PC-Technik, Software

Die eigene Website

PC-Technik

Die eigene Website

Hier wird beschrieben, wie aufgrund bestimmter Anforderungen das Grundgerüst einer Website entworfen werden kann. Da diese Beschreibung im Nachgang des Website-Entwurfs entstand, dient sie gleichzeitig als Dokumentation für die Website www.pegons-web.de und als Vorlage für Websites mit ähnlichen Anforderungen.

Die allgemeine Erläuterung und Anwendung von HTML und CSS ist nicht Gegenstand dieser Beschreibung. Es werden nur bestimmte Aspekte der HTML-Erstellung beschrieben, die für den Aufbau der eigenen Website von Bedeutung sind.

Ziele und Anforderungen

Je nach Einsatzzweck und Zielgruppe der Website sind ganz unterschiedliche Anforderungen an den Website-Aufbau zu stellen. Auch die Werkzeuge zur Erstellung der Website (z.B. HTML-Editor) müssen so gewählt werden, dass sich die Website-Inhalte problemlos realisieren lassen. Sind z.B. bestimmte Inhalte aus einer Datenbank einzubinden, sind andere Werkzeuge nötig im Gegensatz zu Websites mit rein statischem Inhalt. Auch die Verfügbarkeit und die Fertigkeit, solche Werkzeuge zu handhaben (Qualifikation des Website-Erstellers) ist Voraussetzung, die Website entsprechend der gestellten Anforderungen realisieren zu können.

Auch das Zielsystem zur Darstellung der Website hat Einfluss auf die Website-Gestaltung. Neben den üblichen Browsern auf Desktop-PCs und Laptops müssen Anforderungen für den Druck (auf Papier) und/oder bei der Darstellung auf kleineren Bildschirmen (Tablets, Handys u.ä.) berücksichtigt werden.

Hier werden nur die Anforderungen an die eigene Website (pegons-web) beschrieben. Daraus folgt dann der spezielle Layout-Aufbau in Verbindung mit dem HTML-Grundgerüst und der zugehörigen CSS-Eigenschaften.

Allgemeine Zielstellung

Allgemeines Ziel soll eine Website mit statischen Inhalten sein, für deren Erstellung nur HTML-Code in Verbindung mit CSS verwendet wird. Daraus ergibt sich, dass notfalls kein oder nur ein einfacher HTML-Editor nötig ist. Außerdem soll die Website ein Layout haben, dass auf dem Bildschirm nahezu genauso aussieht wie beim Druck. Vorrangig soll die Sicherstellung des Druck-Layouts sein, ohne dass auf übliche Layout-Merkmale druckbasierter Publikationen verzichtet wird.

Besonders für die letzte Forderung ist ein Spagat zwischen der sequentiellen Inhalts-Darstellung auf dem Bildschirm und der seitenbasierten (A4) Darstellung auf dem Papier nötig. Auch die Verlinkung innerhalb und außerhalb der Website muss restriktiv gehandhabt werden, da bekanntlich auf dem Papier keine elektronischen Links möglich sind. Die wichtigste "Verlinkung" in einem Buch ist das Inhaltsverzeichnis im Zusammenhang mit der Seitennummer. Diese Verlinkung sollte ansatzweise auch auf der Website realisiert werden. Dabei werden zwar viele Link-Möglichkeiten auf dem Browser verschenkt, aber das druckgerechte Layout soll im Vordergrund stehen.

Aus dem Vorgenannten ergibt sich auch, dass das Zielsystem der Website so gut wie ausschließlich Bildschirme sind, die mindestens eine bestimmte Pixelzahl breit sind. Diese Breite ergibt sich aus dem Layout auf der A4-Seite. Es war zu fordern, dass der Bildschirm mindestens 800 Pixel breit ist. Nur so lässt sich die gleiche Information einer Zeile sowohl auf einer A4-Seite als auch auf dem Bildschirm bei vernünftiger (und gleicher) Schriftgröße unterbringen.

Es ergeben sich folgende Haupt-Merkmale, die erfüllt werden sollen:

- Auf A4-Seite basierendes Web-Design mit festem Drucklayout
- Statische Inhalte (nur HTML/CSS), sparsame Verwendung von Links
- Jede HTML-Datei bildet eine themenbezogene Publikation mit Inhaltsverzeichnis und Inhalt.
- Zu jeder HTML-Datei gibt es zusätzlich eine PDF-Datei für den Druck.

Navigation und externe Links

Oft sind auf einer HTML-Seite dermaßen viele interne und externe Links angegeben, dass die Übersicht schnell verloren gehen kann und dass man dort landet, wo man eigentlich nicht hinwollte. Insbesondere bei sich dynamisch verändernden Seiten ist dies vielleicht nicht zu umgehen. Sind dann manche Links nicht unmittelbar als solche erkennbar, ohne dass man sie anklickt, läuft der eigentlich gewollte Verweis auf eine weitere Information ins Leere. Selbst Links, die erst beim Überfahren mit dem Mauszeiger als Links erkennbar werden, fallen meiner Meinung nach in die Kategorie sinnloser Links. Absolut verwerflich sind natürlich schlecht erkennbare (oder auch getarnte) Links, die beim Anklicken irgend ein nicht gewolltes Ereignis auslösen, das vielleicht dann noch kostenpflichtig ist oder auf Websites unseriöser Angebote führt.

Entsprechend dieser Erfahrungen sollte auf der eigenen Website eine Navigationsstruktur realisiert sein, die einfach, gut erkennbar und eindeutig ist.

