

IOS – Instituto de
Oportunidade Social

Aula CSS 11 - Grid CSS Layout - Parte 1



Conteúdo

- > O que é Grid.
- > Termos utilizados pelo Grid
- > Propriedades do elemento pai (Grid Container).

IOS – Instituto de
Oportunidade Social

O que é Grid



O que é Grid

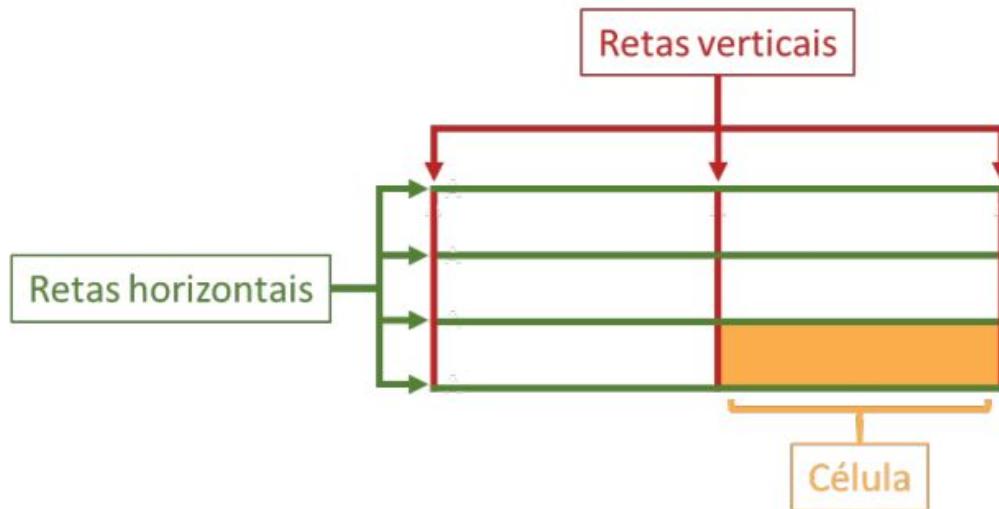


Você tem usado a propriedade **CSS Float** e o **CSS Flexible Box Layout** para criar a aparência de páginas web com várias colunas, porém existe uma técnica bastante eficiente para isso: o **CSS Grid Layout**.

O CSS Grid Layout é uma excelente forma de **dividir uma página em regiões principais ou definir o relacionamento em termos de medidas, posicionamento e camadas entre os diferentes componentes da marcação HTML**.

O que é Grid

Você deve entender o grid como um layout formado por **retas verticais e horizontais**, que **formam as células do grid**. Por exemplo, a imagem abaixo mostra um grid **3x2 (3 linhas e 2 colunas)** formado por três retas verticais e quatro retas horizontais, formando assim seis células.



- Tabelas bidimensionais
- Vantagens sobre tabelas

IOS – Instituto de
Oportunidade Social

Termos utilizados pelo Grid



Termos utilizados pelo Grid



A proposta do Grid é definir o layout bidimensional de uma página web criando itens fixos ou flexíveis que podem ser configurados individualmente. Como tabelas, o layout Grid permite alinhar elementos em linha e colunas por meio de um método bidimensional para criação de layout utilizando CSS.

O **Grid tem vantagem** sobre tabelas por **não se basear em estruturação de conteúdo**. Possibilita uma **enorme variedade de layouts**. O Grid layout é o primeiro módulo no CSS criado especificamente para resolver problemas de layout.

Termos utilizados pelo Grid



Grid Container: é o elemento ou classe que definimos a propriedade `display: grid`. Então, esse elemento será o pai de todos os itens do Grid. Por exemplo, podemos configurar uma classe `.container` em um elemento `<section>` e vários itens dentro dessa seção.

Termos

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

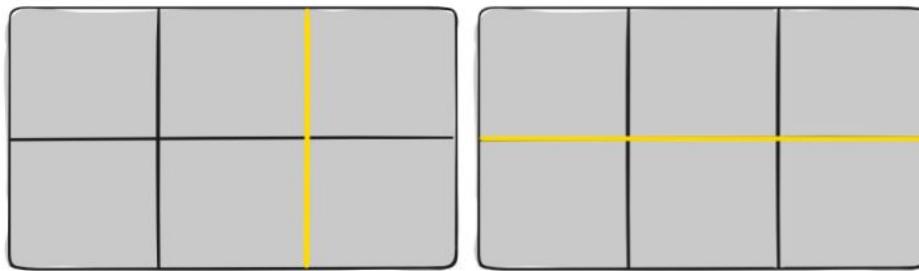
- **display: grid**
- Você sempre precisa definir um container para trabalhar com **Grid layout**

Termos utilizados pelo Grid



Grid Item: são os filhos do grid container.

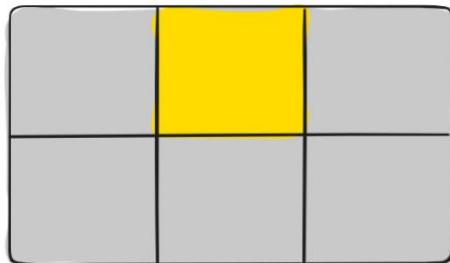
Grid Line: são as linhas divisórias que formam a estrutura do grid. Elas podem ser as retas verticais ou horizontais e estar em qualquer lado da linha ou coluna.



