

Bootstrap

Aula 1



Bootstrap

É o framework mais utilizado para criar layouts e telas de sistemas web, dashboards, sites responsivos e sites comuns, também. Ele facilita muito o trabalho de front-end.

Ele é o responsável por definir um layout estrutural.

Bootstrap – Como Funciona

de acordo com as classes que você colocar nos elementos HTML, o Bootstrap vai definir o visual e o comportamento daquele componente.

Na maioria das vezes, é só colocar as classes CSS certas que você vai ter um componente visual.

Alguns componentes vão precisar de outros atributos a mais. Ou, também, de alguma marcação específica usando várias div's em conjunto (como o Modal ou o Carousel).

Bootstrap – Como Funciona

Ele é composto por arquivos .css e .js, conforme estrutura de diretório a seguir:

```
bootstrap/  
├── css/  
│   ├── bootstrap-grid.css  
│   ├── bootstrap-grid.css.map  
│   ├── bootstrap-grid.min.css  
│   ├── bootstrap-grid.min.css.map  
│   ├── bootstrap-reboot.css  
│   ├── bootstrap-reboot.css.map  
│   ├── bootstrap-reboot.min.css  
│   ├── bootstrap-reboot.min.css.map  
│   ├── bootstrap.css  
│   ├── bootstrap.css.map  
│   ├── bootstrap.min.css  
│   └── bootstrap.min.css.map  
└── js/  
    ├── bootstrap.bundle.js  
    ├── bootstrap.bundle.js.map  
    ├── bootstrap.bundle.min.js  
    ├── bootstrap.bundle.min.js.map  
    ├── bootstrap.js  
    ├── bootstrap.js.map  
    ├── bootstrap.min.js  
    └── bootstrap.min.js.map
```

Trabalhando com o Bootstrap por CDN

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <!-- meta tags obrigatorias -->
  <meta charset="utf-8">
  <!-- meta tags responsiva -->
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <title>Olá Bootstrap</title>

  <!-- CSS do Bootstrap -->
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.css">
</head>
<body>
  <div class="container">
    <h1 Olá, Bootstrap! Vamos dar início!</h1>
  </div>

  <!-- Primeiro o jQuery, depois o Popper.js, e depois o Bootstrap JS -->
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.js"></script>
</body>
</html>
```

Cores do Texto

O Bootstrap 4 tem algumas classes contextuais que podem ser usadas para fornecer "significado por meio de cores". As classes de cores de texto são: `.text-muted`, `.text-primary`, `.text-success`, `.text-info`, `.text-warning`, `.text-danger`, `.text-secondary`, `.text-white`, `.text-dark`, `.text-body` (cor do corpo default / muitas vezes preto) e `.text-light`:

primary = primária

secondary = secundário

success = sucesso

danger = perigo

warning = aviso

info = informar

light = claro

Cores do Texto

Exemplo:

```
<!DOCTYPE html> <
<html lang="pt-br">
  <head>
    <title>Bootstrap Example</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
  </head>
  <body>
    <div class="container">
      <h2>Contextual Colors</h2>
      <p>Use the contextual classes to provide "meaning through colors":</p>
      <p class="text-muted">This text is muted.</p>
      <p class="text-primary">This text is important.</p>
      <p class="text-success">This text indicates success.</p>
      <p class="text-info">This text represents some information.</p>
      <p class="text-warning">This text represents a warning.</p>
      <p class="text-danger">This text represents danger.</p>
      <p class="text-secondary">Secondary text.</p>
      <p class="text-dark">This text is dark grey.</p>
      <p class="text-body">Default body color (often black).</p>
      <p class="text-light">This text is light grey (on white background).</p>
      <p class="text-white">This text is white (on white background).</p>
    </div>
  </body>
</html>
```

Cores de Fundo

As classes de cores do fundo são: `.bg-primary`, `.bg-success`, `.bg-info`, `.bg-warning`, `.bg-danger`, `.bg-secondary`, `.bg-dark` e `.bg-light`. Observe que as cores de fundo não definem a cor do texto, portanto, em alguns casos, você desejará usá-las junto com uma classe `.text-*`.

This text is important.

This text indicates success.

This text represents some information.

This text represents a warning.

This text represents danger.

Secondary background color.

Dark grey background color.

Light grey background color.

Cores de Fundo

Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <!-- meta tags obrigatorias -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <title>Olá Bootstrap</title>

    <!-- CSS do Bootstrap -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.css">
  </head>
  <body>
    <div class="container">
      <h2>Trabalho com as cores de fundo</h2>
      <p>CORES DE FUNDO aplicada na fonte</p>
      <p class="bg-primary text-white">This text is important.</p>
      <p class="bg-success text-white">This text indicates success.</p>
      <p class="bg-info text-white">This text represents some information.</p>
      <p class="bg-warning text-white">This text represents a warning.</p>
      <p class="bg-danger text-white">This text represents danger.</p>
      <p class="bg-secondary text-white">Secondary background color.</p>
      <p class="bg-dark text-white">Dark grey background color.</p>
      <p class="bg-light text-dark">Light grey background color.</p>
    </div>
  </body>
</html>
```

