reicht hier auch `id="navigation"` und wird mit hoher Wahrscheinlichkeit von jedem als Hauptnavigation verstanden). Auch `id="navi"` wäre ein vertretbare und wenig fehleranfällige Bezeichner.

Die horizontalen Blöcke, die im Layout ermittelt wurden, werden möglichst sprechend benannt. `id="kopf"` ist dabei der komplette Kopf, `id="kopf-oben"` dann der obere Block innerhalb des Kopfbereichs, `id="kopf-unten"` der untere. Entsprechend werden die `id`- und `class`-Attribute für das ganze HTML vergeben.

3.3 CSS-Grundlayout aufsetzen

Dieses HTML-Grundgerüst ist ausreichend, um damit alle Seiten aufzubauen. Jetzt kann mit der eigentlichen visuellen Umsetzung begonnen werden.

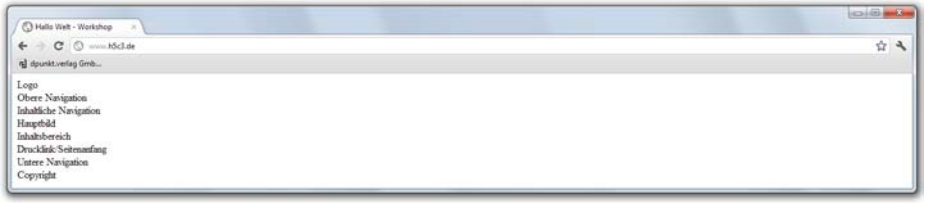


Abb. 3–4
Darstellung des HTML-Grundgerüsts ohne jegliches CSS-Design

In Abb. 3–4 werden die Platzhaltertexte in der Browseransicht sichtbar, die im Grundgerüst eingesetzt wurden. Allerdings erscheinen alle Texte auf weißem Grund. Obwohl diverse HTML-Tags bereits im Quelltext vorhanden sind, können Sie nicht erkennen, wie breit, wie hoch und mit welchen Abständen die einzelnen HTML-Elemente zueinander angezeigt werden. Das »natürliche Verhalten« des HTML wird noch nicht deutlich, oder anders gesagt: Es wird noch nicht klar, wie sich die HTML-Elemente automatisch im Browser anordnen.

Um Licht ins Dunkel zu bringen, werden ein paar CSS-Anweisungen definiert, die ausschließlich den Sinn und Zweck haben, die HTML-Elemente »sichtbar« zu machen, damit Sie erkennen können, welche Auswirkungen weitere CSS-Anweisungen haben. Diese CSS-Regeln können als Debug-Regeln bezeichnet werden, also solche, die ausschließlich zu Entwicklungszwecken definiert werden. Sie sind temporär und werden wieder gelöscht, sobald sie nicht mehr benötigt werden.

Es sind CSS-Eigenschaften notwendig, welche die Dimensionen der HTML-Elemente erkennbar machen, beispielsweise eine Hintergrundfarbe (`background-color`) oder ein Rand (`outline`).

Die Theorie zum Aufbau von CSS-Regeln finden Sie in Kapitel 31.

```
div {
    outline: 2px solid blue;
}
nav {
    outline: 2px solid red;
}
```

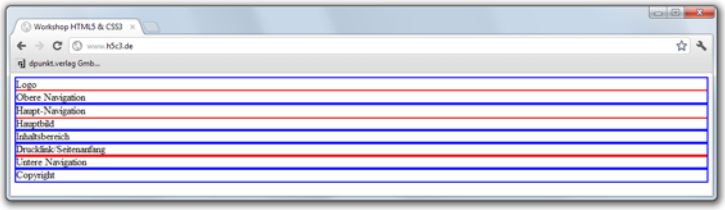


Abb. 3–5
Linien zeigen, welche Dimensionen die HTML-Elemente einnehmen.

