



# Cascading Stylesheets

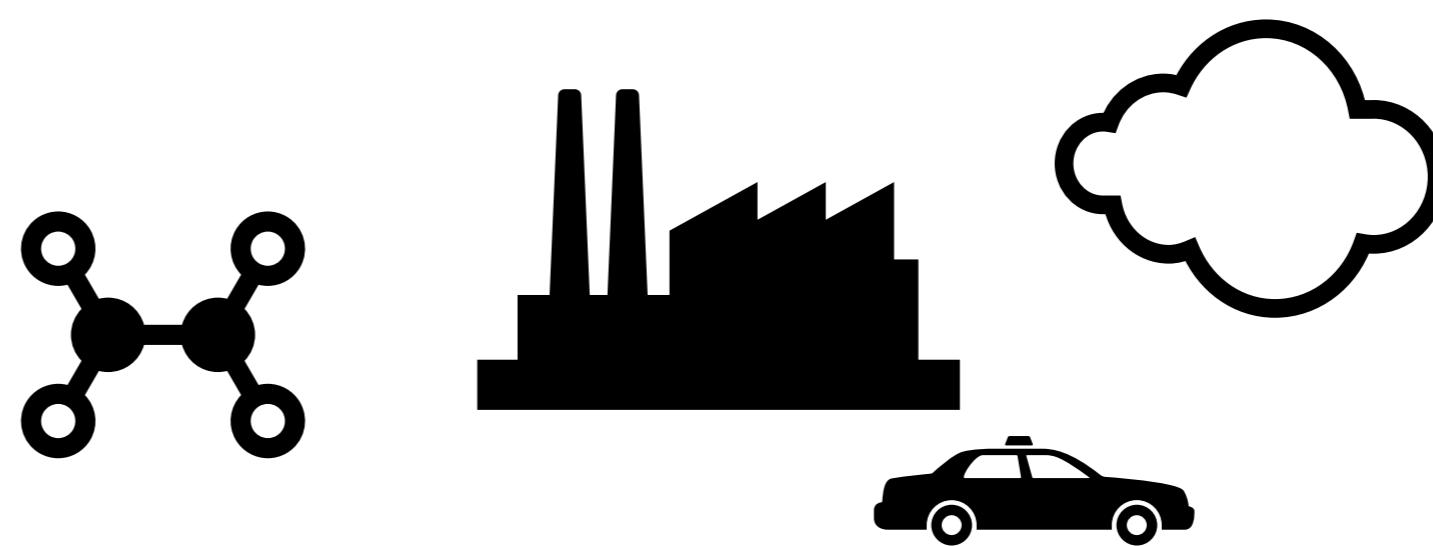
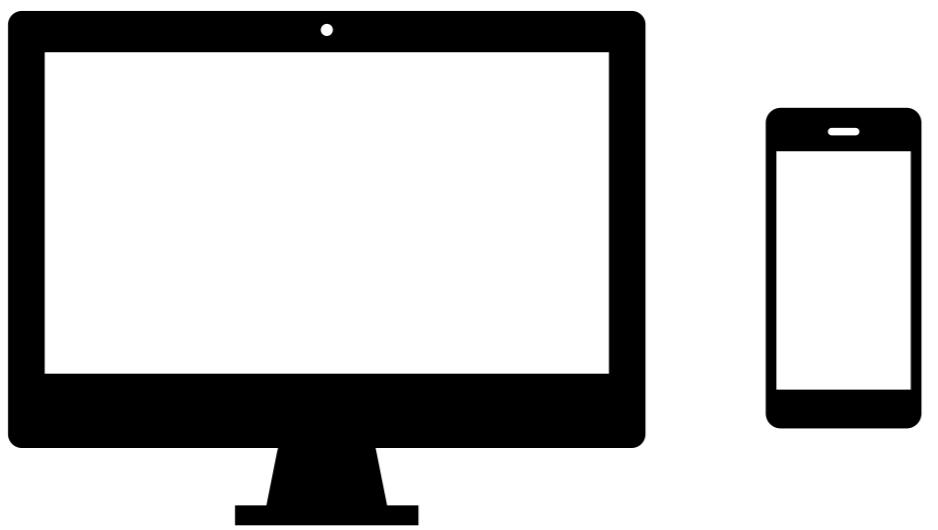
2024, Michael Reichart  
GFU Cyrus AG, Köln

Wir bilden weiter.

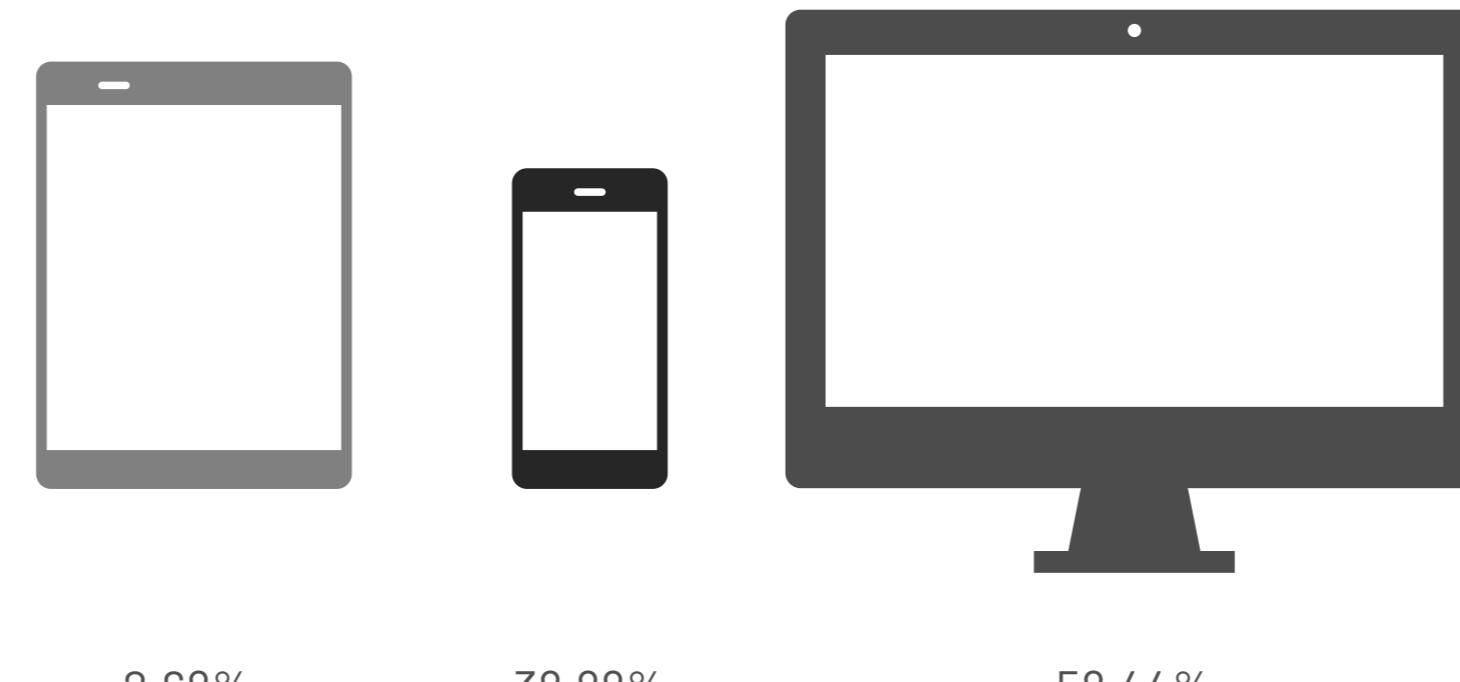
# Warum Stylesheets?

- + Stylesheets legen das Aussehen, sowie ein Teil des Verhaltens (Behaviour) eines HTML-Dokuments fest.
- + So bleibt das HTML Dokument als semantisches Struktur- und Inhaltedokument sauber und frei.
- + Völlig unabhängig vom Inhalt.

- + Aussehen und Verhalten von Information wird durch **verschiedene Stylesheets** an die jeweiligen Ausgabegerät und die jeweiligen Situationen angepasst.



## Usage of Mobile, Tablet and Desktop in Germany in September 2021



Source: <https://gs.statcounter.com>

<https://github.com/zenbox/workshop>

1. Repository herunterladen (Zip-Datei).
2. Datei entpacken und den Ordner 'workshop-master' umbenennen zu 'workshop'.
3. Den 'workshop'-Ordner über 'Open Folder...' in Visual Studio Code öffnen.
4. Extensions für VS Code:
  - Prettier - Code Formatter (Prettier)
  - Live Server (Ritwick Dey)
  - Live Sass Compiler (Glenn Marks)
  - Better Comments (Aaron Bond)

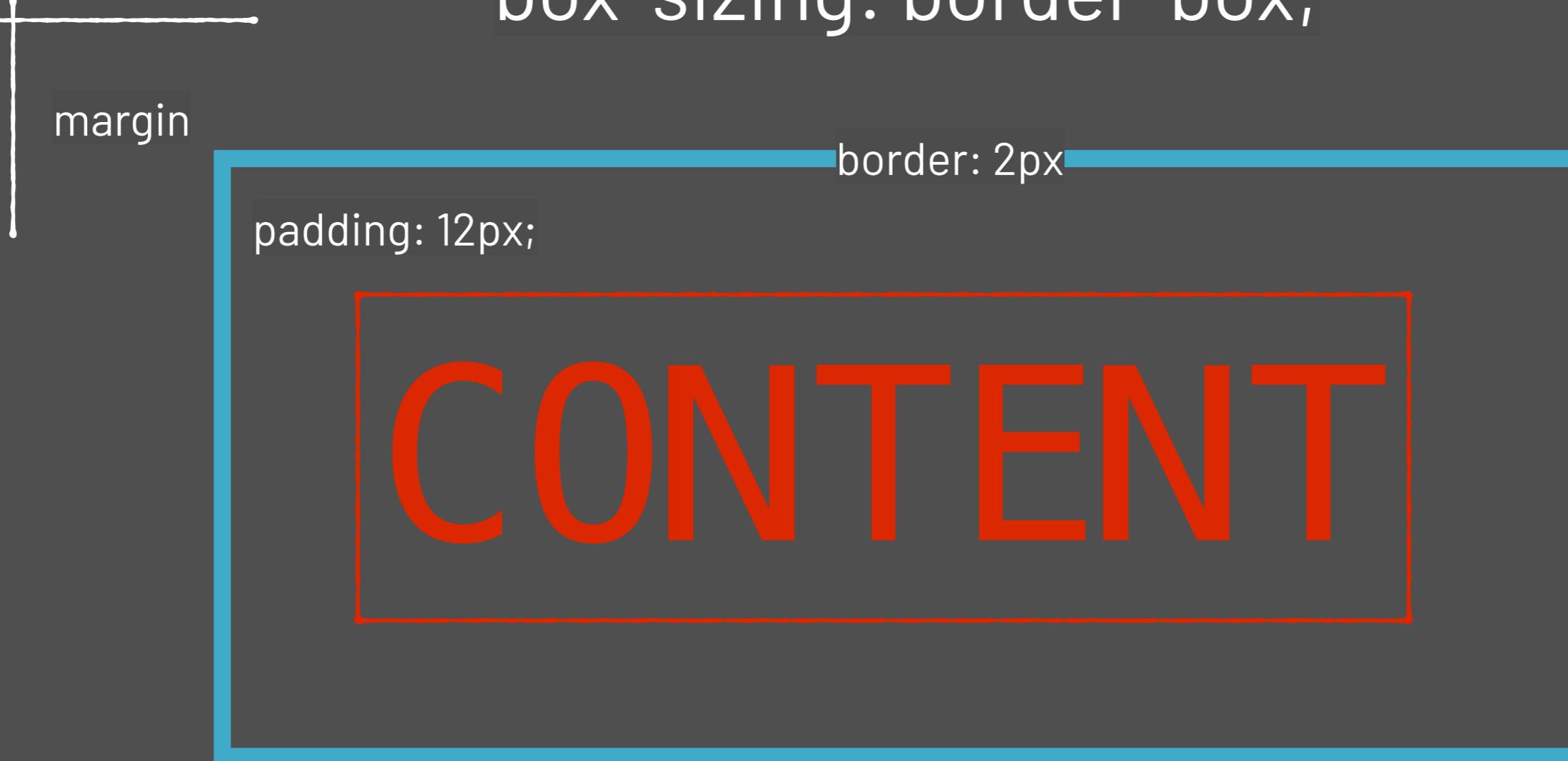
## Arbeitsumgebung herstellen

Die Daten liegen in meinem Git-Repository.

# Skizzen

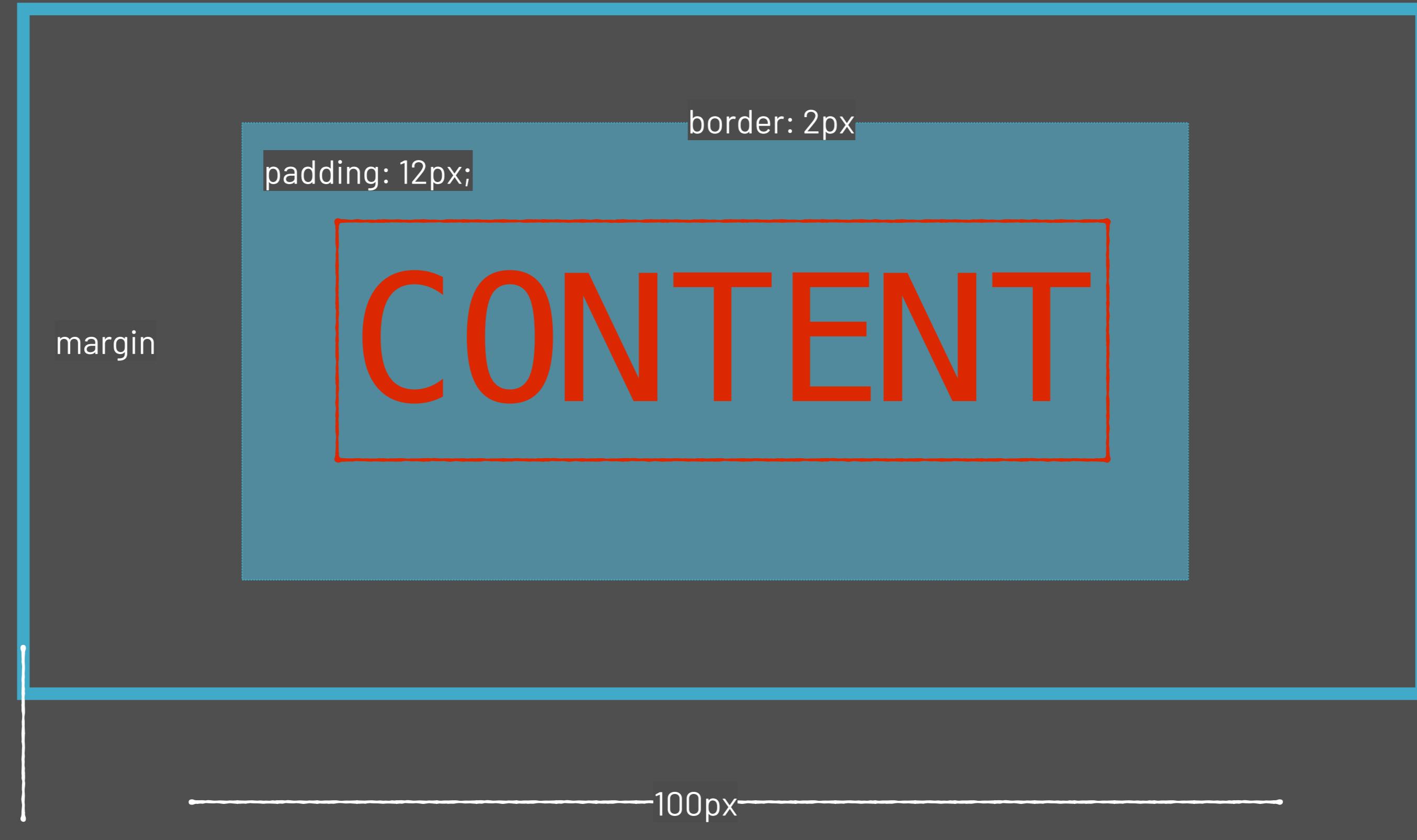
```
display: block;  
width: 100px;  
padding: 12px;
```

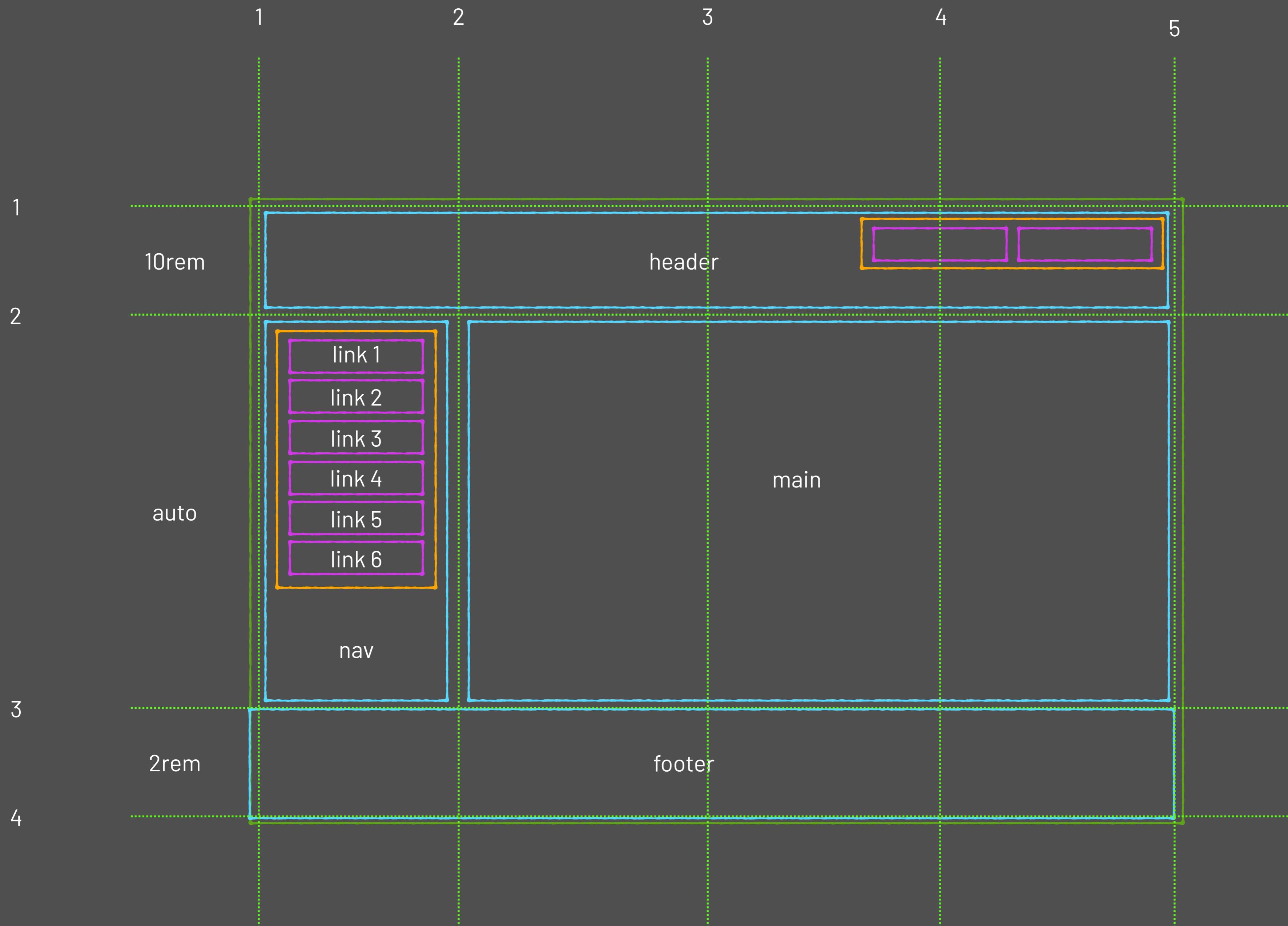
box-sizing: border-box;

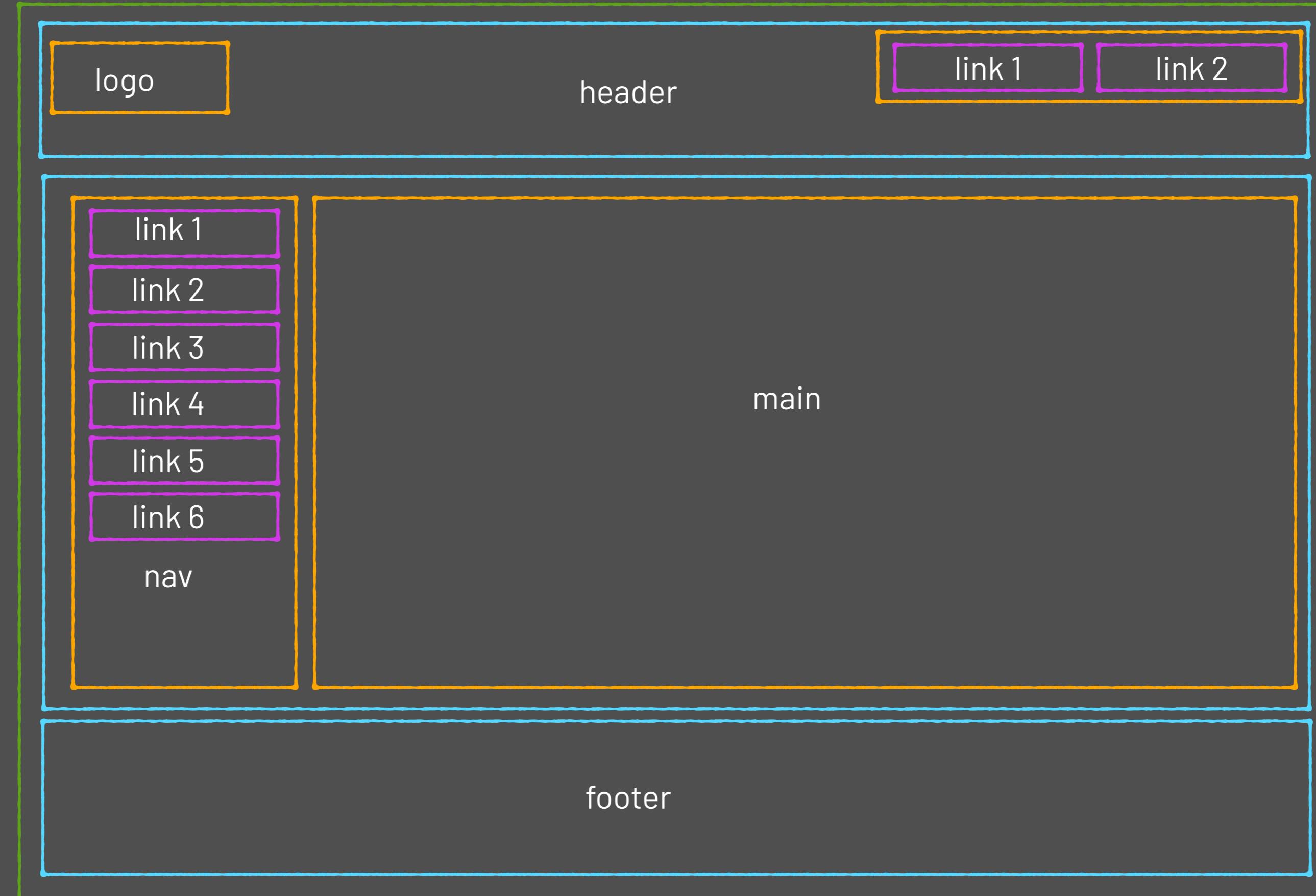


```
width: 100px;  
incl. padding: 2 * 12px ;  
incl. border: 2 * 2px ;
```

100px!!

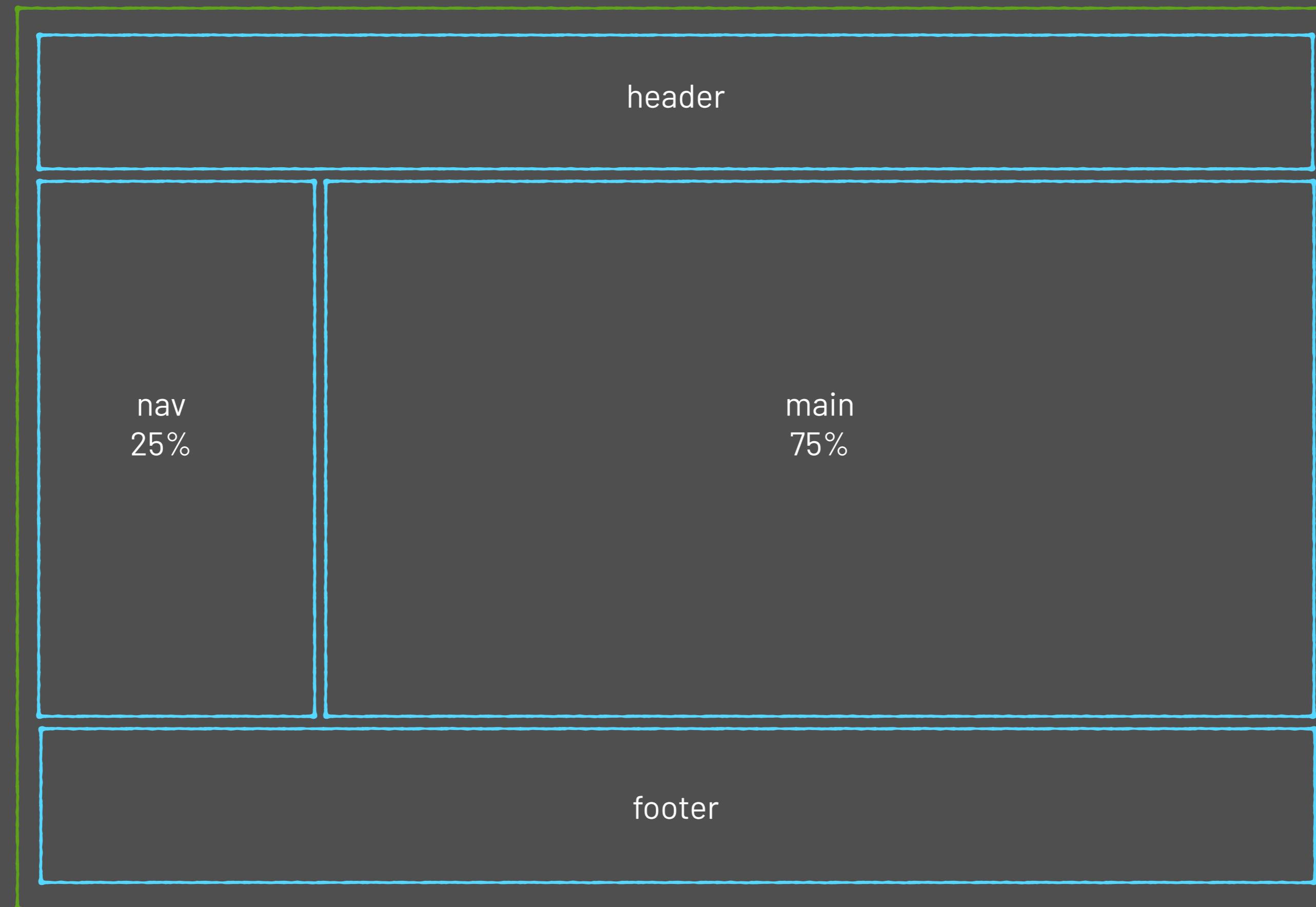






Ein Template bauen: '\_template-1.scss'

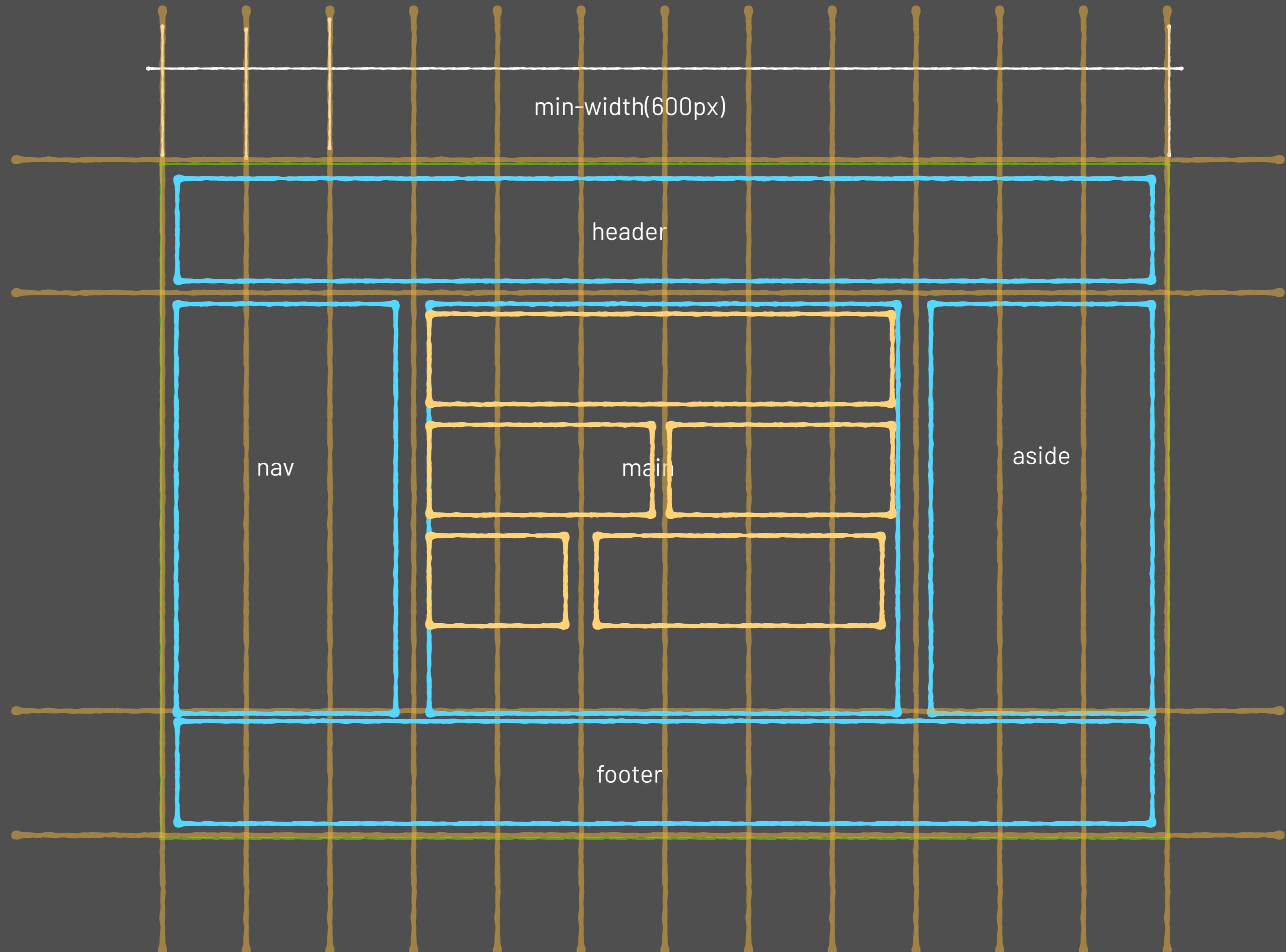
wahlweise mit 'flex' oder mit 'grid'

