Bootstrap 5 - Grid System

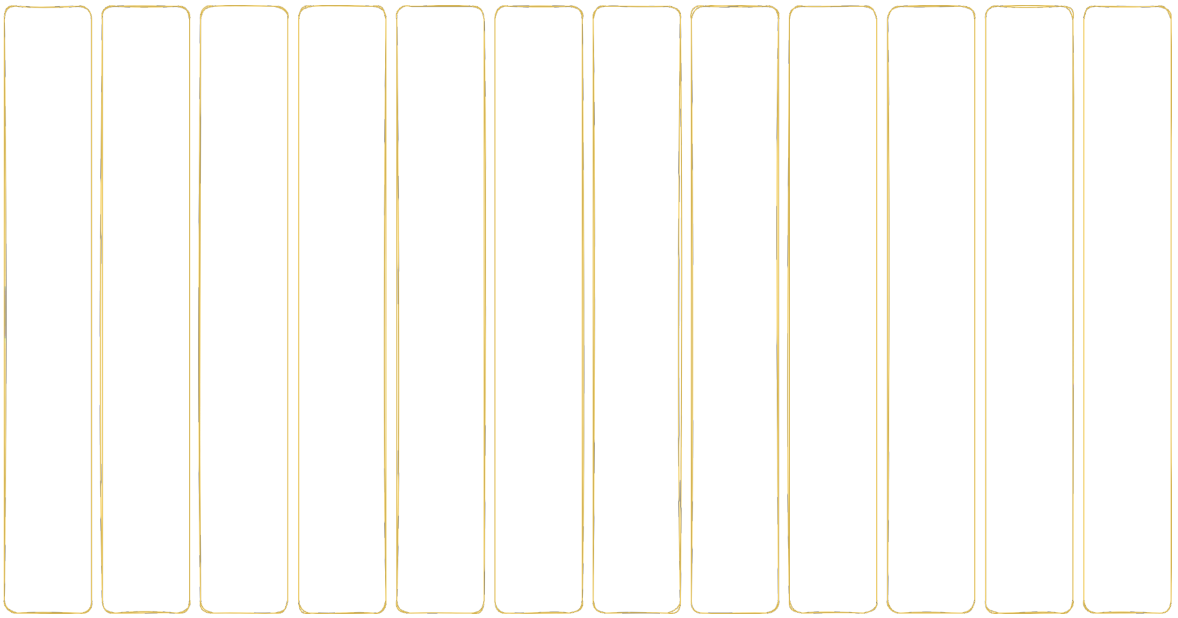
1 Introdução

Nos dias atuais, é fundamental desenvolver páginas que sejam responsivas, o que quer dizer que a forma como o conteúdo é apresentado depende do tamanho da tela do dispositivo sendo utilizado. É possível obter responsividade usando medidas relativas e media queries, por exemplo. Além disso, há frameworks que oferecem mecanismos para a construção de páginas responsivas, além de componentes reutilizáveis. É o caso do Bootstrap, hoje um dos frameworks CSS mais utilizados. Neste tutorial, iremos aprender como funciona seu conhecido **grid system**.

2 Desenvolvimento

2.1 Responsividade (Grid System) O uso de media queries permite que nossas páginas se tornem responsivas. Entretanto, pode ser bastante trabalhoso e repetitivo. Há diversos arcabouços - como o Bootstrap e a PrimeFlex - que nos fornecem a implementação de um recurso chamado de **Grid System**. Um Grid System nos permite tornar nossas páginas responsivas sem ter de lidar diretamente com muitas media queries.

- Um Grid System divide a tela em **colunas**. Um número bastante comum é 12 colunas, mas pode variar:

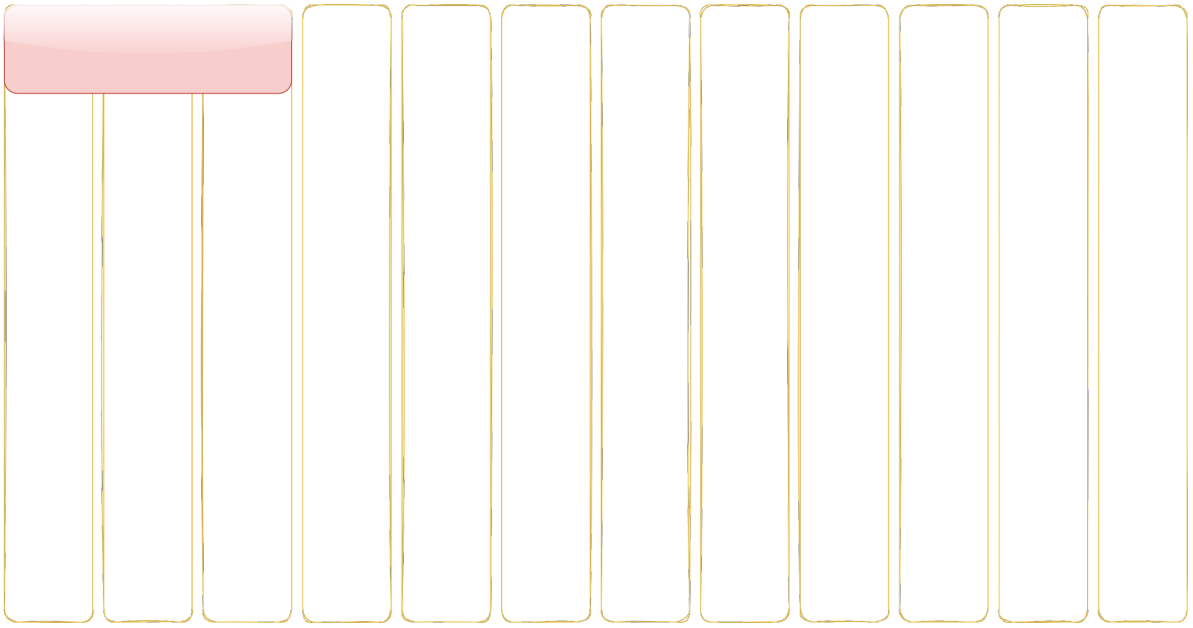


- Há um espaço entre uma coluna e outra. Esse espaço geralmente leva o nome de **gutter**.

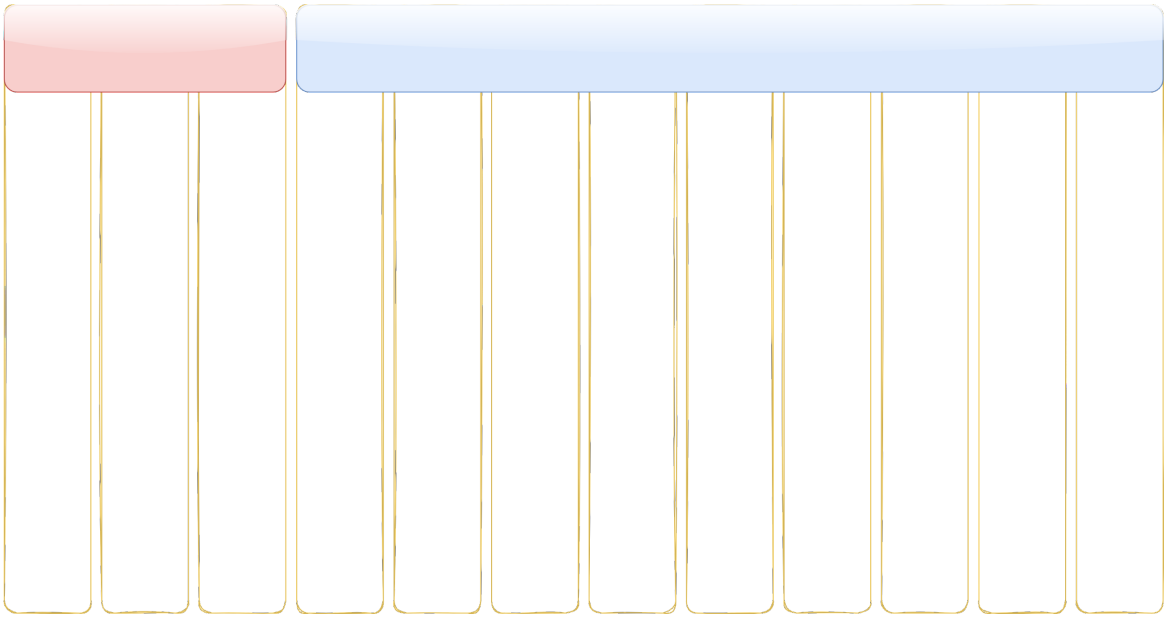
Nota. Uma possível tradução para gutter é “valeta”.

