

Aplicações para Internet

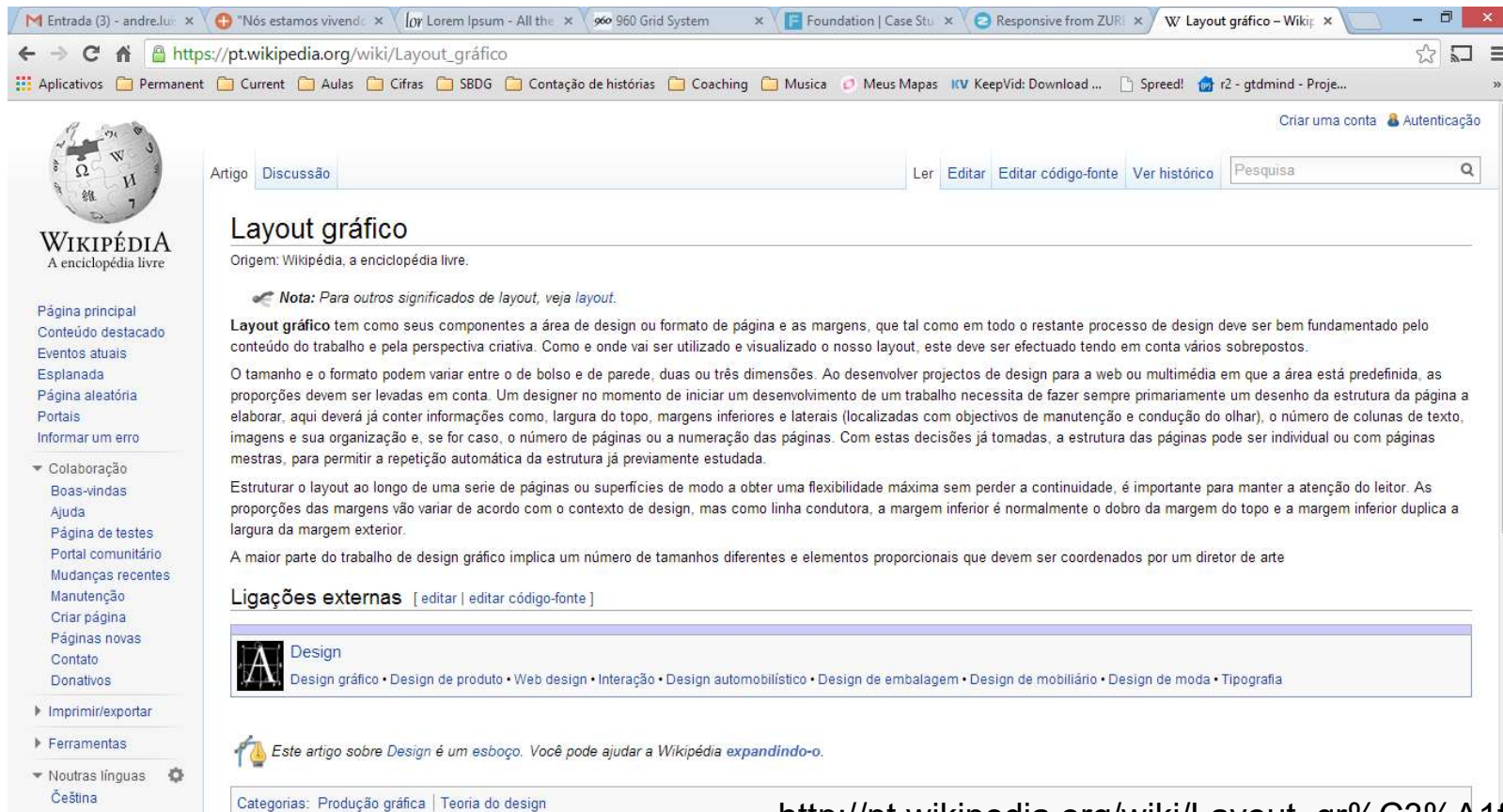
Conteúdo:

- *Layout com Float*
- *Layout Líquido e Congelado*
- *Frameworks para Layout em Grid*
- *Layout responsivo*
- *@media*

Desenvolvido por:
Alcides Teixeira
Ana Paula
Cristiane Camilo
Manuel Ledón

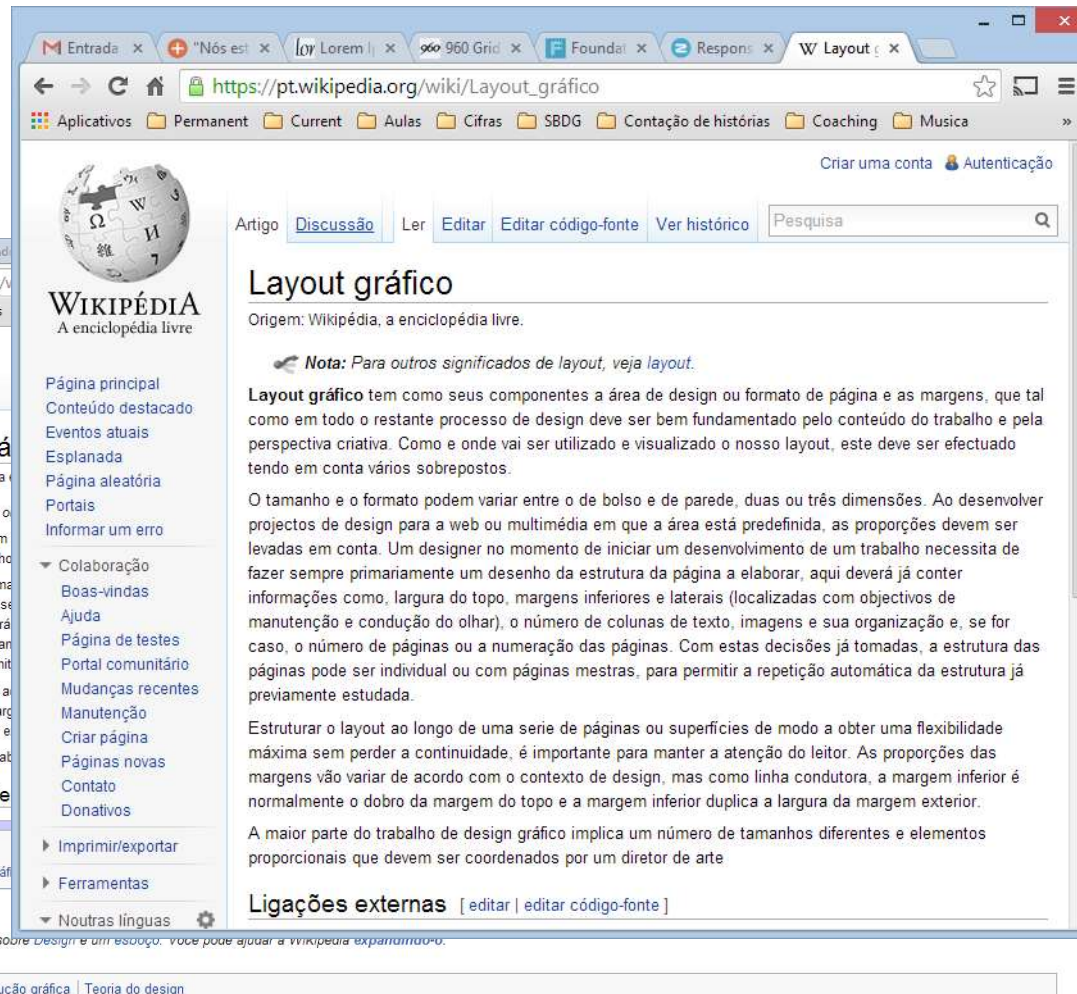
Layout Líquido

- Trata-se de uma organização de conteúdo que se adapta ou se molda ao tamanho da janela



