Hypertext Markup Language und Cascading Style Sheets

Inhaltsverzeichnis

1 Hypertext Markup Language (HTML)

HTML, vom Web-Gründer Tim Berners-Lee entwickelt, wurde zum erfolgreichsten und meist verbreiteten Dateiformat der Welt. Die aktuelle Version von HTML ist HTML5.

Mit HTML kann man z.B. Überschriften, Textabsätze, Listen und Tabellen erzeugen. Man definiert oft anklickbare Verweise (=Hyperlinks) für den Zugriff auf beliebige andere Web-Seiten im Internet. Zusätzlich werden manchmal Formulare in den Text integrieren und HTML bietet Schnittstellen für sehr viele Erweiterungssprachen wie z.B. Cascading-Style-Sheets (CSS) oder JavaScript an, mit deren Hilfe man HTML-Elemente nach Wunsch gestalten und Interaktionen mit dem Anwender realisieren kann.

Kann der Benutzer durch Interaktionen einen Einfluss auf die Web-Seite ausüben, spricht man von einer grafischen Benutzeroberfläche (= Graphical User Interface, GUI) in einer Webanwendungen (=Webapplikation, Web-App).

Das konkrete Aussehen, die Gestaltung einer mit HTML erstellten Benutzeroberfläche sollte ausschließlich mit Cascading-Style-Sheets festgelegt werden. Das W3-Konsortium ist für die Standardisierung von HTML zuständig

1.1 Kleines Lexikon

HTML steht für „Hypertext Markup Language“ und ist das Format, in dem die Text- und Hypertext-Informationen im WWW (World Wide Web) gespeichert und übertragen werden.

Unter **Hypertext** versteht man Texte mit Querverweisen, die ähnlich wie in einem Lexikon oder in einer Literaturliste die Verbindung zu weiteren Informationen herstellen.

Im WWW wird ein solcher Verweise mit Hilfe eines **URL** (Uniform Resource Locator) realisiert. Eine derartige Quelle wird oft auch als **URI** (Uniform Resource Identifier) bezeichnet. URL ist die „Adresse“, die das Client-Programm benötigt, um eine bestimmte Information vom jeweiligen Server-Computer zu erhalten. Der URL enthält zu diesem Zweck Informationen wie die Art des Zugriffs (Protokoll), die Adresse des Server-Computers (Hostname), eventuell mit einem Username und Passwort oder einer Port-Nummer, und das Directory (Ordner) und den Filenamen der Datei, in der die gewünschte Information gespeichert ist.

Das **WWW** ist ein Informationssystem, das einen Zugriff auf Informationen, die auf vielen verschiedenen Computern gespeichert sind ermöglicht. Der Zugriff erfolgt nach dem Prinzip von Client und Server über das Internet mit dem Protokoll HTTP (Hypertext Transfer Protocol).

Server sind Computer, auf denen die Informationen gespeichert sind. **Clients** sind die Benutzer, die Informationen haben wollen. Client-Programme sind Programme, mit denen die Benutzer von ihren eigenen Rechnern (PCs) aus auf die Informationen, die auf den Servern gespeichert sind, zugreifen. WWW-Client-Programme werden auch als Web-Browser bezeichnet.

Web-Browser laufen meist auf einer grafischen Benutzeroberfläche mit Maus oder Touch-Screen. Spezielle Browser-Programme können die Informationen auch in Zeilen-orientierten oder in Blindenschrift oder akustisch (als gesprochener Text) ausgeben. Typische Web-

Browser sind z.B. Mozilla Firefox, Google Chrome, Microsoft Edge, Internet Explorer, Opera, Safari,…

Unter einem **HTML-Parser** versteht man eine Software, die HTML-Syntax analysiert und in eine strukturierte Oberfläche und Text umwandelt. Jeder Web-Browser verfügt über einen HTML-Parser.

1.2 Grundgerüst einer HTML-Datei

Eine gewöhnliche HTML-Datei besteht grundsätzlich aus folgenden Teilen:

Dokumenttyp-Angabe (Angabe zur verwendeten HTML-Version)

Header (Kopfdaten. z.B. Angaben zu Titel u.ä.)

Body (Körper - Inhalt, also Text mit Überschriften, Verweisen, Grafikreferenzen usw.)

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Der Titel des Dokuments</title>
</head>
<body>
Der Inhalt des Dokuments …
</body>
</html>
```

In der Praxis sind die HTML-Dokumenttypen wichtig, weil sie bestimmen, wie ein Browser ein HTML-Dokument darstellt.

Verwendete HTML-Tags

Der gesamte übrige Inhalt einer HTML-Datei wird in die Tags `<html>` bzw. `</html>` eingeschlossen. Das `html`-Element wird auch als Wurzelelement einer HTML-Datei bezeichnet.

Hinter dem einleitenden HTML-Tag folgt das einleitende Tag für den Kopf `<head>`. Zwischen diesem Tag und seinem Gegenstück `</head>` werden Kopfdaten einer HTML-Datei notiert. Die wichtigste dieser Angaben ist der Titel der HTML-Datei, markiert durch `<title>` bzw. `</title>`.

Unterhalb davon folgt der Textkörper, markiert durch `<body>` bzw. `</body>`. Dazwischen wird dann der eigentliche Inhalt der Datei notiert, also das, was im Anzeigefenster des WWW-Browsers angezeigt werden soll.

1.3 Meta-Tag

In Meta-Angaben können Sie verschiedene nützliche Anweisungen für Webserver, Browser und Suchmaschinen-Programme notieren. Meta-Angaben können z.B. Angaben zum Autor, zum Inhalt der Webseite und zur Zeichenkodierung enthalten. Sie können aber auch HTTP-

Kommandos absetzen, zum Beispiel zum automatischen Weiterleiten des Web-Browsers zu einer anderen Adresse.

Leider wird die neue, einfache HTML5-Angabe zur Zeichenkodierung nicht von älteren Browsern erkannt. Deshalb ist es sicherer, zusätzlich die ältere Notationsweise anzugeben. Es spricht nichts dagegen, beide Angaben zu verwenden:

```
<head>
<meta charset="UTF-8">
<meta http-equiv="content-type" content="text/html; charset=utf-8">
<!-- ... andere Angaben im Dateikopf ... -->
</head>
```

UTF-8 ist der de-facto-Standard für die Zeichenkodierung des Internets und damit verbundener Dokumenttypen.

1.4 Kommentar

Ein Kommentar ist zwischen `<!--  -->` zu finden und wird vom Webbrowser nicht ausgeführt.

1.5 Attribute

HTML-Tags können zusätzliche Angaben (=Attribute) enthalten:

Attribute mit freier Wertzuweisung, wobei jedoch ein bestimmter Datentyp oder eine bestimmte Konvention erwartet wird, z.B. `<table style="border:1;" >` (Tabelle mit Rahmen von 1 Pixel Stärke - hier wird eine numerische Angabe erwartet)

Attribute mit freier Wertzuweisung ohne weitere Konventionen, z.B. `<p title="Aussage mit Vorbehalt">` - hier kann ein ganzer Text zugewiesen werden.

Alleinstehende Attribute, z.B. `<input type="text" readonly>` (nur lesbares Textfeld).

Neben Attributen, die nur in bestimmten HTML-Elementen vorkommen können, gibt es auch so genannte **Universalattribute**, die in vielen HTML-Elementen erlaubt sind. Z.B. `<p id="Einleitung">Text</p>` Das Beispiel definiert einen Textabsatz mit den HTML-Tags `<p>` und `</p>`. Im `<p>`-Tag wird ein Universalattribut notiert, nämlich das Attribut `id=`. Damit kann man eindeutige Namen für einzelne HTML-Elemente vergeben.

Alle Werte, die du Attributen zuweist, müssen in Anführungszeichen stehen. Bei herkömmlichem HTML spielt es keine Rolle, ob die Attributnamen in Klein- oder Großbuchstaben notiert werden. In der neueren HTML-Variante, in XHTML, müssen die Attributnamen dagegen klein geschrieben werden.

1.6 Cascading Style Sheets

CSS (Cascading Style Sheets) ist eine Sprache und jede Sprache kennt Vokabeln und Grammatik. Der schematische Aufbau einer CSS-Regel und die wichtigsten Vokabeln dazu:

```
Selektor {
  Eigenschaft: Wert;
  Eigenschaft: Wert;
}
```

Um z.B. in einer Webseite die Hintergrundfarbe auf Rot und die Schriftfarbe auf Blau zu setzen ist folgende CSS-Regel erforderlich:

```
body{background-color:yellow;color:blue;}
```

Die Begriffe **Style**, **CSS-Regel**, **Gestaltungsanweisung** und **Formatvorlage** werden mehr oder weniger synonym gebraucht. Manche Autoren verwenden auch **Stil** oder **Stilregel**.

Eine CSS-Regel (style oder css rule) besteht aus folgenden Einzelteilen:

1.6.1 Selektor

Der Selektor (Selector) steht vor der geschweiften Klammer und wählt aus (selektiert), welche Elemente auf der Seite gestaltet werden sollen.

1.6.2 Deklaration

Zwischen geschweiften Klammern stehen eine oder mehrere Deklarationen (Declarations), auch Anweisungen genannt. Sie beschreiben die Gestaltung der Elemente, auf die der Selektor zutrifft. Jede Deklaration besteht aus einem Eigenschaft-Wert-Paar und wird mit einem Semikolon (Strichpunkt) beendet.

1.6.3 Eigenschaft

Die zu gestaltende Eigenschaft (Property, Farbe, Schriftart etc.) des Elements steht vor dem Doppelpunkt. Leerstellen vor und nach dem Doppelpunkt sind optional. Die Eigenschaft wird auch Attribut genannt.

1.6.4 Wert

Der Wert (Value), den die Eigenschaft annehmen soll, steht nach dem Doppelpunkt. Danach folgt ein Semikolon, um die Deklaration zu beenden.

1.7 Textformatierung

1.7.1 Farbe / Größe / Art

Soll ein Textbereich mit Farbe, Größe und Art formatiert werden, so kann direkt um diesem Textbereich ein Block-Element mit den entsprechenden CSS-Regel definiert werden.

```
<span style="color:green;font-size:40px;font-family:Helvetica;">
Mein Text
</span>
```

Eine bessere Lösung ist es, im Kopfbereich der Webseite im `<style>`-Bereich die entsprechenden CSS-Regel für die neue Klasse zum Beispiel `.mytext` zu definieren und beim Text zu nutzen:

```
<html><head><meta charset="UTF-8">
<title>Farbe-Größe-Art</title>
<style>
.mytext{color:green;font-size:40px;font-family:Helvetica;}
</style>
</head>
<body>
<span class="mytext">Mein Text</span><br>
</body></html>
```

Die optimale Lösung wäre die Auslagerung der CSS-Regel des `<style>`-Bereichs in eine externe Datei zum Beispiel `my.css` im selben Ordner/Verzeichnis. Diese CSS-Datei wird in der HTML-Seite mit dem `<link>`-Tag folgenderweise eingebunden:

```
<html><head><meta charset="UTF-8">
<title>Farbe-Größe-Art</title>
<link rel="stylesheet" type="text/css" href="my.css">
</head>
<body>
<span class="mytext">Mein Text</span><br>
</body></html>
```

1.7.2 Überschriften

```
<style>#ort{color:red;}</style>
…
<h1 id="ort">HTML die Sprache des Web</h1>
```

Das Beispiel zeigt eine Überschrift 1. Ordnung. Das einleitende Tag `<h1>` signalisiert, dass eine Überschrift 1. Ordnung folgt (*h* = *heading* = *Überschrift*). Das abschließende Tag `</h1>` signalisiert das Ende der Überschrift. Ein abschließendes Tag beginnt mit einem Schrägstrich "/".

Mit dem HTML-Universattribut `id` und der CSS_regel `#ort{color:red;}` wird die Farbe auf Rot gesetzt.

1.7.3 Textauszeichnung

