Übung 5 - CSS-Einbau

(a) Erstellen Sie eine neue CSS-Datei und referenzieren Sie von allen drei Unterseiten darauf.

(b) Suchen Sie von Folie 59 (Auswahl der CSS-Eigenschaften) mindestens 16 Eigenschaften aus (jeweils vier beliebige aus jeder der vier Spalten) und bauen Sie sie (sinnvoll) in die bestehende CSS-Datei ein.

(c) Ergänzen Sie die CSS-Datei, so dass die ersten vier Selektor-Typen der vorigen Folie in der CSS-Datei (sinnvoll) vorkommen.

(d) Erweitern Sie Ihre CSS-Datei, so dass alle Elemente explizit formatiert sind und achten Sie dabei auch auf gute Farbgebung, Schriftart, Schriftgröße, Abstände, Rahmenlinien, Design, ...

Einzelarbeit, 30 Minuten

CSS-Selektoren: Codebeispiel

```
<body>

<h2>Obst</h2>
<ul>
  <li>Äpfel</li>
  <li id="birnen">Birnen</li>
  <li>Bananen</li>
  <li class="sonderangebot">Kirschen</li>
  <li>Pflaumen</li>
</ul>

<h2>Gemüse</h2>
<ul>
...
```

```
...
  <li>Paprika</li>
  <li id="salat">Salat
  <ul>
    <li><em>Feld</em>salat</li>
    <li><em>Kopf</em>salat</li>
    <li>Rucola</li>
  </ul>
</li>
<li class="sonderangebot">Tomaten</li>
<li>Karotten</li>
<li>Gurken</li>
</ul>

</body>
```

Einfache Selektoren

Universalselektor (Universal selector) wählt alle Elemente aus:

```
* {
  background-color: lightgray;
}
```

Typselektor (Type selector) wählt alle Elemente eines bestimmten HTML-Elementtyps aus:

```
h2 {
  color: darkgreen;
}
```

ID-Selektor (ID selector) wählt das Element mit einer bestimmten ID aus (HTML-Tag `id`):

```
#birnen {
font-weight: bold;
}
```

Klassenselektor (Class selector) wählt alle Elemente einer bestimmten Klasse (HTML-Tag `class`) aus:

```
.sonderangebot {
  background-color: yellow;
}
```

Komplexe Selektoren

Kind-Selektor (Child selector) wählt alle Elemente von der direkt darunterliegenden Ebene aus:

```
body > ul {
  list-style-type: square;
}
```

Nachfahrenselektor (Descendant selector) wählt alle Elemente von allen darunterliegenden Ebenen aus:

```
#salat em {
  text-decoration: underline;
  font-style: normal;
}
```

Nachbarselektor (Adjacent sibling selector) wählt das auf gleicher Ebene direkt folgende Elemente aus:

```
h2 + ul {
  border: 2px solid darkblue;
}
```

Geschwisterselektor (Sibling selector) wählt alle auf gleicher Ebene folgenden Elemente aus:

```
.sonderangebot ~ li {
  color: darkgray;
}
```

CSS-Selektoren: Ergebnis des Codebeispiels

Pseudo-Elemente

Es gibt die Pseudo-Elemente `::before` und `::after`, die Inhalte unmittelbar vor oder nach ein Element hinzufügen.

Beispiel: Pfeil „→“ direkt vor jeden Link und „←“ nach jedem Link einfügen:

```
...
a::before {
  content: "→";
}
a::after {
  content: "←";
}
...
```

Ergänzung: Mehrere Klassen

Die Angabe mehrerer Klassen für ein HTML-Element ist erlaubt und möglich:

Beispiel:

```
<a class="navlink kontaktlink" href="...">
   ...
</a>
```

Mehrfacher Klassenselektor „.“ für Elemente, die beiden Klassen angehören:

```
.navlink.kontaktlink {
   ...
}
```

Ergänzung der CSS-Selektoren: Auszug einiger Pseudoklassen

Wähle erstes Kindelement:

```
:first-child {
  …
}
```

Wähle letztes Kindelement:

```
:last-child {
 …
}
```

Wähle Element, auf das der Mauszeiger aktuell zeigt:

```
:hover {
 …
}
```

Beispiel für Links:

```
a:hover {
  color: orange;
  background-color: gray;
}
```