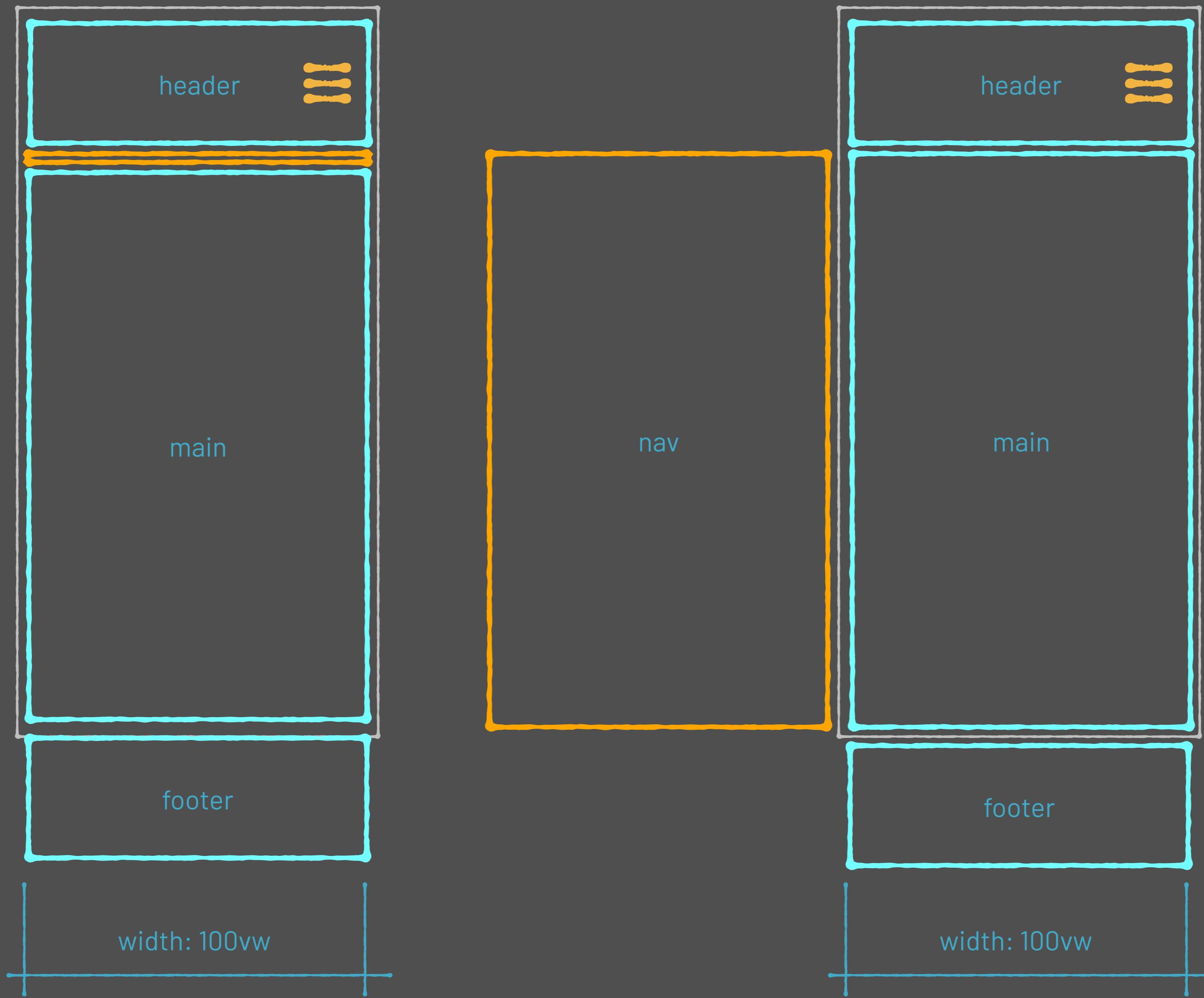


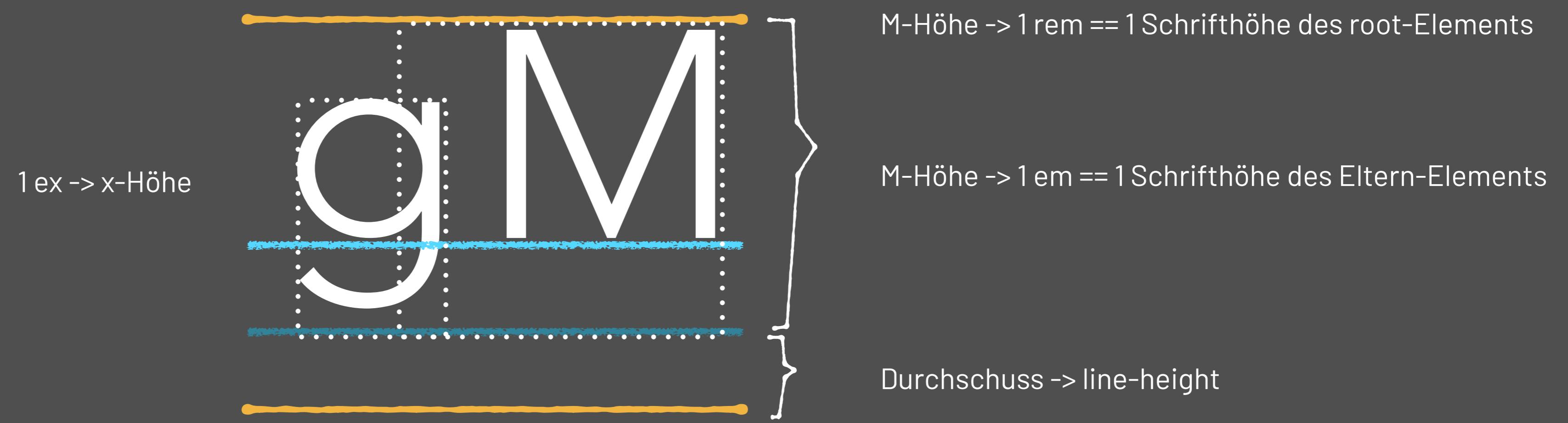
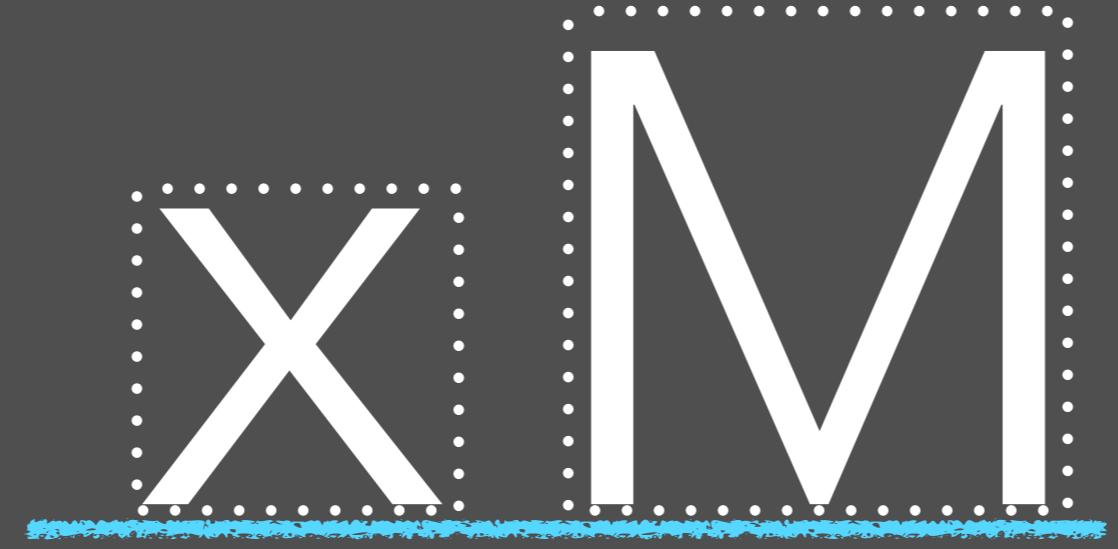












# CSS Syntax

# CSS besteht aus Deklarationen

```
body {  
    background-color : rgb(0, 0, 200);  
}
```

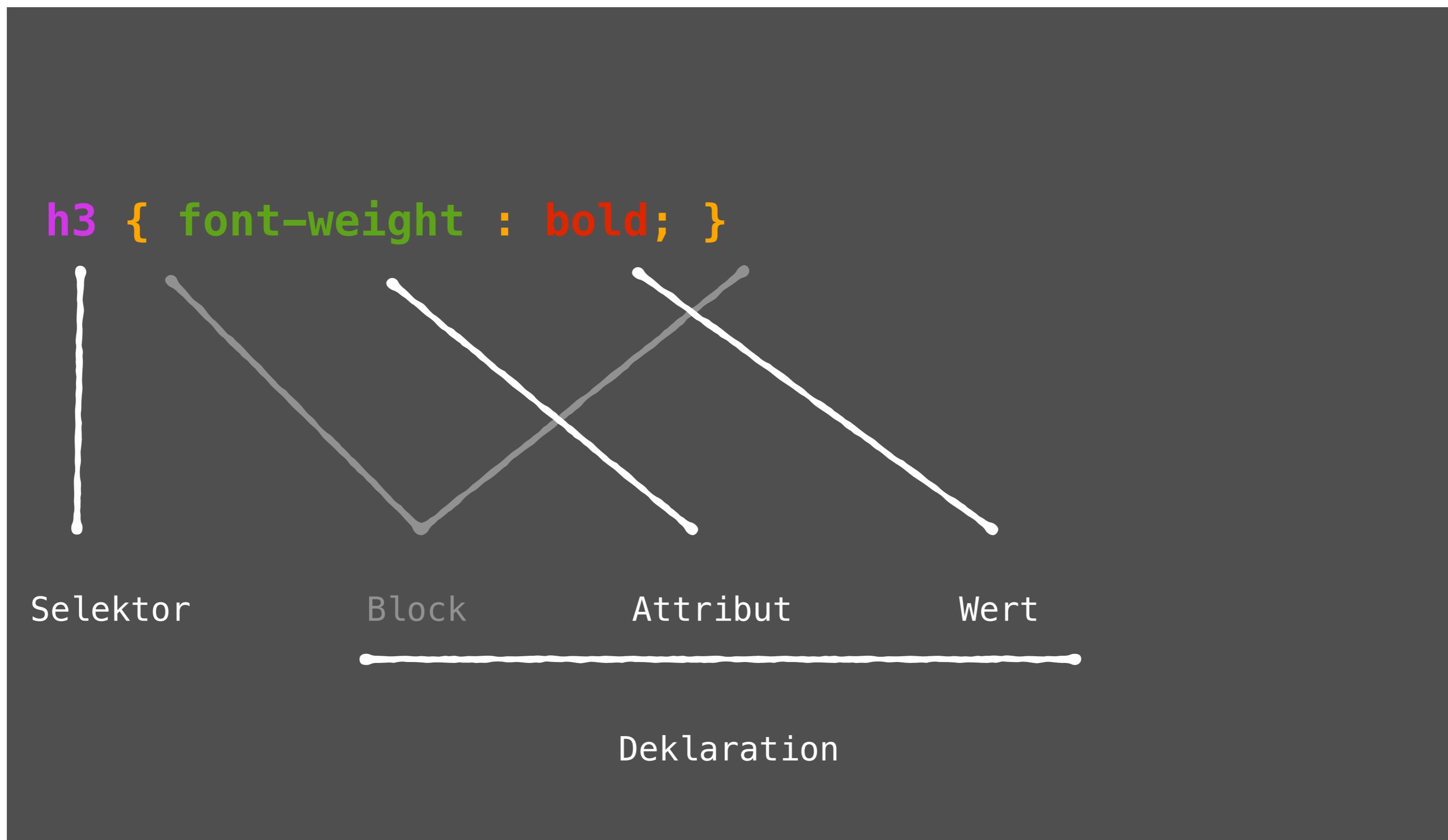
# Jede beliebige Eigenschaft ist ansprechbar.

```
background-color : rgba(0,0,0,0.15);  
background-image : url(meinbild.jpg);  
background-repeat : no-repeat;  
background-attachment : fixed;  
background-position : 10px 10px;
```

# Eigenschaften für Text, Schrift und Schriftverhalten (keinesfalls vollständig).

color	-> Textfarbe
direction	-> Schreibrichtung
line-height	-> Zeilenabstand
letter-spacing	-> Buchstabenabstand
unicode-bidi	-> Bidirektionalität
vertical-align	-> Ausrichtung, senkr.
white-space	-> Leerzeichenanzeige
word-spacing	-> Wortabstand
text-align	-> Textausrichtung
text-decoration	-> Textlinie
text-indent	-> Einrückung
text-shadow	-> Schatten
text-transform	-> Kapitälchen
font-family	-> Schriftart
font-size	-> Schriftgrösse
font-weight	-> Schriftgewicht

# Aufbau einer Deklaration



# Werteangaben, Maßsysteme & Wertebereiche

```
display: inline;  
  
font-family:  
    Arial, "Times New Roman";  
  
font-size:  
    0 | 12pt | 10pc | 13px  
    1em | 1ex | 1rem | 0.5ch  
  
width: 1in | 2.54cm | 25.4mm | 96px  
  
width: 100% | 100vh | 50 vw | 50vmin | 100vmax | 1fr
```

# Schriftgrößen

```
html { font-size : 16px;  
      line-height : 1.3 }  
  
h1 { font-size : 2rem; }  
h2 { font-size : 1.8 rem; }  
p { font-size : 1rem; }
```



# Box-Größen

```
div.header { width: 100vw; }

div.header { width: 1024px; }

body { width: 1024px; }
div.header { width: 100%; }
```



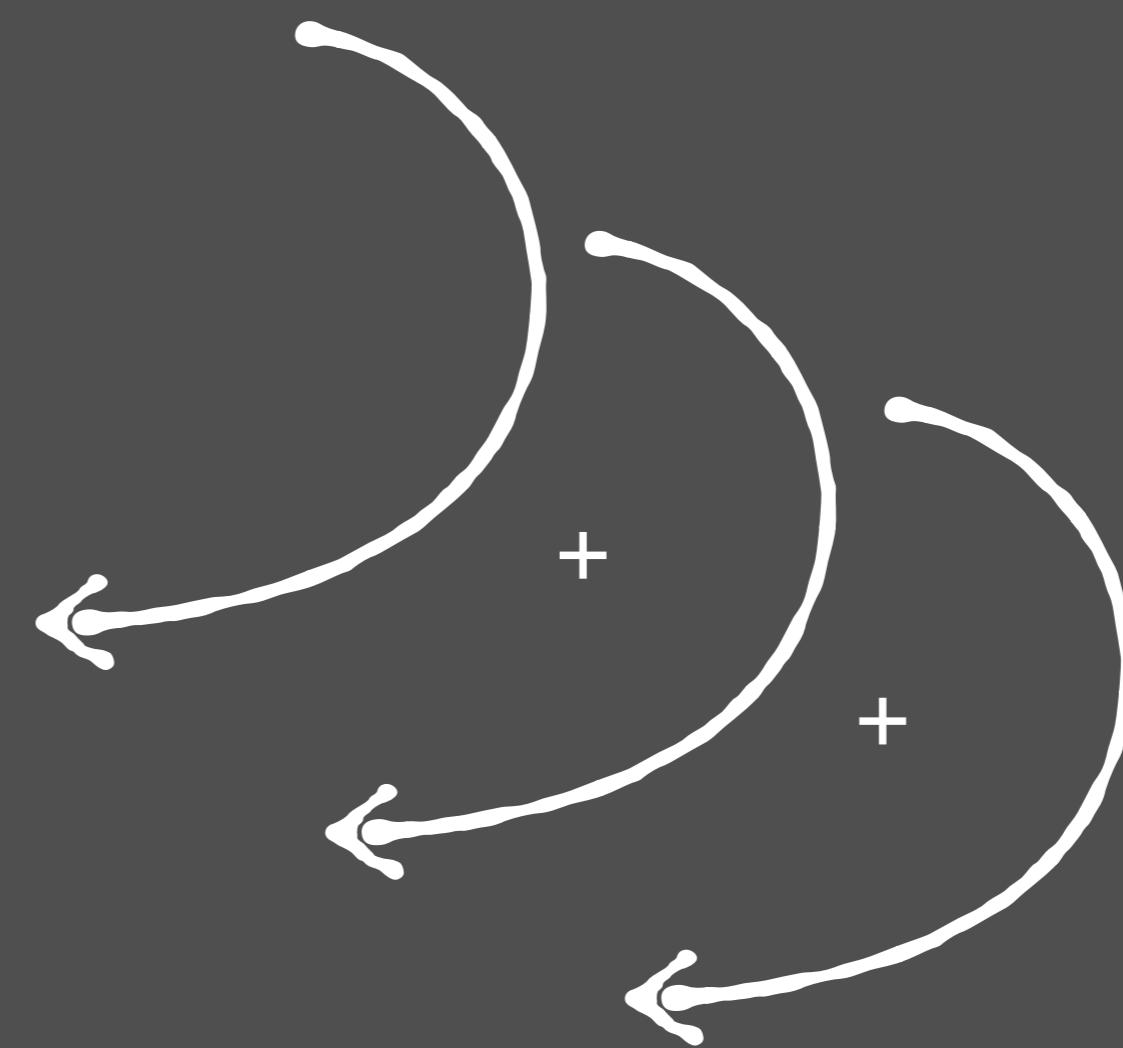
# Zusammenfassen mehrerer Anweisungen

```
h1 { font-weight: bold; }          font-weight : bold;  
h1 { font-size: 12pt; }           font-size   : 12pt;  
h1 { line-height: 14pt; }         font-family : Arial;  
h1 { font-family: Arial; }        font-style   : normal;  
h1 { font-style: normal; }       line-height : 14pt;  
}
```



# Ergänzen durch Kaskadierung

```
html { font-family : Arial;  
      font-size : 16px; }  
  
h1 { font-size : 2rem; }  
  
p { font-size : 0.75rem; }  
  
<html>  
  <body>  
    <h1>...</h1>  
    <p>...</p>  
  </body>  
</html>
```



# Kurzschreibweisen

```
h1 {  
    font-family: Arial, Verdana, sans-serif;  
    font-size: 12pt;  
    line-height: 14pt;  
    color: #000000;  
}  
  
h1 {  
    font: Arial 12pt/14pt #000;  
}
```

# Kurzschreibweisen

```
h1 {  
    margin-top      : 1em;  
    margin-right    : 1em;  
    margin-bottom   : 1em;  
    margin-left     : 1em;  
}  
  
h1 {  
    margin : 1em; /* TRBL */  
}  
h1 {  
    margin : 1em 2em; /* TB RL */  
}  
h1 {  
    margin : 1em 0 0 2em; /* T R B L */  
}
```

# Kaskadierende externe CSS-Dateien

```
@import base.css;  
@import typography.css;  
@import module.css;  
@import skin.css;
```

# Einbindung in HTML

# Extern: CSS-Datei via <link>

```
<link href="assets/css/style.css" rel="stylesheet">  
  
// mehrfach verwendbar  
// Gerätespezifisch  
// Möglichkeit der Modularität  
// Standard!
```

# Interner Stylecontainer

```
<style>
    h1 { font-family: Arial; }
</style>

// einfach, nicht mehrfach verwendbar
// Anwendungsspezifisch.
// Möglichkeit der punktuellen, temporären
Ergänzung.
// Einbindung durch dynamischen Nachladen!
// Kann wieder entfernt werden.
```

# Inlinestyles

```
<nav style="display : block;">
    Navigation
</nav>

// Einfach und elementenbezogen verwendbar
// Anwendungsspezifisch.
// Möglichkeit der punktuellen/temporären Ergänzung.
// Einbindung für dynamisches Nachladen!
// Kann wieder entfernt werden.
// Wird dynamisch und zeitlich begrenzt,
// zum Beispiel von jQuery, erzeugt.
```

## USER AGENT STYLESHEET

ZURÜCKSETZEN – reset.css, normalize.css  
FRAMEWORKS: bootstrap.css

BASE, LAYOUT, RESPONSIVE, MODULES, THEMES  
– Mehrere externe Stylesheets – style.css, concludis.css

AJAX – Dynamische temporäre interne Styleelemente  
`<style> ... </style>`

DOM – ELEMENTEBENE  
Durch Javascript dynamisch verwaltete  
Inlinestyles – `style="background:red; width:50%;"`

# Skalierbares, modulares CSS

# Stylesheets strukturiert aufbauen

- + CSS können jedes Element und jede Eigenschaft einer Weboberfläche ansprechen und gestalten.
- + CSS ist eine deklarative Sprache.
- + CSS hat (noch) KEINE Struktur-Befehle.
- + Struktur muss konzeptionell erreicht werden.

# kategorisieren!

- + Die Aufgaben von Stylesheets können in immer gleiche Kategorien eingeteilt werden:
  - A. Grundsätzliches (base), Semantisches
  - B. Layout
  - C. Zustandsveränderungen
  - D. Inhaltliche Module
  - E. Projekt oder Kundenanpassungen

# Selektoren

# Selektoren und deren Spezifität (Gewichtung)

* {}	> Genereller Selektor	0
body {}	> Elementeselektor	1
[type=text] {}	> Attributselektor	10
[data-id]	> verkürzter Attributselektor	10
:hover {}	> Pseudoklasse (hier state)	10
.nav{}	> Klassenselektor	10
#my-id {}	> ID-Selektor	100

# Selektoren und was sie einfangen ...

body {}	-> <body></body>
p {}	-> <p></p>, <p></p>
.row {}	-> <div class="row open"></div>
:open {}	
[type=email] {}	-> <input type="email">
.icon::before {}	-> <span class="icon"></span>
a[href]:hover {}	-> <a href="..."></a>
section#data {}	-> <section id="data"></section>

# Genereller Selektor

- + \* Stimmt mit jedem Element überein.
- + Er besitzt eine Spezifität von 0 und wird daher von jeder anderen Styleanweisung überschrieben.

# Der Element Selektor

- + Sammelt alle Elemente (hier h1) aus einem Dokument ein.
- + `h1 {}`

Collection

<h1>

<h1>

<h1>

⋮

# Der ID Selektor

- + Eine ID muss im Dokument unique sein!
- + `section#content {}`  
`#content {}`

Collection

```
<section  
id="content">
```

# Der Klassenselektor

- + Klassen sind allgemeine Formatvorlagen. Sie können an Elemente gebunden werden.
- + `section.page-content {}`
  - `.page-content {}`

## Collection

```
<section  
class="page-  
content">
```

```
<section  
class="page-  
content">
```



# Attribut- selektoren

- + Sammelt alle Elemente ein, die ein Attribut (hier href) besitzen, ganz gleich, welchen Wert es hat.  
`+ a[href]`
- + Sammelt alle Elemente ein, die ein Attribut (hier type) besitzen, dessen Wert "text" ist.  
`+ input[type=text]`

```
<a href="...">
```

```
<a name="...">
```

```
<input type="text">
```

```
<input type="email">
```

# Attribut- selektoren

- + Liste von durch Kommas voneinander getrennten Werten, und einer dieser Werte ist gleich „bar“.
- + `div[class*=column]`
- + Liste mit durch Trennstriche voneinander getrennten Werten, die (von links) mit „en“ beginnen.
- + `[lang|=en]`

```
<div class="green  
module bold  
nicecolumn">
```

```
<figure  
lang="en | fr |  
de">
```

# Verneinung in Selektoren

- + `:not(.box){ ... }`
- + `:not(span){ ... }`
- + `h2:not(.teaser){ ... }`

# Verneinung in Attribut-Selektoren

- + `input[type=text]{ ... }`
- + `input[type!=text]{ ... }`
  
- + `a[href] { ... }`
- + `a[!href] { ... }`

# Pseudoklassen, States und Pseudoelemente

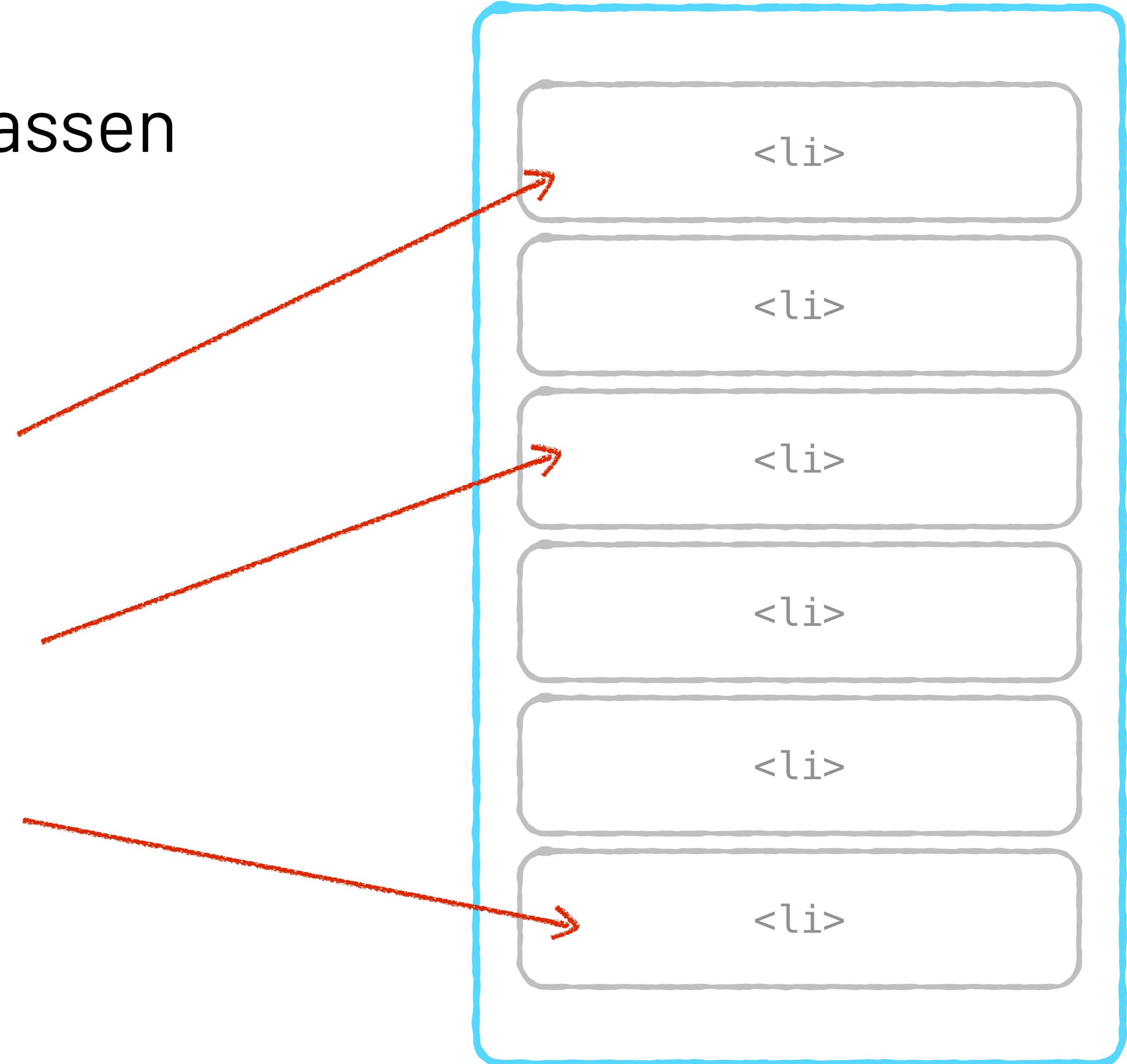
- + Pseudoklasse
  - + nav li:last-child { ... }
- + State Selector
  - + a[href]:hover { ... }
- + Pseudoelement
  - + .icon::before { ... }
- + Language Selector
  - + html:lang(de-DE) { ... }

# Pseudo-klassen

+ `ul li:first-child`

+ `ul li:nth-child(3)`

+ `ul li:last-child`



# Nth-Child

```
h2:first-child { ... }

h2:last-child { ... }

.row:nth-child(even) { /*Gerade Zeilen */
  background: #dde;
}
.row:nth-child(odd) { /* Ungeraden Zeilen */
  background: white;
}
.row:nth-child(5n) { /* Jede fünfte Zeile */
  background: #dde;
}
.row:nth-child(5n+1) { /* Jede fünfte Zeile */
  background: #dde;
}
```

# Nth-of-type()

```
div#test p:nth-of-type(4) { ... }  
div#test p:first-of-type { ... }  
div#test p:last-of-type { ... }  
div#test p:only-of-type { ... }
```

# Unterschied zwischen nth-child und nth-of-type

Mehrere Collections mit  
footer a:nth-child(odd)

```
<footer>
    <a href="">link</a>
    <span>Span</span>
    <a href="">link</a>
    <a href="">link</a>
    <span>Span</span>
    <span>Span</span>
    <a href="">link</a>
    <a href="">link</a>
    <a href="">link</a>
    <a href="">link</a>
</footer>
```

Eine Collection mit  
footer a:nth-of-type(odd)

```
<footer>
    <a href="">link</a>
    <span>Span</span>
    <a href="">link</a>
    <a href="">link</a>
    <span>Span</span>
    <span>Span</span>
    <a href="">link</a>
    <a href="">link</a>
    <a href="">link</a>
    <a href="">link</a>
</footer>
```

# State-Selektoren für Links

- + Nicht besuchter Link:
  - + a[href]:link
- + besuchter Link:
  - + a[href]:visited
- + Link, wenn die Maus darüber steht:
  - + a[href]:hover
- + Link, während er angeklickt wird:
  - + a[href]:active

# State-Selektoren für Formulare

- + `input:focus { ... }`
- + `input:blur { ... }`
- + `input:valid { ... }`
- + `input:invalid { ... }`
- + `input:optional { ... }`
- + `input:required { ... }`
- + `input:required:focus { ... }`
- + `input:required:hover { ... }`
- + `input:out-of-range { ... }`
- + `input:in-range { ... }`

<https://drafts.csswg.org/selectors-4/>

Selectors Level 4 Specification

# ::after, :: before für Pseudelemente

```
<span data-currency=„ €“>10</span>

span[data-currency]::after {
    content : attr(data-currency);
}

<span data-currency=„ €“>10 €</span>
```

# Toolipp per Pseudoelement

```
<span data-desc="tooltip text content">?</span> Text

span[data-descr] {
    position : relative;
    color    : #00F;
    cursor   : help;
    text-decoration : underline;
}

span[data-descr]:hover::after {
    content      : attr(data-descr);
    position     : absolute;
    left         : 0;
    top          : 24px;
    min-width   : 200px;
    background-color : #ffffcc;
    padding      : 12px;
    color        : #000000;
    font-size    : 14px;
    z-index      : 1;
    border       : 1px #aaaaaa solid;
    border-radius : 10px;
}
```

# Selektoren kombinieren

# CSS Selektoren Kombinationen

```
div.page-container {}          -> beschränkter Klassenselektor  
  
.group label {}              -> Descendantselektor  
.nav li {}                   -> Descendantselektor  
  
.nav > li > ul > li,  
ol.nav > li {}               -> Childselektoren-Kette  
  
h2 ~ p {}                    -> General Following Siblings  
h1 + p {}                     -> Immediate Following Sibling  
  
h1 ~ p:nth-of-type(1)         ->  
  
ul > li:first-child {}       -> numerischer Filter  
ul li:nth-child(odd) {}        -> logischer Filter  
  
.hamburger > *:nth-child(1)
```

# Descendant Selector

```
-> nav li { }
```

## Collection



```
<nav>
<ul>
  <li> </li>
  <li> </li>
  <li> </li>
</ul>
</nav>
```

# Child Selector

```
-> nav>ul>li { }
```

# Collection

```
<nav>
  <ul>
    <li> </li>
    <li> </li>
    <li><ul>
      <li> ... </li>
    </ul></li>
  </ul>
</nav>
```

~

General Sibling  
(All Followers)

-> h1 ~ p {}

Collection



```
<h1>
<p> </p>
<h2> </h2>
<p> </p>
<p> </p>
```

+

Adjactent Sibling  
(Immediate Follower)

-> h1 + p {}

# Collection

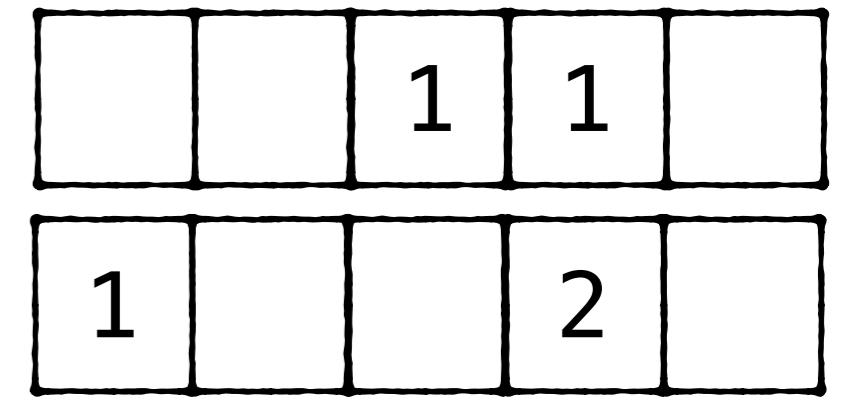
↳ <h1>  
<p> </p>  
<p> </p>  
<p> </p>

Collections: erst sammeln, dann filtern:  
Kombinationen sind lesbar, aber langsam.



