

RESPONSIVE DESIGN

2

MMWP2024 - LV04

INHALTSVERZEICHNIS

- Organisation
- Media Rules
- Fluide Medien
- Bilder- und Video-Design

INHALTSSCHWERPUNKTE

- Gerätearten und Darstellung
- Adaptive Web Design
- Responsive Web Design
- Layout-Muster
- Seitennavigation

VORAUSSETZUNG

Der Ausgangspunkt dieser Vorlesungsreihe ist das Wissen über HTML-CSS-Layouting

- Fixed-Box layout
- Flex-Box layout
- Grid-Layout
- Responsive CSS-Regeln

ZIELE

- Vorstellung aktueller Media-Rules
- Erklärung von fluiden Medien
- Nennung von Text-, Bilder- und Video-Design
Möglichkeiten in HTML5 und CSS3

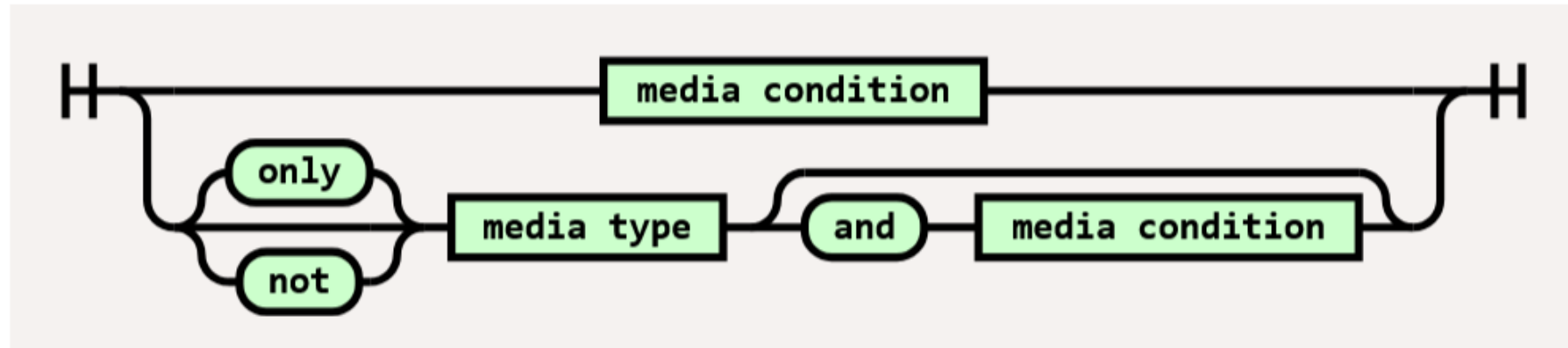
MEDIA QUERIES

- Media Types und Queries sind Elemente von CSS2, die in CSS3 ausgebaut und verändert worden sind
- Für Responsive Design ist die Größe des Zielbildschirms der aktuellen Anzeige entscheidend, die nach ihrer Breite in CSS-px abgefragt wird
- Anhand der Tabellen der verbreitetsten Bildschirmgrößen und anhand des eigenen Designentwurfs werden Änderungspunkte im Design festgelegt
- Danach erfolgen die unterschiedlichen CSS-Angaben
- Neuste Funktionen kamen im Mai 2024, [siehe W3C](#)

MEDIA QUERIES - DEFINITION

- @media Regeln werden in Media Queries verwendet, um verschiedene Styles für verschiedene Medien anzubieten
- Mit Media Queries können verschiedene Eigenschaften von Medien abgefragt und geprüft werden
 - Breite und Höhe des Bildschirms (Viewport)
 - Breite und Höhe des Geräts (Device)
 - Orientierung des Bildschirms (Landscape oder Portrait)
 - Die Auflösung des Bildschirms


LOGIK VON MEDIA QUERIES



Quelle

CSS-SYNTAX VON MEDIA QUERIES

```
@media not|only mediatype and (mediafeature and|or|not mediafeature) {  
    CSS-Code;  
}
```



Expression/Mediakondition

Quelle

- not – ist eine logische Verneinung (das Gegenteil gilt!)
- only – ist ein Schlüsselwort als Hinweis für ältere Browser das Query anzuwenden
- and - kombiniert die Medieneigenschaften (mediafeatures) mit dem konkreten Typ Medium oder miteinander
- or - ist ein logisches oder

VERWENDUNG VON MEDIA QUERIES

```
<link rel="stylesheet" media="screen and (min-width: 900px)" href="widescreen.css">  
<link rel="stylesheet" media="screen and (max-width: 600px)" href="smallscreen.css">  
....
```

https://www.w3schools.com/cssref/css3_pr_mediaquery.asp

```
1 | @import url;  
2 | @import url list-of-media-queries;
```

Beispiele:

```
1 | @import url("fineprint.css") print;  
2 | @import url("bluish.css") projection, tv;  
3 | @import 'custom.css';  
4 | @import url("chrome://communicator/skin/");  
5 | @import "common.css" screen, projection;  
6 | @import url('landscape.css') screen and (orientation:landscape);
```

<https://developer.mozilla.org/de/docs/Web/CSS/@import>

Quelle

MÖGLICHE WERTE FÜR MEDIA QUERIES

- width, device-width
- height, device-height
- orientation, aspect-ratio, device-aspect-ratio, resolution
- (seltener) color, color-index, monochrome
- scan (TV), Grid
- Erklärung und Beispiele

MEDIEN TYPEN

Browser Support

The numbers in the table specifies the first browser version that fully supports the @media rule.