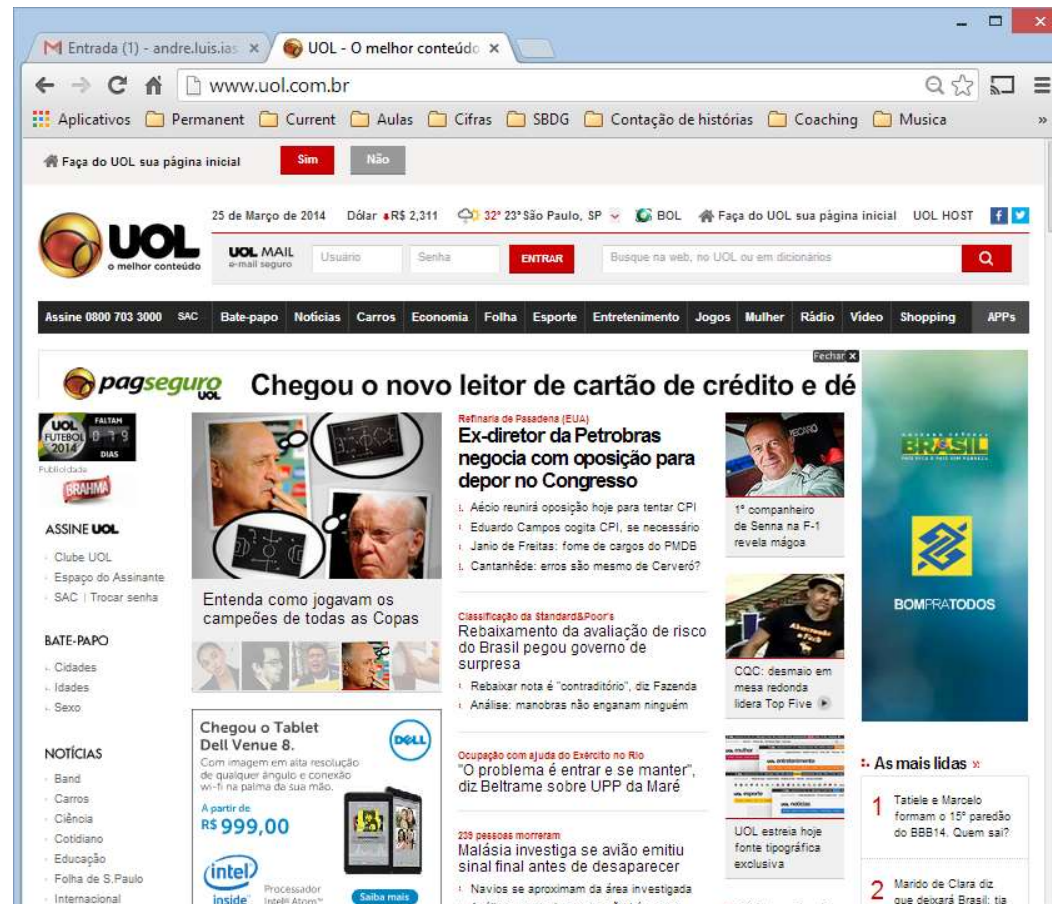
http://pt.wikipedia.org/wiki/Layout_gr%C3%A1fico

Layout Líquido



Layout Congelado

Trata-se de uma organização de conteúdo que permanece a mesma, mesmo que o tamanho da janela mude.



<http://www.uol.com.br/>

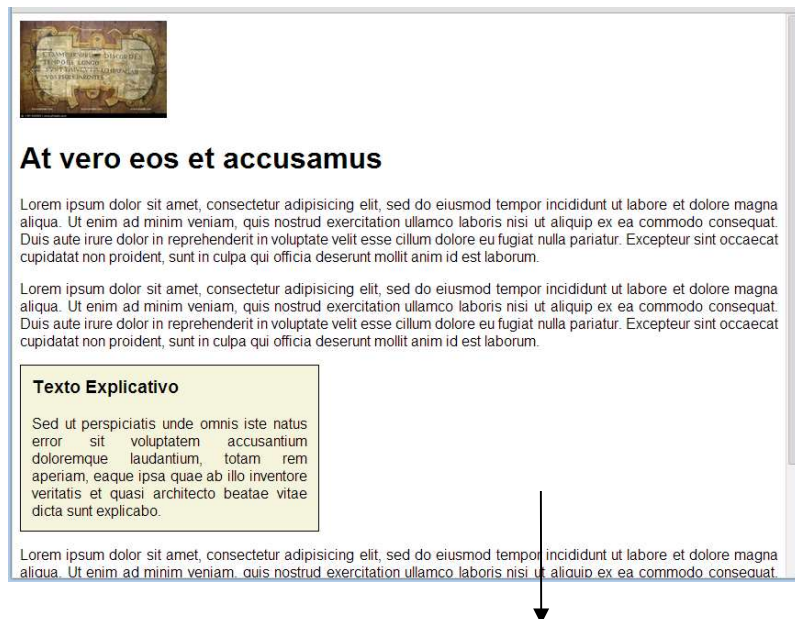
Layout Congelado



Mesmo a janela aumentando o conteúdo permanece ocupando o mesmo espaço

Flutuação – float

- A propriedade **float** foi criada para fazer elementos flutuarem à direita ou à esquerda de um conteúdo




At vero eos et accusamus

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Texto Explicativo

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo.

sem usar float



float: left

float: right

At vero eos et accusamus

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Texto Explicativo

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo.

Layout (completamente) líquido com **float**

- Todos os blocos (divs) tem seu tamanho definidos com porcentagem.
- Margens e “Paddings” devem ser definidos com porcentagens.
- Todos os blocos flutam para a esquerda.
- O bloco que inicia uma nova linha deve ter a propriedade **clear: left;**
- **Problema:**
 - não é desejável que as margens, os ‘paddings’ sejam de tamanho variável;
 - as bordas somem.

Layout parcialmente líquido com **float**

- Todos os blocos (divs) tem seu tamanho definidos em pixels, menos o bloco central.
- Margens, “Paddings” e bordas devem ser definidos em pixels.
- Há blocos que flutam para a esquerda e outros para direita
- O bloco que inicia uma nova linha deve ter a propriedade **clear: both;**

layout_com_float-2.html

- **Problemas:**

- Os blocos à direita tem que vir antes do bloco central no HTML.
- Definir a margem do bloco central pode ser confuso.

Layout congelado com **float**

- Todos os blocos de layout devem estar dentro de um bloco que os contém, comumente chamado “wrapper” ou “container”
- Este bloco deve ter uma largura (**width**) determinada e ser centralizado. Para isso se define uma margem automática, o que faz o sistema colocar a mesma margem do lado direito e esquerdo: **margin: 0 auto;**

Problemas de Layout usando **float**

- Dependência da ordem dos blocos no HTML (quando se usa float: right; a ordem fica estranha)
- Margem e borda de tamanho variável ou de cálculo complicado
- Fica no controle do programador manter as larguras das colunas coerentes em linhas diferentes.
- A altura dos blocos em uma mesma linha ou devem ter tamanho fixado (o que pode ser ruim se o conteúdo for alterável) ou ficam de tamanhos diferentes.

