HTML 5

HTML ist keine Programmiersprache. Es kann keine Entscheidungen treffen oder Berechnungen durchführen. HTML ist statisch, fest.

Seit 2004 wird an HTML5 gearbeitet. Federführend bei der Definition ist die Web Hypertext Application Technology Working Group (WHATWG) und nicht mehr das W3C. Bei HTML5 handelt es sich um eine Sammlung von vielen Standards, von denen viele bereits jetzt genutzt werden. Dies entspricht der Grundidee, dass entsprechend der Weiterentwicklung einzelner Aspekte HTML5 kontinuierlich aktualisiert werden soll. Daher gab es auch lange kein Datum für den Abschluss der Standardisierung. Dann wurde aber doch noch das Jahr 2014 kommuniziert und ein Feature-Freeze beschlossen.

HTML5 hat einen enormen Schub für die Web-Apps ermöglicht. Die Entwicklung zeigt die Homepage http://html5readiness.com sehr gut auf. Sie illustriert sowohl die zeitliche Entwicklung als auch die Unterstützung bei den einzelnen Browsern.

Editor: Womit schreibt man Programme? Man braucht einen Texteditor um den Quellcode zu schreiben und zu bearbeiten. Für den Anfang reicht ein ganz einfaches Exemplar. Ein Editor ist, wie ein Browser, auf jedem Rechner vorhanden. Nicht jedoch auf Smartphones und Tablets. Unter Linux kann das „Leafpad“ sein, auf einem Mac „TextEdit“ und unter Windows „Editor“ oder „Notepad“. Bessere Editoren sind z.B.

- www.brackets.io
- www.geany.org
- https://code.visualstudio.com von Microsoft (relativ neu)
- Adobe Dreamweaver

Bessere Editoren, wie die hier erwähnten haben alle gute Funktionen:

- Syntax-Highlighting: farbliche Gestaltung des Quellcodes. Befehle, Variablen und Sonderzeichen werden jeweils in einer anderen Farbe präsentiert. Dadurch ist der Code übersichtlicher und Schreibfehler werden schnell erkannt. Falsch geschriebene Befehle werden nämlich erst gar nicht farblich markiert.
- Man kann beliebig viele Dateien gleichzeitig geöffnet haben.
- Code Folding: zusammengehörige Programmteile werden erkannt. Und über kleine quadratische Kästchen am Rand kann man die Teile zusammenklappen und auch wieder aufklappen. Damit wird das Programm übersichtlicher.
- Eine Website kann mit einem Klick auf ein bestimmtes Icon direkt im Browser geöffnet werden.
- Fertige Plugins erweitern die Fähigkeiten, wenn gewünscht.

KISS-Prinzip: Keep it simple, stupid!

Eine einfache Lösung ist einer komplexen, umständlichen Lösung vorzuziehen. Denn dieser hat meist weniger versteckte Fehler, ist einfacher zu lesen und einfacher zu warten.

Das Grundgerüst ist Pflicht.

Am Anfang gehört eine kurze Deklaration. Die sieht wie folgt aus:

```
<!DOCTYPE html>
```

Auf diese Weise erkennt jeder Browser sofort, dass nun ein Dokument in der Sprache HTML folgt. Eröffnet man z.B. mit dem Editor „Dreamweaver“ ein neues Dokument, wird diese Deklaration automatisch geschrieben und man muss es nicht selber machen (siehe unten).

Jedes HTML-Dokument wird sauber geöffnet und am Ende wieder geschlossen. Der Befehl ist zwingend vorgeschrieben:

```
<html>
Inhalt
</html>
```

1)Der Aufbau von HTML-Dokumenten:

- **<head> - der Kopfbereich**
  Dieser enthält zusätzliche Informationen zum Dokument. Das können sein, Titel, Name des Autors oder Hinweis auf die Sprache, relevante Stichwörter. Diese Angaben sind nicht verpflichtend notwendig. **Sie sind nicht sichtbar.** Der Kopf ist jedoch Pflicht in jedem HTML-Dokument. Selbst wenn man hier keine Angaben macht, muss man ihn deklarieren und dann leer lassen.

- **Charset – den Zeichensatz definieren**
  <meta charset=“utf-8“>
  Der Zeichensatz UTF-8 ist Standard im www.
  Deutsche Umlaute und Sonderzeichen sind kein Problem.

- **TITLE – dem Dokument einen Titel geben**
  Jedes HTML-Dokument benötigt einen Titel. Die meisten Suchmaschinen sortieren Dokumente nach ihrem Titel und werten diesen aus. Der <title> steht innerhalb des <head>-Bereiches.

- **BODY**
  Hier steht der Inhalt des Dokumentes. Alle sichtbaren Inhalte stehen hier.

- **H1 – Überschriften**
  H für Headline. Auch hier ist ein Start- und End-Tag notwendig.
  Besonderheiten:
  - Vor und nach der Überschrift erfolgt automatisch ein Zeilenumbruch mit einem größeren Zeilenabstand.
  - Der Text wird automatisch deutlich größer dargestellt.
  - Überschriften werden immer fett dargestellt.

