Outra ref

<http://get.hike.in/>

background-size

<https://github.com/louisremi/background-size-polyfill>



Design Responsivo III – Media Queries e Compatibilidade

Ok. Então você decidiu fazer o seu layout responsivo…  Você não esta sozinho na sua aventura. Este post contém muitas informações e ferramentas úteis para criar imagens fluidas, fazer consultas de mídia, vencer os browsers antigos e testar o resultado final para você desenvolver um layout like a boss.

Nos capítulos anteriores nós discutimos as [vantagens de se trabalhar com design responsivo](http://blog.popupdesign.com.br/design-responsivo-i-o-que-e-e-por-que-usar/)e como criar um [grid flexível](http://blog.popupdesign.com.br/design-responsivo-grids-e-texto/). Agora está na hora de colocar a mão na massa e construir um layout! Por onde começar? Eu diria que por um bom wireframe! Imprimir estes [templates](http://jeremypalford.com/arch-journal/responsive-web-design-sketch-sheets" \o "Responsive web design sketch sheets) e desenhar a mão o esqueleto do seu site em diversas resoluções é uma boa dica para organizar todas as informações. Ok, você já tem um design bacana. É hora de desenvolver!

Imagens Fluidas

O primeiro passo é garantir que as imagens vão se adaptar de acordo com o layout. A solução mais prática é forçar as imagens a se redimensionarem proporcionalmente de acordo com o tamanho de um container. Para fazer isto basta uma linha de código:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3 | img {      max-width: 100%;  } |

Esta técnica também funciona para outros objetos como vídeos:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | img,  embed,  object,  video {      max-width: 100%;  } |

Se você (ainda) dá suporte ao Internet Explorer 6 você deve colocar o seguinte parâmetro em uma folha de estilos especifica para ele:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | img,  embed,  object,  video {      width: 100%;  } |

Parece a mesma coisa, mas não é. A diferença é que no max-width:100% as imagens nunca ultrapassarão o tamanho do container. Já no segundo caso as imagens SEMPRE estarão EXATAMENTE do tamanho do container. Se você for usar esta regra para o IE certifique-se de que suas imagens são do tamanho certo para não “estourar” os pixels, ok?

Outra coisa importante de se ter em mente é que o Windows não renderiza muito bem imagens redimensionadas em browsers antigos (como IE7 ou anterior). O artigo [Fluid Images](http://unstoppablerobotninja.com/entry/fluid-images" \o "Fluid Images) do Unstoppable Robot Ninja ensina a usar o filtro AlphaImageLoader para corrigir o problema. Realmente ninja!

Outra técnica bacana é cortar as laterais da imagem, escondendo ou revelando algumas partes de acordo com o tamanho do browser. Uma maneira simples de fazer isto seria determinar um overflow:hidden para o container da imagem.

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | .uma-div-feliz {     overflow: hidden;  }    .uma-div-feliz img {     display: block;     max-width: auto;  } |

Só lembrando que esta técnica vai cropar (existe este verbo? :)) a imagem indiscriminadamente. Se você quiser ter um controle maior sobre que parte da imagem é cortada o artigo [Hiding and Revealing Portions of Images](http://zomigi.com/blog/hiding-and-revealing-portions-of-images/" \o "Hiding and Revealing Portions of Images) ensina algumas maneiras de fazer isto.

O que são Media Queries?

A explicação rápida: são expressões de CSS utilizadas para mudar o layout em diferentes aparelhos sem mudar o conteúdo. A explicação longa: No CSS2 existia uma função chamada Media Type. Ela servia, como o nome já diz, para reconhecer um determinado tipo de dispositivo. Ao todo eram 10 diferentes Media Types:

* all – todos os dispositivos
* aural – sintetizadores de voz
* braille – leitores de Braille
* embossed – impressoras de Braille
* handheld – dispositivos de mão. Por exemplo: celulares com telas pequenas.
* print – impressoras convencionais
* projection – apresentações de slides
* screen – monitores coloridas
* tty – teleimpressores e terminais
* tv – televisores

Como isto funciona na prática? Vamos supor que você queira determinar estilos específicos para a impressão em papel, por exemplo. Você pode fazer isto basicamente de duas maneiras. Acrescentando um link para uma outra folha de estilos no cabeçalho do seu documento:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1 | <link rel="stylesheet" media="print" href="print.css" /> |

Ou utilizando um CSS unificado com a função @media:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3 | @media print {      /\* estilos \*/  } |

