

# **BILDER IN HTML5**

**MMWP2024 - LV07**

## INHALTSVERZEICHNIS

- Organisation
- Content von Webseiten
- High Density
- Responsive Images

## INHALTSSCHWERPUNKTE

- Erklärung der Bedeutung von Bildern für Webseiten
- Vorstellung von modernen Bildformaten
- Asynchrones Laden von Inhalten
- Responsive Bilder

## VORAUSSETZUNG

Der Ausgangspunkt dieser Vorlesungsreihe ist das Wissen über funktionsweise von BrowserAPIs

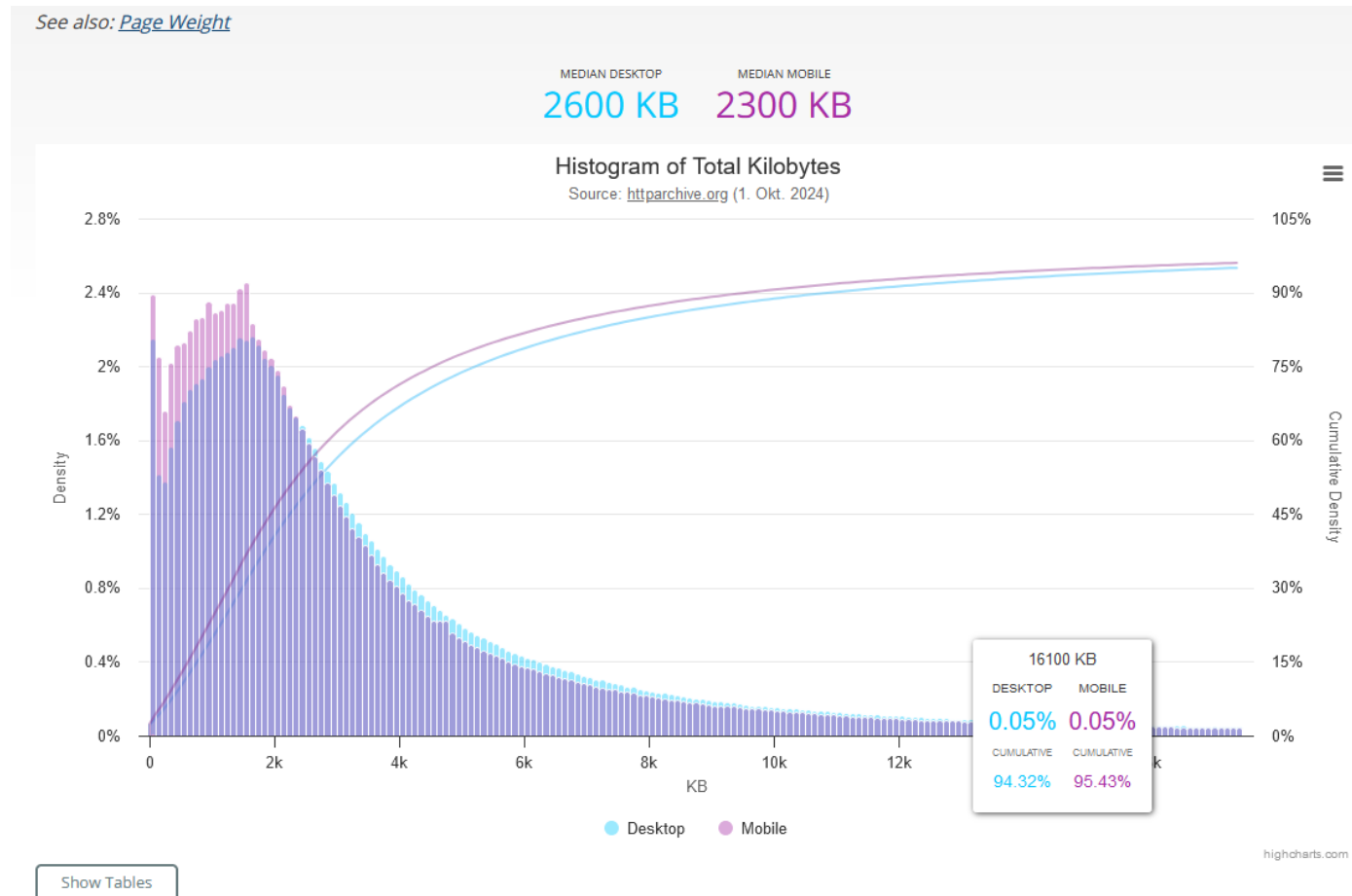
- Verständnis von Inhaltsplatzierung in HTML
- Responsive Webdesign
- Asynchrones Javascript zum Laden von Inhalten

## ZIELE

Vorstellung von:

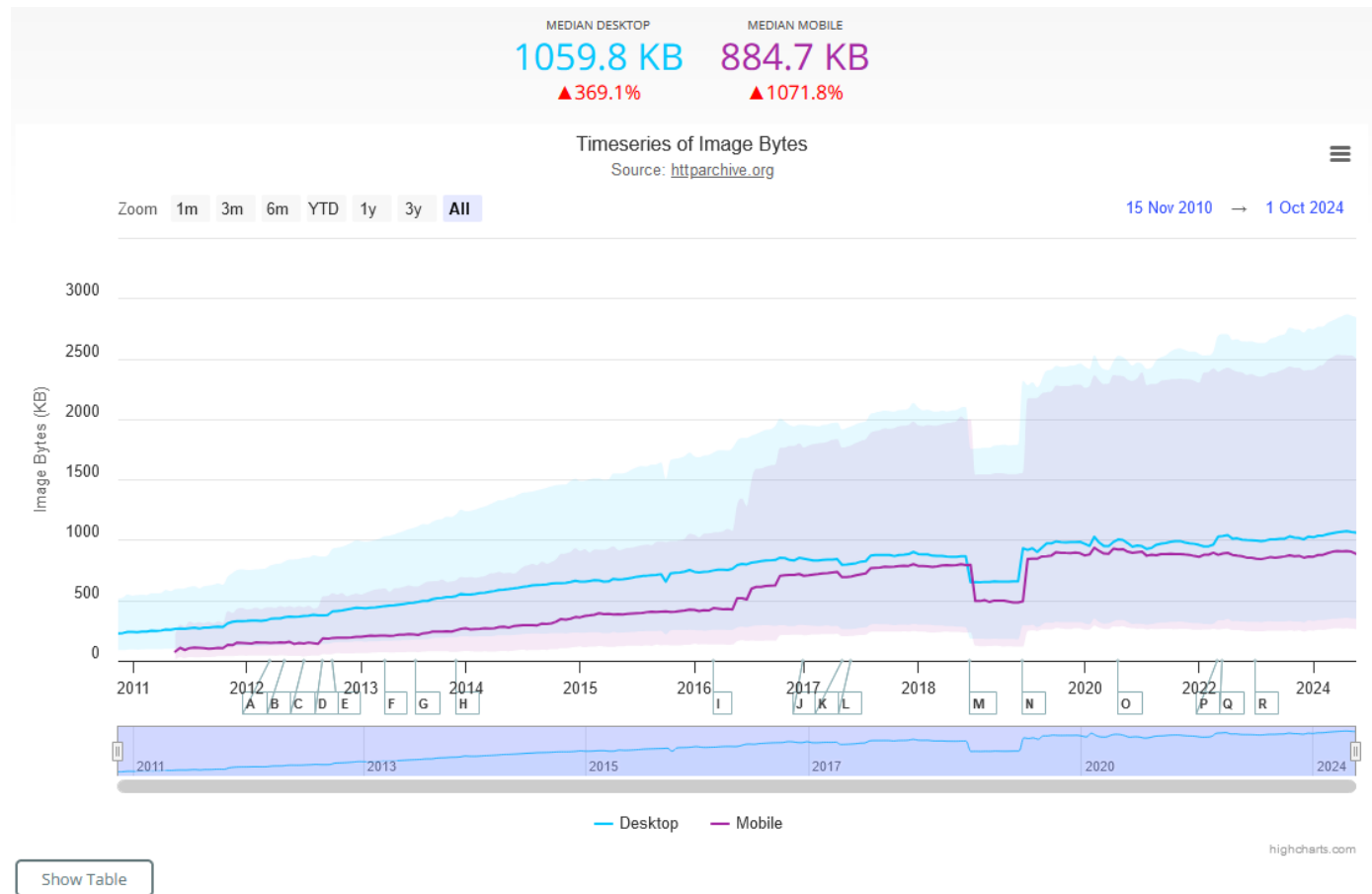
- Modernen Bilderformaten (APNG, AVIF, WebP)
- Beispiele von Inhaltsdarstellung
- Asynchrones Verhalten von Inhalten
- High Density Displays
- Bilder und Media Queries

# SPEICHERGRÖSSEN VON WEBSEITEN



Quelle

# SPEICHERGRÖSSEN VON BILDERN AUF WEBSEITEN

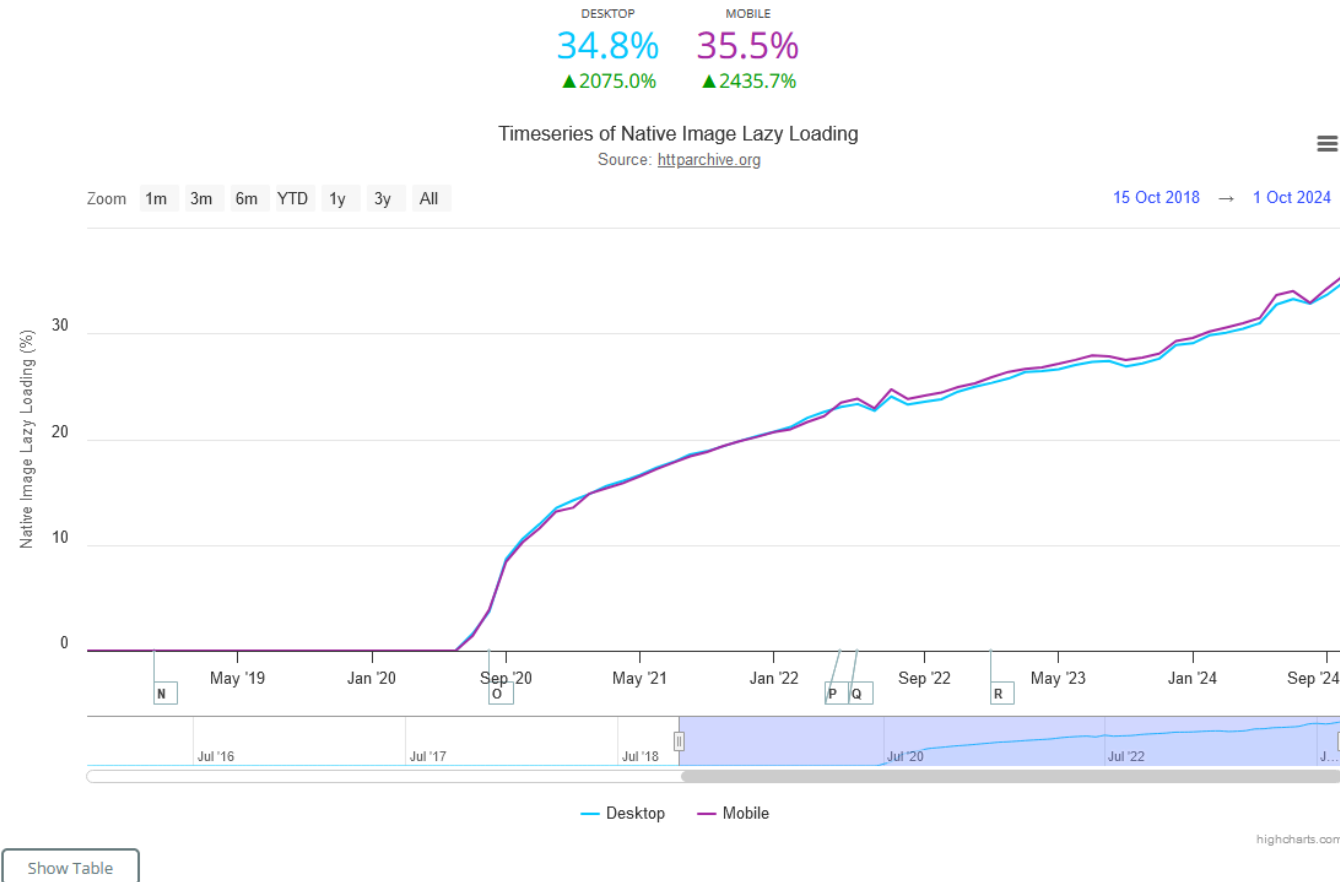


Quelle

# ANTEIL VON ASYNCHRON GELADENEN BILDERN

## Native Image Lazy Loading

The percent of pages that have the `loading=lazy` attribute on `img` elements.