Überschriften können in einer Abstufung von 1 bis 6 markiert werden. Überschriften dienen der Organisation im Text, für die Formatierung gibt es bessere Befehle. Wie groß eine Schrift genau dargestellt wird, kann man mit diesem Befehl nicht definieren. Die Angabe mit den Ebenen 1 bis 6 ist immer relativ zum Haupttext und der Browser entscheidet selbst, welche Schriftgröße er wählt. Lediglich die Relation zum Fließtext ist gleich. Die Einstellung der Eigenschaften erfolgt über CSS.

Info:

Leerräume, sogenannter Whitespace wird vom Browser ignoriert. Daher ist es sogar sinnvoll eine bessere Lesbarkeit des Codes zu erzeugen indem man Zeilenumbrüche mit der Enter-Taste erzeugt, da man dann mehr Übersicht hat und es keinen Einfluss auf die Ausgabe im Browser hat.

Nur ein einziges Leerzeichen zwischen zwei Wörtern wird berücksichtigt. Schreibt man mehrere Leerzeichen, wird trotzdem immer nur eins im Browser angezeigt. Ähnlich ist es bei Zeilenumbrüchen: sie werden überhaupt nicht bei der Darstellung im Browser berücksichtigt. Nur wenn ein Zeilenumbruch zwischen zwei Wörtern steht, wird es wie ein Leerzeichen interpretiert, aber auch nur, falls kein anderes Leerzeichen da ist.

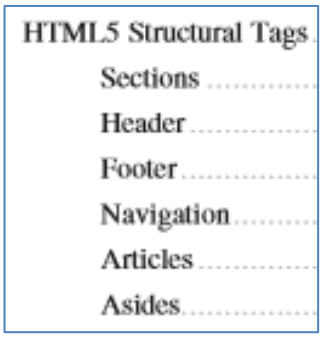
Will man mehr Zwischenraum haben, muss man ein „geschütztes“ Leerzeichen einfügen, oder eben mehrere. Dafür kann man folgendes eingeben:

** **

Das bedeutet eigentlich „**non breacking space**“. Leichter geht es mit der Tastenkombination

STRG + Shift + Leertaste

Semantik in HTML5



Zu diesem schon immer vorhandenen Aufbau kommen in HTML5 spezielle Strukturen hinzu, die man unter dem Begriff „SEMANTIK“ zusammenfasst. Diese sind in jede moderne Website aufzunehmen.

Diese HTML5-Elemente bieten eine weit bessere Möglichkeit, den **Quellcode zu strukturieren** als früher, als man das auf Basis von div-Elementen durchführte. Nun bezeichnen die Elemente nämlich allein durch ihren Namen bereits die Art ihres Inhalts. Zum Beispiel bezeichnet „section“ einen Abschnitt eines zusammengehörigen Textes oder „nav“ eine Navigation (Menü).

Diese Elemente sollen Suchmaschinen helfen, die Inhalte richtig zu deuten.

- **SECTION – Sinnabschnitte definieren**
  Große Dokumente werden oft nicht durch Überschriften unterteilt, sondern durch Abschnitte oder Kapitel. Ein Dokument kann in beliebig viele Sektionen unterteilt werden, um es so besser zu strukturieren. Diese Einteilung hat aber keine Auswirkung auf die Optik. Der Browser stellt den Text nicht anders dar, nur weil sie logisch voneinander getrennt sind. Innerhalb einer Sektion sind alle HTML-Befehle wie gewohnt erlaubt. Das <section> Element hat sinnvollerweise (aber nicht zwingend) eine eigene Überschrift. Es kann <header>, <footer> und <aside> enthalten.

- **NAV – Navigationselemente kennzeichnen**
  Damit werden Navigationselemente klar deklariert. Das <nav>-Element sollte für wichtige Navigationsblöcke wie die Hauptnavigation einer Website, die Service- oder Footer-navigation verwendet werden. Es eignet sich somit auch zur Kennzeichnung einer kurzen Linksammlung. Es kann daher durchaus mehrfach in einer Seite eingesetzt werden.
  Der NAV-Befehl bringt keinerlei optische Wirkung mit sich. Trotzdem bringt er Vorteile, da das Dokument besser strukturiert ist. Außerdem lassen sich später mittels CSS sehr gezielt diese Elemente mit einem schönen Layout steuern.

- **HEADER**
  Der <header> ist nicht zu verwechseln mit dem <head>. Es ist mehr ein Einführungsbereich und bestens geeignet, die oberste H-Überschrift, einen Untertitel, ein Logo oder Ähnliches zu platzieren. Sind mehrere <section> oder <article> vorhanden, lässt sich problemlos in jeder einzelnen ein <header> platzieren. Somit lässt sich ein <header> problemlos mehrfach auf einer Seite verwenden.

