechnung in einem Feld abgelegt werden. Der Zugriff auf die einzelnen Ergebnisse erfolgt jeweils über ihre Position innerhalb des Feldes.



Felder in PHP

Felder unterscheiden sich in der Art, wie auf die Werte innerhalb des Feldes zugegriffen werden kann, und in ihrem Aufbau.

Auf Werte in Feldern zugreifen:

- Im numerisch indizierten Feld werden die einzelnen Werte (Value) innerhalb des Feldes über eine laufende Nummer, auch **Index oder Schlüssel** genannt, angesprochen. Der **Schlüssel** ist eine **laufende Nummer**, die automatisiert vergeben wird und bei **Null beginnt**.

```php
$myArray = array();

$myArray[0] = "Susi";
$myArray[1] = "Maxi";

//oder

$myArray = array("Susi","Maxi");
```

Schlüssel & Inhalt des Arrays $myArray

key (Ort)	value (PLZ)
0	Susi
1	Maxi

- Im assoziativen Feld werden die einzelnen Werte (Value) innerhalb des Feldes über einen eindeutigen Schlüssel, Key genannt, angesprochen.

```
$myArray = array();
$myArray['key'] = "value";


//oder

$staedte = array('Guetersloh' => 98000,
            'Bielefeld' => 326715);

echo $staedte["Guetersloh"]; //gibt 98000 aus
```

Schlüssel & Inhalt des Arrays $staedte

key (Ort)	value (PLZ)
Guetersloh	98000
Bielefeld	326715

Aufbau von Feldern

Felder können ein- oder mehrdimensional sein:

- In einem eindimensionalen Feld können Sie beispielsweise eine Liste von Städten darstellen.
- Wenn Sie z. B. eine Tabelle mit verschiedenen Feldern abbilden möchten, können Sie dies mit einem zweidimensionalen Feld tun.
- Sie haben auch die Möglichkeit, Felder mit mehr als zwei Dimensionen zu nutzen.

Eigenschaften von Feldern

- In PHP müssen die Elemente von Feldern nicht vom selben Datentyp sein.
- Felder-Indizes bei numerisch indizierten Feldern beginnen mit 0. Das erste Element hat dementsprechend den Index 0, das zweite Element hat den Index 1, das dritte Element hat den Index 2 und so weiter. Der Index eines Feldes mit n Elementen reicht somit von bis $n-1$. Der Index eines Feldes mit vier Elementen reicht somit von 0 - 3.
- Beim Anlegen eines Feldes muss nicht angegeben werden, wie viele Elemente in diesem Feld gespeichert werden. Sie können auch nach der Erstellung dem Feld zusätzliche Elemente hinzufügen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_18

Last update: **2019/05/16 17:06**

1.8.2) Indizierte Felder

Indizierte Felder erstellen

Ein indiziertes Feld anlegen

PHP bietet Ihnen eine einfache und schnelle Möglichkeit, Felder mit verschiedenen Werten zu füllen. Dies wird über die array-Anweisung realisiert.

Syntax und Bedeutung der array() -Anweisung

```
$Feldvariable = array(Wertl,Wert2,Wert3,...) ;
```

- Die `array()`-Anweisung beginnt mit dem reservierten Wort array.
- Der Variablen `$Feldvariable` werden die einzelnen Werte übergeben, indem sie durch Kommata voneinander getrennt angegeben werden.
- Die Indizierung der einzelnen Elemente erfolgt automatisch in der Reihenfolge der Angabe.

Beispiel

```
$staedte = array("Frankfurt","Berlin","Bern");
```

Es befinden sich drei Werte in dem Feld `$staedte`; Frankfurt, Berlin und Bern. Die Reihenfolge, in der die Werte zugewiesen worden sind, bestimmt die Indizes der Werte.

Index	0	1	2
Wert	Frankfurt	Berlin	Bern

Auf indizierte Felder zugreifen

Um auf einzelne Elemente in einem Feld zugreifen zu können, muss ihnen eine eindeutige Kennung zugewiesen werden. Bei numerisch indizierten Feldern entspricht der Kennung der automatisch erzeugte fortlaufende Index.

Auf einen bestimmten Wert eines Feldes greifen Sie zu, indem Sie zusätzlich zum Namen der Feldvariablen in den eckigen Klammern den Indexwert angeben.

```
$staedte[Index];
```

Beispiel

```
$stadt = $staedte[2] ;
```

Der Variablen `$stadt` wird der dritte Wert des Feldes `$staedte`, die Zeichenkette Bern, zugewiesen.

Mit indizierten Feldern arbeiten

Indizierte Felder ändern

Um einen Wert innerhalb einer Feldvariablen zu ändern, geben Sie bei der Wertzuweisung bei der Feldvariablen den entsprechenden Index des zu ändernden Wertes an und weisen Sie den neuen Wert zu.

```
$staedte[l] = "München";
```

In dem bisherigen Beispiel würde somit die Zeichenkette Berlin durch München ersetzt werden.

Indizierte Felder erweitern

Soll ein Feld um einen weiteren Wert am Ende des Feldes ergänzt werden, erzeugt PHP den entsprechenden Index automatisch.

Syntax und Bedeutung

```
$Feldvariable[] = Wert Zuweisung;
```

- Um einem Feld einen weiteren Wert (oder mehrere Werte) hinzuzufügen, brauchen Sie bei der Wertzuweisung keinen Index anzugeben. Das Feld wird am Ende um den Wert (bzw. die Werte) ergänzt und der Index automatisch erhöht.
- Sollte das Feld noch nicht bestehen, erstellt PHP automatisch ein Feld, sobald Sie der Variablen einen Wert zugewiesen haben.

Beispiel

```
$staedte = array("Frankfurt","Berlin","Bern");
$staedte[] = "Graz";
$staedte[] = "Rom";
```

ergibt:

Index	0	1	2	3	4
Wert	Frankfurt	Berlin	Bern	Graz	Rom

Anzahl der Feldelemente ermitteln

Der Befehl `COUNT($bereich);` liefert die Anzahl der Feldelemente des Arrays `$bereich`.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_19

Last update: **2019/05/16 17:06**

1.8.3) Assoziative Felder

Assoziative Felder erstellen

Ein assoziatives Feld anlegen

Eine weitere Möglichkeit, Feldvariablen anzulegen, sind die assoziativen Felder. Bei ihnen werden für den Zugriff keine fortlaufenden Indizes benutzt, sondern Schlüssel, über deren Wert auf die einzelnen Werte zugegriffen werden kann, so genannte Schlüsselwörter.

Syntax der `array()`-Anweisung bei assoziativen Feldern

Auch die assoziativen Felder lassen sich über die `array()`-Funktion füllen.

```
$Feldvariable = array (Schlüssel1 => Wertl,Schlüssel2 => Wert2, ...) ;
```

Beispiel:

```
$hauptstaedte = array("Schweiz" => "Bern", "Frankreich" => "Paris");
```

- Die array ()-Anweisung beginnt mit dem reservierten Wort array.
- Der Variablen werden die einzelnen Wertpaare, bestehend aus Schlüsselwort (Key) und Wert (Value), übergeben, indem sie durch Kommata voneinander getrennt angegeben werden.
- Die Indizierung der einzelnen Werte erfolgt über die Angabe eines Schlüssels in Form einer beliebigen, eindeutigen Zeichenkette oder eindeutiger Integerwerte. Die Zuweisung erfolgt über ⇒.

Auf assoziative Felder zugreifen

Beim Zugriff auf einen Wert des Feldes kann der Schlüsselwert direkt angegeben oder als Variable übergeben werden:

	Ergebnis
`$stadt = $hauptstaedte["Schweiz"];`	Variable `$stadt` hat den Wert „Bern“

oder:

	Ergebnis
`$k = "Schweiz"; $stadt = $hauptstaedte[$k];`	Variable `$stadt` hat den Wert „Bern“

Beispiel: arr_assoz.php Hier wurde die Zuweisung der Städtenamen mit der `array()`-Anweisung realisiert.

```
<html>
<body>
<?php
$hauptstaedte = array("Schweiz" => "Bern",
                      "Frankreich" => "Paris",
                      "Deutschland" => "Berlin",
                      "Österreich" => "Wien");
$k = "Österreich";
echo "<p>Hauptstadt von $k: " . $hauptstaedte[$k] ."</p>";
echo "Die Hauptstadt von $k ist " .$hauptstaedte["Österreich"];
?>
</body>
</html>
```

- Das assoziative Feld wird über die `array()`-Funktion und die Zuweisung der Schlüssel sowie Feldinhalte angelegt.
- Der Variablen $k wird die Zeichenkette Österreich zugewiesen.
- Mithilfe der Variablen $k wird das entsprechende Element aus dem Feld ausgelesen und angezeigt.
- Hier wird das Element durch die direkte Angabe des Schlüssels herausgefiltert. Die Bezeichnung des Schlüssels wird als Zeichenkette in den eckigen Klammern angegeben.

Mit assoziativen Feldern arbeiten

Daten an Variablen übergeben

Mit einer `foreach()`-Schleife sind Sie in der Lage, die Werte der Feldelemente einzeln auszulesen. Dabei wird jedes Element in einer neuen Variable zwischengespeichert.

```
foreach($Feld as $wert) Anweisung;
foreach($Feld as $index => $wert) Anweisung;
```

- Die Funktion `foreach()` erwartet die Angabe des assoziativen Feldes, dessen Elemente durchlaufen werden sollen.
- In der ersten Variante werden mithilfe von `foreach` bei jedem Schleifendurchlauf der Variablen ($wert) nacheinander die Werte der Arrayelemente zugewiesen.
- Mithilfe der zweiten Variante können Sie zusätzlich den Index des Arrayelements auslesen und der Variablen $index zuweisen, was beispielsweise bei assoziativen Feldern hilfreich ist.

Beispiel: foreach.php
Verschiedene Länder der Welt und deren Hauptstädte werden in einem assoziativen Feld gespeichert. Zur Ausgabe sollen das Land (= Schlüssel) und die dazugehörige Hauptstadt (= Wert) in einer Tabelle angezeigt werden:

```
<?php

 $hauptstaedte = array("Schweiz" => "Bern",
```

```
                        "Frankreich" => "Paris",
                        "Deutschland" => "Berlin");
 $hauptstaedte["Polen"] = "Warschau";
 $hauptstaedte["Italien"] = "Rom";
 $hauptstaedte["Österreich"] = "Wien";

 echo "<table border>";
 echo "<tr><td><b>Land</b></td>" ;
 echo "<td><b>Hauptstadt</b></td></tr>" ;
 foreach($hauptstaedte as $land=>$stadt)
 {echo "<tr><td>$land</td><td align='left'>$stadt</td></tr>" ;
 }
 echo "</table>";

?>
```

- Das assoziative Feld $hauptstaedte wird über die `array`-Anweisung gefüllt.
- Dem assoziativen Feld $hauptstaedte werden weitere Elemente hinzugefügt.
- Der Tabellenkopf wird erstellt.
- Mithilfe einer foreach-Schleife können Sie auf jedes Element des Feldes zugreifen. Die Schleife wird so lange durchgeführt, bis alle Elemente des Feldes angesprochen wurden. In jedem Schleifendurchlauf wird der Variablen `$land` der jeweilige Schlüsselwert und der Variablen `$stadt` der Wert des Feldes an der durch das Schlüsselwort festgelegten Position zugeordnet.
- Die Werte werden in die Tabelle eintragen und über den Ausgabebefehl echo am Bildschirm ausgegeben.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_20

Last update: **2019/05/16 17:06**

1.8.4) Mehrdimensionale Felder

Mehrdimensionale indizierte Felder erstellen

Grundlagen zu mehrdimensionalen Feldern

Im folgenden Beispiel sehen Sie mehrere Angaben (Vorname, Nationalität und Alter), die jeweils einen Eintrag in einer Feldvariablen bilden. Damit ist dieses Feld ein mehrdimensionales Feld. Bei jeder der Angaben sind Mehrfachwerte möglich. Mehrdimensionale Felder sind verschachtelte Felder, man spricht hier von äußeren und inneren Feldern.

Mehrdimensionale Felder können sowohl numerisch indiziert als auch assoziativ sein. Eine Mischform aus numerisch indizierten und assoziativen Feldern ist ebenfalls möglich. Im Beispiel bietet sich durch mögliche Mehrfachwerte z. B. ein indiziertes Feld an, das jeden Satz an Angaben mit einem eindeutigen numerischen Index versieht.

Index 1	Index 2		
	0	1	2
0	Oliver	spanisch	37 Jahre
1	Maria	deutsch	23 Jahre
2	Oliver	englisch	46 Jahre

In dieser Tabelle wird bereits die Arbeitsweise mehrdimensionaler Felder ersichtlich. Um beispielsweise an die Informationen von Maria zu gelangen, suchen Sie die Zeile mit der Angabe Maria und lesen die einzelnen Werte dieser Zeile aus: `Maria, deutsch, 23 Jahre`.

Mit mehrdimensionalen indizierten Feldern arbeiten

Ein mehrdimensionales indiziertes Feld hat statt *eines* Indexes - in Abhängigkeit von der Anzahl der Verschachtelungen - *mehrere* Indizes.

Syntax eines mehrdimensionalen indizierten Feldes

Die bereits erwähnten Informationen sollen im Folgenden als mehrdimensionales Feld realisiert werden.

Auf eine bestimmte Angabe in einem mehrdimensionalen indizierten Feld zugreifen

Ausschnitt aus der Beispieldatei„mehrdimensional.php“:

```
$person = array(array("Oliver", "spanisch", "37    Jahre"),
           array("Maria",    "deutsch",  "23 Jahre"),
```

```
            array("Oliver",    "englisch", "46    Jahre"));
$vornamel  = $person[l][0]    ;
$nationalitaetl    - $person[l][1]    ;
$alterl    = $person[l][2]    ;
```

Ergebnis:

```
$vorname1       = "Maria";
$nationalitaet1 = "deutsch";
$alter1         = "23 Jahre".
```

Ein mehrdimensionales indiziertes Feld erweitern

```
$person[3][0] = "Johanna";
$person[3][1] = "schwedisch";
$person[3][2] = "19 Jahre";
```

oder alternativ:

```
$person[] = array("Johanna", "schwedisch", "19 Jahre");
```

Mit mehrdimensionalen assoziativen Feldern arbeiten

Syntax eines mehrdimensionalen assoziativen Feldes

Die assoziativen Felder besitzen zum Ansprechen der einzelnen Elemente einen sogenannten Schlüssel. Um ein mehrdimensionales Feld anzulegen, wird jedem Schlüssel ein weiteres Feld mit Schlüssel-Wert-Paaren übergeben (Sonderfall zweidimensional). Verwenden Sie aus Gründen der Übersichtlichkeit und Nachvollziehbarkeit der Programmierung nicht mehr als drei Dimensionen bzw. Verschachtelungen.

Beispiel: mehrdimensional.php

```
$land = array("Spanien"  => array("Hauptstadt"    =>    "Madrid",
                                  "Sprache"  =>    "Spanisch",
                                  "Waehrung" =>    "Euro",
                                  "Flaeche"  =>    "504645 qkm"),
              "England"  => array("Hauptstadt"   =>    "London",
                                  "Sprache"  =>    "Englisch",
                                  "Waehrung" =>    "Pfund Sterling",
                                  "Flaeche"  =>    "130395 qkm"),
              "Portugal" => array("Hauptstadt"   =>    "Lissabon",
                                  "Sprache"  =>    "Portugiesisch",
                                  "Waehrung" =>    "Euro",
                                  "Flaeche"  =>    "92345 qkm"));
```

Hiermit ist jeder Satz an Eintragungen mit entsprechenden Länderinformationen über einen Schlüssel mit aussagekräftiger Bezeichnung - `Spanien, England` oder `Portugal` - direkt ansprechbar.

Auf eine bestimmte Angabe in einem mehrdimensionalen assoziativen Feld zugreifen

Ausschnitt aus Beispieldatei„ mehrdimensional.php“

```
$hauptstadt2  = $land["Portugal"]["Hauptstadt"];
$sprache2     = $land["Portugal"]["Sprache"];
$waehrung2    = $land["Portugal"]["Waehrung"];
$flaeche2     = $land["Portugal"]["Flaeche"];
```

Ergebnis:

```
$hauptstadt2  = "Lissabon";
$sprache2     = "Portugiesisch";
$waehrung2    = "Euro";
$flaeche2     = "92345 qkm";
```

Ein mehrdimensionales assoziatives Feld erweitern

```
$land["Ungarn"]["Hauptstadt"]    = "Budapest";
$land["Ungarn"]["Sprache"]   = "Ungarisch";
$land["Ungarn"]["Waehrung"]  = "Forint";
$land["Ungarn"]["Flaeche"]   = "93036 qkm";
```

oder alternativ:

```
$land["Ungarn"]   = array("Hauptstadt" =>    "Budapest",
                          "Sprache"    => "Ungarisch",
                          "Waehrung    => "Forint",
                          "Flaeche"    => "93036 qkm");
```

Daten aus mehrdimensionalen Feldern extrahieren

Bei der Arbeit mit mehrdimensionalen Feldern kommt es häufig vor, dass Sie die kompletten Daten eines inneren Feldes benötigen. Mit einer `foreach`-Schleife können Sie alle Werte einer Feldvariablen auslesen. Durch die Verwendung einer `foreach`-Schleife müssen Sie keinen Index oder Schlüssel kennen. Bei jedem Schleifendurchlauf wird das aktuelle Element in einer von Ihnen angegebenen Variablen zwischengespeichert. Hierbei handelt es sich auch um eine Feldvariable mit Angaben zum aktuellen „Datensatz“.

Syntax:

```
foreach($feld as $wert);
```

Um die einzelnen Elemente auszulesen und jeden Wert verschiedenen Variablen zu übergeben, verwenden Sie innerhalb der Schleife die `list()`-Funktion.

Syntax

```
list($variable1, $variable2, ...) = $wert;
```

- Der Funktion list () wird das entsprechende Element übergeben, das in die angegebenen Variablen aufgeteilt werden soll.
- Die Variablen werden nacheinander, durch Kommata voneinander getrennt, in den runden Klammern der Funktion list () angegeben.
- Die Anzahl und die Position der Variablen richten sich nach der Anzahl der Elemente im Feld.
- Bei der an dieser Stelle verwendeten Variablen $wert handelt es sich um die Variable, die als letztes Argument in der foreach () -Schleife definiert wurde.

Beispiel: array_zdim_list.php

Verschiedene Angaben zu Ländern werden in einem mehrdimensionalen Feld gespeichert. Zur Ausgabe sollen der Name des Landes, die Hauptstadt, die Sprache und die Landeswährung in separaten Variablen abgelegt werden.

```
<body>
<table width="500">
<tr>
<td width="125"><b>Land</b></td>
<td width="125"><b>Hauptstadt</b></td>
<td width="125"><b>Sprache</b></td>
<td width="125"><b>W&auml;hrung</b></td>
</tr>
    <?php
      $staedte = array(
①         "Japan"         => array("Tokio", "Japanisch", "Yen"),
          "Niederlande"   => array("Amsterdam", "Niederländisch", "Euro"),
          "Polen"         => array("Warschau", "Polnisch", "Z&#322;oty"),
          "Indien"        => array("Neu Delhi", "Indisch", "Rupie"),
          "Island"        => array("Reykjavik", "Isländisch", "Krone"),
          "Italien"       => array("Rom", "Italienisch", "Euro"),
          "Frankreich"    => array("Paris", "Französisch", "Euro"),
          "Spanien"       => array("Madrid", "Spanisch", "Euro"),
          "England"       => array("London", "Englisch", "Pfund Sterling")
          );
```

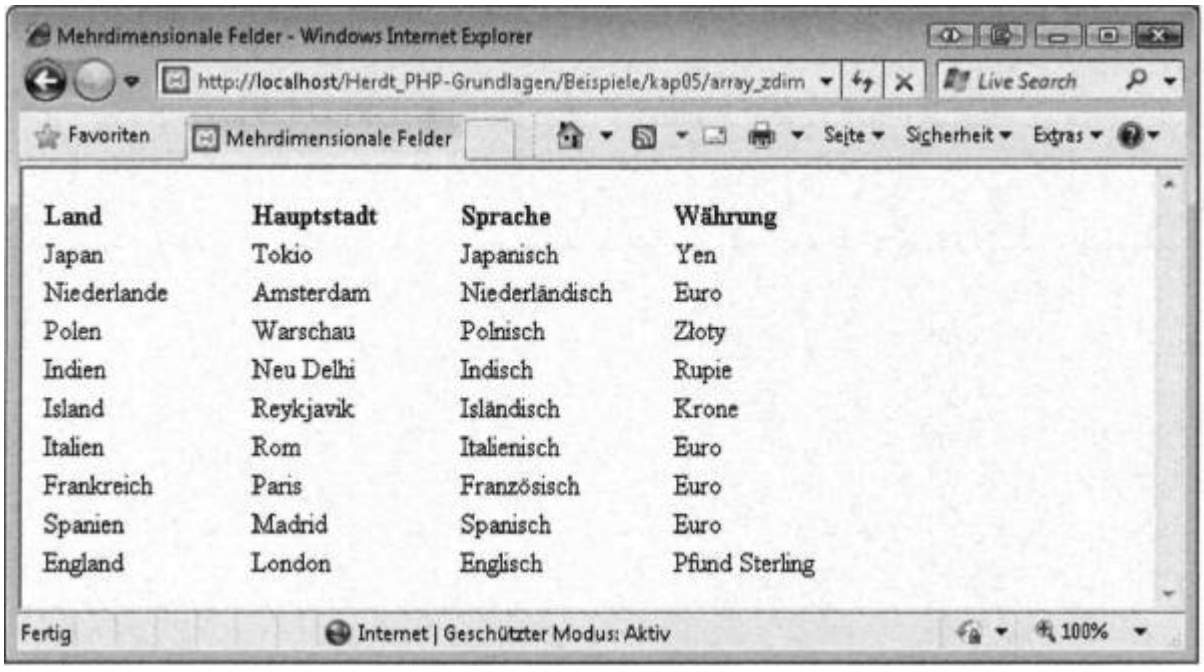
```
②       foreach($staedte as $key=>$ausgabe)
            {
③           list($hauptstadt, $sprache, $waehrung) = $ausgabe;
④           echo "<tr><td>" .$key ."</td>";
            echo "<td>" .$hauptstadt ."</td>";
            echo "<td>" .$sprache ."</td>";
            echo "<td>" .$waehrung ."</td></tr>";
            }
    ?>
</table>
</body>
```

(1) Das assoziative Feld `$staedte` wird neu definiert und mit Werten gefüllt. Die Daten enthalten Informationen zur Hauptstadt eines Landes, zur Sprache und zur im Land verwendeten Währung. Der Name des Landes selbst wird als eindeutiger Schlüssel verwendet.

(2) Mithilfe einer `foreach`-Schleife können Sie auf jedes Element des Feldes zugreifen, indem die Schleife so oft durchgeführt wird, bis auch das letzte Element angesprochen wurde. Da es sich in diesem Beispiel um mehrere Werte pro Element handelt, müssen die zusammenhängenden Werte eines Landes in dem neuen Feld `$ausgabe` zwischengespeichert werden. Im ersten Schleifendurchlauf enthält die Variable `$ausgabe` nur die Werte des zuerst definierten Landes `Japan (Tokio, Japanisch, Yen)`, im zweiten Schleifendurchlauf die Werte der `Niederlande` etc.

(3) Über den Befehl `list()` werden die Daten zum gerade abgerufenen Land (Hauptstadt, Sprache und Währung) in die entsprechenden Variablen `$hauptstadt, $sprache` und `$waehrung` aufgeteilt.

(4) Über den Ausgabebefehl echo werden die Werte der Variablen am Bildschirm ausgegeben. Bei der Variablen `$key` handelt es sich um den Schlüssel (Landesnamen), der in (2) definiert wurde.



Beispiel aus SÜ

```
<?php
// Nicht assoziatives Array
$matrix = array (
            array(1,2,3,4),
            array(5,6,7,8),
            array(9,10,11,12),
            array(13,14,15,16),
            array(17,18,19,20)
          );
// Ausgabe als Zeilen und Spalten
for ($i=0;$i<count($matrix);$i++){
    for ($j=0;$j<count($matrix[$i]);$j++) {echo $matrix[$i][$j].' ';}
    echo '<br>';
}

// Assoziatives Array
$user = array(
          'person'=>array('name'=>'Thomas', 'nickname'=>'Baumi'),
          'lang'
=>array('deutsch','mostviertlerisch','english','spanisch','französisch'),
          'kontakt'=>array('email'=>'thomas.baumgartner@bgamstetten.ac.at',
'web' =>'www.baumi.net')
        );

// Ausgabe
foreach ($user as $eigenschaft => $arrayinhalt) {
  echo $eigenschaft.':';
  foreach ($arrayinhalt as $index =>$wert)
     echo ' '.$index.' ist '.$wert.' | ';
  echo '<br>';
}
?>
```

Verwenden des passenden Feldtyps

Der passende Feldtyp hängt von den Daten ab, die Sie in der Feldvariablen speichern wollen. Die folgende Tabelle soll Ihnen bei der Auswahl des Feldvariablentyps helfen:

Daten	Anzahl an Informationen pro Eintrag	Bevorzugter Feldtyp
Einfache Liste (z. B. Namen, Länder, Automarken etc.)	1	✔ eindimensional/indiziert
Wertepaare (z. B. Länder - Hauptstädte, Artikel - Preis etc.)	2	✔ eindimensional/assoziativ, sofern ein Wert eindeutig ist (= Schlüssel), sonst ✔ mehrdimensional/indiziert oder assoziativ
Mehrere evtl. mehrfach verschachtelte Werte (z. B. Mitarbeiter und ihre Stammdaten, Artikelkategorien - Artikel - Artikeldetails etc.)	> 2	✔ mehrdimensional/indiziert oder assoziativ

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_21

Last update: **2019/05/16 17:06**

1.8.5) Weitere Informationen zu Feldern in PHP

Felder begegnen uns in PHP nicht nur, wenn wir selbst Feldvariablen definieren. PHP arbeitet intern automatisch mit Feldern, und zwar bei folgenden Aktionen:

- Die in einem HTML-Formular übermittelten Daten speichert PHP in einer Feldvariablen.
- Wenn man mit Sessions arbeitet, werden die zur Session gehörigen Daten automatisch in einem Feld abgelegt.
- Bei Abfragen aus Datenbanken befinden sich die Ergebnisse ebenfalls üblicherweise in einer Feldvariablen.

Weitere Feld-Funktionen

PHP kennt annähernd 100 Funktionen für den Umgang mit Feldern. Für nahezu jede Fragestellung liefert PHP eine passende Funktion. Folgende Aufgabengebiete werden abgedeckt:

- Auslesen
- Auswerten (u.a. Dopplungen finden)
- Sortieren
- Suchen
- Teilen und Zusammensetzen
- Verändern (u.a. Füllen, Hinzufügen, Löschen)

Eine Auswahl zusätzlicher wichtiger Feldfunktionen wird kurz in der nachfolgenden Tabelle vorgestellt:

Feldfunktion	**Beschreibung**
`array_flip($feld)`	Im Feld werden Indizes mit Werten vertauscht.
`array_key_exists(wert,$feld)`	Prüfung, ob ein Schlüssel in einem Feld vorhanden ist.
`array_keys($feld)`	Liefert die Indizes des angegebenen Feldes zurück.
`array_merge($feldl,$feld2…)`	Fügt die Elemente mehrerer Felder zu einem Feld zusammen.
`array_push($feld,werte)`	Das Feld wird um den oder die angegebenen Werte am Ende des Feldes erweitert.
`array_search(wert,$feld)`	Das Feld wird nach dem angegebenen Wert durchsucht.
`array_sum($feld)`	Addiert die Werte des Feldes und liefert das Ergebnis zurück.
`array_unique($feld)`	Es wird ein neues Feld erstellt, aus dem doppelte Werte des angegebenen Feldes gelöscht wurden.
`array_values($feld)`	Alle Werte des Feldes werden zurückgeliefert.
`count($feld)`	Gibt die Anzahl der Elemente des Feldes zurück.
`sort($feld)` `rsort($feld)`	Sortiert die Werte des angegebenen Feldes aufsteigend bzw. absteigend.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_22

Last update: **2019/05/16 17:06**

1.8.6) Aufgaben 12 bis 18

Aufgabe 12 - Indizierter Array

1. Schreibe mittels PHP-Code einen „indizierten Array“, der folgende Elemente auflistet:

- Englisch
- Informatik
- Deutsch
- Geografie
- Bewegung und Sport

2. Gib alle eingegebenen Fächer aus.

3. Schreibe folgenden Satz im PHP-Block und greife auf den Array zu, um ihn zu vervollständigen:
„Mein Lieblingsfach ist [.....] „

Aufgabe 13 - Assoziativer Array

1. Schreibe mittels PHP-Code einen „assoziativen Array“. Beziehe folgende Werte ein:

- Hauptfach - Englisch
- Nebenfach - Geschichte

2. Schreibe die folgenden Sätze im PHP-Block und greife auf den Array zu, um den Satz zu vervollständigen:

[Englisch] ist ein Hauptfach. [Geschichte] ist ein Nebenfach.

Aufgabe 14 - Jahreszeit

Erstelle ein Dokument namens jahreszeit.php. Hier sollen die vier Jahreszeiten in einer Werteliste namens

```
$jahreszeit[0]
```

abgebildet werden können. Nutze dafür die **Kurzform**, indem du alle Werte auf einmal zuweist. Gib den Herbst aus.

Aufgabe 15 - Monatsnamen

Erstelle ein Dokument namens monat.php. Speichere alle Monatsnamen in einem Array namens `$monat`. Nutze dafür die **Langform**, indem du die Monate einzeln zuweist! Gib sodann folgenden

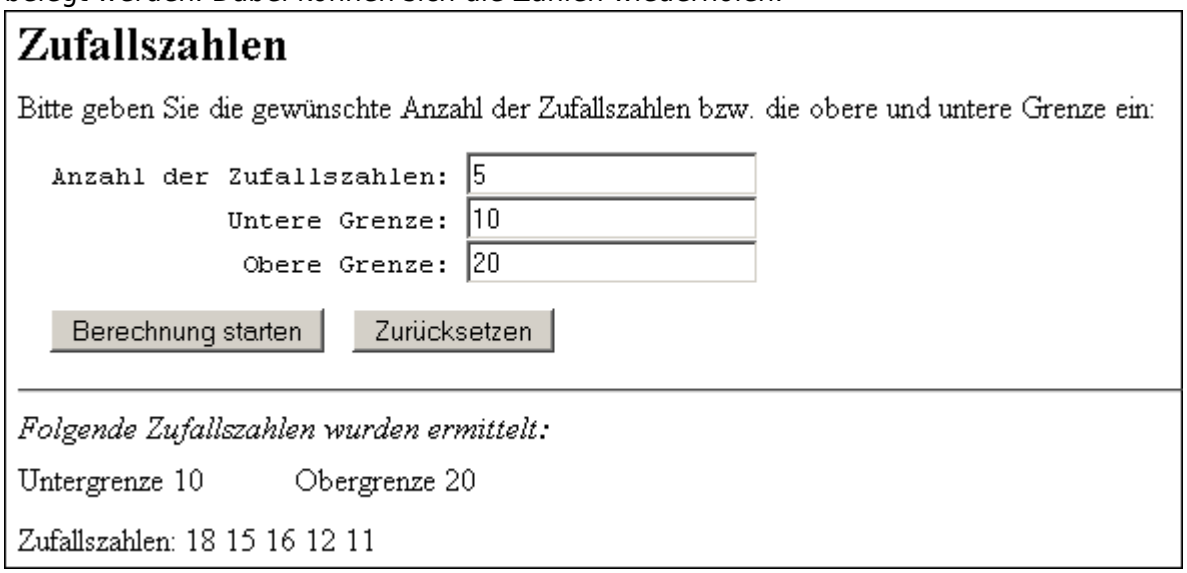
String aus:

„Ein Jahr hat 12* Monate. Der aktuelle Monat ist der Mai.“ (* Zählfunktion einsetzen)

Überlege dabei, welcher Arraytyp sinnvoll ist (indiziert oder assoziativ)!

Aufgabe 16 - Zufallszahlen

Ein Array soll mit einer vorgegebenen Anzahl von Zufallszahlen zwischen den Grenzen ug und og belegt werden. Dabei können sich die Zahlen wiederholen.



Hinweis: Wie man eine Zufallszahl zwischen den Grenzen ug und og zu erhalten kann, zeigt die folgende Zeile:

```
        $zahl[$i]=rand($ug,$og);
```

1. Nach der Generierung der Zufallszahlen soll das Array ausgegeben werden.
2. Der Mittelwert aller Zahlen soll ausgegeben werden. 3. Die Standardabweichung soll ausgegeben werden.

Aufgabe 17 - Primzahlenberechnung mit dem Sieb des Erathostenes

Zur Berechnung aller Primzahlen bis zu einer Obergrenze n gibt es ein Verfahren, das vom griechischem Mahtematiker *Erathostenes* stammt. Es wird das *Sieb des Erathostenes* genannt und basiert auf der Idee, dass die Vielfachen einer Primzahl mit Sicherheit keine Primzahlen sind. Wenn wir z.B. wissen, dass 2 eine Primzahl ist, können wir alle Vielfache von 2 aus der Menge der Primzahlenkandidaten streichen. Die kleinste Zahl der dann verbleibenden Menge ist die nächste Primzahl. Wir eliminieren wiederum alle ihre Vielfache.

Implementieren lässt sich dieser Algorithmus sehr schön mit Hilfe eines Arrays, das zunächst lauter „p“ (Primzahl) in den Feldern stehen hat. Wird eine Zahl gestrichen, ersetzt man das „p“ durch ein „z“

(zusammengesetzte Zahl).

Die Primzahlen sind dann die Indizes jener Felder, in denen „p“ steht.

Aufgabe 18 - Leichtathletik

Erstelle eine neue Datei unter dem Namen `uebung_mehrdim.php` mit einem mehrdimensionales indiziertes Feld mit folgenden Inhalten und gib die Daten anschließend in Tabellenform (mit Überschrift) auf dem Bildschirm aus:

Beginn	Disziplin	Ort	Bemerkung
09:30 Uhr	Diskuswurf	Nebenplatz	Jugendmeisterschaften
10:00 Uhr	5-km-Lauf	Stadion - Laufbahn	Offener Lauf
11:00 Uhr	Halbmarathon	Waldgebiet	Teilnahme ab 18 Jahren
12:00 Uhr	Stabhochsprung	Stadion - Stabhochsprunganlage	Nur Frauen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_23

Last update: **2019/05/16 17:06**

1.8.7) Fragen zu Feldern

- Erkläre die Begriffe „eindimensionales“ und „mehrdimensionales Feld“ und gib jeweils ein sinnvolles bzw. typisches Beispiel an.
- Erkläre die Begriffe „indiziertes“ und „assoziatives Feld“. Worin besteht der Unterschied.
- Wie kann man bei indizierten bzw. bei assoziativen Felder auf die Feldwerte zugreifen?
- Welche typischen Befehle für den Zugriff auf Feldern sind dir bekannt?

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_24

Last update: **2019/05/16 17:06**

1.9.1) HIDDEN-Feld

Beim Formularfeldtyp **hidden** handelt es sich um ein verstecktes input-Feld, das für den Betrachter nicht sichtbar ist. Dennoch kann in ihm eine Variable gespeichert und auch übergeben werden. Genau darin besteht die Aufgabe eines versteckten Feldes, nämlich im Hintergrund verborgen Daten zu übergeben.

Datei einkauf.html

```
<html>
 <head>
  <title>Datenübergabe mittels HIDDEN-Feld</title>
 </head>
 <body>
  Heute kosten alle Artikel 5,99 Euro
  <form action="kaufen.php" method="POST">
     <input type="Hidden" name="preis" value="5.99">
     Artikel <input type="Text" name="artikel">
     Menge <input type="Text" name="menge">
     <input type="Submit" name="submit" value="Kaufen">
  </form>
  </body>
</html>
```

Datei kaufen.php

```
<html>
 <head>
  <title>Datenübergabe mittels HIDDEN-Feld</title>
 </head>
 <body>
  <?php
    echo "Sie haben den Artikel <b>".$_POST["artikel"] ."</b>".
$_POST["menge"]."
    mal zum Gesamtpreis von " . ($_POST["menge"] * $_POST["preis"]) . " Euro
gekauft.";
  ?>
 </body>
</html>
```

Hidden-Input-Felder stellen eine geeignete Möglichkeit dar, Daten über mehrere Webseiten hinweg zu transportieren. Schließlich muss es ausreichen, wenn der Benutzer beispielsweise seinen Vornamen nur auf der ersten Seite eines fünfseitigen Fragebogens eingibt und wir ihn trotzdem auf der fünften Seite noch immer freundlich beim Vornamen ansprechen können.

Wird die Eingabe auf der ersten Seite mittels eines Texteingabefeldes getätigt, so steht sein Vorname auf der zweiten Seite des Fragebogens in $_GET bzw. $_POST zur Verfügung. Diesen Wert können wir dank PHP in ein Hidden-Input-Feld übernehmen und transportieren den Wert damit sicher zur nächsten Seite, sobald der Benutzer die aktuelle Seite durch Aufruf des Absenden-Buttons verlässt.

Diese Kette kann, solange auf der nächsten Seite ein Formular folgt, beliebig lang sein.

Hier zur Verdeutlichung die entsprechenden Skriptbeispiele:

Erste Seite

```
Name : <input type="Text" name="name">
```

Nutzung des Hidden Input Feldes in den folgenden Seiten

```
<input type="Hidden" name="name" value="<?php echo $_POST["name"]; ?>">
```

Zu beachten ist, dass dieses Beispiel nur funktioniert, wenn keine Anführungszeichen im Namen vorkommen!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_25

Last update: **2019/05/16 17:06**

1.9.2) Textarea

Bei der Textarea handelt es sich um den großen Bruder des bereits mehrfach erwähnten Textfeldes. Die beiden Varianten unterscheiden sich nur dadurch, dass die Textarea aus mehreren Zeilen bestehen kann und somit für größere Eingaben besser geeignet ist, als ein einzeiliges Texteingabefeld. Im vorliegenden Beispiel wird beim Abschicken des Formulars die Variable `$bemerkung` mit dem Inhalt des Textareafeldes als Wert übergeben. Will man der Textarea einen Wert vorgeben, so muss man diesen Wert zwischen Anfangs- und Endtag des Textfeldes schreiben.

```
<textarea name="bemerkung"></textarea>
```

so sieht es aus:

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_26

Last update: **2019/05/16 17:06**

1.9.3) Checkbox

Checkbox ohne Array und implode()

Wird ein Formular mit Checkboxen abgeschickt, werden in PHP nur die Einträge im Array $_GET bzw. $_POST erzeugt, deren Felder beim Ausfüllen ausgewählt werden. Wenn also nur das 2. und 3. Kästchen angeklickt wird, dann werden nur die Variablen `$_GET["artikel2"]` mit dem Wert Artikel 2 und `$_GET["artikel3"]` mit dem Wert Artikel 3 erzeugt.

Wenn eine oder mehrere Checkboxen standardmäßig den Zustand „Ausgewählt“ besitzen sollen, kann man innerhalb des input-Tags das Attribut checked verwenden.

```
<html>

 <head>
  <title>Checkbox</title>
 </head>
<body>

 <?php
    echo "<b>Sie haben folgende Artikel bestellt:</b><br>";
    echo $_GET["artikel1"] . "<br>";
    echo $_GET["artikel2"] . "<br>";
    echo $_GET["artikel3"] . "<br>";
    echo $_GET["artikel4"] . "<br>";
 ?>

<form action="check.php" method="get">
   <input type="Hidden" name="preis" value="5.99">
   <input type="Checkbox" name="artikel1" value="Artikel 1">Artikel 1<br>
   <input type="Checkbox" name="artikel2" value="Artikel 2">Artikel 2<br>
   <input type="Checkbox" name="artikel3" value="Artikel 3">Artikel 3<br>
   <input type="Checkbox" name="artikel4" value="Artikel 4">Artikel 4<br>
   <input type="Submit" name="submit" value="Kaufen">
</form>
</body>
</html>
```

so sieht es aus:

☐Artikel 1
☐Artikel 2
☐Artikel 3
☐Artikel 4
Kaufen

Checkbox mit Array und implode()

Checkbox lassen sich auch mittels einer Array - Anweisung erstellen. Wobei es bei der Ausgabe wichtig ist, nicht nur ein Element auszugeben, sondern es könnten möglicherweise alle Elemente, oder nur zwei sein.

Dazu dient die implode() - Anweisung

```
implode ( string $glue , arrayname )
Verbindet die Elemente eines Array mittels glue zu einer Zeichenkette
```

Nachdem er nur etwas im Array findet, wenn etwas angeklickt ist, sollte die Ausgabe nur erfolgen, wenn das Array nicht leer ist (eingebaute IF - Anweisung).

```
<html>
 <head>
  <title>Checkbox</title>
 </head>
<body>
<form action="check.php" method="POST">
   <input type="hidden" name="preis" value="5.99">
   <input type="checkbox" name="artikel[]" value="Artikel 1">Artikel 1<br>
   <input type="checkbox" name="artikel[]" value="Artikel 2">Artikel 2<br>
   <input type="checkbox" name="artikel[]" value="Artikel 3">Artikel 3<br>
   <input type="checkbox" name="artikel[]" value="Artikel 4">Artikel 4<br>
   <input type="submit" name="submit" value="Kaufen">
</form>

<?php
   if(isset($_POST["artikel"])){
    $var = $_POST["artikel"];
    $ausgabe = implode(", ", $var);
    echo "<b>Sie haben folgende Artikel bestellt: $ausgabe</b><br>"; }
 ?>
</body>
</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_27

Last update: **2019/05/16 17:06**

1.9.4) Radio-Button

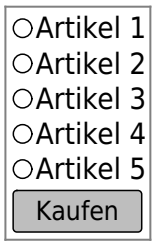
Radio-Buttons ohne Arrays erstellen

Während man bei Checkboxen immer beliebig viele Möglichkeiten auswählen kann, ist man bei Radio-Buttons auf nur eine einzige Antwortmöglichkeit beschränkt. Wie man sehen kann, haben bei Radio-Buttons im Gegensatz zu Checkboxen alle zusammengehörigen Auswahlmöglichkeiten denselben Namen. Daher kann auch nur ein Array-Eintrag erstellt werden, dem dann beim Abschicken der entsprechende Wert des angeklickten Feldes zugewiesen wird.

Ähnlich wie bei Checkboxen kann man hier für einen Radio-Button den Zustand „ausgewählt“ vorgeben, indem man im entsprechenden input-Tag das Attribut checked angibt. Zu beachten ist, dass die Angabe des Attributs nur dann sinnvoll ist, wenn sie sich auf genau einen Radio-Button beschränkt.

```
<html>
 <head>
  <title>Radio-Buttons</title>
 </head>
<body>
<?php
if (isset($_POST["submit"]))
  {
  echo "Sie haben Artikel <b>" . $_POST["artikel"] . "</b> bestellt!";
  }
 ?>
<form action="radiobutton.php" method="POST">
<input type="Hidden" name="preis" value="5.99">
<input type="Radio" name="artikel" value="1" checked>Artikel 1<br>
<input type="Radio" name="artikel" value="2">Artikel 2<br>
<input type="Radio" name="artikel" value="3">Artikel 3<br>
<input type="Radio" name="artikel" value="4">Artikel 4<br>
<input type="Radio" name="artikel" value="5">Artikel 5<br>
<input type="Submit" name="submit" value="Kaufen">
</form>
</body>
</html>
```

so sieht es aus:



Radio-Buttons mit Arrays erstellen

Radio Buttons lassen sich auch mittels einer Array - Anweisung erstellen. Wozu es einem PHP Code im HTML Teil benötigt.
Die Werte der RadioButtons werden in einem Array abgelegt und danch mittels forach ausgegeben. Dabei bekommt der Schüssel, der auch als value-Wert des Radio-Buttens dient, und der Ausgabewert denselben Inhalt. Nämlich jenen der im Array abgespeichert ist.

ACHTUNG!: Übermittelt wird nur der Indexwert, aber nicht der Inhalt des Arrays!

```
<html>
 <head>
  <title>Radio-Buttons</title>
 </head>
<body>
<?php
if (isset($_POST["submit"]))
  { $nummer=$_POST["artikel"]+1;
  echo "Sie haben Artikel <b>" . $nummer . "</b> bestellt!";
  }
?>
<form action="radiobutton.php" method="POST">
<input type="Hidden" name="preis" value="5.99">
<?php
$artikel = array ("Artikel1", "Artikel2", "Artikel3", "Artikel4",
"Artikel5");
 foreach($artikel as $key => $artikelnr){
 echo "<input type='radio' name='artikel' value='$key' />$artikelnr<br>"; }
 ?>
<input type="Submit" name="submit" value="Kaufen">
</form>
</body>
</html>
```

Für komplexere Ausgaben, lässt sich diese auch mit einer **switch-case Anweisung** lösen.

```
<?php
if (isset($_POST["submit"]))
  { echo "Sie haben ";
    switch($_POST["artikel"])
    {case "0":
    echo "Artikel1";
    break;
    case "1":
    echo "Artikel2";
    break;
    case "2":
    echo "Artikel3";
    break;
    case "3":
```

```
    echo "Artikel4";
    break;
    case "4":
    echo "Artikel5";
    break;
    }
    echo " bestellt";
 }
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_28

Last update: **2019/05/16 17:06**

1.9.5) Auswahlfelder

Als Alternative zu den Radiobuttons stehen noch die Select-Auswahlfelder zur Verfügung:

Select-Auswahlfelder:

```
<select name="artikel">
 <option value="1">Auswahl 1</option>
 <option value="2">Auswahl 2</option>
 <option value="3">Auswahl 3</option>
 <option value="4">Auswahl 4</option>
 <option value="5">Auswahl 5</option>
</select>
```

so sieht es aus:

Auswahl 1 ▼

Hier wird analog zu den Radiobuttons nur ein Eintrag im Array $_POST erzeugt, dessen Name durch die Angabe im Element select vorgegeben ist. Der Wert des Eintrags ergibt sich aus der gewählten Option. Ähnlich wie bei einem Radio-Button kann man angeben, dass ein Eintrag einer Auswahlliste standardmäßig ausgewählt sein soll. Dazu muss man beim entsprechenden option-Tag das Attribut selected eintragen. Genau wie bei Radio-Buttons macht die Angabe natürlich nur Sinn, solange sie sich auf einen Eintrag der Liste beschränkt.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_29

Last update: **2019/05/16 17:06**

1.9.6) Aufgabe 20

Erstelle das nachstehende Formular für die Preisberechnung auf der Website eines Winzerbetriebs:

Weinbestellung

Hier können Sie den Preis für Ihre Weinbestellung einfach berechnen:
☐ Flaschen (1l) Grüner Veltliner Kabinett 2006 (€ 2,20)
☐ Flaschen (0,75 l) Chardonnay 2009 (€ 5,00)
☐ Flaschen (0,75 l) Muskateller 2009 (€ 5,50)
☐ Flaschen (0,75 l) Merlot 2008 (€ 4,50)
○Selbstabholung
○Lieferung innerhalb des Bezirks (zzgl. € 25,00)
Ab einem Gesamtbestellwert (inkl. eventueller Liefergebühr) von insg. 150€ wird ein 4%iger Rabatt gewährt.
☐Ja, ich möchte den original Winzer-Newsletter erhalten!

Die Ausgabe soll auf der selben Seite unterhalb des Submit-Buttons erfolgen und etwa wie folgt aussehen:

Sie haben

- 5 Flaschen Grüner Veltliner (€ 11,00)
- 10 Flaschen Chardonnay (€ 50,00)
- 15 Flaschen Merlot (€ 67,50)

bestellt und wollen die Ware geliefert (€ 25,00) bekommen. Da der Gesamtbestellwert (€ 153,50) mind. € 150,00 beträgt, erhalten Sie einen 4%igen Rabatt und zahlen daher insgesamt nur **€ 147,36**.

Ja, Sie werden in Zukunft den original Winzer-Newsletter erhalten.
Herzlichen Dank für Ihre Bestellung!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_30

Last update: **2019/05/16 17:06**

1.10.1) Nutzung von externen Dateien

Beispiel für die Nutzung von Daten aus externen Dateien

Wenn Sie Daten, die die Benutzer in ein Formular eingegeben haben, z. B. in einer Excel-Tabelle auswerten oder bearbeiten möchten, können Sie die Daten außerhalb des PHP-Programms in einer externen Datei speichern.

So können Sie beispielsweise auf diese Datei zugreifen, die Daten lesen, aber auch neue Daten hinzufügen bzw. die Daten ersetzen. Beispiele für die sinnvolle Nutzung von externen Dateien sind:

- Dynamische Daten in eine Webseite einbauen: Beispielsweise können Sie einen Hinweis auf das Angebot des Tages in einer externen Textdatei speichern. So brauchen Sie, um die Daten zu verändern, nur die externe Datei zu bearbeiten.
- Eingaben aus einem Formular speichern
- Logdateien schreiben: Um z. B. die Anzahl der Aufrufe einer Seite zu protokollieren, können Sie die Daten in einer eigenen Datei speichern.

Wichtig bei der Bearbeitung von externen Dateien ist die Art des Zugriffs auf die Datei. Hierbei kann PHP unterscheiden zwischen dem Zugriff auf sequenzielle und binäre Dateien.

Sequenzielle Dateien, z. B. Textdateien (`*.txt`) oder Dateien im Tabellenformat (`*.csv`), beinhalten Zeilen unterschiedlicher Länge. Sie werden sowohl beim Schreiben als auch beim Lesen sequenziell (also eine Zeile nach der anderen) bearbeitet.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_31

Last update: **2019/05/16 17:06**

1.10.2) Dateien öffnen, lesen und schließen

Dateien mit fopen() öffnen

Bevor Sie auf die Daten in Dateien zugreifen können, müssen Sie die entsprechende Datei öffnen. Dazu können Sie die `fopen()`-Anweisung verwenden.

Syntax und Bedeutung der fopen() -Anweisung

```
fopen(Dateiname,Modus);

$datei =  fopen("user.txt","r")
```

- Zusätzlich zur Angabe der zu öffnenden Datei wird der Modus, in dem die Datei geöffnet werden soll, angegeben:

- `r` ... Die Datei wird zum Lesen (r = read) geöffnet; Schreiben ist nicht erlaubt.
- `r+` ... Die Datei wird zum Lesen und Schreiben geöffnet, und der Dateizeiger an den Anfang der Datei gesetzt.
- w ... Öffnet eine Datei zum Schreiben (w = write)
- w+ ... Hier handelt es sich um dieselbe Option wie die Option w, nur wird hier die Datei zum Lesen und Schreiben geöffnet. Der Dateizeiger wird auf den Anfang der Datei gesetzt sowie die Länge der Datei auf 0 Byte.
- a ... Die Datei wird zum Schreiben geöffnet, wobei die neuen Daten die vorhandenen Daten ergänzen (a = append).
- a+ ... Die Datei wird zusätzlich zum Lesen geöffnet.
- b ... Zusätzlich kann der Option Modus der Buchstabe 'b' hinzugefügt werden, der die Behandlung von Binär-Dateien erlaubt.

Dies ist auf Systemen sinnvoll, wenn diese zwischen Binär- und Text-Dateien unterscheiden (wie z. B. Windows).

- Der Rückgabewert der Funktion `fopen()` ist ein Zeiger auf die geöffnete Datei und wird für weitere Zugriffe auf die Datei genutzt. Im Beispiel wird der Dateizeiger in der Variablen `$datei` gespeichert.

Wenn die angegebene Datei nicht gefunden wird, gibt die Funktion fopen Oden Wert FALSE zurück.

Dateien mit fgets() lesen

Mithilfe der Funktion `fgets()` können Sie den Inhalt einer Datei auslesen.

Syntax und Bedeutung der fgets() -Anweisung

```
fgets(Dateizeiger[,Modus]);

$zeile  = fgets($datei);
```

- Im ersten Parameter (Dateizeiger) finden Sie die Angabe zur Datei, aus der gelesen werden soll. Hierbei wird der Dateizeiger im Beispiel `$datei` aus der vorher geöffneten Datei verwendet.
- `fgets()` liest die angegebene Datei zeilenweise aus, wobei Sie optional im Parameter Modus die Länge angeben können. Lassen Sie diesen Parameter weg, ist automatisch eine Länge von maximal 1024 Zeichen für eine zu lesende Zeile festgelegt.
- `fgets()` liest bis zu dem ersten Auftreten eines Zeilenumbruchs bzw. der maximalen Zeilenlänge, dem Ende der Datei (EOF = End of File) oder bis zu der im Parameter Modus ausgegebenen Länge.
- Wenn Sie `fgets()` erneut aufrufen, wird die nächste Zeile gelesen.
- Der Rückgabewert der Funktion fgets() ist eine Zeichenkette. Im Beispiel wird der Rückgabewert in der Variablen `$zeile` gespeichert.

Dateien mit fclose() schließen

Nachdem Sie eine Datei geöffnet und die Daten ausgelesen haben, müssen Sie die Datei wieder schließen, um sie für andere Prozesse oder Benutzer nutzbar zu machen.

Syntax und Bedeutung der fclose()-Anweisung

```
fclose(Dateizeiger);

fclose($datei);
```

- Mit der Funktion `fclose()` schließen Sie die Datei, auf die der Dateizeiger weist. Der Dateizeiger muss gültig, also mit der Funktion `fopen()` geöffnet worden sein.
- Wurde die Datei erfolgreich geschlossen, liefert die Funktion `fclose()` den Wert TRUE zurück, sonst FALSE.

Beispiel zu externe Datei öffnen, lesen und schließen: fgets.php, user.txt

```
<html>
<body>
<?php
 if (file_exists("Gedicht.txt"))
     echo "Datei ist vorhanden!<br><hr>";

 $datei = fopen("Gedicht.txt","r");
 if ($datei)
 {
    echo "<p>Datei Gedicht.txt:</p>";
    while (!feof($datei))
    {
      $zeile = fgets($datei);
     echo $zeile."<br>";
    }
  fclose($datei);
 }
```

```
 else
  echo " Es trat ein Fehler auf! " ;
?>
</body>
</html>
</html>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_32

Last update: **2019/05/16 17:06**

1.10.3) Weitere Möglichkeiten zum Lesen von Dateien

Dateien mit readfile() oder file() lesen

Während die Funktion `fgets()` den Inhalt einer Datei zeilenweise einliest, können Sie mit den Befehlen `readfile()` und `file()` den gesamten Inhalt einer Datei auf einmal wiedergeben.

Der Unterschied zwischen den beiden Funktionen `readfile()` und `file()` liegt in der Ausgabe der Daten der Datei:

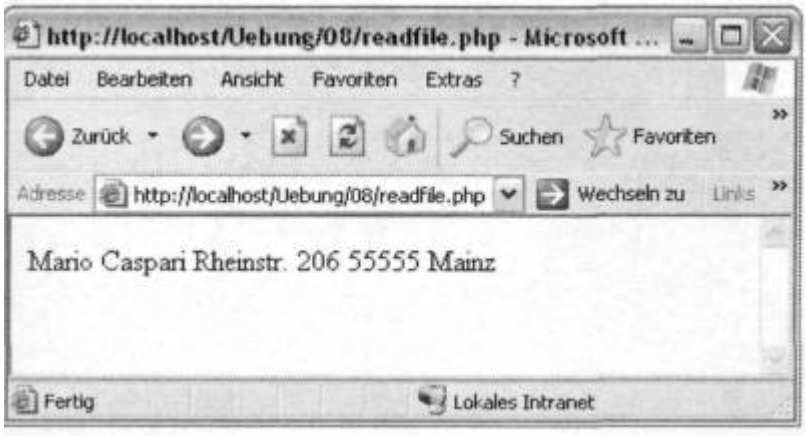
- `readfile()` liest vollständig den Inhalt der Datei aus und sendet das Ergebnis ohne weitere Bearbeitung direkt an den Browser.
- `file()` hingegen liest den vollständigen Inhalt der Datei zeilenweise in einen eindimensionales array. Hierbei wird jeweils der Inhalt einer Zeile zu einem Element des Feldes.

```
readfile(Datei);

file(Datei);
```

Beispiel readfile.php

```
<html>
<body>
 <h4>Auslesen einer Textdatei</h4>
 <?php readfile("user.txt"); ?>
</body>
</html>
```



Beispiel file.php

```
<html>
<body>
<h4>Auslesen einer Textdatei</h4>
<?php
 $feld = file("user.txt");
```

```
 echo "<p>";
 $i=1;
 foreach($feld as $zeile)
 {echo "Zeile ".$i++.":  ";
  echo $zeile."<br>";
 }
?>
</body>
</html>
```



Inhalt einer Datei in eine Zeichenkette einlesen

Sie können die gesamte Datei in einen String einlesen. Hierbei verbinden Sie die einzelnen ausgelesenen Zeilen mithilfe der Zeichenkettenfunktion `join()`.

Sie können mit der Funktion `join()` ein Feld anhand eines Trennzeichens zu einem String zusammenfügen. Dabei werden die Elemente des Feldes nacheinander an den String angehängt.

```
join(Trennzeichen,Feld);
```

Beispiel join.php

```
<?php
 if (file_exists("user.txt"))
 {$finhalt = join (";",file("user.txt"));
  echo $finhalt;
 }
?>
```

Beispieldatei „join.php“

- Mithilfe der Funktion `file_exists()` wird überprüft, ob die Datei existiert. Die Funktion `file_ exists()`erwartet als einzigen Parameter den Namen einer Datei auf dem Webserver. Dateien anderer Server (URL) können nicht auf ihre Existenz überprüft werden.
- Der Befehl `join()` verbindet die mithilfe der Funktion `file()` ausgelesenen Datei *user.txt* zu einer Zeichenkette. Als Trennzeichen wird hierbei das Semikolon „;“ verwendet.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_33

Last update: **2019/05/16 17:06**

1.10.4) In Dateien schreiben

Dateien zum Schreiben öffnen

Wollen Sie Daten in eine Datei schreiben, müssen Sie die Datei zuerst öffnen. Beim Öffnen der Datei können Sie bestimmen, ob Sie die bestehenden Daten der Datei überschreiben wollen oder der bestehenden Datei weitere Daten hinzufügen möchten.

(1) Dateien überschreiben

Wenn Sie eine Datei mit der Funktion `fopen()` und dem Modus 'w' für „write“ öffnen, wird die Datei zum Schreiben geöffnet und der Dateizeiger auf den Anfang der Datei verschoben. Gleichzeitig wird die Länge der Datei auf 0 Byte gesetzt. Wenn die Datei nicht existiert, wird sie angelegt.

(2) Daten in Dateien hinzufügen

Um beispielsweise Daten, die in ein Formular eingegeben wurden, fortlaufend in eine Datei zu schreiben, verwenden Sie den Befehl `fopen()` mit dem Modus „a“. Die Datei wird zum Schreiben geöffnet und der Dateizeiger an das Ende der Datei gesetzt, sodass die neuen Daten hinzugefügt werden. Existiert sie nicht, legt PHP sie an.

Daten in Dateien schreiben

Um eine Zeichenkette in eine geöffnete Datei zu schreiben, verwenden Sie die Funktion `fputs()`. Syntax und Bedeutung der fputs-Anweisung

```
fputs (Dateizeiger,Zeichenkette,[Länge])
```

- Mit der Funktion `fputs()` können Sie eine beliebige Zeichenkette in eine Datei schreiben.
- Die entsprechende Datei wird über den mit `fopen()` festgelegten Dateizeiger angesprochen.
- Ist der optionale Parameter Länge angegeben, wird das Schreiben nach der angegebenen Anzahl Bytes beendet. Geben Sie ihn nicht an, wird die gesamte Zeichenkette geschrieben.
- `fputs()` gibt bei Erfolg die Anzahl der geschriebenen Bytes zurück, andernfalls FALSE.

Auf Systemen, die zwischen Binär- und Textdateien (z. B. Windows) unterscheiden, muss die Datei mit der Option 'b' in der `fopen()`-Funktion geöffnet werden.

Nachdem Daten in die Datei geschrieben wurden, muss die Datei geschlossen werden.

Beispiel: Dateidaten ergänzen bestellformular.html, bestellung.php, bestellung_daten.csv

Die Daten, die der Benutzer in das Bestellformular für Äpfel eingibt, sollen in einer `*.csv`-Datei an das Ende der schon vorhandenen Daten gehängt werden.



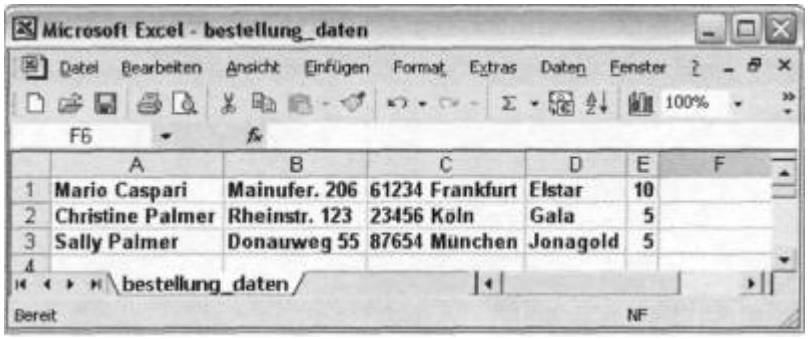

```
<html>
<body>
 <?php
 $datei = fopen("bestellung_daten.csv","a");
 if ($datei == false)
 { echo  "<p>Datei konnte nicht zum Schreiben geöffnet werden</p>";
   exit ;
 }
 $name    = $_POST["name"];
 $strasse = $_POST["strasse"];
 $ort     = $_POST["ort"];
 $sorte   = $_POST["sorte"];
 $menge      = $_POST["menge"];
 fputs ($datei, "$name;$strasse;$ort;$sorte;$menge;n");
 echo "<p>Folgende Angaben wurden gespeichert</p>";
 echo "$name<br>"; echo "$strasse<br>";
 echo "$ort<br>";  echo "$menge kg $sorte<br>";
 fclose($datei);
?>
</body>
</html>
```

Die Datei `bestellungdaten.csv` wird mit `fopen()` und dem Modus 'a' geöffnet, d.h. zum Schreiben geöffnet, und der Dateizeiger wird an das Ende der Datei gesetzt, sodass die neuen Daten hinzugefügt werden.

- Es wird geprüft, ob das Öffnen der Datei erfolgreich war.
- Wenn die Datei nicht erfolgreich geöffnet werden kann, wird mit exit das Programm gestoppt. Mit exit beenden Sie - ohne Möglichkeit der Rückkehr - die Skriptausführung.
- Die Variablen `$name, $strasse` etc. werden mit den Daten, die aus dem Formular übergeben wurden, gefüllt.
- Über die Funktion `fputs()` werden die Werte der Variablen in die geöffnete Datei geschrieben. Hierbei werden die Daten jeweils durch ein Semikolon voneinander getrennt, und am Ende einer

Zeile wird mit-hilfe von „\n“ ein Zeilenvorschub erzeugt, um die neuen Daten in eine eigene Zeile zu schreiben.
- Die geöffnete Datei wird geschlossen.

*.csv können Sie mit Programmen, wie z. B. Microsoft Excel, öffnen und bearbeiten.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_34

Last update: **2019/05/16 17:06**

1.10.5) Gästebuch

In der hier vorgestellten Variante eines Gästebuches werden nur der Name des Gastes und ein Text in der zugehörigen Textdatei gastbuch.txt gespeichert. Diese Textdatei muss Lese- und Schreibrechte für alle aufweisen.

- Die Funktion `file_exists()` testet, ob die genannte Datei im angegebenen Verzeichnis vorliegt.
- Die Funktion `feof()` testet, ob das Ende der Textdatei erreicht wurde. Der Ausdruck `!feof($fp)` liefert also so lange true, so lange das Dateiende nicht erreicht wurde.
- Die Funktion `fgets()` liest aus der mit der Ressource-ID festgelegten Datei einen so langen Datenstrom, wie das zweite Argument (im Beispiel 1000) angibt.
- Die Funktion `is_writeable()` testet, ob Schreibrechte für die angegebene Datei gewährt wurden.
- Die Funktion `fputs()` schreibt die angegebene Zeichenkette bzw. die Werte der angegebenen Variablen in die Datei, die mit der Ressource-ID identifiziert wurde.

Zusatz-Aufgabe

- Zusätzliche Informationen - etwa über Homepage, Datum, IP-Adresse, der Mitteilung, etc. sollen in das Gästebuch eingetragen werden. Informiere dich bezüglich des Einfügens des aktuellen Datums im Internet.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_35

Last update: **2019/05/16 17:06**

1.10.6) Beispiel: Besucherzähler

Eine häufige Anwendung in Verbindung mit Dateizugriffen ist die Einbettung eines Counters (Zugriffszählers) in eine Webseite. Ein Counter zählt die Anzahl der Webseiten-Zugriffe. Dabei wird bei jedem Zugriff die Zahl der bisherigen Besucher um eins erhöht. Das heißt, es wird auf eine bestehende Datei zugegriffen und mit ihrem Inhalt gerechnet. Das Ergebnis der Berechnung überschreibt dann den Inhalt der Datei.

Hinweise zum Beispiel:

- In den HTML-Text wird die Funktion `counter()` zur Anzeige der Besucherzahl eingebunden.
- Die Funktion `counter()` wird angelegt.
- Der Name der Datei, in der die Anzahl der Besucher abgelegt werden soll, lautet *counter.txt* und wird der Variablen `$name` zugewiesen. Die Variable `$count` erhält den Ausgangswert 0.
- Die Datei *counter.txt* wird über die Funktion fopen() zum Lesen und Schreiben geöffnet. Der Zugriff auf diese Datei erfolgt über den Dateizeiger `$datei`. Der Dateizeiger wird mit dem Modus `r+` an den Anfang der Datei gesetzt. War der Zugriff auf die Datei nicht erfolgreich, wird das Skript mit einer Meldung verlassen.
- Damit während der Arbeit mit der Datei kein weiterer Zugriff und somit ein gleichzeitiges Überschreiben der Daten möglich ist, wird die Datei exklusiv (LOCK_EX) gesperrt. LOCK_EX steht für eine exklusive, nur schreibende Verriegelung einer Datei (vgl. folgenden Abschnitt).
- Mit `fgets()` werden die ersten 10 Zeichen der Datei ausgelesen. Der Wert 10 wurde gewählt, da hiermit ein Zählen der Besucherzahl mit 10 Stellen (bis 9999999999) gewährleistet ist.
- Damit der neue Wert den alten Wert in der Datei überschreiben kann, wird der Dateizeiger mittels `fseek()` und der Angabe 0 an den Anfang der Datei gesetzt.
- Nach dem Einlesen des Wertes wird er um eins erhöht und am Bildschirm fett formatiert ausgegeben. Über die Funktion `fwrite()` wird der neue Wert `$count` in die Datei geschrieben.
- `fclose()` schließt die Datei *counter.txt* und beendet zugleich die Zugriffssperre, die über `flock()` veranlasst wurde.



Mit dem Befehl `flock()` können Sie eine Datei in der Zeit, in der das Skript in eine Datei schreibt, zusätzlich für weitere Zugriffe sperren.

Syntax und Bedeutung der flock()-Anweisung

```
flock (Dateizeiger,Modus);
```

- `flock()` sperrt den Zugriff auf die Datei, auf die der Parameter Dateizeiger verweist. Der Modus ermöglicht verschiedene Sperrzustände der Datei.
- Darf während Ihres Zugriffs die Datei von anderen nur gelesen werden, setzen Sie den Modus auf LOCK_SH (Shared Lock, Lesezugriff). Soll kein anderer Nutzer zeitgleich die Datei nutzen dürfen, setzen Sie die Option LOCK_EX (Exclusive Lock), `flock()` wartet, bis die Datei wie angegeben benutzt werden kann. Geben Sie locknb (No Block) an, kehrt die Funktion mit dem Wert FALSE zurück, wenn die Datei bereits von einem anderen Programm gesperrt ist. Möchten Sie die Verriegelung wieder freigeben, geben Sie LOCK_UN (Unlock) an.
- Die Funktion liefert bei Erfolg den Wert TRUE zurück bzw. FALSE, wenn ein Fehler auftrat. Ob die Zugriffssicherung funktioniert, erfahren Sie, indem Sie den Rückgabewert überprüfen.
- Die Sperre wird über die Funktion `fclose()` oder am Ende des Skripts automatisch aufgehoben.
- Bei manchen Betriebssystemen ist die Funktion `flock()` auf Prozess-Ebene (Teil einer Applikation) implementiert. Hierbei können Sie sich nicht auf `flock()` verlassen, um Dateien vor dem Zugriff von anderen PHP-Skripten zu schützen. Zudem wird `flock()` nicht von allen Dateisystemen unterstützt und gibt deshalb in solchen Umgebungen immer FALSE zurück.

Den Dateizeiger mit "fseek()" an eine bestimmte Position setzen

Im vorher gezeigten Beispiel ist es notwendig, den Dateizeiger an den Anfang der Datei zu setzen, um den alten Wert des Zählers mit dem neuen Wert überschreiben zu können. Den Dateizeiger können Sie über `fseek()` positionieren.

```
fseek (Dateizeiger,Stelle[,Wie]);
```

- Hierbei geben Sie für den Parameter Stelle die Anzahl in Bytes, bezogen auf den Dateianfang, der durch den Dateizeiger festgelegt wird, an.
- Der optionale Parameter `[,Wie]` legt bestimmte Bezugspunkte für die Ermittlung der Position fest:
  - Der Parameter SEEK_CUR ermöglicht die Verschiebung von der aktuellen Position des Dateizeigers plus Stelle, SEEK_END vom Dateiende und SEEK_SET vom Dateianfang.
- Fehlt der Parameter Wie, ist standardmäßig die Option SEEK_SET gewählt.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_36

Last update: **2019/05/16 17:06**

1.10.7) Übungsaufgaben

Aufgabe 23: Textdatei einlesen und zeilenweise ausgeben

Gedicht.txt

```
Wer will was Lebendig's erkennen und beschreiben,
Sucht erst den Geist heraus zu treiben,
Dann hat er die Teile in seiner Hand,
Fehlt leider! nur das geistige Band
```

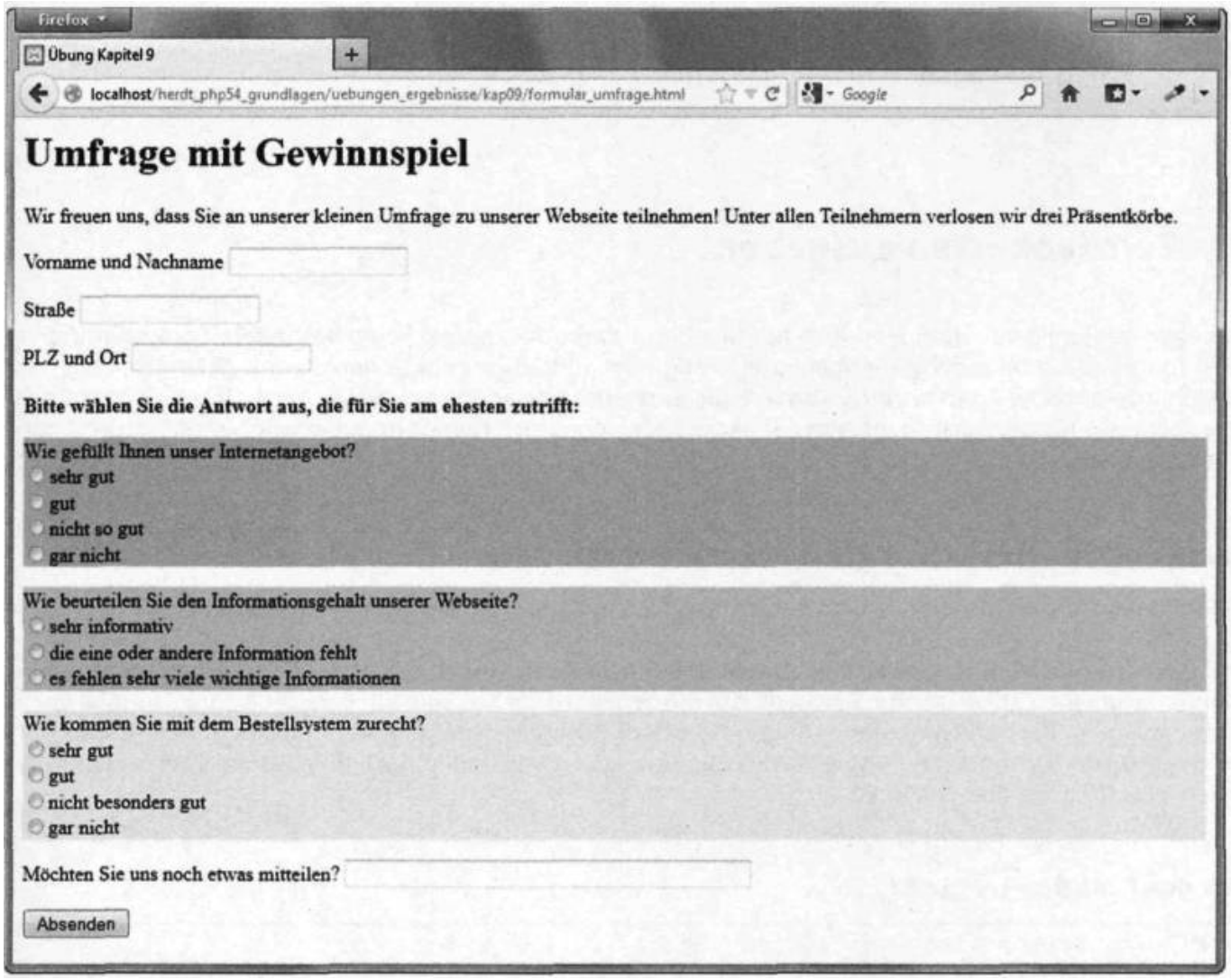
Aufgabe24: Apfel-Bestellungen aufgeben und csv-Datei ausgeben

bestellung_daten.csv

Aufgabe 25: Umfrage mit Gewinnspiel

a) Du möchtest eine Umfrage durchführen. Hierfür erstellst Du ein Formular in das Daten eingegeben werden. Um die Daten auswerten zu können, schreibe ein PHP-Skript, das die Daten an eine Tabelle (*.csv) übergibt und dort speichert.

Das Umfrageformular, das auch das PHP-Script beinhalten soll, speichere unter dem Namen `umfrage.php` und die übergebenen Daten werden in der Datei `umfrage_daten.csv` gespeichert. Hinweis: Die gewählten Optionen werden als Zahlen in die Datendatei geschrieben (1 für 1. Option, 2 für 2. Option, ...).



b) Baue (wie bei Aufgabe 2 aus der Schulübung) die Datei `umfrage.php` zur einem Verwaltungstool aus, dass es ermöglicht, neben der Eingabe von neuen Daten auch die Datei `umfrage_daten.csv` übersichtlich am Schirm auszugeben.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_37

Last update: **2019/05/16 17:06**

1.11.1) Datum und Zeit

getdate()

- `$datum = getdate()` liefert Informationen zur aktuellen Zeit, zum Wochentag, Monat und vielem mehr. Diese Funktion liefert das Ergebnis in einem assoziativen Feld zurück.
- Der Anweisung `getdate()` kann man den Parameter `Zeitstempel` hinzufügen, mit dem man das auszulesende Datum angibt. Der Parameter `Zeitstempel` erwartet als Wert die Sekundenanzahl seit dem 01.01.1970. Sämtliche Zeitangaben beruhen auf diesem Datum, das auch als Beginn der UNIX-Epoche bezeichnet wird.
- `print_r($datum)` veranschaulicht den Rückgabewert der Variablen `$datum`.

Übung 1

Versuche mittels `$datum = getdate()` folgende Ausgabe des aktuellen Datums durchzuführen:
Stunde: 12
Minute: 36
Sekunde: 20
Tag der Woche: 3 = Wednesday
Tag des Monats: 17
Tag des Jahres: 321
Monat: 11 = November
Jahr: 2010

Lösung

```
<?php
$datum=getdate();
echo "Sekunden ".$datum[seconds] . "<br>";
echo "Minuten ".$datum[minutes] . "<br>";
echo "Stunden ".$datum[hours] . "<br>";
echo "Tag des Monats ".$datum[mday] . "<br>";
echo "Tag der Woche ".$datum[wday] . "<br>";
echo "Monat ".$datum[mon] . "<br>";
echo "Jahr ".$datum[year] . "<br>";
echo "Tag des Jahres ".$datum[yday] . "<br>";
echo "Wochentag ".$datum[weekday] . "<br>";
echo "Monat ".$datum[month] . "<br><br>";
 ?>
```

Datum und Zeit formatieren

Die `date()`-Anweisung erwartet als Parameter die Formatanweisungen, welche Datum-Segmente angezeigt werden sollen.

Bsp: `echo date("l d")." of ". date("F Y h:i:s A");` liefert:
Wednesday 17 of November 2010 12:36:16 PM

Übung 2

Versuche mittels `date()` folgende Ausgabe zu erstellen:
17.11.10 um 12:36:16
17.11.10 = 321. Tag des Jahres 2010
17-11-2010
2010-11-17

Lösung

```
 <?php
echo date("d.m.y"). " um ".date("H:i:s")."<br>";
echo date("d.m.y"). " = ".date("z").". Tag des Jahres ".date("Y")."<br>";
echo date("d-m-Y")."<br>";
echo date("Y-m-d")."<br>";
 ?>
```

Mit date() kann man eine Zeitangabe formatieren oder auswerten. Die Zeitangabe übergeben Sie im Parameter timestamp. Lassen Sie diesen Parameter leer, nimmt die Funktion die aktuelle Zeit. Der Parameter format ist ein String, der festlegt, welche Informationen über die Zeitangabe Sie benötigen. In diesem String sind folgende Platzhalter möglich (*: Ausgabe mit führenden Nullen):

- a - „am“ oder „pm“
- A - „AM“ oder „PM“
- B - Tage bis Jahresende
- d - Tag des Monats *(01 – 31)
- D - Tag der Woche (Wed – 3stellig)
- F - Monatsangabe (December – ganzes Wort)
- g - Stunde im 12-Stunden-Format (1-12)
- G - Stunde im 24-Stunden-Format (0-23)
- h - Stunde im 12-Stunden-Format *(01-12)
- H - Stunde im 24-Stunden-Format *(00-23)
- i - Minuten *(00-59)
- I - (großes i) 1 bei Sommerzeit, 0 bei Winterzeit
- j - Tag des Monats (1-31)
- l - (kleines L) ausgeschriebener Wochentag (Monday)
- L - Schaltjahr = 1, kein Schaltjahr = 0
- m - Monat *(01-12)
- n - Monat (1-12)
- M - Monatsangabe (Feb – 3stellig)
- O - Zeitunterschied gegenüber Greenwich (GMT) in Stunden (z.B.: +0100)
- r - Formatiertes Datum (z.B.: Tue, 6 Jul 2004 22:58:15 +0200)
- s - Sekunden *(00 – 59)
- S - Englische Aufzählung (th für 2(second))
- t - Anzahl der Tage des Monats (28 – 31)
- T - Zeitzoneneinstellung des Rechners (z.B. CEST)

- U - Sekunden seit Beginn der UNIX-Epoche (1.1.1970)
- w - Wochentag (0(Sonntag) bis 6(Samstag))
- W - Wochennummer des Jahres (z.B.: 28)
- Y - Jahreszahl, vierstellig (2001)
- y - Jahreszahl, zweistellig (01)
- z - Tag des Jahres (z.B. 148 (entspricht 29.05.2001))
- Z - Offset der Zeitzone gegenüber GTM (-43200 – 43200) in Minuten

Länder- und Spracheinstellungen ändern

Mit folgendem Code wird die deutsche Sprache für alle Werte eingestellt.