Termos utilizados pelo Grid



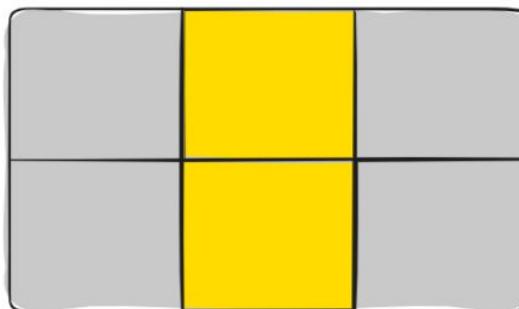
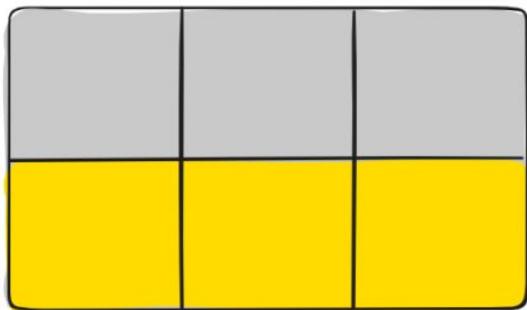
Grid Cell: é a célula do grid (unidade única do grid), ou seja, o espaço entre duas retas horizontais ou verticais adjacentes. Abaixo é mostrado o Grid Cell entre as retas 1 e 2 horizontais e as retas 2 e 3 verticais.



Termos utilizados pelo Grid



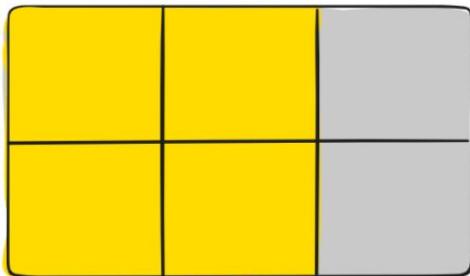
Grid Track: é uma trilha do grid, que pode ser uma linha ou uma coluna inteira.



Termos utilizados pelo Grid



Grid Area: O espaço total rodeado por quatro retas do grid. Grid Area pode ser composta de qualquer número de células do grid. Por exemplo, abaixo a Grid Area está entre as retas horizontais 1 e 3 e as retas verticais 1 e 3.



IOS – Instituto de
Oportunidade Social

Elemento pai (Grid Container)



Elemento pai (Grid Container)

- display
- grid-template-columns
- grid-template-rows
- grid-template-areas
- grid-template
- column-gap
- row-gap
- grid-column-gap
- grid-row-gap
- gap
- grid-gap
- justify-items
- align-items
- place-items
- justify-content
- align-content
- place-content
- grid-auto-columns
- grid-auto-rows
- grid-auto-flow
- grid

Elementos filhos (Grid Items)



> Propriedade display

- grid: gera um block grid
- inline-grid: gera um grid alinhado

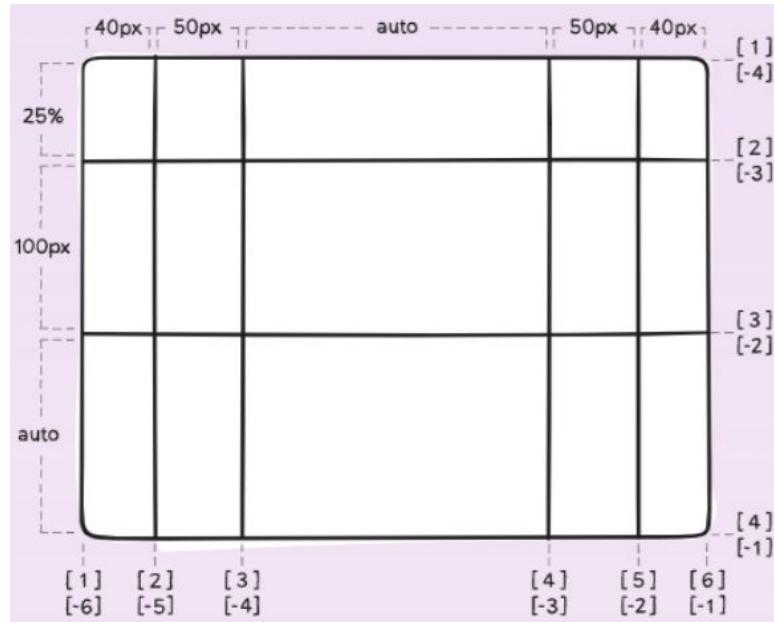
```
.container {  
    display: grid;  
}
```

```
.container {  
    display: inline-grid;  
}
```

Elementos filhos (Grid Items)

> grid-template-columns / grid-template-rows

```
.container {  
    grid-template-columns: 40px 50px auto 50px 40px;  
    grid-template-rows: 25% 100px auto;  
}
```



