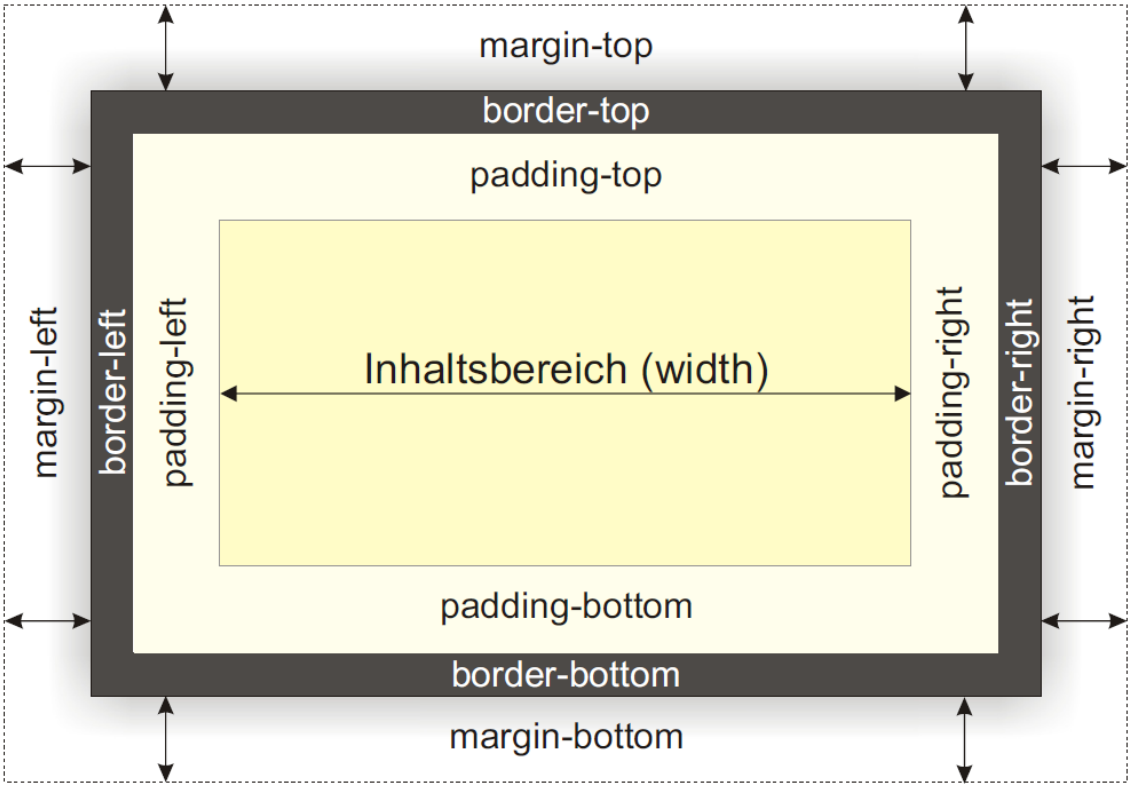
CSS – 2. Teil

Inhalt

1) Das Box-Modell
2) CSS-Eigenschaft float
3) Display: Block-Elemente und Inline-Elemente
4) Border, hover
5) Seite zentrieren mit „margin: auto“
6) Hyperlinks gestalten

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.



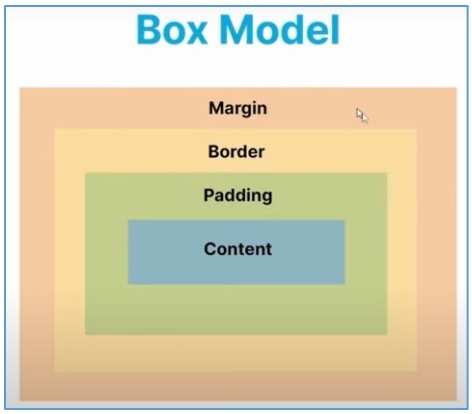
Ein solche Box besteht immer aus vier Teilen:

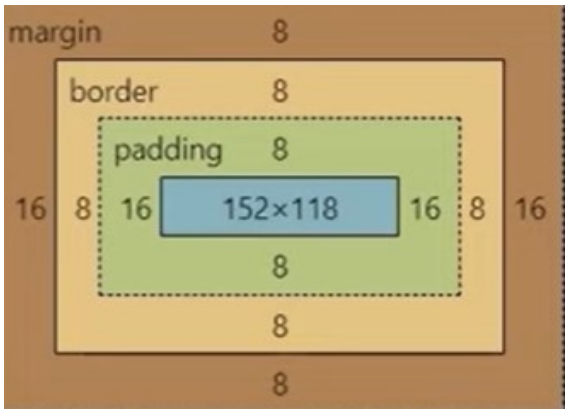
- dem eigentlichen Inhaltsbereich,
- dem Innenabstand (**padding**),
- einem definierten Rahmen (**border**) sowie
- dem Außenabstand (**margin**).