1. alle href: ['href', 'href', 'href', 'href', 'href', 'href']
2. in <a>: ['href', 'href', 'href', 'href', 'href', ~~'href'~~]
3. in <li>: ['href', 'href', 'href', ~~'href'~~, 'href', 'href']
4. in <ul> ['href', 'href', 'href', ~~'href'~~, 'href', 'href']
5. in <nav>: ['href', 'href', 'href', ~~'href'~~, 'href', 'href']

# Gewichtungen (Spezifitäten)



# CSS Gewichtungen (Spezifitäten)

*	0 0 0 0 0	0
body	0 0 0 0 1	1
:pseudo	0 0 0 1 0	10
[href]	0 0 0 1 0	10
.my-class	0 0 0 1 0	10
#id	0 0 1 0 0	100
layer	- - - - -	- - -
<p style="">	0 1 0 0 0	1000
!important	1 0 0 0 0	10000

# CSS Gewichtungen

nav > ul > li	0 0 0 0 3
a[href]	0 0 0 1 1
a	0 0 0 0 1
[href]	0 0 0 1 0
#form-login	0 0 1 0 0
a[href] { ... !important; }	1 0 0 1 1

# CSS Gewichtungen

HTML:

```
<body class=„red“> ... </body>
```

CSS:

```
.red {  
    background-color : red;  
}  
  
body {  
    background-color : black !important;  
}
```

# CSS Gewichtungen

```
body { background-color: green; } 1!  
* {background-color: red; } 0!  
  
<body> // green!  
</body>
```

@layer

# CSS Layer

```
@layer first-layer-name {  
    .red { color: red; } // 1  
}  
  
@layer second-layer-name {  
    p { color: green; } // 1  
}
```

# Einführung: Was sind Kaskadenebenen?

- + CSS-Kaskadenebenen sind dazu gedacht, knifflige Probleme in CSS zu lösen.
- + *Problem: Spezifitätskonflikte eskalieren und wir müssen Stile aus anderen Bereichen unseres Codes (oder eines Drittanbieter-Tools) aufgrund von widersprüchlichen Selektoren außer Kraft setzen.*
- + Der Umgang mit Kaskadenkonflikten und Selektorenspezifität ist oft verwirrend. Kaskaden begründen stark auf Heuristiken, also eine in den Code eingebaute Vermutung oder Annahme.
- + Spezifität basiert zum Beispiel auf der Annahme, dass IDs, die nur einmal verwendet werden, wichtiger sind als wiederverwendbare Klassen und Attribute.
- + Das ist aber keine absolut zuverlässige Regel.  
Es fehlt Kontrolle.

# Kaskadenebenen bieten Kontrolle

- + Kaskadenebenen geben CSS-Autoren mehr direkte Kontrolle über die Kaskade.
- + Mit der @layer at-Regel und den @imports können wir unsere eigenen Kaskadenebenen erstellen.
- + **Ebenen sortieren Stile nach Priorität: niedrig für `reset` oder `base` zu höher wie `theme`, `framework` zu am höchsten wie `component` oder `theme`.**
- + Konflikte innerhalb einer Ebene werden über Spezifität augelöst.
- + **Konflikte zwischen Ebenen werden immer durch die Verwendung der Stile der höheren Ebene gelöst.**

# CSS Layer im CSS Dokument

```
@layer legacy {}

@layer base {
    p {  }
}

@layer service-provider {
    #anzeige p { !important} // 100101
}

@layer customer-theme {
    p {  } // 1
}

p {  } // non layered selectors win always!
```

# CSS Layer im CSS Dokument

```
// Just ordering the layers:  
  
@layer base, service-provider, customer-theme  
  
  
// Once ordered:  
// Reordering will be ignored:  
  
@layer reset, customer-theme, service-provider,  
      base, utilities;
```

# Import mit layer-Definition

```
// style.css
@layer base, service-provider, customer-theme, prefers

@import url(service-provider.css) layer(service-provider)
@import url(customer-theme.css) layer(customer-theme)
@import url(base.css) layer(base)
@import url(prefers.css) layer(prefers)
```

@layer haben weniger Gewicht  
als `not layered`-Deklarationen

```
@layer defaults {  
    a:any-link { color: orange; }  
}  
  
/* un-layered styles have the highest priority */  
a {  
    color: mediumvioletred;  
}
```

> a link to the end of the galaxy

# Verschachtelte Layer

```
// Ordering nested layer:  
@layer components, components.button  
  
// Writing nested layer  
@layer components {  
    p { }  
  
    @layer button {  
        Button { }  
    }  
}  
  
// Import as nested layer  
@import url('button.css') layer(components.button);
```

# Variablen, Funktionen

# CSS4 Variablen

```
// Variablen haben einen Scope,  
// `:root` oder `html` ist der globale Scope  
  
:root{  
    --myVariable : any-value;  
    --$primaryColor: #34fc76;  
    ...  
}  
  
// Mit `var()` aufrufen:  
.my-class {  
    color: var(--$primaryColor);  
}  
  
// CSS Variablen werden erst im Browser ausgewertet,  
// sie haben Zugriff auf aktuelle `viewport` Eigenschaften.
```

# `calc()`

```
// Berechnungen

body {
    margin-inline: 4rem;
    width : calc(100vw - 2 * 4 rem);
}
```

# Generisches CSS mit SASS

# Stylesheets generisch

- + Mit SASS (Syntactically Awesome Stylesheets) können Stylesheets "programmiert" werden.
- + Variablen halten wiederverwendbare Werte für Farben, Schriften, Breiten und mehr.
- + "Mixins" beinhalten wiederverwendbare Deklarationen und aufrufbare Funktionen.
- + Ein Compiler generiert aus den SASS-Codes CSS-Zeilen und legt diese in einem Verzeichnis ab.

# \_variables.scss

```
// LAYOUT
$default-padding : 0.5rem;
$default-margin  : 0;

// FONTS
$font-size      : 22px;
$line-height    : 1.8rem;

// COLORS
$text-hue       : 0;
$text-saturation: 0;
$text-lightness : 10%;
$text-alpha     : 1;

$color-hue      : 210;
$color-saturation: 80%;
$color-lightness : 50%;
$color-alpha    : 1;
```

# \_mixins.scss

```
@mixin list-reset {
  list-style: none;
  padding: 0;
  margin: 0;
}

@mixin shadow($d: 0.25rem, $r: 5px) {
  box-shadow: $d $d $r $shadow-color;
}

@mixin border-radius($radius) {
  -webkit-border-radius: $radius;
  -moz-border-radius: $radius;
  -ms-border-radius: $radius;
  border-radius: $radius;
}

.box {
  @include list-reset;
  @include shadow(0.25rem, 8px);
  @include border-radius(10px);
}
```

# @import

```
// _reset.scss
html, body, ul, ol {
  margin: 0;
  padding: 0;
}

// base.scss
@import 'reset';

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

# Mit Einrückungen arbeiten

```
nav {  
    ul {  
        margin: 0;  
        padding: 0;  
        list-style: none;  
    }  
  
    li {display:inline-block;}  
  
    a {  
        display: block;  
        padding: 6px 12px;  
        text-decoration: none;  
    }  
}  
  
nav ul {  
    margin: 0;  
    padding: 0;  
    list-style: none;  
}  
  
nav li {  
    display: inline-block;  
}  
  
nav a {  
    display: block;  
    padding: 6px 12px;  
    text-decoration: none;  
}
```

# Kompiler einstellen

```
$ cd path/to/web/directory  
$ sass --watch assets/scss:assets/css  
  
--style=expanded | compact | compressed | nested  
--sourcemap=None
```

# @function

```
@function add-numbers($first, $second) {  
  @return $first + $second  
}  
  
.box1 {  
  padding: add-numbers(5px, 10px);  
}
```

# @extend

```
.error {  
  border: 1px #f00;  
  background-color: #fdd;  
  
  &--serious {  
    @extend .error;  
    border-width: 3px;  
  }  
}  
  
<p class="error error--serious">lorem ipsum</p>
```

## @use - statt @import

```
// foundation/_lists.scss
ul, ol {
  text-align: left;

  & & {
    padding: {
      bottom: 0;
      left: 0;
    }
  }
}

// style.scss
@use 'foundation/lists';
```

## @use - Module

```
// src/_corners.scss           // style.scss
$radius: 3px;                 @use "src/corners";

@mixin rounded {              .button {
    border-radius:           @include
    $radius;                  corners.rounded;
}                                padding: 5px +
                                         corners.$radius;
}
```

# @forward

```
// src/_list.scss
@mixin list-reset {
  margin: 0;
  padding: 0;
  list-style: none;
}

// bootstrap.scss
@forward "src/list";

// styles.scss
@use "bootstrap";
```

```
li {
  @include
  bootstrap.list-reset;
}
```

# @if, @else

```
$light-background : #f2ece4;
$light-text       : #036;
$dark-background : #6b717f;
$dark-text       : #d2e1dd;

@mixin theme-colors($light-
theme: true) {
  @if $light-theme {
    background-color:
      $light-background;
    color: $light-text;
  } @else {
    background-color:
      $dark-background;
    color: $dark-text;
  }
}

.banner {
  @include
    theme-colors(
      $light-theme: true
    );
}

body.dark & {
  @include
    theme-colors(
      $light-theme: false
    );
}
```

# @each - mit `lists` und `maps`

```
$sizes: 40px, 50px, 80px;      $icons: (  
@each $size in $sizes {        "eye": "\f112",  
    .icon-#$size {                "start": "\f12e",  
        font-size : $size;          "stop": "\f12f"  
        height   : $size;          );  
        width    : $size;          @each  
    }                                $name, $glyph in $icons {  
}                                .icon-#$name:before {  
                                display:  
                                inline-block;  
                                font-family:  
                                "Icon Font";  
                                content: $glyph;  
                                }  
                                }
```

# @for

```
$base-color: #036;

@for $i from 1 through 3 {
  ul:nth-child(3n + #{$i}) {
    background-color: lighten($base-color, $i * 5%);
  }
}
```

# Nomenklaturen: Block-Element-Modifier

# BEM - Block-Element-Modifier

```
// Block
.form { }

// Block--Modifier
.form--theme-xmas { }
.form--simple { }

// Block__Element
.form__input { }
.form__submit { }

// Block__Element--
// Modifier
.form__submit--disabled
{ }

// SASS
.form {
  &--theme-xmas { ... }
  &--simple { ... }

  &__input { ... }
  &__submit {
    &--disabled { ... }
  }
}
```

# Der typografische Raster

2024 Michael Reichart

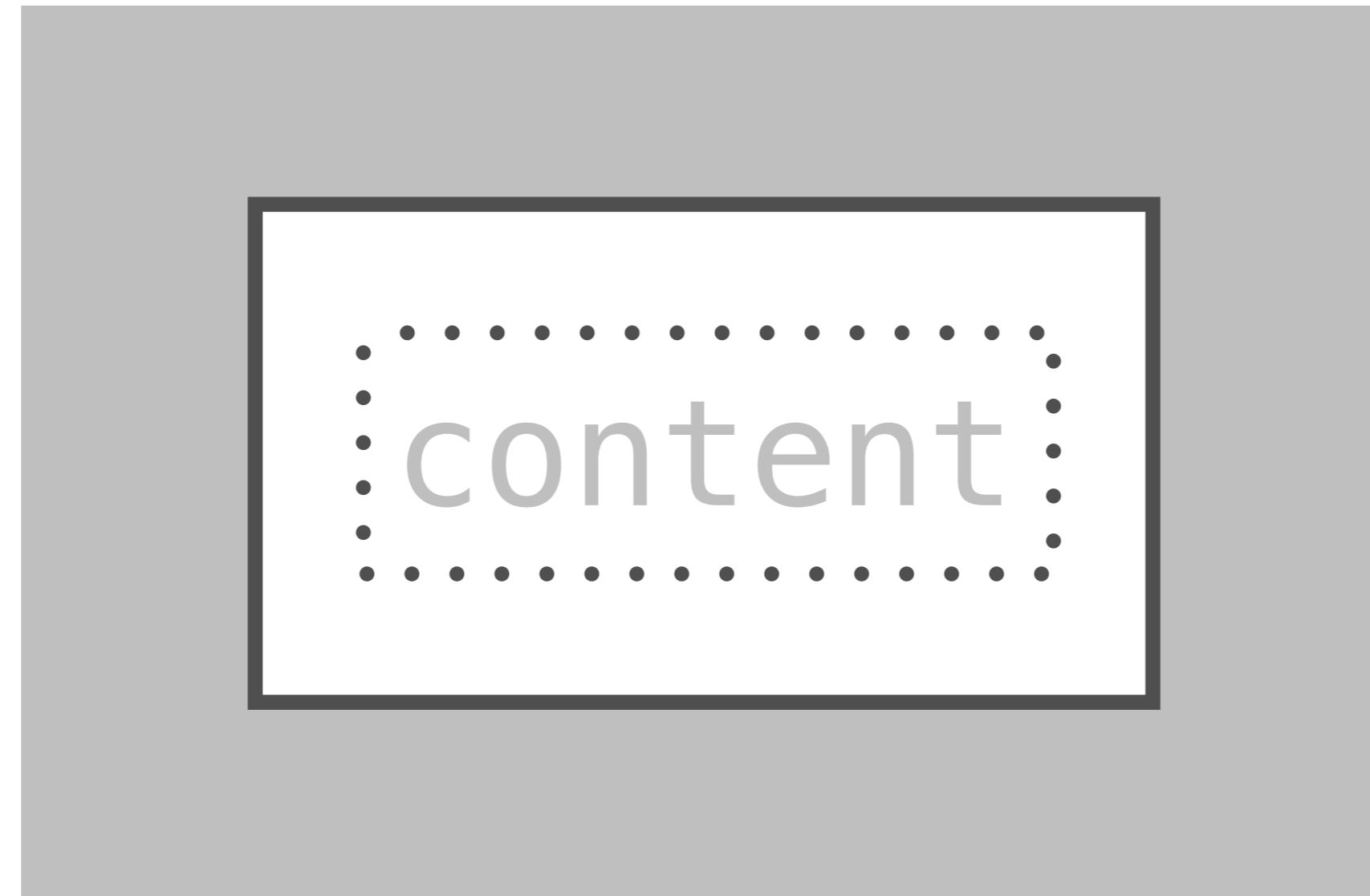
$$M = 1\text{em} = M\text{-Höhe}$$

<html>	root
...	
<body>	body
<h1>...</h1>	headlines
<p>...</p>	paragraphs
...	
</body>	margins
</html>	paddings



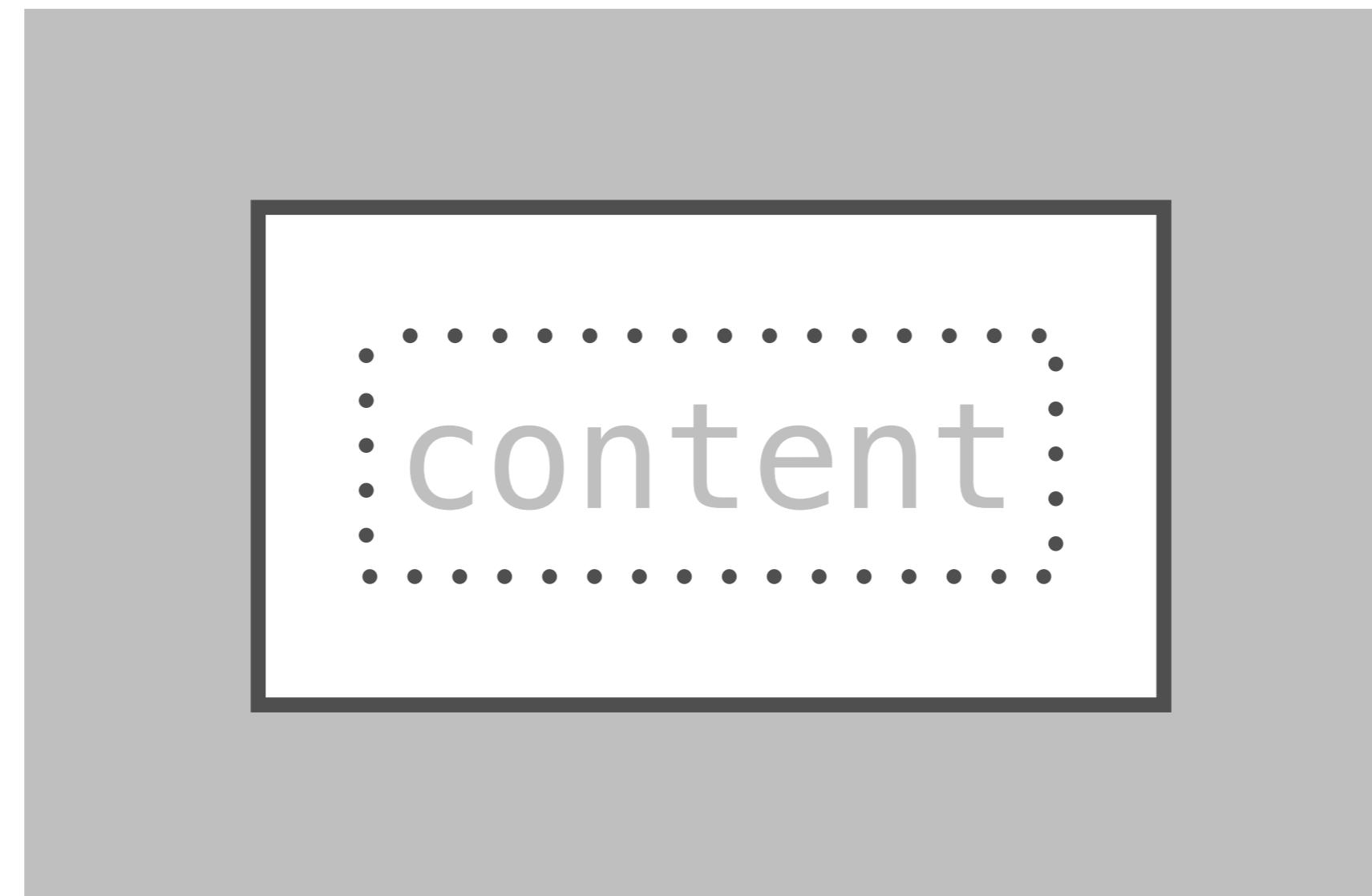
font-family: Barlow;  
font-size: 16px; =1rem  
line-height: 1.35; 21.6px

h1 { font-size: 2rem; }  
h2 { font-size: 1.8rem;  
margin-bottom: 1.2rem; }  
...  
p { font-size: 1rem; }



height : 1rem;  
padding : 1rem;  
border-width : 1px;  
margin-bottom : 1.2rem;

```
box-sizing: content-box;  
box-sizing: border-box;
```



```
display          : block;  
width           : 5rem; (80px)  
padding         : 1rem; (32px)  
border-width  : 1px; (2px)  
                           80px!  
  
margin-bottom   : 1.2rem;
```

# Typography

---

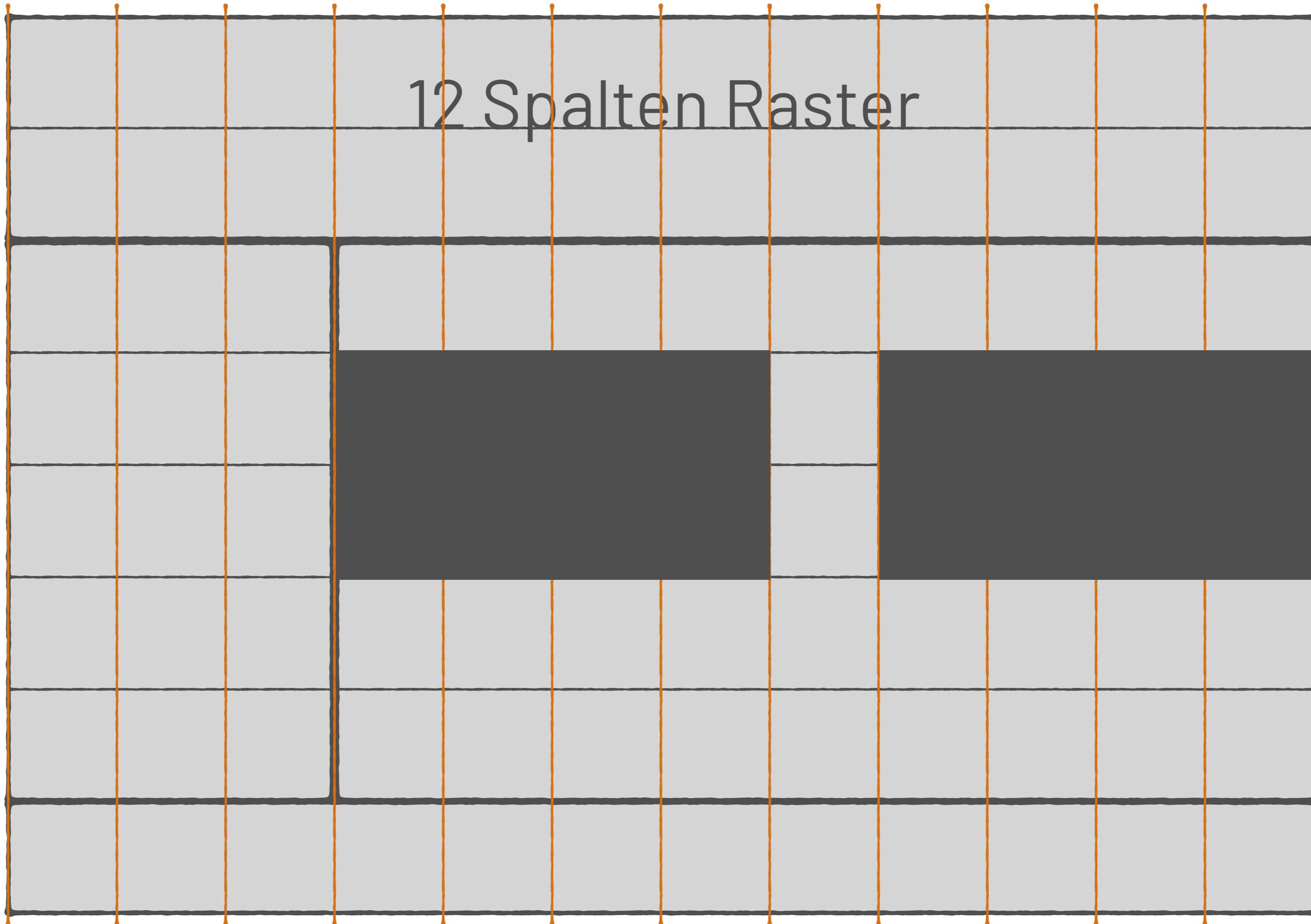
Paragraph (1rem) 1.35

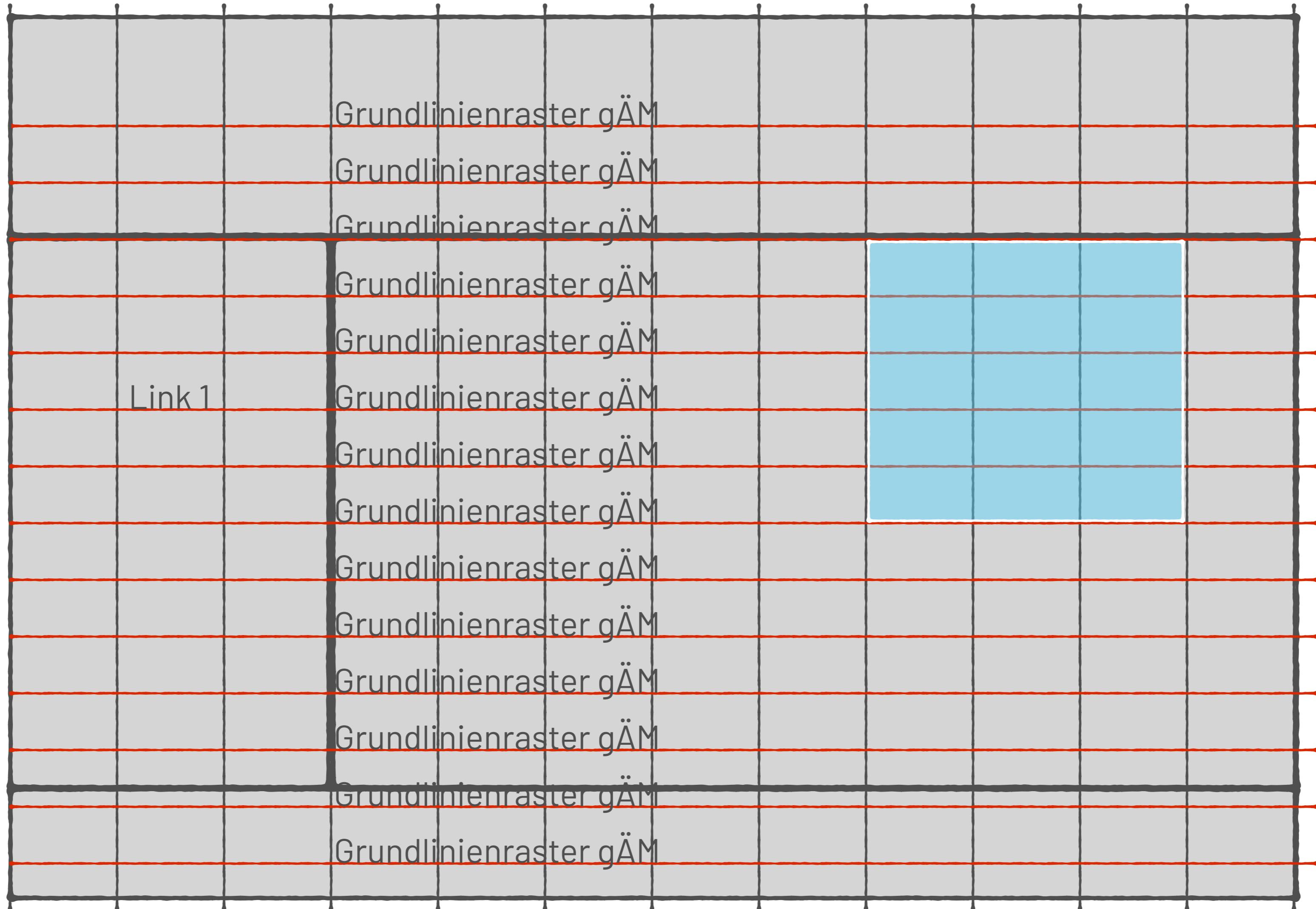
Headline (2rem) 1.35

Headline (1.8rem) 1.35

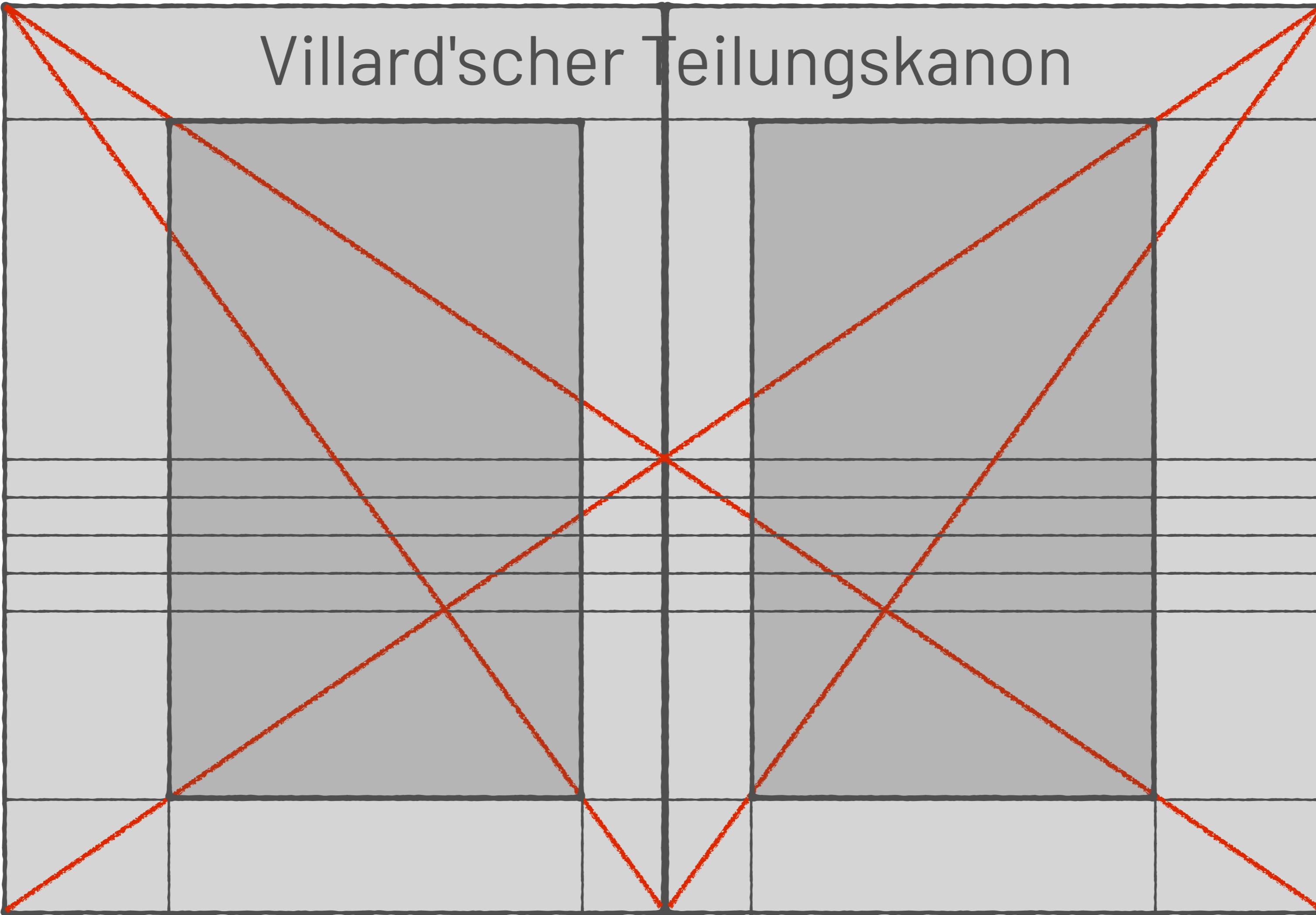


12 Spalten Raster





# Villard'scher Teilungskanon



# Responsive Layouts

2024 Michael Reichart

# Geräteunterscheidung mit CSS und Mediaqueries

Die wichtigen Änderungen und Erneuerungen

# Media Queries mit Attributen

---

- + HTML5 präzisiert Mediatypen.
- + Erweiterung durch Media Attributes wie 'width', und 'orientation'
- + Damit können medienabhängige Stylesheets angesprochen werden.

# Medientypen werden bereits in HTML 4 definiert

+ screen	Bildschirme	
projection	Beamer	wird nicht benutzt
handheld	kleine Computer	wird nicht benutzt
tv	Fernsehgeräte	wird nicht benutzt
+ print	Drucker	
tty	Nadeldrucker	
+ aural	auditiv	deprecated seit CSS2
braille	Blindenschrift	
embossed	Blindenschrift	seit CSS2
speech	Sprachausgabe	seit CSS2
+ all	Alle Medientypen	seit CSS2

## Externes medientyp-abhängiges Stylesheet

```
<link  
    rel="stylesheet"  
    type="text/css"  
    media="screen"  
    href="sans-serif.css"  
  >  
  
<link  
    rel="stylesheet"  
    type="text/css"  
    media="print"  
    href="serif.css"  
  >
```

# Mediablock in einer CSS Datei

```
@media screen {  
    html { font-family: sans-serif  
}  
    ...  
}
```

# not als Negation

---

@media not screen and (color)

# only

---

Das Keyword 'only' kann verwendet werden, um ein Stylesheet vor älteren User Agenten zu verbergen. Neue Browser ignorieren das Wort einfach.

`@media only screen and (color)`

# Mehrere Media Queries als kommaseparierte Liste

```
@media  
  screen and (color),  
  projection and (color)  
{  
  ...  
}
```

# Mediaqueries mit Fehlern

Wenn ein Media Feature auf ein Ausgabegerät nicht zutrifft, so liefert der User Agent ein false. Das Query wird abgelehnt.