```
<?php
  setlocale(LC_ALL, "german");
?>
```

strftime(Format [,Zeitstempel])

- Die Anweisung erwartet als Parameter die Formatanweisungen
- Ohne die Angabe des zweiten Parameters Zeitstempel liefert die Funktion das aktuelle Datum zurück analog der date()-Funktion.

Zeitfunktionen

time()

- Diese Funktion ist ohne Parameter zu verwenden und gibt die Zeit, die seit dem 1. 1. 1970 um 00:00:00 Uhr vergangen ist, in Sekunden zurück.

strtotime()

- Die Funktion liefert die Anzahl der Sekunden zwischen dem 1.1.1970 und dem angegebenen Datum zurück.

```
<?php
     echo strtotime("07.05.2019 10:00:00");
?>
```

mktime([Stunde[, Minute[, Sekunde[, Monat[, Tag[, Jahr);

- Die Funktion liefert die Anzahl der Sekunden zwischen dem 1. 1. 1970 und dem angegebenen Datum zurück.

```
<?php
  $tag   = 24;
```

```
  $monat = 8;
  $jahr  = 1978;

  $geburt = mktime(0,0,0,$monat,$tag,$jahr);
  $differenz = time() - $geburt;

  echo ((integer)($differenz/86400))." Tage liegen zwischen dem heutigen
Datum, ";
  echo date("j.n.Y")." und dem ".date("j.n.Y", $geburt).".";
?>
```

Datumsangaben überprüfen

checkdate(Monat, Tag, Jahr)

- Als Rückgabewert liefert die Funktion den Wert TRUE, falls das Datum existiert, sonst FALSE.

Zeit mittels microtime

microtime(true) liefert den aktuellen UNIX-Zeitstempel mit Mikrosekunden

```
$zeitstempel=microtime(true);
```

Beispiel:

```
<?php
$time_start = microtime(true);

// Sleep for a while
usleep(100);

$time_end = microtime(true);
$time = $time_end - $time_start;

echo "Did nothing in $time seconds\n";
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_38

Last update: **2019/05/16 17:06**

1.11.2) ÜBUNG 26 - GÄSTEBUCH ERWEITERUNG

Erweitere das Gästebuch aus der ÜBUNG 21 dahingehend, dass du nicht nur den Namen und den Text speicherst, sondern auch das Datum des Eintrages.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_39

Last update: **2019/05/16 17:06**

1.12.1) Funktionen erstellen und aufrufen

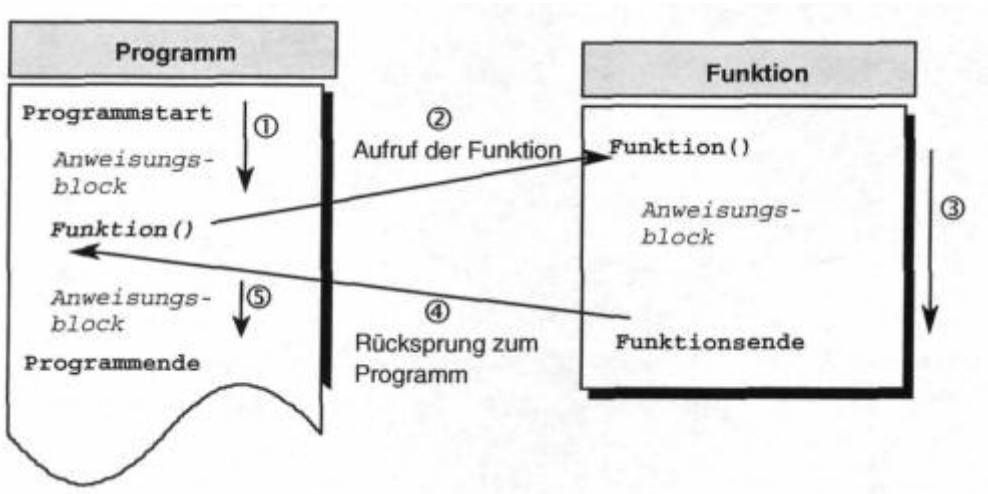
Was sind Funktionen?

Funktionen sind eigenständige Programmteile, die vom Skript beliebig oft aufgerufen und abgearbeitet werden können. Funktionen beinhalten Anweisungen, die innerhalb des Programms mehrmals benötigt werden. Anstatt die Anweisungen mehrfach im Programm zu codieren, wird die entsprechende Funktion aufgerufen, die die gewünschten Anweisungen durchführt.

PHP bietet eine Reihe vordefinierter Funktionen, um bestimmte Standardaufgaben zu lösen. Funktionen, die Sie selbst zur Lösung Ihrer Aufgaben erstellen, werden benutzerdefinierte Funktionen genannt.

Vorteile von Funktionen

- Immer wiederkehrende Abläufe werden nur einmal beschrieben und können danach beliebig oft ausgeführt werden.
- Der Programm-Code wird mithilfe von Funktionen strukturiert und ist dadurch einfacher zu pflegen.
- Änderungen am Programm, die zu einem späteren Zeitpunkt auftreten, lassen sich schneller durchführen, da sich eine Änderung in der Funktion auch auf den Quellcode auswirkt, der diese Funktion verwendet.
- Die Struktur des Programms lässt sich leichter nachvollziehen, da der Quellcode übersichtlicher ist.



<tab>

Eine Funktion wird erst ausgeführt, wenn sie im Programm aufgerufen wird. Dies geschieht über den Namen der Funktion.
Das PHP-Programm wird bis zum Aufruf einer Funktion abgearbeitet (1).
Der Funktionsaufruf (2) erzwingt einen Sprung in die angegebene Funktion.
Jetzt werden die Anweisungen der Funktion abgearbeitet (3).
Mit dem Verlassen der Funktion wird zurück zum Programm gesprungen (4).
Das Programm wird weiter ausgeführt (5).

Eine Funktion erstellen

Syntax und Beschreibung der `function`-Anweisung

```
function Name([Parameter])
{Anweisungsblock;
}
```

- Die function-Anweisung leitet eine Funktion ein.
- Name ist die Bezeichnung der Funktion und sollte einen Bezug zum Inhalt haben, z. B. können Sie eine Funktion, die das Quadrat einer Zahl berechnet, *BerechneQuadratzahl* oder *Quadratzahl* nennen.
- Für den Namen einer Funktion gelten folgende Regeln:
  - Das erste Zeichen muss entweder ein Buchstabe oder ein Unterstrich sein.
  - Der Name darf nur aus Buchstaben und Ziffern bestehen, aber keine Umlaute oder „ß“ und keine Sonderzeichen außer dem Unterstrich „_“ enthalten.
  - Der Name kann Groß- oder Kleinbuchstaben enthalten, wobei zwischen Groß- und Kleinschreibung unterschieden wird: `function Quadratzahl()` ist nicht gleich `function quadratzahl()`.
  - Der Name darf nicht identisch sein mit einem so genannten reservierten Wort (z. B. Befehl aus PHP).
- In den runden Klammern werden die Bezeichnungen der einzelnen Parameter angegeben, für die beim Aufruf der Funktion Werte übergeben werden können.
- Eine Funktion kann keinen, einen oder mehrere Parameter besitzen.

Eine Funktion mit retum-Anweisung erstellen

Syntax und Beschreibung der `return`-Anweisung

- Mit der `return`-Anweisung kann ein Wert zurückgeliefert werden. Sobald die `return`-Anweisung ausgeführt wird, wird eine Funktion verlassen und an die aufrufende Stelle zurückgekehrt.
- Die `return`-Anweisung kann ohne Rückgabewert auch ausschließlich zum Verlassen einer Funktion verwendet werden.

```
function Name([Parameter])
{Anweisungsblock;
 return [$Wert];
}
```

Eine Funktion aufrufen

Syntax und Beschreibung eines Funktionsaufrufs

- Eine Funktion können Sie aus einer beliebigen Stelle des PHP-Codes heraus aufrufen.
- Sie können eine Funktion auch aus einer anderen Funktion heraus aufrufen.
- Eine Funktion wird mit ihrem Namen aufgerufen.

- Die runden Klammern nach dem Funktionsnamen können, falls benötigt, Parameter zur Übergabe von Werten enthalten. Falls keine Werte übergeben werden, bleiben die Klammern leer.

```
<?php
  Anweisungsblock

  Funktionsname();

  Anweisungsblock
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_40

Last update: **2019/05/16 17:06**

1.12.2) Mit Funktionen arbeiten

Je nach Verwendungszweck einer Funktion kann eine Funktion mit oder ohne Parameter bzw. mit oder ohne Rückgabewert erforderlich sein.

Funktionen ohne Parameter

Bei einer Funktion ohne Parameter werden bei jedem Aufruf dieselben Anweisungen ausgeführt.

Beispiel: funktionen1.php

```php
<?php

 function Text()
 {
    echo "<p>Hallo Welt!</p>";
 }

 function Linie()
 {
    echo "<p><hr></p>";
 }

 Text();
 Linie();
 Text();
 Linie();

?>
```



Funktionen mit einem oder mehreren Parametern

Bei einer Funktion mit einem oder mehreren Parametern werden bei jedem Aufruf der Funktion in Abhängigkeit von den Parametern ähnliche Aufgaben ausgeführt. Hierbei kann ein Parameter eine Konstante oder eine Variable sein.
Wenn Sie eine Funktion mit mehreren Parametern aufrufen, müssen die Anzahl und die Reihenfolge der Parameter übereinstimmen.

Beispiel: funktionen2.php

Im folgenden Skript werden die Preise pro kg mithilfe einer Funktion mit einem Parameter berechnet.

```
<?php

 function Preis($menge)
 {
    $betrag = 1.2*$menge;
    echo "Der Preis von $menge kg ist: $betrag<br>";
 }

 Preis(2);
 Preis(5);
 Preis(10);
 $einkaufsmenge=20;
 Preis($einkaufsmenge);
?>
```

Die Funktion `Preis()` wird dreimal aufgerufen. Der übergebene Parameter (der Wert in der Klammer) ist bei jedem Aufruf ein anderer. Der Wert in der Klammer wird der Variablen `$menge` übergeben. Mithilfe der Variablen `$menge` wird beim Aufruf der Funktion die Variable `$betrag` berechnet und am Bildschirm ausgegeben.



Beispiel: funktionen3.php

Wird eine Funktion mit mehreren Parametern aufgerufen, wird der erste Wert an den ersten Parameter, der zweite Wert an den zweiten Parameter usw. übergeben.

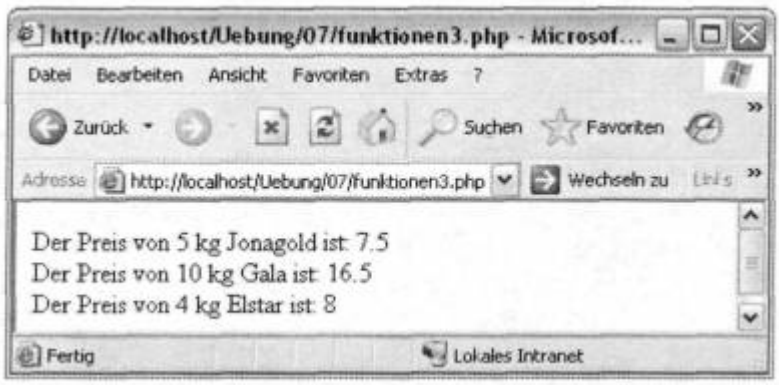
```
<?php
```

```
function Preis($sorte,$preis,$menge)
{
   $betrag = $preis * $menge;
   echo "Der Preis von $menge kg $sorte ist: $betrag<br>";
}

Preis(Jonagold,1.5,5);
Preis(Gala,1.65,10);
Preis(Elstar,2.00,4);

?>
```

Die Funktion `Preis()` wurde mit 3 Parametern versehen: `$sorte, $preis, $menge`.
Beim Aufruf der Funktion werden durch Kommata getrennt drei Parameter übergeben.



Veränderung von Parametern innerhalb einer Funktion

Wenn Sie eine Variable als Parameter an eine Funktion übergeben, dann erhält die Funktion eine Kopie der übergebenen Variablen. Die Übergabe der Parameter als Kopie wird auch als **call-by-value** bezeichnet. Um den Wert einer übergebenen Variablen zu ändern, verändern Sie nur ihre lokale Kopie. Eine Veränderung der Kopie hat keinerlei Rückwirkung auf das Original.

Manchmal ist aber eine Veränderung der übergebenen Variablen erwünscht, beispielsweise um den Preis einer Bestellung abhängig zu machen von der vorgegebenen Menge. Um dies zu erreichen, übergeben Sie den Parameter als Referenz auf das Original. Sie übergeben also nicht, wie bei dem call-by-value-Verfahren, den Inhalt bzw. den Wert einer Variablen, sondern einen Verweis auf die Variable. Diese Vorgehensweise wird **call-by-reference** genannt. Eine Veränderung hat, anders als bei der Übergabe als Kopie, Rückwirkung auf das Original. Eine Referenz auf das Original erhalten Sie, indem Sie vor den Variablennamen ein & setzen, z. B. `&$preis`.

Beispiel: byreference.php

```
<?php

 function Preis($sorte, &$preis, $menge)
 {if ($menge > 10)
  {$preis = 1;
  }
  $betrag = $preis * $menge;
  echo "Der Preis von $menge kg $sorte ist: $betrag EUR <br>";
```

```
 }

 $sorte = Elstar;
 $preis = 2;
 $menge = 8;
 Preis($sorte, &$preis, $menge);

 $sorte = Elstar;
 $preis = 2;
 $menge = 15;
 Preis($sorte, &$preis, $menge);
?>
```

- Die Funktion Preis() wurde mit 3 Parametern versehen: `$sorte,&$preis,$menge`. Die Variable `$preis` wird mithilfe des vorangestellten & als Referenz gekennzeichnet.
- Mithilfe einer `if`-Schleife wird festgelegt, dass der Wert der Variablen `$preis = 1`, wenn `$menge > 10` ist. Beim Aufruf der Funktion werden drei Parameter durch Kommata getrennt übergeben. Da der Wert der Variablen `$menge < = 10` ist, wird zur Preisberechnung der ursprüngliche Wert der Variablen `$preis = 2` verwendet. Wenn aber der Wert der Variablen `$menge > 10` ist, wird der Wert der Variablen `$preis` aufgrund der Bedingung in der `if`-Schleife auf 1 verändert.

Funktionen mit Rückgabewerten

Wenn Sie innerhalb einer Funktion ein Ergebnis ermitteln und dies an den Aufrufer zurückliefern möchten, verwenden Sie die `return`-Anweisung.
Allerdings ist hier zu beachten, dass eine return-Anweisung nicht nur den Rückgabewert festlegt, sondern auch die Ausführung der Funktion sofort beendet. Alle Anweisungen nach einer `return`-Anweisung werden nicht mehr ausgeführt, und der vorhandene Rückgabewert wird an das Programm ausgegeben.

Beispiel: funktionen4.php

```
<?php
 function Summe ($zahll, $zahl2)
 {
     $ergebnis = $zahll + $zahl2;
     return $ergebnis;
 }

 $summel = Summe(10,20);
 echo "Summe1: $summel<br>";
 $zahl3 = 0.11;
 $summe2 = Summe (19,$zahl3);
 echo "Summe2: $summe2<br>";
?>
```

- Die Funktion `Summe()` hat die beiden Parameter `$zahl1` und `$zahl2`.
- Die beiden Parameter werden innerhalb der Funktion addiert und in der Variablen `$ergebnis` gespeichert.

- Der Wert der Variablen `$ergebnis` wird mithilfe der Anweisung return an die aufrufende Stelle zurückgeliefert.
- Der Funktionsaufruf `Summe(l0,20)` liefert in `$ergebnis` den Rückgabewert zurück. Dieser Wert wird der Variablen `$summe` zugewiesen.
- Parameter für Funktionen können sowohl als konstante Werte, als auch über Variablen übergeben werden.

PHP kann nur einen Wert mit `return()` zurückgeben. Wenn Sie mehrer Werte aus einer Funktion zurückgeben möchten, können Sie diese Werte in einem Array zusammenfassen.

Um Variablen die Elemente des zurückgelieferten Arrays zuzuweisen, verwenden Sie die `list()`-Anweisung.

```php
<?php

function zahlen()
{
   $ret = array(2,4,6);
   return $ret;
}

list($var1,$var2,$var3) = zahlen();
echo "Die zurückgelieferten Variablen sind: $var1, $var2, $var3";
?>
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_41

Last update: **2019/05/16 17:06**

1.12.3) Der Gültigkeitsbereich von Variablen

Auf Variablen, die Sie außerhalb einer Funktion im PHP-Programm einsetzen, können Sie in Funktionen nicht ohne weiteres zugreifen und mit ihnen arbeiten. Ebenso gibt es innerhalb einer Funktion Variablen, auf die Sie außerhalb der Funktion nicht zugreifen können.

Variablen haben, je nachdem, wo sie definiert sind, unterschiedliche Gültigkeitsbereiche.

- **Lokale Variablen** sind nur innerhalb der Funktion, in der sie definiert wurden, gültig.
- **Globale Variablen**, die außerhalb von Funktionen definiert wurden, haben standardmäßig nur außerhalb der Funktionen ihren Gültigkeitsbereich. Die globalen Variablen stehen innerhalb einer Funktion nicht zur Verfügung. Sollen die globalen Variablen innerhalb einer Funktion gültig sein, müssen sie dort mit dem Schlüsselwort `global` bekannt gemacht werden, z. B. `global $zahl`.
- **Superglobale Variablen** stehen sowohl innerhalb als auch außerhalb von Funktionen zur Verfügung, beispielsweise das assoziative Feld `$_POST`.

Im folgenden Beispiel werden globale und superglobale Variablen gegenübergestellt. Hierzu wird ein Bestell-Formular für Äpfel erstellt. Anhand des PHP-Codes im Auswertungsprogramm wird gezeigt, wann welche Variablen gültig sind.

Beispiel: formular_funktion.html, funktion_var.php

```
<html>
 <body>
  <h2>Apfelkauf</h2>
  Bitte geben Sie die gewünschte Menge ein und wählen Sie eine Apfelsorte:
  <form action="funktion_var.php" method="post">
   Menge: <input name="menge"><br><br>
   Apfelsorte:<br>
   <input type="radio"  name="sorte" value="Jonagold">Jonagold
   <input type="radio"  name="sorte" value="Gala">Gala
   <input type="radio"  name="sorte" value="Elstar">Elstar<br><br>
   <input type="Submit" value="Abschicken">
   <input type="Reset"  value="Zurücksetzen">
  </form>
 </body>
</html>
```

- Mithilfe der Methode POST werden nach Absendung des Formulars folgende Variablen an das Auswertungsprogramm `funktion_var.php` weitergegeben:
- die Variable `$menge`, die vom Benutzer eingetragen wird, und
- die Variable `$sorte`, die der Benutzer mithilfe eines Optionsfeldes auswählt




```php
<?php
 $bestellnummer = "TT1911";
 $bearbeiter = "Mario Caspari";

 function bestellung()
 {
  global $bestellnummer;      // globale Variable
  echo "Bestellnummer: " .$bestellnummer ."<br>";
  echo "Bearbeiter: " .$bearbeiter ."<br>";
  echo "Die von Ihnen eingegebene Menge: ".$_POST["menge"]." kg<br>";
  switch($_POST["sorte"])
  {
     case "Jonagold": $preis=$_POST["menge"]*1.50;  // lokale Variable
                      break;
     case "Gala":     $preis=$_POST["menge"]*1.65;  // lokale Variable
                      break;
     case "Elstar":   $preis=$_POST["menge"]*2.00;  // lokale Variable
                      break;
  }
  echo "Ausgabe des Preises innerhalb der Funktion: $preis<br>";
 }

 echo "Die von Ihnen gewählte Sorte: ".$_POST["sorte"]."<br>";
 bestellung() ;
 echo "Ausgabe des Preises außerhalb der Funktion: $preis<br>";
?>
```

- Die Variablen `$bestellnummer` und `$bearbeiter` werden als **globale** Variablen im Programm definiert.
- Die **globale** Variable `$bestellnummer` wird innerhalb der Funktion `bestellung()` mithilfe des Schlüsselwortes `global` bekannt gemacht.
- Die Variablen `$bestellnummer` und `$bearbeiter` sollen innerhalb der Funktion ausgegeben werden. Da die Variable `$bearbeiter` innerhalb der Funktion nicht mithilfe des Schlüsselwortes `global` bekannt gemacht wurde, wird sie bei der Ausgabe nicht angezeigt.

- Der Wert des **superglobalen** Feldes `$_POST["menge"]` wird innerhalb der Funktion `bestellung()` ausgegeben.
- Innerhalb der Funktion `bestellung()` wird der Wert der **lokalen** Variablen `$preis` berechnet.
- Die **lokale** Variable `''$preis` wird innerhalb der Funktion ausgegeben.
- Der Wert des **superglobalen** Feldes `$_POST ["sorte"]` wird außerhalb der Funktion `bestellung()` ausgegeben.
- Die Funktion `bestellung()` wird aufgerufen.
- Die **lokale** Variable `$preis` soll auch außerhalb der Funktion ausgegeben werden.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_42

Last update: **2019/05/16 17:06**

1.12.4) PHP-Dateien einbinden mit include und require

Manchmal ist es sinnvoll, wenn Sie selbst definierte Funktionen in eine separate PHP-Datei auslagern, um sie in mehreren Skripten nutzen zu können.

Mithilfe der `include`- bzw. `require`-Anweisung können Sie Dateien, z.B. Funktionen, die Sie benötigen, in Ihr PHP-Programm einbinden.

Mit include und require arbeiten

Die `include()`- bzw. die `require()`-Anweisung binden eine bestimmte Datei, deren Pfad als Argument übergeben wird, in den aktuellen Programmcode ein.

Unterschiede zwischen include und require

Der Unterschied zwischen den beiden Anweisungen zeigt sich, wenn eine Datei fehlerhaft eingebunden wurde:

- Wenn beispielsweise die angegebene Datei nicht vorhanden ist, erzeugt `include()` eine Warnung. Das Skript wird weiter ausgeführt.
- Der Befehl `require()` beendet bei einer fehlenden Datei das Skript sofort mit einer Fehlermeldung.

Syntax und Bedeutung der include()- und require()-Anweisungen

- Die include- und require-Befehle erwarten als Parameter die einzubindende Datei. Befindet sich die Datei nicht im selben Ordner, muss der Pfad angegeben werden.
- Beinhaltet die eingebundene Datei keinen PHP-Code, wird der Inhalt unverändert an den Browser weitergegeben und somit am Bildschirm dargestellt.
- Ist in der Konfigurationsdatei `php.ini` im Bereich `fopen wrappers` die Option `allow_url_fopen = on` gesetzt, so können Sie auch einen URL als Parameter angeben.
- Eingebundene Dateien, die eine Funktion mit Rückgabewert enthalten, werden ebenso behandelt wie Funktionen mit Rückgabewert.

So kann z. B. der Rückgabewert in einer Variablen gespeichert werden mit: `$variable = include(Datei)` .

Innerhalb der eingebundenen Dateien muss der PHP-Code von gültigen PHP-Tags `<?php … ?>` umschlossen sein. Befindet sich kein PHP-Code in den Dateien, können die Tags weggelassen werden.

Um Skripte, die später in einen PHP-Code eingebunden werden sollen, zu kennzeichnen, ist es empfehlenswert, die Dateierweiterung `.inc.php` zu vergeben. Die Bezeichnung `.inc.php` steht hierbei für die Abkürzung von include.

Datei mit include() einbinden

Beispiel: include.php

Zur Veranschaulichung der Arbeitsweise der include()-Anweisung erstellen Sie ein Loginverfahren. Die Prüfung, ob der eingegebene Nickname und das Kennwort korrekt sind, wird in die Datei `prueflogin.inc` codiert. Vom Loginformular `formular_login.html` wird die Datei `include.php` aufgerufen.

Datei: include.php

```
<html>
<body>
 <h4> Loginergebnis über include </h4>
 <?php
  echo "Start<br>Nun wird ein PHP-Skript eingebunden:<hr>";
  echo "Die Rückgabe der Datei ergibt:<br>";
  include ("prueflogin.inc");
  echo "<hr>Hier können weitere Anweisungen der Datei include.php ausgeführt
  werden.";
 ?>
</body>
</html>
```

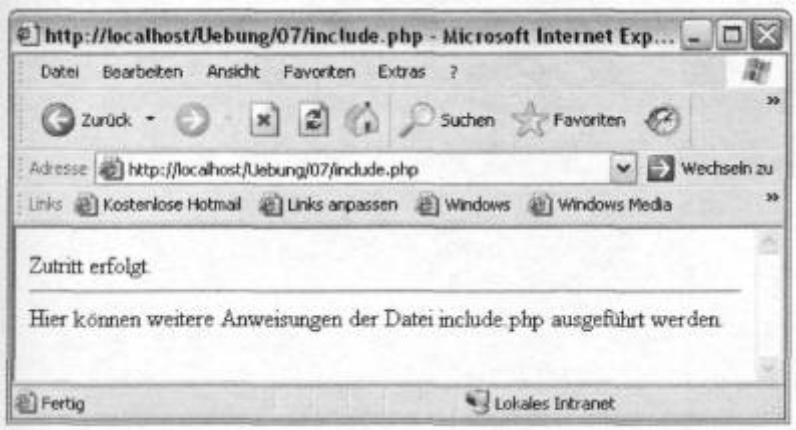
Die `include()`-Anweisung bindet die Datei `prueflogin.inc` ein.

Datei: prueflogin.inc

Innerhalb der Datei prueflogin.inc wird die Kennwortprüfung vorgenommen.

```
<?php
if ($_POST["nickname"] == "Mario" && $_POST["kennwort"] == "geheim")
  echo "Zutritt erfolgt";
else
  echo "Zutritt erfolgt nicht";
?>
```

<tab>



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_43

Last update: **2019/05/16 17:06**

1.12.5) Andere Dateitypen einbinden

Sie können auch andere Dateien, z. B. reine Text- oder HTML-Dateien, in Ihr Programm einbinden.

Innerhalb eines PHP-Skripts soll beispielsweise eine einfache Textdatei eingebunden werden. Der Inhalt der Textdatei wird sofort am Bildschirm ausgegeben. Zuvor wird eine Textdatei mit folgendem Inhalt gespeichert.

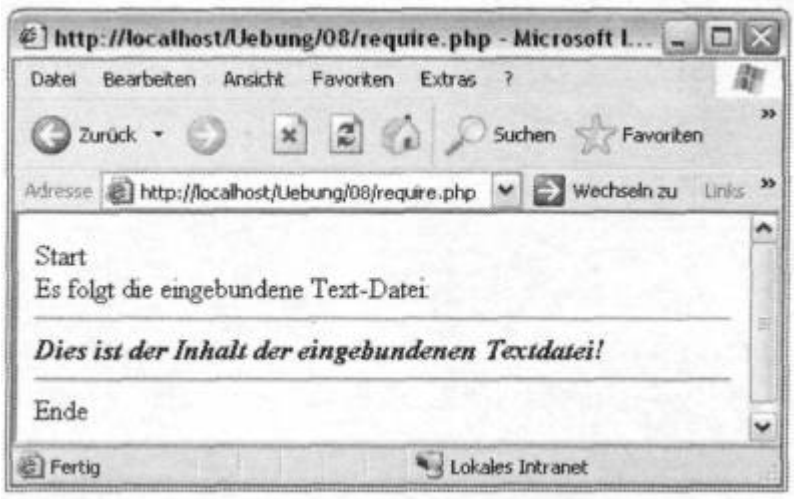
Datei: einfueg.txt

```
<b><i>Dies ist der Inhalt der eingebundenen Textdatei!</i></b>
```

Beispiel: require.php

```
<html>
<body>
 <h4>Eine Textdatei über den Befehl require einbinden</h4>
 <?php
  echo "Start<br>Es folgt die eingebundene Text-Datei:<hr>";
  require("einfueg.txt") ;
  echo "<hr>Ende";
 ?>
</body>
</html>
```

Inmitten des PHP-Codes wird über den Befehl require () eine Textdatei eingebunden. Da diese keinen weiteren PHP-Code, sondern nur Text mit HTML-Befehlen beinhaltet, wird diese von PHP direkt an den Browser weitergeleitet.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_44

Last update: **2019/05/16 17:06**

1.12.6) Übungen zu Funktionen

Aufgabe 27 - Maximum

Erstelle eine PHP-Seite (mit einer Funktion) welches folgende Aufgabe realisiert:

```
1. Zahl: -Benutzereingabe-
2. Zahl: -Benutzereingabe-
größte Zahl: -gibt die größere der beiden Zahlen aus
```

Aufgabe 28 - Bruch

Schreibe eine PHP-Seite, in der ein Bruch soweit als möglich gekürzt wird. Dazu gibt der Benutzer einen Zähler und Nenner ein. Mit Hilfe einer Funktion, die den ggT der beiden Zahlen mit dem euklidischen Algorithmus berechnet, soll der Bruch gekürzt und wieder ausgegeben werden.

Aufgabe 29 - Quader

Schreibe eine PHP-Seite, in der alle möglichen Berechnungen für einen Quader durchgeführt werden.

- Lese dazu vom Benutzer die Länge, Breite und Höhe ein.
- Überprüfe ob alle Werte eingegeben wurden
- Überprüfe mit is_numeric() ob die eingegebenen Werte Zahlen sind
- Berechne in einer Funktion die Grundfläche
- Berechne in einer Funktion das Volumen
- Berechne in einer Funktion die Oberfläche
- Berechne in einer Funktion die Raumdiagonale
- Speichere bei jedem Programmaufruf, die Ergebnisse wie folgt in eine CSV-Datei.

Bspielhafte CSV-Datei:

```
Laenge;Breite;Hoehe;Grundflaeche;Volumen;Oberflaeche;Raumdiagonale
3;5;3;15;45;78;6.557;
3;2;2;6;12;32;4.123;
```

- Gib am Ende jeweils die ganze CSV-Datei als Tabelle aus!

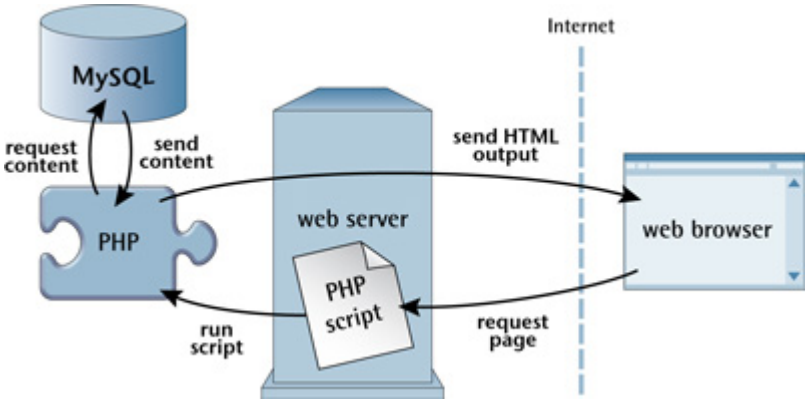
Laenge	Breite	Hoehe	Grundflaeche	Volumen	Oberflaeche	Raumdiagonale
3	5	3	15	45	78	6.577
3	2	2	6	12	32	4.123
...						

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:12:12_45

Last update: **2019/05/16 17:06**

SQL-Anbindung via PHP



- SQL Anbindung mit PHP
  - 13.1.1) Übungen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:13

Last update: **2019/05/16 17:13**

2.8) SQL-Datenbankzugriff mit PHP

Um auf eine MySQL-Datenbank zuzugreifen, muss zuerst eine Verbindung (***mysqli_connect***) hergestellt werden. Dazu ist der **Host**, auf dem der MySQL-Server läuft, der **Benutzername** sowie das **Kennwort** anzugeben. Anschließend muss noch eine Datenbank auf dem DBS ausgewählt werden (***mysqli_select_db***). Danach können die SQL-Anweisungen folgen (***mysqli_query***). Im Falle einer SQL-Abfrage liefert diese Funktion alle Tupel zurück, welche der Reihe nach mit ***mysqli_fetch_array*** abgearbeitet werden können.

Im Folgenden soll als Beispiel ein Gästebuch realisiert werden. Dazu wurde eine Tabelle **meldung** erstellt, welche die einzelnen Nachrichten aufnehmen soll:

```
CREATE TABLE meldung (
  id INT NOT NULL AUTO_INCREMENT,
  datum DATETIME DEFAULT NULL,
  name VARCHAR(200) DEFAULT NULL,
  eintrag TEXT,
  CONSTRAINT PK_id PRIMARY KEY (id)
);
```

Zusätzlich zu den Daten **datum, name** und **eintrag** wurde ein Attribut **id** eingeführt, welches als Schlüssel dient.

Das folgende Skript **show.php** liest alle Einträge der Datenbank aus und gibt sie in Tabellenform aus:

```
<?php
    $DBHost = "127.0.0.1";
    $DBUser = "root";
    $DBPasswd = "";
    $DBName = "db";

    //Verbindung zu DB-Server herstellen
    $con=mysqli_connect($DBHost, $DBUser, $DBPasswd, $DBName)
      OR die("Konnte DB-Server nicht erreichen!");
    mysqli_select_db($con,$DBName);
?>
<html>
    <head>
        <title>Alle Meldungen</title>
    </head>
    <body>
    <?php
        $erg=mysqli_query($con, "SELECT datum, name, eintrag FROM meldung
ORDER BY datum DESC");
        echo mysqli_error($con);

        while($row=mysqli_fetch_array($erg))
        {
            echo "<table bgcolor=\"#dddddd\" border=\"0\" width=\"600\">
```

```
\n";
            echo "<tr><td>Datum: </td><td>".$row["datum"]."</td></tr> \n";
            echo "<tr><td>Name:</td><td>" .$row ["name"]. "</td></tr> \n";
            echo "<tr><td valign=\"top\">Eintrag:</td> \n";
            echo "<td>".nl2br(htmlentities($row["eintrag"]))."</td></tr>
\n";
            echo "</table>\n<br>\n";
        }
    ?>
    <hr><a href="insert.php">neuen Eintrag hinzufügen</a>
    </body>
</html>
```

Datum:	2018-11-15 20:19:20
Name:	Prof. Andreas Lahmer
Eintrag:	Hier sehen sie die SQL-Anbindung mittels PHP in Form eines Gästebuchs!

---

neuen Eintrag hinzufügen

Um neue Einträge für das Gästebuch zu erstellen, existiert ein weiteres Skript insert.php:

```
<?php
    $DBHost = "127.0.0.1";
    $DBUser = "root";
    $DBPasswd = "";
    $DBName = "db";

    //Verbindung zu DB-Server herstellen
    $con=mysqli_connect($DBHost, $DBUser, $DBPasswd)
      OR die("Konnte DB-Server nicht erreichen!");
    mysqli_select_db($con,$DBName);
?>
<html>
    <head>
        <title>Neuer Eintrag in unser Gästebuch</title>
    </head>
    <body>
        <?php

            if(isset($_GET["submit"]))
            {
                $submit = $_GET["submit"];
                $name = $_GET["name"];
                $eintrag = $_GET["eintrag"];

                //Der Submit-Button wurde gedrückt --> die Werte müssen
überprüft werden
                //und bei Gültigkeit in die DB eingefügt werden
```

```
                $DatenOK=1;         //wir gehen prinzipiell von der
Gültigkeit der Daten aus
                $error="";         //es gab noch keine Fehlermeldung bis hier
hier

                if($name=="")    //Kein Name eingegeben
                {
                    $DatenOK=0;
                    $error.="Es muss ein Name eingegeben werden!<br>\n";
                }
                if($eintrag=="")    //Kein Kommentar eingegeben
                {
                    $DatenOK=0;
                    $error.="Ein Eintrag ohne Kommentar macht nicht viel
Sinn!<br>\n";
                }
                if($DatenOK)        //Daten OK -> also in DB eintragen
                {
                                        $timestamp=date("Y-m-d h:i:s",
time());
                    mysqli_query($con,"INSERT INTO eintraege (datum, name,
eintrag) VALUES (\"$timestamp\", \"$name\", \"$eintrag\" );");
                    echo mysqli_error($con);
                    echo "<b>Daten wurden eingetragen.<b>";
                }
                else
                {
                    echo "<h2>Fehler: </h2>\n"; //Fehlermeldung
                    echo $error;
                }
            }

            //Formular
        ?>



        <form action="insert.php" method="GET">
            Name: <input type="text" name="name" size="30" maxlength="200"
value=""><br>
            Text: <br><textarea rows="10" cols="50" wrap="virtual"
name="eintrag"></textarea>
            <br><input type="submit" name="submit" value="Absenden">
        </form>
        <a href="show.php"> Alle Einträge anzeigen</a>
    </body>
</html>
```

Name:

Text:

Absenden

Alle Einträge anzeigen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:13:13_01

Last update: **2019/05/16 17:13**

Aufgabe 1

Erstellen Sie eine php-Seite welche dem Benutzer ein Formular (siehe Abbildung) präsentiert. In diesem Formular kann der Benutzer auswählen, welche der 6 Aufgaben (a bis f) aus der DQL-Übung angezeigt werden sollen.



Ergebnis der SQL-Abfrage

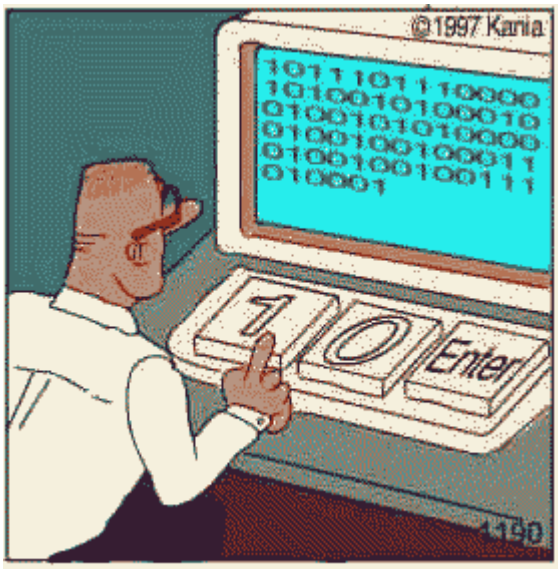
name	durchschnittspreis	gesamtanzahl
Weingut F.X.Pichler	23.50	60
Weingut Prager	21.00	30
Weingut Emmerich Knoll	19.00	15
Lackner Tinnacher	12.50	127
Freie Weingärten Wachau	9.90	80
Weingut Biegler	9.00	16

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:13:13_01:13_01_01

Last update: **2019/05/16 17:13**

14) Algorithmik und Programmierung



Real programmers code in binary.

- 14.1) Grundlagen zur Programmierung
- 14.2) Programmiersprachen
- 14.3) Compiler & Interpreter
- 14.4) Programmierstile
- 14.5) Visualisierung der Programmlogik/von Algorithmen
- 14.6) Code.org
- 14.7) Scratch
- 14.8) C++

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14

Last update: **2019/05/16 17:15**

6.1) Grundlagen zur Programmierung

Programm

Im Alltag sind uns Programme geläufig, wie z.B.:

- das Programm einer Feier,
- ein Kochrezept,
- oder das Programm einer politischen Partei.

Diese beschreiben einen Ablauf d.h. was wie zu tun ist.
In der Informatik wird der Begriff enger gefasst und man spricht von der Programmierung.

Programmierung

Unter Programmierung versteht man das Schreiben solcher Programme in Form von Quellcode d.h. Befehle in einer Text Datei.

Das Verfassen von Quellcode erfolgt in einer Programmiersprache (C++, Java, Python, Fortran, …)

Diese Programmiersprachen werden von Softwareprogrammen in Maschinensprache übersetzt, welche ein Computer versteht und ausführen kann.

Somit kann man mit Programmierung das Verhalten eines Computers steuern.

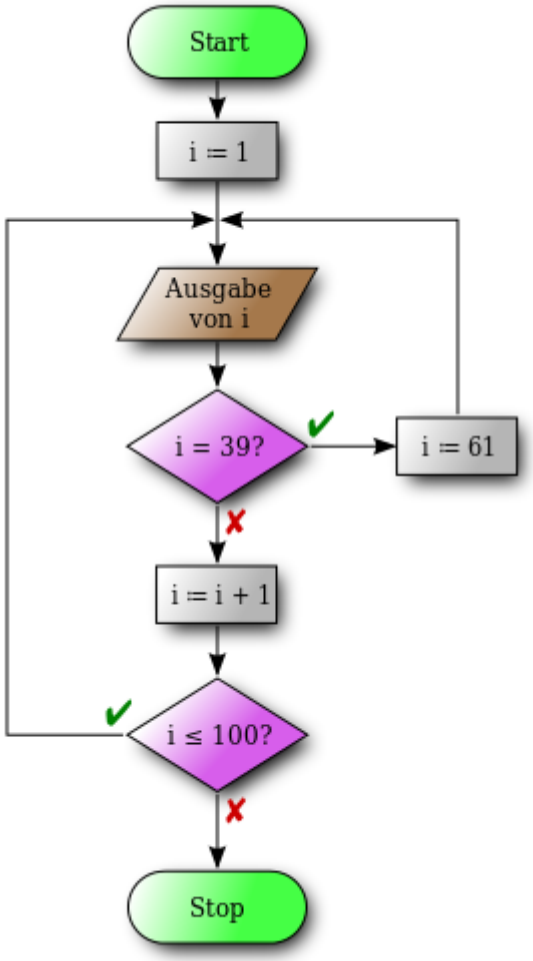
Algorithmus

Ein Algorithmus ist eine eindeutige Abfolge von Anweisungen (Befehlen) zur Lösung eines Problems.

Beispiel eines Algorithmus:

```
ALGORITHMUS mach_was()
   ANFANG mach eine Aufgabenliste
   SOLANGE was zu tun ist
   WENN nicht erledigt
      mach weiter
   SONST
      geh zur nächsten Aufgabe
   WIEDERHOLE
ENDE
```

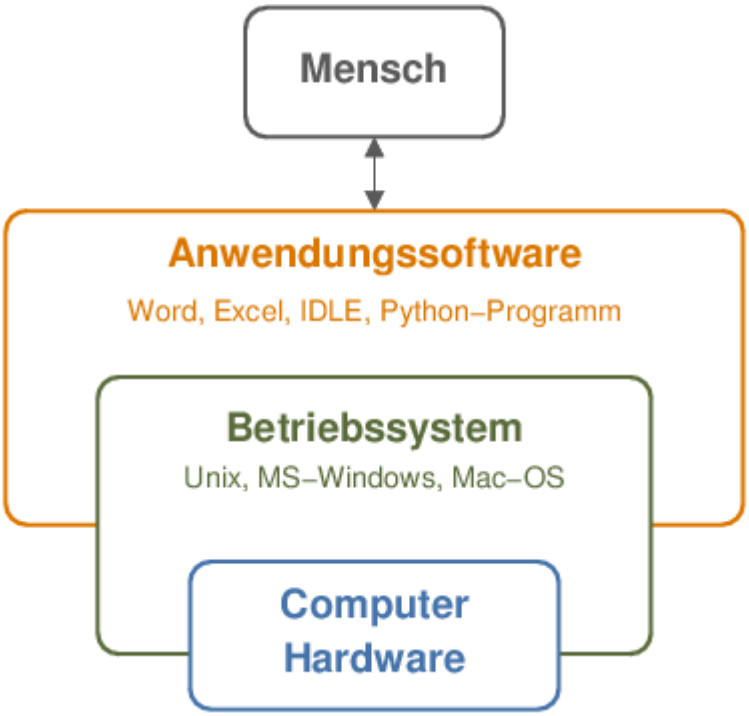
Ein Algorithmus kann auch graphisch in Form eines Flussdiagramms dargestellt werden.



Ein Programm besteht im Allgemeinen aus vielen Algorithmen!

Computersystem

Ein Computersystem kann als Schichtmodell gesehen werden.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_01

Last update: **2019/05/16 17:16**

6.2) Programmiersprachen

Eine Programmiersprache zeichnet sich - wie jede Sprache - durch Syntax und Semantik aus.

Syntax

Legt fest in welcher Folge Zeichen den Programmtext bilden.

Z.B. Python-Syntax zur Definition einer Variablen:

```
a = 1.843
```

Nicht erlaubt wäre z.B.:

```
a 1.843
```

Bei anderen Programmiersprachen würde dies wie folgt aussehen:

```
real a = 1.843     # Fortran
float a = 1.843;   # C++, Java
$a = 1.843;        # Perl
```

Semantik

Beschreibt die Bedeutung eines Programmtextes

```
print "Alles Python oder was?"
```

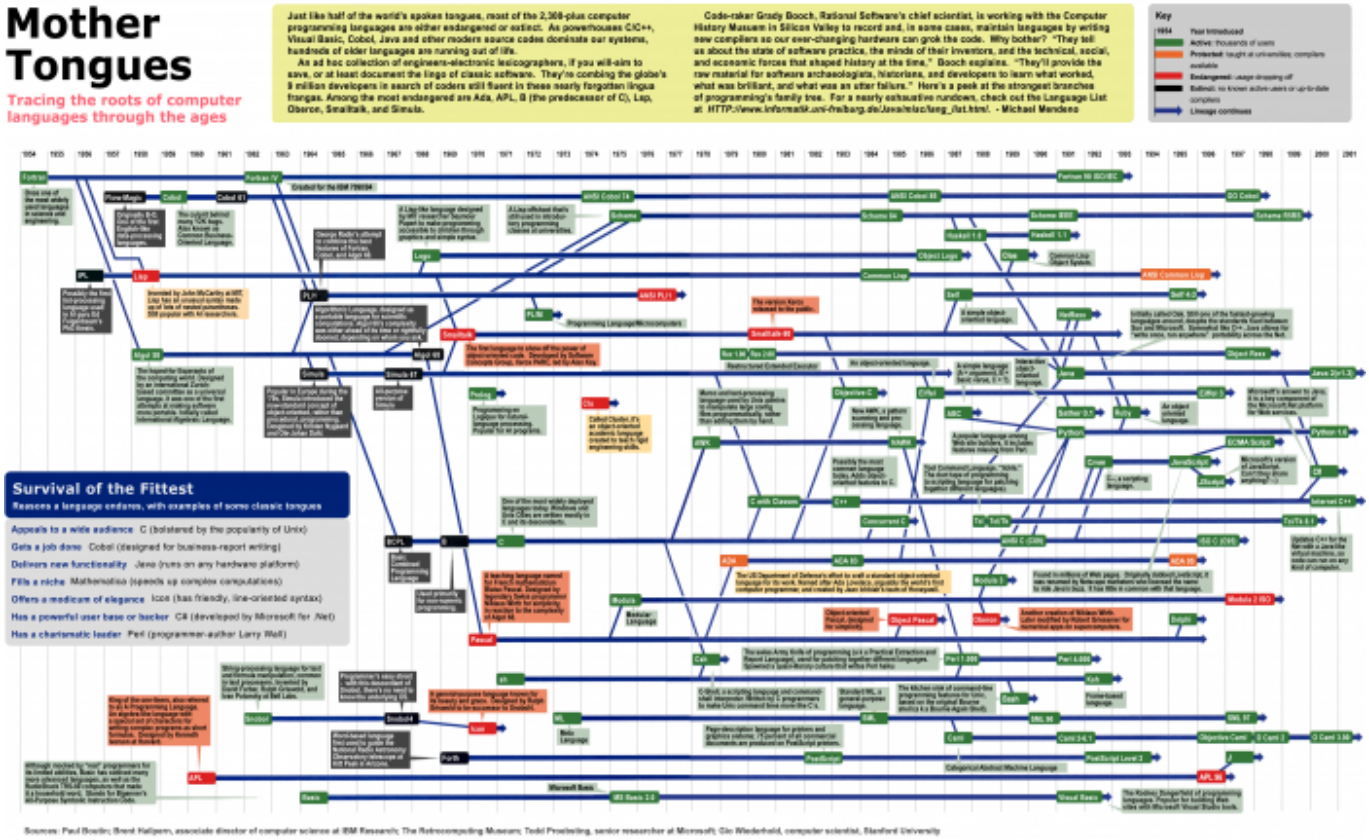
Dieser Programmtext gibt einen Text am Bildschirm aus. Ein weiteres Beispiel :

```
if a < 0 :
  a = a*(-1)
```

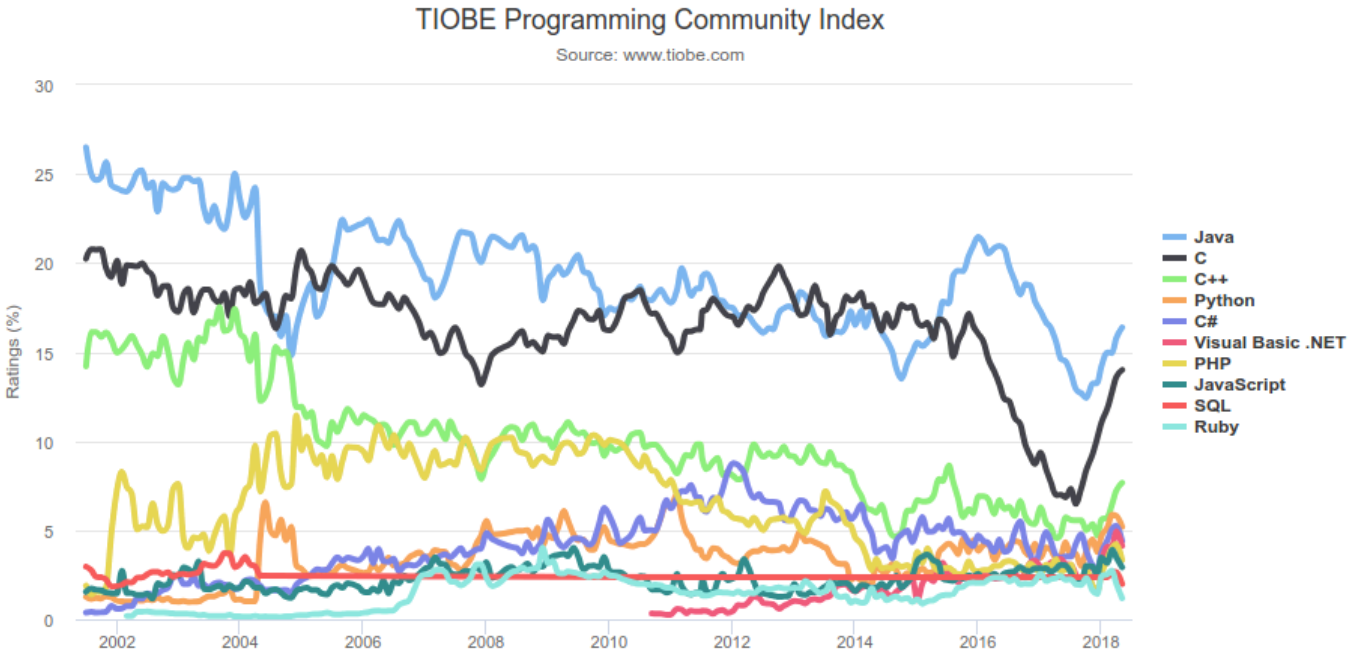
Dieser Code macht aus a eine positive Zahl.

Wichtige Programmiersprachen

Die Liste der Programmiersprachen ist lang. Anbei eine Abbildung über die Entwicklung wichtiger Programmiersprachen.



Anbei noch eine Abbildung über die Popularität von Programmiersprachen



Aber auch Experten wie **Bjarne Stroustrup** - der **Erfinder von C++** meint dazu im Video zu den 5 wichtigsten Programmiersprachen:



Video

… Experten kennen mehr als nur eine Sprache, kennt man diesen Cluster, kennt man alle anderen!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_02

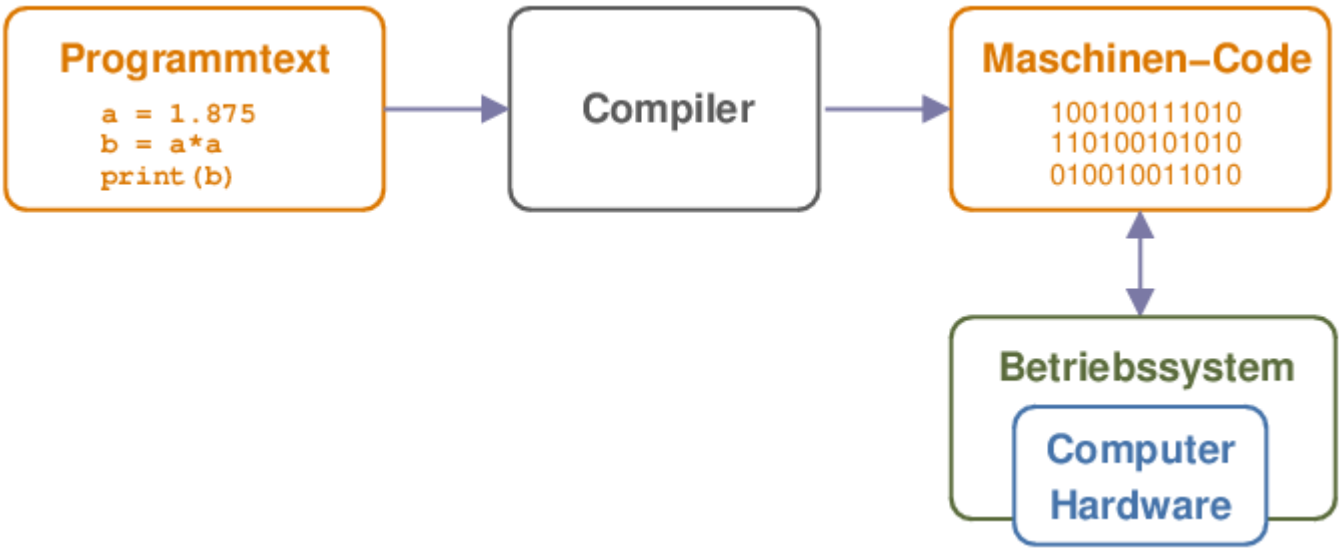
Last update: **2019/05/16 17:16**

6.3) Übersetzung von Quellcode in Maschinensprache

Es gibt **2 unterschiedliche Methoden** wie **Programmtext (Quelltext, source code) in Computersprache übersetzt** wird d.h. vom Computer **verstanden** wird.

Compiler

Der Programmtext wird nach Erstellung komplett in Maschinen-Code (executeable) übersetzt (siehe Arbeitsweise eines Compilers) und ausgeführt.

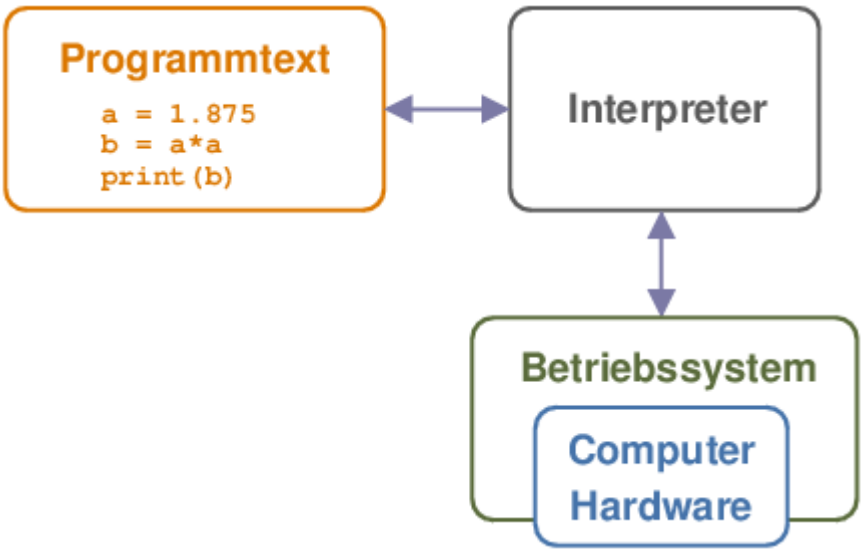


Vor- und Nachteile

- benötigt Compiler (=eigene Anwendungssoftware)
- läuft nur auf dem Betriebssystem, wo Programm kompiliert wurde
- i.a. sehr schnell (high performance)
- Quelltext nach kompilieren nicht einsehbar
- typische Sprachen: Fortran, C++

Interpreter

Der Programmtext wird Zeile für Zeile an das Betriebssystem geschickt und ausgeführt (siehe Arbeitsweise eines Interpreters).



Vor- und Nachteile

- benötigt Interpreter (=eigene Anwendungssoftware)
- unabhängig vom Betriebssystem
- schnell implementiert, langsam exekutiert
- Quelltext einsehbar, schnell änderbar
- typische Sprachen: JavaScript, Python

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_03

Last update: **2019/05/16 17:16**

6.4) Programmierstile

Quellcode kann **imperativ (z.B. prozedural)** oder **objektorientiert** geschrieben werden.

Imperativ

Kommt aus dem lateinischen imperare = anordnen d.h. einzelnen Befehle werden **sequentiell hintereinander** geschrieben mittels:

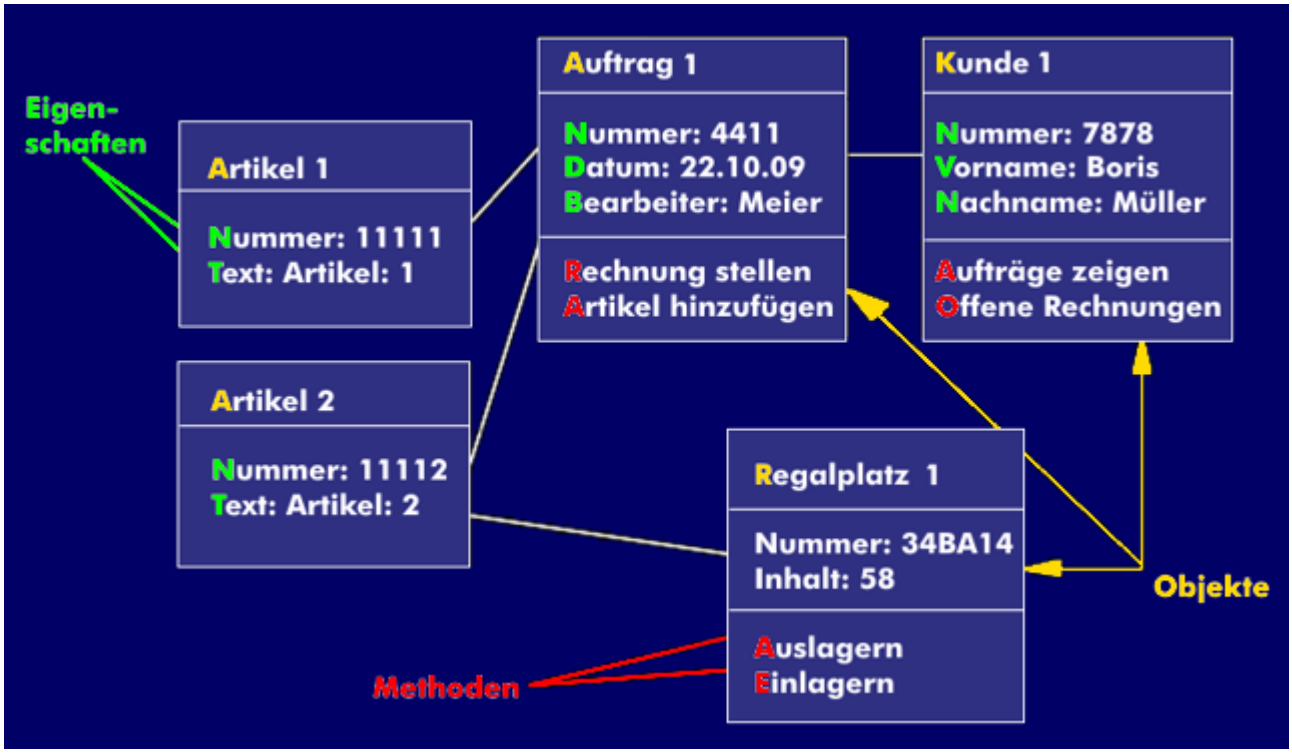
- Kontrollstrukturen = Strukturierte Programmierung
- Überschaubare Unterprogrammen = Prozedurale Programmierung
- Logisch zusammengefassten Unterprogrammen = Modulare Programmierung

Vor- und Nachteile

- für kleine Programme geeignet
- i.a. sehr schnell, da Compiler den Code gut optimieren kann
- wird schnell unübersichtlich, schwer wartbar, ...

Objektorientiert

Das Programm wird als **System von Objekten** dargestellt, die **miteinander kommunizieren** (siehe Beispiel objektorientierte Programmierung).



Erklärungen:

- Zunächst definiert man **sogenannte Klassen** (z.B. Klasse “Artikel”). Ein Programm kann beliebig viele unterschiedliche Klassen haben (Artikel, Auftrag, Kunde, ...).
- Erzeugt man nun einen bestimmten “Artikel” (z.B. Artikel 1, anders gesagt gibt man diesen spezifische Daten), spricht man von einem **Objekt**.
- Diesen Vorgang bezeichnet man auch als **das Objekt (Artikel 1) wird von der Klasse (Artikel) instantiiert**. Man kann viele Objekte einer Klasse erzeugen (Artikel 1, Artikel 2, ..).
- Jedes Objekt besitzt (anhand der Klassendefinition) **Eigenschaften (Attribute)**, dies sind die Daten (z.B. Nummer, Text, ...) und **Methoden (Algorithmen)**, welche diese Daten ändern, anzeigen, bereitstellen, etc (z.B. auslagern, einlagern, ...).
- Ein Hauptprogramm enthält z.B. alle hier dargestellten Objekte in Form von Listen und steuert den Programmablauf inklusive Daten Ein- und Ausgabe.

Vor- und Nachteile

- für große Programmierprojekte gut geeignet
- Konzepte wie Vererbung, Datenkapselung, Polymorhismus, etc umsetzbar

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_04

Last update: **2019/05/16 17:16**

Darstellung von Algorithmen bzw. von Programmlogiken

Es gibt verschiedene Darstellungsformen, um algorithmische Grundbausteine und Algorithmen darzustellen. An dieser Stelle soll auf die drei wichtigsten eingegangen werden:

- Pseudocode
- Struktogramm
- Programmablaufplan

Pseudocode

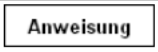
Der Pseudocode stellt einen **Algorithmus in einer Schreibweise** in der Art einer Programmiersprache dar, die aber in der Regel **näher an der natürlichen Sprache ist als am formalen Programmcode**. Es gibt **keine Standardisierung bzw. Vorschrift**, wie ein Pseudocode auszusehen hat; die Ablaufstrukturen werden grob durch Texte und fest vorgeschriebene Schlüsselwörter beschrieben. Guter Pseudocode zeichnet sich durch eine **kurze präzise Beschreibung des Algorithmus** aus.

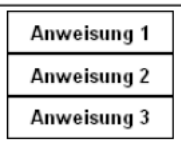
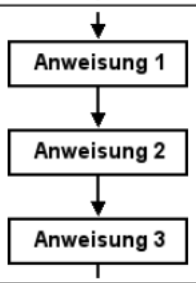
Struktogramm

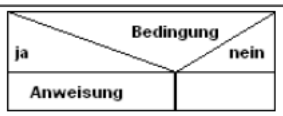
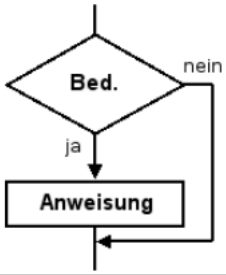
Struktogramme (auch nach ihren Entwicklern **Nassi-Shneidermann-Diagramme** genannt) ist eine **genormte (DIN 66261) Entwurfsmethode für Algorithmen**. Die Methode **zerlegt ein Gesamtproblem**, das man mit dem gewünschten Algorithmus lösen will, in immer **kleinere Teilprobleme** bis schließlich nur noch **elementare Grundstrukturen wie Sequenzen und Kontrollstrukturen** zur Lösung des Problems übrig bleiben.

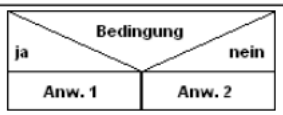
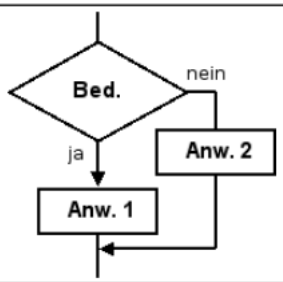
Programmablaufplan (PAP)

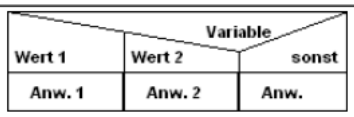
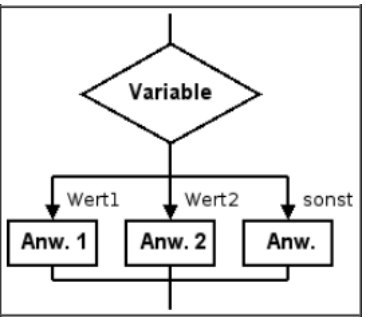
Der **Programmablaufplan (auch Flussdiagramm oder Programmstrukturplan)** ist eine **graphische Darstellung zur Umsetzung eines Algorithmus** in einem Programm und beschreibt die **Folge von Operationen zur Lösung einer Aufgabe**. Die Symbole für Programmablaufpläne sind in der DIN 66001 genormt.

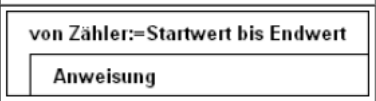
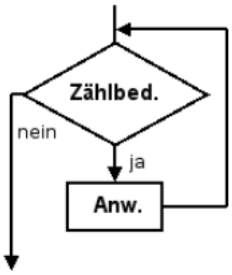
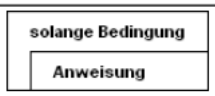
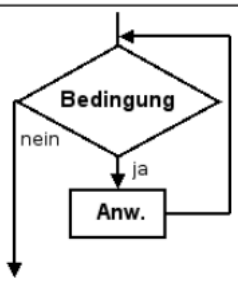
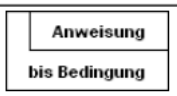
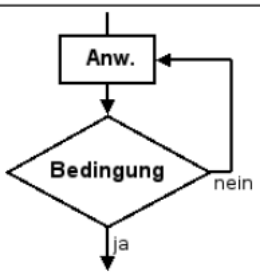
Strukturelement	Pseudocode	Struktogramm	Programmablaufplan
Verarbeitung (Elementarblock)	`Anweisung`		

Reihenfolge (Sequenz)	`Anweisung1`
	`Anweisung2`
	`Anweisung3`
			

Verzweigung (Alternative)	`wenn Bedingung` `  dann Anweisung`		

	`wenn Bedingung`
	`  dann Anweisung1`
	`  sonst Anweisung2`
			

Fallunterscheidung	für Variable
	  Wert 1 : Anweisung1
	  Wert 2 : Anweisung2
	  ...
	  [sonst Anweisung]
			

Zählschleife	von Zähler:=Startwert bis Endwert Anweisung		
anfangsgeprüfte Schleife	solange Bedingung Anweisung		
endgeprüfte Schleife	wiederhole Anweisung bis Bedingung		

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_05

Last update: **2019/05/16 17:16**

Code.org

Code.org ist eine 2013 in den USA gegründete Non-Profit-Organisation und gleichnamige Website, die möglichst viele Menschen und insbesondere Kinder für das Thema Informatik (im englischen „Computer Science“) und Programmieren begeistern will.

- Kurs 3
- Kurs 4

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_06

Last update: **2019/05/16 17:16**

Scratch

Mit Scratch kannst du deine eigenen interaktiven Geschichten, Spiele und Animationen programmieren und deine Kreationen mit anderen in der Gemeinschaft online teilen.

Scratch hilft jungen Leuten, kreativ zu denken, systematisch zu schließen und miteinander zusammenzuarbeiten — grundlegende Fähigkeiten für das Leben im 21. Jahrhundert.

Beispielhafte Scratch-Projekte

- Scratch-Games

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_07

Last update: **2019/05/16 17:16**

C++ bzw. CPP

C + + (zeh plus plus ausgesprochen) ist eine kompilierte Allzweck Programmiersprache. Ihre Komplexität liegt zwischen Einsteiger und Fortgeschritten, da sie sowohl High-Level als auch Low-Level Sprachelemente vereint. Sie bietet imperative, objekt-orientierte und generische Programmiermerkmale.

C + + ist eine der beliebtesten Programmiersprachen und ist auf vielen Hardwareplattformen und Betriebssystemem verfügbar. Da sie besonders effizient ist, wird es für Systemprogramme, Applikationen, Gerätetreiber, eingebettete Software, Server-Client und Unterhaltungsprogramme, z. B. Videospiele, eingesetzt. Verschiedene Hersteller bieten Open Source und proprietäre C + + Compiler an, bspw. FSF, LLVM, Microsoft und Intel.

Entwicklungsumgebung DEV C++

- Download DEV C++

Los gehts!

- Einstieg in C++
- WikiBook C++
- Learn C++
- C++ spielerisch lernen

Programmierung in C++

- 14.8.1) Grundgerüst
- 14.8.2) Variablen
- 14.8.3) Datentypen
- 14.8.4) Verarbeitung
- 14.8.5) Ein- und Ausgabe
- 14.8.6) Ablaufsteuerung

Übungen

- 14.8.6.3) Übungen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08

Last update: **2019/05/16 17:21**

6.8.1) Grundgerüst eines Programms

Ein **Programm** ist eine **Aneinanderreihung von Befehlen**, die dem Computer zur Abarbeitung zugeführt werden können. Wie schon erwähnt, wird ein **C++-Programm in einem Texteditor** geschrieben und dann von einem **Compiler in ausführbaren Code übersetzt**. Danach kann es gestartet werden.

Hauptprogramm - main()

Ein C++-Programm besteht **mindestens** aus der **Funktion main()**. Wichtig ist, dass Sie in **jedem C++-Programm irgendwo den Namen main** mit einem **Klammerpaar** finden werden. In manchen Fällen stehen **zwischen den Klammern Variablen für die Parameterübergabe**. Aber auch das sollte Sie zu Anfang nicht stören. Ignorieren Sie es von ganzem Herzen, bis Sie wissen, wozu Sie es brauchen.
Die **Hauptfunktion beginnt direkt nach einer öffnenden geschweiften Klammer und endet mit der schließenden Klammer**. **Zwischen den geschweiften Klammern** stehen die **Befehle der Hauptfunktion**. Sie werden nacheinander ausgeführt. Diese Hauptfunktion enthält nichts und ist damit das kleinste denkbare C++ Programm.

```
int main()   //Hauptfunktion main()
{

   return 0;  //Optionale Rückgabe an das Betriebssystem
}
```

Dieses Programm tut gar nichts. Wenn es etwas täte, stünden die Befehle zwischen den geschweiften Klammern.