Atividade Prática 1

Alunos irão postar o código para demonstração

Criar um arquivo .html para atender as demandas abaixo:

1. Criar uma tag <head> e aplicar uma tag <title>. “Trabalhando com Cores”;
2. Criar uma tag <meta> para usar o charset;
3. Criar uma outra tag <meta> para tratar da questão do name e conteúdo;
4. Criar uma tag para cada CDN para a ligação com o bootstrap;
5. Criar uma <h1> para destacar um Título;
6. Criar um parágrafo com <style> para destacar um texto referente ao uso das cores;
7. Criar uma <div> e usar igual a containers;
8. Criar dois textos diferentes trabalhando com as cores: **.text-warning**, **.text-danger**;
9. Os textos devem ficar centralizados na página;
10. Criar uma <h1> para destacar um novo Título;
11. Criar um parágrafo com <style> para destacar um texto referente ao uso dos fundos das cores: trabalhar a fonte da letra e a cor;
12. Criar uma <div> e usar igual a containers;
13. Criar quatro textos diferentes trabalhando com os: **bg-secondary/ bg-info/ bg-warning / bg-primary**;

Flexbox

A maior diferença entre o Bootstrap 3 e o Bootstrap 4 é que o Bootstrap 4 agora usa flexbox, em vez de floats, para lidar com o layout.

O Módulo de Layout de Caixa Flexível torna mais fácil projetar uma estrutura de layout responsiva e flexível sem usar flutuação ou posicionamento.

Flexbox

A maior diferença entre o Bootstrap 3 e o Bootstrap 4 é que o Bootstrap 4 agora usa flexbox, em vez de floats, para lidar com o layout.

O Módulo de Layout de Caixa Flexível torna mais fácil projetar uma estrutura de layout responsiva e flexível sem usar flutuação ou posicionamento.

Para criar um contêiner flexbox e transformar filhos diretos em flex items, use a classe **d-flex**.

Flexbox

Para começar a usar o modelo Flexbox, você precisa primeiro definir um flex container.



Exemplo:

```
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
</div>
```

Flexbox



Exemplo:

```
<div class="d-flex p-3 bg-secondary text-white">  
  <div class="p-2 bg-info">Flex item 1</div>  
  <div class="p-2 bg-warning">Flex item 2</div>  
  <div class="p-2 bg-primary">Flex item 3</div>  
</div>
```

p-3 → tamanho do container

bg → background (veremos quando falarmos das cores)

text-white → cor do texto

Dentro da `<div>` é possível colocar a tag `<a>`.

Atividade Prática 2

Exercício: arquivo exercflexbox.html

Criar um arquivo .html para atender as demandas abaixo:

1. Criar uma tag <head> e aplicar uma tag <title>. Texto livre;
2. Criar uma tag <meta> para usar o charset;
3. Criar uma outra tag <meta> para tratar da questão do name e conteúdo;
4. Criar uma tag para cada CDN para a ligação com o bootstrap (exemplo já enviado por mim no arquivo flexbox.html);
5. Usar a classe d-flex para os containers;
6. Criar quatro containers;
7. Criar um background na cor diferente dos quatro containers (usar cores bg-secondary/ bg-info/ bg-warning / bg-primary);
8. Inserir em cada um dos quatro containers informações diferentes referente a uma temática conforme sua escolha.

Containers

Ele é o responsável por definir um layout estrutural, proporcionalmente alinhado na página. Ele não é nada mais, e nada menos, do que uma classe CSS com propriedades pré-definidas, que podemos usar normalmente em um elemento da mesma forma que faríamos como se fosse uma classe CSS que criamos em nosso projeto.