Quelle



## BILDER IN WEBSEITEN - 1

- Bilder im Kontext von Webseiten wurden bisher so betrachtet, dass die Pixelanzahl proportional zu den Abmaßen des Bildes bei der Darstellung auf einem gegebenen Bildschirm im Browserfenster ist
- Das Verhältnis zwischen Pixelanzahl und Abmaßen von Bildschirmen stimmt aber nicht mehr
- Es gibt zunehmend kleine Bildschirme hoher Auflösung (Siehe vorherige Vorlesungen)

## BILDER IN WEBSEITEN - 2

- Standpunkt der Pixeldichte / Auflösung her die richtigen Bildversionen gleichen Datentyps den richtigen Bildschirmen zuzuordnen?
- Die Abmaße eines Bildes in Pixeln entspricht genau seinen Abmaßen in der Verarbeitung in CSS-Pixeln (dank viewport einstellung)
- Auch der Ansatz One-Size-Fits-All ist nicht korrekt (50MB 4K Bilder auf kleinen mobil Geräten bei 3G)

## "HISTORISCHE" BILDFORMATE - 1

- JPEG
  - Familie von Standards verlustbehafteter Bildkompressionsalgorithmen
  - Umfangreiche Metadaten in EXIF-Header möglich
  - Unterstützt keine Transparenzen oder Animationen
- GIF
  - Verlustfreie Kompression
  - geringe Farbtiefe (8/16/32 Bit), Speicherung von Farbpaletten möglich (Kompression)
  - Transparenzen und Animationen möglich
  - Speicherung interlaced möglich

## "HISTORISCHE" BILDFORMATE - 2

- PNG
  - Verlustfreie Kompression
  - blockbasierte Kodierung (gut zum asynchronen Laden)
  - Farbpaletten möglich
  - RGB-Kanäle und zusätzlicher Alpha-Kanal für Transparenzen
  - Animationen nicht möglich

# WEITERE BILDFORMATE

- SVG
  - Vektorgrafikformat
  - Breite Browserunterstützung
  - Komprimiert oder im Quelltext (lesbares Format)
  - Einschluss von Bitmap-Grafiken z.B. JPEG, GIF, PNG möglich
  - Animationen nicht möglich
- WEBP
  - Breite Browserunterstützung erst seit kurzem
  - Verlustbehaftete und verlustfreie Kompression
  - Blockbasierte Kodierung
  - Entropiekodierung

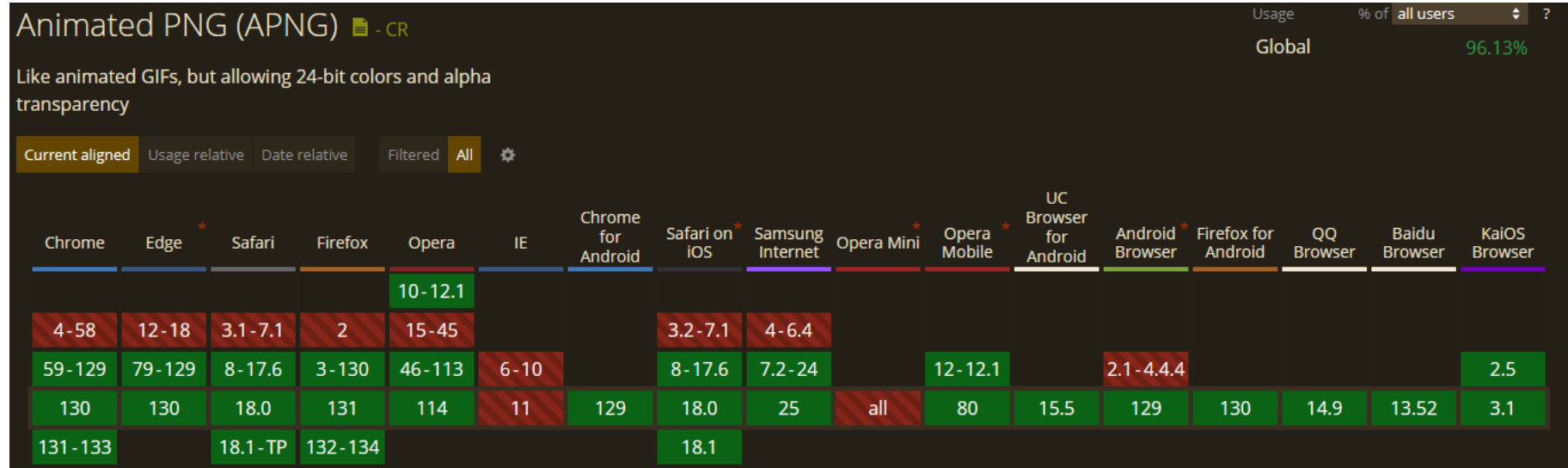
## WEBP - ENTROPIECODIERUNG

- David A. Huffman hat 1952 einen Algorithmus zur Konstruktion von optimalen präfixfreien Codes angegeben
- Die häufigsten Elemente werden möglichst kurz codiert
- WebP setzt auf diese Technik, um kleinere Dateigrößen zu ermöglichen
- Statisch und nicht mehr einfach änderbar
- [Mehr zu Entropiecodierung](#)