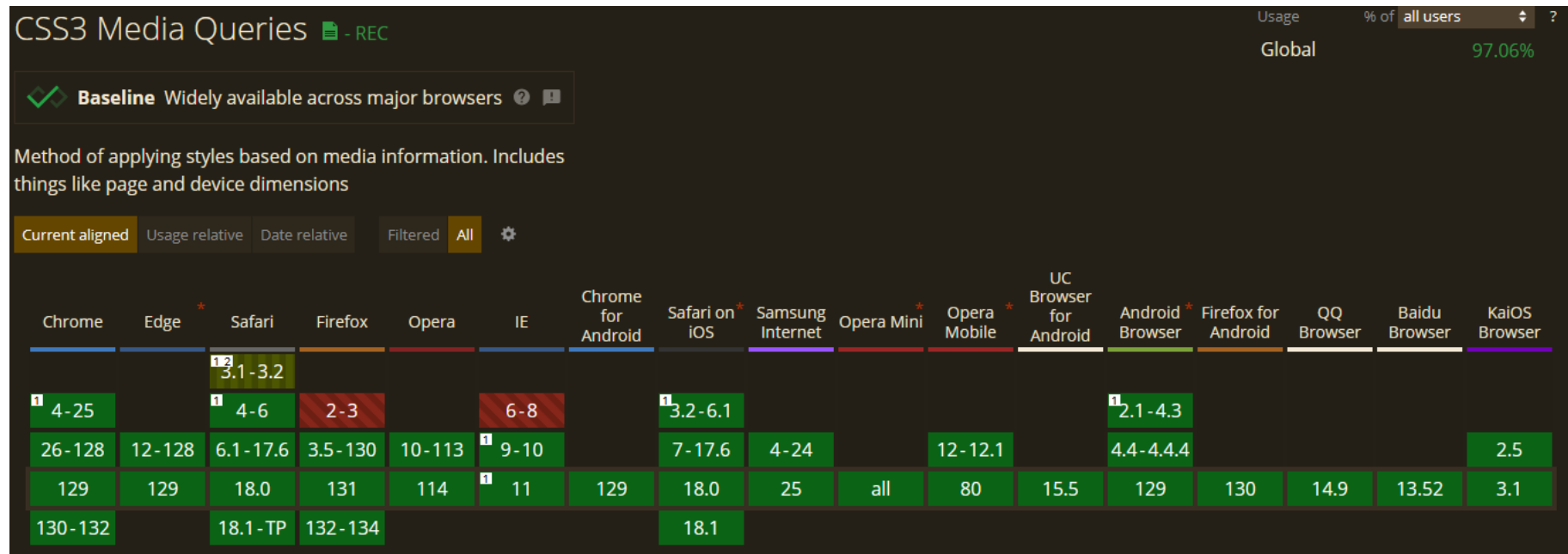
Property					
@media	21	9	3.5	4.0	9

Media Types

Value	Description
all	Default. Used for all media type devices
print	Used for printers
screen	Used for computer screens, tablets, smart-phones etc.
speech	Used for screenreaders that "reads" the page out loud


Quelle

BROWSER SUPPORT - ALLGEMEIN




Quelle

BROWSER SUPPORT - FEATURES WIE AUFLÖSUNG


Media Queries: resolution feature  - CR















Allows a media query to be set based on the device pixels used per CSS unit. While the standard uses `min / max-resolution` for this, some browsers support the older non-standard `device-pixel-ratio` media query.

Usage % of all users  ?

Global 93.62% + 3.45% = 97.06%

unprefixed: 93.62% + 1.11% = 94.72%

Current aligned Usage relative Date relative Filtered All 

Chrome	Edge *	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	Opera Mobile *	UC Browser for Android	Android Browser *	Firefox for Android	QQ Browser	Baidu Browser	KaiOS Browser
			2-3	10-11.5 												
 4-28 		3.1-3.2	 3.5-15	12.1			3.2					2.1-2.2				
 29-67	 12-18	 4-15.6 	 16-61	 15-54	6-8		 4-15.8 	 4-9.2		 12 		 2.3-4.3 				
68-128	79-128	16.0-17.6	62-130	55-113	 9-10		16.0-17.6	10.1-24		12.1		 4.4-4.4.4				2.5
129	129	18.0	131	114	 11	129	18.0	25	 all	80	15.5	129	130	14.9	13.52	3.1
130-132		18.1-TP	132-134				18.1									

Quelle

MEDIA QUERIES LEVEL 4

- Originale Spezifikation: [Quelle](#)
- Grundlegende Referenz: [Quelle](#)
- Hilfreiche Erläuterungen von Jordan Moore zu:
script, hover, pointer - [Quelle](#)

MEDIA QUERIES LEVEL 4

Um für Smartphones, Tablet-PCs und Desktop-PCs spezifisch CSS zu definieren, werden nach dem allgemeinen Teil der CSS-Datei weitere CSS-Abschnitte definiert:

```
@media only screen and (max-width: 480 px) { ...}  
@media only screen and (min-width: 480 px) and (max-  
width: 960 px) { ... }  
@media onlyscreen and (min-width: 960 px) { ... }
```

Das Design für übergroße Bildschirme muss ggf. im letzten Eintrag abgefangen werden, z.B. durch Zentrieren der Webseite im Bild

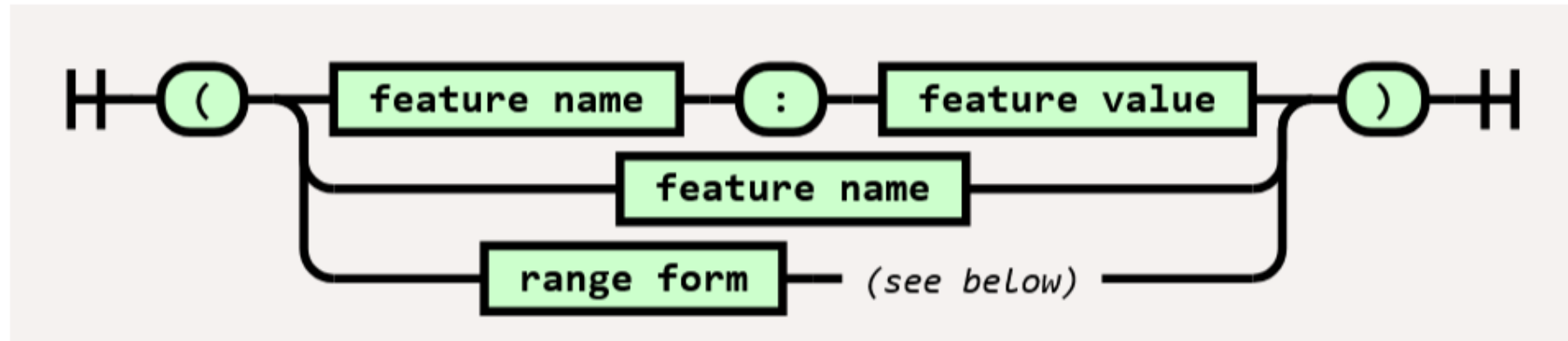
DAS SCHLÜSSELWORT ONLY

- Das Wort “only” sorgt also dafür, dass ältere Webbrowser, die den Media Type screen nicht kennen, die Anweisung ignorieren werden
- Standard: [Quelle](#)