- **FOOTER**
  Das Gegenstück zum <header> ist der <footer>. Damit deklariert man einen Fußbereich innerhalb einer Sektion oder eines Artikels. Er eignet sich sehr gut für den Namen des Autors, Links, Urheberrechtsangaben, Copyrights usw.
  Genau wie der <header> handelt es sich nicht um eine klassische Fußzeile wie in der Textverarbeitung. Es ist nicht nötig, einen <header> zu erstellen, damit man einen <footer> erstellen darf, dieser ist somit auch ohne diesen möglich.

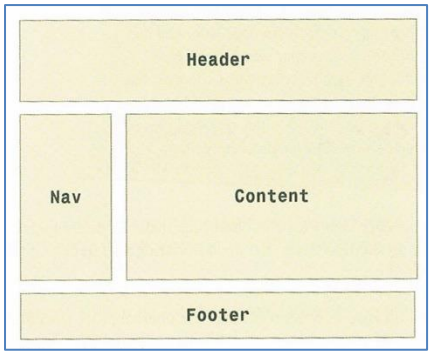
- **ARTICLE**
  Dient zur Strukturierung des Dokuments in Abschnitte und Texte. Er ist für in sich abgeschlossene Artikel gedacht, was für sich alleine stehen kann, wie z.B. Blog- oder Newsartikel oder auch Forenbeiträge. Ein <article>-Element kann in einzelnen <section>-Elemente untergliedert werden, um z.B. einen langen Bericht in mehrere Kapitel zu unterteilen.

- **ASIDE**
  Mit dem <aside>-Element werden ergänzende Informationen ausgezeichnet, die im Kontext nicht zwingend nötig sind. Das kann z.B. ein Zitat sein oder weiterführende Links oder Ähnliches. Ein typischer Einsatz wäre die sogenannte „Marginalie“ (eine Bemerkung die nur am Rand steht). Sie wird oft rechts von einem <article>-Element verwendet.

passender Code dazu:

```
41
42                  <aside>
43                      <h3>Über das Buch</h3>
44                      <p>Die 2. Auflage von "Responsive
    Webdesign - Anpassungsfähige Websites programmieren und
    gestalten" erschien Ende 2014 im Galileo Press Verlag.</p>
45
46                  </aside>
47              </div><!-- .main // Ende -->
```

Weiteres Beispiel:




```
<body>
        <header><div id="header"></div></header>
        <nav><div id="nav"></div></nav>
        <div id="content"></div>
        <footer><div id="footer"></div></footer>
</body>
```

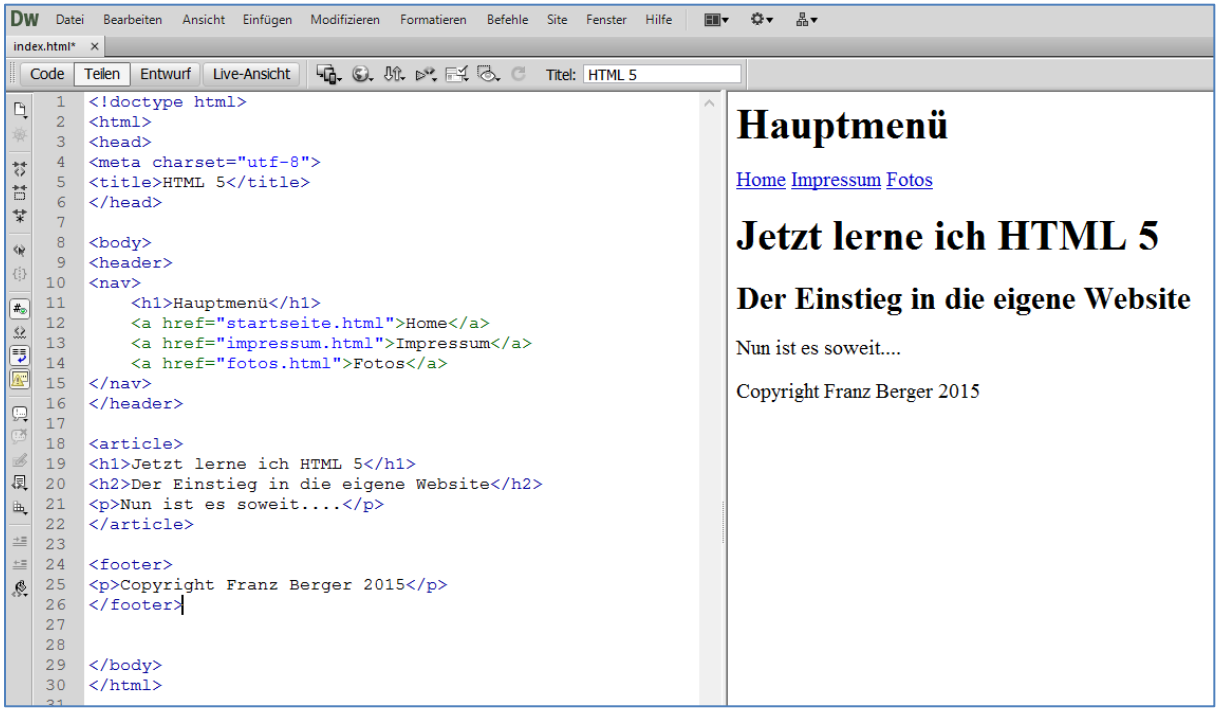
Weblinks zu den HTML5-Elementen:
http://www.html5rocks.com
http://www.html5doctor.com
http://www.selfhtml5.org