```
<link  
  rel="stylesheet"  
  media="aural and (device-aspect-ratio: 16/9)"  
  href="example.css">
```

```
<link  
  rel="stylesheet"  
  media="speech and (min-device-width: 800px)"  
  href="example.css">
```

# Unterscheidung Dark Mode / Bright Mode

`prefs-color-scheme`

# dark-/bright mode des Users

---

```
@media (prefers-color-scheme: dark)  
{}
```

```
@media (prefers-color-scheme: bright)  
{}
```

# Medienattribute

Einsatz und Anwendungsbeispiele.

Ab HTML5 gibt es Medieneigenschaften, die zusätzlich zum Medientyp abgefragt werden können

- + **width**, height,
- + ~~device-width~~, ~~device-height~~
- + orientation
- + aspect-ratio, device-aspect-ratio
- + color, color-index, monochrome
- + resolution
- + scan
- + grid
- + all
- + z.B. @media screen and (min-width:768px)

# width, height

- + Wert: <length>, <height>
- + Anwendbar für visuelle und taktile Medientypen.
- + Kann mit min/max kombiniert werden.
- + 'width', 'height' bezeichnen die Breite/Höhe des Darstellungsbereiches.
- + Für kontinuierliche Medien sind das Breite und Höhe des Viewports inklusive der Scrollbar.
- + Für Medien, die mit Seiten arbeiten, sind das Breite und Höhe des Seitenlayouts oder der Textbox.
- + Breiten und Höhen dürfen nicht negativ sein.

# Für Druckseiten mit einer Breite größer 25 cm.

```
<link  
    rel="stylesheet"  
    media="print and (min-width: 25cm)"  
    href="http://..."  
>
```

Für Geräte mit einem Viewport zwischen 400 und 700 Pixeln:

```
@media  
    screen  
    and (min-width: 400px)  
    and (max-width: 700px)  
    { ... }
```

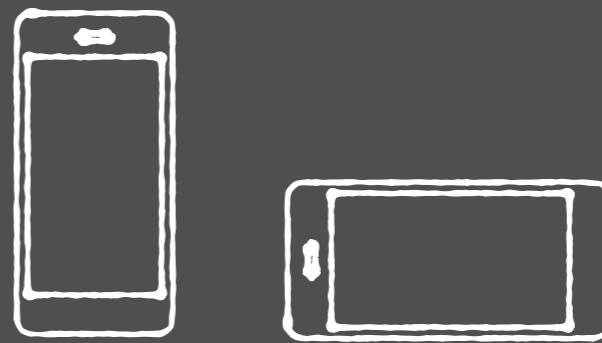
# device-width, device-height

- + Wert: <length>, <height>
- + Anwendbar für visuelle und taktile Medientypen.
- + Kann mit min/max kombiniert werden.
- + 'device-width' und 'device-height' bezeichnen die Breite/Höhe des Ausgabefläche. Bei kontinuierlichen Medien ist das die Bildschirmbreite. Bei seitenbasierten Medien ist das die Blattgröße.
- + Breiten und Höhen dürfen nicht negativ sein.

# orientation

- + Value: portrait | landscape
- + Anwendbar für bitmap basierte Medien
- + Kann nicht mit min/max kombiniert werden.
- + 'orientation' ist 'portrait', wenn die Höhe größer oder gleich ist, als die Breite.
- + Sonst gilt 'landscape'.

# Anwendung der Orientation



```
@media all and (orientation:portrait) { ... }  
@media all and (orientation:landscape) { ... }
```

# aspect-ratio

- + Value: <ratio>
- + Anwendbar für bitmap basierte Medien
- + Kann mit min/max kombiniert werden.
- + 'aspect-ratio' definiert das Verhältnis von 'width' und 'height' eines Asugabebereiches.

# device-aspect-ratio

- + Value: <ratio>
- + Anwendbar für bitmap basierte Medien
- + Kann mit min/max kombiniert werden.
- + The 'device-aspect-ratio' media feature is defined as the ratio of the value of the 'device-width' media feature to the value of the 'device-height' media feature.

# Das Verhältnis 16:9

```
@media  
  screen and (device-aspect-ratio: 16/9) { ... }  
  
@media  
  screen and (device-aspect-ratio: 32/18) { ... }  
  
@media  
  screen and (device-aspect-ratio: 1280/720) { ... }
```

# color

- + Wert: <integer>
- + Anwendbar für visuelle Medientypen.
- + Kann mit min/max kombiniert werden.
- + 'color' beschreibt die Anzahl von Bits pro Farbkanal des Ausgabegerätes.
- + Wenn das Gerät kein Farbgerät ist, dann ist der Wert 0.

Die beiden folgenden Queries gelten  
für jedes Farbausgabegerät

```
@media all and (color) { ... }  
@media all and (min-color: 8) { ... }
```

# color-index

- + Wert: <integer>
- + Anwendbar für visuelle Medientypen.
- + Kann mit min/max kombiniert werden.
- + The 'color-index' media feature describes the number of entries in the color lookup table of the output device. If the device does not use a color lookup table, the value is zero.

# Zwei Wege, um Geräte mit indizierter Farbausgabe anzusprechen

```
@media all and (color-index) { ... }  
  
@media all and (min-color-index: 1) { ... }
```

# monochrome

- + Wert: <integer>
- + Bei nicht monochromen Geräten wird 0 geliefert.
- + Anwendbar für visuelle Medientypen.
- + Kann mit min/max kombiniert werden.
- + Die monochrome - Angabe bezeichnet die Anzahl der bits pro Pixel eines einfarbigen Bildschirmspuffers.

# Zwei Wege für ein monochromes Gerät

```
@media all and (monochrome) { ... }  
  
@media all and (min-monochrome: 1) { ... }
```

# resolution

- + Wert: <resolution>
- + Anwendbar für bitmap basierte Medien
- + Kann mit min/max kombiniert werden.
- + resolution beschreibt die Auflösung des Ausgabegerätes, zum Beispiel die Pixeldichte. Medien mit nicht-quadratischen Pixeln muss die least-dense mit dem Wert von min-resolution verglichen werden, max-resolution mit der most-dense Dimension. Ein einfache 'resolution'-Angabe funktioniert niemals mit einem Gerät mit nicht-quadratischen Pixeln.
- + Bei Drucken korrespondiert die 'resolution' mit der Druckauflösung (Anzahl der Druckpunkte) bei beliebigen Farbmodell.

# Beispiele für Drucker

```
@media print and (min-resolution: 300dpi) { ... }  
@media print and (min-resolution: 118dpcm) { ... }
```

# scan

- + Wert: progressive | interlace
- + Anwendbar für 'tv' Medien
- + Kann nicht mit min/max kombiniert werden.
- + 'scan' beschreibt das Zeilenaufbauverfahren eines TV- Ausgabegerätes.

# Ein Beispiel für TV Geräte mit progressiven Scanning (Zeilenaufbau)

```
@media tv and (scan: progressive) { ... }
```

# grid

- + Wert: <integer>
- + Anwendbar für visuelle und taktile Medien.
- + Kann nicht mit min/max kombiniert werden.
- + 'grid' wird verwendet, um Geräte mit einer festen Bildschirmmatrix (grid) zu verwenden. Das kann ein 'tty'-Terminal sein oder auch ein Handydisplay mit einer festgelegten Zeichenanzahl- und Größe. 'grid' liefert bei solchen Geräten eine '1'. Bitmapbasierte Displays erzeugen eine '0'.
- + Andere Werte als '0' oder '1' erzeugen ein falsches Mediaquery.

Zwei Beispiele:

1em entspricht dabei einer Zeichenbreite oder -höhe.

```
@media  
  handheld  
  and (grid)  
  and (max-width: 15em) { ... }
```

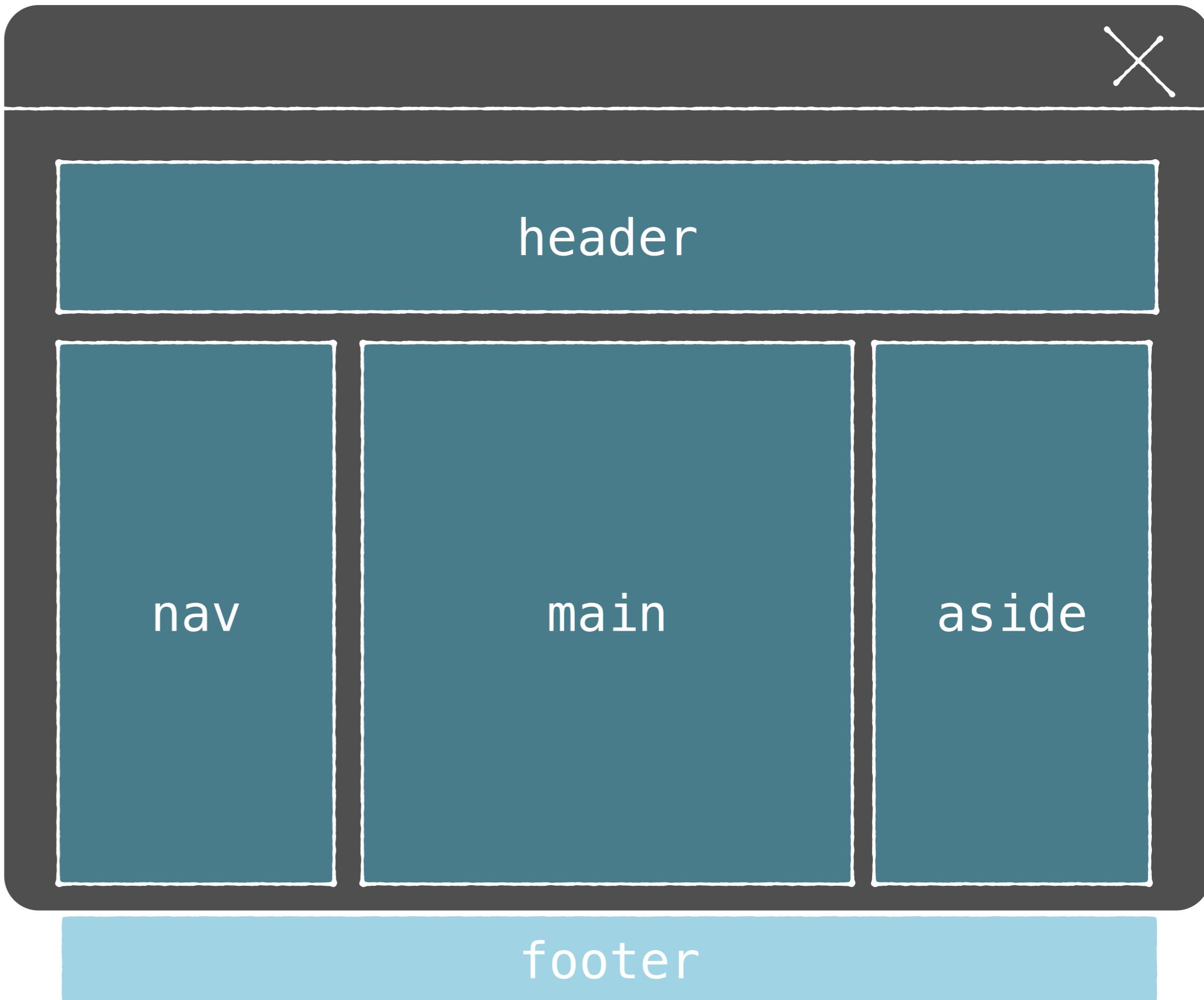
```
@media  
  handheld  
  and (grid)  
  and (device-max-height: 7em) { ... }
```

# Adaptives Design.

Für mobile Geräte heute Standard und ausreichend. Außerdem: Mobile First – Content First.

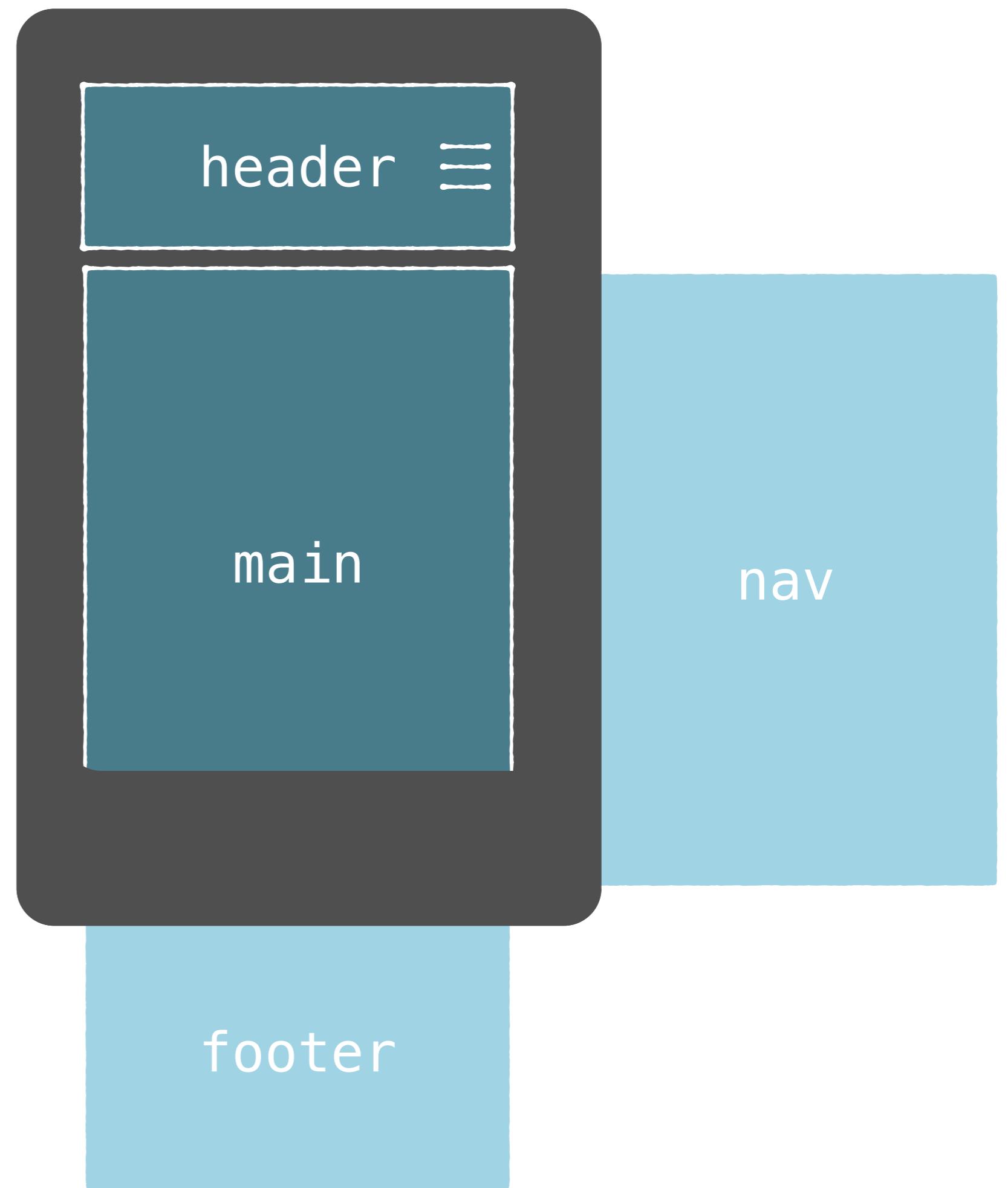
# Desktop-layouts sind mehrspaltig.

- + Es gibt genügend Raum, um Navigation und Inhalte nebeneinander zu platzieren.



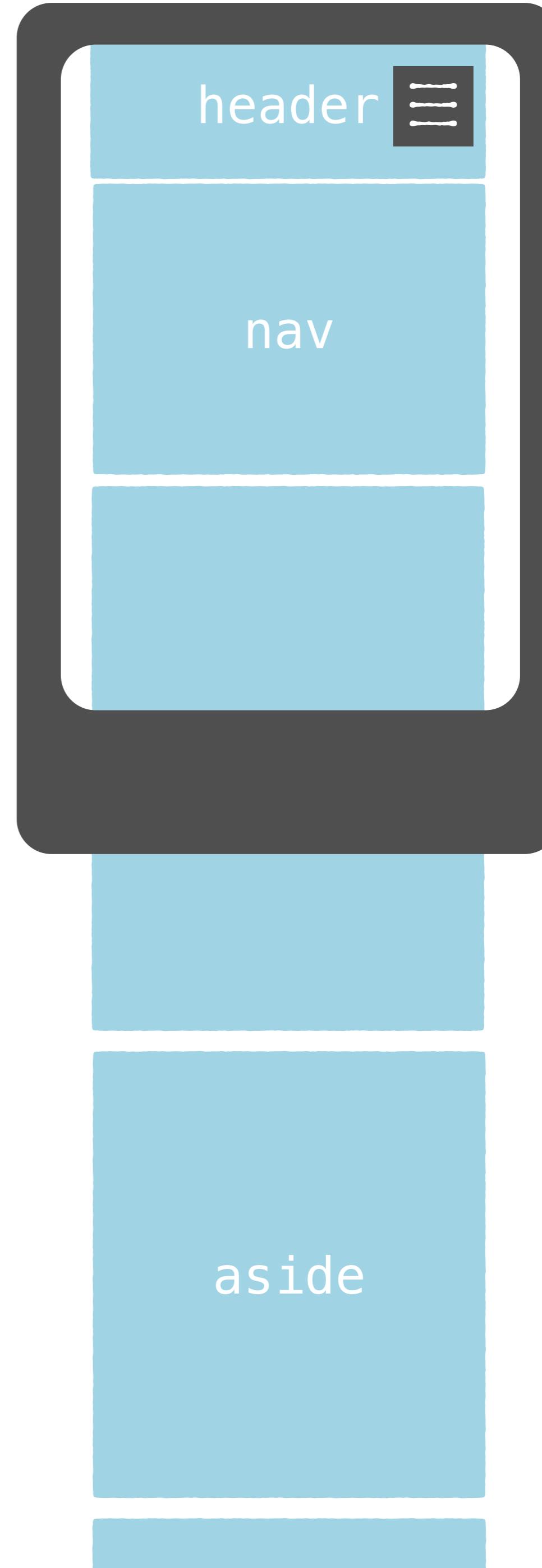
# Smartphone layouts nicht.

- + Besonders im Hochformat bietet das Layout oft nur die Hälfte oder ein Drittel des Raumes.
- + Die Anordnung der Layoutelemente muss geändert werden.

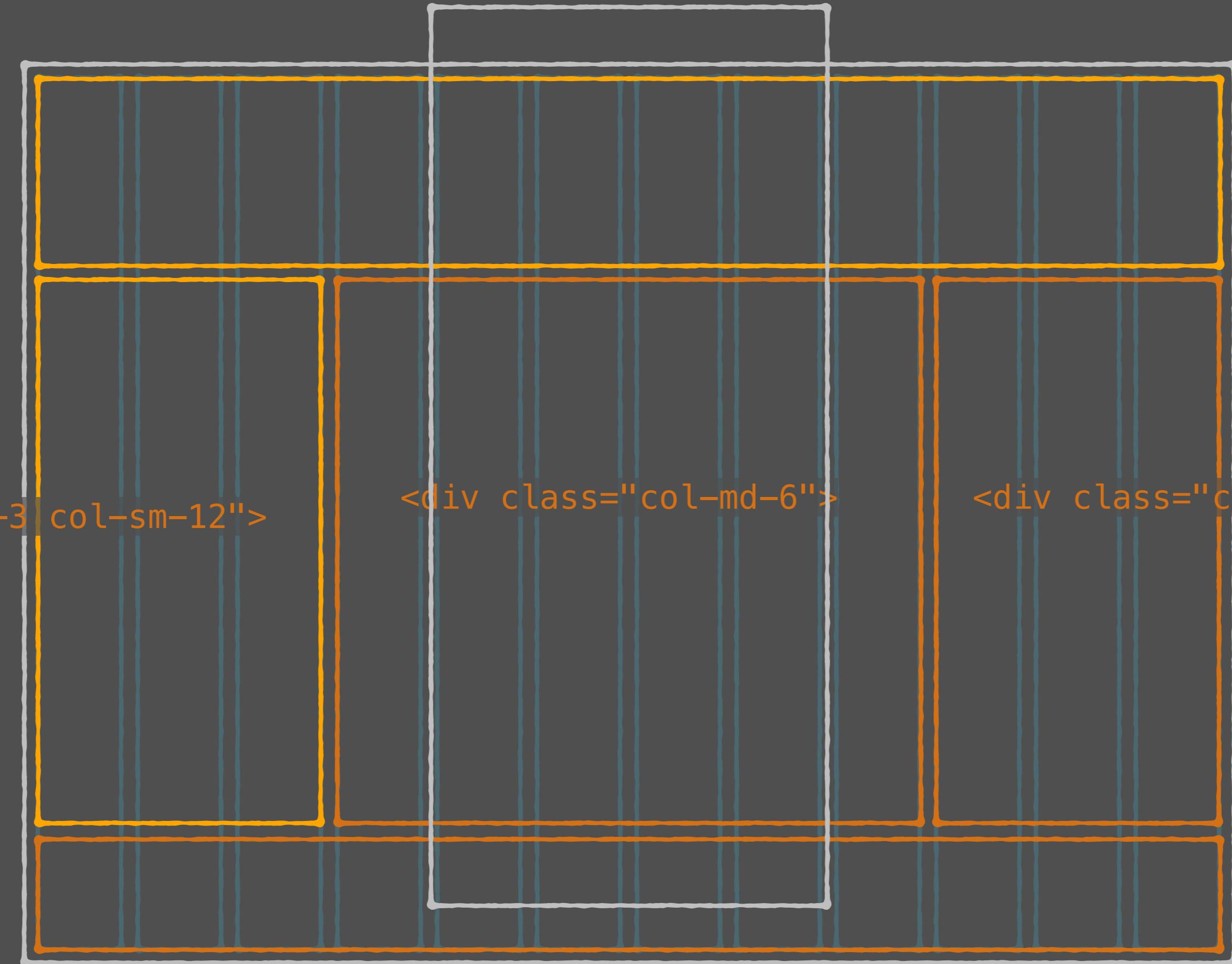


# Linearisierung der Inhalte.

- + Die Inhalte werden untereinander angeordnet, (Flow), einige Elemente werden gar nicht angezeigt.
- + Mobile First  
= Content First



```
<div class="container"> width: 100%
  <div class="col-md-12">
```



```
<div class="col-md-12">
```





# Die Mediaqueries aus Bootstrap

```
/* Extra small devices (phones, less than 768px) */
/* No media query since this is the default in
Bootstrap */

/* Small devices (tablets, 768px and up) */
@media (min-width: @screen-sm-min) { ... }

/* Medium devices (desktops, 992px and up) */
@media (min-width: @screen-md-min) { ... }

/* Large devices (large desktops, 1200px and up) */
@media (min-width: @screen-lg-min) { ... }
```

# Bootstrap: Für alle Displaybreiten eine passende Variante.

	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Grid behavior	Horizontal at all times		Collapsed to start, horizontal above breakpoints	
Max container width	None (auto)	750px	970px	1170px
Class prefix	.col-xs-	.col-sm-	.col-md	.col-lg
# of columns	12			
Max column width	Auto	60px	78px	95px
Gutter width		30px (15px on each side of a column)		
Nestable				
Offsets	N/A	Yes		
Column ordering	N/A	Yes		

# Mobile First Approach

Responsive Layouts

2024 Michael Reichart

# Das Attribut width, um Geräte zu unterscheiden: Vom kleinsten zum größten Gerät

```
really very tiny devices:  
@media screen and (max-width:480px) { ... }  
  
tiny devices:  
@media screen and (min-width:481px) { ... }  
  
small devices:  
@media screen and (min-width:768px) { ... }  
  
medium devices:  
@media screen and (min-width:1024px) { ... }  
  
large devices:  
@media screen and (min-width:1280px) { ... }  
  
xtra large:  
@media screen and (min-width:1600px) { ... }  
  
full hd device:  
@media screen and (min-width:1920px) { ... }
```

# Mobile First - Progressive Enhancement

```
ALL DEVICES
@media screen {
    declarations for all devices,
    optimized for extra small mobile devices
}

XTRA SMALL
@media screen and (max-width:767px) {
    changes/specials for extra small
}

SMALL
MEDIUM
LARGE
@media screen and (min-width:768px) { ... }
@media screen and (min-width:992px) { ... } // 1024px
@media screen and (min-width:1200px) { ... } // 1280px

HD READY
??
FULL HD
4K
@media screen and (min-width:1440px) { ... }
@media screen and (min-width:1600px) { ... }
@media screen and (min-width:1920px) { ... }
@media screen and (min-width:3840px) { ... }

PRINTER
@media print {}
@media print and (min-width:16cm) {}
```

# Mobile First - Material Design

```
@mixin listReset() {
  list-style-type: none;
  margin: 0;
  padding: 0;
}
@media screen {
  // All devices fallback

  // Extra-small
  @media (max-width: 599px) {}

  // Small (tablet)
  @media (min-width: 600px) {}
  @media (min-width: 905px) {}

  // Medium (laptop)
  @media (min-width: 1240px) {}

  // Large (desktop)
  @media (min-width: 1440px) {}
}
```

# DESKTOP First GRACEFUL DEGRADATION

ALL DEVICES

```
@media screen {  
    declarations for all devices,  
    optimized for all, even old ones, devices  
}
```

MEDIUM

```
@media screen and (min-width:1024px) and (max-width:1279px) {  
    changes/specials for medium  
}
```

SMALL

```
@media screen and (max-width:767px) { ... }
```

XTRA SMALL

```
@media screen and (max-width:480px) { ... }
```

LARGE

```
@media screen and (min-width:1280px) { ... }
```

XTRA LARGE

```
@media screen and (min-width:1600px) { ... }
```

HD

```
@media screen and (min-width:1920px) { ... }
```

# Andere Attribute

```
@media screen and (orientation:portrait) { ... }

@media print and (resolution:300dpi) { ... }

@media screen and (min-resolution:96dpi) { ... }
@media screen and (min-resolution:192dpi) { ... } // Retina

@media print and (color) { ... }

@media screen and (aspect-ratio:16/9) { ... }
```

# Flex Box

Neues Layouten

2024 Michael Reichart

# Was sind flex boxes?

- + Flex boxes bieten einen sehr effizienten Weg, um Layoutelemente anzurichten.
- + In einem „Container“ werden „Items“ angeordnet, ausgerichtet und verteilt, auch wenn deren Größe nicht bekannt oder dynamisch ist.
- + Responsive Layouts lassen sich mit flex boxes einfacher und stabiler umsetzen, als mit dem bisherigen float/clear Verfahren.

# Flex Container

```
<div class="page" id="news">  
    <div class="page-header"> . . . </div>  
    <div class="page-content">  
        <div class="content-nav"> . . . </div>  
        <div class="content-main"> . . . </div>  
        <div class="content-aside"> . . . </div>  
    </div>  
    <div class="page-footer"> . . . </div>  
</div>
```

# Flex items

```
<div class="page" id="news">  
    <div class="page-header"> . . . </div>  
  
    <div class="page-content">  
        <div class="content-nav"> . . . </div>  
        <div class="content-main"> . . . </div>  
        <div class="content-aside"> . . . </div>  
    </div>  
  
    <div class="page-footer"> . . . </div>  
</div>
```

# Der Flexbox-Container

```
.page-content {  
    display        : flex;          /* Kindelemente werden Items      */  
    flex-direction : row;          /* Items zeilenweise anordnen   */  
    flex-wrap      : nowrap;        /* Items dürfen nicht umbrechen */  
    flex-flow      : /* Shorthand für flex-direction/flex-wrap */  
  
    align-items    : stretch;       /* Höhe der Items anpassen     */  
    align-content  : flex-start;    /* vertikale Ausrichtung der Items */  
  
    justify-content: space-between /* horizontale Itemausrichtung  */  
}
```

# Flex-Items