Im CSS-Code sind die Namen der HTML-Elemente verwendet, von daher wird diese Schreibweise als **Elementselektor** bezeichnet. Als Selektor wird generell das bezeichnet, was vor den geschweiften Klammern steht. Darüber werden nun alle vorkommenden Elemente dieser Typen (hier `div` und `nav`) angesprochen und beeinflussen damit deren Darstellung. Die erste CSS-Regel (die ersten drei Codezeilen) links neben Abb. 3–5 sorgt also dafür, dass **alle** `div`-Elemente auf der Seite mit einem blauen Umriss versehen werden.

Grund-lagen

Mehr zu Elementselektoren finden Sie in Kapitel 36.1.2.

Im weiteren Verlauf des Workshops werden Sie weitere Selektoren kennen lernen, in Kapitel 36 finden Sie Details zu den wichtigsten Selektoren sowie eine Übersicht aller Selektoren.

WWW

Alle Selektoren im Überblick einschließlich Browsersupport: www.h5c3.de/link-3-1

3.3.1 CSS-Struktur vorbereiten

Bevor es an die ersten »richtigen« Styles geht, wird die CSS-Datei vorbereitet. Im Gegensatz zum HTML, wo sich die Reihenfolge der einzelnen HTML-Elemente aus dem logischen Aufbau der Dokumentstruktur ergibt, entsteht in CSS keine automatische Reihenfolge. CSS-Regeln können im Grunde quer durcheinandergeschrieben und es muss keine bestimmte Reihenfolge in der CSS-Datei eingehalten werden, damit eine Regel wirken kann. Um sich jedoch langfristig im CSS-Code zurechtzufinden, hilft eine Grundstruktur, die über CSS-Kommentare angelegt wird.

Grund-lagen

Syntax CSS-Kommentare: Kapitel 32

```
/*
Übergreifende Formate
*/

/*
Kopf
*/

/*
Navigation
*/

/*
Inhalt
*/

/*
Fuß
*/
```

```
/**************************
* Übergreifende Formate
**************************/

/**************************
* Kopf
**************************/

/**************************
* Navigation
**************************/

/**************************
* Inhalt
**************************/

/**************************
* Fuß
**************************/
```

Der linke CSS-Quelltext zeigt dabei den minimalen Aufbau eines CSS-Kommentars, er wird von /* und */ umschlossen. Dazwischen können alle Zeichen eingefügt werden. Der Quelltext rechts zeigt, wie zusätzliche Zeichen die Struktur optisch noch deutlicher hervorheben.

Im Workshop wird diese Struktur wiederkehrend aufgegriffen, um das CSS nicht irgendwie, sondern organisiert aufzusetzen.

3.3.2 Seitenbreite

Die Seitenbreite ist der erste Schritt im CSS-Design. Hier geht es im Wesentlichen darum, wie sich die »Seite an sich« im Browserfenster verhält. Es gibt zwei Ansätze, wie mit der Breite von Webseiten umgegangen wird.

1. Seitenbreite ist 100 % der Fensterbreite. Damit nimmt die Seite immer die vollständige Breite des Fensters ein. Gerade bei großen Monitoren wird damit der Platz auf dem Bildschirm optimal genutzt. Beispiele sind Amazon oder die Bildersuche von Google. Allerdings sind die Proportionen der Seite variabel und stets abhängig von der Fenstergröße.
2. Seiten bekommen eine feste Breite. Damit sind die Proportionen der Webseite unabhängig vom Browserfenster gleichbleibend, der Gesamteindruck der Seite ist auf unterschiedlichen Monitoren einheitlich.