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  }
```

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:

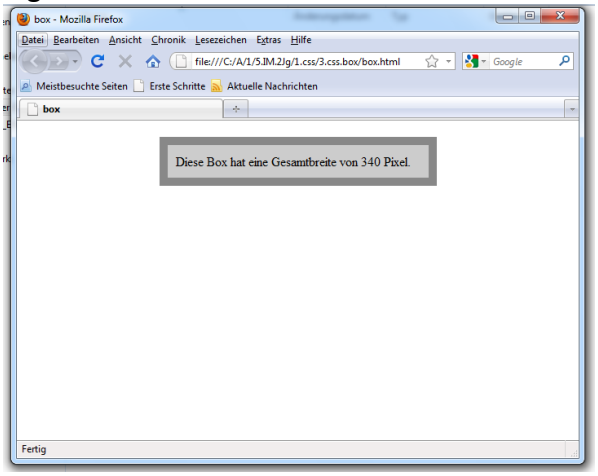
width: 720px;
margin: 20px auto;

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)CSS-Eigenschaft float

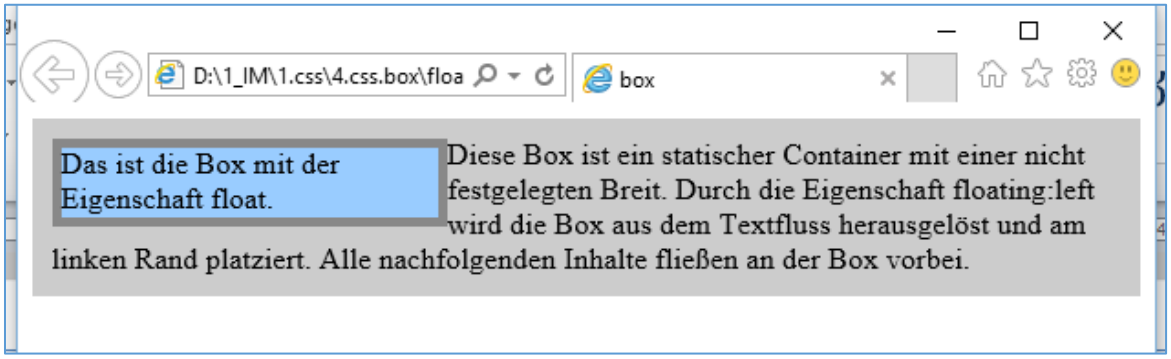
Man kennt diese Art der Positionierung aus der Zeitung oder von Textverarbeitungs-programmen:
Ein Bild wird innerhalb eines Textbereichs am linken oder rechten Seitenrand platziert, und die Wörter umfließen das Bild. **Nachfolgende Elemente fließen einfach an ihnen vorbei.**
Als Wert der Eigenschaft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

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

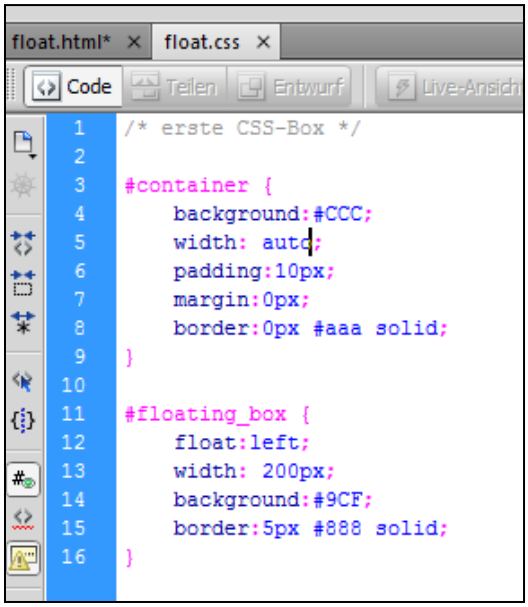
Beispiel:
float: left;

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

Ergebnis: wenn man den Bildschirm etwas verkleinert



float.css:



3)Block-Elemente/Inline-Elemente

Block-Elemente wie <p> oder erzeugen um sich herum

- **automatisch einen Rahmen**. Grundsätzlich sind alle Elemente
- rechteckig. Wenn man den Rahmen sehen würde, sehen diese Elemente wie Kisten aus. Dank dieser Tatsache lässt sich die Gestaltung auf drei Ebenen aufteilen:
  1. Anordnung der Kisten: Wo soll man diese auf der Website platzieren? Wie sieht der Aufbau der Seite aus?
  2. Gestaltung der Kisten: Welche Rahmenart, -dicke, und –farbe will man? Welche Abstände sollen eingehalten werden?
  3. Inhalt der Kiste: Welche Schrift, Schriftgröße und Schriftfarbe wählt man?

, <p>, <h1> bis <h6> und Listenelemente ul, ol und li

sind sogenannte Block-Elemente und sie veranlassen somit autmatisch einen Zeilenumbruch.

- Sie fangen immer auf einer neue Zeile an.
- Nehmen immer 100% der verfügbaren „width“

Der feine Unterschied zwischen <p> und ist, dass

- <p> vor und nach dem Absatz eine Leerzeile einfügt,
- dagegenen nicht.

Das wird daher als Conatiner-Element genutzt, also als eine Art Hülle. Es bietet sich somit an, um andere Elemente zusammenzufassen und wie ein Behälter zu umschließen.

Inline-Elemete

, <a> und <button>

- Starten nicht in einer neue Zeile
- Nehmen nur die nötige „width“
- Man kann nicht „width“ nutzen

Mit kann man sehr gut ein einzelnes Wort oder sogar einzelne Zeichen gestalten. Es ist kein Block-Element, sondern ein INLINE-Element und erzeugt daher auch keinen Zeilenumbruch.

Die **Unterscheidung zwischen Block- und Inline-Elementen** ist sehr wichig. Es wäre nicht richtig, wenn man z.B. ein Wort gestalten möchte und dafür ein Block-Element verwenden würde. Nach dem Wort würde eine neue Zeile entstehen und das wäre wohl nicht gewünscht.

Unterschiede:

- Blockelemente werden so breit, wie es geht

Blockelemente ähneln Absatzformaten in Word: Der Kasten eines Blockelements wird automatisch so breit, wie es geht. Nachfolgende Elemente stehen unterhalb des Kastens in der nächsten Zeile. Blockelemente enthalten normalen Text, Inline-Elemente und manchmal auch andere Blockelemente.

- Inline-Elemente werden nur so breit wie ihr Inhalt

Inline-Elemente ähneln den Zeichenformaten aus Word: Der Kasten eines Inline-Elements wird nur so breit wie sein Inhalt. Nachfolgender Text fließt direkt nach dem Element weiter. Inline-Elemente erzeugen keine neue Zeile und sind den Blockelementen untergeordnet. Sie dürfen normalen Text und andere Inline-Elemente enthalten, aber keine Blockelemente.

Beispiele für Inline-Elemente sind strong und em sowie Hyperlinks (a) und Grafiken (img).

4)border, hover

Bilder mit runden Ecken