```
.content-nav {  
    flex-basis : 25%; /* initiale Breite des Elements */  
    flex-grow : 1;    /* Proportion beim Vergrößern */  
    flex-shrink : 1;  /* Proportion beim Verkleinern */  
}  
.content-main {  
    flex-basis : 50%;  
    flex-grow : 2;  
    flex-shrink : 2;  
}  
.content-aside {  
    flex-basis : 25%;  
    flex-grow : 1;  
    flex-shrink : 1;  
}
```

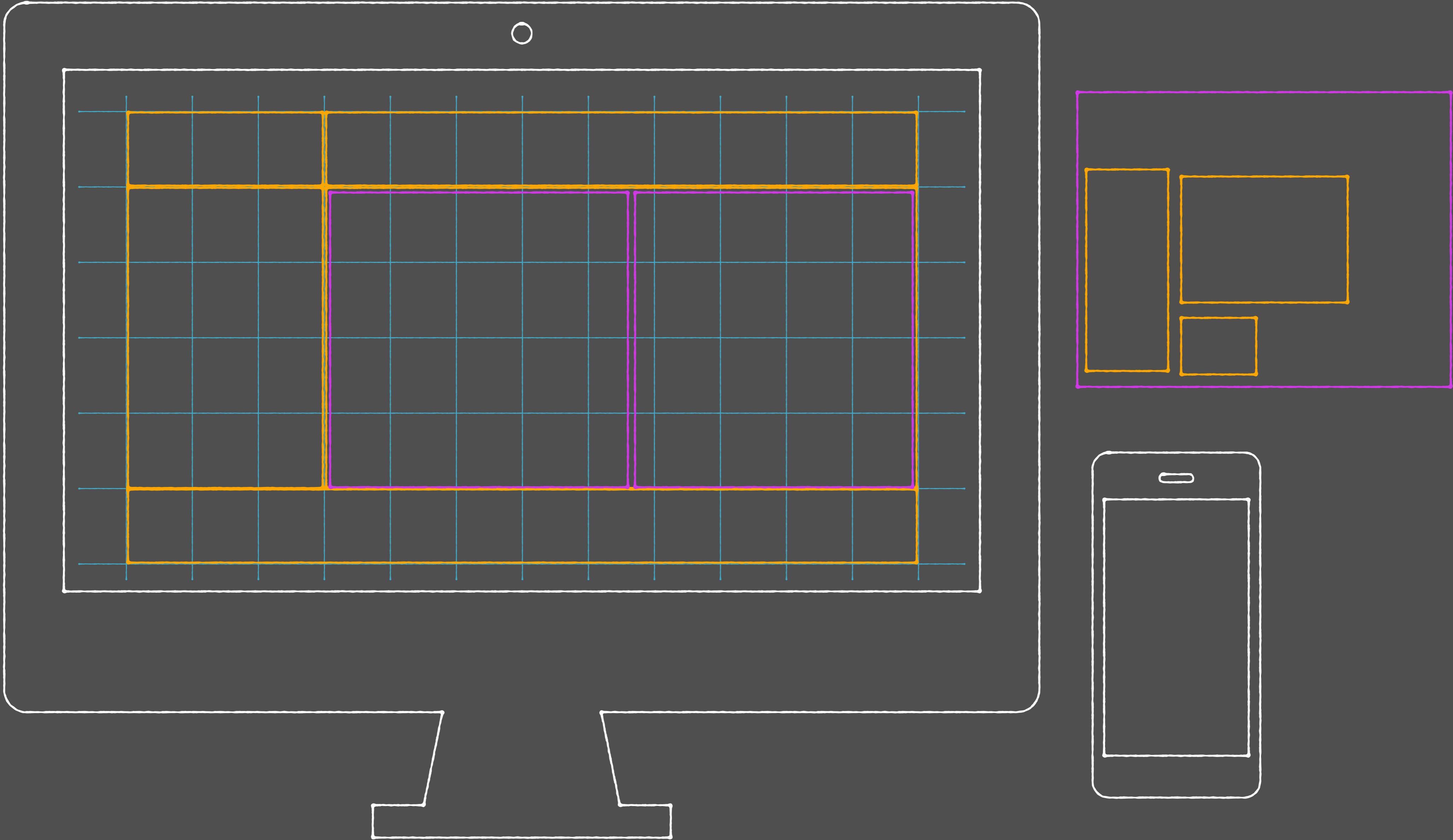
Andere Eigenschaften für Items sind: `order` (Reihenfolge) und `align-self` (Vertikale Ausrichtung des Items)

<http://css-tricks.com/snippets/css/a-guide-to-flexbox/>

- Chris Coyier

# Grids

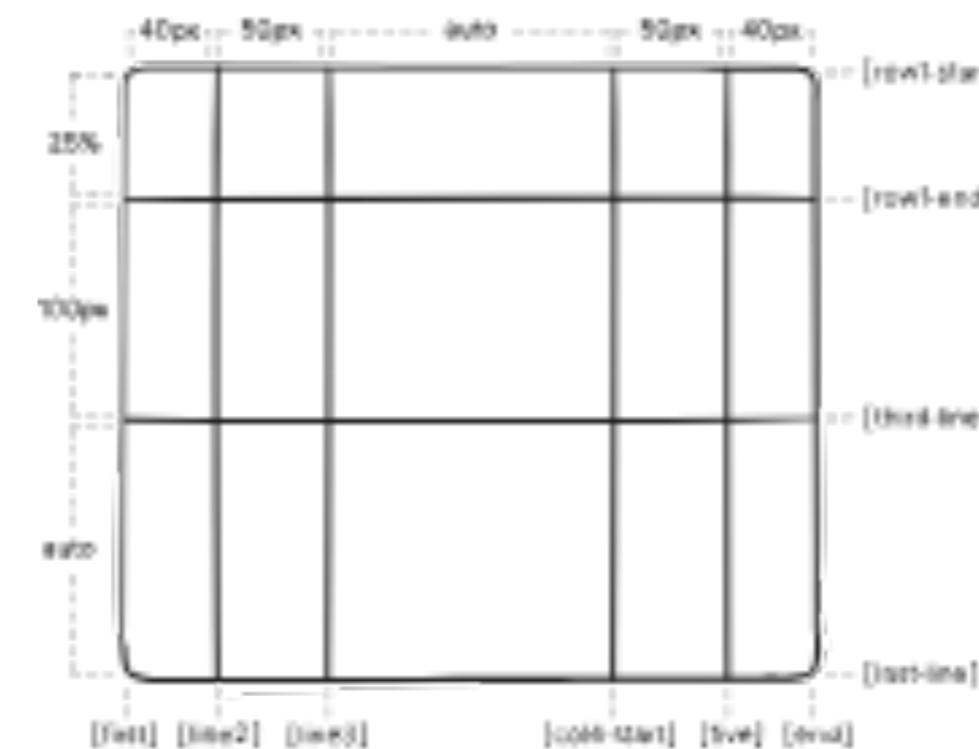
Neues Layouten



# Der Grid-Container

```
.container {  
  display: grid | inline-grid;  
}  
  
.container {  
  grid-template-columns: ... ...;  
  /* e.g.  
   1fr 1fr  
   minmax(10px, 1fr) 3fr  
   repeat(5, 1fr)  
   50px auto 100px 1fr  
  */  
  grid-template-rows: ... ...;  
  /* e.g.  
   min-content 1fr min-content  
   100px 1fr max-content  
  */  
}
```

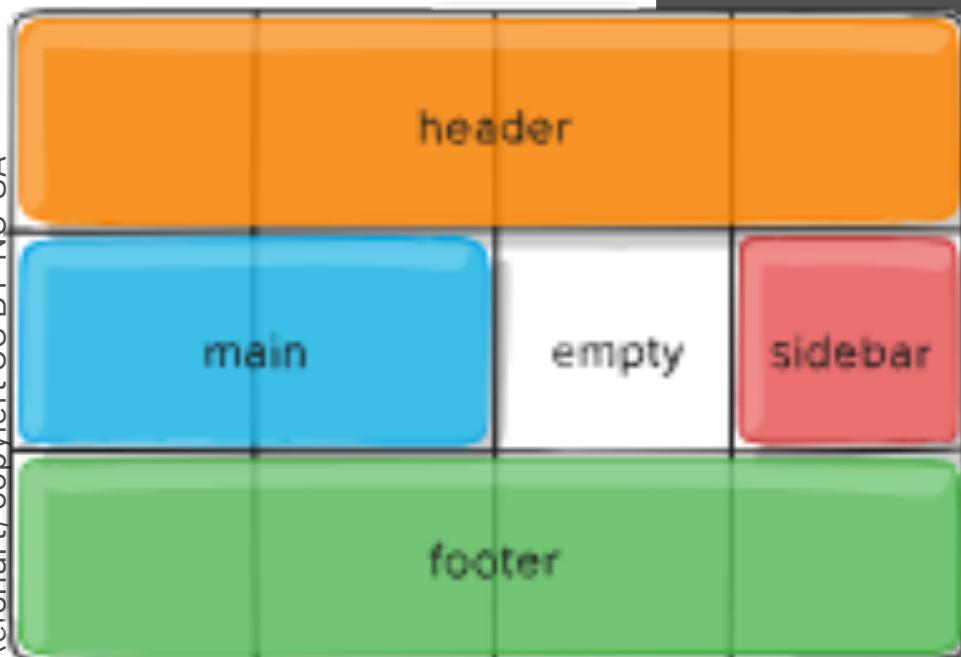
# Grid-Linien benennen



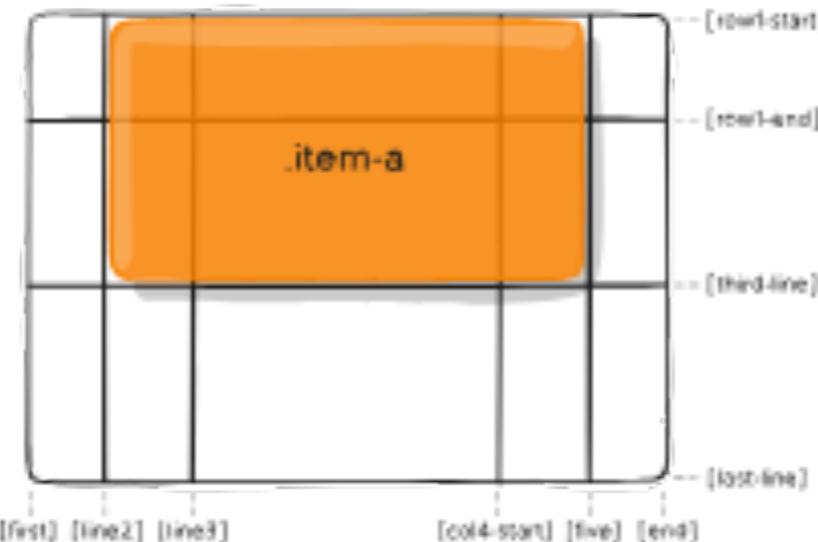
```
.container {  
    grid-template-columns:  
        [first]  
        40px [line2]  
        50px [line3]  
        auto [col4-start]  
        50px [five]  
        40px [end];  
  
    grid-template-rows:  
        [row1-start]  
        25% [row1-end]  
        100px [third-line]  
        auto [last-line];  
}
```

# grid-areas

```
.container {  
    display: grid;  
    grid-template-columns:  
        50px 50px 50px 50px;  
    grid-template-rows:  
        auto;  
    grid-template-areas:  
        "header header  
        header header"  
        "main main  
        . sidebar"  
        "footer footer  
        footer footer";  
  
.item-a {  
    grid-area: header;  
}  
.item-b {  
    grid-area: main;  
}  
.item-c {  
    grid-area: sidebar;  
}  
.item-d {  
    grid-area: footer;  
}
```



# grid-items



```
.item-a {  
  grid-column-start: 2;  
  grid-column-end: five;  
  grid-row-start: row1-start;  
  grid-row-end: 3;  
}  
  
// Shorthand  
.item-a {  
  grid-column: 2 / five;  
  grid-row: row1-start / 3;  
}
```

[https://css-tricks.com/snippets/css/  
complete-guide-grid/](https://css-tricks.com/snippets/css/complete-guide-grid/)

- Chris House

# Ein Template entwerfen und umsetzen

Praktisches

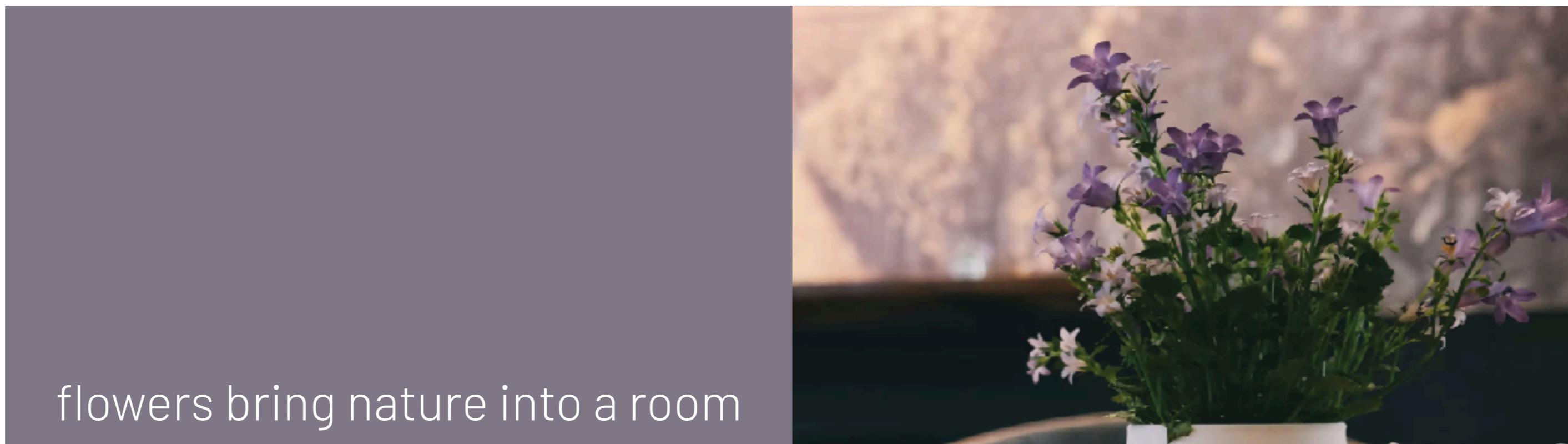
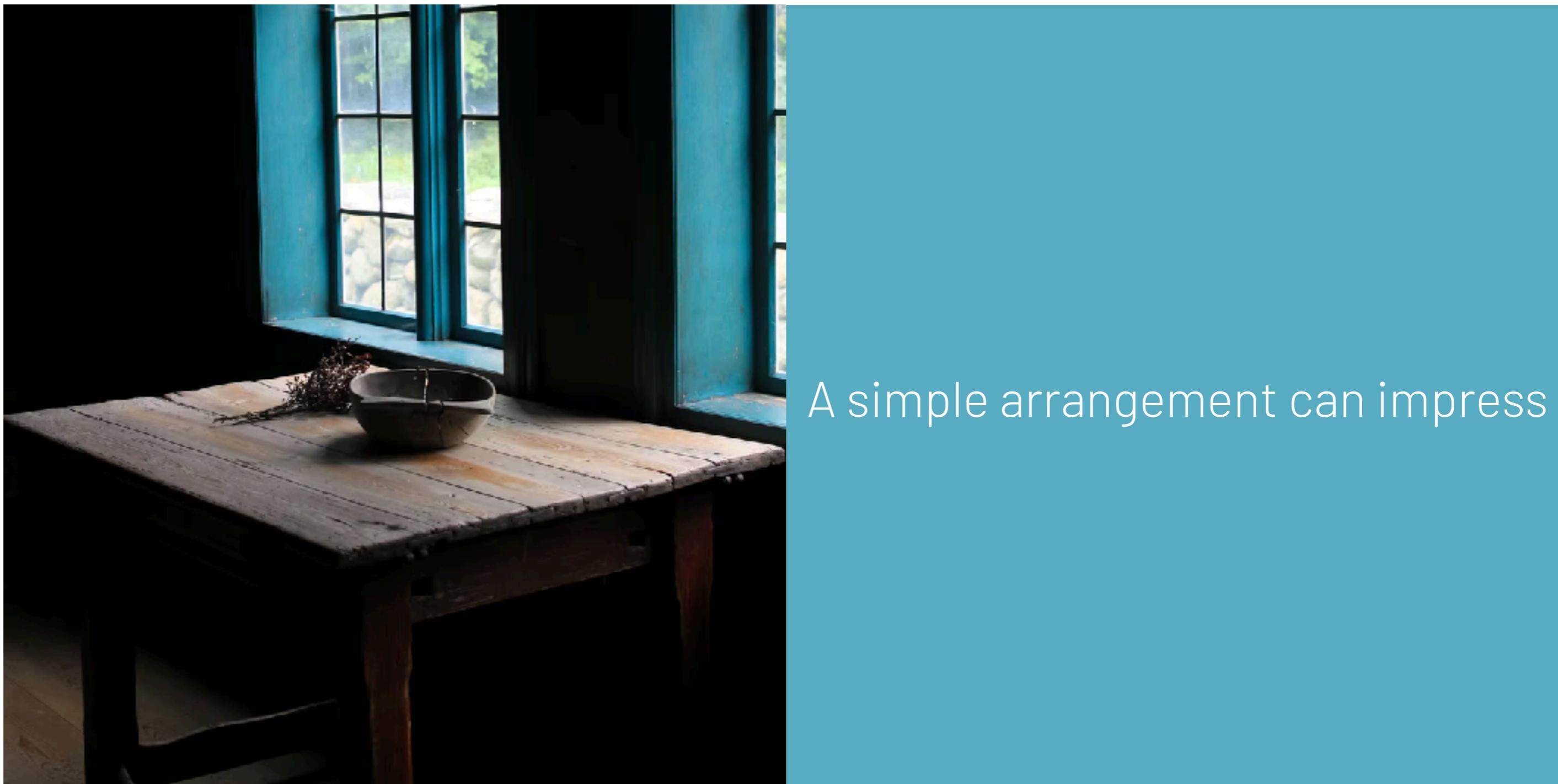


>= 1024px

D E C O R



New Autumn Christmas Living Light Objects Furniture Kitchen & Tables Gifts Sale Inspiration



768px - 1023px



A simple arrangement  
can impress

The header features a menu icon (three horizontal lines), a search icon (magnifying glass), the word "decor" in a bold, dark blue sans-serif font, and three user interaction icons: a profile, a heart, and a shopping bag.



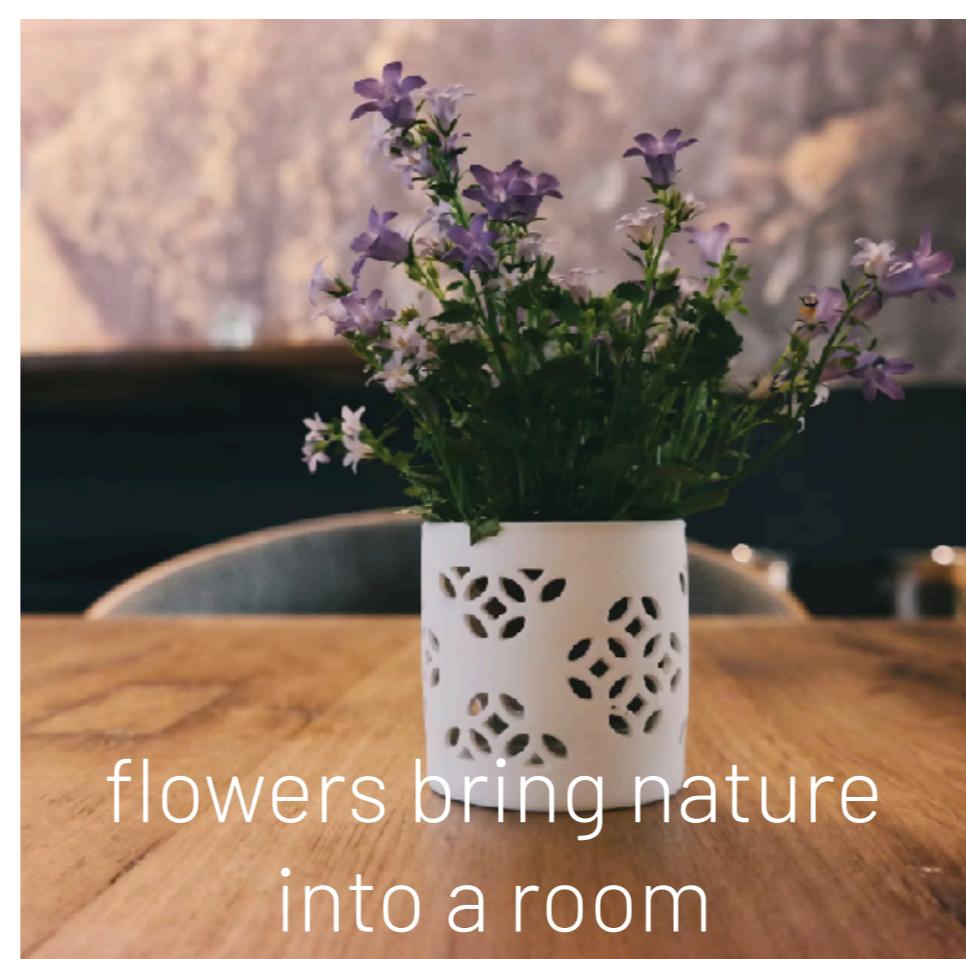
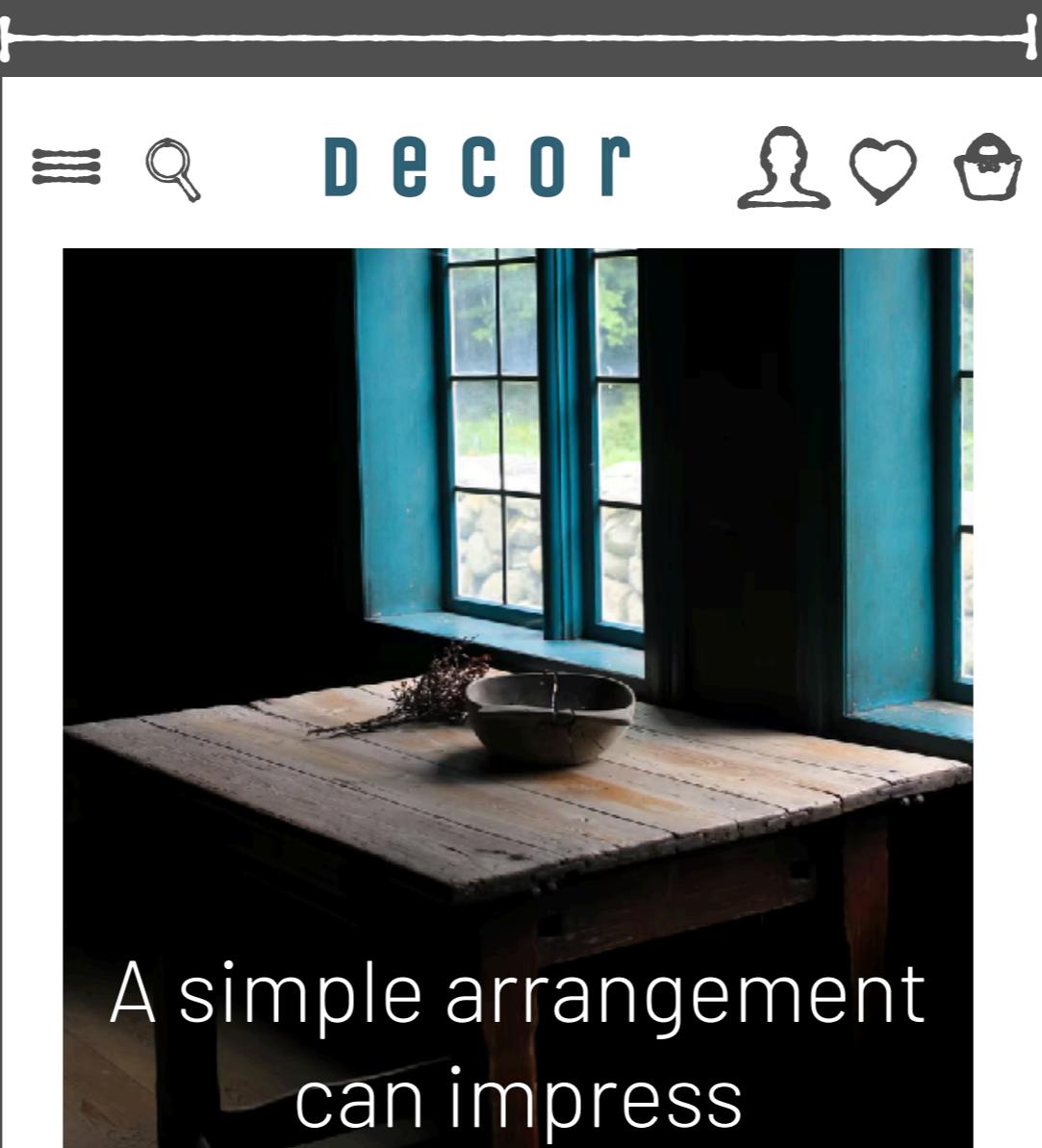
flowers bring nature  
into a room

This section includes a large gray overlay box containing the text "flowers bring nature into a room". To its right is a smaller image of a white vase with purple flowers on a wooden surface.

discover articles



>=767px



discover articles



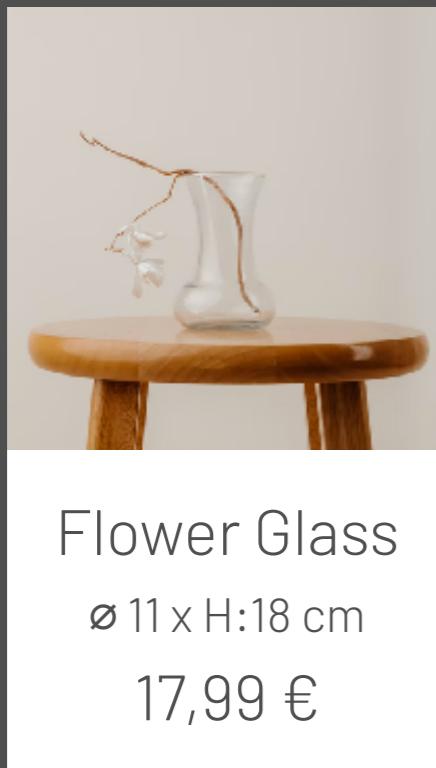
Flower Glass



Mug



# Aufgabe für 30 Minuten Teamarbeit



Baut die Produktansicht.

- > 1. entwickelt semantisches und layoutfähiges HTML.
- > 2. entwickelt in article.css einen neuen Abschnitt .article-product mit passenden Selektoren, um Bild und Typografie festlegen zu können.
- > 3. Diskutiert um passende Lösungsansätze.
- > 4. Präsentiert Eure Ergebnisse.

# Responsive Content

Typographie, Farben, Verhalten

2024 Michael Reichart

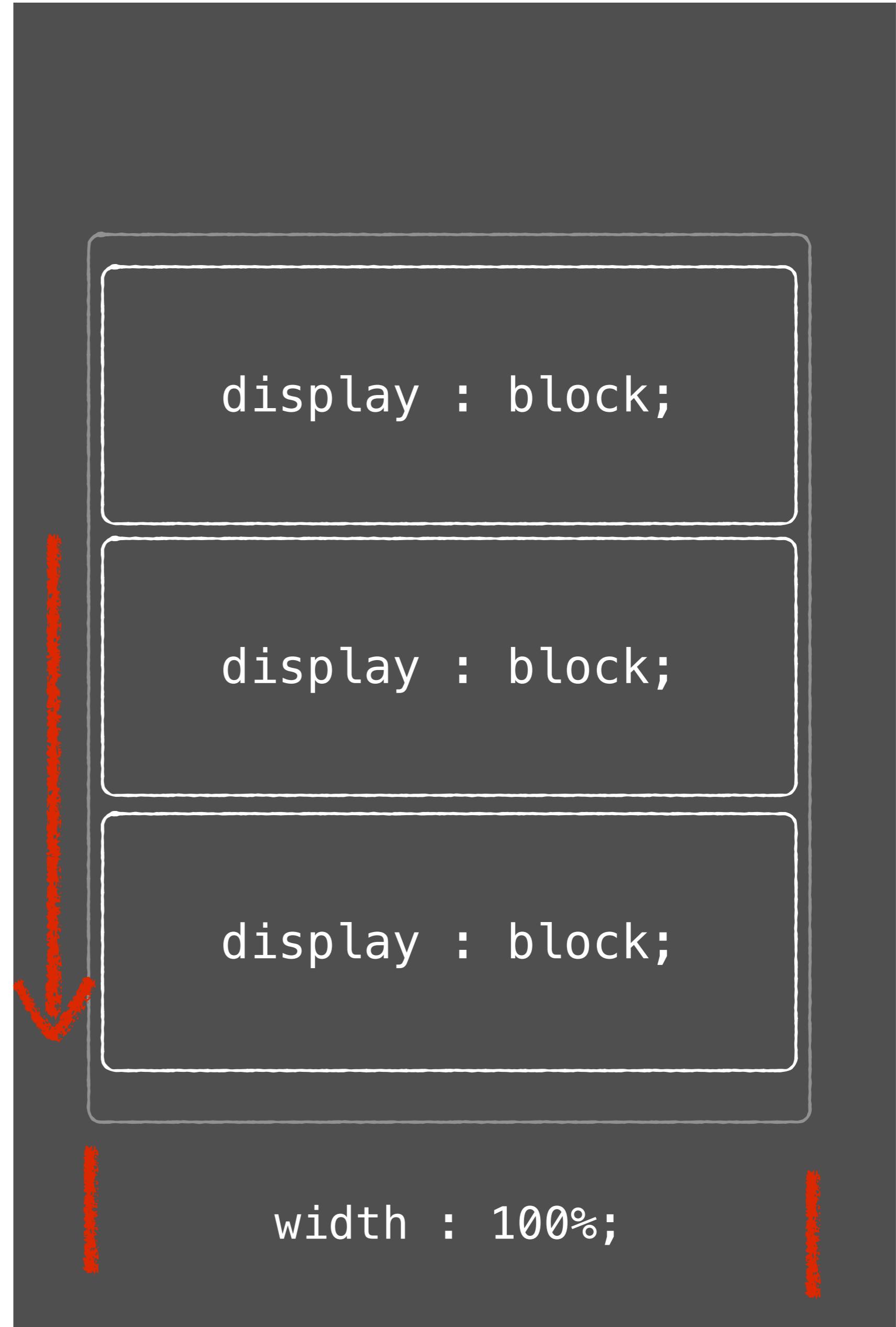
# Die Anzeige von Elementen steuern

display - appearance - text-overflow - box-sizing

# display: block;

- + Default-Eigenschaften sind:
- + 

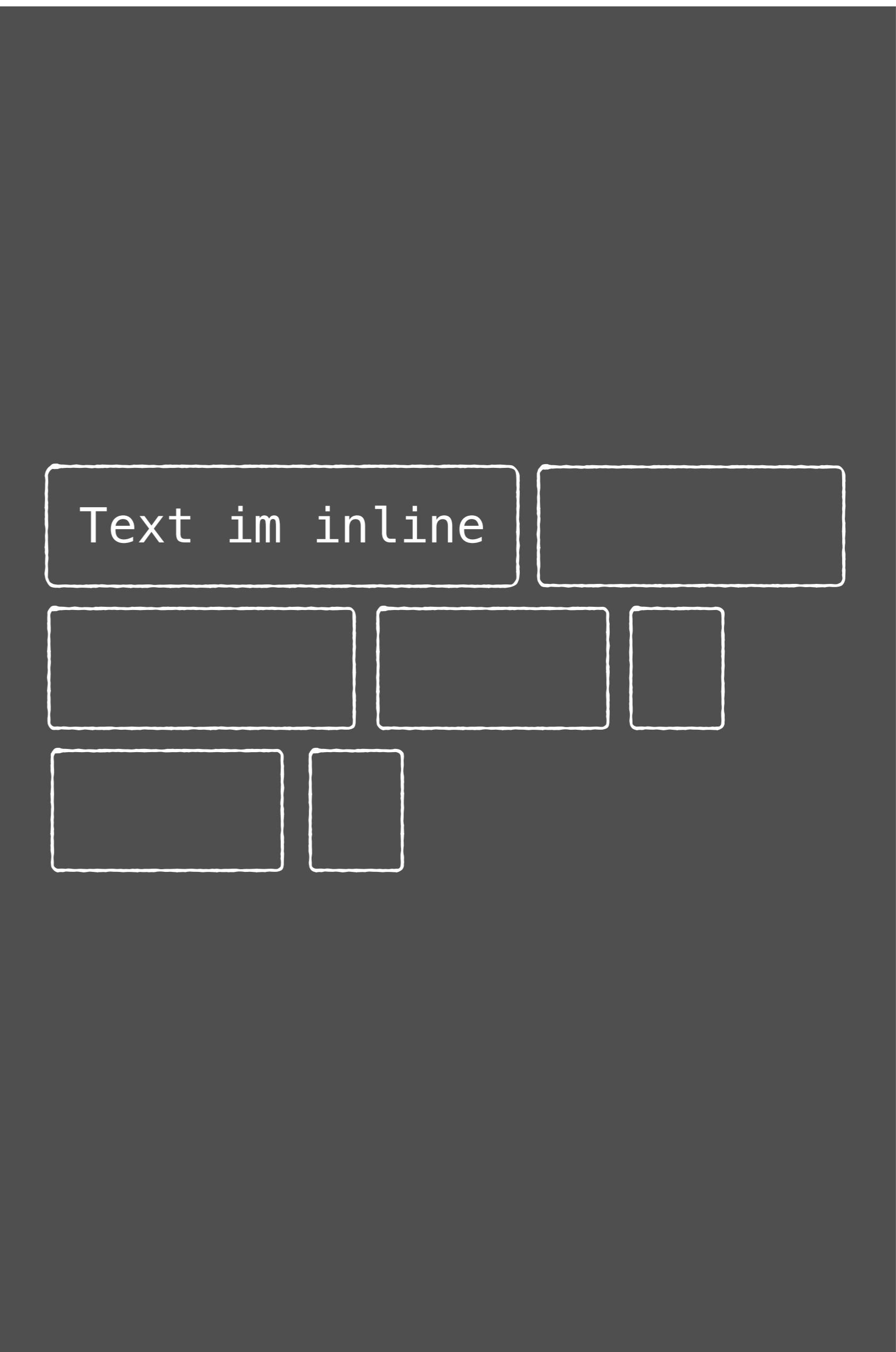
```
div {  
    display : block;  
    width   : 100%;  
    height  : auto;  
    margin   : 0;  
    padding  : 0;  
}
```



# display: inline

- + 

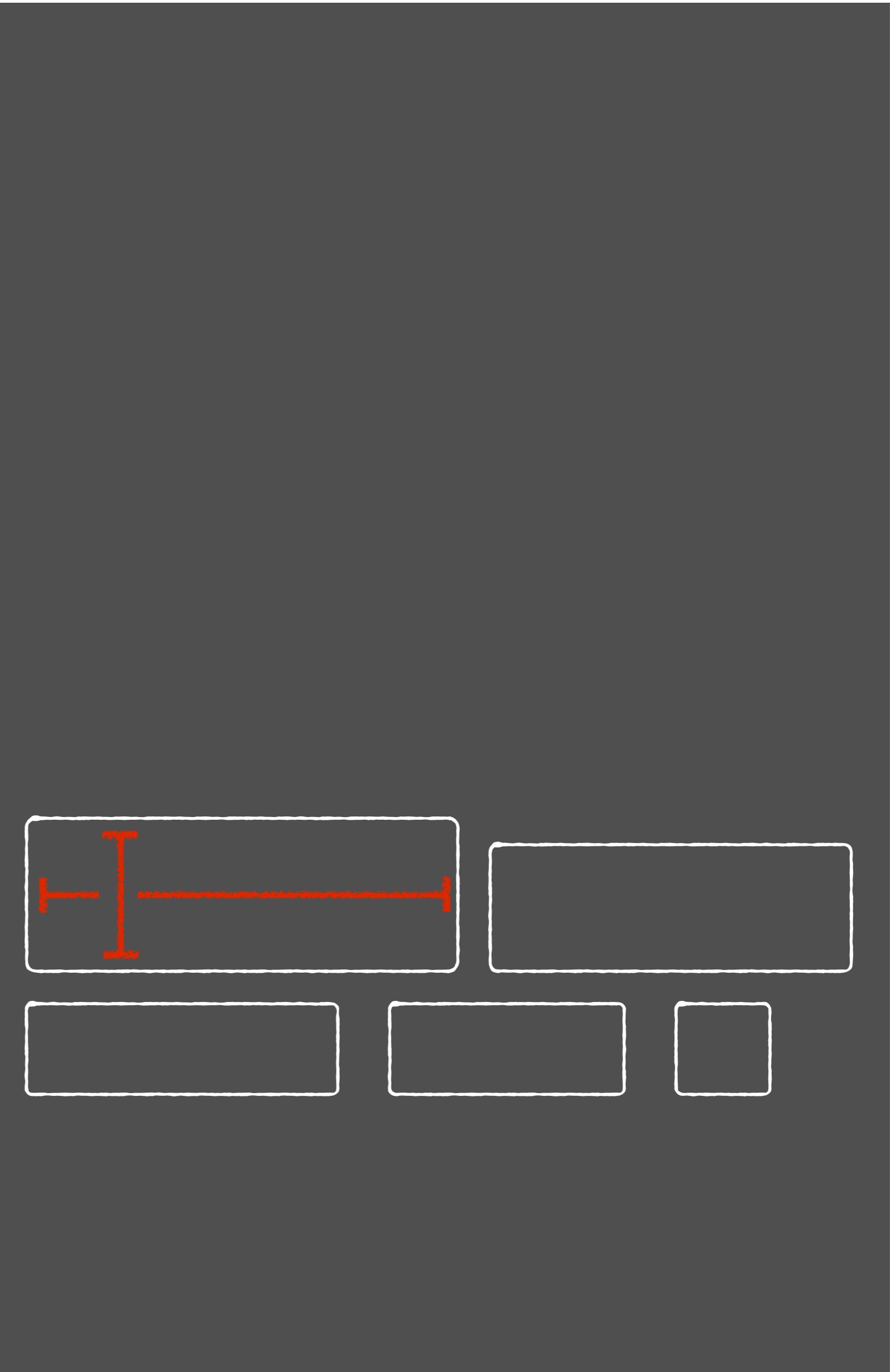
```
div {  
    display : inline;  
}
```
- + Kein width, height, margin oder padding,
- + verhält sich wie Text.



# inline-block