E isto serviu bem. Por um tempo… Com a evolução dos dispositivos alguns aparelhos não se encaixavam em nenhuma destas categorias. Um smartphone moderno tem a tela e resolução muito melhores do que os celulares que se encaixam na categoria handheld… E o que dizer dos recém inventados tablet? Para a nossa salvação, no CSS3 a coisa ficou mais complexa com as tais Media Queries, uma espécie de upgrade dos Media Types. Media Querie, do inglês Consulta de Mídia, é uma expressão lógica que não verifica apenas o tipo do dispositivo, mas também a capacidade dele. É como se o Media Querie fizesse uma pergunta para o browser (sendo que a resposta é sempre verdadeira ou falsa). A vantagem disto é um controle muito maior sobre o layout! Você consegue utilizar estas consultas para checar coisas como a altura e a largura do browser, orientação, resolução, etc.

Parâmetros do Media Queries

Os Media features lembram bastante as propriedades de CSS. Cada uma tem um nome e aceita certos valores. Basicamente é a lista de perguntas disponíveis que você pode fazer para o browser. São elas:

Width

Largura do viewport (janela do browser incluindo a barra de rolagem).  
Valor: medidas de comprimento.  
Aceita prefixo min/max: Sim.

Height

Altura do viewport (janela do browser incluindo a barra de rolagem).  
Valor: medidas de comprimento.  
Aceita prefixo min/max: Sim.

Device-width

Largura da mídia. No caso de uma mídia digital é o tamanho da tela. No caso de impressão é o tamanho da folha.  
Valor: medidas de comprimento.  
Aceita prefixo min/max: Sim.

Device-height

Altura da mídia.  
Valor: medidas de comprimento.  
Aceita prefixo min/max: Sim.

Orientation

Orientação da mídia.  
Valor: portrait (retrato) ou landscape (paisagem)  
Aceita prefixo min/max: Não.

Aspect-ratio

Proporção. Razão entre os valores do ‘width’ e ‘height’. Apenas aplicável a mídias do tipo bitmap.  
Aceita prefixo min/max: Não.

Device-aspect-ratio

Proporção da tela do dispositivo.  
Aceita prefixo min/max: Não.  
Exemplo de aplicação:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3 | @media screen and (device-aspect-ratio: 16/9) {        /\* estilos \*/  } |

Color

Número de bits por cor. Se o valor for zero o dispositivo é monocromático.  
Valor: numérico  
Aceita prefixo min/max: Sim.

Color-index

Número de entradas na tabela de pesquisa de cores do dispositivo de saída. Se o dispositivo não utiliza uma tabela de pesquisa de cor, o valor é igual a zero.  
Valor: numérico  
Aceita prefixo min/max: Sim.

Monochrome

Este recurso de mídia descreve o número de bits por pixel em um buffer de quadros monocromáticos. Se o dispositivo não é um monocromático o valor será 0.  
Valor: numérico  
Aceita prefixo min/max: Sim.

Resolution

Resolução do dispositivo (densidade por pixel). Apenas aplicável a mídias do tipo bitmap.  
Valor: número em DPI ou DCM  
Aceita prefixo min/max: Sim.

Scan

Tipo de formação de imagens especifico para televisores.  
Valor: progressive (progressivo) ou interlace (entrelaçado)  
Aceita prefixo min/max: Não.

Grid

Determina se o dispositivo é baseado em bitmap ou em um grid (exemplo: um terminal tty ou um telefone com apenas uma fonte fixa).  
Valor: Se o dispositivo de saída é baseada em Grid o valor será 1. Caso contrário, o valor será 0. Aceita prefixo min/max: Não.  
Grid neste contexto é um tipo de dispositivo. Não tem nada a ver com o grid do design, ok? :)

Operadores

Através dos operadores not (não), and (e) e only (apenas) você consegue ter um controle muito mais preciso. Você também pode utilizar o sinal de virgula que equivale a um “ou”. Combine isto com media types e você está pronto para dominar o mundo! :)

Resolução de tela, sua nova melhor amiga!