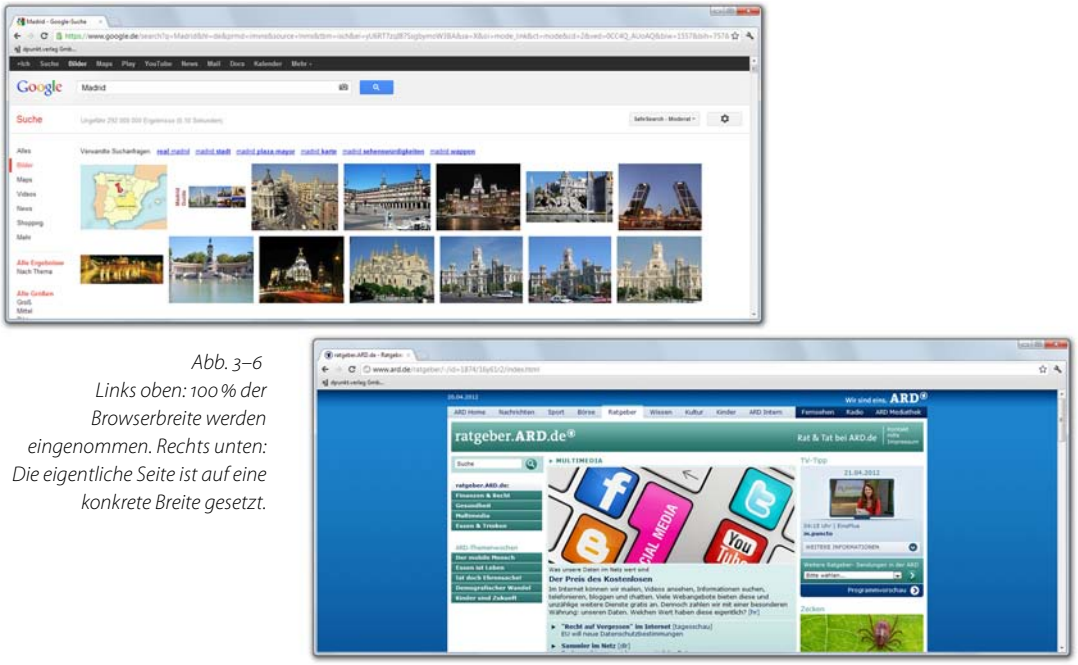


Abb. 3–6
Links oben: 100 % der Browserbreite werden eingenommen. Rechts unten: Die eigentliche Seite ist auf eine konkrete Breite gesetzt.

Im Fall des Beispiellayouts soll die Breite, mit der das Layout gestaltet wurde, auch so erhalten bleiben und sich nicht nach der Fenstergröße richten. Idealerweise bekommen Sie die Information, wie sich die Seite im Browser verhalten soll, mit dem Layout.

Darüber hinaus liefern Grafiker das Seitenraster und damit die Seitenbreite auch schon mal als separate Information, sodass Werte direkt ins CSS übernommen werden können. Stehen diese Informationen nicht zur Verfügung, müssen Sie das Layout selbst ausmessen.

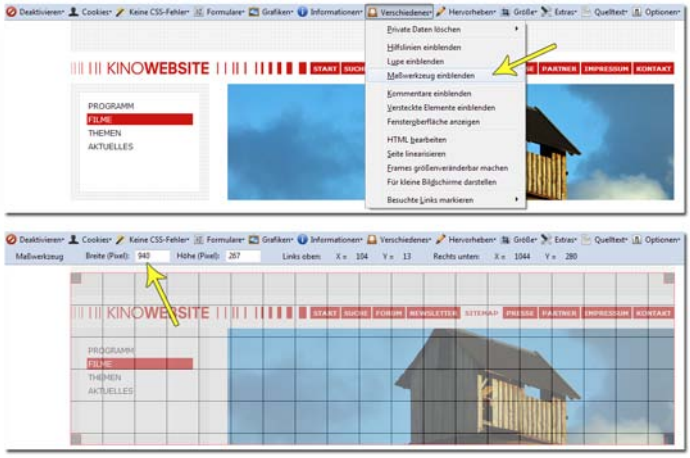
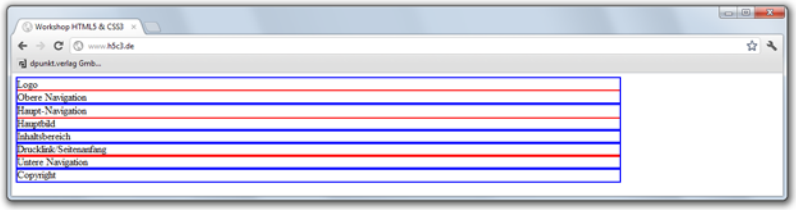


Abb. 3–7
Die Seitenbreite kann über die Toolbar »Web Developer« ausgemessen werden.