```
img {
    border-radius:8px;
}
```

Prinzipiell hat „border-radius“ vier Werte. Diese stehen für die vier Ecken in folgender Reihenfolge: oben links, oben rechts, unten rechts und unten links. Die Angaben werden also im Uhrzeigersinn gemacht.

Beispiel: border-radius: 5px 20px 40px 60px;

Wird nur ein Wert verwendet, beim Beispiel ganz oben „border-radius:8px“ haben alle vier Ecken einen Radius von 8 Pixeln.

Beispiel:

Bild als Link inklusive Rahmen und abgerundetem Bild:

```
a {
    display: inline-block;
    border: 1px solid #ddd;
    border-radius:4px;
    padding:5px;
    transition:0.3s;
}
```

Hover dazu: es erscheint ein Schatten als Hover-Effekt, wenn man mit der Maus drüberfährt:

```
a:hover {
    box-shadow: 0 0 2px 1px;
    rgba(0,140,186,0.5);
}
```

Tipp hover:

The :hover pseudo-class is used to select elements when you mouse over them.
The :hover pseudo-class can be used on all elements, not only on links.

5)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, daher 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.

6)Hyperlinks gestalten

Hyperlinks werden in „Kisten“ mit **a** dargestellt, z.B.

```
<a href="kontakt.html" title="Zum Kontaktformular">Kontakt</a>
```

Unbesuchte Hyperlinks werden von den Browsern standardmäßig blau, besuchte Hyperlinks werden lila dargestellt.
Pseudo-Klassen helfen bei der Definition der Zustände von Hyperlinks:

- ***a:link*** = ein unbesuchter Hyperlink (kein Zwischenraum vor und nach dem Doppelpunkt)
- ***a:visited*** = besuchter Link
- ***a:hover*** = wenn sich die Maus darüber befindet
- ***a:focus*** = wenn der Hyperlink per Tab-Taste ausgewählt wurde
- ***a:activ*** = im Moment des Anklickens

Dabei sollte man sich an genau diese Reihenfolge der Pseudolinks halten!

text-decoration: none = entfernt die Hyperlink-Unterstreichung
text-decoration: normal = bringt wieder die Unterstreichung zurück