DESIGN RESPONSIVO NA PRÁTICA 2: DO LAYOUT AO HTML

Acompanhe o desenvolvimento de um HTML/CSS responsivo passo-a-passo a partir de um layout fixo, com direito a muitas dicas, comentários e demo.

por Dani Guerrato 28/04/2014

~ 13 min. / 2671 palavras

Há um tempinho atrás eu escrevi aqui no Tableless um artigo chamado Design Responsivo na Prática – Do Rascunho ao Digital. A abordagem na época foi mais voltada para os designers, que estavam carentes de conteúdo bacana sobre o assunto. Mas, no entanto, todavia, algumas pessoas comentaram que gostariam de ver o desenvolvimento do dito cujo do layout criado passo-a-passo ou ao menos uma demo. Como eu não poderia deixar de atender os devs também, é justamente isto que vou fazer hoje. Portanto, pegue seu cafézinho pois hoje é dia de codar! 🙂

Hora da Revisão

Não vou perder tempo aqui falando o que é design responsivo (conjunto de técnicas para melhorar a experiência do usuário independente do dispositivo que ele esteja utilizando), quais são as principais características de um site responsivo (um único conteúdo em uma única URL em um único código) ou por que você deve utilizar (são tantas que este parênteses seria gigante). Mas, como este artigo é mão na massa mesmo, antes de começar tenham certeza de entender os três pilares básicos do RWD (Responsive Web Design, que abrasileiramos carinhosamente para Design Responsivo).

Quais são?

...

...

Spoiler: fundação flexível, imagens adaptáveis e media queries. Bem, se você conseguiu lembrar todos os três de memória e está aí resmungando consigo mesmo ganhou um passe "pule a revisão": vá direto para a prática. Se não conseguiu lembrar, sem problemas. Vamos refrescar a memória!

1. Fundação Flexível

Este é o esqueleto básico do layout. A fundação pode ser construída através de um sistema de grid fluído ou na unha combinando medidas relativas e um pouquinho de matemática. Como gostamos de desafios, vou ensinar por aqui a segunda opção.

(Quase) tudo é relativo

Para que o seu site possa se adaptar a múltiplos dispositivos e toda aquela ladainha que vocês já estão cansadinhos de saber é preciso colocar um pouco de lado os pixels. Note bem: colocar de lado, não jogar fora! Você ainda vai utilizá-los para definir alturas fixas e para definir, se for determinado no layout, um container inicial com uma largura máxima.

Para todo o resto conheça agora seus novos melhores amigos: % e EM.

(Existem também algumas outras medidas responsivas super bacanas como REM, VW, VH e FR. Mas vamos nos concentrar no básico.)

O que diabos é EM?

EM ou quadratim é uma medida relativa que nasceu na tipografia. 1 EM era originalmente correspondente ao tamanho da letra M maiúscula de uma determinada fonte. Isto era útil lá naquela época em que os tipógrafos utilizavam blocos de madeira ou metal para diagramar impressos, afinal, através desta letra eles teriam uma "chave" para todos as outras.

Já em se tratando de CSS, 1 EM é correspondente ao valor de font-size, que, por padrão do browser é em média 16px. E isto é bacana para a gente por que é um tamanho dinâmico. Ou seja, vamos supor que o seu usuário possua alguma deficiência de visão e precise aumentar e diminuir o tamanho do texto... Como pixel é uma medida fixa é impossível fazer isto de maneira proporcional, sem distorções

Conversão de PX para EM

Como fazer contas em base 16 é meio chatinho podemos usar um truque de CSS para facilitar a conversão de pixels para EM.

Isto significa que alteramos o valor da fonte padrão de 16px para 10px. Desta forma, 12px passa a equivaler a 1.2em, 14px fica equivalente a 1.4em, etc.

Lembre-se que 1 EM é relativo ao font-size. Então uma vez que você altere este valor dentro de um article, por exemplo, todos os elementos filhos também serão alterados.

Conversão de PX para %

Para construir a tal da fundação flexível é preciso seguir uma formula estrutural básica: objeto : contexto = resultado.

