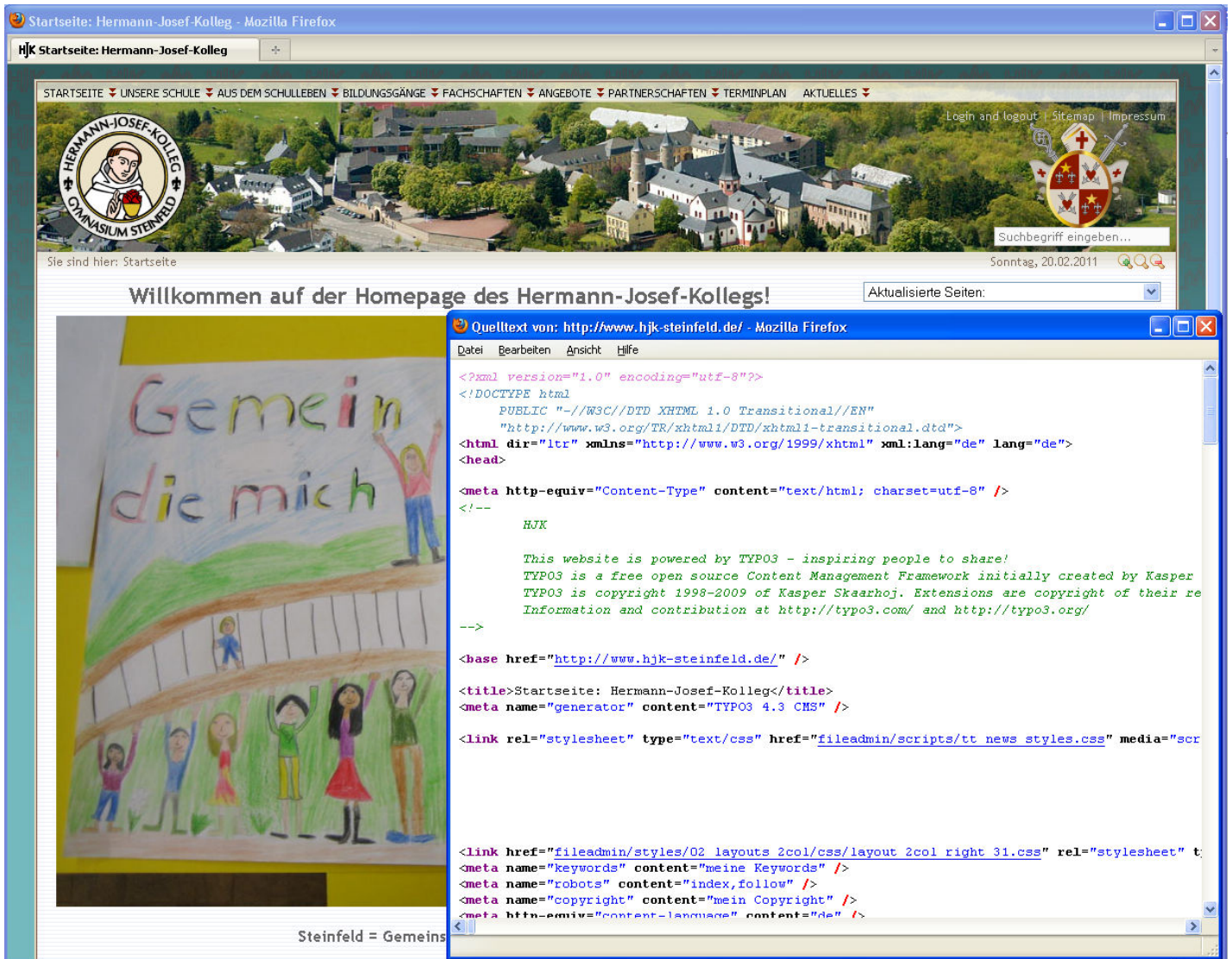


Einführung

in die

WEBPROGRAMMIERUNG



Teil 1

Webdesign mit

HTML und CSS

HJK 2012

INHALTSVERZEICHNIS

1. ALLGEMEINE INFORMATIONEN ÜBER DAS INTERNET	4
1.1. ENTWICKLUNG DES INTERNETS	4
1.2. AUFBAU DES INTERNETS	4
1.3. VERWENDUNG VON PROTOKOLLEN IM INTERNET	5
1.4. AUFBAU DER INTERNET-ADRESSE	6
1.5. DIE GEBRÄUCHLICHSTEN DIENSTE IM INTERNET	6
1.6. <i>HTML UND CSS</i>	7
1.7. AUFGABEN	7
2. ERSTE SCHRITTE IN HTML	8
2.1. TAGS ALS STRUKTURMARKER IN HTML	8
2.2. GRUNDGERÜST EINES HTML-DOKUMENTS	8
2.3. TEXTSTRUKTUREN	9
2.4. WICHTIGE SONDERZEICHEN	10
2.5. AUFGABEN	10
3. ERSTE OPTISCHE FORMATIERUNGEN MIT CSS	11
3.1. AUFBAU DER CSS-REGELN (STYLESHEET-REGELN)	11
3.2. CSS-REGELN AN HTML EINBINDEN	11
3.3. DIE EINFACHEN SELEKTOREN UND VERERBUNGSPRINZIP	12
3.4. DAS AUSSEHEN VON TEXT MIT CSS GESTALTEN	14
3.5. FARBEN IN HTML	14
3.6. AUFGABEN	15
4. ONLINE-GRAFIKEN	16
4.1. GELÄUFIGE GRAFIKFORMATE IM INTERNET.	16
4.2. EINBINDEN VON GRAFIKEN IN HTML	17
4.3. GRAFIKEN UND TEXT MIT CSS STYLEN	17
4.4. MIT CSS ELEMENTE MIT HINTERGRUNDFARBEN UND –BILDERN VERSEHEN	18
4.5. AUFGABEN	18
5. MULTIMEDIA ELEMENTE	19
5.1. EINBINDUNG VON AUDIodateien MIT <AUDIO> TAG.	19
5.2. EINBINDUNG VON VIDEodateien MIT <VIDEO> TAG.	19
5.3. INHALTE ZWISCHEN DEM ANFANGS- UND END-TAG.	19
5.4. AUFGABEN	19
6. LISTEN	20
6.1. UNGEORDNETE LISTEN	20
6.2. NUMMERIERTE AUFGÄHlungen	20
6.3. DEFINITIONSListen	21
6.4. GESCHACHTELTE LISTEN	21
6.5. AUFGABEN	21
7. ORDNERSTRUKTUR UND HYPERLINKS	22
7.1. INTERNETPRÄSENZ IN ORDNER ORGANISIEREN	22
7.2. RELATIVE UND ABSOLUTE PFADE DEFINIEREN.	22
7.3. HYPERLINKS	23
7.4. AUFGABEN	23

8.	WEITERE NÜTZLICHE CSS – SELEKTOREN.....	24
8.1.	KONTEXT-SELEKTOREN	24
8.2.	ATTRIBUT-SELEKTOREN	25
8.3.	AUFGABEN	25
9.	TABELLEN	26
9.1.	DER GRUNDLEGENDE AUFBAU EINER TABELLE	26
9.2.	SPALTEN- UND TABELLENÜBERSCHRIFTEN	26
9.3.	BEREICHE EINER TABELLE ORGANISIEREN	26
9.4.	SPALTEN ORGANISIEREN	27
9.5.	SPALTEN UND ZEILEN VERBINDEN	27
9.6.	EIN KOMPLEXERES BEISPIEL EINER TABELLE	27
9.7.	AUFGABEN	28
10.	CSS – PSEUDOFORMATE UND DEREN ANWENDUNG	29
10.1.	WICHTIGSTE PSEUDOKLASSEN	29
10.2.	WICHTIGSTE PSEUDOELEMENTE	29
10.3.	BEISPIELE FÜR DIE ANWENDUNG DER PSEUDOFORMATE	30
10.4.	AUFGABEN	31
11.	LAYOUT EINER WEBSEITE	32
11.1.	WEBSEITEN-BEREICHE (SECTIONING)	32
11.2.	ANZEIGEART DER ELEMENTE.....	32
11.3.	POSITIONIEREN DER ELEMENTE	33
11.4.	GRÖÖE DER ELEMENTE.....	33
11.5.	DAS BOX-MODELL.....	34
11.6.	AUFGABEN	35
12.	WEITERE WICHTIGEN ASPEKTE WEBDESIGNS.....	36
12.1.	INTERNET-PRÄSENZ FÜR DIE SUCHMASCHINEN VORBEREITEN	36
12.2.	KLARE STRUKTUR EINER WEBPRÄSENZ	37
12.3.	LAYOUT.....	37
13.	RECHTLICHE ASPEKTE EINER WEBPRÄSENZ	38
14.	PROJEKTAUFGABE.....	39
15.	KURZE HTML REFERENZ.....	I
16.	KURZE CSS REFERENZ.....	IV

1. Allgemeine Informationen über das Internet

1.1. Entwicklung des Internets

Die Geschichte des **Internets (Netz der Netze)** beginnt beim US-Militär, genauer dem Department of Defense (DoD). 1969 wurde das ARPANET ins Leben gerufen, um Rechner innerhalb der USA zu vernetzen. Der entstehende Rechnerverbund sollte auch bei Teilausfall (Ausfall von einigen Recheneinheiten z.B. bei einem Atomkrieg) eine zuverlässige Kommunikation gewährleisten. Das ARPANET wurde bereits 1972, schon weitgehend auf TCP/IP basierend, der Öffentlichkeit zugänglich gemacht. Unabhängig vom ARPANET entwickelte sich ein vorwiegend wissenschaftlich genutztes Netz (Computer Science Network - CSnet). Die Vernetzung dieses Netzes (CSnet) mit dem bereits ausgebauten ARPANET im Jahre 1982 führte zur Etablierung des Internet. Parallel dazu wurde ein Netzverbund, das National Science Foundation Network (NSFnet), von der National Science Foundation erstellt. 1983 wurde ein Teil des ARPANET abgetrennt und zum „militärisch orientierten“ MILNET. Durch die Gestaltung eines Netzüberganges vom ARPANET zum NSFnet wurde der bestehende Netzwerkverbund ausgebaut, wobei die NSF-Netzstruktur „die Hauptlast im INTERNET-Verbund“ übernahm. Daher wurde im Jahre 1990 das alte ARPANET endgültig aufgelöst.

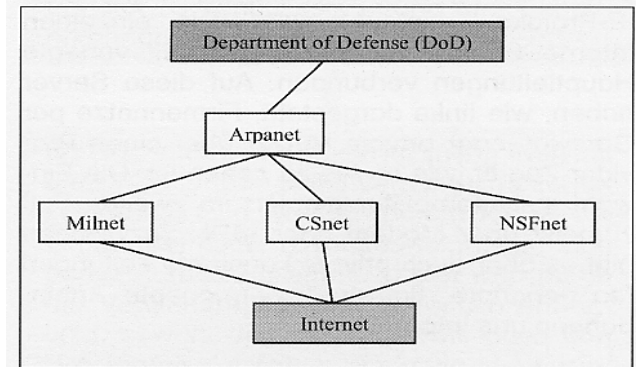


Abb. 1: Bildung des Internets

Das Internet ist, wie Abbildung 1 zeigt, aus den beschriebenen Netzwerken und Netzwerkstrukturen entstanden und zu einem globalen, weltumspannenden Informationssystem zusammengewachsen. In dieser Grafik ist im obersten Bereich das DoD als Auftraggeber für die Entwicklung des Arpa-Netzes dargestellt. In zeitlicher Reihenfolge (von oben nach unten) lässt sich die Entwicklung bis hin zum heutigen Internet verfolgen.

1.2. Aufbau des Internets

Um die Kommunikation zwischen zwei Computern zu ermöglichen, ist spezielle Software erforderlich, die den Datentransfer steuert. Hierbei spricht man vom **Client/Server-Prinzip**. Ein Rechner funktioniert als Client (grob gesagt „Empfänger“), der andere als Server („Sender“). Abb. 2 zeigt dieses Prinzip für das Internet. Ein **Server** bietet Internetdienste an, die dann vom **Client** in Anspruch genommen werden können.

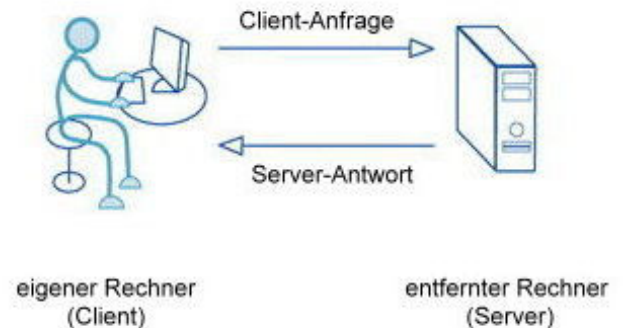


Abb. 2: Das Client/Server-Prinzip

Abbildung 3 stellt schematisch den Internetaufbau dar. Der blau unterlegte Bereich ist „das Internet“, in dem die Kommunikation der einzelnen angeschlossenen Rechner per TCP/IP-Protokoll gesteuert wird. Die einzelnen Internet-Server sind durch weltweit verlegte Hauptleitungen verbunden. Auf diese Server haben, wie links dargestellt, Firmennetze per **Gateway** oder private Nutzer über einen **Provider** Zugriff, wie rechts zu sehen ist. Die Einwahl ins Internet funktioniert im zweiten Fall zumeist über Modem oder ISDN, inzwischen gibt es aber auch private konstante Leitungen (so genannte „**flatrates**“ - permanente Anbindungen ans Internet).

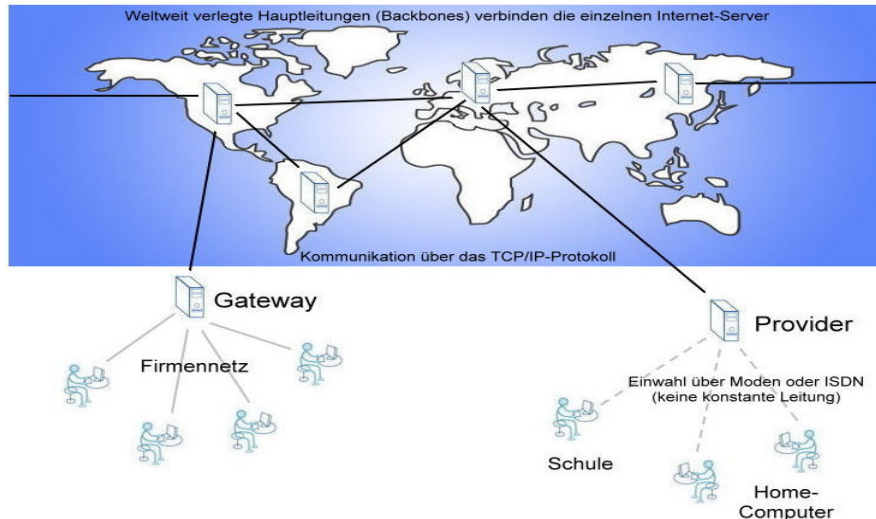


Abb. 3: Internetaufbau

1.3. Verwendung von Protokollen im Internet

Ein Netzwerkprotokoll ermöglicht die Kommunikation zwischen zwei Rechnern. Befinden sich beide Rechner in demselben Netzwerk, so können sie ungehindert miteinander kommunizieren. Ferner legt das Protokoll die Größe der zu übermittelnden Datenpakete fest, steuert die Übertragung der Daten und zeigt den Verlust von Daten an, falls diese ihr Ziel nicht erreichen.

- **TCP:** Das Transmission Control Protocol ist Grundlage für verschiedene Protokolle, so z.B. FTP. TCP ist speziell für Datentransfer, Datenflusssteuerung und Fehlererkennung zuständig.
- **IP:** Das **Internet Protokoll** ist zusammen mit dem TCP für den Datenversand im Internet zuständig. Dabei wird durch die Verwendung von IP-Adressen ermöglicht, sowohl den Absender als auch den Empfänger der Daten eindeutig identifizieren zu.
- **UDP:** Das **User Datagram Protocol** steuert den Transport von Daten ähnlich wie das TCP. Das UDP ist jedoch ein ungesichertes Protokoll (es wird nicht überprüft, ob die Daten das Ziel erreichen; somit besitzt es kein „Fehlermanagement“), es ist verbindungslos und kann somit den Datenfluss auch nicht kontrollieren. Darüber hinaus ist die Menge der zu transferierenden Daten begrenzt.
- **DNS:** Das **Domain Name System** ist ein Verfahren, um die IP-Adressen der Rechner im Internet in Domainnamen zu „verwandeln“. Auf Namens-Servern sind die erforderlichen Daten, IP-Adresse und Domainname gespeichert. Durch diese Server ist es nicht erforderlich, einen Rechner im Internet mit seiner IP-Adresse aufzurufen, sondern man kann auf ihn mittels seines leichter zu merkenden Domainnamens (ergänzt um die Top-Level-Domain) zugreifen.

1.4. Aufbau der Internet-Adresse

Jeder an das Internet angeschlossene Computer besitzt eine **IP-Adresse** bzw. bekommt diese beim Einwählen ins Internet vom Provider zugewiesen, um erkannt zu werden und Kommunikation zu ermöglichen. Diese Adresse darf nur genau einmal im gesamten Netzwerk vergeben sein. Die IP-Adresse besteht aus vier dreistelligen Zahlen (von 0 bis 255), die durch Punkte voneinander getrennt sind (z.B. **194.120.12.110**). Bei der Eingabe von ein- oder zweistelligen Zahlen werden die restlichen Stellen automatisch ergänzt (mit Nullen aufgefüllt).

Das Ansteuern eines speziellen Rechners über seine IP-Adresse ist aufgrund der verwendeten Zahlencodes kompliziert (für jeden anzusteuern den Rechner muss sich der Benutzer bis zu zwölf Ziffern merken). Um diesen Vorgang zu vereinfachen, wurde der DNS begründet. Der IP-Adresse wird ein Name zugeordnet, wobei diese Zuordnung auf den DNS-Servern gespeichert wird (z.B. 194.120.12.110-www.bundestag.de). Durch die Namensvergabe soll die Anwahl der jeweiligen Adresse vereinfacht werden; welcher Benutzer will sich schon 12-stellige Zahlencodes merken, um eine Internet-Präsenz aufsuchen zu können?

Die **DNS-Adresse** wiederum setzt sich aus mindestens drei Komponenten zusammen: **Servername.(Subdomain).Domainname.Top-Level-Domain**.

- Der Servername gibt den Rechner in seiner Funktion an; in www.bundestag.de handelt es sich beispielsweise um einen Rechner, der die Webseiten des Bundestages enthält.
- Der Domainname ist der eigentliche Name; hier z.B. **bundestag** (.de).
- Die Subdomain ist fakultativ anzugeben und stellt eine weitere Untergliederung dar; z. B. www.archiv.bundestag.de.
- Die Top-Level-Domain bezeichnet beispielsweise das Land, aber auch den Bereich (Verwendungszweck), in dem der Rechner registriert ist (**.de** steht für Deutschland; **.tv** für Television).

1.5. Die gebräuchlichsten Dienste im Internet

- **WWW:** Das Web (neben der Bezeichnung WWW die häufigste Kurzform von: World Wide Web) ist einer der Internetdienste, die heute am stärksten Verwendung finden. Dieser Dienst basiert auf Texten und Dokumenten, die miteinander durch Links verknüpft sind (Hypertext).
- **E-Mail:** Das Versenden von E-Mails (Electronic Mails; dt. elektronische Post) ermöglicht den Austausch schriftlicher Nachrichten nahezu ohne zeitliche Verzögerung. Wie bei der analogen Post sind die E-Mails aus einer Adresse (E-Mail-Adresse; z.B. christian.wulff@bundes-tag.de), einem Absender (in Form der eigenen E-Mail-Adresse und weiteren Angaben), einem Betreff und dem Inhalt zusammengesetzt.
- **FTP: (File Transfer Protocol)** FTP bezeichnet zugleich das Protokoll und den damit verbundenen Dienst, Daten im Netz(-werk) von einem auf den anderen Rechner zu übertragen. Mithilfe FTP-Servern werden im Internet „immense Mengen“ verschiedenster Daten zum Download und somit zur Verfügung gestellt.

