

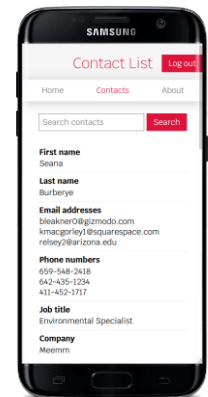
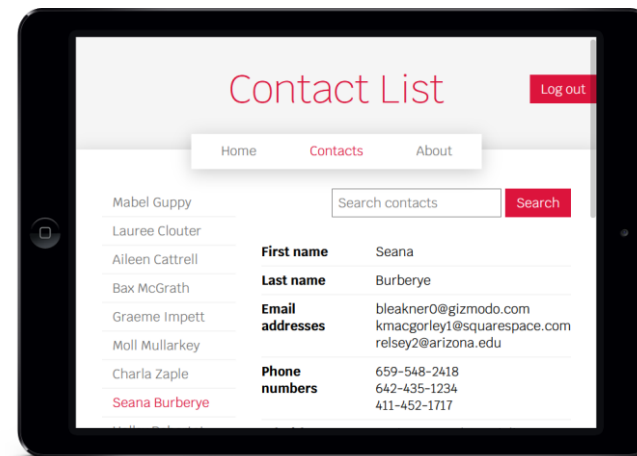
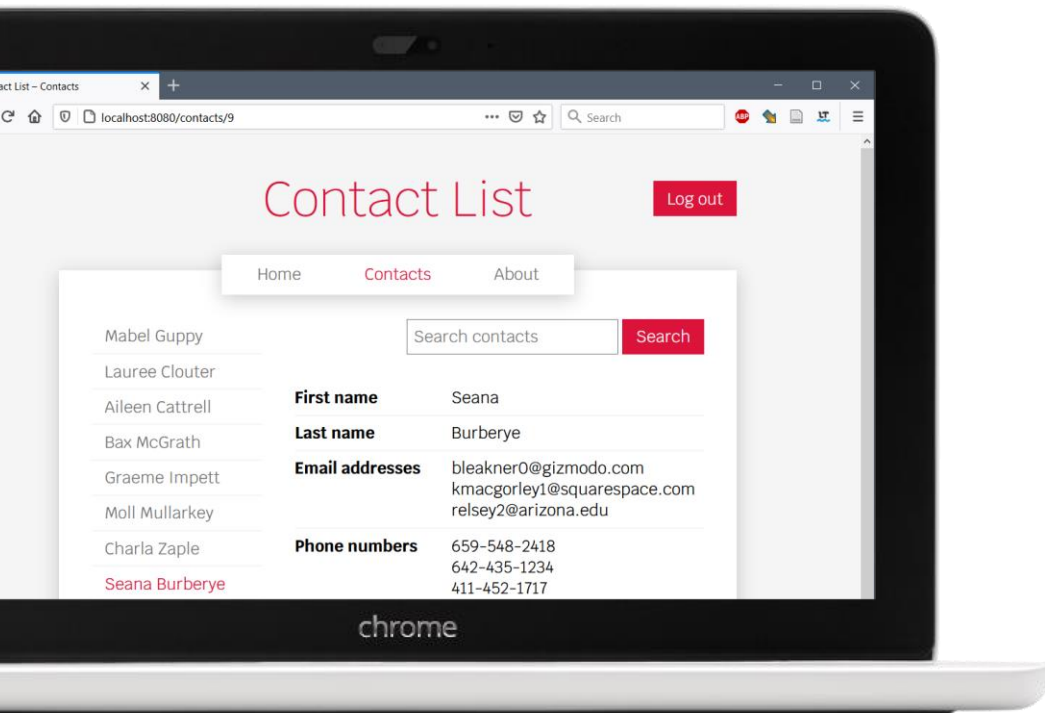
Web Engineering

Responsive Web Design

Adrian Herzog

(basierend auf der Arbeit von Michael Faes, Michael Heinrichs & Prof. Dierk König)