```
<h1><i>HTML</i> - die Sprache des Web</h1>
```

Das i-Element steht für *italic* (= *kursiver Text*). Der Text zwischen `<i>` und `</i>` wird als kursiv interpretiert, abhängig von der eingestellten Schriftart und Schriftgröße für die Überschrift 1. Ordnung. Die Tabelle zeigt einige Textformatierungen:

`<b>...</b>`	zeichnet einen Text als fett aus
`<strong>...</strong>`	zeichnet einen Text als wichtig aus
`<i>...</i>`	zeichnet einen Text als kursiv aus
`<em>...</em>`	zeichnet einen Text als hervorgehoben aus

`<mark>...</mark>`	zeichnet einen Text als markiert aus
`<small>...</small>`	zeichnet einen Text kleiner als normal aus
`<del>...</del>`	zeichnet einen Text als durchgestrichen aus
`<ins>...</ins>`	zeichnet einen Text als eingefügt (unterstrichen) aus
`<sup>...</sup>`	zeichnet einen Text als hochgestellt aus
`<sub>...</sub>`	zeichnet einen Text als tiefgestellt aus

Elemente können ineinander verschachtelt werden. Auf diese Weise entsteht eine hierarchische Struktur. Komplexere HTML-Dateien enthalten sehr viele Verschachtelungen.

1.7.4 Umlaute und Sonderzeichen

In HTML gibt es noch viel mehr Sonderzeichen, als jene der nachfolgenden Tabelle. Wird kein UTF-8 verwendet, müssen auch die Umlaute und ß wie nachfolgend erzeugt werden:

Zeichen	Beschreibung	HTML-Code
ä	Umlaut ä	ä
Ä	Umlaut Ä	Ä
ö	Umlaut ö	ö
Ö	Umlaut Ö	Ö
ü	Umlaut ü	ü
Ü	Umlaut Ü	Ü
ß	ß-Zeichen	ß
©	Copyright-Zeichen	©
®	Registriermarke-Zeichen	®
™	Trademark-Zeichen	™
€	Euro-Zeichen	€
§	Paragraph-Zeichen	§

£	Pfund-Zeichen	£
°	Grad-Zeichen	°
µ	Mikro-Zeichen	µ
<	schließende spitze Klammer	<
>	öffnende spitze Klammer	>

1.8 Textstrukturierung

1.8.1 Zeilenumbruch

`<br>` bewirkt einen manueller Zeilenumbruch (*br = break = Umbruch*).

1.8.2 Absatz

Absätze (*p = paragraph*) dienen der optischen Gliederung eines Textes. Textabsätze werden linksbündig ausgerichtet, wenn nichts anderes festlegt ist. Dieses Block-Element `<p>` beginnt in einer neuen Zeile und bewirkt, dass vor und nach diesem Rechteck-Bereich ein klar ersichtlicher Abstand entsteht.

Durch folgende CSS-Regel wird der Textabsatz zentriert ausgerichtet:

```
<style>
p{text-align:center;}
</style>
…
<p>Dies ist ein zentrierter Absatz.</p>
…
```

`right` bewirkt rechtsbündig, `left` bewirkt linksbündig, `justify` bewirkt Blocksatz.

1.8.3 Allgemeine Bereiche

Sie können mehrere Elemente wie Text, Grafiken, Tabellen usw. in einen gemeinsamen Bereich einschließen. `<div>` (*div = division = Bereich*) bewirkt, dass vor und nach diesem Rechteck-Bereich eine neuen Zeile beginnt. Soll kein Zeilenumbruch erfolgen, wird `<span>` verwendet.

`<div>` ist ein Block-Element. Ein Block-Element beginnt in einer neuen Zeile und nimmt die komplette Breite ein, sofern nichts anderes in CSS umdefiniert wurde. `<span>` ist ein Inline-Element. Ein Inline-Element beginnt in der aktuellen Zeile und hat nur jene Breite, die es tatsächlich benötigt.

Beide Elemente ermöglichen mit Hilfe von CSS eine Ausrichtung festzulegen (`left, center, right`) oder die Hintergrundfarbe zu setzen.

```
<div style="text-align:center;">zentrierter Text</div>
<span style="text-align:center;">zentrierter Text</span>
```

Das Beispiel zeigt einen zentrierten roten Block-Bereich.

```
<div style="position:absolute; top:10px; left:20px; width:100px;
            height:30px; background-color:red;">
Hallo liebe Leute ...
</div>
```

Dieses rechteckige Block-Element wird z.B. absolut zum Browserfenster positioniert. Der Abstand von oben beträgt 10 Pixel (Bildpunkte), von der linken Seite 20 Pixel. Das Rechteck hat eine Breite von 100 Pixel und eine Höhe von 30 Pixel. Die Hintergrundfarbe ist auf Rot gesetzt.

Wenn ein z.B. 200 Pixel breiten Block-Elementes zentriert werden soll:

```
<html><head><meta charset="UTF-8">
<style>
p.blocktext
{
   margin-left:auto;
   margin-right:auto;
   width:200px;
   background-color:red;
   text-align:center;
}
</style>
</head><body>
<p class="blocktext">
<a href="http://www.orf.at">ORF</a>
</p>
</body></html>
```

Auch der Inhalt des Blockelements ist durch `text-align:center;` zentriert.

1.8.4 Trennlinie

`<hr>` erzeugt eine waagrechte Linie *(hr = horizontal rule)*.

1.8.5 Präformatierter Text

Wenn du aus besonderen Gründen Text im Web-Browser so anzeigen willst wie du ihn eingeben hast (mit allen Einrückungen, Umbrüchen usw.), kannst du das HTML-Element für vorformatierten Text verwenden (`<pre>` und `</pre>`).

1.9 Farben

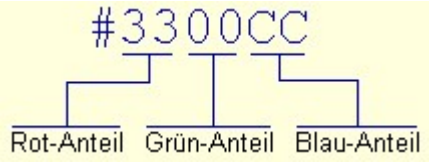
Grundsätzlich gibt es zwei Möglichkeiten, Farben in HTML zu definieren:

durch Angabe der RGB-Werte der gewünschten Farbe in Hexadezimalform
(RGB = Rot/Grün/Blau-Wert der Farbe)

durch Angabe eines Farbnamens. Derzeit sind jedoch nur 16 Farbnamen offiziell.

