# web57 photography

## Template für CMSimple erstellen





## Einleitung

Meine Entscheidung für CMSimple war, unter Anderem, der erfreuliche Umstand, daß ich bei meinem zweiten Versuch CMSimple zu benutzen auf CMSimple_XH 1.2 von Gert Ebersbach stieß. Mein erster Anlauf scheiterte ja primär daran, daß es 2005 kaum nutzbare, attraktive Templates gab (für mich) und das Angebotene zu 99% auf Tabellen basierte. In einer vielfach verschachtelten Tabelle, geschickterweise vielleicht auch noch mit Hintergrund Gifs, irgendetwas anzupassen oder zu ändern, sowas als frustrierend zu umschreiben wäre geschmeichelt.

Dannach kamen 5 Jahre Wordpress, was in dieser Hinsicht auch nicht anfängertauglicher war/ist, wenn man selber Änderungen in Thementemplates vornehmen wollte, aber man stieß in der Regel auf Templates deren Layout durch CSS Statements strukturiert wurden. Auf den ersten Blick nicht wirklich besser (hier trennt sich die Sicht des Web Designers extrem von der des unbedarften Anwenders), auf den Zweiten aber doch deutlich übersichtlicher und in der Praxis leichter den eigenen Bedürfnissen anpassbar. Selbst wenn man, wie ich, wenig Ahnung von HTML und CSS hat (einen Zustand den ich nicht wirklich ändern möchte, ich habe andere Interessen und Hobbys!). Mit ein bißchen Mut beim copy&paste, gelegentlichen Blicken in die Online zur Verfügung stehenden Dokumentationen zu HTML/CSS, allgemeines rumgoogeln und einer guten Backupstrategie, ließ sich da doch einiges bewirken.

Die Templates von Gert Ebersbach sahen ordentlich aus, waren sauber strukturiert und kommentiert, gut um einzusteigen. Das passende Template zu finden war einfach, es an meine Bedürfnisse anzupassen schon fast trivial (zumindest nach 5 Jahren mit WP Templates) und das wars dann auch für ca. 2 Jahre. Aber wie ich hier schon schrieb

> *Es war solide und zuverlässig, aber nach zwei Jahren Nutzung inzwischen in etwa so spannend wie ein Parkhaus nach Mitternacht*

und in einer wilden copy&paste Orgie, verbunden mit exzessiven Löschaktionen, adaptierte ich einen meiner alten Wordpress Favoriten und inzwischen bin ich ganz zufrieden mit gonzo-minimal.

Andererseits fiel mir auf, daß sich in einer Hinsicht zu "früher" nichts geändert hat, die zur Verfügung stehende Dokumentation um sich selbst ein Template zu erstellen ist a. veraltet und b. nicht wirklich Anfängertauglich. Das einzige deutschsprachige Tutorial zu dem Thema fand ich bei Lars Ellmauer (leider nicht mehr online), das eigentlich sehr gut anfängt und nachvollziehbar ist, zumindest bis zu dem Abschnitt, in dem dann ohne weitere Erklärungen die komplette stylesheet.css gezeigt wird. Bei mir (und nicht nur bei mir) war da einfach Schicht im Schacht. Es war für mich schlußendlich einfacher ein Wordpress Template für CMSimple umzuändern, als aus einem CMSimple Tutorial Template etwas für mich gefälliges zu drechseln.

Deswegen, wie schon im Editor´s Blog angedroht: Wir klöppeln uns unser eigenes Template und gehen weiter zu den Grundlagen.

Seitenanfang

## Grundlagen

Wir haben auf der einen Seite unsere Inhalte und auf der anderen Seite CMSimple und seine diversen Funktionen darauf zuzugreifen Unter diesem Aspekt sollten zuerst einmal die CMSimple Funktionen erwähnt werden, mit denen man Daten zugänglich macht. Hier mal eine Liste von Funktionen, auf die man in den meisten Templates stößt:

head(): fügt Metatags und andere Head-Informationen ein
onload(): fügt Javascript ein, das der Editor benötigt
sitename(): zeigt den Seitentitel an
toc(): zeigt das normale Navigationsmenu an
locator(): zeigt den Navigationspfad als Breadcrumb an (z.B. Home > News> CMSimple)
searchbox(): zeigt die Suchfunktion an
editmenu(): zeigt im angemeldeten Zustand die Editor-Werkzeugleiste an
content(): zeigt den Inhalt im Template an
submenu(): zeigt ein Untermenu an, falls entsprechende Seiten vorhanden sind

loginlink(): dieser Link ermöglicht es dem Administrator sich anzumelden
sitemaplink(): erzeugt den Link zur Sitemap
printlink(): erzeugt den Link zur Druckversion der Seite
lastupdate(): zeigt das Datum der letzen Änderung an
mailformlink(): erzeugt den Link zum E-Mail Formular, falls ein Emailadresse eingetragen wurde.