Grundforderungen an die interne Navigation

Die interne Navigation soll folgende Grundforderungen erfüllen:

- Alle HTML-Seiten mit gleicher Navigationsstruktur
- Möglichst einfache Navigation innerhalb einer HTML-Seite
- Oben angeordnete Navigationszeile als Verteiler zu weiteren HTML-Seiten
- Nur geringe Tiefe der HTML-Seiten-Verknüpfung
- Kurzer Weg (2 Klicks) bis zum Datenschutz bzw. Website-Betreiber
- Kurzer Weg (1 Klick) bis zur Index-Datei (Home)
- Verlinkung nur statischer Inhalte nötig
- Leichte Wartbarkeit der internen Links
- Selbsterklärende Navigations-Symbole und eindeutige Kennzeichnung der Link-Texte
- Einbeziehung der Browser-Navigationspfeile

Externe Links

Unter externen Links werden hier nur solche Links (Verbindungen) verstanden, die durch einen einzigen Klick von der eigenen Website aus zu einer fremden Website und damit zu fremden Inhalten führen. Das bedeutet, dass die Angabe einer fremden Webadresse in Textform ohne direkte Verlinkung auf die angegebene Webadresse nicht als (direkter) externer Link gilt.

Es ist nicht unbedingt nötig, aber einfacher gestaltet sich der Umgang mit Verweisen auf fremde Inhalte, wenn keine direkten externen Links zur Anwendung kommen. Es ergeben sich folgende Vor- und Nachteile:

- Ohne direkte externe Links wird die eigene Verantwortlichkeit für fremde Inhalte, die aufgrund einer Information auf der eigenen Website von dieser aus erreichbar sind, regelmäßig ausgeschlossen.
- Es müssen keine direkten externen Links gepflegt (überprüft) werden, um sicherzustellen, dass die Ziele auch nach längerer Zeit mit nur einem Klick noch erreichbar sind.
- Direkte externe Links können zum schnellen Verlassen der eigenen Website führen. Sind sie nicht vorhanden, wird dies erschwert. Letzteres kann vom Besucher als Nachteil empfunden werden, da eine gewünschte fremde Information nicht mehr mit nur einem Klick aufrufbar ist.
- Der Verweis auf fremde Inhalte ist ohne direkte externe Links trotzdem möglich, wenn die entsprechende Webadresse in reiner Textform angegeben wird. Zwingend für die Angabe einer Webadresse ist z.B. der vom Urheber geforderte Verweis auf lizenzierte fremde Werke, die in der eigenen Website verwendet werden. Da aber in gedruckten Werken ebenfalls kein direkter Link möglich ist, reicht normalerweise in allen Fällen der Urheberrechtsnachweis in reiner Textform aus.

HTML-Erstellung

HTML und CSS

Sämtliche Design-Eigenschaften sind als CSS (Cascading StyleSheet) definiert, die als Ergänzung zu HTML die Festlegung der HTML-Element-Formateigenschaften gestattet. Die Definition zentraler Formate (Stylesheets) in einer CSS-Datei ermöglicht nicht nur einen schlanken HTML-Code, sondern sichert auch das durchgängig einheitliche Aussehen bei der Darstellung im Browser und später beim Druck. Die zentrale Formatierung ist vergleichbar mit den zentralen Formatfestlegungen in einem Satzprogramm.

Eine direkte Formatierung im HTML-Code in Form von Attributen (Eigenschaft eines HTML-Elements) gibt es bis auf wenige Ausnahmen nicht. Eine solche Ausnahme ist z.B. der Zeilenumbruch
, der direkt als Standalone-Tag im Text steht.

Einbinden der CSS-Anweisungen

Alle CSS-Anweisungen (oder auch CSS-Regeln) stehen in separaten (externen) CSS-Dateien. Eigentlich reicht eine einzige CSS-Datei aus. Sollen aber Publikationen mit unterschiedlichem Layout realisiert werden, ist für jedes Layout eine neue CSS-Datei erforderlich. Die Layouts unterscheiden sich aber nur in wenigen Eigenschaften (z.B. Schrifttyp und Schriftgröße oder auch Ausrichtung beim Satz, wie Blocksatz oder Flattersatz). Viele Layout-Eigenschaften sind identisch. Es ist deshalb sinnvoll, das Grundlayout in einer Basis-CSS-Datei festzulegen. Die abweichenden CSS-Formate können dann in einer weiteren dem jeweiligen Layout zugeordneten CSS-Datei definiert sein. Zu jeder HTML-Datei gehören dann also 2 CSS-Dateien.

Anmerkung:
Eine Aufteilung der CSS-Anweisungen in Abhängigkeit vom Ausgabemedium (Bildschirm oder Druck) wäre auch möglich. Es würden dann zu jeder HTML-Datei eine CSS-Datei für das Medium "screen" und eine CSS-Datei für das Medium "print" existieren. Wenn dann noch abweichende CSS-Formate für eine bestimmte Gruppe von HTML-Dateien (z.B. Reiseberichte) definiert werden sollen, werden evtl. 3 oder 4 CSS-Dateien nötig. Das ist mir zu kompliziert.

Die Verlinkung der beiden CSS-Dateien im Header sieht so aus:

```
<head>
  ... ((weitere Befehle))
  <link href="styles/pebu0.css" type="text/css" rel="stylesheet">
  <link href="styles/pebu1.css" type="text/css" rel="stylesheet">
</head>
```

Die Datei pebu0.css ist die Basis-CSS-Datei. Sie ist für alle HTML-Dateien gültig. Die Datei pebu1.css ist eine der zusätzlichen CSS-Dateien, die für bestimmte Gruppen von HTML-Dateien definiert sind. Alle CSS-Dateien stehen im Unterverzeichnis "styles/...".