- Para posicionar elementos quaisquer na tela, especificamos o número de colunas que cada um deve ocupar. No exemplo a seguir, a caixa inserida ocupa **três colunas**:

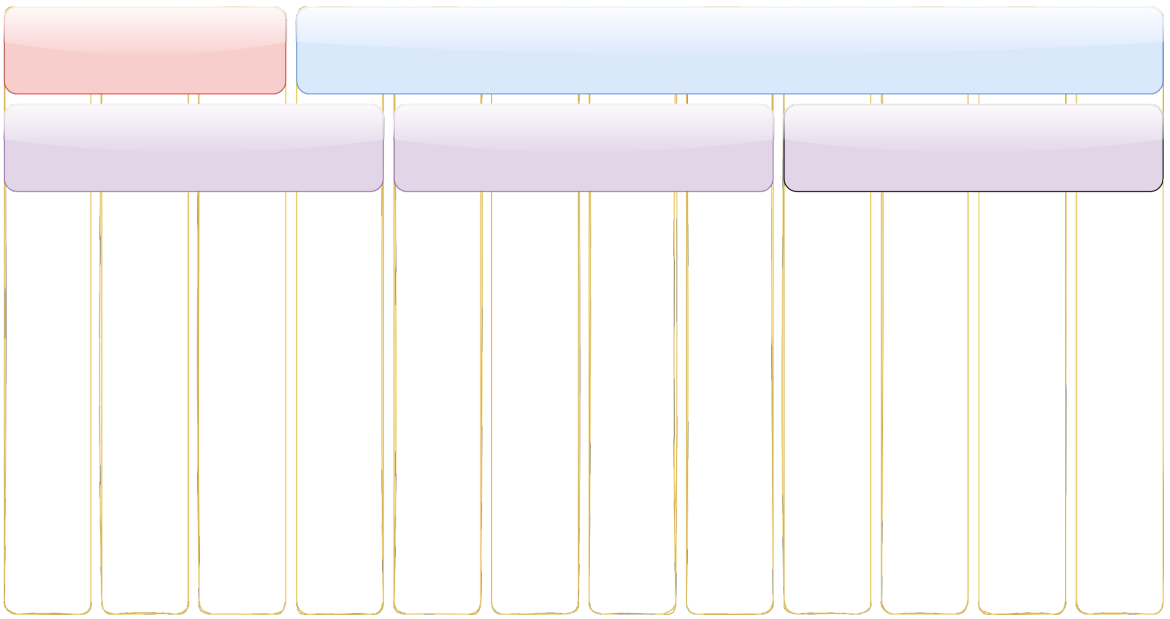
Nota. As cores das colunas e das caixas não têm nenhum significado especial.



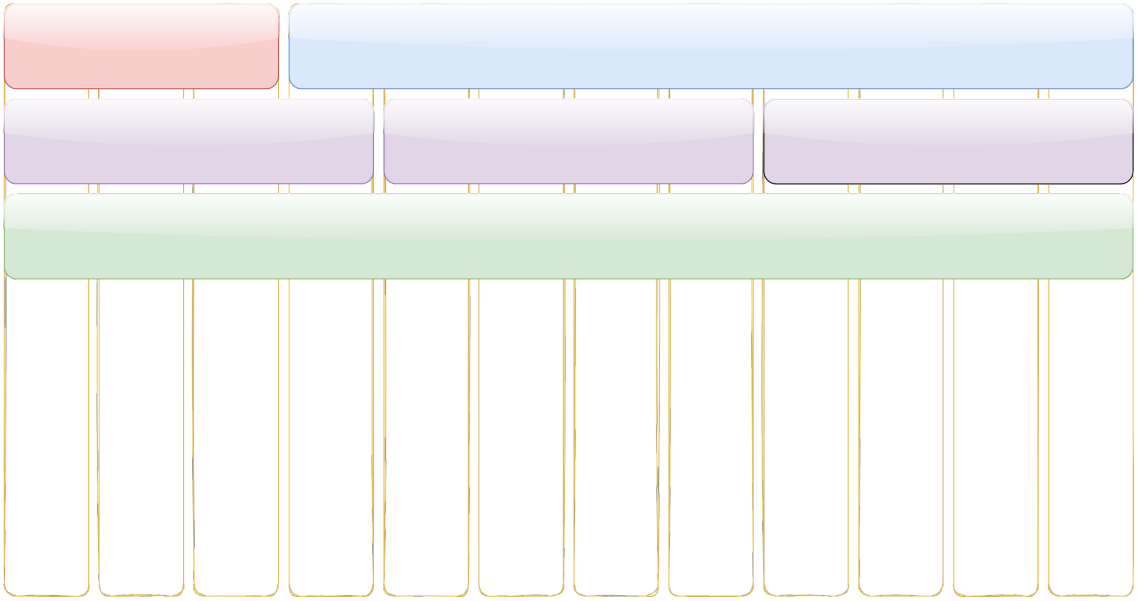
- A seguir, indicamos que a próxima caixa ocupa **nove colunas**. Ele cabe na mesma linha, observe:



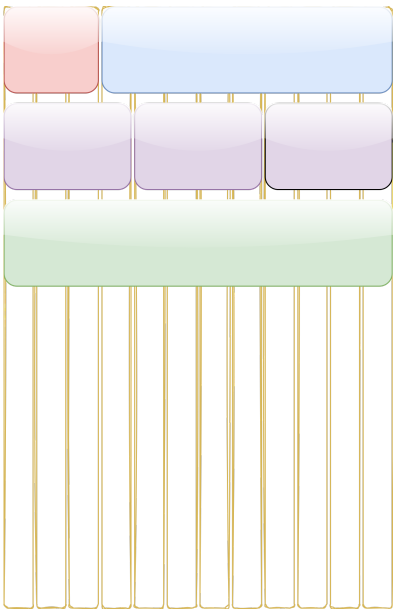
- A seguir, vamos adicionar três caixas de modo que elas ocupem espaço igual. Elas não cabem na primeira linha, portanto vão ser posicionadas logo abaixo. Como queremos espaço igual para três caixas e temos doze colunas, cada uma terá direito a **quatro colunas**:



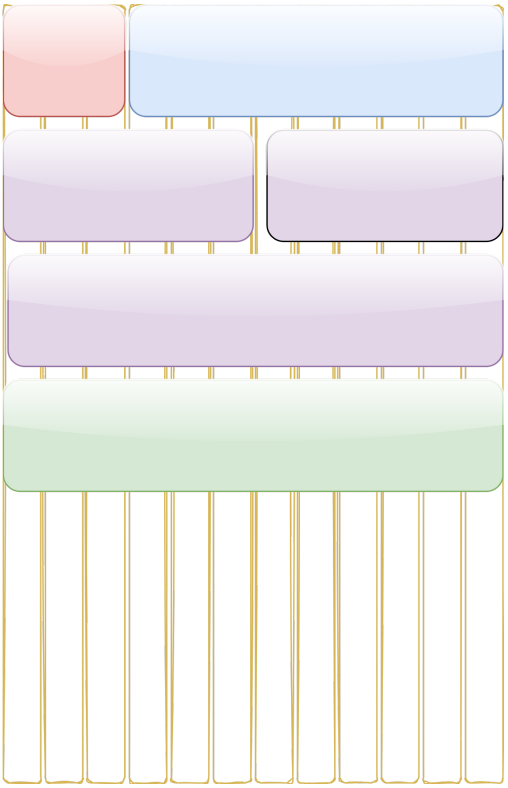
- Digamos que a próxima caixa deve ocupar a tela inteira, horizontalmente. Devemos portanto reservar-lhe **doze colunas**:



- **(E as telas pequenas?)** Nesta sequência de exemplos, estamos supondo que o usuário está utilizando um dispositivo que tenha largura suficiente para que a exibição das caixas lado a lado seja razoável, de modo que ele possa interagir com elas sem que fiquem muito apertadas. O que ocorre, no entanto, se o usuário estiver utilizando uma tela muito pequena? Veja:

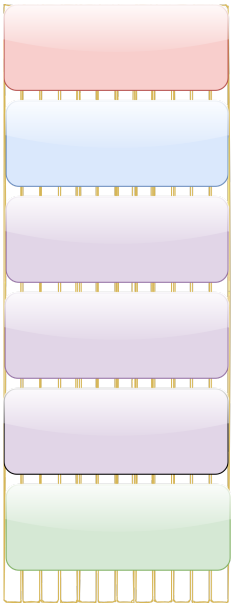


Neste caso, as caixas que estão na mesma linha tendem a ficar muito apertadas e o usuário poderá ter dificuldades para interagir com elas. É aí que entra a utilidade de um Grid System. Ele detecta o uso de telas pequenas (nos blindando do uso direto de media queries) e nos permite especificar quantas colunas cada caixa deve ocupar, agora de maneira condicional, dependendo justamente do tamanho da tela. Neste exemplo, provavelmente, será muito mais interessante empilhar algumas caixas. Talvez da seguinte forma:



Neste caso, optamos por exibir duas das caixas roxas lado a lado, cada uma ocupando metade da tela, ou seja, seis colunas. Para não ficar muito apertado, a outra fica na próxima

linha, ocupando todo o espaço. Mas e se a tela for **menor ainda**? Podemos chegar no caso extremo de reservar doze colunas para cada caixa, fazendo com que todas elas empilhem. Essa organização é melhor para o usuário, pois assim as caixas não ficarão apertadas mesmo em telas muito pequenas. Veja:



2.2 (Novo arquivo) Comece criando um arquivo chamado **gridsystem-bootstrap.html**. Seu conteúdo inicial é dado a seguir.. É o mesmo que pode ser encontrado na documentação oficial, que pode ser visitada por meio do link a seguir:

https://getbootstrap.com/docs/5.2/getting-started/introduction/

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap demo</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+fzT"
crossorigin="anonymous">
  </head>
  <body>
    <h1>Hello, world!</h1>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C8"
crossorigin="anonymous"></script>
  </body>
</html>
```

2.3 (As classes container) As classes “container” do Bootstrap nos permitem determinar qual será a área (horizontalmente) a ser utilizada pela aplicação, em função do tamanho da tela.

Nota. O Bootstrap utiliza nomes específicos para se referir a seus breakpoints. Esses nomes sempre aparecem em suas classes CSS que têm como finalidade fazer algum tipo de manipulação envolvendo a responsividade. São eles:

xs: extra small (menos de 576px)

sm: small (pelo menos 576px)

md: medium (pelo menos 768px)

lg: large (pelo menos 992px)

xl: extra large (pelo menos 1200px)

xxl: extra extra large (pelo menos 1400px)

No que diz respeito às classes “container”, veja seu funcionamento:

	Extra small <576px	**Small** ≥576px	**Medium** ≥768px	**Large** ≥992px	**X-Large** ≥1200px	**XX-Large** ≥1400px
`.container`	100%	540px	720px	960px	1140px	1320px
`.container-sm`	100%	540px	720px	960px	1140px	1320px
`.container-md`	100%	100%	720px	960px	1140px	1320px
`.container-lg`	100%	100%	100%	960px	1140px	1320px
`.container-xl`	100%	100%	100%	100%	1140px	1320px
`.container-xxl`	100%	100%	100%	100%	100%	1320px
`.container-fluid`	100%	100%	100%	100%	100%	100%

Nota. A ideia por trás da definição destas classes é muito simples:

- Queremos controlar quanto de largura utilizar em função do tamanho da tela
-
- Se ela for muito pequena, queremos ocupar 100%
-

- Caso contrário, queremos estabelecer um número de pixels fixo, em função do tamanho da tela. Assim, o conteúdo fica centralizado e sobra espaço para margens laterais.
-
- A classe container-sm nos permite dizer que queremos deixar de ocupar 100% da tela a partir do breakpoint sm. A classe container-md nos permite dizer que queremos deixar de ocupar 100% da tela a partir do breakpoint md. E assim por diante.

Nota. Observe que as classes **container** e **container-sm** têm exatamente o mesmo comportamento. Ocorre que em versões anteriores à versão 4.4 do Bootstrap, somente existiam as classes container e container-fluid. As outras opções foram adicionadas à versão 4.4. Assim, embora tenha o mesmo funcionamento de container, a classe container-sm foi também adicionada, já que é natural esperar pela sua existência, dada a existência das demais.

Veja os exemplos a seguir.

Nota: A classe **mt-4** coloca 4 unidades de medida como **margin top**. A classe **bg-primary** troca a cor de fundo da caixa para que ela seja aquela associada ao nome primary do tema do Bootstrap que estamos utilizando.