```
+ div {  
    display : inline-block;  
    width   : [value];  
    height  : [value];  
    margin  : [value];  
    padding : [value];  
}
```

- + Stattet das Inlineelement mit width/height, padding/ margin aus,
- + verhält sich sonst wie Text.



# table - Display-Typen

- + display: table;
- display: table-cell;
- display: table-column;
- display: table-colgroup;
- display: table-header-group;
- display: table-row-group;
- display: table-footer-group;
- display: table-row;
- display: table-caption;

# Sieht aus wie eine Tabelle, ist aber keine:

```
+ <div style="display: table;">
  <div style="display: table-row;">
    <div style="display: table-cell;">
      Some content ...
    </div>
  </div>
</div>
```

# weitere Display-Typen

- + `display: grid;`
- + `display: flex;`
- + `display: flow-root;`
- + `display: run-in;` (nicht in Mozilla)

# Box Model Addition

- + `box-sizing: content-box`
- + Dimensionsverhalten wie in CSS 2 - Höhe, Breite werden definiert, padding, border, margin und outline werden **dazu** gerechnet.
- + `box-sizing: border-box`
- + In die Boxbreiten- und Höhenangabe werden padding und border **eingerechnet**.
- + `box-sizing: padding-box`
- + Padding mit in der Boxbreite, nicht aber der Rahmen.

# Universelles box-sizing mit Vererbung für alle Elemente und Pesudoelemente

- + Üblicherweise werden alle Elemente auf border-box umgestellt (base.css).

```
+ html {  
    box-sizing: border-box;  
}  
  
+ *,  
*:before,  
*:after {  
    box-sizing: inherit;  
}
```

# Appearance

- + Mit appearance kann man HTML-Elemente so aussehen lassen, dass sie wie Bedienelemente aussehen.
- + `appearance:`  
`checkbox | radio | push-button`  
`| square-button | button |`  
`button-bevel | listbox |`  
`listitem | menulist | menulist-`  
`button | menulist-text |`  
`menulist-textfield |`  
`scrollbarbutton-up |`  
`scrollbarbutton-down |`  
`scrollbarbutton-left |`  
`scrollbarbutton-right |`  
`scrollbartrack-horizontal | ...`
- + Um einen Link wie eine Checkbox aussehen zu lassen, macht man etwa folgendes:
- + `section a {`  
`appearance:checkbox;`  
`-webkit-appearance:checkbox;`  
`-moz-appearance:checkbox;`  
`text-indent: 1.2em;`  
`}`

<https://css-tricks.com/almanac/properties/a/appearance/>

# Schriften

2024 Michael Reichart

# Typografie und Layout

- + Im heutigen Schriftgebrauch gibt es zwei wichtige Schriftformen.
- + **Antiquaschriften** (Serifen-Schrift)
- + **Groteskschriften** (serifenlose Schrift).

Antiqua  
Grotesk

# Times Roman

- + Die Antiquaschriften sind serifen-betonte Schriften mit starken Unterscheidungen der Strichstärken.
- + Häufiger Einsatz für Texte in Büchern und Zeitungen.

- + Groteskschriften sind serifenlos, ohne oder mit geringen Unterschieden in der Strichstärke.
- + Häufiger Einsatz für Titel, Überschriften und in der Werbung.

# Helvetica

- + Schriftschnitte und Auszeichnungen dienen der Differenzierung der Schriftaussage, z. B. "Überschrift", "wichtig", "Zitat"
- + Mit Schriftschnitten erreicht man ein interessantes Schriftbild.
- + Manches sollte man wegen Unleserlichkeit vermeiden.

Extrafein

Fein

normal

**Halbfett**

**fett**

*kursiv*

Extraschmal

**schmal, fett**

**schmal, sehr fett**

~~kleinbuchstaben~~

**GROSSBUCHSTABEN**

~~KAPITÄLCHEN~~

~~gesperrt~~

Versalhöhe (Oberlänge)

Mittelhöhe

Unterlänge

Durchschuss (Zeilenabstand)

**Mixgetränk** Grundlinie

**Mixgetränk** Grundlinie  
Durchschuss (Zeilenabstand)

**Grundlinien-**  
**Raster im**  
**Verbund bilden**  
**die Basis für**  
**jedes Layout.**

# Schriftgrößen

## und Zeilenabstand

Sie werden so gewählt, dass sie

sich in ein einheitliches

Grundlinienraster einfügen.

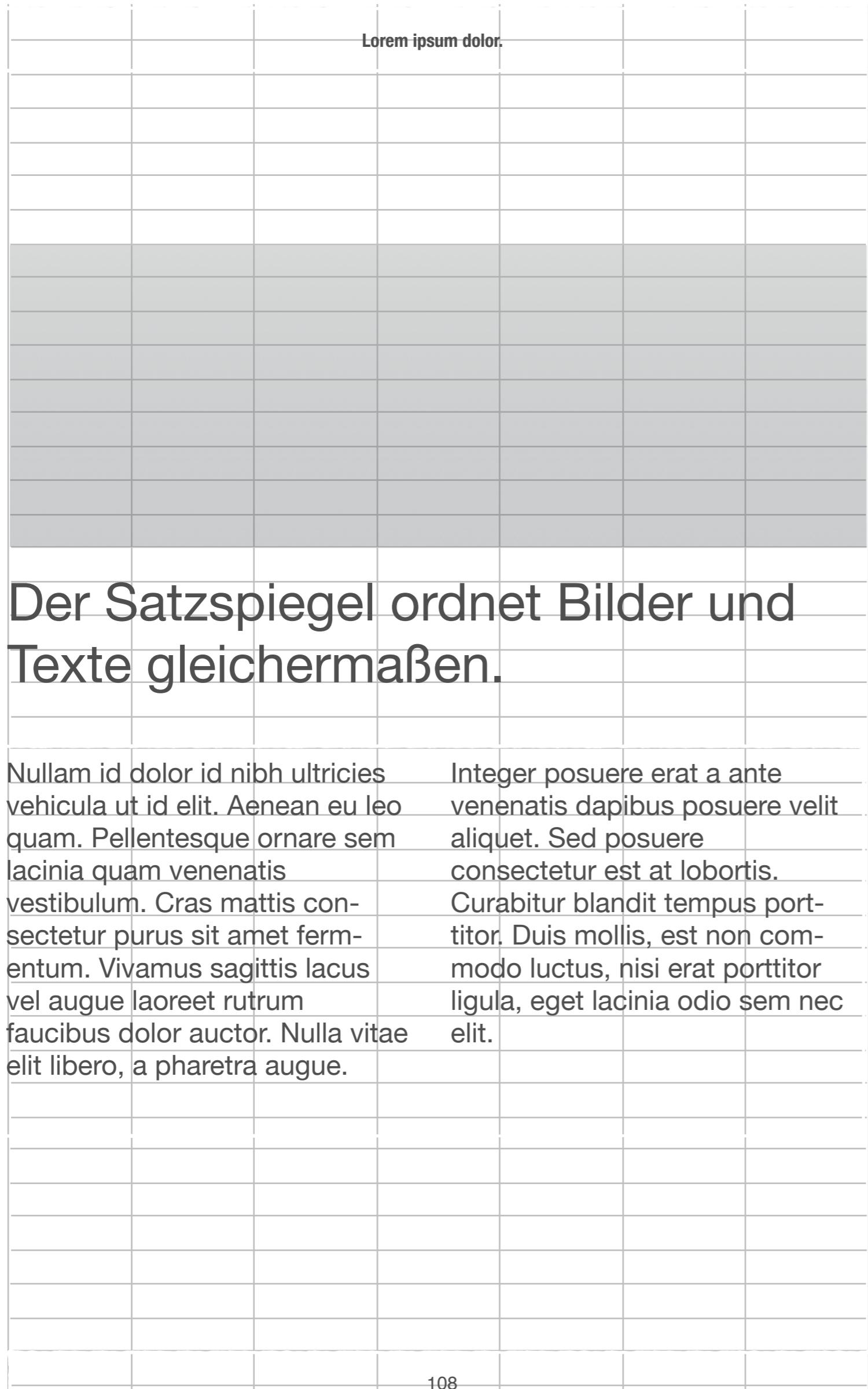
Dies erzeugt einen ruhigen

Rhythmus und Lesbarkeit.

# Rhythmus durch Grundlinien

Nullam id dolor id nibh ultricies  
vehicula ut id elit. Aenean eu leo  
quam. Pellentesque ornare sem  
lacinia quam venenatis  
vestibulum. Cras mattis con-  
sectetur purus sit amet ferm-  
entum. Vivamus sagittis lacus  
vel augue laoreet rutrum  
faucibus dolor auctor. Nulla  
vitae elit libero, a pharetra  
augue.

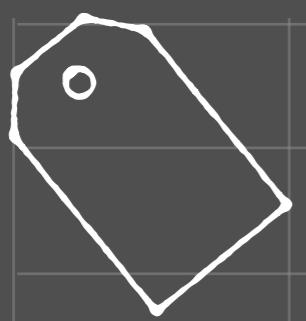
Integer posuere erat a ante  
venenatis dapibus posuere velit  
aliquet. Sed posuere  
consectetur est at lobortis.  
Curabitur blandit tempus port-  
titor. Duis mollis, est non com-  
modo luctus, nisi erat porttitor  
ligula, eget lacinia odio sem nec  
elit.



Der Satzspiegel ordnet Bilder und  
Texte gleichermaßen.

Nullam id dolor id nibh ultricies vehicula ut id elit. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum. Cras mattis consectetur purus sit amet fermentum. Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Nulla vitae elit libero, a pharetra augue.

Integer posuere erat a ante  
venenatis dapibus posuere velit  
aliquet. Sed posuere  
consectetur est at lobortis.  
Curabitur blandit tempus port-  
titor. Duis mollis, est non com-  
modo luctus, nisi erat porttitor  
ligula, eget lacinia odio sem nec  
elit.



Logo

search

### Fringilla

+  
Nullam id dolor id nibh  
ultricies vehicula ut id elit.

Aenean eu leo quam.

Pellentesque ornare sem lacinia quam  
venenatis vestibulum.

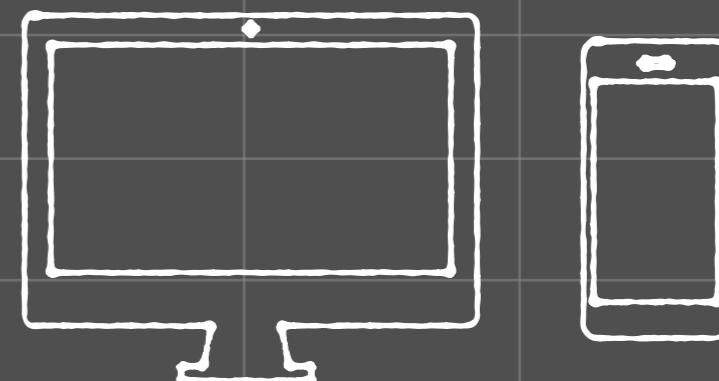
Cras mattis con-sectetur purus sit  
amet ferm-entum.

Vivamus sagittis lacus vel augue  
laoreet rutrum faucibus dolor auctor.  
Nulla vitae elit libero, a pharetra  
augue.

## Bildschirmraster verwenden sehr häufig zwölf Spalten.

Nullam id dolor id nibh ultricies  
vehicula ut id elit. Aenean eu leo  
quam. Pellentesque ornare sem  
lacinia quam venenatis vestibulum.  
Cras mattis con-sectetur purus sit  
amet ferm-entum. Vivamus sagittis  
lacus vel augue laoreet rutrum  
faucibus dolor auctor. Nulla vitae elit  
libero, a pharetra augue.

Integer posuere erat a ante venenatis  
dapibus posuere velit aliquet. Sed  
posuere consectetur est at lobortis.  
Curabitur blandit tempus port-titor.  
Duis mollis, est non com-modo  
luctus, nisi erat porttitor ligula, eget  
lacinia odio sem nec elit.

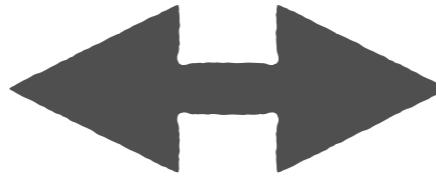
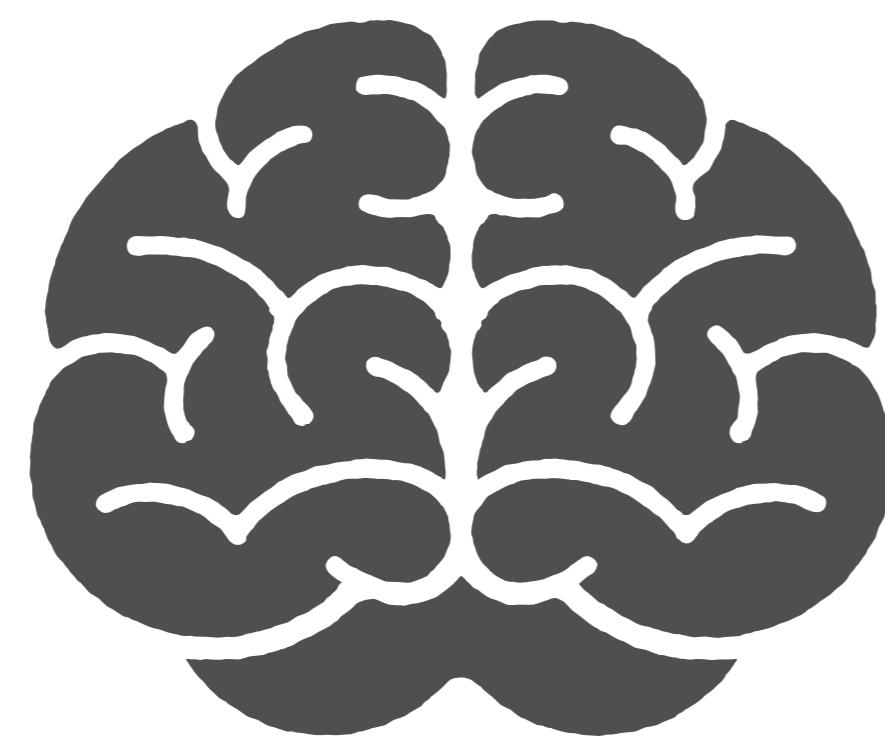


(c) 2017 Cras mattis con-sectetur purus sit amet ferm-entum.

Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Nulla vitae elit libero, a pharetra augue.

# Wie liest jemand?

Wie liest jemand?



# Wei leits jeamnd?

Wotre wrdeen als Mutsre odre

Konuetrn esrsfat. Deis ghet

sslbet aus dem Auegnwnikel

heuras, onhe akvties

Hinhascuen. Der Txet muss

ncoh nicht eimnal rchitig

gchrieseben sein.

# Wie liest jemand?

Ein Textabsatz wird selten Wort für Wort gelesen. Der Text wird eher gescannt. Am Anfang des Textes wird vielleicht die Zeile noch bis zum Ende gelesen, danach nur noch soviel, dass sich der Inhalt erschließt.

## Ebenso die Länge der Zeilen. Zu lange Zeilen werden oft nicht bis zu Ende gelesen.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Maecenas faucibus mollis interdum. Sed posuere consectetur est at lobortis. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Praesent commodo cursus magna, vel scelerisque nisl consectetur et. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec sed odio dui. Cras justo odio, dapibus ac facilisis in, egestas eget quam. Aenean lacinia bibendum nulla sed consectetur. Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Praesent commodo cursus magna, vel scelerisque nisl consectetur et. Cras mattis consectetur purus sit amet fermentum. Nullam quis risus eget urna mollis ornare vel eu leo. Aenean lacinia bibendum nulla sed consectetur. Cras justo odio, dapibus ac facilisis in, egestas eget quam.



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Maecenas faucibus mollis interdum. Sed posuere consectetur est at lobortis. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Praesent commodo cursus magna, vel scelerisque nisl consectetur et. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec sed odio dui.

Cras justo odio, dapibus ac facilisis in, egestas eget quam. Aenean lacinia bibendum nulla sed consectetur. Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Praesent commodo cursus magna, vel scelerisque nisl consectetur et. Cras mattis consectetur purus sit amet fermentum. Nullam quis risus eget urna mollis ornare vel eu leo. Aenean lacinia bibendum nulla sed consectetur. Cras justo odio, dapibus ac facilisis in, egestas eget quam.

# Richtiger Zeilenabstand ist wichtig, um den Zeilen folgen zu können

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Maecenas faucibus mollis interdum. Sed posuere consectetur est at lobortis. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Praesent commodo cursus magna, vel scelerisque nisl consectetur et.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Maecenas faucibus mollis interdum. Sed posuere consectetur est at lobortis. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Praesent commodo cursus magna, vel scelerisque nisl consectetur et.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Maecenas faucibus mollis interdum. Sed posuere consectetur est at lobortis. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Praesent commodo cursus magna, vel scelerisque nisl consectetur et.

# Schrifteinbindung

2024 Michael Reichart

# Einbindung von Zeichensätzen

```
@font-face {  
    font-family: LeagueGothic;  
    src: url(LeagueGothic.otf);  
}  
  
header {  
    font-family: LeagueGothic, Arial;  
}  
  
@font-face {  
    font-family: "Droid Sans";  
    src: url(Droid_Sans.ttf);  
}  
@font-face {  
    font-family: "Droid Sans";  
    src: url(Droid_Sans.woff2);  
}
```

# Manuelles Einbinden verschiedener Formate

```
@font-face {  
    font-family : OpenSans;  
  
    src         : url("/fonts/Open_Sans/OpenSans-Light.ttf")  
                  format("ttf"),  
                  url("/fonts/Open_Sans/OpenSans-Light.eot")  
                  format("eot");  
    font-weight : 300;  
  
    src         : url("/fonts/Open_Sans/OpenSans-Regular.ttf");  
    font-weight : 400;  
  
    src         : url("/fonts/Open_Sans/OpenSans-Semibold.ttf");  
    font-weight : 600;  
}
```

# Schriftschnitte

100 thin, extrafein  
200  
300 light (fein)  
400 regular (normal)  
500  
600 semibold  
(halbfett)  
700 bold (fett)  
800 black  
900 heavy

Extrafein

Fein

normal

**Halbfett**

**fett**

*kursiv*

Extraschmal

**schmal, fett**

**schmal, sehr fett**

# Google Fonts API

```
@import url(http://fonts.googleapis.com/css?  
family=Roboto:100,300,400,600,700,900);
```

## \* Schrift als Iconset nutzen

```
@font-face {  
    font-family : anyIconFont;  
  
    src         : url("/fonts/anyIconFont.ttf")  
                  format("ttf");  
}  
  
.icon-asterisk::before {  
    content: "\2a";           /* der UCS2 code für ein  
Zeichen */  
}  
.icon-plus::before {  
    content: "\2b";  
}  
  
<span class="icon icon-asterisk"></span>
```

# Die Größe von Zeichensätzen justieren

```
p.beispiel {  
    font-family:  
        Verdana, Arial, sans-serif;  
    font-size:12px;  
    font-size-adjust:0.58;  
}
```

Mit `font-size-adjust` kann man eine bestimmte x-Höhe erzwingen, ganz egal welche Schrift aus `font-family` eingesetzt wird. Im Beispiel wird die Schriftliste auf den Adjustwert der Verdana eingestellt.

Siehe auch

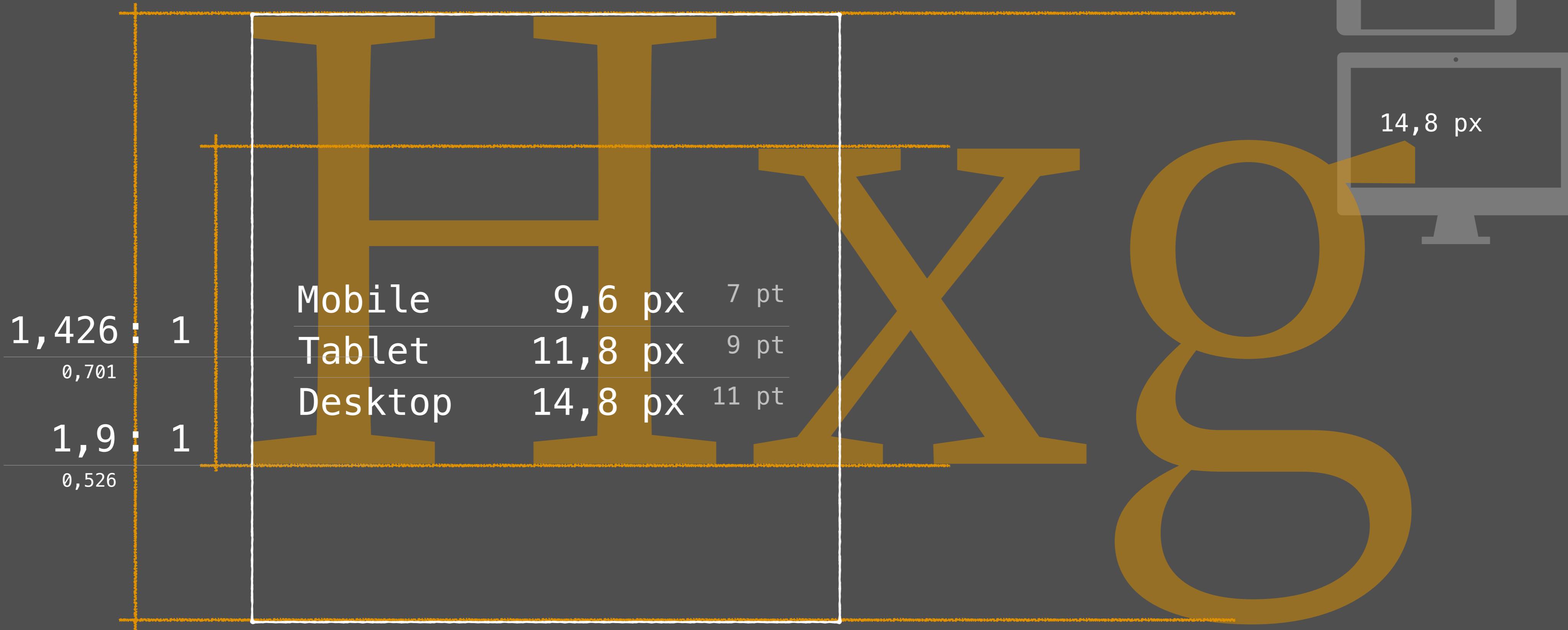
<http://www.webspaceworks.com/resources/fonts-web-typography/43/>

# Textkonturen

```
h1 {  
    text-shadow: 0 0 1px black;  
}
```

	Unterlän	Grundli	Majuskel	Minuskel
	30,88	25,55	9,94	14,53
Majuskelhöhe		15,61		
Mittelänge	11,02	1,41652		
Schriftgröße	20,94	1,9002		

# Beispiel: "Noto Serif"



Auflösung	155 dpi	150 dpi	100 dpi
Abstand	30 cm	40 cm	70 cm
Visus	0,7	0,7	0,7
Mindestlänge	6,7 px	8,3 px	10,4 px

1 pt    1,3 px  
1 inch    2,54 cm

# Mehrspaltensatz

2024 Michael Reichart

# Mehrspaltensatz - das Multicolumn Model

```
+ div { column-count: 3      }  
  
+ div { column-width: 12rem }  
  
+ div { columns: 3 12rem    }  
  
+ div {  
  column-gap: 1rem;  
  column-rule-width: 1px;  
  column-rule-style: solid;  
  column-rule-color: black;  
}
```

Nulla vitae elit libero, a pharetra augue. Nullam quis risus eget urna mollis ornare vel eu leo. Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus.

Nulla vitae elit libero, a pharetra augue. Integer posuere erat a ante venenatis dapibus posuere velit aliquet. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Nullam id dolor id nibh ultricies vehicula ut id elit.

Cras mattis consectetur purus sit amet fermentum. Nullam id dolor id nibh ultricies vehicula ut id elit. Aenean lacinia bibendum nulla sed consectetur. Maecenas faucibus mollis interdum. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor.

# Umbruchsteuerung bei Mehrspaltensatz

```
+ h1 {  
    break-before: column;  
    break-inside: avoid-column;  
    break-after: avoid-column;  
}  
  
+ div {  
    width: 100px;  
    column-width: 45px;  
    column-gap: 0;  
    column-rule: none;  
}
```

# Silbentrennung

- + p {  
    hyphens : none | manual | auto;  
    hyphenate-character : '=';  
    hyphenate-limit-chars : 10 4 4  
                              | auto;  
}
  
  - + limit-chars ->
    1. Mindestwortlänge
    2. Mindestanzahl der Buchstaben vor der Trennung
    3. Mindestanzahl der Buchstaben nach der Trennung
  
  - + manual -> &shy;  
none   -> keine Trennungen
  
  - + auto   -> trennt, wo getrennt werden muss.
- + **Lang-Attribut verwenden, damit die Trennung nach Sprachregeln funktioniert!**

# Erzwungener Wortumbruch

- + Für Wörter mit Überlänge und ohne Trennregel:
- + 

```
p {  
    word-wrap: normal | break-word;  
}
```
- + XML&shy;Http<wbr>Request

# Alle Satzregeln auf einen Blick