Containers

Use a classe `.container` para criar um contêiner responsivo de largura fixa.

Exemplo:

```
<div class="container">  
  <p>Dando início ao jogo<p>  
</div>
```

Uma área de visão com menor utilização na tela



Containers

Classe `.container-fluid` para um container com de largura total, abrangendo toda a largura da sua área de visualização.

Exemplo:

```
<div class="container-fluid">  
  <p>Dando início ao jogo<p>  
</div>
```



Containers Responsivos

Você também pode usar as classes `.container-sm|md|lg|xl` para criar contêineres responsivos.

O `max-width` do container mudará em diferentes tamanhos de tela / janelas de exibição:

Classe	Extra pequeno <576 px	Pequeno ≥576px	Médio ≥768 px	Grande ≥992 px	Extra grande ≥1200px
<code>.container-sm</code>	100%	540px	720px	960px	1140px
<code>.container-md</code>	100%	100%	720px	960px	1140px
<code>.container-lg</code>	100%	100%	100%	960px	1140px
<code>.container-xl</code>	100%	100%	100%	100%	1140px

Containers - Preenchimento

Por padrão, os contêineres têm 15px de preenchimento esquerdo e direito, sem preenchimento superior ou inferior. Portanto, costumamos usar **utilitários de espaçamento**, como preenchimento extra e margens para torná-los ainda melhores.

Por exemplo, **.pt-3** significa "adicionar um preenchimento superior de 16 px":

Grid

O sistema grid Bootstrap usa vários containers, linhas e colunas para arranjar e alinhar conteúdo.

O sistema de grade do Bootstrap é construído com flexbox e permite até 12 colunas na página.

Se não quiser usar todas as 12 colunas individualmente, você pode agrupar as colunas para criar colunas mais largas:

Grid

Classes do grid

O sistema de grade Bootstrap 4 tem cinco classes:

- **.col** → (dispositivos extra pequenos - largura da tela menor que 576 px).
- **.col-sm** → (dispositivos pequenos - largura da tela igual ou superior a 576 px);
- **.col-md** → (dispositivos médios - largura da tela igual ou superior a 768 px);
- **.col-lg** → (dispositivos grandes - largura da tela igual ou superior a 992 px);
- **.col-xl** → (dispositivos xlarge - largura da tela igual ou superior a 1200 px).

As classes acima podem ser combinadas para criar layouts mais dinâmicos e flexíveis.

Dica: Cada classe aumenta, então se você deseja definir as mesmas larguras para **sm** e **md**, você só precisa especificar **sm**.

Grid

Estrutura Básica de uma Grade Bootstrap 4

Exemplo clássico (não responsivo):

```
<div class="row">  
  <div class="col"></div>  
  <div class="col"></div>  
  <div class="col"></div>  
</div>
```

Exemplo para diferentes dispositivos (responsivo):

```
<div class="row">  
  <div class="col-*-*"></div>  
  <div class="col-*-*"></div>  
  <div class="col-*-*"></div>  
</div>
```

Grid

A primeira estrela (*) representa a responsividade: `sm`, `md`, `lg` ou `xl`, enquanto a segunda estrela representa um número, que deve somar 12 colunas para cada linha.

Outra opção é, em vez de adicionar um número a cada um `col`, deixe o bootstrap lidar com o layout, para criar colunas de largura igual: duas colunas `"col"` elementos = 50% de largura para cada coluna, três colunas = 33,33% de largura para cada coluna, quatro `.col-sm | md | lg | xl` colunas = 25% de largura, etc.

Você também pode usar para tornar as colunas responsivas.

Grid

- Containers criam meios para centralizar e, horizontalmente, preencher os conteúdos de seu site:
 - Use `.container` para ter uma largura responsiva em pixel ou `.container-fluid` para ter `width: 100%`, em todas viewports e tamanhos de dispositivos.
- Rows são elementos para envolver colunas:
 - Cada coluna tem `padding` horizontal (gutter) para controlar o espaço, entre elas.
 - Este `padding`, depois, é cancelado com rows usando margens negativas. Assim, todo conteúdo em suas colunas é, visualmente, alinhado à esquerda.

