FH Salzburg

Responsive Web Design

DI Eduard Hirsch

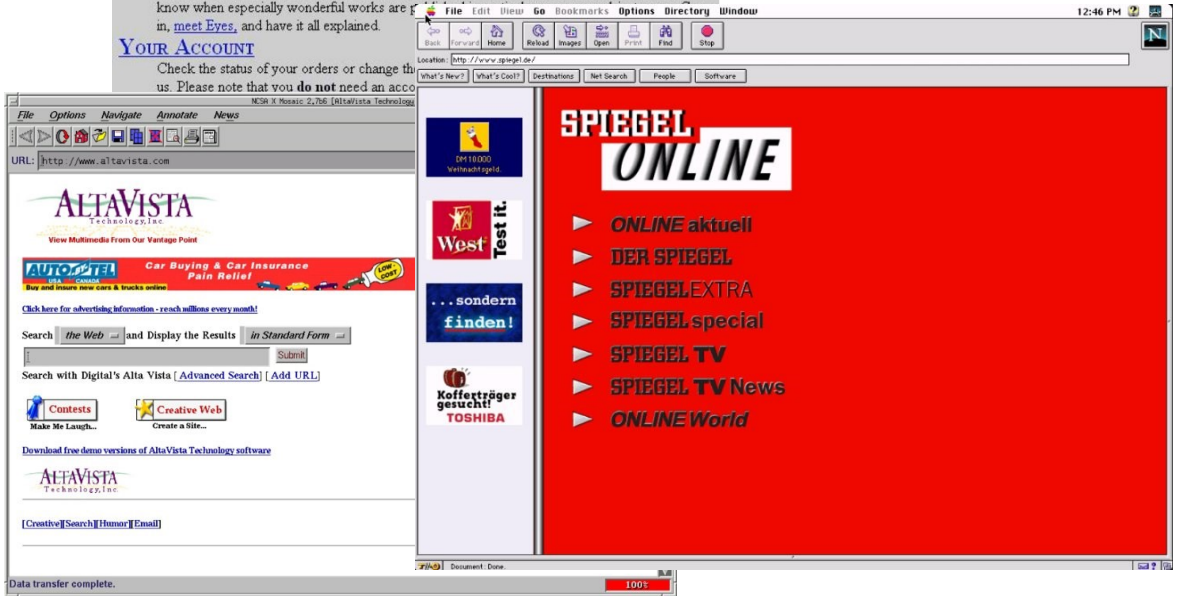
DI(FH) DI Bernhard Danninger

Technik
Gesundheit
Medien

Webseiten

Klassisches Design mit einer fixen in Pixel definierten Breite => fixe Auflösung, fixer Browser und/oder fixes Betriebssystem

Fast schon ausgestorben…



Webseiten – Entwicklung „damals“

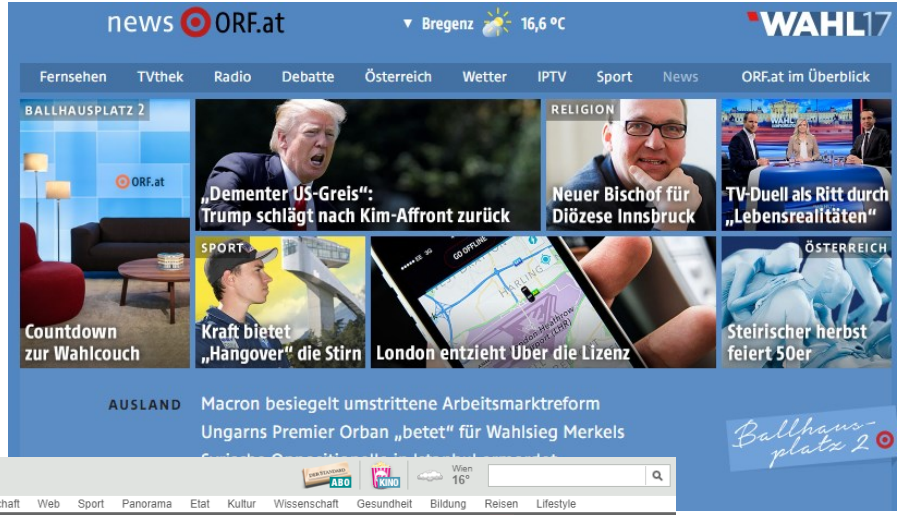
Best Viewed in

Internet Explorer @ 1024

Browser Plugins

Webseiten

Klassisches Design mit einer fixen in Pixel definierten Breite => fixe Auflösung, fixer Browser und/oder fixes Betriebssystem



