HTML5 LAYOUTING UND EINFÜHRUNG IN BOOTSTRAP

MMWP2024 - LV02

INHALTSVERZEICHNIS

- Organisation
- Klassisches HTML-Layout
- Modernes HTML5
- Bootstrap 5

INHALTSSCHWERPUNKTE

- Historische HTML-Layouts
- HTML5 Layout-Modelle
- CSS3 Stylings
- Einführung in Bootstrap5

VORAUSSETZUNG

Der Ausgangspunkt dieser Vorlesungsreihe ist das Wissen in klassischer Webprogrammierung

- HTML 4.01, XHTML 1.0 – Programmierung statischer Webseiten mit Medien
- CSS 2.0 – Kaskade der CSS-Anweisungen, Boxenmodell (Standard), klassischer div-Boxen-Programmierstil
- JavaScript – DOM-Modell, AJAX-Methoden, Reaktion auf Nutzerinteraktionen, Manipulations-möglichkeiten von HTML-Dokumenten im Browser

ZIELE VON MODERNEN WEBDESIGNS

- Neue Herangehensweisen an klassische Webprogrammierung aufzeigen
- Dabei sollen die in HTML5, HTML5 APIs, CSS 3 und JavaScript-Frameworks auftauchenden neuen Programmiermöglichkeiten in wesentlichen Elementen mit eingeführt werden
- Vorstellung aktueller Browserkompatibilitäten bei HTML5 Elementen und CSS-Regeln

HERANGEHENSWEISE

- Egal, ob künstlerisches Design der Website oder strikt rechteckiges, lückenloses Boxendesign – am Anfang steht eine Vorstellung von der zukünftigen Flächenaufteilung der Webseiten
- Weiterhin ist entscheidend, ob man zuerst für Smartphones und kleine Bildschirme entwirft, oder zuerst für Desktops und Netbooks
- Beachtung der Mobile-First-Regel, falls die geplante Webseite von Suchmaschinen gut bewertet werden soll
- Beachtung von verschiedenen Bildgrößen Mithilfe von Responsive- bzw. Fluid Design

TABLE-LAYOUT - 1

"Früher" wurden Webseiten über Tabellen aufgebaut, um somit verschiedene Reihen und Spalten zu ermöglichen

WEBSITE LOGO

Menu
link
Menu
link
Menu
link
Menu

TABLE-LAYOUT - 2

Erst wurden Webseiten über Tabellen aufgebaut, um somit verschiedene Reihen und Spalten zu ermöglichen

```
<table width="100%" style="height: 100%;" cellpadding="10" cellspacing="
  <tr>
    <!-- ============ HEADER SECTION ============== -->
    <td colspan="3" style="height: 100px;" bgcolor="#777d6a"><h1>Website
  <tr>
    <!-- ============ LEFT COLUMN (MENU) ============== -->
    <td width="20%" valign="top" bgcolor="#999f8e">
      <a href="#">Menu link</a><br>
      <a href="#">Menu link</a><br>
      <a href="#">Menu link</a><br>
      <a href="#">Menu link</a><br>
      <a href="#">Menu link</a>
    </td>
    <!-- ============ MIDDLE COLUMN (CONTENT) ============== -->
    <td width="55%" valign="top" bgcolor="#d2d8c7">
```

EINFACHE HTML SEITE OHNE STYLING

```
<!DOCTYPE html>
<html lang="de">
<head>
    <meta charset="utf-8">
    <meta name="Beschreibung" content="HTML5-Beispiel">
    <meta name="keywords" content="HTML5, CSS3, JS">
    <title>Dieser Text ist der Dokumenttitel </title>
</head>
<body>
    <div id="#wrapper">
        <div id="main_header">
            <h1>Dies ist der Haupttitel der Website</h1>
        </div>
        <div id="main_menu">
```

Source

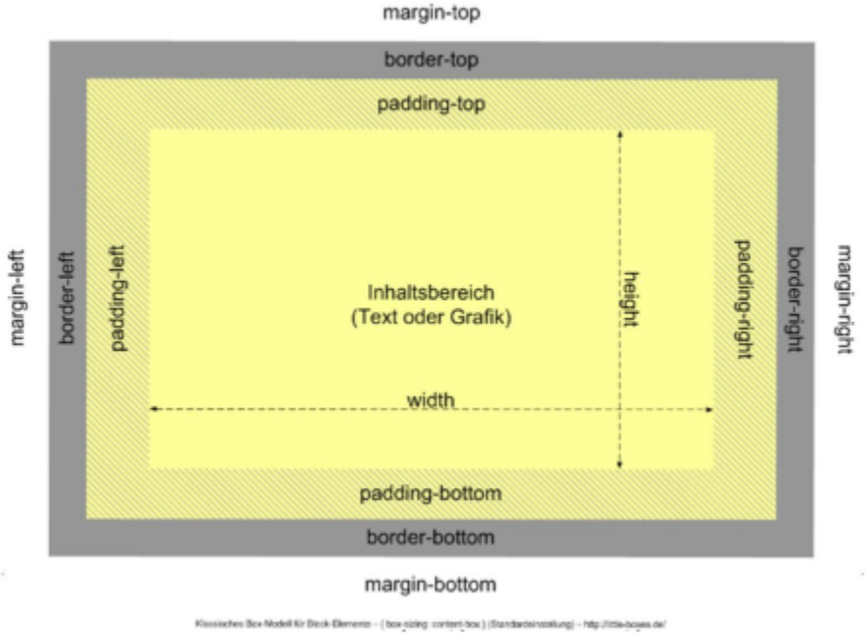
LAYOUT MIT KLASSISCHEN BOXEN - 1

- Darauf folgend wurden neue Möglichkeiten von Inhaltsplatzierung definiert (2009)

```
display: box;
```

