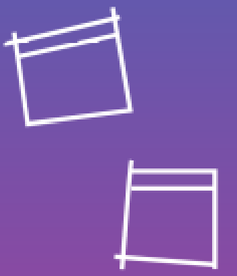
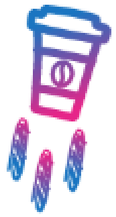
DBC
DIGITAL BUSINESS COMPANY®

VEM SER
DBC



Módulo 2 - HTML e CSS

Aula 6 - calc, Responsividade e Media Queries



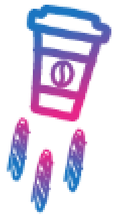
calc

calc

calc é uma função do CSS que permite realizar cálculos simples.

Podem ser realizados cálculos de:

- Adição +
- Subtração -
- Divisão /
- Multiplicação *



Responsividade

Responsividade

Um site responsivo é aquele cujo conteúdo se adapta ao formato de tela do dispositivo utilizado para a sua visualização.

O que isso significa?

Independentemente do dispositivo com que o site for acessado, um site responsivo deve exibir corretamente o *layout* e o conteúdo.

Qual a importância?

Com a popularização dos *smartphones*, as pessoas acessam à internet cada vez mais pelo celular. Ter um site responsivo garante que este público seja atendido.

Curiosidade: *mobile first*

Mobile first é uma metodologia cujo foco de desenvolvimento principal é a usabilidade em dispositivos móveis.

O que isso significa?

Diferentemente da metologia de desenvolvimento comumente praticada, em que a versão *desktop* é desenvolvida primeiro e somente depois a *mobile*, quando se adota o *mobile first*, a primeira versão é desenvolvida para *mobile* e então adaptada para outras plataformas.

Tamanhos de tela

Vamos relembrar o que vimos sobre *breakpoint* no módulo de Boostrap.

Alguns tamanhos de tela são:
- Celular;
- Tablet;
- Desktop.

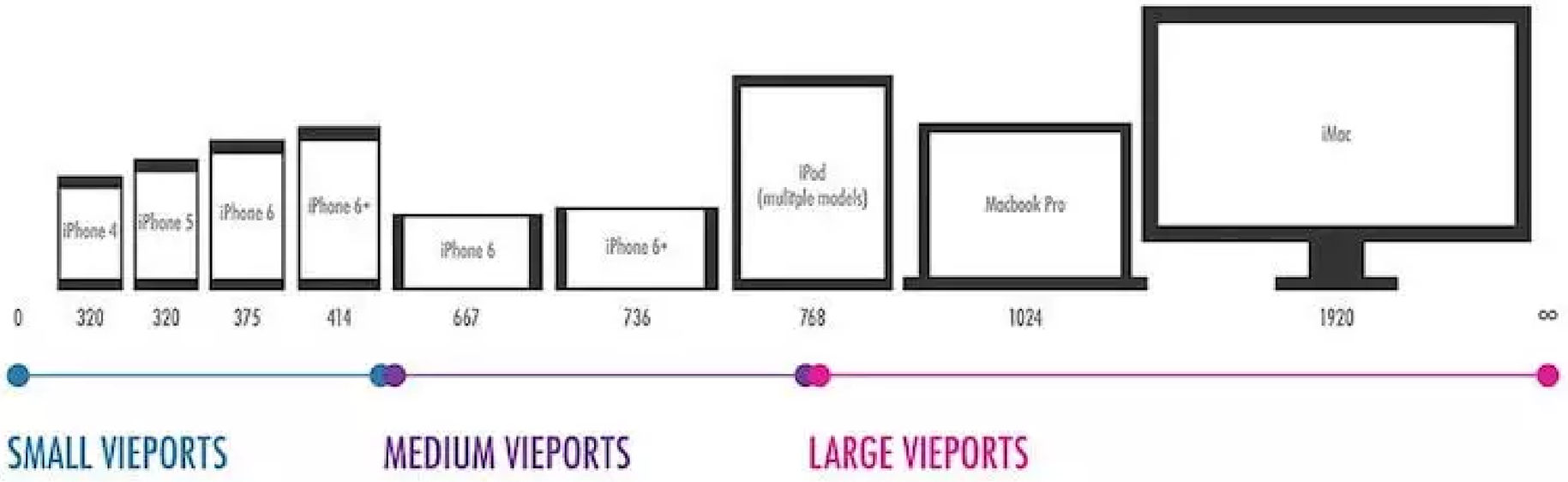
Breakpoints

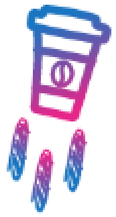
Quantos** breakpoints **utilizar?