Frameworks

- Um framework é um conjunto de código para desenvolvimento de aplicações. Ele fornece uma base sobre a qual os desenvolvedores podem construir seus códigos.
- Simplificam o processo de desenvolvimento, já que os programadores não precisam reinventar a roda cada vez que forem desenvolver uma nova aplicação.
- Existem para diferentes tipos de linguagens e aplicações
- Veja alguns frameworks no link abaixo:
 - <http://usablica.github.io/front-end-frameworks/compare.html>



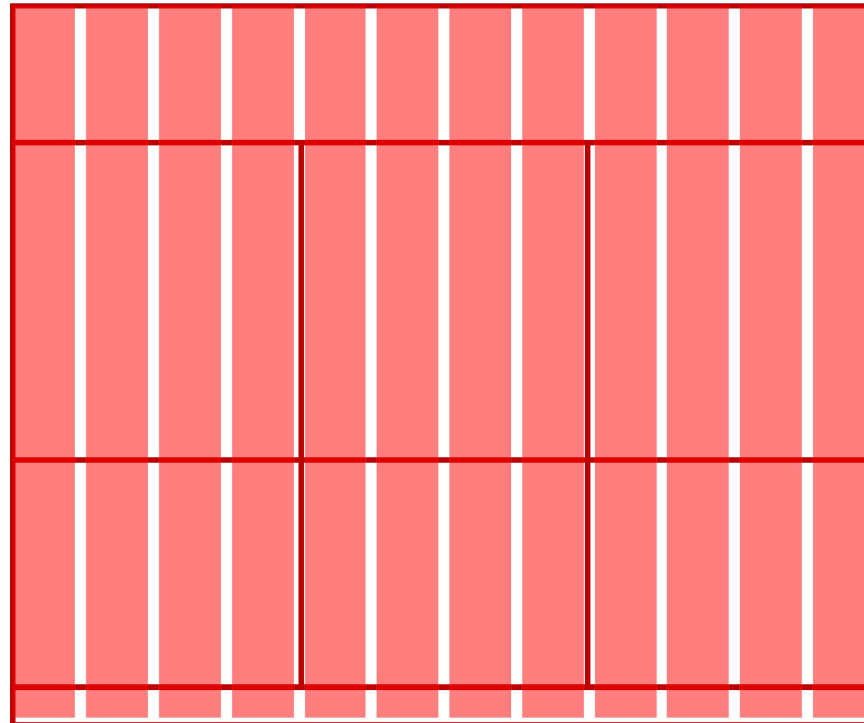
Frameworks de Grid

Framework é um conjunto de códigos (ou biblioteca) que estendem uma linguagem, ou seja, que ‘criam’ novos termos com novas funcionalidades

Os **Frameworks de Grid** normalmente são criados em um arquivo CSS. Este arquivo deve ser anexado à página onde o Framework deve ser usado.

Na sua grande maioria, eles oferecem classes preparadas para criar o *wrapper* (que pode ser de tamanho fixo ou não) e blocos cuja largura ocupa uma certa quantidade de ‘colunas’ do grid

Framework 960.css



O usual é que hajam 12 ou 16 colunas

Alguns Frameworks de Grids

- <http://960.gs/>
- <http://foundation.zurb.com/>
- <http://www.responsivegridsystem.com/>
- <http://gumbyframework.com/>
- <http://semantic.gs/>
- <http://unsemantic.com/>

Usando o 960.gs (parte 1)

- Baixe o framework (use o link do slide anterior)
- Crie a página HTML com **divs** para o layout dentro de uma **div wrapper** e dispostos na ordem correta, mas sem nenhuma informação de layout no CSS
- Coloque o arquivo 960.css na pasta e faça o link para ele no arquivo HTML:

<link rel="stylesheet" href="960.css" />

- No wrapper, adicione a classe **container_12**

Usando o 960.gs (parte 2)

- Em cada **div**, adicione uma classe de acordo com o espaço que o bloco irá ocupar no layout:

Classes:

•grid_12

•grid_1

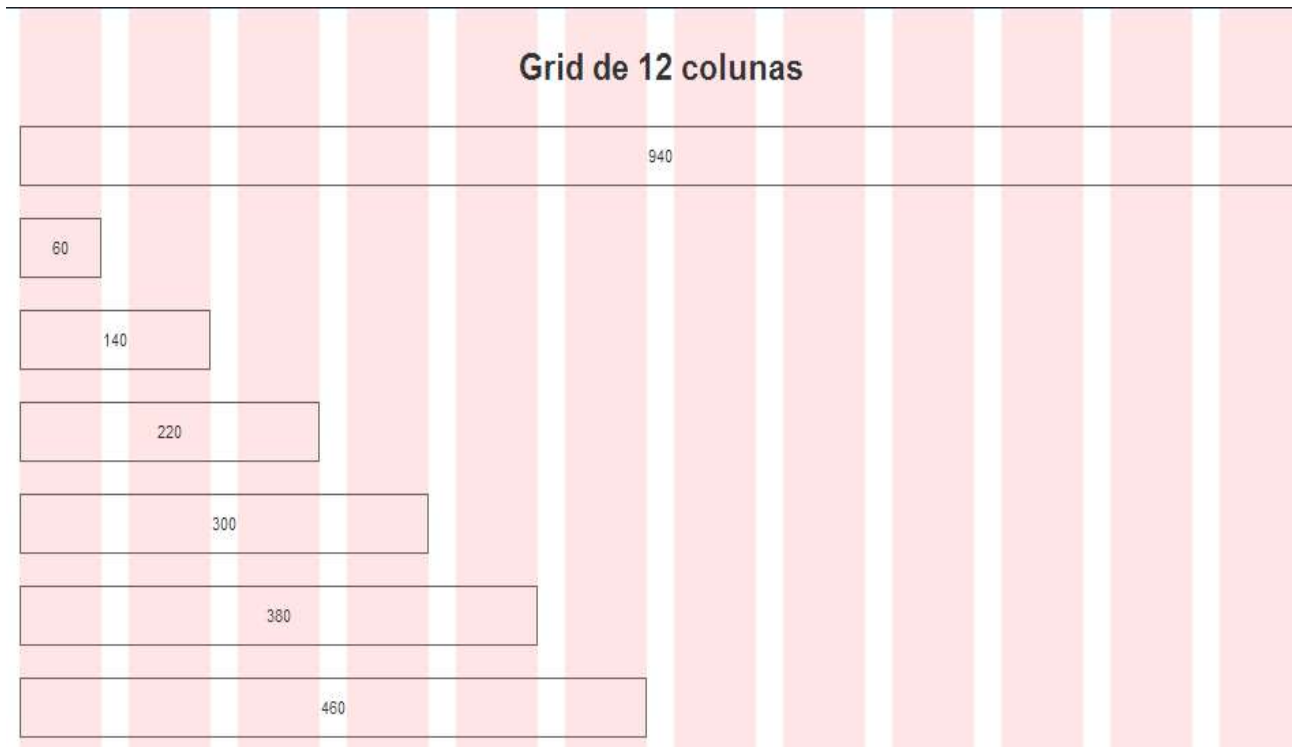
•grid_2

•grid_3

•grid_4

•grid_5

•grid_6



Importante!: Assegure-se de em cada linha ocupar exatamente doze colunas

layout_960.html

Usando o 960.gs (parte 3)

- Se, nas linhas do grid, o conteúdo fizer com que as alturas dos blocos fiquem diferentes, é necessário criar uma separação (para que uma linha não invada outra), colocando no HTML uma **div** sem conteúdo entre as linhas do grid. Essa **div** deve ser da classe “clear”:

```
<div class="clear"></div>
```


Frameworks – prós e contras

Prós:

- Tamanhos padronizados.
- Ordem natural dos blocos mantida.
- Margens incluídas.