```
<!doctype html>
<html lang="en">

<head>
 <meta charset="utf-8">
 <meta name="viewport" content="width=device-width, initial-scale=1">
 <title>Bootstrap demo</title>
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
rel="stylesheet"
   integrity="sha384-iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+fzT"
crossorigin="anonymous">
</head>

<body>
 <div class="container bg-primary mt-4">
   container
 </div>

 <div class="container-fluid bg-primary mt-4">
   container-fluid
 </div>

 <div class="container-sm bg-primary mt-4">
   container-sm
 </div>
 <div class="container-md bg-primary mt-4">
   container-md
 </div>
 <div class="container-lg bg-primary mt-4">
   container-lg
 </div>
```

```
 <div class="container-xl bg-primary mt-4">
    container-xl
 </div>

 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"
   integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C8"
   crossorigin="anonymous"></script>
</body>
</html>
```

Faça testes ajustando a largura do navegador. Veja o que acontece quando a tela tem menos de 576px:

Para telas extra pequenas, todas as classes aplicam 100% de width.

Caso a tela tenha pelo menos 576px, ela passa a ser considerada sm(small). As classes container e container-sm alteram. Repare como elas já não ocupam 100% da tela. Segundo a tabela, estão ocupando 540px.

Para telas que tenham pelo menos 768px, a classe container-md também passa a operar:

E assim por diante.

2.4 (Colunas e linhas) O Bootstrap usa um modelo baseado em **12 colunas**. Ao colocar um elemento em um container, podemos decidir quantas colunas desejamos que ele ocupe. Assim, elementos podem se ajustar lado a lado. Elementos filhos de um elemento que tenha a classe row aplicada “disputam” espaço horizontalmente Observe que **removemos o conteúdo anterior para simplificar os testes.**

Nota. Veja o significado das classes utilizadas:

mt-2: duas unidades de medida de margin top

p-5: 5 unidades de medida de padding

border: adição de uma borda simples

. Veja os exemplos das listagens a seguir. Dê bastante atenção aos **comentários no código**.

```
...
<body>
 <div class="container mt-2 p-5 border">

   <!-- uma linha com um único elemento ocupando as 12 colunas -->
   <div class="row">
     <div class="col-12 border">
       12 colunas
     </div>
   </div>
   <!-- uma linha com um único elemento ocupando 2 colunas -->
   <div class="row">
     <div class="col-2 mt-2 border">
       2
     </div>
   </div>
 </div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body>
```

```
...
<body>
 <div class="container mt-2 p-5 border">

   <!-- uma linha com um único elemento ocupando as 12 colunas -->
   <div class="row">
     <div class="col-12 border">
       12 colunas
     </div>
   </div>
   <!-- uma linha com um único elemento ocupando 2 colunas -->
   <div class="row">
     <div class="col-2 mt-2 border">
       2
     </div>
```

```
   </div>
 </div>

 <div class="row">
   <!-- aqui temos dois elementos div disputando espaço na mesma row -->
   <!-- se a tela for pequena, cada um requer 12 colunas, causando empilhando -->
   <!-- se a tela for pelo menos md, cada uma requer apenas 6 colunas, ficando
portanto uma do lado da outra -->
   <div class="col-sm-12 col-md-6 mt-2 border">
      sm: 12, md: 6
   </div>
   <div class="col-sm-12 col-md-6 mt-2 border">
      sm: 12, md: 6
   </div>
 </div>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body>
```

```
…
<body>
 <div class="container mt-2 p-5 border">

   <!-- uma linha com um único elemento ocupando as 12 colunas -->
   <div class="row">
    …
   </div>
    <div class="row">
    …
    </div>
    <div class="row">
   <!-- três elementos disputando espaço na mesma linha -->
   <!-- todos requerem 12 colunas se a tela for pelo menos sm (isso já é padrão,
mesmo para telas xs) -->
   <!-- todos requerem 4 colunas caso a tela seja pelo menos md -->
   <!-- se a tela for pelo menos lg, 5 colunas para o primeiro elemento, 4
colunas para o segundo e três para o terceiro -->
       <div class="col-sm-12 col-md-4 col-lg-5 mt-2 border">
           sm:12, md: 4, lg: 5
       </div>
       <div class="col-sm-12 col-md-4 col-lg-4 mt-2 border">
```