Kommentare

Wenn wir schon einmal ein Programm haben, das nichts tut, wollen wir es auch um **Befehle** erweitern, **die nichts tun: KOMMENTARE.**

Freundlichkeit unter KollegInnen

Sie können in das Programm Kommentare einfügen, die **vom Compiler völlig ignoriert** werden. Der **Zweck** solcher Kommentare ist es zu **dokumentieren**, warum das Programm so und nicht anders geschrieben wurde. Die **Kommentare richten sich an Programmierer**. Dies können **Kollegen** sein, die Ihr **Programm korrigieren** oder **erweitern** sollen. Aber noch öfter helfen Sie sich selbst damit. In der Praxis ist es so, dass Sie vermutlich auch herangezogen werden, wenn eines Ihrer Programme nicht korrekt läuft oder ergänzt werden soll. Da Sie zwischendurch andere Programme geschrieben haben werden, vielleicht geheiratet haben, umgezogen sind und noch drei weitere Programmiersprachen gelernt haben, werden Sie sich **nicht mehr an jedes Detail erinnern**. Sie werden dankbar sein, wenn Sie in den Programmen ausführliche Hinweise finden, wie und warum es

so funktioniert oder zumindest funktionieren soll.

Es gibt **zwei Arten**, einen Text davor zu schützen, dass der Compiler versucht, ihn als Programm zu interpretieren.

Zeilenweise

Die einfachere Art der Kommentierung ist es, **zwei Schrägstriche** direkt hintereinander zu setzen. Damit gilt der **Rest der Zeile als Kommentar**.

```
int main()
{
    // Hier beginnt der Kommentar.
    // Die naechste Zeile braucht ihr
    // eigenes Kommentarzeichen.

    return 0;
}
```

Kommentarblock

Daneben gibt es die Möglichkeit, einen **größeren Text in Kommentarklammern einzuschließen** und damit dem Compiler zu entziehen. Der Anfang des Kommentars wird durch den Schrägstrich, gefolgt von einem Stern, festgelegt. Der Kommentar endet mit der umgekehrten Zeichenfolge, also mit einem Stern und dann einem Schrägstrich.

```
int main()
{
  /* Hier beginnt der Kommentar.
     Die naechste Zeile braucht kein
     eigenes Kommentarzeichen.
   */

   return 0;
}
```

!! ACHTUNG !! : Eine Verschachtelung von Kommentarblöcken ist nicht möglich!

```
int main()
{
  /* Hier beginnt der Kommentar
     /*
       Die naechste Zeile braucht kein
       eigenes Kommentarzeichen
      */
     Dies wird der Compiler wieder uebersetzen wollen.
  */
```

```
  return 0;
}
```

Anweisungen

Anweisungen sind die **Befehle, aus denen ein Programm** besteht. Eine Anweisung kann dazu dienen, etwas zu **speichern**, zu **berechnen**, zu **definieren** oder auf dem Bildschirm **auszugeben**. **Anweisungen sind die Grundeinheiten**, aus denen **Programme bestehen**.

Blöcke

Zusammenfassen von Anweisungen

Mehrere Anweisungen können **zu einem Block zusammengefasst** werden. Ein Block wird durch **geschweifte Klammern** eingefasst. Der **Compiler** wird einen **Block wie eine einzige Anweisung** behandeln. Sicher ist Ihnen schon aufgefallen, dass die Hauptfunktion main() ebenfalls solche **geschweiften Klammern** hat. In der Tat ist dies der Block, in dem die Befehle des Programms zusammengefasst werden.

Einrückungen

Um die **Übersicht zu erhöhen**, pflegt man **alle Befehle innerhalb eines Blocks einzurücken**. Achten Sie vor allem am Anfang darauf, dass die **zusammengehörigen geschweiften Klammern auf einer Ebene** stehen. Das folgende Beispiel mit Pseudobefehlen zeigt, wie das Einrücken korrekt ist.

```
int main()
{
    Hier sind Pseudo-Anweisungen;
    Diese gehört dazu;
    {
        Neuer Block, also einrücken;
        Wir bleiben auf diesem Niveau;
        Das kann ewig weitergehen;
    }
    Nun ist die Klammer geschlossen;
    Und die Einrückung ist auch zurück;

    return 0;
}
```

Hello World!

Es ist eine **alte Tradition**, eine **neue Programmiersprache** mit einem **„Hello-World“-Programm**

einzuweihen. Auch dieses Buch soll mit der Tradition nicht brechen, hier ist das „Hello-World“-Programm in C++:

```
#include <iostream>                                    // Ein- und
Ausgabebibliothek

int main(){                                            // Hauptfunktion
    std::cout << "Hallo, du schöne Welt!" << std::endl; // Ausgabe

    return 0;                                          // Optionale
Rückgabe an das Betriebssystem
}
```

Namensraum - namespace std

Die C++-Standardbibliotheken verwenden für alle Funktionen und Variablen den Namensraum std. Bei Verwendung von Typen, Funktionen oder Variablen aus den Standardbibliotheken müssten Sie jeweils std:: voranstellen. Sie können aber auch mit dem Befehl **using** erklären, dass Sie den Namensraum **std** verwenden wollen, als seien alle darin definierten Variablen und Funktionen ohne Namensraum definiert. Die Anweisung dazu lautet:

```
using namespace std;
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_01

Last update: **2019/05/16 17:16**

6.8.2) Variablen

Die folgenden 4 Aussagen erläutern die Notwendigkeit von Variablen:

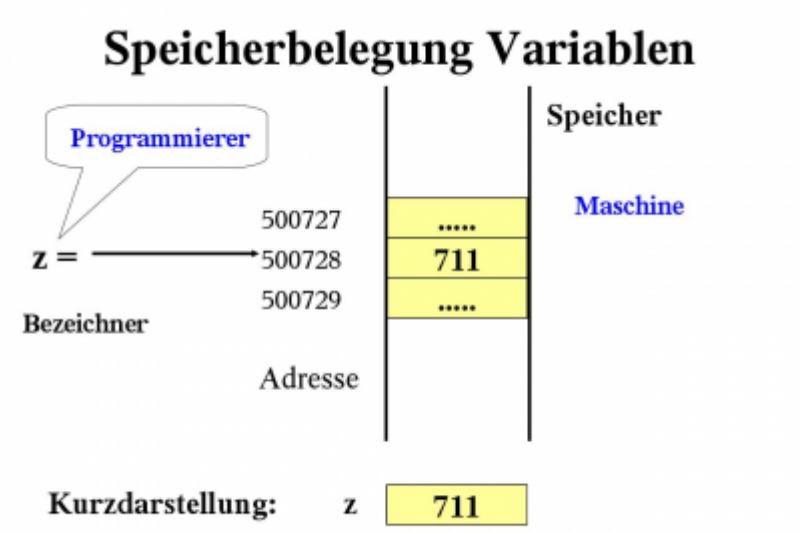
- Jedes Programm verarbeitet Informationen.
- Diese Informationen liegen im Speicher.
- Höhere Programmiersprachen greifen nicht direkt auf den Speicher zu, sondern verwenden Variablen.

⇒ **Variablen sind also die Behälter, in denen das Programm Zahlen und Texte ablegt.**

Eigenschaften einer Variable

SPEICHER

Die **Variable benötigt** zum Ablegen der Informationen immer **Speicher**. Der Speicher wird **vom Compiler zur Verfügung gestellt**. Über die **Lage und Größe muss sich der Programmierer NICHT kümmern**. Der Compiler ermittelt die Größe vom Datentyp und reserviert den dafür notwendigen Speicher.



NAME

Die Variable wird im Programm über einen **weitgehend frei wählbaren Namen** angesprochen. Dieser **Name** identifiziert die Variable **innerhalb eines Blocks eindeutig**. C++ unterscheidet sogar zwischen **Groß- und Kleinschreibung**!

DATENTYP

Der **Datentyp** einer Variablen **bestimmt, welche Informationen abgelegt werden können**. So kann eine Variable je nach ihrem Typ beispielsweise einen Buchstaben oder eine Zahl speichern. Der **Datentyp bestimmt** natürlich auch die **Speichergröße**, die benötigt wird. Der Typ bestimmt aber auch, **welche Operationen auf die Variable angewendet werden können**. So können Sie beispielsweise Zahlen durch einander dividieren. Mit Texten geht das nicht.

Variablendefinition

Das folgende Beispiel zeigt eine Variablendefinition innerhalb der Hauptfunktion main():

```
int main()
{
   int Gehalt;
}
```

Datentyp

Der Datentyp ist in diesem Fall **int**. Dieser Datentyp steht für ganzzahlige Zahlen.

Name

Durch ein Leerzeichen abgesetzt, beginnt der Name der Variablen. Den **Namen** sollten Sie immer so wählen, dass Sie **auf den Inhalt schließen können**. Hier lässt der Name Einkommen bereits auf die Verwendung der Variablen im Programm schließen. **Verwenden Sie ruhig lange Namen**. Abkürzungen ersparen zwar Tipparbeit, Sie werden aber auf lange Sicht mehr Zeit verschwenden, wenn Sie darüber nachdenken müssen, was in welcher Variable abgelegt ist. Das gilt auch dann, wenn Sie extrem langsam tippen. Noch mehr Zeit werden Sie brauchen, wenn Sie einen Fehler suchen müssen, der entstand, weil Sie zwei Variablen verwechselt haben, nur weil der Name unklar war.

Semikolon (=Strich-Punkt)

Zu guter Letzt folgt das Semikolon. **Damit wird jede Anweisung abgeschlossen, auch eine Variablendefinition.**

Initialisierung

Sie können **Variablen gleich bei ihrer Definition mit einem Wert vorbelegen**. Dazu setzen Sie hinter den Variablennamen ein Gleichheitszeichen. Das **initialisiert die Variable mit dem nachfolgenden Wert**.

```
   int Gehalt=2000;
```

Damit wird bei der Erstellung die Variable Gehalt auf 2000 gesetzt. Normalerweise initialisiert man

einen neue Variable immer mit 0.

```
   int Gehalt=0;
```

Mehrfache Definition

Es können mehrere Variablen mit dem gleichen Datentyp hintereinander definiert werden, in dem sie durch Komma getrennt werden.

```
   int zahl, Gehalt=0, Freunde=1, Feinde=5;
```

Geltungsbereich einer Variable

Eine Variable in C++ kann je nach ihrer Definition unterschiedliche Geltungsbereiche besitzen.

Blockgrenzen

Während **in C** Variablendefinitionen **nur am Blockanfang vor der ersten Anweisung** erlaubt sind, kann in **C++ eine Variable überall definiert werden**.

WICHTIG: Sie **gilt** dann **für den gesamten restlichen Bereich des aktuellen Blocks und aller darin verschachtelten Blöcke**.

Es können sogar **in verschachtelten Blöcken Variablen mit dem gleichen Namen** verwendet werden. Dabei **überdeckt die innen liegende Definition diejenige, die außerhalb des Blocks liegt**. Folgendes Beispiel macht das deutlich:

```
int main()
{
    int a = 5;
    {
    // hier hat a den Inhalt 5
    }
    {
        int a = 3;
        // hier hat a den Inhalt 3
        {
            // a ist immer noch 3
        }
    }
    // hier ist a wieder 5

    return 0;
}
```

Lokale Variable

Die **innere Variable wird als lokale Variable** bezeichnet. Für ihren Geltungsbereich **überdeckt sie die außen liegende Variable a**. Die **außen liegende Variable existiert durchaus noch**, aber **aufgrund der Namensgleichheit** mit der lokalen Variablen **kann man bis zu deren Auflösung nicht auf sie zugreifen**. Sobald das Programm den Block verlässt, in dem die lokale Variable definiert ist, wird die äußere Variable wieder sichtbar. **Alle Operationen auf der lokalen Variablen berühren die äußere Variable nicht**.

Globale Variable

Variablen, die **außerhalb jedes Blocks definiert** werden, nennt man globale Variablen. Sie **gelten für alle Blöcke**, die nach der Definition im Quelltext auftreten, **sofern darin nicht eine Variable gleichen Namens lokal definiert wird**.

Konstante

Eine Konstante ist ein **unveränderlicher Wert**. Alle Zahlen in einem Programm sind Konstanten, da ihr Wert, im Gegensatz zu Variablen, nicht geändert werden kann. In C++ haben **auch Konstanten einen Datentyp**.

Deklaration einer Konstanten

Zahlenkonstanten die ohne weitere Erklärung im Programm (=Quellcode) stehen, haben keinen Dokumentationswert und können zu Verwechslungen bzw. Verwirrungen führen. Um die Lesbarkeit des Programms zu erhöhen, könnte man anstatt der Zahl eine Konstante definieren.
Nehmen Sie an, sie möchten ein Programm für die Zeiterfassung ihrer Mitarbeiter schreiben. Jeder Mitarbeiter soll jeden Tag 8 Stunden arbeiten. Anstatt die Zahl 8 zu verwenden, könnten Sie eine Konstante namens Regelarbeitszeit verwenden.
Ist diese Konstante einmal definiert so kann sie bei Programmausführung nicht mehr verändert werden.
Notwendig dafür, ist neben dem Datentyp und dem Variablennamen noch das Schlüsselwort **const**.

```
   const int Regelarbeitszeit=8;
```

Manchmal findet man in C++ auch noch die Form aus der Programmiersprache C.

```
  #define Regelarbeitszeit 8
```

Beispiel für Variablen und ihren Geltungsbereich

```
#include <iostream>         //Bibliothek für Ein- und Ausgabe

#define Wochentage 7      //Alte Definition einer Konstanten namens
Wochentage mit dem Wert 7
```

```
int Arbeitsstunden=8;       //globale Variable

int main()       //Hauptfunktion
{
    const int Arbeitstage=5;       // Deklaration einer Konstanten namens
Arbeitstage initialisiert mit 5
    std::cout << Arbeitsstunden;      //Ausgabe der globalen Variable
Arbeitsstunden => 8
    {
        int Arbeitsstunden=5;         //Deklaration einer lokalen Variable
namens Arbeitsstunden initialisiert mit 5
        std::cout << Arbeitsstunden;        //Ausgabe der lokalen Variable
Arbeitsstunden => 5
    }

    std::cout << Arbeitsstunden;  //Ausgabe der globalen Variable
Arbeitsstunden => 8
    //Arbeitstage=4;                  --> NICHT ERLAUBT, DA Arbeitstage EINE
KONSTANTE IST
    std::cout << Arbeitstage;     //Ausgabe der Konstante Arbeitstage => 5
    std::cout << Wochentage;      //Ausgabe der Konstante Wochentage => 7

    return 0;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_02

Last update: **2019/05/16 17:16**

6.8.3) Datentypen

C++ bietet viele eingebaute Datentypen als auch Möglichkeiten Datentypen selbst zu definieren.

In C++ gibt es prinzipiell 7 grundlegende Datentypen:

- bool ⇒ true oder false
- char ⇒ Buchstabe (8 Bit)
- int ⇒ Ganze Zahl (32 Bit)
- float ⇒ Fließkommazahl (32 Bit)
- double ⇒ Fließkommazahl doppelte Genauigkeit (64 Bit)
- void ⇒ nichts
- enum ⇒ Aufzählung

In der Standard Template Library (STL) ⇒ Standardbibliothek sind dann noch weitere Datentypen enthalten, wie z.B.:

- string ⇒ Zeichenketten
- complex ⇒ Komplexe Zahlen
- vector ⇒ Ähnlich wie Arrays
- valarray ⇒ Vektorrechnung
- list ⇒ Listen
- map ⇒ Assoziative Felder (Wörterbuch)

Wahrheitswerte - bool

Der **Datentyp für Wahrheitswerte heißt in C++ bool**, was eine **Abkürzung für boolean** ist. Er kann nur **zwei Zustände** annehmen: **true (wahr)** oder **false (falsch)**. Obwohl eigentlich 1 Bit ausreichen würde, hat bool mindestens eine Größe von einem Byte (also 8 Bit), denn 1 Byte ist die kleinste adressierbare Einheit und somit die Minimalgröße für jeden Datentyp.

```
bool betrunken=false;
```

Zeichen - char

Zeichen sind eigentlich Ganzzahlen. Sie unterscheiden sich von diesen nur bezüglich der Ein- und Ausgabe. **Jeder Zahl ist ein Zeichen zugeordnet**. Mit den Zahlen lässt sich ganz normal rechnen aber bei der Ausgabe erscheint das zugeordnete Zeichen auf dem Bildschirm. Welches Zeichen welcher Zahl entspricht, wird durch den verwendeten Zeichensatz festgelegt.

Die meisten Zeichensätze beinhalten den **sogenannten ASCII-Code (American Standard Code for Information Interchange)**, welcher die Zeichen 0 – 127 belegt. Er enthält **32 Steuerzeichen (0 – 31) und 96 druckbare Zeichen (32 – 127)**.

char ist der Standard-Datentyp für Zeichen. Er ist in der Regel **1 Byte** groß und kann somit **256 verschiedene Zeichen** darstellen. Diese genügen für einen erweiterten ASCII-Code, welcher zum

Beispiel auch deutsche Umlaute definiert. Für Unicode-Zeichen gibt es die Datentypen char16_t mit einer Größe von 2 Byte und char32_t mit einer Größe von 4 Byte. Früher nutzte man für Unicode auch den Datentyp wchar_t, welcher je nach System 2 oder 4 Byte groß war. Dieser Datentyp sollte jedoch nicht mehr eingesetzt werden.

Ganzzahlen - int

Dieser Datentyp umfasst sowohl **ganzzahlige positive als auch negative Zahlen** aus dem Wertebereich von -2.147.483.648 bis 2.147.483.647. Es werden also vom Compiler für eine Variable des Datentyps int **32 Bit (4 Byte)** reserviert.

```
int x; // Variablendeklaration; die Variable x soll vom Typ int sein
x = 3; //Wertzuweisung, Initialisierung; die Variable x enthält den Wert 3
```

Fließkommazahlen

In der realen Welt sind **ganzzahlige Werte oft nicht ausreichend**.

Bei Gewichten, Geschwindigkeiten und anderen Werten aus der Physik ist immer mit Nachkommastellen zu rechnen. Dieser Tatsache kann sich auch eine Computersprache nicht entziehen. Eine solche Fließkommazahl besitzt zwei Komponenten.

Die eine ist die **Mantisse**, und die andere ist der **Exponent zur Basis 10**.

Der Exponent wird durch ein kleines oder großes E abgetrennt. Die Zahl **1.5E2** hat den Gegenwert von **150**. Dabei ist **1.5** die **Mantisse**, **2** ist der **Exponent**.

float

Der **einfachste Typ mit Nachkommastellen** heißt **float**. Die Zahlen dieses Typs sind binär kodiert, damit sie effizient im Computer verarbeitet werden können. Die Zahl lässt **ganzzahlige, aber auch negative Exponenten** zu. Dadurch sind nicht nur extrem große Zahlen darstellbar, sondern auch sehr kleine Brüche. Die Speicheranforderung einer float-Variablen ist nicht sehr hoch, typischerweise liegt sie bei **4 Bytes**.

```
float pi=3.14159265;
```

double

Reicht die Genauigkeit nicht aus oder werden Größenordnungen benötigt, die über die Kapazität einer float-Variablen hinausgehen, steht der Typ double zur Verfügung. Der Name bedeutet »doppelt« und bezieht sich auf die **Genauigkeit**. Diese ist **doppelt so hoch wie bei float**.

```
double pi=3.14159265;
```

Nichts - void

Der Datentyp **void** bedeutet **nichts**, ist **kein echter Datentyp** und wird überall dort **verwendet, wo kein Wert benötigt wird** oder vorhanden ist. Bei **Funktionen** wird void verwendet, wenn eine Funktion keinen Wert zurückgibt oder die Funktion keinen Parameter hat.

Aufzählungen - enum

Mit **enum können Aufzählungstypen** definiert werden. Dazu gehören **Wochentage** oder **Farben**. Den **Elementen** eines Aufzählungstyps werden **Namen zugeordnet**, obwohl sie **intern natürlich als Zahlen codiert** werden

```
enum Farbe {ROT, GELB, GRUEN, BLAU};
```

Zeichenketten - string

Die C++-Standardbibliothek enthält eine Klasse namens **string**. Um diese Klasse nutzen zu können, müssen Sie die gleichnamige Headerdatei string einbinden.

```
#include <iostream>  //Bibliothek für Ein- und Ausgabe-Befehle
#include <string>  // Bibliothek für Zeichenketten

using namespace std;

int main() {
    string zeichenkette;
    zeichenkette = "Hallo Welt!";
    cout << zeichenkette << endl;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_03

Last update: **2019/05/16 17:16**

6.8.4) Verarbeitung

Zuweisung

Gleichheitszeichen

Bei der **Initialisierung** wurde **bereits das Gleichheitszeichen verwendet**, um eine **Variable mit einem Wert vorzubelegen**. Das Gleichheitszeichen wird auch ansonsten verwendet, wenn eine Variable einen **neuen Wert bekommen soll**. Dabei gibt es einen feinen Unterschied zwischen der Vorbelegung oder **Initalisierung**, die **beim Anlegen der Variablen** erfolgt, und der **Zuweisung**, wie die **Belegung** mit Werten genannt wird, **wenn die Variable bereits existiert**.

Links vom Gleichheitszeichen steht immer das Ziel der Zuweisung. Im Allgemeinen ist das eine Variable. Auf der **rechten Seite des Gleichheitszeichens steht die Datenquelle**. Das kann eine andere Variable, ein Zahlenwert oder eine Berechnung sein. Man bezeichnet die **Datenquelle auf der rechten Seite** einer Zuweisung **allgemein als Ausdruck**. Die englische Bezeichnung dafür ist **»expression«**. Ein Ausdruck ist Konstrukt, das einen Wert liefert und so als Datenquelle dienen kann. Das folgende Beispiel zeigt mehrere Zuweisungen. Dabei wird auch schon den Rechenoperationen ein wenig vorgegriffen.

```
int main()
{

    //Initialisierungen
    int MWStSatz=0, Netto=0;
    double MWStBetrag=0.0, Brutto=0.0;

    //Zuweisungen
    MWStSatz = 16;
    Netto = 200;
    MWStBetrag = Netto * MWStSatz / 100;
    Brutto = Netto + MWStBetrag;

    return 0;
}
```

Kaskadierende Zuweisung

C++ hat die Besonderheit, dass Sie in einer Anweisung mehreren Variablen den gleichen Wert zuweisen können. Sie müssen sich das so vorstellen, dass eine Zuweisung ihren Wert nach links durchreicht. Diese Fähigkeit ermöglicht eine Zeile wie die folgende:

```
int main()
{
    int a,b,c,d;
```

```
    a = b = c = d = 5 + 2;

    return 0;
}
```

Grundrechenarten

Letztlich sieht es nicht sehr viel anders aus, als wenn Sie sich eine Rechenaufgabe auf einen Zettel schreiben. Etwas ungewohnt ist lediglich, dass auf der linken Seite das Zuweisungsziel und ein Gleichheitszeichen steht. Das **Multiplikationszeichen ist der Stern**, und das **Divisionszeichen der Schrägstrich**. **Plus- und Minuszeichen** sehen so aus, wie man es erwartet. Sie können sogar das **Minuszeichen wie gewohnt als Vorzeichen** verwenden.

Restrechnung bzw. Modulo

Eine besondere Rechenart ist die **Modulo-Rechnung**. Sie liefert den **Rest einer ganzzahligen Division**. Wenn Sie sich daran erinnern, wie Sie in den ersten Schulklassen dividiert haben, dann fallen Ihnen vielleicht noch Sätze ein wie: »25 geteilt durch 7 sind 3, Rest 4«. Diese Restberechnung gibt es auch unter C++. Man bezeichnet sie als die Modulo-Rechnung. Als **Operatorzeichen wird das Prozentzeichen** verwendet.

```
Rest = 25 % 7; // Rest ist also 4
```

Punkt vor Strich

Auch in C++ werden die Gesetze der Mathematik geachtet. So wird die alte Regel **»Punktrechnung geht vor Strichrechnung«** auch hier eingehalten. Diese Regel sagt aus, dass die Multiplikation und die Division vor einer Addition oder Subtraktion ausgeführt werden, wenn beide gleichwertig nebeneinander stehen.

Links nach rechts

Binäre Operatoren, also Rechensymbole, die zwei Ausdrücke miteinander verknüpfen, sind im Allgemeinen **linksbindend**. Das bedeutet, dass sie von links nach rechts ausgeführt werden, wenn die Priorität gleich ist. Das heißt, dass **a*b/c als (a*b)/c** ausgewertet wird. Eine **Ausnahme ist die Zuweisung**. Hier wird **a=b=c als a=(b=c)** ausgewertet. Der **Zuweisungsoperator ist also rechtsbindend**. Auch einstellige Operatoren sind rechtsbindend. Dazu gehört auch der Operator ++, den Sie im Laufe des Abschnitts noch kennen lernen werden.

Abkürzungen

Der Mensch neigt zur Bequemlichkeit. Nicht anders geht es Programmierern. Sie handeln nach dem Grundgedanken, niemals etwas zu tun, was ein Computer für sie tun kann, und so ist es naheliegend,

dass es Wege gibt, wiederkehrende Aufgaben möglichst kurz zu formulieren.

Inkrementieren

Um den Wert einer Variablen um 1 zu erhöhen, können Sie die folgende Zeile schreiben:

```
int Zaehler=0;
Zaehler=Zaehler+1;
```

Sind sie schreibfaul, so funktioniert auch folgende Variante:

```
int Zaehler=0;
Zaehler+=1;
```

Hier wird das Pluszeichen mit dem Gleichheitszeichen kombiniert. Damit das Plus nicht fälschlicherweise als Vorzeichen der 1 interpretiert wird, muss es vor dem Gleichheitszeichen stehen. Zwischen Plus- und Gleichheitszeichen darf kein Leerzeichen stehen. Die Bedeutung der Zeile ist also: **»Addiere der Variablen Zaehler den Wert 1 hinzu.«**

Das geht mit allen Rechenarten:

kurze Schreibweise	lange Schreibweise
a+=b	a=a+b
a-=b	a=a-b
a/=b	a=a/b
a*=b	a=a*b
a%=b	a=a%b

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_04

Last update: **2019/05/16 17:16**

6.8.5) Ein- und Ausgabe

Ausgabestrom nach cout

C++ verwendet für die **Ein- und Ausgabe** das **Datenstrom-Modell**. Der Datenstrom zur Ein- und Ausgabe wird in einer **eigenen Bibliothek (iostream)** behandelt. Um sie zu verwenden, muss vor der ersten Verwendung die folgende Zeile eingegeben werden:

```
#include <iostream.h>
```

In einigen Listings werden Sie auch sehen, dass iostream ohne die Erweiterung .h verwendet wird. Wenn Sie einen halbwegs aktuellen Compiler verwenden, ist beides möglich. Der Unterschied hat mit der Verwendung von Namensräumen zu tun.

Um Daten anzuzeigen, werden sie auf die **Datenausgabe cout** (Das »out« in cout ist englisch und bedeutet »aus«). Zunächst wird das Datenziel genannt. Dann werden zwei **Kleiner-Zeichen** verwendet, um die **Richtung der Daten** in Richtung der Ausgabe anzuzeigen.

```
cout << MeinZahlenWert;
cout << endl;
```

Im Beispiel wird die **Variable MeinZahlenWert ausgegeben**. In der Zeile darunter wird **endl** auf den Datenstrom gesendet. Dies ist keine selbst definierte Variable, sondern eine **vordefinierte Konstante für das Zeilenende**. Die Verwendung von endl sorgt auch dafür, dass alle anzuzeigenden Daten sofort auf dem Bildschirm erscheinen.

Kaskadierung

Der Übersicht halber oder um sich Tipparbeit zu sparen, können mehrere Ausgabeobjekte direkt hintereinander, durch die doppelten Kleiner-Zeichen getrennt, aufgeführt werden.

```
cout << MeinZahlenWert << endl;
```

Textausgabe

Sie können nicht nur Variablen, sondern auch **Konstanten (Text)** auf diesem Weg ausgeben. Das ist besonders interessant bei Zeichenketten, mit denen Sie Ihren Programmausgaben ein paar **erläuternde Texte beifügen** können.

```
cout << "Ergebnis: " << MeinZahlenWert << endl;
```

Im Sinne der Benutzerfreundlichkeit eines Programms, sagt der Programmierer dem Benutzer, dass der Wert, der jetzt auf dem Bildschirm erscheint, das Ergebnis des Programms ist.

Eingabestrom aus cin

Um **Eingaben von der Tastatur zu lesen**, wird die Datenquelle **cin** (Das »in« in cin steht für »ein«. Es ist also das Eingabe-Objekt. Der Eingabeoperator besteht genau **spiegelverkehrt zum Ausgabeoperator** aus **zwei Größer-Zeichen**. Sie weisen quasi vom cin auf die Variable, in der die Eingabe abgelegt werden soll.

```
#include <iostream.h>

int main(void)
{
    int Zahleingabe;
    int Doppel;

    cout << "Bitte geben Sie eine Zahl ein!" << endl;
    cin >> Zahleingabe;
    Doppel = Zahleingabe * 2;
    cout << "Das Doppelte dieser Zahl ist "
        << Doppel << "." << endl;

    return 0;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_05

Last update: **2019/05/16 17:16**

6.8.6) Ablaufsteuerung

Bisher bestanden die Programme nur aus aneinander gehängten Anweisungen. Jedoch ist es oft notwendig, den Ablauf eines Programms aufgrund einer Bedingung bzw. von Abhängigkeit von Zuständen zu verändern. Hier ein Beispielfür ein besseres Verständnis zum Thema Kaffeekochen.

Ablaufsteuerung - Beispiel "Kaffee kochen"

Folgende Anweisungen werden beim Kaffeekochen nacheinander ausgeführt:

```
Filtertüte in den Filter stecken
6 Löffel Kaffeepulver in die Filtertüte einfüllen
1 Liter Wasser in den Wasserbehälter einfüllen
Maschine einschalten
Warten
Maschine ausschalten
Kaffee entnehmen
```

Das ist allerdings eine sehr unvollständige Beschreibung des Ablaufs. In der Praxis werden Abläufe in Abhängigkeit von Zuständen erfolgen und Vorgänge wiederholt. Eine detailliertere Beschreibung des Kaffeekochens sieht so aus:

```
Wenn noch eine benutzte Filtertüte in der Maschine steckt
{
    Entnimm die Filtertüte
    Wirf sie in den Müll
}

Neue Filtertüte aus der Packung nehmen
Filtertüte in den Filter stecken

Wiederhole 6-mal:
{
    fülle einen Löffel Kaffeepulver in die Filtertüte
}

1 Liter Wasser in den Wasserbehälter einfüllen
Maschine einschalten

Wiederhole:
{
    Warte eine Minute
} bis kein Kaffee mehr aus dem Filter tropft

Maschine ausschalten
Kaffee entnehmen
```

- Verzweigungen
- Schleifen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_06

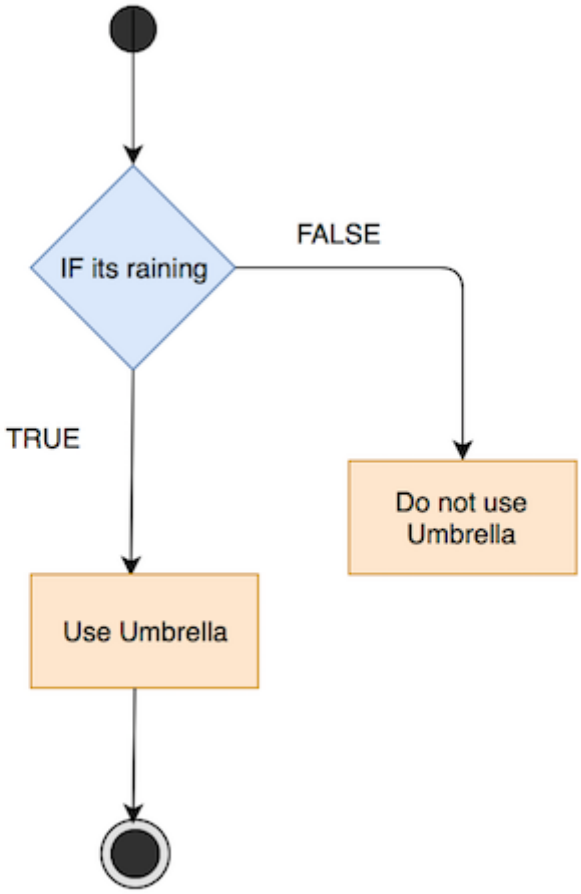
Last update: **2019/05/16 17:21**

Verzweigungen

Es gibt diverse Gründe, warum ein Programm in Abhängigkeit von Variablen bestimmte Operationen nicht ausführen soll. Hier sind ein paar typische Beispiele zusammengestellt:

- Ganz offensichtlich darf ein Programm nicht dividieren, wenn der Nenner 0 ist.
- Vor dem Ziehen einer Wurzel ist es sinnvoll zu prüfen, ob der Operand negativ ist.
- Das Programm darf nicht mehr auf Aliens schießen, wenn die letzte Rakete des eigenen Raumschiffs verschossen ist.

Das Programm soll in all diesen **Fällen** in **Abhängigkeit von einem Variableninhalt** seinen **Ablauf ändern**. Dazu brauchen wir einen **Befehl, der unter einer Bedingung eine Anweisung oder einen Anweisungsblock** ausführt. Um die Bedingung zu formulieren brauchen wir **Operatoren**, mit denen wir **Werte vergleichen** können.



Verzweigung - IF (Wenn-Dann)

Folgende Sätze wären Bedingungen, die man

```
Wenn eine Frau ihren Mann aus dem Fenster wirft, liest man das in der
Kronen-Zeitung.
Wenn ein Mann seine Frau aus dem Fenster wirft, liest man das in Schöner
Wohnen.
```

Zitat unbekannter Herkunft.

Mit dem Befehl **»if«** ist es möglich, eine **Befehlssequenz unter einer Bedingung** auszuführen. Das Schlüsselwort if kommt aus dem Englischen und bedeutet **»falls«** oder **»wenn«**. Der Syntax einer einfach if-Verzweigung sieht folgendermaßen aus:

```
if( «Bedingung» ){
    «Anweisungen»;
}
```

Dem **Schlüsselwort if** folgt eine Klammer, in der die Bedingung formuliert wird, unter der der nachfolgende **Anweisungsblock** ausgeführt wird. Ein **Anweisungsblock** ist entweder **eine einzelne Anweisung oder mehrere Anweisungen**, die durch **geschweifte Klammern zusammengefasst** werden.

```
int Gemeindeeinwohner=12000;

if (Gemeindeeinwohner > 10000)
{
    cout << "Diese Gemeinde ist eine Gemeinde mit Stadtrecht!" << endl;
}
```

Einrücken

Es hat sich bewährt, **Anweisungen**, die unter Bedingungen oder in Schleifen ausgeführt werden, **einzurücken**. Einige Programmeditoren führen dies automatisch durch. Für das Einrücken können die **Tabulatortaste oder Leerzeichen** verwendet werden.

```
if (Divisor != 0)
{
    Ergebnis = Dividend / Divisor;
}
```

Die Division wird nur durchgeführt, wenn die Bedingung zutrifft, also der Divisor ungleich 0 ist.

Verschachtelte IF-Verzweigungen

Die Anweisung, die hinter der if-Bedingung steht, kann selbst wieder eine if-Anweisung sein. Sie wird nur ausgeführt, wenn die erste if-Bedingung eintritt. Eine solche Folge von Abfragen nennt man Verschachtelung.

```
int a=0, b=0;
int c=2;
if (a==0)
    if (b==0)
        c=0;
cout << c << endl;
```

Die Abfrage, ob die Variable b 0 enthält, wird nur gestellt, wenn die Variable a den Inhalt 0 hat. Nur wenn beide Variablen 0 sind, wird die Variable c auf 0 gesetzt. Ansonsten enthält sie weiterhin 2.

Andernfalls: ELSE (Wenn-Dann-Sonst)

Kehren wir zum Divisionsbeispiel zurück. Es wäre schön, wenn das Programm nicht nur prüft, ob der Divisor ungleich 0 ist, sondern den Anwender darüber informieren würde, dass es die Division gar nicht durchführt. Diese Meldung sollte genau dann erscheinen, wenn die if-Bedingung nicht zutrifft. Das könnten Sie erreichen, indem Sie zwei gegensätzliche if-Konstruktionen hintereinander setzen:

```
if (Divisor != 0)
    Ergebnis = Dividend / Divisor;
if (Divisor == 0)
    cout << "Divisor ist 0! Keine Berechnung!";
```

Da solche Konstruktionen immer wieder gebraucht werden, bietet C++ einen eigenen Befehl an, um den gegenteiligen Fall der Bedingung abzudecken. Das Schlüsselwort dazu heißt **else**.

```
if (Divisor != 0)
    Ergebnis = Dividend / Divisor;
else
    cout << "Divisor ist 0! Keine Berechnung!";
```

Mehrfachverzweigung: switch case

Die Abfrage mit if ermöglicht die Unterscheidung zweier Fälle. Wenn auf Grund verschiedener Werte unterschiedliche Anweisungen ausgeführt werden sollen, können **mehrere verschachtelte if-Anweisungen** hintereinander gesetzt werden oder es kann eine **spezielle Fallunterscheidung** verwendet werden. Ein schönes Beispiel ist ein Fahrstuhl in einem Kaufhaus, in dem eine Vielzahl von Stockwerken zur Auswahl stehen und in jedem Stockwerk andere Waren angeboten werden. Zunächst wird die Aufgabe durch **kaskadierende if-Anweisungen** gelöst:

```
if (Stockwerk == 1)
{
     cout << "Süssigkeiten, Bücher" << endl;
}
else if (Stockwerk == 2)
{
    cout << "Bekleidung" << endl;
}
else if (Stockwerk == 3)
{
    cout << "Bekleidung" << endl;
}
else if (Stockwerk == 4)
{
    cout << "Spielzeug" << endl;
}
```

```
else if (Stockwerk == 5)
{
    cout << "Unterhaltungselektronik" << endl;
}
else
{
    cout << "Garage" << endl;
}
```

Diese Konstruktion lässt sich durch die Mehrfachverzweigung vereinfachen.

```
int Stockwerk;

cin >> Stockwerk;

switch (Stockwerk)
{
    case 1:
        cout << "Süssigkeiten, Bücher" << endl;
        break;
    case 2:
    case 3:
        cout << "Bekleidung" << endl;
        break;
    case 4:
        cout << "Spielzeug" << endl;
        break;
    case 5:
        cout << "Unterhaltungselektronik" << endl;
        break;
    default:
        cout << "Garage" << endl;
        break;
}
```

Die Fallunterscheidung beginnt mit dem **Schlüsselwort switch**. In der darauf folgenden Klammer steht der **ganzzahlige Ausdruck**, dessen **Ergebnis die Verzweigung steuert**. Ein Ausdruck kann eine Variable sein, wie im Beispiel die Variable Stockwerk. Hier könnte aber auch eine Berechnung stehen, die zu einem ganzzahligen Ergebnis führt. Es können **auch Buchstaben** verwendet werden, da Buchstaben aus Sicht von C++ letztlich nichts anderes als getarnte Zahlen sind. Im Beispiel wird die **Variable Stockwerk ausgewertet**, die offensichtlich eine ganze Zahl aufnehmen kann.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_06:14_08_06_01

Last update: **2019/05/16 17:16**

Schleifen

Es gibt viele Aufgaben, in denen Befehle wiederholt werden sollen, bis ein bestimmtes Ergebnis erzielt worden ist. Eine solche Wiederholung nennt man in Programmen eine Schleife. Jede Form des Zählens wird im Programm in einer Schleife realisiert. In Computerspielen dürfen Sie so lange ballern, bis Ihr letztes Raumschiff abgeschossen wurde. Eine Schleife hat den Wiederholungscharakter, enthält aber auch immer eine Bedingung, unter der die Schleife verlassen wird, oder positiv formuliert: Die Schleife wird so lange durchlaufen, wie die Schleifenbedingung gilt.

Kopfgesteuerte Schleife : for-loop

Die beiden bisher gezeigten Schleifen benötigten drei Elemente:

- Eine Initialisierung vor Beginn der Schleife
- Eine Bedingung, unter der die Schleife durchlaufen wird
- Eine Aktion, die die Bedingung verändern kann

Fehlt eines dieser Elemente oder ist es nicht korrekt, dann ist das Risiko einer Endlosschleife groß. Die for-Schleife enthält alle drei Elemente in ihrem Kopf. So sind sie an einer Stelle zusammengefasst und können leichter unter Kontrolle gehalten werden.

Eine sehr typische Anwendung für Schleifen ist das Zählen; und gerade hier zeigt sich besonders die Übersichtlichkeit der for-Schleife. Das folgende Beispiel zählt von 0 bis 9.

```
for (i=0; i<10; i++)
{
    cout << i << endl;
}
```

Zusammengefasst

In der Klammer, die dem Schlüsselwort for folgt, findet sich als Erstes eine Startanweisung, die die Variable i auf 0 setzt. Bei den while-Schleifen wurde diese Initialisierung vor der Schleife durchgeführt. Als Zweites ist die Schleifenbedingung zu erkennen. Die dritte Anweisung ist das Inkrementieren der Zählervariablen, die üblicherweise am Ende der Schleife steht. Diese drei Elemente werden je durch ein Semikolon getrennt.

Die for-Schleife ist der while-Schleife sehr ähnlich. Sie verpackt ihre Fähigkeiten nur eleganter und bewirkt, dass der Programmierer nicht so leicht wichtige Elemente vergisst. Sie könnten das obige Schema leicht auch als while-Schleife darstellen.

```
Startanweisung;
while ( Bedingung)
{
    Schleifenkörper;
    Schlussanweisung;
}
```

Entsprechend unterscheidet sich das Struktogramm nicht von dem der while-Schleife. Es wird lediglich inhaltlich der Kopf der Schleife in den Abfragebereich geschrieben.

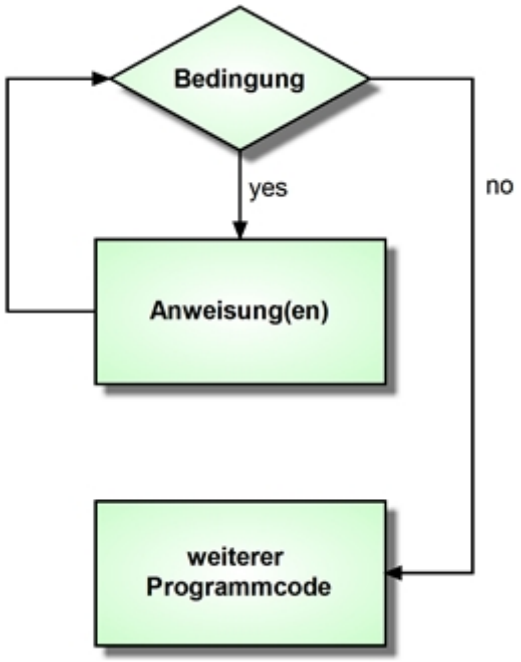
Kopfgesteuerte Schleife : while-loop

Solange

Die **einfachste Schleife** wird durch das **Schlüsselwort while** eingeleitet. Die deutsche Übersetzung lautet **»solange«**.

Sie **wiederholt Vorgänge, solange eine bestimmte Bedingung erfüllt ist**.

Auch im Alltag verwenden wir solche Schleifen. Solange die Suppe noch nicht salzig genug ist, schüttet man noch einen Teelöffel Salz hinein. Und wie in diesem Beispiel zuerst der Salzgehalt geprüft wird, so prüft auch die while-Schleife zuerst die Bedingung und führt dann erst die Anweisung aus.



Analogie zu if

Der Syntax ähnelt sehr einer if-Anweisung. Und tatsächlich bestehen die Unterschiede nur darin, dass die **Schleife die Anweisung wiederholt**. Ist die **Bedingung von vornherein nicht erfüllt**, verhält sich die Schleife exakt wie die if-Anweisung - **Sie führt die Anweisung nicht aus**.

Die Schleife beginnt mit dem **Schlüsselwort while**. In Klammern folgt die Bedingung, unter der die direkt anschließende Anweisung ausgeführt wird. Auch hier können mehrere Anweisungen zu einem Anweisungsblock zusammengefasst werden, wenn sie in geschweiften Klammern stehen.

Zählen für Anfänger

Als erstes Beispiel bringen wir dem Computer das Zählen bei. Denken Sie darüber nach, was Sie tun, wenn Sie zählen. Zunächst beginnen Sie bei einer bestimmten Zahl, typischerweise bei eins. Dann folgt die Wiederholung, in der Sie immer den bisherigen Wert um eins erhöhen. Da Sie auch irgendwann einmal etwas anderes tun wollen als zählen, werden Sie bei einem bestimmten Wert die Wiederholung abbrechen.

Zählen für Computer

Im Programm verwenden Sie eine Variable, die Sie mit dem Startwert vorbesetzen. Dieser Startwert muss vor der Schleife gesetzt werden, sonst würde die Variable in jeder Runde auf ihren Ausgangspunkt zurück gesetzt, und die Schleife würde ewig rotieren. Danach soll das Programm in die Schleife eintreten, in der die Variable immer um eins erhöht wird. Die Schleife beginnt mit dem Schlüsselwort while. Direkt dahinter wird in Klammern die Bedingung formuliert. Solange diese Bedingung gilt, bleibt das Programm innerhalb der Schleife. Wenn Sie bis zehn zählen wollen, müssen Sie hier prüfen, ob die Variable kleiner oder gleich 10 ist. Direkt anschließend an die Bedingung können Sie eine Anweisung angeben, die in der Schleife wiederholt werden soll. Da Sie ja einerseits die Variable um eins erhöhen wollen und andererseits den aktuellen Stand auf dem Bildschirm ausgeben wollen, brauchen Sie aber zwei Anweisungen, die in der Schleife ausgeführt werden sollen. Das kann aber leicht mit einem Block, also geschweiften Klammern gelöst werden. In den Block wird also die Ausgabe der Variablen und die anschließende Erhöhung der Variablen gesetzt.

```
#include <iostream>
using namespace std;
int main()
{
    i = 1;
    while (i <= 10) // Schleifenbedingung
    {
        cout << i << endl;  // Aktion
        i++; // Ohne diese Erhöhung wird es eine Endlosschleife
    }

    return 0;
}
```

Grenzfragen

Leicht entstehen Fehler bei der Festlegung der Schleifengrenzen. Würden Sie statt ⇐ nur das Kleinerzeichen verwenden, würde das Programm nur bis neun zählen. Der Grund ist, dass sobald der Inhalt von i den Wert 10 erreicht, die Bedingung nicht mehr gilt und damit das Schleifeninnere nicht mehr betreten wird. Sie können auch eine Schleife bilden, die zehnmal durchläuft, indem Sie bei 0 beginnen und abfragen, ob die Variable kleiner als zehn ist. Dann würde die Variable i von 0 bis 9 laufen. Wenn Sie i allerdings vor der Ausgabe erhöhen, erhalten Sie wieder die Zahlen von 1 bis 10.

```
#include <iostream>
using namespace std;
int main()
{
```

```
    int i = 0;
    while (i < 10)
    {
        i++;
        cout << i << endl;
    }
    return 0;
}
```

Test am Kopf

Die **while-Schleife** prüft die Bedingung am Schleifenkopf. Das bedeutet, dass die Anweisungen in der Schleife gar nicht ausgeführt werden, wenn die Bedingung vor dem Betreten der Schleife unwahr ist.

Endlosschleife

Achten Sie besonders auf die Formulierung der Schleifenbedingung. Nicht nur Anfängern passiert es, dass die Schleife niemals endet, weil die Schleifenbedingung niemals unwahr wird. Eine solche Schleife wird Endlosschleife genannt. Eine andere Ursache für eine Endlosschleife kann darin bestehen, dass die Bedingung zwar richtig formuliert ist, aber dass der Programmierer vergisst, die Daten, die in der Bedingung abgefragt werden, innerhalb der Schleife zu ändern.

```
while (true)
{

}
```

Fußgesteuerte Schleife : do....while-loop

In manchen Situationen ist die Prüfung am Kopf der Schleife ungünstig. Manchmal wollen Sie eine Aktion durchführen und werden erst an ihrem Ende erkennen können, ob die Aktion wiederholt werden soll. Ein Kind wird beispielsweise seinen Opa so lange um Süßigkeiten anbetteln, bis dieser keine mehr hat. (Erfahrene Eltern werden zu Recht einwerfen, dass die Abschlussbedingung von dem System Opa nicht eingehalten wird.) Das Kind wird in jedem Fall erst einmal betteln, ansonsten kann es die Bedingung gar nicht erst prüfen. In diesem Fall findet also die Prüfung erst nach dem Durchlauf des Schleifenkörpers statt.

Eingabeprüfung

In Programmen wird eine fußgesteuerte Schleife besonders bei Eingabeprüfungen eine typische Anwendung. Innerhalb der Schleife fordert das Programm vom Anwender eine Eingabe an. Erst dann wird geprüft, ob die Eingabe korrekt ist. Ist die Eingabe nicht zufriedenstellend, wird die Schleife wiederholt. Ein weiteres Beispiel ist das Spiel Minesweeper. Der Benutzer soll auf ein Feld klicken. Solange unter dem Feld keine Mine liegt, darf er weiterspielen. Sie können erst am Ende der Schleife

feststellen, wohin der Benutzer geklickt hat.

Auch hier werden Sie in den meisten Fällen einen Block aus geschweiften Klammern verwenden, da eine Schleife meist mehrere Anweisungen umfasst.

Im folgenden Beispiel wird rückwärts gezählt. Betrachten Sie das Listing, und überlegen Sie, ob die Zahlen 10, 1 und 0 ausgegeben werden! Prüfen Sie, ob Ihre Vermutungen stimmen!

```
#include <iostream>
using namespace std;

int main()
{
    int i = 10;
    do
    {
        cout << i << endl;
        i--;
    }
    while(i>0);

    return 0;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_06:14_08_06_02

Last update: **2019/05/16 17:16**

Übungen

Bsp 1 - Hello World

Schreibe ein Programm, das den Text „Hello World“ am Bildschirm ausgibt.

Bsp 2 - Summe

Zwei Zahlen sollen addiert werden. Das Summe wird ausgegeben.

Bsp 3 - Temperatur

Mit diesem Programm wird festgestellt, welche Umstände bei entsprechenden Temperaturen und Niederschlag zu beachten sind. Die Temperatur soll eingegeben werden. Weiters wird eingegeben, ob es Niederschlag gibt (0 für kein Niederschlag, 1 für Niederschlag).

Das Programm gibt je nach Eingaben folgende Werte zurück:

- Glatteisgefahr, Schneefahrbahn, Freie Fahrt, Aquaplaninggefahr.

Bsp 4 - Bestellung

Zwei Produkte werden bestellt (Eingabe der Anzahl). Der Gesamtpreis wird ausgegeben.

Bsp 5 - Division

Nach Eingabe von Dividend und Divisor wird das Ergebnis und der Rest der Division ausgegeben.

Bsp 6 - Fahrenheit

Ein Fahrenheit-Wert wird eingegeben und in Grad Celsius umgerechnet. Weiters wird ausgegeben, ob Glatteisgefahr besteht oder nicht.

Bsp 7 - Kreisumfang und -flächeninhalt

Nach Eingabe von Radius wird der Umfang und Flächeninhalt eines Kreises ausgegeben.

Bsp 8 - Sortiertest

Drei Zahlen werden eingegeben. Das Programm überprüft, ob die drei eingegebenen Zahlen sortiert

sind. Wenn dies zutrifft, wird die größte Zahl ausgegeben.

Bsp 9 - Dreieck

Die drei Seitenlängen a, b und c eines Dreiecks eingegeben. Es wird ausgegeben, ob es sich um ein Dreieck handelt. Weiters wird ermittelt, ob ein gleichschenkeliges, gleichseitiges, rechtwinkeliges oder allgemeines Dreieck vorliegt.

Bsp 10 - Body Mass Index

Im Programm wird eingegeben, ob man eine Frau oder ein Mann ist (z.B. 1 oder 2). Weiters gibt man die Körpergröße in cm sowie das Gewicht in kg an. Mittels folgender Formel wird der Body Mass Index berechnet und ausgegeben:

`bmi=(kg)/m^2`

Weiters soll nach folgender Tabelle ermittelt werden, ob Unter-, Normal- oder Übergewicht vorliegt.

	Frauen	Männer
Untergewicht	<19.1	<20.7
Normalgewicht	19.1-25.8	20.7-26.4
Übergewicht	>25.8	>26.4

Bsp 11 - Geschwindigkeit

Je nach Höhe der Geschwindigkeit gibt es entweder keine Strafe oder gestaffelte Strafzahlungen.

Bsp 12 - Wochentag

Das Programm wandelt die Zahlen 0 bis 6 in den entsprechenden Wochentag um.

Bsp 13 - Noten

Das Programm gibt die eingegebene Schulnote verbal aus.

Bsp 14 - Textbasierter Rechner

Der Benutzer kann zwei Zahlen eingeben.

Anschließend gibt er mittels Eingabe von 1 (addieren), 2 (subtrahieren), 3 (multiplizieren) oder 4 (dividieren) an, welche Grundrechnungsart er wählen möchte. Das Ergebnis wird anschließend ausgegeben.

Wird eine andere Zahl eingegeben, bricht das Programm ab.

Bsp 15 - Uhrzeit

Das Programm wandelt die digitale Uhrzeit (z.B. 12:45) in die gesprochene Uhrzeit (Dreiviertel eins) um!

Bsp 16 - Gaußsche Wochentagsformel

Das Programm berechnet nach eingegebenem Datum den entsprechenden Wochentag.

Formel siehe: Gaußsche Wochentagsformel

Bsp 17 - Euroscheinüberprüfung

Schleifen und char nötig!!!

Jeder Euroschein hat eine Nummer, zum Beispiel N27175276569.

Der erste Buchstabe (hier N) ist ein Ländercode für das Herstellungsland, z.B. N für Österreich.

Darauf folgt eine zehnstellige, fortlaufende Seriennummer (hier: 2717527656). Dann folgt am Ende eine einstellige Prüfziffer, im Beispiel ist dies die 9.

Um zu prüfen, ob die Seriennummer bzw. der Schein echt ist, ist folgendes Verfahren anzuwenden:

1. Buchstaben durch die Position des Buchstaben im Alphabet ersetzen: A=1, B=2, ..., Y=25, Z=26.
2. Die Summe aus der Buchstabenzahl, den zehn Ziffern der Seriennummer und der Prüfziffer bilden.
3. Von dieser Summe die Ziffernsumme (und wieder die Ziffernsumme ...) bilden, bis die Zahl einstellig wird.

Beispiel: 14+2+7+1+7+5+2+7+6+5+6+9=71

7+1=8

Das Ergebnis muss 8 sein, damit die Seriennummer echt ist!

Welcher Buchstabe für welches Land steht, findest du z.B. auf der Seite der Europäischen Zentralbank: http://www.ecb.int/euro/banknotes/html/index.de.html

Bsp 18 - Umwandlung Dualzahl in Dezimalzahl

Schreibe ein Programm, welches eine Dualzahl in eine Dezimalzahl umwandelt.
Bei der Eingabe dürfen nur die Ziffern 0 und 1 oder e eingegeben werden. Sobald e eingegeben wird, ist die Zahl zu Ende.
Die entsprechende Dezimalzahl wird ausgegeben.

Bsp 19 - Umwandlung Hexadezimalzahl in Dezimalzahl

Schreibe ein Programm, welches eine Hexadezimalzahl in eine Dezimalzahl umwandelt.
Bei der Eingabe dürfen nur die Ziffern 0 bis 9 bzw. A bis F oder e eingegeben werden. Sobald e eingegeben wird, ist die Zahl zu Ende. Die entsprechende Dezimalzahl wird ausgegeben.

Bsp 20 - "Tempo, kleine Fische"

Simuliere das Spiel „Tempo, kleine Fische“, indem du den Spielverlauf in geeigneter Form in ein Programm umsetzt.

Spielanleitung

a) Via Zufallswürfel soll der Verlauf einer Spielrunde simuliert werden.
b) Alternativ dazu soll der Benutzer bestimmen können, wie oft das Spiel durchgeführt wird (z.B. 100 mal). Danach gibt das Programm aus, wie oft die verschiedenen Möglichkeiten des Spielendes (Alle Fische erreichen das Meer, etc.) aufgetreten sind.

Bsp 21 - Malreihen

Schreibe ein Programm, dass alle 1*1-Reihen (1-10) auf einmal ausgegeben werden.

Hinweise: Schleifen können auch ineinander geschachtelt werden!

Ausgabebeispiel:

```
Das folgende Programm gibt alle Einmaleins-Reihen aus:

1er-Reihe:
1*1=1
2*1=2
...

2er-Reihe:
1*2=2
2*2=4
...
```

Bsp 22 - Quadratzahlen

Schreibe ein Programm, das eine Quadratzahltabelle von 1 bis zu einem eingegebenen Wert *n* ausgibt. (Quadratzahl.cpp)

```
Eingabe n=9

1 -> 1
2 -> 4
```

```
3 -> 9
4 -> 16
5 -> 25
6 -> 36
7 -> 49
8 -> 64
9 -> 81
```

Bsp 23 - Fibonacci

Schreibe ein Programm, das die ersten *n* Fibonacci-Zahlen erzeugt und ausgibt! Eine Fibonacci-Zahl berechnet sich aus der Summe der beiden vorhergehenden Zahlen. Die ersten beiden Fibonacci-Zahlen sind gleich 1! (Fibonacci.cpp)

z.B.: 1, 1, 2, 3, 5, 8, 13, 21,...

Bsp 24 - Dezimal-Dual

Schreibe ein Programm, das die eingegebene Dezimalzahl in eine Dualzahl umwandelt.
Auf welches Problem stößt du dabei? (DeziDual.cpp)

Bsp 25 - Teiler

a) Schreibe ein Programm, das die Teiler einer eingegebenen natürlichen Zahl n auflistet. (Teilerliste.cpp)
b) Schreibe ein Programm, das die Anzahl der Teiler eingegebenen natürlichen Zahl n ausgibt. (Teileranzahl.cpp)

Bsp 26 - Staffelrechnung/Rechenturm

Erstelle ein Programm, das eine Staffelrechnung / einen Rechenturm durchführt. Der Benutzer gibt ein, mit welcher Zahl er die Staffelrechnung beginnen will und bis zu welcher Multiplikation die Rechnungen durchgeführt werden:

Z.B.
Anfangszahl: 385
Multiplikationen bis: 6

Ausgabe:

```
385*2=770
770*3=2310
2310*4=9240
9240*5=46200
46200*6=277200
277200/2=138600
```

```
138600/3=46200
46200/4=11550
11550/5=2310
2310/6=385
```

Bsp 27 - verschachtelte Schleifen

Rechteck

Schreibe ein Programm, das mit Sternchen ein Rechteck ausgibt! Das Programm soll Breite, Höhe und einen Versatz einlesen. Der Versatz bestimmt, um viele Zeichenbreiten das gesamte Rechteck nach rechts verschoben wird.

```
Ein Rechteck

    Breite?
    17

    Hoehe?
    4

    Versatz?
    2

      ******************
      ******************
      ******************
      ******************
```

Parallelogramm

Schreibe ein Programm, das mit Sternchen ein Parallelogramm ausgibt! Das Programm soll die Breite, Höhe und die Versatzänderung (die Verschiebung nach rechts pro Zeile) einlesen.

```
Ein Parallelogramm

    Breite?
    11

    Hoehe?
    5

    Versatzaenderung?
    1

    ***********
     ***********
```

```
      ***********
       ***********
        ***********
```

Dreieck

Schreibe ein Programm, das mit Sternchen ein Dreieck ausgibt! Das Programm soll die Höhe und die Verbreiterung pro Zeile einlesen.

```
Ein Dreieck

    Hoehe?
    6

    Verbreiterung?
    1

    *
    **
    ***
    ****
    *****
    ******
```

Bsp 28 - Regelmäßige Zahlungen

Schreibe ein Programm, das den Betrag eines regelmäßigen jährlichen Einzahlungsbetrages, die Verzinsung und den gewünschten Endbetrag einliest und ausgibt, nach wie vielen Jahren der gewünschte Endbetrag erreicht (oder erstmals überschritten) ist.

Am Anfang des ersten Jahres wird zunächst ein Einzahlungsbetrages, z.B. 1000 Euro, eingezahlt. Am Ende des ersten Jahres (nach einem Jahr) kommt der Zins, bei 5 % z.B. 50 Euro, und ein weiterer Einzahlungsbetrag, wieder 1000 Euro, hinzu. Nach einem Jahr ist der Kontostand dann also beispielsweise 1000 + 50 + 1000 Euro = 2050 Euro. Sollten also beispielsweise 2000 Euro angespart werden, so wäre dies unter diesen Umständen nach einem Jahr erreicht.

Man kann also in einer Schleife die Jahre mitzählen und den Kontostand fortschreiben. Dann kann man jedesmal prüfen, ob der gewünschte Endbetrag schon erreicht oder überschritten wurde und gegebenenfalls die Zahl der abgelaufenen Jahre und den Kontostand ausgeben. In der Endfassung darf das Programm aber nichts anderes ausgeben als das Endergebnis (keine Zwischenstände).

Das Programm muß auch dann eine richtige (sinnvolle) Antwort geben, wenn der gewünschte Endbetrag kleiner oder gleich dem Einzahlungsbetrag ist.

Die Lösung der Übungsaufgabe sollte mindestens mit den beiden folgenden Testsituationen geprüft werden und muß dann identische Ergebisse liefern.

```
Beispiel 1
```

```
    jaehrlicher Zahlbetrag?
    1000

    Verzinsung, in Prozent?
    5

    gewuenschter Endbetrag?
    2000

    Es wird 1 Jahr benoetigt.
    Der erreichte Endbetrag ist dann 2050.


Beispiel 2

    jaehrlicher Zahlbetrag?
    100

    Verzinsung, in Prozent?
    5

    gewuenschter Endbetrag?
    10000

    Es werden 36 Jahre benoetigt.
    Der erreichte Endbetrag ist dann 10162.8.
```

Bsp 29 - Statistische Kenndaten

Schreibe Sie ein Programm, das in einer Schleife solange natürliche Zahlen einliest, bis -1 eingegeben wird und dann Maximum, Minimum und das arithmetische Mittel der Zahlen ausgibt. Da die Zahl -1 nur als Stoppwert verwendet wird, soll sie bei der Statistik nicht berücksichtigt werden.

Hinweis: Wenn die Variablen summe und n beide vom Typ integer sind, dann wird bei einer Division das Ergebnis (arithmetisches Mittel am) auch ganzzahlig berechnet. Um dies zu verhindern konvertiert man den Zähler mittels `(float)`.

```
am=summe/(float)n;
```

Bsp 30 - Harshad-Zahlen

Eine natürliche Zahl, die durch ihre eigene Ziffernsumme teilbar ist, heißt Harshad- oder Niven-Zahl. Zum Beispiel:
777 ist durch 7 + 7 + 7 = 21 teilbar und damit eine Harshad-Zahl.

Schreibe ein Programm, das alle Zahlen bis n testet, ob sie Harshad-Zahlen sind. Das Programm soll alle Harshad-Zahlen ausgeben.

Bsp 31 - Werfen von 2 Würfeln

Stochastische Feststellung: Kommt die 7 wirklich am häufigsten vor?

Wenn man mit 2 Würfeln würfelt, so tritt die 7 am häufigsten auf.

Wie oft tritt welche Zahl auf, wenn man n mal würfelt?

```
Werfen von 2 Würfeln...
Gib an, wie oft gewürfelt werden soll: 100
Gib an, welche Summe gezählt werden soll: 6

Insgesamt kam die Augensumme 6 genau 12 mal vor.
```

Bsp 32 - Prüfen der Sozialversicherungsnummer

Schreibe ein Programm, welche die Gültigkeit einer Sozialversicherungsnummer prüft.
Der Benutzer gibt die Sozialversicherungsnummer ein, das Programm gibt aus, ob es sich um eine gültige Sozialversicherungsnummer handelt.

Informationen zur Sozialversicherungsnummer:

http://www.pruefziffernberechnung.de/V/VSNR-AT.shtml

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:14:14_08:14_08_06:14_08_06_03

Last update: **2019/05/16 17:16**

Grundlagen des (objektorientierten) visuellen Programmierens

Theorie

Objektorientiertes Programmieren

- OOP - Klassen

Programme mit grafischer Benutzeroberfläche

GUI

Grafische Benutzeroberfläche oder auch grafische Benutzerschnittstelle (Abk. GUI von englisch graphical user interface) bezeichnet eine Form von Benutzerschnittstelle eines Computers. Sie hat die Aufgabe, Anwendungssoftware auf einem Rechner mittels grafischer Symbole, Steuerelemente oder auch Widgets genannt, bedienbar zu machen. Dies geschieht bei Computern meistens mittels einer Maus als Steuergerät, mit der die grafischen Elemente bedient oder ausgewählt werden, bei Smartphones, Tablets und Kiosksystemen in der Regel durch Berührung eines Sensorbildschirms.

Programmen mit GUI vs. Konsolenanwendungen

Vorteile Konsolenanwendung

- **schneller, schlanker** und benötigen **weniger Arbeitsspeicher**
- **übersichtlicher** und **leichter zu warten**
- **leichter portierbar** – die **GUI-Programmierung funktioniert auf jedem Betriebssystem unterschiedlich**. Bei der Trennung von GUI und Hauptprogramm kann das Hauptprogramm oft ohne tiefgreifende Änderungen für ein anderes System kompiliert werden. So muss jeweils nur eine kleine GUI für jede Plattform von Grund auf neu erstellt werden. Diese kann dann auch besser auf die Eigenheiten der jeweiligen Plattform abgestimmt werden.
- **vielseitiger verwendbar** – die Funktion von Konsolenprogramms kann oft per **Verkettung (Piping) mit anderen Programmen** – auch mit GUI-Programmen – kombiniert werden; Ein- und Ausgaben von Kommandozeilenprogrammen kann durch Ein-/Ausgabeumleitung aus und in Dateien geleitet werden (Batchbetrieb).
- **leicht erweiterbar** ohne **Zusatzkomplexität**

Vorteile GUI-Programme

- **Einfachere, intuitivere Benutzung** des Programms durch sichtbare Objekte
- **Learning-by-doing** Konzept - unerfahrene Benutzer können die Funktionen des Programms erahnen

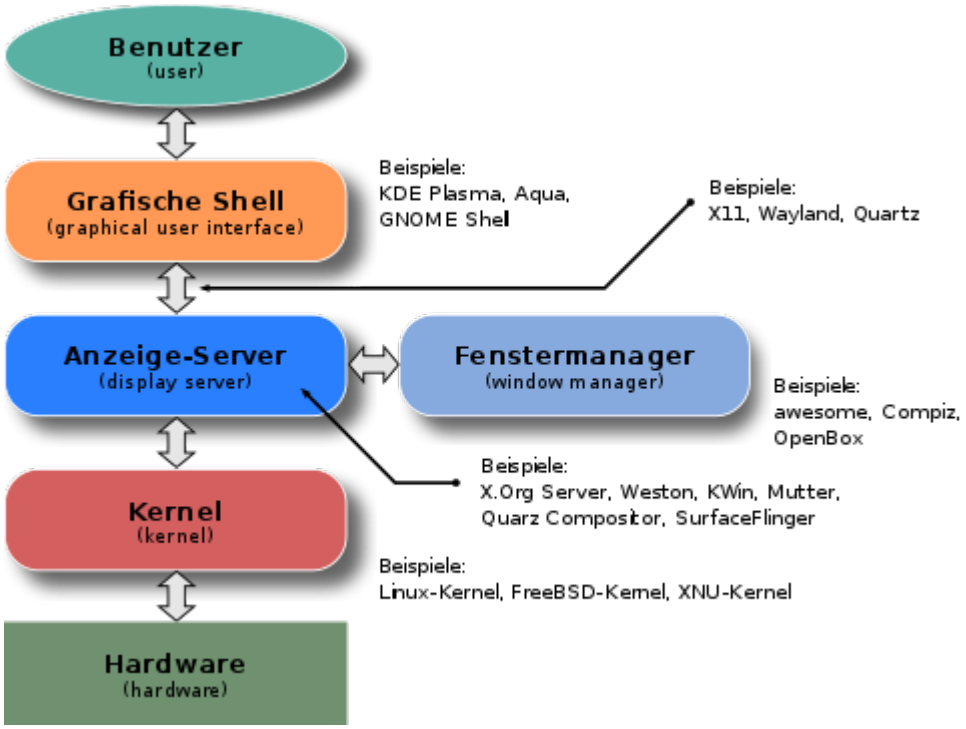
Fenstersystem

Ein Fenstersystem (englisch windowing system) ist der Unterbau einer grafischen Benutzeroberfläche (GUI), deren Hauptaufgabe die Verwaltung von Programmfenstern ist. Im Normalfall ist es Teil einer größeren Desktop-Umgebung.

Aus der Sicht eines Programmierers implementiert das Fenstersystem die grafischen Basisfunktionen, wie das Darstellen von Schriftarten, Zeichnen von Linien, Kurven und Pixelgrafiken, und das Abstrahieren der Grafikhardware (Grafikkarte).

Das Fenstersystem gestattet es dem Anwender mit mehreren Programmen gleichzeitig zu arbeiten, indem jedes Programm „in“ einem oder mehreren eigenen Bereichen des Bildschirms, den Fenstern, ausgeführt wird, die üblicherweise rechteckig sind, mit dem Zeigegerät (Maus) frei bewegt werden können und einander überlappen dürfen.

Einige Fenstersysteme, wie das X Window System in Unix-artigen Umgebungen, haben erweiterte Fähigkeiten wie Netzwerkstransparenz, die es dem Anwender gestatten die grafische Oberfläche einer Anwendung auf einem anderen Computer darzustellen. Das X-Window-System implementiert auch kein festes Aussehen der Umgebung, wodurch die Fenstermanager, GUI-Toolkits und Desktop-Umgebungen volle Freiheit bei der optischen Gestaltung und Handhabung haben.



GUI in C++

C + + bringt von Haus aus kein GUI mit, da es als hardwarenahe plattformunabhängige Sprache erstellt wurde und GUIs stark vom verwendeten Betriebssystem abhängen. Dieser „Mangel“ könnte einige Neueinsteiger vielleicht auf den ersten Blick abschrecken, da sie sich einen Einstieg in C + + oder gar in die gesamte Programmierung mit vielen Buttons, Textfeldern, Statusanzeigen und Menüeintragen erhofft haben.

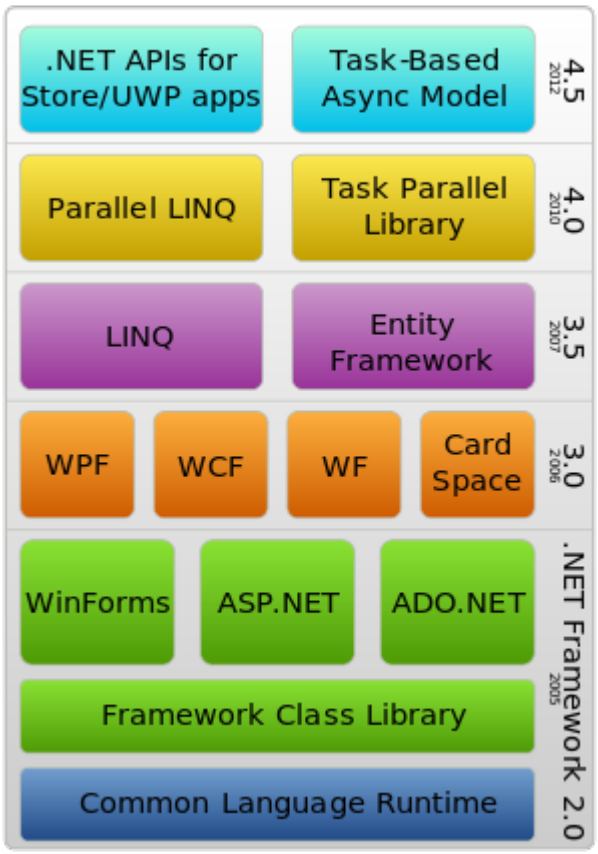
Sobald man mit C + + dann doch ein GUI programmieren will, sollte auf externe Bibliotheken

zurückgegriffen werden. Alle GUIs erfordern ein Verständnis der C + + Grundlagen. Nachfolgend sind einige Bibliotheken zur GUI-Programmierung aufgelistet. Viele der Bibliotheken lassen sich auch mit anderen Programmiersprachen einsetzen.

.NET-Framework

.NET Framework (auch mit dotNetFx oder nur mit NetFx abgekürzt) ist ein **Teil von Microsofts Software-Plattform .NET.** Es umfasst eine Sammlung zahlreicher **Werkzeuge und Schnittstellen**, die die **Entwicklung von Anwendungsprogrammen unterstützen und vereinfachen** sollen, sowie eine Laufzeitumgebung für Programme, die mithilfe dieses Frameworks geschrieben wurden. Da diese Programme hierbei nicht herkömmlich direkt in Instruktionen für einen Prozessor übersetzt werden, braucht ein System für die Ausführung der Programme zusätzliche Software (welche die Laufzeitumgebung des Frameworks bildet). Programme, die mit dem .NET Framework geschrieben wurden, werden in der Regel so ausgeliefert, dass benötigte Komponenten des Frameworks automatisch mitinstalliert werden. Der Benutzer muss sich also nur im Ausnahmefall um das Vorhandensein der Laufzeitumgebung kümmern. Außerdem ist das Framework seit vielen Jahren bereits Bestandteil von Windows Installationen, wenn auch nicht immer in der aktuellsten Version und daher ist diese für manche Zwecke nicht ausreichend.

Ziele des Frameworks, wie auch der Plattform .NET, sind unter anderem **Kompatibilität zwischen verschiedenen Plattformen**, **Zusammenführung bisher inkompatibler Microsoft Technologien** und **Verbesserung veralteter Standards**. Aus Anwendersicht birgt das Framework zwar keine direkten Vorteile, da sich seine Funktionen und Anwendungsbereiche hauptsächlich auf Softwareentwicklung, -betrieb und -wartung beschränken, dennoch ist es eine notwendige Voraussetzung für den Betrieb zahlreicher Anwendungsprogramme.



Qt

Qt ist eine leistungsstarke plattformübergreifende Klassenbibliothek mit Meta-Object Compiler (gennant: moc), die von der norwegischen Firma Trolltech entwickelt wurde. Trolltech wurde 2008 von Nokia übernommen, welches jedoch wirtschaftliche Schwierigkeiten hatte und in der Folge Qt 2011 an das finnische Softwareunternehmen Digia verkaufte.

Die Entwicklung wird inzwischen vom Qt Project vorangetrieben, so dass jeder sich daran beteiligen kann. Die Klassenbibliothek ist unter der GNU General Public License (GPL) und der GNU Lesser General Public License (LGPL) lizenziert. Es gibt außerdem eine proprietäre Lizenz von Digia, die allerdings lediglich zusätzlichen technischen Support beinhaltet.

Speziell aufgrund der Veränderungen, die mit Qt 5 und C + + Einzug in das Framework halten, ist Qt eine der mächtigsten Bibliotheken für C + +. Ein weiterer Vorteil von Qt ist die vollständig freie Entwicklungsumgebung Qt Creator, die insbesondere Anfängern das Programmieren leichter macht.

VCL - Visual Component Library

Die **Visual Component Library (VCL)** ist ein GUI-Toolkit für Windows-Anwendungen. Sie wurde von **Embarcadero (Rad Studio)**, vormals Borland, Inprise und CodeGear, erstellt. Die VCL kann in den Programmiersprachen Borland Delphi, C + +, C, C# verwendet werden. Sie wird von den meisten Borland-Entwicklungsumgebungen als Komponentensammlung benutzt.

MFC - Microsoft Foundation Classes

Die Microsoft Foundation Classes (MFC) sind eine Sammlung objektorientierter Klassenbibliotheken (GUI-Toolkit), die von Microsoft für die Programmierung von Anwendungen mit grafischen Benutzeroberflächen für Windows mit C + + entwickelt wurden.

WF - Windows Forms

Windows Forms ist ein **GUI-Toolkit des Microsoft .NET Frameworks**. Es ermöglicht die Erstellung grafischer Benutzeroberflächen (GUIs) für Windows in C#. Im Vergleich zu Microsoft Foundation Classes (MFC), die auf der Programmiersprache C + + basiert, ist der Einstieg in die Programmierung mit Windows Forms einfacher. Das Framework basiert nicht auf dem Paradigma Model View Controller (MVC).

Mit .NET Framework 3.0 wurde von Microsoft eine Alternative zu Windows Forms bereitgestellt, die Windows Presentation Foundation, welche eine stärkere Trennung der grafischen Oberfläche vom Programmcode und – unter Zuhilfenahme von XAML, einer XML-basierenden Sprache – ein dynamischeres Layout ermöglicht.

GUI in C#

WPF - Windows Presentation Foundation

Windows Presentation Foundation (kurz WPF), auch bekannt unter dem Codenamen Avalon, ist ein Grafik-Framework und Fenstersystem des .NET Frameworks von Microsoft. Es wird seit Windows Vista mit Windows ausgeliefert und lässt sich auf Windows XP (bis zur Version 4.0) und Windows Server 2003 nachinstallieren. Für das neue Framework .NET Core soll WPF ab der Version 3.0 (für 2019 erwartet) unter Windows zur Verfügung stehen.[2]

WPF ist eine 2006 neu eingeführte Klassenbibliothek, die zur Gestaltung von grafischen Benutzeroberflächen und zur Integration von Multimedia-Komponenten und Animationen dient. Sie vereint DirectX, Windows Forms, Adobe Flash, HTML und CSS.

WPF stellt ein umfangreiches Modell für den Programmierer bereit. Dabei werden die Präsentation und die Geschäftslogik getrennt, dies wird vor allem durch die Auszeichnungssprache XAML (basierend auf XML) unterstützt. XAML beschreibt Oberflächen-Hierarchien deklarativ als XML-Code. WPF-Anwendungen können sowohl Desktop- als auch Web-Anwendungen sein und benutzen, wenn möglich, Hardwarebeschleunigung. Das Framework versucht, die verschiedenen Bereiche, die für die Präsentation wichtig sind (Benutzerschnittstelle, Zeichnen und Grafiken, Audio und Video, Dokumente, Typographie), zu vereinen.

Praxis

Visual C ++

- Kurzmeldungen und Kurzeingaben
- Listbox
- Memobox
- Strings
- RichEdit
- Datenstrukturen
- Zugriff auf Datenbanken
- Image-Komponente
- Vertiefende Übungen

Links:

- Turbo C++ Buch Inhaltsverzeichnis
- Kapitel 1
- Kapitel 2
- Kapitel 3,1-9
- Kapitel 3,10-19
- Kapitel 3,20-24
- Kapitel 4
- Kapitel 5
- Kapitel 6
- Kapitel 7
- Kapitel 8

- Kapitel 9
- Kapitel 10 visuelle Komponenten

- Entwicklungsumgebung für C++ (RAD Studio XE)
- Entwicklungsumgebung für C++ (Visual Studio 2013)
- APP-Programmierung (unter RAD Studio XE7)
- Erstellen eines ausführbaren EXE-Projekts

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:15

Last update: **2019/05/26 06:59**

Objektorientierte Programmierung (OOP)

Am Anfang steht wohl die Frage was Objektorientierte Programmierung überhaupt ist. Dies ist nicht ganz einfach zu erklären und es gibt viele Definitionen die dies versuchen.

Grob gesagt ist **Objektorientierte Programmierung (OOP) ein Verfahren zur Strukturierung von Programmen, bei dem Programmlogik zusammen mit Zustandsinformationen (Datenstrukturen) in Einheiten, den Objekten, zusammengefasst werden**.

Es ist also möglich mehrere Objekte des gleichen Typs zu haben. Jedes dieser Objekte hat dann seinen individuellen Zustand. Dies bietet die Möglichkeit einer besseren Modularisierung der Programme, sowie einer höheren Wartbarkeit des Quellcodes.

Vorteile

Wiederverwendung

Ein Vorteil des auf der Objektorientierung beruhenden Programmierparadigmas, der Objektorientierten Programmierung (kurz: OOP), stellt die Möglichkeit dar, Programmteile wiederzuverwenden. Konsistent entwickelte Programmteile können in verschiedenen Konfigurationen und verschiedenen logischen Zusammenhängen genutzt werden

Flexibilität

Durch die Aufgliederung des Programms in Objekte ist es einfacher, Änderungen an der Programmierung vorzunehmen. Oft ist es ausreichend, nur ein Objekt abzuändern, um Auswirkungen auf das gesamte Programm zu erzielen. Durch objektorientierte Programmierung wird eine evolutionäre Arbeitsweise unterstützt, d.h. das schrittweise Ausbauen eines Programms ist gegenüber der herkömmlichen Programmierung vereinfacht.

Modularität & Struktur

Die unterschiedlichen Programmteile sind somit unabhängig voneinander und können einzeln und von verschiedenen Personen entwickelt werden.
Außerdem verbessert die Modularität und Struktur von objektorientierten Programmen die Lesbarkeit des Programms.

Fehlersuche & Wartung

Ein weiterer Vorteil ist, dass durch die Strukturierung in mehrere Programmteile sich die Fehlersuche & Wartung des Programms vereinfacht.

Klassen

Die Klasse (class) ist die zentrale Datenstruktur in C + +. Sie kapselt zusammengehörige Daten und Funktionen vom Rest des Programmes ab. Sie ist das Herz der objektorientierten Programmierung (OOP).

Klassen sind ein erweitertes Konzept von Datenstrukturen denen es erlaubt ist, außer Daten, noch Methoden zu beinhalten. Ein Objekt ist eine Instanz einer Klasse, welches sich die Eigenschaften der Klasse zunutze machen kann. Es ist möglich mehrere Objekte einer Klasse zu instanziieren, wobei jedes dieser Objekte zwar die selben Eigenschaften hat, intern aber einen ganz individuellen Zustand haben kann.

Beispiel: Zwei Instanzen der Klasse „Gegner“ werden erzeugt. Nachdem ein Gegner Schaden erlitten hat sind seine Lebenspunkte auf 50 gesunken. Die des anderen Gegners haben allerdings noch den Wert 100.

Klassen werden normalerweise mit dem Schlüsselwort **class** deklariert. Außerdem haben sie immer folgendes Format:

```
class Klassenname
{
Zugriffsspezifizierung 1:
Member 1;
...
Zugriffsspezifizierung 2:
Member 2;
...
...
};
```

Der Klassenname identifiziert die Klasse, der Objektname ist eine optionale Liste von Namen von Objekten dieser Klasse. Der Körper der Klasse wird durch geschweifte Klammern gekennzeichnet und kann verschiedene Member (Methoden, Membervariablen) beinhalten. Diesen können verschiedene Zugriffsspezifierungen zugewiesen werden.

Zugriffsrechte

public: Auf Members kann von überall zugegriffen werden von wo das Objekt sichtbar ist

private: Auf Member kann nur innerhalb anderer Member zugegriffen werden oder aus befreundeten Klassen und Funktionen

protected: Members sind verfügbar für Member der selben Klasse, befreundeten Funktionen/Klassen und Abgeleiteten Klassen

Attribute

Die Festlegung, welche Daten zu einem Typ gehören, erfolgt bei der Definition der Klasse. Die Objekte

enthalten jeweils unabhängig voneinander einen Satz von Datenkomponenten (Attributen). Ihre Startwerte können durch Konstruktoren festgelegt werden. C + + erlaubt auch die Zuweisung von Anfangswerten bei der Definition:

```
class Enemy {
    public:   //Zugriffsrecht public
        void setHealth(int h);
        int getHealth();
        string name;
                int id;
    private:  //Zugriffsrecht private
        int health;
};
```

Elementzugriff

Im Beispiel davor wird ein Objekt vom **Typ Enemy** angelegt. Das **Objekt** enthält die **3 Variablen health (integer), id (int) und name (String)**, wie es in der Klassendefinition zu sehen ist. Um auf eine öffentliche Elementvariable zugreifen zu können, wird an den Objektnamen ein **Punkt** und dann der in der Klassendefinition verwendete Elementname gehängt. Im Beispiel wird der Name auf Andreas gesetzt.

