

DevWeb

Capítulo 25

Media Queries

No início, só se podia ter acesso à Internet através de terminais específicos e com configurações limitadas. Hoje em dia a coisa está completamente diferente. Tente enumerar em quais tipos de aparelhos e dispositivos podemos ter acesso à rede. Computadores desktop, notebooks, tablets, smartphones, aparelhos de TV, vídeo-games, vestíveis como relógios e óculos, aparelhos com assistentes pessoais embutidos, eletrodomésticos,... A lista é gigante! Precisamos nos preocupar com essa exibição em múltiplos meios.



Você tem todo o direito de usar esse material para seu próprio aprendizado. Professores também podem ter acesso a todo o conteúdo e usá-lo com seus alunos. Porém todos os que usarem esse material - seja para qual for a finalidade - deverão manter a referência ao material original, criado pelo **Prof. Gustavo Guanabara** e disponível no endereço do seu repositório público <https://github.com/gustavoguanabara/>. Este material não poderá ser utilizado em nenhuma hipótese para ser replicada - integral ou parcialmente - por autores/editoras para criar livros ou apostilas, com finalidade de obter ganho financeiro com ele.



O desafio de mostrar conteúdo, não importa a mídia

Dê uma bela olhada na imagem a seguir. Ela mostra como o conteúdo deve ser adaptado para poder aparecer em vários tamanhos e formatos de tela.



Esse sempre foi um grande desafio para todos aqueles que desenvolvem qualquer coisa que será exibida nessas telas, ainda mais agora que o tamanho, formato, resolução e capacidade dessas temas varia tanto.

O primeiro passo para adaptar o conteúdo ao tamanho da tela nós já demos! Quando começamos a estudar CSS, falamos bastante sobre recursos que facilitariam a **responsividade**, adaptando o conteúdo ao tamanho da tela usando valores percentuais e as medidas *vh* e *vw*. Porém, é preciso aprender mais para adaptar ainda mais os conteúdos e é aí que entram as **Media Queries**.

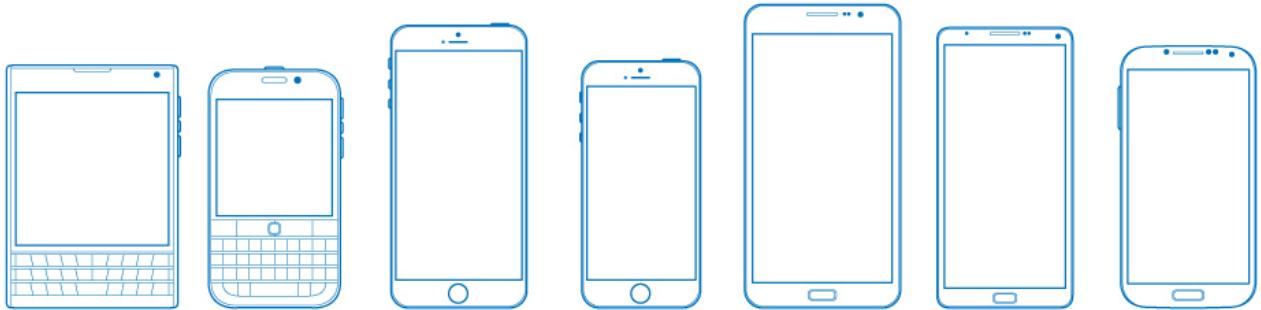


SINGULAR, PLURAL. Basta uma busca rápida pela Internet e você vai encontrar os termos **Media Query** e **Media Queries**. Pois saiba que os dois são corretos, e o primeiro é a representação do termo no singular e o segundo está no plural!

Esse termo não possui uma tradução para o Português, mas de forma simples, **media** significa *meio* ou *mídia* e **query** significa *solicitação* ou *indagação*. Sendo assim, a explicação para o termo seria algo como o seu navegador solicitando um formato especial (CSS) para a exibição em meios diferentes.

Os primeiros estudos sobre essa tecnologia surgiram com os **media types**, que basicamente criavam um formato específico para determinado tipo de mídia (tela, impressora, dispositivos para braille, portáteis handheld, projetores, conteúdo lido, TVs, telas de grade fixa, etc).

