Gymnasium Kirchenfeld

Grundlegende Web-Technologien

Stefan Rothe

2. Februar 2011

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Grundlagen

1.1 Webtechnologien

1.1.1 Einleitung

Nebst der E-Mail ist das *World Wide Web* die verbreitetste Anwendung des Internets. Als World Wide Web bezeichnet man die weltweit verknüpften *Webseiten*, welche man zu Gesicht bekommt, wenn man im "Internet surft". In diesem Skript erfahren Sie, wie eine Webseite technisch aufgebaut ist. Sie lernen die grundlegenden Technologien kennen, um den Aufbau einer Webseite zu verstehen und selbst eine Webseite erstellen zu können.

Technisch gesehen ist eine Webseite aus einer Vielzahl von Dateien wie Bildern, Dokumenten oder Programmen aufgebaut. Die wichtigste Datei, welche bei keiner Webseite fehlen darf, ist eine HTML[1]-Datei. Sie enthält den *Bauplan* der Webseite, der beschreibt, welche anderen Dateien zu der Webseite gehören und auf welche Weise sie zusammengesetzt werden sollen (siehe Abbildung 1.1). Alle diese Dateien sind meist nicht auf dem eigenen Computer vorhanden.

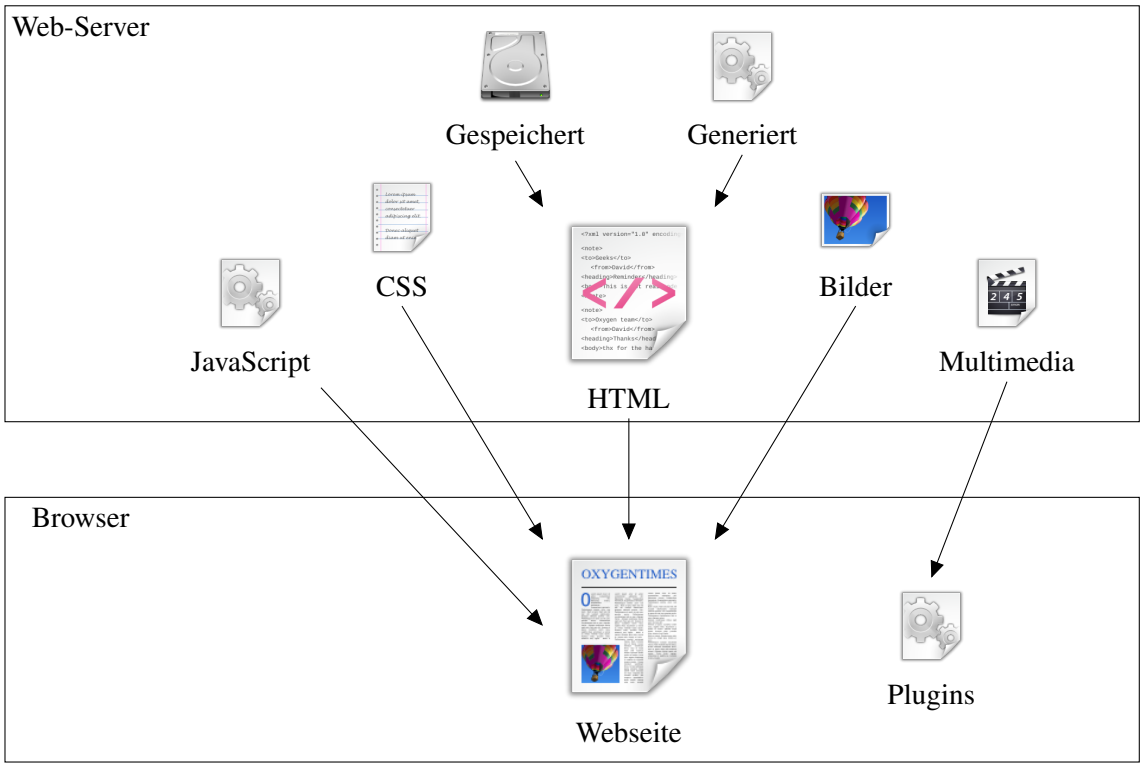

Abbildung 1.1: Aufbau einer Webseite

[1] HTML: Hypertext Markup Language

1.1.2 Webbrowser

Ein Webbrowser ist ein Programm, welches sich auf dem eigenen Computer befindet und für das Anzeigen einer Webseite verantwortlich ist. Dazu muss der Webbrowser die Dateien, welche zu einer Webseite gehören, "herholen". Anschliessend analysiert er die Dateien und baut die Darstellung der Webseite entsprechend auf. Der Browser führt auch Programme, welche in der Webseite enthalten sind, aus.

1.1.3 Webserver

Die Dateien, aus welchen eine Webseite zusammengesetzt ist, werden von einem *Webserver* zu Verfügung gestellt. Dabei wird zwischen zwei Arten von Webseiten unterschieden:

Bei *statischen* Webseiten sind die Dateien auf dem Webserver gespeichert. Dies ist die einfachere Variante und macht Sinn für Webseiten, welche immer gleich aussehen. Für sich ändernde oder personalisierte Webseiten ist dieser Ansatz jedoch nicht geeignet.

Solche *dynamischen* Webseiten werden deshalb bei jeder Anfrage neu erstellt. Dazu sind auf dem Webserver Programme hinterlegt, welche den Aufbau der Webseite steuern. Diese Programme können in unterschiedlichen Programmiersprachen geschrieben werden. Sie kann meist an der Endung der Adresse der Webseite erkannt werden. Die folgende Tabelle zeigt eine Übersicht:

Endung	**Technologie**
`html`, `htm`	HTML-Datei, statisch
`cgi`	allgemeines Programm (CGI: Common Gateway Interface), dynamisch
`jsp`	Java-Programm (JSP: Java Server Pages), dynamisch
`php`	PHP-Programm (PHP Hypertext Preprocessor), dynamisch
`asp`, `aspx`	ASP-Programm (ASP: Active Server Pages), dynamisch

1.1.4 Hypertext Markup Language (HTML)

Die *Hypertext Markup Language* ist die zentrale Technologie hinter Webseiten. Mit HTML wird die Struktur einer Webseite festgelegt. Es wird definiert, welche Dateien zur Webseite gehören. Ausserdem enthält die HTML-Datei den Text einer Webseite.