# "NEUE" BILDFORMATE

- APNG
  - APNG ist eine Erweiterung des PNG-Formats mit Unterstützung für animierte Bilder
  - Es ist als Ersatz für einfache animierte Bilder gedacht (Gif)
  - Unterstützt gleichzeitig 24-Bit-Bilder und 8-Bit-Transparenz
  - APNG ist rückwärtskompatibel mit PNG
  - Jeder PNG-Decoder kann die APNG-spezifischen Chunks ignorieren und ein einzelnes Bild anzeigen
  - Siehe [Spezifikation](#)
- AVIF (AV1 Image File Format)
  - Das AV1 Image File Format (AVIF) ist eine Codierung, die auf dem Open-Source-AV1-Videocodec basiert
  - Seit 2020 in Chromium unterstützt aber seit 2022 von Safari (Webkit) nicht mehr unterstützt
  - firefoxbasierte Browser haben im moment nur Teilsupport bis keinen (letzte Änderungen 2023)
  - Zeigt im Vergleich zu JPEG oder WebP deutliche Reduzierung der Dateigrößen (ohne Qualitätverluste oder Kompressionen)
  - Siehe [AVIF BLog](#)

# APNG SUPPORT



Quelle

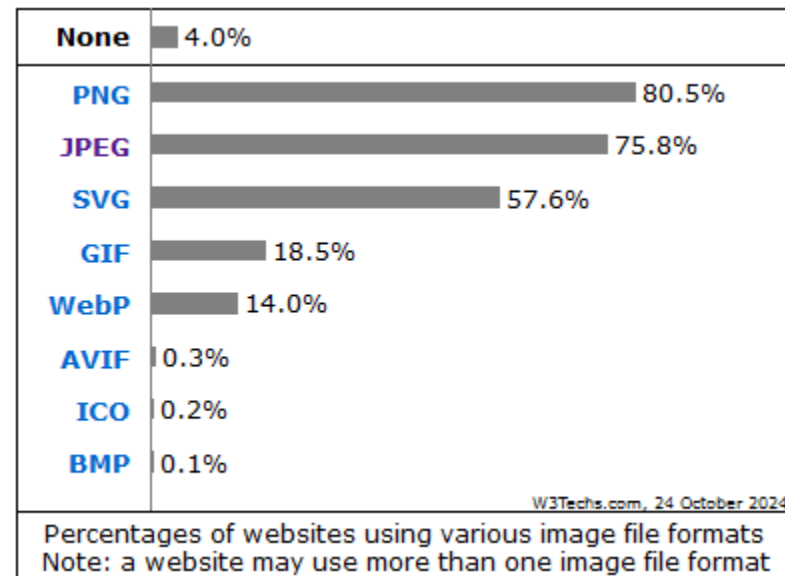


# AVIF SUPPORT



Quelle

# ANTEILE DER BILDFORMATE AUF WEBSEITEN (2024)



Quelle

# VERGLEICH DER ÄLTEREN BILDFORMATE

Eigenschaft	JPEG	PNG	GIF	WebP	SVG
fotorealistisch	+	+		+	
Computergrafik		+	+	+	+
Verlustbehaftet	+			+	
Verlustlos		+	+	+	+
Transparenz		+	+	+	+
Animation			+	+	+
Verlustfrei skalierbar					+
Browserunterstützung	+	+	+	+	+

Quelle

# DAS BILDELEMENT <IMG> - 1

Attribut:	Funktion:	Verwendung:
<b>src</b>	Bindet ein Bild von einer Datei ein	<b>Pflichtattribut</b>
<b>alt</b>	Eine Bildbeschreibung als Text	<b>Pflichtattribut</b> (zur Barrierefreiheit)
<b>crossorigin</b> ( <i>anonymous / use-credentials</i> )	Anonyme Nutzung oder mit Erlaubnis	Optional, bei Zugriff auf fremde Ressourcen
<b>height</b>	Höhe des Bildes	Optional, bei Wunschanpassung
<b>sizes</b>	Liste alternativ wählbarer Größen	Eine Bildgröße wird abhängig vom Bildschirm eingestellt
<b>srcset</b>	Liste alternativ wählbarer Bilddateien als Ressource	Eine Bilddatei wird abhängig vom Bildschirm geladen

Quelle

## DAS BILDELEMENT <IMG> - 2

Attribut:	Funktion:	Verwendung:
<b>title</b>	Tooltip (angezeigt bei Mouse-Over)	Optional (dient zur Barrierefreiheit)
<b>width</b>	Legt die Breite des Bildes fest	Optional bei Wunschanpassung der Größe des Bildes
<b>align</b>	Ausrichtung	in CSS (links, rechts, mittig)
<b>border</b>	Rahmen	Nur in CSS verwenden
<b>hspace</b>	Abstand horizontal	Nur in CSS verwenden
<b>vspace</b>	Abstand vertikal	Nur in CSS verwenden

Quelle

## BILDER IN HTML UND CSS

```
1 <img src='cat.jpg' alt='Katze'> <!-- als Bildelement im HTML-->
2 <body background=„hintergrund.jpg“> <!-- als Hintergrund im HTML -->
3 .main {background-image: url('pattern.jpg');} // als Hintergrund im CSS
4 .main:after {content: url('pattern.jpg');} // als Inhalt im CSS
5 .main:before {content: url('pattern.jpg');} // als Inhalt im CSS
```

## BEFORE AND AFTER IM CSS

```
1  /* CSS2 Syntax */ element:after { Stileigenschaften }
2  /* CSS3 Syntax */ element::after { Stileigenschaften }
3
4  Die ::after Notation (mit zwei Doppelpunkten) wurde in CSS 3
5  eingeführt, um eine Unterscheidung zwischen Pseudoklassen
6  und Pseudoelementen einzuführen. Browser unterstützen auch
7  die :after Notation, wie sie in CSS 2 eingeführt wurde
8
9  selector : pseudo-class { property: value; }
10 selector :: pseudo-element { property: value; }
```