Startseite benennen:

Die Startseite wird standardmäßig angezeigt, wenn ein Besucher die Website aufruft. Wie diese Seite benannt sein muss schreibt eigentlich der Webspace-Provider vor. Aber zu 99,9% ist dies immer

- index.html
- default.html
- index.php
- index.asp

Übung:



Ordner und Dateien benennen

- Kleinschreibung verwenden
- Keine Umlaute
- Keine Sonderzeichen wie z.B. ß,=,@ oder %
- Keine Leerzeichen

2)Auszeichnungen für Absätze, Wörter und spezielle Elemente

- **P – Absätze erzeugen**
  Paragraph-Befehl <p>
  Um einen Text sinnvoll zu strukturieren muss man Absätze einfügen.
  Am besten man schreibt reinen Fließtext ausschließlich innerhalb des <p>-Befehls und vermeidet es, einen Text ohne jede Auszeichnung zu schreiben. Man sollte den <p> Befehl auch nicht verwenden, um nur einen Umbruch zu erzeugen, dafür ist er nicht gedacht.

- **BR – Zeilenumbrüche einfügen**
  Break-Befehl stellt eine Besonderheit dar, da er nicht geöffnet oder geschlossen werden muss. Er steht für sich alleine.

  NOBR oder WBR – Zeilenumbrüche steuern **(=weniger wichtig, nur zur INFO)**
  Das Gegenstück zu
 heißt <nobr> und verhindert, dass der Text umgebrochen wird. Dieser Befehl wird selten eingesetzt, dient eher dazu, etwas längere Elemente, wie z.B. Zitate oder Internetadressen zu in einer Zeile zu halten.
  Vor und nach dem <nobr> Befehl wird automatisch eine neue Zeile angefangen.
  Beispiel:

```
<nobr>
Dieser Text wird im Browser nicht umgebrochen. Der Besucher muss also nach rechts scrollen…
</nobr>
```

  <wbr> - Befehl:
  Damit lassen sich in langen Wörtern Umbruchoptionen deklarieren. Sie geben dem Browser an, wo er das Wort umbrechen darf. Ist ein Zeilenumbruch nicht erforderlich, hat der Befehl keine Auswirkung. Der Befehl steht für sich alleine, er muss nicht geöffnet und geschlossen werden.
  Beispiel:

```
<p>
Donau<wbr>damspschiff<wbr>kapitän.
</p>
```

- **HR – Linie**
  <hr> erzeugt eine Linie. Dieser Befehl steht für sich alleine.
  Vor und hinter der Linie wird automatisch ein Absatz eingefügt, sodass die Linie immer in einer eigenen Zeile steht.

- **Kommentar einfügen**
  In spitzen Klammern, mit jeweils zwei Bindestrichen am Anfang und am Ende sowie einem Ausrufezeichen am Anfang.

```
<!-- Hier steht ein Kommentar -->
```

- **EM und STRONG** – wichtige Wörter betonen
   steht für „Emphasis“ (= Betonung).
  Alle Browser heben die so markierten Wörter deutlich hervor, meist durch einen Fettdruck oder kursive Darstellung.
  - strong bedeutet »stark hervorheben« und wird in visuellen Browsern meist fett gedruckt.

- em hingegen wird meist kursiv dargestellt.

Als Faustregel benutze strong, um den Text bereits vor dem Lesen hervorzuheben, und em, wenn er erst während des Lesens auffallen soll.
 stellt eigentlich eine Steigerung zum Befehl dar, wird aber leider auch nur fett oder kursiv dargestellt.

Beispiel:

```
<p>
Der Chef schrie <em> laut </em> auf.
</p>
```

- **B – Wörter hervorheben**

  Zum Hervorheben eines Wortes im Text. Früher stand der Befehl <b> für „bold“, also fett, was jetzt nicht so gemeint ist.
  Beispiel:

```
<p>
Viele Browser unterstützen <b>HTML5</b> noch nicht
vollständig.
</p>
```

3)Aufzählungen und Listen

- **UL – unsortierte Listen (unordered list)**

  Die einzelnen Punkte werden in dieser <ul> einfach als Listeneintrag <li> (=list item) deklariert. Diese müssen nicht zwingend geschlossen werden, um einen sauberen Code zu erhalten, sollte man es trotzdem tun.
  Jede Liste wird automatisch ein wenig formatiert, damit sie sich deutlich vom Fließtext abhebt:
  - Vor und nach der Liste wird ein Absatz eingefügt.
  - Alle Listeneinträge erhalten eine eigene Zeile.
  - Vor jedem Listeneintrag wird ein Punkt als Aufzählungszeichen gesetzt.

  Man hat aber keinen Einfluss auf das Aussehen der Liste, weder Abstände noch die Einrückungen kann man bestimmen. Das geht erst mit Hilfe von CSS.