1.6. HTML und CSS

Mit **HTML** (Hypertext Markup Language) kam im Jahr 1990 das Web. **Hypertext** ist dabei ein System, mit dem Objekte wie Text und Bilder miteinander verknüpft werden können. **Markup Language** ist eine *Auszeichnungssprache* mit der der Inhalt eines Dokumentes beschrieben werden kann. HTML enthält Befehle zum Auszeichnen von vielen Elementen eines Dokuments, wie Überschriften, Textabsätzen, Listen, Tabellen oder Grafiken. Die Definitionen dieser Elemente werden von einem **Browser** gelesen, interpretiert und am Bildschirm sichtbar gemacht. Für die Übertragung der HTML Dokumente im Internet ist **HTTP** (Hypertext Transfer Protocol) zuständig. Als eine Auszeichnungssprache wird HTML nicht programmiert, sondern schlicht geschrieben.

Im Jahr 1994 kam der Netscape Navigator und 1995 der Internet Explorer von Windows auf den Markt. Beide Firmen haben neue markeneigene HTML-Erweiterungen entwickelt, um ihre Marktanteile zu erhöhen. Während des sogenannten Browserkriegs wurden die Unterschiede in der Darstellung von HTML-Code zwischen den zwei Browsern so extrem, dass Webdesigner teilweise zwei auf den jeweiligen Browser angepasste Webseiten erstellen mussten.

Als Antwort auf dieses Problem entwickelte das W3C die Formatierungssprache **CSS (Cascading Style Sheets)**. CSS gilt mittlerweile als Standard und wird von allen gängigen Browsern unterstützt. Mithilfe von CSS können Webseitenentwickler den Inhalt ihrer Webseiten von ihrer Darstellung trennen. Der Inhalt von Webseiten wird in HTML und die Formatierung in CSS geschrieben. Die Erweiterung mit CSS bietet nicht nur mehr Formatierungsmöglichkeiten als HTML, sondern auch andere wichtige Vorteile gegenüber von reinen HTML-Dokumenten:

- *Webseiten werden schneller geladen.*
- *Die Formatierung kann zentral verwaltet werden.*
- *Sowohl der Formatierungscode wie auch der Inhalt von Webseiten bleiben übersichtlich.*
- *Der gleiche Inhalt kann unterschiedlich dargestellt werden.*

Aktuell befinden wir uns in einer Übergangszeit zwischen HTML4.01 und HTML5 und zwischen CSS2.1 und CSS3. Das kann zu einigen Problemen führen. Die Syntax von HTML5 ist noch nicht definitiv festgelegt. Die neuen Befehle werden von den älteren Browser nicht unterstützt, manche der alten Befehle werden in der Zukunft nicht mehr unterstützt. Die **völlig neuen Befehle** werden in diesem Script **blau** markiert. Auf die schon heute veralteten Befehle wird dieser Script verzichtet.

1.7. Aufgaben

1. *Erzähle kurz mit eigenen Worten die Geschichte der Internetentstehung.*
2. *Nenne und beschreibe die oben beschriebenen Hauptprotokolle des Internets.*
3. *Erkläre genau die Fachbegriffe (Nicht alle sind im Text vorgekommen!): Internet, Client, Server, Gateway, Provider, flaterate, Browser, download, upload, URL, Homepage, Domain, HTTP...*
4. *Neben den oben genannten Diensten im Internet gibt es noch eine ganze Menge anderen. Finde und erkläre mindestens drei weitere.*
5. *Erkläre, wie eine Internetadresse aufgebaut ist. Bediene dich dabei des Beispiels: <http://www.cs.uni-dortmund.de/nps/de/Home/Personen/index.html>*
6. *Was versteht man unter HTML und CSS?*

2. Erste Schritte in HTML

2.1. TAGS als Strukturmarker in HTML

HTML-Code besteht aus Text und **Marker, die Tags genannt werden**. Jeder einzelne Marker ist immer in spitze Klammern „<Tag>“ eingefügt. Dabei sind folgenden Grundregeln zu beachten:

a) Alle Tags müssen korrekt geschlossen werden

Jeder Tag besteht aus zwei Teilen, Anfang- und End-Tag, z.B. <title> und </title>. (der End-Tag hat dabei einen zusätzlichen Schrägstrich „/“ vor dem Taginhalt).

- Daneben gibt es noch s.g. **Leer-Tags**. Diese schließen keinen Inhalt ein vereinigen Anfang- und End-Tag zu einem einzigen Tag (z.B. Tag für „harte“ Zeilenumbrüche
)

b) Anfang-Tags können zusätzliche Angaben zur Formatierung besitzen.

Die Wirkung von Tags kann zusätzlich durch die Angaben von Attributen verfeinert werden:

<Tag Attribut1="Wert" Attribut2 ...>...</Tag> Für die Attribute gilt es dabei:

- Es gibt Attribute, denen ein Wert zugewiesen werden muss, (z.B.) sowie Attribute, die keine Wertzuweisung benötigen. (z.B. <ol compact>)
- Mehrere Attribute (mit oder ohne Werte) werden durch Leerzeichen voneinander getrennt.

c) Regeln bei der Schachtelung von Tags

Man verschachtelt keine Tags wie z. B. <i> xxxx xxxx </i>.

Stattdessen sollte man schreiben: <i> xxxx </i> <i> xxx </i>.

d) Keine Unterscheidung von Groß- und Kleinschreibung

Bei der Verwendung von HTML-Tags braucht die Groß- und Kleinschreibung nicht beachtet werden. Die meisten Browser akzeptieren inzwischen sowohl <body> als auch <BODY>, ja sogar <BOdy> oder <BoDy>. *(Man sollte jedoch alles klein schreiben.)*

2.2. Grundgerüst eines HTML-Dokuments

Jedes HTML-Dokument sollte mit der Angabe der HTML-Definition (DOCTYPE) beginnen. Das HTML-Dokument selbst besteht aus zwei wesentlichen Bereichen: Kopfbereich und Rumpfbereich. Beide Bereiche werden vom Tag <html> eingerahmt. Der Kopf des Dokuments enthält Angaben wie den Titel der Seite oder auch Stichwörter für Suchmaschinen (sogenannte Metatags). Der Textkörper enthält den eigentlichen Text des zu erzeugenden Dokuments sowie Grafiken, Tabellen etc., also den Inhalt.

```
<!doctype html>
<html>
  <head>
    <title>
      Titel der Seite
    </title>
  </head>
  <body>
    Text, Graphiken, Tabellen etc.
  </body>
</html>
```


2.3. Textstrukturen

Wenn man einfachen **Fließtext** in einem HTML-Dokument verwendet, formatiert der Browser diesen Text nach bestimmten Regeln.

- Mehrere Leerzeichen und Zeilenumbrüche werden im Browser automatisch zusammengefasst und nur als ein Leerzeichen angezeigt.
- Führende Leerzeichen einer Textzeile werden ignoriert.

Wenn man also eine bestimmte Textstruktur erzielen will, muss man bestimmte Tags nutzen. Früher wurden auch bestimmte **optische Formatierungen** mit HTML-Tags definiert. Solche Tags, wie z.B. für fett, <i> für kursiv, <u> für unterstrichen, sind durch Einführung von CSS überflüssig geworden. Einige von denen, wie z.B. <u> gehören nicht mehr zum HTML5 Standard, der Anwendungsbereich von den anderen z.B. <i> wurde neu definiert. Heute legt man nämlich Wert vor allem auf die **logische Textauszeichnungen**. Bei denen entscheidet der Browser selbst, wie der Text hervorgehoben wird, z.B. kursiv oder fett. Wenn man es will, kann man mit CSS immer noch die gezielte optische Formatierung erzwingen. Folgende Tags werden (heutzutage) von den meisten wichtigen Browsern verwendet.

Befehl	Verwendung	Beschreibung/Standardformatierung
<!-- Anmerkungen -->	Kommentar	Über Kommentare fügt man Erläuterungen hinzu. Ein HTML Kommentar kann Text über mehrere Zeilen umschließen. Kommentare sind nur im Quelltext lesbar. Ein Browser zeigt den Inhalt von Kommentaren nicht an
<h(1-6)> ... </h(1-6)>	Überschriften	Es gibt sechs verschiedene Überschriftgrößen (h1 bis h6).
<p> ... </p>	Textabsatz	Funktioniert ähnlich wie RETURN beim Textverarbeitung
 	Zeilenumbruch	Der Leer-Tag bricht eine Zeile um.
<no br> ... </no br>	Zeilenumbruch verhindern	Der umschlossene Text wird nicht automatisch getrennt.
 ... 	Inline-Blöcke	Kürzere Textabschnitte werden durch den Tag eingeschlossen.
 ... 	Betonter Text	Mit HTML5 soll mit diesem Element ein Abschnitt hervorgehoben werden, der aber keine wichtigere Bedeutung als der gesamte Text hat.
<i> ... </i>	Kursiver Text	Es sollen damit Texte markiert werden, die im Normalfall kursiv dargestellt werden sollen - z. B. Gedanken, Titel usw.
 ... 	Gelöschter Text.	Wird oft als durchgestrichener Text dargestellt.
<adress> ... </adress>	Adressangaben	Kursiv
<blockquote> ... </blockquote>	Längere Zitate	Rechts eingerückt dargestellt
<cite> ... </cite>	Quellangaben	Kursiv
<code> ... </code>	Quelltext	Schreibmaschinenschrift wie z.B. Courier
<dfn> ... </dfn>	Begriffsdefinitionen	Kursiv
 ... 	Betonung	Kursiv
<kbd> ... </kbd>	Tastatureingaben	Schreibmaschinenschrift wie z.B. Courier
<pre> ... </pre>	Vorformatierter Text	Der Text wird samt eingegebener Leerschritte bzw. -zeilen vom Browser übernommen.
<q> ... </q>	Kurze Zitate	Anführungszeichen
<samp> ... </samp>	Beispiele	Schreibmaschinenschrift wie z.B. Courier
_{...}	Tiefgestellter Text	z.B. Normal tiefgestellt
^{...}	Hochgestellter Text	z.B. 10 ³⁰
 ... 	Starke Betonung	Fett
<hr />	Trennlinien	Bisher erzeugte dieses Element eine waagerechte Linie. In HTML5 hat sich daran auch nichts geändert, jedoch symbolisiert die Linie zusätzlich einen thematischen Umbruch zwischen einzelnen Textabschnitten.

2.4. Wichtige Sonderzeichen

Wenn man sicher gehen will, dass alle Sonderzeichen im Browser (z.B. auch in Japan) korrekt ausgegeben werden, sollte man sie auf jeden Fall kodieren.

HTML-Code	Zeichen	HTML-Code	Zeichen	HTML-Code	Zeichen
ä	ä	©	©	µ	μ
ö	ö	°	°	 	„Leerzeichen“
ü	ü	€	€	±	±
ß	ß	½	½	£	£
&ScAuml;	Ä	¼	¼	®	®
Ö	Ö	¾	¾	§	§
Ü	Ü	>	>	²	²
"	"	<	<	³	³

Eine andere Möglichkeit alle Zeichen zu codieren besteht in der Angabe des entsprechenden Wertes des Unicodes. Den Wert kann man in einer dezimalen (z.B. ß für ß) oder in einer hexadezimalen (z.B. ß für ß) Form darstellen.

2.5. Aufgaben

1. Welche Regel muss man bei der Anwendung der HTML-Tags beachten?
2. Im Folgenden wird ein fehlerhaftes HTML-Dokument gezeigt. Gib die gefundenen Fehler an.

```
</html>

<-- Mein Kommentar

<body>

  <head> Willkommen auf der Homepage von HJK

    <title> HJK

  </head>    </title>

</body> >

-- ! >

<html>
```

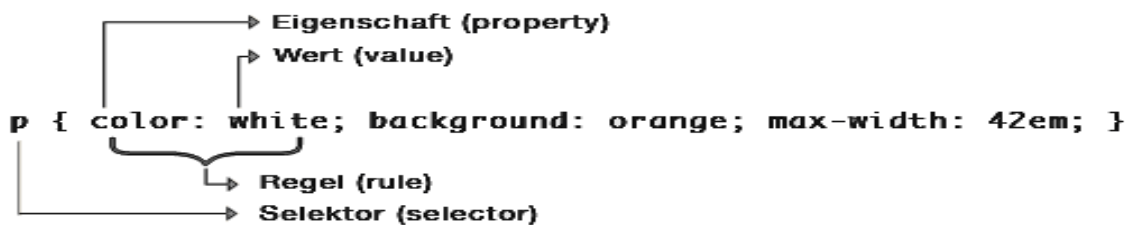
3. Kopiere den Text vom Hermann-Josef Lied in den Notepad-Editor, Speichere ihn als HTML-Datei und Bearbeite ihn auf folgende Weise:
 - i. Definiere jede einzelne Strophe als Absatz
 - ii. Erzeuge die gewünschte Zeilenstruktur.
 - iii. mach alle Nomen **fett**
 - iv. setze alle Adjektive kursiv.
4. Kopiere den Text „Unser Schulprofil“ in den Editor und wandle ihn in ein gut formatiertes HTML-Dokument um.
5. Alle HTML-Dokumente im Notepad-Editor zu schreiben ist sehr mühsam.
 - a. Informiere dich, wie unterscheiden sich die Code- und optische Editoren voneinander.
 - b. Untersuche den bei uns benutzten Editor „Scriptly“.
6. Informiere dich im Internet, welche Inhalte ein Impressum beinhalten sollte. Schreibe die erste Fassung des Impressum-Textes für dein Projekt und erstelle daraus eine HTML-Seite.

3. Erste optische Formatierungen mit CSS

Früher haben Webseitenentwickler die Formatierung ihrer Webseiten direkt in den Tags festgelegt. So wurde z.B. die zentrierte Ausrichtung des Textes in einem Absatz dem Konstrukt `<p align="center">` festgelegt. Wenn man das für 100 `<p>` Tags tun wollte, schrieb man 100-mal `align="center"`. Außerdem wurde damals Inhalt und Layout so eng miteinander vermischt, dass jede Layoutänderung sehr umständlich war. Mit Hilfe von CSS kann man heute Inhalt (rein HTML) von Layout trennen. CSS ist also für Layout der Homepage zuständig. Die sogenannten CSS-Regeln teilen dem Browser mit, wie bestimmte HTML-Elemente formatiert werden sollen.

3.1. Aufbau der CSS-Regeln (Stylesheet-Regeln)

Die Syntax von CSS unterscheidet sich von HTML. Statt Tags kommen Regeln zum Einsatz. Hier legen Sie mit **Selektoren** fest, welche HTML-Elemente Sie formatieren möchten. Die gewünschte Formatierung bestimmt man in einer **Deklaration (Regel)**. Eine Deklaration besteht aus geschweiften Klammern, die mehrere **Eigenschafts-/Wertpaare** beinhalten. Die Eigenschaft wird von ihrem Wert durch einen Doppelpunkt getrennt, und Paare werden mit einem Semikolon voneinander getrennt. Die Syntax sieht wie folgt aus:



Das Semikolon nach dem letzten Eigenschafts-/Wertpaar ist nicht notwendig. Es handelt sich hier jedoch um eine gute Codepraxis. Wie bei HTML sind Leerschritte und Zeilenumbrüche nur im Code ersichtlich. Auch hier ist das Kleinschreiben eine erprobte Praxis. Beim Selektor wird sogar zwischen Groß- und Kleinschreibung unterschieden. Zeilenumbrüche und -eintrückungen im Code verbessern die Lesbarkeit des Codes. So kann man die obere CSS-Regel auch schreiben als:

Selektor	CSS-Eigenschaft	Wert
<code>p</code>	<code>{ color:</code> <code>background:</code> <code>max-width::</code> <code>}</code>	<code>white;</code> <code>orange;</code> <code>42em;</code>

3.2. CSS-Regeln an HTML einbinden

Man kann Stylesheets auf vier verschiedene Arten in Ihr HTML-Dokument einbinden:

- Als Definition für ein bestimmtes Tag z.B. `<p style="color:gray;text-align:center;">`
- Als Definition im Kopf (head) des HTML-Dokuments: `<style type="text/css"> ... </style>`
- Als Definition in einer separaten Datei:

z.B.: `<link rel="stylesheet" type="text/css" href="datei.css" />`

- Als importierte Definition z.B.: `@import url(datei.css);`

Es gilt dabei: Bei den konturierenden Regeln wird die angewandt, die am nächsten dem Ziel formuliert wurde.

3.3. Die einfachen Selektoren und Vererbungsprinzip

Selektoren filtern HTML-Elemente mit Suchmustern, die auch miteinander kombiniert werden können, um aus einfachen Selektoren komplexe Filter zu erzeugen. Die einfachsten davon sind:

Einfache Selektoren	Beispiel	Bezeichnung / Beschreibung
*	<code>* {color: red }</code>	Universal-Selektor Universal selector Alle Element der HTML-Seite
Tag	<code>p { font-family: fantasy }</code>	(Element) Typ-Selektor Type Selectors Alle E-Tags der HTML-Seite
.klasse Tag.klasse	<code>.extra { font-size: 1.6em; }</code> oder <code>p.extra { font-size: 1.6em; }</code>	Klassen-Selektor Class Selectors Alle E-Tags mit dem Attribut class="klasse"
#einzig Tag#einzig	<code>#nav { width: 200px; }</code> oder <code>div#nav { width: 200px; }</code>	ID-Selektor ID selectors Alle HTML-Tags mit dem Attribut id=""

Universal-Selektor

```
* { Eigenschaft: Wert }
```

Hier werden auf einen Schlag alle Elemente des HTML-Dokumentes. Wenn man z.B. schreibt: `* {color: red }`, dann werden alle Elemente rot dargestellt.

(Element) Typ-Selektor

```
Tag { Eigenschaft: Wert }
```

Hier wird ein HTML-Tag einfach mit einer Stylesheet-Angabe verknüpft. Als Selektor wird dabei der HTML-Tag ohne die spitzen Klammern angegeben. Beispielsweise werden mit dem Selektor `p` alle Tags `<p>` und `</p>` im HTML-Dokument angesprochen und die umschlossenen Inhalte entsprechend formatiert.

Möchte man das Format gleichzeitig mehreren HTML-Tags zuweisen, gibt man alle entsprechenden Tags, durch Kommas voneinander getrennt, an. Folgende Angaben sind gleichwertig:

```
h2 {color: black}
h3 {color: black}  oder wenn Sie die Kurzform verwenden: h2,h3,p {color: black}
p  {color: black}                                     h1 {color: blue}
h1 {color: blue}
```

Der Typ-Selektor hilft, die Umformatierung eines HTML-Dokuments vorzunehmen, da man nur den Tag als Selektor und die entsprechenden Eigenschaftswerte angeben muss. Der Nachteil hierbei ist, dass beispielsweise alle Absätze `<p>` gleich formatiert werden. Eine Unterteilung, dass der erste Absatz anders aussieht als der zweite, funktioniert nicht mit dem Typ-Selektor.

Klassen-Selektor

```
.Klassenname { Eigenschaft: Wert }
```

Dieser Selektor ist vermutlich der am häufigsten eingesetzte Selektor. Mit ihm selektiert man keine Tags einer Webseite, sondern man kann die Formatierung über den Klassennamen gezielt einem einzelnen Element zuweisen. Die Definition des Klassenselektors beginnt mit einem Punkt, gefolgt von einer beliebigen Bezeichnung.