Übung 6 – erweiterter CSS-Einbau

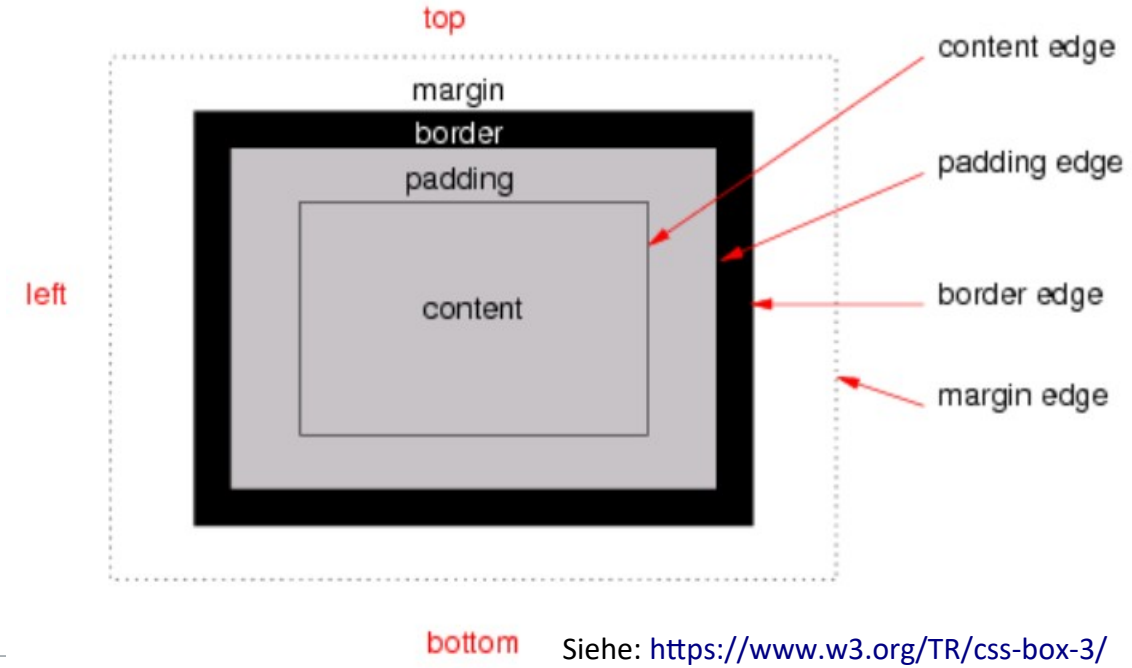
(a) Ergänzen Sie die CSS-Datei, so dass die letzten vier Selektor-Typen der Folie 63 in der CSS-Datei (so weit wie möglich sinnvoll) vorkommen.

(b) Erweitern Sie Ihre CSS-Datei, so dass auch `::hover`, `::before`, `::after` sinnvoll zum Einsatz kommen.

Einzelarbeit, 30 Minuten

Box model: Margin/Padding

- Fast jeder Inhalt (content) kann einen Rahmen (border) erhalten
- Größenangaben `height` und `width` beziehen sich nur auf den Inhalt (content), mit `box-sizing: border-box;` wird auch `padding` und `border` mit einbezogen
- Am besten keine festen Größenangaben machen, sondern Flexbox oder Grid Layout verwenden
- padding = „Polsterung“ ist der Abstand zwischen Inhalt und Rahmen, wird mit eingefärbt
- margin = „Rand“ ist der Abstand zwischen Rahmen und anderen Elementen (margins überlappen jedoch)
- jeweils `-top, -right, -bottom, -left`