Grid

- Em um layout grid, o conteúdo deve ser posicionado dentro de colunas e só elas podem ser filhos imediatos de `.rows`;
- Graças ao flexbox, colunas grid sem `width` declarado vão, automaticamente, se dimensionar com larguras iguais;
 - Por exemplo, quatro exemplos de `.col-sm` vão, automaticamente, ter 25% de largura, no breakpoint `sm` ou maior;
- Classes de colunas indicam o número de colunas que você quer usar, dentro de uma possibilidade de 12, por row;
 - Se você quiser três colunas de larguras idênticas, pode usar `.col-4`, por exemplo.

Grid

- Largura de colunas são definidas em porcentagem para que, então, elas sejam sempre fluidas e dimensionadas com relação a seus elementos pais;
- Colunas possuem `padding` horizontal para criar gutters, entre cada coluna:
 - No entanto, você pode remover a `margin` das rows e `padding` das colunas, usando `.no-gutters` na `.row`.
- Para fazer o grid responsivo, existem cinco breakpoints, um para cada `breakpoint responsivo`: extra small (implícito), small, medium, large e extra large;

Grid

- Breakpoints grid são baseados em media queries `min-width`, significando que **elas aplicam estilos para o dado breakpoint e outros maiores**:
 - Exemplo: `.col-sm-4` aplica ao small, medium, large e extra large, mas não ao primeiro breakpoint `xs`.

Grid Responsivo

Texto 1 de 3

Texto 2 de 3

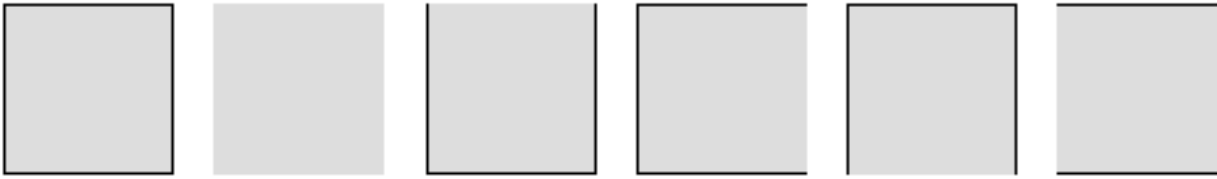
Texto 3 de 3

Exemplo:

```
<div class="container">  
  <div class="row">  
    <div class="col-sm">  
      Uma de três colunas  
    </div>  
    <div class="col-sm">  
      Duas de três colunas  
    </div>  
    <div class="col-sm">  
      Três de três colunas  
    </div>  
  </div>  
</div>
```

Grid - Bordas

Use a classe **border** para adicionar ou remover bordas de um elemento:



Exemplo:

```
<span class="border"></span>  
<span class="border border-0"></span>  
<span class="border border-top-0"></span>  
<span class="border border-right-0"></span>  
<span class="border border-bottom-0"></span>  
<span class="border border-left-0"></span>
```

Grid – Cores das Bordas

Use a classe **border** para adicionar ou remover bordas de um elemento:



Exemplo:

```
<span class="border border-primary"></span>  
<span class="border border-secondary"></span>  
<span class="border border-success"></span>  
<span class="border border-danger"></span>  
<span class="border border-warning"></span>  
<span class="border border-info"></span>  
<span class="border border-light"></span>  
<span class="border border-dark"></span>  
<span class="border border-white"></span>
```

Atividade Prática 2

Exercício:

Criar um arquivo .html

1. Fazer o designer de um jogo da velha usando usando os modelos de borda aprendido no slide anterior;
2. Serão 9 caixas e em cada caixa você poderá colocar a marcação do jogo da velha que quiser;
3. É preciso que haja um título de cabeçalho;
4. É preciso que haja um título para a página.