- Das klassische Boxenmodell berechnet den Platz für den Inhalt einer div-Box nach width, height
- Die Maße für Padding, Border und Margin kommen als realer Platzbedarf im Bild dazu

LAYOUT MIT KLASSISCHEN BOXEN - 2



Quelle

LAYOUT MIT KLASSISCHEN BOXEN - 3

- Um dem Boxenmodell generell konsistente Maße für margin und padding geben zu können, werden Reset-Rules in der CSS-Datei empfohlen:

```
* { margin: 0px; padding: 0px; }
```

- Weitere Reset-Rules rules können für gleichmäßigere Anzeige von Webseiten auf verschiedenen Browser benutzt werden (Quelle)

LAYOUT MIT KLASSISCHEN BOXEN - 4

- Manchmal ist es gewünscht, auf überbreiten Bildschirmen (16x10) mit einer Maximalbreite der Website zu arbeiten, und diese dann zu zentrieren

```
body {text-align: center }
```

- Für die flächige Anordnung der div-Boxen ist es mitunter erforderlich, Gruppen von Boxen mit unsichtbaren Wrapper-div-Boxen zu kapseln

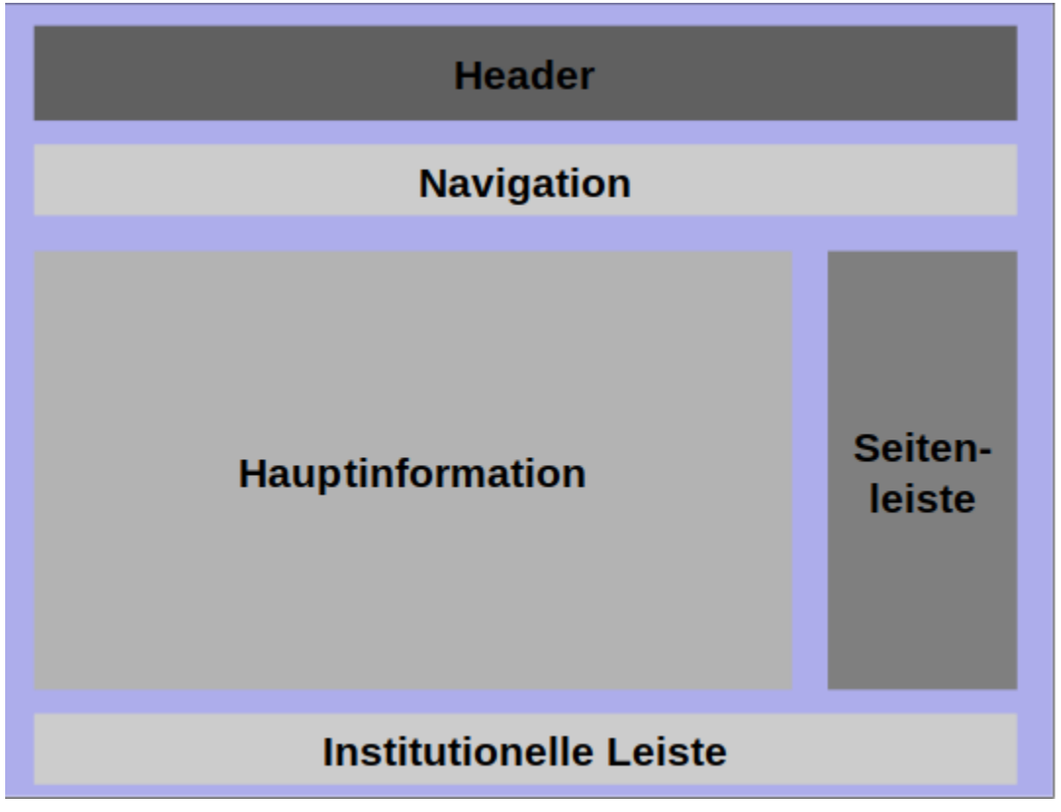
LAYOUT MIT KLASSISCHEN BOXEN - 5

- So wird alles in <body> mit umgeben:

```
1 #wrapper { width: 960px;
2 margin: 15px auto;
3 text-align: left }
```

- Der margin-Befehl wirkt auf Blockelemente und bewirkt deren Zentrierung. Der text-align-Befehl hebt den Befehl zu "<body>" wieder auf, für alle inneren Folgeelemente

