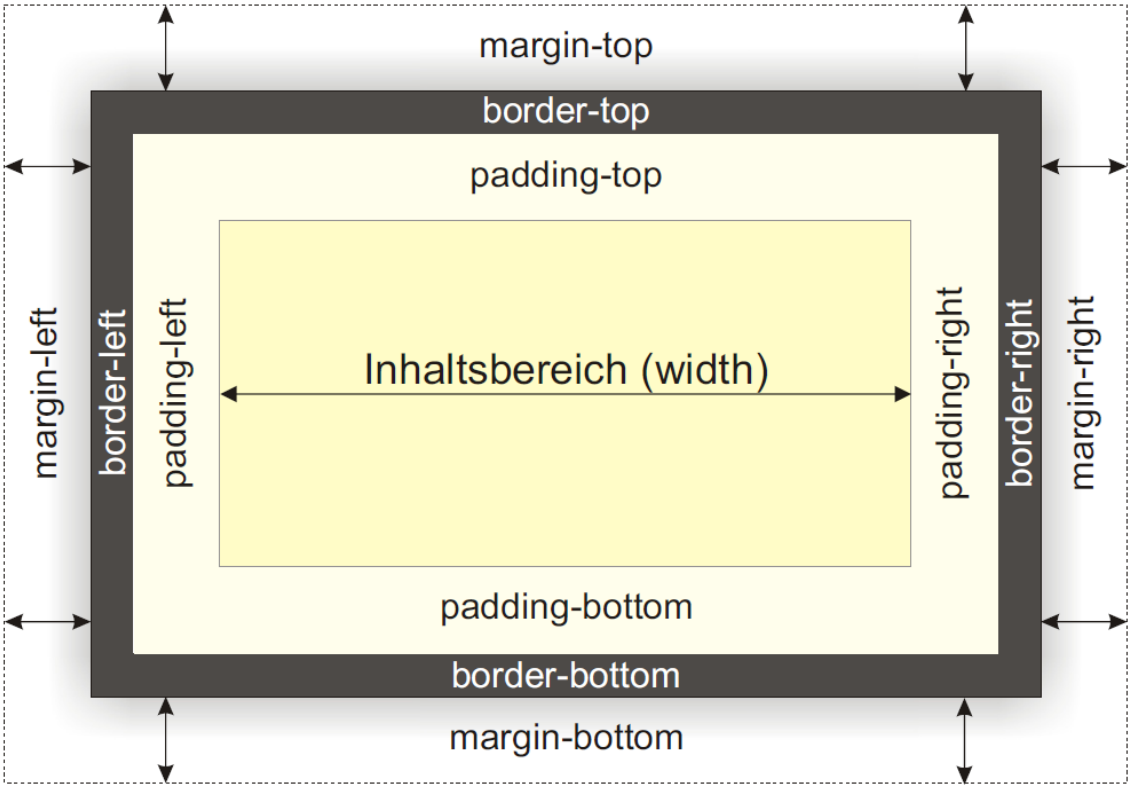
Inhalt:

1)Das Box-Modell

Die Box (auch „Kiste“, „Kästchen“, „Kontainer“) ist der Baustein einer Website und das grundlegende Gestaltungselement bei CSS. **Alle Boxes sind nach diesem Schema aufgebaut!** Es beschreibt einen rechteckigen Bereich, der für alle Elemente des Dokumentenbaums generiert wird.



Ein solche Box besteht immer aus vier Teilen:

- dem eigentlichen Inhaltsbereich,
- dem Innenabstand (**padding**),
- einem definierten Rahmen (**border**) sowie
- dem Außenabstand (**margin**).

❖ **Innenabstand (padding)** beschreibt den Abstand des Inhaltsbereiches vom Rahmen der Box. Übernimmt die Hintergrundfarbe des Inhaltsbereiches

- **Rahmen (boder)** schließt den sichtbaren Bereich des Box-Modells nach außen hin ab. Kann eine eigene Farbe, einen eigenen Stil und eine eigene Dicke erhalten
- **Außenabstand (margin)** ist generell transparent und beschreibt den Abstand der Box zum nächsten Element. Er übernimmt die Hintergrundfarbe des umgebenden Elements, auch „Elternelement“ genannt und kann keine eigene Farbe definiert bekommen. Die Außenabstände zweier Boxen innerhalb einerr Gliederungsebene addieren sich dabei nicht. Sie greifen übereinander, wodurch der größere Wert für den Abstand zwischen den beiden Boxen maßgebend wird.