```
            sm:12, md: 4, lg: 4
        </div>
        <div class="col-sm-12 col-md-4 col-lg-3 mt-2 border">
            sm:12, md: 4, lg: 3
        </div>
    </div>
 </div>


 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body>
```

(Largura igual, sem empilhar) Pode ser de interesse que elementos de uma mesma linha possuam largura igual e que eles não empilhem mesmo em telas pequenas. Para obter esse efeito, usamos a classe **col**. Mais uma vez, apagamos o conteúdo anterior mantendo apenas o container. Veja.

```
...
<body>
 <div class="container mt-2 p-5 border">

   <div class="row">
     <div class="col border mt-2">
       largura igual, sem empilhar
     </div>
     <div class="col border mt-2">
       largura igual, sem empilhar
     </div>
     <div class="col border mt-2">
       largura igual, sem empilhar
     </div>
   </div>
 </div>
 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
```

```
</body>
...
```

2.6 (Leiaute automático) Podemos especificar o número de colunas de alguns elementos e deixar de especificar para outros. O Bootstrap irá atribuir o número que resta para chegar a 12 àquele que não tiver o número definido explicitamente.

```
<body>
 <div class="container mt-2 p-5 border">

   <div class="row">
     <div class="col border mt-2">
       auto
     </div>
     <div class="col-5 border mt-2">
       5
     </div>
     <div class="col-4 border mt-2">
       4
     </div>
   </div>

 </div>

 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body>
```

2.7 (Largura igual com empilhamento até um breakpoint desejado) No modelo do Bootstrap, cada elemento ocupa, por padrão, as 12 colunas, considerando a metodologia “mobile first”. Assim, eles empilham, ainda que pertençam à mesma linha. Podemos especificar o tamanho a partir do qual desejamos que os elementos deixem de ser empilhados. Quando não especificamos um número de colunas, fica implícito que a distribuição deve ser igual.

```
...
<body>
 <div class="container mt-2 p-5 border">

   <div class="row">
     <div class="col-md border mt-2">
       md
     </div>
     <div class="col-md border mt-2">
       md
     </div>
     <div class="col-md border mt-2">
       md
     </div>
   </div>
 </div>

 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body>
....
```

2.8 (Ordenação) As classes com prefixo **order-** permitem especificar a posição em que desejamos que um elemento seja exibido.

```
...
<body>
 <div class="container mt-2 p-5 border">

    <div class="row">
      <div class="col-4 border mt-2 order-2">
      sou o 1
      </div>
      <div class="col-4 border mt-2 order-3">
      Sou o 2
      </div>
      <div class="col-4 border mt-2 order-1">
      Sou o 3
      </div>
    </div>

 </div>

 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body>


...
```

2.9 (Deslocamento com offset) Colunas podem ser deslocadas dentro de uma linha com as classes offset. Especificamos a medida a partir da qual o deslocamento deve ocorrer e o número de colunas de deslocamento.

```
...
<body>
 <div class="container mt-2 p-5 border">

   <div class="row">
     <div class="col-md-6 offset-md-3 border my-2">
       col: 6, off: 3
     </div>
   </div>

   <div class="row">
     <div class="col-md-4 offset-md-2 border my-2">
       col:4, off:2
     </div>
     <div class="col-md-3 offset-md-3 border my-2">
       col:3, off:3
     </div>
   </div>

 </div>

 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body
...
```

2.10 (Alinhamento com align-items (eixo perpendicular ao principal)) O **display flex**, usado por padrão pelas classes row do Bootstrap, define dois eixos:

- o principal e o
-
- perpendicular ao principal.
-
- Usando a propriedade **flex-direction**, podemos especificar qual eixo desejamos que seja o principal. Os valores possíveis são **row** e **column**.
-
- O valor padrão desta propriedade é **row**, o que quer dizer que o eixo principal é horizontal.
-
- A propriedade **align-items** tem efeito no eixo perpendicular ao principal.