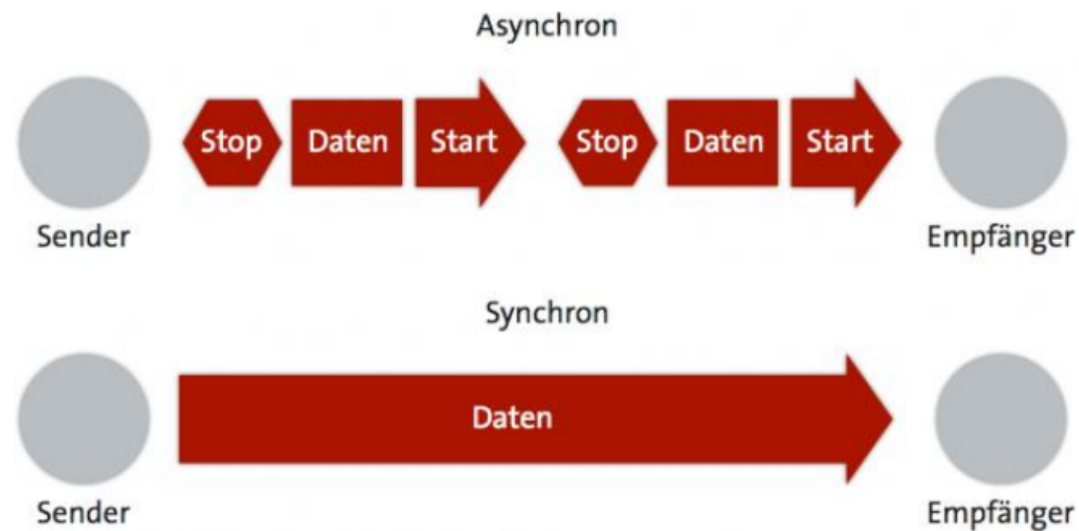
## BILDER MIT JAVASCRIPT LADEN

```
1 function lazyload () {  
2     if(lazyloadThrottleTimeout) {  
3         clearTimeout(lazyloadThrottleTimeout);  
4     }  
5  
6     lazyloadThrottleTimeout = setTimeout(function() {  
7         var scrollTop = window.pageYOffset;  
8         lazyloadImages.forEach(function(img) {  
9             if(img.offsetTop < (window.innerHeight + scrollTop)) {  
10                img.src = img.dataset.src;  
11                img.classList.remove('lazy');  
12            }  
13        });  
14  
15        if(lazyloadImages.length == 0) {
```

Siehe [CSS-Tricks](#)



## BILDER ASYNCHRON LADEN - IDEE



Quelle

# AJAX (ASYNCHRONOUS JAVASCRIPT AND XML)

```
1 var xmlhttp = false; //globale Instanz von XMLHttpRequest
2 //XMLHttpRequest-Instanz erstellen
3 try {
4     xmlhttp = new ActiveXObject("Msxml2.XMLHTTP");
5 } catch(e) { try {
6     xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
7 } catch(e) { xmlhttp = false; } } //... für Internet Explorer
8 //... für Mozilla, Opera, Safari usw.
9 if (!xmlhttp && typeof XMLHttpRequest != 'undefined') {
10     xmlhttp = new XMLHttpRequest(); }
11
12 function loadData(url,id) {
13     if (xmlhttp) {
14         xmlhttp.open('GET', 'loadimage.php?id=' + id + '&url=' + url, true);
15         xmlhttp.onreadystatechange = answer;
```

Siehe [CSS-Tricks](#)

## JQUERY ASYNCHRONOUS IMAGE LOADER (JAIL)

```
1 
2 <noscript>
3 
4 </noscript>
5 <script src="/lib/jquery.js"></script>
6 <script src="/src/jail.js"></script>
7 <script>
8   $(function(){
9     $('img.lazy').jail();
10  });
11 </script>
```

Siehe [Jail Beispiel](#)

## VORTEILE VON ASYNCHRONEN LADEN

- Entkoppelung von Darstellung relevanter Inhalte und der vollständigen Anzeige und Nutzbarkeit der Website
- Der Nutzer erhält schneller ein Ergebnis
- Die Website reagiert agiler, weil früher mehr zu sehen ist und genutzt werden kann
- Außerdem lässt sich steuern, wann der Inhalt angezeigt wird (z.B. über javascript)
- Dieser lässt sich optisch ansprechend darstellen, was den Joy-of-Use erhöhen kann

## NACHTEILE VON ASYNCHRONEN LADEN

- Die Voraussetzung, dass JavaScript zur Verfügung stehen muss (Noscript)
- In der Wirkung kann die Seite allgemein als „unruhig“ empfunden werden (Bewegungen)
- Durch zu viele AJAX-Requests kann Ladezeit auch negativ beeinflusst werden
- Caching-management, damit Inhalte nicht mehrfach geladen werden (z. B. nach einem Seitenwechsel)
- Siehe [Blogbeitrag zur Ladezeitenoptimierung](#)

# MASSNAHMEN ZUR OPTIMIERUNG

Nach Testergebnissen von E. Schlag sind folgende Optimierungen effektiv:

Maßnahme	Ladezeitreduktion [%]	Datenreduktion [%]
Caching (client- und serverseitig)	42-82	99-100
Infinite Scroll	40-62	63-80
Lazy Loading	36-58	
W-Descriptor	32	23-53
WebP über picture	19-26	
Inline-SVG		9

User-Agent-Sniffing/srcset allein: ohne Wirkung

## LIVE-BEISPIEL

- Unoptimiert
- Asynchron
- Mehrfachsupport
- Zugeschnittene Bilder

# HIGH DENSITY PIXEL

## Die Pixeldichte von Bildschirmen

$$\text{Pixeldichte} = \frac{\sqrt{(\text{Breite in Pixel})^2 + (\text{Höhe in Pixel})^2}}{\text{Bildschirmdiagonale in Zoll}}$$

- Pixel per Inch (PPI) ist das Analogon zur Größe DPI in der Druckereiindustrie
- ab 240 PPI spricht man von High Density
- ab 287 PPI soll ein durchschnittlicher Mensch die Einzelpixel im Bild nicht mehr sehen können (modulo Sehschärfe und Geräteabstand)



# HIGH DENSITY DISPLAYS

JPEG (unkomprimiert) 71,0 KB JPEG (50% komprimiert) 11,9 KB Ausschnitt-Vergrößerung