Neue Ansätze

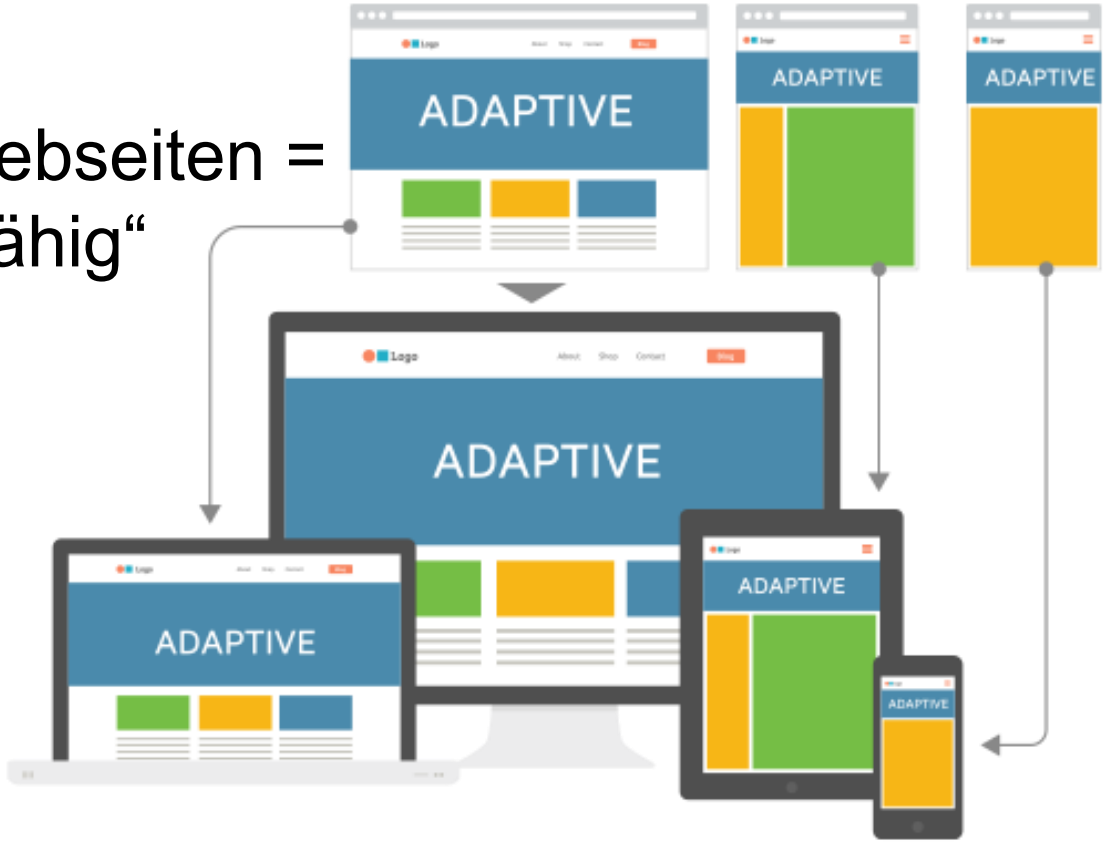
Der klassischer Ansatz **nicht** mehr praktikabel

Neue Wege:

- Adaptives Design
- Responsives Design

Webseiten Typen

„**Adaptive**“ Webseiten =
„anpassungsfähig“



„Adaptive“ Webseiten

- Ein „Adaptive“ Layout oder „Adaptive Web Design“ **AWD** beschreibt ein für bestimmte Displaygrößen optimiertes Web-Layout.
- Nicht für alle Displaygrößen
- Also exakte Viewports wie: Desktop, Tablet, Smartphone
- Breitendefinition mit Media Queries
- Oft mit Serverseitiger „Automatic Client Detection“

Webseiten Typen

„**Responsive**“ Webseiten =
„reaktionsfähig“

„Responsive“ Webseiten

Ein „Responsive“ Layout oder „Responsive Web Design“ beschreibt eine Lösung um **alle erdenklichen Geräte** zu optimieren. Praktisch stark auf Displaygrößen bezogen, sollten **auch andere Geräteeigenschaften** im Design berücksichtigt werden.

- Design Elemente sollten entsprechend dem Gerät verwendet bzw. dargestellt werden (z.B. Handy: statt pop-up Dialog, eine Sidebar)
- Design Features werden je nach Geräteunterstützung eingeblendet

RWD vs AWD

Quelle: https://css-tricks.com/the-difference-between-responsive-and-adaptive-design/

„Adaptive“ Webseiten - Vorteile

- Es kann gut mit **klassischen Mockups, Wireframe** und Skizzen gearbeitet werden, da feste Abmessungen existieren
- Inhalte müssen nur **für klar definierte Abmessungen optimiert** werden, aber nicht vollkommen flexibel sein
- Viel **gestalterischer Freiraum**, da mit einem starren Raster gearbeitet wird
- Technisch **recht unkompliziert** umzusetzen
- **Zeitsparendere** Umsetzung pro Layout