```
<style>
body {background-color:#FF0000;} <!-- roter Seitenhintergrund -->
table {background-color:#00FF00;} <!-- grüner Tabellenhintergrund -->
div {background-color:#0000FF;}  <!-- blauer Block-Bereich -->
</style>
```

Jede hexadezimale Farbdefinition ist 6-stellig. Ein Gatter (`#`, für die hexadezimale Schreibweise) und dahinter 6 Stellen definieren die Farbe. Die ersten beiden Stellen legen die Intensität des Rot-Anteiles fest. Dann folgt die Intensität der Farben Grün und Blau.



Hexadezimale Ziffern sind:

Hexadezimal Ziffer	Dezimalwert
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

Eine hexadezimale Ziffer kann also 16 Zustände haben. Für jeden Farbwert (Rot, Grün, Blau) stehen 2 Ziffern zur Verfügung. Das macht 16 x 16 (= 256) mögliche Zustände pro Farbwert.

Die definierten Farbnamen sind: (von Schwarz bis Weiss): black, gray, maroon, red, green, lime, olive, yellow, navy, blue, purple, fuchsia, teal, aqua, silver, white

1.9.1 Hintergrundfarbe und Textfarbe

```
<html><head><meta charset="UTF-8">
<style>
body{background-color:yellow;color:blue;}
</style>
</head><body>
Blauer Text und gelber Hintergrund
</body></html>
```

Die Angabe zur Hintergrundfarbe der HTML-Datei erfolgt mit einer CSS-Regel für das `<body>`-Tag der HTML-Datei. Mit dem Attribut `background-color` bestimmst du die Farbe für den Bildschirmhintergrund (*background-color* = *Hintergrund-Farbe*). Mit dem Attribut `color` bestimmst du die Farbe für die Textfarbe.

1.10 Bild

1.10.1 Im Vordergrund

```
<html><head><meta charset="UTF-8">
<style>
img{display:block;margin-left:auto;margin-right:auto;border:5px solid red;}
<body>
</style>
</head><body>
<img src="http://www.leiserberge.at/buschberg.jpg" alt="Berg im
Weinviertel" title="Berg im Weinviertel" width="387" height="195">
</body></html>
```

`<img ...>` (*img* = *image* = *Bild*) bindet ein Bild in die HTML-Seite ein. Durch das Attribut `src` wird die Quelle (=Ort) der Bilddatei festgelegt (URL oder relativer Dateipfad). Höhe und Breite des Bildes wird durch `height` und `width` (Einheit: Pixel oder Prozent) bestimmt. Das Attribut `alt` oder `title` bewirkt für alle Webbrowser, dass die festgelegte Information ausgegeben wird, sobald sich der Benutzer mit der Maus über dem Bild befindet. Das viel genutzte HTML-Attribut `align` sollte nicht mehr verwendet werden.

Auch andere Attribute sollten ab HTML5 durch CSS-Regeln festgelegt werden.

Durch die CSS-Attribute display:block;margin-left:auto;margin-right:auto; kann das Bild horizontal zentriert werden. Mit diesen Attributen kann auch ein beliebiger Textblock `<p>`,`<div>` oder `<span>` zentriert werden. Das CSS-Attribut `border:5px solid red;` legt einen roten Rand (Einheit: 5 Pixel) fest.

Der horizontale und vertikale Abstand eines Textes rund um ein Bild von z.B. 30 Pixel wird durch `margin:30px;` festgelegt. Der Abstand zum Bild kann aber auch durch die CSS-Attribute `margin-top`, `margin-right`, `margin-bottom` und `margin-left` detaillierter festlegt werden.

1.10.2 Im Hintergrund

```
Body { background-image:url("relative Dateipfad oder URL zum Bild"); }
```

Die Angabe des Hintergrundbildes der HTML-Datei erfolgt mit einer CSS-Regel für das `<body>`-Tag der HTML-Datei. Mit dem Attribut `background-image` bestimmst du das Hintergrundbild (*background-image = Hintergrund-Bild*). Als Wert gibst du für `url()` entweder eine relativen Dateipfad zur lokale Bilddatei oder einen gültigen URL zu einem Bild im Internet an. Das Bild wird dann allerdings so oft angezeigt, bis der Hintergrund vollständig von Bild ausgefüllt ist. Damit das Bild nicht widerholt angezeigt wird, kann zusätzlich das Attribut/Werte-Paar `background-repeat:no-repeat` verwendet werden.

Um ein zentriertes Hintergrundbild auf roten Hintergrund zu erhalten, kann folgende CSS-Regel verwendet werden:

```
body
{
   background-color:red;
   background-image:url("http://www.leiserberge.at/buschberg.jpg ");
   background-repeat:no-repeat;
   background-attachment:fixed;
   background-position:center;
}
```

Wenn das Hintergrundbild ohne Verzerrung möglichst den gesamten Fensterbereich ausfüllen und sich auch der Fenstergröße anpassen soll, kann folgende CSS-Regel genutzt werden:

```
body
{
   background-image:url("http://www.leiserberge.at/buschberg.jpg");
   background-repeat:no-repeat;
   background-attachment:fixed;
   background-position:center;
   background-size:cover;
}
```

1.11 Hyperlinks

Ein Web-Projekt (Homepage, Webshop, etc...) besteht typischerweise aus mehreren bis vielen Einzelseiten, die miteinander durch Hyperlinks (=Links, Verweise) miteinander verknüpft sind. Um einen Link auf eine andere Webseiten zu erstellen, muss eine exakte Angabe zum Verweisziel gemacht werden. Dieses Verweisziel ist eine beliebige Datei. Klickt ein Benutzer auf einen Link, so gelangt er zu diesem Verweisziel. Kann der Browser das Verweisziel nicht anzeigen, so bietet er einen Download dieser Datei an.

Die Definition eines Links erfolgt durch `<a href=""></a>` . Der Text zu Anklicken des Links ist nach dem einleitenden `<a>`-Tags festzulegen. Soll statt des Textes ein Bild angeklickt werden können, muss nach dem einleitenden `<a>`-Tag der HTML-Tag für ein Bild eingefügt sein. Durch das Attribut `href` wird das Verweisziel festgelegt.

1.11.1 Text als Hyperlink

```
<a href ="Verweisziel">Diese Text wird als Link angezeigt</a>
```

1.11.2 Bild als Hyperlink

```
<a href ="Verweisziel"><img src="Bildname.Extension"></a>
```

1.11.3 Dateiname als Verweisziel

Ist das Verweisziel im selben Ordner (=Verzeichnis) wie die Quelldatei (=Datei, die den Link enthält) zu finden, genügt es, beim Attribut `href` den vollständigen Namen der Datei inklusive Dateierweiterung (=Verweisziel) anzugeben.