```
    ...
    //Zugriff auf öffentliche Variable name
    e1.name="Andreas";
    //Zugriff auf öffentliche Variable id
    e1.id=1;
    ...
    //e1.health=100; ist nicht erlaubt, da health das Schlüsselwort private
besitzt!!!
```

Klassenmethoden

Nun können Sie ein Objekt von der Klasse Enemy erzeugen. Aber was nützt Ihnen die schönste Datenstruktur in Ihrem Programm, wenn sie nicht durch Funktionen zum Leben erweckt wird? Sie werden z.B. einen Namen und Lebenspunkte eingeben und ausgeben wollen. Vielleicht wollen Sie die Lebenspunkte verringern oder erhöhen. Kurz gesagt, ein Datenverbund ist nichts wert ohne Funktionen, die auf ihn wirken. Aber die Funktionen sind auch nur im Zusammenhang mit ihrem Datenverbund sinnvoll. Aus diesem Grund werden die Funktionen ebenso in die Klasse integriert wie die Datenelemente. Eine Funktion, die zu einer Klasse gehört, nennt man Elementfunktion oder auf englisch member function. In anderen objektorientierten Programmiersprachen spricht man auch von einer Methode oder Operation. Aufruf So, wie Sie auf Datenelemente nur über ein real existierendes Objekt, also eine Variable dieser Klasse zugreifen können, kann auch eine Funktion nur über ein Objekt aufgerufen werden. Objekt und Funktionsnamen werden dabei durch einen Punkt getrennt. Die Funktion arbeitet mit den Daten des Objekts, über das sie gerufen wurde. Aus prozeduraler Sicht könnte man es so sehen, dass eine Elementfunktion immer bereits einen Parameter mit sich trägt, nämlich das Objekt, über das sie aufgerufen wurde.

Beispiel:

```
#include <iostream>
#include <conio.h>
#include <string.h>

using namespace std;

//Klasse Enemy
class Enemy {
    public:   //Zugriffsrecht public
        void setHealth(int h);
        int getHealth();
        string name;
                int id;
    private:  //Zugriffsrecht private
        int health;
};
//Methode setHealth zum Setzen der Lebenspunkte
void Enemy::setHealth(int h)
{
    health=h;
}

//Methode getHealth() zum Auslesen der Lebenspunkte
int Enemy::getHealth()
{
    return health;
}

//Hauptprogramm
int main()
{
    Enemy e1;   //Objekt e1 der Klasse Enemy wird erzeugt

    //Zugriff auf öffentliche Variable name
    e1.name="Andreas";
    //Zugriff auf öffentliche Variable id
    e1.id=1;

    //Aufruf der Methode setHealth
    e1.setHealth( 100 );
    //Aufruf der Methode getHealth und Zugriff auf die öffentliche Variable
name
    cout << "Der Gegner " << e1.name << " hat " << e1.getHealth() << "
Lebenspunkte.\n";
    //Aufruf der Methode setHealth
    e1.setHealth( 50 );
    //Aufruf der Methode getHealth und Zugriff auf die öffentliche Variable
name
    cout << "Der Gegner " << e1.name << " hat " << e1.getHealth() << "
```

```
Lebenspunkte.n";

    return 0;
}
```

Methodendefinition innerhalb einer Klasse

Alternativ können Sie die Elementfunktion auch direkt in der Klasse definieren. Das wird leicht unübersichtlich, darum sollten Sie das nur bei sehr kurzen Funktionen tun.

```
#include <iostream>
#include <conio.h>
#include <string.h>

using namespace std;

class Enemy {
    public:     //Zugriffsrecht public
                /***** INLINE - METHODENDEFINITION *****/
        //Methode setHealth zum Setzen der Lebenspunkte
        void setHealth(int h)
        {
            health=h;
        }
        //Methode getHealth() zum Auslesen der Lebenspunkte
        int getHealth()
        {
            return health;
        }

        string name;
                int id;


    private:     //Zugriffsrecht private
        int health;
};

//Hauptprogramm
int main()
{
    Enemy e1;   //Objekt e1 der Klasse Enemy wird erzeugt

    //Zugriff auf öffentliche Variable name
    e1.name="Andreas"

    //Aufruf der Methode setHealth
    e1.setHealth( 100 );
    //Aufruf der Methode getHealth und Zugriff auf die öffentliche Variable
name
```

```
    cout << "Der Gegner " << e1.name << " hat " << e1.getHealth() << "
Lebenspunkte.\n";
    //Aufruf der Methode setHealth
    e1.setHealth( 50 );
    //Aufruf der Methode getHealth und Zugriff auf die öffentliche Variable
name
    cout << "Der Gegner " << e1.name << " hat " << e1.getHealth() << "
Lebenspunkte.\n";

    return 0;
}
```

Aufruf

So, wie Sie auf Datenelemente nur über ein real existierendes Objekt, also eine Variable dieser Klasse zugreifen können, kann auch eine Funktion **nur über ein Objekt aufgerufen werden**.
Objekt und **Funktionsnamen** werden dabei durch einen **Punkt** getrennt. Die Funktion arbeitet mit den Daten des Objekts, über das sie gerufen wurde. Aus prozeduraler Sicht könnte man es so sehen, dass eine Elementfunktion immer bereits einen Parameter mit sich trägt, nämlich das Objekt, über das sie aufgerufen wurde.

```
    ....
    Enemy e1;   //Objekt e1 der Klasse Enemy wird erzeugt

    //Aufruf der Methode setHealth
    e1.setHealth( 100 );
    ....
```

Konstruktor

Die Elementfunktion, die beim Erzeugen eines Objekts aufgerufen wird, nennt man Konstruktor. In dieser Funktion können Sie dafür sorgen, dass alle Elemente des Objekts korrekt initialisiert sind. Der Konstruktor trägt immer den Namen der Klasse selbst und hat keinen Rückgabetyp, auch nicht void. Der Standardkonstruktor hat keine Parameter.

```
class Enemy {
    public:     //Zugriffsrecht public

                /**** Konstruktor ****/
                Enemy()
                {
                        health=0;
                        name="";
                        id=0;
                }

        string name;
                int id;
```

```
    private:     //Zugriffsrecht private
        int health;
};
```

Dekonstruktor

Im Falle einer Datumsklasse wäre es sinnvoll, dass der Konstruktor alle Elemente auf 0 setzt. Daran kann jede Elementfunktion leicht erkennen, dass das Datum noch nicht festgelegt wurde. Sie könnten alternativ das aktuelle Datum ermitteln und eintragen. Im Beispiel ist auch ein Destruktor definiert worden, obwohl er im Falle eines Datums keine Aufgabe hat.

```
class Enemy {
    public:     //Zugriffsrecht public

                /**** Destruktor ****/
                ~Enemy()
                {
                     cout << "Ressourcen von " <<  name << " wurden
freigegeben!";
                }

        char name[50];
                int id;


    private:     //Zugriffsrecht private
        int health;
};
```

Überladen von Konstruktor

Konstruktoren können genauso überladen werden wie normale Funktionen auch. Es kann neben dem Standardkonstruktor auch mehrere weitere Konstruktoren mit verschiedenen Parametern geben. Der Compiler wird anhand der Aufrufparameter unterscheiden, welcher Konstruktor verwendet wird.

```
class Enemy {
    public:     //Zugriffsrecht public


                /**** Konstruktor ****/
                Enemy()
                {
                     name="Max Mustermann";
                }
                Enemy(string n)
                {
```

```
                    name=n;
                }
                Enemy(string n, int h)
                {
                    name=n;
                    health=h;
                }
                /**** Destruktor ****/
                ~Enemy()
                {
                    cout << "Ressourcen von " <<  name << " wurden
freigegeben!";
                }

        char name[50];
                int id;


    private:     //Zugriffsrecht private
        int health;
};
```

Je nachdem wie viele Parameter beim Konstruktoraufruf übergeben werden, wird der passende Konstruktor ausgewählt. Existiert gar kein Konstruktor so wird vom Compiler ein leerer Konstruktor eingefügt.

```
int main (void)
{

    Enemy e;                    //führt den Standardkonstruktor Enemy() aus
-> name="Max Mustermann"
    Enemy e("Fritz Phantom");   //führt den 2. Konstruktor mit den
passenden Argumenten aus -> name="Fritz Phantom"
    Enemy e("Hans Wurst", 100); //führt den 3. Konstruktor mit den
passenden Argumenten aus -> name="Hans Wurst", health=100

    return 0;
}
```

Vererbung

Die Informatik steckt voller schöner Analogien. So hat die objektorientierte Programmierung den Begriff der »Vererbung« eingeführt, wenn eine Klasse von einer anderen Klasse abgeleitet wird und deren Eigenschaften übernimmt.

Basisklasse

Eine Klasse kann als Basis zur Entwicklung einer neuen Klasse dienen, ohne dass ihr Code geändert

werden muss. Dazu wird die neue Klasse definiert und dabei angegeben, dass sie eine abgeleitete Klasse der Basisklasse ist. Daraufhin gehören alle öffentlichen Elemente der Basisklasse auch zur neuen Klasse, ohne dass sie erneut deklariert werden müssen. Man sagt, die neue Klasse erbt die Eigenschaften der Basisklasse.

Spezialisierung

Durch die Elemente, die in der abgeleiteten Klasse definiert werden, wird die abgeleitete Klasse zu einem besonderen Fall der Basisklasse. Sie besitzt alle Eigenschaften der Basisklasse. Sie können neue Elemente hinzufügen. Am folgenden Beispiel wird deutlich, warum das Hinzufügen von Eigenschaften eine Spezialisierung ist.

Ein Beispiel

Aus Sicht eines Computerprogramms haben alle **Personen** (=Basisklasse)

- Namen
- Adressen und
- Telefonnummern.

Geschäftspartner haben darüber hinaus eine

- Bankverbindung.

Da die Geschäftspartner auch Personen sind, haben sie neben ihrer Bankverbindung auch Namen, Adressen und Telefonnummern (von der Basisklasse).

Einige Geschäftspartner können auch **Kunden** sein. **Kunden** haben zusätzlich zu den Eigenschaften eines Geschäftspartners noch eine

- Lieferanschrift.

Lieferanten sind keine Kunden, aber auch Geschäftspartner. Sie haben noch

- eine Anzahl an offenen Rechnungen.

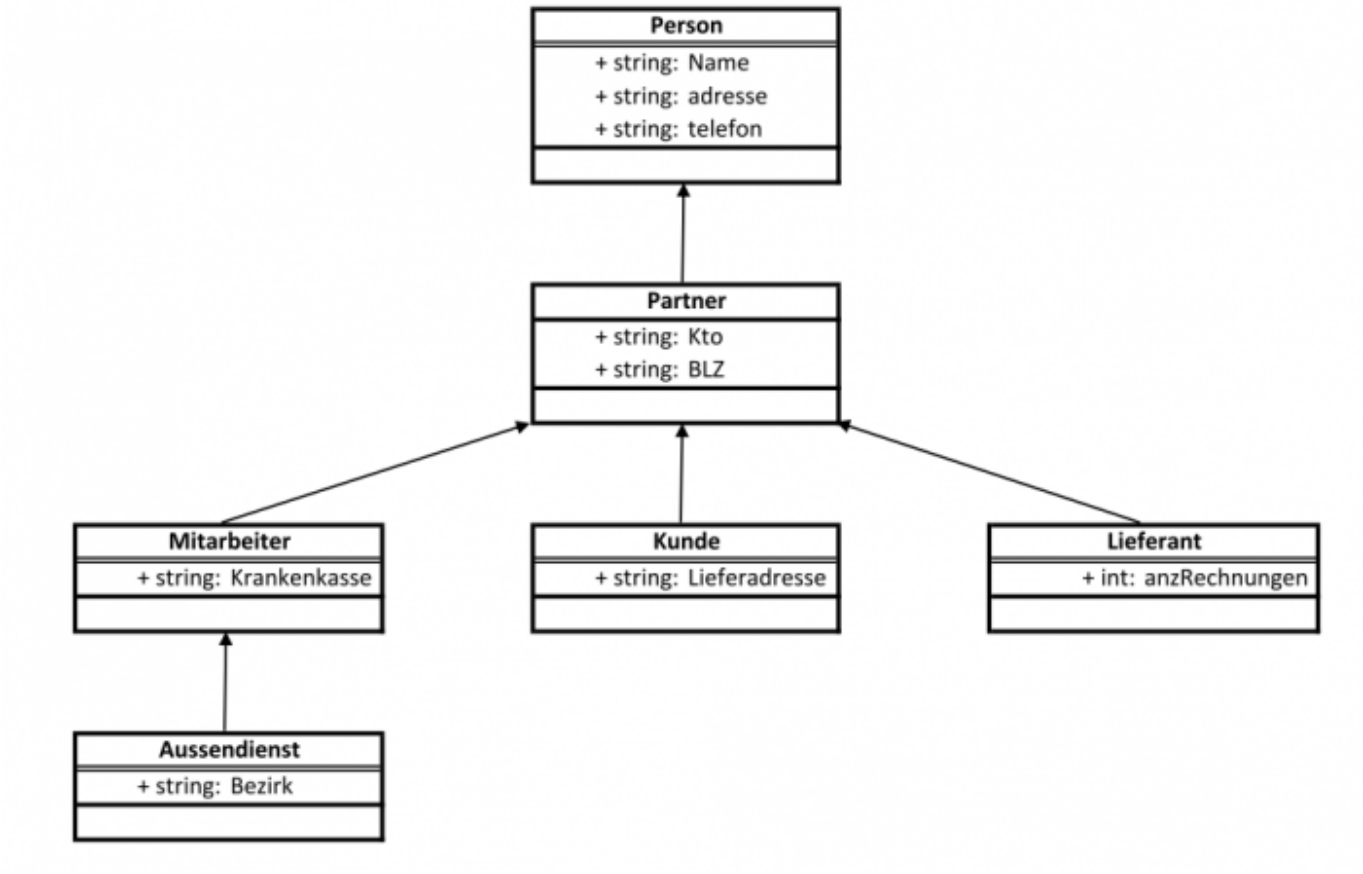
Selbst **Mitarbeiter** sind eigentlich Geschäftspartner, denn sie haben eine Bankverbindung. Darüber hinaus haben sie eine

- Krankenkasse.

Außendienstler haben alle Eigenschaften eines Mitarbeiters und zusätzlich ihren

- Bezirk.

Beispiel als UML-Diagramm:



Beispiel in C + +

```
class Person
{
    public:
       string Name, Adresse, Telefon;
};

class Partner : public Person
{
    public:
       string Kto, BLZ;
};

class Mitarbeiter : public Partner
{
    public:
       string Krankenkasse;
};

class Kunde : public Partner
{
    public:
       string Lieferadresse;
};

class Lieferant : public Partner
```

```
{
    public:
       int anzRechnungen;
};

class Aussendienst : public Mitarbeiter
{
    public:
       string Bezirk;
};

int main (void) {

   Aussendienst a1;

   a1.Bezirk="Amstetten";
   a1.Name="Hans Wurst";
   a1.Krankenkasse="Gebietskrankenkasse";
   a1.BLZ="14200";

   return 0;
}
```

Vorteil von Vererbungen:

Damit stellt die Ausgangsklasse Person die Verallgemeinerung dar und jede abgeleitete Klasse eine Spezialisierung. Der Vorteil dieser Technik ist, dass der bestehende Code der Basisklasse nicht noch einmal für die neu geschaffene Klasse geschrieben werden muss. Beispielsweise würde eine Prüffunktion der Adresse, die der Klasse Person hinzugefügt wird, automatisch auch allen anderen Klassen, die direkt oder indirekt von Person abgeleitet wurden, hinzugefügt, ohne dass eine Zeile Code mehr geschrieben werden müsste. Eine Änderung in der Klasse Mitarbeiter würde immer auch auf die Außendienstler durchschlagen. Es gibt aber keine Rückwirkung auf die Geschäftspartner.

Zusammenfassung

Die objektorientierte Programmierung hat einige neue Begriffe aufgebracht.

Objekt und Klasse

Der zentrale Begriff des Objekts bezeichnet einen Speicherbereich, der durch eine Klasse beschrieben wird. Prinzipiell kann sich der Anfänger ein Objekt als eine besondere Art einer Variablen vorstellen und die Klasse als die Typbeschreibung. Im Buch wird ein Objekt auch hin und wieder als Variable bezeichnet, insbesondere dann, wenn sich ein Objekt an dieser Stelle wie jede andere Variable verhält.

Attribut

Die Daten-Elemente einer Klasse werden in diesem Buch meist Elementvariable genannt. In der objektorientierten Literatur findet sich dafür auch die Bezeichnung Attribut. Damit wird angedeutet, dass die Elementvariablen die Eigenschaften eines Objekts beschreiben.

Methode und Operation

Die Funktionen einer Klasse, die hier als Elementfunktionen bezeichnet werden, finden sich in der objektorientierten Literatur unter dem Namen Methode oder Operation wieder. Diese Bezeichnung bringt zum Ausdruck, dass die Funktion nur über das Objekt erreichbar und damit eine Aktion des Objekts ist.

Konstruktor

Die Elementfunktion, die beim Erzeugen eines Objekts aufgerufen wird, nennt man Konstruktor. In dieser Funktion können Sie dafür sorgen, dass alle Elemente des Objekts korrekt initialisiert sind. Der Konstruktor trägt immer den Namen der Klasse selbst und hat keinen Rückgabetyp, auch nicht void. Der Standardkonstruktor hat keine Parameter.

Dekonstruktor

Im Falle einer Datumsklasse wäre es sinnvoll, dass der Konstruktor alle Elemente auf 0 setzt. Daran kann jede Elementfunktion leicht erkennen, dass das Datum noch nicht festgelegt wurde. Sie könnten alternativ das aktuelle Datum ermitteln und eintragen. Im Beispiel ist auch ein Destruktor definiert worden, obwohl er im Falle eines Datums keine Aufgabe hat

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:15:15_01

Last update: **2019/05/26 07:19**

C++

- 3.1) Zeiger (Wdhg. 7.Klasse)
  - 3.1.1) Zeiger-Übung (Wdhg. 7.Klasse)
- 3.2) Rekursionen
  - 3.2.1) Rekursion-Übung
- 3.3) Datenstruktur struct
- 3.4) Einfach verkettete Listen
- 3.5) Doppeltverkettete Listen
- 3.6) Binäre Bäume

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16

Last update: **2019/05/16 17:24**

Zeiger (Pointer)

Zeiger (engl. pointers) sind Variablen, die als Wert die Speicheradresse einer anderen Variable enthalten.

Jede Variable wird in C++ an einer bestimmten Position im Hauptspeicher abgelegt. Diese Position nennt man Speicheradresse (engl. memory address). C++ bietet die Möglichkeit, die Adresse jeder Variable zu ermitteln. Solange eine Variable gültig ist, bleibt sie an ein und derselben Stelle im Speicher.

Am einfachsten vergegenwärtigt man sich dieses Konzept anhand der globalen Variablen. Diese werden außerhalb aller Funktionen und Klassen deklariert und sind überall gültig. Auf sie kann man von jeder Klasse und jeder Funktion aus zugreifen. Über globale Variablen ist bereits zur Kompilierzeit bekannt, wo sie sich innerhalb des Speichers befinden (also kennt das Programm ihre Adresse).

Zeiger sind nichts anderes als normale Variablen. Sie werden deklariert (und definiert), besitzen einen Gültigkeitsbereich, eine Adresse und einen Wert. Dieser Wert, der Inhalt der Zeigervariable, ist aber nicht wie in unseren bisherigen Beispielen eine Zahl, sondern die Adresse einer anderen Variable oder eines Speicherbereichs. Bei der Deklaration einer Zeigervariable wird der Typ der Variable festgelegt, auf den sie verweisen soll.

```
#include <iostream>

int main() {
     int   Wert;      // eine int-Variable
     int *pWert;      // eine Zeigervariable, zeigt auf einen int
     int *pZahl;      // ein weiterer "Zeiger auf int"

     Wert = 10;       // Zuweisung eines Wertes an eine int-Variable

     pWert = &Wert;   // Adressoperator '&' liefert die Adresse einer
Variable
     pZahl = pWert;   // pZahl und pWert zeigen jetzt auf dieselbe Variable
```

Beispielhafte Speicherbelegung des Programms im Hauptspeicher:

Variable	Adresse	Wert
Wert	0x0001	10
*pWert	0x0005	0x0001
*pZahl	0x0009	0x0001
...	.	.
...	.	.

Der Adressoperator & kann auf jede Variable angewandt werden und liefert deren Adresse, die man einer (dem Variablentyp entsprechenden) Zeigervariablen zuweisen kann. Wie im Beispiel gezeigt, können Zeiger gleichen Typs einander zugewiesen werden. Zeiger verschiedenen Typs bedürfen einer Typumwandlung. Die Zeigervariablen pWert und pZahl sind an verschiedenen Stellen im Speicher abgelegt, nur die Inhalte sind gleich.

Wollen Sie auf den Wert zugreifen, der sich hinter der im Zeiger gespeicherten Adresse verbirgt, so verwenden Sie den Dereferenzierungsoperator *.

```
  *pWert += 5;
  *pZahl += 8;

   std::cout << "Wert = " << Wert << std::endl;
```

Beispielhafte Speicherbelegung des Programms im Hauptspeicher:

Variable	Adresse	Wert
Wert	0x0001	10 –> 15 –> 23
*pWert	0x0005	0x0001
*pZahl	0x0009	0x0001
...	.	.
...	.	.

Ausgabe:

```
Wert = 23
```

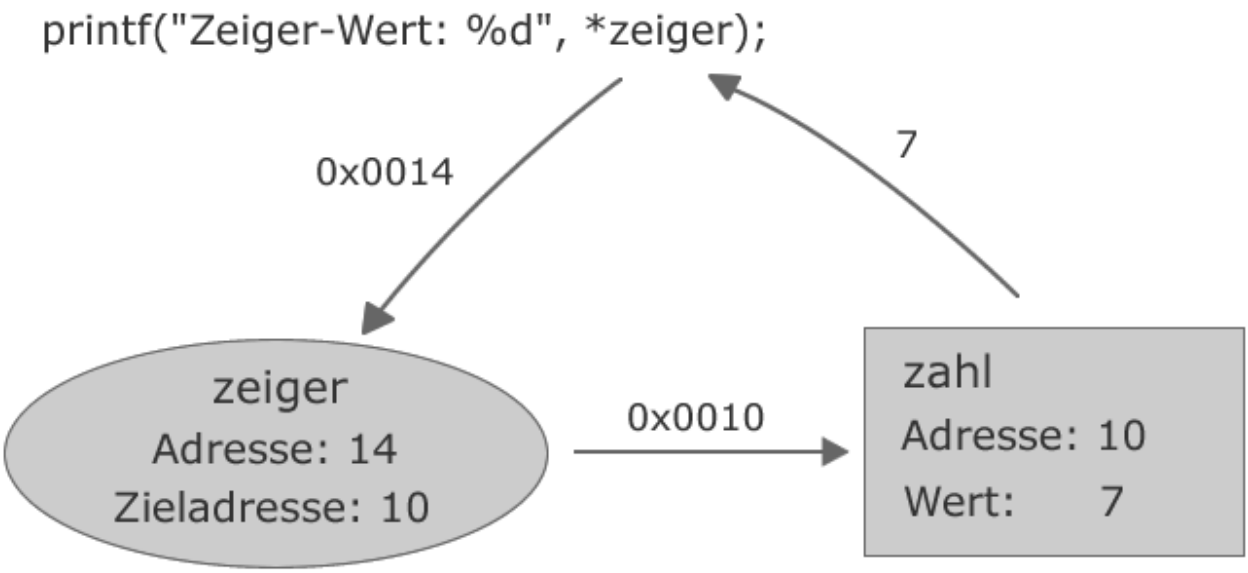
Man nennt das den Zeiger dereferenzieren. Im Beispiel erhalten Sie die Ausgabe Wert = 23, denn pWert und pZahl verweisen ja beide auf die Variable Wert.

Um es noch einmal hervorzuheben: Zeiger auf Integer (int) sind selbst keine Integer. Den Versuch, einer Zeigervariablen eine Zahl zuzuweisen, beantwortet der Compiler mit einer Fehlermeldung oder mindestens einer Warnung. Hier gibt es nur eine Ausnahme: die Zahl 0 darf jedem beliebigen Zeiger zugewiesen werden. Ein solcher Nullzeiger zeigt nirgendwohin. Der Versuch, ihn zu dereferenzieren, führt zu einem Laufzeitfehler.

Ein weiteres Beispiel....

```
int zahl = 7;
int *zeiger;
zeiger = &zahl;
printf("Zeiger-Wert: %d\n", *zeiger);
```

Ein **Zeiger repräsentiert eine Adresse** und nicht wie eine Variable einen Wert. Will man auf den Wert der Adresse zugreifen, auf die ein Zeiger zeigt, muss der Stern * vor den Namen gesetzt werden.



Zeiger auf Zeiger

Zeiger zeigen auf Adressen. Sie können nicht nur auf die Adressen von Variablen, sondern auch auf die Adressen von Zeigern verweisen. Dies erreicht man mit dem doppelten Stern-Operator * * .

```
int zahl=7;
int *zeiger = &zahl;
int **zeigerAufZeiger = &zeiger;

cout << "Wert von zeigerAufZeiger -> zeiger -> zahl:" << **zeigerAufZeiger;
```

Ausgabe

```
Wert von zeigerAufZeiger -> zeiger -> zahl: 7
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_01

Last update: **2019/05/16 17:22**

Aufgabe 3.1.1

Gib nach jeden Programmierbefehl alle Adressen und Inhalte der einzelnen Variablen aus!

```
int a=2, b=5, *c=&a, *d=&b;

a = *c * *d;
*d -= 3;
b = a * b;
c = d;
b = 7;
a = *c + *d;
```

Aufgabe 3.1.2

Gib nach jeden Programmierbefehl alle Adressen und Inhalte der einzelnen Variablen aus!

```
int a=2, b=5, *c=&a, *d=&b;
int **zz=NULL;

a = *c + *d;
zz=&d;
**zz=*zz-10;
*c *= 3;
b = a * *c;
c = d;
a = 7-*d;
b = *c * *d;
*c = *c + **zz;
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_01:16_01_01

Last update: **2019/05/16 17:22**

Rekursionen

Die **Rekursion** ist ein **spezieller Aufruf von Funktionen**, nämlich wenn Funktionen sich **selbst aufrufen**. Da bei einem Aufruf sich die Funktion wieder selbst aufruft, benötigt die Funktion wie bei den Schleifen eine **Abbruchbedingung**, damit die Selbstaufrufe nicht endlos sind.

Beispiel

In dem folgenden Beispiel, welches lediglich eine Bildschirmausgabe via Rekursion zeigt, wird der **Abbruch anhand einer Zählvariable** entschieden – wie bei den Schleifen. Ist x größer 0 erfolgt eine Ausgabe und ein rekursiver Aufruf mit einem dekrementierten (=verringerten) Zählwert. Ist der Zählwert bei 0 angelangt, erfolgt kein rekursiver Aufruf mehr.

```
#include<stdio.h>

using namespace std;

printLines(int x);

int main() {
    printLines(5);
    return 0;
}

printLines(int x){
    if(x > 0) {
        cout << "\nZeile Nr. " << x);
        printLines(x-1);
    }
}
```

Ausgabe des Programms

```
Zeile Nr. 5
Zeile Nr. 4
Zeile Nr. 3
Zeile Nr. 2
Zeile Nr. 1
```

Beispiel Fakultät

Nun ein sinnvolleres und gern verwendetes Beispiel, die **Berechnung der Fakultät mittels Rekursion**. Bei der Berechnung der Fakultät wird solange ein Produkt mit der dekrementierten Zahl gebildet, bis die Zahl bei der 1 angelangt ist. Zum Beispiel ist die Fakultät Vier: 4! = 4 * 3 * 2 * 1 =

24.

```
#include<stdio.h>

using namespace std;

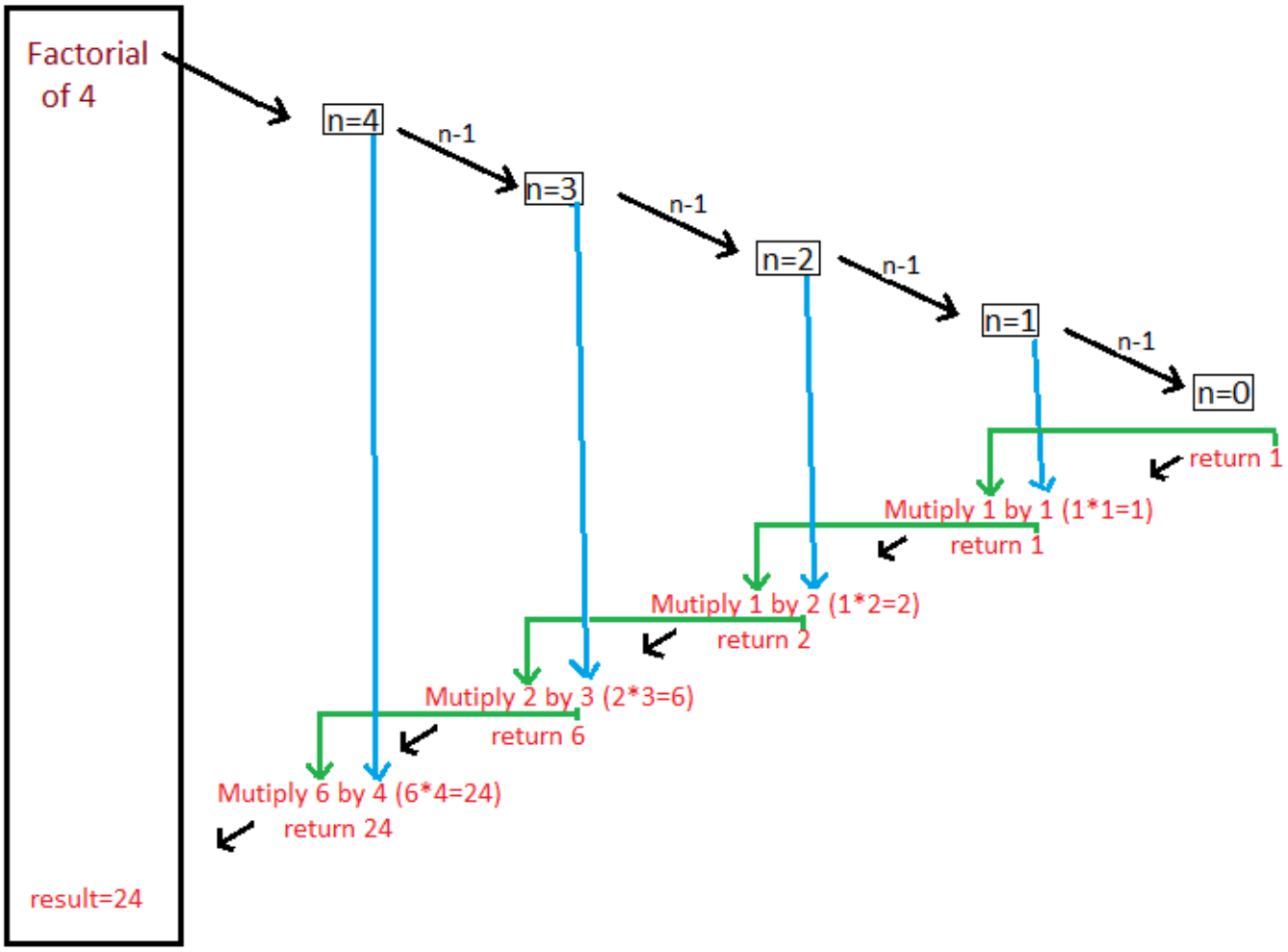
int fakultaet(int x);

int main() {
    int a = 6;
    cout << "Fakultaet von " << a << " ist " << fakultaet(a);
    return 0;
}

int fakultaet(int x) {
    if(x > 1) {
        return x * fakultaet(x-1);
    }else {
        return 1;
    }
}
```

Ausgabe

```
Fakultaet von 6 ist 720
```



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_02

Last update: **2019/05/16 17:22**

Aufgabe 3.2.1

Schreibe ein C + + - Programm, das die Summe aller Zahlen von 1 bis n (einer vom User eingegebenen Zahl) mithilfe einer Rekursion berechnet. Das Ergebnis der Summe soll im Hauptprogramm ausgegeben werden.

Beispiel

Eingabe:

```
6
```

Summe:

```
1+2+3+4+5+6 = 21
```

Aufgabe 3.2.2

Schreibe ein C + + - Programm, das die Potenzfunktion y = x^n mithilfe einer Rekursion berechnet. Der Funktionswert y, soll im Hauptprogramm ausgegeben werden.

Beispiel

Eingabe:

```
x=2
n=4
```

Ausgabe:

```
y=16 (2*2*2*2)
```

Aufgabe 3.2.3

Schreibe ein C + + - Programm, das die Fibonacci-Zahlen rekursiv berechnet. Die Fibonacci-Zahlen berechnen sich außer bei der Zahl 1 und 2 immer aus den beiden Vorgängern der Zahl.

```
z. B. 1, 2, 3, 5, 8, 13, 21, ...
```

Errechnet werden können sie mittels

```
1+2=3, 2+3=5, 3+5=8, 5+8=13.
```

Die Formel lautet also:

```
F(n+2)=F(n+1) +F(n)
```

Beispiel

Eingabe der Anzahl von Fibonacci-Zahlen:

```
6
```

Ausgabe:

```
1
2
3
5
8
13
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_02:16_02_01

Last update: **2019/05/16 17:22**

Struktur - eine zusammengesetzter Datentyp

Mit Arrays können Variablen gleichen Typs zusammengestellt werden. In der realen Welt gehören aber meist Daten unterschiedlichen Typs zusammen. So hat ein Auto einen Markennamen und eine Typbezeichnung, die als Zeichenkette unterzubringen ist. Dagegen eignet sich für Kilometerzahl und Leistung eher der Typ Integer. Für den Preis bietet sich der Typ float an. Bei bestimmten Autohändlern könnte auch double erforderlich sein. Alles zusammen beschreibt ein Auto.

Modell

Vielleicht werden Sie einwerfen, dass ein Auto noch mehr Bestandteile hat. Da gibt es Bremsscheiben, Turbolader und Scheibenwischer. Das ist in der Realität richtig. Ein Programm interessiert sich aber immer nur für bestimmte Eigenschaften, die der Programmierer mit dem Kunden zusammen festlegt. Unser Beispiel würde für einen kleinen Autohändler vielleicht schon reichen. Eine Autovermietung interessiert sich vielleicht überhaupt nicht für den Wert des Autos, aber möchte festhalten, ob es für Nichtraucher reserviert ist. Eine Werkstatt dagegen könnte sich tatsächlich für alle Teile interessieren. Ein Programm, das die Verteilung der Firmenfahrzeuge verwaltet, interessiert sich vielleicht nur für das Kennzeichen. Es entsteht also ein Modell eines Autos, das bestimmte Bestandteile enthält und andere vernachlässigt, je nachdem was das Programm benötigt. Bereits in C gab es für solche Zwecke die Struktur, die mehrere Variablen zu einer zusammenfasst. Das Schlüsselwort für die Bezeichnung solch zusammengesetzter Variablen lautet struct. Nach diesem Schlüsselwort folgt der Name des neuen Typen. In dem folgenden geschweiften Klammernblock werden die Bestandteile der neuen Struktur aufgezählt. Diese unterscheiden sich nicht von der bekannten Variablendefinition. Den Abschluss bildet ein Semikolon.

struct

Um ein Auto zu modellieren, wird ein neuer Variablentyp namens TAutoTyp geschaffen, der ein Verbund mehrerer Elemente ist.

```
struct TAutoTyp // Definiere den Typ
{
    char Marke[MaxMarke];
    char Modell[MaxModell];
    long km;
    int kW;
    float Preis;
};  // Hier vergisst man leicht das Semikolon!
```

Syntaxbeschreibung

Das Schlüsselwort struct leitet die Typdefinition ein. Es folgt der Name des neu geschaffenen Typs, hier TAutoTyp. In dem nachfolgenden geschweiften Klammerpaar werden alle Bestandteile der

Struktur nacheinander aufgeführt. Am Ende steht ein Semikolon, das man selbst als erfahrener Programmierer immer wieder einmal vergisst. Variablendefinition Damit haben wir den Datentyp TAutoTyp geschaffen. Er kann in vieler Hinsicht verwendet werden wie der Datentyp int. Sie können beispielsweise eine Variable von diesem Datentyp anlegen. Ja, Sie können sogar ein Array und einen Zeiger von diesem Datentyp definieren.

```
TAutoTyp MeinRostSammler; // Variable anlegen
TAutoTyp Fuhrpark[100];   // Array von Autos
TAutoTyp *ParkhausKarte;  // Zeiger auf ein Auto
```

Elementzugriff

Die Variable MeinRostSammler enthält nun alle Informationen, die in der Deklaration von TAutoTyp festgelegt sind. Um von der Variablen auf die Einzelteile zu kommen, wird an den Variablenname ein Punkt und daran der Name des Bestandteils gehängt.

```
// Auf die Details zugreifen
MeinRostSammler.km = 128000;
MeinRostSammler.kW = 25;
MeinRostSammler.Preis = 25000.00;
```

Zeigerzeichen

Wenn Sie über einen Zeiger auf ein Strukturelement zugreifen wollten, müssten Sie über den Stern referenzieren und dann über den Punkt auf das Element zugreifen. Da aber der Punkt vor dem Stern ausgewertet wird, müssen Sie eine Klammer um den Stern und den Zeigernamen legen.

```
TAutoTyp *ParkhausKarte = 0;      // Erst einmal keine Zuordnung
ParkhausKarte = &MeinRostSammler; // Nun zeigt sie auf ein Auto
(*ParkhausKarte).Preis = 12500;   // Preis für MeinRostSammler
```

Das mag zwar logisch sein, aber es ist weder elegant noch leicht zu merken. Zum Glück gibt es in C und C++ eine etwas hübschere Variante, über einen Zeiger auf Strukturelemente zuzugreifen. Dazu wird aus Minuszeichen und Größer-Zeichen ein Symbol zusammengesetzt, das an einen Pfeil erinnert.

```
ParkhausKarte->Preis = 12500;
```

L-Value

Strukturen sind L-Values. Sie können also auf der linken Seite einer Zuweisung stehen. Andere Strukturen des gleichen Typs können ihnen zugewiesen werden. Dabei wird die Quellvariable Bit für Bit der Zielvariable zugewiesen. TAutoTyp MeinNaechstesAuto, MeinTraumAuto; MeinNaechstesAuto = MeinTraumAuto;

Trotzdem die beiden Strukturvariablen nach dieser Operation ganz offensichtlich gleich sind, kann

man dies nicht einfach durch eine Anwendung des doppelten Gleichheitszeichen nachprüfen. Sie können bei Strukturen die Typdeklaration und die Variablendefinition zusammenfassen, indem der Name der Variablen direkt nach der geschweiften Klammer eingetragen wird.

```
struct // hier wird kein Typ namentlich festgelegt
{
    char Marke[MaxMarke];
    char Modell[MaxModell];
    long km;
    int kW;
    float Preis;
} MeinErstesAuto, MeinTraumAuto;
```

Hier werden im Beispiel die Variablen MeinErstesAuto und MeinTraumAuto gleich mit ihrer Struktur definiert. Werden auf diese Weise gleich Variablen dieser Struktur gebildet, muss ein Name für den Typ nicht unbedingt angegeben werden. Damit ist dann natürlich keine spätere Erzeugung von Variablen dieses Typs möglich. Initialisierung Auch Strukturen lassen sich initialisieren. Dazu werden wie bei den Arrays geschweifte Klammern verwendet. Auch hier werden die Werte durch Kommata getrennt.

```
TAutoTyp JB = {"Aston Martin", "DB5", 12000, 90, 12.95};
TAutoTyp GWB = {0};
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_03

Last update: **2019/05/16 17:22**

Lineare Listen (=Einfach verkettete Listen)

Einfach **verkettete Listen** oder **linked lists** sind eine **fundamentale Datenstruktur**, die ich hier anhand von Code-Beispielen und Grafiken erklären will.

Einfach verkettete Listen zeichnen sich dadurch aus, dass man besonders einfach Elemente einfügen kann, wodurch sie sich besonders gut für Insertion Sort eignen.

Vorteile

- Wenn das Suchen erledigt, der Einfügepunkt gefunden ist, ist der Aufwand fürs Einfügen an jeder Stelle O(1), wogegen beim Array O(n) Datensätze umkopiert werden müssen.
- Geringer zusätzlicher Speicherbedarf (1 Zeiger).

Nachteile

- Die Kosten fürs Suchen sind {\displaystyle O(n)} O(n), da ungünstigstenfalls über jeden Knoten iteriert werden muss.

Knoten

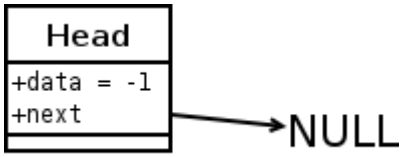
Eine einfach verkettete Liste besteht aus **Knoten, Englisch nodes**, die einen **Zeiger auf das nächste Element** und Daten beinhalten.

```
typedef struct list{

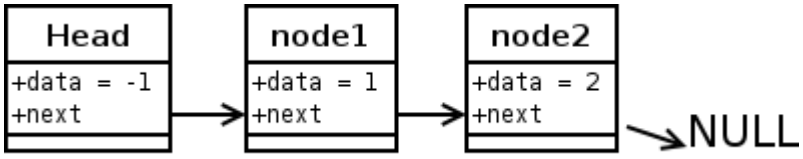
    int data;
    list *next;

};
```

Eine **leere Liste** besteht aus einem Kopf (Head) und nichts sonst:



Wenn man mehrere Elemente einfügt, sieht das so aus:



Eine einfach verkettete Liste mit einem Kopf und zwei Knoten.

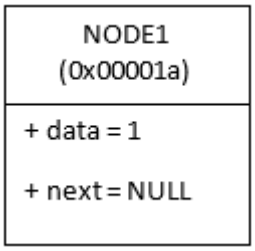
Knoten befüllen & einfügen

Wenn man einen Zeiger auf ein Element der Liste hat, ist es einfach, ein Element dahinter einzufügen.

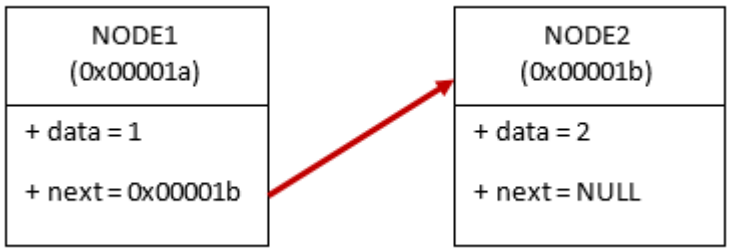
Dazu muss man den next-Zeiger der Liste auf das neue Element setzen, und den next-Zeiger des neuen Element auf den alten Wert des next-Zeigers der Liste:

```
    //1. Knoten erstellen
    list *node1;

    //1. Knoten befüllen
    node1->data=1;    //oder (*node1).data=1;
    node1->next=NULL;
```

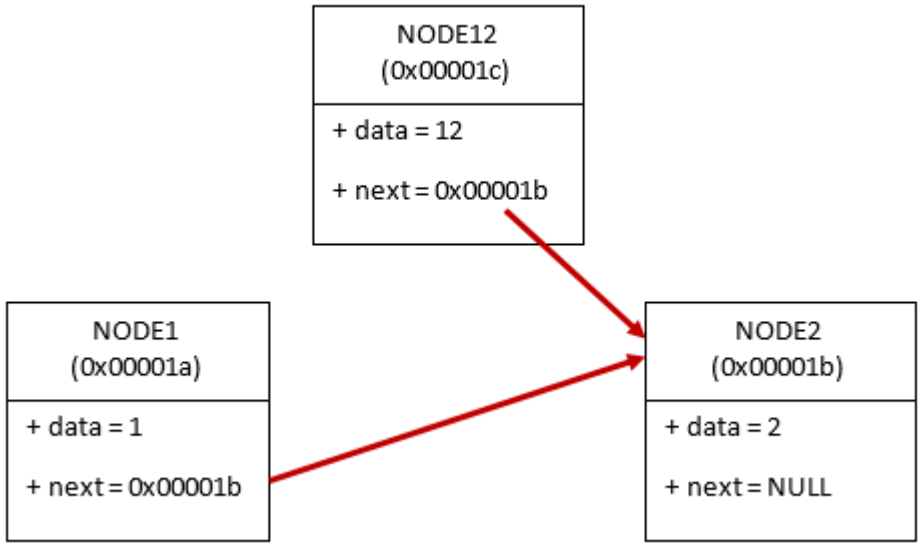



```
    //2. Knoten erstellen
    list *node2;
    node2->data=2;
    node2->next=NULL;
    //1. Knoten zeigt auf 2. Knoten
    node1->next=node2;
```

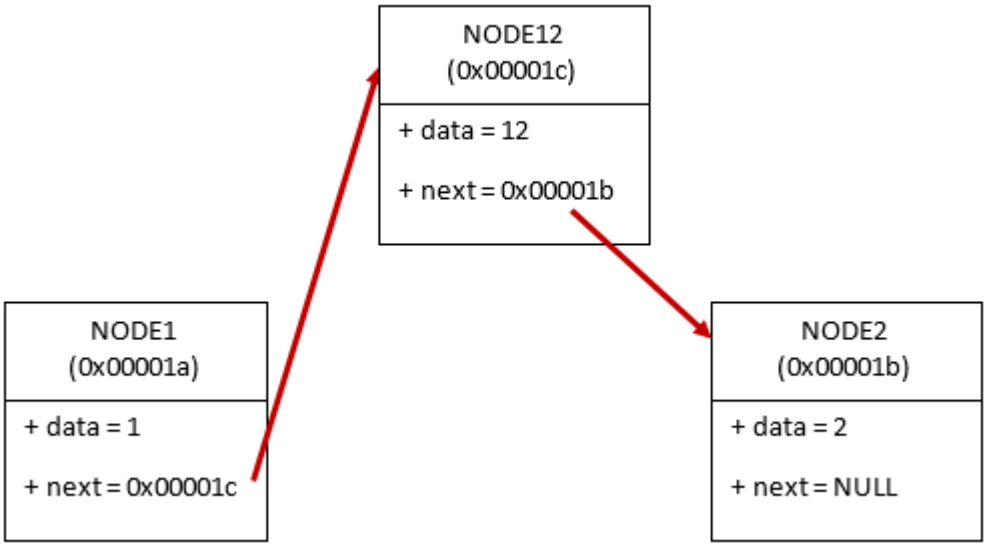



```
    //Knoten 12 zwischen Knoten 1 und 2 einfügen
    list *node12;
    node12->data=12;
    node12->next=node2;
    node1->next=node12;
```

Schritt 1



Schritt 2



Automatisiertes Hinzufügen von Knoten

Ein neuer Listenknoten wird durch Aufruf von new erzeugt. Dabei muss darauf geachtet werden, dass der Zeiger next gleich korrekt gesetzt wird. Wenn Sie nicht sofort den Nachfolger einhängen können, setzen Sie den Zeiger auf NULL.

```
#include <iostream>


using namespace std;
```

```
typedef struct list{

    int data;
    list *next;

};

int main(int argc, char** argv) {


    //Kopf der Liste
    list *head=NULL;


    //10 Knoten hinzufügen
    for(int i=1; i<10; i++)
    {
        list *node = new list;
        node->data=i;
        node->next=head;
        head=node;
    }


    //Knoten ausgeben
    list *help=NULL;
    help=head;

    while(help!=NULL)        //Solange Hilfszeiger nicht auf NULL zeigt
    {
        cout << help->data << endl;
        help=help->next;
    }

        return 0;
}
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_04

Last update: **2019/05/26 07:32**

Doppelt verkettete Listen

Doppelt verkettete Listen (oder doubly linked lists) sind häufig benutze Datenstrukturen und eine Verallgemeinerung der einfach verketteten Listen, die ich hier anhand von Beispielen in der Programmiersprache C++ vorstellen will.

Sie funktioniert in jeder Programmiersprache genauso, die Zeiger oder Referenzen und so etwas wie Klassen oder Structs zur Verfügung stellt.

Wozu?

Doppelt verkettete Listen, oder „double linked lists“ braucht man immer dann, wenn man sich in einer Liste leicht vorwärts und rückwärts bewegen können muss, und wenn man schnell und einfach Elemente der Liste an beliebigen Positionen löschen und neue einfügen muss.

Denn einfach verkettete Listen haben den Nachteil, dass man sie nur in einer Richtung durchlaufen kann.

Darüber hinaus kann man ein referenziertes Listenelement nicht unmittelbar löschen, da man von diesem Element keinen Zugriff auf das Vorgängerelement hat.

Doppelt verkettete Listen umgehen dieses Problem, indem sie in jedem Knoten zusätzlich noch eine Referenz auf den Vorgängerknoten speichern.

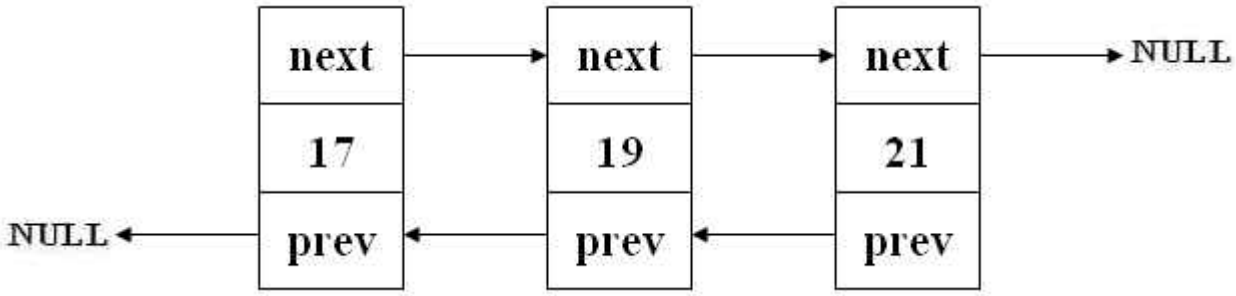
Die Grundidee

Im Gegensatz zu einfach verketteten Listen haben doppelt verkettete Listen in den Knoten eine zusätzliche Instanzvariable für die Referenz auf den Vorgängerknoten

Die Idee einer doppelt verketteten Liste ist es also, für jedes Element zwei Zeiger zu speichern, einen nach links und einen nach rechts:

```
struct DList{
    DList*left;
    DList *right;
    int data;
};
```

Die eigentlichen Daten, die in der Liste gespeichert werden sollen, stehen in data, das hier als int definiert ist. Jeder andere Datentyp, z.B. ein Zeiger auf beliebige andere Strukturen, funktioniert genau so.



Einfügen, Löschen und Ausgeben von Elementen

```
#include <iostream>

using namespace std;

struct DListe{
    DListe *next;
    DListe *prev;
    int zahl;
};


int main(int argc, char** argv) {

    DListe *head=NULL;        //Anlegen des Head-Pointers
    DListe *tail=NULL;        //Anlegen des Tail-Pointers
    DListe *help=NULL;        //Anlegen des Help-Pointers (für das Löschen
und Ausgeben)


    //Einfügen von 10 Elementen in eine doppelt verkettete Liste
    for(int i=0;i<10;i++)
    {
        DListe *elem=new DListe();        //Element erzeugen
        elem->zahl=i;                     //Attribut zahl befüllen
        if(head==NULL && tail==NULL)    //noch kein Element in der Liste
        {
            head=elem;                    //Head zeigt auf das neue Element
            tail=elem;                    //Tail zeigt auf das neue Element
            elem->next=NULL;            //Next zeigt auf NULL
            elem->prev=NULL;            //Prev zeigt auf NULL
        }
        else                            //Elemente sind bereits vorhanden &
Element wird am Ende eingefügt
        {
            tail->next=elem;            //tail->next auf das neue Element
            elem->prev=tail;            //elem->prev zeigt auf das
ursprünglich letzte Element
            elem->next=NULL;            //elem->next zeigt auf NULL
```

```
            tail=elem;                    //tail zeigt auf das neue letzte
Element
        }
    }

    cout << "Geben Sie an, welches Element (0-9) Sie loeschen wollen!" <<
endl;
    int loesche=0;
    cin >> loesche; //zB. Element mit der zahl=2 wird gelöscht

    help=head;                    //help zeigt auf head (Anfang der Liste)
    while(help->zahl!=loesche)    //Wenn help->zahl!=loesche, dann wandert
help in der Liste weiter
    {
        help=help->next;
    }
    //Überprüfe nochmals mit Ausgabe, ob beim richtigen Element (=2)
angekommen?
    cout << help->zahl;
    //Sonderfall 1. Element soll gelöscht werden (zahl=0)
    if(help==head)
    {
        head->next->prev=NULL;
        head=head->next;
        head->prev=NULL;
    }
    else if(help==tail)  //Sonderfall letztes Element soll gelöscht werden
    {
        tail=help->prev;
        help->prev->next=help->next;
    }
    else    //Element ist zwischen 2 anderen Elementen
    {
        help->prev->next=help->next;
        help->next->prev=help->prev;
    }

    //Ausgabe der doppelt verketteten Liste
    cout << endl << endl;
    help=head;
    while(help!=NULL)
    {
        cout << help->zahl << " -> ";
        help=help->next;
    }

    return 0;
}
```