Se você quiser desenvolver um CSS tendo um dispositivo específico em mente basta você saber a resolução de uma tela (altura e largura em pixels). Já se você quer uma gama maiores de aparelhos  é preciso fazer algumas generalizações para determinar quais vão ser os pontos que o seu layout deverá se transformar. Para isto vamos considerar as seguintes larguras:

* ￼320 pixels - Smartphones no modo retrato.
* 480 pixels - Smartphones no modo paisagem.
* 600 pixels - Tablets pequenos. Ex: Amazon Kindle (600×800)
* ￼768 pixels - Tablets maiores em modo retrato. Ex: iPad (768×1024)
* 1024 pixels - Tablets maiores em modo paisagem, monitores antigos.
* ￼1200 pixels - Monitores wide.

Alguns exemplos

Um iPhone em modo retrato, por exemplo, possui 320px de width. Se você desenvolver um CSS para o smartphone da Apple basta utilizar o seguinte Media Querie:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3  4 | /\* Smartphone em modo retrato \*/  @media only screen and (max-width : 320px) {        /\* estilos \*/  } |

É como se você perguntasse para o browser: “Hey, o seu dispositivo é uma tela e a largura máxima é 320 pixels?”. Se a resposta for sim o navegador aplica os estilos. Fácil, né? Já se você quiser determinar os estilos de um iPad independente da orientação:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | /\* iPads (restrato e paisagem) \*/  @media only screen  and (min-device-width : 768px)  and (max-device-width : 1024px) {        /\* estilos \*/  } |

Estas são algumas opções. As possibilidades de combinação são infinitas. Se você quiser mais algumas idéias de uma olhadinha no artigo [Hardboiled CSS3 Media Queries](http://stuffandnonsense.co.uk/blog/about/hardboiled_css3_media_queries/" \o "Hardboiled CSS3 Media Queries) do Stuff and Nonsence.

Um exemplo legal de aplicação é utilizar um container e ir variando a max-width de acordo com a resolução da tela. Assim você garante que o seu conteúdo responda a diferentes tipos de dispositivos. Uma ferramenta legal para testar estes pontos de quebra é o [Responsivepx](http://responsivepx.com/" \o "Responsive Px). Você consegue visualizar o site e ir diminuindo e aumentando a área de corte até perceber em qual ponto começa algum problema… E aí é só usar um media query para botar a ordem na casa. :)

Para mais informações sobre Media Queries leia a [documentação da W3C](http://www.w3.org/TR/2010/CR-css3-mediaqueries-20100727/#media1) sobre o assunto.

OK. Mas isso funciona em TODOS os browsers?

Bem, tecnicamente os navegadores que trabalham legal com Media Queries são  Firefox 3.5+, Opera 9.5+, Safari 3+, Google Chrome 4+ e Internet Explorer 9+.  Segundo [esta pesquisa](http://caniuse.com/css-mediaqueries) cerca de 70% de todos os browsers no mercado aceitam Media Queries. Mas existem algumas maneiras de fazer os outros 30% participarem também da brincadeira…

[](http://johnmartz.com/)creditos: John Martz

Modernizr

O [Modernizr](http://www.modernizr.com/" \o "Modernizr) é uma biblioteca de JavaScript que detecta quais features de HTML5 e CSS3 um browser suporta. E isso, como vocês podem imaginar, pode ser aplicado aos Media Queries. Através de uma função em JS você pode determinar que uma folha de estilos seja carregada apenas se aquele browser aceitar Media Queries. É importante ler com cuidado toda a documentação da biblioteca para entender tudo que é possível fazer. Se você não precisa de tudo o que o Modernizr oferece, no site da biblioteca você pode baixar uma versão customizada escolhendo apenas as funções que deseja utilizar.

Ficou curioso para saber o que o seu browser atual aceita?  Sem problemas! A [suite de testes](http://modernizr.github.com/Modernizr/test/index.html" \o "Modernizr Test) do Modernizr informa tudo o que o seu navegador pode ou não fazer. Mas se bater uma preguiça de testar a aceitação em browser por browser você pode conferir algumas tabelas em alguns sites como [HTML5 Please](http://html5please.com/) ou [HTML5 & CSS3 Support](http://www.findmebyip.com/litmus).