Die Reihenfolge der Verlinkung ist wichtig, da für den Browser immer die zuletzt gelesene CSS-Anweisung gültig ist. Das bedeutet, wurde z.B. in pebu0.css für den p-Text die Schriftgröße 10 pt festgelegt, in der pebu1.css aber 12 pt, dann wird der p-Text in 12 pt dargestellt.

CSS-Anweisungen im HTML-Header
Sollen darüber hinaus für eine einzelne HTML-Datei nur einige Formate geändert werden, ist die Festlegung als CSS-Anweisung auch direkt im HTML-Header möglich. Die zugrundeliegenden CSS-Dateien müssen dafür nicht geändert werden, da auch hier immer die zuletzt vom Browser gelesene CSS-Anweisung gültig ist. Frühere Anweisungen des gleichen Elements werden überschrieben.

Zusätzliche CSS-Anweisungen im Header:

```
<head>
  ... ((weitere Befehle))
  <link href="styles/pebu0.css" type="text/css" rel="stylesheet">
  <link href="styles/pebu3.css" type="text/css" rel="stylesheet">
  <style type="text/css">
    #inh h3 { font-size: 11pt; }
  </style>
</head>
```

Hier wurde (abweichend von Anweisungen in den CSS-Dateien) die h3-Überschriften-Größe auf 11 pt festgelegt, wenn sie sich im div-Container mit der id = inh befindet. Dabei gilt diese Änderung sowohl für die Darstellung auf dem Bildschirm als auch für den Druck, da die Medien-Gültigkeit der CSS-Anweisung nicht zusätzlich definiert ist.

Medien-abhängige CSS-Anweisungen

Im folgenden Beispiel wird festgelegt, dass die h3-Überschriften-Größe nur beim Druck ("print") auf 11 pt geändert wird. Bei der Darstellung auf dem Bildschirm ("screen") erfolgt keine Änderung.

Zusätzliche CSS-Anweisungen im Header für den Druck:

```
<head>
  ... ((weitere Befehle))
  <link href="styles/pebu0.css" type="text/css" rel="stylesheet">
  <link href="styles/pebu3.css" type="text/css" rel="stylesheet">
  <style media="print" type="text/css">
    #inh h3 { font-size: 11pt; }
  </style>
</head>
```

Sind bestimmte CSS-Anweisungen nur für ein bestimmtes Medium vorgesehen, wird dies nacheinander in die entsprechende CSS-Datei geschrieben. Fehlt die Angabe "@media ...", gelten die Anweisungen für alle Medien, also sowohl für den Bildschirm als auch für den Druck.

Medien-abhängige CSS-Anweisungen in der CSS-Datei:

```
@media screen {
a:link { text-decoration: none; } /* default=blau */
a:visited { color: #A00000; text-decoration: none; }
a:hover { color: #FF0000; text-decoration: underline; }
a:active { color: #FF0000; text-decoration: underline; }
}
@media print {
a:link { color: #000000; text-decoration: none; } /* default=blau */
a:visited { color: #000000; text-decoration: none; }
a:hover { color: #000000; text-decoration: none; }
a:active { color: #000000; text-decoration: none; }
}
```

Auf dem Bildschirm sind wie üblich je nach History farbige und teils unterstrichene Links vorgesehen. Beim Druck sind diese Attribute nicht sinnvoll. Deshalb ist bei der Druckausgabe für Links generell die Farbe schwarz und keine Unterstreichung eingestellt. Das hat allerdings zur Folge, dass im Drucklayout Links nicht so gut erkennbar sind. Link-Texte sollten deshalb in Anführungszeichen ("...") stehen und mit eindeutigem Verweistext formuliert werden.

Eine weitere Besonderheit dieser Website ist, dass keine externen Links existieren. Verweise auf externe Inhalte werden aber trotzdem gegeben, allerdings als reiner Text.

Beispiel für einen externen Link, der auch beim Druck als solcher gut erkennbar ist:

SELFHTML ist eine Befehls-Beschreibung zu HTML und CSS (SELFHTML Version 8.0 vom 27.10.2001 von Stefan Münz, als Offline-Version). Die Original-Website dieser Version 8.0 existiert nicht mehr (in 2017). Eine Kopie ist unter "https://www2.informatik.hu-berlin.de/Themen/www/selfhtml/" zu finden.

Hinweis zu Farbflächen beim Druck:
Da beim Drucken Farbflächen nicht vorkommen sollen (außer natürlich in Bildern), ist in der CSS-Datei definiert, dass die obige linke Farbfläche durch 2 senkrechte Linien ersetzt wird.

Die medien-abhängigen CSS-Anweisungen in der CSS-Datei sehen so aus:

```
@media screen {
.elook { border-left-style: solid; border-left-width: 10px; border-left-color: #FF9900;
padding-left: 5px; }
}
@media print {
.elook { border-left-style: double; border-left-width: 6px; border-left-color: #F8B82A;
padding-left: 5px; }
}
```

Natürlich können normalerweise die Farbflächen gedruckt werden. Ich vermeide dies aber, wo es geht. Man spart so Tinte bzw. Toner. Der Hauptgrund ist aber, dass Farbflächen beim S/W-Druck (schwarz/weiß) gerastert werden, was mir persönlich nicht so gut gefällt.