Quelle

## HIGH DENSITY DISPLAYS - MASSEINHEITEN

- Um Bilder nicht sinnlos klein dargestellt zu bekommen, wurden verschiedene PixelMaße eingeführt
- Physische Pixel: Tatsächliche Ausmaße eines Objektes auf einem Display in Pixeln (Breite mal Höhe)
- Logische Pixel: Konstanter, für die interne Kalkulation verwendeter Wert, auch dips (device-independent pixels) oder CSS-Pixel genannt
- Bei unterschiedlichen Zoom-Leveln einer Webseite werden immer die CSS-Pixel-Werte angegeben

## VIEWPORT

- Für Desktops ist bzw. war der sichtbare Bereich im Browserfenster gleich dem Bereich für Layout-Berechnungen
- Bei Smartphones, Tablets etc. wurde dies direkt eingeführt
- Visual Viewport: sichtbarer Bereich
- Layout Viewport: Bereich für Layout-Berechnungen (auch teilweise unsichtbar)
- Standard ist somit, den Layout Viewport zu rendern und das Ergebnis in den Visual Viewport hineinzuzoomen (meist zu verkleinern)

## HIGH DENSITY DISPLAYS - SMARTHPONES

- Im Ergebnis sieht man die gesamte Webseite auf dem Display des Smartphones, Tablets etc.:  
Gerätedisplaybreite ist gleich Webseitenbreite.
- Um vor dem Rendern den Standard Layout Viewport zu überschreiben und den Visual Viewport mit dem Layout Viewport gleichzusetzen, gibt es zwei alternative Anweisungen:

### HTML

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

### CSS

```
@viewport {width : device-width; zoom : 100%;}
```

# VIEWPORT METHODEN UND PARAMETER

Eigenschaft	CSS-Deklaration		meta-Tag	
	Schlüssel	Werte	Schlüssel	Werte
Breite	width	auto, Länge, Prozentsatz	width	device-width, Zahl
Start-Zoom	zoom	auto, Zahl, Prozentsatz	initial-scale	Zahl
Zoom-Veränderung	user-zoom	fixed, zoom	user-scale	yes, no
Ausrichtung	orientation	auto, portrait, landscape	-	-

Quelle

# VIEWPORT SUPPORT



Quelle

# MIT VIEWPORT TAG UND OHNE VOREINSTELLUNG

Beispiel

## TESTSEITE VON PETER-PAUL KOCH

- Eine sehr gute Testseite für Browser gibt es unter
- [Grid und responsive Testseite](#)
- Dort kann man über zwei Links die Weite des Viewports setzen bzw. im Standardmodus die natürliche Weite (DPR) des Viewports abfragen



## HISTORISCHE DATEN ÜBER VIEWPORT

- Das meta-Element viewport wurde von Apples iPhone-Safari-Browser etwa 2008 eingeführt
- „CSS Device Adaption Module Level 1“ – Editor‘ Draft, 05. 07. 2018 (siehe [Draft Approach](#))
- Vernünftiges Aussehen erzielt man mit responsivem Design und seinen Methoden

## DEVICE PIXEL RATIO (DPR)

- Displays unter 200 DPI haben DPR=1
- Displays zwischen 200 und 300 DPI haben DPR=2
- Displays mit mehr als 300 PPI:  $DPI = (PPI / 150)$  abgerundet
- Neuere Algorithmen zur Bestimmung der Darstellungsgröße von Bildern bauen auf der Größe DPR auf

## DEVICE PIXEL RATIO (DPR) - BEISPIELE - 1

- Beispiel zur Nutzung eines bestimmten Bildes bei mindestens DPR=2 bzw. mindestens DPI=192:
- So kann vor dem Hochladen eines Bildes entschieden werden, welches Bild es sein soll (96dpi = 1dppx, Fallback)
- [Resolution Dokumentation](#)

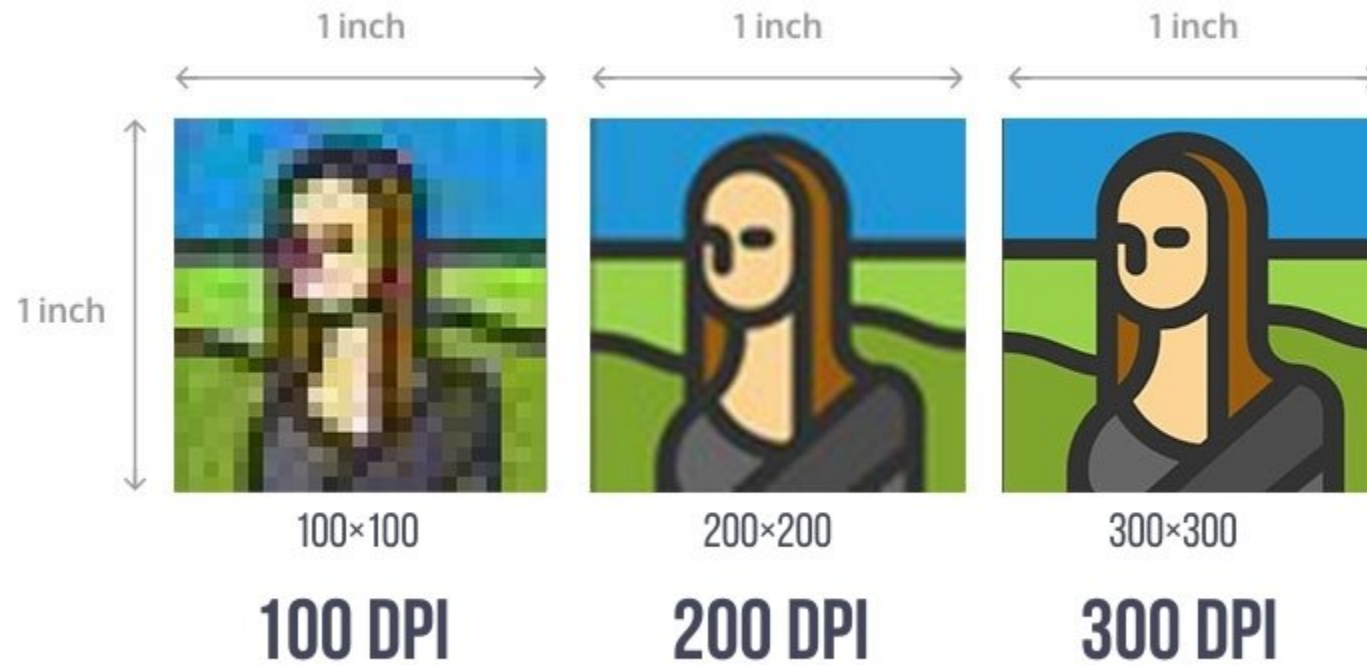
```
1 @media only screen and (-webkit-min-device-pixel-ratio: 2),  
2   only screen and (min-resolution: 2dppx),  
3   only screen and (min-resolution: 192dpi),  
4   .btn {background-image: url(image@2x.jpg);  
5 }
```