Inhaltsbereich:
Breite und Höhe des Inhaltsbereichs richten sich entweder nach der Größe der Inhalte des Elements (z.B. eines Bildes oder einer Tabelle) oder sie können über die CSS Eigenschaften **width** und **height** explizit vorgegeben werden.

Zur **Gesamtbreite** einer Box kommen die Innenabstände (**padding**) und die Rahmenbreite (**border-width**) hinzu. Die Innen- und Außenabstände sowie die Rahmenbreite und Rahmenhöhe können per CSS auf null gesetzt werden. Die Gesamtbreite errechnet sich somit aus der Addition aller sichtbaren Bestandteile (Inhaltsbereich, Innenabstand, Rahmen) des Box-Modells.

Beispiel:

Erstelle eine „box.html“ und „box.css“
Erstelle eine Box mit einem Inhaltsbereich von 300 Pixel, einen umlaufenden Innenabstand von 10 Pixel sowie einen ebenfalls umlaufenden und 10 Pixel breiten, dunkelgrauen Rahmen. Den Außenabstand setze auf null.

box.css × box.html ×

Code | Teilen | Entwurf | Live-Ans

```
1  /* erste CSS-Box */
2
3  .box {
4      background:#CCC;
5      width:300px;
6      padding:10px;
7      margin:20px;
8      border:10px #888 solid;
9  }
```

Man kann Klassen auch selbst definieren. Sie müssen immer mit einem Punkt beginnen, danach folgt der Name der Klasse.

```
<style type="text/css">
.codeschrift
 {
}
</style>
```

box.html Code für die Box – mit „div class“

```
<!doctype html>
<html>
<head> <meta charset="utf-8">
<title>Box</title>
<link rel="stylesheet" href="box.css" /> </head>
<body>
<div class="box">Diese Box hat eine Gesamtbreite von 340 Pixel</div>
</body> </html>
```

*box.html

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Box</title>
<link rel="stylesheet" href="box.css" />
</head>
<body>
    <div class="box">Diese Box hat eine Gesamtbreite von 340 Pixel</div>
</body>
</html>
```

Die Box zentrieren

Um die Box zentriert darstellen zu können, kann man **nicht** auf den bekannten HTML-Code ***<text align="center">*** zurückgreifen.
Es genügen die beiden CSS-Eigenschaften „width“ und „margin“:
Beispiel:

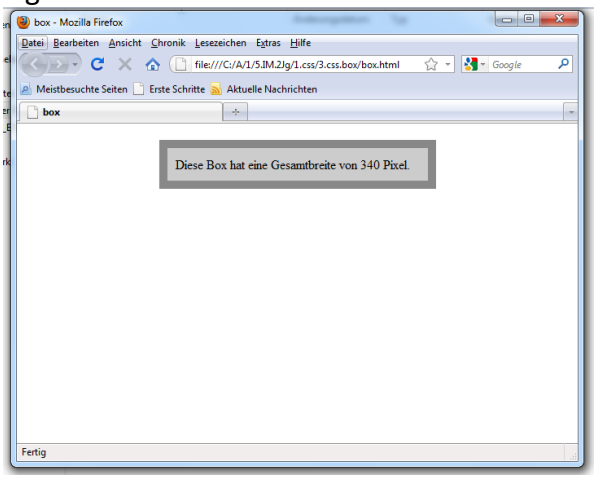
width: 720px;
margin: 20px auto;

Auto bedeutet, dass automatisch von beiden Seiten der gleiche Abstand gewählt wird.

Im Angabenbeispiel füge die Eigenschaft auto bei margin hinzu:

```
.box {
    background:#CCC;
    width:300px;
    padding:10px;
    margin:20px auto;
    border:10px #888 solid;
}
```

Ergebnis im Browser:



Angaben verkürzt darstellen:

Die vier Seiten der Box können einzeln deklariert werden, z.B.:

```
padding-top: 10px;
padding-right: 20px;
padding-bottom: 0;
padding-left: 20px;
```

Einfacher ist es aber diese in einem darzustellen: (im Uhrzeigersinn)

```
padding: 10px 20px 0 20px;
```

Weiter Vereinfachung:

- Hat ein Style nur **2 Werte**, dann gilt dies immer für oben und unten, bzw. rechts und links. Diese Regelmäßigkeit nutzt man für folgende Darstellung:

```
padding: 5px 20px;
```

= padding: 5px 20px 5px 20px;

- Wird nur **ein Wert** angeben, dann sind damit alle vier Seiten gemeint und erhalten diesen Wert!

Übung:

Erstelle im obigen Beispiel ein padding von 30px für alle 4 Seiten.

2)Tipp: Die Seite zentrieren mit „margin: auto“

Text und Inline-Elemente kann man mit der Anweisung text-align: center zentrieren, aber für Blockelemente funktioniert das leider nicht. Da es keinen Befehl wie »Zentriere ein Blockelement auf der Seite« gibt, benutzt man einen kleinen Trick:

- Wenn die Außenabstände (margin) für links und rechts auf *automatisch* gesetzt werden, sind sie immer gleich groß.
- Wenn die Außenabstände links und rechts gleich groß sind, ist das Element zentriert.

3)CSS-Eigenschaft float

Man kennt diese Art der Positionierung aus der Zeitung oder von Textverarbeitungs-programmen:
Ein Bild wird innerhalb eines Textbereichs am linken oder rechten Seitenrand platziert, und die Wörter umfließen das Bild. **Nachfolgende Elemente fließen einfach an ihnen vorbei.**
Als Wert der Eigenschaft float kann man

- left oder right angeben, je nachdem, auf welcher Seite des Elternelements das *floatende* oder *schwebende* Element platziert werden soll.

Beispiel:
```
float: left;
```

Das Verhalten von Elementen mit der Eigenschaft float ist etwas gewöhnungsbedürftig, denn ein floatendes Element beeinflusst auch die umgebenden Inhalte:

Beispiel:
Erstelle eine neue Datei „float.html“ und eine „float.css“:
Die Box ***#floating_box*** befindet sich im HTML-Code innerhalb des statischen Containers. Die Box ***#container*** hat eine flexible Breite, während die floating Box eine Breite von 200 Pixel hat.

Beachte im HTML-Code:

*** ***
*** … ***
Hier kommt der Text des Containers und nicht vor dem div id=”floating_box”…..
*** ***

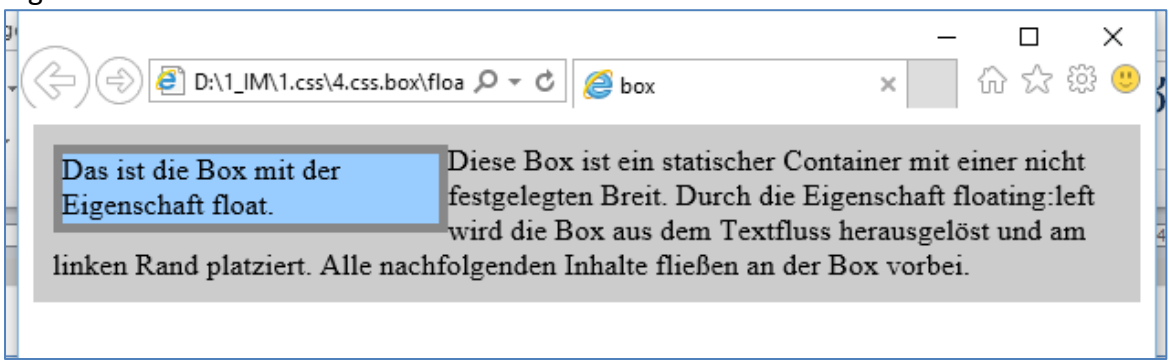
Text zum Kopieren:

“Diese Box ist ein statischer Container mit einer nicht festgelegten Breite. Durch die Eigenschaft floating:left wird die Box aus dem Textfluss herausgelöst und am linken Rand platziert. Alle nachfolgenden Inhalte fließen an der Box vorbei.”

Code:

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>box</title>
<link rel="stylesheet" href="float.css" type="text/css">
</head>

<body>
<div id="container">
    <div id="floating_box">Das ist die Box mit der Eigenschaft float.</div>
    Diese Box ist ein statischer Container mit einer nicht festgelegten Breit.
    Durch die Eigenschaft floating:left wird die Box aus dem Textfluss herausgelöst
    und am linken Rand platziert. Alle nachfolgenden Inhalte fließen an der Box vorbei.
    </div>
</body>
</html>
```