GEPLANTE MEDIA QUERIES FEATURES - MEDIA QUERIES LEVEL 5

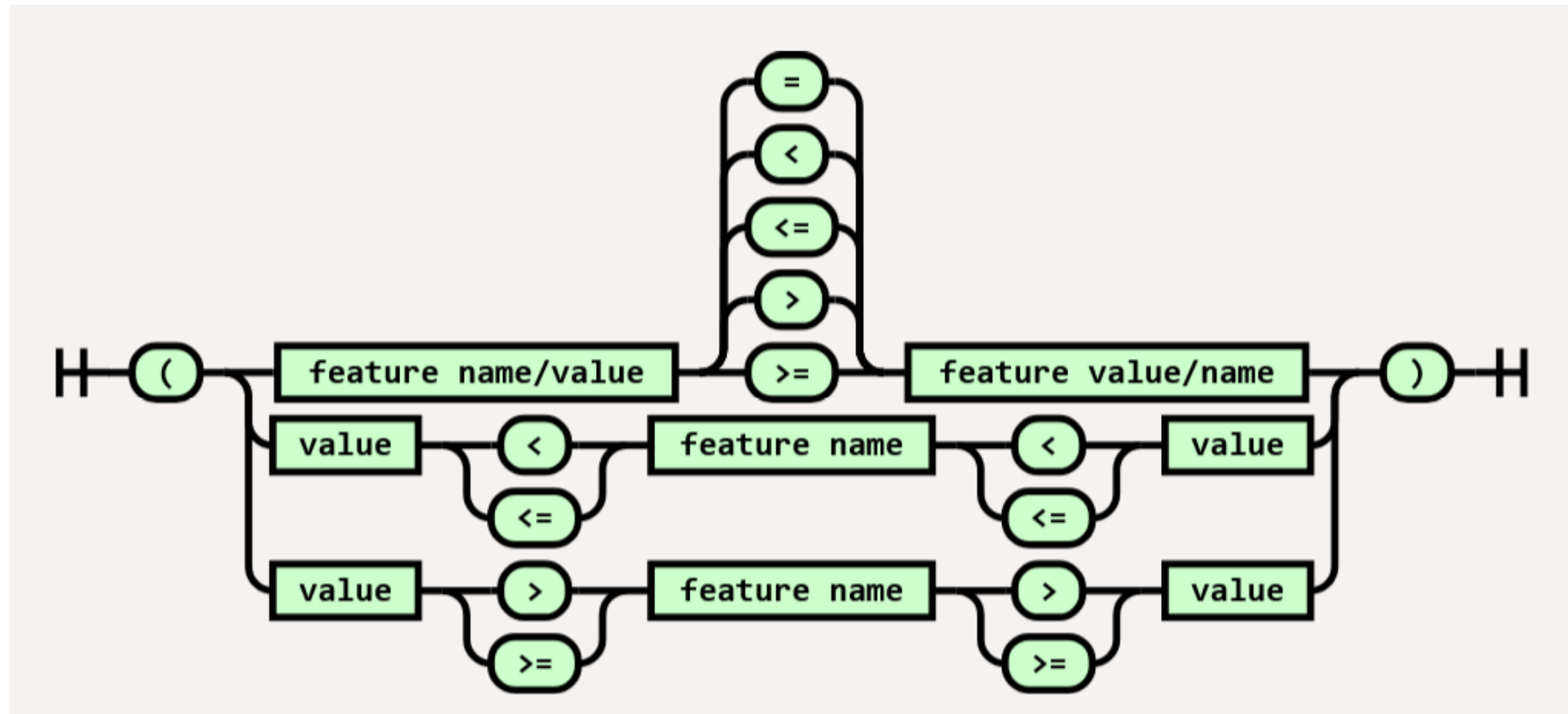
- Seit 2021 ist ein neues Level 5 von Media queries in planung und wird seit dem vom W3C erarbeitet
- Neuerungen sind z.B.
 - Combining Media Queries
 - tty, tv, projection, handheld, braille, embossed, aural, speech
 - Ranges, prefers-reduced-data, prefers-light, prefers-dark