1.1.5 Cascading Stylesheets (CSS)

Stylesheets sind eine Ergänzung zu HTML. Mit ihnen kann das Aussehen von Webseiten festgelegt werden. Mit CSS kann genau definiert werden, wie einzelne Elemente der Webseite aussehen sollen. So können Eigenschaften wie Schriftart, Grösse, Farbe, Abstände, Rahmen und Hintergrundbilder festgelegt werden.

Stylesheets sind so konzipiert, dass mit einer CSS-Datei das Aussehen mehrere Webseiten definiert werden kann. Umgekehrt können für eine Webseite unterschiedliche Darstellungen festgelegt werden (zum Beispiel für die Darstellung am PC, auf dem Mobiltelefon oder für den Ausdruck).

1.1.6 JavaScript

JavaScript ist eine Programmiersprache welche speziell für den Einsatz in Webseiten entwickelt worden ist. Die Sprache hat aber nicht viel mit Java gemeinsam, sie wurde aus Marketinggründen so genannt. Im Ge-

gensatz zu serverseitigen Technologien wie PHP wird die *JavaScript* im Webbrowser ausgeführt. So können Webseiten verändert werden, ohne dass die Dateien neu vom Webserver angefordert werden müssen.

Mit JavaScript können Webseiten auch im Hintergrund mit dem Server Daten austauschen und so Teile der Webseite nachladen und aktualisieren. In diesem Fall spricht man auch von AJAX (Asynchronous JavaScript and XML). Moderne Webanwendungen machen intensiven Gebrauch von dieser Technik.

JavaScript stellt aber auch eine grosse Angriffsfläche dar. Für Angriffe aus dem Internet ist diese Möglichkeit, Programmcode auf dem anzugreifenden Computer ausführen zu lassen, natürlich sehr verlockend. Es ist daher zu empfehlen, JavaScript nach Möglichkeit zu deaktivieren.

1.1.7 Plugins

Das die Möglichkeiten der bisher genannten Technologien nicht als genügend erachtet worden sind, haben verschiedene Softwarehersteller haben eigene Browser-Erweiterungen, sogenannte *Plugins*, definiert und entwickelt. Heute werden diese vor allem für das Abspielen von Videos oder für browserbasierte Anwendungen und Spiele verwendet. Folgend eine Aufzählung einiger verbreiteter Plugins:

- ActiveX (von Microsoft)
- Flash (von Adobe)
- Java (von Oracle)
- Quicktime (von Apple)
- Silverlight (von Microsoft)

1.2 Uniform Resource Locators (URLs)

Die die Dateien einer Webseite befinden sich wie schon erwähnt meist auf einem anderen Computer. Wir müssen dem Webbrowser mitteilen können, wie er diese Dateien finden kann. Ein *Uniform Resource Locator* (URL) ist eine globale, weltweit eindeutige Adresse für eine Datei. Die URL definiert, *wie* mit dem anderen Computer kommuniziert wird, *wo* dieser Computer zu finden ist, und *welche* Datei auf dem Computer gemeint ist. Die URL setzt sich entsprechend aus drei Teilen zusammen:

- *Protokoll*: normalerweise `http://` oder `https://`, die Übertragungsart der Datei.
- *Computername, Domäne*: Weltweit eindeutiger Name des Computers, auf welchem die Datei gespeichert ist (z.B. `www.selfhtml.de`)
- *Dateipfad*: Eindeutige Bezeichnung einer Datei (z.B. `/navigation/css.html`)

1.3 Hilfsmittel

Folgende Hilfsmittel sind grundlegend, um mit HTML zu arbeiten:

- Ein Texteditor mit HTML-Unterstützung zur Bearbeitung des HTML-Quelltextes, zum Beispiel Notepad++ oder NetBeans
- Ein Browser zum Betrachten der Webseite, zum Beispiel Mozilla Firefox.
- Die SELFHTML-Referenz (`http://www.selfhtml.de`)

- Mit dem W3C-Validator (`http://validator.w3.org`) können HTML-Seiten auf ihre Korrektheit überprüft werden.

Weiter werden folgende Hilfsmittel empfohlen:

- Online-Tutorials (z.B. w3schools.com)
- Web Developer Plugin für Mozilla Firefox

2 Hypertext Markup Language (HTML)

2.1 Auszeichnungssprache

Haben Sie schon einmal **lach** oder etwas Ähnliches geschrieben? Dann haben Sie eine Auszeichnungssprache benutzt. Die Idee dahinter ist, dass bestimmte Teile eines Textes nicht den eigentlichen Inhalt darstellen, sondern definieren, wie der Inhalt zu verstehen ist. Im vorangehenden Beispiel bedeuten die Asterisken, dass mit *lach* eine Tätigkeit gemeint ist. Auszeichnungssprachen dienten ursprünglich als Anweisungen für die Setzer im Buchdruck. Auch HTML ist eine Auszeichnungssprache.

2.2 Aufbau von HTML

2.2.1 Erster Kontakt

HTML-Quelltext besteht aus strukturierten *Elementen*, welche durch sogenannte *Tags* begrenzt werden. HTML-Tags sind immer durch spitze Klammern (< und >) gekennzeichnet. Hier sehen Sie ein erstes Beispiel eines HTML-Quelltextes:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Meine erste Webseite</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  </head>
  <body>
    <p>Willkommen auf meiner ersten Webseite!</p>
    <p>Gefällt sie Ihnen?</p>
  </body>
</html>
```

Mit Hilfe der Tags wird also im HTML-Dokument eine *verschachtelte Struktur* von Elementen definiert. Für das obenstehende Beispiel sieht diese Struktur folgendermassen aus:

2.2.2 Elemente

Die meisten Elemente haben einen Inhalt, welche durch die entsprechenden *Anfangstags* und *Endtags* begrenzt werden. Dabei haben die beiden Tags den gleichen Namen (das ist das erste Wort innerhalb der spitzen Klammern). Beim Endtag wird dem Namen einen Schrägstrich vorangestellt.

Beispiel: Dieses Beispiel zeigt ein `<p>`-Element mit dem Inhalt "Das ist ein Beispiel.". Das `<p>`-Element kennzeichnet einen Paragraphen:

```
<p>Das ist ein Beispiel.</p>
```

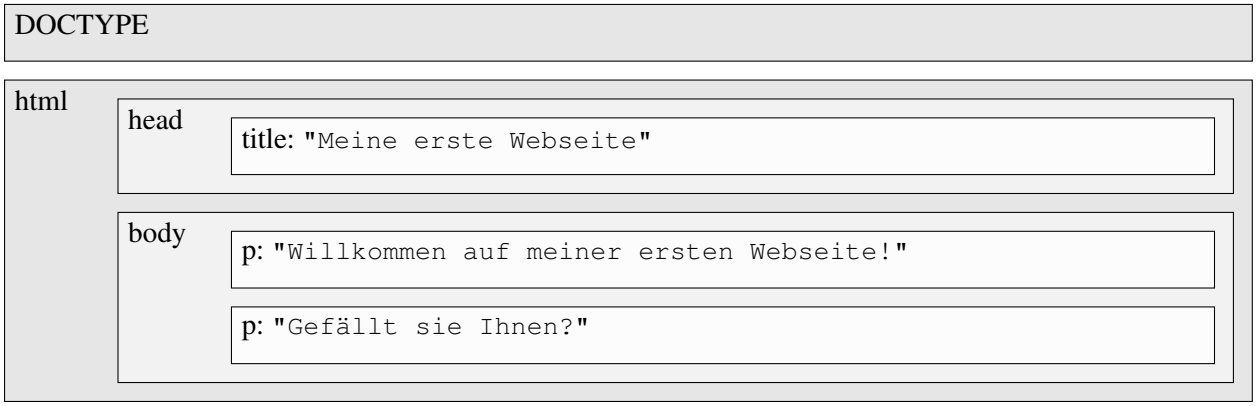

Abbildung 2.1: Struktur eines HTML-Dokuments

Elemente können ineinander verschachtelt sein. Das heisst, dass im Inhalt eines Elementes wieder Elemente vorkommen können. Hier ist beispielsweise das `<p>`-Element aus dem vorangehenden Beispiel nun selbst Inhalt eines `<body>`-Elementes:

```
<body><p>Das ist ein Beispiel.</p></body>
```

Elemente dürfen sich aber nicht überkreuzen, Anfangs- und Endtags müssen immer in umgekehrter Reihenfolge vorkommen.

```
<body><p>ACHTUNG! Das ist kein gültiger HTML-Code.</body></p>
```

Einige Elemente haben keinen Inhalt und besitzen somit kein Endtag. Beispiel: Das `<br>`-Element stellt einen Zeilenumbruch dar, es hat keinen Inhalt:

```
Zeile eins.<br>Zeile zwei.
```

2.2.3 Block- und Inline-Elemente

In HTML findet die Formatierung eines Dokuments auf zwei Ebenen statt:

- Mit sogenannten Inline-Elemente kann das Aussehen einzelner Zeichen gesteuert werden. Solche Elemente unterbrechen den Textfluss nicht.
- Durch Blockelemente werden Absätze mit einem bestimmten Layout definiert.

Daraus folgt, dass Inline-Elemente mehrfach geschachtelt werden dürfen. Innerhalb von Blockelementen dürfen aber nur Inline-Elemente verwendet werden.

Beispiel: Das Blockelement `<p>` definiert einen normalen Absatz, das Inline-Element `<i>` erzwingt eine kursive Schrift.

```
<p>Das ist ein <i>tolles</i> Beispiel.</p>
```

2.2.4 Kommentare

Kommentare sind Abschnitte im Dokument, welche Hinweise für den Autor enthalten, aber nicht dargestellt werden sollen. Ein Kommentar beginnt mit der Zeichenfolge `<!-` und endet mit `-->`. Der Kommentar selbst darf weder `>` noch `-` enthalten. Das ist ein gültiger Kommentar:

```
<!-- Dies ist ein Kommentar. -->
```

Dies ist kein gültiger Kommentar:

```
<!-- '>' ist nicht erlaubt, '--' auch nicht. -->
```

2.2.5 Attribute

Manchmal ist es wünschenswert oder nötig, zu HTML-Elementen zusätzliche Angaben zu machen. Zum Beispiel muss für einen Link die Zieladresse angegeben werden. Solche Informationen werden als sogenannte Attribute in das Anfangstag des Elements geschrieben. Dabei wird jedes Attribut in der Form `name="wert"` geschrieben. Ein Anfangstag mit zwei Attributen sieht also so aus:

```
<elementname attr1="wert1" attr2="wert2">
```

Beispiel: Das `<img>`-Element bindet ein Bild in die Webseite ein. Mit dem Attribut `src` wird der Speicherort des Bildes angegeben:

```
<img src="images/example.png">
```

2.3 Grundgerüst einer HTML-Datei

Jede HTML-Datei besteht aus folgenden drei Teilen:

- *Beschreibung des Dokumenttyps*: Hier wird die verwendete HTML-Version angegeben, damit der Browser das Dokument korrekt darstellen kann.
- *Kopfdaten*: Dieser Bereich enthält Meta-Informationen, also Angaben *über* das Dokument wie dessen Titel, den Autor oder Stichwörter für Suchmaschinen. Dieser Bereich wird mit dem `<head>`-Tag umschlossen.
- *Körper*: In diesem Bereich steht der eigentliche Inhalt des Dokuments. Er wird mit dem `<body>`-Tag umschlossen.

Die Kopfdaten und der Körper des Dokuments werden zusätzlich mit dem `<html>`-Tag zusammengefasst. Dies führt zu folgender Grundstruktur:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Titel der Webseite</title>
     <!-- Hier stehen Informationen ueber das Dokument. -->
  </head>
  <body>
     <!-- Hier steht der eigentliche Inhalt. -->
  </body>
</html>
```

2.4 Grundlegende HTML-Elemente

2.4.1 Zeichenformatierung

Die Darstellung einzelner Zeichen kann durch Inline-Elemente gesteuert werden. In HTML gibt es zwei Möglichkeiten der Zeichenformatierung:

- Durch ein logisches Format wird die Art des Inhalts angegeben (z.B. ein Zitat, eine Hervorhebung, ...). Die Darstellung wird dem Browser überlassen.
- Durch ein physisches Format wird direkt eine Angabe zur gewünschten Darstellung gemacht (z.B. fett, kursiv, ...).