Não fez o menor sentido? É, para mim também não fez a primeira vez que eu li. Vamos adiantar um pouquinho a prática e pensar no seguinte: temos uma div de largura 1128px. Dentro dela uma coluna com 264px de largura. A coluna é, portantom filha do container. Então vamos pegar o valor em px da coluna (objeto = 264px) e dividir pelo valor em pixel do elemento pai (contexto = 1128px). O resultado deu 0,23404255319149. Agora basta andar duas casas para a esquerda com a virgula e acrescentar um ponto que temos o valor 23.404255319149. E esta é a correspondência da nossa coluna em porcentagem: 23.404255319149%. Este número é realmente grande e a tentação é grande para chamar de 23% e acabar com a história. Mas, se você arredondar, uma hora a soma vai quebrar. Computadores são bem mais exatos que a gente. Eles sabem lidar bem com matemática...

Enfim, design responsivo é repetir esta continha a exaustão, meus caros. Usar um sistema de Grid ou Framework pronto é mais fácil? Muito! Como usar um elevador é mais fácil do que subir vários lances de escadas de um prédio a pé. Mas o dia que o nosso prédio metafórico pegar fogo saber COMO subir as tais das escadas é um conhecimento bem útil para salvar sua pele, certo? O mesmo vale para grids e frameworks. São ferramentas legais, mas tenha certeza que você sabe se virar sem elas. Pode ser que caia no seu colo um projeto onde, por uma incompatibilidade de linguagens ou escolha das outras pessoas envolvidas, você não possa usar um sistema de Grid pronto. Vai por mim. É melhor aprender a fazer na unha primeiro. Depois não diga que eu não avisei...

2. Imagens adaptáveis

Já que vamos trabalhar com porcentagens precisamos garantir que as imagens não vão se distorcer, certo? Então podemos acrescentar o seguinte código CSS.

Isto significa que o tamanho final da imagem no browser nunca vai ultrapassar o tamanho original dela. Mas, para isto funcionar, é necessário sempre envelopar as imagens em um container. Pode ser um figure, uma div, enfim, vai dá sua escolha para aquele contexto.

3. Consulta de mídia

O terceiro e último passo da nossa revisão é consulta de mídia ou media queries. Estes chuchuzinhos do CSS3 servem para identificar qual é o tipo, resolução e densidade do dispositivo e tornar a nossa vida mais fácil.

Para isto basta utilizar a regrinha @media combinada com parâmetros como min-width, max-width, min-height, max-height, aspect ratio, etc. e operadores como and, only e not.

Vamos supor que você queira trabalhar apenas com telas de largura máxima 1024px. O media querie ficaria assim:

A tradução disto de CSS para português seria “tipo de mídia: tela E largura máxima: 1024px”.

Este tema sozinho daria um artigo inteiro. Na verdade já deu! Então, no melhor estilo escolha sua própria aventura:

– Se você tem dúvidas a respeito do funcionamento da consulta de mídia de uma lida em Design Responsivo III – Media Queries e Compatibilidade escrito pela autora que vos fala.

– Se você entendeu tudo desta explicação relâmpago e quer simplesmente ver mais exemplos de media queries dê uma olhadinha em Logic in media queries.

– Se você já entendeu tudo e não precisa de mais exemplos siga em frente. 🙄

Round Bonus – Viewport

Precisamos dizer para todos os dispositivos que a escala inicial do nosso layout é equivalente ao

por conta própria e o design responsivo só vai funcionar no desktop! Para isto vamos manipular a metatag viewport. Cole isto no head do seu documento

Quer saber mais sobre viewport? O artigo Desenvolvimento Responsivo e Viewport é super completo.

Fim da revisão!

O HTML

Hora de colocar a mão no código!

Eu pessoalmente acredito ser mais fácil construir todo o HTML para depois desenvolver o CSS. Mas esta é uma preferência pessoal e não vou ficar aqui cagando regra. Vou manter a estrutura bem simples para nos concentrarmos no CSS. Vamos relembrar o layout que construímos juntos no artigo anterior.

Podemos dividir este HTML em cinco estruturas básicas.