```
+ p {  
    text-align      : justify | left | center | right;  
  
    text-justify   : inter-word | inter-character | auto | none;  
    [word-spacing   : normal;]  
    [letter-spacing: normal;]  
  
    hyphens        : auto;  
  
    widows         : 3;  
    orphans         : 3;  
  
    word-break     : normal | break-all | keep-all;  
  
    hanging-punctuation: first last;  
}
```

# Texte abschneiden

- + 

```
div {  
    text-overflow: ellipsis;  
}
```
- + Nur bei einzeiligen  
Texten!

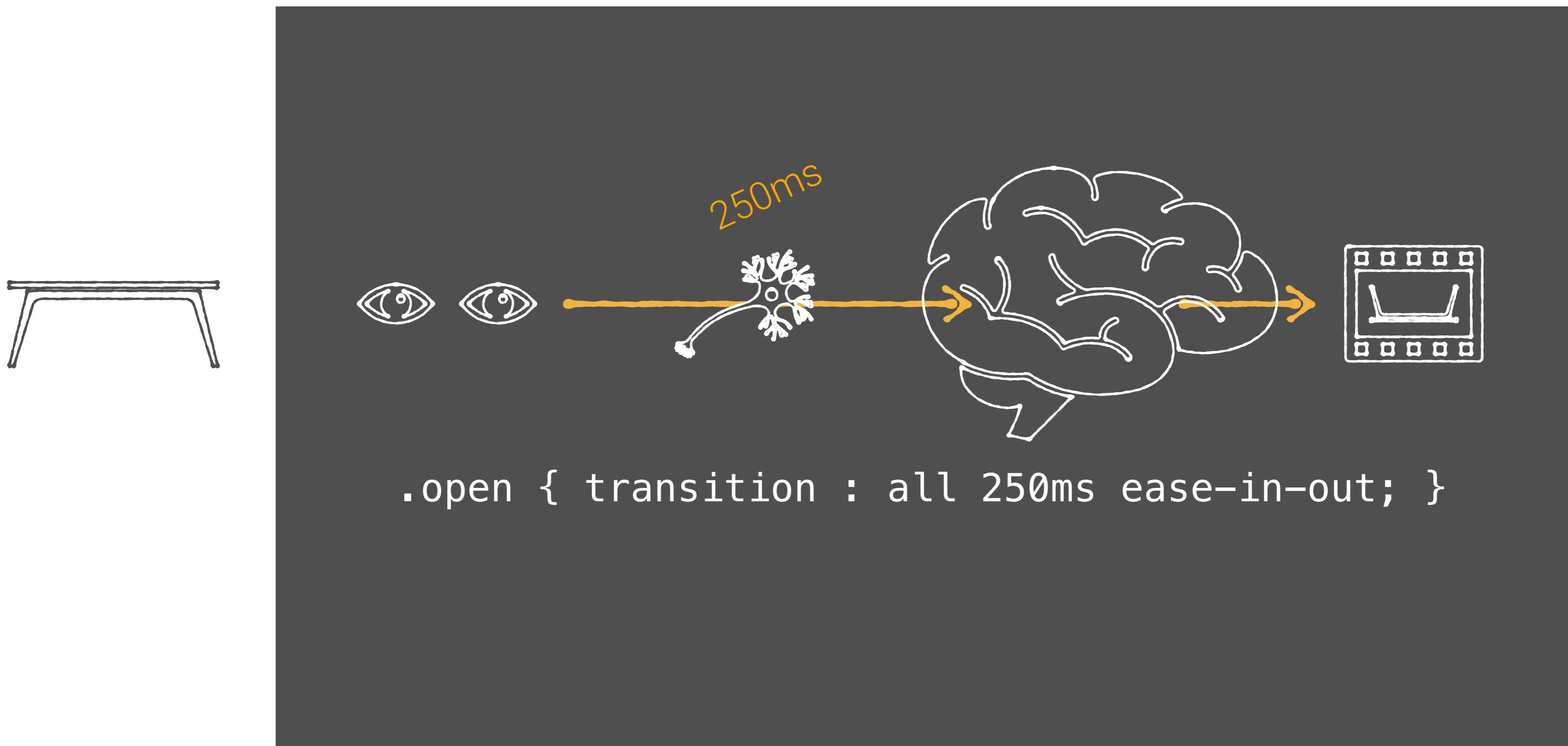


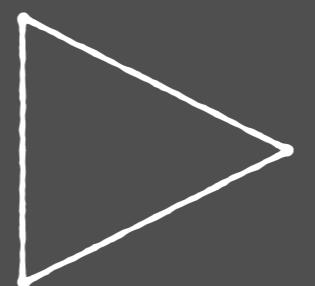
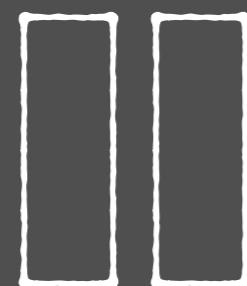
Text im Eleme ...

# Animationen und Übergänge

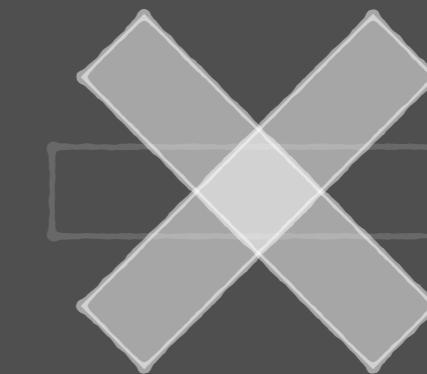
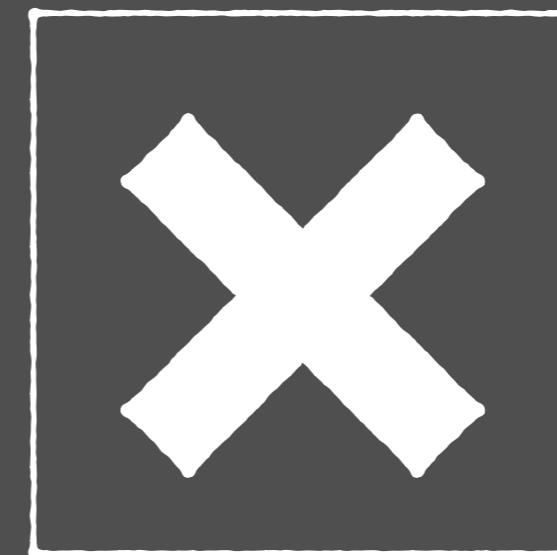
2024 Michael Reichart

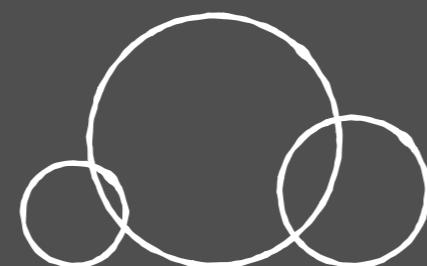
Erkenntnis ist kein Ereignis,  
sondern ein Prozess!



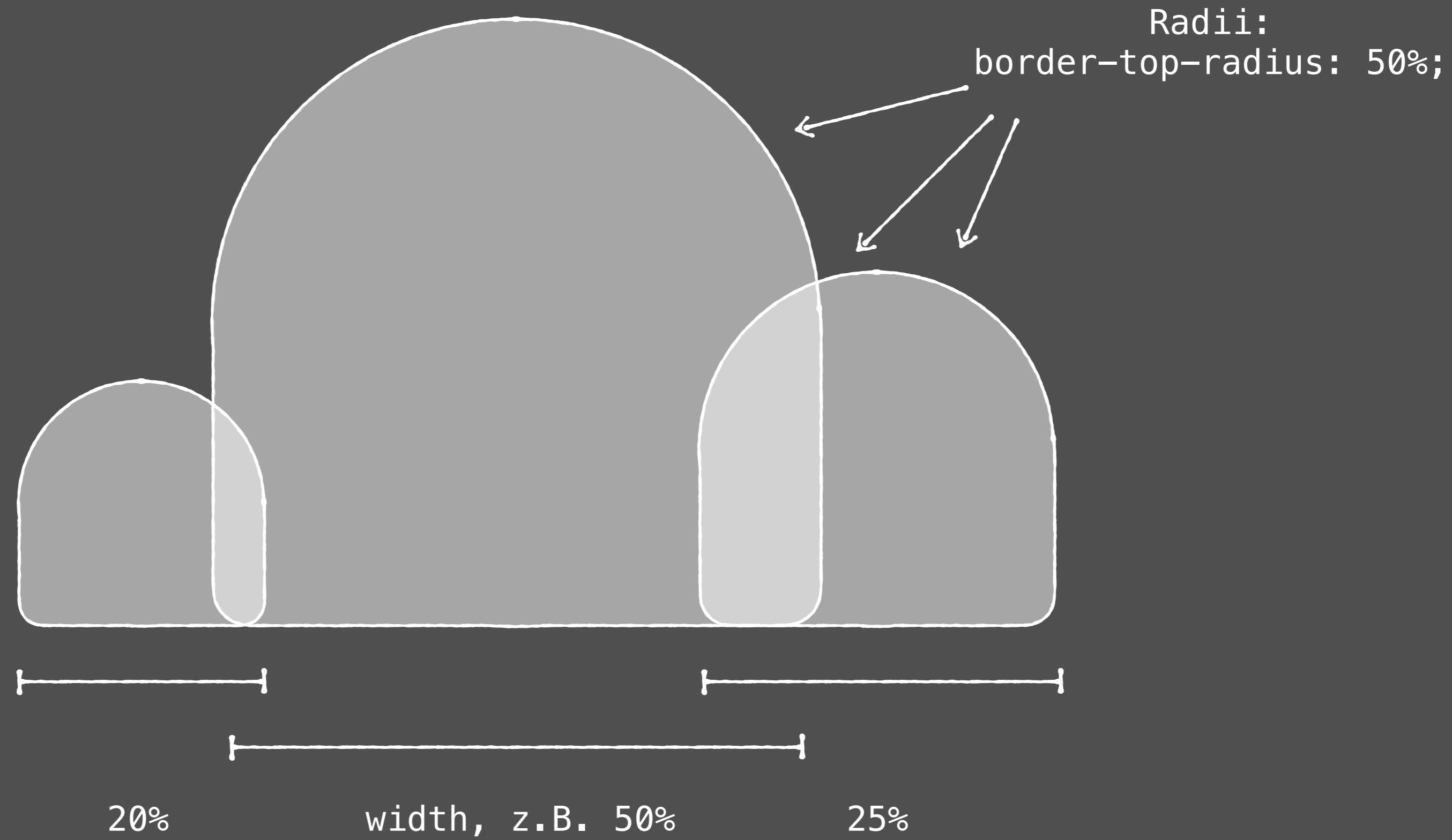


Transformation eines Hamburger Menüs.





## Konstruktion einer Wolke mit CSS



# CSS Transitions

- + Eine CSS Transition ändert den Wert einer CSS-Eigenschaft über eine festgelegte Zeit.
- + Dabei entstehen Animationen wie das Ein- und Ausblenden und Bewegen von HTML-Elementen ohne Javascript oder Flash.

# Aufbau einer Transition

```
#box.left { margin-left: 0; }
#box.right { margin-left: 1000px; }

#box {
    transition : margin-left 1s ease-in-out;
}

document.getElementById( 'box' ).className = 'left';
document.getElementById( 'box' ).className = 'right';
```

# transition property, duration, timing

**transition-property**

Die Eigenschaft, die während des Übergangs geändert wird.

**transition-duration**

Die Zeit, in der ein Übergang stattfinden soll.

**transition-delay**

Zeit bis zum Start des Übergangs.

**transition-timing-function**

Zeitlicher Verlauf des Übergangs.

# Ein aufklappendes Menu

```
.content-nav {  
    display : block;  
    height  : 0;  
    overflow: hidden;  
  
    transition-property : height;  
    transition-duration : 0.25s;  
    transition-timing-function : ease-in-out;  
}  
.menu-show {  
    display : block;  
    height  : 5em;  
}  
  
$('button#menu-toggle').on('click', function () {  
    $('.content-nav').toggleClass('menu-show');  
});
```

# Übergänge mit States

```
<div class="button"><a href="#">Mit Transition</a></div>  
...  
.button a {  
    background-color: #9FC0D0;  
    transition-property: background-color;  
    transition-duration: 1s; }  
  
.button a:hover { background-color: #759DB2; }
```

# Die Easing Effekte

## **ease**

Ohne weitere Angaben starten Transitions mit CSS langsam, beschleunigen in der Mitte und werden am Ende wieder langsamer.

## **linear**

ist eine gleichförmiger Übergang,

## **ease-in**

beginnt langsam und wird schneller,

## **ease-out**

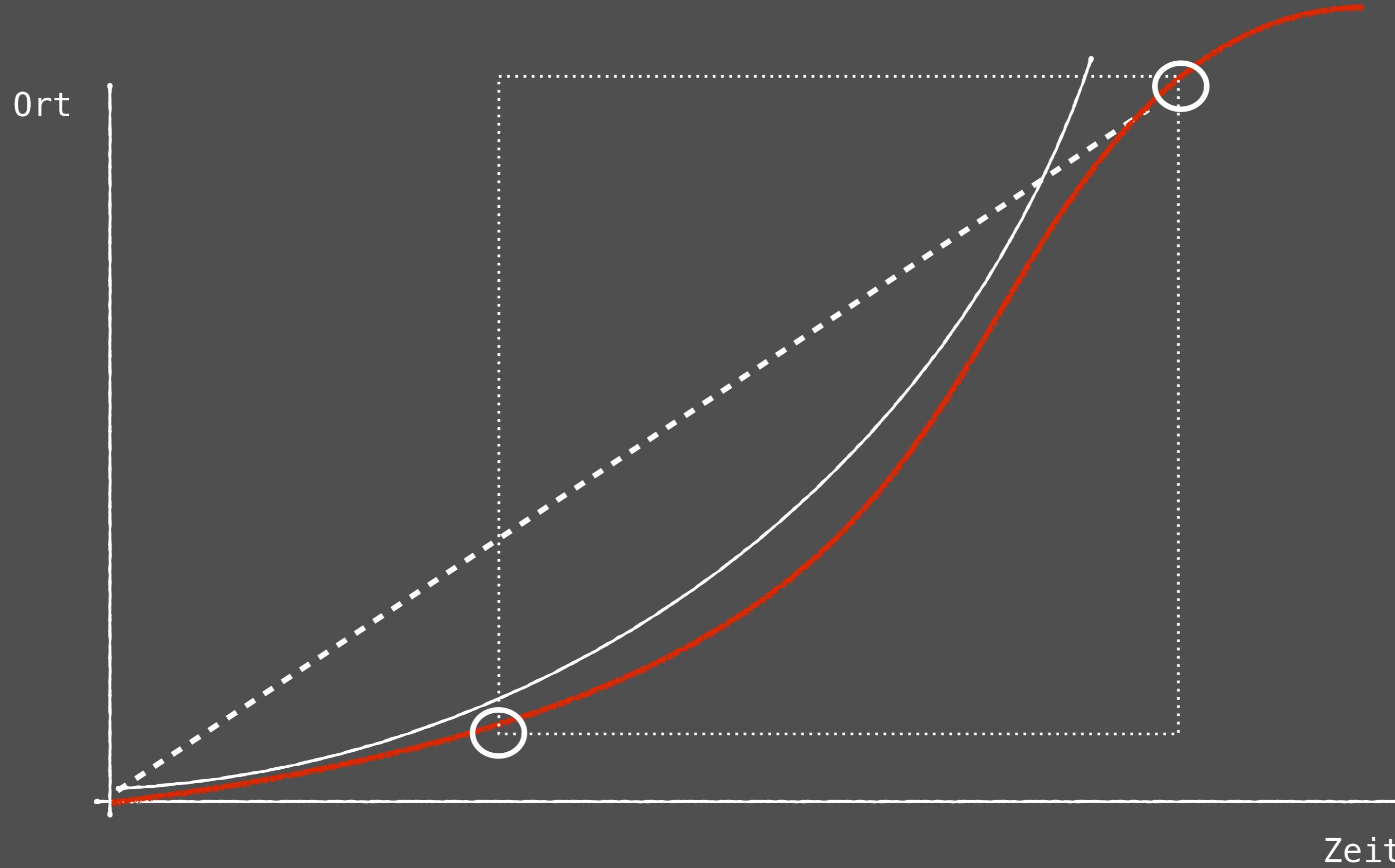
beginnt schnell und läuft langsamer aus

## **ease-in-out**

kombiniert das langsame Anlaufen mit einem langsamen Auslaufen, aber einem deutlich schnelleren Teil in der Mitte.

## **cubic-bezier(x1,y1,x2,y2)**

ist Fine-Tuning für den Verlauf der Animation



# Animationen

```
@keyframes my-animation {  
    0% { background-color : black; }  
    25% { background-color : red; }  
    50% { background-color : green; }  
    75% { background-color : yellow; }  
    100% { background-color : black; }  
}  
  
.background {  
    animation: my-animation 4s infinite;  
}
```

# Transformationen

```
transform : rotateZ(45deg);
transform : scaleX(2.5%);
transform : translate3d(0, 0, 90px);
transform : perspective(500px)

#threed-example {
    transform : rotateZ(5deg);
    transition : transform 2s ease-in-out;
}

#threed-example:hover {
    transform : rotateZ(-5deg);
}
```

# Farben und Farbverläufe, image processing

2024 Michael Reichart

# Responsives Logo

```
<div class="brand"><h1 class="sr-only">Company Name</h1></div>

.brand {
    width      : 100%;
    height     : 100%;
    background : url(logo.svg) center center no-repeat;
    background-size   : contain; /* cover */
}

.sr-only {
    display   : block;
    position  : absolute;
    left     : -10000px
}
```

# Responsiver Vollbild Hintergrund

```
body {  
    background : url(background.jpg) center center  
no-repeat;  
    background-size : cover;  
}
```

# Background Stacking

```
div {  
  background :  
    rgba(200,0,0,0.25),  
    linear-gradient(  
      to bottom,  
      #1e5799 0%,  
      #2989d8 50%,  
      #207cca 51%,  
      transparent 100%  
    );  
  repeat,  
  repeat-x;  
}url(src/overlay.png) 10px center no-  
url(src/background.png) 10px center
```

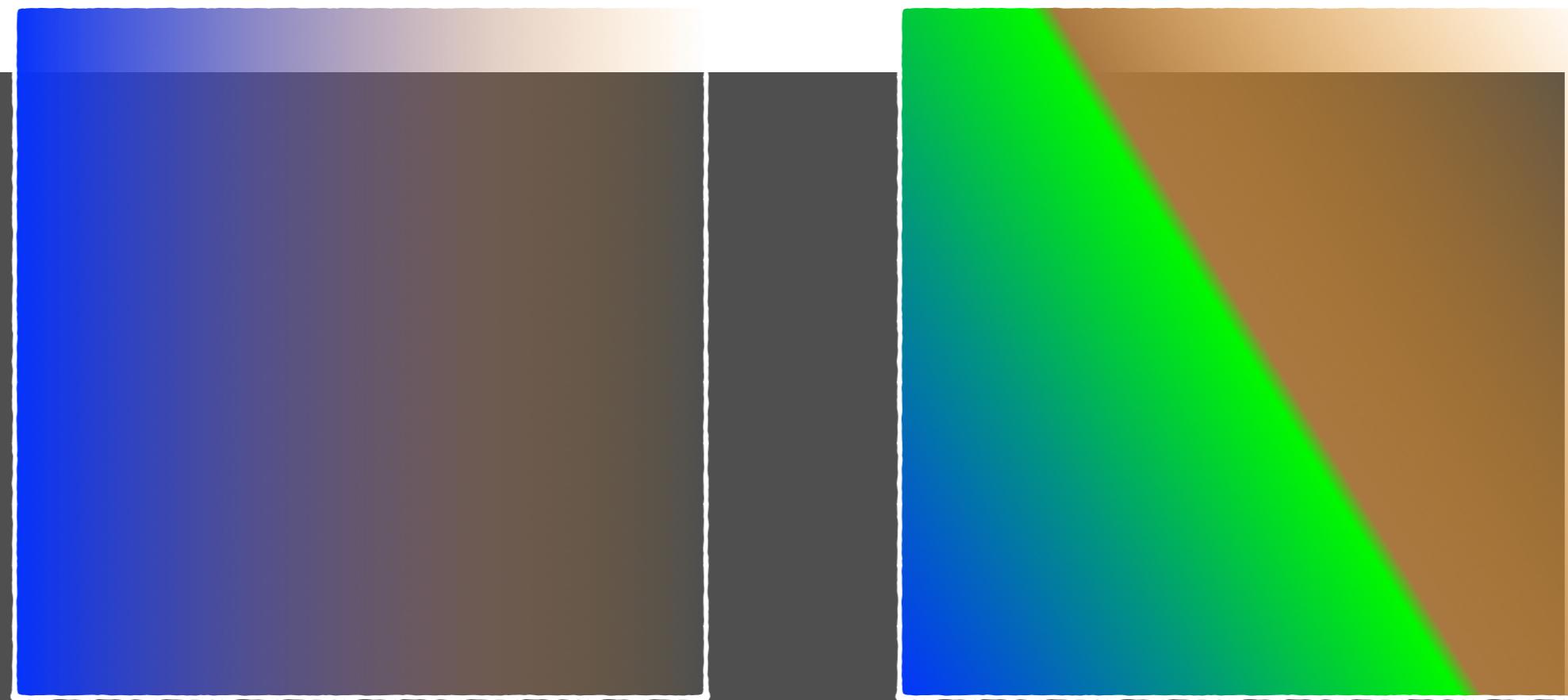
# Neue Farbmodelle mit Transparenz

Für Farben kann nun eine Deckkraft eingestellt werden. Der vierte Wert steht für einen Alpha-Kanal. Im Gegensatz zu Opacity, welches jeweils für das ganze Element auswirkt, können mit den Alpha Farben Fläche, Rand oder Text einzeln eingestellt werden.

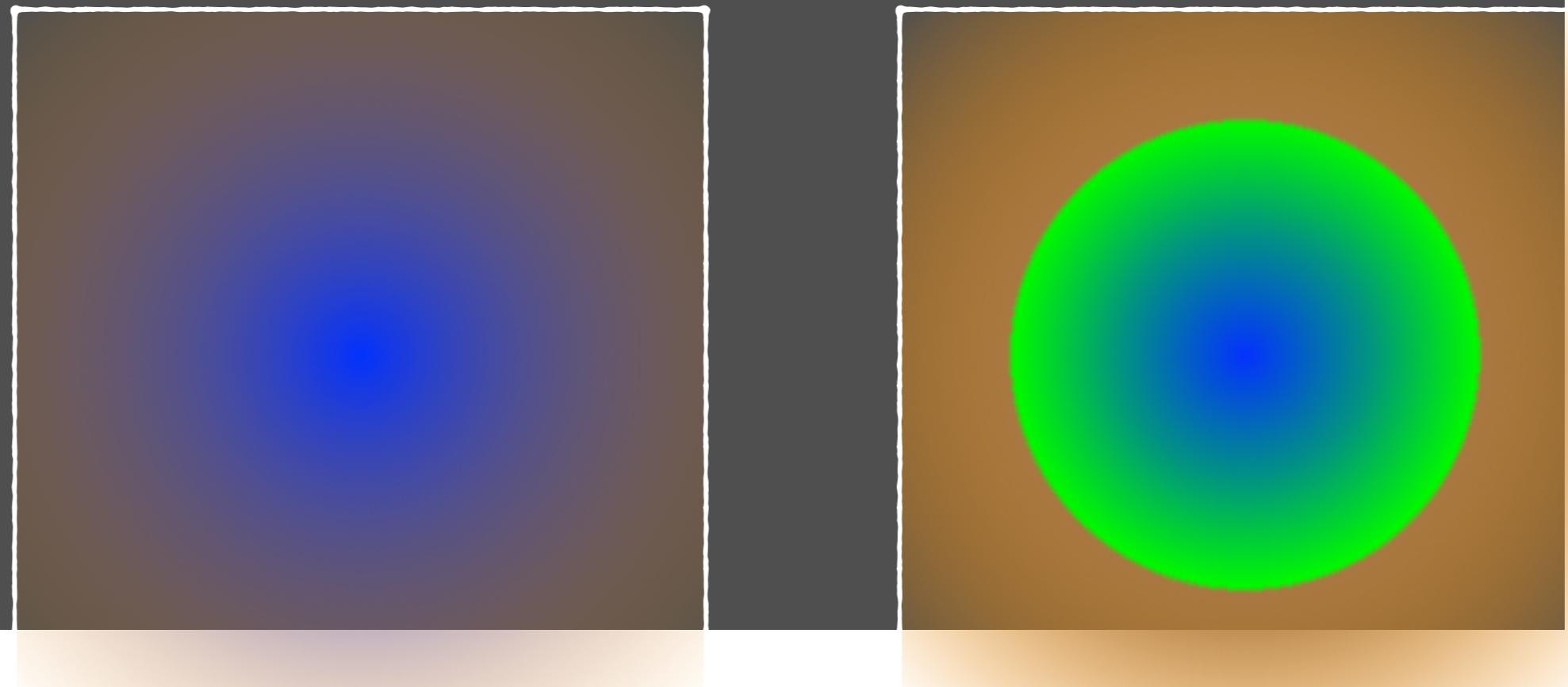
```
color: rgba(255, 0, 0, 0.75);  
background-color: hsla(360, 100%, 50%, 0.75);
```

# Farbverläufe - linear und radial

```
background:  
linear-gradient(  
    32deg,  
    blue 0%,  
    green 50%,  
    brown 51%,  
    transparent 100%  
) ;
```



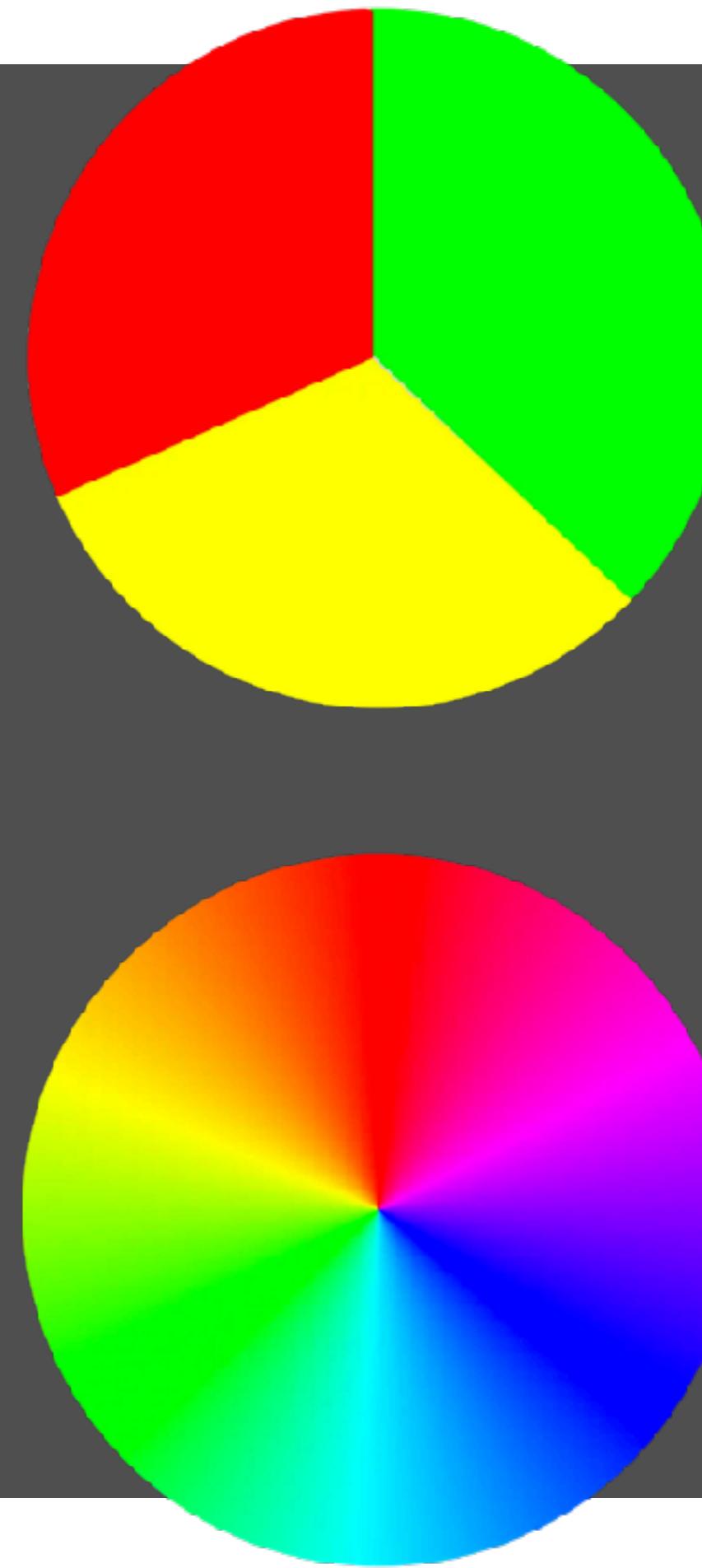
```
background:  
radial-gradient(  
    ipat center,  
    blue 0%,  
    green 50%,  
    brown 51%,  
    transparent 100%  
) ;
```



# Konischer Farbverlauf

```
.conic-gradient {  
  background:  
    conic-gradient(  
      lime 40%,  
      yellow 0 70%,  
      red 0);  
}
```

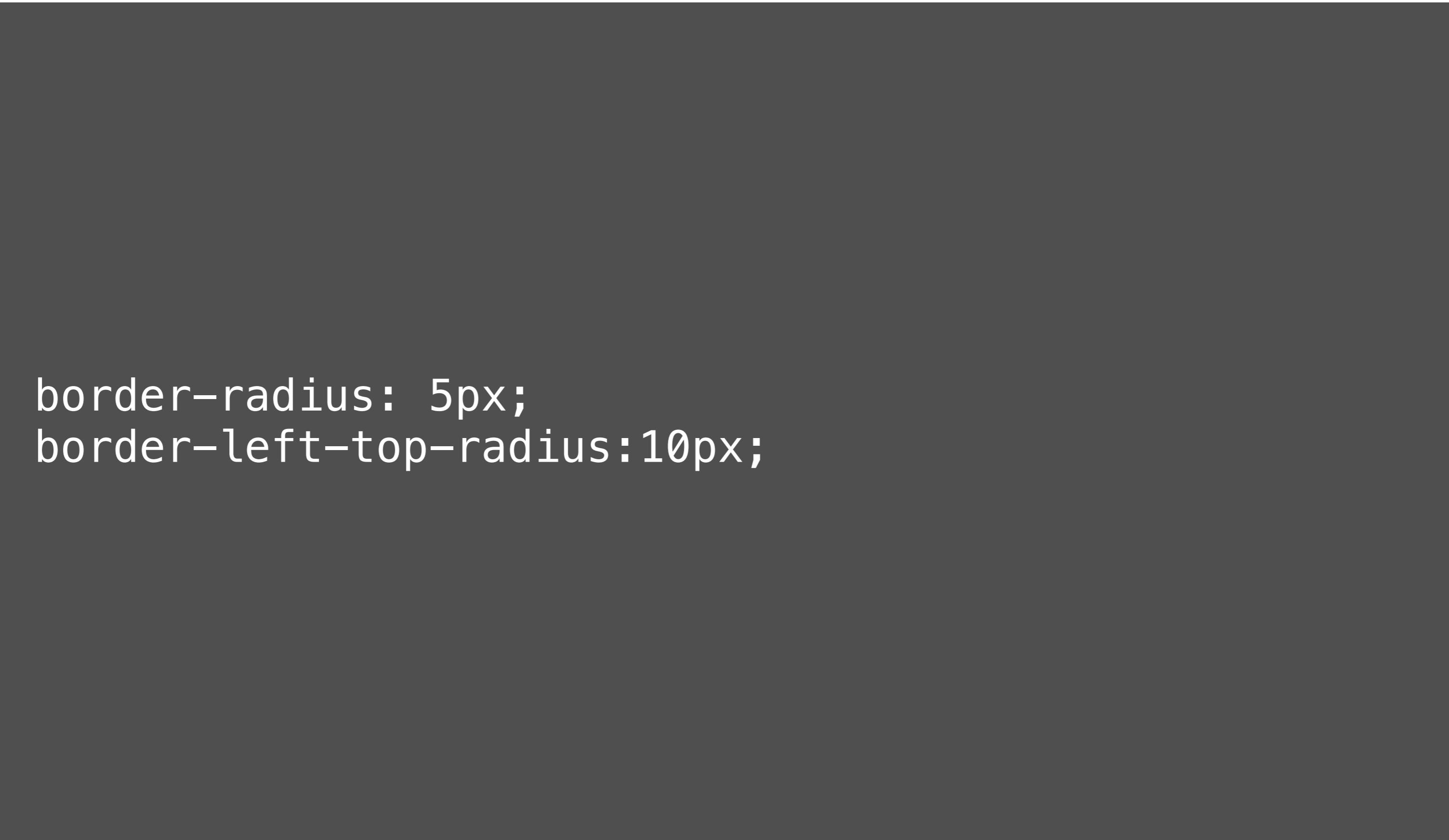
```
.conic-gradient {  
  background:  
    conic-gradient(  
      red,  
      yellow,  
      lime,  
      aqua,  
      blue,  
      magenta,  
      red);  
}
```



<https://css-tricks.com/css3-gradients/>

Unbedingt lesen!

# Abgerundete Ecken



