Responsive Webdesign

Mobile Version

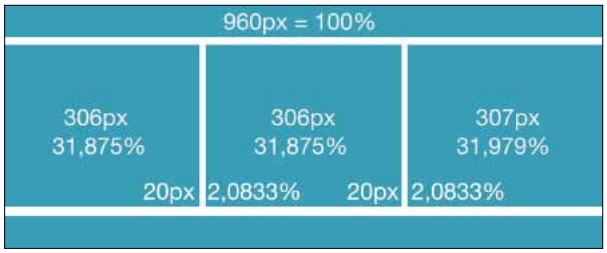
Ob auf dem Smartphone, dem Tablet oder dem Desktop-PC – Optik und Funktionalität der Seite sollen sich an das Endgerät des Nutzers anpassen. **Von Sabrina Sauter**

AUTORIN

Sabrina Sauter
ist Webentwicklerin bei der Agentur S2 Intermedia GmbH, die sich auf hochwertige Unternehmens-Webseiten, E-Business-Lösungen und individuelle browserbasierte Softwarelösungen spezialisiert hat. Neben dem CMS TYPO3 und dem Shop-System Magento interessiert sie sich auch für Javascript-Frameworks.
▸ www.s2intermedia.de

Inhalt
Mobile Webseiten per Responsive Webdesign.

Sowohl als Designer wie auch als Entwickler nähern wir uns im Bereich der Webentwicklung immer schneller dem Punkt an, wo man alleine als Person kaum mehr in der Lage ist, alle Anforderungen an eine Webseite zu beachten. So hat auch die Vielfalt der Smartphones und Tablet-PCs mit unterschiedlichen Bildschirmgrößen, -orientierungen und -auflösungen in den vergangenen fünf Jahren stetig zugenommen – und damit auch die Zahl der Nutzer, die sich auf mobilen Endgeräten durch das Netz bewegen. Im Vergleich zum Vorjahr haben sich 2011 die Zugriffe auf das Internet per Smartphone von 10 Prozent auf 18 Prozent fast verdoppelt. Durch die stetigen Neuerungen und Verbesserungen an Smartphones wird sich dieser Trend sicherlich fortsetzen und verstärken.



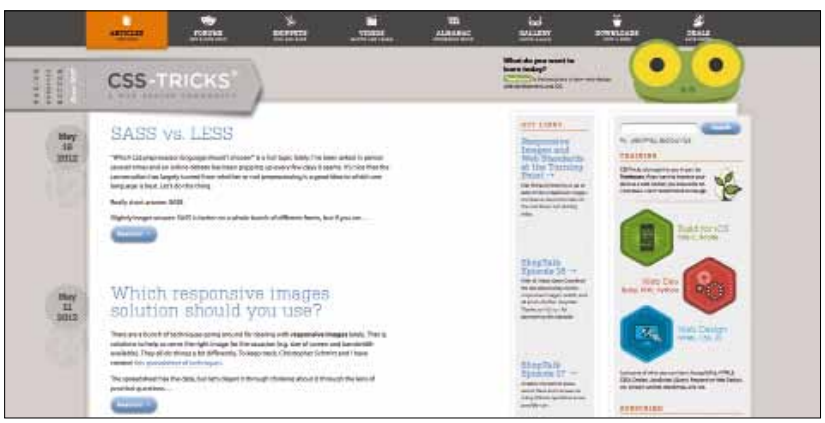
Wie man ein pixelbasiertes, in seinen Größen starres Design in ein Fluid Grid umwandeln kann (Bild 1)

Doch was soll man dieser Flut an Anforderungen entgegenhalten? Ein maßgeschneidertes Design für jede Auflösung und jedes neue Gerät zu entwerfen scheint unmöglich, oder zumindest unpraktisch und ineffizient. Hier gibt es jetzt aber eine andere Möglichkeit.

Responsive Webdesign ist ein Webentwicklungskonzept, das es auf Basis der neuen Technologien wie HTML5 und CSS3 ermöglicht, Webseiten so zu gestalten, dass diese auf das Verhalten und die Umgebung des Nutzers reagieren. Responsive kommt von »to respond«, also auf etwas antworten beziehungsweise reagieren. In der technischen Umsetzung bedeutet dies einen ausgewogenen und intelligenten Einsatz von flexiblen Layouts, den sogenannten Fluid Grids.