O principal problema é que só os **media types** não foram suficientes, pois quando falávamos em telas, tínhamos tamanhos, formatos, suporte máximo de cores, resoluções, proporções e compatibilidades muito diferentes. Um exemplo típico é comparar as telas de alguns modelos de smartphones:



E olha que a imagem acima nem mostrou os aparelhos mais recentes onde a tela ocupa a parte frontal completa do dispositivo e foram adicionados aqueles recortes bizarros para caber as câmeras e sensores.



A ADAPTAÇÃO A TELAS DIFERENTES NÃO É AUTOMÁTICA?

Provavelmente você deve estar se perguntando isso. Pois eu tenho uma PÉSSIMA notícia para te dar. Como desenvolvedores, devemos nos preocupar MUITO com essas resoluções diferentes. Usuários comuns podem pensar que tudo é adaptado como mágica, mas é aí que está a beleza da Tecnologia!

É aí que surgiram as **media queries**. Elas são basicamente as **media types** somadas às **media features**, que são características extras que foram criadas para que possamos personalizar ainda mais nossas folhas de estilo. A seguir, enumero algumas dessas características que podem ser testadas:

- width e height
- device-height e device-width
- aspect-ratio e device-aspect-ratio
- orientation
- resolution
- color e color-index
- grid
- scan

Como usar uma Media Query?

A declaração de um estilo personalizado para um mídia específica através de uma *media query* pode ser feita de duas maneiras: através da declaração de atributo media em arquivos HTML ou usando uma regra @media em arquivos CSS.

Media query definida na HTML

No exemplo que vou apresentar a seguir, note que antes de mais nada, usamos uma configuração que está presente desde o início dos nossos exercícios. É uma linha que está dentro da área `<head>` do nosso documento e faz a configuração básica da viewport, fazendo com que ela ocupe 100% da largura disponível do dispositivo onde

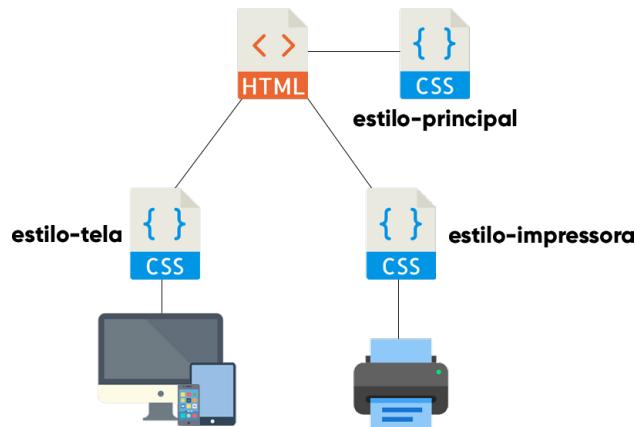
a página esteja sendo exibida, em uma escala 1:1. Já discutimos sobre isso anteriormente quando discutimos os princípios de responsividade.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Testando o Media Query</title>
7   <link rel="stylesheet" href="estilo-principal.css" media="all">
8   <link rel="stylesheet" href="estilo-tela.css" media="screen">
9   <link rel="stylesheet" href="estilo-impressora.css" media="print">
10 </head>
```

Perceba que a **linha 5** do código acima faz a configuração da *viewport* e essa linha é gerada automaticamente pelo VSCode. Se você está usando outro editor de código, por favor, adicione essa linha! Sem ela, nada feito!

Na **linha 7**, adicionamos o primeiro documento de folha de estilo chamado *estilo-principal.css* que vai servir como base para todos (*all*) os dispositivos, independente da característica.

Já nas **linhas 8 e 9**, fazemos a carga dos documentos CSS com configurações específicas para **telas** e **impressoras**, respectivamente. O funcionamento disso será o seguinte:



Note que o arquivo HTML será o mesmo para todos os dispositivos, assim como o estilo principal CSS, que servirá para todos. Porém, na hora de carregar o site em dispositivos com tela ou em impressoras, teremos estilos específicos para cada um. Neste caso, o arquivo *estilo-principal.css* é quem vai fazer as configurações gerais e elas servirão para todo tipo de dispositivo. Já o *estilo-impressora.css* ou *estilo-tela.css* vão ter as configurações extras, que vão adicionar ou modificar características definidas no estilo principal.

O problema da declaração acima é que definimos apenas os *media types*, o que generaliza muito quando dizemos “dispositivos com tela” e diz que todos eles possuem a mesma configuração. Precisamos adicionar configurações de *media features* para aumentar a personalização. Vamos fazer algumas alterações nas linhas para aumentar as possibilidades:

```

3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Testando o Media Query</title>
7   <link rel="stylesheet" href="estilo-geral.css" media="all">
8   <link rel="stylesheet" href="small-screen.css" media="screen and (max-width: 600px)">
9   <link rel="stylesheet" href="phone.css" media="screen and (min-width: 600px) and (max-width: 768px)">
10  <link rel="stylesheet" href="tablet.css" media="screen and (min-width: 768px) and (max-width: 992px)">
11  <link rel="stylesheet" href="pc.css" media="screen and (min-width: 992px) and (max-width: 1200px)">
12  <link rel="stylesheet" href="tv.css" media="screen and (min-width: 1200px)">
13  <link rel="stylesheet" href="print-portrait.css" media="print and (orientation: portrait)">
14  <link rel="stylesheet" href="print-landscape.css" media="print and (orientation: landscape)">
15 </head>

```

Note que agora, além da configuração geral (**linha 7**) temos configurações especiais para telas pequenas (**linha 8**), smartphones (**linha 9**), tablets (**linha 10**), PCs desktop/notebooks (linha 11) e TVs/telas gigantes (**linha 12**) com tudo adaptável. Também adicionamos configurações específicas para impressoras no modo retrato (folha em pé, **linha 13**) e paisagem (folha deitada, **linha 14**).



QUE MEDIDAS SÃO ESSAS? As medidas de tela que usamos acima foram especificadas pela W3Schools como sendo os "*Typical Device Breakpoints*", mas você não é obrigado a seguir esses valores e pode adaptá-los de acordo com medidas mais atualizadas, já que as telas evoluem a cada dia e as resoluções estão melhorando bastante.

W3Schools: https://www.w3schools.com/howto/howto_css_media_query_breakpoints.asp

É claro que você não precisa de todas essas configurações para todos os sites que for construir, mas saiba que é possível criar estilos personalizados para todas as possibilidades que seu projeto necessitar.



IMPORTANTE! Perceba que nas declarações de uma *media query*, em primeiro lugar indicamos o *media type* e depois as *media features* delimitadas por parênteses. Isso é obrigatório. Usamos também os operadores *and*, *or*, *not* ou *only* para manter o relacionamento entre as especificações.

Media query definida nas CSS

Como vimos anteriormente, existem duas maneiras de especificar estilos personalizados para mídias diferentes e já vimos como incluí-las dentro das declarações HTML. A outra maneira é especificando regras @media dentro de uma declaração de folhas de estilo.

Vamos criar uma declaração CSS que suporte vários formatos de tela, tudo junto em um único documento. Analise as linhas a seguir com muito cuidado, já que agora teremos níveis diferentes de seletores.

```

/* Configurações gerais */
body {
    background-color: ■blue;
    color: □white;
}

/* Tablets */
@media screen and (min-width: 768px) and (max-width: 992px) {
    body {
        background-color: ■red;
    }
}

/* Monitores desktop/notebook */
@media screen and (min-width: 992px) and (max-width: 1200px) {
    body {
        background-color: □yellow;
        color: ■red;
    }
}

/* Telas grandes e TVs */
@media screen and (min-width: 1200px) {
    body {
        background-color: ■orange;
        color: □yellow;
    }
}

/* Impressoras */
@media print {
    body {
        background-color: □white;
        color: ■black;
    }
}