Bildschirm- und Druck-Layout

Druckseiten und Satzspiegel

Eigentlich gibt es bei HTML wegen der sequentiellen Darstellung keinen Satzspiegel. Der Satzspiegel ist seitenorientiert und bezeichnet den grundsätzlichen Aufbau einer Druckseite. Dazu gehören die Seitenränder, die Breite des Bodytextes (Fließtext) und seine Spaltenaufteilung, die Schriftarten und -größen u.a.

Mit einigen Kniffen kann aber auch im HTML-Layout eine Abfolge von A4-Seiten nachgebildet werden. Dazu ist es nötig, entsprechende Kopf- und Fußzeilen einzufügen, die sowohl in HTML als auch beim Druck gültig sind. Auch die Seiten-Nummerierung und ein Inhaltsverzeichnis gelten für den Bildschirm und für die Druckausgabe gleichermaßen. Durch die zusätzlich generierten PDF-Dateien wird die Sicherheit für das gewollte Drucklayout erhöht, da irgendwelche Browser-Einstellungen nicht mehr wirksam sind. Das seitenorientierte PDF wurde ja extra dafür entwickelt, zu druckende Inhalte zu portieren und auf allen Druckern gleich aussehen zu lassen.

Kopfzeile und Seiten-Nummerierung

Jede A4-Druckseite ist bereits im sequentiell anzeigenden Browser als solche erkennbar. Dazu gibt es für jede Druckseite eine Kopfzeile, die im Browser den Beginn einer A4-Druckseite markiert, so wie die Kopfzeile auch auf der A4-Druckseite am oberen Rand des A4-Blatts zu sehen ist. Die Kopfzeile enthält linksbündig eine Information zur Publikation und zum Kapitel, rechtsbündig den Copyright-Vermerk mit Jahreszahl der Erstellung bzw. der letzten Änderung oder Ergänzung. Die Kopfzeile ist im Browser und auf der A4-Seite identisch, beim Druck allerdings ohne hinterlegter Farbfläche.

Zusätzlich zur Kopfzeile ist eine automatische Seiten-Nummerierung eingebaut. Das wird mit einem Zähler realisiert, der mit jeder Kopfzeile um 1 weiterzählt. Am Ende der Publikation steht somit im Zähler die Seitenzahl der HTML-Datei bzw. die der HTML-Datei zugeordnete Anzahl von A4-Seiten.

Die aktuelle Seiten-Nummer wird in der HTML-Datei rechts außerhalb des Body-Bereichs sichtbar dargestellt. Sie dient als Kennzeichen für das Ende des laufenden HTML-Abschnitts, der einer A4-Seite entspricht. Das heißt, bei der HTML-Erstellung ist spätestens kurz nach Erscheinen der Seiten-Nummer eine neue Kopfzeile einzufügen, um damit auch eine neue A4-Seite zu beginnen.

Fußzeile

Im Drucklayout erscheit die gleiche Seiten-Nummer in der automatisch generierten Fußzeile unten rechts. Linksbündig enthält die Fußzeile die Domain (www.pegons-web.de).

Bei der HTML-Darstellung im Browser gibt es zur Kennzeichnung des Endes einer A4-Druckseite nur die Seiten-Nummer außerhalb des Body-Bereichs. Es gibt aber keine eigentliche Fußzeile zu jeder A4-Seite. Dafür gibt es aber eine Fußzeile für die gesamte HTML-Datei, die ja einer Publikation entspricht. Diese Fußzeile wird von Hand in die HTML-Datei geschrieben und nicht mitgedruckt.

CSS-Festlegungen für Kopf-/Fußzeilen und Seiten-Nummer

In pebu0.css definiert:

```
@media screen {
div.kopf1 { background-color: #F4E99C; margin-top: 2pt; padding-bottom: 1pt; }
div.kopf { clear: both; background-color: #F4E99C; margin-top: 4pt; margin-bottom: 4pt; }
div.footer { margin-top: -6pt; position: relative; top: 1090px; margin-right: -18px; }
div.reseiz:after {counter-increment: seiz; content: "" counter(seiz) ""; }
/* reseiz = rechts-seitenzahl */
.nos { display: none } /* existiert nur im Screen-Stylesheet */
}
@media print {
div.kopf1 { margin-bottom: 6pt; border-bottom-style: solid; border-bottom-color: #F8B82A;
border-bottom-width: 0.5pt; padding-bottom: 1pt; }
div.kopf { page-break-before: always; clear: both; margin-bottom: 6pt; }
div.footer { margin-top: -6pt; position: relative; top: 1095px; }
div.reseiz:after {counter-increment: seiz; content: "Seite " counter(seiz) "";}
.nop { display: none }
/* existiert nur im Print-Stylesheet, für Wiederholungs-Zeichnung und Navigation */
}
.hlin { border-bottom-style: solid; border-bottom-color: #F8B82A; border-bottom-width:
0.5pt; padding-bottom: 1pt; }
/* 1 horizontale farbige Linie */
```

In pebu1.css definiert:

```
div.reseiz { text-align: right; font-family: Verdana,Tahoma,Arial,sans-serif;
  font-size: 8pt; } /* rechter Text (Seitenzahl) */
.le { clear: both; float: left; font-family: Verdana,Tahoma,Arial,sans-serif;
  font-size: 8pt; } /* linker Text für Kopf- oder h1-Zeile */
.re { text-align: right; font-family: Verdana,Tahoma,Arial,sans-serif;
  font-size: 8pt; } /* rechter Text bzw. PDF-Symbol für Kopf- oder h1-Zeile */
```

div.kopf1 und div.kopf

Beide div-Container umschließen die Kopfzeile, und zwar div.kopf1 die 1. Kopfzeile in der HTML-Datei und div.kopf jede weitere Kopfzeile. Bei der 1. Kopfzeile erfolgt vor der Kopfzeile kein A4-Seitenumbruch, dies geschieht erst vor jeder weiteren Kopfzeile mit "page-break-before: always". Etwa noch aktive Spalten werden mit "clear: both" beendet. Die Kopfzeilen-Hintergrundfarbe im Browser wird mit "background-color: #F4E99C;" festgelegt.