Responsive Design



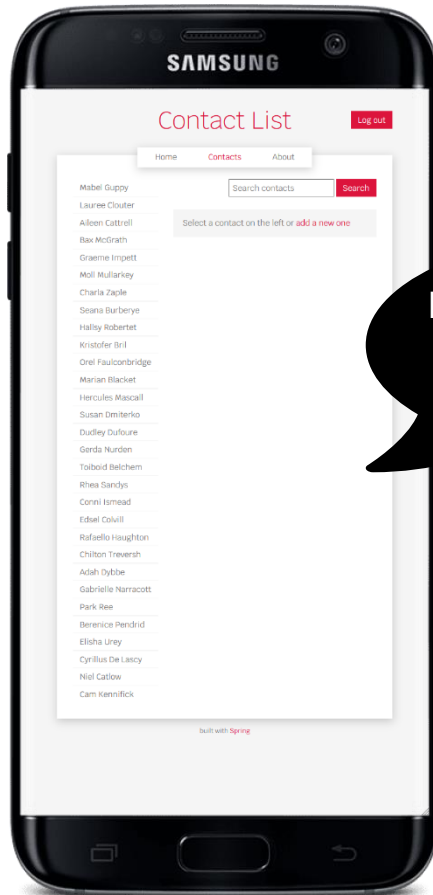
Ziel: Web-App soll auf grossen *und kleinen* Geräten gut aussehen und bedienbar sein. Seite «antwortet» auf Änderungen der Displaygrösse.

Wo steht unsere Web-App?

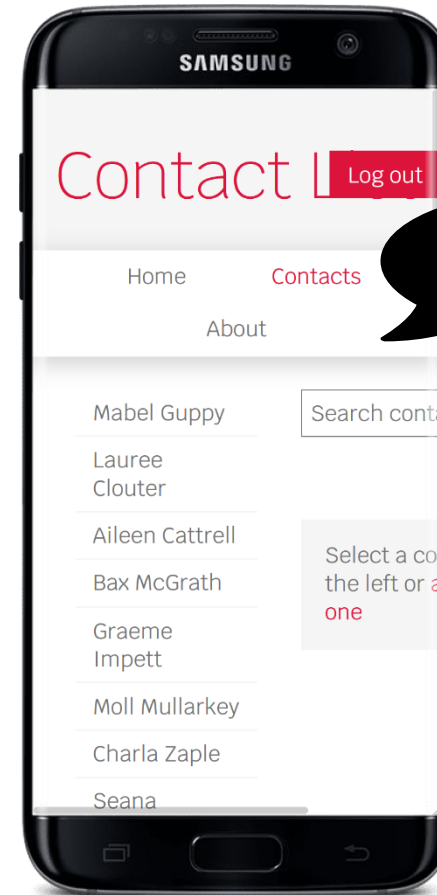
Grundlage für
Responsive Design

Komplett ohne Anpassung:

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



Browser von Phones
«lügen» über
Bildschirmgröße



Auch nicht
besser...

Lösungs-Ansätze

webeC



Flexibles Layout (CSS)

Relative Schriftgrößen, relative Element-Breiten, Grids, ...

Media-Queries (CSS)

Displaygrößen-spezifische Regeln

Libraries (CSS/JS)

z. B. *Bootstrap* getbootstrap.com

Dynamische Seitenlogik (JS)

z. B. `onresize="..."`

Geräte-spezifische Views (MVC)

z. B. github.com/blueconic/browscap-java

Flexibles Layout

Grundideen:

- Grösse von Elementen mit *relativen Einheiten* definieren
- Elemente *flexibel platzieren* (basierend auf Grösse & Platz)

Einheiten in CSS:

Einheiten	Beschreibung
cm, mm, in, ...	<i>absolute</i> Einheiten (für Druckansicht)
px	<i>absolute</i> Einheit, aber nicht immer = 1 Pixel! (abhängig von Pixeldichte des Displays)
em, ex, ch	<i>relativ</i> zur Schriftgrösse des (äusseren) Elements (<code><html></code> -Element hat Defaultgrösse 16px)
rem	<i>relativ</i> zur Schriftgrösse des <code><html></code> -Elements
vw, vh, vmin, vmax	<i>relativ</i> zur Fenster-/Gerätegrösse : 1% davon
%	<i>relativ</i> zur Breite/Höhe/Schrift des äuss. Elem.