STATIC BOX-LAYOUT - 1



STATIC BOX-LAYOUT - 2

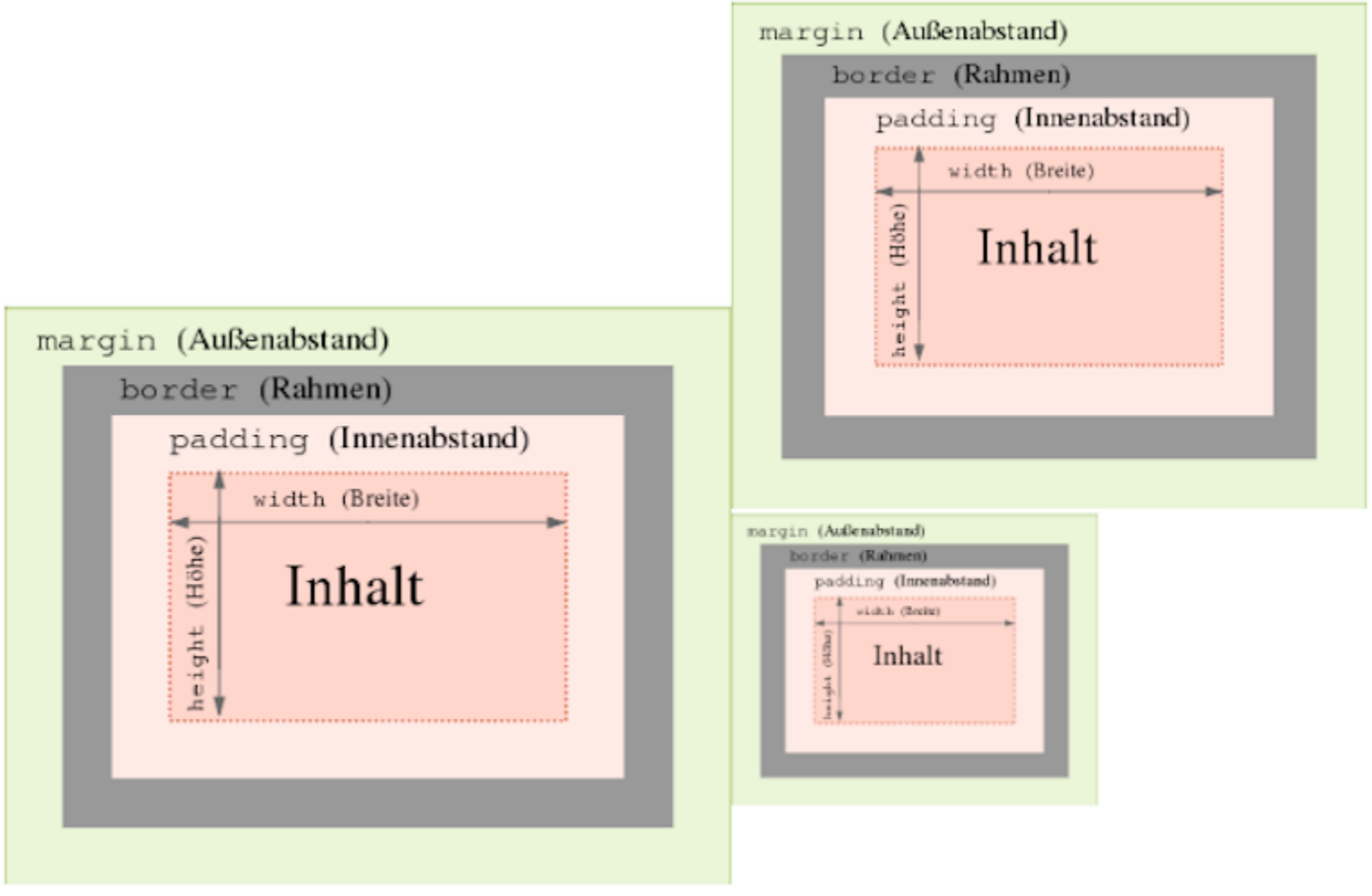
```
* {
    margin: 0px;
    padding: 0px;
}

h1 {
    font: bold 20px verdana, sans-serif;
}
h2 {
    font: bold 14px verdana, sans-serif;
}

header, section, footer, aside, nav, article, figure, figcaption, hgroup
    display: block;
}
```

Source

PROBLEME DES KLASSISCHEN BOXEN-MODELLS - 1

- Größe der Boxen lässt sich schlecht auf verschiedenen Bildgrößen berechnen
- Abstände zu anderen Boxen können von Box zu Box verschieden sein und beeinträchtigen die Positionierung

PROBLEME DES KLASSISCHEN BOXEN-MODELLS - 2



Source

"NEUE" MÖGLICHKEITEN VON WEBELEMENTEN IN HTML5

- Barrierefreiheit-Regeln wurden als Standard ausgearbeitet
- Anpassbarer Inhalt an die Gerätegröße
  - Auflösung (1920x1080)
  - Pixeldichte (Pixel density)
  - Farbunterstützung (LCD, OLED, LED, HDRI)
- Mobile Geräte benötigen andere Layouts und Funktionen (z.B. Touch-Navigation)

DIE HTML5 SPEZIFIKATION

- Grundsätzlich werden die Standards für den deklarativen Teil von HTML5 und für CSS3 von zwei Organisationen gesetzt:
  - **World Wide Web Consortium (W3C)** - **https://www.w3.org**
    Ziel: fortsetzende Spezifikation von HTML5 und CSS3
  - **Web Hypertext Application Technology Working Group (WHATWG) – https://www.whatwg.org**
    Ziel: Eine fortwährende Standardisierung dessen propagieren, was sich mittlerweile unter der Bezeichnung „HTML5“ etabliert hat (z.B. Browser-APIs)
- Es werden stetig neue Proposals veröffentlicht: https://html.spec.whatwg.org/