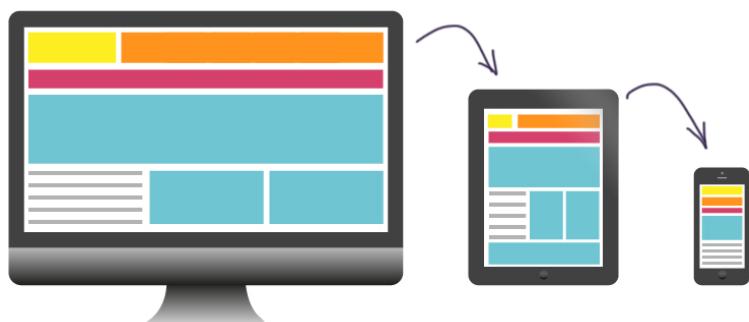
```

Olhando o código acima, perceba que as configurações para Tablets não possuem a configuração de color. Se ela não foi definida na media query personalizada, o que vai valer é o que foi definido nas configurações gerais no início do código.

Outra coisa que provavelmente você deve estar perguntando: "*mas onde estão as configurações especiais para telas pequenas e smartphones?*". Pois é, mas para conseguirmos falar sobre esse aparente "esquecimento", precisamos antes conversar sobre outro assunto.

O conceito de Mobile First

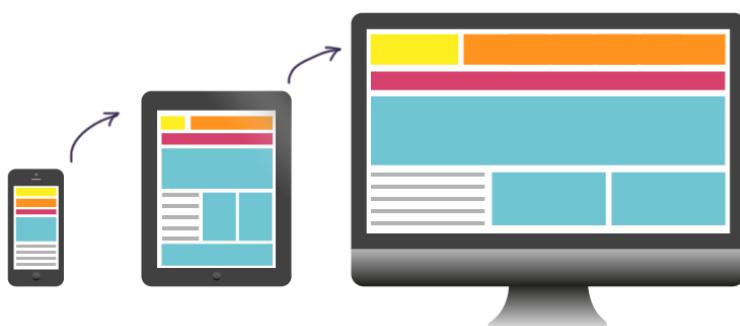
Em uma abordagem mais tradicional, quando pensamos na construção de um site, idealizamos toda a estrutura e os recursos visuais que ele pode ter, criamos o layout completo, implementamos o projeto em um computador e depois nos preocupamos em adaptar toda a sua estrutura para que as telas menores possam exibir o site sem maiores problemas. Isso é o que chamamos de **design responsivo**.



Porém, em uma abordagem um pouco diferente, podemos trilhar exatamente o caminho contrário, pensando inicialmente no layout e funcionalidades da versão móvel e depois adaptando tudo a telas maiores. Esse é o conceito proposto pela linha chamada de **mobile first** (o móvel primeiro), esquema proposto por **Luke Wroblewski** (foto) em 2009. Atualmente chefe de produtos **Google**, LukeW é considerado um dos “pais” dessa abordagem de desenvolvimento.



E esse conceito acaba fazendo muito sentido, pois segundo a GSMA- uma entidade mundial que representa as operadoras móveis - o Brasil é o país da América Latina com mais *smartphones* conectados de toda a América Latina, seguido da Argentina. Isso leva a uma grande probabilidade do seu site ser acessado em um dispositivo portátil conectado a um 4G (ou até 3G) e com tamanho limitado de tela.



E para contribuir ainda mais com essa visão, em 2018 o **Google** anunciou que passaria a valorizar mais na sua indexação os sites que tivessem a preocupação em gerar uma melhor experiência móvel ao usuário. E quem não quer ser valorizado na indexação orgânica da maior ferramenta de buscas do mundo?

Segundo o princípio fundamental da filosofia **mobile first**, pensar primeiro na versão móvel trás algumas vantagens:

- Uma maior divulgação do site, pois como vimos anteriormente, ferramentas de busca vão valorizá-lo.
- Uma melhor experiência do usuário, que vai ter acesso ao conteúdo que foi pensado para que pudesse ser consumido de forma mais confortável e sem confusões.