– Uma div com a classe container envolvendo todo o layout;

– Um cabeçalho com a classe header;

– Uma div “hero” com a classe banner;

– Quatro blocos de texto e imagem com a classe coluna;

– Um rodapé com a classe footer.

Até aqui não tem segredo nenhum. É um HTML normalzinho. Você pode fazer o download deste HTML no meu [repositório do GitHub][5] e acompanhar passo-a-passo.

Note que todas as imagens estão dentro de outros elementos (div ou figure).

O CSS [Desktop]

Container – 1128px

Logotipo – 234px x 36px

Menu – maximo 840px

Banner – 1128px x 450px

Caixa de texto – 480px

Colunas de texto – 264px

Fotos – 264px x 218px

Margens – 24px

Rodapé – 1128px

Primeiro vamos centralizar e determinar a largura máxima do container no CSS.

Clearfix

Esta é uma técnica utilizada para conter os floats e evitar que elementos entrem em colapso. Funciona basicamente adicionando um espaço vazio antes e depois dos elementos e dando um "clear" nos dois lados. Tudo isto sem precisar escrever marcação adicional. Como em design responsivo estaremos utilizando muito floats é útil conhecer. Vamos aplicar esta classe ao container do nosso layout para que ele possa conter todas as divs.

Proporção

A seguir vamos garantir que todas as imagens, videos e conteúdos embedados fiquem com a largura máxima de 100% do tamanho original.

Aqui cabe uma observação. Como no nosso mock-up final as imagens em smartphones ocuparão o tamanho de uma coluna eu preferi utilizar um tamanho proporcionalmente maior (500x413px). Assim

Border-box

Outro truque bacana de CSS para design responsivo é o box-sizing border box acompanhado do seletor *. Basicamente esta regra diz que todos os elementos agora levarão em conta apenas a largura e altura determinada, sem somar a este valor a borda e o padding. Ou seja, uma coisa a menos para nos preocuparmos.

O Header

A seguir vamos especificar o tamanho do header. Este é bem fácil: 100%!

Eu dei a ele também uma altura fixa (48px) e especifiquei margens utilizando a medida EM. A minha intenção aqui foi criar uma margem dinâmica que ficasse em um tamanho bacana em relação ao texto em qualquer resolução. Por isto utilizei EM ao invés de px.

Dentro deste header temos um logotipo e um menu. Lembra da formulinha? Objeto : Contexto = Resultado. Então 234 : 1128 = 0,20744680851064. Andando duas casas para direita temos 20,744680851064. Este é o valor em porcentagem do logotipo. Eu gosto de deixar comentado o valor original em pixels para facilitar caso for preciso recalcular os elementos.

Ah, fique atento para substituir a virgula por ponto no CSS.

O menu ficará flutuando a direita e a largura máxima “segura” é 840px.

Coloquei também alguns elementos de estilo dos links e texto. Mas isto você já sabe fazer, certo? 😜

Banner

O banner também é bem padrão, com 100% de largura. Aqui poderiamos ter um slider ou carrossel em JavaScript. Mas vou ficar apenas no HTML por este artigo.

Repare que a imagem está dentro do background do banner. Neste caso ela vai sendo cortada de acordo com a altura e largura que eu especificar para ele. Dentro do banner temos uma caixa que eu carinhosamente dei a classe de “caixa”. Aqui é a mesma ladainha de sempre. Objeto : Contexto =

Novamente vamos pular os estilos puramente estéticos para ganharmos tempo. Mas você pode conferir todos eles na [demo][6].

Colunas

Temos quatro destaques contendo texto e imagens que chamamos de colunas. A margem da direita também foi calculada em porcentagem. Como o último bloco não tem margem coisa nenhuma utilizamos o parâmetro last-child para especificar isto. Lembrando que versões antigas do IE não suportam isto.

Footer

O rodapé do nosso layout é bem simples e ocupa 100% de largura.

CSS [Mobile]

Vamos adaptar o layout aqui para tamanhos menores do que1128px. Mas poderíamos utilizar media queries para criar versões alternativas para televisores, impressão, dispositivos com maior densidade de pixel (como as telas retinas), etc.