- **OL – nummerierte Listen (ordered list)**

  Beispiel:

```
<ol>
<li>Texteditor</li>
<li>Webbrowser</li>
</ol>
```

1. Texteditor
2. Webbrowser

  Eine nummerierte Liste beginnt immer mit der Nummer 1. Das lässt sich ändern. Hierzu dient das Attribut „start“, ab dem die Einträge normal weitergezählt werden.

```
<ol start="5">
```

- Listen ineinander verschachteln

  Das funktioniert bei <ul> als auch bei <ol>.

Beispiel für <ol>:
In HTML wird jede Unterliste für sich alleine durchnummeriert. Es gibt keine durchgehende Nummerierung wie z.B. 1.1.1.

Übung: Erstelle folgende <ol> - Liste:

Wir benötigen folgende Tools:

1. Texteditor
    1. Dreamweaver
    2. Notepad++
2. Webbrowser
    1. Firefox
    2. Google Chrome

Lösung:

```
<h2>Wir benötigen folgende Tools:</h2>
<ol>
   <li>Texteditor</li>
          <ol>
               <li>Dreamweaver</li>
               <li>Notepad++</li>
          </ol>
     <li>Webbrowser</li>
          <ol>
               <li>Firefox</li>
               <li>Google Chrome</li>
         </ol>
</ol>
```

Übung:

Ändere die obige nummerierte Liste in eine unsortierte Liste.

4)Links – auf andere Websites verweisen

a)Sprungmarken und Anker

Alle Browser zeigen Links standardmäßig in einer blauen Schriftfarbe an und unterstreichen den Text.

Das Prinzip ist einfach: Es gibt einen Startpunkt und ein Ziel.

- Der Startpunkt wird als „Sprungmarke“ bezeichnet, das Ziel als „Anker“. Diese beiden müssen genau definiert sein, damit ein Link funktioniert.
- Der Anker bildet das Ziel. Meistens stellt ein Anker eine Webseite dar, wie z.B. http://www.superclass.eu
  Es kann aber auch eine Datei und ein Verzeichnis sein, eine PDF-Datei, ein Word-Dokument oder ein Programm, das dann heruntergeladen werden kann.

Der HTML-Befehl für alle Links lautet „a“ für „anchor“ (=Anker). Das bedeutet aber nicht, dass dies der Anker ist, sondern dass man nun auf einen verweist. Der Wert des Ankers, also die Zieladresse, gibt man mit dem Attribut „href“ an. Dies steht für „Hyper reference“.

```
<a href="http://www.superclass.eu">Link zur Website</a>
```

Den Link-Text kann man frei bestimmen. Er wird als anklickbarer Link sichtbar.

Natürlich kann man auch auf eine bekannte Unterseite verweisen, wie z.B.:

```
<a href="http://www.superclass.eu/audacity.html"> Link zu Audacity-Informationen </a>
```

Ziel zum Öffnen von Links angeben:

Der Browser lädt die neue Seite immer im aktuellen Fenster, sodass die Ursprungsseite, auf der der Link angeklickt wurde, automatisch verlassen wird.
Will man einen Link in einem neuen Fenster öffnen, dann sieht der Besucher in einem Fenster weiterhin die erste Homepage und in einem zweiten Browserfenster die neue Website.

Als Attribut wird „target“ (=Ziel) verwendet. Als Ziel sieht HTML zwei Standartwerte vor:

- _blank es wird ein neues Fenster geöffnet (blank = leer)
- _self es wird im selben Fenster geöffnet (self = selbst)

Beispiel:

```
<a href="http://www.superclass.eu" target="_blank">Link zur Website</a>
```

b)E-Mails per Klick erstellen

Klickt ein Besucher auf einen E-Mail-Link öffnet sich sein E-Mail Programm und setzt in das Empfänger-Feld die im Link hinterlegte Adresse ein. Somit kann man ganz einfach mit jemandem in Kontakt treten, ohne dass man extra eine Adresse abtippen muss.

```
<a href="mailto:adresse@gmx.at"> E-Mail schreiben</a>
```

Man muss also nur das Protokoll mailto: angeben. Außerdem muss der doppelte Slash (//) weggelassen werden.

Zusätzlich bietet HTML die Option Parameter an die Mail anzuhängen. Dazu muss man lediglich ein Fragezeichen (?) anhängen. Damit können spezielle Parameter übergeben werden. Mit einem kaufmännischen „und" (&) können mehrere Parameter kombiniert werden, wie z.B. Betreff (=subject) und ein kurzer Mail-Text (=body).

Übung:
Sinnvoll ist es, gleich einen Betreff hinzuzufügen, damit man in der Mailbox weiß, dass man Post von der eigenen Website bekommen hat.