KONTEXTBEZOGENE (SEMANTISCHE) HTML5 ELEMENTE

<article>	Definiert einen unabhängigen, in sich abgeschlossenen Inhalt.
<aside>	Definiert einen Abschnitt mit zusätzlichen Informationen zum Inl
<bdi>	Isoliert bidirektionalen Text (wenn eine Sprache mit Rechts-nacl
<details>	Enthält zusätzliche Details, die der Benutzer öffnen und anzeig
<dialog>	Gibt ein Dialogfeld oder Fenster an.
<figcaption>	Fügt eine Beschriftung oder Erklärung zum Inhalt des <figure> T
<figure>	Legt einen in sich geschlossenen Inhalt fest.
<footer>	Definiert die Fußzeile einer Webseite oder eines Abschnitts.
<header>	Definiert eine Kopfzeile einer Seite oder eines Abschnitts.
<main>	Gibt den Hauptinhalt eines Dokuments an.
<mark>	Markiert einen Teil des Textes, der von Bedeutung ist.
<meter>	Definiert eine skalare Messung im bekannten Bereich oder die gra
<nav>	Definiert einen Block von Navigationslinks, entweder innerhalb d
<progress>	Zeigt den Fortschritt der Aufgabe an (Fortschrittsbalken).
<rp>	Definiert einen alternativen Text, der in den Browsern angezeigt

MEDIA ELEMENTE

```
<audio>   Zeigt die Variationen der gleichen Audiodatei an.
<embed>   Wird als Container für externe Anwendungen, Multimedia und interal
<source>  Definiert mehrere Medienressourcen in verschiedenen Formaten: Vide
<track>   Legt Textspuren für Medienelemente fest.
<video>   Bettet Video in ein HTML-Dokument ein.
```

WEITERE ELEMENTE:

FORMULAR:

```
<datalist>  Creates a list of input options, predefined by the <input> tag.
<output>  Defines a place for representing the result of a calculation perfo
```

GRAFIK:

```
<canvas>  Defines an area on the web page, where we can create different obj
<svg>   Draws scalable vector graphics.
```

BARRIEREFREIHEIT VON WEBSEITEN - 1

- Verbesserung der Zugänglichkeit von Webseiten, insbesondere von dynamischen Inhalten und Komponenten bei Webseiten
- Ajax, HTML, JavaScript und verwandten Technologien werden hierfür verwendet
- Seit März 2014 ist Accessible-Rich-Internet-Applications (ARIA) ein empfohlener Webstandard des World Wide Web Consortium (W3C)

BARRIEREFREIHEIT VON WEBSEITEN - 2

- Webseitensteuerungen und Inhaltsaktualisierungen sind für Nutzende mit Behinderungen oft nicht zugänglich, insbesondere für Nutzende mit Screen-Reader
- Zu kleine Schrift, oder fehlender Kontrast stellen auch Problem dar
- Erweiterung: Accessible Rich Internet Applications suite of web standards (WIA-ARIA)

SEMANTISCHE ELEMENTE IM HTML5 - 1

- Statt reinem div-Boxen-Design sollen semantisch orientierte Elemente von HTML5 für die Hauptboxen im Quellcode eingesetzt werden
- Zusätzlich werden gleich „Landmark Roles“ nach WAI-ARIA mit eingebaut
- Screenreadern können „Landmark Roles“ als Orientierung dienen

SEMANTISCHE ELEMENTE IM HTML5 - 2

- Chromium und Gecko-Browser ermöglichen automatisierte Screenreader-Navigation
- Landmark Roles werden von den Screenreadern JAWS, NVDA, ORCA, Chromevox, Window Eyes und VoiceOver verstanden, und via FireFox addon durch Tastaturnutzer
- Neuste Elemente für ARIA-Unterstützung

GRUNDSÄTZLICHE REGELN VON ARIA - 1

- ARIA-Elemente sollten nicht verwenden werden, wenn die gewünschte Semantik durch die Verwendung eines nativen HTML-Elements oder -Attributs erreicht werden kann
- Es ist nicht ratsam, die Semantik von nativem HTML zu verändern, es sei denn, es gibt einen zwingenden Grund, dies zu tun
- Es ist zwingend erforderlich, dass alle interaktiven ARIA-Steuerelemente über die Tastatur bedient werden können

GRUNDSÄTZLICHE REGELN VON ARIA - 2

- Es ist nicht ratsam, die Semantik zu entfernen oder fokussierbare Elemente zu verbergen
- Es ist zwingend erforderlich, dass allen interaktiven Elementen ein zugänglicher Name zugewiesen wird, der die Accessibility API verwendet

SEMANTIK ELEMENTE - 1

```
<body>
  <header role="banner">
    <h1>Dies ist der Haupttitel der Website</h1>
  </header>
  <nav role="navigation">
    <ul>
      <li>Start</li>
      <li>Fotos</li>
      <li>Videos</li>
      <li>Kontakt</li>
    </ul>
  </nav>
</body>
```

SEMANTIK ELEMENTE - 2

```
<section role="main">
  <article>
    <header>
      <h1>Titel von Beitrag eins</h1>
      <h2>Untertitel von Beitrag eins</h2>
      <p>erschienen am 10.12.2016</p>
    </header>
    <p>Dies ist der Text meines ersten Beitrags</p>
    <footer>
      <p>Kommentare (42)</p>
    </footer>
  </article>
</section>
```

SEMANTIK ELEMENTE - 3

```
<aside role="complementary">
  <blockquote>Artikel Nummer eins</blockquote>
  <blockquote>Artikel Nummer zwei</blockquote>
</aside>
<footer role="contentinfo"> Copyright &copy; Sybex,
J.D. Gauchat, 2023-2024</footer>
```