Die folgenden logischen Formate stehen zu Verfügung:

Element	**Bedeutung**
`<em> ... </em>`	betont, wichtig (*emphasis*)
`<strong> ... </strong>`	stark betont, sehr wichtig
`<code> ... </code>`	Quelltext (*source code*)
`<samp> ... </samp>`	ein Beispiel (*sample*)
`<kbd> ... </kbd>`	eine Benutzereingabe (*keyboard*)
`<var> ... </var>`	eine Variable
`<cite> ... </cite>`	Quellenangabe eines Zitats

Die folgenden physischen Zeichenformate stehen zu Verfügung:

Element	**Bedeutung**
`<b> ... </b>`	fett (bold)
`<i> ... </i>`	kursiv (italic)
`<big> ... </big>`	grösser als normal
`<small> ... </small>`	kleiner als normal
`<sup> ... </sup>`	hochgestellt (superscript)
`<sub> ... </sup>`	tiefgestellt (subscript)

2.4.2 Textgliederung

Die Gliederung des Textes wird durch verschiedene Blockelemente erreicht.

Für normale Textabsätze wird das `<p>`-Element (von *paragraph*) verwendet.

Für Zitate steht das `<blockquote>`-Element zu Verfügung. Die Quellenangabe des Zitats kann mit dem `<cite>`-Element (siehe oben) eingefügt werden.

Überschriften (*headings*) werden mit den Elementen `<h1>`, `<h2>` bis `<h6>` dargestellt, wobei `<h1>` die grösste Überschrift darstellt.

Listen werden mit dem `<ul>`-Element (von *unordered list*, ohne Nummerierung) oder dem `<ol>`-Element (von *odered list*, mit Nummerierung) erzeugt. Die einzelnen Listeneinträge werden als `<li>`-Elementen (von *list item*) innerhalb des Listen-Elements definiert:

```
<ul>
  <li>Erstens</li>
  <li>Zweitens</li>
```

```
  <li>Drittens</li>
</ul>
```

2.4.3 Verweise (Links)

Verweise werden mit Hilfe des `<a>`-Elementes (von *anchor*) erstellt. In diesem Element stehen die folgenden Attribute zu Verfügung:

- href definiert die Zieladresse des Verweises
- target definiert das Browserfenster die referenzierte Webseite dargestellt werden soll.

Wird das Attribut target mit mit dem Wert `"_blank"` angegeben, so wird die referenzierte Webseite in einem neuen Browserfenster geöffnet.

Beispiel: Beim Klicken auf diesen Link wird die Einstiegsseite von SELFHTML in einem neuen Browserfenster geöffnet:

```
<a href="http://www.selfhtml.de" target="_blank">SELFHTML-Homepage</a>
```

Die Zieladresse des Verweises muss eine sogenannte URL (*uniform resource locator*) sein. Eine URL beschreibt eindeutig den Ort einer Ressource (also z.B. einer HTML-Datei) im Web.

```
http://www.selfhtml.ch/navigation/css.html
```

Eine URL setzt sich aus verschiedenen Teilen zusammen:

- *Protokoll*: normalerweise `http://` oder `https://`, die Übermittlungsart der Webseite.
- *Computername*: Weltweit eindeutiger Name des Computers, auf welchem die Webseite gespeichert ist (z.B. `www.selfhtml.de`)
- *Dateipfad*: Eindeutige Bezeichnung einer Datei (z.B. `/navigation/css.html`)

Wenn eine URL auf eine andere HTML-Datei auf dem gleichen Webserver verweisen soll, so werden das Protokoll und der Computername weggelassen. Es kann einfach eine absolute oder relative Pfadangabe verwendet werden:

```
../sample/index.html
```

2.4.4 Bilder

Bilder können mit dem Inline-Element `<img>` eingebunden werden. Im Attribut `src` wird die Adresse (URL) des Bildes angegeben. Das `alt`-Attribut definiert einen Alternativtext für eine andere Darstellung (z.B. für die Sprachausgabe oder als Braille für Sehbehinderte). Mit den Attributen `width` und `height` können die Breite und Höhe des Bildes in Pixel definiert werden.

```
<img src="sample.jpg" alt="Ein Beispielbild" witdh="20" height="20">
```

2.4.5 Tabellen

Für die Definition von Tabellen sind gleich drei verschiedene, ineinander verschachtelte Elemente nötig:

- Die gesamte Tabelle wird von einem `<table>`-Element umschlossen.

- Jede Zeile wird durch ein `<tr>`-Element (von *table row*) definiert.
- Für jede Zelle innerhalb einer Zeile wird ein `<td>`-Element (von *table data*) benötigt.