Auf Einige davon kann man verzichten, Andere sind existenziell. Desweiteren hat das typische CMSimple Template einen Kopfbereich, den man, so man es nicht besser weiß, einfach mal ungefragt übernehmen sollte:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<?php echo head();?>
</head>
<body <?php echo onload();?>>
```

und am Ende des Templates, sollte man auch nicht vergessen body und html zu schließen :

```
</body>
</html>
```

Rein Theoretisch könnte man nun darin die CMSimple Funktionen auflisten und hätte eine funktionierende Installation, die sogar als valides HTML 4.01 Transitional durchgeht:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<?php echo head();?>
</head>
<body <?php echo onload();?>><br>
<?php echo sitename();?><br>
<?php echo locator();?><br>
<?php echo searchbox();?><br>
<?php echo toc();?><br>
<?php echo sitemaplink();?><br>
<?php echo printlink();?><br>
<?php echo mailformlink();?><br>
<?php echo loginlink();?><br>
<?php echo lastupdate();?><br>
<?php echo editmenu();?><br>
<?php echo content();?><br>
<?php echo submenu();?><br>
</body>
</html>
```

Funktioniert tatsächlich, sieht aber ziemlich, äh, bescheiden aus, nichts was man ohne Not freiwillig benutzen würde. Aber das ist das Gerüst, das, wenn es erstmal mit (html/css) Leben gefüllt ist, eine richtige Webseite ergibt, die sowohl funktional als auch optisch unseren Vorstellungen entspricht. Ebenfalls wichtig, sie sollte von uns ohne größeren Aufwand modifizierbar sein, also "zukunftssicher" gestaltet werden.

Mit "wir" und "uns" meine ich alle, die sich hinsichtlich der Benutzung von HTML und CSS auf meinem oder unterhalb meines Kenntnisstand bewegen und metaphorisch betrachtet befinde ich mich auch nur irgendwo in der ersten Etage eines sehr, sehr hohen Hochhauses und ein "ungefragt übernehmen sollte" wird noch öfter zu lesen sein.

Wie kommen wir nun zu einem attraktiven, oder zumindest ohne Spontankrätze ansehbarem, funktionierenden html/css Rohbau, der sich mit dem CMSimple Bauplan verheiraten lässt? Viele Wege führen nach Rom:

a. bei Null anfangen. Nicht wirklich attraktiv, wenn man aus dem Umgang mit html/css keine Passion und sein nächstes Hobby machen möchte
b. ein existierendes CMSimple Template. Akzeptabel, aber größere nachträgliche Änderungen sind eher aufwändig.
c. der Griff zu einem der Zahllosen im Web zu findenden CSS Template Generatoren, die ein funktionierendes Grundgerüst zur Verfügung stellen. Großartig, absolut legitim für Laien, damit hat man in den meisten Fällen schonmal die halbe Miete.
d. Template Beispiele im Web analysieren und die Teile, die man soweit versteht um sie nutzen zu können, übernehmen.

Ich habe mich für eine Mischung von c. und d. entschieden, das Thema ist ja primär "Template für CMSimple ... für Laien" und nicht "wie werde ich zum CSS Guru".

Das geplante Template soll einen Kopfbereich haben, in dem der Seitentitel und/oder ein Logo und/oder ein Bannerbild zu sehen ist. Oder auch nur eines davon. Desweiteren soll es entweder eine oder zwei Sidebars haben, die entweder auf einer Seite oder auch auf beiden Seiten des Inhaltes positioniert sein können. Ein abschliessender Fußbereich gehört natürlich auch dazu. Da ich Templates mit festen Abmessungen nicht mag, soll es ein Template werden, das sich in Grenzen auf die Breite des Browsers einstellen kann. Ich werde auch einige CMSimple Funktionen weglassen (da ich sie nicht benötige) was aber kein Problem sein wird, weil, wenn wir fertig sind, es (hoffentlich) klar ist, wie man sie dort einfügen kann, wo man sie haben möchte.