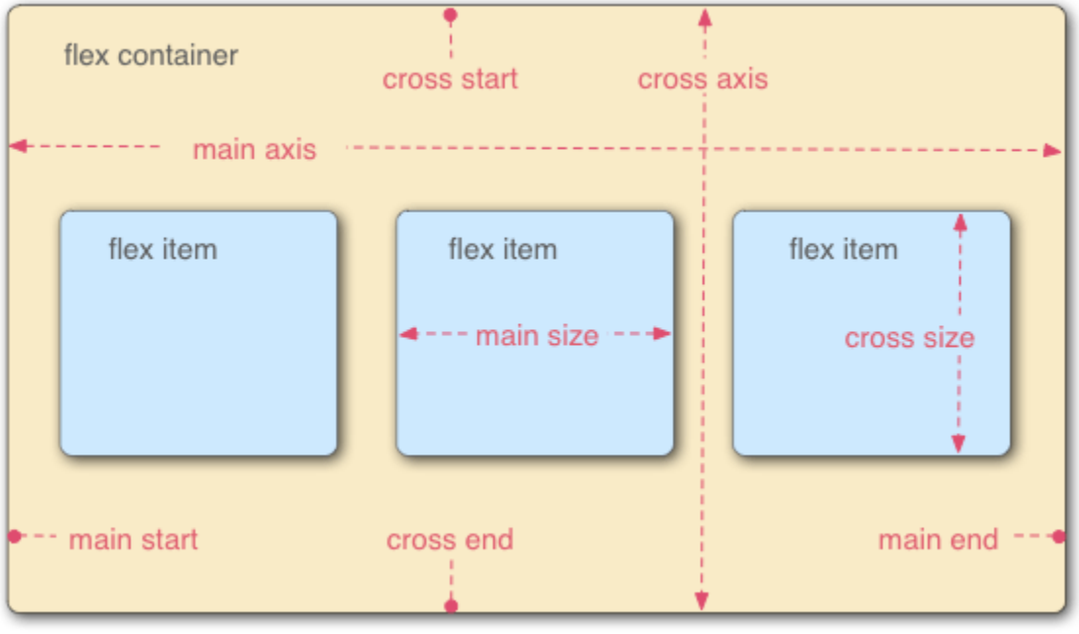
RESULTAT NACH EINFÜGEN DER SEMANTISCHEN ELEMENTE

```
<!DOCTYPE html>
<html lang="de">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0
    <meta name="Beschreibung" content="HTML5-Beispiel">
    <meta name="keywords" content="HTML5, CSS3, JS">
    <title>Dieser Text ist der Dokumenttitel </title>
</head>
<body>
    <header role="banner">
        <h1>Dies ist der Haupttitel der Website</h1>
    </header>
    <nav role="navigation">
        <ul>
```

FLEX-BOX-LAYOUT - 1

- Das flexible Boxenmodell aus CSS3 geht im Gegensatz zum klassischen Boxenmodell davon aus, dass width für eine Box immer Inhaltsbreite+Padding+Border ist
- Die Margin bleibt als Platz zur nächsten Box ausgenommen
- **Hier wird erstmalig deutlich, dass HTML5 und CSS3 hinter einfachen Befehlen in der Syntax komplexe Modelle verbirgt!**