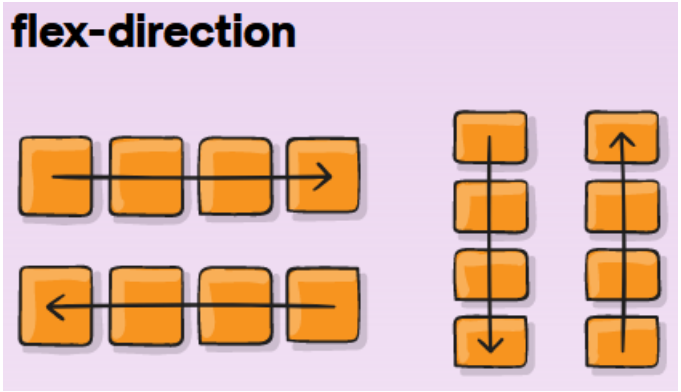
Siehe: https://www.w3.org/TR/css-box-3/

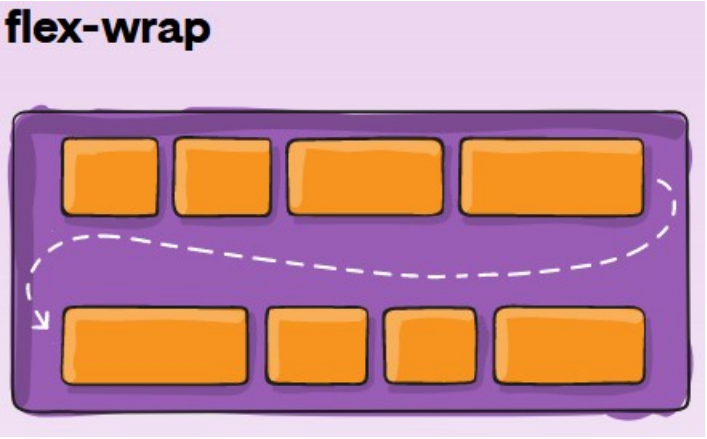
Layouting mit Flexbox (1/3)

- Einfache Layout-Möglichkeit in CSS3, noch nicht final standardisiert
- Sehr gute Seitenreaktion bei unterschiedlichen Fenstergrößen
- Übersichtliche und gut bedienbare Webseiten möglich
- Früher: Layouting mit unsichtbaren Tabellen oder mit prozentualen Angaben von Boxen oder anderen Flexbox-Vorgängern
- Aktuelle Flexbox-Container nur mit CSS-Eigenschaft: `display: flex;`
- Alle Unterelemente eines Flex-Containers sind automatisch ebenfalls flex-fähig

Layouting mit Flexbox (2/3)

```
flex-direction: row | column | row-reverse | column-reverse;
flex-wrap: nowrap | wrap;
```



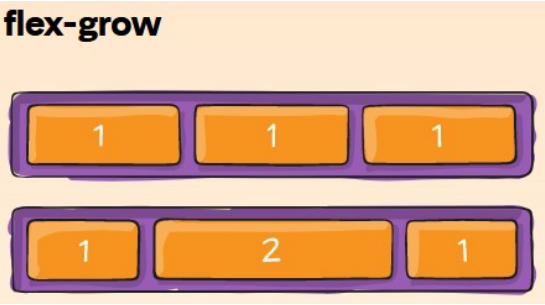


Kombinierte Eigenschaft `flex-flow`, z. B.: `flex-flow: column wrap;`

Bildquelle: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

Layouting mit Flexbox (3/3)

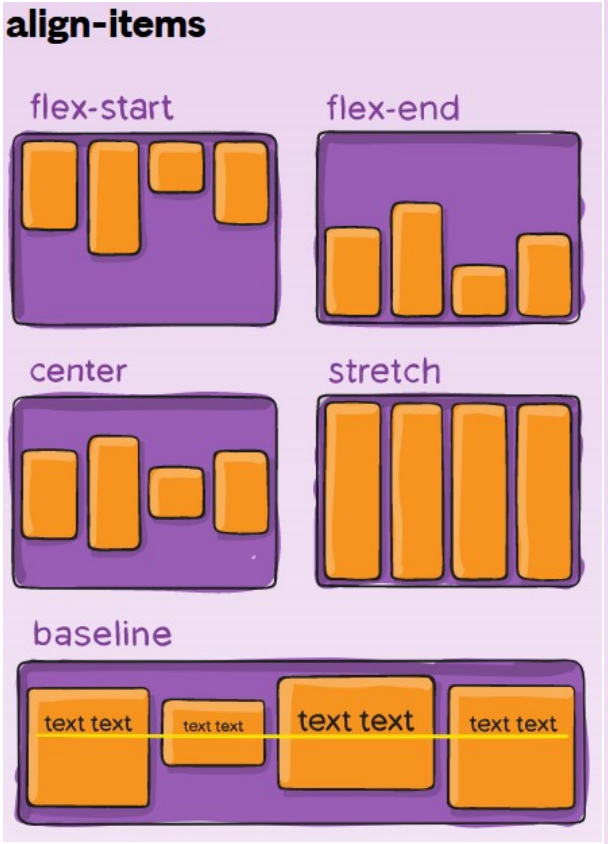
Relative Faktoren `flex-grow` und `flex-shrink`