Contras:

Intervenção no HTML

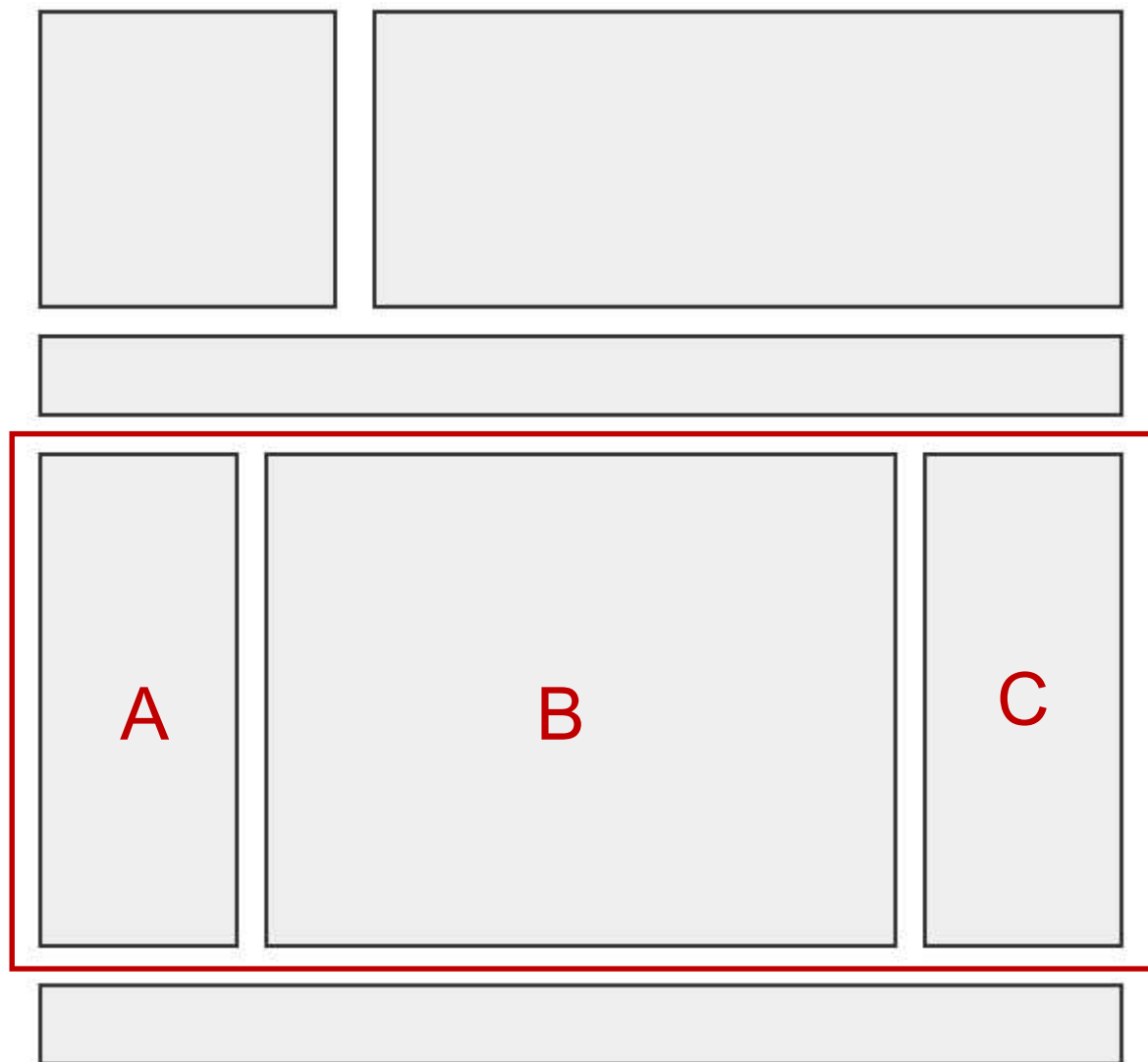
Alturas dos blocos têm que ser pré-fixadas ou...

Para controlar o isolamento de cada linha precisa de uma div para separação e as alturas não são normalizadas.

Layout com FlexBox (CSS 3)

- FlexBox é uma especificação que estende o CSS, criada originalmente pela Mozilla e incorporada na versão 3.0 da linguagem.
- Em termos da construção de um Grid, o FlexBox resolve a organização de uma linha, desde que seus blocos estejam envolvidos em uma div para este grupo.

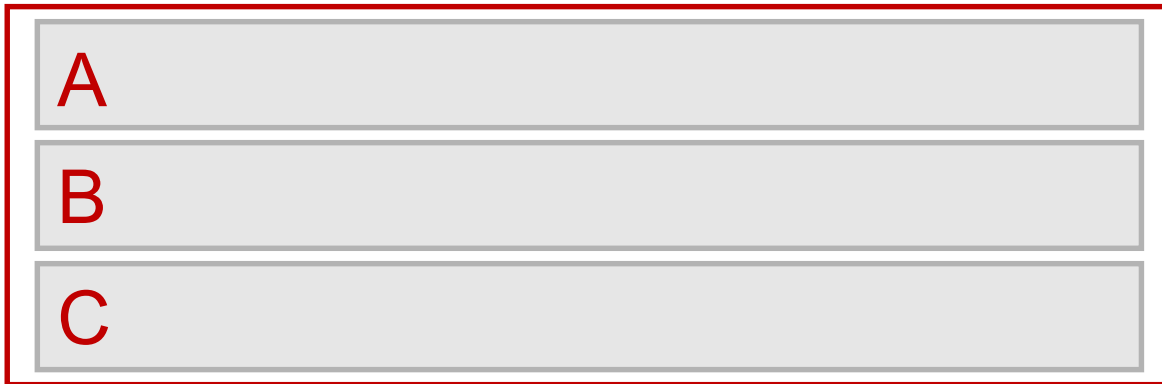
Layout com FlexBox - construindo uma linha - 1



```
<div id="linha">  
  <div id="A">A</div>  
  <div id="B">B</div>  
  <div id="C">C</div>  
</div>
```

Layout com FlexBox - construindo uma linha - 2

Normalmente...



```
<div id="linha">  
  <div id="A">A</div>  
  <div id="B">B</div>  
  <div id="C">C</div>  
</div>
```


Layout com FlexBox - construindo uma linha - 3

CSS com FlexBox:

```
#linha {  
  display: -webkit-flex;  
}
```



```
<div id="linha">  
  <div id="A">A</div>  
  <div id="B">B</div>  
  <div id="C">C</div>  
</div>
```

Layout com FlexBox - construindo uma linha - 4

CSS com FlexBox:

```
#linha {  
  display: -webkit-flex;  
  -webkit-align-items: stretch;  
  -height: 200px;  
}
```



```
<div id="linha">  
  <div id="A">A</div>  
  <div id="B">B</div>  
  <div id="C">C</div>  
</div>
```

Layout com FlexBox - construindo uma linha - 5

CSS com FlexBox:

```
#linha {  
  display: -webkit-flex;  
  -webkit-align-items: stretch;  
  -height: 200px;  
}
```

```
#A, #C {  
  min-width: 200px;  
}  
  
#B {  
  -webkit-flex-grow: 1;  
}
```

```
<div id="linha">  
  <div id="A">A</div>  
  <div id="B">B</div>  
  <div id="C">C</div>  
</div>
```



Ou...

Layout com FlexBox - construindo uma linha - 6

CSS com FlexBox:

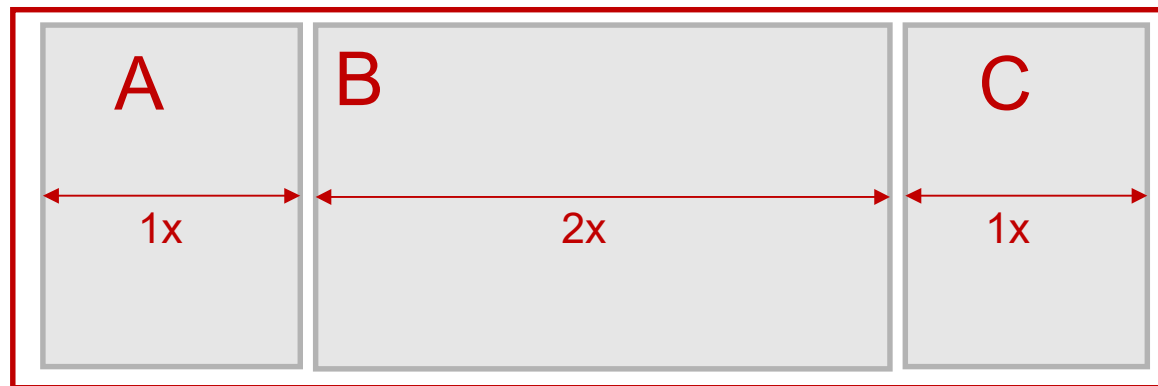
```
#linha {  
  display: -webkit-flex;  
  -webkit-align-items: stretch;  
  -height: 200px;  
}
```

```
#A, #C { -webkit-flex-grow : 1; }  
  
#B      { -webkit-flex-grow: 2; }  
  
#A, #B, #C { -webkit-flex-basis : 0; }
```

flexBoxLayout.html

```
<div id="linha">  
  <div id="A">A</div>  
  <div id="B">B</div>  
  <div id="C">C</div>  
</div>
```