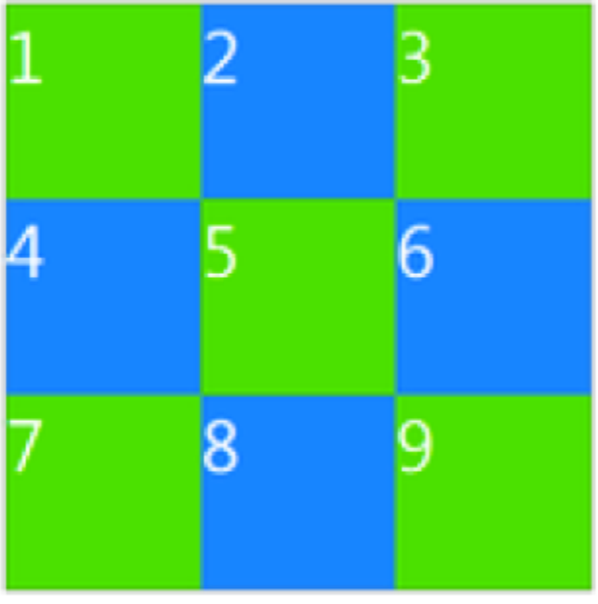
FLEX-BOX-LAYOUT - 2



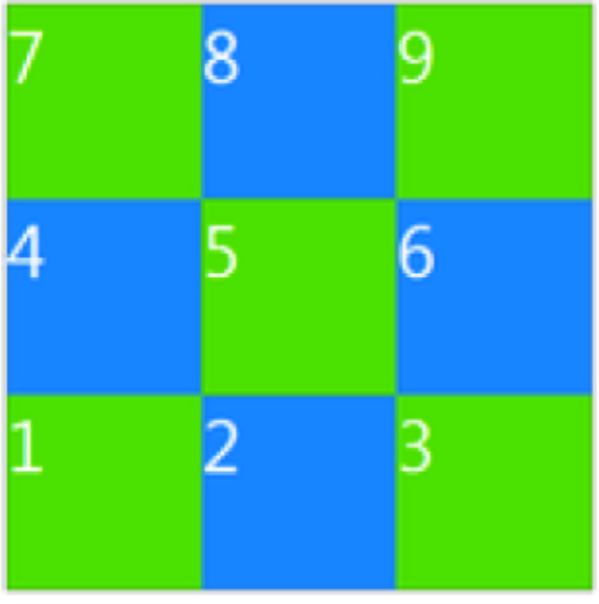
Source

FLEX-BOX-LAYOUT - 2

flexbox mit Umbrüchen



`-ms-flex-wrap: wrap;`

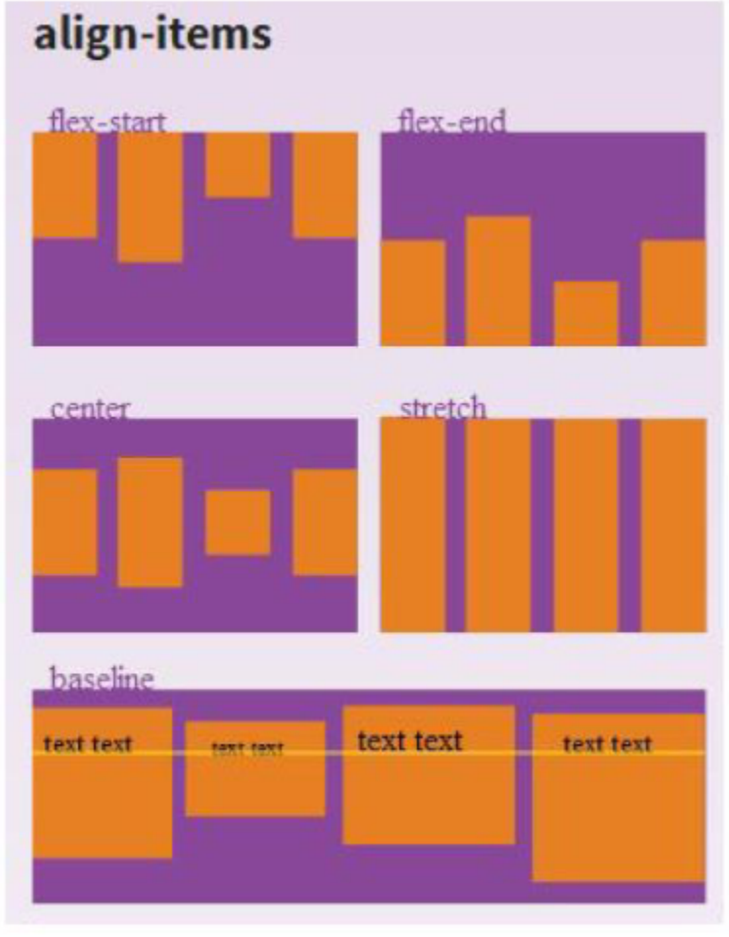


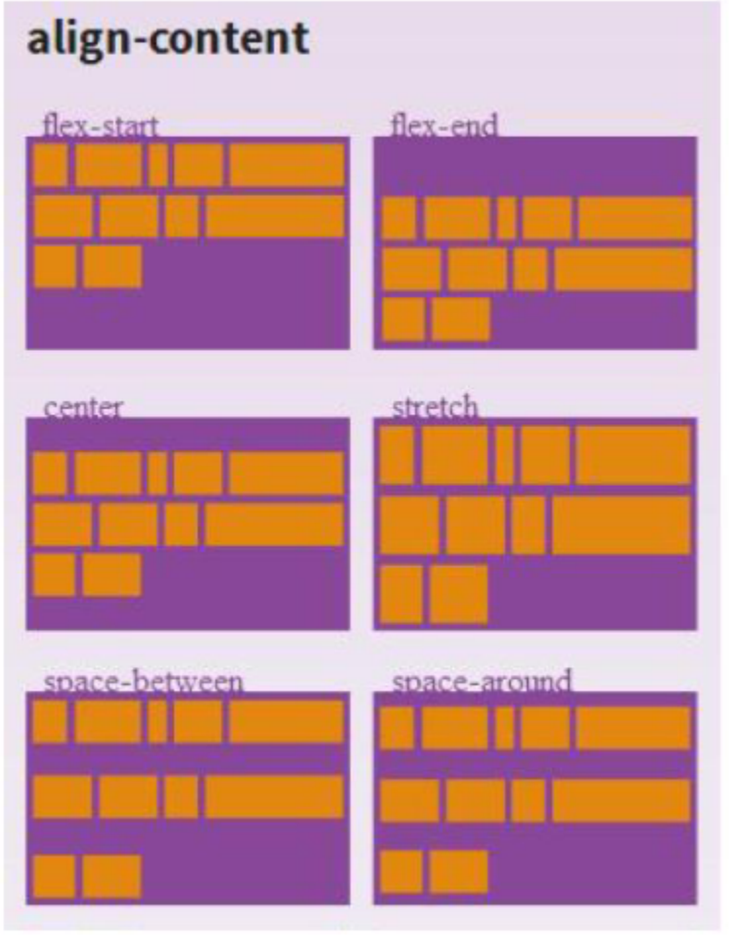
`-ms-flex-wrap: wrap-reverse;`

In den meisten Browsern bereits ohne Präfix möglich.

Source

FLEX-BOX-LAYOUT - 3





Source

NUTZEN VON FLEX-BOXEN

- Damit sind Breitenangaben genauer, es muss nur noch auf die Werte für margin geachtet werden
- Die Angaben für **padding und border** beeinflussen nicht mehr globale Abstände
- Zusammen mit der Einstellung des Viewports

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Können Flexboxen besser genutzt werden: z.B.