MEDIA QUERIES LEVEL 5 - FEATURES



Quelle

MEDIA QUERIES LEVEL 5 - FEATURES AND RANGES

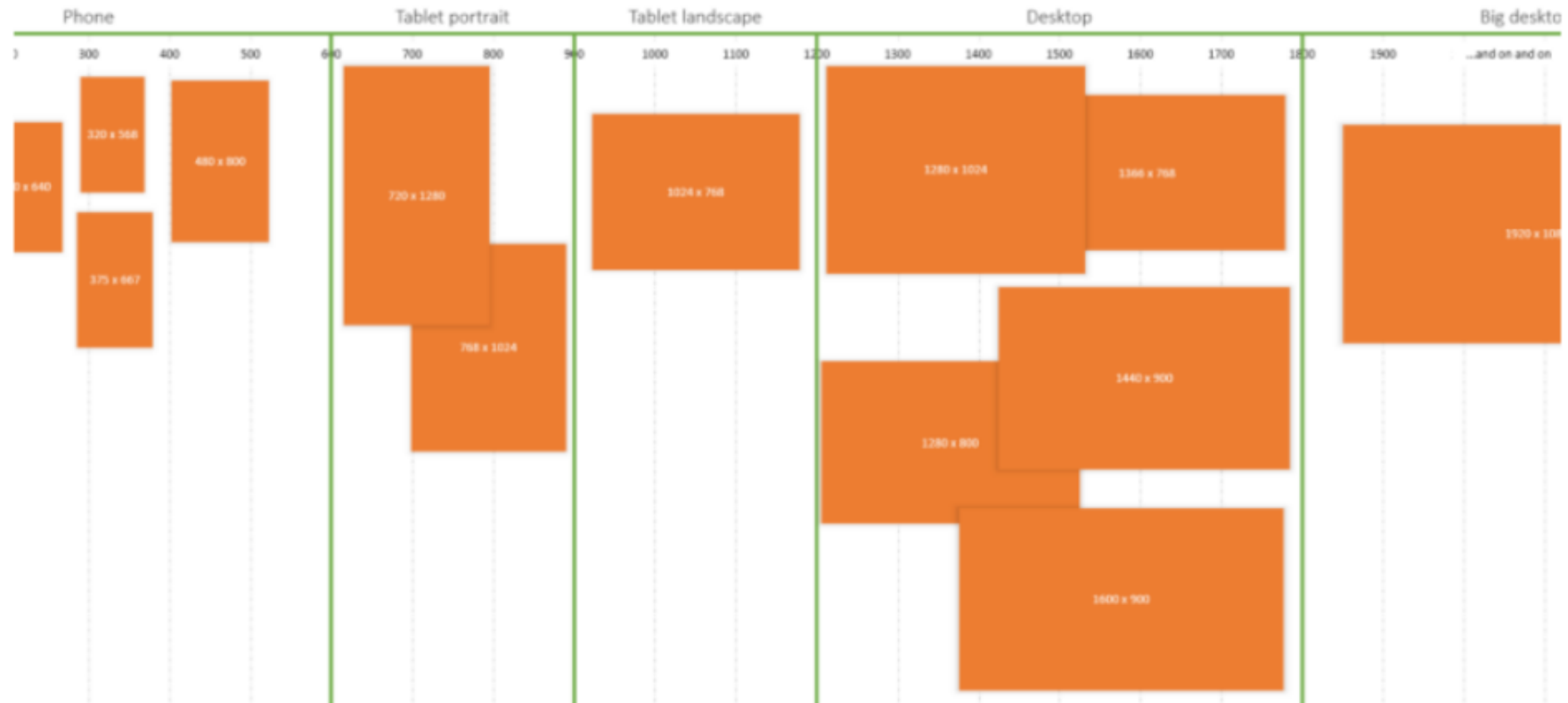


Quelle

HILFREICHE LINKS ZU MEDIA QUERIES

- Media queries: [Quelle](#)
- Ein Diskussionsbeitrag, warum device-orientierte Media Queries fehlerbehaftet sein können: [Quelle](#) (Justin Avery, 2014)

MEDIA QUERIES - GEEIGNETE BREAKPOINTS



“For complex sites, life is much easier if you pick a handful of breakpoints to use across the site”

An excerpt from The 100% correct way to do CSS breakpoints

Quelle

ZUSÄTZLICH: ZOOM-WERT BEI SMARTPHONES - 1

- Viewport Einstellungen kennen wir bereits
- Ohne die Anweisung `width=device-width` wird die Webseite für Smartphones erst gerendert und dann auf die Breite (Android: 800px, iOS: 980px) des Viewports herunterskaliert
- Im Ergebnis werden Umbruchspunkte (break points) zur Steuerung durch die Media Queries gar nicht erst angesprochen und bleiben unwirksam
- Nur bei `initial-scale=1` wird die Webseite in beiden Modi des Smartphones gleich angezeigt

ZUSÄTZLICH: ZOOM-WERT BEI SMARTPHONES - 2

- Wenn man maximum-scale nicht verwendet und Zoomen verhindern will, hilft folgendes: user-scalable
- Mit diesem Attribut kann man definieren, ob der Nutzer auf der Seite zoomen kann (=yes) (default) oder nicht (=no)
- Manche Autoren fangen Smartphones beim Wechsel der Ansichten portrait und landscape mit Media Queries ab
- Z.B. [Quelle](#)

SHRINK-TO-FIT=NO

- shrink-to-fit=no ist eine sinnvolle Ergänzung bei der viewport-Angabe, gilt aber nur für Safari, und zwar für iOS ab V. 9 bei Safari im Split oder Side-View-Modus
- Dieser Modus erlaubt es, mehrere Apps gleichzeitig nutzen zu können oder eben zwei Fenster gleichzeitig zu öffnen
- Ohne die Angabe wird alles (vgl. Elemente fixer Breite) so verkleinert, dass es auf den geringeren verfügbaren Platz passt
- Wenn wir beispielsweise ein 900px breites Element haben, so wird die Webseite gesamt zu verkleinert, dass dieses Element ganz zu sehen ist
- Mit shrink-to-fit=no findet diese Verkleinerung hingegen nicht statt, der Benutzer muss bei einem 900px breiten Element beispielsweise horizontal scrollen

FLEXIBLE BREITEN - 1

- Um in Browserbreiten zwischen den Punkten des radikalen Designwechsels flexible Boxenbreiten realisieren zu können, sind Prozentangaben zu den Breiten und zu den Margins (bei flexbox-Einsatz) das Mittel der Wahl
- Ein einfaches dreispaltiges Layout für Desktops könnte man z.B. wie folgt anlegen, wobei die Summe der Breiten je Design nachzurechnen wäre (max. 100%, besser 99%)

FLEXIBLE BREITEN - 2

```
1 .flexible-main-parent-container {  
2   width: 100%, ... }  
3 .left-sidebar { width: 17%; max-width: 17%;  
4   min-width: 17%; margin-left: 1.5%; ... }  
5 .right-sidebar {width: 18%; max-width: 18%;  
6   min-width: 18%; margin-right: 1.5%; ... }  
7  
8 @media screen and (min-width: 1280){  
9   ....flexible-main-parent-container {  
10    min-width: 56%; margin: 0 2% 0 2%; ... }  
11  ... }
```

FLUID DESIGN MEDIEN

- Beachtung von Bildschirmgröße, Bildschirmauflösung und Betrachtungsabstand bei Texten, Bildern und Videos
- Diese Maßnahmen entstanden unabhängig von den Überlegungen zum Grunddesign von Webseiten
- Ein Stichwort dazu ist Fluid Design

FLUID DESIGN TEXT - 1

- Es sollen Schriften der Größe relativ angepasst werden
- Browser haben meist 16px Standard-schriftgröße, die man fast immer so belässt

```
body { font: normal 100% sans-serif; }  
body { font-size: 100%; }
```

FLUID DESIGN TEXT - 2

- Eine gute Website zur Adaption von Fonts ist:
WebDesignerDepot.com
- Css-Tricks bietet auch eine gute Anleitung für fluide
Typography: [Quelle](#)

FLUID DESIGN TEXT - BEISPIELE

- Will man die Standardschriftgröße z.B. auf 15px bringen, muss man erst rechnen, dann zuweisen
- $(\text{Zielgröße} / \text{Kontextgröße}) \cdot 100 = (15 / 16) \cdot 100 = 93,75\%$

```
body { font-size: 93.75%; }
```

- Weitere Elemente errechnen ihre Schriftgröße analog, wobei immer die Standardschriftgröße des unmittelbaren Elterncontainers zählt:

```
h1 { font-size: 1.466666667 em; }
```

- bei 22px Ziel- und 15px Kontextgröße

FLUID DESIGN TEXT - PROBLEME

- Das Verfahren ist mühevoll, da man in der Hierarchie der Elemente in HTML5 immer erst einmal die Größe des Textes in px festlegen
- Relativ zur Textgröße muss dann in Prozent im Elternelement ermittelt werden
- Hilfe bieten Frameworks wie Bootstrap, oder auf die Textgröße im Element `<html>` oder `<body>` direkt bezug nehmen
- CSS-Clamp ist auch eine Möglichkeit für fluiden Text: [Quelle](#)
- Es gibt auch Bibliotheken, welche Javascript zur Berechnung nehmen: [Quelle](#)