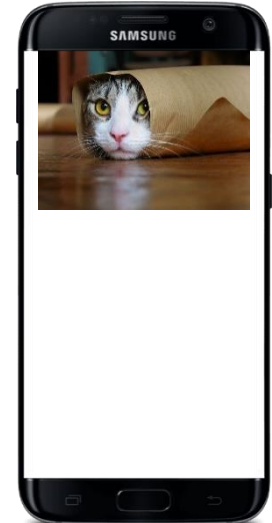
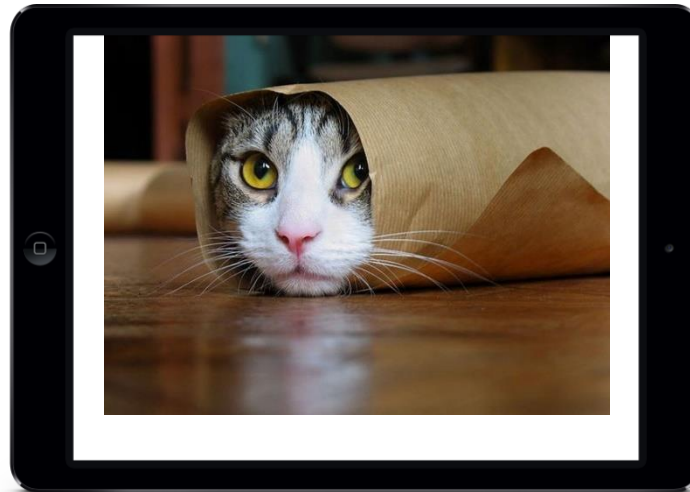
z. B. Galaxy S20:
1px = 4 Pixel!

«Viewport»

Relative Einheiten: Beispiele

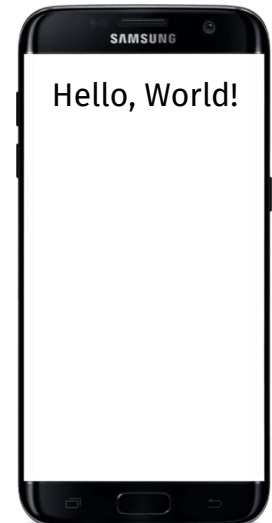
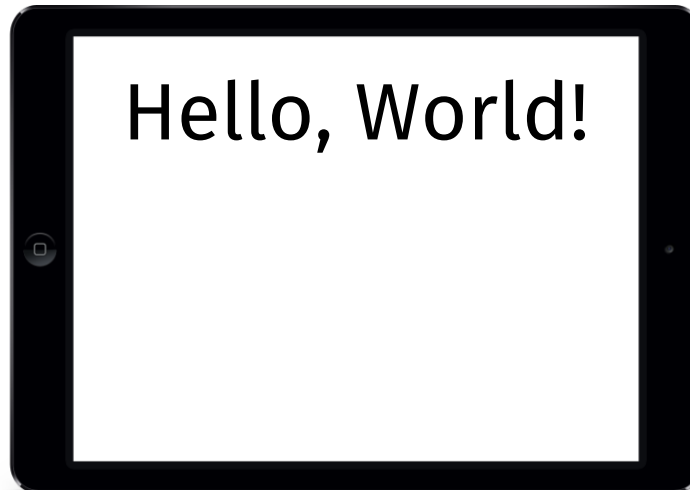
Relative Breite:

```
#cat-img {  
  width: 100%;  
}
```



Relative Schriftgröße:

```
h1 {  
  font-size: 8vw;  
}
```



Übung 1: Erste Schritte mit Responsive

Lade die neue Vorlage in deine IDE, starte die App und betrachte sie im *Responsive Design Mode* in deinem Browser. Wähle als Gerät ein Smartphone mit 360 – 400 px Breite.

1. Füge das in den Folien gezeigte `<meta>`-Element innerhalb von `<head>` zum App-Layout hinzu und beobachte die Änderung im Browser.
2. Mache einen ersten Schritt Richtung Responsive Design, indem du die Titelgrösse für `<h1>` anpasst. Definiere die Grösse relativ zum Viewport, aber nur bis zu einem bestimmten Maximum. Überprüfe im Browser, dass die Titelgrösse jetzt responsive ist.