É interessante que um site em produção tenha, pelo menos, três *breakpoints*. Um para desktop, outro para tablet e outro para celulares.

Os sites atualmente costumam ter de **três** a **quatro** *breakpoints*. Este quarto último seria para monitores com baixa resolução de tela, garantindo que todo o público que acessar o site consiga navegar da forma esperada.

Breakpoints





Media Queries: Responsividade na prática

Iniciando com media queries

É possível adicionar *media queries* ao projeto de duas formas, vamos à primeira:

Usando *media queries* dentro do *style.css*.

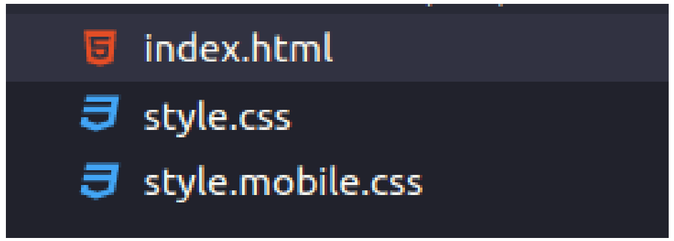
```css
.container {
  width: 100px;
  height: 100px;
  background-color: blue;
}

@media (max-width: 900px) {
  .container {
    background-color: red;
  }
}
```

Iniciando com media queries

Usando *media queries em um arquivo externo*

Vamos começar criando dois arquivos css, um *style.css* e outro *style.mobile.css*



Agora vamos dizer para o HTML qual arquivo deve ser usado em cada tamanho de tela:

```
1  <link rel="stylesheet" href="styles.css" />
2  <link rel="stylesheet" href="styles.mobile.css" media="(max-width: 900px)" />
```

Iniciando com media queries

Repare que, como não sinalizamos nenhum padrão de *width* na primeira importação, o arquivo *style.css* é o padrão.

E o arquivo *style.mobile.css* é o arquivo de estilo utilizado para telas cujo tamanho não ultrapasse 900px de largura.

Media queries

Imagine que tenhamos a seguinte estilização:

HTML

```html
<div class="container"></div>
```

style.css

```css
.container {
  width: 100px;
  height: 100px;
  background-color: blue;
}
```