Als Werkzeuge benötigen wir:

a. eine CMSimple Installation, für die Entwicklungsphase ist Portable_XH (windows) ganz nützlich, für die später notwendigen Validierungen sollte es dann schon eine Testinstallation auf der eigenen Domain sein. Die Validierung dient dabei weniger dem Zweck eines "Gütesiegels" für die eigene Seite, sie ist ein sehr hilfreiches Instrument zur Fehlersuche. Ein Großteil meiner Fehler waren bisher simple Tippfehler, ein vergessenes / oder > kann enorme Auswirkungen haben.
b. einen vernünftigen Texteditor, ich bevorzuge dabei Notepad++, der durch Syntax Highlighting and Syntax Folding das Editieren in html und css Dateien enorm erleichtern kann.

c. einen Template Generator, der mir das passende Grundgerüst zur Verfügung stellt. Das Angebot ist vielfältig, die Unterschiede können beträchtlich sein.

und damit geht es weiter zum Kapitel Inhalte suchen Struktur

Seitenanfang

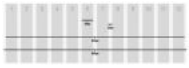
---

## Inhalte suchen Struktur

Unsere Inhalte werden durch die entsprechenden CMSimple Aufrufe repräsentiert, die Struktur bestimmt wo sie positioniert werden. In der Regel besteht eine Webseite aus einem Kopfbereich, dem Rumpf und zum Abschluß einen Fußbereich. Im Kopfbereich befindet sich der Name der Seite, manchmal auch ein Bild oder Logo.
Im darunterliegenden Rumpf ist der primäre Inhaltsbereich, meistens aufgeteilt in einen breiten Bereich für die Inhalte und Links oder Rechts ein schmalerer Bereich, in dem sich Menü und weitere Elemente wie Links, News oder andere kleine Textelemte befinden.
Abgeschlossen wird das alles durch den Fußbereich in dem Copyrighthinweise, bei CMSimple der entsprechende Link auf cmsimple.org oder cmsimple_xh (je nachdem welche Variante man benutzt) und weitere Links stehen können (bei mir z.B. zum Impressum und der loginlink).
Übliche Begriff dafür sind auch Header, Footer, Content, Sidebar, muß heuer ja alles Englisch sein ...

Um die Grundlegende Struktur zur Verfügung zu stellen habe ich mich schlußendlich für ein sog. Grid (Raster) System entschieden. Template Generatoren dafür gibt es genug (google), aber unabhängig davon sollte man das Konzept in seinen Grundzügen verstehen, damit man zumindest Andeutungsweise weiß, was man da eigentlich macht.
Ein Grid ist nichts anderes als ein festes Raster aus Zeilen und Spalten, in dem die einzelnen Elemente der Seite positioniert werden. Um ausreichende Flexibilität zu ermöglichen ist eine gewisse Mindestanzahl von Spalten unumgänglich, ein beliebter Ausgangswert sind 12 Spalten, wobei es aber auch genug Layoutgeneratoren gibt die deutlich mehr Spalten ermöglichen. Der bekannteste Vertreter der Gattung dürfte das 960 Grid System sein (das 960 steht für die Breite in Pixel) und einen ersten Überblick erhält man durch diese Bilder, die Grids mit 12, 16 und 20 Spalten zeigen

Die 12 hat den Vorteil, daß sie durch 2, 3 und 4 teilbar ist und mehr Spalten bieten jetzt nicht unbedingt mehr Möglichkeiten. Also sehen wir uns das 12er Grid mal genauer an.



12 Spalten a 60px, ein Zwischenraum von 20px, linker/rechter Rand von 10px und wir sind bei den mystischen 960px gelandet. Das funktioniert natürlich auch mit anderen Werten für die Anzahl der Zellen, ihrer Größe und den Abständen, ermöglicht dann andere Breiten für die Seite.
Die zu Grunde liegende Idee ist ganz einfach. Es gibt Zeilen und Zellen. Eine Zeile geht über die gesamte Breite und Zellen können 1 Spalte, 2 Spalten, 3 Spalten, ....., 11 Spalten, 12 Spalten groß sein. Der Rest ist kleines 1*1, man befüllt eine Zeile mit soviel Zellen, bis die 12 Spalten gefüllt sind. In der Praxis könnte das Skelett dann so aussehen:


Die oberste Zeile könnte sein: Logo (2), Sitename (7), Loginlink oder Suchfeld, oder Beides (3).
Zweite Zeile: bei mir z.B. als CoCo Bereich (12)
Dritte Zeile: Content (8), Sidebar mit Menü (4)
Vierte Zeile : 4x Newsboxen oder CoCo Bereiche
vergessen, die fünfte Zeile mit Copyrighthinweisen (12)

Mein generelles Problem dabei ist aber, es sind alles feste Werte und ich persönlich bevorzuge flexible Seiten, die sich innerhalb gewisser Grenzen der Breite des Browsers anpassen. Das macht es jetzt ein bißchen schwerer und die Lösung, das Umrechnen in Prozentwerte, hat nicht nur Vorteile. Und nun mal wieder etwas praktisches, das Basis CSS aus dem Grid System für unsere stylesheet.css

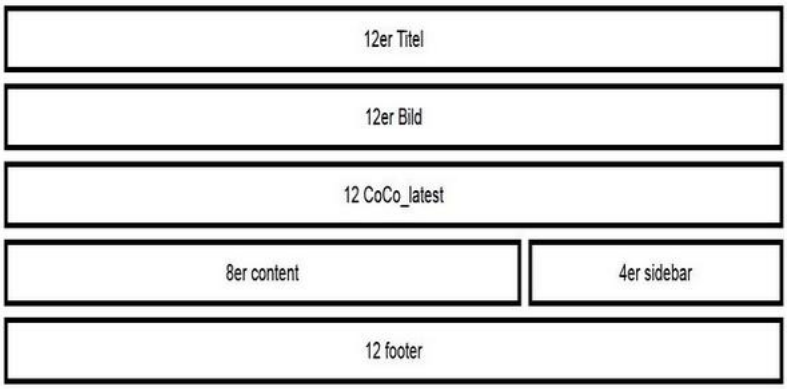
```
.grid_1 { width:6.25%; }
.grid_2 { width:14.583%; }
.grid_3 { width:22.917%; }
.grid_4 { width:31.25%; }
.grid_5 { width:41.667%; }
.grid_6 { width:47.917%; }
.grid_7 { width:56.25%; }
.grid_8 { width:64.583%; }
.grid_9 { width:72.917%; }
.grid_10 { width:81.25%; }
.grid_11 { width:89.583%; }
.grid_12 { width:97.917%; }
.column { margin: 0 1.04%; overflow: hidden; float: left; }
.row { min-width: 960px; max-width: 1280px; margin: 0 auto; overflow: hidden; }
```

.grid_1 bis grid_12 sind unsere Zellen mit der jeweils enthaltenen Anzahl von Spalten, dahinter steht die Breite in Prozent, .column gibt noch den Rand dazu und in .row wird die Zeile definiert. Die Mindestbreite wird in diesem Falle mit "min-width: 960px" festgelegt, die größte Breite mit "max-width: 1280p"; die Werte für min/max sind dabei ganz dem eigenen Geschmack/Bedürfnissen unterworfen.

Und dazu kommt nun noch unser Basis HTML für die template.htm, der Einfachheit halber bedienen wir uns dafür eines vorhandenen Templates, dem von gonzo-minimal. Das ist in seinem Aussehen und Struktur ziemlich klassisch; Header mit Bild, Content links, Sidebar mit Menü und Footer. Zusätzlich werden 2 CoCo Bereiche eingesetzt, dafür aber auf einige andere CMSimple Funktionen verzichtet.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<?php echo head();?>
</head>
<body <?php echo onload();?>>
<?php echo sitename();?>
<img src="./templates/gonzo-minimal/images/bfhoechst1980.jpg" alt="" width="100%" height="175">
<?php echo coco('coco_latest');?>
<?php echo editmenu();?>
<?php echo content();?>
<?php echo toc();?>
<?php echo coco('coco_sidebar');?>
<p><?php echo top();?> | 2001-2012 | © mein.name | <a href="?Startseite:Impressum">Impressum</a> | Powered by <a
href="http://www.cmsimple.org/">CMSimple 4</a> | Template by <a href="http://web57.ws/cms/">w.scharff</a> | <?php echo
loginlink();?></p>
</body>
</html>
```

Was jetzt kommt ist, wenn man das oben gezeigte Grid System ein bißchen verinnerlicht hat, fast schon erschreckend einfach.




Die template.htm sieht dannach folgendermaßen aus:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<?php echo head();?>
</head>
<body <?php echo onload();?>>

<!-- Header: Name der Seite -->
<div class="row">
    <div class="column grid_12">
        <?php echo sitename();?>
    </div>
</div>
<!-- Header: Bild über gesamte Breite -->
<div class="row">
    <div class="column grid_12">
        <img src="./templates/gonzo-minimal/images/bfhoechst1980.jpg" alt="" width="100%" height="175">
    </div>
</div>
<!-- CoCo Bereich für Leitartikel -->
<div class="row">
    <div class="column grid_12">
        <?php echo coco('coco_latest');?>
    </div>
</div>
<!-- Content/Editbereich und sidebar mit toc/coco_sidebar -->
<div class="row">
    <div class="column grid_8">
        <?php echo editmenu();?>
        <?php echo content();?>
    </div>
    <div class="column grid_4">
        <?php echo toc();?>
        <?php echo coco('coco_sidebar');?>
    </div>
</div>
<!-- footer -->
<div class="row">
```

```
    <div class="column grid_12">
         <?php echo top();?> | 2001-2012 | © mein.name | <a href="?Startseite:Impressum">Impressum</a> | Powered by <a
href="http://www.cmsimple.org/">CMSimple 4</a> | Template by <a href="http://web57.ws/cms/">w.scharff</a> | <?php echo
loginlink();?>
    </div>
</div>
</body>
</html>
```