 E-Mail schreiben

Übung:
Betreff und kurzer Text:
 E-Mail schreiben

c)Hyperlinks auf andere Dokumenttypen erstellen

- Textdatei: <a href="text.txt"> eine Textdatei wird direkt im Browser als einfacher Text angezeigt
- Grafik: <a href="grafik.jpg"> Die Grafik wird in den Browser geladen und angezeigt.
- Video: a href="video.mpg"> existiert ein entsprechendes Programm, wird dieses geöffnet und das geladene Video abgespielt.
- Musik: <a href="musik.mp3">
- Andere Formate: (pdf, docx) <a href="dokument.pdf">

5)Bilder in die Webseite einbauen

Der Befehl <img> bedeutet „image“ (=Bild) und steht für sich alleine und muss daher nicht geschlossen werden.

Die Angabe der gewünschten Bilddatei erfolgt durch das Attribut „src“ was für „Source“ (=Quelle) bedeutet.

```
<img src="bild.jpg">
```

Liegt das Bild im selben Verzeichnis wie die html-Datei genügt diese obige Angabe.

Meistens liegen die Bilder gut geordnet in einem eigenen Ordner, wie z.B. „images“. Daher muss dieses Verzeichnis mit angegeben werden:

```
<img src="/images/bild.jpg">
```

Gibt man das Verzeichnis mit zwei Punkten an (..) steht dies für ein Bildverzeichnis in der nächsthöheren Ebene

```
<img src="../images/bild.jpg">
```

Der HTML-Standard schreibt zu jedem Bild vor, einen alternativen Text anzugeben. Dies wird mit dem Attribut <alt> gemacht:

```
<img src="/images/bild.jpg" alt="Dies ist unsere Schule in
Mistelbach">
```

a)Die Größe der Bilder anpassen

Jeder Webbrowser zeigt Bilder in ihrer Originalgröße an. Bei kleinen Bildern kein Problem, werden aber übergroße Bilder im Browser geöffnet, verkleinert es der Browser so, dass es vollständig angezeigt werden kann.

HTML bietet die Möglichkeit, die Anzeigegröße von Bildern direkt zu beeinflussen. Hierzu dienen die Attribute WIDTH für die Breite und HEIGHT für die Höhe. Sie werden direkt innerhalb des IMG-Tags als ganze Pixel-Zahl angegeben.

```
<img src="/images/bild.jpg" width="300" height="250"
alt="Dies ist unsere Schule in Mistelbach">
```

Tipp: Hat man ein zu großes Bild und will eine gewünschte Breite erreichen, aber nicht die verzerrungsfreie Höhe ausrechnen verwendet man **einfach „auto“**

```
<img src="/images/bild.jpg" width="300" height="auto">
```

Vorteil: man kann ein Bild in verschiedenen Dokumenten aber auch im selben Dokument jeweils mit einer anderen Größe angeben, obwohl man immer auf dieselbe Datei zugreift.

b)Grafische Links erstellen

Dabei werden Abbildungen als Links verwendet. In HTML 5 erfolgt dies durch eine Kombination des <a>-Links und durch Einfügen des gewünschten Bildes anstelle des Textes.

```
<a href="http://www.homepage.at"><img src="/images/bild.jpg"
alt="Bildbeschreibung"></a>
```

Es darf auch der Befehl <target> verwendet werden, um ein Ziel für den Link anzugeben, z.B. `target ="_blank"` für ein neues Fenster.

Alle Browser sind bemüht, sämtliche Links in einem Dokument sichtbar zu machen und stellt diese in einer blauen Umrahmung dar. Um diese unschönen blauen Rahmen abzuschalten, gibt es in HTML den Befehl <border>.

Füge das BORDER-Attribut einfach dem IMG-Tag hinzu und setze den Wert auf NULL. Der Rahmen hat somit eine Breite von Null und ist somit unsichtbar.

```
<a href="http://www.homepage.at"><img src="/images/bild.jpg"
alt="Bildbeschreibung" border="0"></a>
```

6)Tabellen

Um Daten übersichtlich zu präsentieren, gibt es spezielle Befehle für den Tabellenkopf, den Fußbereich, Spaltenüberschriften, Tabellenüberschriften und Ähnliches.

Tabellen werden mit dem Befehl <table> definiert. Manche Browser zeigen bei Tabellen immer einen Rahmen an, manche nicht.

Mit dem Befehl <border> (=Rand oder Rahmen) blendet der Browser immer einen Rahmen für die Tabelle ein. Ohne weitere Angaben hat der Rand die Breite 1.

```
<table border>
  Inhalt
</table>
```

<tr> und <td> - Reihen und Spalten

<tr> Tabellenreihe, damit legt man die Zeile fest
<td> Tabellenspalte, damit legt man die Spalte fest

<th> Header

Grundregeln:

- Man muss zuerst eine <tr> öffnen, erst dann kann man eine <td> einfügen. Umgekehrt ist es nicht möglich.
- Innerhalb von <td> steht der Inhalt.

a)Spaltenüberschriften hinzufügen

Hierzu dient der Befehlt <th> (=table head). Er wird genau wie der Befehl für eine Spalte verwendet. Allerdings sollte er nur in der ersten Zeile vorkommen. Alle Browser heben diese Zeile optisch hervor und geben mehr Struktur.

```
<table border>
<tr>
        <th>Browser</th>
        <th>Werkzeuge</th>
</tr>
<tr>
        <td>Internet Explorer</td>
        <td>Computer</td>
</tr>
<tr>
        <td>Firefox</td>
        <td>Text-Editor</td>
</tr>
</table>
```

Ergebnis:

Browser	Werkeuge
Internet Explorer	Computer
Firefox	Text-Editor

b)Überschrift für die Tabelle

Hierzu dient der Befehl <caption>. Er muss direkt nach dem Eröffnen der Tabelle, vor der ersten Zeile, gesetzt werden.
Dieser Befehl erlaubt aber auch einen längeren Einführungstext. Daher sind innerhalb des <caption>-Befehls auch grundlegende Formatierungen erlaubt.