Sortiertes Einfügen

```
      do{
        DListe *elem=new DListe();
        cout << endl << "Geben Sie ein Element ein, das sie einfuegen
wollen!" << endl;
        cin >> elem->zahl;
        elem->next=NULL;
        elem->prev=NULL;
        help=head;

        while((elem->zahl>help->zahl) && (help->next!=NULL))
        {
            help=help->next;
        }

        if(head==help)    //1. Fall => Element wird an erster Stelle
eingefügt
        {
            elem->next=help;
            help->prev=elem;
            head=elem;
        }
        else if(tail==help) //2. Fall => Element wird an letzter Stelle
eingefügt
        {
            elem->prev=help;
            help->next=elem;
            tail=elem;
        }
        else    //3. Fall => Element wird in der Mitte, also zwischen 2
anderen Elemente eingefügt
        {
            elem->next=help;
            help->prev->next=elem;
            elem->prev=help->prev;
            help->prev=elem;
        }

        //Ausgabe der Doppelt verketteten Liste
        ausgabe(head);

        cout << "Wollen Sie noch ein Element einfuegen (j/n)" << endl;
    }while(getch()=='j');
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_05

Last update: **2019/05/16 17:22**

(Binäre) Bäume

Was haben Bäume mit der Informatik zu tun?? – werden Sie Sich wahrscheinlich gefragt haben, als Sie zum ersten Mal erfahren haben, dass Bäume auch in der digitalen Welt existieren.

Sie werden sehen, dass Bäume in der Informatik einige Gemeinsamkeiten mit den natürlichen Bäumen haben. Zum Beispiel besitzen beide Blätter und Wurzel. Und genau so, wie es in der Natur etliche Arten von Bäumen gibt, existieren auch in der Informatik verschiedene Arten. Jede Art ist spezialisiert und für eine bestimmte Anwendung optimiert. Es gibt jedoch auch Unterschiede zwischen Bäumen der Informatik und natürlichen Bäumen: Sie „wachsen“ nicht in die gleiche Richtung. Bäume in der Informatik wachsen nach unten und haben dementsprechend die Wurzel oben wie ein natürlicher Baum, den Sie um 180 Grad drehen.

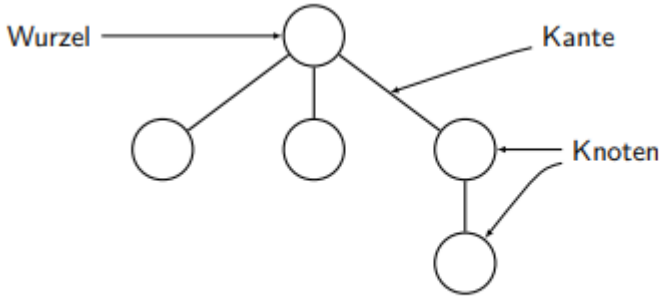


Abbildung Ein Baum mit der Wurzel, mehreren Knoten und Kanten.

Komponenten eines Baums

Ein **Baum (tree)** besteht aus verschiedenen Komponenten. Die wohl wichtigste ist der **Knoten**

Knoten (node) & Kanten (edges)

Ein **Knoten (node)** dient zur Speicherung von Daten. Knoten sind untereinander mit **Kanten (edges)** verbunden.

Wurzel (root) & Blatt (leaf)

Besitzt ein Baum keine Knoten, so ist dieser Baum **leer**. Besitzt er hingegen Knoten, so können diese vom Typ **Wurzel (root)**, **innerer Knoten** oder **Blatt (leaf)** sein.
Jeder nicht leere Baum besitzt genau eine **Wurzel (root)**, welche normalerweise oben gezeichnet wird. Da die Wurzel ganz oben ist, wächst der Baum nach **unten**.

Kind (child) & Vater (father)

Ist ein Knoten nach unten mit **anderen Knoten durch Kanten verbunden**, so sind diese unteren

Knoten seine **Kinder (child)**. Der obere Knoten wird als **Vater (father)** dieser Kinder bezeichnet.

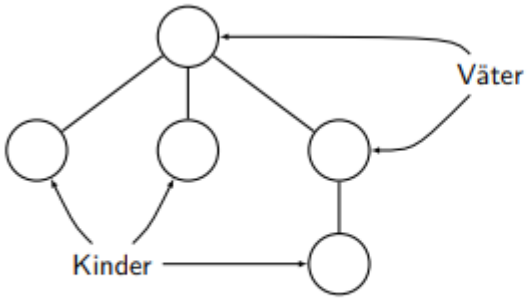


Abbildung Die Wurzel ist Vater von drei Kindern und eines ihrer Kinder (ganz rechts) ist selber Vater eines Kindes.

Ein **innerer Knoten (inner node)** ist ein Knoten mit mindestens einem Kind. Ist ein Knoten kinderlos, so spricht man von einem **Blatt (leaf)**.

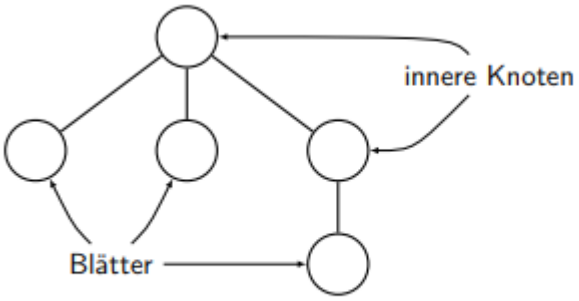


Abbildung Ein Baum mit inneren Knoten und Blättern.

Masse eines Baumes

Es gibt verschiedene Messgrößen, welche benutzt werden, um Eigenschaften eines Baumes zu beschreiben.

Ordnung (order)

Die **Ordnung (order)** gibt an, wie viele Kinder ein innerer Knoten *höchstens* haben darf. Hat ein Baum zum Beispiel die Ordnung 2, so dürfen seine Knoten maximal 2 Kinder haben. Ist jedoch nur die grafische Darstellung eines Baumes vorgegeben, so lässt sich seine Ordnung nicht ablesen. An Hand einer Grafik kann nur die Ordnung bestimmt werden, welche der Baum mindestens hat.

Grad (degree)

Der **Grad (degree)** eines Knotens sagt aus, wie viele Kinder ein Knoten effektiv hat. Dabei gilt, dass der Grad jedes Knotens kleiner oder gleich der Ordnung des Baumes sein muss. In der folgenden Abbildung haben zum Beispiel die Knoten a und b den Grad 2 und der Knoten e hat den Grad 5.

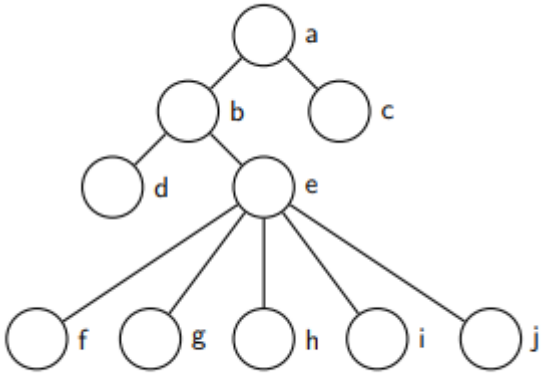


Abbildung Ein Baum mit Ordnung ≥ 5.

Pfad (path)

Der **Pfad (path)** von Knoten *v* nach Knoten *w* gib an, welche Kanten und Knoten besucht werden, wenn man von *v* nach *w* wandert.

Tiefe (depth)

Die **Tiefe (depth)** eines Knotens *v* gibt an, wie lange der Pfad zur Wurzel ist.

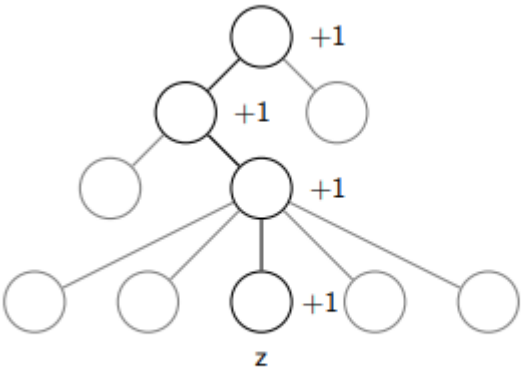


Abbildung Die Tiefe vom Knoten z ist $1 + 1 + 1 + 1 = 4$.

Dabei werden alle Knoten gezählt, welche auf dem direkten Pfad von der Wurzel zum Knoten *v* liegen - inklusive der Wurzel und dem Knoten *v* selbst.

Niveaus (level)

Knoten, die die gleiche Tiefe haben, werden zu **Niveaus (level)** zusammengefasst.

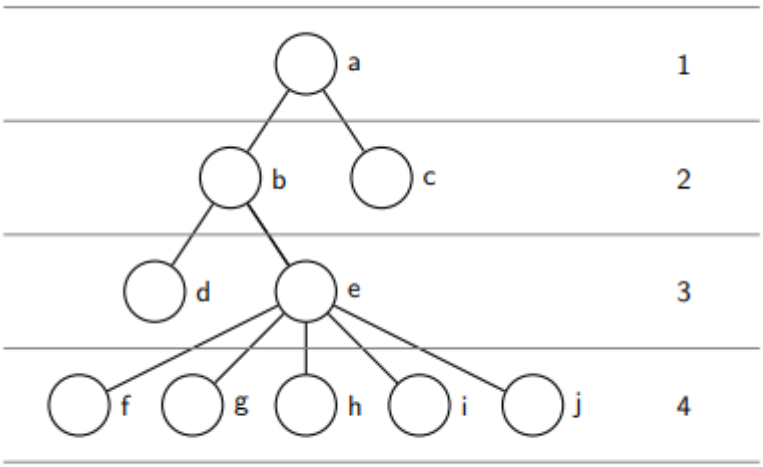


Abbildung Die Knoten d und e haben das gleiche Niveau von 3.

Höhe (height)

Die **Höhe (height)** eines Baumes entspricht der Tiefe des Knotens, welcher am weitesten von der Wurzel entfernt ist.
Somit entspricht die Höhe des Baumes dem größten Niveau aller Knoten des Baumes.

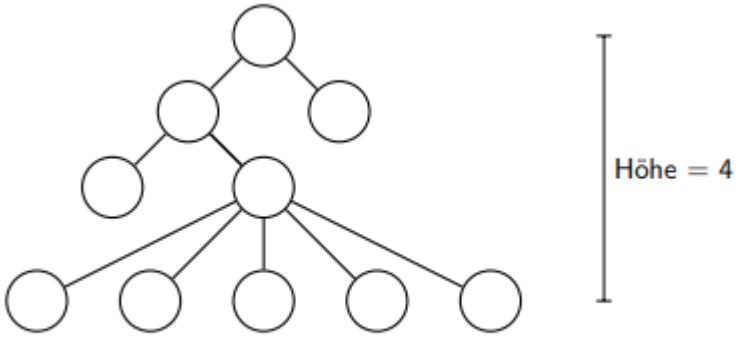


Abbildung Die Höhe dieses Baumes beträgt 4.

Ein Baum heißt ausgefüllt, wenn alle inneren Knoten die maximale Anzahl Kinder haben. Somit ist der Grad jedes Knotens entweder gleich der Ordnung des Baumes (innerer Knoten) oder 0 (Blatt).

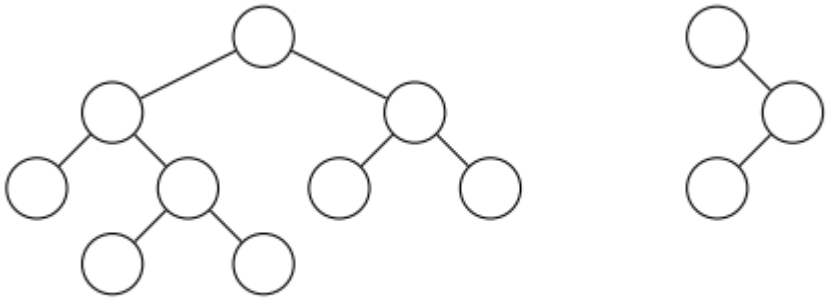


Abbildung Wenn beide Bäume Ordnung 2 haben, dann ist nur der Linke ein ausgefüllter Baum.

Vollständig

Ein Baum heißt **vollständig**, wenn jedes Niveau die maximale Anzahl an Knoten hat. Jeder vollständige Baum ist auch ausgefüllt.

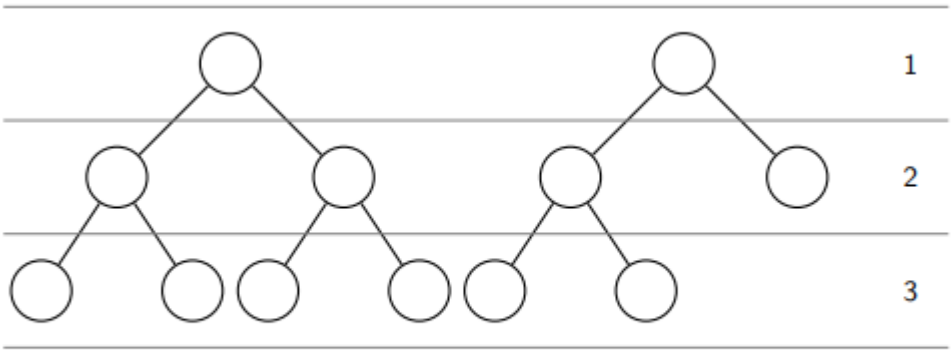


Abbildung Wenn beide Bäume Ordnung 2 haben, dann ist der linke Baum vollständig. Der rechte Baum ist zwar ausgefüllt, aber nicht vollständig.

Brüder/Geschwister

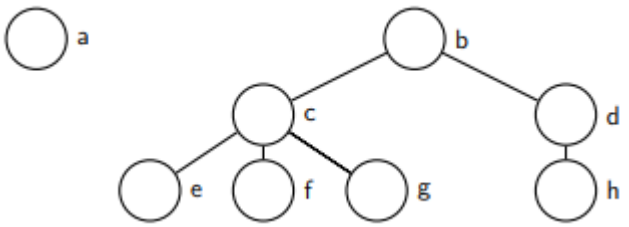
Knoten, welche den gleichen Vater haben, werden auch **Brüder** oder **Geschwister (sibling)** genannt.

⇒ Nun kennst du alle relevanten Begriffe der Bäume, welche für die anstehende Programmierung ein notwendige Grundlage sind.

Aufgaben

1)

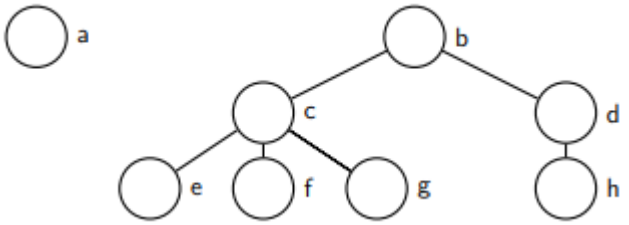
Bestimmen den Typ (Wurzel, innerer Knoten, Blatt, Vater, Kind) aller Knoten in folgender Abbildung. Beachte, dass einzelne Knoten auch von mehreren Typen sein können.



2)

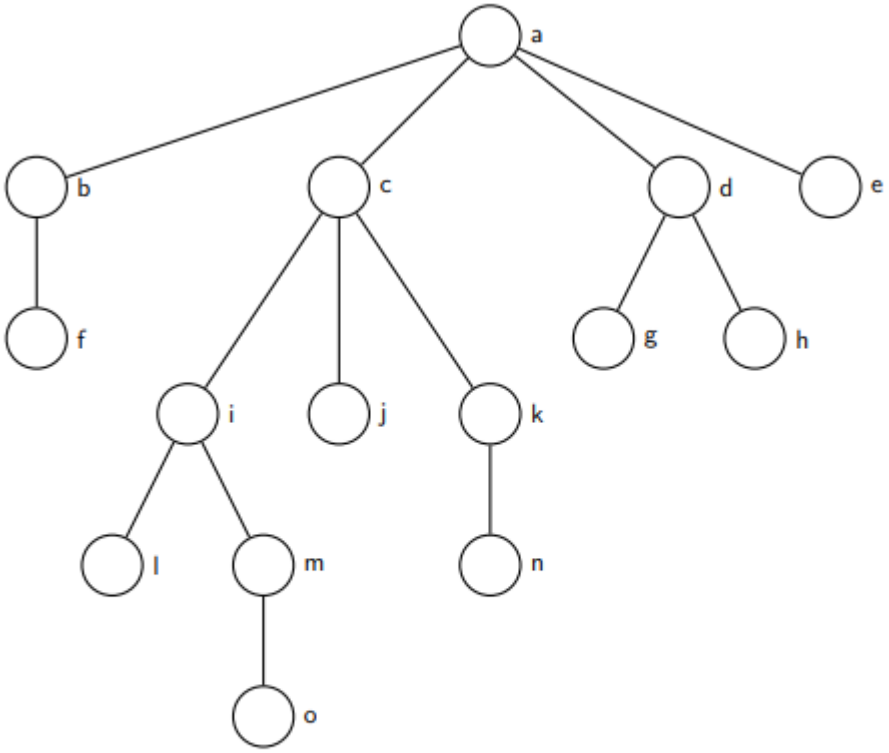
Bestimmen Sie für die folgenden zwei Bäume :

- a) die Höhe,
- b) die Ordnung,
- c) die Niveaus,
- d) ob der Baum ausgefüllt ist,
- e) den Grad aller Knoten
- f) und die Tiefe jedes Knotens



3)

Bestimmen Sie im folgenden Baum die Typen der Knoten. Füllen Sie dazu die vorgegebene Tabelle aus.



	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
Wurzel															
Blatt															
Innerer Knoten															

4)

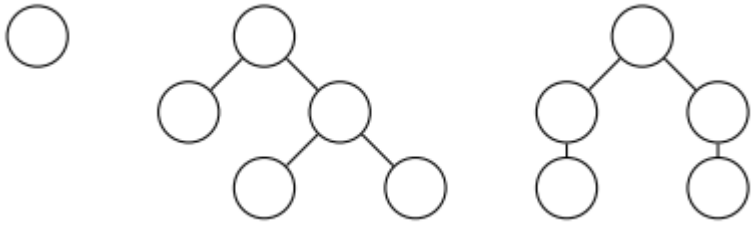
Bestimmen Sie für jeden Knoten des Baumes der Aufgabe 3 die Tiefe und den Grad.

5)

Zeichnen Sie im Baum der Aufgabe 3 die Niveaus und die Höhe ein. Überlegen Sie sich, welche Ordnung dieser Baum haben könnte, und begründen Sie ihre Entscheidung.

6)

Gegeben sind die drei folgenden Bäume der Ordnung 2. Entscheiden Sie, welche Bäume vollständig und welche ausgefüllt sind, und begründen Sie Ihre Antwort



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:16:16_06

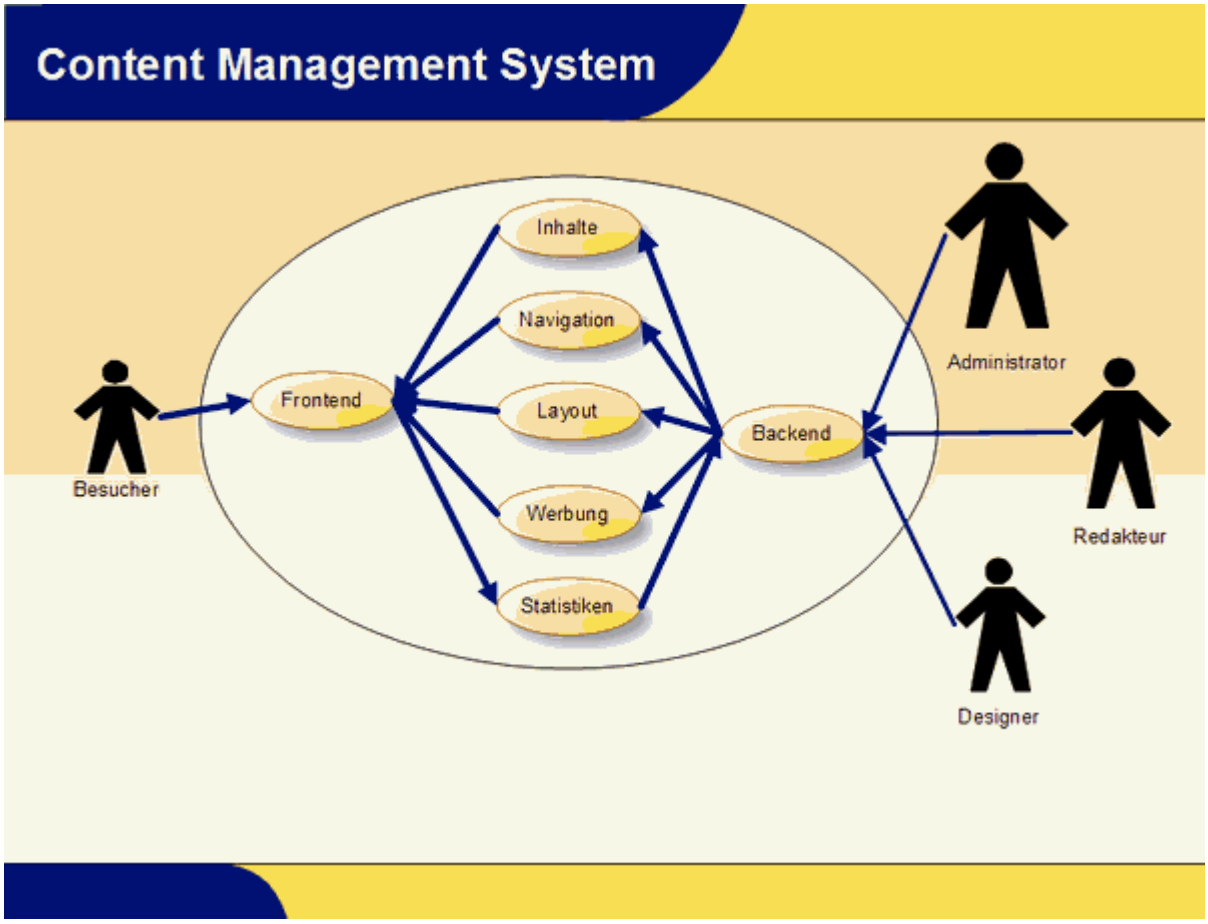
Last update: **2019/05/26 07:37**

17) Content Management System (CMS-System)

Ein CMS, ein **Content-Management-System**, dient zur **Verwaltung der Inhalte (Einpflege von Texten, Bildern, Videos und Daten)**. Ein WCMS (Web Content Management System) verwaltet Inhalte von einer oder auch mehrerer **Websites**. WCMS sind **mandantenfähig**, d.h. sie können mehrere Kunden (Mandanten) bedienen, ohne dass diese Einblick in die Benutzerverwaltung des anderen haben. In weiterer Folge, wird immer von einem WCMS ausgegangen, jedoch der Einfachheit halber wird immer der Begriff CMS verwendet.

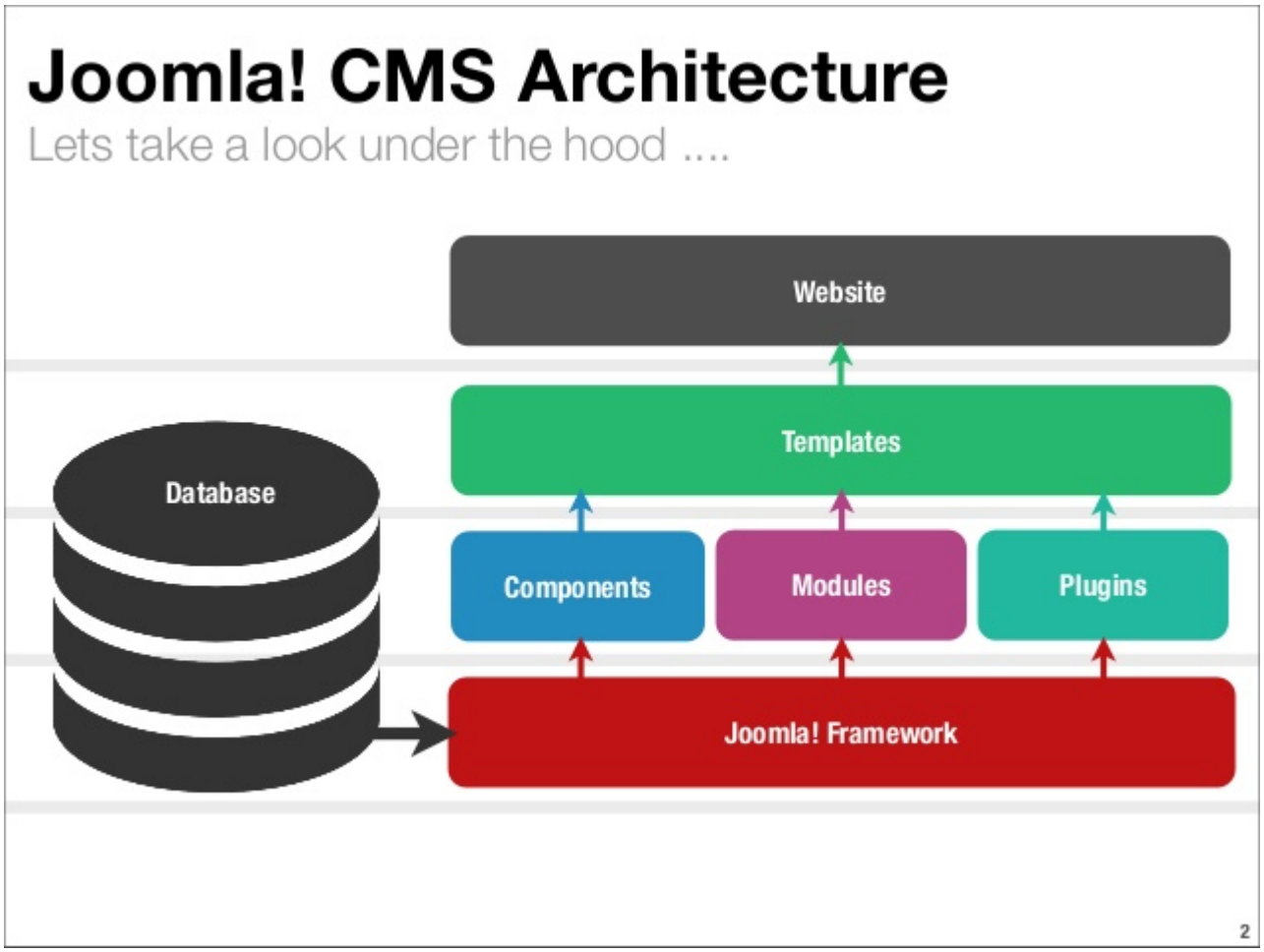
Frontend & Backend eines CMS

Ein CMS wird meist in ein Frontend- und ein Backend-Bereich unterteilt. Das Frontend eines CMS (wörtlich übersetzt „Vorder-Ansicht“) ist die Webseite, die ein Besucher angezeigt bekommt.



Das Backend eines CMS (wörtlich übersetzt „Hinter-Ansicht“) ist ein geschützter Bereich, in dem die Administration des CMS stattfindet. Hier stellt das CMS nach einer Anmeldung alle Funktionen zur Verfügung, die der jeweilige Benutzer zum Verwalten der Webseite benötigt. Häufig können für unterschiedliche CMS-Benutzer auch unterschiedliche Zugriffsrechte festgelegt werden, z.B. der Designer auf die Templates des CMS, der Redakteur auf die Inhalte des CMS.

CMS-Architektur anhand von Joomla



Datenbank

Alle Inhalte (Text, Menüs, Benutzer, ...) außer Bilddateien & Dokumente sind in einer Datenbank (z.B. MySQL) gespeichert.

Framework

Das Joomla-Framework (Rahmenstruktur/Programmiergerüst) ist ein modulares PHP-Framework und bietet eine Auswahl an vorgefertigten Funktionen und Bibliotheken, die einfach anzupassen bzw. leicht zu erweitern sind.

Extensions

Durch den modularen Aufbau des Joomla-Framework kann man dieses einfach mithilfe von Components, Modules, Plugins oder Templates erweitern.

Components

Komponenten sind die Anwendungen innerhalb von Joomla!. Unter einer Joomla! Komponente versteht man eigene Applikationen, die innerhalb von Joomla! laufen. Einige dieser Komponenten sind bereits in Joomla! integriert wie zum Beispiel das Artikelsystem zur Darstellung der Artikel oder eine einfache Weblinks-Verwaltung.

Komponenten sind also Anwendungen zur Verarbeitung und Darstellung von Informationen. Diese Komponenten unterliegen dabei einer eigenen Administration, in der Sie Ihre Daten verwalten oder auch die Verarbeitungsweise von vom Besucher eingegeben Informationen bestimmen.

Neben den bereits in Joomla! integrierten Komponenten findet man auf dem Markt eine Vielzahl kostenloser (das heißt lizenzfreier), aber auch einige kostenpflichtige Komponenten. Sehr bekannt sind Komponenten wie Foren, Gästebücher und Kommentarmöglichkeiten. Aber auch komplexe spezifische Anwendungen wie Kleinanzeigenmärkte, Buchungssysteme und Übersetzungstools für multilinguale Websites sind erhältlich.

Module

Unter Modulen bei Joomla! versteht man die Tools zur Anzeige von Informationen an verschiedenen Stellen auf der Seite. Diese Informationen bezieht Joomla! dabei in der Regel aus der Datenbank. Das heißt, sämtliche Anwendungen (Joomla! selbst, andere Komponenten), die ihre Daten in der Datenbank speichern, können abgefragt werden. Typische Beispiele sind das „wer ist online“-Modul, das die Anzahl der Besucher anzeigt oder das „latest news“-Modul, das die letzten Nachrichten kompakt in einer Liste anzeigt.

Module können aber auch dafür genutzt werden, selbst definierten Inhalt anzuzeigen: beispielsweise ein einfaches Bild oder einen eigenen Text.

Diese Module werden meist neben dem Hauptinhalt in einer Spalte angezeigt und lockern damit das Erscheinungsbild auf und liefern weitere Informationen.

Plugins

Plugins haben vielerlei Funktionen und arbeiten teilweise mit Komponenten zusammen. Der Editor ist beispielsweise ein Plugin, ebenso wie die Möglichkeit, Bilder einzubinden. Erweiterungen für alle drei Elemente, zum Beispiel umfangreichere Editoren oder erweiterte Sprachpakete, lassen sich zusätzlich installieren.

Templates

Unter Template versteht man nicht nur bei Joomla! den gestalterischen „Rahmen“ der Website.

Der Sinn eines Content Management Systems ist die Trennung von Inhalt und Design. In der Datenbank stehen nur die reinen Texte, das Template sorgt vor der Ausgabe auf dem Bildschirm dafür, dass diese - wie vom Designer festgelegt - „passend“ angezeigt werden.

Sinn und Zweck dieser Trennung liegt auf der Hand: Auch nicht mit der Web-Programmierung vertraute Nutzer können die Inhalte pflegen und müssen sich nicht um die Darstellung kümmern. Der Designer legt verschiedene Formatierungen fest, aus denen der Nutzer wählen kann (klein, groß, Überschrift, fett etc.), so dass diese auf der gesamten Website konsistent angezeigt werden.

Natürlich ist ein solches Template noch mehr: Es bestimmt den generellen Aufbau der Seite, ist also auch für den Gesamteindruck Ihrer Website verantwortlich. Denn die besten Inhalte nutzen nichts, wenn diese aufgrund einer nicht durchdringbaren Struktur unauffindbar bleiben. Ebenso sorgt das Template für den stimmigen Transport Ihrer CI.

Website

Zum Schluss entsteht eine Webseite, mit der die Benutzer & Besucher interagieren können.

Vorteile

Dezentralisierte Wartung

Ein gutes webbasiertes Content Management System funktioniert mit jedem gewöhnlichen Webbrowser (zum Beispiel Internet Explorer, Mozilla Firefox oder Opera). Sie benötigen also keine zusätzliche Software und können ohne Engpässe Ihre Internet-Präsentation warten und erweitern, wann und wo Sie wollen.

Trennung Inhalt & Gestaltung

Zum einen werden bei einem CMS die G**estaltung der Webseite (Layout & Design)** und die **Verwaltung der Inhalte getrennt** verwaltet. Dadurch muss man sich beim Einsatz eines CMS nicht bei Inhalten um das Layout kümmern und umgekehrt. Außerdem kann so **mit wenigen Klicks** im CMS das **Layout der kompletten Webseite geändert** werden.

Für Autoren ohne technischen Background

Wer Textverarbeitung beherrscht, kann auch in einem modernen CMS System online Inhalte erstellen, Bilder veröffentlichen oder neue Produkte anlegen. Über ein modernes Content Management System können Sie ohne HTML-Kenntnisse Ihre Homepage pflegen.

Konfigurierbare Zugriffsbeschränkung

Ein CMS System beinhaltet in der Regel eine eigene Benutzerverwaltung. Benutzern können Rollen und Berechtigungen zugewiesen werden, wodurch die unautorisierte Veränderung von Inhalten effektiv verhindert wird.

Einhalten von Design-Vorgaben

In einem CMS können Inhalte vom Design getrennt gespeichert werden. Folglich wird der gesamte Content aller Autoren in einem Design zentral ausgegeben.

Automatische Navigations-Generierung

Menüs können in CMS Lösungen aus den Datenbankinhalten automatisch erzeugt bzw. generiert werden. Hyperlinks werden nur dargestellt, wenn sie auf gültige Seiten verweisen. Wenn beispielsweise diese Seiten gelöscht werden, können auch die Hyperlinks automatisch ausgeblendet werden.

Speicherung der Inhalte in einer Datenbank

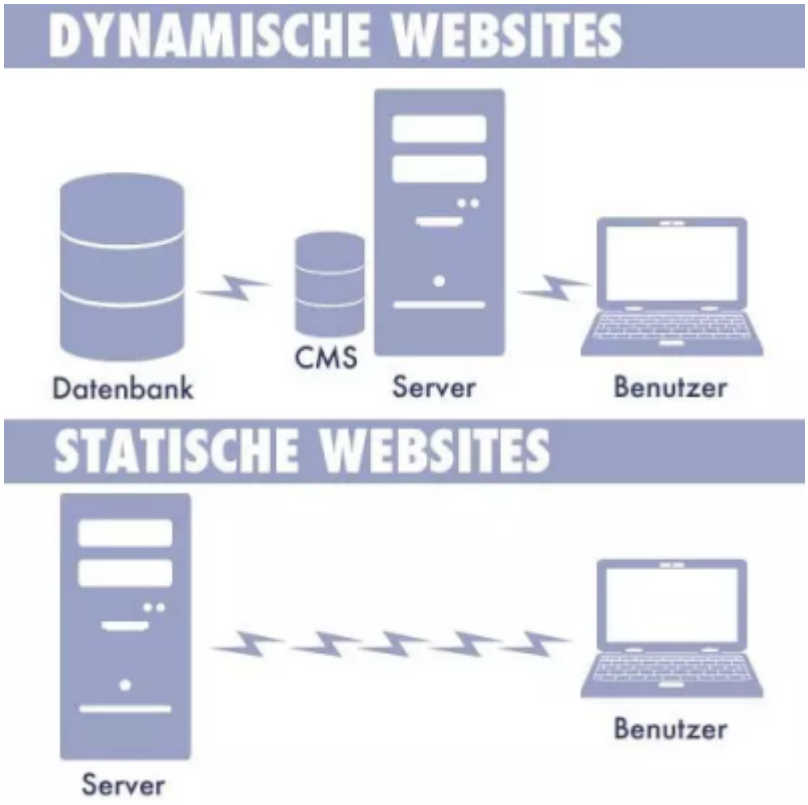
In einem guten CMS System werden die Inhalte in der zentraler Datenbank abgelegt. Eine zentrale Content-Speicherung bedeutet, dass Inhalte an verschiedenen Stellen wiederverwendet werden und für das jeweilige Medium (Webbrowser, Mobiltelefon/Wap, PDA, Print) adäquat formatiert werden können.

Dynamische Inhalte

Modulare Erweiterungen wie z.B. Foren, Umfragen, Shops, Anwendungen, Suchfunktionen und News-Management stehen als Module zum Einsatz bereit.

Content just in time

Die Publikation von Inhalten ist zeitgenau steuerbar. Inhalte können im Hintergrund vorbereitet und nur von berechtigten Usern in der sog. „hidden preview“ vorab eingesehen werden.



Nachteile

Eingeschränkte Gestaltung der Website

Durch die festgelegte Struktur eines CMS bestehen nicht vielen Freiräume was Widgets oder Features zur Integration in die Website betrifft. Es können nur verfügbare Features in die Website integriert werden.

Abhängigkeit vom Dienstleister

Entscheidet man sich für ein CMS, so ist man von der Aktualisierung und Service der Software vom jeweiligen Hersteller des Website-CMS abhängig.

Sicherheitsaspekt

Gerade **weitverbreitete Systeme (TYPO3, Joomla, Wordpress,...)** werden (genau wie bei den Betriebssystemen auch) natürlich **mit Vorliebe gehackt**. **Sicherheitslücken verbreiten sich schnell** in der Szene. Die vollständige Absicherung des Systems - zum Beispiel durch Ändern aller üblichen Installationspfade - kann sehr schnell sehr aufwändig werden. Allerdings sind bei den oben genannten Systemen auch die Entwickler sehr schnell mit der Erstellung von Patches. Es bedarf allerdings dann auch der entsprechenden Beobachtung der bekannten Foren und Mailinglisten.

Geringere Performance

Da die Inhalte dynamisch beim Aufruf der Seite generiert werden, kann der Auslieferung der kompletten Seite zumeist nur langsamer als bei einer statischen HTML-Seite erfolgen. Gute CMS-Systeme bieten hier allerdings geeignete Caching-Lösungen (Zwischenspeicher) an, die diesen Nachteil wieder ausgleichen.

Mehr Funktionalität als benötigt

Ein CMS muss als Gesamtsystem angesehen werden, in dem sich viele Programmteile bedingen oder voneinander abhängen. Will man das System um nicht benötigte Funktionionen reduzieren kann es häufig zu unerwünschten Nebenwirkungen kommen.

Aufwendigere Backups

Genügt bei statischen Seiten meist ein einfaches Kopieren der Inhalte per FTP, so ist ein Backup eines CMS aufwändiger. Hier muss zusätzlich die Datenbank gesichert werden, sowie eventuell auch Systemeinstellungen des Servers. Auch das Rückspielen von Backups im Falles eines Datenverlustes sollte in regelmäßigen Abständen getestet werden.

Verbreitete CMS-Systeme

WordPress

Ursprünglich ein reines Blogsystem ist es heute mit über 70 Mio. Installationen das am weitesten verbreitete CMS.

Vorteile

- Installations- und Einrichtungsaufwand überschaubar
- Riesige Auswahl an kostenlosen bzw. -günstigen Designs
- Vielfältige Erweiterungsmöglichkeiten durch Plugins
- Ideal für SEO-Maßnahmen (sehr gute Tools: z.B. YOAST)

Nachteile

- WordPress braucht jede Menge Systemressourcen
- Ladegeschwindigkeit bei hohem Traffic oft langsam
- Viele Updates, teilweise mit Sicherheitsrisiken

TYPO3



Content-Management-Systeme: Typo3-LogoDas „Open Source Flaggschiff“ mit mehr als 9 Mio. Installationen und 500.000+ Websites.

Vorteile

- Weitverbreitet, viele Experten und Entwickler
- Viele Funktionen
- Erheblich schneller als WordPress, auch bei hoher Besucherfrequenz

Nachteile

- Das Setup ist komplex und für den Laien ungeeignet
- Auch bei Backend-Anpassungen sind profunde Kenntnisse im Administrationsbereich erforderlich

---

Joomla

Content-Management-Systeme: Joomla-LogoIst schon lange auf dem Markt und hat mit 1,2 Mio.+ Downloads eine eingeschworene Fangemeinde. Durch einen Entwicklungsstopp vor ein paar Jahren ist Joomla ein wenig ins Hintertreffen geraten. Derzeit holt dieses CMS allerdings wieder mächtig auf.



Vorteile

- Installation und Einrichtung einfach und gut dokumentiert
- Viele Erweiterungen und vorgefertigte Designs (ähnlich wie bei WordPress)

Nachteile

- Beliebtes Ziel von Hackern, nicht zuletzt aufgrund der vielen Erweiterungsmöglichkeiten (gilt übrigens auch für WordPress)
- Beim Rollen- und Rechtemanagement gibt’s Luft nach oben

⇒ 2.01) Joomla Installation

Drupal

Content-Management-Systeme: Drupal-LogoDrupal ist ein Baukaustensystem mit einer riesigen Auswahl an Features. Obwohl es unter die Content-Management-Systeme fällt, ist es eigentlich ein Content-Management-Framework (CMF). Drupal funktioniert im Grunde wie Lego – man kann alles damit bauen.



Vorteile

- Drupal verfügt über ein differenziertes Rollen- und Rechtesystem
- Es hat jede Menge Funktionen, die als Baustein in das System integriert werden können
- Das Backend ist voll individualisierbar

Nachteile

- Das Drupal-Setup ist vergleichsweise kompliziert
- Eingriffe im Backend und am Server erfordern Routine

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:17

Last update: **2019/05/14 21:55**

18) NETZWERKE

Allgemeine Netzwerkgrundlagen

- 18.0) Grundlagen
- 18.1) Topologien
- 18.2) Übertragungsmedien
- 18.3) Ethernet + Zugriffsverfahren CSMA/CD
- 18.4) Schichtenmodell
- 18.5) Netzwerkgeräte
- 18.6)Adressierung
    - 18.6.1) Adressierung Übungen
- 18.7) Routing
- 18.8) Netzwerkbefehle
- 18.9) Protokolle

Übungsaufgaben (Praxis)

- 18.10) Aufbau von Netzwerken
    - Routing
- 18.11) Netzwerkübungen (Subnetting, DNS-Server, DHCP-Server, WEB-Server, Mail-Server)

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

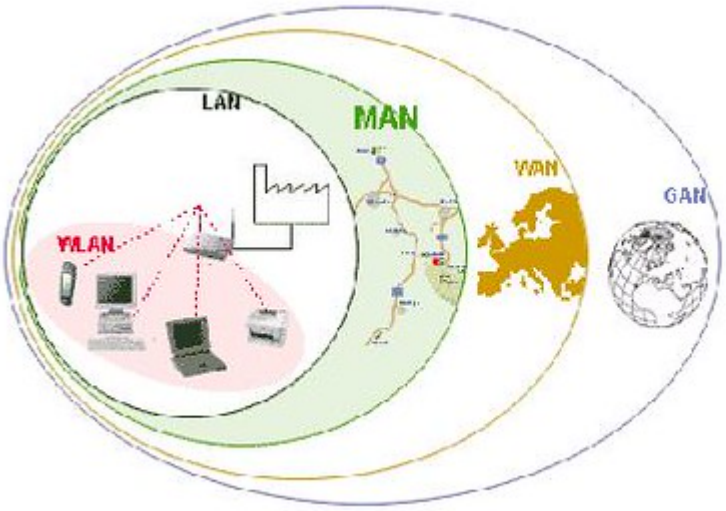
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18

Last update: **2019/05/16 18:33**

Netzwerk-Grundlagen

Größenordnung von Netzwerken

- **LAN** (Local Area Network): lokale Netze (meist innerhalb eines Gebäudekomplexes)
- **WAN** (Wide Area Network): große bis weltumspannende Netze; Beispiel: Telefonnetz, ISDN-Netz, VNET (IBM-eigenes Netzwerk)
- Der Begriff **MAN** (Metropolitan Area Network) ist eigentlich öffentlichen Netzen vorbehalten; in letzter Zeit verwenden aber auch Anwender mit vielen vernetzten Betriebsstellen (Banken) diesen Ausdruck.
- Netzwerke wie das Internet (die aus vielen, weltweit miteinander verbundenen Netzwerken bestehen), werden manchmal auch als **GAN** (Global Area Network) bezeichnet.

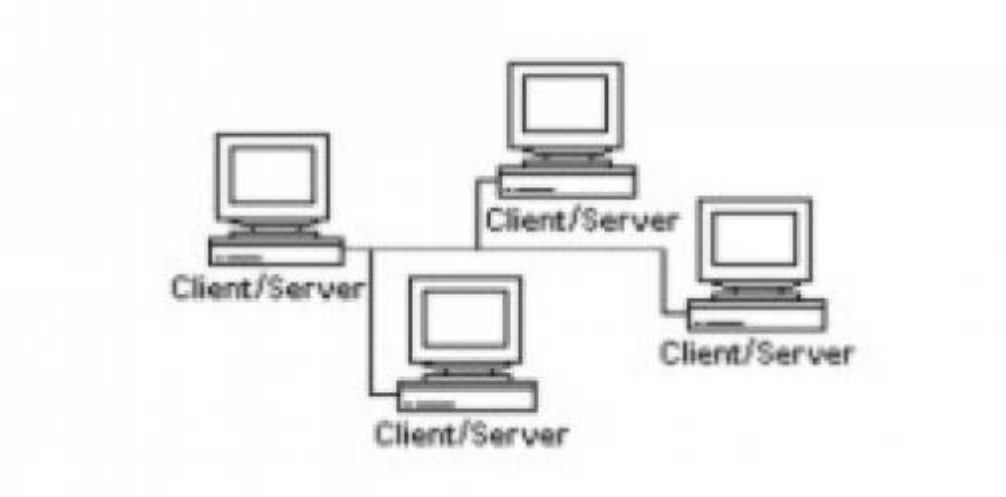


Peer-to-Peer-Netze und Client-Server-Architekturen

Man unterscheidet zwei „Philosophien“:

Peer-to-Peer-Netzwerke

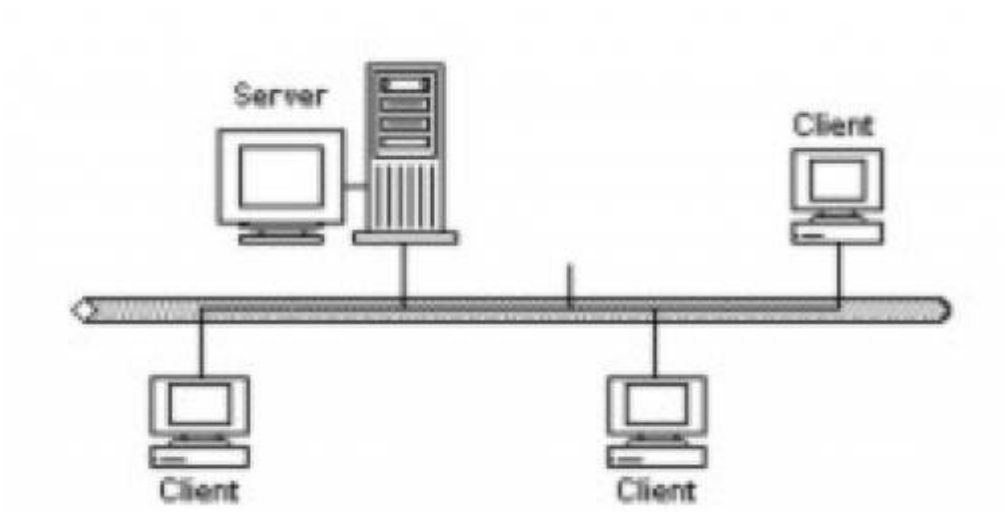
Bei einem solchen Netz können prinzipiell alle in das Netz eingebauten PCs ihre Ressourcen anderen PCs bzw. Anwendern zur Verfügung stellen. Typische Vertreter: NetWare Lite (Novell), Windows for Workgroups (Microsoft), LANtastic (Artisoft), Windows 95/98/ME (Microsoft), Windows NT Workstation (Microsoft), Windows 2000/XP Professional (Microsoft).



Peer-to-Peer-Netze brauchen keinen eigenen Server-Rechner, da jeder PC Server-Funktionen übernehmen kann.

Client/Server-Architekturen

Hier gibt es eine Trennung der Ein-/Ausgabefunktion von der eigentlichen Verarbeitung. Auf der Workstation laufen Programme, die nur für die Ein- und Ausgabe zuständig sind (Frontend-Software), während – unbemerkt vom Anwender – das entsprechende Backend-Programm auf dem Server seine Aufgaben erfüllt (z.B. Speicherung, Suche von Daten). Das grundlegendste Backend-Programm ist das Netzwerk-Betriebssystem.



Die meisten Netzwerke arbeiten so, dass der Server dabei seine Fähigkeiten den anderen Rechnern (Workstations) zur Verfügung stellt. Einen Server, der ausschließlich das Netzwerk und die Datenübertragungen im Netzwerk verwaltet und kontrolliert, bezeichnet man als Dedicated Server. Ist der Server selbst gleichzeitig als Workstation verwendet, so spricht man von einem Non-Dedicated Server.

Server-Betriebssysteme

Für einen Client/Server-Netzwerkbetrieb benötigt man für den Server eigene Betriebssysteme. Netzwerk-Betriebssysteme müssen Multiprocessing unterstützen.

Typische Netzwerk-Betriebssysteme

Novell Open Enterprise Server (früher NetWare)

Dieses Betriebssystem hat alle Kommunikationseigenschaften, die für Netzwerkbetrieb notwendig sind; außerdem werden verschiedenste Einzelplatz-Betriebssysteme unterstützt. Novell NetWare eignet sich daher besonders gut für heterogene Netzwerke. Die Grundidee ist, dass bei Novell NetWare meist dedicated servers eingesetzt werden, auf denen nicht gearbeitet werden kann. Versionen von Novell NetWare 2.11, 3.11 (Nachfolger 3.12), 4.x, 5.x, 6.x, OES1, NetWare for SAA (für Anbindung von Großrechnern wie AS/400)

Microsoft Windows-Serverbetriebssysteme

Typisch für die Microsoft-Serverproduktlinie ist die Möglichkeit, auch Anwendersoftware einsetzen zu können. Damit sind verbesserte Möglichkeiten der Protokollierung und Auswertung gegeben. Die aus den Microsoft Client-Betriebssystemen bekannte Oberfläche ermöglicht rasches Einarbeiten und die Konzentration auf die eigentlichen Systembetreuungsaufgaben. Versionen Windows NT 4.0 Server-Familie, Windows 2000 Server-Familie, Windows Server 2003-Familie

Unix Dialekte

SCO-Unix (SCO = Santa Cruz Operation), Xenix, Sinix, AIX, ULTRIX, Irix, Linux, ...
Auf den Unix-Dialekt Linux soll gesondert verwiesen werden, da es – im Vergleich zu den anderen Dialekten – sehr preisgünstig ist. Linux bietet (mit kleinen Einschränkungen) die volle Unix-Funktionalität!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_00

Last update: **2019/05/28 11:13**

18.1) Netzwerk-Topologien

Die Struktur eines Netzwerks bezeichnet man als Topologie. Wie wichtig die Struktur eines Netzwerks ist, merkt man bei einem Leitungsausfall: ein gutes Netzwerk findet bei einem Leitungsausfall selbstständig einen neuen Pfad zum Empfänger.

Physikalische und logische Topologie

Interessant ist, dass sich die **sichtbare Topologie** (also die physische Verkabelungsstruktur) vom tatsächlichen Datenfluss unterscheiden kann. Deshalb verwendet man für die hardwaremäßige Realisierung den Begriff **physikalische Topologie** während man für den tatsächlichen Datenfluss den Begriff „logische Topologie“ verwendet.

Die wichtigsten Netzwerktopologien sind:

Bus-Topologie



ALLE GERÄTE nutzen DASSELBE KABEL

Bei einem Bussystem sind alle Rechner hintereinander geschaltet und über Abzweige (T-Stücke) an das Netzwerkkabel angeschlossen. Problem: Eine Verbindungsunterbrechung betrifft den ganzen Bus!

Vorteile

- Relativ niedrige Kosten, da geringe Kabelkosten
- Ausfall einer Station führt zu keinem Netzausfall

Nachteile

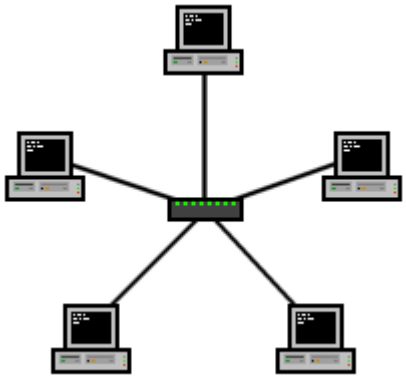
- Alle Daten über ein Kabel
- Nur eine Station kann senden. Alle anderen sind blockiert.
- Eine Störung an einer Stelle (z.B.: Defektes Kabel) führt zu einem Netzausfall (⇒ aufwendige Fehlersuche)
- Unverschlüsselter Netzwerkverkehr kann direkt am Bus (=Kabel) mitgelesen werden

Einsatzgebiet

Früher aufgrund der niedrigen Kosten häufig verwendet, heute spielt die Bus-Topologie keine Rolle mehr und wurde von der Stern-Topologie verdrängt.

W Bus-Topologie - Details

Stern-Topologie



JEDES GERÄT nutzt EIN KABEL.
Damit ist es zu einem Verteiler verbunden. Es existiert eine Punkt-zu-Punkt Verbindung zwischen Verteiler und Gerät. Als Verteiler kann ein HUB oder ein SWITCH dienen.

Vorteile

- Ausfall einer Station oder eines Defekts an einem Kabel führt zu keinem Netzausfall
- Aktive Verteiler (Switch, Hub) dienen gleichzeitig als Signalverstärker
- Bei richtiger Konfiguration können zwei Stationen die volle Bandbreite des Übertragungsmediums für ihre Kommunikation nutzen, ohne andere Stationen dabei zu behindern. Diese physikalische Topologie erlaubt somit sehr hohe Datendurchsatzraten.
- Weitere Stationen oder Verteiler können einfach hinzugefügt werden. Sehr leicht skalierbar.

Nachteile

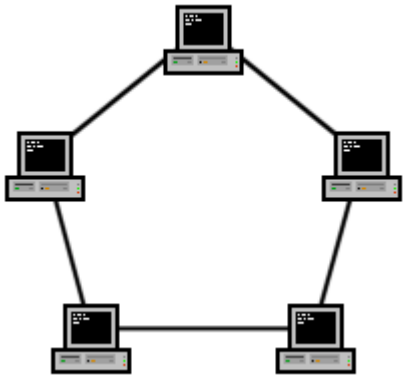
- Große Kabelmengen
- Beim Ausfall des Verteilers ist kein Netzverkehr mehr möglich

Einsatzgebiet

Im praktischen Einsatz bei lokalen Netzwerken findet die Stern-Topologie Verwendung. Häufigste Form der Verkabelung.

W Stern-Topologie - Details

Ring-Topologie



JEDES GERÄT ist mit ZWEI NACHBARN verbunden.

Die Ring-Topologie ist eine geschlossene Form, es gibt keinen Kabelanfang und kein Kabelende.
Es handelt sich jeweils um eine Punkt-zu-Punkt Verbindung zwischen den Rechnern.
Jede Station hat genau einen Vorgänger und einen Nachfolger.
Datenverkehr findet immer nur in eine Richtung statt.

Vorteile

- Vorgänger und Nachfolger sind festgelegt
- Alle Stationen verstärken das Signal
- Alle Stationen haben gleiche Zugriffsmöglichkeit
- Leicht umsetzbar

Nachteile

- Ausfall einer Station oder eines Kabelteils führt zu einen Netzausfall
- Hoher Aufwand bei der Verkabelung (Jede Station braucht 2 Netzwerkkarten)
- Leicht abhörbar
- langsame Datenübertragung bei vielen Stationen

Einsatzgebiet

Physikalische Anwendung gibt es heute keine mehr.
Logische Anwendung findet sie im Token Ring.

W Ring-Topologie - Details

Mischformen

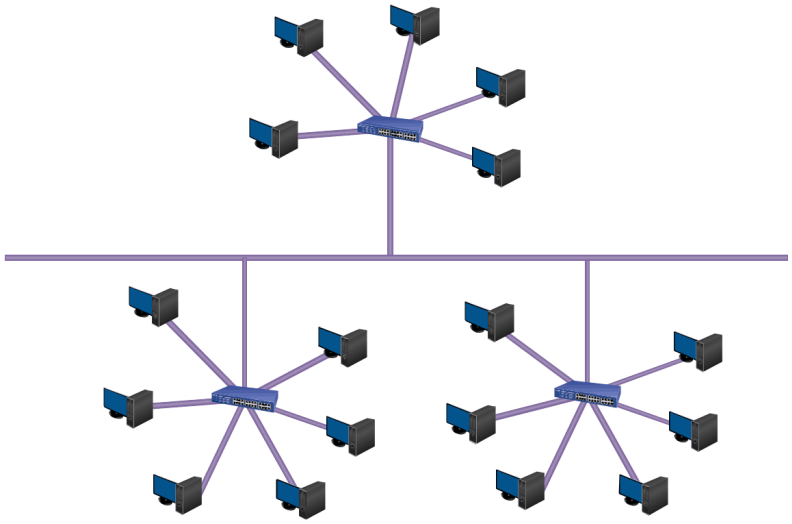
Sind zumeist **Kombinationen aus Bus, Stern und Ring**.

Backbone

Unter einem Backbone („Rückgrat“) wird die physikalische Verbindung zwischen einzelner Teilnetze verstanden. Es wird auch oft als Hintergrundnetz betitelt und verbindet z.B. mehrere Gebäude.

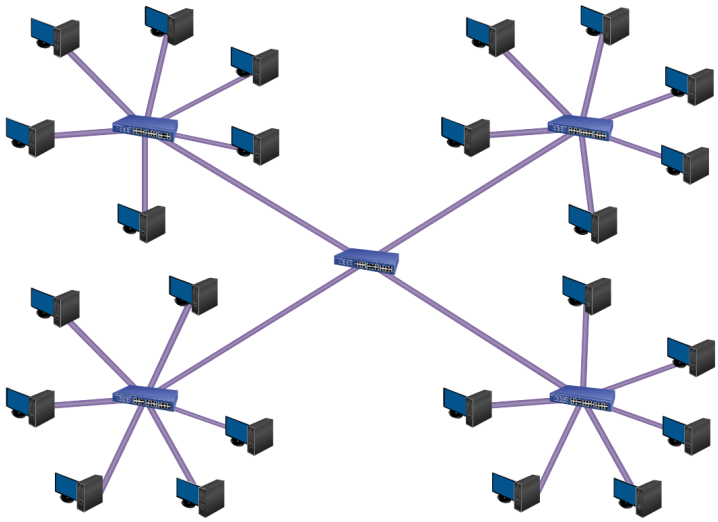
Stern-Bus-Netz

Ein Stern-Bus-Netz entsteht, wenn mehrere Verteiler über einen Bus miteinander verbunden sind. Häufig sind so mehrere Stockwerke miteinander verbunden.



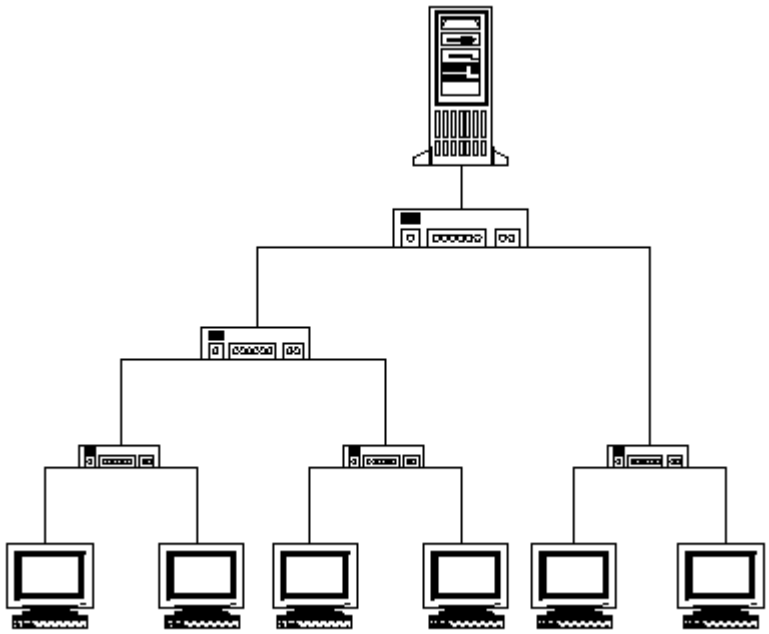
Stern-Stern-Netz

Ein Stern-Stern-Netz entsteht, wenn mehrere Verteiler wiederum über einen Verteiler verbunden sind. Häufigste Anwendung ist wiederum das Verbinden von mehreren Subnetzen (z.B. Netze in verschiedenen Stockwerken).
Fällt der Hauptverteiler aus, so kann zwischen den Stockwerken nicht mehr kommuniziert werden. Jedoch kann man auch die Hauptverteiler redundant auslegen.



Baum

Eine Baum-Topologie wird so aufgebaut, dass, ausgehend von der Wurzel, eine Menge von Verzweigungen zu weiteren Verteilungsstellen existiert.
Es handelt sich somit um eine Erweiterung der Stern-Stern-Topologie auf mehrere Ebenen.



Maschen-Topologie



Vorherrschende Netzstruktur in großflächigen Netzen (z.B. öffentliche Telekommunikationsnetze).
Maschen-Topologie - Details

Zell-Topologie

Die Zell-Topologie kommt hauptsächlich bei drahtlosen Netzen zum Einsatz. Eine Zelle ist der Bereich um eine Basisstation (z.B. Wireless Access Point), in dem eine Kommunikation zwischen den Endgeräten und der Basisstation möglich ist.
Zell-Topologie - Details

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_01

Last update: **2019/05/16 18:34**

18.2) Übertragungsmedien

Die Übertragungsmedien sind die Straßen der Daten. Der Aufbau dieser Straßen muss sehr gut geplant werden, um alle aktuellen Anforderungen bzw. eventuelle zukünftige Anforderungen ohne großartige Veränderungen zu erfüllen.

Als **Maßeinheit für die Übertragungsgeschwindigkeit** werden die Werte in **bit/s, b/s bps, -> also Bit pro Sekunde** angegeben. Achtung: Nicht zu verwechseln mit **Byte/s -> Byte pro Sekunde**!!

`C= D/t` `(bits)/(s)`

1) Rechenbeispiel:

Es werden 100MB in 10s übertragen. Wie hoch ist die Übertragungsgeschwindigkeit?

- D=100MByte
- t=10s

Rechenschritt	**Berechnung**
D umwandeln in bits	`100*1024*1024*8 = 838860800` `bits`
C berechnen	`C=D/t=(838 860 800)/10=83886080` `(bit)/(s)`
C umwandeln in Mbit/s	`(83886080)/(1024)/(1024)=80` `(Mbit)/(s)`

2) Rechenbeispiel:

Max hat eine Datentransferrate von 10Mbit/s Download und 2Mbit/s Upload.

a) Wie lange braucht er, um 10MB runterzuladen?

- D=10MB
- C=10Mbit/s

Rechenschritt	**Berechnung**
D umwandeln in bits	`10*1024*1024*8 = 83886080` `bits`
C umwandeln in bit/s	`10*1024*1024=10485760` `(bit)/(s)`
Formel umformeln	`t=(D)/(C)` `(bits)/((bit)/s)`
In Formel einsetzen	`t=83886080/10485760=8` `s`

b) Wie lange braucht er, um 10MB hochzuladen?

Rechenschritt	**Berechnung**
D umwandeln in bits	`10*1024*1024*8 = 83886080` `bits`
C umwandeln in bit/s	`2*1024*1024=2097152` `(bit)/(s)`
Formel umformeln	`t=(D)/(C)` `(bits)/((bit)/s)`
In Formel einsetzen	`t=83886080/2097152=40` `s`

Leitergebundene Übertragung

Bei einer leitergebundenen Übertragung werden Medien in Form von Kabeln benötigt (Metallische Leiter, Glasfaser).
Ein Kabel besteht zumindest aus einer Ader (=Faser).
Mehrere Adern sind durch entsprechende Isolationssschichten getrennt.
Alle Adern wiederum werden von einem Mantel als Schutz umgeben.
Die Übertragung selbst erfolgt durch elektromagnetische Schwingungen.

Koaxialkabel

Das früher verwendete Koaxialkabel ist in modernen Netzen praktisch vollständig von Twisted Pair-Kabeln (TP) und Lichtwellenleiter (LWL) abgelöst worden.



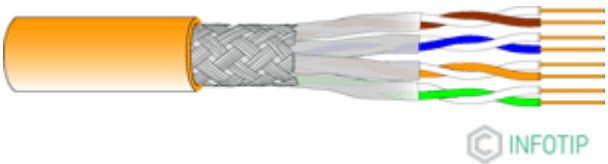
Es besteht aus

- einem Innenleiter (Kupfer, Stahlkupfer)
- einer Isolation (Dielektrikum)
- einer Abschirmung (Metallgeflecht schützt vor magnetischen Störungen –> Rauschen & Übersprechen)
- einem Mantel

Es waren bis zu 10Mbps möglich:

- Thicknet (10Base5)
- Thinnet (10Base2) - Heute noch bei Satellitenempfang im Einsatz

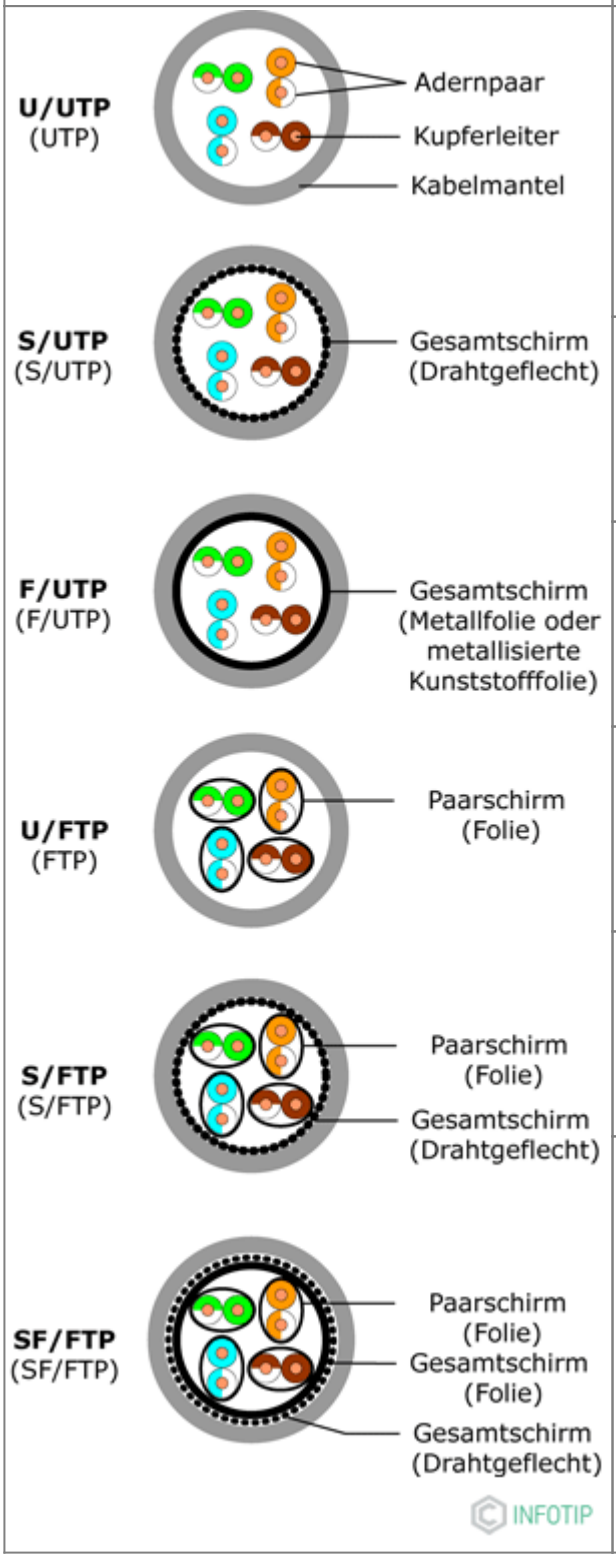
Twisted-Pair Kabel



Twisted Pair ist ein vier-, acht- oder mehr-adriges Kupferkabel, bei dem jeweils zwei Adern miteinander verdrillt sind. Durch die Verdrillung kompensieren sich Leitungskapazität und -induktivität. Dadurch steigt die Übertragungsbandbreite und die mögliche Übertragungsreichweite wird praktisch nur durch die Dämpfung des Wirkwiderstandes begrenzt. Die Verwendung von symmetrischen Signalen (Differentialspannungen) erhöht die Festigkeit gegen elektromagnetische Störstrahlung.

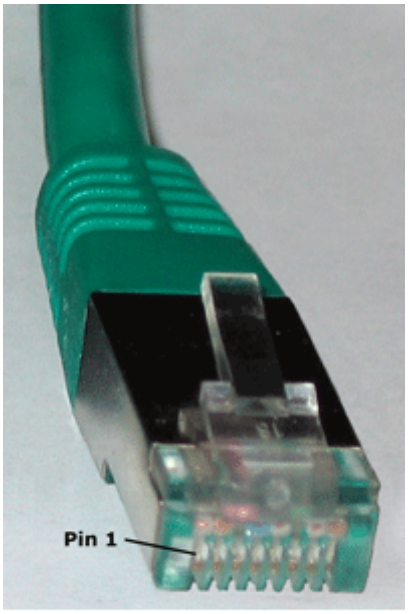


Twisted Pair-Kabel gibt es in zahlreichen Varianten. Twisted Pair-Verbindungen werden außer in der Kommunikationstechnik (Netzwerkkabel, Telefonkabel) auch bei HDMI-, DVI- und LVDS-(in LCD/Plasma-TV zwischen Signalprozessor und Display) Verbindungen eingesetzt. Die Anzahl der Leiterpaare im Kabel hängt dabei von der benötigten Datenübertragungsrate ab. In Netzwerken wird für jede Übertragungsrichtung (senden, empfangen) wird jeweils ein Adernpaar (bei 100BaseT4 und 1000BaseT jeweils zwei) genutzt. Die Übertragungsreichweite ist abhängig vom Aufbau des Kabels, von der Dämpfung (=Länge) des Kabels und von den externen Störeinflüssen. Twisted Pair-Kabel für Netzwerke gibt es in zahlreichen Varianten:

Bild	Benennung	Beschreibung
	**U/UTP-Kabel**	**Unshielded/Unshielded Twisted Pair** sind nicht abgeschirmte verdrillte Leitungen und gehörten früher typischerweise der CAT3 an. UTP-Kabel sollten im industriellen Bereich oder in der Datentechnik mit hohen Datenraten nicht verwendet werden.
	S/UTP-Kabel	**Screened/Unshielded Twisted Pair** haben einen Gesamtschirm aus einem Kupfergeflecht zur Reduktion der äußeren Störeinflüsse
	F/UTP-Kabel	**Foilshielded/Unshielded Twisted Pair** besitzen zur Abschirmung einen Gesamtschirm, zumeist aus einer alukaschierten Kunststofffolie
	U/FTP-Kabel	**Unshielded/Foilshielded Twisted Pair** auch genannt **CAT5** oder **CAT5e**. Die Leitungsadern sind paarweise mit Folie abgeschirmt
	S/FTP-Kabel	**Screened/ Foilshielded Twisted Pair** auch genannt **CAT6** sollten in Bereichen mit hoher Störstrahlung (z.B. Büros mit mehreren PCs) eingesetzt werden.
	SF/FTP-Kabel	**Screened Foilshielded/Foilshielded Twisted Pair** auch genannt **CAT6e** oder **CAT7** besitzen eine Abschirmung für jedes Kabelpaar sowie eine doppelte Gesamtschirmung. Hierdurch kann eine optimale Störleistungsunterdrückung erreicht werden. Auch das Übersprechen zwischen den einzelnen Adernpaaren wird so wirksam unterdrückt

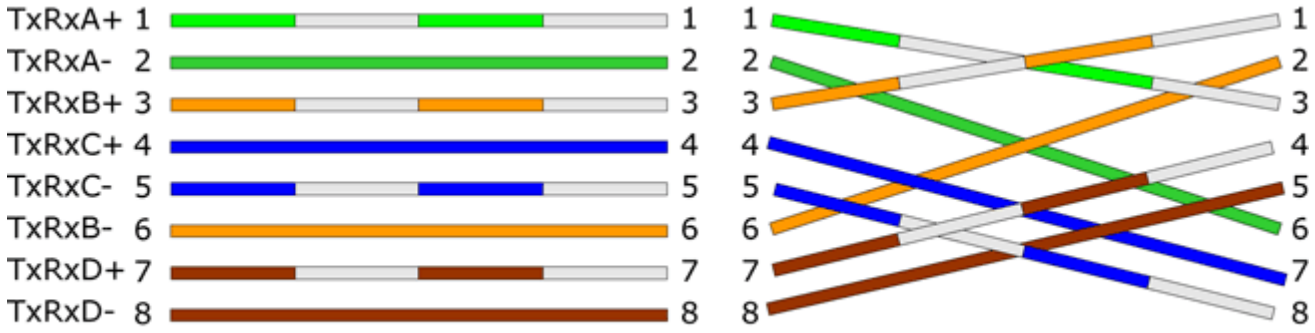
Die Preisunterschiede zwischen CAT-5e- Kabeln und CAT-7-Kabeln ist so gering, dass es sich bei Neuinstallation auf jeden Fall empfiehlt, CAT-7-Kabel einzusetzen. Dieses ist als einziges Kupfermedium in der Lage mit dem kommenden 10GBit-LAN verwendet zu werden.

Verbinder - RJ45



Der typische Standardverbinder für die Twisted-Pair-Verkabelung eines kupfergebundenen Ethernet-Netzwerkes ist der **8polige Western-Modularstecker RJ-45** (8P8C), auch RJ-48 oder RJ-49 genannt. RJ-45 Steckverbindungen können auf zwei Arten belegt sein, wobei die Belegung nach T568B am weitesten verbreitet zu sein scheint:

Belegung nach EIA/T-T568A		**Belegung nach EIA/T-T568B**	
Pin	**Farbe**	**Pin**	**Farbe**
1	weiß-grün	1	weiß-orange
2	grün	2	orange
3	weiß-orange	3	weiß-grün
4	blau	4	blau
5	weiß-blau	5	weiß-blau
6	orange	6	grün
7	weiß-braun	7	weiß-braun
8	braun	8	braun



Bei 1:1-Verbindungen sind beide Beschaltungen sind elektrisch zueinander kompatibel. Nur bei Erweiterungen von fest verdrahteten Netzen ist festzustellen, welche Belegung bereits vorgegeben ist. Normale Verbindungskabel („ Patchkabel“) mit RJ-45-Steckern sind 1:1 verschaltet, d.h. Pin 1 des einen Steckers geht auf den Pin 1 des anderen Steckers usw. Nur in besonderen Fällen, wenn z.B.

zwei Netzwerkkarten direkt miteinander verbunden werden sollen oder wenn Netzwerkkomponenten (z.B. Hubs älterer Bauart) über keinen dedizierten Uplink-Port verfügen, kann der Einsatz von Crossover-Kabeln notwendig werden.



LichtWellenLeiter (LWL)

Sind mit der Netzwerkverkabelung weite Strecken zu überwinden, z.B. zwischen einzelnen Gebäuden auf einem Fabrikgelände („Campusbereich“), sind sehr hohe Datenübertragungsraten (z.Zt. bis zu 170Gb/s) gefordert oder wenn sich die Datenübertragung per Kupferkabel aus technischen Gründen (z.B. bei extremer Störstrahlung) oder aus Gründen der Sicherheit verbietet, werden Lichtwellenleiter (LWL, Glasfasern) als Übertragungsmedium eingesetzt. Die Lieferprogramme der Hersteller erlauben mittlerweile die Übertragungsstrecken bis zum Einzelplatz komplett auf der Basis von LWL auszuführen.



Aufbau und Prinzip

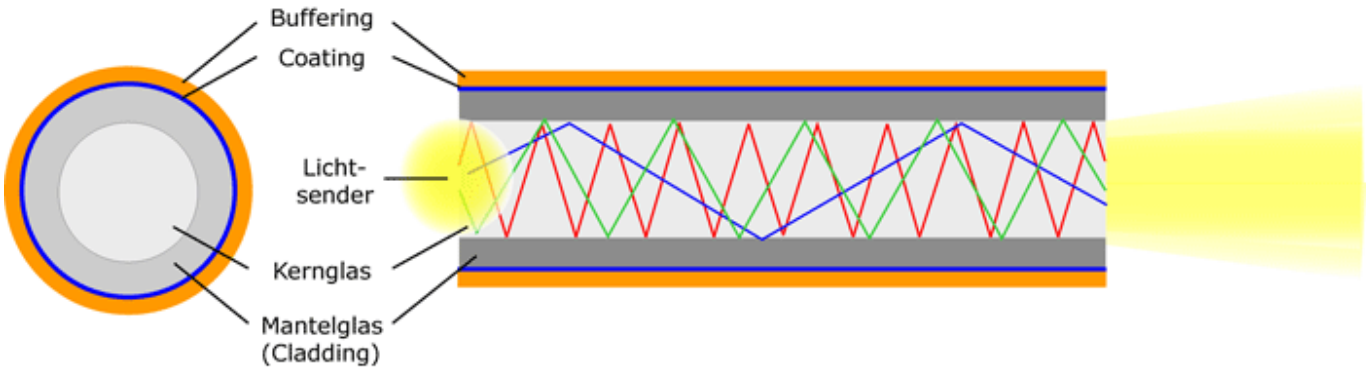
In einem LWL werden die Informationen nicht, wie in einem Kupferkabel, elektrisch übertragen, sondern mit **Licht**.
Der eigentliche LWL ist eine Faser aus Glas oder Kunststoff. Jede Faser besteht aus zwei Schichten. Der konzentrische Kern besteht aus einem optischen Material mit einem hohen Brechungsindex, das Mantelglas („ Cladding“) aus einem Material mit niedrigem. Licht, das in einem bestimmten Winkelbereich auf den Übergang von Kern zum Mantel trifft wird dort vollständig reflektiert. Über