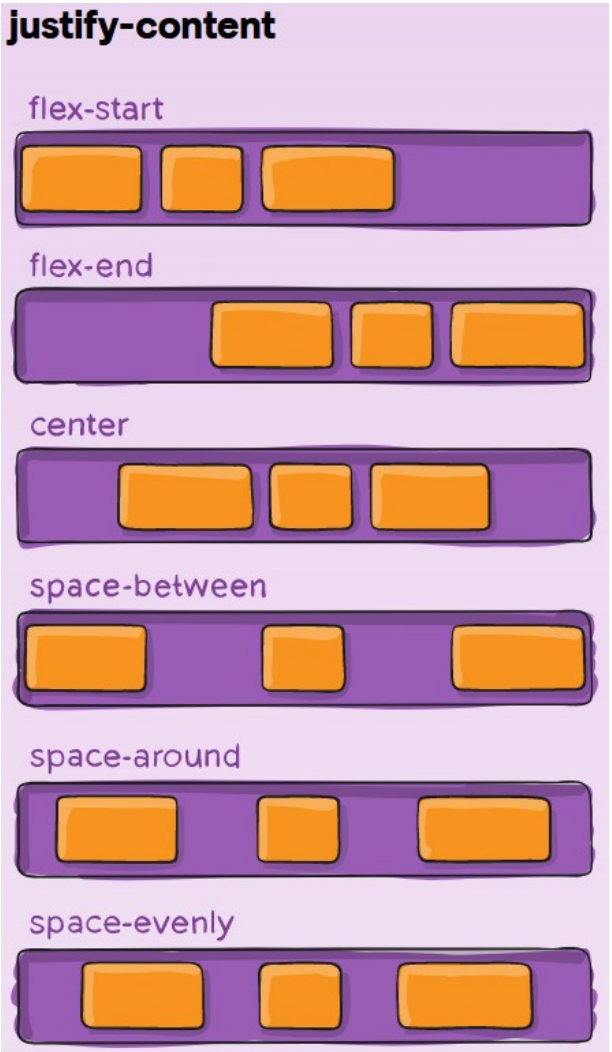


`flex-basis` definiert Default-Größe eines Elements (Wert oder `auto`)

`flex` kombiniert alle drei. Beispiel:

```
flex: 0 1 auto;
```





Bildquelle: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

→ **Live-Demo:** `NavBar.html & NavBar.css`

Displayabhängige Anpassung des Layouts

Abhängig vom aktuellen Ansichtsfenster („Viewport“) können mit *Media Queries* sehr flexible, live anpassende Layouts erstellt werden. Beispiel:

```
@media all and (max-width: 600px) {
  /* Too narrow to support three columns */
  main { flex-flow: column; }
  main > article, main > nav, main > aside {
    /* Return them to document order */
    order: 0; width: auto;
  }
}
```

Quelle: https://www.w3.org/TR/css-flexbox-1/

Weitere Beispiele:

https://www.w3schools.com/css/css_rwd_mediaqueries.asp

Übung 7

(a) Layouten Sie Ihre gesamte Webseite mithilfe von Flexboxen. Verwenden Sie das Schema von Folie 84 als Basis.

(b) Testen Sie Ihre Webseite für unterschiedliche Fenstergrößen und ergänzen Sie den Code um diplayabhängige Neuanordnungen für mindestens drei verschiedene Displaybreiten (Bildschirm, Tablet, Handy), analog zum Live-Demo-Beispiel `navBar.css`.

(c) Testen Sie verschiedene Werte von `align-items` und `justify-content` und beobachten Sie, wie sich das Layout jeweils verändert.

Einzelarbeit, 60 Minuten