Sie können die Breite des Layouts über ein Grafikprogramm ermitteln. Wie Sie dabei vorgehen müssen, hängt von der konkreten Software als auch von Ihrer Erfahrung damit ab.

Alternativ ist das Ausmessen mit Browserwerkzeugen möglich. Da das Layout als JPG-Datei vorliegt, kann dieses mit dem Browser geöffnet werden (am schnellsten per »Drag & Drop«). Abb. 3–7 zeigt das Werkzeug »Web Developer« (dieses Werkzeug wird in Kapitel 20.2.1 vorgestellt), das für Firefox und Google Chrome zur Verfügung steht. Unter »Verschiedenes« können Sie »Maßwerkzeug einblenden« auswählen, woraufhin ein Linienraster erscheint. Über die Greifer an allen vier Ecken kann die Größe des Messwerkzeugs verändert und die Fläche des Layouts überspannt werden. Die ausgewählte Breite wird in der Toolbarleiste angezeigt.




```
body {
    width: 940px;
}
```

Das Layout ist mit einer Breite von `940px` aufgesetzt. Über die CSS-Eigenschaft `width` wird diese definiert, der Effekt ist in Abb. 3–8 zu sehen.

Ohne eine eigene Breitendeklaration nehmen Block-Elemente 100 % der verfügbaren Breite ein. Aktuell sind ausschließlich Block-Elemente im Einsatz. In Abb. 3–5 ist das automatische Verhalten der HTML-Elemente zu erkennen, die ohne weiteres Dazutun die Breite des `body`-Elements einnehmen. Dabei handelt es sich um die Referenzgröße, die oben als »verfügbare Breite« bezeichnet ist.

Die Seite hat bereits jetzt ihre endgültige Breite, wird aber linksbündig angezeigt. Es ist Geschmacksache, ob man dies so belässt oder die Seite »zentriert«. Die westliche Leseweise von links nach rechts spricht dafür, Seiten linksbündig anzuzeigen. Diese Entscheidung würde ich vom konkreten Webangebot anhängig machen.

Dass die Seite mit der definierten Breite linksbündig angezeigt wird, hat folgenden Grund: Die meisten HTML-Elemente weisen einen Außenabstand `margin` von `0px auf`, `body` hat einen Standardwert von `8px` in alle vier Richtungen. Allerdings kann der `body`, wenn er 940 px in der Breite misst, das Fenster hingegen breiter ausfällt, 8 px nach links **und** nach rechts nicht einhalten. Falls Werte im Konflikt stehen, »überbietet« der linke Abstand `margin-left` den rechten Abstand `margin-`

Work-shop

Weitere Entwicklungswerkzeuge sind in Kapitel 20 beschrieben.

Abb. 3–8
Im ersten Schritt wird die Seitenbreite festgelegt.

Grund-lagen

Grundlagen zu Block-Elementen finden Sie in Kapitel 38.3.