„Adaptive“ Webseiten - Nachteile

- Es wird nur für **bestimmte** Viewports / bestimmte Geräte optimiert
- **Häufige Fehldarstellungen** auf abweichenden Endgeräten
- **Aufwändige Zielgruppenanalyse** um die relevanten Viewports zu bestimmen
- Häufig **mehr CSS-Code** als notwendig

„Responsive“ Webseiten - Vorteile

- **„Jede“ Displaygröße** wird optimal berücksichtigt
- Es wird **kein Platz verschenkt**
- Die **Information** steht im **Vordergrund**
- **Zukünftige** mobile Endgeräte werden automatisch mit abgedeckt

„Responsive“ Webseiten - Nachteile

- Mockups, Wireframes und Skizzen stoßen an ihre Grenzen. Häufig muss mit **Prototypen** gearbeitet werden um Kunden das Verhalten der Website zu zeigen
- **Komplexer** in
  - der Gestaltung
  - der technischen Umsetzung
  - der Anpassung der Seiteninhalte
- **Zeitintensivere** Umsetzung

Zusammenfassung & Fragen

Fragen

- Was ist Responsive Web Design
- Was ist Adaptives Web Design
- Was sind Vor- & Nachteile der jeweiligen Methoden
- Warum sind moderne Designs wichtig

Inhalte

- Responsive Web Design
- RWD vs AWD
- alte, einfach und moderne Designs
- Plugins

Responsive Web Design

Wird häufig als Überbegriff verwendet

Aber wie funktioniert es??

Breakpoints - Umbruch

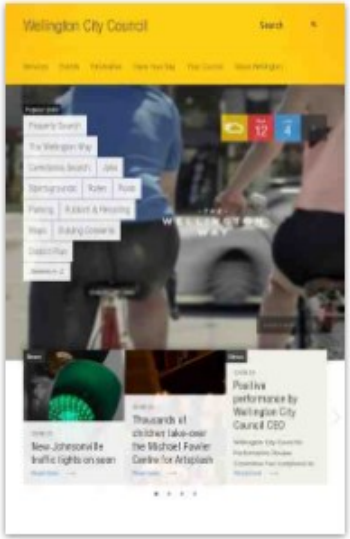
- … ist der Punkt, an dem das Design für die Größe des Darstellungsfeld umspringt
  - Major breakpoints
  - Minor breakpoints

- wird durch ein @media-Regel (Media Query) beschrieben:

```
.column { width: 90%; margin: 1em auto; }
.column h2 { font-size: 1.2em; color: firebrick; }

@media only screen and (min-device-width: 40em) {
        .column { width: 48%; float: left; }
}
```

Breakpoints - Beispiel



Quelle: http://mediaqueri.es

Breakpoints

```
/* Small screens: Define mobile styles */
@media only screen { }
/* max-width 640px, mobile-only styles */
@media only screen and (max-width: 40em) { }

/* Medium screens: min-width 641px, medium screens */
@media only screen and (min-width: 40.063em) { }
/* min-width 641px and max-width 1024px */
@media only screen and (min-width: 40.063em) and (max-width: 64em) { }

/* Large screens: min-width 1025px, large screens */
@media only screen and (min-width: 64.063em) { }
/* min-width 1025px and max-width 1440px*/
@media only screen and (min-width: 64.063em) and (max-width: 90em) { }

/* XLarge screens: min-width 1441px, xlarge screens */
@media only screen and (min-width: 90.063em) { }
/* min-width 1441px and max-width 1920px */
@media only screen and (min-width: 90.063em) and (max-width: 120em) { }

/* XXLarge screens: min-width 1921px, xxlarge screens */
@media only screen and (min-width: 120.063em) { }
```

Viewport festlegen

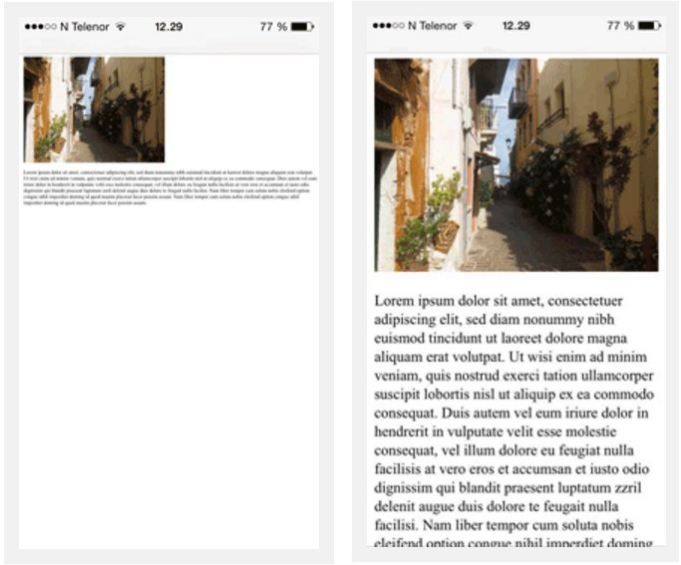
- Der Sichtbare Bereich der Useransicht

⇒ Viewport (Browser Fenster Inhalt)