- Aumento na credibilidade, já que os visitantes vão ter a percepção (ainda que inconsciente) de que quem criou o site se preocupou com a sua experiência.
- Otimização do carregamento, uma vez que sofremos com conexões lentas e dispositivos populares sem tanto poder de processamento.

Dessa maneira, respondendo à dúvida que havia sido levantada anteriormente, o código CSS que criamos não possui uma regra @media específica para dispositivos móveis pois ela simplesmente é a configuração padrão das nossas folhas de estilo. Não esquecemos as configurações para pequenas telas, simplesmente estamos usando o conceito de **mobile first**.

Colocando a mão na massa

Agora que já sabemos para que serve *media query* e entendemos o conceito de *mobile first*, chegou a hora de fazer um projeto que crie versões personalizadas de um site simples para vários tipos de mídia.

Vamos começar separando as imagens e ícones que usaremos no projeto. As imagens que vamos utilizar foram baixadas diretamente do site **Pexels**, que apresentamos durante o **capítulo 11** e os ícones foram baixados no site **Flaticon**. Esses dois serviços disponibilizam conteúdos livres para uso em nossos projetos.

Todos os ícones também foram redimensionados no **Gimp** para que tenham tamanho padrão de **200x200 pixels** e colocados todos na mesma pasta de imagens.



back-pc.jpg



back-phone.jpg



back-print.jpg



back-tablet.jpg



back-tv.jpg



icon-pc.png



icon-phone.png



icon-print.png



icon-tablet.png



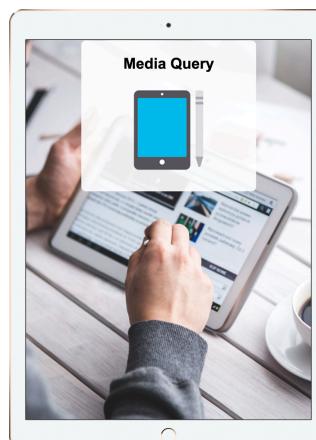
icon-tv.png

Os resultados esperados estão representados nas imagens a seguir. Note que a personalização simples que criamos para cada mídia gera resultados visuais que podem até fazer olhos destreinados perceberem que são sites diferentes, mas se trata da mesma página, apenas com *media query* aplicada.

Versão para smartphone



Versão para tablets



Versão para desktop/ notebooks



Versão para grandes telas/TVs



Versão para impressão



DOWNLOAD DAS IMAGENS: Mesmo te dando o caminho das pedras e indicando os sites **Pexels** e **Flaticon**, estou deixando as imagens utilizadas no nosso repositório aberto do GitHub. Basta acessar a pasta **ex??** e baixar o arquivo **pacote-imagens-cap25.zip** que está lá.

Repositório:

Para iniciar, vamos criar o arquivo `index.html` e colocar no mesmo local a pasta `imagens` com todos os arquivos `JPG` e `PNG` que vamos usar no projeto.

A base do arquivo `HTML` será apenas uma área principal `<main>` com um título e todos os ícones.