Ganz wichtig und unbedingt zu Beachten: wenn man Plugins wie CoCo in einem Template aufruft, muß dieses Plugin auch installiert sein! Ist dies nicht der Fall funktioniert das Template nicht, man sieht nur eine weise Seite oder bekommt eine Fehlermeldung von Server.

Es ist ganz nützlich sich das Kommentieren im Template anzugewöhnen. Mit **<!--** und **-->** werden Kommentare umschlossen. Ebenfalls sehr hilfreich betrachte ich eine Formatierung, die durch Einrückung sichtbar macht, wo ein **div** beginnt und wo es endet. Ein wirklich beliebter Fehler ist es, bei tiefer verschachtelten divs, einfach mal eines zu vergessen, wobei es auch schon reicht ein **<**,/ oder **>** zu unterschlagen. Wer notepad++ benutzt hat es leichter.
Obiges Template in Verbindung mit dem Stylesheet sieht tatsächlich dem original gonzo-minimal schon ziemlich ähnlich, auf den ersten Blick. Auf den Zweiten fällt auf, daß es nur eine Schriftart gibt, die Links sehen wieder aus wie anno dunnemal im web 0.9 und ein paar Formatierungen sind weggefallen. Deswegen folgt noch ein Kapitel, wo wir uns ein bißchen mit dem Thema Schriften, Farben, allgemeines aufhübschen, sozusagen dem "ich hab die Haare schön" des Seitendesigns widmen.

Seitenanfang

---

## Aufhübschen (ich hab die Haare schön)

Wenn man dem Browser des Besuchers keine besonderen Anweisungen gibt, benutzt er zur Darstellung der besuchten Webseite vorgegebene Standards.  Das sieht nicht unbedingt so aus, wie man es gerne hätte. Also muß man die entsprechenden Anweisugen hinsichtlich Schriften, Farben, Formatierungen jeglicher Art selber in der stylesheet.css definieren.

Fangen wir mit ein paar grundlegenden Elementen und Voreinstellungen an, den Farben für Hintergrund und Schrift, die primäre Schrift, Zeilenhöhe und die Formatierung der H(Überschrift) Elemente.  Bei den Schriften kann man beliebig viele auflisten, der Browser nimmt die Erste die er anzeigen kann. In unterem Falle wäre das auch gleich "Times",  diese Schrift dürfte für die meisten Browser auf den meisten Betriebsystemen vorhanden sein. Die anderen Schriften kann man auch mal ausprobieren, die Größe in der die Schrift angezeigt wird, legt man dabei mit font-size:"wert" fest. Als "wert" kann man %, px oder em benutzen.

```
/* typography.css */
html {font-size:100.01%;}
body {font-size:90%;color:#222;background:#fff;font-family:Times, Arial, Helvetica, sans-serif;}
h1, h2, h3, h4, h5, h6 {font-weight:normal;color:#111;}
h1 {font-size:2em;line-height:1;}
h2 {font-size:1.5em;}
h3 {font-size:1em;line-height:1;}
h4 {font-size:1.5em;line-height:1.25;}
h5 {font-size:1.2em;font-weight:bold;}
h6 {font-size:1em;font-weight:bold;}
```