A1	B1
A2	B2

Beispiel: Die obenstehende Tabelle kann in HTML so dargestellt werden:

```
<table>
  <tr><td>A1</td><td>B1</td></tr>
  <tr><td>A2</td><td>B2</td></tr>
</table>
```

2.4.6 Zeichendarstellung

Habe Sie sich überlegt, wie Sie das Zeichen < in HTML darstellen können? Direkt hinschreiben geht nicht, dann wird es als Anfang eines HTML-Elements interpretiert. Deshalb gibt es in HTML für Zeichen mit einer besonderen Bedeutung eine alternative Darstellung, welche Sie der nachfolgenden Tabelle entnehmen können:

Zeichen	**Codierung**	**Bedeutung**
¨	`&quot;`	*quote*
&	`&`	*ampersand*
<	`<`	*less than*
>	`>`	*greater than*
	` `	*non-breaking space*

Beachten Sie, dass eine solche Zeichenumschreibung immer mit einem Et-Zeichen beginnt und mit einem Semikolon endet. Die Umschreibung ` ` stellt ein spezielles Leerzeichen dar, bei welchem kein Zeilenumbruch erlaubt ist. Besonders vor oder nach Zahlangaben in einem Text kann dies erwünscht sein. Beim folgenden Beispiel wird sichergestellt dass die Satzteile “Kirchenfeldstrasse 25” und “42 km” nicht getrennt werden:

```
Die Adresse Lautet Kirchenfeldstrasse 25.
Die Strecke ist 42 km lang.
```

3 Cascading Stylesheets (CSS)

3.1 Trennung von Inhalt und Layout

Mit HTML und CSS wird die grundlegende Idee der Trennung von Inhalt und Layout umgesetzt. Anstatt dem Inhalt direkt eine Formatierung zuzuweisen, wird eine Struktur definiert. Den einzelnen Strukturelementen kann dann die Formatierung zugewiesen werden. So können sowohl Inhalt als auch Layout auf einfache Weise ausgetauscht werden.

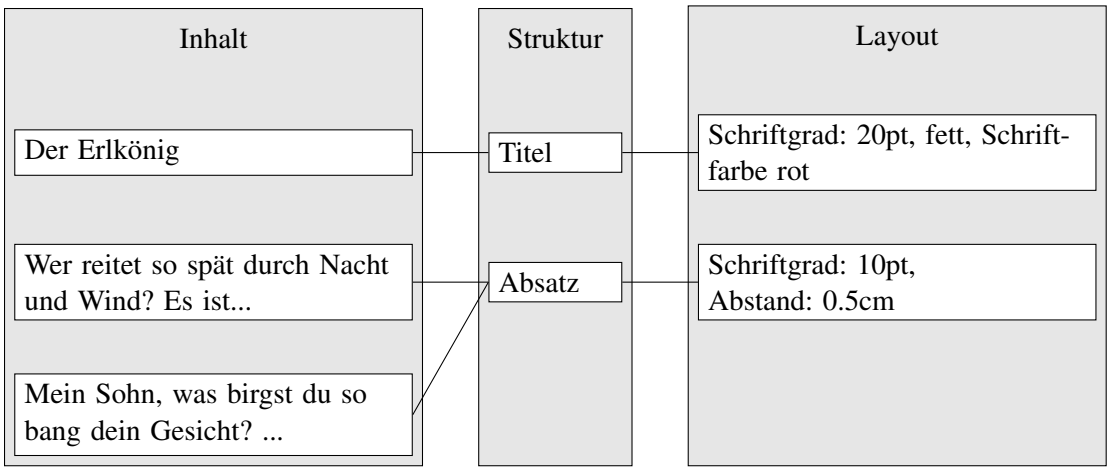

Abbildung 3.1: Trennung von Inhalt und Layout

Durch diese Trennung kann das gleiche Layout für viele unterschiedliche Inhalte (z.B. Webseiten oder Textdokumente) wiederverwendet werden. Andererseits kann der gleiche Inhalt einfach in verschiedenen Layouts dargestellt werden (z.B. als Webseite, al Druckvorlage, für mobile Geräte, usw.)

Mit HTML und CSS wird dieser grundlegende Idee folgendermassen umgesetzt: Die vorgegebenen HTML-Elemente geben die mögliche Struktur eines Dokumentes vor. In einer HTML-Datei wird der Inhalt durch Tags den entsprechenden Strukturen zugeordnet. In einer CSS-Datei wird den HTML-Elementen eine Formatierung zugeordnet:

3.2 Aufbau von CSS

3.2.1 Regeln

Eine CSS-Datei ist einfach aufgebaut: Sie setzt sich aus einer Reihe von *Regeln* zusammen, welche die Formatierung von bestimmten HTML-Elementen festlegen. Eine solche Formatierungsregel ist folgendermassen aufgebaut:

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

Der *Selektor* legt fest, auf welche HTML-Elemente die Regel angewendet werden soll. In den geschweiften Klammern stehen eine oder mehrere *Deklarationen*, welche den *Wert* einer bestimmten *Eigenschaft* definieren. Zum Beispiel selektiert die folgende Regel sämtliche `<h1>`-Elemente und legt die Schriftgrösse auf 48 Pixel fest:

```
h1 { font-size: 48px; }
```

Die Anzahl Leerzeichen und Zeilenumbrüche innerhalb einer Regel spielen dabei keine Rolle. Regeln mit mehreren Deklarationen werden üblicherweise auf mehrere Zeilen verteilt:

```
table {
  border-width: 1px;
  border-color: rgb(90%, 90%, 100%);
  border-style: double;
}
```

3.2.2 Selektoren

Mit einem Selektor werden die HTML-Elemente ausgewählt, welche zu formatieren sind. Die einfachste Möglichkeit, einen Selektor zu definieren, ist der Name des HTML-Elements anzugeben (ohne die spitzen Klammern). Zum Beispiel wählt der folgende Selektor alle alle `<blockquote>`-Elemente:

```
blockquote
```

Mit dem sogenannten *Universalselektor* $*$ (dem Asterix-Zeichen) werden sämtliche HTML-Elemente ausgewählt. So kann zum Beispiel die Schriftart für eine ganze Webseite festgelegt werden:

```
* { font-family: sans-serif; }
```

Wenn nicht alle Vorkommen eines HTML-Elements gleich formatiert werden sollen, kann mit sogenannten *Klassen* gearbeitet werden. Einem HTML-Element kann mit dem Attribut class eine oder mehrere Klassen zugeordnet werden:

```
<p class="wichtig">1. Wichtiger Text, hevorgehoben.</p>
<p>2. Normaler Text mit <i class="wichtig">wichtigem</i> Wort.</p>
```

Im Stylesheet kann dann der HTML-Elementname und der Klassenname mit einem Punkt zu einem Selektor kombiniert werden. Die folgende Regel wird also auf alle `<p>`-Elemente mit der Klasse `wichtig` angewendet:

```
p.wichtig { background-color: yellow; }
```