Damit ein Klassenselektor angewandt werden kann, muss man jedoch zunächst im HTML – Dokument die Klassen definieren. Das erreicht man, indem man im einen Anfangs-Tag dem Attribut

class den gewünschten Klassen-Namen zuweist. Der gleichen Klasse darf man dabei mehrere, sogar unterschiedlichen Tags zuweisen. Ein Abschnitt vom HTML Dokument kann z.B. so aussehen:

```
<h1 class="himmel"> Himmlische &Uuml;berschriften</h>
<p> Ein normaler Sterbliche kann am Ende</p>
<p class="himmel"> in den Himmel kommen </p>
<p class="naja"> oder auch anderswo gelangen </p>
```

Wenn man nun im CSS- Dokument schreibt:

```
.himmel { color: blue; }
.naja { color: red; }
```

wird Textinhalt der Überschrift und des zweiten Abschnitts blau erscheinen. Der erste Abschnitt wird (normal) schwarz erscheinen und der letzte Abschnitt rot.

Den Klassenselektor kann man auch an einen bestimmten Tag binden. Dazu gibt man den Tag vor dem Punkt an. Die Definition sieht dann folgendermaßen aus:

Tag.Klassenname { Eigenschaft: Wert }

Für den HTML-Tag <p> kann man beispielsweise folgende Formatierung definieren:

```
p.himmel { color: blue; }
```

Wenn man nun diese CSS-Regel auf den o.g. HTML-Beispiel anwendet, wird nur der zweite Abschnitt blau erscheinen (Überschrift bleibt „normal“).

ID-Selektor

ID { Eigenschaft: Wert }
Tag # ID { Eigenschaft: Wert }

Der ID-Selektor ist dem Klassen-Selektor sehr ähnlich. In HTML-Syntax hat man die Möglichkeit, einem Tag über das Attribut **id** eine eindeutige Kennung zuzuweisen z.B. <div id="page">. Da eine ID innerhalb einer Webseite **nur einmal vorkommen darf**, liegt der Hauptunterschied darin, dass Sie mit dem ID-Selektor nur ein bestimmtes Element formatieren können. Den Klassen-Selektor können Sie mehrfach einsetzen.

Vererbung

Das Konzept der Formatierung von Elementen über CSS beruht auf dem Prinzip der Vererbung. Dies bedeutet, dass die Formatierungseigenschaften eines bestimmten HTML-Elements auf seine Unterelemente automatisch weitergegeben werden. Anschaulich wird dieses System anhand des Elemente-Baums einer Webseite.

Der sichtbare Teil einer Webseite befindet sich zwischen den Tags <body> und </body>. Definiert man z. B. für den Tag <body> eine schwarze Schriftfarbe, werden alle textlichen Elemente der Webseite schwarz dargestellt.

Legt man dann für die geordnete Liste eine rote Schrift fest, werden auch die Unterelemente rot dargestellt. Sie haben die Eigenschaft von dem übergeordneten Element geerbt.

```
<html>
  <head>
    <title></title>
  </head>
  <body> (schwarz)
    <h1>(schwarz)</h1>
    <p>(schwarz)
      <b>(schwarz)</b>
    </p>
    <ol> (rot)
      <li>(rot)</li>
      <li>(rot)</li>
    </ol>
    <p>(schwarz)</p>
  </body>
</html>
```

Der Sinn liegt darin, dass man zuerst über die obersten Elemente die Basis-Formatierungen festlegt. Alle Unterelemente werden genau so dar- gestellt, können aber jederzeit durch eine neue Formatierung am entsprechenden Element überschrieben werden. Dann wird ab diesem Element für alle seine Unterelemente die neue Formatierung vererbt.

3.4. Das Aussehen von Text mit CSS gestalten

Man kann verschiedene Eigenschaften Textes mit CSS definieren. Über einen Selektor legt man fest, welches HTML-Element Sie stylen möchten, und dann gibt man ein Eigenschafts/Wertpaar ein.

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
font-family	Arial Comic Sans Courier New Georgia serif für Serifenschriften wie Times New Roman sans-serif für serifenlose Schriften wie Arial	Name der Schriftart bzw. Bezeichnung für eine Schriftfamilie Serifen sind Abschlussstriche von Buchstaben: L L Schrift mit (serif) und ohne Serifen (sans-serif)
font-size	xx-small, x-small, small, medium, large, x-large, xx-large smaller , larger 12px 14pt 150% 1.2em	- Vordefinierte Schriftgrößen - im Bezug auf Maß des Übergeordneten Elementes - Eingabe in Pixel - Eingabe in Punkten - Prozent gegenüber dem übergeordneten Element - x-fache der Größe des übergeordneten Elements
font-style	normal (normal) italic , oblique (kursiv)	Schriftschnitt
color	blue #0000FF 0,0,255	Schriftfarbe
font-variant	normal , small-caps	Kapitälchen
font-weight	normal , bold	Gewicht
text-align	left, right, center, justify	Ausrichtung
text-decoration	none (normale Schreibweise) underline (unterstrichen) line-through (durchgestrichen) overline (Linie über dem Element) blink (blinken)	Texteffekte
text-indent	Wir wünschen Ihnen einen angenehmen und erholsamen Aufenthalt.	Einzug der ersten Zeile
text-transform	none (normale Schreibweise) capitalize (erster Buchstabe groß) lowercase (Kleinschreibung) uppercase (Großschreibung)	Groß / Kleinschreibung erzwingen
line-height	3px) Hotel Vallora } 3 p x	Zeilenabstand
word-spacing	3px Hotel Vallora 3px	Wortabstand
Letter-spacing	3px H o t e l 3px	Zeichenabstand

3.5. Farben in HTML

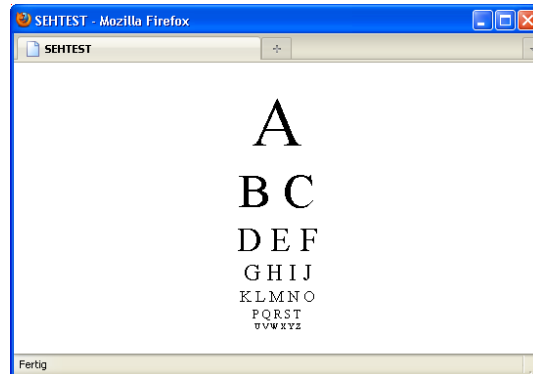
Farbdefinitionen in HTML (im RGB-Modell) entstehen durch die Mischung aus drei Grundfarben: Rot, Grün und Blau (#RRGGBB). Die Intensität jeder Grundfarbe kann dabei den Wert zwischen 0 und 255 annehmen (je größer, desto heller). Da dieser Intensitätsgrad im hexadezimalen System angegeben wird, braucht man für die Intensität einer Grundfarbe nur zwei Zeichen.

Beispiel: #FF0000– für Rot #0000FF – für Blau #000000 – für Schwarz
 #00FF00 – für Grün #FFFFFF – für Weiß

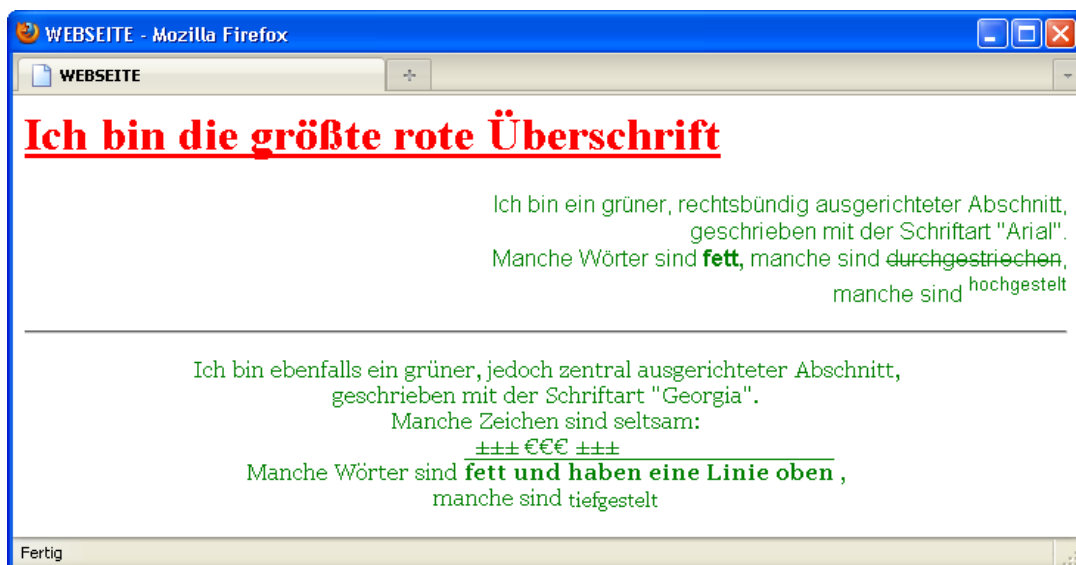
Für manche Farben kann man in HTML darüber hinaus die standardisierte englische Farbnamen benutzen, z.B. black, red, green, blue, white, yellow, aqua, purple,...

3.6. Aufgaben

1. Erkläre den Aufbau der CSS-Regeln.
2. Auf welchen Weisen kann man CSS-Regeln mit dem HTML-Dokument verbinden?
3. Zähle auf, welche Selektoren würdest Du definieren, um folgende Aufgaben zu lösen:
 - a. Du möchtest ein einziges Element der Webseite formatieren. (Zähle 2 Möglichkeiten auf.)
 - b. Du möchtest allen Absätzen das gleiche Format zuweisen.
 - c. Du möchtest unterschiedlich die ungeraden und die geraden Absätze formatieren.
4. Erstelle (mit möglichst wenigen Regeln) eine HTML-Datei, die wie dieser „Sehtest“ aussieht!



5. Erstelle ein HTML – Dokument mit CSS - Regeln für die folgende Webseite. Versuche dabei jeweils so wenig - wie möglich - Regeln anzuwenden.



- a. Schreibe zunächst alle CSS-Regeln innerhalb von `<body>` Bereich des HTML-Dokumentes
 - b. Lagere die CSS-Formatierungen in den `<head>`-Bereich.
 - c. Lagere die CSS-Formatierungen in eine `probe.css` Datei.
6. Öffne die Seite: „P. Franziskus Jordan - Gründer der Salvatorianer“. Erstelle ein HTML/CSS
 7. Schreibe einen passenden Text und erstelle daraus erste „richtige“ HTML-Seite für dein Projekt. Formatiere nach Deinen Geschmack mit CSS das Layout der einzelne Elemente dieser Seite und lagere dann alle CSS-Regeln in eine `elemente.css` Datei aus.
 8. Wenden die CSS-Regeln aus der Datei `elemente.css` auf deine Impressums-Seite an.

4. Online-Grafiken

Bei großen Grafiken wird der Platz auf dem Server stark im Anspruch genommen und die Ladezeiten verlängert sich. Deswegen versucht man, die Größe von Bilddateien zu reduzieren. Trotzdem will man dabei weder (zu große) Qualitätsverluste noch Einsatz-Beschränkungen hinnehmen. Wie weit das gelingen kann, ist unter anderem von der Art der Grafik abhängig.

4.1. Geläufige Grafikformate im Internet.

GIF-Grafiken

- hohe Komprimierungsrate - somit kleine Dateigröße
- maximale Anzahl darstellbarer Farben ist auf 256 begrenzt
- kaum Verlust von Bilddaten durch die Komprimierung - ermöglicht angemessene Darstellung
- Das „interlaced format“ sorgt dafür, dass sich die Grafik während des Ladevorgangs nicht mehr in Einzelzeilen aufbaut, sondern erst unscharf erscheint und mit der Menge der übertragenen Daten immer schärfer wird und besser zu erkennen ist.
- Es lassen sich auch Teile der Grafik ausblenden, indem man sie durchsichtig macht.
- Es lassen sich mehrere Bilder hintereinander hängen, um so eine kleine Animation zu erstellen - die Ladezeiten dieser Dateien werden allerdings länger.

Das GIF-Dateiformat eignet sich für Grafiken mit großen, klaren Flächen und einer überschaubaren Anzahl von Farben.

JPEG-oder JPG-Grafiken

- Farbtiefe der Grafik bleibt bei der Komprimierung erhalten - es gehen jedoch Bildinformationen unwiederbringlich verloren.
- Aus den Farben benachbarter Pixel wird ein Mittelwert gebildet: der neu berechnete Wert ersetzt die vorherigen Pixel, indem sie zu einem Pixel zusammengefasst werden.
- Die entstehenden Dateien können sehr klein werden - eine Komprimierungsrate über 95% kann dazu führen, dass durch fehlende Zwischentöne Teile der Grafik zu großen farbgleichen Blöcken zusammengefasst werden - Komprimierungsrate zwischen 50 und 80 ist sinnvoll
Tipp: Mehrere Versionen mit unterschiedlichen Komprimierungsfaktoren erstellen und die geeignete auswählen.
- Durch den neuen Standard „progressive JPEG“ wird das Bild bereits bei 15% der übertragenen Bilddaten (zunächst undeutlich) angezeigt und mit der Menge der übermittelten Daten immer schärfer und detaillierter abgebildet.
- JPEG kann keine homogenen (einfarbig) Flächen erzeugen bei der Komprimierung werden durch die Interpolation Zwischentöne erzeugt, wodurch keine klar abgrenzbaren Farbbereiche(-flächen) entstehen Übergänge „verschwimmen“ oder „verwaschen“.
- Durch Zwischentöne werden mehr Farben für die Grafik erzeugt - Vergrößerung der Dateigrößen - lange Ladezeiten.

JPEG wird nur für Fotografien und Bilder ohne homogene Flächen eingesetzt.

PNG-Grafiken

- In den 90er-Jahren wurde speziell für das Internet ein neues Format entwickelt, das die Vorteile der bisherigen vereinen, gleichzeitig aber deren Probleme beseitigen sollte.
- Mit dem PNG-Format können Bilddateien verlustfrei komprimiert werden, wobei auch transparente Hintergründe eingefügt werden können (wie bei GIF-Dateien).
- Zugleich besitzt dieses Format die Farbtiefe(-darstellung) wie JPEG-Grafiken.

Nachteil: geringe Komprimierungsrate, da verlustfreie Komprimierung größere Dateien erzeugt

4.2. Einbinden von Grafiken in HTML

Standardmäßig wird eine Grafik im Fließtext wie ein Buchstabe eingefügt. Die Höhe der Graphik bestimmt dabei den Zeilenanstand. Das Einbinden der Grafik erfolgt durch folgende HTML-Befehle.

Tag/Attribute	Werte/Beispiele	Beschreibung
<code></code>	z.B. <code>src="../bild.gif"</code> <code>200px, 30%</code> <code>100px, 20%</code> <code>alt="Unsere Schule"</code>	Leeres Element für die Graphikanbindung - (source) - die URL zu einer -Grafik wird angegeben - Die Angabe der Breite der Grafik (auch mit CSS machbar) - Die Angabe der Höhe der Grafik (auch mit CSS machbar) - Der alternative Text wird angezeigt, wenn die Grafik nicht darstellbar ist.

4.3. Grafiken und Text mit CSS stylen

Graphiken sind nur eine von vielen Elementes-Arten, die man auf einer Homepage einbinden kann. In CSS gibt es viele allgemeine Konzepte, die man nicht nur für Graphiken anwenden kann. Einige von ihnen, wie z.B. Boxmodell (mit der gesamten Rahmenproblematik), werden also später, als ein autonomes Kapitell beschrieben. An dieser Stelle wird nur (als Exempel) besprochen, wie man gezielt eine Graphik im Bezug zu einem benachbarten Text positionieren kann.

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
vertical-align	text-top middle text-bottom	Der direkt vorhergehende und nachfolgender Text kann oben mittig oder unten zur Grafik ausgerichtet werden
float	left right	Horizontale Ausgerichtet der Grafik in einem um diese Graphik fließenden Text. (Bei dem Wert „left“ wird sich z.B. die Graphik links und der Text rechts befinden.)
clear	left right both	beendet das Umfließen andere Elemente. Ein mit clear formatiertes Element ist das Erste, dass nicht mehr neben anderen Elementen steht. Es kann aber trotzdem mit der float Eigenschaft formatiert werden, damit nachfolgende Elemente um dieses Element wieder herum fließen.
margin-left margin-right margin-top margin-bottom	12px 14px 1px 55px	Die Breite von dem linken, rechten, oberen, bzw. unteren äußeren Rand der Graphik. (damit werden die Abstände zum Fließtext festgelegt)

Beispiel:

```
<h2>Pseudo HJK-Latein</h2>
<p style="text-align: justify">
  Fortune plango vulnera... stilantibus ocellis,
  
  quod sua michi munera subtrahit rebellis.
!...
</p>
```

Pseudo HJK-Latein

Fortune plango vulnera... stilantibus ocellis,
quod sua michi munera
subtrahit rebellis. quicquid
enim florui felix et beatus,
nunc a summo corruui gloria
privatus. Fortune rota
volvitur descendo minoratus,
alter in altum tollitur, nimis exaltatus rex sedet
in vertice; caveat ruinam!...



4.4. Mit CSS Elemente mit Hintergrundfarben und -Bildern versehen

Für alle Elemente innerhalb des <body> -Tags können mittels CSS Hintergrundfarben oder auch Hintergrundbilder definiert werden.

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
background-color	blue #0000FF 0,0,255	Hintergrundfarbe wird zugewiesen
background-image	url(Pfad/Graphikname)	Syntax für die Einführung eines Hintergrundbildes
background-repeat	repeat (wiederholen) no-repeat (nicht wiederholen) repeat-x (waagerecht wiederholen) repeat-y (senkrecht wiederholen)	Diese Eigenschaft gibt an, ob oder wie eine Grafik wiederholt wird.
background-position	Wert, z. B. 50px 30px 20% top (horizontale Ausrichtung oben) center (horizontale Ausrichtung mittig) bottom (horizontale Ausrichtung unten) left (vertikale Ausrichtung links) center (vertikale Ausrichtung mittig) right (vertikale Ausrichtung rechts) z. B. top center	Mit dieser Eigenschaft legt man; die Werte und die gewünschte Maßeinheit fest, um die Position des Bilds relativ zur oberen, linken Ecke des Elements anzugeben. Zur Angabe der Positionierung gibt man zuerst den Wert für die horizontale Ausrichtung und anschließend den Wert für die vertikale Ausrichtung an.
background-attachment	scroll fixed	- Beim Scrollen bewegt sich das Bild mit - Hintergrundbild bleibt stehen

Beispiel:

```
<body style=" background-color:#07A99E;  
                background-image: url(hg.jpg);  
                background-repeat: repeat-x;" >
```



4.5. Aufgaben

1. Nenne und beschreibe die typischen Grafikformate im Web.
2. Öffne das in der Aufgabe 3.6.6 erstellte Dokument *pater_jordan.html* und füge dort das Bild von P. Jordan nach dem Vorbild der Schulhomepage.
3. Erstelle ein HTML-Dokumente mit folgenden Eigenschaften:
 - a) Hintergrundfarbe ist gelb und das HJK-Logo fest in der Mitte steht
 - b) Hintergrundfarbe ist blau und das HJK-Logo sich vertikal am rechten Rand wiederholt.
 - c) Hintergrundfarbe ist rot und das HJK-Logo sich horizontal am unteren Rand wiederholt.
4. Auf den meisten Webseiten befinden sich Fotos. Nehme mit einem Fotoapparat, bzw. Handy mindest zwei Bilder für dein Projekt auf. Mit dem Programm Gimp formatiere diese Bilder passend und binde dann diese Bilder an eine Text-Seite deines Projektes (bzw. erstelle eine weitere Seite) so, dass sie die mit dem Text umflossen werden. Gestalte auch passend den Hintergrund dieser HTML-Seite.