Das Drucken erfolgt ohne Kopfzeilen-Hintergrundfarbe. Dafür ist beim Druck für alle Kopfzeilen eine farbige Linie unten vorgesehen (border-bottom-style: solid; border-bottom-color: #F8B82A; border-bottom-width: 0.5pt; padding-bottom: 1pt;). Die gleiche Linie erhalten auch die Kopfzeilen div.kopf im Browser (außer der 1. Kopfzeile). Bei den Kopfzeilen div.kopf ist für die Linie Klasse .hlin zuständig, d.h. hlin muss separat angegeben werden. Das ergibt eine größere Flexibilität, da so eine Kopfzeile auch ohne Linie realisierbar ist.

div.footer

Der HTML-Code des div-Containers div.footer folgt unmittelbar nach einer Kopfzeile. Er ist relativ positioniert (position: relative; top: 1095px;). Das bedeutet, div.footer wird erst 1095 Pixel später wirksam. 1095 px ist der Platz, der für eine A4-Seite benötigt wird. Die im footer enthaltene Seiten-Nummer (div.reseiz) wird also am Ende der aktuellen Seite positioniert. Die relative Positionierung ist im Browser Firefox im Vergleich zur Druckansicht des IE etwas unterschiedlich und wurde experimentell ermittelt.

div.reseiz

Die Anzeige für die Seiten-Nummer erfolgt mit "div.reseiz:after {...}". Der Seitenzähler seiz wird um 1 erhöht (counter-increment) und zur Anzeige gebracht. Textausrichtung und Schrift werden in der Klasse div.reseiz in einer weiteren CSS-Datei definiert, da diese Attribute für verschiedene HTML-Dateien unterschiedlich sein können.

Klassen nop und nos

NoPrint (.nop { display: none }) im Print-Stylesheet angewendet bedeutet, dass kein Ausdruck erfolgt. NoScreen (.nos { display: none }) im Screen-Stylesheet angewendet bedeutet, dass keine Anzeige auf dem Bildschirm erfolgt.

Klassen le und re

Diese Klassen werden für Text auf einer Zeile angewendet, der zum Teil linksbündig (.le) und zum Teil rechtsbündig (.re) stehen soll. Das betrifft auch z.B. den linksbündigen Titel und den rechtsbündigen Copyright-Vermerk auf der Kopfzeile. Zusätzlich zur Ausrichtung sind die Schrift und deren Größe festgelegt.

Der HTML-Code für eine Kopfzeile mit Seiten-Nr. sieht so aus (Beispiel):

```
<div class="kopf hlin">
  <div class="le">PC/Software: Die eigene Website</div>
  <div class="re">2017 © Peter E. Burkhardt</div>
</div>
<div class="footer">
  <div class="le nos">www.pegons-web.de</div>
  <div class="reseiz"></div>
</div>
```

Der gesamte HTML-Code im Kasten ist in der HTML-Datei auf eine Zeile geschrieben, davor eine Leerzeile. Dadurch wird die HTML-Datei strukturiert und übersichtlicher. Die Leerzeilen trennen die zu einer A4-Seite gehörenden Abschnitte.

Spaltensatz

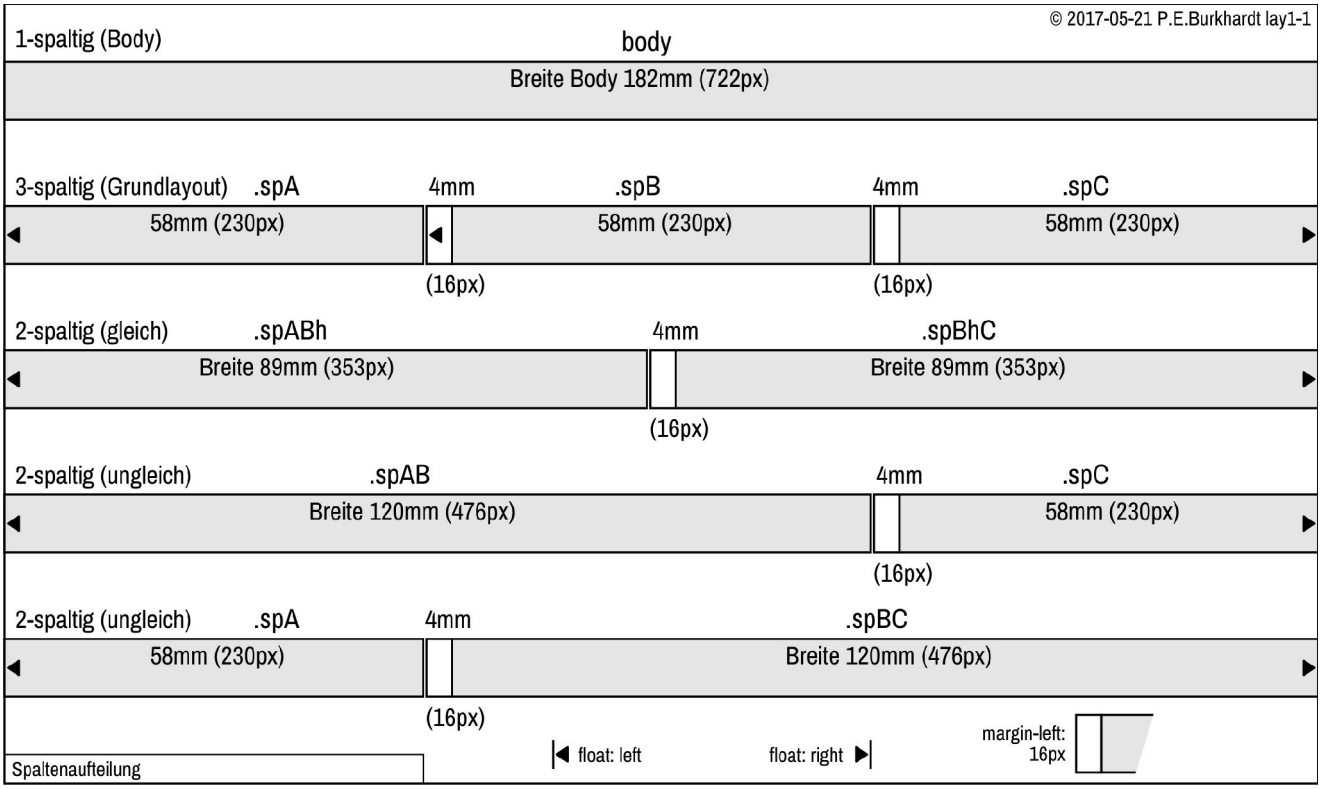
Der Bodytext kann ein- oder mehrspaltig gesetzt sein, das gilt auch für Zeichnungen und Bilder. Dabei kann die Anzahl der Spalten jederzeit gewechselt werden, d.h. auch innerhalb einer A4-Seite. Die Spalten-Steuerung erfolgt über Klassendefinitionen. Vorteilhaft sind div-Container, aber auch direkt z.B. im img-Tag oder im p-Tag.