style.mobile.css

```css
.container {
  background-color: red;
}
```

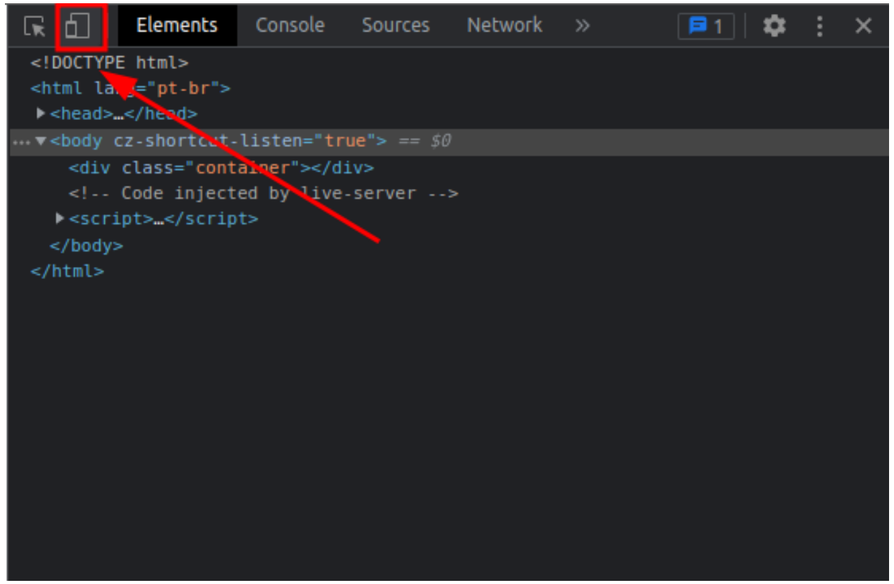
Resultado

Padrão

Fazendo o *Chrome* simular a resolução de um celular

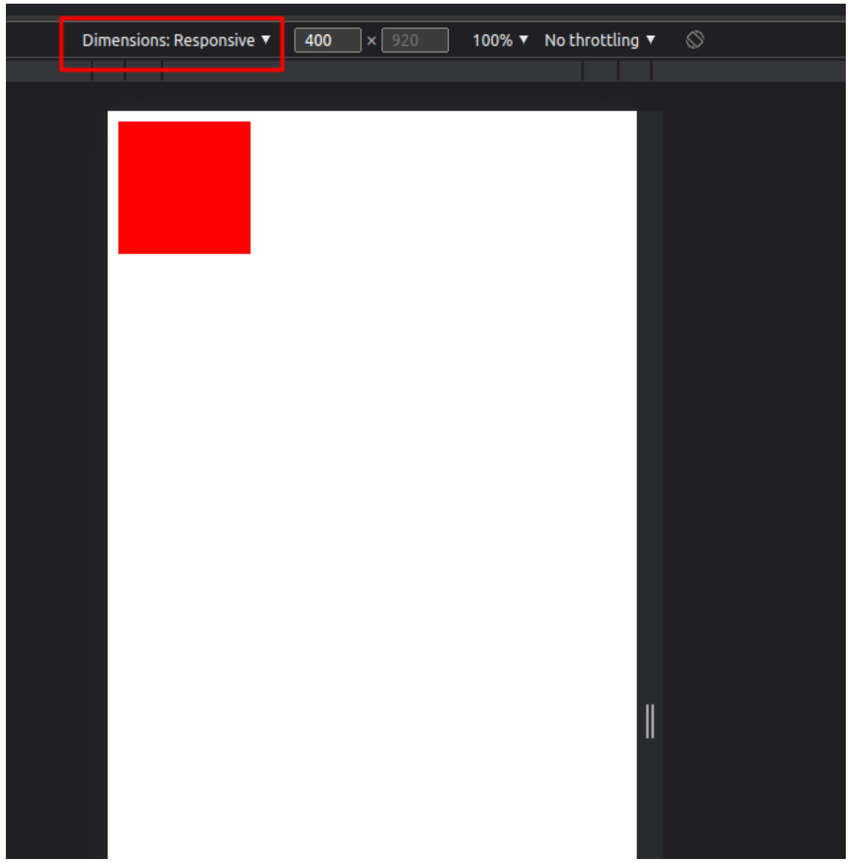
Aperte **Ctrl** + **Shift** + **I**

Será aberta uma janela lateral, conforme a mostrada abaixo, vamos clicar no ícone de dispositivos.



Fazendo o *Chrome* simular a resolução de um celular

Agora é possível definir as dimensões. Veja que o estilo mudou:



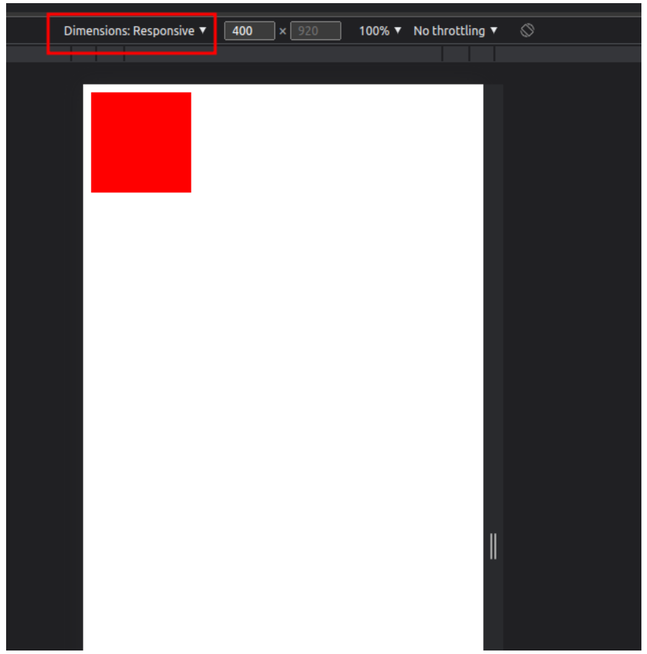
Resultados

Tamanho de tela padrão

style.css

Tamanho de tela menor que 900px

style.mobile.css



Escondendo elementos

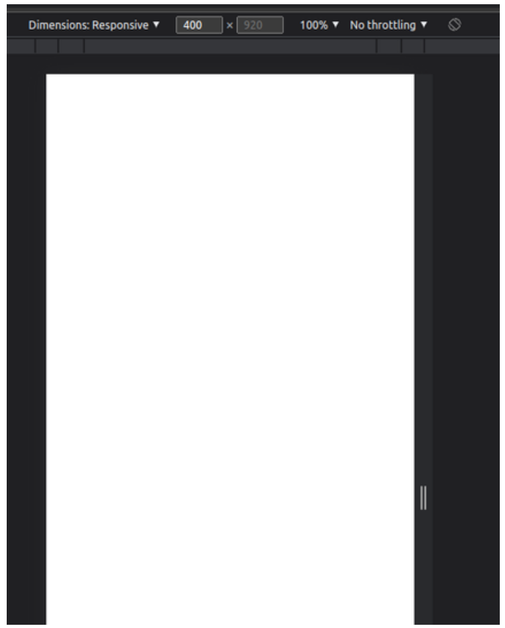
Também podemos "esconder" elementos de tela que queremos que funcionem apenas um tamanho de tela específico. Para isso, usamos ***display*** ou ***visibility***.

Códigos do *style.mobile.css*

```css
.container {
  display: none;
}
```

```css
.container {
  visibility: hidden;
}
```

Resultado prático



Escondendo elementos

Apesar do efeito prático de ***display*** e ***visibility***. Parecer o mesmo à primeira vista, existem algumas diferenças.

Para entender as diferenças, vamos adicionar um segundo container no HTML:

```html
<div class="container"></div>
<div class="segundo-container"></div>
```

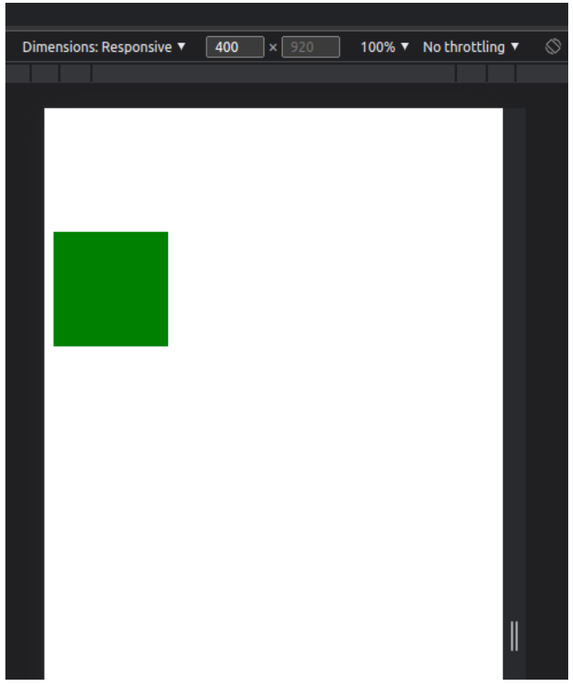
Com *visibility*

Agora vamos "esconder" o elemento de classe *.container* com ***visibility***:

style.mobile.css

```css
.container {
  visibility: hidden;
}
```

Resultado



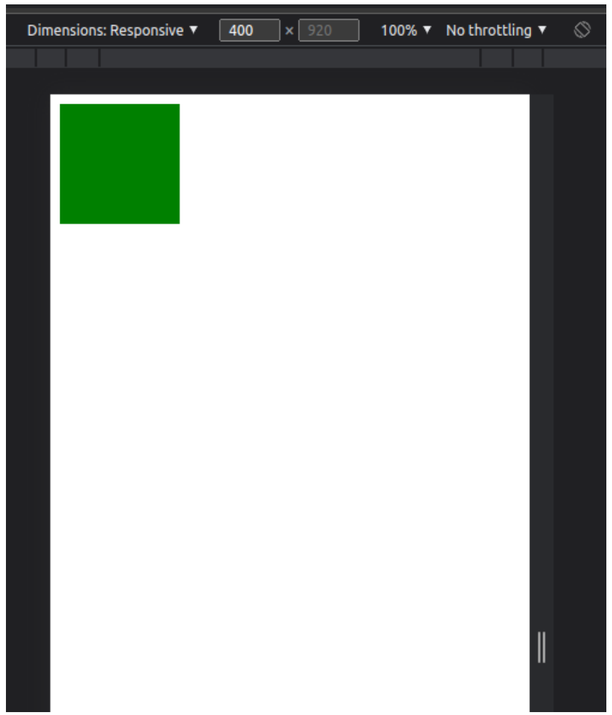
Com *display*

Agora vamos "esconder" o elemento de classe *.container* com ***display***:

style.mobile.css

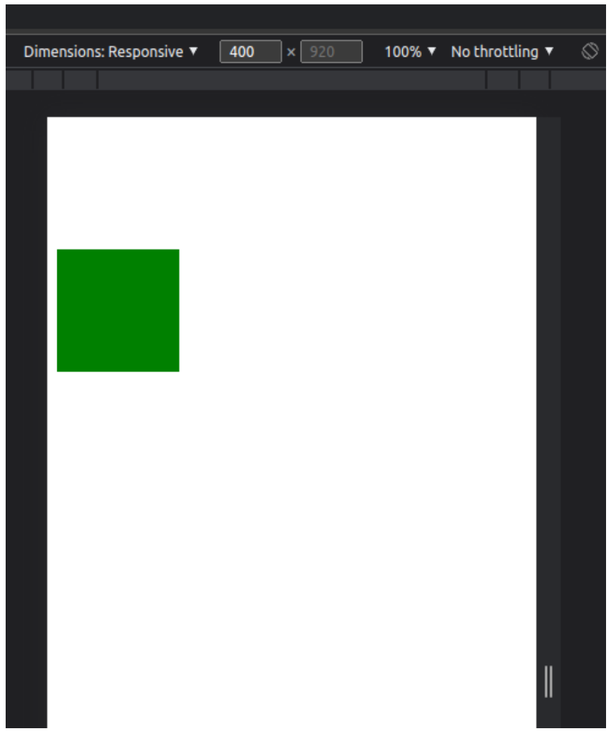
```css
.container {
  display: none;
}
```