Grid - Colunas

Largura

Definir a largura de um elemento com as classes W- * (`.w-25`, `.w-50`, `.w-75`, `.w-100`, `.mw-100`):

Exemplo:

```
<div class="w-25 bg-warning">Width 25%</div>  
<div class="w-50 bg-warning">Width 50%</div>  
<div class="w-75 bg-warning">Width 75%</div>  
<div class="w-100 bg-warning">Width 100%</div>  
<div class="mw-100 bg-warning">Max Width 100%</div>
```

Grid - Colunas

Altura

Definir a altura de um elemento com as classes de h- * (`.h-25`, `.h-50`, `.h-75`, `.h-100`, `.mh-100`):

Exemplo:

```
<div style="height:200px; background-color:#ddd">  
  <div class="h-25 bg-warning">Height 25%</div>  
</div>
```

Grid - Colunas

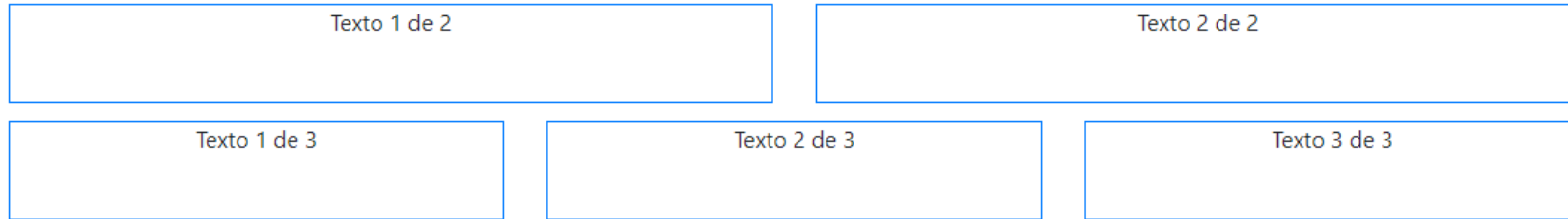
Layout automático de colunas

Use classes de breakpoints específicos para dimensionamento de colunas fácil, sem uma classe numerada explícita, como `.col-sm-6`.

Largura idêntica

Por exemplo, aqui estão dois layouts grid que aparecem em todo dispositivo e viewport, desde `xs` até `xl`. Use qualquer quantidade de classes sem números, para cada breakpoint necessário e, então, todas colunas terão a mesma largura.

Grid - Columnas



Exemplo:

```
<div class="container text-center">  
  <div class="row">  
    <div class="col">  
      1 de 2  
    </div>  
    <div class="col">  
      2 de 2  
    </div>  
  </div>  
  <div class="row">  
    <div class="col">  
      1 de 3  
    </div>  
    <div class="col">  
      2 de 3  
    </div>  
    <div class="col">  
      3 de 3  
    </div>  
  </div>  
</div>
```

Atividade Prática 3

Exercício: arquivo exercontainer.html

Criar um arquivo .html para atender as demandas abaixo usando o cabeçalho padrão definido na atividade anterior:

1. Usar a classe container;
2. Os textos deverão estar centralizados;
3. Criar uma linha com um container;
4. Criar uma outra linha com dois containers;
5. Na primeira linha:
 1. Inserir uma informação conforme sua escolha;
 2. Colocar uma borda no container;
 3. Colocar uma cor no texto;
6. Na segunda linha:
 1. Primeiro container inserir uma imagem, tag ;
 2. Segundo container inserir um link para um outro site, tag <a>;
 3. Colocar uma borda no container;
 4. Colocar uma cor no texto;

Grid - Colunas

Coluna	Coluna
Coluna	Coluna

Exemplo:

```
<div class="container">
  <div class="row">
    <div class="col">
      Coluna
    </div>
    <div class="col">
      Coluna
    </div>
  <div class="w-100"> </div>
    <div class="col">
      Coluna
    </div>
    <div class="col">
      Coluna
    </div>
  </div>
</div>
```

Grid - Colunas

Definindo largura de uma coluna

Layout automático em colunas **flexbox** do grid também significa que você pode definir a largura de uma coluna e ter suas colunas irmãs redimensionadas, automaticamente. Você também pode usar **classes grid** pré-definidas, **mixins** grid ou larguras **inline**. Perceba que as outras colunas vão se redimensionar, não importa a largura da coluna do meio.

Grid - Colunas

1 de 3	2 de 3 (maior)	3 de 3
--------	----------------	--------

Exemplo:

```
<div class="container">  
  <div class="row">  
    <div class="col">  
      1 de 3  
    </div>  
    <div class="col-6">  
      2 de 3 (maior)  
    </div>  
    <div class="col">  
      3 de 3  
    </div>  
  </div>  
</div>
```


Grid - Colunas

Conteúdo com largura variável

Use classes `col-{breakpoint}-auto` para dimensionar colunas, baseando-se na largura natural de seus conteúdos.