Estes aparelhos podem ser divididos em alguns tamanhos médios de largura:

1024px – Tablets em modo paisagem;

768px – Tablets em modo retrato;

600px – eReaders;

480px – Smartphones;

Mas estes valores não passam de chutes calculados. Existem dezenas de resoluções intermediárias. Aqui vai entrar o conceito de break-point, ou, ponto-de-quebra. Funciona mais ou menos assim:

1. Redimensione o seu browser até identificar um problema

2. Concerte o problema.

3. Repita.

Sério. Não tem segredo. Para isto você pode utilizar algum bookmarklet de resolução. Copie o valor para o seu Media Querie.

Por exemplo, em exatos 1128px de largura temos o primeiro problema. Não existe nenhuma margem e o layout fica coladinho na janela do browser. Entra o primeiro Media Querie.

Tradução: em telas de largura máxima 1128px aplicar os seguintes estilos: espaçamento de 2.4em para a esquerda e para a direita.

O próximo problema é que lá por volta do 768px a nossa caixa de banner parece meio apertada demais. Podemos aumentar o tamanho dela.

O mesmo acontece com as colunas... Dentro do mesmo media querie vamos especificar duas colunas com largura de 48% e margem de 2%.

Lembrando que agora a regrinha é diferente. Para cada coluna de número par zeramos a margem através do :nth-child(even).

O logotipo poderia ter uma margem maior por aqui.

O próximo problema é o banner. Por volta de 718px ele poderia receber menos destaque na imagem e mais destaque no texto. Diminuimos a altura dele para 150px, e colocamos a caixa logo abaixo. O resto é correção de margem e perfumaria.

Diminua mais ainda a janela do browser. Eventualmente o menu cairá para duas linhas, certo? Como temos apenas quatro links na navegação ele pode muito bem simplesmente ocupar 100% de largura e zerar a margem das linhas. Vamos também diminuir um pouco o tamanho dos links.

Poderíamos, através de JavaScript e/ou CSS3, criar outras soluções de navegação como um menu retratil, com overlay, etc. Mas como já existem alguns artigos aqui no Tableless sobre o tema vamos seguir adiante.

Neste ponto já temos nosso layout em tablets. Esta é uma screenshot em um iPad 2 modo retrato.

O próximo, e último passo, é criar o layout em smartphones em si. Para isto vamos especificar que todas as colunas também ocupem 100% da tela.

E aqui esta o layout em um smartphone.

Saiba mais

E está finalizada nossa demo. [Baixem o layout][6] e brinquem com o código. Ainda está com dúvida em algum tema específico ou quer se aprofundar em um dos aspectos? Segue uma listinha básica de alguns outros artigos sobre o assunto:

Design Responsivo na Prática: Do rascunho ao digital

[Design Responsivo I – O que é e por que usar][7]

[Design Responsivo II – Grids e Texto][8]

[Design Responsivo III – Media Queries e Compatibilidade][9]

[Desenvolvimento Responsivo e Viewport][10]

[Design Responsivo & Retina Display: desenvolvimento web em alta resolução][11]

[Imagens em alta resolução utilizando SVG][12]

[Grids semânticos com LESS][13]

[Menu Restrátil com CSS e jQuery][14]

[Como testar Design Responsivo][15]

[Imagens Responsivas de Alta Performance][16]

[Responsive Web Design][17]

Espero que tenha sido proveitoso para vocês. Bons estudos, um abraço e até a próxima! 😉

Update:

O Diego Eis fez um [Micro Workshop Online][18] mostrando como implementar um layout responsivo, mostrando o básico sobre Grids fluídos, Imagens (vídeos etc) fluídos, Media Queries, Fonts com REM e algumas outras coisas. Vale a pena ver.