Allerdings ist es nicht möglich, den Fließtext über mehrere Spalten fließen zu lassen, wie es bei Satzprogrammen der Fall ist. Text und Bilder müssen der jeweiligen Spaltenaufteilung angepasst werden.

Tabellen als Spaltenersatz sind zwar in den CSS-Dateien definiert (Klasse .blind), werden aber nur in ganz wenigen Ausnahmefällen angewendet.

Spaltenaufteilung

Die Body-Breite ist so aufgeteilt, dass sich verschiedene Spaltenbreiten realisieren lassen, die einem bestimmten Grundraster folgen. Die Abstände der Spalten sind gleich.



Für die Spalten ergeben sich folgende CSS-Klassen-Selektoren:

```
/* CSS-Spaltendefinition */
.spA { width: 230px; float: left; } /* Spalte A */
.spB { width: 230px; float: left; margin-left: 16px; } /* Spalte B */
.spC { width: 230px; float: right; margin-left: 16px; } /* Spalte C */
.spAB { width: 476px; float: left; } /* Spalte A+B */
.spBC { width: 476px; float: right; margin-left: 16px; } /* Spalte B+C */
.spABh { width: 353px; float: left; } /* Spalte A+Bh (h=halbe) */
.spBhC { width: 353px; float: right; margin-left: 16px; } /* Spalte Bh+C (h=halbe) */

.sp_c { clear: both; } /* löscht Spalten-Umfließung */
```

Angabe der Bildbreite in einer Spalte

Bilder oder andere Inhalte, die innerhalb einer Spalte keinen Platz finden, werden über die Spaltengrenzen hinweg in voller Größe dargestellt. Das bedeutet, dass die Breite z.B. eines Bildes maximal die Breite der Spalte haben darf, wenn es richtig dargestellt werden soll. Für ein Bild in voller Breite (Body) ist also width="722" anzugeben, für ein Bild z.B. in Spalte A width="230". Die Angabe erfolgt in Pixel.

3-spaltiges Grundlayout

Das Grundlayout ist 3-spaltig (Spalten A, B und C) und füllt die Breite des Bodytextes aus, der in den Body-Eigenschaften definiert ist.

Body-Eigenschaften

```
/* CSS-body-Deklaration */
body { margin-left: 0px; margin-top: 4px; background-color: #FFFFE0; min-width: 722px; }
div#divbody { width: 722px; margin: auto; }
```

Diese Body-Deklaration ist nicht vollständig, enthält aber die zur Spalten-Definition nötigen Angaben. Der Bodybereich füllt das ganze Browserfenster aus und hat als Hintergrund die Farbe hellgelb (background-color: #FFFFE0;). Die Mindesbreite ist auf 722 px festgelegt (min-width: 722px;), unabhängig vom Bodyinhalt.

Der gesamte HTML-Code innerhalb Body befindet sich im div-Container mit dem id-Selektor divbody. Der div-Container divbody wird zentriert (duch margin: auto;) und ist ebenfalls 722 Pixel breit.

Der HTML-Code sieht dann so aus:

```
<body>
<div id="divbody">
...((Inhalt der Webseite))
</div> <!-- div id="divbody" -->
</body>
```

Spalten-Eigenschaften

Die Festlegung auf ein maximal 3-spaltiges Layout erfolgte, um einen Spaltensatz realisieren zu können, wie er auch in Dokumentationen und Zeitschriften üblich ist. Vorteil ist nicht nur die reduzierte Zeilenlänge (bessere Lesbarkeit), sondern auch die mögliche Anordnung von Bildern direkt neben dem beschreibenden Text.