Es wird schon, aber die Links sehen noch nicht wirklich schön aus und bei den Newseinträgen hängt das Datum direkt an der Überschrift. Wie Links aufgehübscht werden steht in den ersten 4 Zeilen, die nächsten beiden Zeilen widmen sich der Darstellung von Listen (was auch auswirkungen auf die Links hat, die ja meist als Liste angezeigt werden) und der letzte Eintrag erscheint in der Format Combobox und dient dazu, dementsprechend markierten Text, nach rechts an den rand zu schieben.

```
a{text-decoration:none;color:#111111;font-weight:inherit;font-style:inherit}
a:focus{color:#c0c0c0}
a:hover{text-decoration:underline;color:#000080}
a:visited{color:#666}

li{line-height:1.4em}
ul,ol{margin-bottom:20px;list-style:none}

.news-date {float: right; font-size: 10pt; font-style: italic; font-weight: normal;}
```

jetzt sieht es schon deutlich manierlicher aus, fehlt aber immer noch einiges, z.B. die Quotes sind, äah, langweilig?

nächste Zeile für das css

```
blockquote{font-style:italic;border-left:5px solid;padding:1px 0 1px 12px;margin:16px 0 16px 16px;color:#777;font-
size:1.2em}
```

und damit Bilder, unabhängig von ihrer echten Größe, vom Template skaliert werden, kommt noch ein weiteres Statement hinzu:

```
.column img {max-width:100%;}
```

man sollte dann aber bei der Einbindung der Bilder die width und height Felder bei den Bildeigenschaften leer lassen!

Und damit wäre dieses kleine Tuorial, ohne jetzt tiefer in die Materie CSS einzutauchen, fertig. Das Template funktioniert und ist valide.

nochmal als copy und paste, die template.htm, für die eigene Nutzung nicht vergessen die Pfade an die eigenen Bedürfnisse anzupassen:

---

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
```

```
<head>
<?php echo head();?>
</head>
<body <?php echo onload();?>>

<!-- Header: Name der Seite -->
<div class="row">
    <div class="column grid_12">
           <h1><a href="http://www.web57.ws/cms" title="Startseite"><?php echo sitename();?></a></h1>
    </div>
</div>
<!-- Header: Bild über gesamte Breite -->
<div class="row">
    <div class="column grid_12">
        <img src="./templates/gonzo-minimal/images/bfhoechst1980.jpg" alt="" width="100%" height="175">
    </div>
</div>
<!-- CoCo Bereich für Leitartikel -->
<div class="row">
    <div class="column grid_12">
        <?php echo coco('coco_latest');?>
    </div>
</div>
<!-- Content/Editbereich und sidebar mit toc/coco_sidebar -->
<div class="row">
    <div class="column grid_9">
        <?php echo editmenu();?>
        <?php echo content();?>
    </div>
    <div class="column grid_3">
        <?php echo toc();?>
        <?php echo coco('coco_sidebar');?>
    </div>
</div>
<!-- footer -->
<div class="row">
    <div class="column grid_12">
        <?php echo top();?> | 2001-2012 | © mein.name | <a href="?Startseite:Impressum">Impressum</a> | Powered by <a
href="http://www.cmsimple.org/">CMSimple 4</a> | Template by <a href="http://web57.ws/cms/">w.scharff</a> | <?php
echo loginlink();?>
    </div>
</div>
</body>
</html>
```

---

und die aktuelle stylesheet.css:

---

```
 /* gonzo-grid */

/* typography.css */
html {font-size:100.01%;}
body {font-size:90%;color:#222;background:#fff;font-family:Times, Arial, Helvetica, sans-serif;}
h1, h2, h3, h4, h5, h6 {font-weight:normal;color:#111;}
h1 {font-size:2em;line-height:1;}
h2 {font-size:1.5em;}
h3 {font-size:1em;line-height:1;}
h4 {font-size:1.5em;line-height:1.25;}
h5 {font-size:1.2em;font-weight:bold;}
h6 {font-size:1em;font-weight:bold;}

a{text-decoration:none;color:#111111;font-weight:inherit;font-style:inherit}
a:focus{color:#c0c0c0}
a:hover{text-decoration:underline;color:#000080}
a:visited{color:#666}

li{line-height:1.4em}
ul,ol{margin-bottom:20px;list-style:none}

.news-date {float: right; font-size: 10pt; font-style: italic; font-weight: normal;}

blockquote{font-style:italic;border-left:5px solid;padding:1px 0 1px 12px;margin:16px 0 16px 16px;color:#777;font-
size:1.2em}