5. Multimedia Elemente

Seit der Version 5 kann man sehr einfach Multimediadateien in HTML-Quelltext einbinden. Für uns werden zunächst allein die reinen Audio- und Videodateien von Interesse. Für die „Formatierung“ dieser Elemente benutzt man zurzeit überwiegend einen reinen HTML-Text (ohne CSS)

5.1. Einbindung von Audiodateien mit <audio> Tag.

Der `<audio> ... </audio>` Tag verfügt über folgende Formatierungsmöglichkeiten:

TAG-Eigenschaft	Werte/Beispiele	Beschreibung
<code>autoplay</code>	ohne Wert <code>autoplay</code>	Der Browser wird den Inhalt direkt abgespielt werden, sobald genug geladen ist.
<code>controls</code>	ohne Wert <code>Controls</code>	Der Browser wird Steuerungselemente (Play/Pause/Position/Lautstärke) anzeigen
<code>loop</code>	ohne Wert <code>Loop</code>	If present, the audio will start over again, every time it is finished.
<code>preload</code>	ohne Wert <code>auto</code> <code>metadata</code> <code>none</code>	Gibt an, ob die Audio-geladen werden soll, wenn die Seite geladen wird. (Wird ignoriert, wenn autoplay vorhanden ist.)
<code>src</code>	<code>url</code>	Definiert die URL von der Audio-Datei

5.2. Einbindung von Videodateien mit <video> Tag.

Der `<video> ... </video>` Tag verfügt **zusätzlich** über folgenden Formatierungsmöglichkeiten:

TAG-Eigenschaft	Werte/Beispiele	Beschreibung
<code>audio</code>	<code>muted</code>	Definiert den Grundzustand von Audio. Derzeit werden nur "stumm" ist zulässig
<code>height</code>	<code>pixels</code>	Setzt die Höhe des Video-Players
<code>width</code>	<code>pixels</code>	Legt die Breite des Video-Players

5.3. Inhalte zwischen dem Anfangs- und End-Tag.

1. Da beide o.g. Tags erst ab HTML5 verfügbar sind, sollte man für ältere Browser eine Bemerkung angeben, dass der Browser den audio bzw. Video Tag nicht unterstützt.
2. Da unterschiedliche Browser unterschiedliche Formate von Medial-Dateien unterstützen ist es oft sinnvoll, statt eine Datei mit `src` – Attributt innerhalb eines `<audio>` bzw. `<video>` Tags, mehrere alternativen mit zusätzlichen `<source>` Tags einzubinden.

Beispiel:

```
<video width="320" height="240" controls="controls">
  <source src="movie.ogg" type="video/ogg" />
  <source src="movie.mp4" type="video/mp4" />
  Your browser does not support the video tag.
</video>
```

5.4. Aufgaben

1. *Wie kann man eine Hintergrundmusik an eine Web-Seite anbinden? Unterlege die Seite mit dem Hermann-Josef-Lied, mit der selbst erstellten Audiodatei dieses Liedes.*
2. Erstelle selbst zu deinem Projekt passende Audio- und Video-Dateien. An eine weitere HTML-Seite deines Projektes binde dann diese Multimediadateien an. Probiere die unterschiedlichen Attributbelegungen aus.

6. Listen


6.1. Ungeordnete Listen

Ein häufig benutztes Element zum optischen Auszeichnen oder Hervorheben von Textabsätzen sind unsortierte Listen oder Aufzählung. Eine ungeordnete Liste leitet man mit `` an und schließt man mit `` ab. Jeder einzelne Eintrag wird mit den Tags ` ... ` umschlossen. CSS liefert dazu u.a. folgende Formatierungsmöglichkeiten:

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
list-style-type	none circle square disc disc	- kein Aufzählungszeichen - ungefühlter Kreis, nur Rahmen - gefühltes Quadrat - gefühlter Kreis
List-style-image	url (Wert) z.B. url(burst.png)	- eigene Graphik als Aufzählungszeichen

Beispiel:

```
<ul style="list-style-type: circle">
  <li style="list-style-type: square"> Eintrag 1</li>
  <li style="list-style-type: disc"> Eintrag 2</li>
  <li style="list-style-type: none"> Eintrag 3</li>
  <li> Eintrag 4</li>
  <li style="list-style-image: url(burst.png)"> Eintrag 5</li>
</ul>
```

- Eintrag 1
- Eintrag 2
- Eintrag 3
- Eintrag 4
-  Eintrag 5

6.2. Nummerierte Aufzählungen

Neben Aufzählungen lassen sich auch Nummerierungen zum Hervorheben von Textabsätzen verwenden. Die Nummerierung erfolgt dabei automatisch mit dem verwendeten Typ. Eine solche Nummerierung wird in HTML als sortierte Liste (ordered list) bezeichnet und mit dem Tag `` eingeleitet. Der Browser nummeriert dabei jeden der angegebenen Listeneinträge in aufsteigender Reihenfolge. Wenn man nicht direkt von Vorne anfangen will, kann man dabei dem Attribut `start` (bei ``) bzw. dem Attribut `value` (bei ``) die gewünschte Anfangsstelle zuordnen.

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
list-style-type	decimal lower-alpha upper-alpha lower-roman upper-roman decimal-leading-zero lower-greek	- Dezimalzahlen (1, 2, 3, ...) - Kleinbuchstaben (a, b, c, ...) - Großbuchstaben (A, B, C, ...) - Römische Zahlen, Kleinschreibung (i, ii, iii, ...) - Römische Zahlen, Großschreibung (I, II, III, ...) - Dezimalzahlen mit führender 0 (01, 02, 03, ...) - Kleine, griechische Nummerierung (α , β , χ , ...)

Beispiel:

```
<ol start="4" style="list-style-type: decimal-leading-zero">
  <li style="list-style-type: lower-alpha"> Eintrag 1</li>
  <li style="list-style-type: upper-alpha"> Eintrag 1</li>
  <li value="13"> Eintrag 3</li>
  <li style="list-style-type: lower-roman"> Eintrag 4</li>
  <li style="list-style-type: upper-roman"> Eintrag 5</li>
  <li style="list-style-type: decimal"> Eintrag 6</li>
  <li style="list-style-type: lower-greek"> Eintrag 7</li>
  <li style="list-style-type: upper-greek"> Eintrag 8</li>
</ol>
```

- d. Eintrag 1
- E. Eintrag 1
- 13. Eintrag 3
- xiv. Eintrag 4
- XV. Eintrag 5
- 16. Eintrag 6
- ρ . Eintrag 7
- Σ . Eintrag 8

6.3. Definitionenlisten

Diese Listenart ist insbesondere für Glossare gedacht. Glossare enthalten Paare von Wörtern und deren Erläuterung. Die Einträge bestehen hier aus einem zu definierenden Ausdruck und der zugehörigen Definition. Für die Erstellung solcher Listen verwendet man in HTML die Tags `<dl>` und `</dl>` zum Umschließen des Glossars, `<dt>...</dt>` zur Definition des Fachbegriffs und `<dd> ...</dd>` zur Erläuterung. Alle drei vorgestellten Aufzählungstypen können das zusätzliche Attribut `compact` enthalten. Damit wird eine Liste in einer kompakteren (platzsparenden) Form dargestellt.

```
<h3>Begriffserklärung</h3>
<dl>
  <dt>AA</dt>
  <dd>Auswärtiges Amt</dd>
</dl>
<h3>kompakte Begriffserklärung</h3>
<dl compact>
  <dt>AA</dt>
  <dd>Auswärtiges Amt</dd>
  <dt>BSE</dt>
  <dd>bovine spongiforme Enzephalopathie
    (Rinderwahnsinn)</dd>
  <dt>Wal</dt>
  <dd>ein Säugetier</dd>
</dl>
```



6.4. Geschachtelte Listen

Alle Listenarten können beliebig miteinander verschachtelt sein. Hier ein Beispiel dafür:

```
<h2> Unser Info-
Angebot:</h2>
<dl>
  <dt>Diff 8</dt>
  <dd>Alles, was Spass
macht,</dd>
  <dt>Diff 9</dt>
  <dd>Manchmal muss man
durch:
  <ol>
    <li>Datenbanken</li>
    <li>Web</li>
  </ol>
    <li>MySQL</li>
    <li>PHP</li>
  </ul>
  <li>Basic</li>
</dd>
</dl>
```

Unser Info-Angebot:

Diff 8
Alles, was Spass macht,
Diff 9
Manchmal muss man durch:
1. Datenbanken
2. Web
 o MySQL
 o PHP
3. Basic
zum Ziel kommen

6.5. Aufgaben

1. Worin besteht der Unterschied zwischen `` und `` Listendefinitionen?
2. Wie kann man in einer Listendefinition die Aufzählungszeichen ändern?
3. Gegeben sind folgende Listen. Schreibe den dazu gehörenden Quelltext auf.

- NRW
 - o Steinfeld
 - Schule
 - o Euskirchen
 - Kino
 - Kaufhaus
 - o Kall
- BAYERN

- II. NRW
 1. Steinfeld
 - a. Schule
 - α Physik
 - c. Kloster
 2. Kall
- III. BAYERN

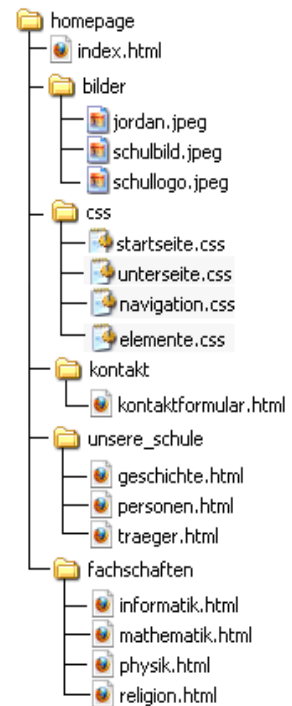
4. Ergänze die erste Text-Seite deines Projektes (bzw. erstelle eine weitere Seite) um eine bzw. zwei Aufzählungen.
5. Erstelle eine weitere Seite deines Projektes mit einer Definitionsliste.

7. Ordnerstruktur und Hyperlinks

7.1. Internetpräsenz in Ordner organisieren

Bereits zu Beginn der Erstellung einer Internetpräsenz muss die spätere Aktualisierung und Wartung bedacht werden. Daher ist es unerlässlich die vielen erzeugten Dateien (HTML, PHP, Grafiken u. a.) in einem logischen Ablagesystem zu verwalten. Die Ablage der Dateien in entsprechenden Ordnern (Grafikdateien, CSS-Dateien, und Multimediaelemente) sollten aus „arbeitstechnischen Gründen“ getrennt von HTML-Dokumenten verweilt werden) sorgt für die nötige Übersicht und erleichtert das erneute Auffinden. Neben der Berücksichtigung der unterschiedlichen Datentypen sollte die Ordnerstruktur auch (mindestens grob) die Inhaltliche Struktur der Internetpräsenz wieder spiegeln.

Die Startseite, also die Seite, die zuerst angezeigt wird, sollte als **index.html** im obersten Verzeichnis der Ablagestruktur liegen. Alle weiteren Dateien können sich in untergeordneten Verzeichnissen befinden.



7.2. Relative und absolute Pfade definieren.

Wenn man in HTML ein Bild, ein Video einbinden will, oder auf ein anderes Dokument verweisen will, muss man zunächst ein Pfad zu dem erwünschten Ziel definieren. Prinzipiell gibt es zwei Adressierungsarten: absolute und relative.

Absolute Adressierung ist unabhängig von der Startposition, von der man die Suche anfängt. Typische absolute Adressen z.B. für die Graphikpfaden sind hier aufgebaut nach dem Muster: `c:/informatik/diff9/homepage/bilder/schullogo.jpeg` (wenn das Bild sich auf eigenem Computer befindet) oder `http://www.domain.de/verzeichnis/bild.jpg` (wenn nach das Bild direkt aus dem Internet holen will).

ACHTUNG: Wenn man bei dieser Art der Adressierung auf die Elemente im eigenen Webauftritt zugreifen will, darf der Speicherort der Homepage nicht verändert werden. Da das jedoch oft erwünscht ist, sollte man in diesem Fall keine absoluten Adressen nutzen.

Relative Adressierung wird verwendet, um auf die anderen Elemente im selben Webauftritt zu verweisen. Relative Pfade weisen den Weg **relativ zum aktuellen Standort** aus. Soll auf einen Element in einem Unterordner verwiesen werden, gibt man zunächst den Namen vom Unterordner gefolgt von dem „/“ Zeichen an. Soll dagegen auf ein Element in einem übergeordneten Ordner verwiesen werden, dann fügt man die Zeichenfolge „../“ an. Z.B.: wenn man in der o.g. Webstruktur im Dokument `religion.html` das Bild `jordan.jpeg` einbinden will muss man dann den Pfad: `src="../../bilder/jordan.jpeg"` angeben.

BEMERKUNG: Mit dem Tag `<base />` im Headbereich kann man absolut die Standardstartposition für die relative Adressierung festlegen.

7.3. Hyperlinks

Verweise im Internet, auch bekannt unter dem Begriff Hyperlinks oder Links, dienen dazu, die Webseiten miteinander zu verbinden. Man kann dabei ein Link zu einem beliebigen HTML-Dokument oder auch direkt zu einer bestimmten Stelle des eigenen oder fremden HTML-Dokumentes erstellen. Für die Festlegung einer konkreten Sprungmarke benutzt man heutzutage nur noch den **ID-Selector** (siehe Kap. 3.3). Wenn sich das Ziel nicht direkt im Ordner der Startposition befindet, muss man davor noch den relativen bzw. absoluten Pfad zum Zielelement setzen. Für die Hyperlinks benutzt man den `<a>` Tag. (a wie englisch „anchor“). Das Attribut `href` (von englisch „hypertext reference“) setzt dabei das Ziel des Links fest. Wenn man auf die Elemente zwischen den Tags `<a>` und `` klickt, springt man zum Ziel. Nun zusammengefasst:

HTML-Tag/Attribute	Werte/Beispiele	Beschreibung
<code>id="Ziel"</code>	<code><p id="Ziel">Sprungmarke</p></code>	- Anfang von diesem Abschnitt wird zur Sprungmarke
<code> Verweistext </code>	<code>http://www.domain.de</code> <code>#Ziel</code> <code>http://www.domain.de#Ziel</code> <code>_blank</code> <code>_self</code> <code>Home</code> <code> </code>	- Haupttag für die Hyperlinks - Verweis auf eine URL im Internet - Verweis auf eine Sprungmarke im eigenem Dokument - Verweis auf eine Sprungmarke eines „fremdes“ Dokuments - Verweisziel wird in einem neuen Fenster geöffnet - Verweisziel wird im aktuellen Fenster geöffnet - Der Sprung wird ausgeführt, wenn man auf einen der Elemente zwischen <code><a></code> und <code></code> klickt

7.4. Aufgaben

1. Erstelle eine sinnvolle Ordnerstruktur für alle bisher von dir erstellten Projekt-Html-Dokumente (analog zum Kap. 7.1) und aktualisiere alle Anbindungen von Bildern, Sounds, Videos und CSS-Dateien, so dass sie funktionieren.
2. Erstelle im Hauptdokument (`index.html`) die Hauptnavigation für alle Unterseiten der Webpräsenz (verwende dazu die ungeordnete Liste).
3. Positioniere auf jeder anderen Html-Seite (oben links) ein Projekt-Logo. Wenn man auf dieses Logo klickt, sollte man außerdem auf die Startseite (`index.html`) springen. Probiere dabei die Wirkung der unterschiedlichen Werte für `target` Attribut und entscheide dich, für eins davon.
4. Erstelle (oder kopiere) einen längeren, zu deinem Projekt passenden Text mit einer inneren Absatz-Struktur (ähnlich wie „Schulprogramm“)
 - a. Wandle den Text zunächst in ein gut formatiertes HTML - Dokument um und speichere es in einem Unterordner deines Projektes.
 - b. Am Anfang dieses Dokumentes erstelle (in der Form einer Liste) ein Inhaltsverzeichnis für die wesentlichen Abschnitte des Textes.
 - c. Positioniere im Dokument geschickt die Sprungmarken und füge danach der Inhaltsverzeichnisliste die Hyperlinks so zu, dass sie als interne Navigation funktioniert.
 - d. Erweitere die Hauptnavigation im Dokument `index.html` um den Eintrag Schulprogramm zusammen mit der internen Navigation für dieses Dokument.

8. Weitere nützliche CSS – Selektoren

Bei der Arbeit allein mit einfachen Selektorarten muss man immer wieder im HTML-Code die Attribute class, bzw. ID einsetzen, was letztlich dem Wunsch nach einer vollständigen Entkopplung der Inhalt von dem Layout widerspricht. Um mit CSS professioneller umgehen zu können, müssen wir also lernen, wie man von Außen (z.B. von einer ausgelagerten CSS-Datei) die CSS-Regeln einfacher, direkter und gezielter den gewünschten Elementen des HTML-Quellcodes zuordnen kann. Dazu werden wir nun weitere Selektorarten kennen (und schätzen) lernen.

8.1. Kontext-Selektoren

Kontext Selektoren	Beispiel	Bezeichnung / Beschreibung
E F	<code>a img { border: 4 px solid blue; }</code> Alle Bilder innerhalb von a-Tags	Nachfahren-Selektor Alle F-Tags, die innerhalb eines E-Tags liegen auch in zweiter oder dritter Ebene
E > F	<code>div>table { border:thin green; }</code> Alle table-Elemente, die direkt innerhalb eines div-Elements liegen	Kind-Selektor: Alle F-Tags, die direkter Nachfahre des E-Tags sind
E + F	<code>h5 + p { font-weight: bold; }</code> Das p-Element, das direkt auf h5 folgt	Nachbar-Selektor: Das F-Element, das unmittelbar hinter dem E-Tag folgt.
E ~ F	<code>h5 ~ p { font-size: smaller; }</code> Alle p-Elemente, die der Überschrift h5 folgen	Indirekter Nachbar-Selektor Alle F-Tags, die dem E-Tag folgen.

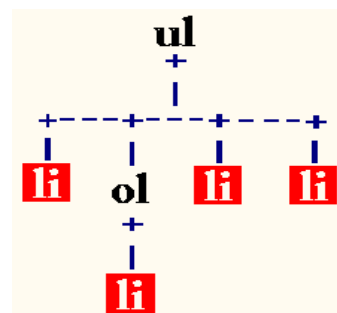
Nachfahre-Selektor

```
Selektor1 Selector2 { Eigenschaft: Wert }
```

Mit diesem Selektor ist es möglich, ein Stylesheet nur dann anwenden zu lassen, wenn mehrere Selektoren, gleichzeitig auftreten.

```
ul li { color: white; background: red; }
```

Alle li-Elemente, die innerhalb von ul-Elementen liegen, werden durch die Regel angesprochen.



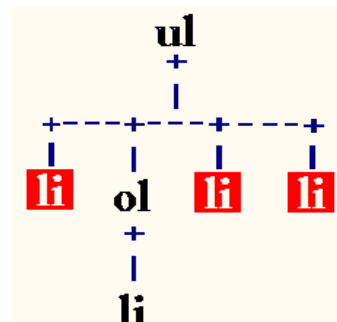
Kind-Selektor

```
Selektor1 > Selector2 { Eigenschaft: Wert }
```

Der Kind-Selektor funktioniert ähnlich den Nachfahre-Selektor. Der Unterschied besteht darin, dass beim Kind-Selektor die entsprechende Tags direkte Nachfolger des vorherigen Tags sein müssen.

```
ul > li { color: white; background: red; }
```

trifft nur auf li-Elemente zu, die direkte „Kinder“ vom ul-Element sind.