Die Spalten A, B und C sind gleich breit (58 mm) und haben einen Abstand von 4 mm. Die Maße ergeben sich aus der Body-Breite von 182 mm. Die äußeren Ränder (Spalte A linker Rand und Spalte C rechter Rand) werden durch das Drucklayout definiert. Bei 210 mm Breite einer A4-Seite verbleiben daher ein rechter und ein linker Rand von 14 mm.

Diese Ränder können im Browser verändert werden. Um sicher zu gehen, dass der Body-Inhalt auch wirklich auf dem A4-Papier landet, ist im Browser die Einstellung "Auf Seitengröße verkleinern" (Firefox) bzw. "An Größe anpassen" (MS IE) zu aktivieren. Damit wird der Seiteninhalt soweit verkleinert, dass er innerhalb der eingestellten Seitenränder vollständig dargestellt wird, ohne beschnitten zu werden.

Die zusätzlichen PDF-Dateien werden so generiert, dass sich die richtigen Seitenränder und die richtige Lage der Kopf- und Fußzeilen ergeben.

Spalten-Umfließung

Jedes mit der float-Eigenschaft versehene Element wird zum Block-Element und ordnet sich bezüglich seines Eltern-Elements entweder links oder rechts an. Im Falle der definierten Spalten ist Body das Eltern-Element. Ist eine Spalte mit "float: left;" definiert, steht diese Spalte links (am linken Body-Rand), alle folgenden Elemente umfließen diese Spalte rechts. Ist eine Spalte mit "float: right;" definiert, steht diese Spalte rechts (am rechten Body-Rand), alle folgenden Elemente umfließen diese Spalte links.

Eine Sonderstellung nimmt die mittlere Spalte B ein. Sie floatet links, wie Spalte A. Damit Spalte B an der richtigen Stelle erscheint, muss also in HTML vorher Spalte A definiert sein.

Jeder Spalte ist die entsprechende Breite zugewiesen, z.B. für Spalte A "width: 230px;" (Ein Block-Element benötigt diese Angabe sowieso.). Damit umfließende Elemente (z.B. Text oder ein Bild) zur links davon stehenden Spalte (z.B. Spalte A) einen Abstand haben, wird das umfließende Element mit dem gewünschten Abstand (Rand) versehen ("margin-left: 16px;"). Jede weiter rechts stehende Spalte wird also mit einem linken Rand versehen.

Da jeder Spalte nur die Breite, nicht aber die Länge zugewiesen ist, würde sie den Platz bis zum Ende des Body-Bereichs einnehmen, auch wenn die Spalte nicht ausgefüllt ist. Die Aufteilung in druckgerechte A4-Seiten wäre nicht möglich.

Deshalb muss die float-Eigenschaft gelöscht werden. Dies erfolgt mit "clear: both;" (Klasse .sp_c) im auf die Spalte(n) folgenden Element. Das bedeutet ein mit "float: left;" oder mit "float: right;" definierter Umfluss wird abgebrochen. Wird die Klasse .sp_c beispielsweise in einem p-Absatz angewendet, verläuft dieser Absatz über die ganze Body-Breite. Vorher definierte Spalten sind beendet.

Damit ist es möglich, in einer HTML-Datei Bereiche mit verschiedenen Spalten zu definieren. Das Gleiche gilt natürlich auch für eine A4-Seite. Außerdem werden mit jeder neuen A4-Seite vorige Spalten automatisch gelöscht.

Beispiel 2-spaltiges Layout

Entsprechend der CSS-Spaltendefinition (siehe weiter oben) können Spalte A mit B (.spAB) oder Spalte B mit C (.spBC) zu einer Spalte zusammengefasst werden. Zusätzlich ist es aber auch möglich, eine Spalte A und die Hälfte der Spalte B (.spABh) bzw. eine halbe Spalte B mit der Spalte C (.spBhC) zusammenzufassen.

Bei all diesen Kombinationen wird das 3-spaltige Grundraster eingehalten. Insbesondere ist die Anordnung von Bildern unterschiedlicher Breite eine leichte Angelegenheit.

Das folgende Beispiel zeigt die HTML-Grundsequenz für ein 2-spaltiges Layout. Dabei ist die CSS-Klasse "ra" nur eingefügt, um die Spaltengrenzen besser sichtbar zu machen. Es ist belanglos, ob zuerst die Spalte spAB oder zuerst die Spalte spC definiert wird. Es ergibt sich das gleiche Layout im Browser. Fehlt allerdings die CSS-Layoutdatei, wird der Text in der Reihenfolge angezeigt, wie er in der HTML-Sequenz steht.

HTML-Code für ein 2-spaltiges Layout:

```
<div class="spAB ra">
<p>Das ist der Inhalt von Spalte A+B mit Rahmen formatiert. Das ist der Inhalt von Spalte
A+B. Das ist der Inhalt von Spalte A+B. Spalte A+B könnte auch ein Bild sein.</p>
</div>
<div class="spC ra">
<p>Das ist der Inhalt der Spalte C mit Rahmen formatiert. Der linke Abstand zur Spalte
A+B ist in Klasse .spC festgelegt. Spalte C könnte auch ein Bild sein.</p>
</div>
<p class="sp_c">Das ist Fließtext. Damit die vorigen Spalten beendet werden, wird
Klasse .sp_c (Spalten clear) angewandt.</p>
```

2-spaltiges Layout-Beispiel:

Das ist der Inhalt von Spalte A+B mit Rahmen formatiert. Das ist der Inhalt von Spalte A+B. Das ist der Inhalt von Spalte A+B. Spalte A+B könnte auch ein Bild sein.