```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Testando o Media Query</title>
7      <!-- Aqui vão entrar os estilos e as media queries -->
8  </head>
9  <body>
10     <main>
11         <h1>Media Query</h1>
12         
13         
14         
15         
16         
17     </main>
18 </body>
19 </html>

```

Olhe atentamente nas **linhas de 12 a 16**, onde demos um id para cada ``. Isso será importante para podermos filtrar qual será a imagem a ser exibida em cada mídia.

Agora vamos criar a chamada aos estilos dentro da área `<head>`, exatamente a partir do local indicado na **linha 7** do código acima. Serão ao todo cinco arquivos de folhas de estilo, o primeiro terá as configurações gerais (optamos por aplicar o fundamento de *mobile first* para ele) e os demais para cada uma das mídias restantes.

```

7      <link rel="stylesheet" href="estilos/style.css"
8          media="all">
9      <link rel="stylesheet" href="estilos/tablet.css"
10         media="screen and (min-width: 768px) and (max-width:
11             992px)">
12      <link rel="stylesheet" href="estilos/pc.css"
13          media="screen and (min-width: 992px) and (max-width:
14              1200px)">
15      <link rel="stylesheet" href="estilos/tv.css"
16          media="screen and (min-width: 1200px)">
17      <link rel="stylesheet" href="estilos/print.css"
18          media="only print">

```

Note que na **linha 11**, usamos o operador `only` para deixar claro que essas configurações só serão aplicados para a mídia `print` e não vai afetar de maneira alguma as demais mídias.

Agora vamos segurar a tecla `Ctrl` (`Command` no Mac) e clicar sobre o arquivo `style.css` que está na **linha 7** do código acima. Note que o VSCode também terá que criar a pasta `estilos` que também está indicada na linha.

A estrutura básica do nosso projeto deverá ser essa: o arquivo index e duas pastas, uma para **estilos** (com os arquivos .css) e outra para **imagens**.



O arquivo style.css vai ter um número maior de linhas, já que terá as configurações gerais:

```
1 @charset "UTF-8";
2
3 html {
4     font-family: Arial, Helvetica, sans-serif;
5     font-size: 1em;
6 }
7
8 body {
9     background: black url(../imagens/back-phone.jpg)
10    no-repeat center center;
11    background-size: cover;
12    background-attachment: fixed;
13    color: black;
14 }
15
16 main {
17     background-color: rgba(255, 255, 255, 0.911);
18     text-align: center;
19     width: 350px;
20     height: 300px;
21     margin: auto;
22     padding: 10px;
23     border-radius: 10px;
24 }
25
26 img {
27     display: block;
28     margin: auto;
29 }
30 img#phone { display: block; }
31 img#tablet { display: none; }
32 img#pc { display: none; }
33 img#tv { display: none; }
34 img#print { display: none; }
```

Analisando todas as linhas acima, com certeza você será capaz de compreender todas as declarações dos seletores, pois vimos todas elas detalhadamente em todos os capítulos anteriores. A jogada diferente aqui ficará por conta das **linhas 30 até 34**, onde vamos usar a propriedade display para esconder (valor none) quase todas as imagens, exibindo apenas (valor block) o que vai mostrar o telefone.

O código anterior já vai gerar automaticamente a versão para smartphone. Nossa missão agora é criar as demais, configurando os arquivos para tablet, computadores, TVs e impressoras. O que vamos definir para cada um são apenas as configurações que serão diferentes daquelas que já foram feitas no arquivo geral que apresentamos acima.

Código do arquivo tablet.css

```
1 @charset "UTF-8";
2
3 body {
4     background-image: url(..../imagens/back-tablet.jpg);
5 }
6
7 img#phone { display: none; }
8 img#tablet { display: block; }
9 img#pc { display: none; }
10 img#tv { display: none; }
11 img#print { display: none; }
```

Código do arquivo pc.css

```
1 @charset "UTF-8";
2
3 body {
4     background-image: url(..../imagens/back-pc.jpg);
5 }
6
7 img#phone { display: none; }
8 img#tablet { display: none; }
9 img#pc { display: block; }
10 img#tv { display: none; }
11 img#print { display: none; }
```

Código do arquivo tv.css

```
1 @charset "UTF-8";
2
3 ∵ body {
4     background-image: url(..../imagens/back-tv.jpg);
5 }
6
7 img#phone { display: none; }
8 img#tablet { display: none; }
9 img#pc { display: none; }
10 img#tv { display: block; }
11 img#print { display: none; }
```

Código do arquivo print.css