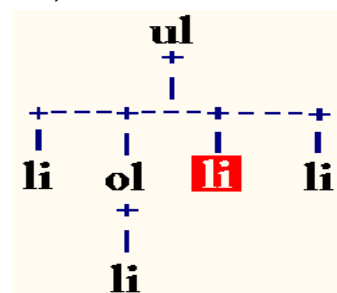
Direkter Nachbar (Geschwister) – Selektor (!!! Probleme mit CHROME!!!)

```
Selektor1 + Selector2 { Eigenschaft: Wert }
```

wählt ein Element aus, welches von demselben Väterelement stammt und sich in unmittelbarer Nachbarschaft befindet. Zum Beispiel:

```
ol + li { color: white; background: red; }
```

li - Element folgt direkt dem ol - Element nach und beide haben dabei denselben Vater.



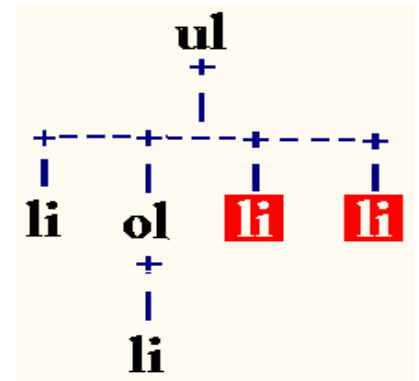
Allgemeiner Nachbar (Geschwister) – Selektor

```
Selektor1 ~ Selector2 { Eigenschaft: Wert }
```

CSS3 bringt mit sich noch einen allgemeinen Geschwister-Selektor. Dieser Selektor trifft auf beliebige Geschwisterelemente des betreffenden Elementes eines bestimmten Typs zu.

```
ol ~ li { color: white; background: red; }
```

Indirekter Nachbar-Selektor, angewendet auf alle li-Elemente, die dem ol-Element folgen. ol und li haben dabei immer dieselben Eltern – liegen also in derselben Ebene.



Kombination mit Klassen und ID-Selektoren

Auch diese Kontext-Selektoren können z.B. durch Klassen- oder id-Selektoren noch weiter eingeschränkt werden.

```
ul#nurHier li.content { color: white; background: green; width: 100px; }
```

Nur Inhalte von li-Elementen mit dem Attribut class="content", die innerhalb des ul-Elements mit dem id-Attribut id="nurHier" liegen, werden den aufgeführten Regeln unterworfen.

8.2. Attribut-Selektoren

Attribut Selektoren	Beispiel	Bezeichnung / Beschreibung
E [attr]	td[width] { background: red; }	Alle E-Elemente, deren "attr"-Attribut gesetzt ist (gleich, mit welchem Wert).
E [attr="xyz"]	input[type="password"] {background: red; }	Alle E-Elemente, deren "attr"-Attribut exakt den Wert "xyz" aufweist

```
[Attribut] { Eigenschaft: Wert; }  
Tag [Attribut] { Eigenschaft: Wert; }  
Tag [Attribut="Wert"] { Eigenschaft: Wert; }
```

Attribut-Selektoren dienen zum Selektieren von Elementen auf Basis eines Attributs oder eines bestimmten Wertes eines Attributs. Sie sind an eckigen Klammern zu erkennen, zum Beispiel trifft

```
[href] { background: red; }
```

auf alle Elemente zu, die über das href-Attribut verfügen (der Universalselektor * wird implizit angenommen).

Attribut-Selektoren lassen sich auch sowohl auf bestimmte Elementtypen als auch auf solche Elemente einschränken, die den vorgegebenen Wert des betreffenden Attributs aufweisen. So trifft zum Beispiel die folgende Regel:

```
a[href="http://www.hjk-stinfeld.de"] {background: pink;}
```

nur auf Hyperlinks, die auf unsere Schulhomepage leiten.

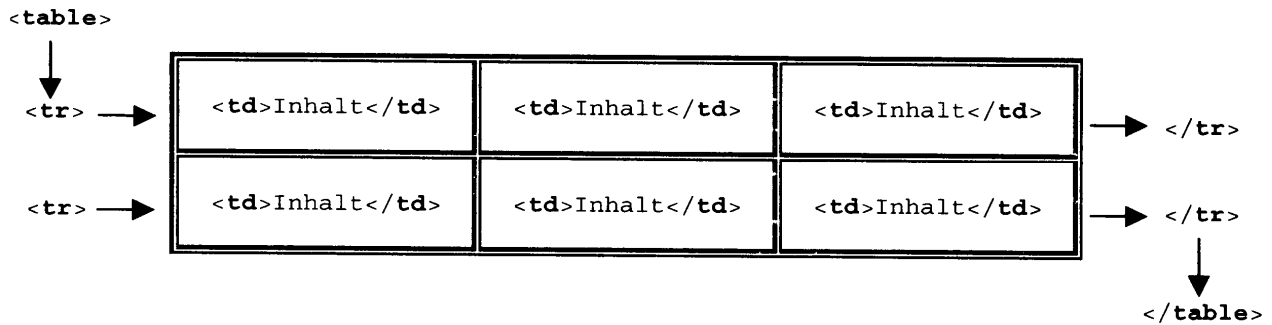
8.3. Aufgaben

1. Im index.html -Dokument befindet sich eine längere „Navigationsliste. Formatiere (nach deinem Geschmack) layout dieser Liste mithilfe der in diesem Kapitel kennengelernten Selektorenarten.

9. Tabellen

Tabellen sind ein wichtiges Hilfs- und Gestaltungsmittel, um tabellarische Daten in einer Gitterstruktur übersichtlich darzustellen.

9.1. Der Grundlegende Aufbau einer Tabelle



HTML-Tag/Attribute	Werte/Beispiele	Beschreibung
<code><table border = ? width = ? > </table></code>	<code>border = „2“</code> <code>width = „300“</code> <code>width = „80%“</code>	<ul style="list-style-type: none">- Leitet eine Tabellendefinition ein.- <code>border</code> legt die Dicke der Umrandung in Pixel fest. (In Html5 nicht mehr unterstützt, fürs Anfang jedoch sehr hilfreich)- Legt die Breite der Tabelle in Pixel bzw. in Prozentangaben, wobei 100% der Breite des Gesamtfensters entspricht. (In Html5 nicht mehr unterstützt, fürs Anfang jedoch sehr hilfreich)
<code><tr height = ?> <td width = „20“> Inhalt </td> </tr></code>	<code>height = „40“</code>	<ul style="list-style-type: none">- Definition einer Zeile (table row)- Angabe der Zeilenhöhe in Pixel- Definition einer Zelle (table data) evtl. mit Breitenangabe

Tabellenzellen dürfen auch leer sein. Die Browser stellen jedoch oft leere Zellen nicht dar. In diesem Fall sollte man die leeren Zellen mit mindestens einem Leerzeichen (` `) füllen.

9.2. Spalten- und Tabellenüberschriften

Tabellen enthalten häufig in der ersten Zeile Zellen mit **Spaltenüberschriften**. Diese Zellen kann man mit dem Tag `<th>` (table header) festlegen. Spaltenüberschriften werden standardmäßig fett hervorgehoben und zentriert dargestellt.

Mithilfe des Tags `<caption>` kann man zusätzlich eine **Tabellenüberschrift** der Tabelle zufügen.

9.3. Bereiche einer Tabelle organisieren

HTML erlaubt, die Zeilen einer Tabelle nach Bedarf logisch in **Kopf-, Rumpf- und Fußbereichen** zu organisieren. Logische bedeutet hierbei, dass keine besondere Formatierung durch den Browser durchgeführt, sondern die Tabelle in einzelne zusammengehörende Bereich aufgeteilt wird. Standardmäßig sind alle Elemente einer Tabelle als Rumpf anzusehen. Der Vorteil der Aufteilung der Tabellenelemente in Kopf, Fuß und Körper liegen darin, dass die verschiedenen Bereiche mithilfe CSS unterschiedlich formatiert werden können.

Die entsprechenden Tags heißen `<thead>`, `<tfoot>` und `<tbody>`. Die optionale Angabe von `<thead>` und `<tfoot>` muss vor dem Tag `<tbody>` erfolgen!!!

9.4. Spalten organisieren

Über die Tags `<col />` und `<colgroup>` kann man dem Browser gleich zu Beginn einer Tabellendefinition mitteilen, wie viele Spalten die folgende Tabelle haben wird. Dadurch kann der Browser die Tabelle schon während des Ladens aufbauen, und die Anzeige der Webseite erfolgt schneller.

Man kann die Spalten in Gruppen zusammenfassen, um sie als Einheit festzulegen. Diese Gruppen können dann Attribute erhalten, die sich auf alle Spalten der Gruppe beziehen. Dies vereinfacht die Konfiguration mehrerer Spalten mit gleichen Eigenschaften.

Für Tabellen mit vielen Spalten ist die Verwendung des Attributs **span** von Interesse. Es weist mehreren aufeinanderfolgenden Spalten dieselben Eigenschaften zu.

9.5. Spalten und Zeilen verbinden

Wenn man komplexere Tabellenstrukturen erzeugen will, kann man Zellen miteinander verbinden.

- Wenn man mehrere Spalten miteinander verbinden will, setzt innerhalb des Tags `<td>` bzw. `<th>` das Attribut **colspan** (column span – Spalten überbrücken) und weist man ihm als Wert die Anzahl der Spalten an, die man vereinigen will. (z.B.: **colspan** = „2“)
- Wenn man mehrere Zeilen miteinander verbinden will, setzt dagegen das Attribut **rowspan** (row span – Zeilen überbrücken) und weist man ihm als Wert die Anzahl der Zeilen an, die man vereinigen will. (z.B.: **rowspan** = „2“)
- Man kann natürlich auch gleichzeitig sowohl die Spalten, wie auch Zeilen verbinden. Dann setzt man beide Attribute innerhalb eines Tags.

9.6. Ein komplexeres Beispiel einer Tabelle

```
<table border="1" >
  <caption> Kunst </caption>
  <colgroup><col width="150" style="background-color: lime" /><col width="200" /></colgroup>
  <colgroup width="250" span="2" style="background-color: yellow"></colgroup>
  <thead>
    <tr><th> Spalte 1 </th><th> Spalte 2 </th><th> Spalte 3 </th><th> Spalte 4 </th></tr>
  </thead>
  <tfoot style="background-color: aqua; text-align: center">
    <tr><td colspan="4">Fußzeile</td></tr>
  </tfoot>
  <tbody>
    <tr><td rowspan="3">&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td><td>&nbsp;</td></tr>
    <tr><td>&nbsp;</td><td rowspan="2" colspan="2">&nbsp;</td></tr>
    <tr><td>&nbsp;</td></tr>
  </tbody>
</table>
```

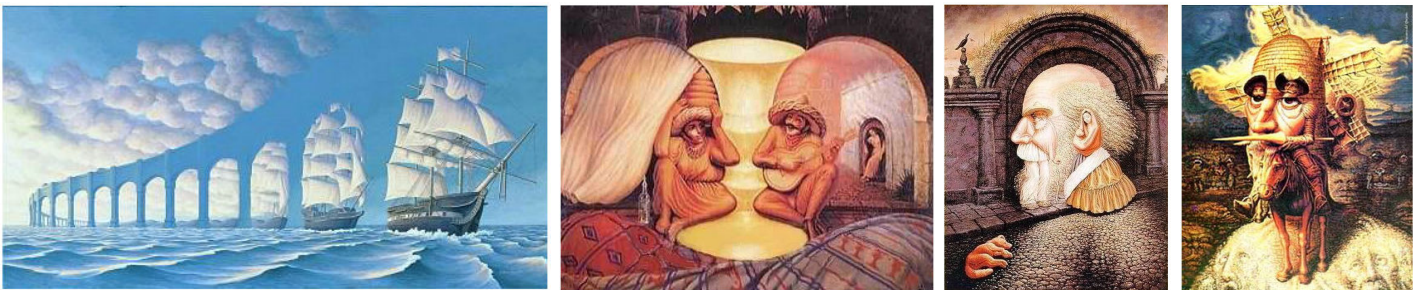
Spalte 1	Spalte 2	Spalte 3	Spalte 4
Fußzeile			

9.7. Aufgaben

1. Erstelle folgende Tabelle mit Gesamthöhe von 80% (500 px) und Gesamtbreite von 50% (300 px). Untersuche dabei die Funktionsweise von CSS Table Attribut `border-collapse: collapse` und `border-collapse: separate`.

Breite: 20% Höhe: 10%	Breite: 50% Höhe: 10%	Breite: 30% Höhe: 10%
Breite: 20% Höhe: 30%	Breite: 50% Höhe: 30%	Breite: 30% Höhe: 30%
Breite: 20% Höhe: 40%	Breite: 50% Höhe: 40%	Breite: 30% Höhe: 40%
Breite: 20% Höhe: 20%	Breite: 50% Höhe: 20%	Breite: 30% Höhe: 20%

2. Im Ordner findest du insgesamt 4 „zerstückelte“ Grafiken. Erstelle eine HTML-Datei, die 4 passende Tabellen enthält. Füge die einzelne Teilgrafiken so in diese Tabellen ein, dass die gesamten Grafiken lückenlos dargestellt werden. (Tipp: „`border-collapse: collapse`“, für die Tabelle und „`padding: 0px`“ für jede Zelle)



3. Erstelle Tabellen, die folgende Strukturen aufweisen.

Zeile 1	<td rowspan=4>
Zeile 2	
Zeile 3	
Zeile 4	

Zeile 1	<td rowspan=2>
<td rowspan=2>	
Zeile 4	<td rowspan=2>

Zeile 1	Zeile 1	Zeile 2
Zeile 2	? ? ?	
Zeile 3		
Zeile 4		

4. Erstelle eine Tabelle für Deinen Stundenplan

Stundenplan						
Einheit	Zeit	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
1. Stunde	8:00-8:45	Informatik EF		Mathematik 9c	Informatik Diff-9	Physik 9c
2. Stunde	8:45-9:15	Informatik EF		Informatik Diff-8	Physik EF	Mathematik 9c
1. Pause	9:15-9:30					
3. Stunde	9:30-10:15			Informatik Diff-8	Physik EF	Mathematik EF
4. Stunde	10:15-11:00	Physik 9c	Informatik Diff-9	Mathematik EF	Informatik EF	
2. Pause	11:00-11:20					
5. Stunde	11:20-12:05	Informatik Diff-8	Informatik 11	Mathematik EF	Informatik 13	Informatik 13
6. Stunde	12:05-12:50	Mathematik 9c	Informatik 11		Informatik 13	Informatik Diff-9
Mittagspause	12:50-13:40					
7-8. Stunde (A)	13:40-15:10				Informatik 11	
7-8. Stunde (B)	13:40-15:10					Physik EF
Giltig ab 01.02.2011						

5. Um die Bilder in einem HTML-Dokument gezielt zu positionieren verwendet man oft eine Tabelle. Erstelle für dein Projekt so eine Tabelle, in der sowohl die Bilder wie auch Textelemente vorhanden sind.

10. CSS – Pseudoformate und deren Anwendung

Durch Pseudoklassen und Pseudoelemente lassen sich spezielle Zustände oder Bereiche von Elementen durch eigene Selektoren ansprechen. Durch **Pseudoklassen** werden die Elemente angesprochen, die über bestimmte Merkmale verfügen. Eine Pseudoklasse kann pauschal für alle Elemente festgelegt werden, die z.B. das erste Kind eines Elternelements sind. **Pseudoelemente** dagegen sind bestimmte Bereiche, die nicht von der Auszeichnungssprache erfasst, aber dennoch mit Stilangaben versehen werden können, z.B. die erste Textzeile in einem Element. Pseudo-Elemente sind immer von einem Selektor abhängig und. Um Pseudoelemente besser von Pseudoklassen unterscheiden zu können sieht CSS 3 vor, dass Pseudoklassen mit einem, Pseudoelemente mit zwei Doppelpunkten markiert werden. In CSS2 markiert man beide mit einfachem Doppelpunkt.

	:Pseudo-Klasse	{	Eigenschaft: Wert	}
Selektor:	Pseudo-Klasse	{	Eigenschaft: Wert	}
Selektor:	Pseudo-Element	{	Eigenschaft: Wert	}

10.1. Wichtigste Pseudoklassen

Pseudoformate	Beispiel	Bezeichnung / Beschreibung
E:link	<code>a:link { color: blue; }</code>	Alle Elemente E, die einen Hyperlink zu einem noch nicht besuchten Ziel darstellen
E:visited	<code>a:visited { color: purple; }</code>	Alle Elemente E, die einen Hyperlink zu einem noch nicht besuchten Ziel darstellen
E:active	<code>a:active { background: #8FBC8F }</code>	Dynamischer Pseudolink E-Elemente während das Element aktiviert ist (z.B. Maustaste auf einen Link gedrückt ist)
E:hover	<code>li:hover { background: #8FBC8F }</code>	Dynamischer Pseudolink: E-Elemente während die Maus über dem Element hovers
E:focus	<code>input:focus { background: red }</code>	Dynamischer Pseudolink: E-Elemente während der Fokus auf dem Element liegt
E:first-child	<code>td:first-child { color: green; }</code>	Erstes Kind eines Eltern-Elements
E:last-child	<code>li:last-child { border-bottom: 1px }</code>	wie :first-child, aber filtert vom letzten Element an
E:nth-child	<code>tr:nth-child(2) { color: pink }</code>	filtert das n-te Kindelement eines HTML-Elements
E:nth-last-child	<code>tr:nth-last-child(2) { color: blue }</code>	wie nth-child, aber beginnt am Ende
E:nth-of-type	<code>p:nth-of-type(odd) { color: black }</code>	wie nth-child, wirkt nur bei Elementen eines Tpps
E:nth-last-of-type	<code>p:nth-last-of-type(odd) { color: red }</code>	wie nth-of-type, aber beginnt am Ende
E:empty	<code>td:empty { background: gray }</code>	ein Element ohne Inhalt

10.2. Wichtigste Pseudoelemente

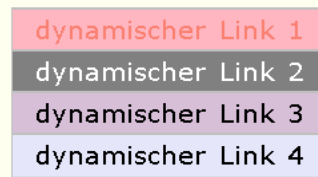
Pseudoelemente	Beispiel	Bezeichnung / Beschreibung
E:first-line	<code>p:first-line { color: purple }</code>	Erste Zeile in einem Text
E:first-letter	<code>p:first-letter { color: purple }</code>	Erstes Zeichen in einem Text
E:before, E:after	<code>td.preis:before { content: " + " }</code> <code>td.preis:after { content: " € " }</code>	:before und :after fügen Inhalte automatisch vor oder nach einem dynamisch generierten Element ein.

10.3. Beispiele für die Anwendung der Pseudoformate

Pseudolinks

Das Abfangen der verschiedenen Zustände eines Links wurde durch Link-Pseudoklassen bewältigt, die schnell von allen Browsern unterstützt wurden.

```
a:link      { background: thistle; }
a:visited  { background: lavender; }
a:hover    { background: pink; }
a:focus    { background: lightpink; }
a:active   { background: plum; }
```



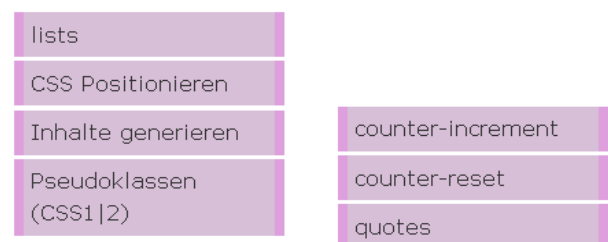
Die Reihenfolge der Pseudoselektoren muss wie hier: `link`, `:visited`, `:hover`, `:focus`, `:active` sein. Ansonsten überschreibt z.B. `:visited` den Pseudoselektor `:hover`.