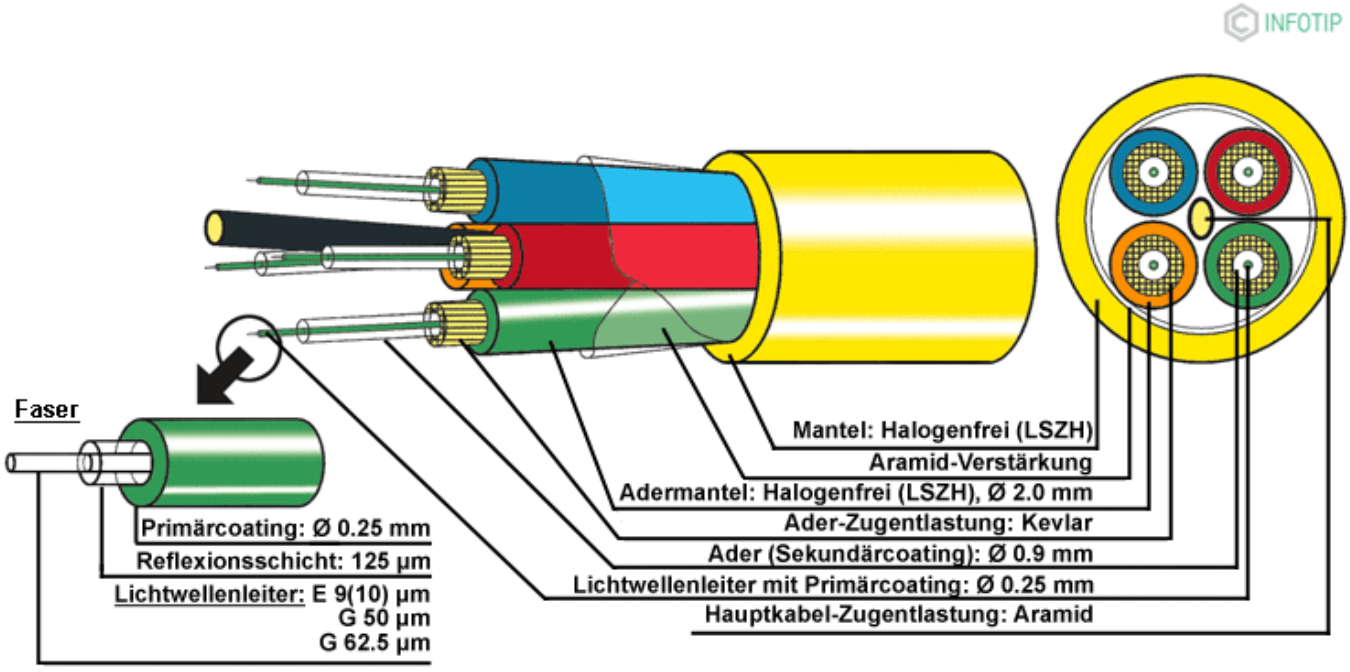
solche fortlaufenden Totalreflexionen pflanzt sich das Licht durch den LWL bis zum Ende der Faser fort.
Je steiler der Einfallswinkel des Lichts bei der Einspeisung in den LWL ist, desto häufiger wird die Lichtwelle reflektiert. Mit jeder Reflektion der Lichtwelle wird der Weg, des sogenannten Modes, länger.

Licht, das wenig häufig reflektiert wird, hat einen kürzeren Weg und durchläuft die Faser schneller. Es ist Licht niedrigen Modes.

Licht, das sehr häufig reflektiert wird, hat eine niedrige Ausbreitungsgeschwindigkeit in der Faser. Er ist Licht hohen Modes.

Erzeugt die Lichtquelle des Senders ein nicht-kohärentes Licht, tritt das Licht mit einer Vielzahl unterschiedlicher Winkel in die Faser ein. Dadurch entstehen natürlich durch die unterschiedlichen Moden Laufzeitunterschiede zwischen den Signalanteilen. Ein Eingangsimpuls mit steilen Flanken wird dadurch verschliffen und in seiner Breite gedehnt. Je länger ein Kabel ist, desto höher wird auch diese sog. Dispersion (Einheit: ns/km). Die Dispersion beeinflusst direkt die Übertragungsbandbreite der Glasfaserverbindung.

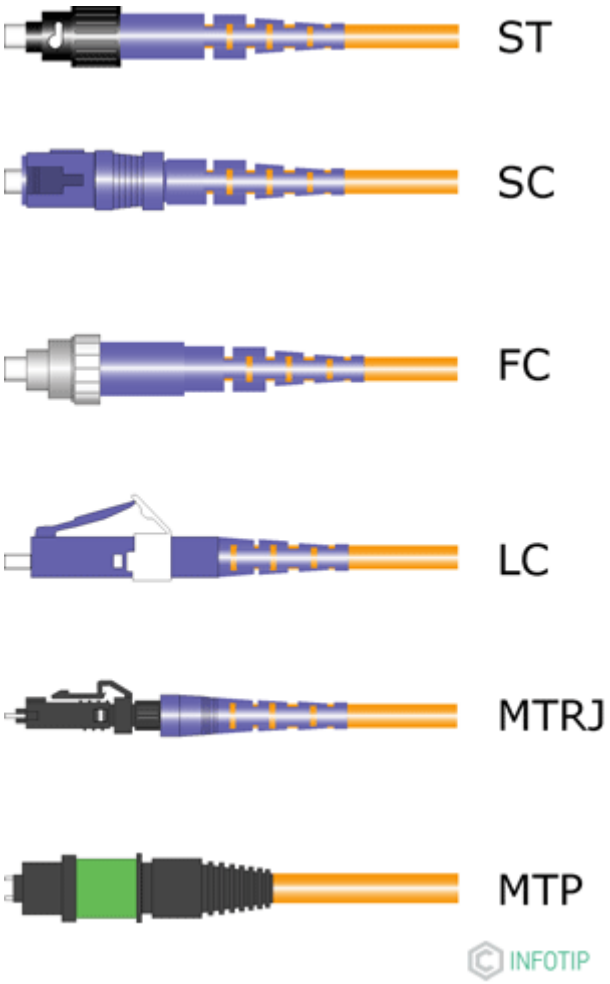




Da die Fasern sehr dünn und empfindlich sind, werden sie zum mechanischen Schutz mit einer Kunststoffbeschichtung („ Coating“) und einem Schutzüberzug versehen. In einem LWL-Kabel können mehrere Fasern, sogar in mehreren Bündeln, zusammen gefasst sein.

Verbinder

Die Hersteller von Netzwerkzubehör bieten konfektionierte Verbindungs- und Patchkabel mit einer Vielzahl von verschiedenen Steckerformen an. Meist sind die Kabel paarweise angelegt um beide Datenflussrichtungen (TX und RX) gleichzeitig herstellen zu können.



Vorteile

- hohe Reichweite
- hohe Übertragungsbandbreite
- Potentialfrei, daher auch für explosionsgefährdete Bereiche geeignet
- hohe Störfestigkeit, LWL können sogar zu Energieversorgungskabeln parallel verlegt werden
- hohe Abhörsicherheit

Nachteile

- Material für die Verkabelung ist teuer
- teure Verbindungstechnik
- Die Montagekosten sind wegen des höheren technischen Aufwandes höher
- komplexe und teure Messtechnik
- zusätzliche Kosten für Medienkonverter auf Kupfer-Ethernet

Leiterungebundene Übertragung

Als leiterungebundene Übertragung bezeichnet man eine Übertragung per

- Funk
- Ultraschall
- Infrarot
- Laser
- Licht

Drahtlose Übertragung (WLAN)

Die Übertragung von Informationen ohne Kabel ist mittlerweile in vielen Lebensbereichen als praktische Alternative eingezogen. Von der Fernbedienung eines Fernsehers über drahtlose Lautsprecher bis zum Smartphone gibt es viele Beispiele für die Umsetzung dieser Technik. Dabei werden Funksignale in frei verfügbaren Frequenzbändern anstelle von Kabeln für die Datenübertragung verwendet.

Vorteile

- Es sind keine baulichen Maßnahmen innerhalb eines Gebäudes nötig.
- Die baulichen Maßnahmen zwischen verschiedenen Gebäuden sind geringer als bei einer Verkabelung.
- Höhere Mobilität, da theoretisch jeder Punkt eines Firmengeländes drahtlos erreichbar ist.

Nachteile

- Oft geringere Datenübertragungsraten als bei Kabeln, die abhängig von Hindernissen sind.
- Anfällig für Störeinflüsse und Abhören durch Unbefugte.
- Probleme mit Ausleuchtung und Reflexionen.
- Bei vielen gleichzeitigen Nutzern an einem WLAN-Zugang bricht die Übertragungsrate ein (Shared Media).

Sicherheit als kritischer Bereich

Gerade im Bereich Sicherheit gibt es bei WLANs einige Punkte zu beachten, die sich auch in einem etwas größeren Konfigurationsaufwand äußern. Die sogenannte **SSID (Service Set Identifier)** kann eine eindeutige Identifikation (Firmenname etc.) enthalten, damit bei Problemen eine Kontaktaufnahme mit dem Betreiber möglich ist. Ein Verbergen bringt nicht viel, da die SSID in jedem Paket mitgeschickt wird und es Programme gibt, die auch verborgene SSIDs auslesen können. Eine versehentliche Verbindung durch Unbefugte ist bei verschlüsselten Zugängen nicht zu erwarten. Es gibt auch Empfehlungen, als SSID eine zufällige Zeichenfolge einzugeben, damit die SSID keine Rückschlüsse auf den Betreiber zulässt.

Letztendlich sollte die Übertragung im WLAN nur **verschlüsselt** erfolgen, wobei die Verschlüsselung mit **WEP (Wired Equivalent Privacy)** unsicher ist. Besser ist der Einsatz von **WPA (Wi-Fi**

Protected Access) bzw. der Nachfolgetechnologie **WPA2**, da hier deutlich stärkere Verschlüsselungsmechanismen mit **AES (Advanced Encryption Standard)** verwendet werden. Zusammen mit **TKIP (Temporal Key Integrity Protocol)** sind allerdings nur max. 54 Mbit/s möglich! Höhere Raten erreicht z. B. WPA2 zusammen mit CCMP (Counter-Mode/CBC-Mac Protocol)

Grundlegende Beschreibung

Kommunikation über WLAN erfolgt entweder als Punkt-zu-Punkt- oder als Mehrpunkt-Kommunikation. Die erste Variante dient z. B. der Überwindung größerer Distanzen durch den Einsatz zweier Richtantennen.

Bei der Mehrpunkt-Kommunikation werden ein oder mehrere sogenannte Access Points eingesetzt, die im Prinzip jeweils wie Zentralen (Verteiler) fungieren und die Datenströme mehrerer Clients koordinieren.

Diese Access Points können bei größeren Installationen über Kabel und geeignete Managed Switches eine Verbindung zu einem sogenannten RADIUS (Remote Authentication Dial-In User Service)-Server erhalten, um zwischen berechtigten und nicht berechtigten Sendern zu unterscheiden (Authentifizierung).

Eine Unterscheidung über die MAC-Adresse sollte nur den zum Zugang berechtigten Geräten vorbehalten bleiben, die sich nicht per RADIUS authentifizieren können. Dabei wird in einer Zugangsliste (Access Control Table) vom Managed Switch die MAC-Adresse eingetragen. Auf einem Access-Point bringt dies keine höhere Sicherheit, da per Funk eine MAC-Adresse unverschlüsselt verschickt wird und somit gefälscht werden kann. Die kleinste Einheit ist eine sogenannte Funkzelle, womit der Bereich gemeint ist, der von einem Sender (=Access Point) abgedeckt werden kann. Er umfasst ca. 30m im Gebäude und bis zu 300m im Freien. Mit speziellen Antennen können auch mehrere Kilometer überbrückt werden.

ISM-Frequenzbänder

In den meisten Fällen wird von den Herstellern ein sogenanntes **ISM-Band (Industrial, Scientific and Medical)** verwendet. Manchmal finden Sie auch die Abkürzung ISMO, wobei der letzte Buchstabe für den Begriff „Office“ (Büro) steht. Der Einsatz dieser Frequenzbänder bietet zwei Vorteile. Sie sind

- **gebührenfrei**
- **genehmigungsfrei**

Hierin liegt aber auch gleichzeitig der Nachteil. Sie werden von sehr vielen Herstellern für die unterschiedlichsten Zwecke genutzt, wie z. B. drahtlose Lautsprecher, elektronische Türöffnung bei Autos oder Garagen, und so ist die Gefahr, dass sich Geräte gegenseitig stören, relativ hoch.

Die für WLAN wichtigsten ISM-Bänder sind

- das **2,4-GHz-Band** (2,3995 bis 2,4845 GHz) mit max. **13 überlappenden Kanälen** von 20 MHz Bandbreite; auch 40 MHz Bandbreite ist möglich, dann aber mit weit weniger nutzbaren Kanälen. Es sind Geschwindigkeiten bis zu 300MBit/s möglich.
- das **5-GHz-Band** (5,150 bis 5,350 GHz für Kanalnummer 36–64 und 5,470 bis 5,725 GHz für Kanalnummer 100–140) mit Kanälen von 20, 40, 80 oder 160 MHz Bandbreite, wobei **max. 19**

Kanäle bei 20 MHz Bandbreite nicht überlappend nutzbar sind. Beim Funken mit 40 MHz Bandbreite sind 2 dieser Kanäle gebündelt erforderlich, mit 80 MHz 4 Kanäle usw. Die **Reichweite ist geringer** als im 2,4 GHz-Band. Es sind Geschwindigkeiten bis zu 600MBit/s möglich.

- zukünftig das 60-GHz-Band (57 bis 66 GHz) mit vier 2000 MHz breiten Funkkanälen für kurze Distanzen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_02

Last update: **2019/05/16 18:34**

18.3) Ethernet

Erklärung

Ethernet ist eine Technik, die **Software (Protokolle usw.) und Hardware (Kabel, Verteiler, Netzwerkkarten usw.) für kabelgebundene Datennetze spezifiziert**, welche ursprünglich für lokale Datennetze (LANs) gedacht war und daher auch als LAN-Technik bezeichnet wird. Sie ermöglicht den Datenaustausch in Form von Datenframes zwischen den in einem lokalen Netz (LAN) angeschlossenen Geräten (Computer, Drucker und dergleichen). Derzeit sind Übertragungsraten von 1, 10, 100 Megabit/s (Fast Ethernet), 1000 Megabit/s (Gigabit-Ethernet), 2,5, 5, 10, 40, 50, 100, 200 und 400 Gigabit/s spezifiziert. In seiner ursprünglichen Form erstreckt sich das LAN dabei nur über ein Gebäude; Ethernet-Varianten über Glasfaser haben eine Reichweite von bis zu 70 km.

Die Ethernet-Protokolle umfassen Festlegungen für Kabeltypen und Stecker sowie für Übertragungsformen (Signale auf der Bitübertragungsschicht, Paketformate). Im OSI-Modell ist mit Ethernet sowohl die **physische Schicht (OSI Layer 1)** als auch die **Data-Link-Schicht (OSI Layer 2)** festgelegt.

Ethernet basiert auf der Idee, dass die Teilnehmer eines LANs Nachrichten durch Hochfrequenz übertragen, allerdings **nur innerhalb eines gemeinsamen Leitungsnetzes**. Jede Netzwerkschnittstelle hat **einen global eindeutigen 48-Bit-Schlüssel**, der als **MAC-Adresse** (=Media-Access-Control-Adress) bezeichnet wird. Das stellt sicher, dass alle Systeme in einem Ethernet unterschiedliche Adressen haben.

Ethernet Frame

Ethernet II Frame Structure and Field Size					
8 Bytes	6 Bytes	6 Bytes	2 Bytes	46 - 1500 Bytes	4 Bytes
Preamble	Destination Address	Source Address	Type	Data	Frame Check Sequence

CSMA/CD

CSMA/CD (Carrier Sense Multiple Access with Collision Detection) ist das Zugriffsverfahren des Ethernets. Die Grundidee dabei ist, dass jede Station zu senden beginnen kann, wann sie will. Die

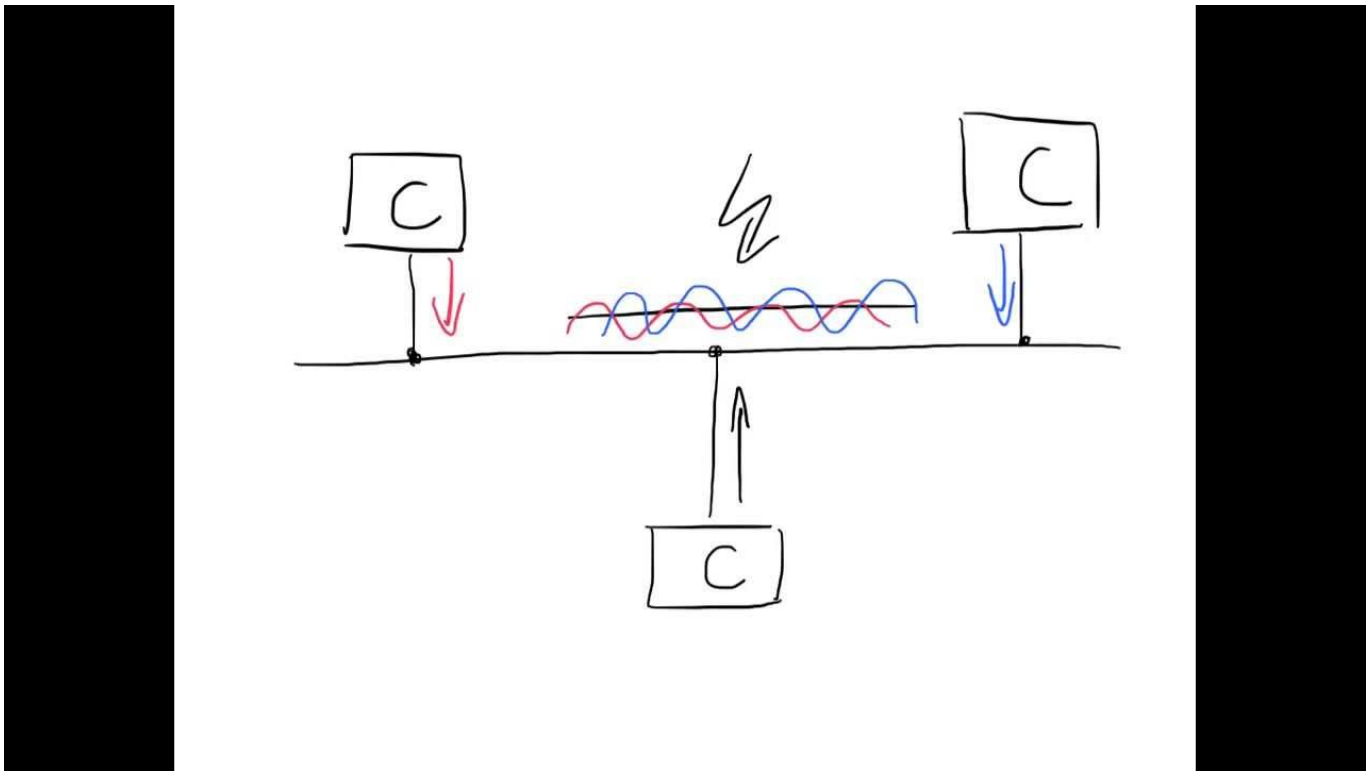
einzelnen Stationen haben jederzeit und konkurrierend Zugang (Multiple Access) zum gemeinsamen Übertragungsmedium. Das grundlegende Motto könnte damit lauten:

Jeder darf, wann er will

Eingesetzt wird dieses Verfahren bei logischen Bus-Topologien. Dabei ist egal, ob physikalisch eine Bus- oder eine Stern-Topologie vorliegt.

Vorgehen

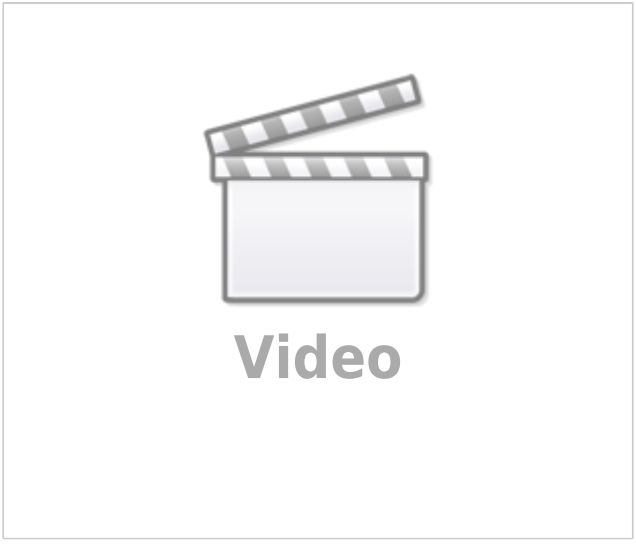
Durch Regelungen wird versucht, das Risiko zu minimieren, dass zwei Stationen ungünstigerweise gleichzeitig zu senden beginnen und somit die Signale auf dem Übertragungsweg zerstört/gestört werden **(=Kollision)**.



1) Leitung prüfen (Carrier Sense)

Der erste Teil der Abkürzung steht für Kollisionsverhinderung. Dabei wird vor einer geplanten Sendung das Übertragungsmedium abgehört, ob dieses frei ist. Wenn das Medium frei ist, wird gesendet. **–> LISTEN BEFORE TALKING**

2) Erkennen von Kollisionen (Collision Detection) Kommt es trotzdem zu einer Kollision, weil zwei Stationen gleichzeitig zu senden beginnen (**Multiple Access**). dann muss diese Kollision erkannt (**Collision Detection**) und reagiert werden. Dabei müssen alle Stationen immer am Medium horchen, ob eine Kollision auftritt. Ist dies der Fall, so sendet die erste Station, die eine Kollision erkennt, ein sogenanntes JAM-Signal aus. Jede Station, die das JAM-Signal registriert, stoppt unmittelbar das Senden von Daten. Nach einer zufälligen Zeitspanne, wird das Medium wieder überprüft und anschließend wieder begonnen zu senden.



Frage: Wie kann bei einer physikalischen Stern-Topologie eine Kollision auftreten?

Vorteile

- Jeder kann zu jeder Zeit senden

Nachteile

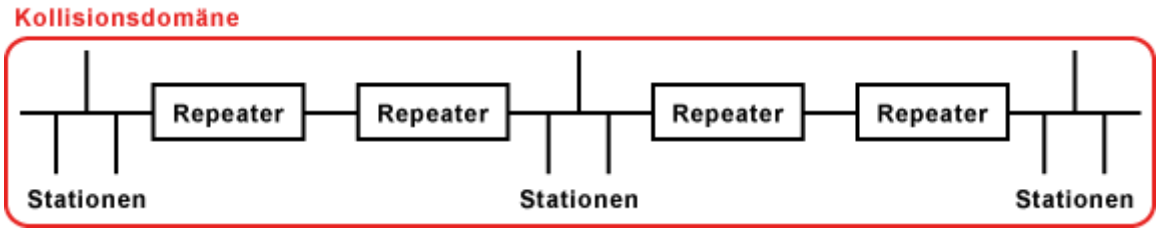
- Je mehr Stationen in einer Kollisionsdomäne, desto häufiger treten Kollisionen auf
- Der Zeitpunkt einer Sendung ist zufällig
- Das Verfahren ist ungeeignet für zeitkritischen Anwendungen

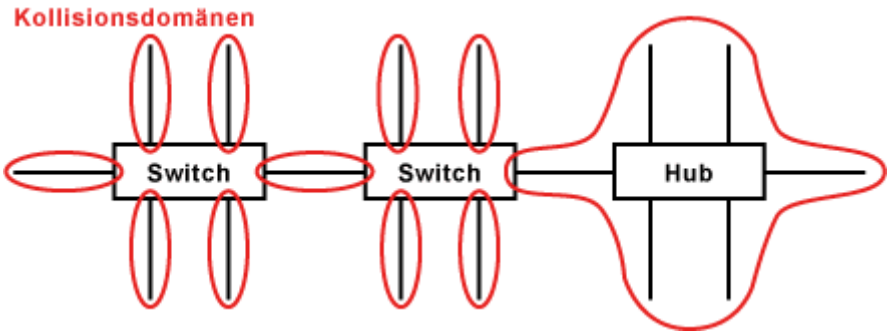
Kollisionsdomäne

Mit dem Begriff Kollisionsdomäne wird in einem Computernetz ein Teilbereich aus Teilnehmerstationen in derselben OSI-Modell-Schicht 1 bezeichnet. Eine Kollisionsdomäne umfasst alle Netzwerkgeräte, die um den Zugriff auf ein gemeinsames Übertragungsmedium konkurrieren. Das Übertragungsmedium ist daher eine zwischen allen Netzstationen geteilte Ressource. Grundlegende Vorstellung dabei ist, dass alle Netzwerkteilnehmer die Chance zur gleichberechtigten Nutzung des Netzwerkes besitzen.

Bei einem gemeinsamen Medium kann zu einer bestimmten Zeit nur jeweils eine Station Informationen übertragen, die an alle anderen Stationen übertragen bzw. von diesen empfangen wird. Fangen in einem derartigen gemeinsamen Schicht-1-Segment zwei Stationen gleichzeitig an zu senden, kommt es zu Kollisionen.

Beispiele für Kollisionsdomänen:





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_03

Last update: **2019/05/16 18:35**

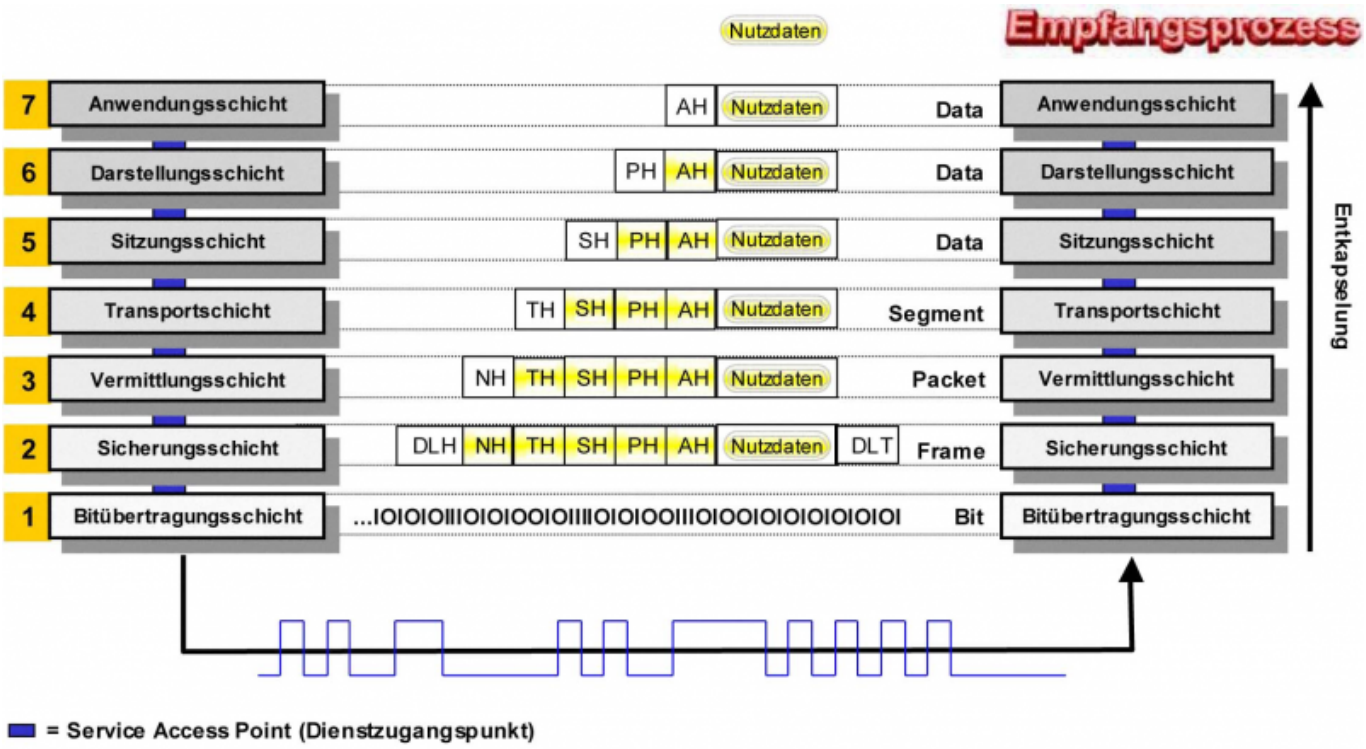
18.4) OSI - Schichtenmodell

Das OSI-7-Schichtenmodell ist ein Referenzmodell für herstellerunabhängige Kommunikationssysteme bzw. eine Design-Grundlage für Kommunikationsprotokolle und Computernetze. OSI steht für Open System Interconnection (Offenes System für Kommunikationsverbindungen) und wurde von der ISO (International Organization for Standardization), das ist die Internationale Organisation für Normung, als Grundlage für die Bildung von offenen Kommunikationsstandards entworfen. Bei allen ISO-Standards handelt es sich um Handlungsempfehlungen. Die Einhaltung einer ISO-Norm ist freiwillig. In der Regel wird die Einhaltung der ISO-Standards von verschiedenen Seiten, zum Beispiel Kooperationspartnern, Herstellern und Kunden, gefordert.



Das Modell

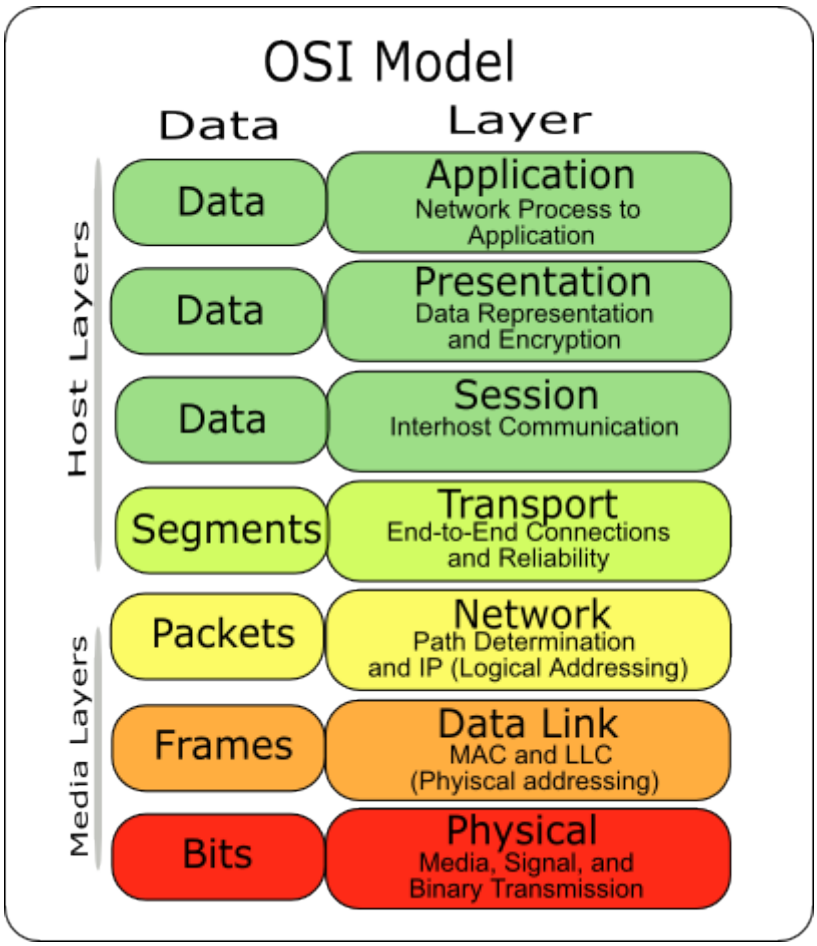
(Offenes System für Kommunikationsverbindungen) und wurde von der ISO (International Organization for Standardization), das ist die Internationale Organisation für Normung, als Grundlage für die Bildung von offenen Kommunikationsstandards entworfen. Bei allen ISO-Standards handelt es sich um Handlungsempfehlungen. Die Einhaltung einer ISO-Norm ist freiwillig. In der Regel wird die Einhaltung der ISO-Standards von verschiedenen Seiten, zum Beispiel Kooperationspartnern, Herstellern und Kunden, gefordert.



Die Schichten im Detail

Application Layer (Anwendungsschicht)	**Benutzerschnittstelle, Dienste, Anwendungen und Netzmanagement**
Schicht 7	Die Anwendungsschicht stellt Funktionen für die Anwendungen zur Verfügung. Diese Schicht stellt die Verbindung zu den unteren Schichten her. Auf dieser Ebene findet auch die Dateneingabe und -ausgabe statt.
Presentation Layer (Darstellungsschicht)	**Übersetzung, Verschlüsselung , Kompression in Standardformate**
Schicht 6	Die Darstellungsschicht setzt die Daten der Anwendungsebene in ein Zwischenformat um. Diese Schicht ist auch für Sicherheitsfragen zuständig. Durch sie werden Dienste zur Verschlüsselung von Daten bereitgestellt und gegebenenfalls Daten komprimiert.
Session Layer (Sitzungsschicht)	**Erstellung einer Verbindung, Freigabe von Verbindungen, Dialogsteuerung**
Schicht 5	Diese Schicht ermöglicht zwei Anwendungen auf verschiedenen Computern, eine gemeinsame Sitzung aufzubauen, damit zu arbeiten und sie zu beenden. Sie übernimmt ebenfalls die Dialogsteuerung zwischen den beiden Computern einer Sitzung und regelt, welcher der beiden wann und wie lange Daten überträgt.
Transport Layer (Transportschicht)	**Logische Ende-zu-Ende-Verbindungen (Transportkontrolle, Paketbildung)**
Schicht 4	Die Transportschicht stellt die zuverlässige Auslieferung der Nachrichten sicher und erkennt sowie behebt allfällige Fehler. Sie ordnet bei Bedarf auch die Nachrichten in Paketen neu, indem sie lange Nachrichten zur Datenübertragung in kleinere Pakete aufteilt. Am Ende des Weges stellt sie die kleinen Pakete wieder zur ursprünglichen Nachricht zusammen. Die empfangene Transportebene sendet auch eine Empfangs bestätigung.
Network Layer (Vermittlungsschicht)	**Routing (Internet), Datenflusskontrolle, Adressierung**

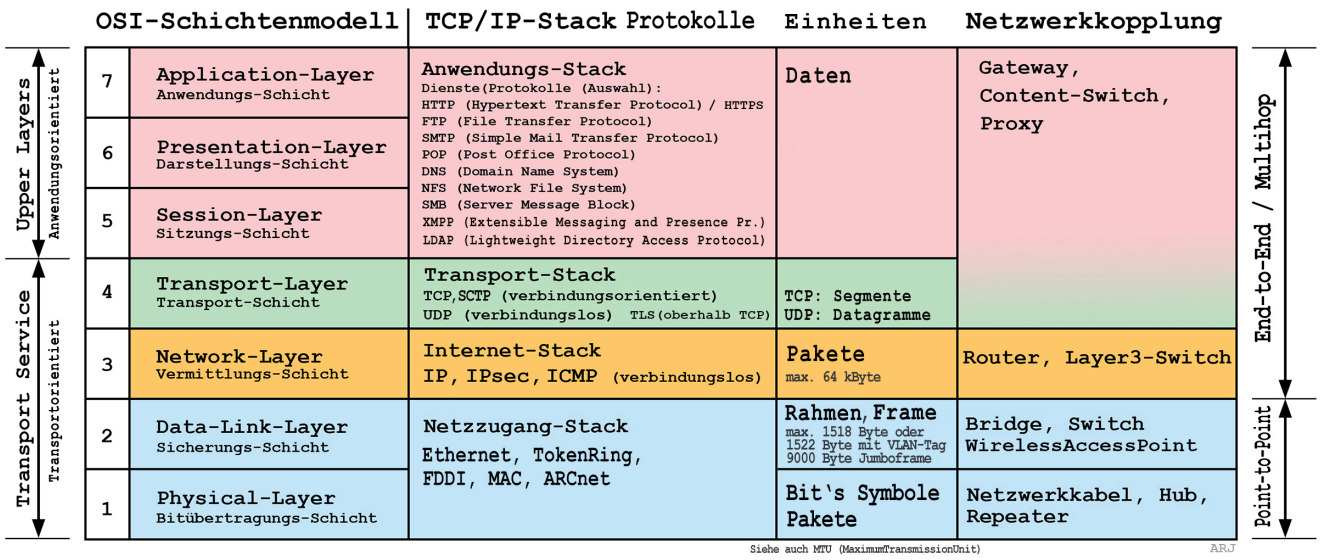
Application Layer (Anwendungsschicht)	**Benutzerschnittstelle, Dienste, Anwendungen und Netzmanagement**
Schicht 3	Die Vermittlungsschicht steuert die zeitliche und logische getrennte Kommunikation zwischen den Endgeräten, unabhängig vom Übertragungsmedium und der Topologie. Auf dieser Schicht erfolgt erstmals die logische Adressierung der Endgeräte. Die Adressierung ist eng mit dem Routing (Wegfindung vom Sender zum Empfänger) verbunden.
Data Link Layer (Sicherungsschicht)	**Logische Verbindungen mit Datenpaketen und elementare Fehlererkennungsmechanismen**
Schicht 2	Die Sicherungsschicht sorgt für eine zuverlässige und funktionierende Verbindung zwischen Endgerät und Übertragungsmedium. Zur Vermeidung von Übertragungsfehlern und Datenverlust enthält diese Schicht Funktionen zur Fehlererkennung, Fehlerbehebung und Datenflusskontrolle. Auf dieser Schicht findet auch die physikalische Adressierung von Datenpaketen statt.
Physical Layer (Physikalische Schicht)	**Maßnahmen und Verfahren zur Übertragung von Bitfolgen**
Schicht 1	Die Bitübertragungsschicht definiert die elektrische, mechanische und funktionale Schnittstelle zum Übertragungsmedium. Die Protokolle dieser Schicht unterscheiden sich nur nach dem eingesetzten Übertragungsmedium und -verfahren. Das Übertragungsmedium ist jedoch kein Bestandteil der Schicht 1.



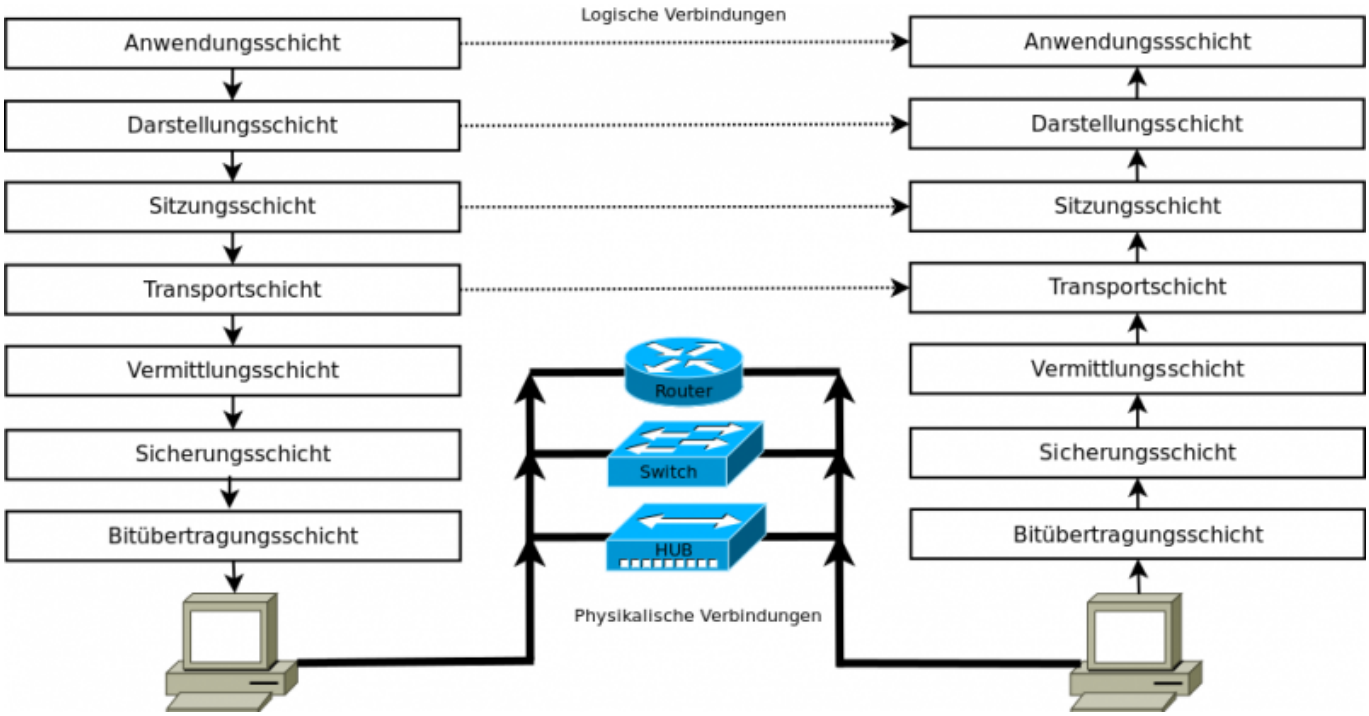
Protokolle

Protokolle sind eine **Sammlung von Regeln zur Kommunikation** auf einer bestimmten Schicht des OSI-Schichtenmodells. Die Protokolle einer Schicht sind zu den Protokollen der über- und untergeordneten Schichten weitestgehend transparent, so dass die Verhaltensweise eines Protokolls sich wie bei einer direkten Kommunikation mit dem Gegenstück auf der Gegenseite darstellt.

Die **Übergänge zwischen den Schichten sind Schnittstellen**, die von den Protokollen verstanden werden müssen. Weil manche Protokolle für ganz bestimmte Anwendungen entwickelt wurden, kommt es auch vor, dass sich **Protokolle über mehrere Schichten** erstrecken und mehrere Aufgaben abdecken. Dabei kommt es vor, dass in manchen Verbindungen einzelne Aufgaben in mehreren Schichten und somit mehrfach ausgeführt werden.



Anmerkung zur Skizze: End-zu-End-Verbindungen finden laut Definition erst ab Schicht 4 statt.



Das wichtigste Protokoll im Netzwerkverkehr, ist das **TCP & IP Protokoll** (Transmission Control Protocol & Internet Protocol), welche sich heute als Standard durchgesetzt haben.

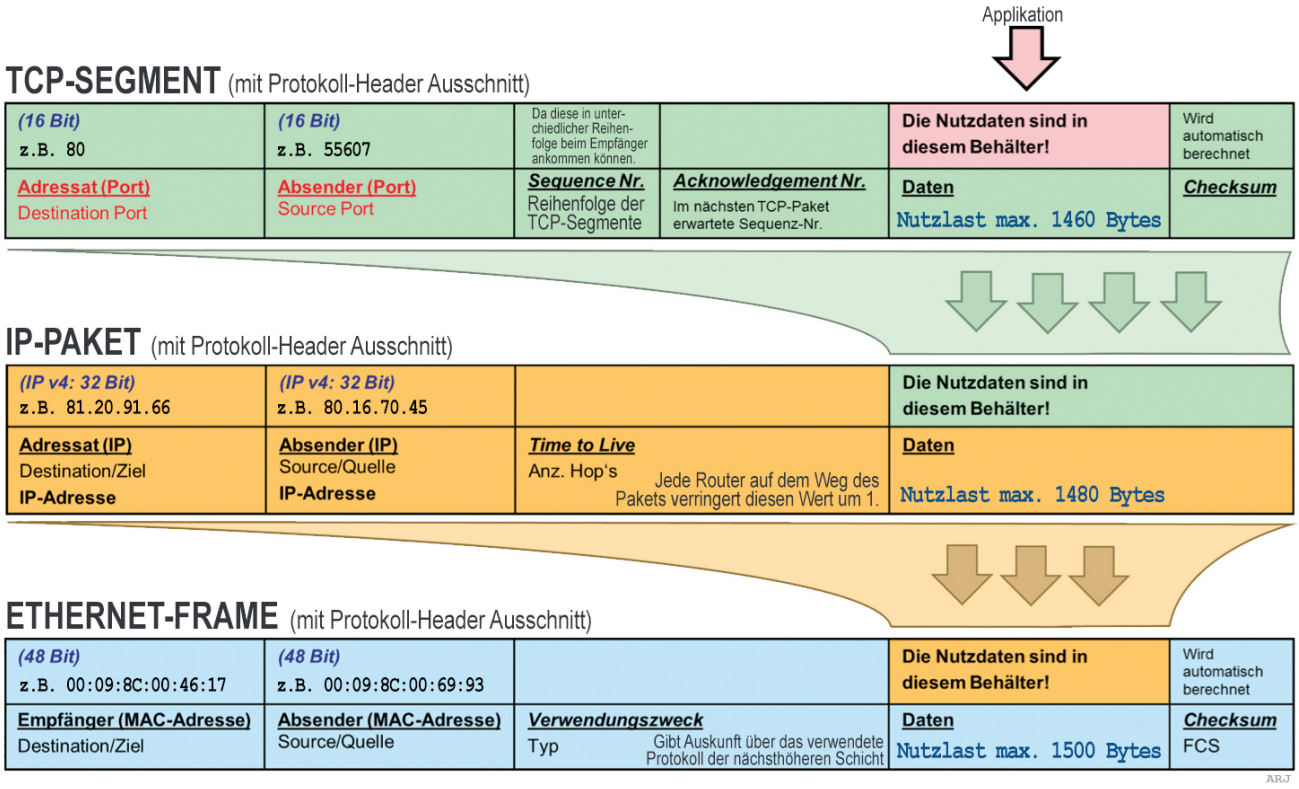
Neben TCP und IP gibt es natürlich noch viele weitere Protokolle, wie in der obigen Abbildung zu sehen ist.

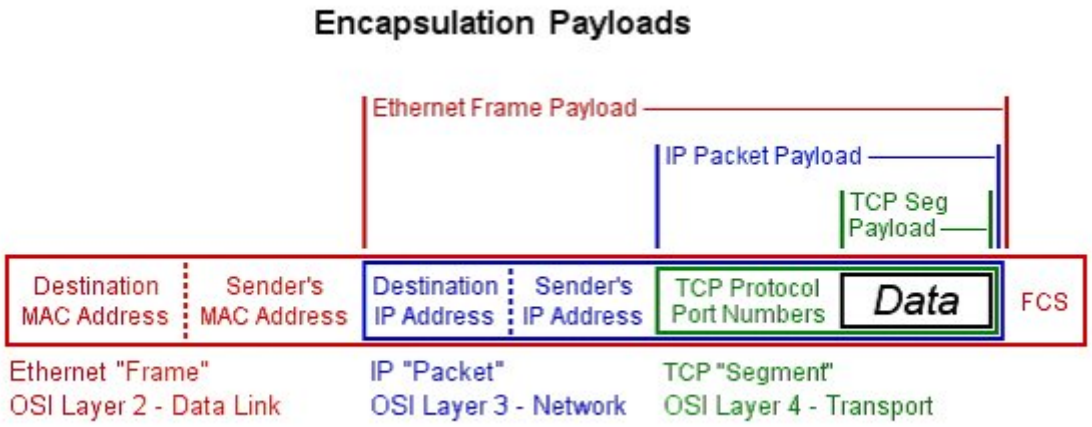
Die folgende Tabelle listet einige dieser Protokolle auf und ordnet sie ins obige Modell ein:

Protokoll	**Schicht**	**Name**	**Beschreibung**
FTP	5	File Transfer Protocol	Datenaustausch zwischen Rechnern
Telnet	5	Telecommunication Network Protocol	Terminalemulation zur Host-Kommunikation
SMTP	5	Simple Mail Transfer Protocol	Versenden von EMails
HTTP	5	Hypertext Transfer Protocol	Übertragen von HTML-Seiten
POP	5	Post Office Protocol	Abrufen von E-Mails
TCP	4	Transmission Control Protocol	Aufbau logischer Verbindungen zwischen Applikationen
UDP	4	User Datagram Protocol	Verbindungsloses Übertragungsprotokoll. Es ist nicht so gesichert wie TCP dafür aber schneller
IP	3	Internet Protocol	Verbindungsloses Protokoll zur Paketlenkung und Paketvermittlung über IP-Adressen
IPSec	3	IP Secure	Erweitert das reguläre IP-Protokoll um ein Bündel von Sicherheitsmechanismen
ARP	3	Address Resolution Protocol	Dien dazu logische IP-Adressen physikalischen MAC-Adressen zuzuordnen

Datenkapselung

Unter der Datenkapselung versteht man den Prozess im OSI-Modell und im TCP/IP-Referenzmodell, der die zu versendenden Daten im Header (und ggf. Trailer) der jeweiligen Schichten ergänzt. Im OSI-Modell betrifft dies die Datagramme der Schichten 2 bis 4, die gekapselt (verpackt) werden.





OSI-Schichtenmodell

TCP/IP-Referenzmodell

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_04

Last update: **2019/05/16 18:35**

18.5) Netzwerkkomponenten

In der Netzwerktechnik unterscheidet man zwischen aktiven und passiven Netzwerk-Komponenten. Während **aktive Netzwerk-Komponenten eine eigene Logik haben**, zählen die passiven Netzwerk-Komponenten zur fest installierten Netzwerk-Infrastruktur. In der Regel dienen Netzwerk-Komponenten zur Kopplung der Netzwerk-Stationen. Man spricht deshalb auch von Kopplungselementen.

Passive Netzwerk-Komponenten

- Patchkabel und Installationskabel
- Anschlussdose
- Steckverbinder
- Patchfeld / Patchpanel
- Netzwerk-Schrank / Patch-Schrank

Hinweis: Zu den passiven Netzwerk-Komponenten zählen die Bestandteile der Verkabelung. Diese ist im OSI-Schichtenmodell nicht definiert.

Aktive Netzwerk-Komponenten

In kleinen privaten Netzwerken, haben Netzwerk-Komponenten noch klare Bezeichnung, wie Switch oder Router. In großen Unternehmensnetzwerken ist die Benennung der Kopplungselemente nicht immer eindeutig.

- Netzwerkkarte
- Repeater
- Hub
- Bridge
- Switch
- Router
- Gateway
- Server

Netzwerkkarte

Eine Netzwerkkarte wird auch als Netzwerkadapter bezeichnet. Die englische Bezeichnung ist Network Interface Card (NIC). Eine Netzwerkkarte ermöglicht es, auf ein Netzwerk zuzugreifen und arbeitet auf der Bitübertragungsschicht (Schicht 1) und der Datensicherungsschicht (Schicht 2) des OSI-Schichtenmodells. Jede Netzwerkkarte hat eine Hardware-Adresse (Format: XX-XX-XX-XX-XX-XX), die es auf der Welt nur einmal gibt. Anhand dieser Adresse lässt sich eine Station auf der Bitübertragungsschicht adressieren.

Im Falle von Ethernet-Netzen besteht die **MAC-Adresse aus 48 Bit (sechs Bytes)**. Die Adressen werden in der Regel **hexadezimal** geschrieben. Üblich ist dabei eine **byteweise Schreibweise**,

wobei die einzelnen Bytes durch Bindestriche oder Doppelpunkte voneinander getrennt werden, z. B. 00-80-41-ae-fd-7e oder 00:80:41:ae:fd:7e. Seltener zu finden sind Angaben wie 008041aefd7e oder 0080.41ae.fd7

In den **ersten 24 Bits** (Bit 3 bis 24) wird eine von der IEEE vergebene **Herstellerkennung** (auch OUI – Organizationally Unique Identifier genannt) beschrieben, die weitgehend in einer Datenbank einsehbar sind[6]. Die **verbleibenden 24 Bits** (Bit 25 bis 48) werden **vom jeweiligen Hersteller** für jede Schnittstelle individuell **festgelegt**.

Repeater

Ein Repeater ist ein Kopplungselement, um die Übertragungsstrecke innerhalb von Netzwerken, zum Beispiel Ethernet, zu verlängern. Ein Repeater empfängt ein Signal und bereitet es neu auf. Danach sendet er es weiter. Auf diese Weise verlängert der Repeater die Übertragungsstrecke und räumliche Ausdehnung des Netzwerks. Im einfachsten Fall hat ein Repeater zwei Ports, die wechselweise als Ein- und Ausgang funktionieren (bidirektional).

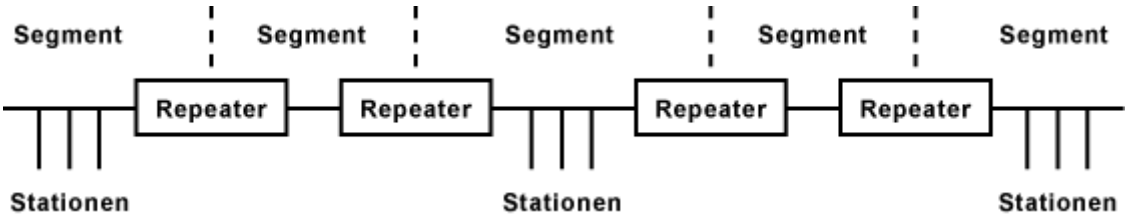
Repeater versteht man in der Regel als Verstärker von Übertragungsstrecken. Die weitere Beschreibung bezieht sich auf Repeater in kabelgebundenen Netzwerken, speziell in Ethernet-Netzwerken.

Ein Repeater arbeitet auf der Schicht 1, der Bitübertragungsschicht des OSI-Schichtenmodells. Der Repeater übernimmt keinerlei regulierende Funktion in einem Netzwerk. Er kann nur Signale empfangen und weiterleiten. Für angeschlossene Geräte ist nicht erkennbar, ob sie an einem Repeater angeschlossen sind. Er verhält sich völlig transparent.

Ein Repeater erweitert somit eine Kollisionsdomäne!!

Ein Repeater mit mehreren Ports wird auch als Hub (Multiport-Repeater) bezeichnet. Er kann mehrere Netzwerk-Segmente miteinander verbinden.

Die Repeater-Regel (5-4-3)



Um ein großes Netzwerk mit einer möglichst großen Reichweite aufzubauen, können mehrere Repeater hintereinandergeschaltet werden. Allerdings, nicht in beliebiger Anzahl. Der Grund liegt im Laufzeitverhalten und der Phasenverschiebung zwischen den Signalen an den Enden des Netzwerks. Deshalb gilt folgende Repeater-Regel:

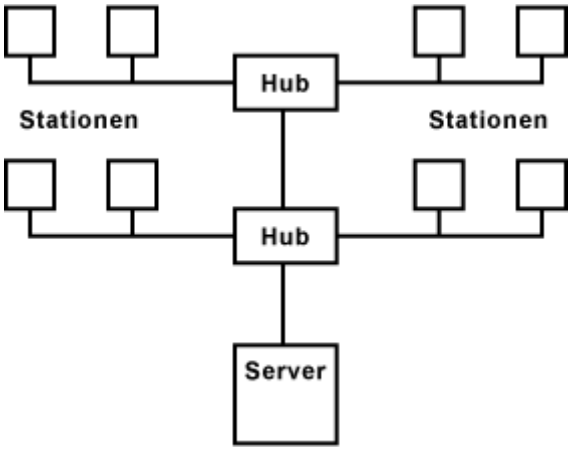
Es dürfen nicht mehr als fünf (5) Kabelsegmente verbunden werden. Dafür werden vier (4) Repeater eingesetzt. An nur drei (3) Segmenten dürfen Endstationen angeschlossen werden.

Diese **Repeater-Regel hat nur in den Ethernet-Netzwerken 10Base2 und 10BASE5** eine

Bedeutung. In Netzwerken, die mit Switches und Router aufgebaut sind, hat diese Repeater-Regel keine Bedeutung. Um die Nachteile von Repeatern in Ethernet-Netzwerken zu umgehen, werden generell Switches zur Kopplung der Hosts eingesetzt. In großen Netzwerken, insbesondere über unterschiedliche Übertragungssysteme hinweg, werden zusätzlich Router eingesetzt.

Hub

Ein Hub ist ein Kopplungselement, das mehrere Hosts in einem Netzwerk miteinander verbindet. In einem Ethernet-Netzwerk, das auf der Stern-Topologie basiert dient ein Hub als Verteiler für die Datenpakete. Hubs arbeiten auf der Bitübertragungsschicht (Schicht 1) des OSI-Schichtenmodells und sind damit auf die **reine Verteilfunktion** beschränkt. **Hubs erweitern somit die Kollisionsdomäne**. Ein Hub nimmt **ein Datenpaket entgegen und sendet es an alle anderen Ports** weiter. Das bedeutet, er **broadcastet**. Dadurch sind nicht nur **alle Ports**, sondern **auch alle Hosts belegt**. Sie bekommen alle Datenpakete zugeschickt, auch wenn sie nicht die Empfänger sind. Für die Hosts bedeutet das auch, dass sie nur dann senden können, wenn der Hub gerade keine Datenpakete sendet. Sonst kommt es zu Kollisionen.



Wenn die Anzahl der Anschlüsse an einem Hub für die Anzahl der Hosts nicht ausreicht, dann benötigt man noch einen zweiten Hub. Zwei Hubs werden über einen Uplink-Port eines der beiden Hubs oder mit einem Crossover-Kabel (Sende- und Empfangsleitungen sind gekreuzt) verbunden. Es gibt auch spezielle „stackable“ Hubs, die sich herstellerspezifisch mit Buskabeln kaskadieren lassen. Durch die Verbindung mehrerer Hubs lässt sich die Anzahl der möglichen Hosts im Netzwerk erhöhen. Allerdings ist die Anzahl der anschließbaren Hosts begrenzt. Hier gilt die Repeater-Regel.

Nachteile

- ineffizient
- unsicher (Jeder bekommt jede Nachricht)

Vorteile

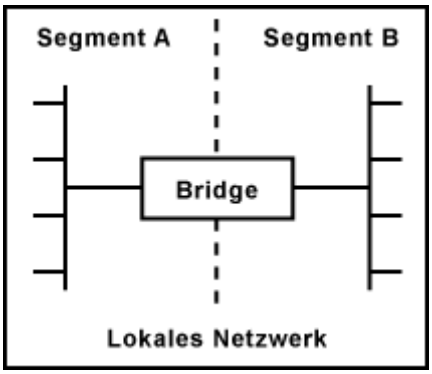
- zentraler Verteiler

Bridge

Eine Bridge ist ein Kopplungselement, das ein lokales Netzwerk in zwei Segmente aufteilt. Dabei

werden die Nachteile von Ethernet, die besonders bei großen Netzwerken auftreten ausgeglichen. Als Kopplungselement ist die Bridge eher untypisch. Man vermeidet die Einschränkungen durch Ethernet heute eher durch Switches.

Eine Bridge teilt eine Kollisionsdomäne!!

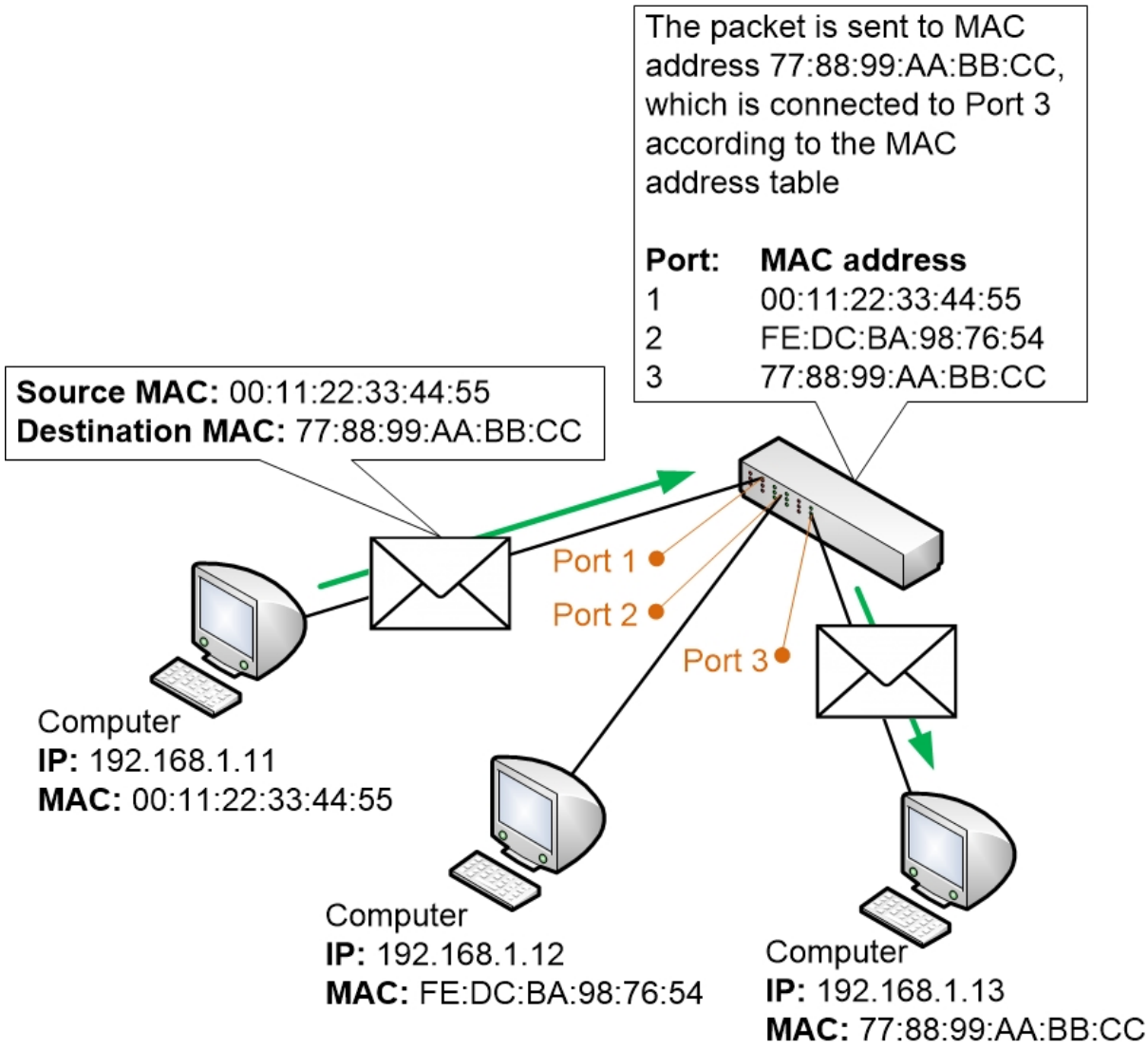


Switch

Ein Switch ist ein Kopplungselement, das mehrere Stationen in einem Netzwerk miteinander verbindet. In einem Ethernet-Netzwerk, das auf der Stern-Topologie basiert dient ein Switch als Verteiler für die Datenübertragung.

Die Funktion ist ähnlich einem Hub, mit dem Unterschied, das ein Switch direkte Verbindungen zwischen den angeschlossenen Geräten schalten kann, sofern ihm die Ports der Datenpaket-Empfänger bekannt sind. Somit kommunizieren wirklich nur jene miteinander, die auch miteinander reduzieren wollen. Wenn nicht, dann broadcastet der Switch die Datenpakete an alle Ports. Wenn die Antwortpakete von den Empfängern zurück kommen, dann merkt sich der Switch die MAC-Adressen der Datenpakete und den dazugehörigen Port und sendet die Datenpakete dann nur noch dorthin. Er baut also eine sogenannte MAC-Adressen-Tabelle auf:

Beispiel:



Während ein Hub die Bandbreite des Netzwerks limitiert, steht der Verbindung zwischen zwei Hosts, die volle Bandbreite der Ende-zu-Ende-Netzwerk-Verbindung zur Verfügung.

Ein Switch arbeitet auf der Sicherungsschicht (Schicht 2) de s OSI-Modells und arbeitet ähnlich wie eine Bridge. Daher haben sich bei den Herstellern auch solche Begriffe durchgesetzt, wie z. B. Bridging Switch oder Switching Bridge. Die verwendet man heute allerdings nicht mehr.

Switches unterscheidet man hinsichtlich ihrer Leistungsfähigkeit mit folgenden Eigenschaften:

- Anzahl der speicherbaren MAC-Adressen für die Quell- und Zielports
- Verfahren, wann ein empfangenes Datenpaket weitervermittelt wird (Switching-Verfahren)
- Latenz (Verzögerungszeit) der vermittelten Datenpakete

Ein Switch ist im Prinzip nichts anderes als ein intelligenter Hub, der sich merkt, über welchen Port welcher Host erreichbar ist. Teure Switches können zusätzlich auf der Schicht 3, der Vermittlungsschicht, des OSI-Schichtenmodells arbeiten (Layer-3-Switch oder Schicht-3-Switch). Sie sind in der Lage, die Datenpakete anhand der IP-Adresse an die Ziel-Ports weiterzuleiten. Im Gegensatz zu normalen Switche lassen sich auch ohne Router logische Abgrenzungen erreichen.

Switching-Verfahren

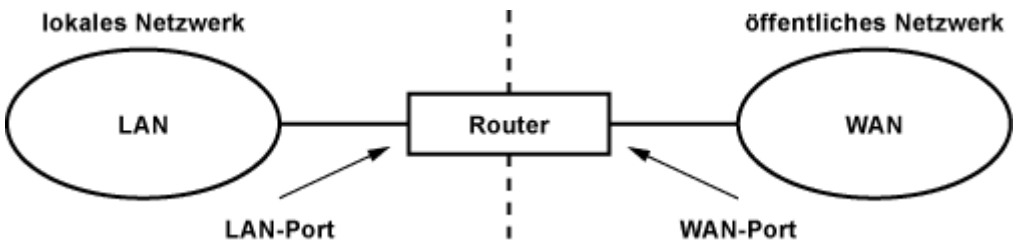
Switching-Verfahren	Beschreibung	Vorteile	Nachteile
Cut-Through	Der Switch leitet das Datenpaket sofort weiter, wenn er die Adresse des Ziels erhalten hat.	Die Latenz, die Verzögerungszeit, zwischen Empfangen und Weiterleiten ist äußerst gering.	Fehlerhafte Datenpakete werden nicht erkannt und trotzdem an den Empfänger weitergeleitet.
Store-and-Forward	Der Switch nimmt das gesamte Datenpaket in Empfang und speichert es in einem Puffer. Dort wird dann das Paket mit verschiedenen Filtern geprüft und bearbeitet. Erst danach wird das Paket an den Ziel-Port weitergeleitet.	Fehlerhafte Datenpakete können so im voraus aussortiert werden.	Die Speicherung und Prüfung der Datenpakete verursacht eine Verzögerung, abhängig von der Größe des Datenpaketes.
Fragment-Free	Der Switch empfängt die ersten 64 Byte des Daten-Paketes. Ist dieser Teil fehlerlos werden die Daten weitergeleitet. Die meisten Fehler und Kollisionen treten während den ersten 64 Byte auf. Dieses Verfahren wird trotz seiner effektiven Arbeitsweise selten genutzt.	sehr effizient	selten genutzt

Router

Ein Router verbindet mehrere Netzwerke mit unterschiedlichen Protokollen und Architekturen. Ein Router befindet sich häufig an den Außengrenzen eines Netzwerks, um es mit dem Internet oder einem anderen, größeren Netzwerk zu verbinden. Über die Routing-Tabelle entscheidet ein Router, welchen Weg ein Datenpaket nimmt. Es handelt sich dabei um ein dynamisches Verfahren, das Ausfälle und Engpässe ohne den Eingriff eines Administrators berücksichtigen kann. Ein Router hat mindestens zwei Netzwerkanschlüssen. Er arbeitet auf der Vermittlungsschicht (Schicht 3) des OSI-Schichtenmodells.

Die Aufgabe eines Routers ist ein komplexer Vorgang, der sich in 4 Schritte einteilen lässt:

- Ermittlung der verfügbaren Routen
- Auswahl der geeignetsten Route unter Berücksichtigung verschiedener Kriterien
- Herstellen einer physikalischen Verbindung zu anderen Netzwerken
- Anpassen der Datenpakete an die Übertragungstechnik (Fragmentierung)

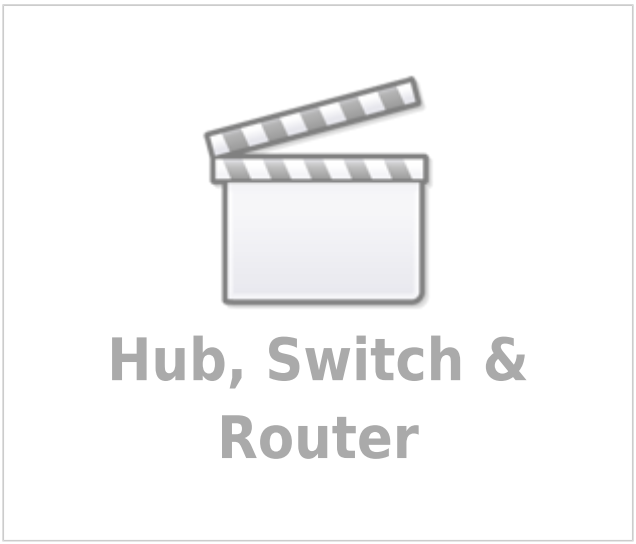


Ein Router hat in der Regel zwei Anschlüsse. Einen für die LAN-Seite und einen für die WAN-Seite.

Häufig sind die Ports mit der Bezeichnung LAN und WAN gekennzeichnet. Manchmal gibt es Port-Beschriftungen, bei denen nicht immer eindeutig ist, um was es sich handelt. Mit LAN ist immer das lokale Netzwerk mit privaten IP-Adressen gemeint, während die WAN-Seite das öffentliche Netzwerk kennzeichnet.



Zusammenfassung



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_05

Last update: **2019/05/16 18:35**

18.6) IP-Adressen

Damit die Wegewahl in Netzwerken und im Internet zur Übermittlung von Datenpaketen vom Senden zum Empfänger funktioniert, wird jedem Rechner im **Internet eine weltweit eindeutige (und einmalige) Adresse (IP-Adresse)** zugeordnet.

Für Rechner die ohne Router direkt mit dem Internet verbunden sind, heißt das, das deren IP-Adressen weltweit eindeutig sind.

Standardmäßig haben diese IP-Adressen eine Länge von **32 Bit (4 x 8Bit)**.

Diese werden aus Gründen der **Übersichtlichkeit in 4 Zahlen zu je 1 Byte** aufgeteilt. Meist werden die einzelnen Bytes durch einen Punkt getrennt und dezimal dargestellt.

Öffentliche IP-Adressen

Wenn man von öffentlichen Adressen spricht, meint man die IP-Adressen, die im Internet erreichbar sind. Die Zuweisung einer öffentlichen IP-Adresse erfolgt in der Regel durch einen Provider (z.B. A1, Telekom, ...) welche diese wiederum in Europa durch die Organisation RIPE-NCC bekommt.

Private IP-Adressen

Private IP-Adressen (abgekürzt Private IP) sind IP-Adressen, die von der **IANA** nicht im Internet vergeben sind. Sie wurden für die **private Nutzung aus dem öffentlichen Adressraum ausgespart**, damit sie ohne administrativen Mehraufwand (Registrierung der IP-Adressen) in lokalen Netzwerken genutzt werden können. Als die **IP-Adressen des Internet Protokolls v4 knapp** wurden und dadurch eine **bewusste Einsparung öffentlicher IP-Adressen** notwendig wurde, war es umso wichtiger, private IP-Adressen in lokalen Netzwerken zur Verfügung zu haben, die **beliebig oft bzw. in beliebigen Netzwerken genutzt** werden können.

Netzklasse: Anzahl Netze (ohne Subnetting)	**Netzadressbereich**	**Subnetzmaske**	**CIDR-Notation**	**Anzahl Adressen**
Klasse A: 1 privates Netz mit 16.777.216 Adressen	10.0.0.0 bis 10.255.255.255	255.0.0.0	10.0.0.0/8	2^{24} = 16.777.216
Klasse B: 16 private Netze mit jeweils 65.536 Adressen	172.16.0.0 bis 172.31.255.255	255.240.0.0	172.16.0.0/12	2^{20} = 1.048.576
Klasse C: 256 private Netze mit jeweils 256 Adressen	192.168.0.0 bis 192.168.255.255	255.255.0.0	192.168.0.0/16	2^{16} = 65.536

Loopback Adresse

Die Class-A-Netzwerkadresse 127 ist weltweit reserviert für das sogenannte local loopback; sie dient zu Testzwecken der Netzwerkschnittstelle des eigenen Rechners. Die IP-Adresse **127.0.0.1** ist standardmäßig dem Loopback-Interface jedes Rechners zugeordnet.

Alle an diese Adresse geschickten Datenpakete werden nicht nach außen ins Netzwerk gesendet, sondern an der Netzwerkschnittstelle reflektiert. Die Datenpakete erscheinen, als kämen sie aus einem angeschlossenem Netzwerk.

Vergleich IP-Adresse vs. Telefonnummer

Ähnlich wie bei einer Telefonnummer setzt sich eine IP-Adresse aus mehreren Segmenten zusammen. Eine Telefonnummer besteht aus einer Vorwahl und einer Teilnehmernummer. Führen Sie ein Ortsgespräch, so muss die Vorwahl nicht angegeben werden. Ähnlich ist es bei IP-Adressen.

Diese bestehen auch aus zwei Teilen, wobei die ID für Identifikation steht:

- **Netzwerk-ID** im vorderen linken Teil entspricht der Vorwahl und gibt das entsprechende IP-Subnetz an.
- **Host-ID** im hinteren rechten Teil kennzeichnet eine einzelne Netzwerkkarte und entspricht dem Teilnehmernummer im Ortsnetz.

Entsprechend können Rechner im selben Subnetz direkt miteinander kommunizieren. Dagegen erfordert Kommunikation zischen Subnetzen eine Vermittlungsstelle, einen Router (Standardgateway), wo alle nicht im selben Netz adressierten Pakete hingeschickt werden.

Um zu erkennen, wo die Netzwerk-ID endet und die Host-ID beginnt, muss zusätzlich zur IP-Adresse zwingend eine sogenannte Subnetzmaske mit angegeben werden.

Subnetzmaske

Eine Subnetzmaske ist ein Bitmuster, das (von links nach rechts) Teile der IP-Adresse „maskiert“, um den Übergang zwischen Netz-ID und Host-ID zu kennzeichnen. Binär betrachtet besteht eine Subnetzmaske aus einer Folge von Einsen, die ab einer bestimmten Stelle umschlägt in eine Folge von Nullen. Dieser Umschlagpunkt gibt an, wie viele Bits zur Netzwerk-ID (Einsen) und zur Host-ID (Nullen) gehören.

Mögliche Subnetzmasken

Maske	Maske (kurze Schreibweise)	Anzahl Hosts pro Netz	Netze	Beispiel Klasse C-Netz (192.168.1.0)
255.255.255.0	/24	256	1	192.168.1.0 - 192.168.1.255
255.255.255.128	/25	128	2	192.168.1.0 - 192.168.1.127 192.168.1.128 - 192.168.1.255

Maske	Maske (kurze Schreibweise)	Anzahl Hosts pro Netz	Netze	Beispiel Klasse C-Netz (192.168.1.0)
255.255.255.192	/26	64	4	192.168.1.0 - 192.168.1.63 192.168.1.64 - 192.168.1.127 192.168.1.128 - 192.168.1.191 192.168.1.192 - 192.168.1.255
255.255.255.224	/27	32	8	192.168.1.0 - 192.168.1.31 192.168.1.32 - 192. 168.1.63 ... 192.168.192 - 192.168.1.223 192.168.1.224 - 192.168.1.255
255.255.255.240	/28	16	16	192.168.1.0 - 192.168.1.15 192.168.1.16 - 192.168.1.31 ... 192.168.1.224 - 192.168.1.239 192.168.1.240 - 192.168.1.255
255.255.255.248	/29	8	32	192.168.1.0 - 192.168.1.7 192.168.1.8 - 192.168.1.15 ... 192.168.1.240 - 192.168.1.247 192.168.1.248 - 192.168.1.255
255.255.255.252	/30	4	64	192.168.1.0 - 192.168.1.3 192.168.1.4 - 192.168.1.7 ... 192.168.1.248 - 192.168.1.251 192.168.1.252 - 192.168.1.255
255.255.255.254	/31	2	128	192.168.1.0 - 192.168.1.1 192.168.1.2 - 192.168.1.3 ... 192.168.1.252 - 192.168.1.253 192.168.1.254 - 192.168.1.255
255.255.255.255	/32	1	256	192.168.1.0 192.168.1.1 ... 192.168.1.254 192.168.1.255



Erklärung

Subnetting

Die Aufteilung eines zusammenhängenden Adressraums von IP-Adressen in mehrere kleinere Adressräume nennt man Subnetting. Ein Subnet, Subnetz bzw. Teilnetz ist ein physikalisches Segment eines Netzwerks, in dem IP-Adressen mit der gleichen Netzwerkadresse benutzt werden. Diese Teilnetze können über Routern miteinander verbunden werden und bilden dann ein großes zusammenhängendes Netzwerk.

Beispiel 1:

IP-Adresse: 192.168.128.17