## DEVICE PIXEL RATIO (DPR) - BEISPIEL - 2

- Beispiel zur Nutzung eines bestimmten Bildes bei mindestens DPR=2 bzw. mindestens DPI=192:
- So kann vor dem Hochladen eines Bildes entschieden werden, welches Bild es sein soll (96dpi = 1dppx, Fallback)

```
1 <figure>
2   <picture>
3     <source media="(min-resolution: 4dppx)" srcset="image@4x.jpg">
4     <source media="(min-resolution: 2dppx)" srcset="image@2x.jpg">
5     
6   </picture>
7   <figcaption>Beispiel</figcaption>
8 </figure>
```

## BEISPIEL



Quelle

## DEVICE PIXEL RATIO (DPR) - BEISPIEL - 2

- HTML Living Standard [HTML-Picture standard](#) (Updated 23 October 2024)
- zur Unterstützung älterer Browser gibt es z.B. Picturefill (Responsive Image Polyfill) (siehe [Picturefill](#))
- Polyfill bzw. Emulation von <picture>-Element (Es stellt Bilder je nach Bildschirmgröße, Viewport-Größe und Auflösung angemessen dar)

## BILDER ZUR WAHL JE NACH DPR - 1

Das WHATWG hat folgende Syntax vorgeschlagen, um minimalistisch eine Auswahlösung zu schaffen:

```

```

## BILDER ZUR WAHL JE NACH DPR - 2

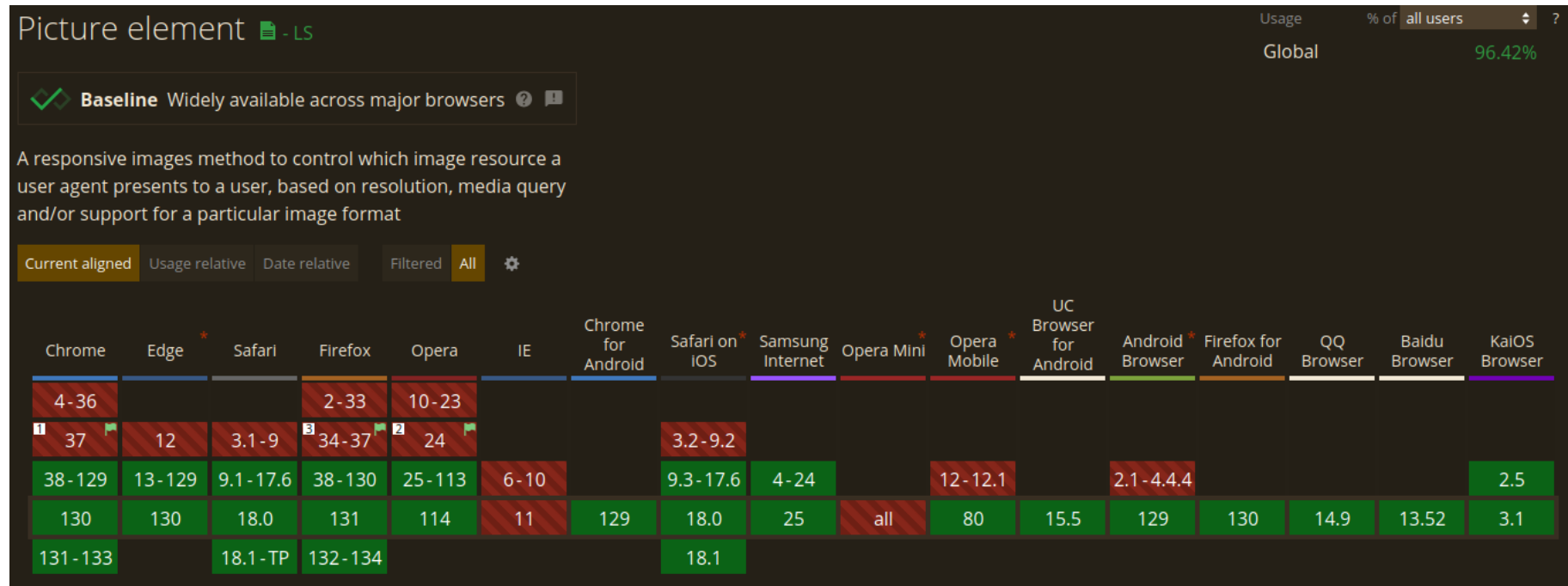
```
1 <div class="container">
2   
14 </div>
```



## BILDER ZUR WAHL JE NACH DPR - 3

```
1 
```

# PICTURE - SUPPORT



Quelle

## RESPONSIVE IMAGES

- Es wurde eine Lösung gesucht, die allein für ein einziges img-Element schon funktioniert und wesentlich flexibler reagiert
- HTML Living Standard: [Embedded content standard](#) (Updated 23 October 2024)
- Picture-Element als Container mit:
  - source-Element zum Anbieten verschiedener Bilder
  - img-Element als Fallback zur Darstellung eines Default-Bildes