FLUID DESIGN TEXT - PIXEL UND EM

px font-size	em equivalent	* Rounded to 3dp	1px in ems	Notes
11	0.689	*	0.091	
12	0.750		0.083	
13	0.814	*	0.077	
14	0.875		0.071	
15	0.938	*	0.067	
16	1.000		0.063	Browser standard default
17	1.064	*	0.059	
18	1.125		0.056	
19	1.188	*	0.053	
20	1.250		0.050	
21	1.313	*	0.048	
22	1.375		0.046	
23	1.438	*	0.044	
24	1.500		0.042	
25	1.563	*	0.040	
26	1.625		0.039	
27	1.688	*	0.037	
28	1.750		0.036	
29	1.813	*	0.035	
30	1.875		0.033	

Quelle

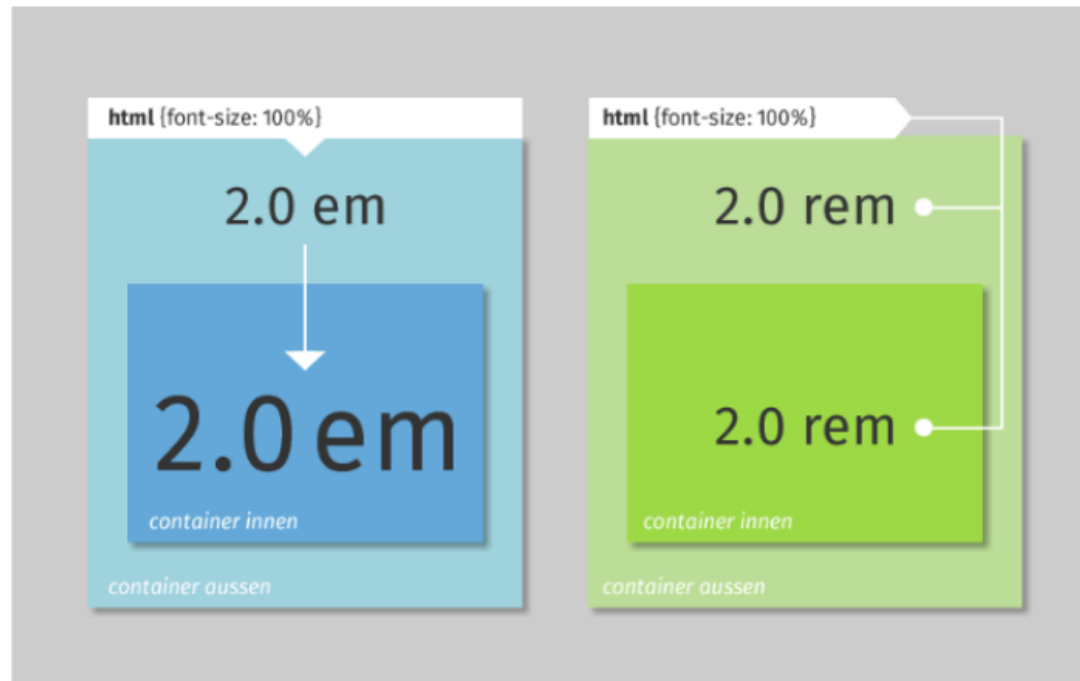
FLUID DESIGN TEXT - EM

- Schriftgrößen können in em mit Bezug zur Schriftgröße 1em im direkten Elternelement oder in rem mit Bezug zur Schriftgröße im <html>- oder <body>-Element angegeben werden
- Dazu belässt man die Zuweisung

```
font-size: 100%; /* 16px */
```

- im entsprechenden Bezugselement zu Beginn. Evtl. ist eine explizite Fontangabe zuvor erforderlich, damit der CSS-Interpreter das realisiert

FLUID DESIGN TEXT - REM UND EM BEDEUTUNG

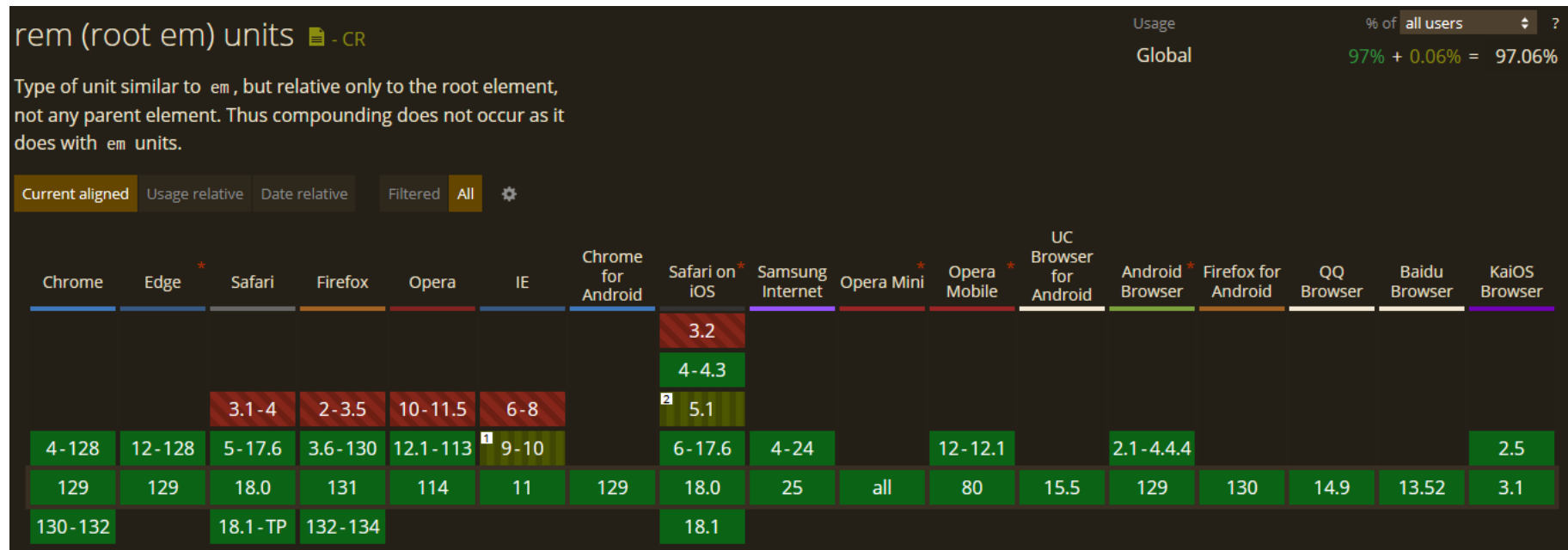


```
html{font-size:100%;}
```

r (wie root) + **em** = **rem**.