Beispiel:

```
<table border>
<caption>
     <p><em>Grundausstattung</em></p>
     <p>Man benötigt nur einen Browser, einen Computer und
einen Texteditor. Mehr ist nicht nötig</p>
</caption>
<tr>….
```

Ergebnis:

Grundausstattung

Man benötigt nur einen Browser, einen Computer und einen Texteditor. Mehr ist nicht nötig

Browser	Werkeuge
Internet Explorer	Computer
Firefox	Text-Editor

c)Tabellenzellen miteinander verbinden <colspan>

Mit dem Befehl <colspan> kann man einer Zelle mitgeben, dass diese in dieser Zeile zwei oder mehr Einzelspalten einnehmen soll. Der Befehl kann ausschließlich innerhalb eines <td>- oder <th>-Tags verwendet werden, weil mit ihm angegeben wird, dass diese Zeile so breit wie zwei oder mehr Spalten sein soll.

Die Spalten, die in eine Zelle zusammengefasst werden, müssen nebeneinander liegen.

Übung: Ersetzte die Begriffe „Browser“ und „Werkzeuge“ durch den einen Begriff „Tools“, der in einer einzigen Zelle stehen soll.

Lösung:

```
<tr>
    <th colspan="2">Tools</th>
</tr>
<tr>
    <td>Internet Explorer</td>
    <td>Computer</td>
</tr>
<tr>
    <td>Firefox</td>
    <td>Text-Editor</td>
</tr>
</table>
```

Tools	
Internet Explorer	Computer
Firefox	Text-Editor

Beachte:

Die Gesamtzahl der Spalten, also tatsächliche Spalten plus <colspan>, muss weiterhin passen und darf nicht mehr oder weniger ergeben, als die restlichen Zeilen der Tabelle. Ansonsten zeigt die Tabelle ein merkwürdig verschobenes Gebilde an.

d)Zellen über mehrere Zeilen spannen <rowspan>

Mit dem Befehl <rowspan> wird eine Zelle auf die Höhe von mehreren Zeilen gebracht. Dieser Befehl kann nur innerhalb des <td> und <th>-Tags verwendet werden.

7)Multimedia – Videos und Musik

a)Objekte einbinden

Der wichtigste Befehl zum Einbinden von multimedialem Inhalt lautet in HTML5 <object>.

Innerhalb dieses Befehls müssen folgende weiteren Attribute angeführt werden:

- type — damit gibt man dem Browser an, um was für einen Inhalt es sich überhaupt handelt
- width & height — damit wird die Anzeigengröße des eingebundenen Objekts bestimmt
- data — damit gibt man den Dateinamen an

```
<object type="image/jpg" data="foto1.jpg" width="400"
height="300">
</object>
```

Ein Objekt kann auch eine Textdatei sein.

Die Breite und Höhe geben dabei die Größe des Fensters an, innerhalb dessen der Text angezeigt werden soll. Beachte jedoch, dass der Text ohne jede Formatierung dargestellt wird. Soll er mit einer Formatierung dargestellt werden, schreibe ihn in eine HTML-Datei. Lege Begrenzungen mit den üblichen Befehlen wie <section> , <article> <p> usw. fest. Dann erscheint der Text in herkömmlicher Darstellung im HTML-Dokument. Die Datei benötigt keinen Header ohne Ähnliches, weil das über die Hauptdatei gesteuert wird.

```
<object type="text/plain" data="artikel.txt" width="400"
height="300">
</object>
```

Attribut <name>

Mit dem Attribut <name> innerhalb des Object-Befehls kann man dem Objekt einen Namen geben. Das hat vor allem in größeren Datenbanken einen Sinn.

```
<object type="text/plain" data="artikel.txt" width="400"
height="300"><param name="Bewerbungsvorlage von Fr. Huber">
</object>
```

b)Videos einbinden

In HTML5 gibt es einen eigenen Befehl nur für Videos. Mit ihm lassen sich alle Film- und Videodateien direkt in die Webseite einbinden. Der Webbrowser kann dann mit HTML5-Unterstützung die Dateien sogar selbst abspielen, da sie einen kleinen Player bereits eingebaut haben.