Ergebnis: wenn man den Bildschirm etwas verkleinert



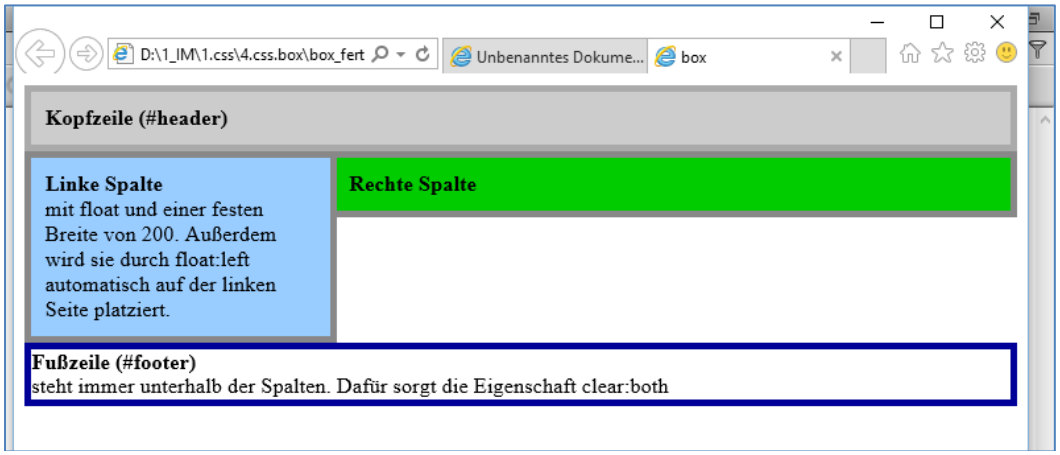
float.css:

```
/* erste CSS-Box */

#container {
    background:#CCC;
    width: auto;
    padding:10px;
    margin:0px;
    border:0px #aaa solid;
}

#floating_box {
    float:left;
    width: 200px;
    background:#9CF;
    border:5px #888 solid;
}
```

4)Übung: Zwei Spalten mit Footer

Durch die Kombination von floatenden und statischen Containern lässt sich auf sehr einfache Weise ein flexibles Layout erstellen.

Ziel: folgendes Layout



Erstelle die CSS-Datei „box_fertig.css“:
Die Positionierung der einzelnen Container erfolgt in folgender Weise mit CSS:

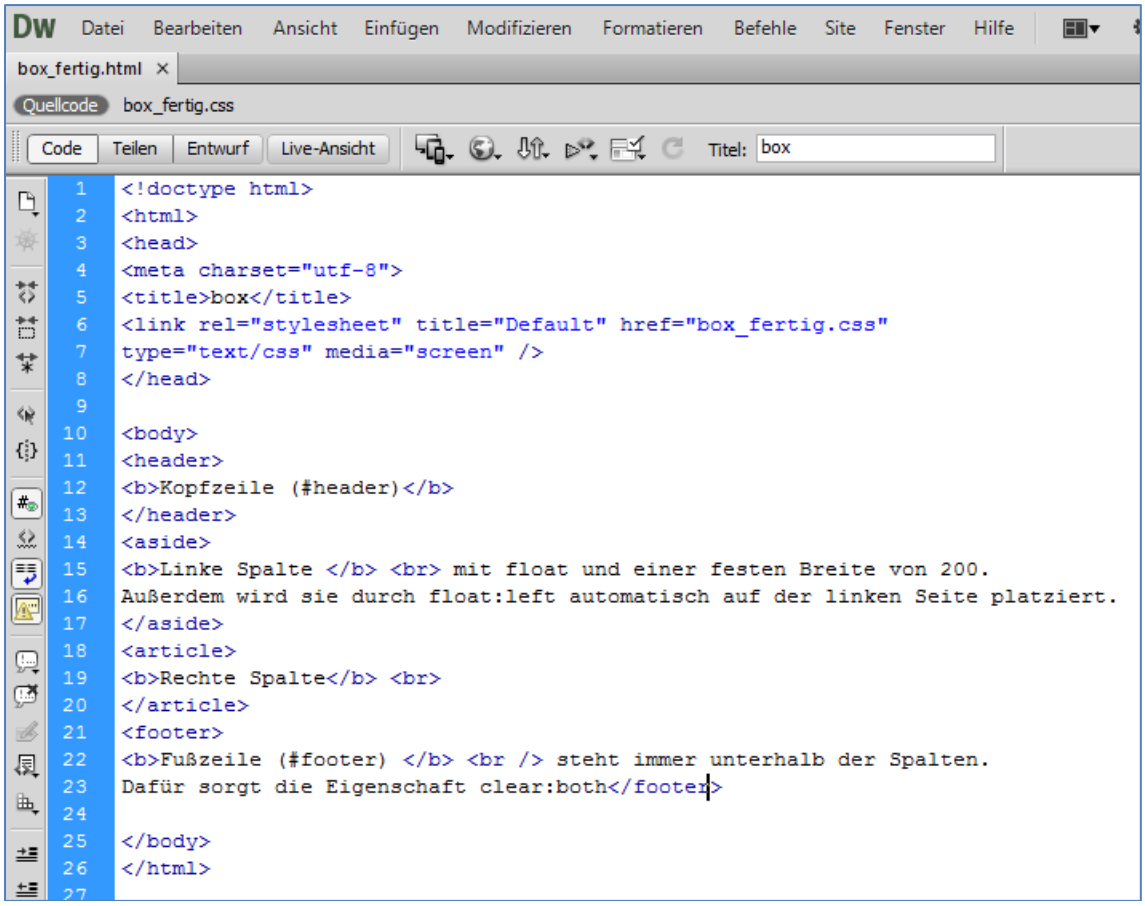
box_fertig.html

Quellcode box_fertig.css

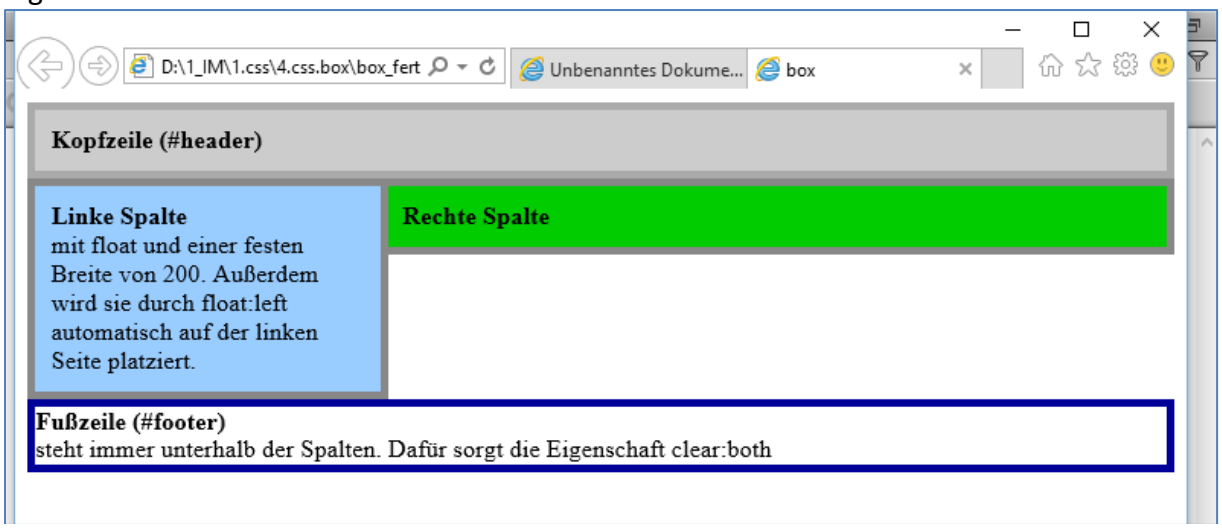
Code | Teilen | Entwurf | Live-Ansicht

```
/* 2 Spalten mit Footer*/

header {
    background:#CCC;
    width: auto;
    padding:10px;
    margin:0px;
    border:5px #aaa solid;
}

aside {
    float:left;
    width: 200px;
    padding:10px;
    background:#9CF;
    border:5px #888 solid;
}

article {
    width: auto;
    padding:10px;
    margin:0 0 0 224px;
    background:#0C0;
    border:5px #888 solid;
}

footer {
    clear:both;
    background:#999 solid;
    border:5px #009 solid;
}
```

Erstelle danach die HTML-Datei „box_fertig.html“ mit „semantischem HTML5“:



Ergebnis:



Den **Kopfbereich** **<header>** bildet ein normaler statischer Container. Die Eigenschaft ***width:auto*** braucht normalerweise nicht expizit angegebene zu werden, da auto sowieso der Standardwert ist.
Die **linke Spalte** <***aside***> erzeugt man durch die Eigenschaft ***float:left***. Sie erhält eine Breite von 200 Pixel und einen Innenabstand von 10 Pixel, sowie einen Rahmen.
Die **rechte Spalte** <***article***> würde durch die Eigenschaft ***width:auto*** die volle Seitenbreite einnehmen und damit teilweise hinter der linken Spalte verschwinden. Um dies zu

vermeiden, erhält <***article>*** einen linken Außenabstnd (***margin-left: 224px***) was exakt der Breite der linken Spalte entspricht.
Der Container ***<footer>*** ist eine statische Box und wird damit automatisch unterhalb von ***<article>***, dem im Quelltext voranstehenden Container, platziert. Es muss nur noch dafür gesorgt werden, dass auch die linke Spalte in die Positionierung mit einbezogen wird. Hier greift der große Vorteil von ***floats***, denn durch die Eigenschaft ***clear:both*** wird ***<footer***> automatisch unter die floatenden Spalten verschoben.