- Browser skaliert „runter“ um eine Standard Webseite einzupassen

⇒ Unterschiedlich für verschiedene Geräte

⇒ Media Queries alleine **nicht ausreichend!**



Quelle: https://www.w3schools.com/css/css_rwd_viewport.asp

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Zusammenfassung und Fragen

Fragen

- Was sind Breakpoints
- Welche Breakpoints gibt es
- Wie könnte man ein spalten orientiertes Design entwerfen
- Wie können Bildschirmgrößen eingeteilt werden

Inhalte

- Umsetzung von RWD
- Breakpoints

Design Strategien

- Mobile First
- Card Layout
- Konkrete Beispiel Layouts (bottom-up)

Mobile First Strategie
durchgängiges Layout generieren

1. man baut das Design von Grund auf und startet mit der am meisten eingeschränkten Ansicht – den mobilen Geräten bzw. dem kleinsten Viewport
2. es notwendig den Inhalt auf das Wichtigste zu reduzieren
3. um so mehr Relevanz ein Inhalt hat um so weiter oben in der Seite sollte er stehen
4. oft „nur“ mehr Blöcke untereinander angeordnet
5. wenn nun der kleinste Viewport korrekt designend ist, dann kommt der nächst höhere dran – also nach Mobiler Ansicht, z.B. eine Tablet Ansicht, usw.
6. danach weiter bis man bei dem größten Viewport angekommen ist, also z.B. XXLarge Desktop

$\Rightarrow$ nutzen “cascading” Eigenschaft von CSS

Beispiele für RWD

- Bilder
  - Größe steuern
  - als Hintergrund
  - mit einfachem Textumbruch
- Card Design
- Menu
- Gesamt Layouts und Spaltendesign

Responsive Images

```
img {
    width: 100%;
    height: auto;
}

img {
    max-width: 100%;
    height: auto;
}
```

Responsive Images - Hintergrund

```
div {
    height: 400px;
    background-image:
        url('img_flowers.jpg');
    background-repeat:
        no-repeat;
    background-size: contain;
    border: 1px solid red;
}
```

Responsive Images - Hintergrund

```
div {
    height: 400px;
    background-image:
        url('img_flowers.jpg');
    background-size: 100% 100%;
    border: 1px solid red;
}
```

Responsive Images - Hintergrund


```
div {
    height: 400px;
    background-image:
        url('img_flowers.jpg');
    background-size: cover;
    border: 1px solid red;
}
```

Artikel mit Bild

```
img {
  float: left;
  width: 150px;
  display: inline-block;
  margin-right: 10px;
  margin-bottom: 5px;
}

<article>
  <h2>The Planet</h2>
  <img src="bild.jpg" alt="ein Bild">
  <p>Lorem ipsum …</p>
  <p>Lorem ipsum …</p>
</article>
```

The Planet

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

jsFiddle: https://jsfiddle.net/hirsche/vo7axht3/11/

Artikel Version 2 mit Bild

```css
* {
  box-sizing: border-box;
}
img {
  float: left;
  width: 20%;
  display: inline-block;
}
p {
  float: right;
  width: 80%;
  padding-left: 1rem;
  margin-top: 0;
}
h2 {
  float: left;
  padding-left: 1rem;
  margin-top: 0;
}
```

The Planet

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

```html
<article>
  <img src="bild.jpg" alt="ein Bild">
  <h2>The Planet</h2>
  <p>Lorem ipsum ...</p>
  <p>Lorem ipsum ...</p>
</article>
```

Card Design

Card title

This is a longer card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

Last updated 3 mins ago

Card title

This card has supporting text below as a natural lead-in to additional content.

Last updated 3 mins ago

Card title

This is a wider card with supporting text below as a natural lead-in to additional content. This card has even longer content than the first to show that equal height action.

Last updated 3 mins ago

Quelle: https://getbootstrap.com/docs/4.1/components/card/#card-decks

Cards

Wie sehen CARD aus

- Hervorgehobene Bilder oder Medieninhalt
- Titel
- Autor
- Zusammenfassung
- Datum
- Kategorie
- Social Share Links
- „Read More“ Button

Was sind CARD

- Für sich abgeschlossen
- unabhängig
- individuell

Zum Nachlesen: https://material.io/design/components/cards.html

Cards Vorteile

- Ist „automatisch“ Responsive
- Simpel und übersichtlich
- Leicht zu organisieren – man kann sich auf Details spezialisieren ohne das Gesamtlayout zu beeinflussen
- Einfach zu lesen
- Gut geeignet für Medien-Inhalte (Bilder, Videos)
- Kann je nach Darstellungsposition bzw. –ort einfach „resized“ werden