```
Dezimale Darstellung:           192.        168.        128.          17
Bit-Darstellung:          1100 0000   1010 1000    1000 0000   0001 0001
Hex-Darstellung:                C0          A8          80          11
```

Subnetzmaske: 255.255.255.0

```
Dezimale Darstellung:           255.        255.        255.          0
Bit-Darstellung:           1111 1111   1111 1111   1111 1111  0000 0000
Hex-Darstellung:                FF          FF          FF          00
```

Verknüpft man nun die binäre IP-Adresse mit der Subnetzmaske mit einem Logischen AND, so bekommt man die Netz-ID (=Netzadresse).

```
    <----------- NETZ-ID ------------><- HOST-ID ->
    1100 0000   1010 1000   1000 0000   0001 0001    (IP-Adresse)
AND 1111 1111   1111 1111   1111 1111   0000 0000    (Subnetzmaske)
    ------------------------------------------------------------
    1100 0000   1010 1000   1000 0000   0000 0000    (Netz-ID)
Netz-ID in Dezimaldarstellung:
 192.168.128
```

Verknüpft man nun die binäre IP-Adresse mit der negierten Subnetzmaske mit einem Logischen AND, so bekommt man die Host-ID.

```
    <----------- NETZ-ID ------------><- HOST-ID ->
    1100 0000   1010 1000   1000 0000   0001 0001    (IP-Adresse)
AND 0000 0000   0000 0000   0000 0000   1111 1111    (negierte Subnetzmaske)
    ------------------------------------------------------------------------
-
    0000 0000   0000 0000   0000 0000   0001 0001    (Host-ID)
Host-ID in Dezimaldarstellung:
 17
```

Das heißt im Netzwerk 192.168.128.0 stehen theoretisch 256 (0-255) Adressen zur Adressierung von Netzwerkgeräten zur Verfügung. Praktisch sind es aber nur 254 Adressen, da zwei Adressen

- die **Netzadresse** (= 1. Adresse im Netz –> 192.168.128.0)
- die **Broadcastadresse** (= letzte Adresse im Netz –> 192.168.128.255)

reserviert sind.

Beispiel 2:

```
IP-Adresse 130.94.122.195/27

              Dezimal          Binär                                          
Berechnung

IP Adresse    130.094.122.195  10000010 01011110 01111010 11000011           
ip-adresse
Netzmaske     255.255.255.224  11111111 11111111 11111111 11100000      AND  
netzmaske
------------------------------------------------------------------------------
----------
Netzwerkteil  130.094.122.192  10000010 01011110 01111010 11000000   =      
netzwerkanteil

IP Adresse    130.094.122.195  10000010 01011110 01111010 11000011           
ip-adresse
Netzmaske     255.255.255.224  00000000 00000000 00000000 00011111  AND(NOT  
netzmaske)
------------------------------------------------------------------------------
----------
Geräteteil                  3  00000000 00000000 00000000 00000011       =  
geräteteil
```

Bei einer Netzmaske mit 27 gesetzten Bits ergibt sich ein Netzwerkteil von 130.94.122.192. Es verbleiben 5 Bits und damit `2^5=32` Adressen für den Geräteteil. Hiervon werden noch je 1 Adresse für das Netzwerk selbst und für den Broadcast benötigt, sodass 30 Adressen für Geräte zur Verfügung stehen.

Beispiel 3:

```
IP-Adresse 130.94.122.117/28

              Dezimal          Binär                                          
Berechnung

IP Adresse   130.094.122.117   10000010 01011110 01111010 01110101           
ip-adresse
Netzmaske    255.255.255.240   11111111 11111111 11111111 11110000      AND  
netzmaske
------------------------------------------------------------------------------
----------
Netzwerkteil 130.094.122.112   10000010 01011110 01111010 01110000   =      
netzwerkanteil
```

```
IP Adresse   130.094.122.117   10000010 01011110 01111010 01110101
ip-adresse
Netzmaske    255.255.255.240   00000000 00000000 00000000 00001111  AND(NOT
netzmaske)
-------------------------------------------------------------------------
----------
Geräteteil     0.  0.  0.  5   00000000 00000000 00000000 00000101       =
geräteteil
```

Bei einer Netzmaske mit 28 gesetzten Bits ergibt sich ein Netzwerkteil von 130.94.122.112. Es verbleiben 4 Bits und damit `2^4=16` Adressen für den Geräteteil. Hiervon werden noch je 1 Adresse für das Netzwerk selbst und für den Broadcast benötigt, sodass 14 Adressen für Geräte zur Verfügung stehen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_06

Last update: **2019/05/16 18:35**

18.6.1) Übungen IP-Adressierung

Bsp 1

Befinden sich 192.168.0.93/27 und 192.168.0.97/27 im gleichen Netzwerk-Segment?

Lösung

```
1) NW-ID: 192.168.0.64
2) NW-ID: 192.168.0.96
=> Nein
```

Bsp 2

Wie lauten die Neztwerkadressen des ersten/letzten Hosts des Netzwerkes 192.168.0.96/27 und die Broadcastadresse?

Lösung

```
1. Host: 192.168.0.97
letzter Host: 192.168.0.126
BC: 192.168.0.127
```

Bsp 3

Das Netz 195.1.31.0 soll in 30 Subnetze aufgeteilt werden. Bestimme die Subnetzmaske, und die Anzahl Adressen pro Subnet!

Lösung

```
Subnetzmaske: 255.255.255.248
Adressen: 6
```

Bsp 4

Das Netz 10.0.0.0 soll in 200 Subnetze aufgeteilt werden. Bestimme die Subnetzmaske, und die Anzahl Adressen pro Subnet!

Lösung

```
Subnetzmaske: 255.255.0.0
```

```
Adressen: 65534
```

Bsp 5

Das Netz 192.168.1.0 soll in Subnetze mit je 18 Host-Adressen aufgeteilt werden. Bestimme die Subnetzmaske, und die Anzahl Subnetze!

Lösung

```
Subnetzmaske: 255.255.255.224
Subnetze: 6
```

Bsp 6

Das Netz 192.168.100.0 soll in Subnetze mit je 5 Host-Adressen aufgeteilt werden. Bestimmen Sie die Subnetzmaske, und die Anzahl Subnetze!

Lösung

```
Subnetzmaske: 255.255.255.248
Subnetze: 30
```

Bsp 7

Bestimme Netz- und Broadcastadresse des Subnets, in dem die Adresse 195.1.31.135 mit Netzmaske 255.255.255.128 liegt!

Lösung

```
NW-ID: 195.1.31.128
BC: 195.1.31.255
```

Bsp 8

Bestimme Netz- und Broadcastadresse des Subnets in dem die Adresse 195.1.31.135 mit Netzmaske 255.255.255.192 liegt!

Lösung

```
NW-ID: 195.1.31.128/26
BC: 195.1.31.191
```

Bsp 9

Bestimme Netz- und Broadcastadresse des Subnets in dem die Adresse 15.3.128.222 mit Netzmaske 255.255.255.240 liegt!

Lösung

```
NW-ID: 15.3.128.208
Broadcast: 15.3.128.223
```

Bsp 10

Gib den Bereich der Rechneradressen an, den das Teilnetz Nummer drei (132.45.96.0/19) hat.

Lösung

```
132.45.96.1
bis
132.45.127.254
```

Bsp 11

Berechne Netz-, Broadcast und Hostadressen des 0.,1.,15. und 31 Subnetz des Netzes 192.168.1.0/29

Lösung

```
0. Netz:
* NW-ID: 192.168.1.0
* BC: 192.168.1.7
* Hosts: 192.168.1.1-6
```

```
1. Netz:
* NW-ID: 192.168.1.8
* BC: 192.168.1.15
* Hosts: 192.168.1.9-14
```

```
15. Netz:
* NW-ID: 192.168.1.120
* BC: 192.168.1.127
* Hosts: 192.168.1.121-126
```

```
31. Netz:
* NW-ID: 192.168.1.248
* BC: 192.168.1.255
* Hosts: 192.168.1.249-254
```

Bsp 12

Vervollständige folgende Tabelle:

IP-Adresse	10.1.1.1	Netzwerkadresse des Subnet	. . .
Netmask	255.255.0.0	Broadcastadresse des Subnet	. . .
Anzahl Host pro Subnet		Anzahl Subnets in diesem Netz	

Lösung

IP-Adresse	10.1.1.1	Netzwerkadresse des Subnet	10.1.0.0
Netmask	255.255.0.0	Broadcastadresse des Subnet	10.1.255.255
Anzahl Host pro Subnet	65536	Anzahl Subnets in diesem Netz	256

Bsp 13

Vervollständige folgende Tabelle:

IP-Adresse	10.1.1.1/24	Netzwerkadresse des Subnet	. . .
Netmask	255.255.255.0	Broadcastadresse des Subnet	. . .
Anzahl Host pro Subnet		Anzahl Subnets in diesem Netz	

Lösung

IP-Adresse	10.1.1.1/24	Netzwerkadresse des Subnet	10.1.1.0
Netmask	255.255.255.0	Broadcastadresse des Subnet	10.1.1.255
Anzahl Host pro Subnet	254	Anzahl Subnets in diesem Netz	65536

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_06:18_06_1

Last update: **2019/05/16 18:35**

18.7) IP-Routing

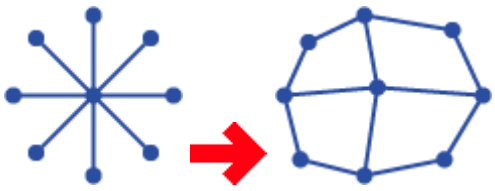
Das **Internet Protocol (IP)** ist ein **routingfähiges Protokoll** und sorgt dafür, dass **Datenpakete über Netzgrenzen** hinweg einen Weg zu anderen Hosts finden. Es kann die Daten über jede Art von physikalischer Verbindung oder Übertragungssystem vermitteln. Der hohen Flexibilität steht ein hohes Maß an Komplexität bei der Wegfindung vom Sender zum Empfänger gegenüber. Der **Vorgang der Wegfindung wird Routing** genannt

Wozu Routing?

Das grundlegende Verbindungselement in einem Ethernet-Netzwerk ist der Hub oder Switch. Daran sind alle Netzwerk-Teilnehmer angeschlossen. Wenn ein Host Datenpakete verschickt, dann werden die Pakete im Hub an alle Stationen verschickt und von diesen angenommen. Jedoch verarbeitet nur der adressierte Host die Pakete weiter. Das bedeutet aber auch, dass sich alle Hosts die Gesamtbandbreite dieses Hubs teilen müssen.

Um die Nachteile von Ethernet in Verbindung mit CSMA/CD auszuschließen, wählt man als Kopplungselement einen Switch und nutzt Fast Ethernet (kein CSMA/CD mehr). Der Switch merkt sich die Hardware-Adressen (MAC-Adressen) der Stationen und leitet die Ethernet-Pakete nur an den Port, hinter dem sich die Station befindet. Ist einem Switch die Hardware-Adresse nicht bekannt, leitet er das Datenpaket an alle seine Ports weiter (Broadcast) und funktioniert in diesem Augenblick wie ein Hub. Neben der begrenzten Speichergröße des Switches machen sich viele unbekannte Hardware-Adressen negativ auf die Performance eines Netzwerks bemerkbar.

Daher eignet sich zum **Verbinden großer Netzwerke** weder ein Hub noch ein Switch. Aus diesem Grund wird ein Netzwerk **durch Router** und IP-Adressen in **logische Segmente bzw. Subnetze** unterteilt. Die Adressierung durch das Internet Protocol ist so konzipiert, dass der Netzwerkverkehr innerhalb der Subnetze bleibt und erst dann das Netzwerk verlässt, wenn das Ziel in einem anderen Netzwerk liegt.



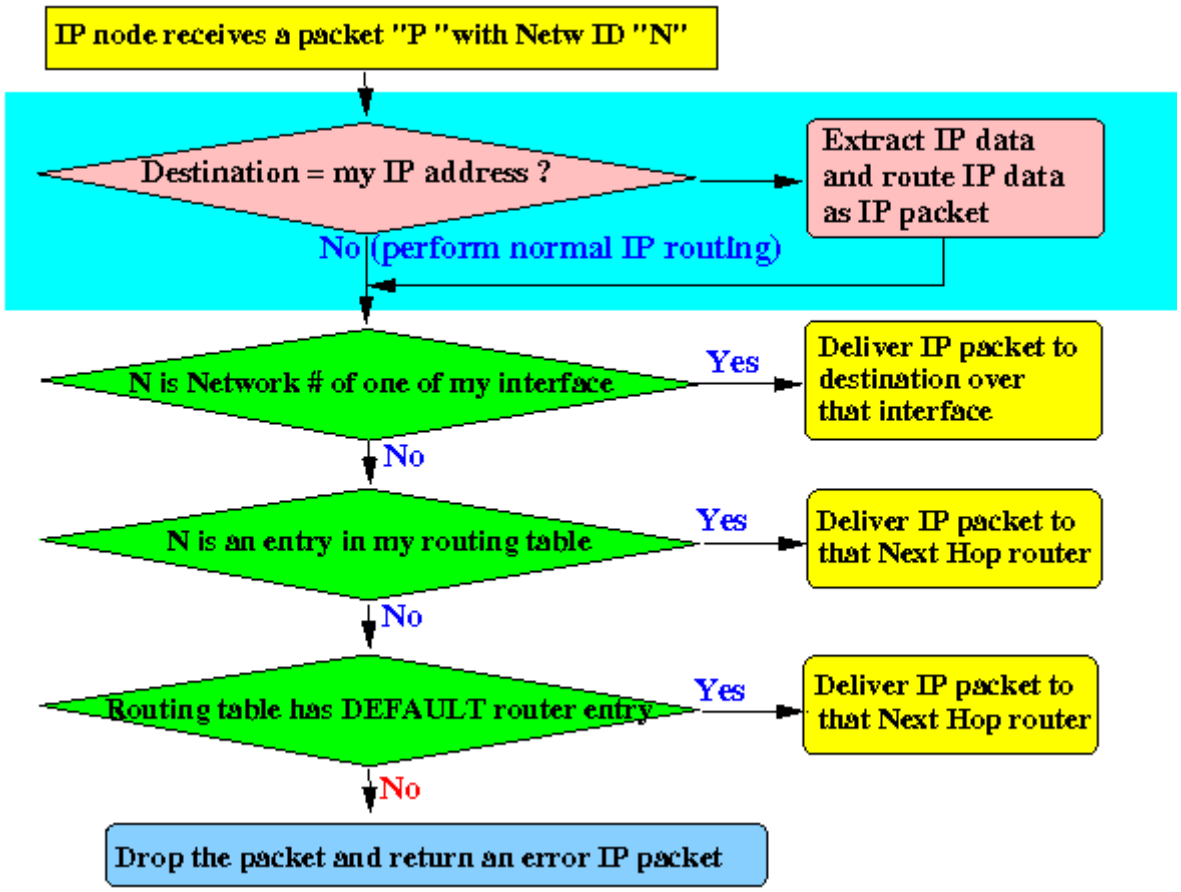
Insbesondere folgende **Probleme** in einem Ethernet-Netzwerk machen **IP-Routing notwendig**:

- **Vermeidung von Kollisionen und Broadcasts durch Begrenzung der Kollisions- und Broadcastdomäne**
- **Routing über unterschiedliche Netzarchitekturen und Übertragungssysteme**
- **Paket-Filter und Firewall**
- **Routing über Backup-Verbindungen bei Netzausfall**

IP-Routing-Algorithmus

Der IP-Routing-Algorithmus gilt nicht nur für IP-Router, sondern für alle Host, die IP-Datenpakete

empfangen können. Die empfangenen Datenpakete durchlaufen diesen Algorithmus bis das Datenpaket zugeordnet oder weitergeleitet werden kann.



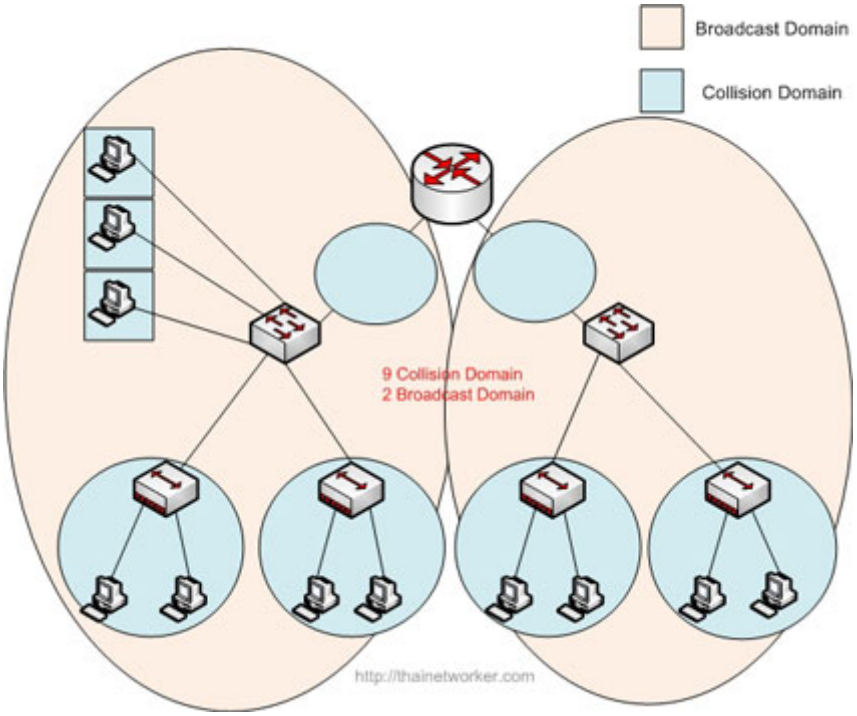
- Prüfe, ob das Datenpaket mir gehört?
  - Wenn ja, dann hat das Datenpaket sein Ziel erreicht und kann verarbeitet werden.
  - Wenn nein, prüfe ob das Datenpaket in mein Subnetz gehört?
    - Wenn ja, schick es in das eigene Subnetz weiter!
    - Wenn nein, prüfe ob die Route zum Empfänger bekannt ist?
      - Wenn ja, schicke es über die bekannte Route zum nächsten Router!
      - Wenn nein, prüfe ob es ein Standard-Gateway gibt?
        - Wenn ja, schick das Paket zum Standard-Gateway!
        - Wenn nein, schreibe eine Fehlermeldung und verwirf das Datenpaket!

Broadcastdomäne

Eine Broadcast-Domäne ist ein logischer Verbund von Netzwerkgeräten in einem lokalen Netzwerk, der sich dadurch auszeichnet, dass ein **Broadcast alle Domänenteilnehmer erreicht**.

Ein lokales Netzwerk auf der 2. Schicht des OSI-Modells (Sicherungsschicht) besteht durch seine Hubs, Switches und/oder Bridges aus einer Broadcast-Domäne. Erst durch die Unterteilung in VLANs oder durch den Einsatz von Routern, die auf Schicht 3 arbeiten, wird die Broadcast-Domäne aufgeteilt.

Eine Broadcast-Domäne besteht aus einer oder mehreren Kollisionsdomänen.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_07

Last update: **2019/05/16 18:36**

18.8) Netzwerkbefehle

Windows-Kommandos

Wichtige Netzwerkbefehle unter Windows XP

- ipconfig
  - /all ... detaillierte Informationen über Netzwerk-Konfiguration
  - /renew ... erneuert IP-Adressen
  - /release ... gibt IP-Adressen frei
- ping (sendet Datenpakete zu Rechner)
  - ping IP-Adresse
  - ping hostname
  - -n Anzahl ... Anzahl der Pakete
- tracert (Route zu Rechner)
- nslookup
  - nslookup (DNS-Abfragen)
  - nslookup IP-Adresse
  - nslookup Domain-Name

Linux-Kommandos

- ifconfig

Zeigt Informationen zu Netzwerk-Interfaces

- ping

```
ping -c 4 www.example.com
```

Pingt 4 mal www.example.com

- tracepath

```
tracepath www.example.com
```

Zeigt Hops zum Host an (benötigt eventuell SU-Rechte)

```
sudo tracepath www.example.com
```

- nslookup

```
nslookup www.example.com
```

Überprüft DNS-Eintrag

weitere hilfreiche Befehle:

- nmap (scannt („mappt“) das Netzwerk, führt Portscans aus und findet die Software eines fremden PCs heraus)
- mtr (kombiniert tracert (ohne su-Rechte) und ping, anschauliche Darstellung)

Fragen - Aufgaben - Arbeitsaufträge

Arbeitsaufträge:

1. Gib deiner/m Nachbarn/in die IP-Adresse deines Computers und notiere dir die IP-Adresse seines/ihres Computers.
2. Versuche deinen Nachbarcomputer anzupingen. Wie lange brauchte das Datenpaket zu diesem Rechner?
3. Gib deine IP-Adresse frei und versuche den Nachbarn anzupingen bzw. dich anpingen zu lassen. Erneuere anschließend deine IP-Adresse und überprüfe, wie sie nun lautet.
4. Versuche deinen Rechner (mit deiner IP-Adresse und der Loopback-Adresse) anzupingen.
5. Finde heraus, welche IP-Adresse der Domain-Name mail.bgamstetten.ac.at hat.
6. Finde einen Rechner im Internet, bis zu welchem ein Datenpaket sehr lange dauert und schicke zu diesem Rechner 20 Datenpakete hintereinander.
7. Finde heraus, welcher Domain-Name zu folgender IP-Adresse gehört 131.130.250.250
8. Lasse dir die Route zu www.yahoo.com anzeigen. Wie viele Rechner liegen zwischen dir und dem Webserver von www.yahoo.com ?
9. Versuche mit dem Internetexplorer die Website von www.google.at aufzurufen, indem du die IP-Adresse von Google in der Browserzeile eingibst.
10. Suche im Internet die Homepage von der australischen Regierung. Welche IP-Adresse hat der Rechner, auf dem die Homepage liegt? Wie viele Rechner liegen zwischen dir und diesem Rechner?
11. Finde einen Rechner im Internet, bis zu dem möglichst viele Hops dazwischen sind (Wer findet die meisten?).

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_08

Last update: **2019/05/16 18:36**

18.9) Protokolle

Protokolle sind eine **Sammlung von Regeln zur Kommunikation** auf einer bestimmten Schicht des OSI-Schichtenmodells. Die Protokolle einer Schicht sind zu den Protokollen der über- und untergeordneten Schichten weitestgehend transparent, so dass die Verhaltensweise eines Protokolls sich wie bei einer direkten Kommunikation mit dem Gegenstück auf der Gegenseite darstellt.

Die **Übergänge zwischen den Schichten sind Schnittstellen**, die von den Protokollen verstanden werden müssen. Weil manche Protokolle für ganz bestimmte Anwendungen entwickelt wurden, kommt es auch vor, dass sich **Protokolle über mehrere Schichten** erstrecken und mehrere Aufgaben abdecken. Dabei kommt es vor, dass in manchen Verbindungen einzelne Aufgaben in mehreren Schichten und somit mehrfach ausgeführt werden.

Protokoll-Stack

Da sich ein einzelnes Protokoll immer nur um eine Teilaufgabe im Rahmen der Kommunikation kümmert, werden mehrere Protokolle zu Protokollsammlungen oder Protokollfamilien, den sogenannten Protokoll-Stacks, zusammengefasst. Die wichigsten Einzel-Protokolle werden dann oft stellvertertretend als Bezeichnung des gesamten Protokoll-Stapels genutzt.

Kommunikation zwischen Netzwerkkomponenten funktioniert nur dann, wenn sie denselben Protokoll-Stack benutzen oder wenn Geräte eingesetzt werden, die zwischen verschiedenen Stacks vermitteln können.

Portnummern (ab Layer 5)

Um die einzelnen Dienste (Protokolle), die bei einem Rechner über dieselbe IP-Adresse ausgeführt werden, voneinander zu differenzieren, wurden Portnummern eingeführt, um bei einer Anfrage deutlich zu machen, welcher Dienst gemeint ist.
Diese Portnummern, die im TCP- oder UDP Header angegeben werden, sind weltweit eindeutig festgelegt und können z.B. auf der Homepage der IANA eingesehen werden.
Allgemein lässt sich also sagen, dass die **IP-Adresse** den Rechner, **und** Die **Port-Nummer** den Dienst auf dem jeweiligen Rechner angibt. Diese beiden Informationen zusammen werden als **Socket** bezeichnet.

Wichtige Protokolle

OSI-LAYER	PROTOKOLL (Port)
5-7	DHCP (67+68/UDP)
	DNS (53/UDP)
	HTTP (80/TCP)
	HTTPS (443/TCP)
	FTP (20+21/TCP)
	SSH (22/TCP)
	SMTP (465 und 587 oder 25/TCP)
	POP3 (995 oder 110/TCP)
	IMAP (993 oder 143/TCP)
4	TCP
	UDP
3	IPv4
	IPv6
	NAT
	ICMP
2	Ethernet 802.3 , Wireless 802.x, MAC, ISDN,...
1	

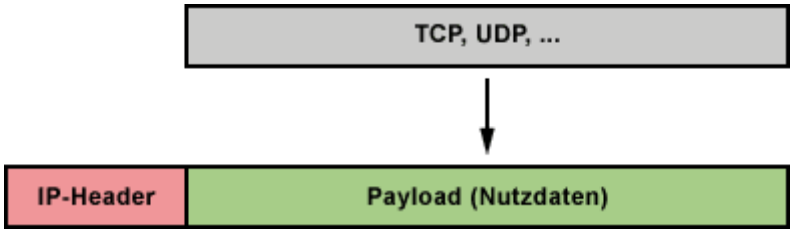
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09

Last update: **2019/05/16 18:36**

IPv4 - Internet Protocol Version 4 (Layer 3)

Das Internet Protocol, kurz IP, wird im Rahmen der Protokollfamilie TCP/IP zur Vermittlung von Datenpaketen verwendet. Es arbeitet auf der **Schicht 3 des OSI-Schichtenmodells** und hat maßgeblich die Aufgabe, **Datenpakete zu adressieren** und in einem dezentralen, **verbindungslosen und paketorientierten Netzwerk zu übertragen**. Dazu haben alle Netzwerk-Teilnehmer eine eigene IP-Adresse im Netzwerk. Sie dient nicht nur zur Identifikation eines Hosts, sondern auch des Netzes, in dem sich der jeweilige Host befindet.

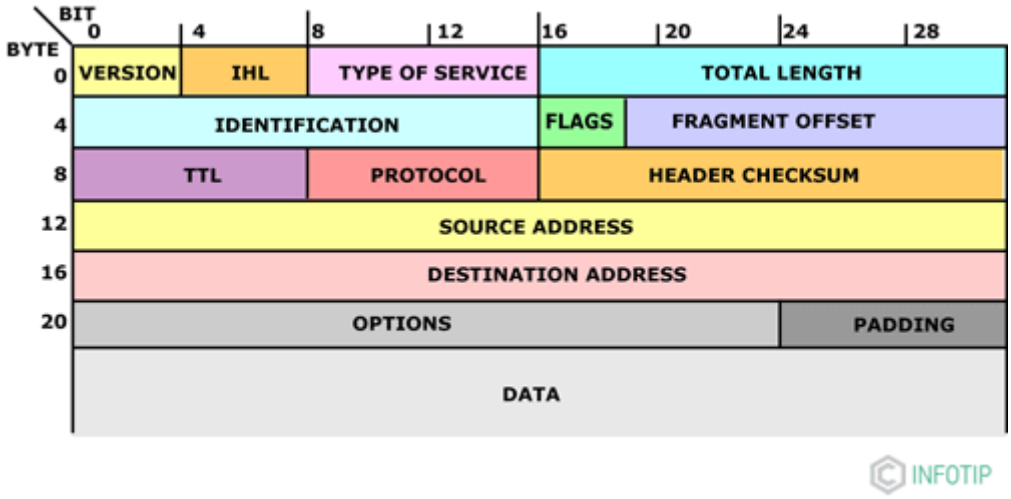


Aufgaben und Funktionen von IPv4

- Logische Adressierung (IPv4-Adresse)
- IPv4-Konfiguration
- IPv4-Header
- IP-Routing
- Namensauflösung (DNS-Dienst)

IPv4 Header

Jedes IPv4-Datenpaket besteht aus einem Header (Kopf) und dem Payload, in dem sich die Nutzdaten befinden. Der Header ist den Nutzdaten vorangestellt. Im IP-Header sind Informationen enthalten, die für die Verarbeitung durch das Internet Protocol notwendig sind.



Der Header ist in jeweils 32-Bit-Blöcke unterteilt. Dort sind Angaben zu Servicetypen, Paketlänge, Sender- und Empfängeradresse abgelegt. Ein IP-Paket muss mindestens 20 Byte Header und 8 Byte Nutzdaten bzw. Nutz- und Fülldaten enthalten. Die Gesamtlänge eines IP-Pakets darf 65.535 Byte nicht überschreiten.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_01

Last update: **2019/05/16 18:20**

IPv6 - Internet Protocol Version 6 (Layer 3)

IPv6 ist als Internet Protocol (Version 6) für die Vermittlung von Datenpaketen durch ein paketvermittelndes Netz, die Adressierung von Netzknoten und -stationen, sowie die Weiterleitung von Datenpaketen zwischen Teilnetzen zuständig. Mit diesen Aufgaben ist IPv6 der Schicht 3 des OSI-Schichtenmodells zugeordnet. Die Aufgabe des Internet-Protokolls besteht im Wesentlichen darin, Datenpakete von einem System über verschiedene Netzwerke hinweg zu einem anderen System zu vermitteln (Routing).

IPv6 ist der direkte Nachfolger von IPv4 und Teil der Protokollfamilie TCP/IP. Seit Dezember 1998 steht IPv6 bereit und wurde hauptsächlich wegen der Adressknappheit und verschiedener Unzulänglichkeiten von IPv4 entwickelt spezifiziert. Da weltweit immer mehr Menschen, Maschinen und Geräte an das Internet mit einer eindeutigen Adresse angeschlossen werden sollen, reichen die 4 Milliarden IPv4-Adressen nicht mehr aus.

Warum IPv6?

IPv6 gilt als Wunderwaffe gegen so manche Probleme mit Netzwerkprotokollen und gleichzeitig wird es als Teufelszeug verdammt, das wieder neue unbekannte Probleme hervorruft. Eine Tatsache ist, dass Administratoren, Programmierer und Hersteller IPv6 neu lernen müssen. Viele Rezepte aus der IPv4-Welt taugen unter IPv6 nicht mehr. Erschwerend kommt hinzu, dass es bei IPv6 allen Beteiligten an Erfahrung fehlt. IPv6-Gurus, die man bei einem großen Problem befragen kann, gibt es nicht so viele.

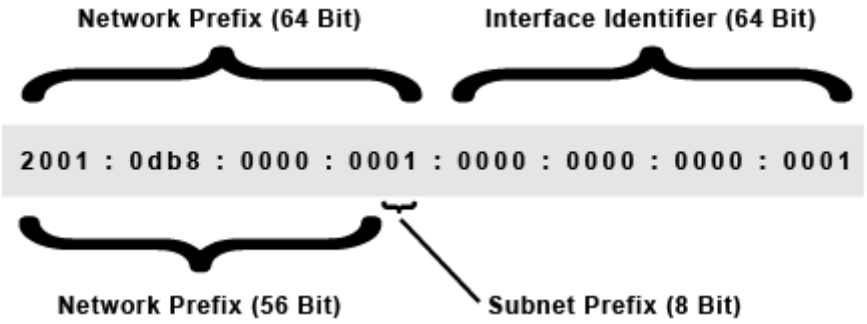
Bei IPv6 ist das Ende-zu-Ende-Prinzip konsequent weiter gedacht. Ein Interface kann mehrere IPv6-Adressen haben und es gibt spezielle IPv6-Adressen, denen mehrere Interfaces zugeordnet sind. IPv6 löst also nicht nur die Adressknappheit, sondern bietet auch Erleichterungen bei der Konfiguration und im Betrieb. Die zustandslose IPv6-Konfiguration und verbindungslokalen Adressen, die bereits nach dem Computerstart verfügbar sind, vereinfachen die Einrichtung und den Betrieb eines lokalen Netzwerks. Damit das gelingt sind Planer und Errichter von IP-Netzen gefordert sich eine neue Denkweise anzueignen.

IPv6 Adressen

Eine IPv6-Adresse ist eine Netzwerk-Adresse, die einen Host eindeutig innerhalb eines IPv6-Netzwerks logisch adressiert. Die Adresse wird auf IP- bzw. Vermittlungsebene (des OSI-Schichtenmodells) benötigt, um Datenpakete verschicken und zustellen zu können. Im Gegensatz zu anderen Adressen hat ein IPv6-Host mehrere IPv6-Adressen, die unterschiedliche Gültigkeitsbereiche haben. Konkret bedeutet das, dass wenn von IPv6-Adressen die Rede ist, dass nicht immer klar ist, welchen Gültigkeitsbereich diese IPv6-Adressen aufweisen. Grob unterscheidet man zwischen verbindungslokalen und globalen IPv6-Adressen. Die verbindungslokale IPv6-Adresse ist nur im lokalen Netzwerk gültig und wird nicht geroutet. Die globale IPv6-Adresse ist über das lokale Netzwerk hinaus im Internet gültig.

Eine IPv6-Adresse hat eine Länge von 128 Bit. Diese Adresslänge erlaubt eine unvorstellbare Menge von 2128 oder 3,4 x 1038 IPv6-Adressen. Das sind 340.282.366.900.000.000.000.000.000.000.000.000.000 IPv6-Adressen, also rund 340 Sextillionen Adressen. Bei IPv4 spricht man von rund 4,3 Milliarden Adressen. Der Adressraum von IPv6 reicht aus,

um umgerechnet jeden Quadratmillimeter der Erdoberfläche inklusive der Ozeane mit rund 600 Billiarden Adressen zu pflastern.



Eine IPv6-Adresse besteht aus 128 Bit. Wegen der unhandlichen Länge werden die 128 Bit in 8 mal 16 Bit unterteilt. Je 4 Bit werden als eine hexadezimale Zahl dargestellt. Je 4 Hexzahlen werden gruppiert und durch einen Doppelpunkt („:“) getrennt. Um die Schreibweise zu vereinfachen lässt man führende Nullen in den Blöcken weg. Eine Folge von 8 Nullen kann man durch zwei Doppelpunkte („::“) ersetzen.

Eine IPv6-Adresse besteht aus zwei Teilen. Dem Network Prefix (Präfix oder Netz-ID) und dem Interface Identifier (Suffix, IID oder EUI). Der Network Prefix kennzeichnet das Netz, Subnetz bzw. Adressbereich. Der Interface Identifier kennzeichnet einen Host in diesem Netz. Er wird aus der 48-Bit-MAC-Adresse des Interfaces gebildet und dabei in eine 64-Bit-Adresse umgewandelt. Es handelt sich dabei um das Modified-EUI-64-Format. Auf diese Weise ist das Interface unabhängig vom Network Prefix eindeutig identifizierbar.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_02

Last update: **2019/05/16 18:20**

NAT - Network Address Translation (Layer 3)

NAT (Network Address Translation) ist ein Verfahren, dass in **IP-Routern eingesetzt** wird, die lokale Netzwerke mit dem Internet verbinden. Weil Internet-Zugänge in der Regel nur über **eine einzige öffentliche** und damit routbare IPv4-Adresse verfügen, müssen sich alle anderen Hosts im lokalen Netzwerk mit privaten IPv4-Adressen begnügen. **Private IP-Adressen** dürfen zwar **mehrfach verwendet** werden, aber besitzen **in öffentlichen Netzen keine Gültigkeit**. Hosts mit einer privaten IPv4-Adresse können somit nicht mit Hosts außerhalb des lokalen Netzwerks kommunizieren.

NAT ist allerdings nur eine mittlerweile sehr lang andauernde Notlösung, um die Adressknappheit von IPv4 zu umgehen. Um die damit einhergehenden Probleme zu lösen muss langfristig auf ein Internet-Protokoll mit einem größeren Adressraum umgestellt werden. IPv6 ist ein solches Protokoll.

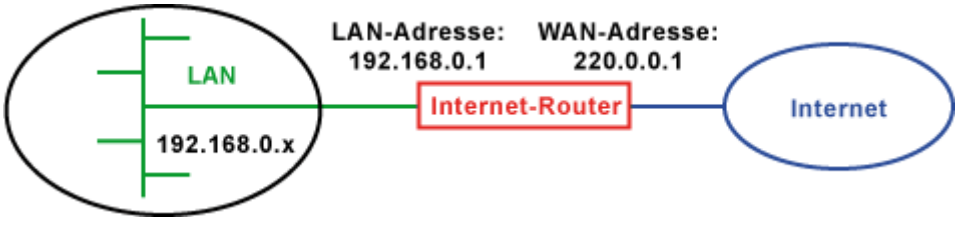
Warum NAT?

Die **ersten IPv4-Netze** waren **anfangs** eigenständige Netz **ohne Verbindung nach außen**. Hier begnügte man sich mit IPv4-Adressen aus den privaten Adressbereichen. Parallel dazu kam es bereits **Ende der 1990er** Jahre zu **Engpässen bei öffentlichen IPv4-Adressen**. Die steigende Anzahl der Einwahlzugänge über das Telefonnetz mussten mit IPv4-Adressen versorgt werden. Bis heute bekommt **ein Internet-Anschluss** nur eine **IPv4-Adresse für ein Gerät**. Damals war es undenkbar, dass an einem Internet-Anschluss ein ganzes Heimnetzwerk betrieben wird. Wenn ein Haushalt einen PC per Modem an das Telefonnetz angeschlossen und sich ins Internet eingewählt hat, dann war das schon etwas besonderes.

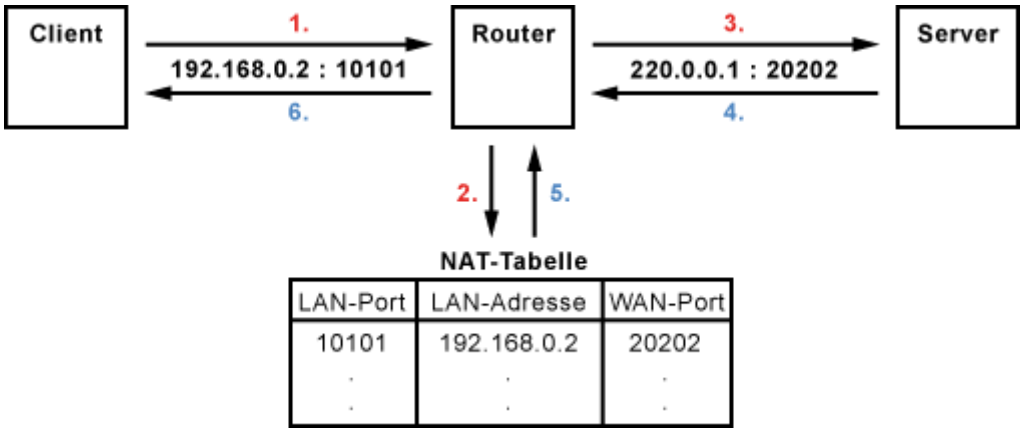
Internet in Österreich

NAT

Der Betrieb eines NAT-Routers ist üblicherweise an einem gewöhnlichen Internet-Anschluss. Zum Beispiel über DSL oder Kabelmodem. Der eingesetzte Router dient als Zugang zum Internet und als Standard-Gateway für das lokale Netzwerk. In der Regel wollen über den Router mehr Geräte ins Internet, als öffentliche IP-Adressen zur Verfügung stehen. In der Regel nur eine einzige. Beispielsweise bekommt der Router des lokalen Netzwerks die öffentliche IP-Adresse 222.0.0.1 für seinen WAN-Port vom Internet Service Provider (ISP) zugewiesen. Weil nur eine öffentliche IP-Adresse vom Internet-Provider zugeteilt wurde, bekommen die Stationen im LAN private IP-Adressen aus speziell dafür reservierten Adressbereichen zugewiesen. Diese Adressen sind nur innerhalb des privaten Netzwerks gültig. Private IP-Adressen werden in öffentlichen Netzen nicht geroutet. Das bedeutet, dass Stationen mit privaten IP-Adressen keine Verbindung ins Internet bekommen können. Damit das trotzdem funktioniert, wurde NAT entwickelt.



Ablauf von NAT



1. Der Client schickt seine Datenpakete mit der IP-Adresse 192.168.0.2 und dem TCP-Port 10101 an sein Standard-Gateway, bei dem es sich um einen NAT-Router handelt.
2. Der NAT-Router tauscht IP-Adresse (LAN-Adresse) und TCP-Port (LAN-Port) aus und speichert beides mit der getauschten Port-Nummer (WAN-Port) in der NAT-Tabelle.
3. Der Router leitet das Datenpaket mit der WAN-Adresse 220.0.0.1 und der neuen TCP-Port 20202 ins Internet weiter.
4. Der Empfänger (Server) verarbeitet das Datenpaket und schickt seine Antwort zurück.
5. Der NAT-Router stellt nun anhand der Port-Nummer 20202 (WAN-Port) fest, für welche IP-Adresse (LAN-Adresse) das Paket im lokalen Netz gedacht ist.
6. Er tauscht die IP-Adresse und die Port-Nummer wieder aus und leitet das Datenpaket ins lokale Netz weiter, wo es der Client entgegen nimmt.

Probleme

- Durch NAT können nur noch die, die über öffentliche IPv4-Adressen und in der Regel auch über das notwendige Kleingeld verfügen, Dienste im Internet anbieten.
- Die Einträge in der NAT-Tabelle des Routers sind nur für eine kurze Zeit gültig. Für eine Anwendung, die nur sehr unregelmäßig Daten austauscht, bedeutet das, dass ständig die Verbindung abgebrochen wird und dadurch die Erreichbarkeit eingeschränkt ist.

Vorteile

- Mehr Sicherheit - PCs im Netzwerk nicht direkt ansprechbar = mehr Privatsphäre
- Firewall - keine von außen eingehenden Verbindungen erlaubt, so fern keine Verbindung von intern aufgebaut wurde

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_03

Last update: **2019/05/16 18:20**

ICMP - Internet Control Message Protocol (Layer 3)

Das Internet Control Message Protocol (ICMP) ist Bestandteil des Internet Protocols (IP). Es wird aber als eigenständiges Protokoll behandelt, das zur Übermittlung von Meldungen über IP dient. Hauptaufgabe von ICMP ist die **Übertragung von Statusinformationen und Fehlermeldungen der Protokolle IP, TCP und UDP**. Die ICMP-Meldungen werden zwischen Rechnern und aktiven Netzknoten, z. B. Routern, benutzt, um sich gegenseitig Probleme mit Datenpaketen mitzuteilen. Ziel ist, die Übertragungsqualität zu verbessert.

Jedes Betriebssystem mit TCP/IP hat Tools, die ICMP nutzen. Zwei bekannte Tools sind Ping und Trace Route. Beides sind sehr einfache Programme, die zur Analyse von Netzwerk-Problemen gedacht sind und damit wesentlich zur Problemlösung beitragen können.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_04

Last update: **2019/05/16 18:20**

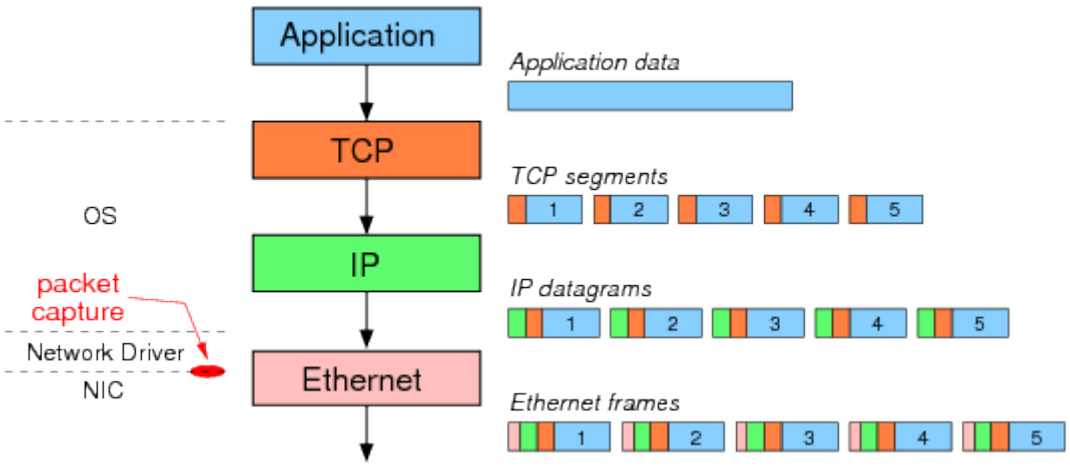
TCP - Transmission Control Protocol (Layer 4)

Das **Transmission Control Protocol** (=Übertragungssteuerungsprotokoll) ist ein Netzwerkprotokoll, das definiert, auf welche Art und Weise Daten zwischen Netzwerkkomponenten ausgetauscht werden sollen. Nahezu sämtliche aktuellen Betriebssysteme moderner Computer beherrschen TCP und nutzen es für den Datenaustausch mit anderen Rechnern. Das Protokoll ist ein zuverlässiges, verbindungsorientiertes, paketvermitteltes Transportprotokoll in Computernetzwerken. Es ist Teil der Internetprotokollfamilie, der Grundlage des Internets.

Aufgaben des TCP-Protokolls

Segmentierung (Data Segmenting)

Eine Funktion von TCP besteht darin, den von den Anwendungen kommenden **Datenstrom in Datenpakete bzw. Segmente aufzuteilen (Segmentierung)** und beim Empfang wieder zusammenzusetzen. Die Segmente werden mit einem **Header** versehen, in dem Steuer- und Kontroll-Informationen enthalten sind. Danach werden die **Segmente an das Internet Protocol (IP) übergeben**. Da beim IP-Routing die **Datenpakete unterschiedliche Wege** gehen können, entstehen unter Umständen **zeitliche Verzögerungen**, die dazu führen, dass die Datenpakete beim Empfänger in einer anderen Reihenfolge eingehen, als sie ursprünglich hatten. Deshalb werden die Segmente beim Empfänger auch wieder in die **richtige Reihenfolge** gebracht und erst dann an die adressierte Anwendung übergeben. Dazu werden die Segmente mit einer **fortlaufenden Sequenznummer** versehen (Sequenzierung).



Verbindungsmanagement (Connection Establishment and Termination)

Als **verbindungsorientiertes Protokoll** ist TCP für den **Verbindungsaufbau und Verbindungsabbau** zwischen zwei Stationen einer **Ende-zu-Ende-Kommunikation** zuständig. Durch TCP stehen Sender und Empfänger ständig in Kontakt zueinander (siehe TCP Kommunikation).

Fehlerbehandlung (Error Detection)

Obwohl es sich eher um eine virtuelle Verbindung handelt, werden während der Datenübertragung

ständig **Kontrollmeldungen** ausgetauscht. Der Empfänger bestätigt dem Sender jedes empfangene Datenpaket. Trifft keine Bestätigung beim Absender ein, wird das Paket noch mal verschickt. Da es bei Übertragungsproblemen zu doppelten Datenpaketen und Quittierungen kommen kann, werden alle TCP-Pakete und TCP-Meldungen mit einer **fortlaufenden Sequenznummer** gekennzeichnet. So sind Sender und Empfänger in der Lage, die Reihenfolge und Zuordnung der Datenpakete und Meldungen zu erkennen.

Flusssteuerung (Flow Control)

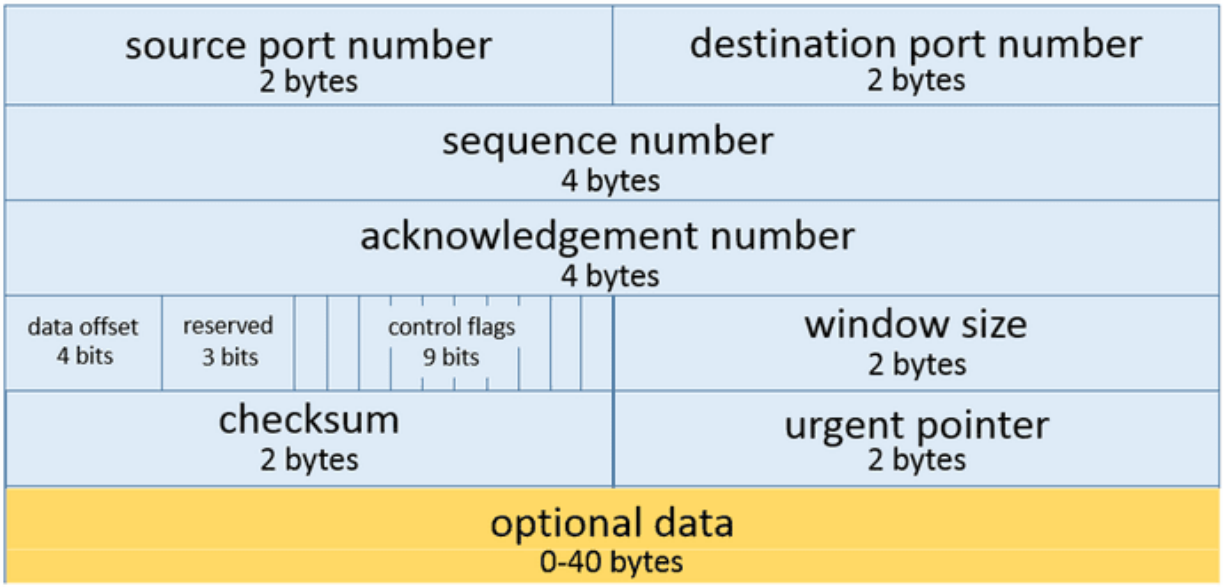
Bei einer paketorientierten Übertragung ohne feste zeitliche Zuordnung und ohne Kenntnis des Übertragungswegs erhält das Transport-Protokoll vom Übertragungssystem **keine Information über die verfügbare Bandbreite**. Mit der **Flusssteuerung** werden beliebig langsame oder schnelle **Übertragungsstrecken dynamisch auszulasten** und auch auf unerwartete Engpässe und Verzögerungen reagiert.

Anwendungsunterstützung (Application Support)

TCP- und UDP-Ports sind eine Software-Abstraktion, um Kommunikationsverbindungen voneinander unterscheiden zu können. Ähnlich wie IP-Adressen Rechner in Netzwerken adressieren, **adressieren Ports spezifische Anwendungen** oder Verbindungen, die auf einem Rechner laufen.

Aufbau eines TCP Headers

Aufbau des TCP-Headers TCP-Pakete setzen sich aus dem Header-Bereich und dem Daten-Bereich zusammen. Im Header sind alle Informationen enthalten, die für eine gesicherte TCP-Verbindung wichtig sind. Der TCP-Header ist in mehrere 32-Bit-Blöcke aufgeteilt. Mindestens enthält der Header 5 solcher Blöcke. Somit hat ein TCP-Header eine Länge von **mindestens 20 Byte**.

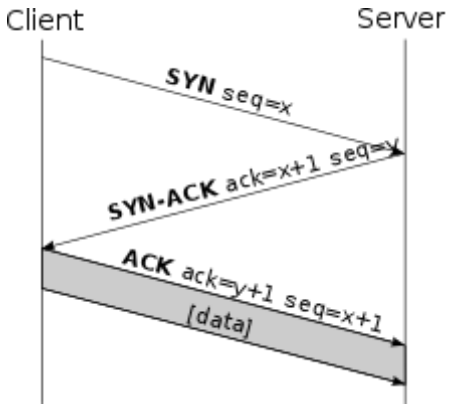


TCP Kommunikation

Verbindungsaufbau - 3-Way-Handshake

Der Client, der eine Verbindung aufbauen will, sendet dem Server ein **SYN-Paket** (von englisch synchronize) mit einer Sequenznummer x. Die Sequenznummern sind dabei für die Sicherstellung einer vollständigen Übertragung in der richtigen Reihenfolge und ohne Duplikate wichtig. Es handelt sich also um ein Paket, dessen SYN-Bit im Paketkopf gesetzt ist (siehe TCP-Header). Die Start-Sequenznummer ist eine beliebige Zahl, deren Generierung von der jeweiligen TCP-Implementierung abhängig ist. Sie sollte jedoch möglichst zufällig sein, um Sicherheitsrisiken zu vermeiden.

Der Server (siehe Abbildung) empfängt das Paket. Ist der Port geschlossen, antwortet er mit einem TCP-RST, um zu signalisieren, dass keine Verbindung aufgebaut werden kann. Ist der Port geöffnet, bestätigt er den Erhalt des ersten SYN-Pakets und stimmt dem Verbindungsaufbau zu, indem er ein **SYN/ACK-Paket** zurückschickt (ACK von engl. acknowledgement ‚Bestätigung'). Das gesetzte ACK-Flag im TCP-Header kennzeichnet diese Pakete, welche die Sequenznummer x+1 des SYN-Pakets im Header enthalten. Zusätzlich sendet er im Gegenzug seine Start-Sequenznummer y, die ebenfalls beliebig und unabhängig von der Start-Sequenznummer des Clients ist.



Der Client bestätigt zuletzt den Erhalt des SYN/ACK-Pakets durch das Senden eines eigenen **ACK-Paket**s mit der Sequenznummer x+1. Dieser Vorgang wird auch als **Forward Acknowledgement** bezeichnet. Aus Sicherheitsgründen sendet der Client den Wert y+1 (die Sequenznummer des Servers + 1) im ACK-Segment zurück. Die Verbindung ist damit aufgebaut. Im folgenden Beispiel wird der Vorgang abstrakt dargestellt:

```
1.  SYN-SENT            →    <SEQ=100><CTL=SYN>              →    SYN-RECEIVED
2.  SYN/ACK-RECEIVED    ←    <SEQ=300><ACK=101><CTL=SYN,ACK>    ←
SYN/ACK-SENT
3.  ACK-SENT            →    <SEQ=101><ACK=301><CTL=ACK>    →    ESTABLISHED
```



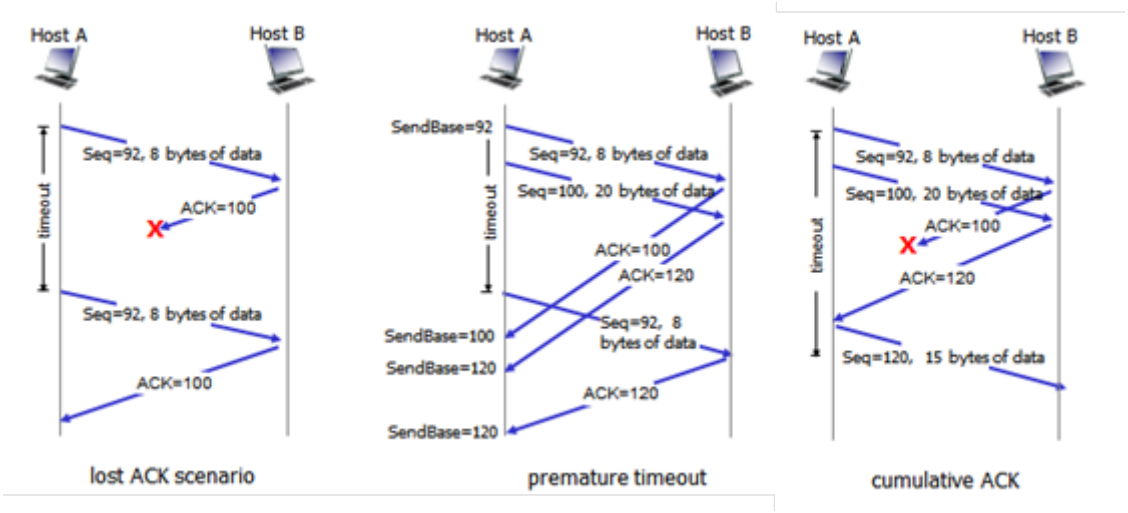
Einmal aufgebaut, ist die Verbindung für beide Kommunikationspartner gleichberechtigt, man kann einer bestehenden Verbindung auf TCP-Ebene nicht ansehen, wer der Server und wer der Client ist. Daher hat eine Unterscheidung dieser beiden Rollen in der weiteren Betrachtung keine Bedeutung mehr.

Aufzeichnung mit Wireshark

Datenaustausch

Der Sender beginnt mit dem Senden des ersten Datenpakets (Send Paket 1). Der Empfänger nimmt

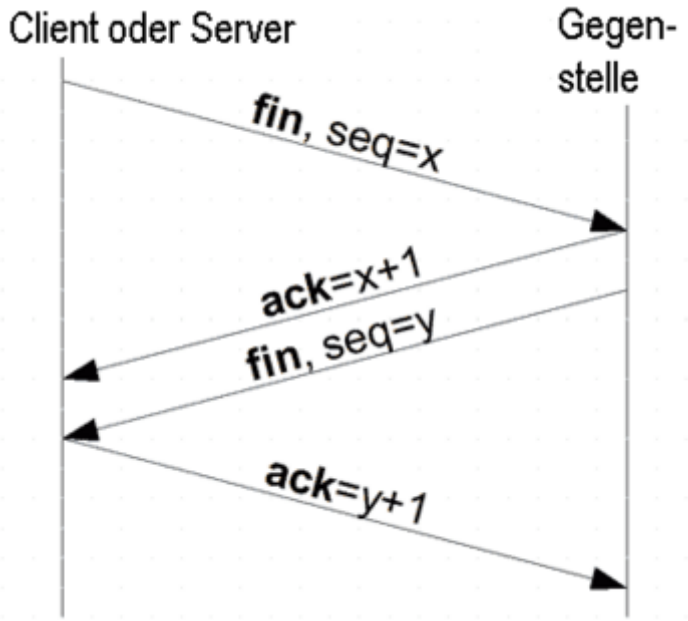
das Paket entgegen (Receive Paket 1) und bestätigt den Empfang (Send ACK Paket 1). Der Sender nimmt die Bestätigung entgegen (Receive ACK Paket 1) und sendet das zweite Datenpaket (Send Paket 2). Der Empfänger nimmt das zweite Paket entgegen (Receive Paket 2) und bestätigt den Empfang (Send ACK Paket 2). Der Sender nimmt die zweite Bestätigung entgegen (Receive ACK Paket 2). Und so läuft der Datenaustausch weiter, bis alle Pakete übertragen wurden.



Aufzeichnung mit Wireshark

Verbindungsabbau

Der Verbindungsabbau kann sowohl vom Client als auch vom Server vorgenommen werden. Zuerst schickt einer der beiden der Gegenstelle einen Verbindungsabbauwunsch (**FIN**). Die Gegenstelle bestätigt den Erhalt der Nachricht (ACK) und schickt gleich darauf ebenfalls einen Verbindungsabbauwunsch (**FIN**). Danach bekommt die Gegenstelle noch mitgeteilt, dass die Verbindung abgebaut ist (**ACK**).



Aufzeichnung mit Wireshark

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_05

Last update: **2019/05/16 18:20**

UDP - User Datagram Protocol (Layer 4)

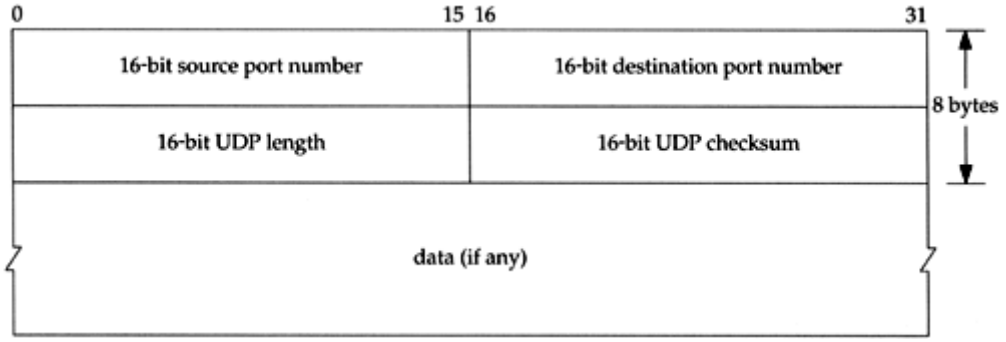
UDP ist ein **verbindungsloses Transport-Protokoll** und arbeitet auf der Schicht 4, der **Transportschicht**, des OSI-Schichtenmodells. Es hat damit eine vergleichbare Aufgabe, wie das verbindungsorientierte TCP. Allerdings arbeitet es **verbindungslos und damit unsicher**. Das bedeutet, der **Absender weiß nicht**, ob seine verschickten **Datenpakete angekommen sind**. Während TCP Bestätigungen beim Datenempfang sendet, verzichtet UDP darauf. Das hat den Vorteil, dass der Paket-Header viel kleiner ist und die Übertragungsstrecke keine Bestätigungen übertragen muss. Typischerweise wird UDP bei DNS-Anfragen, VPN-Verbindungen, Audio- und Video-Streaming verwendet.

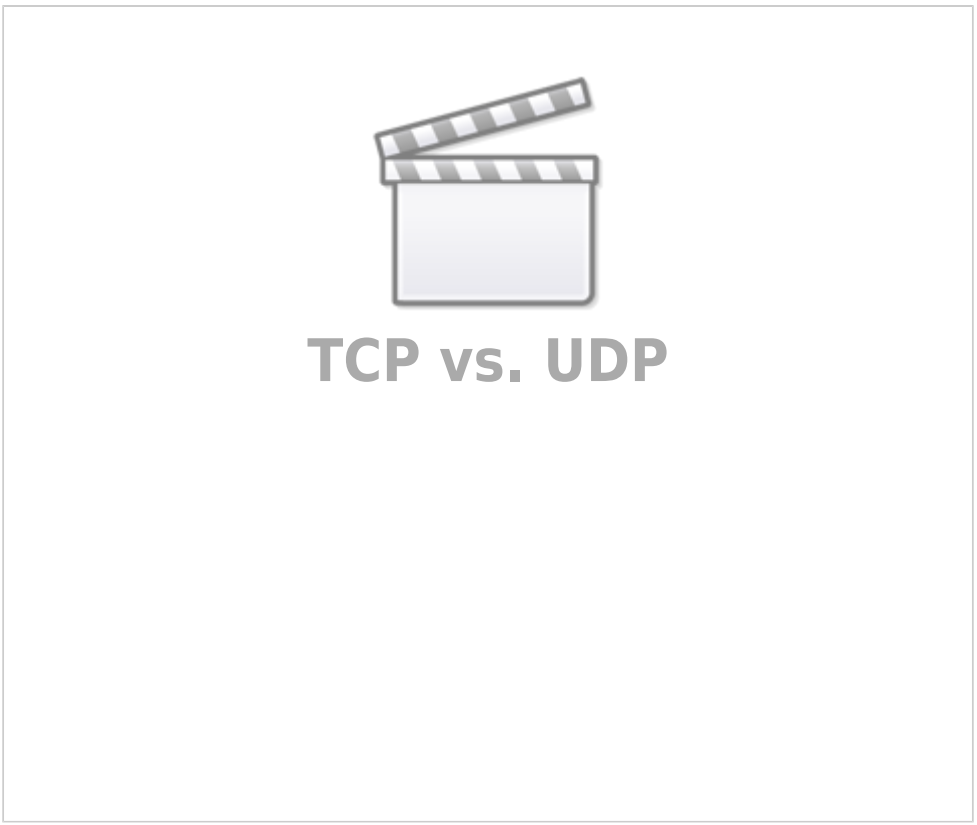
Funktionsweise

UDP hat die selbe Aufgabe wie TCP, nur das nahezu alle Kontrollfunktionen fehlen, dadurch schlanker und einfacher zu verarbeiten ist. So besitzt UDP keinerlei Methoden die sicherstellen, dass ein Datenpaket beim Empfänger ankommt. Ebenso entfällt die Nummerierung der Datenpakete. UDP ist nicht in der Lage die Datenpakete in der richtigen Reihenfolge zusammenzusetzen. Statt dessen werden die UDP-Pakete direkt an die Anwendung weitergeleitet. Für eine sichere Datenübertragung ist deshalb die Anwendung zuständig. In der Regel wird UDP für Anwendungen und Dienste verwendet, die mit Paketverlusten umgehen können oder sich selber um das Verbindungsmanagement kümmern. UDP eignet sich auch für Anwendungen, die nur einzelne, nicht zusammenhängende Datenpakete transportieren müssen.

UDP Header

UDP-Pakete setzen sich aus dem Header-Bereich und dem Daten-Bereich zusammen. Im Header sind alle Informationen enthalten, die eine einigermaßen geordnete Datenübertragung zulässt und die ein UDP-Paket als ein solches erkennen lassen. Der UDP-Header ist in 32-Bit-Blöcke unterteilt. Er besteht aus zwei solcher Blöcke, die den Quell- und Ziel-Port, die Länge des gesamten UDP-Pakets und die Check-Summe enthalten. Der UDP-Header ist mit insgesamt 8 Byte sehr schlank und lässt sich mit wenig Rechenleistung verarbeiten.





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_06

Last update: **2019/05/16 18:20**

DHCP - Dynamic Host Configuration Protocol (UDP, Ports 67 + 68, Layer 7)

DHCP ist ein Protokoll, um **IP-Adressen** in einem TCP/IP-Netzwerk **zu verwalten** und an die anfragenden Hosts zu verteilen. Mit DHCP ist jeder **Netzwerk-Teilnehmer** in der Lage sich selber **automatisch zu konfigurieren**.

Warum DHCP?

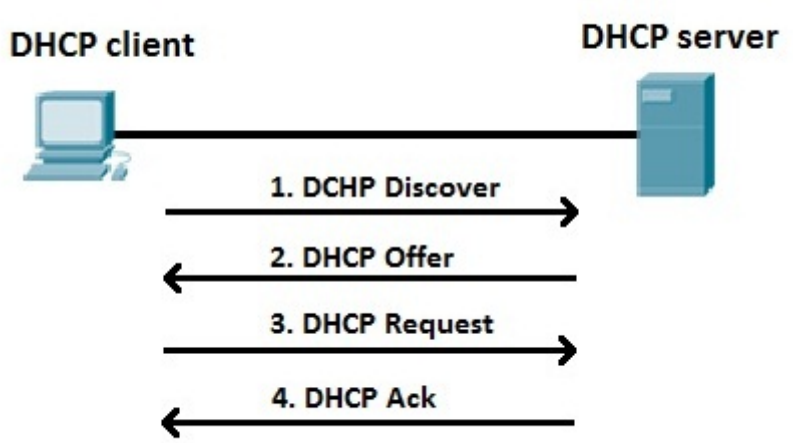
Um ein Netzwerk per TCP/IP aufzubauen ist es notwendig an jedem Host eine IP-Konfiguration vorzunehmen. Für ein TCP/IP-Netzwerk müssen folgende Einstellungen an jedem Host vorgenommen werden:

- Vergabe einer eindeutigen IP-Adresse
- Zuweisen einer Subnetzmaske (Subnetmask)
- Zuweisen des zuständigen Default- bzw. Standard-Gateways
- Zuweisen des zuständigen DNS-Servers

In den ersten IP-Netzen wurden IP-Adressen noch von Hand **aufwendig vergeben** und **fest** in die Systeme **eingetragen**. Die dazu **erforderliche Dokumentation** war jedoch nicht immer fehlerfrei und schon gar nicht aktuell und vollständig. Der Ruf nach einer einfachen und automatischen Adressverwaltung wurde deshalb besonders bei Betreibern großer Netze laut. Hier war durch die manuelle Verwaltung und Konfiguration sehr **viel Planungs- und Arbeitszeit** notwendig. Um für die Betreiber der immer größer werdenden Netze eine Erleichterung zu verschaffen wurde DHCP entwickelt. Mit DHCP kann jede IP-Host die IP-Adresskonfiguration von einem DHCP-Server anfordern und sich selber automatisch konfigurieren. So müssen IP-Adressen nicht mehr manuell verwaltet und zugewiesen werden.

Funktionsweise

1. Der DHCP-Client **schickt an alle Rechner** im Netzwerk (=Broadcast) eine **DHCP-Server-Suchanfrage (DHCP-Discover)**. Bis auf den DHCP-Server verwerfen alle Rechner das Datenpaket.
2. Nur der **DHCP-Server** empfängt den DHCP-Discover und bietet dem Client mittels Broadcast eine **freie IP-Adresse an (DHCP-Offer)**
3. Nur der **DHCP-Client** empfängt den DHCP-Offer und antwortet mit einer eine **DHCP-Anfrage (DHCP-Request)**. Alle anderen Clients verwerfen wiederum das DHCP-Angebot.
4. Der **DHCP-Server** empfängt den DHCP-Request und **bestätigt** dem DHCP-Client die angefragte IP-Adresse **mittels DHCP-ACK**.





From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_07

Last update: **2019/05/16 18:20**

DNS - Domain Name System (UDP, Port 53 , Layer 7)

Das Domain Name System, kurz DNS, wird auch als **Telefonbuch des Internets** bezeichnet. Ähnlich wie man in einem Telefonverzeichnis nach einem Namen sucht, um die Telefonnummer heraus zu bekommen, schaut man im DNS nach einem Computernamen, um die dazugehörige IP-Adresse zu bekommen. Die IP-Adresse wird benötigt, um eine Verbindung zu einem Server aufbauen zu können, über den nur der Computername bekannt ist.

Das Domain Name System ist ein System zur Auflösung von Computernamen in IP-Adressen und umgekehrt. DNS kennt keine zentrale Datenbank. Die Informationen sind auf vielen tausend Nameservern (DNS-Server) verteilt. Möchte man zum Beispiel die Webseite www.orf.at besuchen, dann fragt der Browser einen DNS-Server, der in der IP-Konfiguration hinterlegt ist. Das ist in der Regel der Router des Internet-Zugangs. Je nach dem, ob die DNS-Anfrage beantwortet werden kann oder nicht, wird eine Kette weiterer DNS-Server befragt, bis die Anfrage positiv beantwortet und eine IP-Adresse an den Browser zurück geliefert werden kann.

Wenn ein Computername oder Domain-Name nicht aufgelöst werden kann, dann kann auch keine Verbindung zu dem betreffenden Host aufgebaut werden. Es sei denn, der Nutzer verfügt über das Wissen der IP-Adresse. Das bedeutet, ohne DNS ist die Kommunikation im Netzwerk und im Internet praktisch nicht möglich. Deshalb existieren viele tausend DNS-Server auf der ganzen Welt, die zusätzlich hierarchisch angeordnet sind und sich gegenseitig über Änderungen informieren.

Top-Level-Domain & Second-Level-Domain

Domain-Namen sind hierarchisch von rechts nach links gegliedert. Der ganz rechte Abschnitt nach dem letzten Punkt heißt Top-Level-Domain (TLD), der davor Second-Level-Domain (SLD) oder einfach „Domain“. Alle weiteren Namensteile links davon sind jeweils Sub- bzw. Third-Level-Domains (Fourth Level, Fifth Level, Sixth Level usw.). Ein Beispiel verdeutlicht die Begrifflichkeiten: Der Name „www.example.com“ besteht aus drei Ebenen:

- „.com“: die erste Ebene, auch Top-Level-Domain oder Domain-Endung genannt
- „example“: die zweite Ebene, auch als Second-Level-Domain oder Domain bezeichnet
- „www“: die dritte Ebene, auch Sub- oder Third-Level-Domain genannt

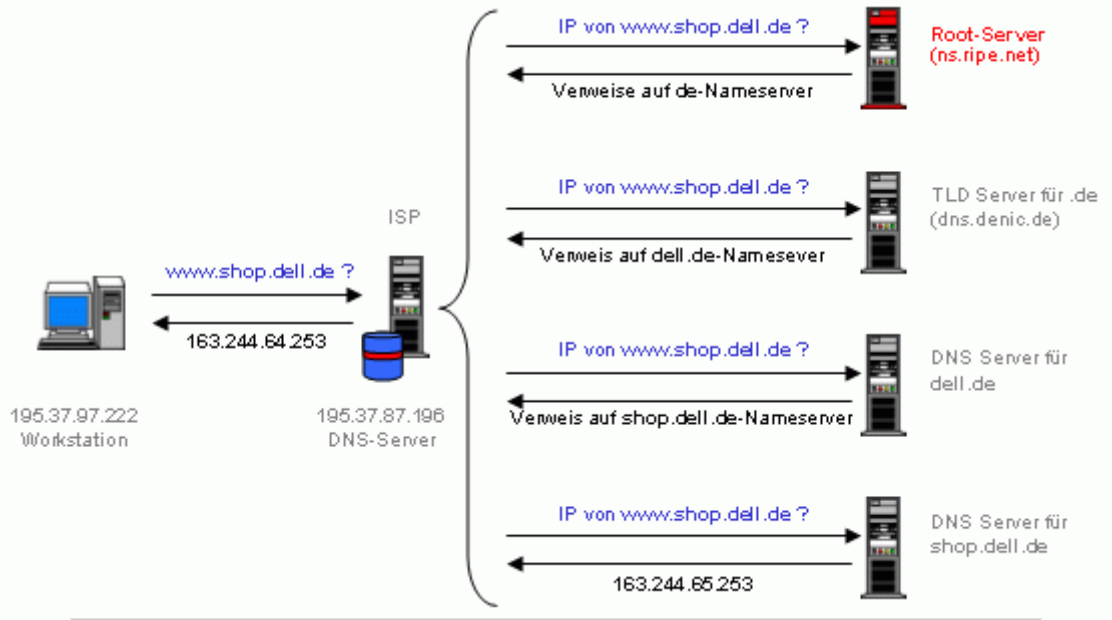
FQDN (Fully Qualified Domain Name)

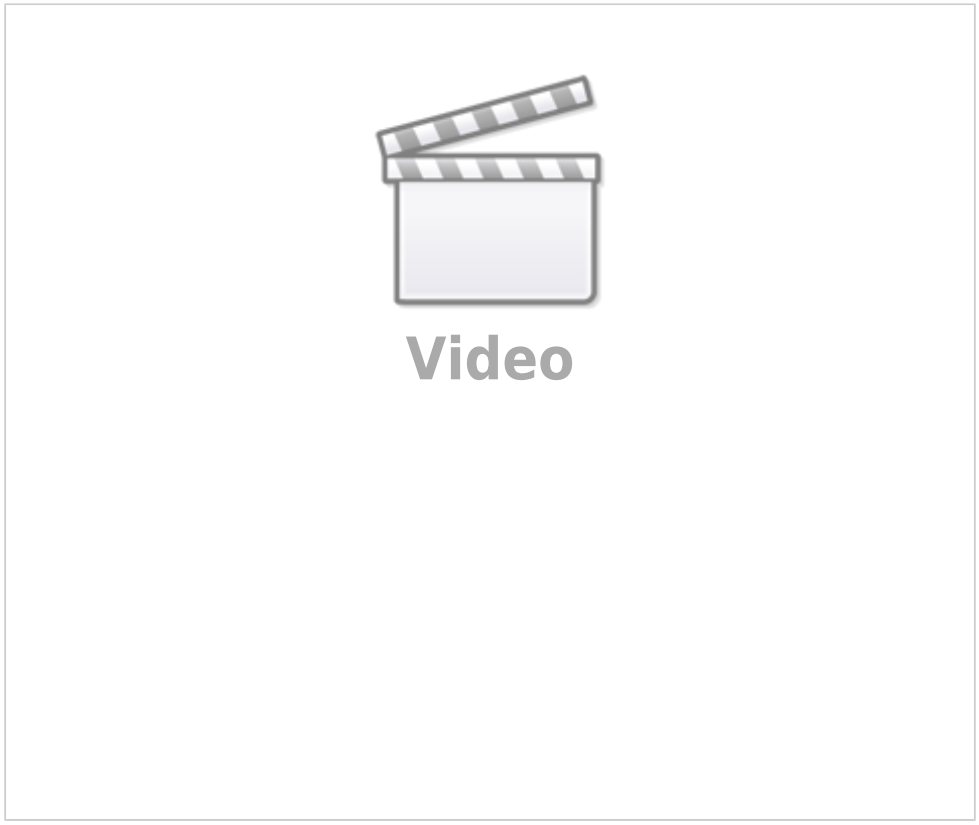
Der vollständige Name einer Domain wird als ihr Fully Qualified Domain Name (FQDN) bezeichnet. Der Domain-Name ist in diesem Fall eine absolute Adresse.

Der FQDN www.example.com. ergibt sich durch:

```
3rd-level-label. 2nd-level-label. Top-Level-Domain. root-label
```

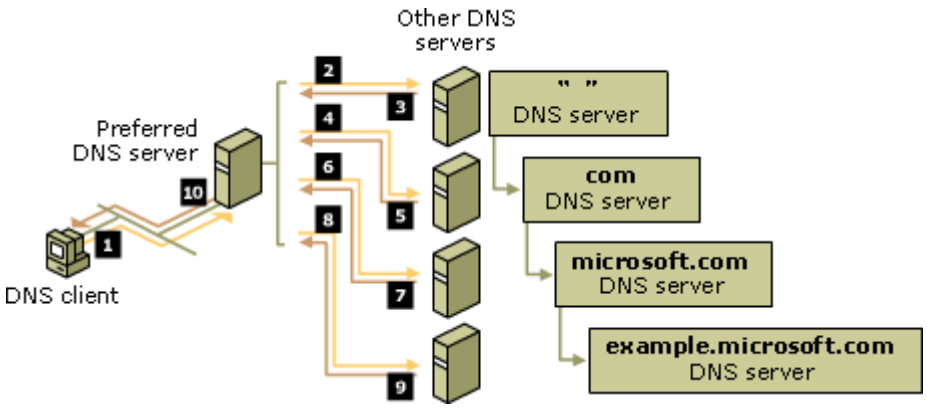
Da das Root-Label immer leer ist (es besteht aus einer leeren Zeichenkette), wird bei den meisten Benutzer-Anwendungen (zum Beispiel Browsern) in der Regel auf die Eingabe des Punktes zwischen dem Label der Top Level Domain und dem root-label verzichtet. Streng genommen handelt es sich bei dieser Schreibweise nicht mehr um eine absolute, sondern um eine relative Adresse und damit nicht mehr um einen FQDN.



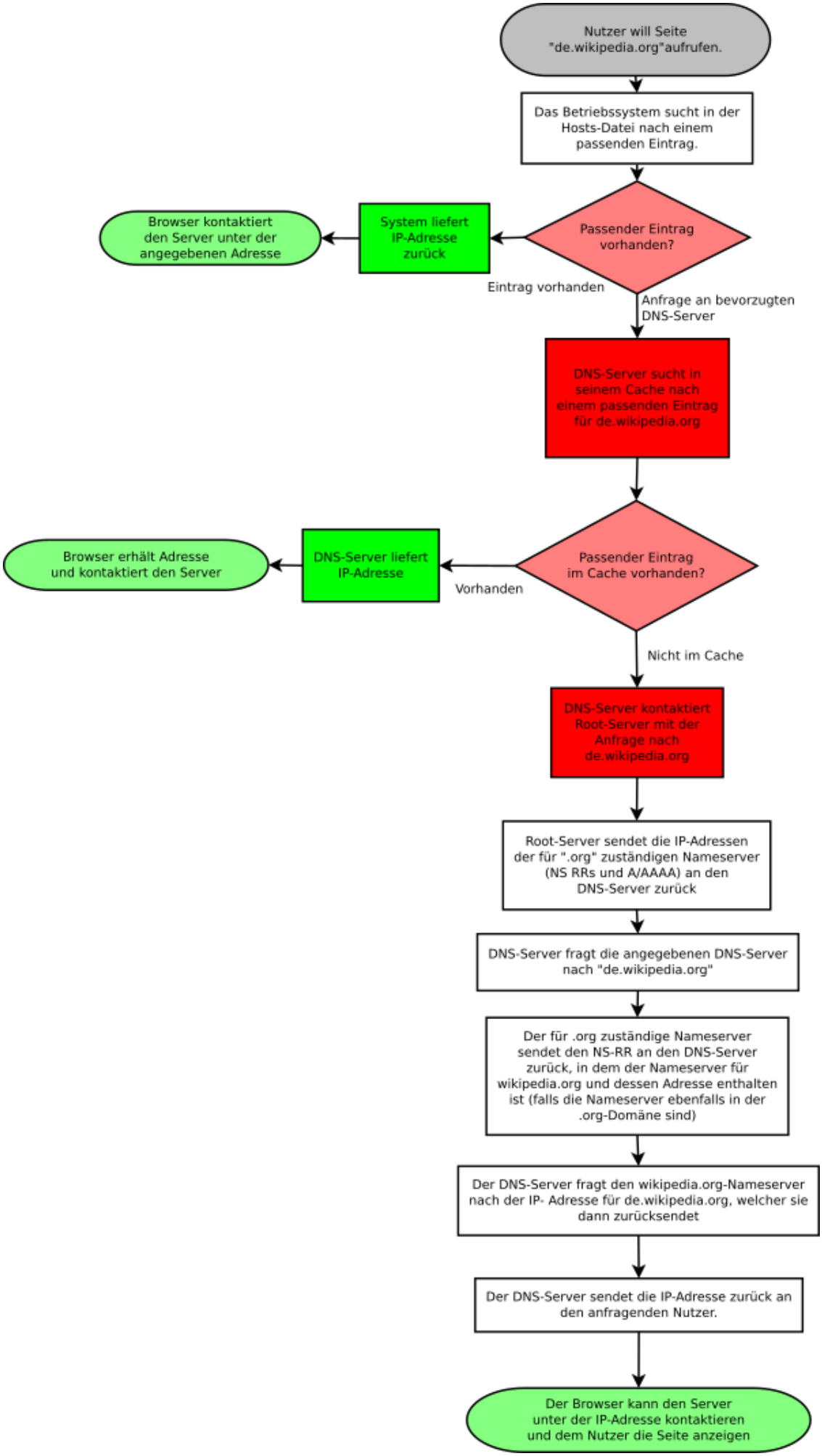


Rekursive Namensauflösung - Rekursive DNS-Serverstruktur

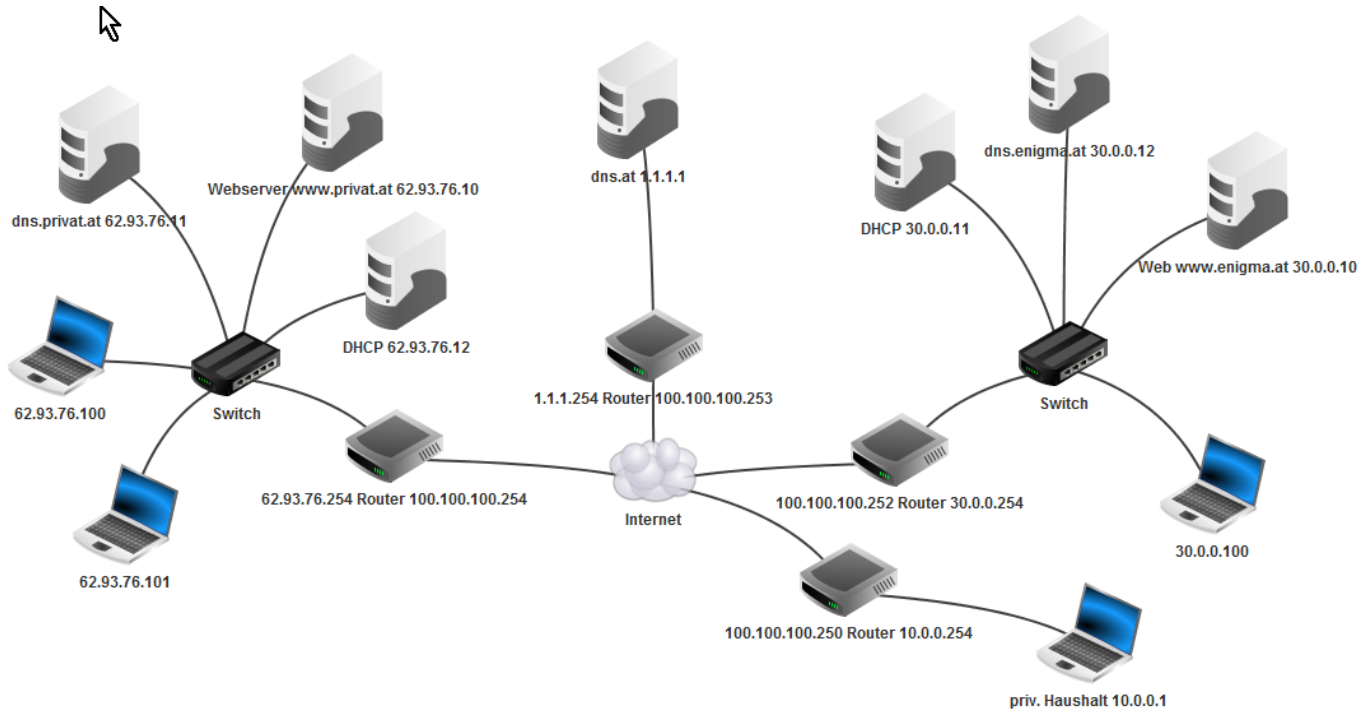
- Einzelne DNS-Server haben nicht jeden Domainnamen gespeichert. Informationen werden hierarchisch von übergeordneten DNS-Servern abgefragt. Die folgenden Grafiken sollen den Prozess veranschaulichen:



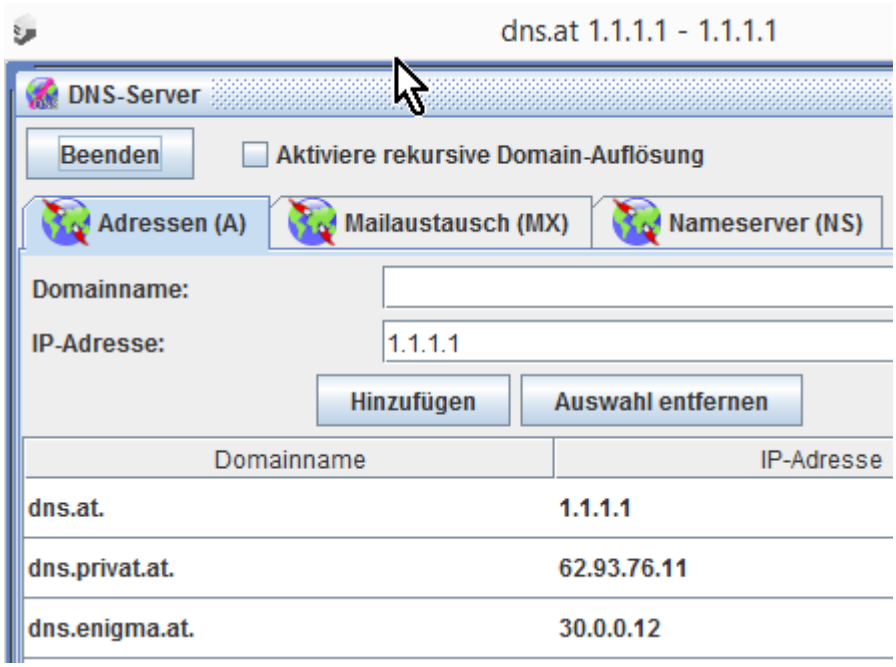
- Hier eine genauere Abfolge der Anfragen:

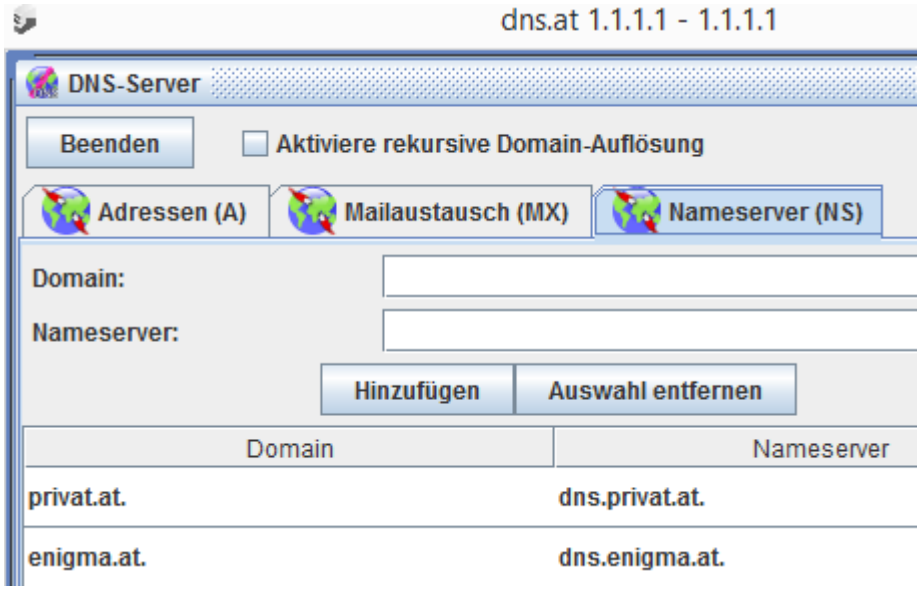

Nutzer will Seite "de.wikipedia.org"aufrufen.
Das Betriebssystem sucht in der Hosts-Datei nach einem passenden Eintrag.
Passender Eintrag vorhanden?
System liefert IP-Adresse zurück
Browser kontaktiert den Server unter der angegebenen Adresse
Eintrag vorhanden
Anfrage an bevorzugten DNS-Server
DNS-Server sucht in seinem Cache nach einem passenden Eintrag für de.wikipedia.org
Passender Eintrag im Cache vorhanden?
DNS-Server liefert IP-Adresse
Browser erhält Adresse und kontaktiert den Server
Vorhanden
Nicht im Cache
DNS-Server kontaktiert Root-Server mit der Anfrage nach de.wikipedia.org
Root-Server sendet die IP-Adressen der für ".org" zuständigen Nameserver (NS RRs und A/AAAA) an den DNS-Server zurück
DNS-Server fragt die angegebenen DNS-Server nach "de.wikipedia.org"
Der für .org zuständige Nameserver sendet den NS-RR an den DNS-Server zurück, in dem der Nameserver für wikipedia.org und dessen Adresse enthalten ist (falls die Nameserver ebenfalls in der .org-Domäne sind)
Der DNS-Server fragt den wikipedia.org-Nameserver nach der IP- Adresse für de.wikipedia.org, welcher sie dann zurücksendet
Der DNS-Server sendet die IP-Adresse zurück an den anfragenden Nutzer.
Der Browser kann den Server unter der IP-Adresse kontaktieren und dem Nutzer die Seite anzeigen

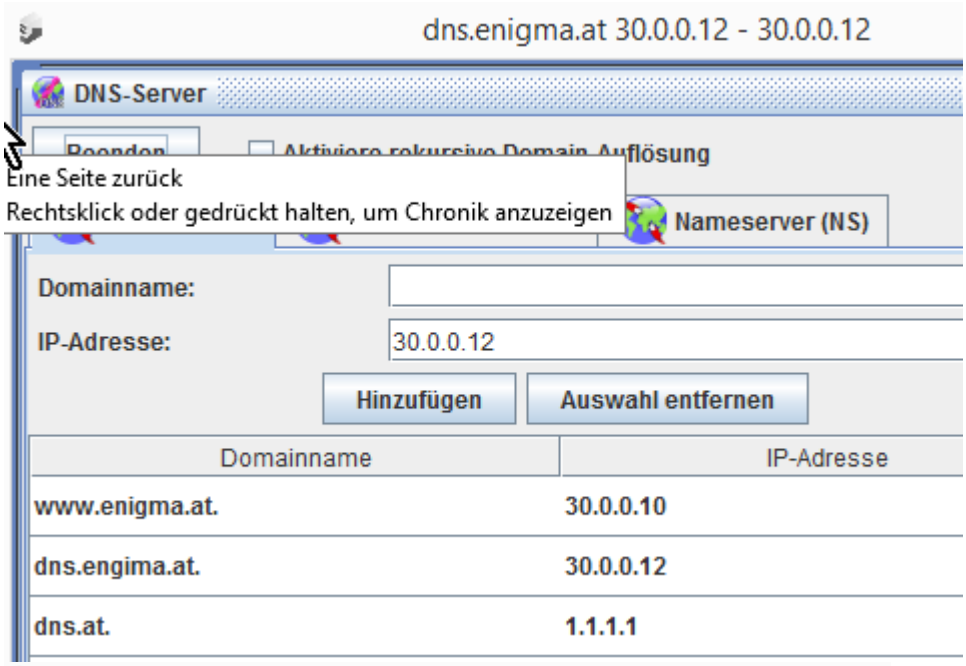
Mittels Filius kann eine hierarchische DNS-Serverstruktur aufgebaut werden:

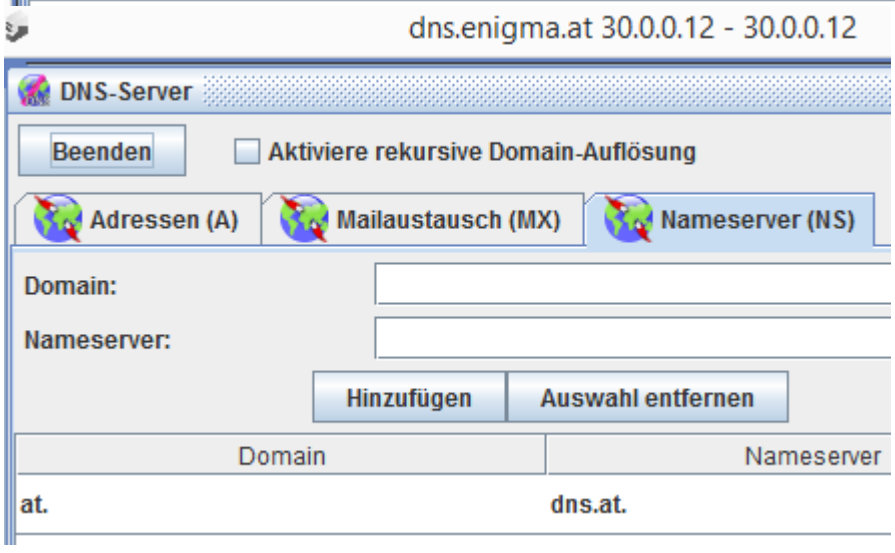


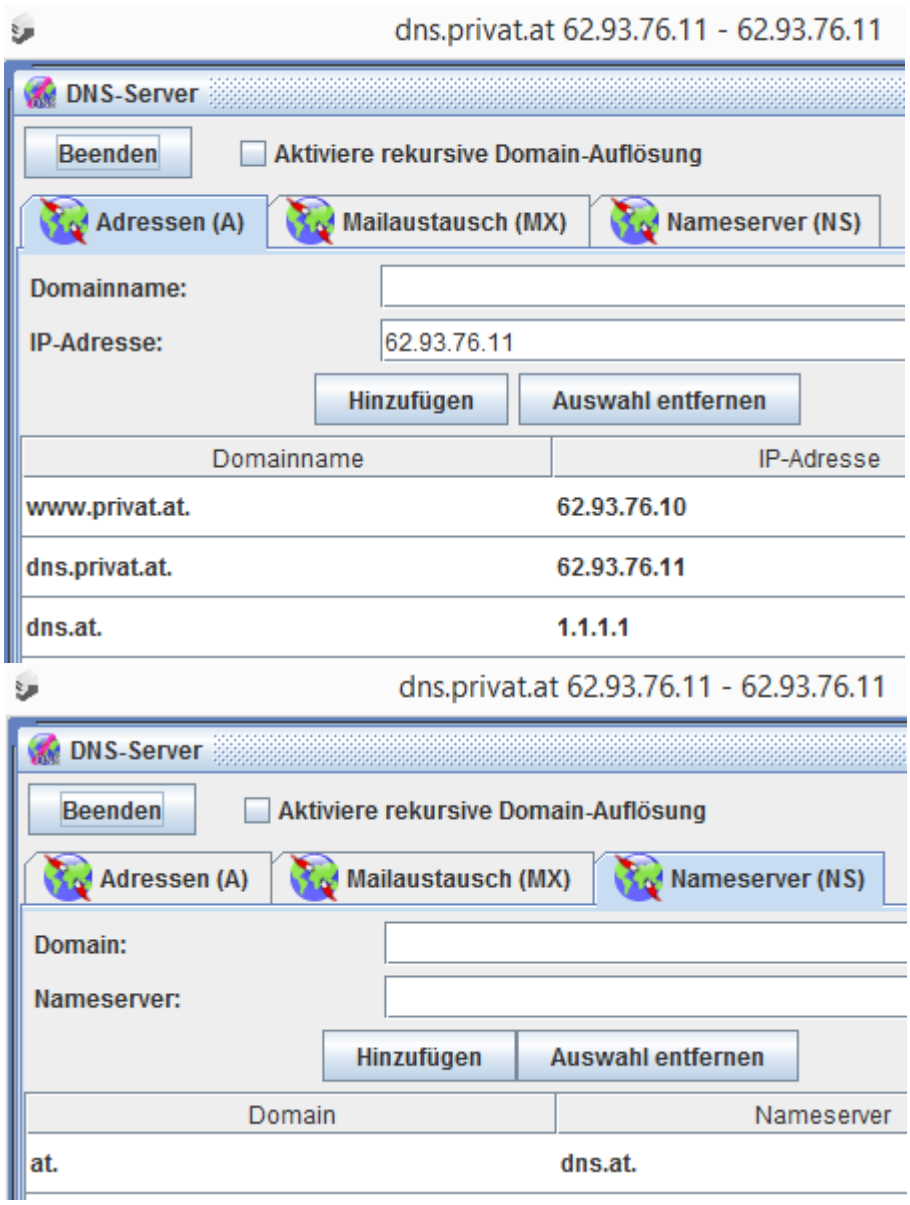
Hier die Screenshots über die entsprechenden Eintragungen in den Nameservern.




dns.at 1.1.1.1 - 1.1.1.1
DNS-Server
Beenden
Aktiviere rekursive Domain-Auflösung
Adressen (A)
Mailaustausch (MX)
Nameserver (NS)
Domain:
Nameserver:
Hinzufügen
Auswahl entfernen
Domain
Nameserver
privat.at.
dns.privat.at.
enigma.at.
dns.enigma.at.


dns.enigma.at 30.0.0.12 - 30.0.0.12
DNS-Server
Eine Seite zurück
Rechtsklick oder gedrückt halten, um Chronik anzuzeigen
Nameserver (NS)
Domainname:
IP-Adresse:
30.0.0.12
Hinzufügen
Auswahl entfernen
Domainname
IP-Adresse
www.enigma.at.
30.0.0.10
dns.engima.at.
30.0.0.12
dns.at.
1.1.1.1


dns.enigma.at 30.0.0.12 - 30.0.0.12
DNS-Server
Beenden
Aktiviere rekursive Domain-Auflösung
Adressen (A)
Mailaustausch (MX)
Nameserver (NS)
Domain:
Nameserver:
Hinzufügen
Auswahl entfernen
Domain
Nameserver
at.
dns.at.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

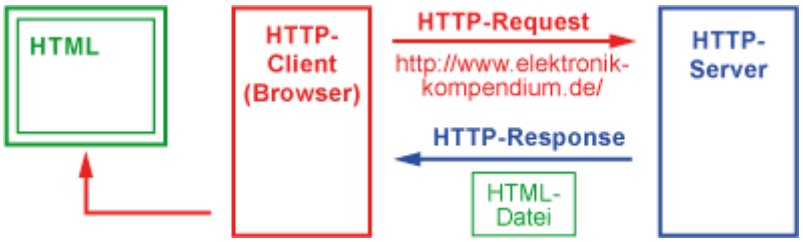
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_08

Last update: **2019/05/16 18:20**

Hypertext Transfer Protocol (HTTP) (Port 80 - Layer 7)

HTTP ist das Kommunikationsprotokoll im World Wide Web (WWW). Die wichtigsten Funktionen sind Dateien vom Webserver anzufordern und in den Browser zu laden. Der Browser übernimmt dann die Darstellung von Texten und Bilder und kümmert sich um das Abspielen von Audio- und Video-Daten.

Funktionsweise



Die Kommunikation findet nach dem **Client-Server-Prinzip** statt. Der **HTTP-Client (Browser)** sendet seine **Anfrage (HTTP-Request)** an den **HTTP-Server (Webserver/Web-Server)**. Dieser bearbeitet die Anfrage und schickt seine **Antwort (HTTP-Response)** zurück. Nach der Antwort durch den Server ist diese Verbindung beendet. Typischerweise finden gleichzeitig mehrere HTTP-Verbindungen statt.

HTTP Adressierung

Damit der Server weiß, was er dem HTTP-Client schicken soll, adressiert der HTTP-Client eine Datei, die sich auf dem HTTP-Server befinden muss. Dazu wird vom HTTP-Client ein **URL (Uniform Resource Locator)** im HTTP-Header an den HTTP-Server übermittelt:

```
http://Servername.Domainname.Top-Level-Domain:TCP-Port/Pfad/Datei

z. B.
http://www.elektronik-kompendium.de:80/sites/kom/0902231.html
```

HTTP Request

Der HTTP-Request ist die Anfrage des HTTP-Clients an den HTTP-Server. Ein HTTP-Request besteht aus den Angaben **Methode, URL und dem Request-Header**. Die häufigste Methoden sind **GET und POST**. Dahinter folgt durch ein Leerzeichen getrennt der URL und die verwendete HTTP-Version. In weiteren Zeilen folgt der Header und bei der Methode POST durch eine Leerzeile (!) getrennt die Formular-Daten.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_09

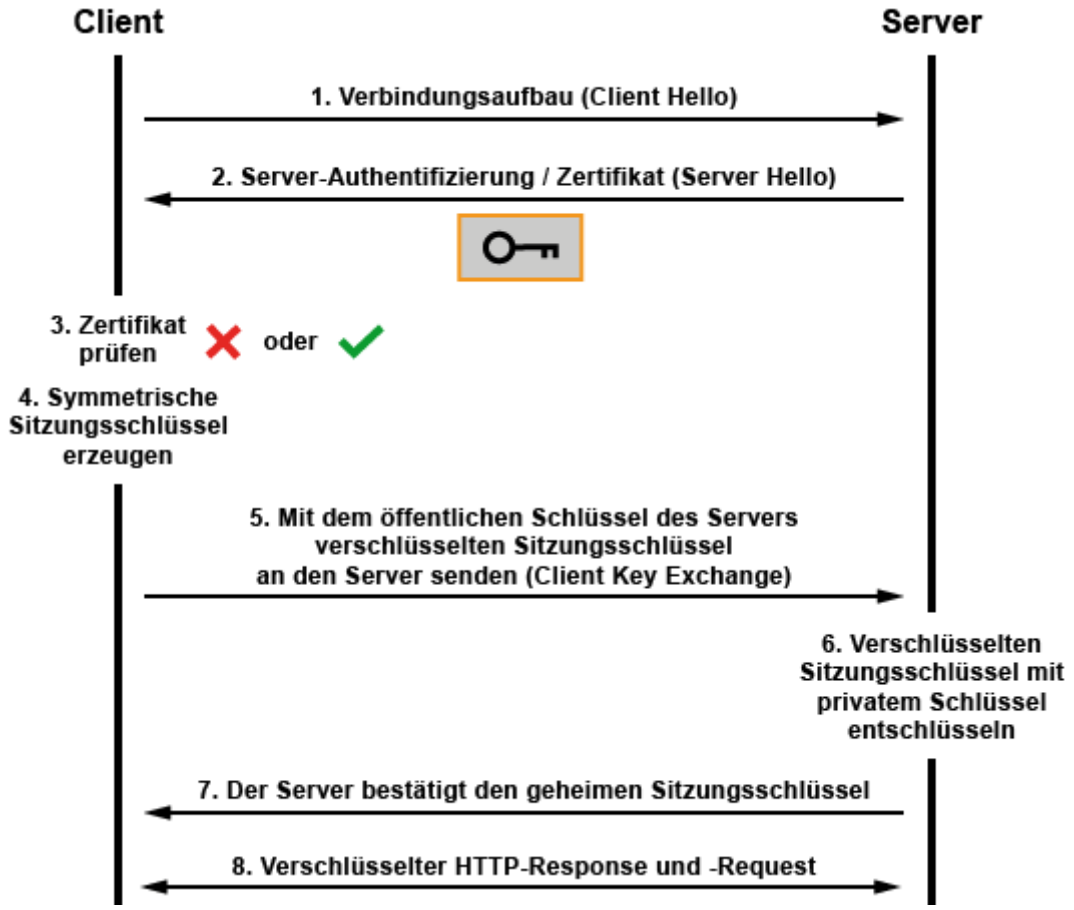
Last update: **2019/05/16 18:20**

HyperText Transfer Protocol Secure (HTTPS) (Port 443 - Layer 7)

HTTPS bzw. HTTP Secure ist die Anwendung von HTTP in **Verbindung mit Verschlüsselung und Authentifizierung**. Wobei in der Regel nur der **angefragte Webserver sich mit einem Zertifikat authentisieren** muss.

Eine **verschlüsselte Verbindung mit einem Browser** signalisiert man mit einem „https:“ *(TCP-Port 443) statt „http:*“ (TCP-Port 80). Dabei muss sich der Webserver dem Client gegenüber authentisieren, ob er tatsächlich der Webserver ist, der sich unter der eingegebenen Adresse befindet. Zusätzlich wird die **Verbindung bzw. Sitzung Ende-zu-Ende-verschlüsselt**. Das bedeutet, die **Stationen zwischen Client und Server** können die Kommunikation **nicht entschlüsseln**.

Für die Authentifizierung und Verschlüsselung ist SSL/TLS verantwortlich. Es schiebt sich zwischen HTTP und dem Transportprotokoll TCP. Damit steht SSL/TLS auch für andere Anwendungsprotokolle zur Verfügung. Beispielsweise SMTPS, IMAPS und FTPS. SSL arbeitet für den Anwender nahezu unsichtbar.



1. Client Hello: Der Client kontaktiert den Server über ein Protokoll mit Verschlüsselungsoptionen.
2. Server Hello, Certificate, Server Key Exchange, Server Hello Done: Der Server nimmt die Verbindung an und schickt sein Zertifikat mit dem öffentlichen Schlüssel seines Schlüsselpaares zur Authentifizierung an den Client.
3. Der Client überprüft das Server-Zertifikat und dessen Gültigkeit (Validierung). Erkennt der Client das Zertifikat als ungültig wird die Verbindung an dieser Stelle abgebrochen.
4. Erkennt der Client das Zertifikat als gültig erzeugt der Client den symmetrischen Sitzungsschlüssel.

5. Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message: Mit dem öffentlichen Schlüssel des Servers verschlüsselt der Client den Sitzungsschlüssel und schickt ihn an den Server.
6. Mit seinem privaten Schlüssel kann der Server den verschlüsselten Sitzungsschlüssel entschlüsseln.
7. Encrypted Handshake Message, Change Cipher Spec, Encrypted Handshake Message: Der Server bestätigt den geheimen Sitzungsschlüssel.
8. Danach werden alle HTTP-Requests und -Responses verschlüsselt, bis die Verbindung abgebaut wird.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

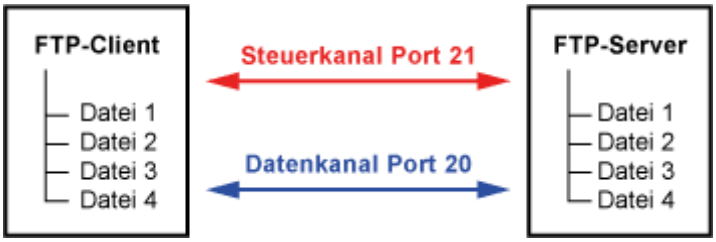
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_10

Last update: **2019/05/16 18:20**

File Transfer Protocol (FTP) (Port 20 und 21 - Layer 7)

FTP ist ein Kommunikationsprotokoll, um Dateien zwischen unterschiedlichen Computersystemen zu übertragen. Die Übertragung findet nach dem Client-Server-Prinzip statt. Ein FTP-Server stellt dem FTP-Client Dateien zur Verfügung. Der FTP-Client kann Dateien auf dem FTP-Server ablegen, löschen oder herunterladen. Mit einem komfortablen FTP-Client arbeitet man ähnlich, wie mit einem Dateimanager.

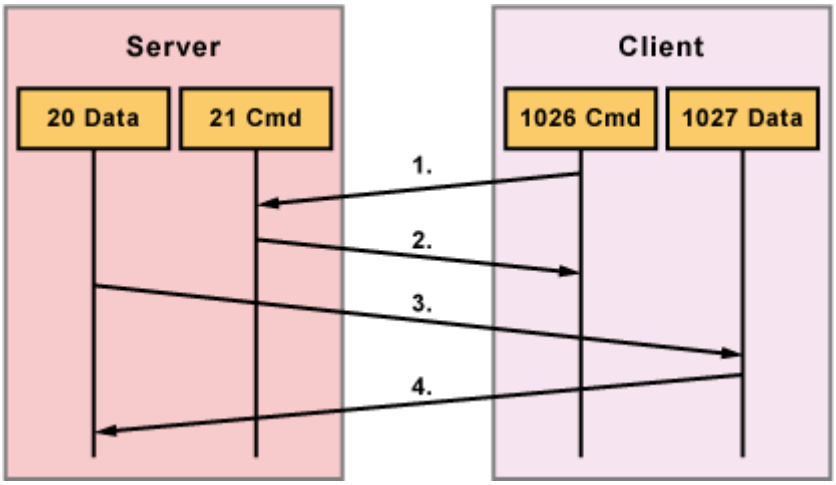
Funktionsweise

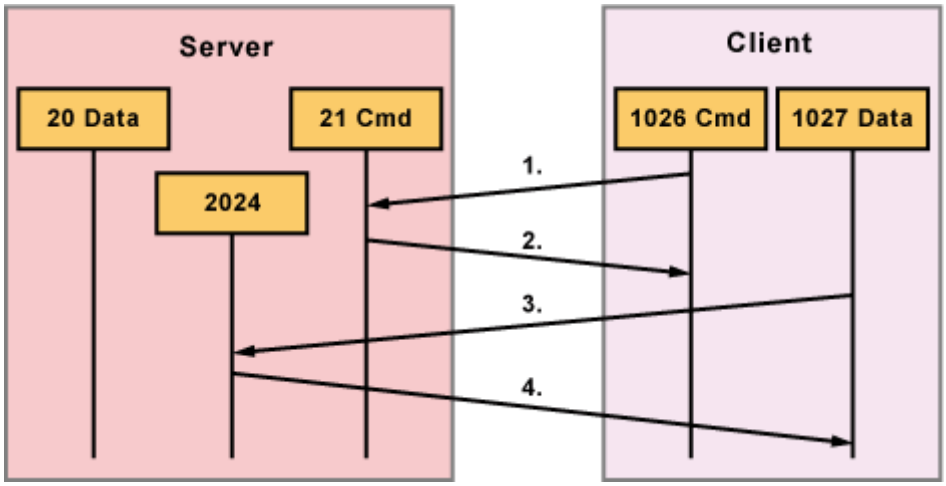


Die Kommunikation findet nach dem Client-Server-Prinzip statt. Wobei FTP zwischen Client und Server zwei logische Verbindungen herstellt. Eine Verbindung ist der Steuerkanal (command channel) über den TCP-Port 21. Dieser Kanal dient ausschließlich zur Übertragung von FTP-Kommandos und deren Antworten. Die zweite Verbindung ist der Datenkanal (data channel) über den TCP-Port 20. Dieser Kanal dient ausschließlich zur Übertragung von Daten. Über den Steuerkanal tauschen Client und Server Kommandos aus, die eine Datenübertragung über den Datenkanal einleiten und beenden.

Aktives vs. Passives FTP

Der FTP-Verbindungsaufbau sieht vor, dass der Steuerkanal vom FTP-Client zum FTP-Server aufgebaut wird. Steht der Steuerkanal wird der Datenkanal vom FTP-Server zum FTP-Client initiiert (aktives FTP). Befindet sich der FTP-Client hinter einem NAT-Router oder einer Firewall und verfügt parallel dazu nur über eine private IPv4-Adresse, dann kommt die Verbindung nicht zustande. Die Verbindungsanforderung vom Server an den Client wird von der Firewall bzw. dem Router abgeblockt, bzw. kann wegen der privaten IPv4-Adresse gar nicht geroutet werden. Für diesen Fall gibt es das passive FTP, bei dem auch der Client den Datenkanal initiiert.





Am Anfang jeder FTP-Verbindung steht die Authentifizierung des Benutzers. Danach erfolgt der Aufbau des Steuerkanals über Port 21 und des Datenkanals über Port 20. Wenn die Dateiübertragungen abgeschlossen sind, werden die Verbindungen vom Benutzer oder vom Server (Timeout) beendet.



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_11

Last update: **2019/05/16 18:20**

Secure Shell (SSH) (Port 22 - Layer 7)

SSH bzw. Secure Shell ist ein **kryptografisches Protokoll** mit dem man auf einen entfernten Rechner mittels einer **verschlüsselten Verbindung über ein unsicheres Netzwerk** zugreifen kann.

Die Shell (Kommandozeile) bietet **vollen Zugriff auf das Dateisystem und alle Funktionen** des Rechners.

Die Funktionen der Secure Shell beinhalten den Login auf entfernte Rechner, die interaktive und nicht interaktive Ausführung von Kommandos und das Kopieren von Dateien zwischen verschiedenen Rechnern eines Netzwerks. SSH bietet dazu eine kryptografisch gesicherte Kommunikation über das unsichere Netzwerk, eine zuverlässige gegenseitige Authentisierung, Verschlüsselung des gesamten Datenverkehrs auf Basis eines Passworts oder Public/Private-Key-Login-Methoden

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

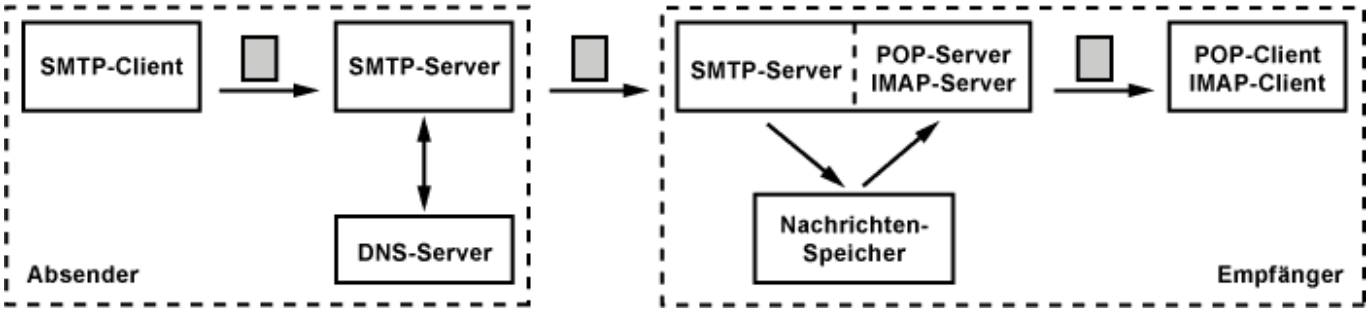
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_12

Last update: **2019/05/16 18:20**

Simple Mail Transfer Protocol (SMTP) (Port 25 - Layer 7)

SMTP ist ein **Kommunikationsprotokoll für die Übertragung von E-Mails**. Die **Kommunikation** erfolgt **zwischen** einem **E-Mail-Client und einem SMTP-Server (Postausgangsserver)** oder zwischen zwei SMTP-Server. Für den **Austausch** der E-Mails sind die **Mail Transfer Agents (MTAs)** zuständig. Untereinander verständigen sich die MTAs mit dem SMTP-Protokoll.

Neben SMTP gibt mit POP und IMAP noch zwei weitere Protokolle für den E-Mail-Austausch. Diese beiden Protokoll dienen jedoch nur dazu, um E-Mail abzuholen oder online zu verwalten. SMTP dagegen ist ein Kommunikationsprotokoll, das **E-Mails entgegennehmen und weiterleiten kann**.



Der Ablauf des E-Mail-Routings sieht in etwa so aus: Der SMTP-Server fragt einen DNS-Server ab und erhält eine Aufstellung von Mail-Servern, die E-Mails für den Ziel-SMTP-Server entgegennehmen. Jeder dieser Mail-Server (Mail Exchange) ist mit einer Priorität versehen. Der SMTP-Server versucht die Mail-Server in der vorgegebenen Reihenfolge zu kontaktieren, um die E-Mail zu übermitteln

Nachteile

- Für versendete E-Mails keine Versandbestätigung
- Geht eine E-Mail verloren, werden weder Sender noch Empfänger darüber informiert
- Nicht vorhandene Authentisierung des Benutzers beim Verbindungsaufbau zwischen SMTP-Client und SMTP-Server. Das führt dazu, dass eine beliebige Absenderadresse beim Versand einer E-Mail angegeben werden kann

Simple Mail Transfer Protocol Secure (SMTPS) (Port 465 und 587- Layer 7)

SMTPS (Simple Mail Transfer Protocol Secure) bezeichnet ein Verfahren zur **Absicherung der Kommunikation** beim E-Mail-Transport via **SMTP über SSL/TLS** und ermöglicht dadurch **Authentifizierung der Kommunikationspartner** auf Transportebene sowie **Integrität und Vertraulichkeit** der übertragenen Nachrichten.

SMTPS ist **kein eigenes Protokoll** und auch keine Erweiterung von SMTP, da es **vollkommen transparent und unabhängig** von diesem auf der Transportschicht arbeitet.

Das bedeutet, dass die Verbindung, über die SMTP abgewickelt wird, softwaremäßig mit den Verfahren **SSL oder TLS abgesichert** wird. Dies geschieht direkt beim Verbindungsaufbau, noch bevor irgendwelche Maildaten ausgetauscht werden. Da also die Verwendung der Sicherungsschicht nicht verhandelt wird, sind SMTPS-Dienste in der Regel auf einem eigenen TCP-Port erreichbar.

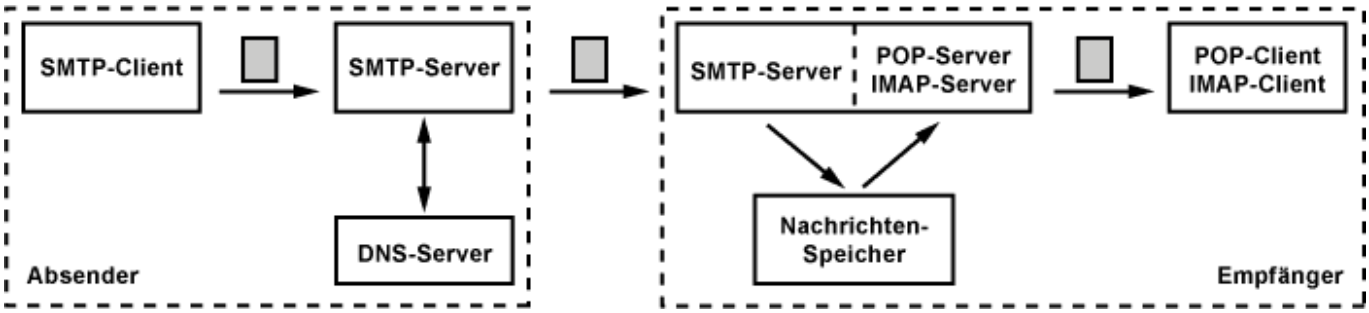
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_13

Last update: **2019/05/16 18:20**

Post Office Protocol Version 3 (POP3) (Port 110 - Layer 7)

POP ist ein Kommunikationsprotokoll, um **E-Mails von einem Posteingangsserver (POP-Server) abzuholen**. Die **Kommunikation** erfolgt **zwischen** einem **E-Mail-Client** und einem **E-Mail-Server (Posteingangsserver)**. Das Protokoll, das diesen Zugriff regelt, nennt sich POP (aus dem Jahr 1984), das in der aktuellen Version 3 vorliegt, und deshalb manchmal auch als POP3 bezeichnet wird.



Per Fernzugriff werden die gespeicherten E-Mails abgerufen und auf dem lokalen Computer gespeichert. POP sieht das Prinzip der Offline-Verarbeitung von E-Mails vor. Online werden die E-Mails vom Posteingangsserver vom E-Mail-Client heruntergeladen. Wenn sich darunter E-Mails mit einem großen Dateianhang befinden, kann der Download schon mal etwas länger dauern. Erst nach erfolgreichem und vollständigem Zugriff werden die E-Mails auf dem Server gelöscht. Die Bearbeitung der eingegangenen E-Mails erfolgt anschließend auf dem lokalen Computer des Benutzers ohne Verbindung (offline) POP-Server.

Die Verbindung zwischen POP-Server und E-Mail-Client erfolgt über TCP auf Port 110.

Post Office Protocol Version 3 Secure (POP3S) (Port 995 - Layer 7)

POP3S bezeichnet ein Netzwerkprotokoll zur Erweiterung des E-Mail-Übertragungsprotokolls POP3 um eine Verschlüsselung durch SSL/TLS. Üblicherweise wird für POP3S TCP auf Port 995 genutzt.

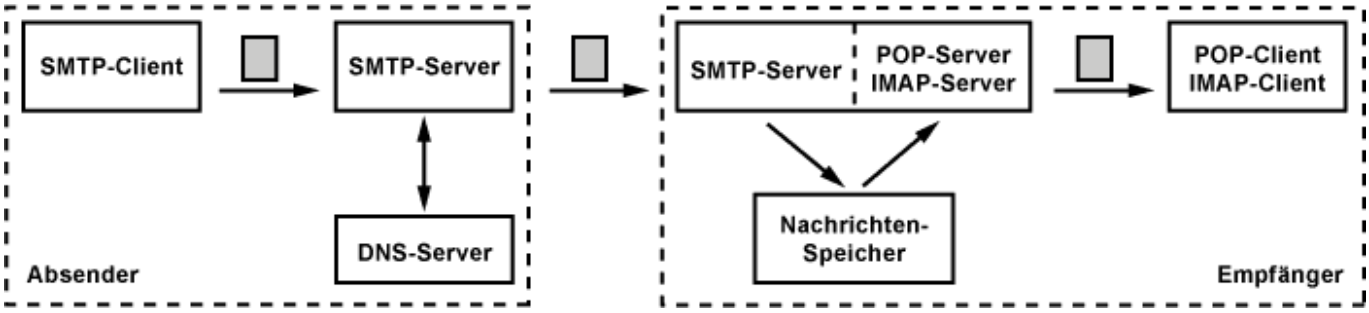
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_14

Last update: **2019/05/16 18:20**

Internet Mail Access Protocol (IMAP) (TCP-Port 143 - Layer 7)

IMAP ist ein Kommunikationsprotokoll, um **E-Mails** auf einem **entfernten Server** ähnlich wie Dateien **zu verwalten**. Dabei **bleiben alle E-Mails auf dem IMAP-Server**. Erst wenn eine E-Mail gelesen werden soll, wird sie heruntergeladen.



IMAP erlaubt den Zugriff auf eine Mailbox, **ähnlich wie mit POP**. Der entscheidende **Unterschied** zwischen beiden Protokollen ist der **Online-Modus von IMAP**, über den der **E-Mail-Client ständig in Verbindung mit dem E-Mail-Server** steht. Während einer IMAP-Sitzung kann auf einzelne E-Mails zugegriffen werden, die so lange auf dem Server bleiben, bis sie gelöscht werden. Dadurch kann von überall auf dem Server zugegriffen werden. Auch mit einem Endgerät, das nur mit geringer Bandbreite am Netzwerk angeschlossen ist. Die **E-Mails werden nur dann heruntergeladen, wenn der Anwender diese Lesen will**. E-Mails mit einem großen Dateianhang verstopfen dann nicht mehr ungewollt den Zugang zum Netzwerk.

Internet Mail Access Protocol Secure (IMAPS) (TCP-Port 143 - Layer 7)

Bei der Verwendung von IMAPS wird die Verbindung zum Server bereits während des Verbindungsaufbaus durch SSL verschlüsselt. Damit der Server das erkennt, muss ein anderer Port verwendet werden. Dafür wurde der Port 993 reserviert.

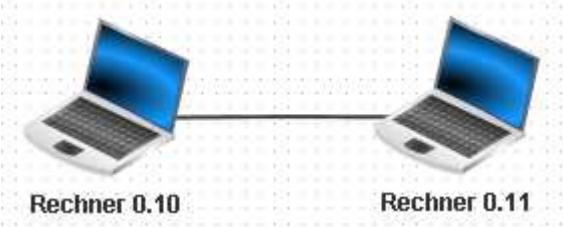
Nach dem Aufbau der SSL-Verbindung wird IMAP verwendet. Die SSL-Schicht ist für das IMAP-Protokoll transparent, d. h., es werden keine Änderungen am IMAP-Protokoll vorgenommen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_09:18_09_15

Last update: **2019/05/16 18:20**

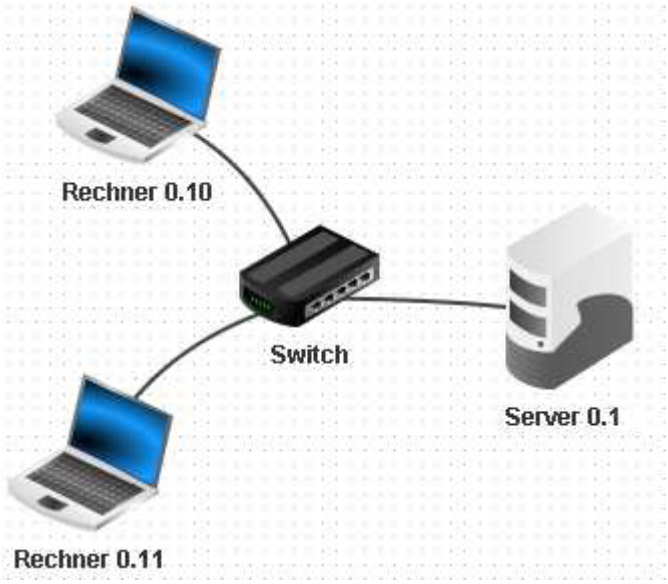
Aufgabe 1



1. Erstellen Sie ein Netzwerk mit zwei vernetzten Computern, welche beide eine Client-Funktion haben (Symbol: Notebook). Die Computer sollen die abgebildeten Namen sowie die IPs 192.168.0.10/24 und 192.168.0.11/24 besitzen. (Durch die richtige Subnetzmaske 255.255.255.0 stellen Sie sicher, dass beide Computer im selben Netzwerk liegen.)
2. Installieren Sie auf dem Rechner 0.10 eine „Befehlszeile“ (Terminal). Starten Sie das Terminal und testen Sie die Verbindung zum Rechner 0.11 mit dem Befehl `ping 192.168.0.11`. Beobachten Sie die Netzwerkaktivität, indem Sie sich den Datenaustausch von Rechner 0.10 anzeigen lassen.
3. Testen Sie auch andere Befehle auf dem Terminal, wie z. B. die Befehle `ipconfig`, oder `host localhost` oder `dir`. Der Sinn des host-Befehls wird zu einem späteren Zeitpunkt im Zusammenhang mit einem DNS-Server evtl. deutlicher.

Aufgabe 2

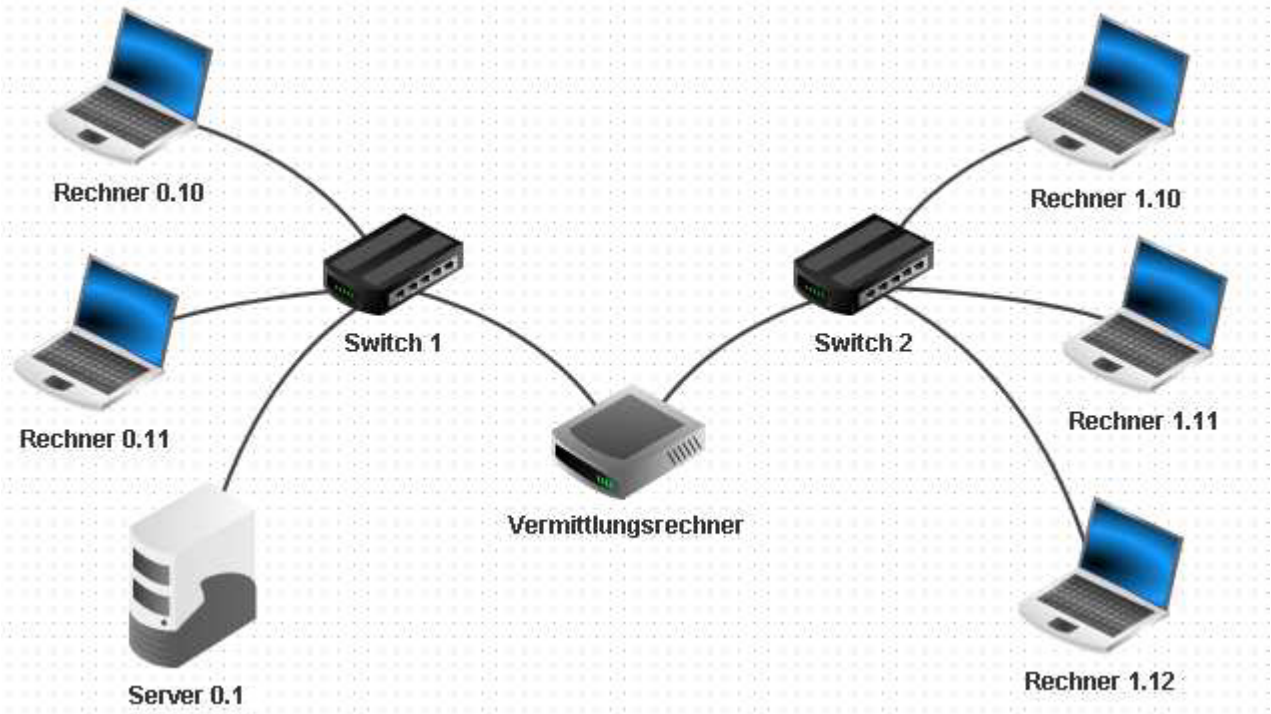
Erweitern Sie nun das Netzwerk um einen dritten Computer, einen Server, mit dem abgebildeten Namen und der IP 192.168.0.1/24, welcher im gleichen Netzwerk liegen soll. Achten Sie darauf, dass Sie von nun an für Computer mit der Funktion eines Servers, das Rechner-Symbol (Standrechner) wählen. Verbinden Sie alle Computer mit einem Switch wie abgebildet.



Installieren Sie auf dem Server 0.1 einen Echo-Server und starten diesen auf dem voreingestellten Port 55555. Installieren Sie auf einem Client einen Echo-Client und verbinden Sie diesen mit dem Server. Senden Sie vom Client einige Textnachrichten und beobachten Sie den Effekt. Schauen Sie sich auch die Netzwerkaktivität im Datenaustausch-Fenster des Clients an.

Aufgabe 3

- Erstellen Sie neben dem bereits vorhanden Netzwerk ein weiteres Netzwerk mit drei Rechnern wie abgebildet. Die neuen Rechner sollen sich in einem logisch anderen Netzwerk befinden. Wählen Sie dafür die IPs 10.1.1.10/16 bis 10.1.1.12/16. Verbinden Sie anschließend die beiden Netzwerke mit einem Vermittlungsrechner (Router), welcher die Netzwerkkarten mit den IPs 192.168.0.254/24 und 10.1.1.254/16 besitzt.



- Prüfen Sie anschließend im Terminal mit einem ping-Befehl die Verbindung der Rechner vom Netzwerk 192.168.0.0/24 zu den Rechnern des Netzes 10.1.0.0/16.
    - Beschreiben Sie, welches Problem auftreten kann und wie man sicherstellen kann, dass die Rechner aus einem fremden Netzwerk erreicht werden können.
- Testen Sie die Netzwerkverbindung auch mit dem Echo-Client und Echo-Server. Installieren Sie dazu auf einem Rechner aus dem Netz 192.168.0.0/24 einen Echo-Server und auf einem Rechner im Netz 10.1.0.0/16 einen Echo-Client.

Aufgabe 4 - Webserver

- Installieren Sie auf dem Server 192.168.0.1 einen Webserver und einen Texteditor. Starten Sie den Texteditor und öffnen Sie hiermit die Datei index.html aus dem virtuellen Verzeichnis root/webserver. Passen Sie den html-Code so an, dass eine Seite mit Ihren Informationen angezeigt wird. Erstellen Sie auch eine neue Seite kontakt.html, welche von der Startseite verlinkt werden soll.
- Starten Sie die Anwendung „Webserver“ mit einem Doppelklick. Starten Sie dann den virtuellen Webserver über den Button Starten.
- Installieren Sie dann auf dem Rechner 10.1.1.10 einen Webbrowser. Starten Sie den Browser und bauen Sie eine Verbindung zum Webserver auf, indem Sie die URL http://192.168.0.1 in die Adressleiste des Webbrowsers eingeben.

Aufgabe 5 - DNS-Server

- Erstellen Sie einen neuen Server mit der IP 1.1.1.1/24 und dem Gateway 1.1.1.254. Ändern Sie die Anzahl der Schnittstellen am Vermittlungsrechner auf drei ab. Ergänzen Sie die auf der neuen Registerkarte zur dritten Netzwerkkarte die Einstellungen: IP-Adresse 1.1.1.254/24 (Netzmaske 255.255.255.0). Verbinden Sie anschließend den neuen Server mit dem Vermittlungsrechner.
- Tragen Sie bei jedem Rechner in den Einstellungen die DNS-Server-Adresse 1.1.1.1 ein. Dies entspricht dem gerade erstellten DNS-Server.
- Installieren Sie auf dem Server 1.1.1.1 die Anwendung „DNS-Server“. Tragen Sie in die Eingabefelder den Domainnamen www.myweb.at und die zugehörige IP-Adresse 1.1.1.1 ein. Starten Sie abschließend den DNS-Server mit dem Button „Starten“. Testen Sie die Verbindung von Ihrem Webbrowser nun mit der URL http://www.myweb.at

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_11

Last update: **2019/05/16 18:20**

Filius Übungen

Übung 01 - Schulnetzwerk

- aufgabe_schulnetzwerk.pdf

Übung 02 - Firma

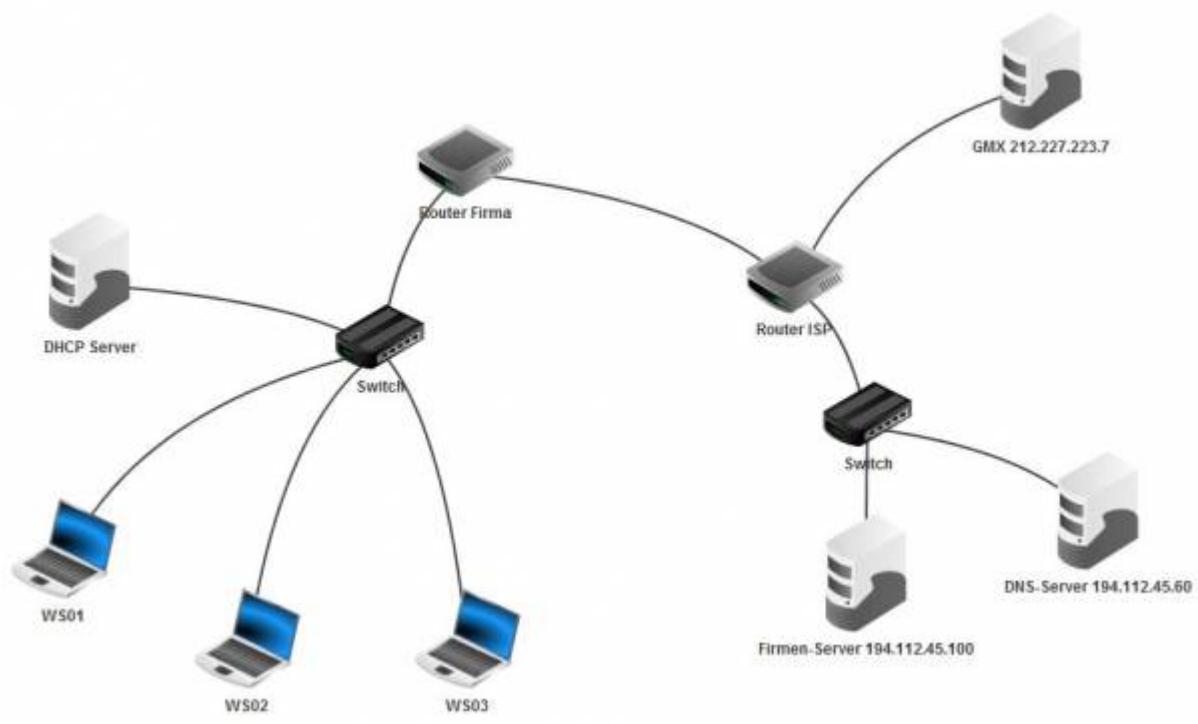
Du bekommst den Auftrag, in einer kleinen Firma mit drei PCs und einem Server das Netzwerk neu einzurichten. Ebenso soll für Web- und Maildienste ein eigener Server beim Provider angemietet und konfiguriert werden.

Starte das Programm Filius und öffne die Datei netzwerk-vorlage.fls.

Die Geräte des Firmennetzwerks und der Firmen-Server müssen vollständig konfiguriert werden, ebenso müssen am DNS-Server zusätzliche Einstellungen getätigt werden.

Alle bereits vorkonfigurierten Einstellungen sind funktionstüchtig und sollten nicht mehr verändert werden!

„Das Internet“ wird repräsentiert durch den Server von GMX, der per „Waterwolf“-Browser über http://www.gmx.at erreichbar ist. Auch sind zwei Emailadressen eingerichtet: alice@gmx.at und bob@gmx.at. POP3-Server: pop3.gmx.at, SMTP-Server: smtp.gmx.at, Passwörter: Benutzername in Großbuchstaben (ALICE bzw. BOB). Der DNS-Server und der beim Provider zu konfigurierende Firmen-Server stellen das Netzwerk des Internet Service Providers (ISP) dar.



Folgende Unterlagen zur Konfiguration bekommst du ausgehändigt:

- Firmennetzwerk:
  - Externe IP: 194.112.174.1
  - IP Router: 10.0.0.254
  - IP DHCP-Server: 10.0.0.100
  - Konfiguration der PCs via DHCP: 10.0.0.1 - 10.0.0.99
- Internetzugang für Firmen-Router
  - Externe IP: 194.112.174.1
  - Gateway: 194.112.174.254
  - DNS-Server: 194.112.45.60
- Firmenserver:
  - IP: 194.112.45.100
  - Gateway: 194.112.45.254
  - Domain (A): www.firma.at
  - Domain (MX): firma.at
  - Emailadresse: office@firma.at, Passwort: OFFICE

---

1. Richte den DHCP-Server und die Workstations ein.
2. Konfiguriere den Firmen-Router und stelle eine Verbindung zum ISP her. Prüfe die Verbindung mit dem Befehl ping zum ISP-Router und zum GMX-Server.
3. Bearbeite die Netzwerkeinstellungen des Firmen-Servers und prüfe die Erreichbarkeit vom DNS-Server und vom Firmennetzwerk aus.
4. Installiere und starte Web- und Mailserver auf dem Firmen-Server. Die Homepage soll umgestelltet werden mit „Herzlich willkommen auf unserer Homepage!“ (o.ä.)
5. Erweitere den DNS-Server um die Einträge für www.firma.at und firma.at
6. Installiere auf der Workstation WS01 Browser und Email-Client (mit der Firmenadresse office@firma.at)
7. Auf der Workstation WS02 und WS03 installiere ebenfalls Browser und Email-Client (WS02: alice@gmx.at, WS03: bob@gmx.at)
8. Sende eine Email von office@firma.at an alice@gmx.at und bob@gmx.at
9. Beantworte beide Mails.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:18:18_12

Last update: **2019/05/16 18:20**

inf8bi_201819_matura

Umfeldfragen

1. Erläutere den Begriff „Cyber-Mobbing“

Absichtliches und über einen längeren Zeitraum anhaltendes Beleidigen, Bedrohen, Bloßstellen, Belästigen oder Ausgrenzen anderer über digitale Medien.
Besonderheiten: Jederzeit, großes Publikum, Anonymität

1. Erkläre verschiedene Möglichkeiten, sich gegen Cyber-Mobbing zu wehren.
    - Cyber-Mobbing Reaktion
2. Nenne verschiedene Anlaufstellen zur Unterstützung und Beratung bei Cyber-Mobbing.
    - Beratungsmöglichkeiten
3. Erläutere den Begriff „Datenschutz“.
    - Datenschutz
4. Nenne verschiedene Grundregeln, persönliche Daten im Internet zu schützen.
    - Grundregeln Daten schützen
5. Erläutere den Terminus „Recht am eigenen Bild“.
    - Recht am eigenen Bild
6. Gib an, wie man am besten vorgesehen sollte, um peinliche Fotos wieder aus dem Netz zu bekommen.
    - Fotos entfernen
7. Nenne Grundregeln zur Erstellung sicherer Passwörter.
    - Sichere Passwörter
8. Erläutere den Begriff „Zwei-Stufen-Authentifizierung“.
    - Zwei-Stufen-Authentifizierung
9. Erläutere, was durch das Urheberrecht geschützt wird und wann Urheberrechte verletzt werden.
    - Urheberrecht
10. Erläutere den Begriff „Creative-Commons-Lizenz“.
    - Creative-Commons-Lizenz
11. Erläutere den Begriff „Phishing“ und wie man sich davor schützen kann.
    - Phishing
12. Erläutere den Begriff „Sextortion“.
    - Sextortion
13. Erläutere den Begriff „Cyber-Grooming“.
    - Cyber-Grooming
14. Nenne verschiedene Betrugsarten, auf die achtlose Internetuser reinfallen können.
    - Betrugsarten
15. Nenne verschiedene Möglichkeiten, seinen Computer vor Viren, Spam, etc. zu schützen.
    - Viren, Spam
16. Erläutere den Begriff „Spam“ und wie man sich davor schützen kann.
    - Spam
17. Erläutere den Begriff „Hoax“.
    - Hoax
18. Nenne verschiedene Möglichkeiten, sicher im Internet einzukaufen/zu bezahlen.

    - Bezahlen im INternet

19. Erläutere, inwieweit Kinder bzw. Minderjährige im Internet einkaufen dürfen.
    - Altersgrenzen Einkaufen im Internet
20. Erläutere den Terminus „Recht auf Gewährleistung“.
    - Recht auf Gewährleistung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19

Last update: **2019/05/16 18:39**

- Kinder unter 7: - nicht geschäftsfähig
- Kinder zwischen 7 und 14: - beschränkt geschäftsfähig (nur wenn der Erziehungsberechtigter zustimmt)
- zwischen 14 und 18: - Geschäfte die den eigenen Lebensunterhalt nicht gefährden → sonst unwirksam
- ab 18: - voll geschäftsfahig

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:altersgrenzen_einkaufen_im_internet

Last update: **2019/05/16 18:38**

Kinder und Jugendliche

- 147 Rat auf Draht (Tel. 147)
- Kinder- und Jugendanwaltschaften Österreichs

Schulen

- Schulpsychologischer Dienst und Bildungsberatung
- Saferinternet.at-Workshops

Eltern

- Familienberatungsstellen
- 147 Rat auf Draht (Tel. 147)

Sonstige

- Opfernotruf (Tel. 0800/112 112)
- Kriminalprävention (Tel. 059-133)
- Frauenhelpline „Halt der Gewalt“ (Tel. 0800/222 555)
- 24-Stunden Frauennotruf der Stadt Wien (Tel. 01/71 71 9)
- 142 Telefonseelsorge (Tel. 142)
- Internet Ombudsmann (ombudsmann.at): Entfernung unerwünschter Online-Inhalte

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:beratungsmoeglichkeiten

Last update: **2019/05/16 18:38**

Betrugsarten

„Phishing“ → Trick um an geheime Daten zu kommen zB. Online-Bankdaten; meist durch betrügerische Email, in denen man aufgefordert wird persönliche Daten anzugeben, oder es werden durch E-Mail-Anhänge Schadsoftwares installiert, um die Daten auszuspionieren

„Vermeintliche GRATIS-Angebote“ → attraktive Websites bieten Gratis-Produkte an → Kostenhinweise sind versteckt → Rechnung kommt meist später

„Abofalle“→ bei „Spaß-Anwendungen“ wird die Eingabe der Handynummer verlangt → teures Mehrwert-SMS-Abo → Geld wird durch den Mobilfunkbetreiber extra verrechnet

„Online-Shopping-Betrug“ → auch wenn Geld überwiesen wurde, erhält man die Ware nie

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:betrugsarten

Last update: **2019/05/16 18:38**

- Vorsicht bei Verkäufer aus dem Ausland und Überweisungen ins Ausland
- Vorsicht bei zu günstigen Angeboten
- bei größeren Beiträgen → mit selbst ausgewählten Treuhandunternehmen zahlen
- nicht kaufen wenn der Verkäufer eine andere Sprache verwendet oder die Nachrichten von einer automatischen Übersetzungssoftware übersetzt wird
- bei Zahlung mit Kreditkarte → wenn mögliche Prepaid- Kreditkarte → kann mit einem bestimmten Betrag aufgeladen werden
- Lieferung per Nachname → gute Alternative (jedoch meist etwas teuerer → aber sehr sicher)
- Bezahlung auf Rechnung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:bezahlen_im_internet

Last update: **2019/05/16 18:38**

11. Creative-Commons-Lizenz:

Was ist Creative Commons?

Um die juristisch eindeutige und korrekte Formulierung der Lizenzbestimmungen freier Lizenzen zu gewährleisten, wurde 2001 die Organisation Creative Commons gegründet. Die Lizenzen, die von Creative Commons entwickelt wurden und frei zur Verfügung stehen, ermöglichen es dem Autor/der Autorin, den NutzerInnen je nach Bedarf bestimmte Verwertungsrechte im Rahmen einer Werknutzungsbewilligung einzuräumen, beispielsweise die kommerzielle oder nicht-kommerzielle Nutzung oder die Weiterverarbeitung.

Woran erkenne ich eine Creative Commons (CC) Lizenz?

Dokumente mit einer Creative-Commons-Lizenz weisen auf der Titelseite zumeist das Logo der jeweilige Lizenz sowie einen Texthinweis auf den Lizenztyp und einen Link zum vollständigen Lizenztext auf.

Beispiel: This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-nd/4.0/.

Welche Creative Commons (CC) Lizenzen gibt es?

Es gibt insgesamt sechs verschiedene Lizenzen, die sich aus vier unterschiedlichen Modulen zusammensetzen.

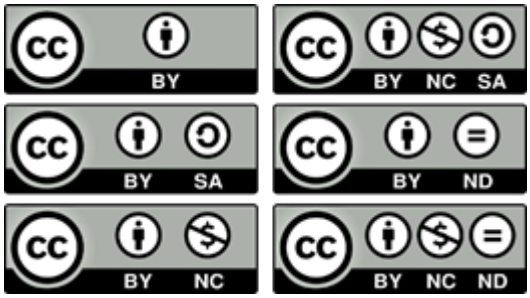
Die vier Module sind:

- BY: Namensnennung - Der Name des Urhebers muss genannt werden.
- NC: Nicht-kommerziell (non-commercial) - Das Werk darf nicht für kommerzielle Zwecke verwendet werden.
- ND: Keine Bearbeitung (no derivatives) - Das Werk darf nicht verändert werden.
- SA: Weitergabe unter gleichen Bedingungen (share alike) - Das Werk muss nach Veränderungen unter der gleichen Lizenz weitergegeben werden.

Daraus ergeben sich folgende sechs Lizenzen:

- CC BY: Namensnennung
- CC BY-SA: Namensnennung, Weitergabe unter gleichen Bedingungen
- CC BY-ND: Namensnennung, keine Bearbeitung
- CC BY-NC: Namensnennung, nicht kommerziell
- CC BY-NC-SA: Namensnennung, nicht kommerziell, Weitergabe unter gleichen Bedingungen
- CC BY-NC-ND: Namensnennung, nicht kommerziell, keine Bearbeitung

Die Icons der sechs Lizenzen sind:



Mehr Information zu den einzelnen Lizenzen sowie die Links zu den Kurzfassungen und juristischen Langfassungen finden Sie hier.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:creative-commons-lizenz

Last update: **2019/05/21 18:09**

„Cyber-Grooming“

Pädophile, die über das Internet Kontakt mit Kindern aufnehmen um sie später auf verschiedenste Art zu missbrauchen; sie geben sich oft als Gleichaltrig oder Modeagenten aus! Wenn Nacktfotos ausgetaucht wurden, kann es dazu kommen, dass:

1. diese an Kinderpornografie-Foren weitergeleitet werden
2. die Kinder später damit erpresst werden, wenn sie den Kontakt abbrechen wollen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:cyber-grooming

Last update: **2019/05/16 18:38**

2. Erkläre verschiedene Möglichkeiten, sich gegen Cyber-Mobbing zu wehren.

- Sichere Beweise: Screenshots von Chats
- Nutzer Sperren: Täter auf Sozialen Netzwerken sperren oder Nummer ändern
- Probleme melden: Die Betreiber der Website kontaktieren
- Rechte: Ernstes Cyber-Mobbing kann für den Täter rechtliche Folgen haben

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:cyber-mobbing_reaktion

Last update: **2019/05/16 18:38**

Datenschutz

Jeder kann selbst entscheiden, welche persönlichen Daten von ihm/ihr veröffentlicht werden (z.B.: Name, Adresse, Alter, E-Mail, IP-Adresse, ...);

sensible Daten: politische Einstellung, Gesundheit, Religion, Sexualleben, ...;

Totale Anonymität ist heutzutage beinahe unmöglich. Deshalb soll überlegt sein, welche Datenspur man im Internet hinterlässt.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:datenschutz

Last update: **2019/05/16 18:38**

7. Peinliche Fotos aus dem Netz löschen lassen:

Bloßstellung MUSS objektiv nachvollziehbar sein Vorgehen: Screenshot der Website speichern Schriftlich die Person oder das Unternehmen kontaktieren und um Löschung bitten (mit Frist - zB 3 Wochen); Link Foto melden Sollte das nichts nutzen → Unterlassungsklage und Schadensersatzforderung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:fotos_entfernen

Last update: **2019/05/16 18:38**

Grundregeln

- Sichere Passwörter verwenden
- Spam ignorieren
- Nicknames nutzen
- So wenig persönlichen Daten wie möglich preisgeben
- Anti-Viren-Programm benutzen
- Vorsicht bei der Nutzung von öffentlichen PCs
- Mehrere E-Mail-Adressen verwenden
- Regelmäßige Updates von Browsern und Anti-Viren-Programme

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:grundregeln_daten_schuetzen

Last update: **2019/05/16 18:38**

Als Hoax wird eine Nachricht bezeichnet, die bewusst als falsche Nachricht in Umlauf gebracht wird. Beispiele dafür sind Scherzmeldungen oder auch Kettenbriefe, die oft über Facebook oder WhatsApp verbreitet werden. Der Großteil der Hoaxes ist jedoch harmlos und der Aspekt des Scherzes soll im Vordergrund stehen. Ein Problem stellen Hoaxes nur da, wenn unfreiwillig Fotos von Personen online gestellt werden und diese als beispielsweise Kinderschänder bezeichnet werden.

Beispiel: Achtung! Ein Virus breitet sich auf Facebook aus! Nehmt auf keinen Fall die Freundschaftsanfrage von XY an!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:hoax

Last update: **2019/05/16 18:38**

12. Phishing: Als Phishing werden Tricks bezeichnet, die benutzt werden um Usern geheime Daten (Online-Banking-Login) zu entlocken. Die häufigste Form des Phishings sind zwielichtige Chat-Nachrichten und E-mails. Wichtig: Seriöse Unternehmen und Banken verlangen niemals vertrauliche Daten wie Logins, Passwörter oder Transaktionsnummern (TANs) per E-Mail oder Chat. Gegen Phishing schützen:

- Auf keine Links in E-Mails klicken
- Keine vertraulichen Daten per Chat/ E-Mail übermitteln
- Keine unbekannten Datei-Anhänge öffnen
- Überraschende Änderungen der Login-Seite sofort den Betreiber melden
- Vertrauliche Daten nur über SSL verschlüsselte Seiten bekannt geben
- Regelmäßige Sicherheits-Updates im Browser und Anti-Viren-Programme/ Firewalls nutzen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:phishing

Last update: **2019/05/16 18:38**

Recht am eigenen Bild

Urheberrechtsgesetz: Bilder, die die „berechtigten Interessen“ von Personen verletzen, dürfen von keinem anderen veröffentlicht werden;

Aufnahmen an öffentlichen Plätzen sind meist kein Problem, außer die Abbildung ist nachtteilig für eine Person (zB Oben-ohne-Foto am Strand)

Abbildungen im privaten Bereich: Veröffentlichung von Fotos die einen bloßstellen sind verboten; Bloßstellung muss aber objektiv nachvollziehbar sein.

Vor der Veröffentlichung: Stellen Sie sich selbst die Frage, ob Sie ein solches Bild von Ihnen selbst im Internet haben wollen. Oder Fragen Sie die Abgebildete Person.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:recht_am_eigenen_bild

Last update: **2019/05/16 18:38**

Recht auf Gewährleistung: besteht bei mangelhafter Leistung durch den Verkäufer

Garantie ist davon unabhängig und für den Verkäufer freiwillig

Gewährleistung ist ein Recht! gilt innerhalb von 2 Jahren nach dem Verkauf

Vorsicht bei Privatpersonen!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:recht_auf_gewaehrleistung

Last update: **2019/05/16 18:38**

„Sextortion“

= Betrugsmasche im Internet: dabei werden Internetnutzer von einem Unbekannten(=meist attraktiv) dazu aufgefordert, ihm sexuelle Fotos oder Videos zu schicken. Diese werden dann dazu verwendet, die Person zu erpressen → Veröffentlichung der Bilder/Videos

Empfohlen: Keines Falls Zahlung leisten → zur Polizei gehen!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:sextortion

Last update: **2019/05/16 18:38**

8. Grundregeln zur Erstellung sicherer Passwörter

- 8-10 Buchstaben, Zahlen und Sonderzeichen
- Keine einfachen Wörter, Namen oder Zahlenkombinationen
- Strategien:
    1. Buchstaben durch Zahlen/Sonderzeichen ersetzen
    2. Sätze unter Verwendung von Zahlen und Sonderzeichen abkürzen
    3. Wortreihenfolgen ohne Zusammenhang bilden
- Verschiedene PWs für untersch. Websites
- 2-Faktor-Authentifizierung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:sichere_passwoerter

Last update: **2019/05/16 18:38**

Als Spam werden unerwünschte E-Mails bezeichnet. Obwohl sich der Nutzer bei keinem Newsletter angemeldet hat, erhält er dennoch Benachrichtigungen zu Produkten, die für ihn nicht relevant sind. Spam ist in Österreich zwar verboten, jedoch dürfen Unternehmen in folgenden Fällen Nachrichten versenden:

- Der Adressat stimmt vorher zu

- Der Versender hat die Kontaktinformationen (z.B. im Zusammenhang mit einem Verkauf) erhalten.

Gegen Spam kann man sich folgendermaßen schützen:

- E-Mail nicht an Unbekannte weitergeben

- Mehr als eine E-Mail Adresse verwenden (eine für private Zwecke, eine für Anmeldungen)

- Nicht auf Spam antworten und keine Dateianhänge öffnen

- Eintragung in die ECG-Liste (Anbieter dürfen keine Nachrichten an Personen senden, die in dieser Liste eingetragen sind)

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:spam

Last update: **2019/05/16 18:38**

10. Urheberrecht:

Urheberrecht schützt die Urheberschaft an einem künstlerischen Werk. Geschützt ist geistiges Eigentum in den Kategorien: Werke der Literatur, der Tonkunst, der bildenden Kunst und der Filmkunst, die ein Mindestmaß an Individualität und Originalität aufwiesen. Jede Abrufbarmachung von geschützten Werken ohne die Zustimmung des Rechteinhabers ist eine Verletzung des Urheberrechts. Ob zu privaten oder kommerziellen Zwecken ist hierbei egal. Es muss also immer die Zustimmung zur weiteren Verwendung eingeholt werden. Das Urheberrecht gilt bis 70 Jahre nach dem Tod des Inhabers.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:urheberrecht

Last update: **2019/05/16 18:38**

Software-Updates, Firewall, Anti-Viren-Programme, Anti-Spyware-Programm

Bei Software, Betriebssystem und Programmen darauf achten, dass diese immer auf dem aktuellen Stand sind, um Sicherheitslücken zu schließen und Viren leichter zu erkennen.

WLAN-Netzwerke sollte man immer mit einem sehr guten Passwort verschlüsseln, damit Unbefugte keinen Zugriff darauf haben. (WPA bzw. WPA2-Verschlüsselung). MAC-Adresse für die eigenen Geräte am Router einrichten.

Internetbrowser sollten bei ein paar Stellen (Cookies, Pop-Up-Fenster...) manuell konfiguriert werden.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:viren_spam

Last update: **2019/05/16 18:38**

9. Zwei-Stufen-Authentifizierung:

Eine zusätzliche Sicherheitsmaßnahme zum Schutz des Benutzerkontos. Neben dem Passwort muss beim Login eine weitere Sicherheitskomponente eingegeben werden, zB ein Bestätigungscode. Diese Codes werden meist zu einer bestimmten Rufnummer geschickt oder von einem Generator erzeugt. Unbefugt haben - auch wenn sie das PW kennen - keinen Zugriff auf das Benutzerkonto.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf8bi_201819_matura:19:zwei-stufen-authentifizierung

Last update: **2019/05/16 18:38**