CSS2 erlaubt eine weitere Differenzierung, bei der die Pseudoklassen hintereinander geschaltet sind. Wenn der Browser diese Funktionalität unterstützt, muss der Hintergrund eines bereits besuchten Links blau werden und seine Schrift weiß, wenn die Maus über ihm `hovers`:

```
a:visited:hover { background: gray; color: white }
```

drop-down Navigation mit Hilfe von `:hover`

Heute unterstützen die meisten Browser das Ändern einer CSS-Eigenschaft beim Hovern mit der Maus nicht mehr nur bei `a`-Tags. So kommen wir dann mit CSS an eine aufpoppende Navigation ganz ohne Javascript.



CSS- Regeln

```
ul.col ul { display: none; }
ul.col li:hover ul {
    display: block;
    position: absolute;
    left: 14em; z-index: 10;
    margin-top: -2em;
    width: 12em;
    list-style: none;
}
```

HTML- Quell text

```
<ul class="col">
  <li>lists</li>
  <li> CSS Positionieren </li>
  <li> Inhalte generieren
    <ul>
      <li>counter-increment</li>
      <li>counter-reset</li>
      <li>quotes</li>
    </ul>
  </li>
  <li> Pseudoklassen (CSS1|2)</li>
</ul>
```

`:first-line`, `:first-letter`, `:first-child`

Die Pseudoklasse: `first-child` selektiert das erste Kind eines Elternelements – gleich, um welches Element es sich handelt. `:first-line` formatiert die erste Zeile in einem Text `:first-letter` formatiert den ersten Buchstaben eines Textes

```
p:first-line { color: purple; font-weight: bold;}
p:first-letter { color: mediumvioletred;
                font-size: x-large; float: left; }
p:first-child { color: mediumvioletred;}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin eu augue. Mauris bibendum, justo vitae molestie malesuada, leo turpis pulvinar massa, et suscipit pede libero sit amet orci.

Donec fringilla diam vulputate nibh luctus rutrum. Suspendisse potenti. In vitae ante. Mauris nunc. Morbi sed nulla. Proin congue lacinia tellus. Sed ligula. Aenean metus.

before: und :after :: Inhalte per CSS einfügen

Die CSS-Eigenschaft *content* fügt automatisch die wiederkehrenden Phrasen »€« und »zzgl. MwSt« vor und nach dem Preis in eine Preisliste ein und verringert den Schreibaufwand.

```
td.preis:before { content: " € " }
td.preis:after { content: " zzgl. MwSt " }
<table>
  <tr><td>Rose Rugora</td> <td class="preis">65</td></tr>
  <tr><td>Tuplpe Melanie </td> <td class="preis">80</td></tr>
  <tr><td>Tauprinzessin</td> <td class="preis">19</td></tr>
  <tr><td>Sommernachtstraum</td> <td class="preis">27</td></tr>
  <tr><td>Bianca</td> <td class="preis">91</td></tr>
</table>
```

Rose Rugora	€ 65 zzgl. MwSt
Tulpe Melanie	€ 80 zzgl. MwSt
Tauprinzessin	€ 19 zzgl. MwSt
Sommernachtstraum	€ 27 zzgl. MwSt
Bianca	€ 91 zzgl. MwSt

:nth-child(odd) und nth-child(even)

Tabellenzeilen lassen sich mit einem geringen Aufwand abwechselnd einfärben – das erleichtert in langen Tabellen die Lesbarkeit.

```
.exc tr:nth-child(odd) { background: lavender; }
.exc tr:nth-child(even) { background: thistle; }
...
<table class="exc">
```

Klassen- und Id-Selektoren	Die ältesten und zuverlässigsten Selektoren
Pseudoelemente CSS1 und CSS2	Insbesondere für die Gestaltung von Links in bestimmten Zuständen verwendet
Pseudoelemente CSS3	Besonders interessant für Listen und Tabellen
Attribut-Selektoren	Filtern HTML-Elemente anhand ihrer Attribute
Kontext-Selektoren	Wählen HTML-Elemente anhand ihrer relativen Position innerhalb der Struktur

:empty

Der Selektor *:empty* filtert leere Tabellenzellen.

```
td:empty { background: gainsboro; }
```

Schwebendes Element
Zwingt das Element an den links bzw. rechten Rand
Position eines absolut positionierten Elements
Position in der dritten Dimension an

10.4. Aufgaben

1. Erstelle für dein Projekt eine längere Tabelle. Mit *:nth-child(odd)* und *nth-child(even)* kann man unterschiedlich die geraden und die ungeraden Zeile einer Tabelle formatieren. Mit *tr :hover* kann einen farbigen Hintergrund der Tabellenzeilen erhalten, wenn die Maus über ihnen hovers. Das sollte man nutzen. Auf der Seite <http://icant.co.uk/csstablegallery/> findest du eine ganze Menge von CSS- TABLE DESIGNS. Erstelle nach diesen Mustern selbst eins, oder binde eine Vorlage in deine Homepage ein. (BEACHT E DABEI ALLE COPYRIGHTS!!!) Verlagere dein CSS- TABLE DESIGN in das Dokument `elemente.css`
2. Programmiere ein vertikales und ein horizontales CSS-drop-down-menu für Deine Homepagenavigation. Entweder machst du es allein (schwierig aber auch für dich machbar) oder verwendest du die Vorlagen. Auf der Seite <http://www.cssplay.co.uk/menus/> findest du eine ganze Menge von CSS-drop-down-menüs. Finde eine, die Dir besonders gefällt und binde sie in deine Homepage (Genau gesagt an deine Dokumente: `index.html` und `unterseite.html`). BEACHT E DABEI ALLE COPYRIGHTS!!! Verlagere die CSS- DATEIEN in ein (oder zwei) Dokument (e) `navigation.css` (`navigation_hor.css`, `navigation_ver.css`)

11. Layout einer Webseite

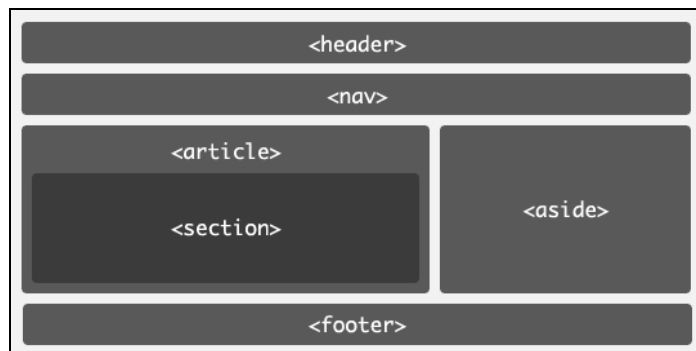
11.1. Webseiten-Bereiche (Sectioning)

Für die Strukturierung des Rupfbereiches sind folgende HTML Konstrukte von Bedeutung:

Bisherige Vorgehensweise



Vorgehensweise in HTML5



HTML-Tag	Verwendung	Beschreibung
<code><div > ... </div></code>	Allgemeine Bereiche (Blöcke)	Ein Bereich (Block) kann in einem HTML-Dokument mehrere, unterschiedliche Objekte beinhalten.
<code><header> ... </header></code>	Für den Kopfbereich	Kann typischerweise den Titel des Dokuments, Logos, ein Formular zur Schnellsuche oder ein Inhaltsverzeichnis enthalten
<code><nav> ... </nav></code>	Für die Hauptnavigation	Navigation einer Webseite
<code><article> ... </article></code>	Für die Hauptinhalte der Seite	Article-Blöcke beheimaten größere zusammenhängende voneinander unabhängige Inhaltsblöcke.
<code><section> ... </section></code>	Logische Bereiche	article-Blöcke können mit <section> in mehrere Abschnitte unterteilt werden und sind außerdem schachtelbar.
<code><aside> ... </aside></code>	Für den Marginalbereich	Ein klassischer Fall für Sidebars (Randfenster), aber auch für inhaltliche Einschübe in einem article.
<code><footer> ... </footer></code>	Für den Fußbereich eines Dokuments findet:	Z.B. für Autor- und Copyright-Informationen. „Footer“ kann, aber muss nicht notwendigerweise am Ende eines Dokuments stehen

11.2. Anzeigeart der Elemente

Wie ein HTML-Element im Browser dargestellt wird, hängt zunächst von dem „Content-Modell“, dem ein Element zugehört. HTML4 unterscheidet dabei zwei Basis-Content-Modelle (Die neuen Modelle von HTML5 werden wir nicht behandeln)

- **Block-Elemente** sind Elemente mit Inhalt, die eine neue Zeile im Textfluss und einen Block erzeugen. Textabsätze gehören dazu, Überschriften, Listen, Trennlinien, Tabellen usw.
- **Inline-Elemente** sind Elemente, die keine neue Zeile im Textfluss erzeugen und reine Auszeichnungen innerhalb von Text sind.

Mit CSS - Eigenschaft **display**: können man jedoch die „angeborene“ Anzeigeart von Elementen unterdrücken und selbst die Art der Anzeige festlegen. Folgende Angaben sind (u.a.) möglich:

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
display	none block inline inline-block compact table table-row table-cell	<ul style="list-style-type: none">- Element wird nicht angezeigt- Erzwingt einen Block - das Element erzeugt eine neue Zeile- Das Element wird im laufenden Textfluss angezeigt- Erzeugt einen Block, belässt das Element jedoch im Textfluss.- Kompakte Darstellung des Elementes- Das Element enthält tabellarisch angeordnete Kindelemente und erzeugt eine neue Zeile. Wirkt wie das table-Element in HTML- Das Element enthält nebeneinander angeordnete Kindelemente. Wirkt wie das tr-Element in HTML- Das Element steht für eine Tabellenzelle. Wirkt wie die Elemente th und td in HTML

11.3. Positionieren der Elemente

Jedes HTML-Element, egal ob Block- oder Inline-Element, „lebt“ in einem (Vater-) Container, der der Bezug für ein gegebenes Element ist. So werden die Positionen und Größen in CSS gewöhnlich abhängig von den Kanten des umschließenden Blocks berechnet. Aber auch wenn die Position jeder Box sich an ihrem umschließenden Block orientiert, so muss sie doch nicht immer innerhalb dieses Blocks liegen. So kann man z. B. durch negative **margins** erreichen, dass sie teilweise oder ganz außerhalb des umschließenden Blocks liegt. Außerdem können die Element auch vollständig oder auch Teilweise aufeinanderliegen, so dass insgesamt mehreren EbeneFür die Positionierung und Elementgröße gelten folgende Attribute:

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
position:	static relative absolute fixed	Beschreibt die Art der Positionierung <ul style="list-style-type: none">- Keine spezielle Formatierung, Element positioniert sich im Textfluß (standard)- Abweichung von der Position, die das Element im Textfluß eingenommen hätte- Positionierung des Elements in Abhängigkeit vom Elternelement.- Feste Positionierung im Browserfenster; beim Scrollen der Webseite bleibt das Element fixiert
top: right: bottom: left:	20px 50px 60px 90px	<ul style="list-style-type: none">- Startposition von oben- Startposition von rechts- Startposition von unten- Startposition von links
z-index:	1 2 3	<ul style="list-style-type: none">- Bestimmt die Reihenfolge von überlappenden Elementen. Je höher die Zahl, desto weiter vorne liegt das Element

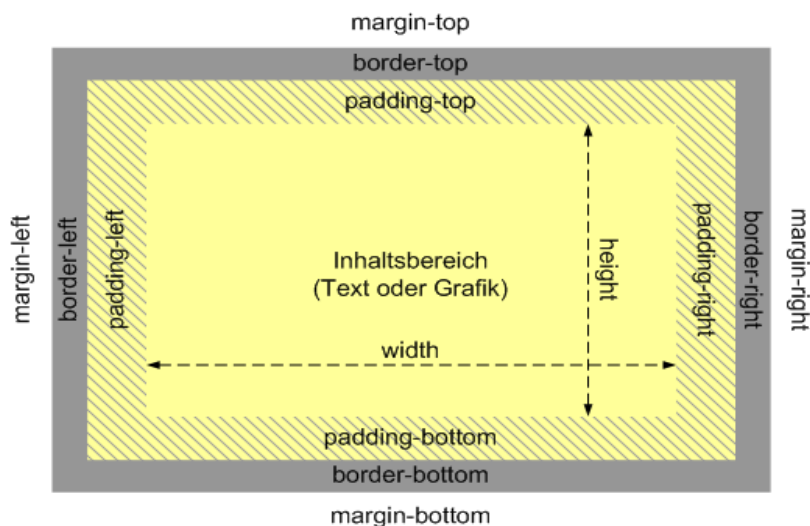
11.4. Größe der Elemente

Grundsätzlich ist innerhalb des Textflusses zwischen Block- und Inline-Elementen zu unterscheiden. Block-Elemente nehmen dabei, wenn nichts anderes erzwungen wird, in der Breite so viel Raum ein wie möglich und in der Höhe so viel Raum wie erforderlich. Inline-Elemente nehmen dagegen, wenn nichts anderes erzwungen wird, sowohl in der Breite als auch in der Höhe nur so viel Raum wie erforderlich. Wenn die Größe des Elementes für die Inhalte (vor allem Text) nicht ausreicht, kommt es zum s.g. Überlauf (overflow). Wie der Browser in einer solchen Situation „reagiert“ kann man ebenfalls mit CSS festlegen. Konkrete CSS-Regeln für den Umgang mit der Größe der Elemente sehen folgendermaßen aus:

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
width: min-width: max-width: height: min-height: max-height:	auto 300px 70%	- Festlegung der Größe von Elementen
overflow:	visible hidden auto scroll	<ul style="list-style-type: none">- Das Element der gesamte Inhalt ist sichtbar- Elementhöhe bleibt; der Inhalt wird abgeschnitten- Browserabhängige Einstellung (meist Scrollen)- Einschalten der horizontalen und vertikalen Bildlaufleisten

11.5. Das Box-Modell

Um die Steuerung der Innenabstände, Rahmen und Außenabständen zu ermöglichen stellt CSS das sogenannte Box-Modell zur Verfügung. Die Attribute **margin** (Außenrand), **border** (Rahmen) und **padding** (Innenabstand) können dabei für alle vier Seiten allgemein oder auch für jede Seite spezifisch mit Werten belegt werden.



CSS-Eigenschaft	Werte/Beispiele	Beschreibung
margin-top (-width): margin-right (-width): margin-bottom (-width): margin-left (-width):	20px 30px 50px 40px	. Dicke des oberen Außenrandes . Dicke des rechten Außenrandes . Dicke des unteren Außenrandes . Dicke des linken Außenrandes analog für border und padding
margin (-width): border (-width): padding (-width):	1 Wert z.B. 20px 2 Werte z.B. 10px 20px 3 Werte z.B. 10px 0px 20px 4 Werte z.B. 3px 10px 5px 20px	. Dieser Wert gilt für alle vier Seiten . Der erste Wert für die obere und untere Seite, der zweite Wert für die rechte und linke Seite . Der erste Wert ist für die obere, zweite für die linke und rechte, dritte für die untere Seite . Der erste Wert ist für die obere, zweite für die rechte, dritte für die untere und der vierte für die linke Seite
border-style:	none solid dashed double dotted groove (3D) ridge (3D Umkehrung von groove) inset (oben und links etwas dunkler) outset (unten und rechts etwas dunkler)	. Nicht sichtbarer Rand <div style="border: 1px solid black; padding: 2px; margin: 2px;">Solider Rahmen</div> <div style="border: 1px dashed black; padding: 2px; margin: 2px;">Gestrichelter Rahmen</div> <div style="border: 1px double black; padding: 2px; margin: 2px;">Doppelter Rahmen</div> <div style="border: 1px dotted black; padding: 2px; margin: 2px;">Gepunkteter Rahmen</div> <div style="border: 1px groove black; padding: 2px; margin: 2px;">Groove Rahmen</div> <div style="border: 1px ridge black; padding: 2px; margin: 2px;">Ridge Rahmen</div> <div style="border: 1px inset black; padding: 2px; margin: 2px;">Inset Rahmen</div> <div style="border: 1px outset black; padding: 2px; margin: 2px;">Outset Rahmen</div>
border-color:	blue (oder z.B. #0000FF)	. Festlegung der Rahmenfarbe
border:	#0000FF 5px solid	. Wenn die Eigenschaften für alle 4 Seiten des Rahmens gleich sein sollen, so können sie zusammen innerhalb einer einzigen Zeile, mit Hilfe der border-Eigenschaft, angegeben werden
border-top-left-radius: border-top-right-radius: border-bottom-right-radius: border-bottom-left-radius:	20px 30px 50px 40px	- Legt die Rundung linken oberen Ecke - Legt die Rundung rechten oberen Ecke - Legt die Rundung rechten unteren Ecke - Legt die Rundung linken unteren Ecke
border-radius:	50px 20px 30px 50px 40px	- Legt die einheitliche Ecken - Rundung - Legt die Rundung aller Ecken in der Reihenfolge wie oben

11.6. Aufgaben

1. Die Wirkung der unterschiedlichen CSS – Regeln tiefer zu verstehen untersuche die Dokumente im Ordner `Kapite9_Vorlagen`. Verändere dabei auch die Attributwerte und beobachte die Veränderungen.
2. Auf der Homepage ist es oft wünschenswert die Bilder zu unterschreiben und sie dann an einer gewünschten Stelle zu positionieren. Erstelle drei Boxen mit einem unterschriebenen Bild und positioniere sie auf einer Diagonale des Browserfensters.
3. Es möglich ist, das Layout einer gesamten Seite entweder mit Hilfe einer Tabelle oder mit Hilfe CSS festzulegen. Heutzutage bevorzugt man die Layoutgestaltung mit CSS. Untersuche die CSS Layoutvorlage `layout-css.htm`, die identisch wie `layout-tabellen.htm` aussieht.
4. Im Dokument `3bereicheineinander.htm` findest Du ein einfaches Beispiel für ein Seitenlayout. Verändere das Beispiel so, dass die Bereiche keine Abstände zueinander aufweisen und der gesamte Bereich mit einer Breite von 700 Pixeln zentriert auf der Webseite platziert wird.



5. Erstelle ein Seitenlayout nach folgendem Muster



6. Baue entsprechende CSS-Seitenlayouts für dein Projekt.
 - a. Ein Layout für die Startseite. (`index.html` + `startseite.css`)
 - b. Ein Layout für die Unterseiten. (`unterseite.html` + `unterseite.css`)

12. Weitere wichtigen Aspekte Webdesigns

Die Anzahl der im Internet befindlichen Webseiten steigt ständig. Um mit eigenen Internet-Präsenz wahrgenommen zu werden und positiv aufzufallen, sollte man sich bei den Suchmaschinen bemerkbar machen und eine besondere Acht auf Struktur und Layout geben.

12.1. Internet-Präsenz für die Suchmaschinen vorbereiten

Search engines (Suchmaschinen im klassischen Sinne) arbeiten mit Programmen (so genannte robots), die regelmäßigen Webseiten selbstständig auf Inhalte untersuchen. Dabei folgen sie den Verlinkungen innerhalb der Webseite und klassifizieren die Seiten nach gefundenen Stichwörtern in Titeln, Fließtext (meist innerhalb der ersten 250 Zeichen) und *Metatags*. Die Ergebnisse der Suche werden in einer zentralen Datenbank gespeichert. Über eine Stichwortsuche kann der Internetnutzer auf die gespeicherten Informationen zugreifen. Ihm werden dann die Internetadressen aufgelistet, die mit seinen eingegebenen Stichworten übereinstimmen.

Metatags können genauso wie der Titel in den <head>-Bereich eines HTML-Dokuments eingebracht werden. Die erforderlichen Tags besitzen alle die Struktur:

```
<meta name="..." content="..." />.
```

Durch „name“ wird festgelegt, um welche Art von Metatag es sich handelt, und der „content“ legt fest, auf welche Weise welche Informationen übermittelt werden. Jeder Typus von Metatag muss in eine eigene Spitzklammer gesetzt werden, damit er eindeutig identifiziert und umgesetzt werden kann. Die nachfolgende Tabelle stellt die wichtigsten Typen von Metatags und ihre Funktionen:

Attribut	Funktion de Attributes („content“)
title	gibt den eingegebenen Text als Titel der Seite wieder; dieser Text wird in der Suchmaschine als Ergebnis ausgegeben
description	dieser Text wird als Beschreibung der Webseite zusammen mit dem Titel angezeigt; die Beschreibung sollte nicht länger als 200 Zeichen sein
keywords	gibt ein Internetnutzer ein Schlüsselwort in die Suchmaschine ein, das hier steht, so wird die Seite als „Treffer“ angezeigt; die einzelnen Keywords werden durch Kommata voneinander getrennt
author	gibt an, wer die Seite erstellt hat; wird nur von wenigen Suchmaschinen verwendet
publisher	gibt an, wer für den Web-Auftritt verantwortlich ist
revisit-after	gibt an, wie oft die Seite aktualisiert wird; der Eingabe entsprechend oft schauen die robots vorbei (Anzahl der Tage)
robots	eine Möglichkeit, die Roboter der Suchmaschinen durch einen Metatag zu steuern: „index“ erlaubt dem Roboter, die Seite zu erfassen; „follow“ lässt ihn den Links in der Seite folgen; mit vorangestelltem „no“ (dieses wird ohne Abstand vor das „follow“ gesetzt), werden die beiden o.g. Anweisungen negiert; durch ein Komma verbunden, kann man die Befehle kombinieren
language	übermittelt die Sprache der Webseite; ist wichtig für Suchmaschinen, die nur nach Seiten mit bestimmten Sprachen suchen

Beispiel;