Wann sollten Cards nicht eingesetzt werden

- als reiner Layout-Mechanismus
- für sehr bereite Layouts
- „zu viel“ Inhalt
- eher weniger für medienarme Inhalte
- für Elemente die viele „Desktop“-Features beinhalten

Cards Beispiel

```
<div class="card">
  <img src="car.jpg" alt="car">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some ...</p>
    <a href="#" class="btn">Go...</a>
  </div>
</div>
```

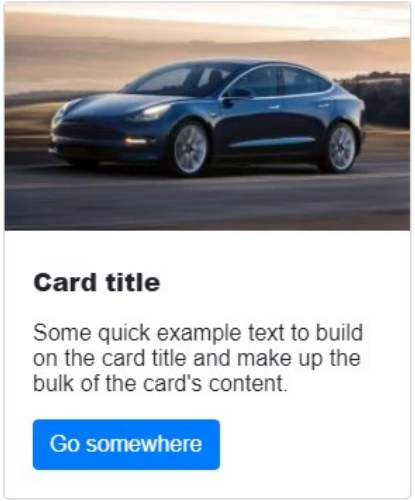
```
.btn {
    color: #fff;
    background-color: #007bff;
    display: inline-block;
    font-weight: 400;
    text-align: center;
    text-decoration: none;
    padding: 0.375rem 0.75rem;
    font-size: 1rem;
    line-height: 1.5;
    border-radius: 0.25rem;
}
.card {
  border: 1px solid lightgray;
  margin-top: 10px;
  margin-right: 10px;
  border-radius: 0.25rem;
  font-family: Arial;
  width: 18rem;
}
```

Cards Beispiel

```html
<div class="card">
  <img src="car.jpg" alt="car">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some …</p>
    <a href="#" class="btn">Go …</a>
  </div>
</div>
```

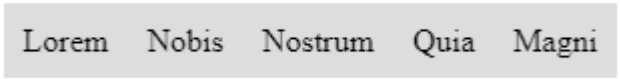
```css
.card img {
  width: 100%;
  display: block;
}
.card h5 {
  font-weight: 800;
  font-size: 1.1rem;
  margin: 0;
}

.card-body {
  color: #212529;
  text-align: left;
  padding: 20px;
}
```



jsFiddle: https://jsfiddle.net/hirsche/v27kzrsm/34/

Menü




```
<nav class="navigation">
  <ul class="menu">
    <li><a href="#">Lorem</a></li>
    <li><a href="#">Nobis</a></li>
    <li><a href="#">Nostrum</a></li>
    <li><a href="#">Quia</a></li>
    <li><a href="#">Magni</a></li>
  </ul><!-- .menu -->
</nav><!-- .navigation -->
```

```
.navigation {
  display: inline-block;
}

/* Listeneigenschaften ändern */
.menu,
.sub-menu {
  margin: 0;
  padding: 0;
}
.navigation .menu {
  list-style: none;
  display: flex;
}
```

jsFiddle: https://jsfiddle.net/hirsche/s9gqjtpe/75/

Menü

Lorem Nobis Nostrum Quia Magni

```
<nav class="navigation">
  <ul class="menu">
    <li><a href="#">Lorem</a></li>
    <li><a href="#">Nobis</a></li>
    <li><a href="#">Nostrum</a></li>
    <li><a href="#">Quia</a></li>
    <li><a href="#">Magni</a></li>
  </ul><!-- .menu -->
</nav><!-- .navigation -->
```

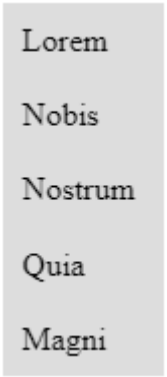
jsFiddle: https://jsfiddle.net/hirsche/s9gqjtpe/75/

```
/* Menueinträge umformen */
.navigation ul.menu li {
  margin: 0;
  background-color: #DDD;
}
.navigation ul.menu a
{
  /*
    einem Link wird die Farbe
    nicht von Haus aus vererbt
  */
  color: inherit;
  text-decoration: none;
  display: inline-block;
  padding: 0.6rem;
}
.navigation ul.menu li:hover {
  color: #DDD;
  background-color: #222;
}
```

Menü

```
<nav class="navigation">
  <ul class="menu vertical">
    <li><a href="#">Lorem</a></li>
    <li><a href="#">Nobis</a></li>
    <li><a href="#">Nostrum</a></li>
    <li><a href="#">Quia</a></li>
    <li><a href="#">Magni</a></li>
  </ul><!-- .menu -->
</nav><!-- .navigation -->
```

```
.navigation .menu.vertical {
  width: 5rem;
  display: inline-block;
}
```



jsFiddle: https://jsfiddle.net/hirsche/s9gqjtpe/75/

Burger – Menü

```
<nav>
 <div class="burger">
   <a>=Menu=</a>
 </div>
 <ul class="menu">
   <li><a href="#home">Home</a></li>
   <li><a href="#news">News</a></li>
   <li><a href="#contact">Contact</a></li>
   <li><a href="#about">About</a></li>
 </ul>
</nav>
```