```
<a href ="Dateiname.Extension">Diese Text wird als Link angezeigt</a>
```

1.11.4 Dateipfad als Verweisziel

Ist das Verweisziel im selben Dateibaum oder am selben Webserver wie die Quelldatei zu finden, muss beim Attribut `href` der korrekte relative Dateipfad zur Zieldatei (=Verweisziel) angegeben werden.

```
<a href ="../../ordner1/ordner2/Dateiname.Extension ">Text als Link</a>
```

Verwende bei Verweisen keinen absoluten Dateipfad, da du sonst den Verweis in einer neuen Umgebung anpassen musst !

1.11.5 URL als Verweisziel:

Ist das Verweisziel im Internet zu finden , muss beim Attribut `href` der korrekte URL (*Uniform Resource Locators* - einheitliche Quellenorter) angegeben werden. Im HTML-Standard wird ein URL auch als URI (*Uniform* Resource Identifier - universelle Quellenbezeichnung) bezeichnet.

```
<a href ="http://www.tgm.ac.at">TGM</a>
```

1.11.6 Anker als Verweisziel

Bei einem Link zu einer HTML-Seite wird im Browser standardmäßig der Beginn dieser Seite geöffnet. Bei großen (langen) HTML-Seiten besteht die Möglichkeit, gezielt an einen bestimmten Punkt (=an eine bestimmte Stelle) einer HTML-Seite zu gelangen. Dazu muss die genaue Stelle in der HTML-Seite für das Verweisziel festgelegt sein. Durch das Attribut `name` beim `<a>`-Tag oder durch das Attribut `id` bei beliebigen HTML-Tags wird eine genaue Stelle als eine Art „Ankerpunkt“ definiert.

```
<a name="oben">
<h1 id="unten">&Uuml;berschrift</h1>
```

Bei einem Link zu diesem Ankerpunkt muss zusätzlich zum Verweisziel das `#`-Zeichen angegeben werden.

```
<a href ="#oben">Link auf dieselbe Datei zum Anker mit dem Namen oben</a>
<a href ="Dateiname.Extension#uten">Link zum Anker mit dem Namen unten</a>
<a href ="../../ordner1/ordner2/Dateiname.Extension#oben">Link zu oben</a>
<a href ="http://www.tgm.ac.at#unten">Link zum Anker unten</a>
```

1.11.7 Farbe für Hyperlinks

```
<html><head><meta charset="UTF-8">
<style>
a:link{color:#FF0000;}
a:visited{color:#00FF00;}
a:hover{color:#FFFF00;}
a:active{color:#000000;}
</style>
</head><body>
<a href="http://www.tgm.ac.at">TGM</a>
</body></html>
```

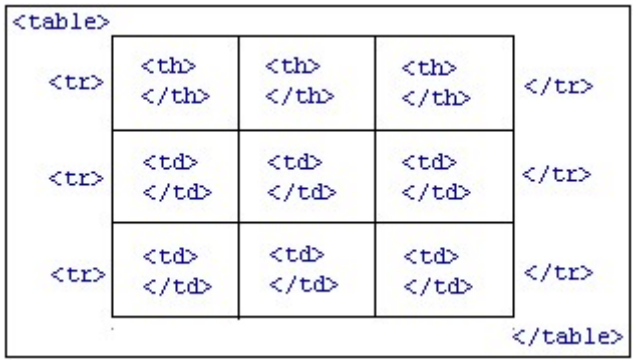
Mit `a:link{`color:#FF0000;`}`definierst du die rote Farbe für Verweise, die du noch nicht besucht (=angeklickt) hast (*link = Verweis*). Mit a:visited{color:#00FF00;}definierst du die grüne Farbe für Verweise, die du bereits besucht hast (*visited link = besuchter Verweis*). Mi a:hover{color:#FFFF00;} definierst du die gelbe Farbe für einen Verweis, wenn du dich mit der Maus gerade darüber befindest (*hover link = über dem Verweis*). Mit a:active{color:#000000;} definierst du die schwarze Farbe für Verweise, die du gerade anklickst (*activated link = aktivierter Verweis*).

Wenn du einen anderen Link in anderen Farben und ohne den Unterstrich haben möchtest, kannst du das zusätzlich definieren und das CSS-Attribut `text-decoration:none;` verwenden:

```
…
a.other:link{color:#00FFFF; text-decoration:none;}
a.other:visited{color:#FFFF00FF;text-decoration:none;}
a.other:hover{color:#FFFFFF;text-decoration:none;}
a.other:active{color:#000000;text-decoration:none;}
…
<a class="other" href="http://www.orf.at">ORF</a>
…
```

1.12 Tabellen

Du kannst in HTML Tabellen definieren, um tabellarische Daten darzustellen oder um Text und Grafik attraktiver am Bildschirm zu verteilen. In der heutigen Praxis des Web-Designs sind Tabellen als Grundgestaltungsmittel für Seitenlayouts nicht mehr wegzudenken.



`<table>` leitet eine Tabelle ein (*table = Tabelle*).

`<tr>` leitet eine neue Tabellenzeile ein (*tr = table row = Tabellenzeile*). Im Anschluss daran werden die Zellen (Spalten) der betreffenden Reihe definiert. Am Ende einer Tabellenzeile wird ein abschließendes Tag `</tr>` notiert.

Eine Tabelle kann Kopfzellen und gewöhnliche Datenzellen enthalten. Text in Kopfzellen wird hervorgehoben (meist fett und zentriert ausgerichtet). `<th>` leitet eine Kopfzelle ein, `<td>` eine normale Datenzelle (*th = table header = Tabellenkopf, td = table data = Tabellendaten*). Der Inhalt einer Zelle wird jeweils hinter dem Tag notiert. Die zugehörigen End-Tags sind `</th>` bzw. `</td>` . In einer Zelle können beliebige Zellenelemente stehen.

Wenn die Tabelle sichtbare Gitternetzlinien enthalten soll, musst du das CSS-Attribut `border` verwenden und ihm einen Wert größer `0` zuweisen. Auch die Angabe einer z.B. schwarzen Rahmenfarbe ist möglich.