[5]: https://github.com/daniauthors: Dani Guerrato paid: true/design-responsivo-demo "Demo" [6]: https://github.com/daniauthors: Dani Guerrato paid: true/design-responsivo-demo "Demo - Design Responsivo na Prática" [7]: https://blog.popupdesign.com.br/design-responsivo-i-o-que-e-e-por-que-usar/ "https://blog.popupdesign.com.br/design-responsivo-i-o-que-e-e-por-que-usar/" [8]: https://blog.popupdesign.com.br/design-responsivo-grids-e-texto/ "https://blog.popupdesign.com.br/design-responsivo-grids-e-texto/" [9]: https://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/ "https://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/" [10]: https://blog.popupdesign.com.br/desenvolvimento-responsivo-e-viewport/ "Desenvolvimento Responsivo e Viewport" [11]: https://blog.popupdesign.com.br/design-responsivo-e-retina-display-desenvolvimento-web-em-tempos-de-alta-resolucao/ "https://blog.popupdesign.com.br/design-responsivo-e-retina-display-desenvolvimento-web-em-tempos-de-alta-resolucao/" [12]: https://tableless.com.br/imagens-em-alta-resolucao-utilizando-svg/ "Imagens em alta resolução utilizando SVG" [13]: https://tableless.com.br/grids-semanticos-com-less/ "Grids semânticos com LESS" [14]: https://tableless.com.br/menu-retratil-com-css-e-jquery/ "Menu Retrátil com CSS e jQuery" [15]: https://tableless.com.br/como-testar-design-responsivo/ "Como testar design responsivo" [16]: https://tableless.com.br/imagens-responsivas-de-alta-performance/ "Imagens Responsivas de Alta Performance" [17]: https://alistapart.com/article/responsive-web-design/ "Responsive Web Design" [18]: https://www.eventials.com/tableless/live-coding-implementando-um-site-responsivo/

Leia mais aqui no Tableless:

- O Cenário do Web Design Responsivo
- O futuro chegou: O elemento picture
- Não tenho versão mobile, faço ou não faço?
- Design responsivo: foco no ser humano
- Introdução aos Progressive Web Apps

TAMBÉM EM **TABLELESS**

Core do Linux no Windows com WSL2

3 anos atrás • 1 comentário

Windows usará kernel do Linux para rodar software binário do Linux no …

React Hooks

3 anos atrás • 1 comentário

Nova feature que chegou na versão 16.8.0, um dos assuntos mais hype do …

Como Criar um Logo de Sucesso: a …

2 anos atrás • 3 comentários

Neste artigo você aprenderá como o logo da Apple foi criado, quem o inventou …

Cor Cor

3 ano

Artig threa tem

TheHentai.net • um ano atrás

Os links do GitHub sumiram, fiz todo o tutorial e chegou na hora de pegar o código HTML e cadê.... :(

Responder • Compartilhar ›

Henrique Dias • 5 anos atrás

Alguém conhece uma lista com os 4 padrões de dimensões Smartphone tablet laptop e desktop?

1 Responder • Compartilhar ›

Sarah Cristina ↗ Henrique Dias • 4 anos atrás

instala esta extensao no chrome q chama i love adaptative
poe o site e depois vc escolhe as varias dimensoes

2 Responder • Compartilhar ›

Henrique Dias • 5 anos atrás

Show

Responder • Compartilhar ›

Igor Cedro • 5 anos atrás

Entrei na faculdade agora e estou fascinado por webdesing (apesar de não ser o foco no curso) e hoje achei este site e seus artigos, já é o terceiro que leio, já adicionei o "temas para tumblr" na "playlist" e pretendo ler todos sobre o assunto. Parabéns pelo conteúdo incrível e muito obrigado!

Responder • Compartilhar ›

Você vai gostar de ler:

O Cenário do Web Design Responsivo

O futuro chegou: O elemento picture

Design responsivo: foco no ser humano

Introdução aos Progressive Web Apps

Categorias

tecnologia-e-tendências (316)

geral (297)

artigos (290)

técnicas-e-práticas (269)

javascript (264)

código (214)

browsers (206)

html (191)

css (183)

mercado (121)

SOBRE O TABLELESS

CONTATO

ANUNCIE

SEJA UM AUTOR

FAZEMOS CÓDIGO FRONT-END

ACOMPANHE

WEBINARS

CANAL NO MEDIUM

CANAL NO TELEGRAM

COMUNIDADE

FEMUG

MEETUPCSS

PODCAST ZOFE

BRAZILJS

DEVNAESTRADA

FRONT-END BRASIL

Escrito pela e para a comunidade web brasileira. Ajude.
Change privacy settings