=Menu= Home News Contact About

Home
News
Contact
About

```
.burger {
  user-select: none;
}
ul.menu {
  list-style: none;
  margin: 0;
  padding: 0;
  display: none;
}
nav > .menu:hover, nav > .burger:active ~ .menu,
nav > .burger:hover ~ .menu { display: block; }


@media screen and (min-width:36.5em) {
  nav > .burger {
    display: none;
  }
  nav > .menu {
    display: block;
  } nav > .menu li {
    display: inline;
  }
}
```

https://jsfiddle.net/bidsoft/67tgkps9/4/

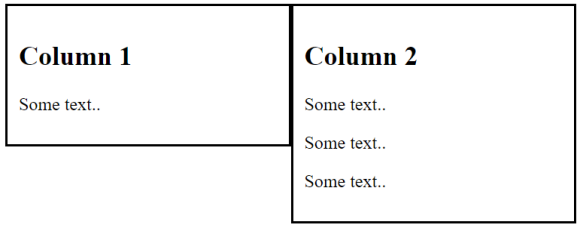
Üblicherweise jedoch in Kombination mit JavaScript umgesetzt

Spalten/Column Layout mit float

„float“ für mobile notwendig?

```
<div class="row">
  <div class="column">
    <h2>Column 1</h2>
    <p>Some text..</p>
  </div>
  <div class="column">
    <h2>Column 2</h2>
    <p>Some text..</p>
  </div>
</div>
```

```
* { box-sizing: border-box; }
.column{
  border-style: solid;
  float: left;
  width: 100%;
  padding: 10px;
}
.row:after {
  content: "";
  display: table;
  clear: both;
}
@media screen and (min-width: 480px) {
  .column { width: 50%; }
}
```



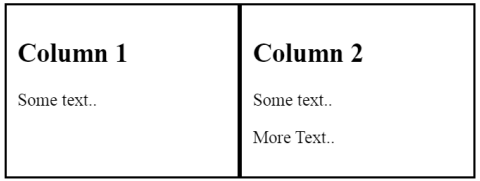
Nicht optimal für unterschiedlich hohe Spalten

https://jsfiddle.net/hirsche/kqt4Lvo1/

Spalten/Column Layout mit Flexbox

```html
<div class="row">
  <div class="column">
    <h2>Column 1</h2>
    <p>Some text..</p>
  </div>
  <div class="column">
    <h2>Column 2</h2>
    <p>Some text..</p>
    <p>More text..</p>
  </div>
</div>
```

```css
* { box-sizing: border-box; }
.column{
  border-style: solid;
  padding: 10px;
}
@media screen and (min-width: 480px) {
  .row{
    display: flex;
    flex-wrap: wrap;
  }
  .column{ flex: 50%; }
}
```



https://jsfiddle.net/hirsche/Lbke6xza/14/

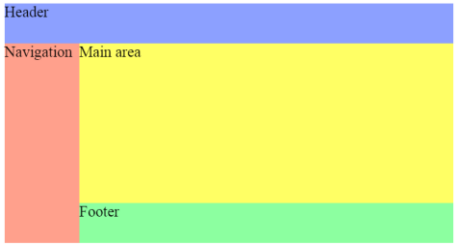
Grid Layout using CSS Grid

```html
<div class="container">
  <header>Header</header>
  <nav>Navigation</nav>
  <main>Main area</main>
  <footer>Footer</footer>
</div>
```

```css
.container {
  display: grid;
  grid-template-areas:
                  "head"
                  "nav"
                  "main"
                  "foot";
  grid-template-rows: 1fr 1fr 8fr 1fr;
  grid-template-columns: 1fr; }
```