```
1 @charset "UTF-8";
2
3 body {
4     background-image: url(..../imagens/back-print.jpg);
5 }
6
7 main {
8     border: 1px solid black;
9 }
10
11 img#phone { display: none; }
12 img#tablet { display: none; }
13 img#pc { display: none; }
14 img#tv { display: none; }
15 img#print { display: block; }
```



IMAGEM DE FUNDO SUMIU? Mesmo colocando uma background-image para a versão de impressora ela não será impressa! Não se preocupe, isso é um padrão para a impressão não ficar visualmente poluída. Se por acaso alguma outra imagem não abrir, verifique com toda atenção a digitação dos comandos e use o *autocomplete* do **VSCode** sempre que possível.

Viu como é fácil criar um site que se adapte a múltiplos tipos de mídia? É claro que você não precisa criar todas as possibilidades em cada projeto, mas isso é totalmente possível.

Com o tempo e muita prática depois, você vai ser capaz de montar o layout de sites já pensando em exibi-los em múltiplos dispositivos.



E paramos por aqui?

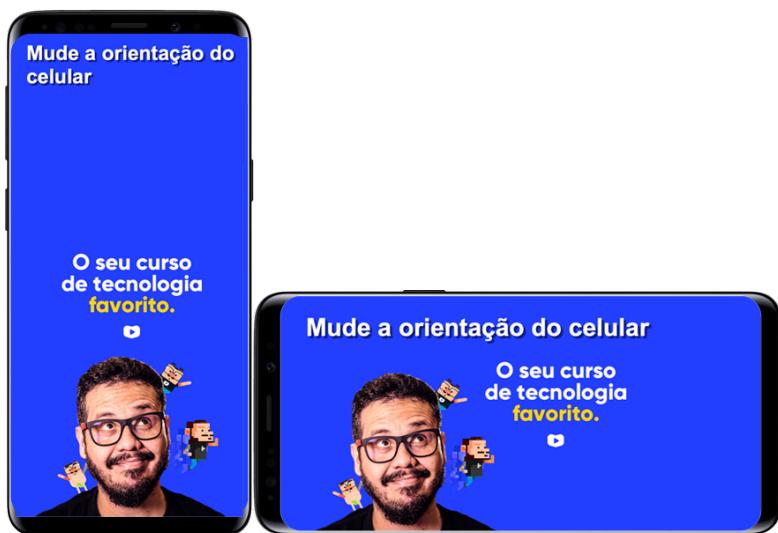
Se você acha que seus estudos sobre *media query* termina por aqui, aí é que você se engana! O objetivo é criar três projetos que vão testar nossos conhecimentos em múltiplas mídias e adaptar os conteúdos a cada uma delas.



ONDE ESSES PROJETOS ESTARÃO? O código para esses projetos serão apresentados em forma de vídeos, já que colocar tudo em formato impresso ficaria muito extenso. Vamos aproveitar as múltiplas mídias por aqui também. A essência de *media query* está explicada aqui, os projetos serão desenvolvidos em vídeos.

Projeto orientação

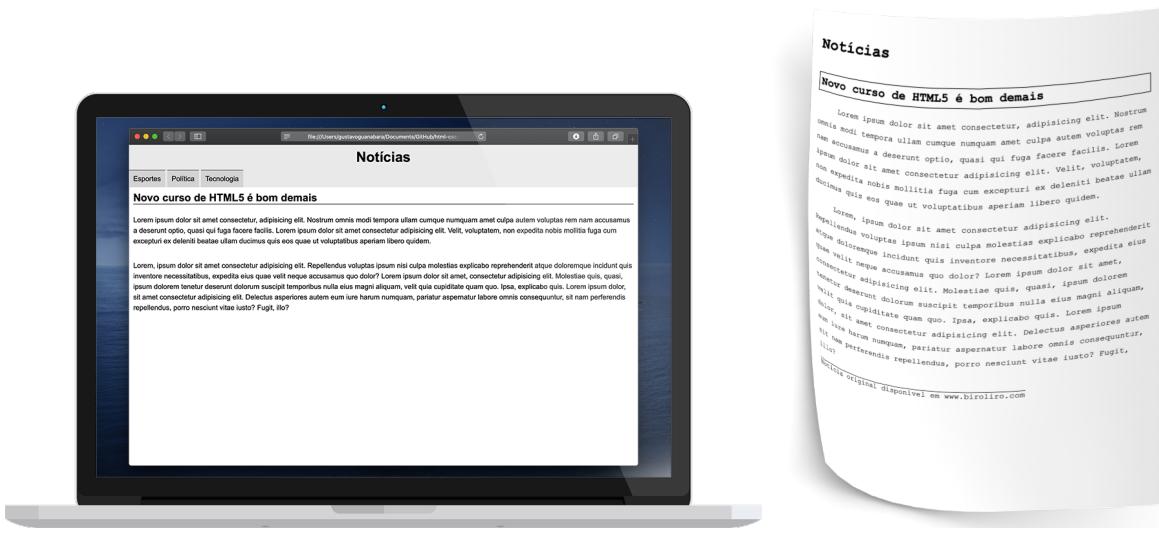
O primeiro projeto é o mais simples de todos e vai criar um site que se adapta ao formato do dispositivo, principalmente os móveis. Estando em pé, a imagem se adapta à parte de baixo do aparelho e a mensagem aparece escrita em cima da figura. Basta deitá-lo e a imagem ficará no canto esquerdo e a mensagem aparecerá na direita.



Nas imagens acima, apresento o resultado esperado. Vamos ver como isso é possível de uma maneira extremamente simples.

Projeto site no papel

O segundo projeto vai mostrar as possibilidades em gerar uma versão especial para impressoras. Provavelmente você já viu isso acontecer em alguns sites, onde a exibição em tela apresenta o site completo, mas na hora de imprimir só será enviado o artigo principal e alguns poucos componentes extras.