Resultado

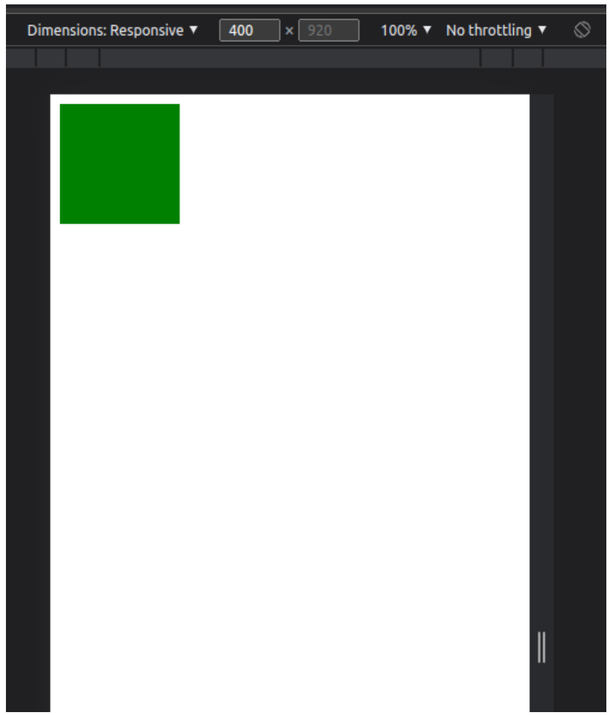


Diferenças entre *visibility* e *display*

Com ***visibility***



Com ***display***

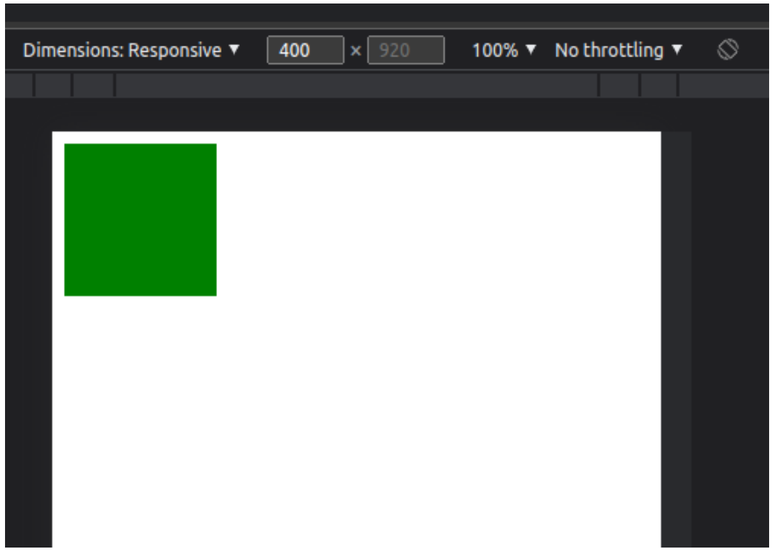


Diferenças entre *visibility* e *display*

Perceba que, enquanto com *visibility*, apesar de termos "escondido" o elemento de classe *.container*, a altura ainda permanece, e o elemento de classe *.segundo-container* só é renderizado 100px abaixo.

Uma forma de corrigir isso seria:

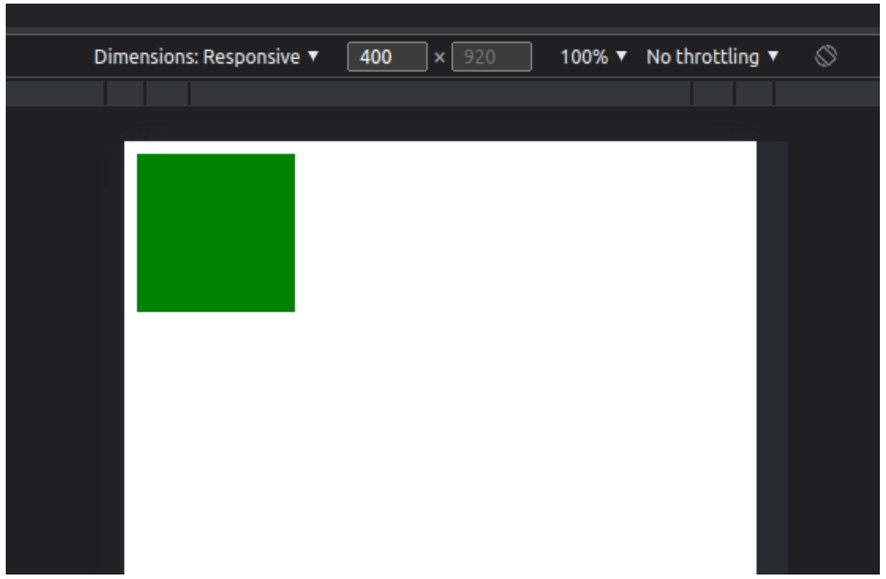
```css
.container {
  visibility: hidden;
  width: 0;
  height: 0;
}
```



Diferenças entre *visibility* e *display*

Ou podemos simplesmente usar *display*. O efeito prático será o mesmo que o do código do *slide anterior.*

```css
.container {
  display: none;
}
```



Obrigado!

DBC

DIGITAL BUSINESS COMPANY®

/dbc.company

/dbccompany.com.br

/company/dbc-company