```
table,tr,td{border:1px solid black;}
```

Der angegebene Wert ist dann die Breite des Rahmens in Pixel. Um eine Tabelle ohne sichtbaren Rahmen und Gitternetzlinien zu erzeugen, definiere folgende die CSS-Regel:

```
<style>
table {border:0px solid black;}
</style>
```

1.12.1 Tabellenüberschrift

```
..
table{caption-side:bottom;}
…
<table>
<caption>&Uuml;berschrift</caption>
…
```

Die Angabe einer Tabellenüberschrift erfolgt durch die `<caption>`-Tags.

Durch das CSS-Attribut `caption-side` kann die Überschrift über der Tabelle oben zentriert (`top`) oder unter der Tabelle zentriert (`bottom`) ausgerichtet werden.

1.12.2 Breite / Höhe / Rahmen

```
<table style="border:2px solid black;" height="1000">
<table style="border:2px solid black;" height="100%">
<table style="border:2px solid black;text-align:center" height="90%">
<tr height="70%">
```

Mit dem HTML-Attribut `height` kann die Höhe der Tabelle festgelegt werden. Die Einheit für die Höhe ist automatisch Pixel, wenn kein %-Zeichen angegeben wird. Einige voneinander unabhängige und unvollständige Beispiele:

```
<style>
table{border:1px solid red;width:1200px;height:800px;text-align:center;}
td{width:200px;}
</style>
```

Mit den CSS-Attributen `width` und `height` können die Breite und Höhe der Tabelle und mit `border` kann die Rahmenstärke mit Rahmenfarbe festgelegt werden.

1.12.3 Zentrierungen

Das nachfolgende Beispiel zeigt sowohl zentrierte Texte in den Tabellenzellen als auch eine zentrierte Tabelle und einen zentrierten Text in einem zentrierten Absatz mit gelbem Hintergrund:

```
<html><head><meta charset="UTF-8">
<style>
p.zentriert{
   margin-left:auto;
   margin-right:auto;
   background-color:yellow;
   width:600px;
   text-align:center;
}
table{
   margin-left:auto;
   margin-right:auto;
   border:2px solid black;
   width:200px;
   height:80px;
   text-align:center}

tr,td{border:2px solid red;}
td{width:200px;}
</style>
</head><body>
<p class="zentriert">
Der zentrierter Text<br><br>
<table><tr><td>Zelle 1</td><td>Zelle 2</td></tr></table>
</p>
</body></html>
```

1.12.4 Spaltenbreite vordefinieren

Die Darstellung einer Tabelle ergibt sich zwar automatisch aus den definierten Zeilen und Spalten. Doch für einen Web-Browser ist es nicht ganz einfach, die Darstellung frühzeitig zu ermitteln. Er muss erst die gesamte Tabelle einlesen, bevor er irgendetwas davon darstellen kann. Bei großen Tabellen kann das zu unschönen leeren Bildschirminhalten während des Seitenaufbaus führen.

Die HTML-Tags `<colgroup>` und `<col>` ermöglichen dem Browser gleich zu Beginn der Tabelle mitzuteilen, wie viele Spalten die Tabelle hat, und wie breit diese sind. Dadurch kann

der Browser die Tabelle schneller aufbauen, d.h. bereits Teile der Tabelle anzeigen, bevor die gesamte Tabelle eingelesen ist.

Beispielschema 1:

```
…
<style>
table,tr,td{border:2px solid black;}
</style>
…


<table>
    <colgroup>
       <col style="width:80;">
       <col style="width:150;">
       <col style="width:320;">
    </colgroup>
    <tr>
       <td>1.Zeile,1.Spalte</td>
       <td>1.Zeile,2.Spalte</td>
       <td>1.Zeile,3.Spalte</td>
    </tr>
</table>
```

Beispielschema 2:

```
…
<style>
table,tr,td{border:2px solid black;}
</style>
…

<table>
    <colgroup>
       <col span="3" style="width:200;">
    </colgroup>
    <tr>
       <td>1.Zeile,1.Spalte</td>
       <td>1.Zeile,2.Spalte</td>
       <td>1.Zeile,3.Spalte</td>
    </tr>
</table>
```

Mit `<colgroup>` leitest du hinter dem einleitenden `<table>`-Tag eine Vorab-Definition der Tabellenspalten ein (*colgroup* = *column group* = *Spaltengruppe*). Dabei hast du zwei Möglichkeiten: entweder du möchtest unterschiedlich breite Tabellenspalten haben. Dann gehe so vor wie im obigen „Beispielschema 1“. Oder du hast eine Tabelle, in der alle Spalten die gleiche einheitliche Breite haben sollen. Dann kannst du so vorgehen wie im obigen Beispielschema 2.

Im „Beispielschema 1“ enthält das `<colgroup>`-Tag keine weiteren Attribute. Dafür notierst du im Anschluss an `<colgroup>` für jede einzelne gewünschte Tabellenspalte je ein `<col>`-Tag. Das erste `<col>`-Tag definiert die erste Spalte, das zweite die zweite Spalte usw. Wenn du keine weiteren Angaben machst, wird die Breite der Spalten automatisch aufgrund des Tabelleninhalts ermittelt. Mit `width=` [Pixel/Prozent] kannst du jedoch eine Spaltenbreite für

die einzelnen Spalten vorgeben (*width* = *Breite*). Mit `width:100` erzwingst du beispielsweise eine Spaltenbreite von 100 Pixel, und mit `width:33%` eine Breite von einem Drittel der Breite der Gesamttabelle.

Im „Beispielschema 2“ enthält das `<colgroup>`-Tag keine weiteren Attribute. Statt mehrerer `<col>`-Tags wird nur eines verwendet, welches das Attribut `span=` (*span* = *spannen*) enthält. Als Wert weist du dem `span` die Anzahl der Spalten zu. Mit dem Attribut `width` kannst du in diesem Fall eine einheitliche Spaltenbreite für alle Spalten definieren.

Bei `width` werden zumeist Pixel oder Prozentwerte anzugeben.

1.12.5 Zelle spaltenweise verbinden

Eine Tabellenzelle kann sich mit Hilfe des Attributs `colspan` (=*columns span* = Spalten verbinden) über mehrere Tabellenspalten erstrecken.

Beispiel 1:

```
…
<style>
table,tr,td{border:1px solid red;}
</style>
…

<table>
<tr><td>Spalte 1</td><td>Spalte 2</td><td>Spalte 3</td><td>Spalte
4</td></tr>
<tr><td colspan="4">Zelle erstreckt sich über 4 Spalten</td></tr>
</table>
```

Spalte 1	Spalte 2	Spalte 3	Spalte 4
Zelle erstreckt sich über 4 Spalten			

Beispiel 2:

```
…
<style>
table,tr,td{border:1px solid red;}
</style>
…

<table>
<tr><td>Spalte 1</td><td>Spalte 2</td><td>Spalte 3</td><td>Spalte
4</td></tr>
<tr><td>Zelle</td><td colspan="3">Zelle über 3 Spalten</td></tr>
</table>
```

Spalte 1	Spalte 2	Spalte 3	Spalte 4
Zelle	Zelle über 3 Spalten		

1.12.6 Zelle zeilenweise verbinden

Eine Tabellenzelle kann sich mit Hilfe des Attributs `rowspan` (=*rows span* = *Zeilen verbinden*) über mehrere Tabellenzeilen erstrecken.

Beispiel 1:

```
…
<style>
table,tr,td{border:1px solid red;}
</style>
…
<table>
<tr><td>Zeile 1</td><td>Zeile 1</td></tr>
<tr><td>Zeile 2</td><td rowspan="3">2 bis 4</td></tr>
<tr><td>Zeile 3</td></tr>
<tr><td>Zeile 4</td></tr>
</table>
…
```

Beispiel 2:

```
…
<style>
table,tr,td{border:1px solid red;}
</style>
…
<table>
<tr><td>Zeile 1</td><td rowspan="2">1+2</td></tr>
<tr><td>Zeile 2</td></tr>
<tr><td>Zeile 3</td><td rowspan="2">3+4</td></tr>
<tr><td>Zeile 4</td></tr>
</table>
```

1.12.7 Kopf-, Rumpf- und Fußaufteilung

Eine Tabelle kann logisch in Bereiche aufgeteilt werden: einen Kopfbereich `<thead>`, einen oder mehrere Datenbereiche `<tbody>` und einen Fußbereich `</tfoot>`. Beim Ausdruck langer Tabellen sollte der Browser Tabellenkopf und Tabellenfuß auf jeder Seite wiederholen.

```
<table style="border:2;width:200";>
<thead style="background-color:red;
              text-align:center;">
<tr><td>zentrierter Tabellenkopf</td></tr>
</thead>
<tbody style="background-color:yellow;
              text-align:right;">
<tr><td>rechtsb&uuml;ndiger Rumpf</td></tr>
<tr><td>mit mehreren</td></tr>
<tr><td>Zeilen</td></tr>
</tbody>
<tfoot style="background-color:green;
              text-align:left;">
<tr><td>linksb&uuml;ndiger
Tabellenfu&szlig;</td></tr>
</tfoot></table>
```

1.13 Listen

1.13.1 Geordnete Liste

`<ol>` leitet eine nummerierte Liste ein (*ol = ordered list = nummerierte Liste*). Mit `<li>` beginnt ein neuer Punkt innerhalb der Liste (*li = list item = Listeneintrag*). `</li>` beendet den Listeneintrag.`</ol>` beendet die Liste.

```
<ol type="1">
<li> Banane </li>
<li> Apfel </li>
<li> Orange </li>
<li> Kiwi </li>
</ol>
```

Die Liste von Elementen `<li>`wird in mit einer Zahlen-Nummerierung dargestellt. Das Attribut `type="1"` ist optional.

`<ol start="17">` Das Attribut `start` legt den Startwert der Liste fest.

```
<ol type="I">
  <li>Anfang</li>
  <li>Hauptteil</li>
  <li>Schluss</li>
</ol>
```

Die sortierte Liste wird durch das Attribut `type="I"` oder `type="i"` mit großen oder kleinen römischen Ziffern dargestellt. Soll nach dem Alphabet geordnet werden, so muss `type="A"` oder `type="a"` verwendet sein.

Anstelle des HTML-Attribut `type` sollte jedoch besser das CSS-Attribut `list-style-type` verwendet werden

`decimal` (dezimale Nummerierung, z.B. 1., 2., 3., 4.,…)
`decimal-leading-zero` (dezimale Nummerierung mit Null, z.B. 01., 02., 03., 04.,…)
`lower-roman` (kleine römischen Zahlen)
`upper-roman` (große römische Zahlen)

`lower-alpha` (kleines Alphabetzeichen)
`lower-latin` (kleines Alphabetzeichen)
`upper-alpha` (großes Alphabetzeichen)
`upper-latin` (kleines Alphabetzeichen)
`none` (keine Nummerierung)

1.13.2 Ungeordnete Liste

Das Element `<ul>` (*ul steht dabei für unordered list, ungeordnete, unsortierte Liste*) beschreibt eine Aufzählungsliste, also eine Liste, bei der die Reihenfolge der Elemente nur eine untergeordnete oder keine Rolle spielt

```
<ul>
<li>Probieren geht über Studieren</li>
<li>Liebe geht über Triebe</li>
<li>Tante fällt über Kante</li>
</ul>
```

Wie das Aufzählungszeichen dargestellt wird, bestimmt dabei der Web-Browser. Dessen Darstellung kann jedoch mit dem CSS-Attribut `list-style-type` beeinflusst werden:

`disc` (gefüllter Kreisfläche)
`circle` (Kreisring)
`square` (gefülltes Quadrat)
`none` (kein Zeichen):

```
<style>
ul.a{list-style-type:disc;}
ul.b{list-style-type:circle;}
ul.c{list-style-type:square;}
</style>
```

1.13.3 Beschreibungsliste

Beschreibungslisten (Definitionslisten) werden zumeist für Abkürzungen verwendet:

```
<dl>
<dt>allg.</dt><dd>allgemein</dd>
<dt>bez.</dt><dd>bezüglich</dd><dd>bezahlt</dd>
<dt>u. a.</dt> <dd>unter anderem, unter anderen</dd>
               <dd>und andere, und anderes</dd>
<dt>zzgl.</dt><dd>zuzüglich</dd>
</dl>
```

Eine Definitions- bzw. Beschreibungsliste wird mit `<dl>` eingeleitet und mit `</dl>` beendet. Der zu erläuternde Ausdruck steht zwischen `<dt>…</dt>` (*description [list] term*). `<dd>…</dd>` (*description [list] description*) umschließt die Erläuterung.

1.14 Framesets

Framesets werden ab HTML5 nicht mehr unterstütz. Die wichtigsten Argumente gegen Framesets sind:

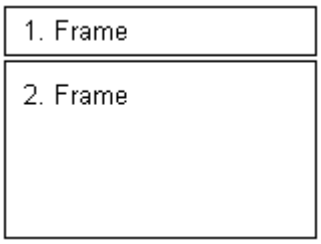
- Suchmaschinen sehen, verlinken und bewerten die falschen Inhalte
- Darstellung bei unterschiedlichen Auflösungen und Endgeräten sehr problematisch
- Einbindung externer Seiten in Framesets rechtlich umstritten
- keine Barrierefreiheit
- keine Gestaltungsfreiheit

Da aber Framesets in vielen HTML-Werkzeugen und HTML-Toolkits noch länger im Einsatz sind, werden diese nachfolgend trotzdem besprochen.

Mit Hilfe von Frames kann der Anzeigebereich des Browsers in verschiedene Teile (Segmente) zerlegt werden. Diese Aufteilung erfolgt durch Framesets `<frameset>` und `</frameset>` in einer eigenen HTML-Datei. Jedes Segment `<frame>` erhält üblicherweise den Inhalt aus einer eigenen HTML-Datei.

1.14.1 Zeilenaufteilung

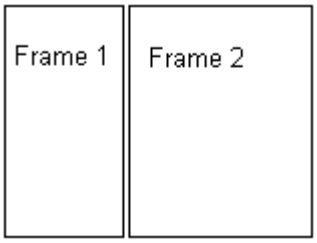
```
<frameset rows="20%,80%">
   <frame src="inhalt1.html">
   <frame src="inhalt2.html">
</frameset>
```



Im obigen Beispiel wird im einleitenden Tag `<frameset>` die zeilenartige Aufteilung festgelegt. Durch das Attribut `rows` (*rows* = *Reihen*) wird das Anzeigefenster in Reihen aufgeteilt. Dahinter wird festgelegt, wie die Aufteilung aussehen soll. Die Aufteilung in zwei Reihen wird mit Hilfe von `rows="20%,80%"` erzwungen, wobei die obere Reihe 20 % des Anzeigefensters in Anspruch nimmt, die untere 80 %. Wird bei den Zahlenwerten kein %-Zeichen angegeben, so ist Pixel die Einheit für die Aufteilung. Durch den Tag `<frame src="inhalt1.html">` wird das Segment durch den Inhalt aus der Datei mit dem Namen `inhalt1.html` gefüllt. Alternativ kann anstelle eines vollständigen Dateinamens ein relativer Pfad oder ein URL (URI) angegeben werden.

1.14.2 Spaltenaufteilung

```
<frameset cols="200,*">
   <frame src="inhalt1.html">
   <frame src="inhalt2.html">
</frameset>
```



Im obigen Beispiel wird im einleitenden Tag `<frameset>` die spaltenartige Aufteilung

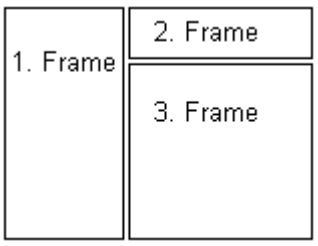
festgelegt. Durch das Attribut `cols` (*columns* = *Spalten*) wird das Anzeigefenster in Spalten aufgeteilt. Dahinter wird festgelegt, wie die Aufteilung aussehen soll.

Die Aufteilung in zwei Spalten wird mit Hilfe von `cols="200,*"` erzwungen, wobei die linke Spalte 200 Pixel des Anzeigefensters in Anspruch nimmt, die rechte Spalte nützt den restlichen Bereich des Browserfensters. Wird bei den Zahlenwerten kein %-Zeichen angegeben, so ist Pixel die Einheit für die Aufteilung. Anstelle des *-Zeichens kann auch ein korrekter Zahlenwert verwendet werden.

Durch den Tag `<frame src="inhalt1.html">` wird das Segment durch den Inhalt aus der Datei mit dem Namen `inhalt1.html` gefüllt. Alternativ kann anstelle eines vollständigen Dateinamens ein relativer Pfad oder ein URL (URI) angegeben werden.

1.14.3 Verschachtelung

```
<frameset cols="40%,60%" border="3"
          bordercolor="red" >
  <frame name="segment1" src="inhalt1.html">
  <frameset rows="20%,80%">
    <frame name="segment2" src="inhalt2.html">
    <frame name="segment3" src="inhalt3.html">
  </frameset>
</frameset>
```



Im obigen Beispiel erfolgt zuerst eine spaltenartige Aufteilung in zwei Segmente im Verhältnis 40 zu 60. Jedes Segment bekommt durch das Attribut name einen eigenen Namen, der sehr oft bei Links über das Attribut target genützt wird. Die linke Spalte wird nicht weiter zerlegt und erhält den Inhalt von der Datei inhalt1.html. Die rechte Spalte wird zeilenartig in zwei Segmente im Verhältnis 20 zu 80 zerlegt und bekommen die Inhalte aus den Dateien inhalt2.html und inhalt3.html. Das nicht genormte Attribut border legt eine gewünschte Umrahmung der Segmente fest. Die Farbe dieses Rahmens kann mit dem nicht genormten Atribut bordercolor festgelegt werden.

1.14.4 Framesets – Diverses

Der Titel `<title>...</title>`, der in der Datei mit der Frameset-Definition angegeben ist, wird während der gesamten Dauer des Frame-Sets angezeigt. Wähle in der Datei, die die Frame-Set-Definitionen enthält, deshalb einen allgemeinen, `aussagekräftigen` Titel, der für das gesamte Projekt Gültigkeit hat. Frame-Sets sollten immer so definiert sein, dass das gesamte Anzeigefenster abgedeckt wird. Verwende dazu Prozentangaben, die in der Summe 100 ergeben, oder das Sternzeichen `*`. Bei Pixelangaben muss der Anwender eventuell scrollen.