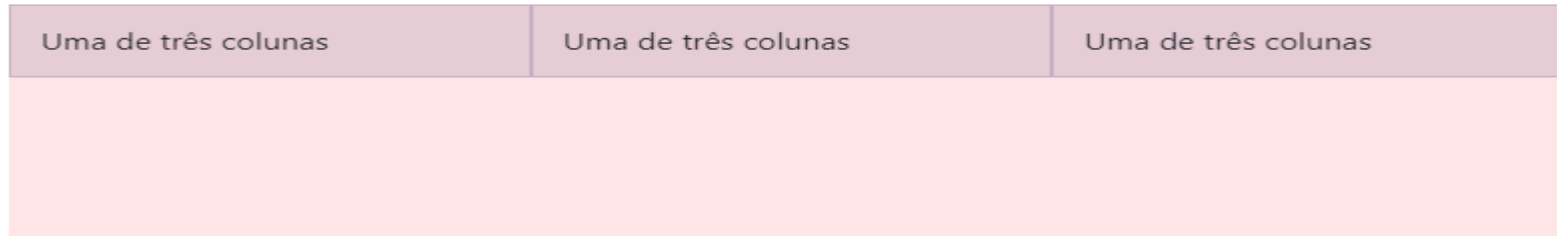
1 de 3	Conteúdo com largura variável	3 de 3
1 de 3	Conteúdo com largura variável	3 de 3

Grid - Colunas

Exemplo:

```
<div class="container">  
  <div class="row justify-content-md-center">  
    <div class="col col-lg-2">  
      1 de 3  
    </div>  
    <div class="col-md-auto">  
      Conteúdo com largura variável  
    </div> <div class="col col-lg-2">  
      3 de 3  
    </div>  
  </div>  
  <div class="row">  
    <div class="col">  
      1 de 3  
    </div>  
    <div class="col-md-auto">  
      Conteúdo com largura variável  
    </div>  
    <div class="col col-lg-2">  
      3 de 3  
    </div>  
  </div>  
</div>
```

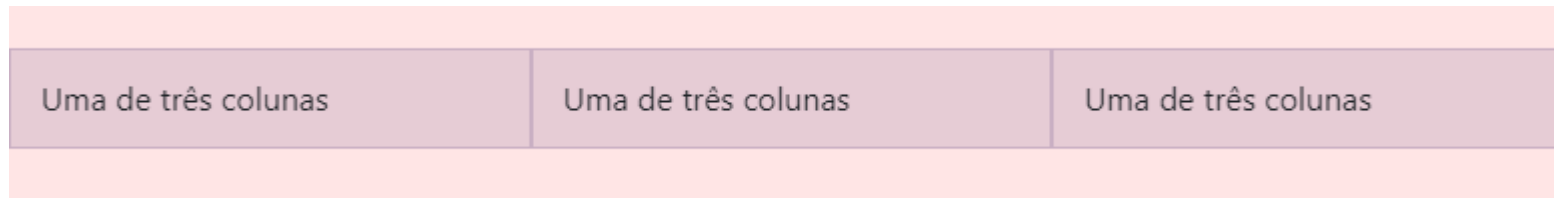
Grid – Alinhamento Vertical de Colunas



Exemplo:

```
<div class="container">  
  <div class="row align-items-start">  
    <div class="col">  
      Uma de três colunas  
    </div>  
    <div class="col">  
      Uma de três colunas  
    </div>  
    <div class="col">  
      Uma de três colunas  
    </div>  
  </div>  
</div>
```

Grid – Alinhamento Vertical de Colunas



Exemplo:

```
<div class="container">  
  <div class="row align-items-center">  
    <div class="col">  
      Uma de três colunas  
    </div>  
    <div class="col">  
      Uma de três colunas  
    </div>  
    <div class="col">  
      Uma de três colunas  
    </div>  
  </div>  
</div>
```