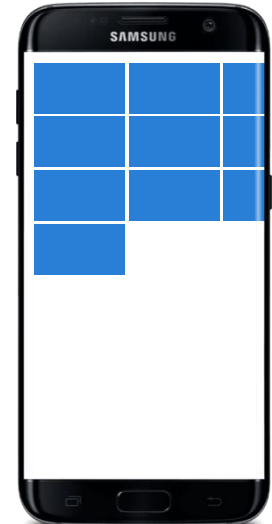
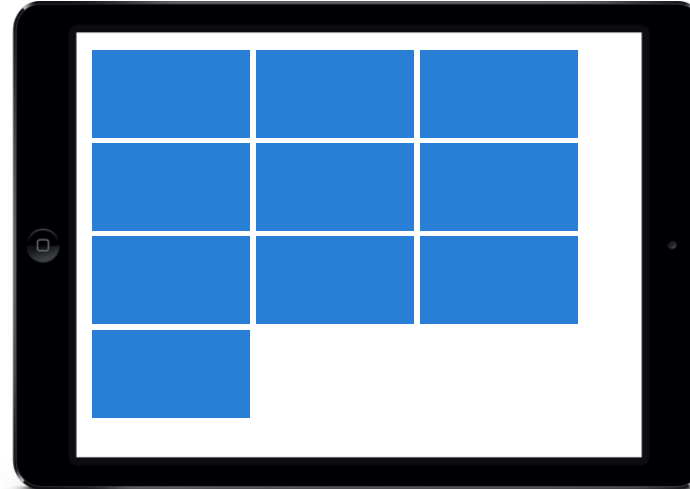
Elemente flexibel platzieren: Grid Layout

Mächtiger Layout-Mechanismus in CSS: *Grid Layout*

Unflexibel:

```
<div class="gallery">
  <div>...</div>
  <div>...</div>
  <div>...</div>
  ...
</div>
```

```
.gallery {
  display: grid;
  grid-template-columns:
    150px 150px 150px;
}
```

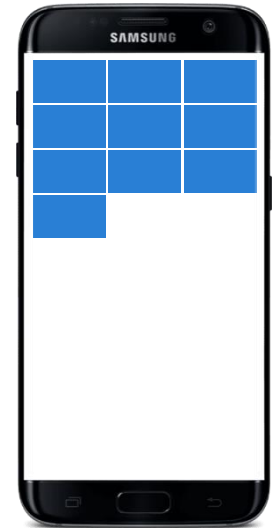
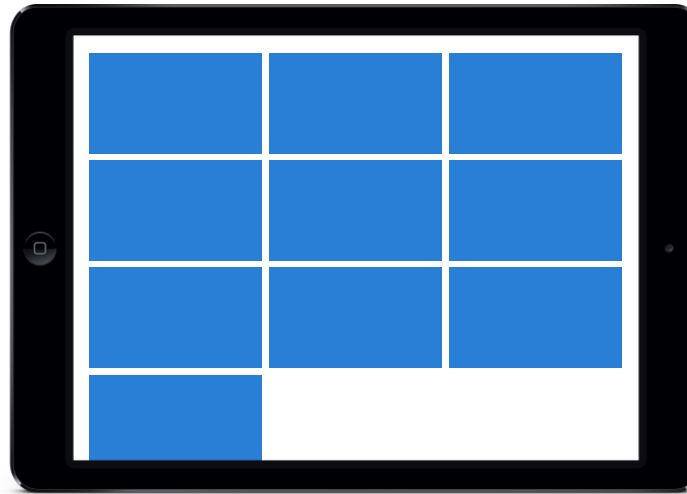


Unflexibel, aber mit *relativer Grösse*: Spezielle Einheit **fr** (fraction)

```
.gallery {  
  display: grid;  
  grid-template-columns:  
    1fr 1fr 1fr;  
}
```

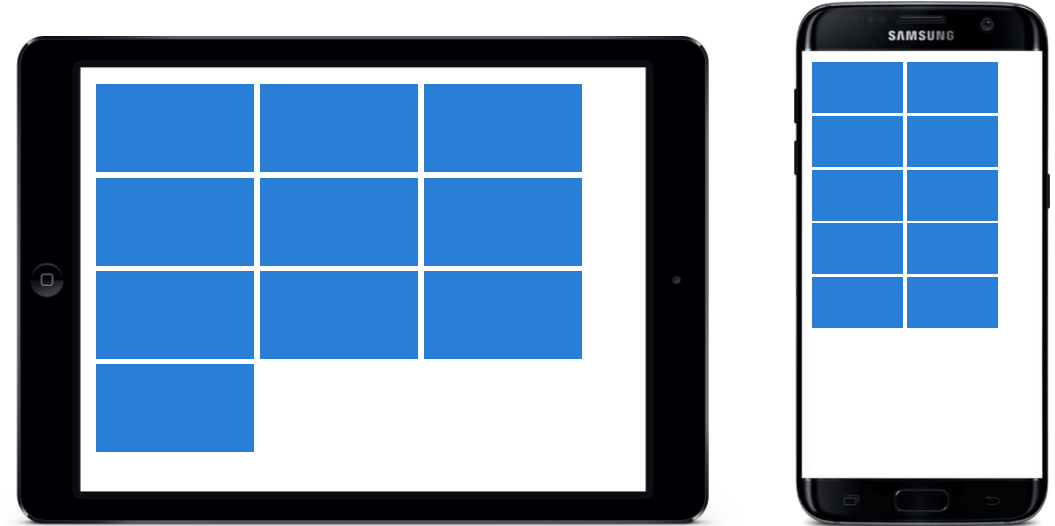
oder (gleichwertig):

```
.gallery {  
  display: grid;  
  grid-template-columns:  
    repeat(3, 1fr);  
}
```