Garante que o conteúdo de um bloco não o obrigue a crescer mais que a proporção estabelecida

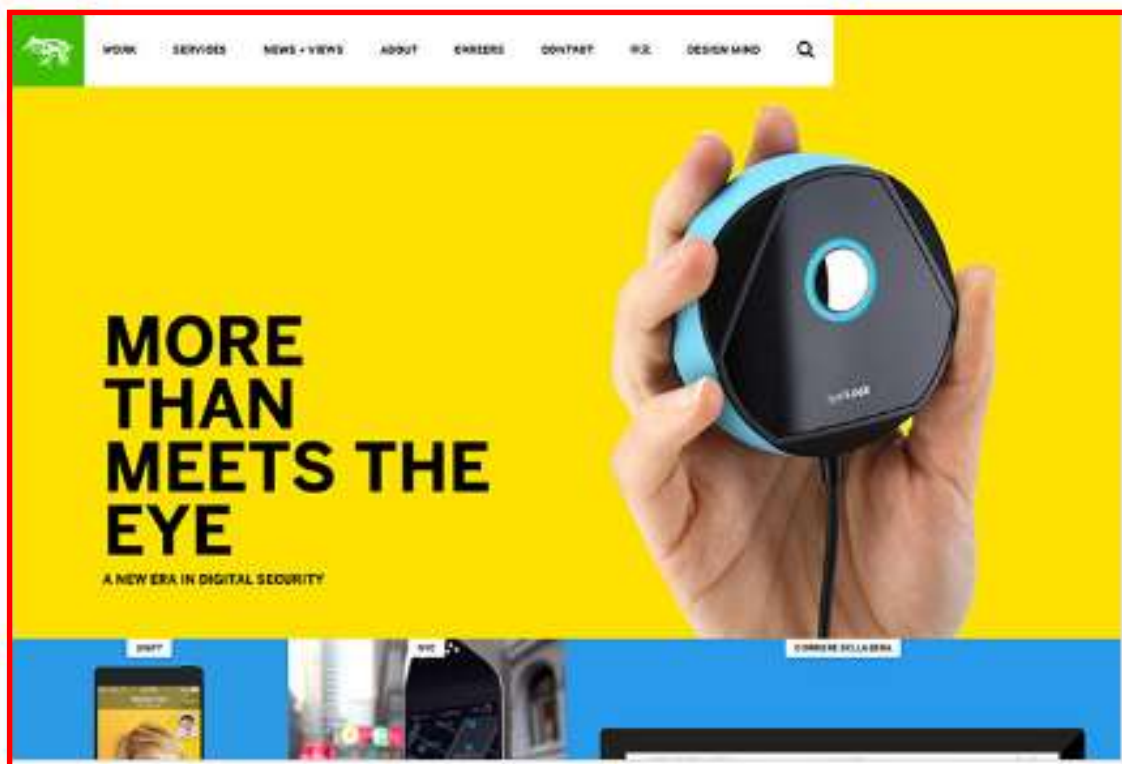


Layout com FlexBox – últimas observações

- Existem muito mais comandos que os apresentados aqui, que permitem trabalhar com coluna em vez de linha, alterar a ordem e fazer todo tipo de alinhamento dos blocos internos
- Tudo que foi escrito nos exemplos com “-webkit-” na frente, deve ser escrito também com “-moz-”, com “-ms” para que rode em todos os navegadores que tem implementação
- Para saber sobre o suporte dos navegadores ao FlexBox, veja o site <http://caniuse.com/flexbox>
- Mais ajuda:
 - <http://tableless.com.br/flexbox-organizando-seu-layout/>
 - <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

Layout adaptados para Contextos específicos

Mesmo HTML – Apresentações diferentes



Desktop



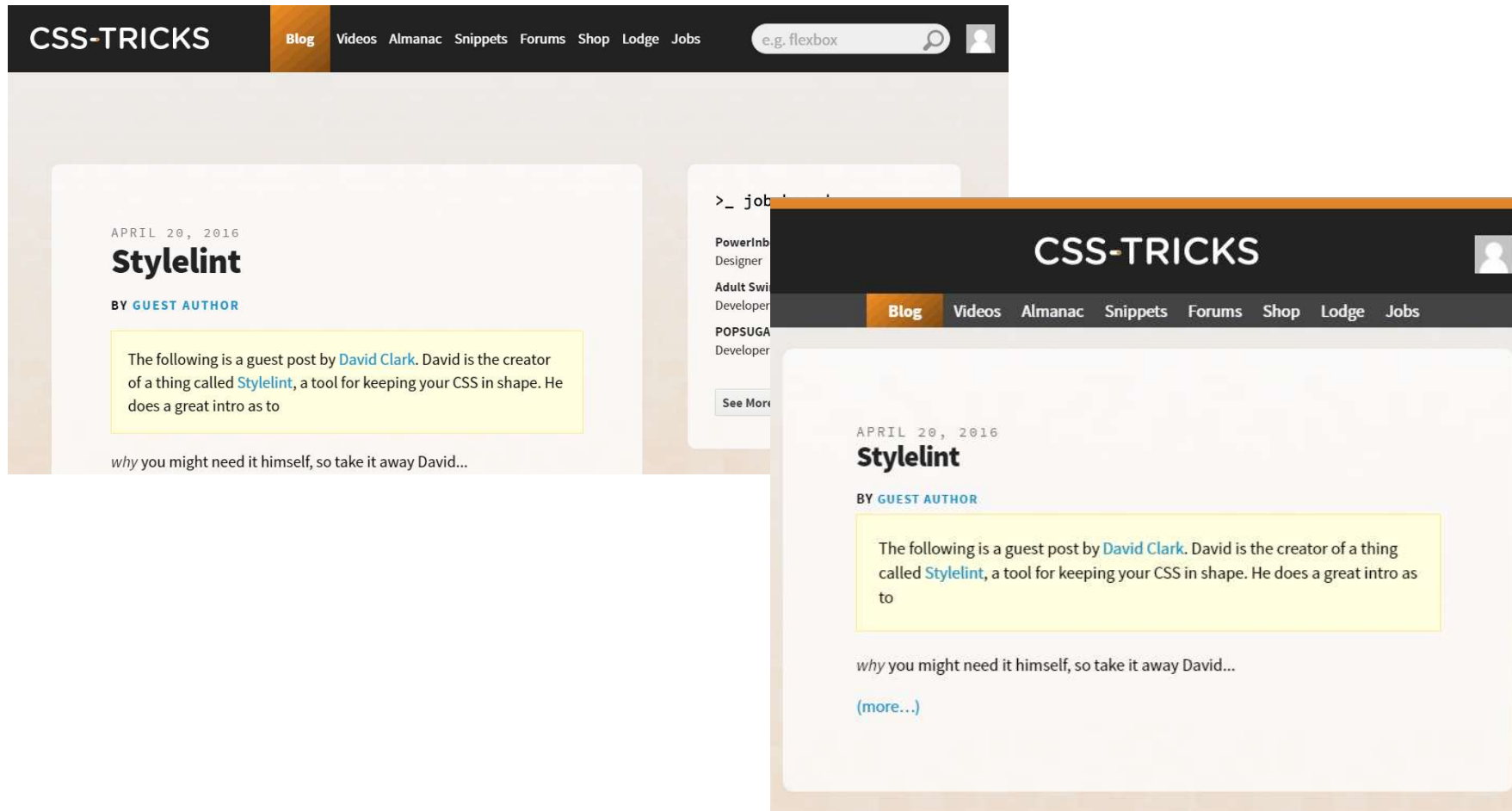
Celular

Layout responsivo

- Acessibilidade de sites em vários tipos de telas/dispositivos.
- Devemos pensar na mobilidade (celular, tablets), além de outras telas como em geladeiras, máquinas de lavar, TVs etc.