Der Befehl lautet <video>. Im einfachsten Fall muss man lediglich das Attribut <src> und die Größe mit <width> und <height> festlegen.

```
<video src="videodatei.mp4" width="400" height="300" controls>
</video>
```

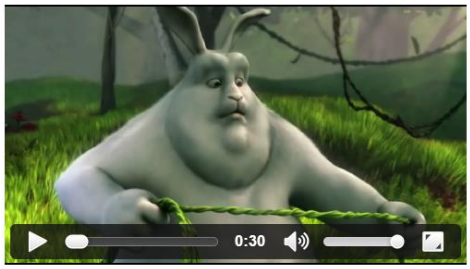
Leider spielen manche Browser die Steuerelemente nicht immer an, daher ist es sinnvoll das Attribut <**controls**> einzublenden. Dann kann der Besucher das Video starten, stoppen oder die Lautstärke des Tons verändern.

<poster>

Wird das Video gerade nicht abgespielt, zeigt es das erste Bild als Vorschau. Um das zu ändern, gibt es in HTML5 einen eigenen Befehl, mit dem sich eine beliebige Vorschaugrafik einblenden lässt. Der Befehl heißt <poster> und er muss als Wert den Link zu einer gültigen Grafikdatei besitzen. Die Grafik kann z.B. ein Filmplakat sein. Achte aber darauf, dass das Bild dieselbe Größe wie das Video hat, da sonst manche Browser die Darstellung verzerren.

```
<video src="videodatei.mp4" width="400" height="300" controls
poster="vorschau.jpg"> </video>
```

Ergebnis:



Weitere Videobefehle:

- preload — damit wird der Browser sofort beim Laden der HTML-Seite auch das Video herunterladen und im Puffer zwischenspeichern
- autoplay — damit startet der Browser automatisch die Wiedergabe des Videos, sobald die Seite aufgerufen wird
- loop — damit wird der Browser angewiesen, das Video in einer Endlosschleife abzuspielen

```
<video src="videodatei.mp4" width="400" height="300"
controls poster="vorschau.jpg" preload autoplay loop>
</video>
```

c)Audio – der Befehl speziell für Musikinhalte

Der Befehl <audio> ist von der Syntax dem <video> Befehl vergleichbar.

Man sollte den Befehl <controls> verwenden, damit der Player sichtbar ist und der Benutzer die Möglichkeit hat, die Wiedergabe zu starten.

```
<audio type="audio/mpeg" src="musik.mp3" controls>
</audio>
```

Ansicht bei Firefox

Ansicht bei Chrome

Im Dreamweaver wurde es in der Live-Ansicht nicht dargestellt, nur im Browser.

Auch hier können die Zusatzbefehle wie bei <video> sinnvoll verwendet werden:

- autoplay — die Wiedergabe beginnt, sobald die HTML-Seite geladen wurde. Das ist für Hintergrundmusik geeignet, ob es vom Besucher immer gewünscht wird, ist fraglich.
- loop — damit wird die eingebundene Musikdatei in einer Endlosschleife abgespielt
- preload — damit lädt der Browser die Audiodatei sofort beim Aufrufen der Website in seinen internen Cache. Sie muss daher nicht live gestreamt werden.

```
<audio type="audio/mpeg" src="musik.mp3" controls preload
autoplay loop>
</audio>
```

8)Scrollbalken

Sollen Scrollbalken nur genau dann angezeigt werden, wenn sie benötigt werden, muss man folgendes verwenden:

overflow: auto

Das genaue Verhalten ist nicht vom Standard vorgeschrieben, sondern es wird der Browser nur bei überlaufendem Inhalt den Scrollbalken-Mechanismus anbieten.

9)Eigene semantische Auszeichnung mit IDs und Klassen

Wenn man spezielle Kennzeichnungen vornehmen möchte, aber der „Wortschatz“ von HTML dafür nicht ausreicht (obwohl fast 100 Elemente in HTML vorhanden sind, wie z.B. <input>, , <adress>, <aside>, <img>, <textarea> usw.) nutzt man IDs und Klassen.

Beispiel:
Man möchte eine Bestellnummer als solche kennzeichnen <bestellnummer>. Dieses fehlende gewünschte Element kann als „class“ emuliert werden: . Auf dieses kann man später nochmals zugreifen.

id- und class-Attribute.:
Die neutralen Elemente und können einen „persönlichen“ Identifizierer oder eine Klasse zugewiesen bekommen, in Form der id- und class-Attribute.

```
<span class=“bestellnummer“>2016-1234</span>
```

Grundregeln:

- wähle den Bezeichner nach semantischen Gesichtspunkten, nicht nach gewünschter Präsentation (class=“wichtig“ ist besser als class=“roteSchrift“).
- Darf die Information nur einmal im Dokument auftreten, wähle das id-Attribut zur Kennzeichnung.
- Tritt die Information mehrfach auf, wähle das class-Attribut, wie z.B. für eine Bestellnummer.
- Bildet die Information einen eigenständigen Block, nutze das -Element als Basis.
- Tritt die Info im Fließtext (also „inline“) auf, baue es auf das -Element auf.

Weblinks:

http://html5demos.com

http://w3schools.com sehr gute Info