Respond.js

Uma alternativa muito mais simples e leve (apenas 3kb) do que o Modernizzer , o[respond.js](https://github.com/scottjehl/Respond) adiciona suporte a min/max-width para o IE6-8 e outros browsers antigos. Útil se você só precisa destes parâmetros. Para utilizar o respond.js você precisa (além de obviamente instalar o plugin) colocar o comentário condicional /\*/mediaquery\*/ ao final de todas as media queries que você utilizar no seu CSS. Por exemplo:

[?](http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/)

|  |  |
| --- | --- |
| 1  2  3 | @media screen and (max-width: 768px) {    ...  }/\*/mediaquery\*/ |

css3-mediaqueries-js

Outra solução baseada em javascript. O [css3-mediaqueries-js](http://code.google.com/p/css3-mediaqueries-js/)  é mais pesado do que o respond, mas em compensação aceita mais funções e promete fazer IE 5+, Firefox 1+ e o Safari 2 aceitarem todas os features do Media Queries. Você pode conferir mais algumas dicas sobre como degradar suavemente Media Queries neste [post da Smashing Magazine](http://coding.smashingmagazine.com/2011/08/10/techniques-for-gracefully-degrading-media-queries/).

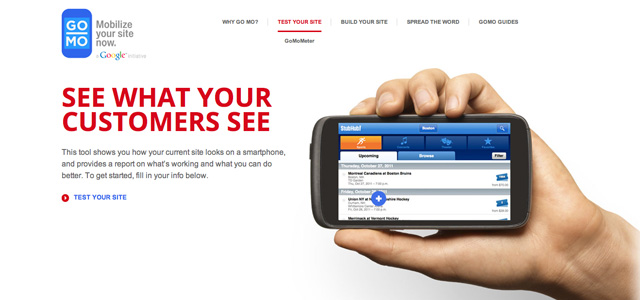
Tudo é relativo…

Qual destas opções utilizar? Depende de qual você se adapta melhor e do que você precisa para o seu projeto. A verdade é que a escolha da melhor solução sempre depende do caso. Na pior das hipóteses faça uma folha de estilos diferente para o IE e pronto. Para um site que exige manutenção constante ter uma folha de estilos a mais pode ser muito trabalhoso, além de representar uma consulta a mais no servidor e mimimi… Ainda assim, na minha opinião, é uma solução válida. Entre todos os visitantes desta semana do BlogUp, por exemplo, apenas 7% utilizaram o Internet Explorer. Não vou nivelar por baixo e entregar uma experiência pior para os outros 93% por conta de um browser lançado a dez anos atrás… Mas esta é uma posição pessoal minha que vai variar muito de acordo com o público alvo. Se você acha que não vale a pena o trabalho extra simplesmente não use design responsivo. Na verdade, fique longe do HTML5 e do CSS3. Só cuidado para não estar desenvolvendo para o passado. Lembra o que aconteceu com o Flash? :)

Olhe para mim ainda falando enquanto temos ciência pra fazer! Hora de testar.

GoMo

Uma ferramenta bem interessante criada pela Google. Através do [GoMo](http://www.howtogomo.com/" \o "GoMo) você pode não apenas visualizar screenshots do seu site em um smartphone como também obter um relatório personalizado com dicas para melhorar a Mobile User Experience.

[](http://www.howtogomo.com/)

Screenfly

Através do [Screenfly](http://quirktools.com/screenfly/) você pode ter um preview do seu site em desktops de 10′ a 24′ polegadas, diversos modelos de tablets, smartphones e até televisores… Outra função legal é poder girar a tela para ver seu site em outra orientação… Uma ferramenta muito útil e completa. <http://quirktools.com/screenfly/>

Browser Add-Ons

É útil também ter instalado no seu navegador alguma ferramenta como o famoso Web Developer Tools – para [Google Chrome](https://chrome.google.com/webstore/detail/bfbameneiokkgbdmiekhjnmfkcnldhhm) ou [Firefox](https://addons.mozilla.org/pt-br/firefox/addon/web-developer/) - ou um bookmarklet como o [este](http://seesparkbox.com/foundry/media_query_bookmarklet) para redimensionar a janela automaticamente em diferentes resoluções.

Vale lembrar que embora ofereçam uma noção muito boa, nenhuma destas ferramentas emuladoras são 100% precisas. Nada substitui o teste no próprio dispositivo! Você pode não ter vários tablets e smartphones, mas vale a pena pedir para os amigos darem uma olhada e conferirem se está tudo certinho…

Isto é tudo pessoal!

Bem, o básico que você precisava saber para lidar com Design Responsivo é isto! Mas fique de olho aqui no blog que vamos sempre postar discas, plugins e best practices sobre este assunto e muitos outros. Espero que vocês tenham curtido. Até a próxima! :)

Tags: [media queries](http://blog.popupdesign.com.br/tag/media-queries/), [responsive design](http://blog.popupdesign.com.br/tag/responsive-design/)