```
<body>
 <div class="container mt-2 p-5 border">

   <div class="row align-items-start border bg-light" style="height:200px">
     <div class="col border">
       top (start de cima para baixo)
     </div>
     <div class="col border">
       top (start de cima para baixo)
     </div>
   </div>

   <div class="row align-items-center border bg-light my-2" style="height:200px">
     <div class="col border">center</div>
     <div class="col border">center</div>
   </div>

   <div class="row align-items-end border bg-light my-2" style="height:200px">
     <div class="col border">bottom</div>
     <div class="col border">bottom</div>
   </div>
 </div>
 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script></body>
```

2.11 (Alinhamento no eixo perpendicular para elementos específicos) A propriedade **align-items** do display flex se aplica a **todos os filhos de um container**.

Caso seja necessário utilizá-la para filhos específicos, utiliza-se a propriedade **align-self**.

```
<body>
 <div class="container mt-2 p-5 border">

   <div class="row border my-2 align-items-center" style="height:200px">
     <div class="col border align-self-start">top</div>
     <div class="col border align-self-center">center</div>
     <div class="col border align-self-end">bottom</div>
   </div>
 </div>

 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
   crossorigin="anonymous"></script>
</body>

</html>
```

2.12 (Alinhamento no eixo principal) A propriedade **justify-content** do display flex se refere ao eixo principal. Como o valor **row** é o padrão, o eixo principal é, por padrão, horizontal. Assim a propriedade **justify-content** opera, a princípio, horizontalmente. Caso alteremos o eixo principal, ela passa a operar verticalmente.

```
<body>
 <div class="container mt-2 p-5 border">
   <div class="row justify-content-start my-2">
     <div class="col-4 border">start</div>
     <div class="col-4 border">start</div>
   </div>

   <div class="row justify-content-center my-2">
     <div class="col-4 border">center</div>
     <div class="col-4 border">center</div>
   </div>

   <div class="row justify-content-end my-2">
     <div class="col-4 border">end</div>
     <div class="col-4 border">end</div>
   </div>
   <div class="row justify-content-around my-2">
```

```
      <div class="col-4 border">around</div>
      <div class="col-4 border">around</div>
    </div>

    <div class="row justify-content-between my-2">
      <div class="col-4 border">between</div>
      <div class="col-4 border">between</div>
    </div>
  </div>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
    crossorigin="anonymous"></script>

</body>
```

2.13 (Alterando eixos) A propriedade **flex-direction** do modelo flex permite a alteração do eixo principal. Valores possíveis são **row, row-reverse, column** e **column-reverse**.

```
<body>
  <div class="container mt-2 p-5 border">

    <div class="row">
      <!-- d-flex: display: flex; -->
      <!-- padrão row da esquerda para a direita -->
      <div class="d-flex bg-light border my-2">
        <div class="p-4 border mx-2">1</div>
        <div class="p-4 border mx-2">2</div>
        <div class="p-4 border mx-2">3</div>
      </div>

      <!-- flex-row-reverse da direita para a esquerda -->
      <div class="d-flex flex-row-reverse bg-light border my-2">
        <div class="p-4 border mx-2">1</div>
        <div class="p-4 border mx-2">2</div>
        <div class="p-4 border mx-2">3</div>
      </div>

      <!-- flex-column: eixo principal se torna o vertical -->
      <!-- posicionamento de cima para baixo -->
      <div class="d-flex flex-column bg-light border my-2">
        <div class="p-4 border mx-2">1</div>
        <div class="p-4 border mx-2">2</div>
        <div class="p-4 border mx-2">3</div>
```

```
      </div>

      <!-- flex-column-reverse: de baixo para cima -->
      <div class="d-flex flex-column-reverse bg-light border my-2">
        <div class="p-4 border mx-2">1</div>
        <div class="p-4 border mx-2">2</div>
        <div class="p-4 border mx-2">3</div>
      </div>
    </div>
  </div>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js
"

integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C
8"
    crossorigin="anonymous"></script>
</body>
```

Referências

Bootstrap · The most popular HTML, CSS, and JS library in the world., 2022. Disponível em: https://getbootstrap.com/. Acesso em setembro de 2022.