```
border-radius: 5px;  
border-left-top-radius:10px;
```

# Schatten

- + **text-shadow:**  
10px /\* X Versatz \*/  
10px /\* Y Versatz \*/  
10px /\* Blurradius\*/  
rgba(64, 64, 64, 0.5)  
;
- + **box-shadow:**  
10px /\* X Versatz \*/  
10px /\* Y Versatz \*/  
10px /\* Blurradius\*/  
rgba(0, 0, 128, 0.25);



Das ist Text mit  
Schlagschatten

# Reflektion

```
box-reflect:  
below | above | left | right  
5px;  
  

```



# Filter Funktionen

```
filter : blur(5px)
         brightness(0.2)
         saturate(50%)
         hue-rotate(30deg)
         contrast(125%)
         invert(100%)
         greyscale(100%)
         sepia(100%)

;

// Ggf. müssen Browser-Prefixes verwendet werden:
// -moz-filter, -webkit-filter, -o-filter, -ms-
filter
```

<https://codepen.io/grayghostvisuals/pen/Cacib>



<https://css-tricks.com/almanac/properties/f/filter/>

# BackgroundMischmodus

```
.blended {  
    background-image : url(face.jpg);  
    background-color : red;  
    background-blend-mode : multiply;  
}  
  
// Weitere Modus sind : normal , multiply, screen,  
// overlay,  
// darken, lighten, color-dodge, saturation, color,  
// luminosity
```

# ElementMischmodus

```
// Für Elemente gilt die Eigenschaft  
mix-blend-mode : normal;
```

Die Modus sind ähnlich: multiply, screen, overlay,  
darken,  
lighten, color-dodge ...

<https://css-tricks.com/basics-css-blend-modes/>

# Filter Funktionen

```
.element {  
  backdrop-filter: blur(5px) contrast(0.8);  
}
```

Properties:

- blur()
- brightness()
- contrast()
- drop-shadow()
- grayscale()
- hue-rotate()
- invert()
- opacity()
- saturate()
- sepia()
- url()



`backdrop-filter: grayscale(1)`

`backdrop-filter: brightness(1.5)`

`backdrop-filter: blur(5px)`

`backdrop-filter: contrast(1.5)`

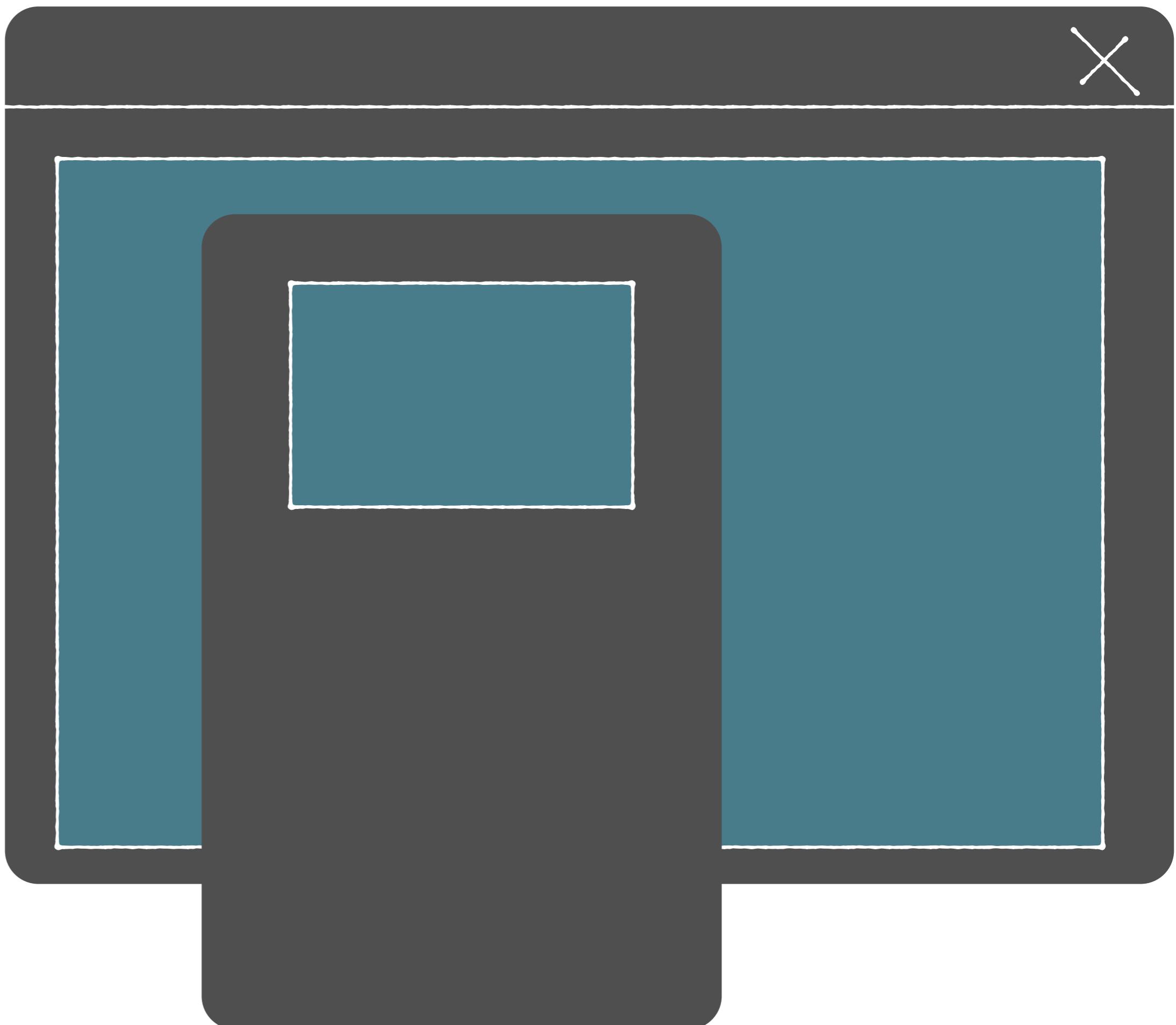
<https://css-tricks.com/the-backdrop-filter-css-property/>

# Responsive Bilder

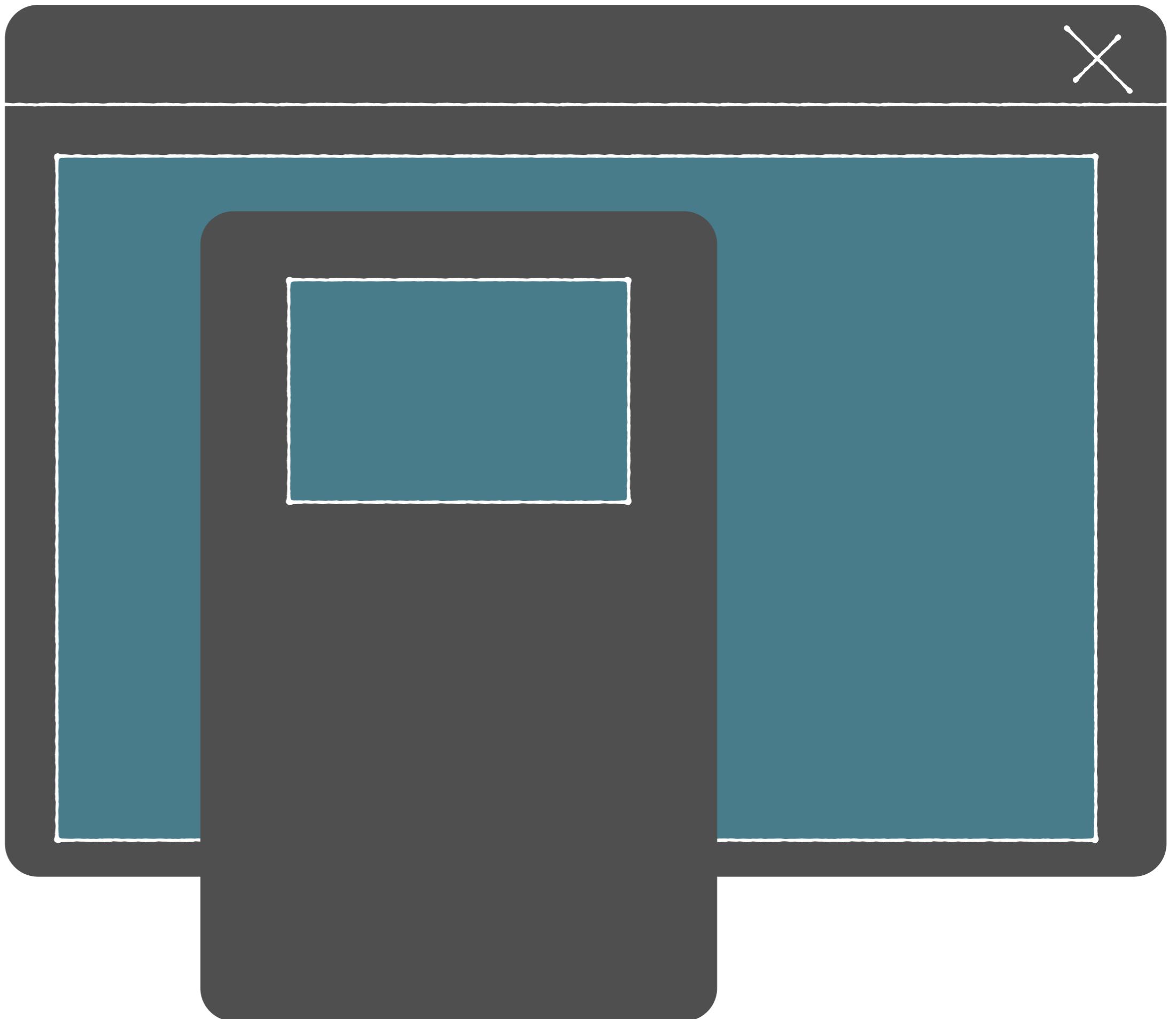
2024 Michael Reichart

# Bilder responsive bemassen

```
+ .portrait {  
    height : auto;  
    max-width : 100%;  
}
```

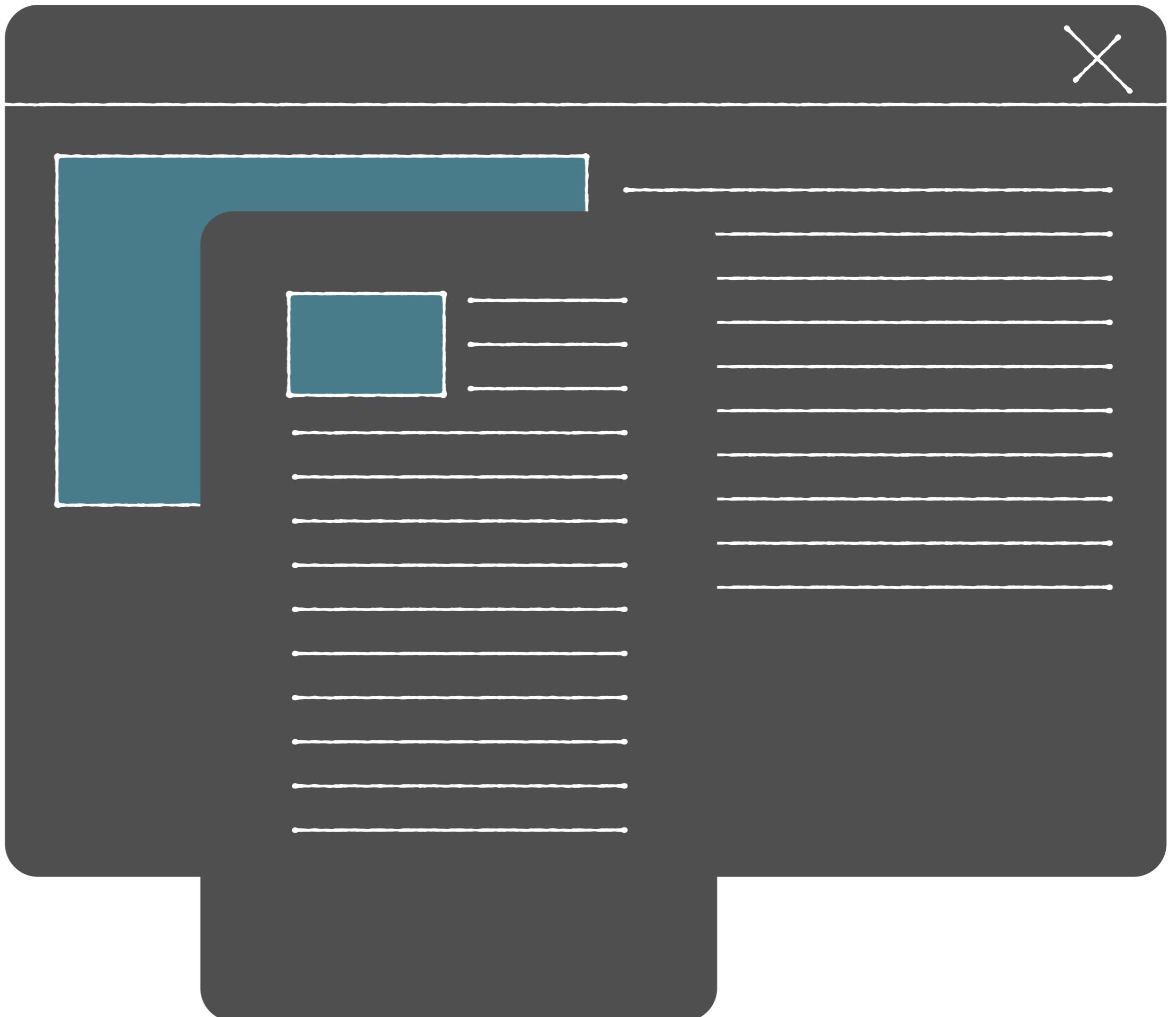


- + Das geht auch mit embed, object, video, canvas!



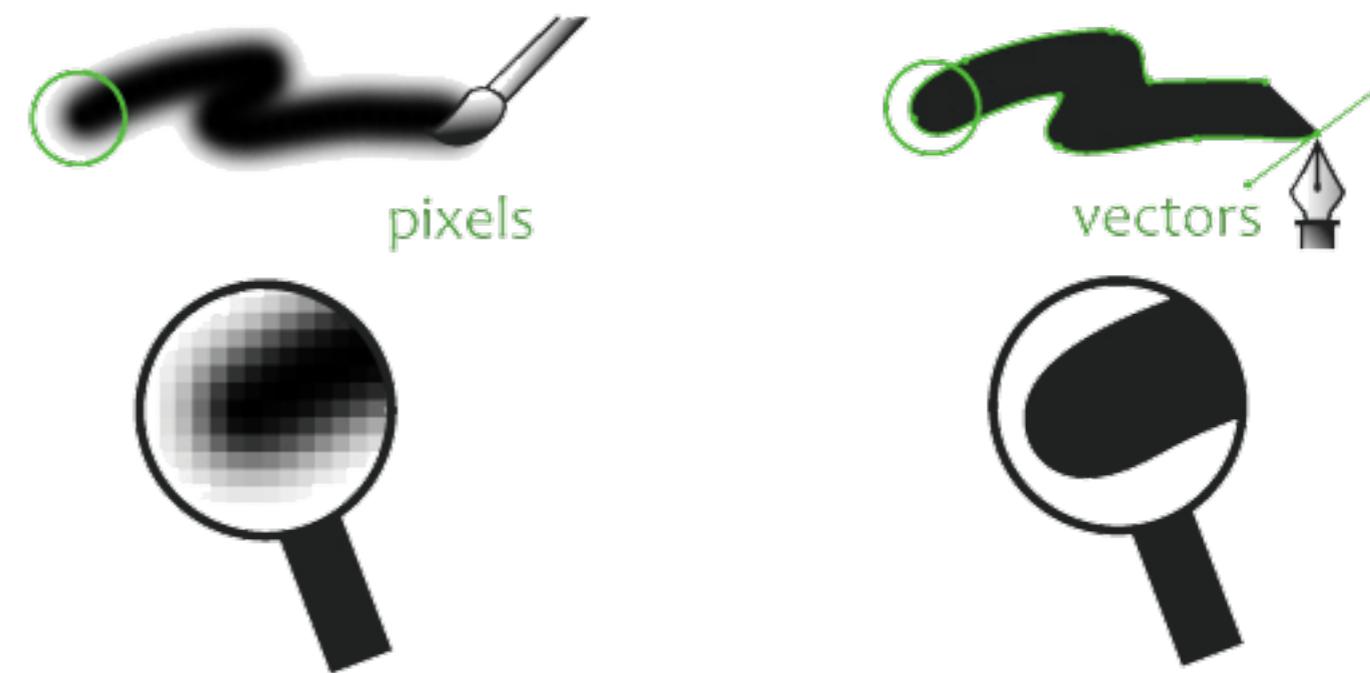
# "Floating"-Bilder responsive bemassen

```
+ .portrait {  
    float: left;  
    height: auto;  
    max-width: 40%;  
    margin: 0 5% 2.5% 0  
}
```



# Auflösungen beachten - Die Grenzen beim Skalieren von Bildern

- + Bilder lassen sich allerdings weder vergrößern, noch beliebig verkleinern. Ein Pixel ist ein Pixel, und wird davon nur die Hälfte gezeigt, geht Qualität verloren.



# Das <picture> Element

Für die Platzierung von Bildern wird ein neues <picture>-Element eingeführt. Damit können auflösungsabhängig verschiedene Bildformate bereitgestellt werden. Dabei müssen die Bilder vom Größten zum Kleinsten hin eingebunden werden.

```
<picture>

    <source media="(min-width: 1200px)" srcset="1440px.jpg">
    <source media="(min-width: 992px)" srcset="1199px.jpg">
    <source media="(min-width: 768px)" srcset="991px.jpg" >
    <source media="(min-width: 481px)" srcset="767px.jpg" >
    <source media="(max-width: 480px)" srcset="480px.jpg" >

        
</picture>
```

# Das srcset Attribut

Die srcset Attribute halten verschiedenen Bildquellen bereit, von denen eine nach der Auswertung des Browsers ausgewählt wird und anstelle der src Angabe im img Element verwendet wird.

```
<picture>
  <source media="(min-width: 1200px)" srcset="1440px.jpg">
  <source media="(min-width: 992px)" srcset="1199px.jpg">
  <source media="(min-width: 768px)" srcset="991px.jpg" >
  <source media="(min-width: 481px)" srcset="767px.jpg" >
  <source media="(max-width: 480px)" srcset="480px.jpg" >
  
</picture>
```

# Das Fallback Bild

Trifft keine der srcset Bilder zu oder kann ein Browser das picture/srcset nicht auswerten, dann wird das img Element als Fallback verwendet.

```
<picture>
  <source media="(min-width: 1200px)" srcset="1440px.jpg">
  <source media="(min-width: 992px)" srcset="1199px.jpg">
  <source media="(min-width: 768px)" srcset="991px.jpg" >
  <source media="(min-width: 481px)" srcset="767px.jpg" >
  <source media="(max-width: 480px)" srcset="480px.jpg" >
  
</picture>
```

# <picture>

Ausser der gerätespezifischen Einstellung kann das <picture> Element auch zur Auswahl aus mehreren alternativen Bildformaten verwendet werden.

```
<picture>
  <source srcset="/uploads/100-marie-lloyd.webp"
          type="image/webp">
  <source srcset="/uploads/100-marie-lloyd.jxr"
          type="image/vnd.ms-photo">

  
</picture>
```

# Retina Displays für <picture>

```
<picture>

    <source media="(min-width: 2400px) and
                  (-webkit-min-device-pixel-ratio: 2) or
                  (min-resolution: 192dpi)" srcset="2880px.jpg">

    <source media="(min-width: 1200px)" srcset="1440px.jpg">
    <source media="(min-width: 992px)" srcset="1199px.jpg">
    <source media="(min-width: 768px)" srcset="991px.jpg" >
    <source media="(min-width: 481px)" srcset="767px.jpg" >
    <source media="(max-width: 480px)" srcset="480px.jpg" >
    
</picture>

@media
(-webkit-min-device-pixel-ratio: 2),
(min-resolution: 192dpi) {
    /* Retina-specific stuff here */
}
```

##  srcset-w

Ebenfalls für auflösungsabhängig Bildformate kann das srcset-Attribut verwendet werden. Im Gegensatz zur picture Gruppe wertet das srcset-Attribut browserseitig und selbstständig aus, welche Bilddatei unter gegebenen Bedingungen die beste ist.

srcset-w:

```

```

#  srcset-x

Mit der x-Angabe können Bilder Displayauflösungsabhängig hinterlegt werden. So lassen sich Retina-fähige Bilder hinterlegen.

srcset-x:

```

```

[https://css-tricks.com/  
responsive-images-youre-just-changing-  
resolutions-use-srcset/](https://css-tricks.com/responsive-images-youre-just-changing-resolutions-use-srcset/)

[https://www.smashingmagazine.com/  
2014/05/responsive-images-done-right-  
guide-picture-srcset/](https://www.smashingmagazine.com/2014/05/responsive-images-done-right-guide-picture-srcset/)

[https://css-tricks.com/snippets/css/  
retina-display-media-query/](https://css-tricks.com/snippets/css/retina-display-media-query/)

# Picturefill

## A responsive image polyfill

- + The picture element and associated features are W3C standard HTML features that allow web developers to deliver an appropriate image to every user depending on a variety of conditions like screen size, viewport size, screen resolution, and more.
- + Picturefill is a JavaScript file (or a polyfill to be more specific) that enables support for the picture element and associated features in browsers that do not yet support them, so you can start using them today!

# Serverside: adaptive-images.php

- + Adaptive Images detects your visitor's screen size and automatically creates, caches, and delivers device appropriate re-scaled versions of your web page's embedded HTML images. No mark-up changes needed. It is intended for use with Responsive Designs and to be combined with Fluid Image techniques.

<http://www.adaptive-images.com>

# Scalable Vector Graphics

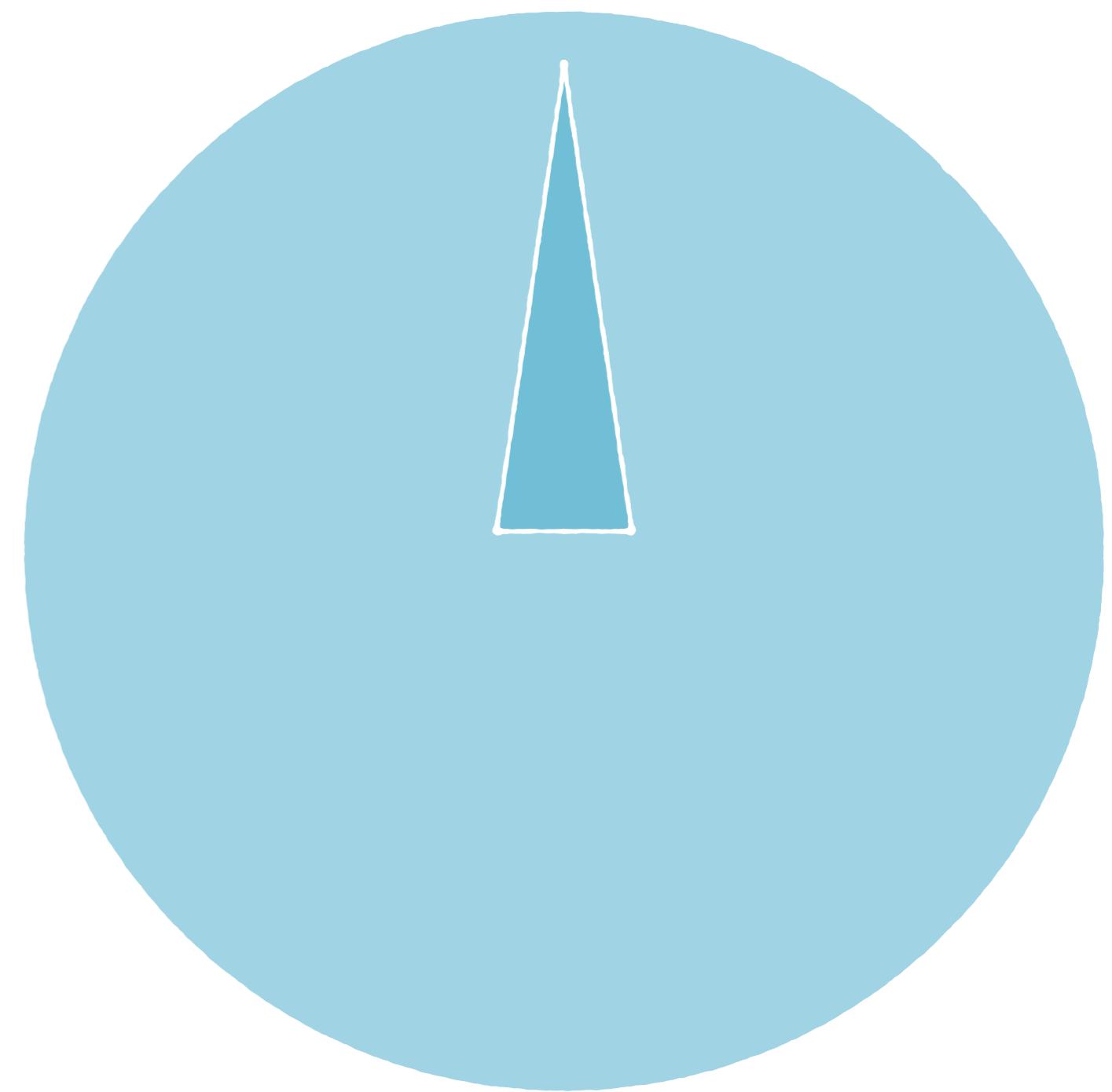
# SVG - Scalable Vector Graphics

- + SVG Graphiken sind Vektorgraphiken auf XML-Basis.
- + SVG kann beliebig in HTML5 Dokumente eingebunden werden.
- + Sie bilden eine DOM-Struktur und können mit CSS gestaltet und mit Javascript programmiert werden.

```
<svg width="400" height="400" viewBox="0 0 400 400">
  <g id="layer">
    <circle
      id="circle" class="circle"
      cx="0" cy="0" r="100"
      transform="translate(200,200) rotate(0)"
    ></circle>

    <polyline id="needle" class="needle"
      points="-10,0 10,0 0,80 -10,0 10,0"
      transform="translate(200,200) rotate(20)"
    ></polyline>

    <polyline id="diagram" class="diagram"
      points="0,0 200,0"
      transform="translate(100,90) rotate(0)"
    ></polyline>
  </g>
</svg>
```



# SVG CSS

```
/* MODULE */
.circle,
.needle,
.diagram {
    fill      : transparent;
    fill-rule : evenodd;

    stroke     : white;
    stroke-width : 5px;
    stroke-linecap : button;
    stroke-linejoin : round;

    transition-property : all;
    transition-duration : 2s;
    transition-timing-function : ease-in-out;

}

/* STATE */
.default { fill : rgba(0,200,0,1); }
.warning { fill : rgba(200,0,0,1); }
```

# Responsives Javascript

# Breakpoints mit Javascript

```
mediaTiny      = window.matchMedia('(max-width: 320px)'),  
mediaXtrasmall = window.matchMedia('(max-width: 479px)'),  
mediaSmall     = window.matchMedia('(max-width: 767px)'),  
mediaMedium    = window.matchMedia('(min-width: 768px)'),
```

# Laden bei Größenänderung des Viewports

```
mediaTiny.onchange = function () {  
    console.log('tiny changed!');  
};
```

# Inhalte beim Öffnen einer Seite laden

```
if ( mediaMedium.matches === true ) {  
    loadVideothek();  
    unloadButton();  
} else {  
    unloadVideothek();  
    loadButton();  
}
```

# Javascript und Geräteerkennung

# Browser- und Geräteinformationen abfragen

```
console.log( window.navigator.userAgent );
console.log( window.navigator.vendor );
```

# mobile-detect.js

# mobile-detect.js

```
<script src="mobile-detect.js"></script>
<script>

  var md = new MobileDetect(window.navigator.userAgent);

</script>
```

# mobile-detect.js

```
var md = new MobileDetect(  
    'Mozilla/5.0 (Linux; U; Android 4.0.3; en-in;  
SonyEricssonMT11i' +  
    ' Build/4.1.A.0.562) AppleWebKit/534.30 (KHTML,  
like Gecko)' +  
    ' Version/4.0 Mobile Safari/534.30');
```

# mobile-detect.js

```
console.log( md.mobile() );           // 'Sony'  
console.log( md.phone() );           // 'Sony'  
console.log( md.tablet() );          // null  
console.log( md.userAgent() );        // 'Safari'  
console.log( md.os() );              // 'AndroidOS'  
console.log( md.is('iPhone') );       // false  
console.log( md.is('bot') );          // false  
console.log( md.version('Webkit') );   // 534.3  
console.log( md.versionStr('Build') ); // '4.1.A.0.562'  
console.log( md.match('playstation|xbox') ); // false
```

<https://github.com/hgoebl/mobile-detect.js/>

-Heinrich Goebel

# Responsive Tabellen

# Tabellen in einen Scrollcontainer packen

```
<div class="table table-responsive">
<table>
  <thead>
    <tr>
      <th>Bibendum Sem Mollis</th>
      <th>Bibendum Sem Mollis</th>
    </tr>
  </thead>
  <tfoot>...<tfoot>
  <tbody>
    <td>Nullam quis risus eget urna mollis leo.</td>
    <td>Nullam quis risus eget urna mollis leo.</td>
  <tbody>
</table>
</div>
```

# Tabellen in einen Scrollcontainer packen

```
.table-responsive {  
    max-width: 100%;  
    overflow: auto;  
}  
.table-responsive td {  
    width : 100%;  
}
```

# Tabellen als „Listen“ darstellen

```
.table-responsive table {  
    display: block; /* ersetzt table */  
}  
  
.table-responsive thead {  
    display: none;  
}  
  
.table-responsive td {  
    display: block; /* ersetzt table-cell */  
}  
  
.table-responsive td::before {  
    content: attr(data-head); /* liest ein data Attribut aus */  
    display: block;  
    font-weight: 600;  
    margin-bottom: 0.5rem;  
}
```