Quelle

FLUID DESIGN TEXT - EM BROWSERSUPPORT





Quelle

FLUID DESIGN TEXT - REM BROWSERSUPPORT

HTML element: em 

Usage % of all users Global 95.45%

Current aligned Usage relative Date relative Filtered All 

Chrome	Edge *	Safari	Firefox	Opera	IE  *	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	Opera Mobile *	UC Browser for Android	Android Browser *	Firefox for Android	QQ Browser	Baidu Browser	KaiOS Browser
		3.1-3.2		10-12.1								2.1-4.3				
4-128	12-128	4-17.6	2-130	15-113	6-10		3.2-17.6	4-24		12-12.1		4.4-4.4.4				2.5
129	129	18.0	131	114	11	129	18.0	25	all	80	15.5	129	130	14.9	13.52	3.1
130-132		18.1-TP	132-134				18.1									

Quelle

FLUID DESIGN TEXT - SUPPORT FÜR ALLE GERÄTE

- Um alle Browser zu bedienen, kann man vor der Angabe in rem eine feste Schriftgröße zusätzlich angeben
- Der Abstand vom Bildschirm macht unterschiedliche Schriftgrößen empfehlenswert, was unter dem Stichwort „Typografische Tonleiter“ nachgesehen werden kann
- Für das Webdesign ist es nun ausreichend, die empfohlenen Schriftgrößen je Augenabstand vom Gerät zu beachten
- Daraus ergeben sich dann Umrechnungsfaktoren je nach Bereich der Media-Query-Angaben
- Hier zum Nachlesen: [Quelle](#)


FLUID DESIGN TEXT - COMMON MISCONCEPTIONS ABOUT TOUCH

Table 1—Minimum sizes for visual targets on various device sizes			
Target	2.5-inch Phone	3.5–5-inch Phone	9–10-inch Tablet
Text	4 pt / 1.4 mm	6 pt / 2.1 mm	8 pt / 2.8 mm
Icons	6 pt / 2.1 mm	8 pt / 2.8 mm	10 pt / 3.5 mm

➤ Touch targets:

- **Minimum**—17 pt / 6 mm
- **Preferred**—23 pt / 8 mm
- **Maximum**—43 pt / 15 mm

➤ Spacing between targets to avoid interference errors, on center:

- **Minimum**—23 pt / 8 mm
- **Preferred**—28 pt / 10 mm 

Quelle

FLUID DESIGN TEXT - BERECHNUNG

Select your body font size	Voilà! Your conversions				Oh la la! Custom conversion			
<small>Choose a body font size. You'll receive the result in rem.</small>	<small>Conversions based on your body font size.</small>	<small>Conversions based on your body font size.</small>	<small>Conversions based on your body font size.</small>	<small>Conversions based on your body font size.</small>	<small>How to use this tool: Enter your custom font size.</small>	<small>How to use this tool: Enter your custom font size.</small>	<small>How to use this tool: Enter your custom font size.</small>	<small>How to use this tool: Enter your custom font size.</small>
Pixels	EM	Percent	Points	Pixels	EM	Percent	Points	1. Enter a base pixel size
6px	0.375em	37.5%	5pt	6px	0.375em	37.5%	5pt	<input type="text" value="16"/> px
7px	0.438em	43.8%	5pt	7px	0.438em	43.8%	5pt	<input type="button" value="Convert"/>
8px	0.500em	50.0%	6pt	8px	0.500em	50.0%	6pt	<input type="button" value="PX to EM"/> <input type="button" value="EM to PX"/>
9px	0.563em	56.3%	7pt	9px	0.563em	56.3%	7pt	<input type="text" value=""/> px or <input type="text" value=""/> em
10px	0.625em	62.5%	8pt	10px	0.625em	62.5%	8pt	<input type="button" value="Convert"/>
11px	0.688em	68.8%	8pt	11px	0.688em	68.8%	8pt	<input type="button" value="3. Result"/>
12px	0.750em	75.0%	9pt	12px	0.750em	75.0%	9pt	
13px	0.813em	81.3%	10pt	13px	0.813em	81.3%	10pt	
14px	0.875em	87.5%	11pt	14px	0.875em	87.5%	11pt	
15px	0.938em	93.8%	11pt	15px	0.938em	93.8%	11pt	
16px	1.000em	100.0%	12pt	16px	1.000em	100.0%	12pt	
17px	1.063em	106.3%	13pt	17px	1.063em	106.3%	13pt	
18px	1.125em	112.5%	14pt	18px	1.125em	112.5%	14pt	
19px	1.188em	118.8%	14pt	19px	1.188em	118.8%	14pt	
20px	1.250em	125.0%	15pt	20px	1.250em	125.0%	15pt	
21px	1.313em	131.3%	16pt	21px	1.313em	131.3%	16pt	
22px	1.375em	137.5%	17pt	22px	1.375em	137.5%	17pt	
23px	1.438em	143.8%	17pt	23px	1.438em	143.8%	17pt	
24px	1.500em	150.0%	18pt	24px	1.500em	150.0%	18pt	

Die Berechnung erfolgt nach der Formel:
 (neue Schriftgröße in px) / (Hauptschriftgröße in px) = Ergebnis
 in rem

Man sollte immer bis 4 Nachkommastellen listen.

Mit der Angabe von `letter-spacing: 0.07rem;` oder
 ähnlich kann man die Laufweite der Schrift steuern.