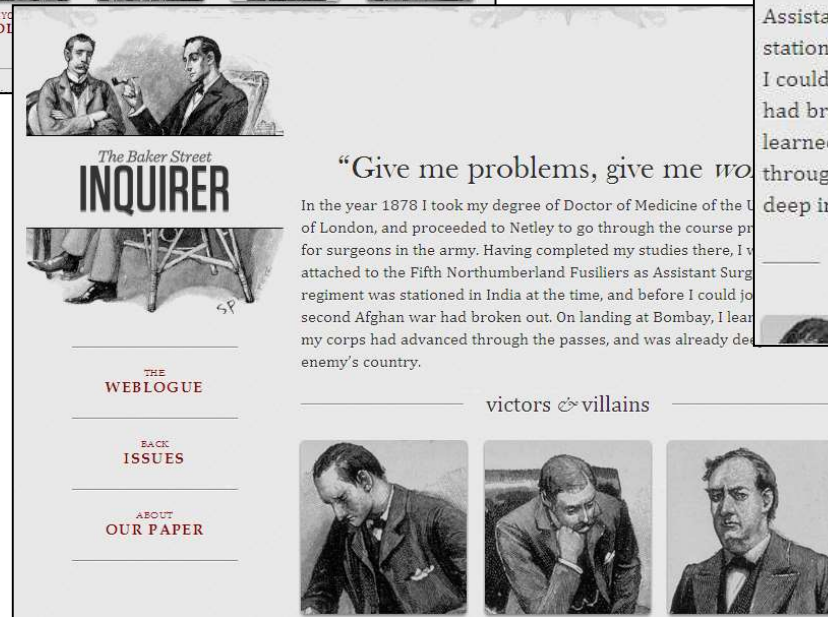
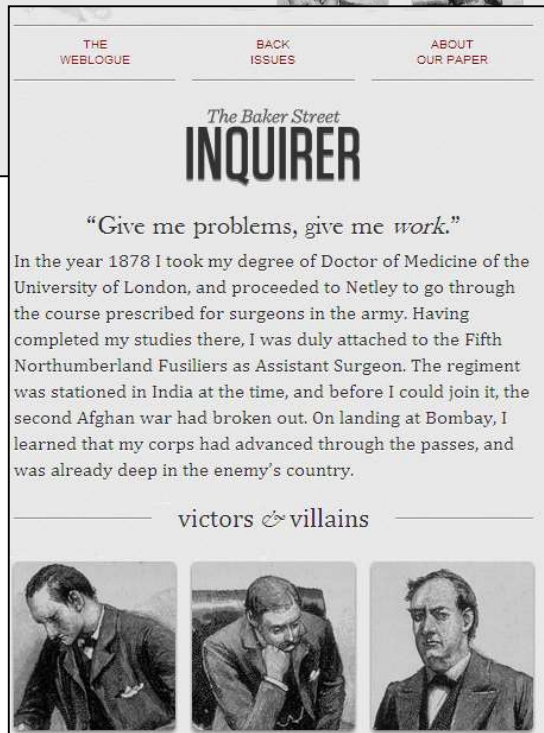
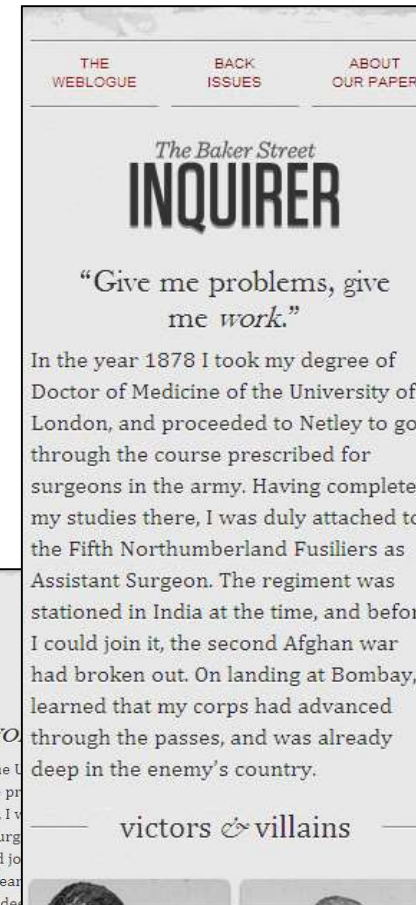
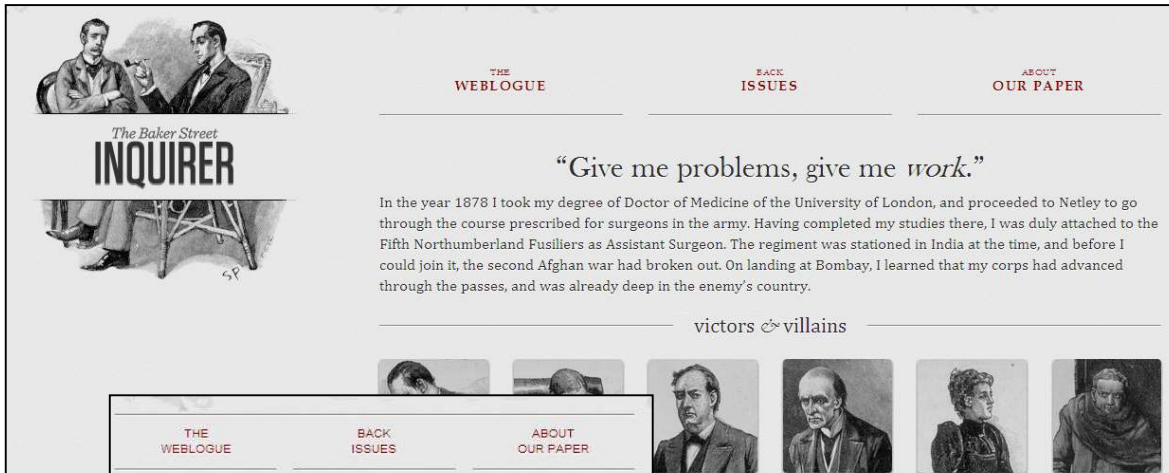


Exemplos



<http://css-tricks.com/>

Exemplos



<http://alistapart.com/d/responsive-web-design/ex/ex-site-FINAL.html>

A regra CSS para mídia: @media

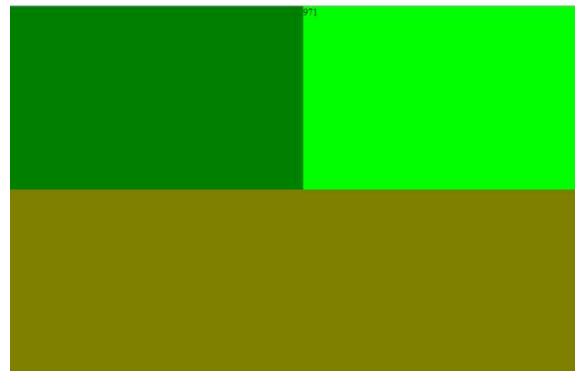
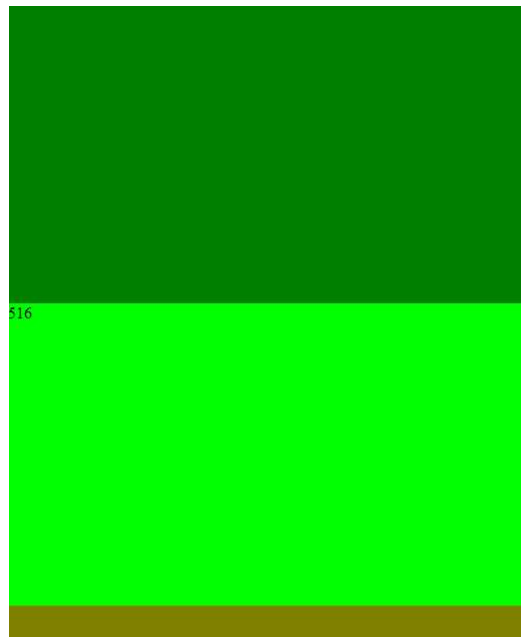
É possível no CSS fazer especificações que só são consideradas em condições determinadas. Isso é feito envolvendo as regras de especificação pela diretiva:

@media <condição> { ... }

Entre as chaves vão as regras CSS, que só serão usadas se a mídia corresponder à <condição> . Exemplos de <condição> são:

- **only screen and (max-width: 768px) and (min-width: 481px)**
(tela com largura entre 481px e 768px)
- **only screen and (max-width: 480px)** (tela de largura menor ou igual a que 480px)

Exemplo @media



exemploBasico_media.html

Exemplo @media

```
<style type="text/css">
body{ margin:0; }
.coluna1,.coluna2,.coluna3{ height:300px; }

@media screen and (min-width: 320px){
    .coluna1{ background-color:green; }
    .coluna2{ background-color:lime; }
    .coluna3{ background-color:olive; }
}

@media screen and (min-width: 780px){
    .coluna1, .coluna2{
        float:left;
        width:50%;
    }
    .coluna3{ clear:both; }
}
```

```
@media screen and (min-width: 1024px){
    .coluna1{
        float:left;
        width:30%;
    }
    .coluna2{
        float:left;
        width:60%;
    }
    .coluna3{
        float:left;
        width:10%;
        clear:none;
    }
}
</style>
```

```
<body>
<div class="coluna1"></div>
<div class="coluna2" id="msg"></div>
<div class="coluna3"></div>
<script> document.getElementById("msg").innerHTML = document.body.offsetWidth;</script>
</body>
```

Outros exemplos

```
1 <!-- CSS media query on a link element -->
2 <link rel="stylesheet" media="(max-width: 800px)" href="example.css" />
3
4 <!-- CSS media query within a style sheet -->
5 <style>
6 @media (max-width: 600px) {
7   .facet_sidebar {
8     display: none;
9   }
10 }
```

```
@media tv and (min-width: 700px) and (orientation: landscape) { ... }
```

Other Media Types

Media Type	Description
all	Used for all media type devices
aural	Used for speech and sound synthesizers
braille	Used for braille tactile feedback devices
embossed	Used for paged braille printers
handheld	Used for small or handheld devices
print	Used for printers
projection	Used for projected presentations, like slides
screen	Used for computer screens
tty	Used for media using a fixed-pitch character grid, like teletypes and terminals
tv	Used for television-type devices

<http://googlesamples.github.io/web-fundamentals/samples/layouts/rwd-fundamentals/media-queries.html>

Bootstrap (layout grid)

- <http://www.w3schools.com/bootstrap/default.asp>
- <http://getbootstrap.com/examples/grid/>

Grid options

See how aspects of the Bootstrap grid system work across multiple devices with a handy table.

	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints		
Container width	None (auto)	750px	970px	1170px
Class prefix	.col-xs-	.col-sm-	.col-md-	.col-lg-
# of columns	12			

```
<div id="b" class="row">
  <div class="col-xs-6">
    Conteúdo aqui.
  </div>
```

```
  <div class="col-xs-6">
    Conteúdo aqui.
  </div>
</div>
```

ou

```
<div id="b" class="row">
  <div class="col-xs-6 col-sm-6 col-lg-6">
    Conteúdo aqui.
  </div>
```

```
  <div class="col-xs-6 col-sm-6 col-lg-6">
    Conteúdo aqui.
  </div>
</div>
```

layout_bootstrap.html

Imagem (responsiva)

- <http://neilimagee.com/article/responsive-background-images>
- http://www.w3schools.com/bootstrap/bootstrap_images.asp
- <http://getbootstrap.com/css/>

Com CSS puro

```
<style>
  img{
    max-width: 100%;
    height:auto;
  }
</style>
```

Com bootstrap

```

```

imagem_resp.html

Imagens e textos em sites responsivos

É desejável que imagens e textos se redimensionem dependendo do tamanho da tela do dispositivo e até com o redimensionamento da janela de um navegador (em um PC, notebook etc.):

- Podemos usar tamanhos escaláveis ou até estilos diferentes para imagens, para diferentes resoluções;
- Usar tamanhos de textos em unidade em, rem ou % em lugar de utilizar as unidades px ou pt para font-size.

Observação

- Para garantir uma boa apresentação dos layouts responsivos em dispositivos móveis e permitir o zoom, insira o código abaixo dentro do bloco da tag head

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

Unidades para tamanhos de texto

- **“Ems” (em):** É uma unidade escalável. Por exemplo, se o tamanho do texto no documento é de 12pt, então 1em é igual a 12pt, 2em será igual a 24pt e .5em igual a 6pt.
- **Pixels (px):** Pixels são unidades fixas (não escalável). Um pixel é igual a um ponto da tela.
- **Pontos (pt):** Unidade de medida fixa (não escalável) bastante utilizada em mídia impressa. Um pt é igual a 1/72 de polegada.
- **Percentual (%):** Unidade de tamanho escalável, parecida com em. Por exemplo, se o tamanho do texto no documento é de 12pt, então 12pt é igual a uma fonte de tamanho 100% e 50% será um tamanho de fonte de 6pt.

Exemplo: tamanhos de texto com **em**

<style>

```
body {  
  font-size: 100%;  
  font-family: Verdana;  
}
```

```
h1 {  
  font-size: 1.4em;  
}
```

```
h2 {  
  font-size: 1.0em;  
}
```

```
p {  
  font-size: 0.8em;  
}
```

```
ul li {  
  font-size: 0.7em;  
}
```

</style>

Exemplo: imagens escaláveis

```
<style>
```

```
#figuragrande {  
  width: auto;  
  padding: 1%;  
  float:left;  
}
```

```
/* para telas com largura abaixo de 1024px: */
```

```
@media screen and (max-width: 1024px) {
```

```
  #figuragrande {  
    width: auto;  
    padding: 1%;  
    float:none;
```

```
  }  
}
```

```
/* telas com largura abaixo de 800px: */
```

```
@media screen and (max-width: 800px) {
```

```
  #figuragrande {  
    width: 80%;  
    padding: 1%;  
    float:none;
```

```
  }
```

```
}
```

```
...
```

```
</style>
```

Exemplo: quatro abordagens para imagens

```
<div id="figura1">
  
</div>
<div id="figura2">
  
</div>
<div id="figura3">
  
</div>
<div id="figura4">
  
</div>
```

img_approachs.html

```
img { /* uma img nunca ultrapassará o tamanho de seu contêiner */
  border: 0;
  max-width: 100%;
  height: auto;
  width: auto\9; /* ie8 */
}

/***** para telas com largura abaixo de 1024px: *****/
@media screen and (max-width: 1024px) {
  #figura4 {
    width: 80%;
    padding: 1%;
    float: none;
  }
}
```

Exemplo: quatro abordagens para imagens (cont.)



As imagens **figura1** e **figura2** foram redimensionadas para tamanhos fixos, menor e maior que o tamanho original (ver no slide anterior).

A **figura3** está sendo mostrada no tamanho original (640 x 360 px).

Nos três casos, se a tela for menor que o tamanho da imagem, a imagem será redimensionada por causa do estilo `max-width:100%;`

Exemplo: quatro abordagens para imagens (cont.)



Suponhamos uma tela menor que 1024px.