Das ist der Inhalt der Spalte C mit Rahmen formatiert. Der linke Abstand zur Spalte A+B ist in Klasse .spC festgelegt. Spalte C könnte auch ein Bild sein.

Das ist Fließtext. Damit die vorigen Spalten beendet werden, wird Klasse .sp_c (Spalten clear) angewandt.

Beispiel 3-spaltiges Layout mit Bildern

Dieses 3-spaltige Grundlayout enthält Bilder mit Bild-Unterschriften, die mit Klasse "capt" (capture) formatiert sind. Alle Bilder sind auf Spaltenbreite width="230" Pixel gesetzt. Die Höhe height="32" ergibt sich durch Verkleinerung des Bildes mit der Option "Seitenverhältnis beibehalten" im verwendeten HTML-Entwicklungs-Tool Webocton-Scriptly.

HTML-Code für ein 3-spaltiges Layout mit Bildern und Bild-Unterschriften:

```
<div class="spA">
<img src="images/0elek1.jpg" width="230" height="32" alt="">
<p class="capt">Bild-Unterschrift zum Bild Spalte A</p>
</div>
<div class="spB">
<img src="images/0elek1.jpg" width="230" height="32" alt="">
<p class="capt">Bild-Unterschrift zum Bild Spalte B</p>
</div>
<div class="spC">
<img src="images/0elek1.jpg" width="230" height="32" alt="">
<p class="capt">Bild-Unterschrift zum Bild Spalte C</p>
</div>
<p class="sp_c">Das ist Fließtext. Damit die vorigen Spalten beendet werden, wird
Klasse .sp_c (Spalten clear) angewandt.</p>
```

3-spaltigen Layout-Beispiel:

Bild-Unterschrift zum Bild Spalte A

Bild-Unterschrift zum Bild Spalte B

Bild-Unterschrift zum Bild Spalte C

Das ist Fließtext. Damit die vorigen Spalten beendet werden, wird Klasse .sp_c (Spalten clear) angewandt.

Nach dem gleichen Schema werden auch andere Spaltenkonfigurationen realisiert. Bei nur einspaltigem Layout ist keine Spaltendefinition erforderlich, da die dann einzige Spalte gleichzeitig der Body ist.

Beispiel 1-spaltiges Layout mit umfließendem Text

Soll z.B. rechts ein Bild stehen und links der beschreibende Text, ist es nicht unbedingt nötig, 2 Spalten zu definieren. Es reicht, wenn z.B. die rechte Spalte (mit dem Bild) mit der entsprechenden Spalten-Klasse festgelegt wird. Der links davon stehende Text muss nicht als Spalte festgelegt sein, er umfließt die rechte Spalte.

HTML-Code für 1-spaltiges Layout mit umfließendem Text:

```
<img class="spBhC" src="images/0elek1.jpg" width="353" height="49" alt="">

<p>Das ist der beschreibende Text (einfacher p-Text) zu nebenstehendem Bild. Da dieser Text nicht als Spalte definiert ist, muss er in HTML nach der rechten Spalte (dem Bild) geschrieben werden. Nur so kann der Text das Bild umfließen. Ist der Text länger als die Bildhöhe, fließt der Text bis zum rechten Body-Rand. Ist der Text kürzer als die Bildhöhe, bestimmt die Bildhöhe den Platz der Kombination p-Text mit Bild.</p>

<p class="sp_c">Um die vorige Spalte zu beenden, wird auch hier Klasse .sp_c (Spalten clear) angewandt. Das ist allerdings nur nötig, wenn der umfließende Text kürzer als die Bildhöhe ist. Andernfalls (wie hier im Beispiel) ist das Spalten-Löschen nicht nötig.</p>
```

Die Leerzeilen im Kasten wurden nur wegen der besseren Übersichtlichkeit eingefügt. In der HTML-Datei besteht der gesamte Kasteninhalt aus 3 Zeilen.

1-spaltiges Layout mit umfließendem Text:

Das ist der beschreibende Text (einfacher p-Text) zu nebenstehendem Bild. Da dieser Text nicht als Spalte definiert ist, muss er in HTML nach der rechten Spalte (dem Bild) geschrieben werden. Nur so kann der Text das Bild umfließen. Ist der Text länger als die Bildhöhe, fließt der Text bis zum rechten Body-Rand. Ist der Text kürzer als die Bildhöhe, bestimmt die Bildhöhe den Platz der Kombination p-Text mit Bild.

Um die vorige Spalte zu beenden, wird auch hier Klasse .sp_c (Spalten clear) angewandt. Das ist allerdings nur nötig, wenn der umfließende Text kürzer als die Bildhöhe ist. Andernfalls (wie hier im Beispiel) ist das Spalten-Löschen nicht nötig.

An diesem Besispiel wird deutlich, dass für eine rechte Spalte mit links umfließendem Text (oder auch linkem Block-Element) nur wenig HTML-Code nötig ist. Es reicht, die rechte Spalte mit der entsprechenden Spalten-Klasse festzulegen. Man muss nur darauf achten, dass zuerst die rechte Spalte in der HTML-Sequenz geschrieben wird, dann den links umfließenden Text (oder Texte) oder auch Blockelemente (z.B. Bilder).

1-spaltiges Layout mit umfließendem (kurzen) Text:

Hier ist der Text kürzer als die Bildhöhe, deshalb der Leerraum.

Dieser Text gehört nicht zum Bild. Um die vorige Spalte zu beenden, wird Klasse .sp_c (Spalten clear) angewandt.

PC-Technik

Monitor-Umschalter