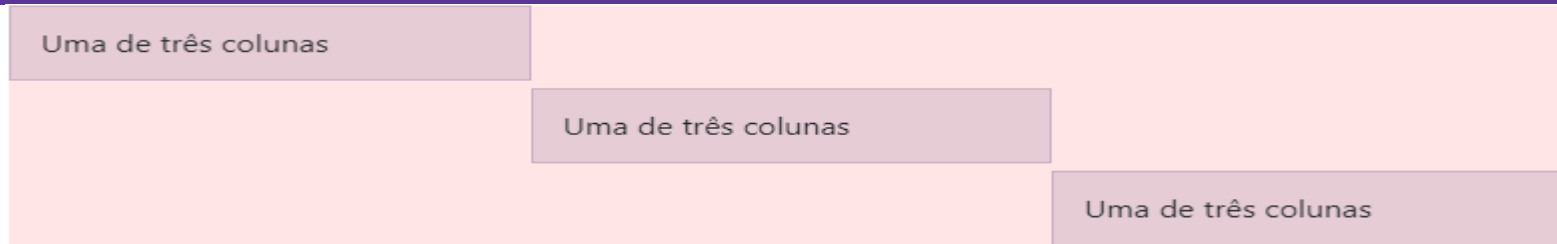
Grid – Alinhamento Vertical de Colunas

Uma de três colunas	Uma de três colunas	Uma de três colunas

Exemplo:

```
<div class="container">  
  <div class="row align-items-end">  
    <div class="col">  
      Uma de três colunas  
    </div>  
    <div class="col">  
      Uma de três colunas  
    </div>  
    <div class="col">  
      Uma de três colunas  
    </div>  
  </div>  
</div>
```

Grid – Alinhamento Vertical de Colunas



Exemplo:

```
<div class="container">  
  <div class="row">  
    <div class="col align-self-start">  
      Uma de três colunas  
    </div>  
    <div class="col align-self-center">  
      Uma de três colunas  
    </div> <div class="col align-self-end">  
      Uma de três colunas  
    </div>  
  </div>  
</div>
```

Grid – Colunas (gutters)

As gutters, entre colunas, podem ser removidas com uma de nossas classes pré-definidas: `.no-gutters`. Isto remove a margem negativa do `.row` e padding horizontal de todas colunas filhas imediatas.

`.col-12 .col-sm-6 .col-md-8`

`.col-6 .col-md-4`

Grid – Columnas (gutters)

Exemplo:

```
<div class="row no-gutters">  
  <div class="col-12 col-sm-6 col-md-8">  
    .col-12 .col-sm-6 .col-md-8  
  </div>  
  <div class="col-6 col-md-4">  
    .col-6 .col-md-4  
  </div>  
</div>
```


Grid – Colunas (Quebra)

Quebrar colunas, em uma nova linha, exige um pequeno hack: adicionar um elemento com `width: 100%`, aonde quer que você queira que suas colunas pulem para uma nova linha. Normalmente, isso é conseguido com múltiplos `.row`, mas isso não se encaixa em toda situação.

.col-6 .col-sm-3	.col-6 .col-sm-3
.col-6 .col-sm-3	.col-6 .col-sm-3

Grid – Colunas (Quebra)

Exemplo:

```
<div class="container">
  <div class="row">
    <div class="col-6 col-sm-3">
      .col-6 .col-sm-3
    </div>
    <div class="col-6 col-sm-3">
      .col-6 .col-sm-3</div>
    <!-- Force as próximas colunas quebrarem, em uma nova linha -->
    <div class="w-100"></div>
    <div class="col-6 col-sm-3">
      .col-6 .col-sm-3
    </div>
    <div class="col-6 col-sm-3">
      .col-6 .col-sm-3
    </div>
  </div>
</div>
```

Grid – Colunas (Reordenamento)

Classes de ordem

Use classes `.order-` para controlar a **ordem visual** de seu conteúdo. Estas classes são responsivas, então, você pode definir a ordem por breakpoint (ex: `.order-1.order-md-2`). Inclui suporte para ordenamento de **1** até **12**, em todos breakpoints do grid.

Primeiro, mas não está ordenado.	Era terceiro, mas é o primeiro ordenado.	Era o segundo, mas é o último ordenado.
----------------------------------	--	---

Grid – Colunas (Reordenamento)

Exemplo:

```
<div class="row no-gutters">  
  <div class="col-12 col-sm-6 col-md-8">  
    .col-12 .col-sm-6 .col-md-8  
  </div>  
  <div class="col-6 col-md-4">  
    .col-6 .col-md-4  
  </div>  
</div>
```

Grid – Colunas (Reordenamento)

Também tem as classes responsivas `.order-first` e `.order-last`, as quais alteram a `order` do elemento, aplicando `order: -1` e `order: 13 (order: $columns + 1)`, respectivamente. Estas classes também podem ser misturadas com as classes numeradas `.order-*`, se necessário.