A **figura3** será mostrada com o tamanho original (640 x 360 px) ou será redimensionada por causa do estilo `max-width:100%`;

A **figura4** foi ajustada a 80% do tamanho de seu contêiner, por causa da condição em `@media screen and (max-width: 1024px)`



Tamanho de telas

► Phones and Handhelds

► Tablets

▼ Galaxy Tablets

► Laptops

► Wearables

CSS

```
/* ----- Galaxy Tab 10.1 ----- */

/* Portrait and Landscape */
@media
  (min-device-width: 800px)
  and (max-device-width: 1280px) {

}

/* Portrait */
@media
  (max-device-width: 800px)
  and (orientation: portrait) {

}

/* Landscape */
@media
```

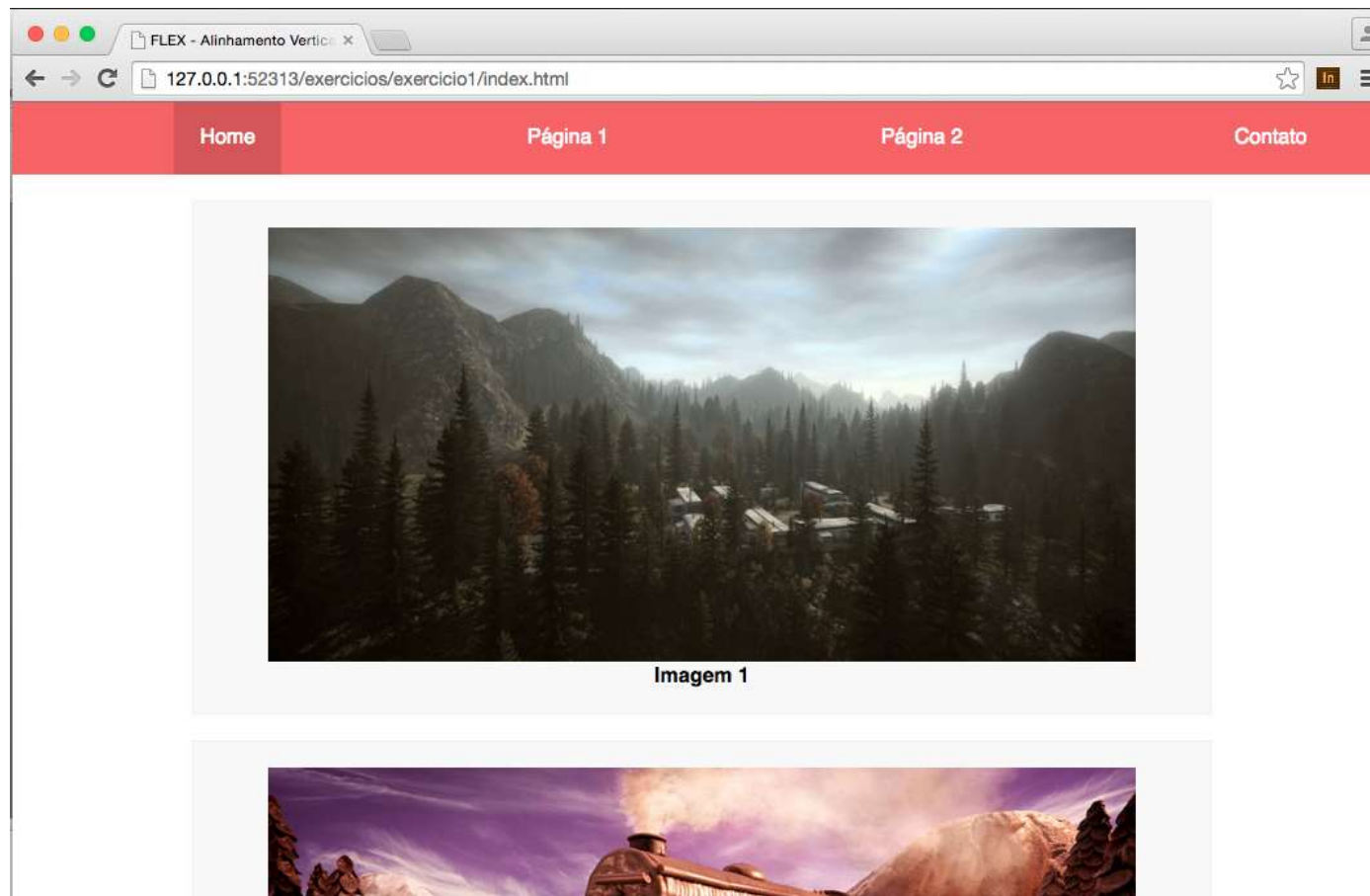
320 x 480 iPhone 3G/3GS	480 x 320 iPhone 3G/3GS	1024 x 600 Most Netbooks
480 x 720 Meizu M8	720 x 480 Meizu M8	1280 x 800 MacBook Air 08
480 x 800 Google Nexus one	800 x 480 Google Nexus one	1366 x 768 Some Laptops
640 x 960 iPhone 4	960 x 640 iPhone 4	1440 x 900 MacBook Pro 15 inches
768 x 1024 iPad	1024 x 768 iPad	Maximum Resize to Maximum

Current Inner 1341 x 593

<http://resizemybrowser.com/>

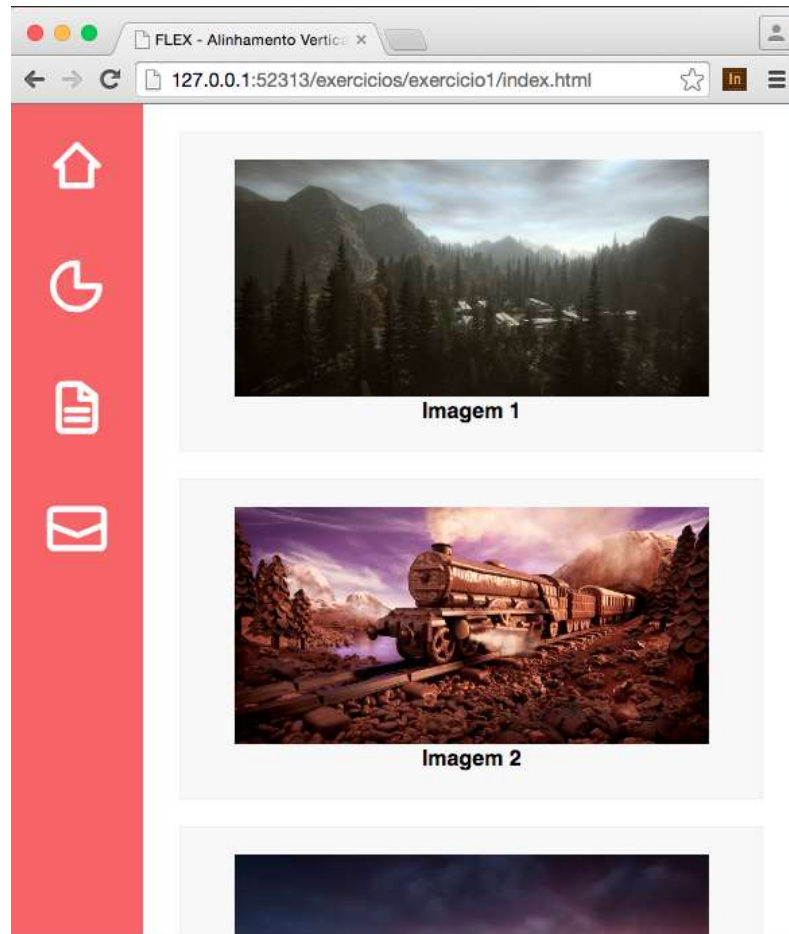
<https://css-tricks.com/snippets/css/media-queries-for-standard-devices/>

Exemplo completo



Exemplo completo

Para largura igual ou menor que 600px, o layout deve ser como mostra a imagem:



Exemplo completo

Cores utilizadas:



Utilize os conceitos de @media para este exercício!

Dicas:

- Para remover o ícone da lista: `list-style-type: none;`
- Para o item da lista: `display: block;`
- Experimente trabalhar com porcentagens para larguras, margins e paddings
- Observe o efeito de mudança de cor do item quando o mouse passa sobre ele.
- Procure colocar `height 100%` no `body` e `html`