Sie unterlegt also den ersten Absatz aus dem obenstehenden HTML-Schnippsel mit gelb. Um auch das Wort ”wichtigem” aus dem zweiten Absatz gelb zu unterlegen, kann der Universalselektor in Kombination mit der Klasse verwendet werden:

```
*.wichtig { background-color: yellow; }
```

3.3 Einbindung in HTML

Um ein Stylesheet in ein HTML-Dokument einzubinden, muss in diesem eine entsprechende Referenz eingetragen werden. Dies geschieht mit einem `<link>`-Element innerhalb des `<head>`-Elements:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
```

```
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" href="mystyle.css">
    <title>Titel der Webseite</title>
  </head>
  <body>
     <!-- Hier steht der eigentliche Inhalt. -->
  </body>
</html>
```

Als Wert des Attributs href wird die URL des Stylesheets angegeben. Zusätzlich kann mit dem Attribut `media` definiert werden, für welche Ausgabemedien das Stylesheet verwendet werden soll. So kann für die Darstellung einer Webseite am Bildschirm, auf mobilen Geräten oder für das Drucken ein unterschiedliches Layout definiert werden.

media-Attributwert	Bedeutung
`all`	Die CSS-Datei gilt für alle Ausgabemedien.
`braille`	Die CSS-Datei wird für Braillezeilen verwendet. Das sind spezielle Ausgabegeräte für sehbehinderte Menschen, welche Brailleschrift darstellen können.
`handheld`	Die CSS-Datei definiert das Layout für mobile Geräte.
`print`	Die CSS-Datei beim Ausdrucken auf Papier verwendet.
`screen`	Die CSS-Datei gilt für die Darstellung am Computerbildschirm.

Um zum Beispiel eine unterschiedliche Darstellung für die Ausgabe auf den Bildschirm und den Drucker zu erhalten, können Sie folgenden HTML-Code verwenden:

```
<link rel="stylesheet" media="print" href="style-print.css">
<link rel="stylesheet" media="screen" href="syle-screen.css">
```

3.4 Werte

Einige typische Werte werden für viele unterschiedliche CSS-Eigenschaften verwendet. Folgend werden Zahlen- und Farbwerte sowie URLs erklärt.

3.4.1 Zahlen / Masse

Die meisten Zahlwerte sind Längen- oder Grössenmasse. Für diese können unterschiedliche Einheiten verwendet werden:

Einheit	Erklärung
pt	Punkt, eine typografische Einheit, 1 pt entspricht 1/72 Zoll (inches).
pc	Pica, eine typografische Einheit, 1 pc entspricht 12 Punkt.
in	Zoll (inch), 1 in entspricht 2.54 cm.
mm	Millimeter
cm	Zentimeter
px	Pixel
%	Prozent

3.4.2 Farben

Für Farbwerte wird das RGB-Farbmodell verwendet. In CSS werden die Rot-, Grün- und Blauanteile einer Farbe in Prozent angegeben. Eine Farbe mit 10% Rotanteil, 20% Grünanteil und 50% Blauanteil wird in CSS so geschrieben:

```
rgb(10%, 20%, 50%)
```

Ausserdem können 16 vordefinierte Farbnamen verwendet werden:

Name	R	G	B	Beschreibung
aqua	0%	100%	100%	aquamarin
black	0%	0%	0%	schwarz
blue	0%	0%	100%	blau
fuchsia	100%	0%	100%	pink
gray	50%	50%	50%	grau
green	0%	50%	0%	grün
lime	0%	100%	0%	hellgrün, neongrün
maroon	0%	100%	0%	kastanienbraun
navy	0%	0%	50%	marineblau
olive	50%	50%	0%	olivgrün
purple	50%	0%	50%	purpurrot, violett
red	100%	0%	0%	rot
silver	80%	80%	80%	silbern, hellgrau
teal	0%	50%	50%	blaugrün, smaragdgrün
white	100%	100%	100%	weiss
yellow	100%	100%	0%	gelb

3.4.3 URLs

In CSS werden URLs zum Beispiel verwendet, um ein Hintergrundbild anzugeben. Wie URLs aufgebaut sind, wird in Abschnitt 1.2 beschrieben. In CSS muss die URL in runden Klammern stehen, ausserdem muss die Zeichenfolge `url` vorangestellt werden. Das sieht dann beispielsweise so aus:

```
url(images/test.png)
url(http://gymkirchenfeld.ch/logo.png)
```

3.5 Eigenschaften

3.5.1 Schriftformatierung

Die Darstellung der Schrift kann mit folgenden Eigenschaften beeinflusst werden:

Eigenschaft	Bedeutung
font-family	Definiert die Schriftart wie Arial, Helvetica oder Times Roman
font-variant	Legt fest, ob Kapitälchen verwendet werden
font-style	Legt fest, ob Kursivschrift verwendet wird
font-size	Bestimmt die Grösse der Schrift
font-weight	Definiert das Gewicht, also die Dicke und Stärke der Schrift
color	Legt die Schriftfarbe fest

Schriftart

Mit der Eigenschaft `font-family` kann die die Schriftart wie Arial, Helvetica oder Times Roman definiert werden. Es können auch mehre Schriftarten durch Kommas getrennt angegeben werden. In diesem Fall wird die erste Schriftart verwendet, die auf dem Gerät verfügbar ist. Folgende generische Schriftarten sind vordefiniert - sie sind immer verfügbar:

font-family	**Bedeutung**
serif	Schriftart mit Serifen
sans-serif	Schriftart ohne Serifen
cursive	Kursive Schriftart
monospace	Schriftart mit gleichmässigem Zeichenabstand

Schriftstil

Der Stil der Schrift kann mit verschiedenen Eigenschaften beeinflusst werden.

Mit `font-variant` kann der Text in Kapitälchen (das sind kleine Grossbuchstaben) dargestellt werden.

```
font-variant: small-caps;
```

Mit der Eigenschaft `font-style` kann der Text in kursiver Schrift dargestellt werden.