ANWENDUNG VON FLEX-BOXEN

```html
<div class="flex-container">
  <div class="flex-item"></div>
  <div class="flex-item"></div>
  <div class="flex-item"></div>
  <div class="flex-item"></div>
</div>
```

```css
.flex-container {
  display: flex;
  flex-direction: row; /* Standardwert */
}
```

LAYOUT MIT FLEXIBLEN BOXEN

Einführung in das flexbox modell

BROWSER SUPPORT

Source

FLEXBOXEN-EINSATZ IM BEISPIEL

```
body {
    width: 100%;

    display: -ms-flex;
    display: -webkit-flex;
    display: -moz-flex;
    display: flex;

    -ms-flex-direction: column;
    -webkit-flex-direction: column;
    -moz-flex-direction: column;
    flex-direction: column;

    -ms-justify-content: flex-start;
    -webkit-justify-content: flex-start;
```

Beispielseite mit FlexBoxen

FLEX-BOXEN EINSTELLUNGEN - 1

- Entscheidend für das Verständnis ist, dass es Eigenschaften für Eltern-Boxen und Eigenschaften für Kind-Boxen gibt
- Jede Box kann beide Rollen zugleich haben, die Eigenschaften gelten aber immer für eine Eltern-Box mit ihren Kind-Boxen
- (Die Begriffe „Eltern“ und „Kind“ kommen aus der SGML-/XML-Hierarchie im DOM-Modell)

FLEX-BOXEN EINSTELLUNGEN - 2

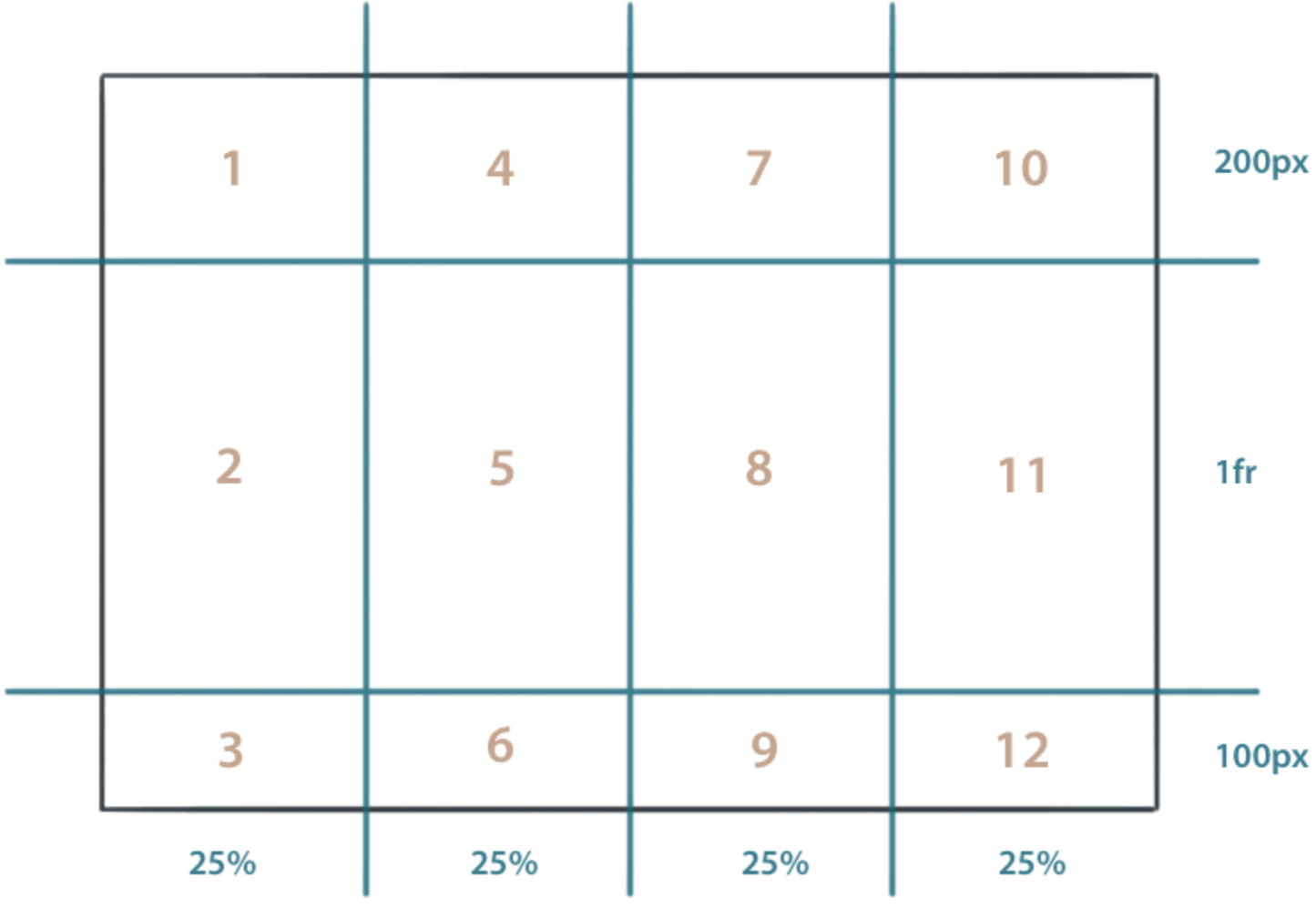
- Das Flexbox-Modell hat kein unterlegtes Grid, es werden nur Zeilen oder Spalten initial als Orientierung festgelegt (main axis)
- Um bei flexiblem Maß in der Breite (row-Stil) eine Mindest-/Maximalbreite der Kindboxen zu garantieren, sind die CSS-Eigenschaften min-width bzw. max-width zu benutzen
- Es geht aber einfacher: ein Grid in der Richtung der cross-axis