Dazu gehören ebenfalls skalierbare Bilder, auch Fluid Images genannt, wie auch Media Queries als gerätespezifische Stilvorgaben. Eine unter diesen Aspekten erstellte Webseite ermöglicht es durch diese Technologie, sich flexibel an die Einstellungen des Nutzers anzupassen. Flexibel beschreibt zum einen die Anpassung von Text, Bildern und Schriftgrößen, aber auch die Möglichkeit, Inhaltselemente individuell zu positionieren oder auszublenden, wenn kein Platz vorhanden ist. Ein nach diesem Ansatz konsequent geplantes Projekt bündelt in Zukunft verschiedene Entwicklungs- und Gestaltungsphasen. Anstatt eine Vielfalt an unterschiedlichen Webseiten für mobile Endgeräte und Desktop-PCs sowie auch Apps zu entwickeln, wird dies in einer einzigen Version für alle Endgeräte und Nutzer erreicht.

Der herkömmliche Weg, ein Webseiten-Design mit HTML und CSS umzusetzen, ist jedem ein Begriff. Das vorgegebene Design wird in festen Pixelbreiten vermessen und in das Stylesheet



Große Darstellung für Desktop-PCs (Bild 2)

Mittlere Darstellung für Desktop-PCs (Bild 3)

übertragen. Das sogenannte Fixed-Width Layout ist beispielsweise in seiner Gesamtheit immer 960 Pixel breit. Die drei beinhalteten Spalten haben hier zum Beispiel immer eine feste Größe von 306 beziehungsweise 307 Pixeln. Eine in diesem Stil aufgebaute Webseite bewegt sich nicht mit, wenn sich die Browserfenstergröße verkleinert. So wird diese irgendwann einfach vom Browserfenster abgeschnitten.

Diesem Layout entgegen steht mit Responsive Design der Ansatz des Fluid Layouts oder das häufiger verwendete Fluid Grid. Hier haben alle Elemente auf der Seite eine definierte Breite, und die Innen- und Außenabstände zu anderen Containern sind in Prozentangaben definiert. Somit passen sich diese Layouts an den Viewport der jeweiligen Geräte an. Der Viewport ist hier das Ausschlaggebende, denn er beschreibt die Breite des Bereichs, die der Webseite auf einem Gerät tatsächlich zur Verfügung steht. Die Webseite bewegt sich somit je nach Größe des Browsers mit allen Inhaltselementen flexibel mit.

Anhand einer einfachen Layoutstruktur kann verdeutlicht werden, wie man ein pixelbasiertes, in seinen Größen starres Design in ein Fluid Grid umwandeln kann **(Bild 1)**.

Als Beispiel dient ein 960 px breites Layout mit drei Spalten, von denen nur die mittlere einen *margin* von jeweils 20 px nach rechts und links erhält. *960px* bilden hier den Ausgangspunkt, der das Gesamtgerüst der Seite definiert und 100 Prozent darstellt. Davon ausgehend können die drei Spalten mit Hilfe eines Dreisatzes in ihre Prozentsätze umgerechnet werden. Mit diesem Ansatz ist eine variable Lösung für Endgeräte gefunden, die sich an verschiedene Orientierungen, aber auch an einen großen Desktop-Monitor oder ein Smartphone anpasst.

Um komplexere Layouts zu entwickeln, gibt es zudem mittlerweile Hilfsmittel, die das Erstellen von Grid-Systemen für Frontend-Entwickler erheblich erleichtern. Auf der Seite http://gridpak.com findet sich ein Responsive Grid Generator, der nach Eingabe der maximalen Breite in Pixeln, der gewünschten Anzahl der Spalten sowie der Innen- und Außenabstände eine Grundstruktur zum Download anbietet. Diese enthält ein PNG für die grafische Weiterbearbeitung des Grids in Photoshop und darüber hinaus auch beispielhaft aufgesetzte CSS-, LESS- und SASS-Stylesheets. An denen orientiert kann man das HTML-Basisgerüst der eigenen Responsive Website dann sehr einfach und schnell aufbauen. Für die mit Photoshop arbeitenden Designer empfiehlt sich außerdem die hervorragende Extension GuideGuide, die basierend auf der Dokumentengröße oder einer selbstgezogenen Auswahl mit Hilfslinien ein Raster erstellt. Unter www.guideguide.me kann das Tool heruntergeladen werden.