Work-shop

*Mehr zu **`margin`** (Außenabstand) in Kapitel 5.1.*

`right` (und ebenfalls dominiert ein oberer Abstand `margin-top` einen unteren `margin-bottom`).

```
body {
    width: 940px;
    margin-left: auto;
}
```

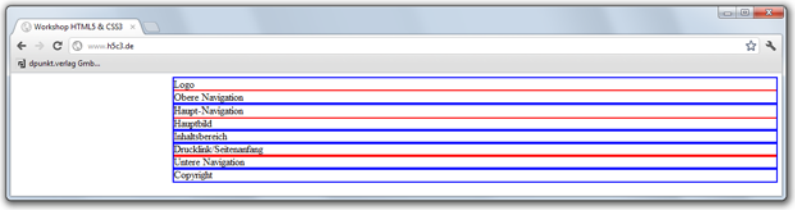


Abb. 3–9
Um die Seite zu zentrieren, muss der Standardwert ***`margin-left:`*** `0px` *»neutralisiert« werden.*

In Abb. 3–9 wurde die CSS-Regel für den `body` um ein `margin-left: auto` ergänzt. Dies löst zwar die Linksbündigkeit auf, führt allerdings dazu, dass der Standardwert von `margin-right` von `8px` greift – die Seite wird nun rechtsbündig angezeigt.

```
body {
    width: 940px;
    margin-left: auto;
    margin-right: auto;
}
```

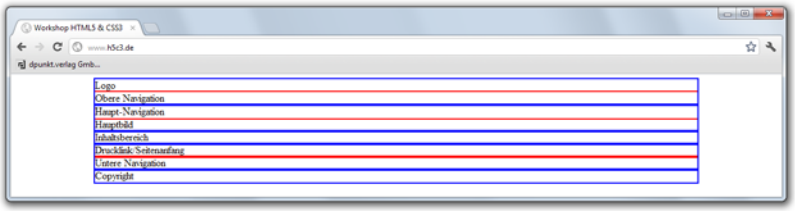


Abb. 3–10
`margin-right:` *muss ebenfalls auf* `auto` *gesetzt werden, damit die Seite mittig angezeigt wird.*

In Abb. 3–10 wird sichtbar, dass die Seite nun zentriert angezeigt wird, sobald `margin-left` und `margin-right` auf `auto` gesetzt sind.

```
body {
    width: 940px;
    margin: auto;
}
```

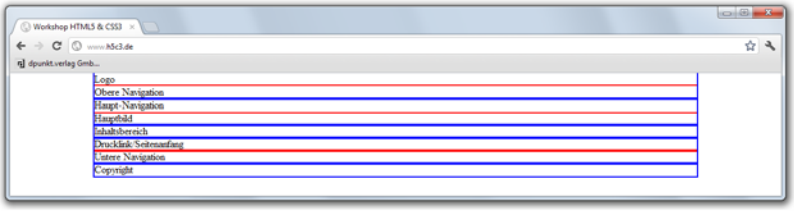


Abb. 3–11
`margin` *ersetzt die Werte für links und rechts, aber auch für oben und unten.*

Über sogenannte Kurzschreibweisen können einzelne CSS-Eigenschaften kombiniert werden. In Abb. 3–11 ersetzt `margin: auto` die beiden einzelnen Anweisungen, aber auch `margin-top` und `margin-bottom` werden über diese Kurzschreibweise auf `auto` gesetzt. In der Darstellung ist zu sehen, dass die Seite jetzt nicht nur zentriert ist, sondern auch am oberen Rand des Browsers sitzt.

Grundlegendes zu Kurzschreibweisen finden Sie in Kapitel 31.2.

Dieser Nebeneffekt der Umstellung auf die Kurzschreibweise ist in diesem Fall gewünscht. `body` ist mit einen Standardwert für den oberen Abstand von `margin-top`: `8px` versehen; dies wird in den seltensten Fällen jedoch so belassen, da so ein Lücke am oberen Rand zur Webseite entsteht.

Bezüglich der Höhe werden zum jetzigen Zeitpunkt noch keine CSS-Anweisungen definiert. Zum großen Teil ergibt sich diese durch den Platzbedarf der darin zu integrierenden Elemente, die Stück für Stück in das Beispiellayout eingebaut werden. Konkrete Abstände zwischen einzelnen Blöcken in vertikaler Richtung lassen sich dann im Nachgang noch dem Layout anpassen.

Kapitel 3 Ergebnispaket: www.h5c3.de/listing-3-3