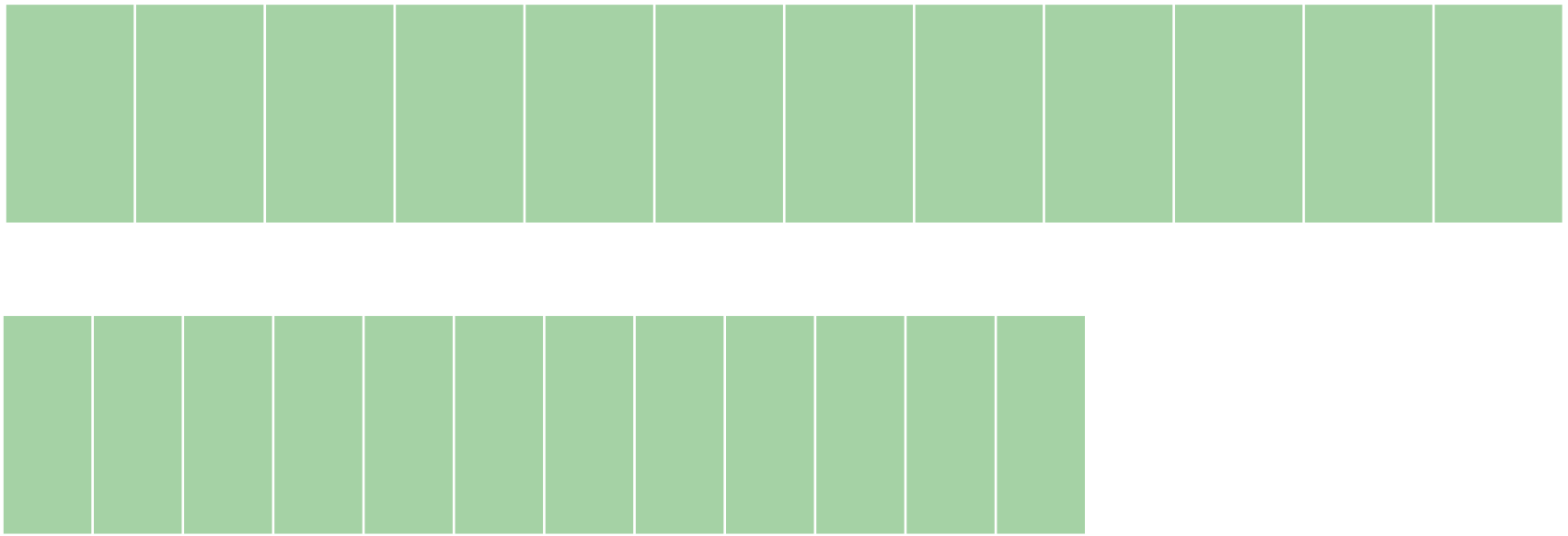
```css
.container > header {
  grid-area: head;
  background-color: #8ca0ff;
}
.container > nav {
  grid-area: nav;
  background-color: #ffa08c;
}
.container > main {
  grid-area: main;
  background-color: #ffff64;
}
.container > footer {
  grid-area: foot;
  background-color: #8cffa0; }
```

```css
@media screen and (min-width: 30.06em) {
  .container {
    grid-template-areas:
                  "head head"
                  "head head"
                  "nav main"
                  "nav foot";
    grid-template-columns: 1fr 5fr;
    grid-template-rows: 1fr 1fr 10fr 1fr;
  }
}
```