GRID-LAYOUT

```
<div class="container">
  <div>Element 1</div>
  <div>Element 2</div>
  …
  <div>Element 11</div>
  <div>Element 12</div>
</div>
```

```
container {
  display: grid;
  grid-template-rows:200px 1fr 100px;
  grid-template-columns:25% 25% 25% 25%;
}
```

GRID-LAYOUT



Source

BROWSER SUPPORT

Source

EINBINDEN VON BILDERN IN FLEX/GRID-BOXEN - 1

- **alt und role** sind für Screenreader und nicht mehr existierende Bilder nützlich
- **alt** sollte immer verwendet werden
- Es können mehrere Bilder aufgezählt werden
- Die Bildunterschrift passt sich der Gesamtbreite zentriert an
- Weitere Elemente werden in späteren Veranstaltungen behandelt

```
1 <figure>
2   <img src="shib.jpg" alt="Much Shib, very Dog, Such Pumpkin" role="img"/
3   <figcaption>
4   Dies ist ein Bild des ersten Beitrags,
5     <a href="https://happygoriley.tumblr.com/" target="_blank">Quelle</a>
6   </figcaption>
7 </figure>
```

EINBINDEN VON BILDERN IN FLEX/GRID-BOXEN - 2

Grundseite mit Flexbox und Bild

EINBINDEN VON VIDEOS IN FLEX/GRID-BOXEN - 1

- Es können mehrere Videoformate (sources) aufgezählt werden
- Metadata, wie Untertitel können (asynchron) vorab geladen werden
- Hinweistext, oder Video beschreibungen, wenn Videoformate nicht unterstützt werden
- Weitere Elemente werden in späteren Veranstaltungen behandelt

```
1 <video preload="metadata" controls="" poster="../images/vl2/shib.jpg" dat
2   <source src="../images/vl2/sample.mp4" type="video/mp4">
3   <source src="../images/vl2/sample.mkv" type="video/mkv">
4   <source src="../images/vl2/sample.vid" type="video/vid">
5   <div class="fallback"> Your browser is not able to display this video.<
6 </video>
```

EINBINDEN VON VIDEOS IN FLEX/GRID-BOXEN - 2

Grundseite mit Flexbox und Video

FLEXBOXEN UND GRID-DESIGN

- Viele Webseiten verwenden Flexboxen oder basieren auf Grids
- Frameworks wie Bootstrap bauen auf diesen Designs auf
- Zusammen mit Javascript ergibt sich eine responsive Webseite mit möglichen Interaktionen

BOOTSTRAP 5 ALS PROGRAMMIERHILFE FÜR RESPONSIVE WEBSEITEN

- Bootstrap ist eine leistungsstarke, funktionsreiche Bibliothek
- Es basiert auf HTML5, CSS3 und JS
- Bootstrap benutzt hauptsächlich Flexboxen als Layout
- Einbindung der CSS-Datei sowie JS-Datei reicht aus, um alle Funktionen nutzen zu können
- Schritt-für-Schritt-Anleitung

BOOTSTRAP 5 BROWSERUNTERSTÜTZUNG

- Bootstrap funktioniert auf allen modernen DesktopBrowsern
- Auch Browser auf Mobilegeräten (Chrome, Firefox, Safari) werden unterstützt
- Quelle
- CSS-"Hacks" und Javascript-Funktionen helfen bei der Unterstützung von alten Browsern (z.B. Internet Explorer)

BOOTSTRAP 5 ELEMENTE

- Layout: Flex- und Grid-Elemente
- Content: Seiteninhalte z.B. Bilder und Texte
- Forms: Elemente rund um Formulare
- Components: Sammlung von nützlichen Elemente wie Pop-Ups, Gallery oder Buttons
- Helpers: HTML-Element-, Klassen- und Javascript-Sammlung wie Farben, Icon-Buttons und
- Utilities: Sammlung von CSS-Klassen und deren Erklärung
- (Neu) Icons: Sammlung von SVG-Icons (Ähnlich wie bei https://fontawesome.com/)

BOOTSTRAP 5 GUIDES

- Bootstrap 5 Documentation
- Bootstrap 5 Beispiel-Seiten
- W3schools Guide

LITERATUR

- J. D. Gauchat, „HTML5, CSS3 & JavaScript“, Sybex, 2013
- https://www.paulirish.com/2012/box-sizing-border-box-ftw/, 01.02.2012
- Ranjan, Alok, Abhilasha Sinha, and Ranjit Battewad. JavaScript for modern web development: building a web application using HTML, CSS, and JavaScript. BPB Publications, 2020.
- CSS-Framework KUBE, minimalistisch und effektiv, https://getbootstrap.com/ – Bootstrap 5
- CSS Flexible Box Layout Module Level 1, 25.09.2014, Working Draft des W3C: http://www.w3.org/TR/css-flexbox-1/
- A Complete Guide to Grid: https://css-tricks.com/snippets/css/complete-guide-grid/

ABSPANN

Zweites Level geschafft weitere Folgen!

Fragen und Feedback?