LISTING 1: MEDIA QUERIES

```
/* Beispiel 1
 * Smartphones im Querformat und kleiner
 */
@media (max-width: 480px) { ... }

/* Beispiel 2
 * Smartphones im Querformat bis zu Tablet-PCs im Hochformat
 */
@media (max-width: 767px) { ... }

/* Beispiel 3
 * Tablet-PCs im Querformat und kleine Desktop-PC-Ansichten
 */
@media (min-width: 768px) and (max-width: 979px) { ... }

/* Beispiel 4
 * Große Desktop-PC-Ansicht
 */
@media (min-width: 1200px) { ... }
```

Ein Problem, das beim Arbeiten mit dem Konzept des Responsive Webdesign gelöst werden muss, ist der Umgang mit Bildern. Es gibt einige gut umzusetzende Techniken, um Bilder proportional in der Größe zu verändern. Es empfiehlt sich, hier die einfache und meistangewandte CSS-Eigenschaft *max-width* zu verwenden, die auch Ethan Marcotte in seinem Artikel über Fluid Images erwähnt. Ausführlich nachzulesen sind seine Ausführungen unter www.alistapart.com/articles/fluid-images:

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

Solange keine CSS-Regel die Größe der Bilder überschreibt, wird jedes Bild auf der Webseite prozentual am zur Verfügung stehenden Platz, aber maximal in seiner Originalgröße dargestellt. Bilder werden somit nicht in festen Pixelgrößen definiert, sondern die Prozentangaben bewirken, dass der Browser die Bilder je nach Bedarf und tatsächlich vorhandenem Platz anpasst. Während *max-width* in allen modernen Browsern funktioniert, muss hier für die älteren Versionen des Internet Explorers in einem separaten Stylesheet eine alternative Lösung hinterlegt werden. Mit dieser einfachen Lösung wird auch im Internet Explorer verhindert, dass zu große Bilder ein flexibles Layout zerstören:

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

Über diese Lösung hinaus kann man an dieser Stelle bereits weiterdenken. ▶

Kleine Darstellung für Desktop-PCs (Bild 4)

LINKS ZUM THEMA

Responsive Grid Generator
- http://gridpak.com

Photoshop-Extension GuideGuide
- www.guideguide.me

Artikel über Fluid Images
- www.alistapart.com/articles/fluid-images

Adaptive Images von Matt Wilcox
- http://adaptive-images.com

Javascript-Fallback von Wouter van der Graaf
- http://code.google.com/p/css3-mediaqueries-js

Bootstrap from Twitter
- http://twitter.github.com/bootstrap

Landscape Smartphone und kleiner (Bild 6)

So haben wir zwar gerade die sichtbare Größe von Bildern einfach an die Darstellungsmöglichkeiten verschiedener Endgeräte angepasst, aber nicht die tatsächliche Dateigröße der Bilder verändert. So werden große Bilder der Desktopversion einer Webseite auf dem mobilen Endgerät kleiner angezeigt, die Dateigröße jedoch nicht reduziert

Adaptive Images von Matt Wilcox baut auf dem Ansatz von Filament's Group Responsive Images auf, Bilder in ihrer Auflösung für kleinere Endgeräte zu verkleinern und damit das anfallende Transfervolumen und die Ladegeschwindigkeit der Webseite zu optimieren. Die zugrunde liegende Technik der Filament's Group hat allerdings den Haken, dass alle <*img*>-Tags der Webseite mit einem Marker versehen werden müssen. Dieser beinhaltet dann die Information über das alternativ zu ladende Bild, das auch als kleinere Datei bereits existieren muss.

Matt Wilcox hat mit seiner Entwicklung diese grundlegende Idee weiter optimiert, indem er nondestruktiv mit einer Kopie des Ursprungsbildes arbeitet. Eine Anpassung des <*img*> ist hierbei nicht nötig und kann einfach in jede Seite integriert werden. Es wird lediglich eine Zeile Javascript im *head* der Seite ausgeführt, die ein Session-Cookie mit der Bildschirmgröße des Webseitenbesuchers speichert:

```
<script>document.cookie=
'resolution='+Math.max(screen.width,
screen.height)+';
path=/';
</script>
```

Die *.htaccess rewrite rules* schicken Serveranfragen bezüglich der Bilder an die gesonderte PHP-Datei *adaptive-images.php* weiter. Hier wird das Cookie ausgelesen und die Bildschirmgröße festgestellt. Dieser Wert wird mit den in der Variable *$resolution* hinterlegten Werten abgeglichen.