Na versão para desktop, o site usa cores, tem um menu superior e o texto em um formato específico para leitura em telas. Já na versão impressa, o conteúdo deixa de exibir cores, perde o menu, ganha bordas, uma apresentação mais coerente com a leitura em papel e um rodapé com o link para acesso futuro.

Projeto menu responsivo

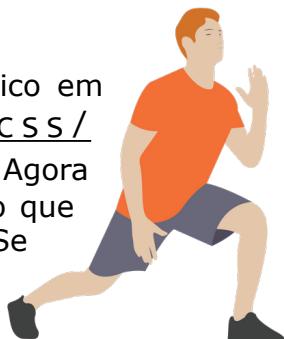
O terceiro e último projeto desse capítulo vai criar um menu adaptável a cada tipo de mídia. Olhe atentamente para as imagens a seguir:



Quando o site é apresentado em uma tela maior, o menu superior é apresentado na forma de pequenos botões organizados um ao lado do outro. Porém, em uma tela mais estreita, o menu muda de forma e dá lugar ao clássico “botão hambúrguer” que pode ser clicado e dá acesso à todas as opções disponíveis no menu. Um resultado muito mais agradável para quem estiver em dispositivos com telas menores.

Hora de exercitar

Chegou a hora de acessar o endereço do nosso repositório público em <https://gustavoguanabara.github.io/html-css/desafios/> e executar os **exercícios ???** no seu computador. Agora tente atingir esse mesmo resultado em casa, sem copiar o código que eu criei. Nesse momento, a prática é algo que você mais precisa. Se por acaso ficar difícil, pode acessar o repositório público de HTML e CSS e dar uma olhada nos comandos, mas **EVITE COPIAR**.



Tenho desafios pra você!

Lá no repositório, além do material em PDF e dos códigos dos exercícios 100% disponíveis, também disponibilizamos alguns **desafios** que devem ser resolvidos. Esses desafios não incluem o código original e você deve tentar chegar à resposta sem copiar nenhum código.

Com todo o conteúdo que vimos até essa aula, você já pode resolver o **desafio ???**. Acesse o repositório público, abra a área do curso de HTML+CSS e clique no link de acesso aos desafios. Manda ver! Só não fica pedindo a resposta! Você consegue resolver isso sozinho(a)!



Repositório em: <https://gustavoguanabara.github.io>

Eu já falei sobre isso no YouTube?

Eu sei que às vezes as pessoas gostam mais de assistir vídeos do que ler livros, e é por isso que eu lanço há anos materiais no canal Curso em Vídeo no YouTube. O link que vou compartilhar contigo tem o conteúdo explicado como você leu aqui, só que de forma mais ilustrada. Reserve um tempo dos seus estudos para assistir esse vídeo todo.



Curso em Vídeo: https://www.youtube.com/playlist?list=PLHz_AreHm4d1AnJ_jJtV29RFxnPHDuk9o