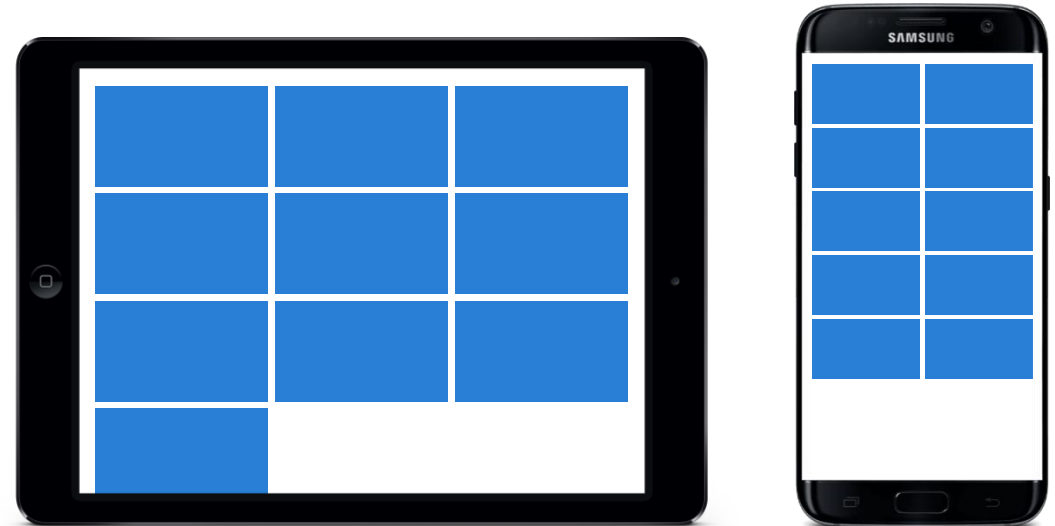
Flexibel: auto-fit

```
display: grid;  
grid-template-columns:  
  repeat(auto-fit, 150px);
```



Flexibel & relative Grösse:

```
display: grid;  
grid-template-columns:  
  repeat(auto-fit,  
    minmax(150px, 1fr));
```



Media Queries

Noch mächtiger: Media Queries. Erlauben es, beliebige CSS-Regeln *unter bestimmten Bedingungen* anzuwenden

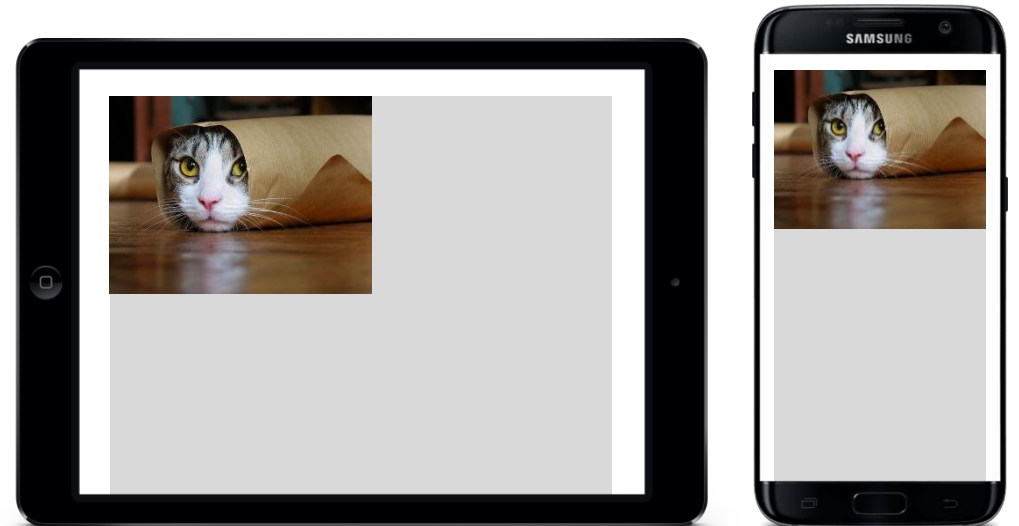
```
@media Medientyp screen and (Medien-Feature min-width: 480px) {  
  body {  
    margin: 30px;  
  }  
}
```