/* Raster, das grid */

.grid_1 { width:6.25%; }
.grid_2 { width:14.583%; }
.grid_3 { width:22.917%; }
.grid_4 { width:31.25%; }
.grid_5 { width:41.667%; }
```

```
.grid_6 { width:47.917%; }
.grid_7 { width:56.25%; }
.grid_8 { width:64.583%; }
.grid_9 { width:72.917%; }
.grid_10 { width:81.25%; }
.grid_11 { width:89.583%; }
.grid_12 { width:97.917%; }
.column { margin: 0 1.04%; overflow: hidden; float: left; }
.column img {max-width:100%;}
.row { min-width: 800px; max-width: 1200px; margin: 0 auto; overflow: hidden; }
```

---

Und ferig ist die Grundlage für weitere Experimente. Viel Raum zum aufhübschen gibt es bei den Schriftarten und ihrer Größe, die Art und Weise wie die Links angezeigt werden hat auch noch eine Menge Entwicklungspotential und natürlich, die Position der Elemente auf der Seite selber. Menü auf der linken Seite und Content Rechts? Kein Problem, einfach in der entsprechenden Zeile (.row) die grid Elemente tauschen:

```
<!-- Content/Editbereich und sidebar mit toc/coco_sidebar -->
<div class="row">
    <div class="column grid_3">
        <?php echo toc();?>
        <?php echo coco('coco_sidebar');?>
    </div>
    <div class="column grid_9">
        <?php echo editmenu();?>
        <?php echo content();?>
    </div>
</div>
```

und schon hat man die Sidebar auf der linken Seite.

Viel Spass beim Experimentieren!

Ein Template, das auf dieses Tutorial aufbaut, gibt es unter dem Namen gonzo-h im Downloadbereich. Da ich in meine downloadbaren Templates auch nachträglich noch Fehlerkorrekturen und Verbesserungen einpflege, sind aber mehr oder weniger subtile Unterschiede in den html/css Dateien zu den hier erarbeiteten möglich.

Das zugrundeliegende Grid System, das in seiner Stringenz und Übersichtlichkeit alle anderen, von mir ausprobierten, Grid Frameworks um Längen schlägt, ohne deswegen jetzt zu sehr in den Möglichkeiten eingeschränkt zu sein, ist :

Bedauerlicherweise wurde die Seite zum 1KB CSS Grid Anfang 2013 vom Netz genommen

Seitenanfang

---

## Verbesserungen und Korrekturen

Nix ist perfekt und auch wenn das bisher Erreichte nett aussieht und den Validator heil übersteht, bedeutet das noch lange nicht, daß aus der Sicht derer, die CMSimple entwickeln (das CMS, Plugins und auch Templates dafür), meine erarbeiteten template.htm/stylesheet.css sonderlich gut oder elegant ist und ich habe dankenswerterweise auch einiges an konstruktiver Kritik dazu erhalten. Hinweise und Tips, so sie leicht nachvollziehbar und nicht zu aufwändig sind, werde ich in diesem Kapitel anhängen.

Als Erstes wurde der Kopfbereich gesäubert, man kann es sogar noch einfacher/kürzer machen und das

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

durch ein `<!DOCTYPE html>` ersetzen. Die daraus resultierende Fehlermeldung des Validators führte dann sozusagen zwingend zum nächsten Schritt, die doch arg unelegante Variante das Headerbild direkt in das Template einzubinden durch eine Zeitgemäßere Variante zu ersetzen, nämlich sowas in der stylesheet.css zu handhaben. In der template.htm wird also diese Zeile entfernt:

```
<img src="./templates/gonzo-minimal/images/bfhoechst1980.jpg" alt="" width="100%" height="175">
```

und durch Folgendes ersetzt:

```
 <div id=banner>
 </div>
```

und in der stylesheet.css muß dann natürlich noch das banner bestimmt werden:

```
#banner { background-image: url("images/bfhoechst1980.jpg"); background-size: 100%; height: 175px; background-repeat:no-repeat;}
```

---

Tags: cmsimple cmsimple templates

Druckansicht

## Kommentare: 8

1
nora