Das Skript sucht nun in einem definierten Ordner nach einem zur Bildschirmgröße passenden Bild. Wird dieses nicht gefunden, wird es über die GDLib generiert, angezeigt und für den weiteren Gebrauch abgespeichert.

Da sie leicht zu implementieren und Cross-Browser-kompatibel ist, stellt diese Technik gerade für Webseiten mit vielen Bildern eine großartige Möglichkeit dar, Traffic und Ladezeiten zu sparen. Eine ausführliche Dokumentation und der Download sind auf der Seite http://adaptive-images.com zu finden.

Zusätzlich wissenswert ist bei diesem Umgang mit Bildern im Responsive Webdesign die spezielle Eigenschaft einiger Smartphones und Tablets. Denn das iPhone zum Beispiel hat die Eigenschaft, dass sich Webseiten automatisch so

Portrait & Landscape für Tablet / Desktop (Bild 5)

verkleinern, dass sie auf den kleinen Bildschirm passen. Der Nutzer kann einfach ein- und auszoomen. Allerdings bemerkten Entwickler und Nutzer mit dem Aufkommen von Responsive Webdesign, dass auf iPhones und iPads die Bilder immer noch mitskalierten, obwohl sie extra für den kleinen Bildschirm angefertigt waren. Deshalb haben sich Texte und andere Elemente auf der Seite unverhältnismäßig verkleinert.

Zur Lösung dieses Problems gibt es nun ein eigenes Meta-Tag, dessen Einsatzmöglichkeiten und erweiterte Funktionen auch in Apples offizieller Entwicklerdokumentation nachgelesen werden können:

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

Durch das Setzen des Wertes *initial-scale* auf *1* wird das Standardverhalten, die Bildergrößen einer Webseite proportional zu verändern, überschrieben.

Media Queries

Die Basis für eine Responsive Website haben wir mit der Erstellung eines Fluid Layouts und dem dazu passenden Handling der Bildelemente geschaffen. Die Seite wird nun zwar richtig skaliert, aber von der Anordnung immer noch für eine große Desktop-Version dargestellt.

Oft sind Inhaltselemente mit einem *float: left;* versehen oder zum Beispiel Navigationen mit Icons ergänzt. Wie auch immer gestaltet, stellt sich dann die Notwendigkeit, das Design auf ein kleineres Layout anzupassen. Hier kommen die Media Queries ins Spiel. Schon seit CSS 2.1 gibt es in Stylesheets ein gewisses Bewusstsein für Endgeräte durch die Eigenschaft der Media

Types. Wenn man jemals ein separates Stylesheet für die Druckversion einer Webseite angelegt hat, ist einem dieses Konzept vertraut:

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

Mit den neuen CSS3-Spezifikationen des W3C wurden diese Media Types um die *media query* erweitert und enthalten nun unter anderem die Eigenschaften *max-width*, *device-width* und *orientation*, mit denen eine Abfrage spezieller Charakteristiken des Endgeräts möglich ist. Mit dem Aufkommen von Smartphones wurde es beispielsweise populär, ein Stylesheet einzubinden, das spezielle Regeln für diese kleinen Geräte enthält:

```
<link rel="stylesheet" media="screen
and (max-width: 480px)"
href="smartphone.css" />
```

Die obige Query besteht aus zwei Teilen, nämlich dem Media Type *screen*, sowie der Abfrage, ob die horizontale Größe maximal *480px* beträgt. Wenn diese Seite also auf einem Gerät mit einem kleinen Display betrachtet wird, greift das Stylesheet *smartphone.css*.

Grundsätzlich ist die Integration von Regeln mehrerer Endgeräte in ein Stylesheet zu empfehlen. So erspart man sich unnötige Anfragen an den Server, die nur aufgrund verschiedener Stylesheets gestartet werden. Twitters Bootstrap beispielsweise enthält nun sehr gut anwendbare Media Queries, welche die gängigen Endgeräte abfragen. Ein Beispiel zeigt Listing 1.