```
font-style: italic;
```

Die Eigenschaft `font-weight` legt die Dicke der Schrift fest. Hier sind folgende Angaben möglich:

font-weight	**Bedeutung**
bold	fett
bolder	sehr fett
lighter	dünn
normal	normale Dicke

Ausserdem können die Werte 100 (sehr dünn) bis 900 (sehr fett) in Schritten von 100 angegeben werden.

```
font-weight: bold;
```

Schriftgrösse

Die Schriftgrösse wird mit der Eigenschaft `font-size` festgelegt. Dabei wird die Grösse als nummerischer Wert angegeben, wie in Abschnitt 3.4.1 beschrieben. Es kann aber auch eine der folgenden Konstanten eingesetzt werden:

font-size	Bedeutung
xx-small	sehr sehr klein
x-small	sehr klein
small	klein
medium	mittel
large	gross
x-large	sehr gross
xx-large	sehr sehr gross
smaller	kleiner als das Elternelement
larger	grösser als das Elternelement

Dabei geben die zwei letzten Konstanten die Grösse relativ zum Elternelement an. Dies ist auch für Prozentwerte der Fall. Eine relative Grössenangabe ist zum Beispiel bei Inline-Elementen sinnvoll. Beispielsweise möchten wir Hervorhebungen (HTML-Inline-Element `<em>`) in Kapitälchen darstellen, aber die Schrift ein wenig kleiner als den normalen Text halten. Dies kann mit folgender CSS-Regel erreicht werden:

```
em {
  font-size: 80%;
  font-variant: small-caps;
}
```

3.5.2 Hintergrund

Der Hintergrund eines HTML-Elements kann mit folgenden CSS-Eigenschaften definiert werden:

Eigenschaft	Bedeutung
background-color	Definiert die Hintergrundfarbe.
background-image	Legt ein Hintergrundbild fest.
background-repeat	Legt Wiederholung des Hintergrundbildes fest.
background-attachment	Legt Scrolling des Hintergrundbildes fest.

Mit `background-color` wird die Farbe des Hintergrundes festgelegt. Hier kann ein Farbwert angegeben werden.

Die restlichen Eigenschaften beziehen sich auf das Hintergrundbild. Mit `background-image` kann der URL einer Bilddatei angegeben werden. Die Eigenschaft `background-repeat` legt fest, ob das Hintergrundbild wiederholt werden soll. Folgende Werte sind möglich:

background-repeat	Bedeutung
repeat	Wiederholung in beiden Richtungen (Standardwert)
repeat-x	Wiederholung nur horizontal
repeat-y	Wiederholung nur vertikal
no-repeat	keine Wiederholung

Mit `background-attachment` kann festgelegt werden, ob das Hintergrundbild mitgescrollt werden soll. Verwenden Sie einen der folgenden Werte für diese Eigenschaft:

background-attachment	Bedeutung
scroll	Das Hintergrundbild bewegt sich mit dem Text (Standardwert)
fixed	Das Hintergrundbild bleibt stehen

Mit der folgenden CSS-Regel wird das ein Hintergrundbild definiert, das in vertikaler Richtung wiederholt wird und beim scrollen fixiert bleibt. Der nicht vom Bild bedeckte Hintergrund wird hellblau eingefärbt:

```
body {
  background-color: rgb(60%,60%,100%);
  background-image: url(wallpaper.png);
  background-repeat: repeat-y;
  background-attachment: fixed;
}
```

3.5.3 Rahmen

Um ein HTML-Element kann ein Rahmen gezeichnet werden. Dazu können folgende CSS-Eigenschaften verwendet werden:

Eigenschaft	Bedeutung
`border-color`	Farbe des Rahmens
`border-style`	Art des Rahmens
`border-width`	Dicke des Rahmens

Für `border-width` wird ein nummerischer Wert angegeben, für die Darstellung auf dem Bildschirm typischerweise in Pixel (`px`). Bei `border-color` muss ein Farbwert angegeben werden. Für `border-style` sind folgende Werte möglich:

`border-style`	**Bedeutung**
`none`	kein Rahmen
`dotted`	gepunktete Linie
`dashed`	gestrichelte Linie
`solid`	durchgezogene Linie
`double`	Doppellinie
`groove`	3D-Effekt
`ridge`	3D-Effekt
`inset`	3D-Effekt, vertieft
`outset`	3D-Effekt, erhoben

Die obenstehenden Eigenschaften legen den Rahmen auf allen Seiten eines HTML-Elements fest. Sie können aber auch für verschiedene Seiten eines Elements unterschiedliche Rahmen definieren. Dazu werden folgende Eigenschaften verwendet:

Farbe	Stil	Breite
`border-top-color`	`border-top-style`	`border-top-width`
`border-bottom-color`	`border-bottom-style`	`border-bottom-width`
`border-left-color`	`border-left-style`	`border-left-width`
`border-right-color`	`border-right-style`	`border-right-width`

Die folgende Regel definiert beispielsweise eine hellblaue Doppellinie als Rahmen für Bilder:

```
img {
  border-color: rgb(90%, 90%, 100%);
  border-style: double;
```

```
  border-width: 1px;
}
```

Normalerweise wird um jedes HTML-Element ein Rahmen gezeichnet. Bei Tabellen erhält somit jede einzelne Zelle einen Rahmen. Oft möchte man jedoch die Rahmen um verschiedene Zellen zusammenfallen lassen. Dies kann mit der CSS-Eigenschaft `border-collapse` gesteuert werden (siehe Abbildung 3.2).

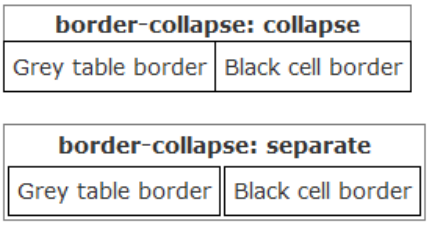

Abbildung 3.2: Darstellung mit *border-collapse*

3.5.4 Abstände

Für jedes HTML-Element können die Abstände zu den Nachbarelementen festgelegt werden. Dabei werden zwischen Innen- und Aussenabständen unterschieden. Die Innenabstände geben den Zwischenraum zwischen dem Elementinhalt und dem Rand des Elements an, die Aussenabstände die Entfernung zwischen den Rändern zweier Elemente. Der Rand eines Elements spielt insbesondere beim Rahmen und Hintergrund eine wichtige Rolle. Die Abbildung 3.3 illustriert dies.

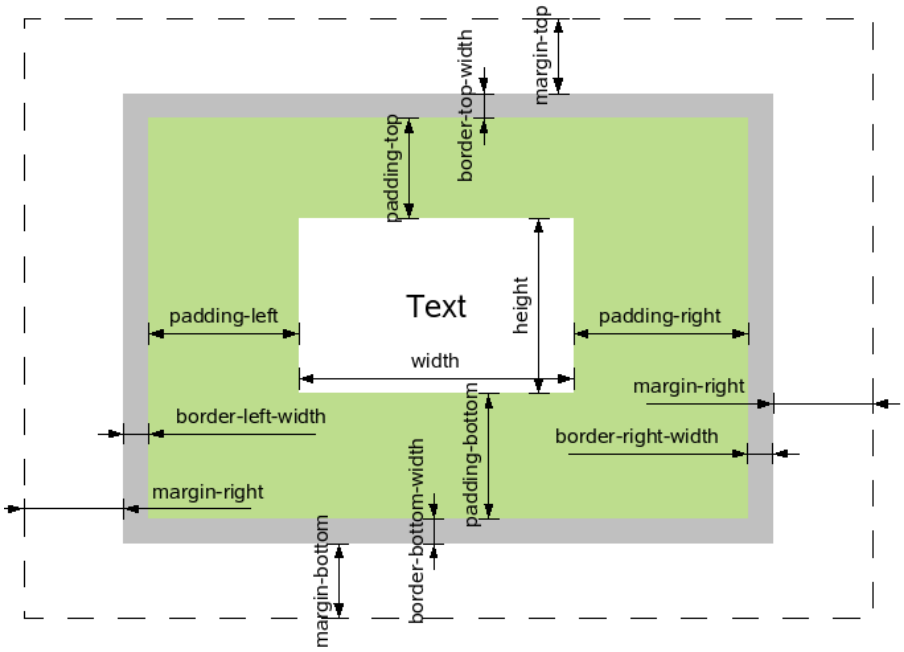

Abbildung 3.3: CSS-Rahmenmodell

In CSS wird der Innenabstand mit der Eigenschaft `padding`, der Aussenabstand mit `margin` festgelegt. Wie bei den Rahmen gibt es hier die Möglichkeit, die Werte für alle vier Seiten einzeln festzulegen. In der folgenden Tabelle sind die Eigenschaften aufgeführt:

Innenabstand	Aussenabstand
`padding`	`margin`
`padding-top`	`margin-top`
`padding-bottom`	`margin-bottom`
`padding-left`	`margin-left`
`padding-right`	`margin-right`

3.5.5 Ausrichtung

Mit der Eigenschaft `text-align` wird die Textausrichtung festgelegt. Folgende vorgegebene Werte sind möglich:

`text-align`	Bedeutung
`left`	linksbündig
`right`	rechtsbündig
`center`	zentriert
`justify`	Blocksatz (links- und rechtsbündig)

3.5.6 Textfluss

Einigen Elemente einer Webseite, etwa Bilder, möchte man häufig vom Text umfliessen lassen. Dazu muss im Element, welches vom Text umflossen werden soll, die CSS-Eigenschaft `float` gesetzt werden. Indem der Wert dieser Eigenschaft auf `left` oder `right`, wird das Element am linken bzw. rechten Rand positioniert und vom folgenden Text umflossen:

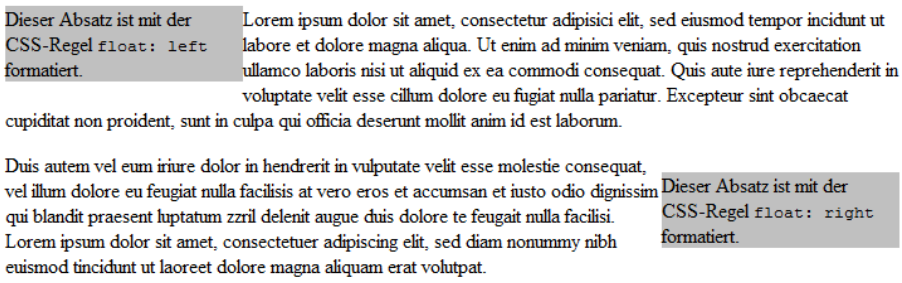

Abbildung 3.4: Effekt von `float`

Um das Umfliessen zu beenden, kann bei einem nachfolgenden Element die Regel `clear: both` angewendet werden. Damit ist sichergestellt, dass das Element unterhalb von sämtlichen `float`-Elementen positioniert wird. Zum Beispiel bei Überschriften kann dies sinnvoll sein.

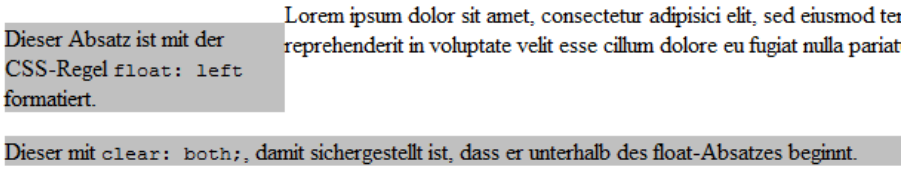

Abbildung 3.5: Effekt von `clear`

3.5.7 Mauszeiger

Mit der Eigenschaft `cursor` kann festgelegt werden, wie der Mauszeiger über dem Element aussehen soll. Folgende Werte sind möglich:

`cursor`	**Bedeutung**
`auto`	Cursor wird automatisch festgelegt (Standard)
`default`	
`crosshair`	Cursor hat die Form eines Fadenkreuzes
`pointer`	Cursor hat die Form eines Zeiger
`text`	Cursor signalisiert, dass Texteingabe möglich ist
`wait`	Cursor signalisiert, dass gewartet werden muss