Auch die Zeilenhöhe kann gesteuert werden:

`line-height: 1.4; /*der schrifthoehe!*`

Quelle

FLUID DESIGN TEXT - SUPPORT FÜR ALLE GERÄTE

- HTML5 mit CSS3 bietet erstmalig die Möglichkeit, Fonts über Typekit ID oder aus Google Web Fonts im Kontext des Aufrufs der Webseite mit auszuliefern
- Eigene Schriftarten können mit Tools - z.B. [Quelle](#) erzeugt werden
- Es wird empfohlen, Fontsdateien immer lokal zu lagern, da sich URLs ändern können
- Google-Fonts und ähnliche Dienste sollten **IMMER** lokal benutzt werden, da der automatische Einsatz (online) meist gegen die DSGVO verstößt
- Zum Nachlesen von aktuellen Gesetzen und Rechtsprechungen zu diesem Thema: [Quelle](#)
- Außerdem müssen oft mehrere Fontsdateiformate angeboten werden, um alle Browser zu bedienen

FLUID DESIGN TEXT - FONTS

- Es gibt weitere zahlreiche Schriftformate: EOT, WOFF, TTF und SVG
- Nicht nur Schrift können über Fonts dargestellt werden sondern auch Bilder
- Um weniger Bilder einzeln nachladen zu müssen, verpacken Webdesigner mitunter Icons von Schaltelementen oder auch Bedienelementen in speziell erzeugten Fonts
- Z.B. [FontAwesome](#) bietet eine große Sammlung an Icons an, welche sich als Font einbinden lassen
- Die Icons werden dann wie Text aufgerufen und einzeln mit Boxen platziert

RESPONSIVE BILDER UND VIDEOS



Quelle

RESPONSIVE IMAGES - GUIDES FÜR ANWENDUNGEN MIT BEISPIELEN

- `<picture>` (z.T. in `<figure>`): `srcset` und `sizes`
- [Google Guide](#)
- [W3Schools](#)

HTML5 - PICTURE ELEMENT - SOURCE

SAMPLE MARKUP FOR `PICTURE`

```
<picture>
  <source media="(min-width: 40em)"
    srcset="big.jpg 1x, big-hd.jpg 2x">
  <source
    srcset="small.jpg 1x, small-hd.jpg 2x">
  
```

Quelle

HTML5 - PICTURE ELEMENT - SRCSET

SAMPLE MARKUP FOR `SRCSET` AND `SIZES`

```

```

Quelle

RESPONSIVE IMAGES - BEISPIEL - 1

Bis 600px Breite – erstes Bild, danach zweites Bild.
Gleichzeitig fallback: `` für ältere Browser

```
1 <figure>
2   <picture>
3     <source srcset="a-square.png" media="(max-width: 600px)">
4     
6   <figcaption>Barney Frank, 2011</figcaption>
7 </figure>
```

RESPONSIVE IMAGES - BEISPIEL - 2

Durch die CSS-Angaben zu den Bildmaßen wird erreicht, dass gleich beim ersten Rendern der Webseite exakt Platz eingerichtet wird, auch wenn das Bild noch im Ladeprozess und damit nicht messbar sein sollte

```
1 <figure>
2   <style scoped>
3     #a { width: 300px; height: 150px; }
4     @media (max-width: 600px) { #a { width: 100px;
5       height: 100px; } }
6   </style>
7   <picture>
8     <source srcset="a-square.png" media="(max-width:600px)">
9     
11  <figcaption>Barney Frank, 2024</figcaption>
12 </figure>
```


RESPONSIVE IMAGES - BEISPIEL - 3

Alternative mit Angabe von height und width

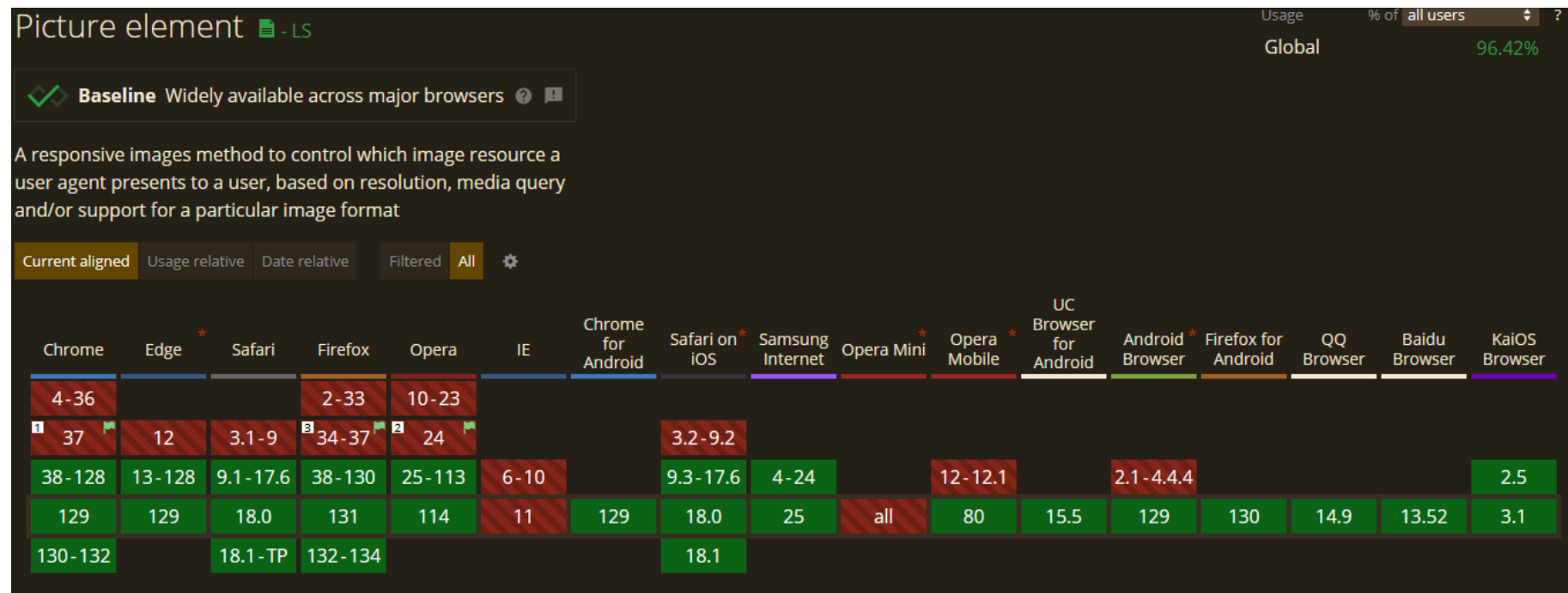
```
1 <figure>
2   <style scoped media="(max-width: 600px)">
3     #a { width: 100px; height: 100px; }
4   </style>
5   <picture>
6     <source srcset="a-square.png" media="(max-width:
7     600px)">
8     
12   </picture>
13   <figcaption>Barney Frank, 2011</figcaption>
14 </figure>
```

RESPONSIVE IMAGES - BEISPIEL - 4

Mehrere Bilder stehen zur Auswahl

```
1 <figure>
2   <picture>
3     <source media="(min-width: 45em)" srcset="lrg.jpg">
4     <source media="(min-width: 32em)" srcset="med.jpg">
5     
7   </picture>
8   <picture>
9     <source srcset="/uploads/100-marie-lloyd.webp"
10       type="image/webp">
11     <source srcset="/uploads/100-marie-lloyd.jxr"
12       type="image/vnd.ms-photo">
13     
15   </picture>
```