Die Eigenschaften *min-width* und *max-width* sind selbsterklärend. So bestimmt *max-width* die maximale Browser- oder Bildschirmgröße, die für bestimme Regeln zutrifft – beispielsweise die vorherige Media Query aus unserem Beispiel mit der maximalen Breite von *480px*. Hier greifen bei Überschreitung der Auflösung keine Regeln mehr, die innerhalb dieser Query definiert wurden. Die Eigenschaft *min-width* macht genau das Gegenteil, wie in Beispiel 3 von Listing 1 zu sehen. Alle hier definierten Regeln treffen nicht zu, wenn die Auflösung *768px* unterschritten wird oder *979px* überschritten werden. Ebenfalls mit *min-width* lassen sich eigene Regeln für besonders große Ansichten definieren. Wie in Beispiel 4 von Listing 1 zu sehen, greifen die dort hinterlegten Eigenschaften erst, wenn eine Mindestgröße von *1200px* gegeben ist. Mit diesem Beispiel 3 wird die Möglichkeit erkennbar, mehrere Queries miteinander zu verbinden. Zudem wird hier auch der Vorteil der Zusammenfassung von Queries in einem Stylesheet klar. Wären diese Regel getrennt, würden beispielsweise für das Umschalten der horizontalen und vertikalen Orientierung eines iPads zwei Stylesheets getrennt geladen werden.

Mit einem Javascript-Fallback von Wouter van der Graaf, das unter http://code.google.com/p/css3-mediaqueries-js zu finden ist, kann man durch die Implementierung nur einer Datei Media Queries auch für alte Browser ohne nativen Support verwenden. Mit der Einbindung des Skripts *css3-mediaqueries.js* sind dann auch IE 5+, Firefox 1+ und Safari 2 in der Lage, Media Queries zu parsen und anzuwenden.

Generell ist dennoch zu bedenken, dass Media Queries natürlich nicht die Lösung für jede Aufgabenstellung, aber eine sehr gute Wahl für die Umsetzung einer Responsive Website sind. Hat man in der Entwicklung zusätzlich die Möglichkeit, mit JavaScript zu arbeiten, stehen durch die Kombination beider Technologien noch mehr Türen offen.

Showcase

Nach den Beispielen für technische Herangehensweisen zeigen wir hier noch ein gestalterisch schön und technisch durchdachtes Responsive Webdesign. Sicherlich gibt es viele gelungene Webseiten, die für variable Endgeräte entwickelt worden sind. Exemplarisch sei hier nun aber die Webseite http://css-tricks.com von Chris Coyier gezeigt. Neben der Darstellung für Smartphone und Tablets ist die Seite auch für unterschiedliche Desktop-PC-Bildschirmgrößen angepasst:

- **Große Darstellung für Desktop-PCs:** Die Webseite setzt sich aus einem dreispaltigen Layout mit zwei Sidebars und viel Platz für den Hauptinhalt der Artikel zusammen (Bild 2).
- **Mittlere Darstellung für Desktop-PCs:** Für kleinere Bildschirme mit einer maximalen Breite von bis zu 1200 px verändert sich die Inhaltsarchitektur das erste Mal (Bild 3).
- **Kleine Darstellung für Desktop-PCs:** Für mittlerweile eher kleine Monitore mit einer Auflösung von bis zu 1024 px fallen nun die Datumsblasen weg, und auch das Logo ist jetzt auf den Seitentitel reduziert (Bild 4).
- **Portrait & Landscape für Tablet / Desktop:** Sehr kleine Browserfenster und Tablet-PCs bekommen die nächste Stufe der Gestaltung zu sehen (Bild 5).
- **Landscape Smartphone und kleiner:** Smartphones bis 480 px bekommen noch ein letztes Mal eine leicht veränderte Gestaltung (Bild 6).

Zur eigenen genaueren Erfahrung empfiehlt es sich natürlich, die Seite selbst mit Smartphone, Tablet-PC und Desktop PC zu betrachten. Durch das Vergrößern und Verkleinern des Browserfensters werden aber auch schon viele Effekte sichtbar. **[mb]**

FAZIT

Fluid Grids, Fluid Images und Media Queries sind das Handwerkszeug, um ein Responsive Webdesign technisch umzusetzen.

Dieses Konzept erfordert allerdings auch eine andere Herangehensweise an die Konzeption einer Seite. Anstatt unseren Inhalt in getrennte, gerätespezifische Webseiten aufzuteilen, können wir Media Queries benutzen, um den verschiedenen Anforderungen an eine Webseite gerecht zu werden.

Sicherlich muss man erst einmal lernen, damit umzugehen. Der Aufwand in der Konzeption und Umsetzung ist dann aber geringer als beim Entwickeln mehrerer einzelner Seiten. Der Wartungsaufwand in der Betreuung sinkt dabei ebenso deutlich wie der redaktionelle Aufwand auf Kundenseite.