# VORSTELLUNGEN

Variable	Dem Entwickler beim schreiben des Codes bekannt?	Dem Browser während des ladens der Seite bekannt?
Viewport-Dimensionen	Nein	Ja
Bildgröße im Verhältnis zum Viewport	Ja	Nein (Abhilfe: sizes)
Bildschirmauflösung	Nein	Ja
Dimensionen der Bilddateien	Ja	Nein (Abhilfe: srcset)

Quelle

## RESPONSIVE BILDER BEISPIELE - 1

```

```

Einheit	Beschreibung
vw: Viewport-Breite	prozentuale Breite des Anzeigebereichs (Viewport): 100vw = Breite des Viewports
vh: Viewport-Höhe	prozentuale Höhe des Anzeigebereichs: 100vh = Höhe des Viewports
w	maximale Breite in CSS-Pixeln, bis zu der das entsprechende Bild zu laden ist

## RESPONSIVE BILDER BEISPIELE - 1

- Wenn der Bildschirm des Nutzers 320 CSS-Pixel groß ist, dann müsste die Spezifikation lauten:  
wolf-400.jpg 1.25x, wolf-800.jpg 2.5x, wolf-1600.jpg 5x
- Wenn der Bildschirm des Nutzers 1200 CSS-Pixel groß ist, dann müsste die Spezifikation lauten:  
wolf-400.jpg 0.33x, wolf-800.jpg 0.67x, wolf-1600.jpg 1.33x
- Der Browser des Nutzers kann die korrekte Wahl des zu ladenden Bildes einleiten

## RESPONSIVE BILDER BEISPIELE - 2

- Mit Responsive Design werden drei Layouts definiert:
- Schmalere Viewport: eine Spalte Bilder
- Mittlerer Viewport: zwei Spalten Bilder
- Breiter Viewport: drei Spalten Bilder
- Beispielsweise Breakpoints: 30em und 50em

## RESPONSIVE BILDER BEISPIELE - 2

```

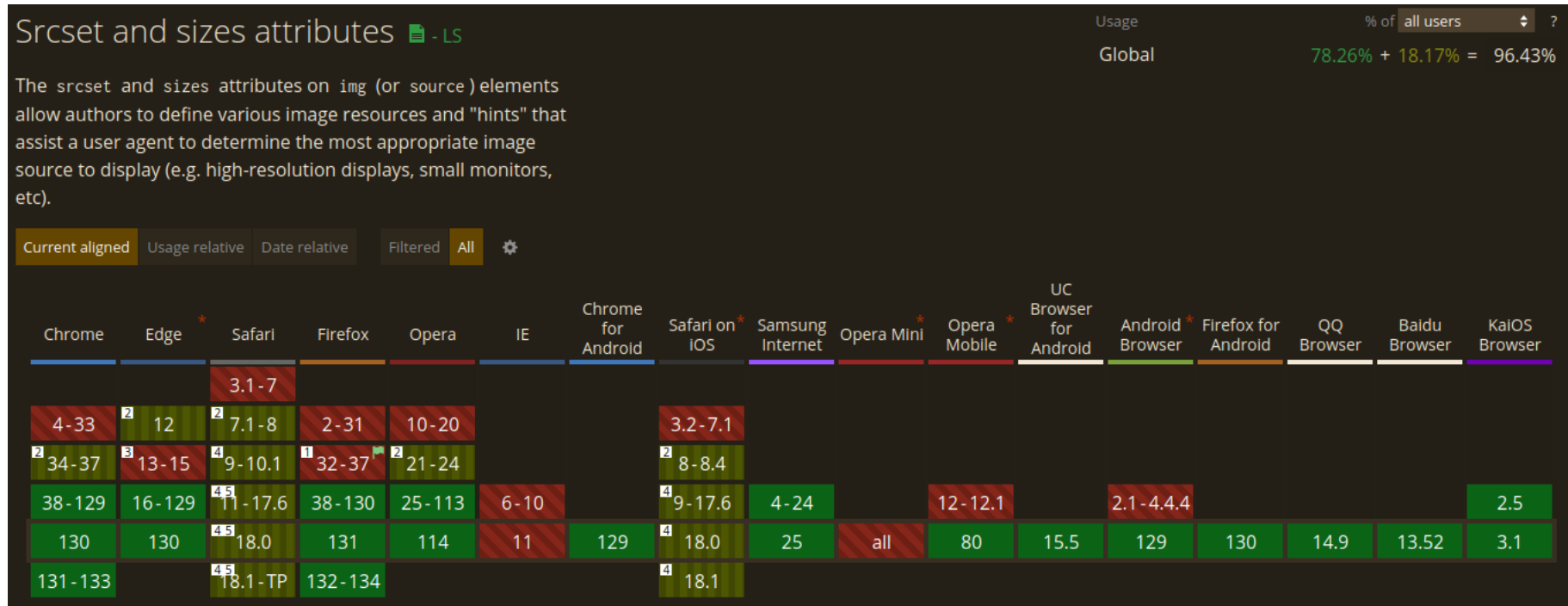
```



## RESPONSIVE BILDER BEISPIELE - 2

- Ist die Viewport-Breite 29em, dann wird die Bedingung (max-width: 30em) erfüllt und 100vw wird benutzt
- Das Bildmaß für die Wahl der Bilddatei ist dann 29em
- Wenn die Viewport-Breite 32em ist, dann wird die Bedingung (max-width: 30em) nicht erfüllt, jedoch die Bedingung (max-width: 50em)
- Also: der Wert 50vw wird benutzt und das Ergebnis ist: die Bilder werden schmaler

# SRCSET - SUPPORT



Quelle

- Größtes Webbild im Internet
- Ladeverhalten von Bildformaten

## WEITERFÜHRENDE INFORMATIONEN

- [HTML Responsive Images Guide \(12.11.2024\)](#)
- [Mozilla - Responsive images \(12.11.2024\)](#)
- [Tutorial auf Kulturbanause](#)

## QUELLEN

- Framework „Adaptive Images“ (minimalistisch, effektiv): [Quelle](#)
- weitere Tipps und Links zu Bibliotheken bzw. Frameworks: [Quelle](#)

## **ABSPANN**

Siebtes Level geschafft weitere Folgen!

Fragen und Feedback?