HTML5 - PICTURE ELEMENT - BROWSERUNTERSTÜTZUNG



Quelle

FLUID DESIGN BILD

Um Bilder und Videos dynamisch in der Größe anzupassen und zugleich immer komplett zu zeigen, bindet man sie normal ein und legt dann in CSS folgendes fest

```
` nutzt, dann bindet man die effektive Bildbreite an das äußere Element

```
1 <figure style="max-width:50%; margin: 1rem 1rem 1rem 1rem;">
2
6 <figcaption>Ein flexibles Bild</figcaption>
7 </figure>
```

Als äußere Container können `<article>`, `<section>` oder `<aside>` dienen, oder auch andere

## FLUID DESIGN BILD - FLUID FIGURE - 2

float:left oder float:right im Element <img> – evtl., um glatte Ränder zu bekommen bzw. clear:both für die Bildunterschrift, sonst kann ein kurzes erstes Wort zwischen die Bilder rutschen

```
1 <figure style="max-width:50%; margin: 1rem 1rem 1rem
2 1rem;">
3
6
9 <figcaption>Zwei flexible Bilder</figcaption>
10 </figure>
```

Als äußere Container können <article>, <section> oder <aside> dienen, oder auch andere

## FLUID DESIGN BILD - FLUID FIGURE - 3

Alternativ können Bilder bei Verkleinerung ihres Containers einfach beschnitten oder beschnitten und zugleich mit Scrollbalken versehen werden

```
1 <div style="max-width:50%; height:297px;
2 margin: 1rem 1rem 1rem 1rem; border: 1px solid #000;
3 overflow:scroll">
4
5 </div>
```

Der Wert für overflow: kann auch auto oder hidden sein. Eine Anweisung float:right für <img> sorgt für eine feste rechte Seite



## FLUID DESIGN BILD - FLUID FIGURE - 4

Fließende Übergänge für ihre Größe verändernde Bilder kann man z.B. wie folgt in CSS anweisen

```
1 -webkit-transition: max-width .5s ease-out;
2 /* Saf3.2+, Chrome */
3 -moz-transition: max-width .5s ease-out;
4 /* FF4+ */
5 -ms-transition: max-width .5s ease-out;
6 /* IE10? */
7 -o-transition: max-width .5s ease-out;
8 /* Opera 10.5+ */
9 transition: max-width .5s ease-out;
```

Der Wert für overflow: kann auch auto oder hidden sein. Eine Anweisung float:right für <img> sorgt für eine feste rechte Seite

## BEISPIEL UND GUIDES FÜR FLUIDE BILDER MIT VERSCHIEDENEN STYLES UND SOURCES

- [Live-Beispiel von responsiven Bildern](#)
- [Guide für responsive Bildern](#)
- [Syntaxhinweise in HTML5](#)
- [Tutorial von Mozilla](#)
- [W3Schools Anleitung](#)

## FLUID DESIGN BILD - LADEZEITENOPTIMIERUNG

- Nicht immer soll das Bild in Originalauflösung übertragen werden, z.B. im mobilen Bereich
- Deshalb wäre es denkbar, mehrere Auflösungen eines Bildes anzubieten oder das Bild vorher entsprechend der Übertragungskapazität umzurechnen
- Die sofortige Angabe des Viewports des Bildes vermeidet einen zweiten Rendervorgang der Webseite nach dem Laden (je-)des Bildes
- Das spart Strom und bietet sofort konsistentes Design
- Im moment gibt es hierfür noch keinen Standard aber JS-Bilbiotheken wie JQuery, die dies Ermöglichen

## FLUID DESIGN VIDEO

- Bei Videos ist wegen der Bilddynamik alles noch etwas komplizierter
- Umrechnen ohne Streaming erfordert sowieso zu viel Zeit
- Aber oft werden Videos von Youtube oder Vimeo eingebunden, was in iframes erfolgt
- Damit funktioniert (aktuell) die Methode max-width noch nicht
- Weiteres: [CSS-Tricks](#)

## FLUID DESIGN VIDEO

```
<video controls>
 <source src="video-small.mp4" type="video/mp4" media="all and (max-width
 <source src="video-small.webm" type="video/webm" media="all and (max-wid
 <source src="video.mp4" type="video/mp4">
 <source src="video.webm" type="video/webm">
</video>
```

Anpassbare Videogrößen

## LITERATUR

- [CSS Tricks Anleitung](#)
- [Google Guide zum responsiven Layouting](#)
- [Mozilla Guide und Beispiele](#)

## QUELLEN - 1

- Michael Jendryschik, „Allen recht“, (zu Media Queries), [Quelle](#), iX 09/2010
- Mozilla Developer Network, „CSS Media Queries“, [Quelle](#), 26.08.2013
- Responsible Webdesign, [Quelle](#) und 4 weitere Teile
- Apple, „Supported Meta Tags“, [Quelle](#)
- Christoph Zillgens, „Responsive Webdesign: Worauf es beim Einsatz reaktionsfähiger Typografie ankommt“, [Quelle](#), (und drei weitere Beiträge verlinkt)

## QUELLEN - 2

- Tom Brown, „More Meaningful Typography“, [Quelle](#), 03.05.2011 – modular scales for typography
- Conor MacNeill = @thefella, „Responsive Type References“, [Quelle](#), 2011, mit Testwebseiten und weiteren Verweisen
- [Quelle](#)
- Picturefill Home: [Quelle](#), Demo: [Quelle](#)
- Adaptive Images Home und Demo: [Quelle](#)
- HiSRC Home: [Quelle](#)
- [Quelle](#)



## QUELLEN - 3

- Responsive Images Home: [Quelle](#)
- Responsive Images Alt Desc: [Quelle](#)
- Foresight.js Home: [Quelle](#)
- [Library für responsive Inhalte](#)
- Rloadr Home: [Quelle](#), Examples: [Quelle](#)
- Responsive Enhance: [Quelle](#)
- rwdImages Home: [jQuery res Images](#)
- Content Aware Resizing Home: [GetImag Library](#)

## QUELLEN - 3

- Doubletake Home und Demo: [Quelle](#)
- Responsive Images mit Cookies Home: [Quelle](#)
- Responsive IMGs: [Quelle](#) und [Quelle](#)
- How TO - Responsive Images: [Quelle](#)
- Why responsive images?: [Quelle](#)

## **ABSPANN**

Viertes Level geschafft weitere Folgen!

Fragen und Feedback?