Era terceiro, mas ordenado como primeiro.	Segundo, mas sem ordenamento.	Era primeiro, mas ordenado como último.
---	-------------------------------	---

Grid – Colunas (Reordenamento)

Exemplo:

```
<div class="container">  
  <div class="row">  
    <div class="col order-last">  
      Era primeiro, mas ordenado como último.  
    </div>  
    <div class="col">  
      Segundo, mas sem ordenamento.  
    </div>  
    <div class="col order-first">  
      Era terceiro, mas ordenado como primeiro.  
    </div>  
  </div>  
</div>
```

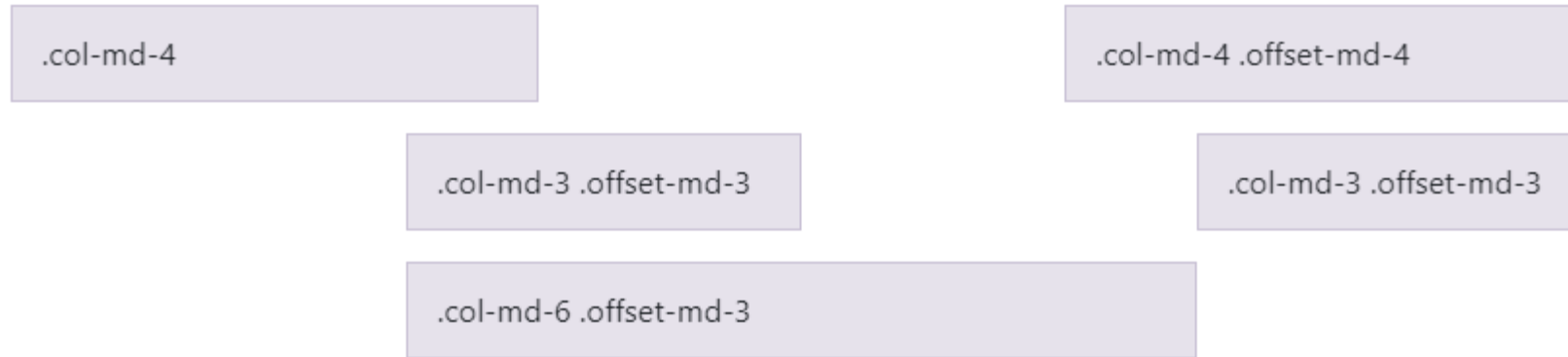
Grid – Colunas (Deslocamento)

Você pode deslocar colunas, de duas maneiras: com nossas classes responsivas `.offset-`. Classes do grid podem ser combinadas para dimensionar colunas, enquanto classes de margem são mais úteis para rápidos layouts, onde a largura do deslocamento é variável.

Classes offset

Mova colunas para a direita, usando classes `.offset-md-*`. Estas classes crescem a margem esquerda da coluna, em `*` colunas. Por exemplo, `.offset-md-4` desloca o elemento `.col-md-4`, por quatro colunas.

Grid – Colunas (Deslocamento)



Exemplo:

```
<div class="container">
  <div class="row">
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>
  </div>
  <div class="row">
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
  </div>
  <div class="row">
    <div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>
  </div>
</div>
```


Grid – Colunas (Aninhamento)

Para aninhar seu conteúdo no grid padrão, coloque um `.row` com um conjunto de colunas `.col-sm-*`, dentro de uma coluna `.col-sm-*` existente. Rows aninhados devem ter um conjunto de 12 colunas ou menos (não é obrigatório usar todo o espaço disponível para 12 colunas).

Level 1: .col-sm-9	
Level 2: .col-8 .col-sm-6	Level 2: .col-4 .col-sm-6

Grid – Colunas (Aninhamento)

Exemplo:

```
<div class="container">  
  <div class="row">  
    <div class="col-sm-9">  
      Level 1: .col-sm-9  
      <div class="row">  
        <div class="col-8 col-sm-6">  
          Level 2: .col-8 .col-sm-6  
        </div>  
        <div class="col-4 col-sm-6">  
          Level 2: .col-4 .col-sm-6  
        </div>  
      </div>  
    </div>  
  </div>  
</div>
```