Medien-Feature-Ausdruck

Media Queries: Beispiel

Unterschiedliche Regeln, je nach Viewport-Grösse:

```
#cat-img {  
  width: 100%;  
}  
  
@media (min-width: 640px) {  
  #cat-img {  
    width: 50%;  
  }  
}
```



Media-Queries & Priorität/Spezifität

Rückblick:

Priorität	Merkmal	Beschreibung
1	Wichtigkeit	Mit !important kann alles überschrieben werden <i>Sollte aber möglichst nie bzw. höchstens als Notlösung verwendet werden!</i>
2	Inline	Inline-Regeln überschreiben andere Regeln
3	Selektoren-Spezifität	Verschiedene Arten von Selektoren haben unterschiedliche Priorität
4	Reihenfolge	Spätere Sheets / Regeln überschreiben frühere
5	Vererbung	Gewisse undefinierte Eigenschaften werden von Eltern-Elementen geerbt

Media Queries haben keinen Einfluss auf Priorität!

Medientypen und -Features

Medientyp	Beschreibung
<code>screen</code>	Anzeige auf einem Bildschirm
<code>print</code>	gedruckte Medien (und Druckvorschau)
<code>speech</code>	Sprach-Synthesizer
<code>all</code>	passend für alle Geräte

Medien-Feature	Beschreibung
<code>width</code>	Breite des Viewports (Fenster/Gerät), z.B. in <code>px</code>
<code>height</code>	Höhe des Viewports
<code>resolution</code>	Pixeldichte des Anzeigegeräts, in <code>dpi</code>
<code>aspect-ratio</code>	Seitenverhältnis des Viewports, Zahl oder Verhältnis
<code>...</code>	

alle mit min-, max-
oder ohne

min-width oder max-width?

`min-width`: Default-Regeln für kleine Geräte, Spezialregeln für grosse

`max-width`: Default-Regeln für grosse Geräte, Spezialregeln für kleine

Theoretisch gleichwertig, aber...

Zuerst für Desktop-Geräte designen und für mobile Geräte anpassen, oder umgekehrt?

Kommt auf Web-App an, aber häufig:

Mobile First

[Internet](#) > [Mobiles Internet & Apps](#)

PREMIUM +

Anteil der mobilen Internetnutzer in Deutschland in den Jahren 2015 bis 2020

Anteil der mobilen Internetnutzer in Deutschland bis 2020

Veröffentlicht von [Statista Research Department](#), 02.03.2021



Durch die zunehmende Verbreitung von mobilen Endgeräten und niedrige Mobilfunkpreise ist der Anteil mobiler Internetnutzer in Deutschland in den vergangenen Jahren stetig gestiegen und belief sich im Jahr 2020 auf 80 Prozent. Im Jahr 2015 lag der Anteil der mobilen Internetnutzer noch bei 54 Prozent. Laut einer [Umfrage zur Internetnutzung nach Endgeräten](#) wurde im Jahr 2018 zudem erstmals häufiger das Smartphone als ein PC/Laptop zum Surfen im

Print Layout

Beim Druck-Layout hilft es oft, gewisse Elemente ausblenden (z.B. Menus und Gestaltungselemente).

Dafür kann z.B. eine Hilfsklasse definiert werden, die dann auf den Elementen gesetzt wird, welche nicht gedruckt werden sollen.

```
@media print {  
    .hide-print {  
        display: none;  
    }  
}
```

```
<div class="hide-print">  
      
</div>
```


Übung 2: Einfache Media Query

Der Text der App ist allgemein zu gross für kleine Geräte.

Implementiere mittels einer Media Query folgende Regeln:

- Für «kleine» Geräte wird als Grundschriftgrösse 100% verwendet.
- Für «grosse» Geräte wird hingegen 120% verwendet (wie bisher).

Entscheide, ab welcher Breite ein Gerät als «gross» gilt und wähle zwischen einer `min-width`- und einer `max-width`-Regel.

Fragen?