```
<head>
  <titel>Titel der Seite</titel>
  <meta name="title" content="Homepage des HJK-Kollegs">
  <meta name="descriptiof content="Dies ist die Webseite des HJK-Kollegs">
  <meta name="keywords" content="HJK-Kolleg HJK Informatik-AG">
  <meta name="robots" content="index,follow">
  <meta name="language" content="deutsch">
</head>
```

12.2. Klare Struktur einer Webpräsenz

Wenn man von der Struktur einer Webpräsenz spricht, meint man dabei einerseits interne Ordnerstruktur (backend), andererseits die Struktur, die ein Onlinebenutzer zu Sicht bekommt (frontend). Die Kriterien für eine interne Struktur wurden schon im Kapitel 7 genannt. Nun ein paar „frontend“-Kriterien;

- Die Strukturierung des Web-Auftritts sollte für die den Benutzer bereits auf der Startseite verdeutlicht werden.
- Klare Bezeichnung der Links. Die Interne Verlinkung von der Startseite aus sollte der hierarchisch gegliederten Strukturierung folgen.
- Beschränkung der Darstellung auf eine Bildschirmseite.
- An der Corporate Identity ausgerichteter, einheitliches Layout und Farbkonzept.
- Gleiche Positionierung der Navigationselemente auf allen Seiten.
- Keine aufwändigen Hintergründe oder schmückende Grafiken
- Durchgängige Verwendung einer minimalen Zahl von selbsterklärenden Icons.

12.3. Layout

Ein gutes Layout ist an die Nutzertypen angepasst, die auf die Webseite „geloct“ werden sollen. Dabei muss jedoch nicht nur das Nutzungsinteresse, sondern auch die technische Ausstattung sowie Erfahrung im Umgang mit dem Medium berücksichtigt werden.

Es sollte also zu Beginn der Erstellung einer Webseite geklärt werden, welchen Zweck die Präsentation verfolgt. Sollen Informationen vermittelt werden oder soll die Seite unterhalten und so Surfer zum Verweilen bewegen? Die Anforderungen an die Technik (z.B. aufwändige Gestaltung von Animationen) und den Menschen (Umgang mit Computer) sollten nicht zu hoch sein. Wird jedoch nur Fließtext geboten, kann dies den Benutzer genauso von einem Besuch abhalten. Man ist also gezwungen, einen Spagat zwischen diesen Aspekten zu machen, um jedem das Gefühl zu vermitteln, ihn mit seinen Bedürfnissen zu berücksichtigen.

Grundsätzlich sollte ein ausgewogenes Verhältnis zwischen Text, Grafiken und Strukturierungselementen (wie z. B. Linien oder Zwischenüberschriften) bestehen.

Text ist die überwiegende Darstellungsform für Inhalte im Internet. Doch gerade das Lesen am Bildschirm ist schwierig und kann ermüden. Aus diesem Grund ist es wichtig, auf folgende Regeln zu achten:

- Text muss immer gut lesbar sein!
- Ausreichend Kontrast zwischen Schrift und Hintergrund erzeugen.
- Die Farben von Schrift und Hintergrund aufeinander abstimmen.
- Aufzeichnungen wie Fettdruck oder Vergrößerung des Textes, Kursivschrift oder farbiger Text nur sparsam einsetzen: wenn zu viel Text auf diese Weise hervorgehoben wird, verliert sich der Effekt besonderer Markierung.

13. Rechtliche Aspekte einer Webpräsenz

Das **Internetrecht** (auch: Onlinerecht) befasst sich mit den rechtlichen Problemen, die mit der Verwendung des Internet einher gehen. Es stellt kein eigenes Rechtsgebiet dar, sondern ist die Schnittstelle aller Rechtsgebiete im Bereich des Internets. Was man auf Ihrer Homepage darf und muss, hängt von vielen Dingen ab. Medien-Recht, Urheber-Recht, Marken-Recht, Datenschutz usw.. Hier ein paar generelle Regeln für eine private Homepage:

Inhalte: Man schreibt auf eigener Homepage nichts und man lässt andere nichts schreiben, was man nicht auch öffentlich sagen oder in der Zeitung drucken darf.

Das bedeutet auch, dass man fremde Einträge auf eigener Homepage (z.B. in einem Forum) überwachen muss. Anmerkungen wie "Ich bin nicht verantwortlich für Postings von Besuchern" befreien den Eigentümer NICHT von der Verantwortung!

Text kopieren ohne Einverständnis des Urhebers darf man, solange das Urheber-Recht daran noch gilt (und das sind viele Jahre) auch nicht. Auch Liedtexte etc. sind davon betroffen. Es gibt einige Ausnahmen, wie z.B. kurze Zitate, aber auch da ist Vorsicht geboten.

Bilder: Man verwendet keine Fotos oder Grafiken, bei denen man nicht das Recht zur Veröffentlichung hat! Auch eigene Fotos von Kunstwerken, bei denen das Urheberrecht noch gilt, darf man nicht veröffentlichen. Vorsicht! Copyright-Verletzungen können sehr teuer werden.

Personen-Bilder: Man darf keine Fotos von Personen ohne deren Einverständnis posten. Man muss die Privatsphäre anderer respektieren.

Musik, Movies: Man darf keine Musik-Files, Movies etc. verwenden, bei denen man nicht das Recht zur Veröffentlichung hat! Copyright-Verletzungen können sehr teuer werden.

Marken und Logos: Man verwendet keine fremden Logos oder Marken- oder Firmennamen, wenn man nicht das Einverständnis des jeweiligen Eigentümers eingeholt hat. Bei eigenem „Firmenauftritt“ sollte man Ähnlichkeiten zu fremden Logos und Markenzeichen vermeiden. Bei Verletzung von Markenrechten bekommt man mit Sicherheit rechtliche Schwierigkeiten.

Werbung/Verkauf: Achtung. Durch Werbung oder Angebote auf eigener Homepage wird die private Homepage sehr schnell zu einer gewerblichen Homepage und den Homepageeigentümer zum „Händler“, mit allen rechtlichen und steuerlichen Konsequenzen.

Steuer: Einnahmen aus Partnerprogrammen und Webshops sind zu versteuern, auch wenn der Betreiber eine Privatperson ist. Man sollte nicht das Finanzamt unterschätzen. Der Provider wird alle Verkaufsdaten auf Antrag des Finanzamtes auf dessen Ersuchen herausgeben müssen.

Impressum: Eine rein private Webseite benötigt kein Impressum, solange sie nicht geschäftsmäßig betrieben wird und keine Hinweise auf Produkte, geschäftsmäßige oder kommerzielle Angebote enthält.

14. Projektaufgabe

Nun ist die Zeit, dein Projekt zu vervollständigen. Der gesamte Aufwand sollte nicht 12 HTML-Seiten übersteigen. Bei der technischen Umsetzung gilt folgende Erwartungshorizont.

Interne Struktur

Hier gelten die Vorgaben aus dem Kap.7.

- Im Hauptverzeichnis sollte nur Startseite mit dem Namen `index.html` stehen.
- Alle CSS-Dateien, Graphiken, Multimedia-Dateien sollten jeweils in separaten Ordner gespeichert werden.
- Weitere Ordner spiegeln die Inhaltliche Struktur der Webpräsenz.

Frontend Struktur

- Die HTML-Seiten sollten prinzipiell durch die Body-Bereiche vom Kapitel 10.1 strukturiert werden.
- Es reichen hier zwei Arten von Grundstrukturen aus: eine für die Startseite, eine andere für die restlichen Seiten.
- Die Navigationstiefe sollte mindestens die Stufe 2 erreichen.

CSS-Regeln

- Die gesamte optische Formatierung wird, soweit das möglich ist, durch CSS gesteuert.
- Die CSS-Regeln sind in mehreren separaten CSS-Dokumenten ausgelagert: z.B. eine CSS-Datei für die Startseite, eine für die restlichen Seiten, eine (oder besser zwei unterschiedliche) für die Navigation und eine CSS-Datei für die einheitliche Gestaltung von solchen HTML-Elementen, wie Tabellen, Überschriften, Listen, Bilder mit Unterschriften,...
- Da die vielen CSS-Regeln sehr schnell unübersichtlich erscheinen, sollten die CSS-Dateien ausführlich kommentiert werden.

Verlangte Elemente

- Es ist (mindestens) eine drop-down – Navigation vorhanden.
- Durch das Anklicken von Logo kann man von jeder Stelle auf die Startseite gelangen.
- Einige Graphiken werden durch Text umflossen. Es gibt Graphiken mit Unterschriften.
- Es sind mindestens zwei Multimediadateien, mehrere Listen, eine sinnvoll gestaltete Tabelle und mehrere externe Hyperlinks vorhanden.

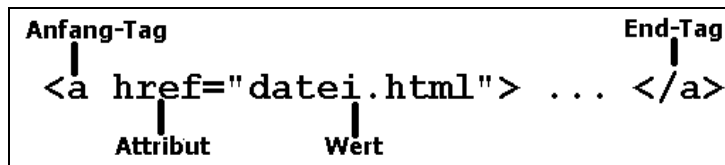
Weitere wichtige Aspekte

- Alle Copyrights werden beachtet. Ein Impressum ist von jeder Seite erreichbar.
- Sinnvolle *Metatags* beschreiben jedes HTML-Dokument
- Struktur und Layout richten sich nach dem im Kapitell 12 genannten Kriterien.

VIEL ERFOLG



15. Kurze HTML Referenz



Globale Attribute

Attribut	Wert	Beschreibung
lang	= „Sprachcode“	Definiert die Landessprache für das Element (z.B. en)
class	= „Text“	Weist einen bzw. mehrere Klassennamen zu.
id	= „Text“	Weist dem Element eine eindeutige Bezeichnung zu.
style	= „CSS-Regeln“	Weist dem Element ein internes Stylesheet zu.
title	= „Text“	Beschreibung eines Elements, die als Tooltip-Text angezeigt wird.

Strukturelemente eines HTML-Dokumentes

TAG	Verwendung	Beschreibung
<!doctype html>	Anfang des HTML5-Dokumentes	Angabe, um welche Art des Dokumentes sich handelt.
<html> ... </html>	Rahmen von HTML-Dokument	Jedes HTML-Dokument ist von diesen Tags umrandet.
<head> ... </head>	Kopfbereich	Enthält Angaben wie den Titel der Seite oder auch Stichwörter für Suchmaschinen (sogenannte Metatags).
<body> ... </body>	Rumpfbereich	Enthält den eigentlichen Text des zu erzeugenden Dokuments sowie Grafiken, Tabellen etc., also den Inhalt.
<div> ... </div>	Blöcke definieren	Ein Block in einem HTML-Dokument kann mehrere, unterschiedliche Objekte beinhalten. Bis HTML5 war das das einzige Tag, um die Blöcke zu definieren
<header> ... </header>	Für den Kopfbereich	Kann typischerweise den Titel des Dokuments, Logos, ein Formular zur Schnellsuche oder ein Inhaltsverzeichnis enthalten
<nav> ... </nav>	Für die Hauptnavigation	
<article> ... </article>	Für die eigentlichen Inhalte der Seite	Die Verwendung soll so erfolgen, dass article-Blöcke, für sich genommen, alleinstehend sind, also beispielsweise auch unverändert als Inhalt eines Newsfeeds verwendet werden könnten.
<section> ... </section>	Sektionen der Seite besser abgrenzen.	article-Blöcke können mit <section> in mehrere Abschnitte unterteilt werden und sind außerdem schachtelbar.
<aside> ... </aside>	Für die „locker“ vom Hauptinhalt abhängende Abschnitte,	Ein klassischer Fall für Sidebars (Randfenster), aber auch für inhaltliche Einschübe in einem article.
<footer> ... </footer>	Für den Fußbereich eines Dokuments findet:	Hier findet man z.B. Autor- und Copyright-Informationen oder Querverweise. „Footer“ kann, aber muss nicht notwendigerweise am Ende eines Dokuments stehen
 ... 	Inline-Blöcke definieren	Kürzere Textabschnitte werden durch den Tag eingeschlossen.
<!-- Anmerkungen -->	Kommentar	Dadurch fügt man dem HTML-Quelltext Erläuterungen hinzu.
<hr />	Trennlinien	Dieses Element erzeugt eine waagerechte Linie und symbolisiert einen thematischen Umbruch zwischen Textabschnitten.

Elemente des Head-Bereiches

TAG / Attribut	Wert	Beschreibung
<title> ... </title>		Bestimmt den Dokumenttitel.
<base href />	= „URL“	Definiert einen Basispfad für alle relative Hyperlinks innerhalb des Dokumentes.
<link charset href rel type />	= „Zeichensatz“ = „URL“ = „stylesheet“ = „MIME TYPE“ z.B. text/css	Verlinkt das Dokument mit einer externen Resource. - Zeichencodierung des verlinkten Dokuments. - Pfad zur Ressource - Beziehung zum Zieldokument - Medientyp der Ressource
<meta name description />	= „Text“ = „Text“	Informationen für Webserver, Browser und Suchmaschinen - Inhaltsbezeichnung, z.B. autor, description, keywords,... - Wert der (name) Eigenschaft.
<style type />	= „text/css“	Definiert einen Bereich für eingebettetes Stylesheet - Medientyp der Stilsprache

Textstrukturen

TAG	Verwendung	Beschreibung/Standardformatierung
<code><h(1-6)> ... </h(1-6)></code>	Überschriften	Es gibt sechs verschiedene Überschriftgrößen (h1 bis h6).
<code><p> ... </p></code>	Textabsatz	Funktioniert ähnlich wie RETURN beim Textverarbeitung
<code>
</code>	Zeilenumbruch	Der Leer-Tag <code>
</code> bricht eine Zeile um.
<code><nobr> ... </nobr></code>	Zeilenumbruch verhindern	Der umschlossene Text wird nicht automatisch getrennt.
<code><acronym> ... </acronym></code>	Abkürzungen	Keine besondere Standardformatierung
<code> ... </code>	Betonter Text	Standardmäßig fett dargestellt.
<code><i> ... </i></code>	Kursiver Text	Es sollen damit Texte markiert werden, die im Normalfall kursiv dargestellt werden sollen - z. B. Gedanken, Titel usw.
<code> ... </code>	Gelöschter Text.	Wird oft als durchgestrichener Text dargestellt.
<code><adress> ... </adress></code>	Adressangaben	Kursiv
<code><blockquote> ... </blockquote></code>	Längere Zitate	Rechts eingerückt dargestellt
<code><cite> ... </cite></code>	Quellangaben von Zitaten	Kursiv
<code><code> ... </code></code>	Quelltext	Schreibmaschinenschrift wie z.B. Courier
<code><dfn> ... </dfn></code>	Begriffsdefinitionen	Kursiv
<code> ... </code>	Betonung	Kursiv
<code><kbd> ... </kbd></code>	Tastatureingaben	Schreibmaschinenschrift wie z.B. Courier
<code><pre> ... </pre></code>	Vorformatierter Text	Der Text wird samt eingegebener Leerschritte bzw. -zeilen vom Browser übernommen.
<code><q> ... </q></code>	Kurze Zitate	Anführungszeichen
<code><samp> ... </samp></code>	Beispiele	Schreibmaschinenschrift wie z.B. Courier
<code><sub> ... </sub></code>	Tiefgestellter Text	z.B. Normal tiefgestellt
<code><sup> ... </sup></code>	Hochgestellter Text	z.B. 10 30
<code> ... </code>	Starke Betonung	Fett
<code><var> ... </var></code>	Variablen	Kursiv

Wichtigsten Sonderzeichen

HTML-Code	Zeichen	HTML-Code	Zeichen	HTML-Code	Zeichen
<code>&auml;</code>	ä	<code>&copy;</code>	©	<code>&micro;</code>	μ
<code>&ouml;</code>	ö	<code>&deg;</code>	°	<code>&nbsp;</code>	„Leerzeichen“
<code>&uuml;</code>	ü	<code>&euro;</code>	€	<code>&plusmn;</code>	±
<code>&szlig;</code>	ß	<code>&frac12;</code>	½	<code>&pound;</code>	£
<code>&ScAuml;</code>	Ä	<code>&frac14;</code>	¼	<code>&reg;</code>	®
<code>&Ouml;</code>	Ö	<code>&frac34;</code>	¾	<code>&sect;</code>	§
<code>&Uuml;</code>	Ü	<code>&gt;</code>	>	<code>&sup2;</code>	²
<code>&quot;</code>	"	<code>&lt;</code>	<	<code>&sup3;</code>	³

Listen

TAG / Attribut	Wert	Beschreibung
<code> ... </code>		Eine Liste ohne Nummerierung.
<code> ... start ... </code>	= „5“	Eine nummerierte Liste. - Die Angabe eines Startwertes
<code> ... </code>		Listeneintrag einer <code></code> oder <code></code> Liste
<code><dt> ... </dt></code>		Beschriebener Ausdruck.
<code><dl> ... </dl></code>		Beschreibung des Ausdruckes.

Graphiken

TAG / Attribute	Werte/Beispiele	Beschreibung
<code></code>	z.B. <code>src="../../bild.gif"</code> 200px, 30% 100px, 20% <code>alt="Unsere Schule"</code>	- Leeres Element für die Graphikanbindung - (source) - die URL zu einer -Grafik wird angegeben - Die Angabe der Breite der Grafik (auch mit CSS machbar) - Die Angabe der Höhe der Grafik (auch mit CSS machbar) - Der alternative Text wird angezeigt, wenn die Grafik nicht darstellbar ist.

Multimedia

TAG-Eigenschaft	Werte/Beispiele	Beschreibung
<audio		Bindet die Audiodateien ein:
src	= „URL“	- Definiert die URL von der Audio-Datei
autoplay	ohne Wert = „autoplay“	- Der Browser wird den Inhalt direkt abgespielt werden, sobald genug geladen ist.
controls	ohne Wert = „controls“	- Der Browser wird Steuerungselemente (Play/Pause/Position/Lautstärke) anzeigen
loop	ohne Wert = „loop“	- Gibt an, dass die Audio sollte wieder von vorn anfangen, wenn es fertig ist.
preload	ohne Wert = „auto“ = „metadata“ = „none“	- Gibt an, ob die Audio-geladen werden soll, wenn die Seite geladen wird. (Wird ignoriert, wenn autoplay vorhanden ist.)
> <source>...</source> </audio>		Als Inhalt zwischen Anfang- und Endtag kann man Bemerkung für die ältere Browser schreiben, oder zwischen <source> unb </source> werden die alternativen Dateien angegeben.
<video>		Bindet die Videodateien ein. Möglich alle Attribute wie bei <audio> plus:
audio	muted	Definiert den Grundzustand von Audio. Derzeit werden nur "stumm" ist zulässig
height	pixels	Setzt die Höhe des Video-Players
width...	pixels	Legt die Breite des Video-Players
</video>		

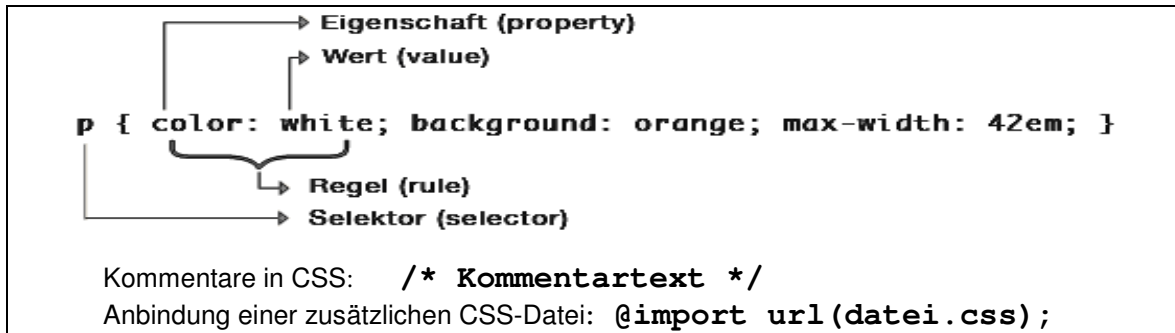
Hyperlinks

Tag/Attribute	Werte/Beispiele	Beschreibung
id="Ziel"	<p id="Ziel">Sprungmarke</p>	- Anfang von diesem Abschnitt wird zur Sprungmarke
 Verweistext 	http://www.domain.de #Ziel http://www.domain.de#Ziel _blank _self Home 	- Haupttag für die Hyperlinks - Verweis auf eine URL im Internet - Verweis auf eine Sprungmarke im eigenem Dokument - Verweis auf eine Sprungmarke eines „fremdes“ Dokuments - Verweisziel wird in einem neuen Fenster geöffnet - Verweisziel wird im aktuellen Fenster geöffnet - Der Sprung wird ausgeführt, wenn man auf einen der Elemente zwischen <a> und klickt

Tabellen

Tag/Attribute	Werte/Beispiele	Beschreibung
<table border = ? width = ? > </table>	border = „2“ width = „300“ width = „80%“	Leitet eine Tabellendefinition ein. - border legt die Dicke der Umrandung in Pixel fest. (In Html5 nicht mehr unterstützt, fürs Anfang jedoch sehr hilfreich) - Legt die Breite der Tabelle in Pixel bzw. in Prozentangaben, wobei 100% der Breite des Gesamtfensters entspricht. (In Html5 nicht mehr unterstützt, fürs Anfang jedoch sehr hilfreich)
<caption>... </caption>		
<colgroup span = ? > <col /> </colgroup>	= „2“	Für die Organisation der Tabellenspalten - Aufeinander folgende Spalten mit gleicher Breite können mit span zusammengefasst werden - Für jede Tabellenspalte wird eine <col>-Angabe benötigt.
<thead> ... </thead>		Legt logisch den Kopfbereich einer Tabelle fest
<tfoot> ... </tfoot>		Legt logisch den Fußbereich einer Tabelle fest (Wenn vorhanden ist, muss direkt dem Kopfbereich folgen)
<tbody> ... </tbody>		Legt logisch den Rumpfbereich einer Tabelle fest.
<tr height = ?> <th width = ? > Inhalt</th> <td colspan = ? rowspan = ?> Inhalt </td> </tr>	= „40“ = „20“ = „3“ = „2“	Definition einer Zeile (table row) - Angabe der Zeilenhöhe in Pixel - Definition einer Zelle für Spaltenüberschriften. <th> (table header) - Angabe der Zeilenbreite in Pixel - Inhalt der Spaltenüberschrift - Definition einer Zelle (table data) evtl. mit Breitenangabe - Verbindung von (3) Spalten - Verbindung von (2) Zeilen - Inhalt der „normalen“ Zelle (bzw. vom Verbund der Zellen)

16. Kurze CSS Referenz



Selektoren in CSS

Einfache Sel.	Beispiel	Bezeichnung / Beschreibung
*	<code>* {color: red }</code>	Universal-Selektor Universal selector Alle Element der HTML-Seite
Tag	<code>p { font-family: fantasy }</code>	(Element) Typ-Selektor Type Selectors Alle E-Tags der HTML-Seite
.klasse Tag.klasse	<code>.extra { font-size: 1.6em; }</code> oder <code>p.extra { font-size: 1.6em; }</code>	Klassen-Selektor Class Selectors Alle E-Tags mit dem Attribut class="klasse"
#einzig Tag#einzig	<code>#nav { width: 200px; }</code> oder <code>div#nav { width: 200px; }</code>	ID-Selektor ID selectors Alle HTML-Tags mit dem Attribut id=""


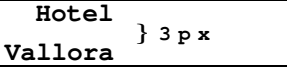
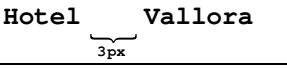
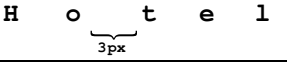
Kontext Sel.	Beispiel	Bezeichnung / Beschreibung
E F	<code>a img { border: 4 px solid blue }</code> Alle Bilder innerhalb von a-Tags	Nachfahren-Selektor Alle F-Tags, die innerhalb eines E-Tags liegen auch in zweiter oder dritter Ebene
E > F	<code>div>table { border:thin green; }</code> Alle table-Elemente, die direkt innerhalb eines div-Elements liegen	Kind-Selektor: Alle F-Tags, die direkter Nachfahre des E-Tags sind
E + F	<code>h5 + p { font-weight: bold }</code> Das p-Element, das direkt auf h5 folgt	Nachbar-Selektor: Das F-Element, das unmittelbar hinter dem E-Tag folgt.
E ~ F	<code>h5 ~ p { font-size: smaller }</code> Alle p-Elemente, die der Überschrift h5 folgen	Indirakter Nachbar-Selektor Alle F-Tags, die dem E-Tag folgen.

Attribut Sel.	Beispiel	Bezeichnung / Beschreibung
E [attr]	<code>td[width] { background: red }</code>	Alle E-Elemente, deren "attr"-Attribut gesetzt ist (gleich, mit welchem Wert).
E [attr="xyz"]	<code>input[type="password"] {background: red; }</code>	Alle E-Elemente, deren "attr"-Attribut exakt den Wert "xyz" aufweist

Pseudoformate	Beispiel	Bezeichnung / Beschreibung
E:link	<code>a:link { color: blue; }</code>	Alle Elemente E, die einen Hyperlink zu einem noch nicht besuchten Ziel darstellen
E:visited	<code>a:visited { color: purple; }</code>	Alle Elemente E, die einen Hyperlink zu einem noch nicht besuchten Ziel darstellen
E:active	<code>a:active { background: #8FBC8F }</code>	Dynamischer Pseudolink E-Elemente während das Element aktiviert ist (z.B. Maustaste auf einen Link gedrückt ist)
E:hover	<code>li:hover { background: #8FBC8F }</code>	Dynamischer Pseudolink: E-Elemente während die Maus über dem Element hovers
E:focus	<code>input:focus { background: red }</code>	Dynamischer Pseudolink: E-Elemente während der Fokus auf dem Element liegt
E:first-child	<code>td:first-child { color: green ;}</code>	Erstes Kind eines Eltern-Elements
E:last-child	<code>li:last-child { border-bottom: 1px}</code>	wie :first-child, aber filtert vom letzten Element an
E:nth-child	<code>tr:nth-Child(2) { color: pink }</code>	filtert das n-te Kindelement eines HTML-Elements
E:nth-last-child	<code>tr:nth-last-Child(2){ color: blue }</code>	wie nth-child, aber beginnt am Ende
E:nth-of-type	<code>p:nth-of-type(odd) { color: black }</code>	wie nth-child, wirkt nur bei Elementen eines Tpps
E:nth-last-of-type	<code>p:nth-last-of-type(odd){color:red }</code>	wie nth-of-type, aber beginnt am Ende
E:empty	<code>td:empty { background: gray }</code>	ein Element ohne Inhalt

Pseudoelemente	Beispiel	Bezeichnung / Beschreibung
E:first-line	p:first-line { color: purple }	Erste Zeile in einem Text
E:first-letter	p:first-letter { color: purple }	Erstes Zeichen in einem Text
E:before, E:after	td.preis:before { content: " + " } td.preis:after { content: " € " }	:before und :after fügen Inhalte automatisch vor oder nach einem dynamisch generierten Element ein.

Text mit CSS gestalten

Eigenschaft	Werte/Beispiele	Beschreibung
font-family	Arial Comic Sans Courier New Georgia serif für Serifenschriften wie Times New Roman sans-serif für serifenlose Schriften wie Arial	Name der Schriftart bzw. Bezeichnung für eine Schriftfamilie Serifen sind Abschlusstriche von Buchstaben:  Schrift mit (serif) und ohne Serifen (sans-serif)
font-size	xx-small, x-small, small, medium, large, x-large, xx-large smaller, larger 12px 14pt 150% 1.2em	- Vordefinierte Schriftgrößen - im Bezug auf Maß des Übergeordneten Elementes - Eingabe in Pixel - Eingabe in Punkten - Prozent gegenüber dem übergeordneten Element - x-fache der Größe des übergeordneten Elements
font-style	normal (normal) italic, oblique (kursiv)	Schriftschnitt
color	blue #0000FF 0,0,255	Schriftfarbe
font-variant	normal, small-caps	Kapitälchen
font-weight	normal, bold	Gewicht
text-align	left, right, center, justify	Horizontale Textausrichtung
vertical-align	text-top, middle, text-bottom	Vertikale Textausrichtung z.B. in einer Tabellen-Zelle.
text-decoration	none (normale Schreibweise) underline (unterstrichen) line-through (durchgestrichen) overline (Linie über dem Element) blink (blinken)	Texteffekte
text-indent	Wir wünschen Ihnen...	Einzug der ersten Zeile
text-transform	none (normale Schreibweise) capitalize (erster Buchstabe groß) lowercase (Kleinschreibung) uppercase (Großschreibung)	Groß / Kleinschreibung erzwingen
line-height	3px) 	Zeilenabstand
word-spacing	3px 	Wortabstand
Letter-spacing	3px 	Zeichenabstand

Listen mit CSS gestalten

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
list-style-type	none circle square disc decimal lower-alpha upper-alpha lower-roman upper-roman decimal-leading-zero lower-greek	- kein Aufzählungszeichen - ungefühlter Kreis, nur Rahmen - gefühltes Quadrat - gefühlter Kreis - Dezimalzahlen (1, 2, 3, ...) - Kleinbuchstaben (a, b, c, ...) - Großbuchstaben (A, B, C, ...) - Römische Zahlen, Kleinschreibung (i, ii, iii, ...) - Römische Zahlen, Großschreibung (I, II, III, ...) - Dezimalzahlen mit führender 0 (01, 02, 03, ...) - Kleine, griechische Nummerierung (α, β, γ, ...)
list-style-image	url (Wert) z.B. url(burst.png)	- eigene Graphik als Aufzählungszeichen

Hintergrund mit CSS gestalten

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
background-color	blue #0000FF 0,0,255	Hintergrundfarbe wird zugewiesen
background-image	url(Pfad/Graphikname)	Syntax für die Einführung eines Hintergrundbildes
background-repeat	repeat (wiederholen) no-repeat (nicht wiederholen) repeat-x (waagerecht wiederholen) repeat-y (senkrecht wiederholen)	Diese Eigenschaft gibt an, ob oder wie eine Grafik wiederholt wird.
background-position	Wert, z. B. 50px 20% top (horizontale Ausrichtung oben) center (hor. Ausrichtung mittig) bottom (horizontale Ausrichtung unten) left (vertikale Ausrichtung links) center (vertikale Ausrichtung mittig) right (vertikale Ausrichtung rechts) z. B. top center	Mit dieser Eigenschaft legt man; die Werte und die gewünschte Maßeinheit fest, um die Position des Bilds relativ zur oberen, linken Ecke des Elements anzugeben. Zur Angabe der Positionierung gibt man zuerst den Wert für die horizontale Ausrichtung und anschließend den Wert für die vertikale Ausrichtung an.
background-attachment	scroll fixed	- Beim Scrollen bewegt sich das Bild mit - Hintergrundbild bleibt stehen

Anzeigeart der Elemente

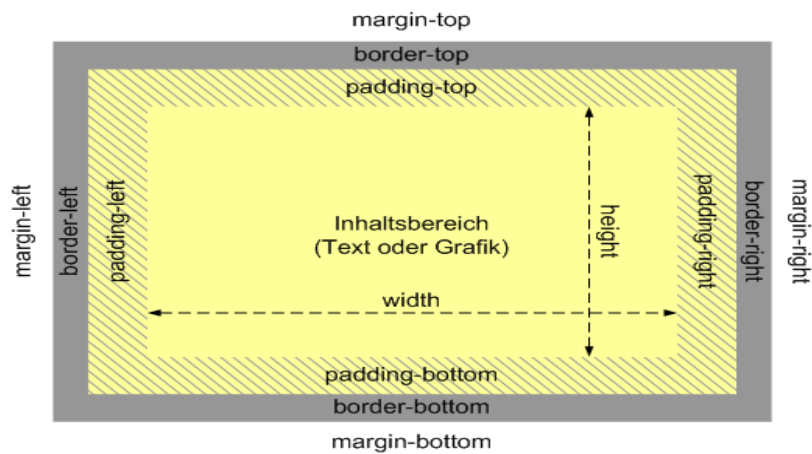
CSS-Eigenschaft	Werte/Beispiele	Beschreibung
display	none block inline inline-block table table-row table-cell	<ul style="list-style-type: none"> - Element wird nicht angezeigt - Erzwingt einen Block - das Element erzeugt eine neue Zeile - Das Element wird im laufenden Textfluss angezeigt - Erzeugt einen Block, belässt das Element jedoch im Textfluss. - Das Element enthält tabellarisch angeordnete Kindelemente und erzeugt eine neue Zeile. Wirkt wie das table-Element in HTML - Das Element enthält nebeneinander angeordnete Kindelemente. Wirkt wie das tr-Element in HTML - Das Element steht für eine Tabellenzelle. Wirkt wie die Elemente th und td in HTML

Positionieren mit CSS

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
position:	static relative absolute fixed	<p>Beschreibt die Art der Positionierung</p> <ul style="list-style-type: none"> - Keine spezielle Formatierung, Element positioniert sich im Textfluß (standard) - Abweichung von der Position, die das Element im Textfluss eingenommen hätte - Positionierung des Elements in Abhängigkeit vom Elternelement. - Feste Positionierung im Browserfenster; beim Scrollen der Webseite bleibt das Element fixiert
top: right: bottom: left:	20px 50px 60px 90px	<ul style="list-style-type: none"> - Startposition von oben - Startposition von rechts - Startposition von unten - Startposition von links
z-index:	1 2	- Bestimmt die Reihenfolge von überlappenden Elementen. Je höher die Zahl, desto weiter vorne liegt das Element

Das Umfließen eines Elementes

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
float	left right	Horizontale Ausrichtung eines Elementes in einem Fluss von anderen Elementen.
clear	left right both	Beendet das Umfließen. Ein mit clear formatiertes Element ist das Erste, dass nicht mehr neben anderen Elementen steht. Es kann aber trotzdem mit der float Eigenschaft formatiert werden, damit nachfolgende Elemente um dieses Element wieder herum fließen.



Box - Modell

CSS-Eigenschaft	Werte/Beispiele	Beschreibung
<code>margin-top (-width):</code> <code>margin-right (-width):</code> <code>margin-bottom (-width):</code> <code>margin-left (-width):</code>	20px 30px 50px 40px	<ul style="list-style-type: none"> · Dicke des oberen Außenrandes · Dicke des rechten Außenrandes · Dicke des unteren Außenrandes · Dicke des linken Außenrandes analog für border und padding
<code>margin(-width):</code> <code>border(-width):</code> <code>padding(-width):</code>	1 Wert z.B. 20px 2 Werte z.B. 10px 20px 3 Werte z.B. 10px 0px 20px 4 Werte z.B. 3px 10px 5px 20px	<ul style="list-style-type: none"> · Dieser Wert gilt für alle vier Seiten · Der erste Wert für die obere und untere Seite, der zweite Wert für die rechte und linke Seite · Der erste Wert ist für die obere, zweite für die linke und rechte, dritte für die untere Seite · Der erste Wert: die obere, zweite: die rechte, dritte: die untere und der vierte: die linke Seite
<code>border-style:</code>	none solid dashed double dotted groove (3D) ridge (3D Umkehrung von groove) inset (oben und links dunkler) outset (unten und rechts dunkler)	<ul style="list-style-type: none"> · Nicht sichtbarer Rand <div style="display: flex; flex-direction: column; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin: 2px;">Solider Rahmen</div> <div style="border: 1px dashed black; padding: 2px; margin: 2px;">Gestrichelter Rahmen</div> <div style="border: 1px double black; padding: 2px; margin: 2px;">Doppelter Rahmen</div> <div style="border: 1px dotted black; padding: 2px; margin: 2px;">Gepunkteter Rahmen</div> <div style="border: 1px groove black; padding: 2px; margin: 2px;">Groove Rahmen</div> <div style="border: 1px ridge black; padding: 2px; margin: 2px;">Ridge Rahmen</div> <div style="border: 1px inset black; padding: 2px; margin: 2px;">Inset Rahmen</div> <div style="border: 1px outset black; padding: 2px; margin: 2px;">Outset Rahmen</div> </div>
<code>border-color:</code>	blue (oder z.B. #0000FF)	· Festlegung der Rahmenfarbe
<code>border-collapse:</code>	collapse	· Einzelrahmen wird zusammenfallen.
<code>border:</code>	#0000FF 5px solid	· Wenn die Eigenschaften für alle 4 Seiten des Rahmens gleich sein sollen, so können sie zusammen innerhalb einer einzigen Zeile, mit Hilfe der border-Eigenschaft angegeben werden
<code>border-top-left-radius:</code> <code>border-top-right-radius:</code> <code>border-bottom-right-radius:</code> <code>border-bottom-left-radius:</code>	20px 30px 50px 40px	<ul style="list-style-type: none"> · Legt die Rundung linken oberen Ecke · Legt die Rundung rechten oberen Ecke · Legt die Rundung rechten unteren Ecke · Legt die Rundung linken unteren Ecke
<code>border-radius:</code>	50px 20px 30px 50px 40px	<ul style="list-style-type: none"> · Legt die einheitliche Ecken - Rundung · Legt die Rundung aller Ecken
<code>width:</code> <code>min-width:</code> <code>max-width:</code> <code>height:</code> <code>min-height:</code> <code>max-height:</code>	auto 300px 70%	· Festlegung der Größe von Elementen
<code>overflow:</code>	visible hidden auto scroll	<ul style="list-style-type: none"> · Das Element der gesamte Inhalt ist sichtbar · Höhe bleibt; der Inhalt wird abgeschnitten · Browserabhängige Einstellung (meist Scrollen) · Einschalten der horizontalen und vertikalen Bildlaufleisten