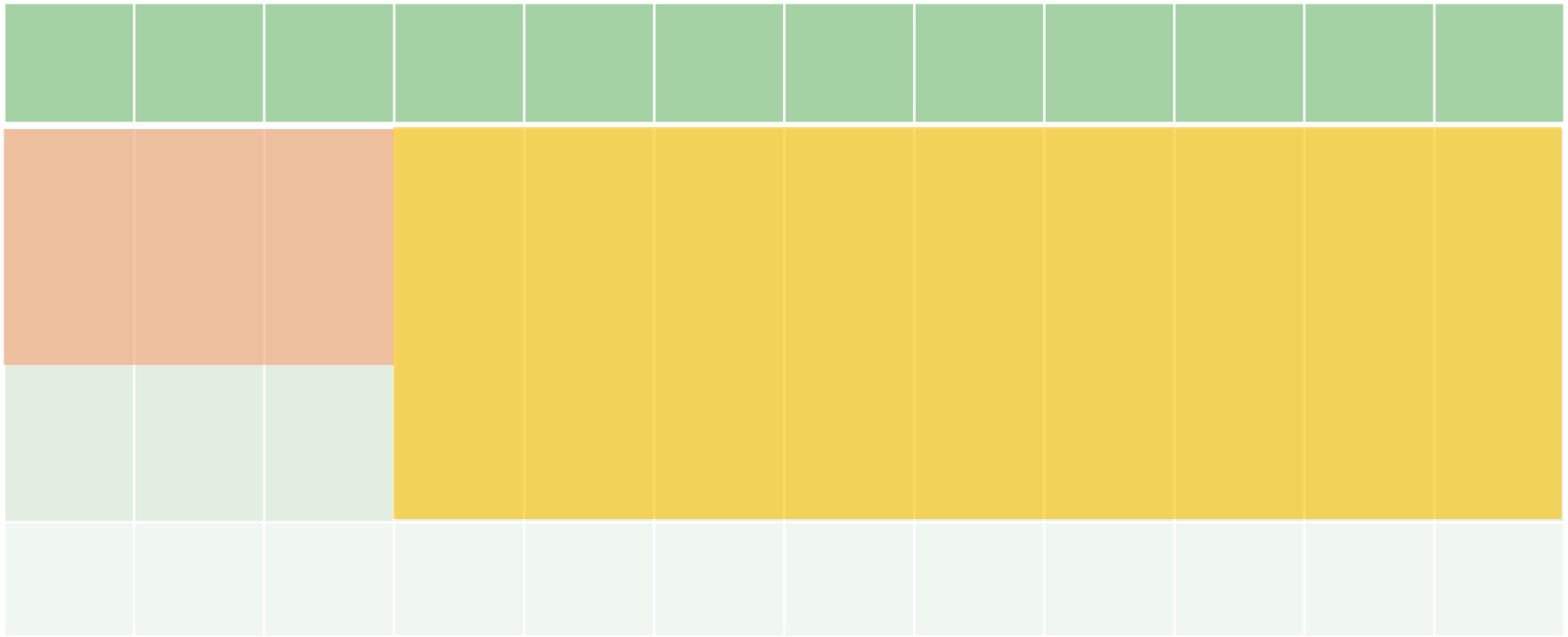


https://jsfiddle.net/bidsoft/kben3zw6/31/

Spalten/Column Layout



Grid Layout



https://getbootstrap.com/docs/4.0/layout/grid/

(Bootstrap) Grid

Spalten Definition mit Klassen:

col-b-n

b … breakpoint für eine bestimmte Bildschirmbreite

n … Gesamtzahl der Spalten

Bootstrap Grid

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-

Bootstrap Grid

```
<div class="container">

  <div class="row">
    <div class="col-sm-4">.col-sm-4</div>
    <div class="col-sm-4">.col-sm-4</div>
    <div class="col-sm-4">.col-sm-4</div>
  </div>

</div>
```

- **container** oder **container-fluid** um die Art des Elternelementes zu bestimmen
- **row** um eine neue Zeile zu beginnen
- **col-sm-4** um eine Spalte zu definieren die einem drittel der Gesamtbreite entspricht. Wobei:
  - **col** … für Spalte steht
  - **sm** … für die „getroffene“ Displaygröße
  - **4** … für einen Teiler von 12

Bootstrap Grid

```
<div class="row">
  <div class="col-4">.col-4</div>
  <div class="col-4">.col-4</div>
  <div class="col-4">.col-4</div>
</div>

<div class="row">
  <div class="col-sm-4">.col-sm-4</div>
  <div class="col-sm-4">.col-sm-4</div>
  <div class="col-sm-4">.col-sm-4</div>
</div>
```

```
<div class="row">
  <div class="col-md-4">.col-md-4</div>
  <div class="col-md-4">.col-md-4</div>
  <div class="col-md-4">.col-md-4</div>
</div>

<div class="row">
  <div class="col-lg-4">.col-lg-4</div>
  <div class="col-lg-4">.col-lg-4</div>
  <div class="col-lg-4">.col-lg-4</div>
</div>

<div class="row">
  <div class="col-xl-4">.col-xl-4</div>
  <div class="col-xl-4">.col-xl-4</div>
  <div class="col-xl-4">.col-xl-4</div>
</div>
```

Bootstrap Grid

Extra Small Screen

.col-4	.col-4	.col-4
.col-sm-4		
.col-sm-4		
.col-sm-4		

Small Screen

.col-4	.col-4	.col-4
.col-sm-4	.col-sm-4	.col-sm-4
.col-md-4		
.col-md-4		
.col-md-4		

Extra Large Screen

.col-4	.col-4	.col-4
.col-sm-4	.col-sm-4	.col-sm-4
.col-md-4	.col-md-4	.col-md-4
.col-lg-4	.col-lg-4	.col-lg-4
.col-xl-4	.col-xl-4	.col-xl-4

Large Screen

.col-4	.col-4	.col-4
.col-sm-4	.col-sm-4	.col-sm-4
.col-md-4	.col-md-4	.col-md-4
.col-lg-4	.col-lg-4	.col-lg-4
.col-xl-4		
.col-xl-4		
.col-xl-4		

RWD/AWD frameworks

- Twitter Bootstrap
- Zurb Foundation
- Google:
  - Material Framework
  - Leaf
- Materialize
- Simple Grid (reines Grid System)

https://colorlib.com/wp/free-css3-frameworks/

Zusammenfassung und Fragen

Fragen

- Wie teil Bootstrap die Spalten ein, warum
- Welche Strategie verfolgt man normalerweise wenn man eine Responsive Webseite gestallten möchte
- Welche Möglichkeiten kennen Sie um das Gesamtlayout einer Webseite umzusetzen?
  - Setzten Sie ein Beispiel in Code um
- Was sind die Vorteile bzw. Nachteile von Cards
- Welche Frameworks kennen Sie, die Sie bei dem Aufbau einer Webseite unterstüzen

Inhalte

- Designstrategien
- Mobile First
- Beispiele
  - Cards
  - Menüs (Burger)
  - Grid & Column Layouts
  - …
- Frameworks: Bootstrap