Elementos filhos (Grid Items)



> grid-template-columns / grid-template-rows

```
.container {  
    grid-template-columns: [first] 40px [line2] 50px [line3] auto [col4-start] 50px  
    [five] 40px [end];  
    grid-template-rows: [row1-start] 25% [row1-end] 100px [third-line] auto [last-line];  
}
```

```
.container {  
    grid-template-rows: [row1-start] 25% [row1-end row2-start] 25% [row2-end];  
}
```

Elementos filhos (Grid Items)



Caso você queira dividir em partes iguais, pode usar a notação **repeat()**. Por exemplo:

```
.container {  
    grid-template-columns: repeat(3, 20px [col-start]);  
}  
  
.container {  
    grid-template-columns: repeat(4, 1fr);  
}
```

Elementos filhos (Grid Items)



É equivalente a:

```
.container {  
    grid-template-columns: 20px [col-start] 20px [col-start] 20px [col-start];  
}  
  
.container {  
    grid-template-columns: 1fr 1fr 1fr 1fr;  
}
```

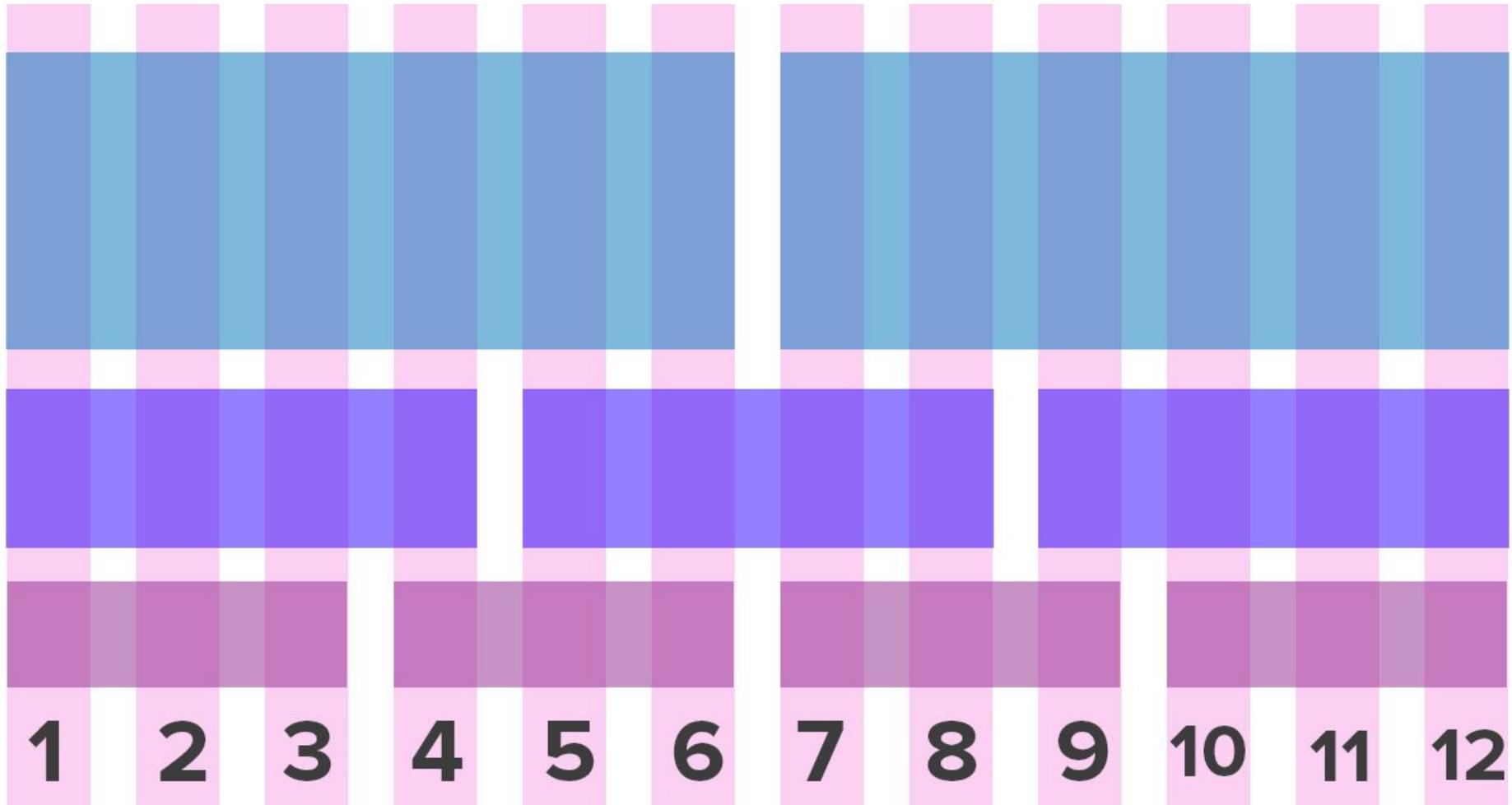
Elementos filhos (Grid Items)



> grid-template-areas

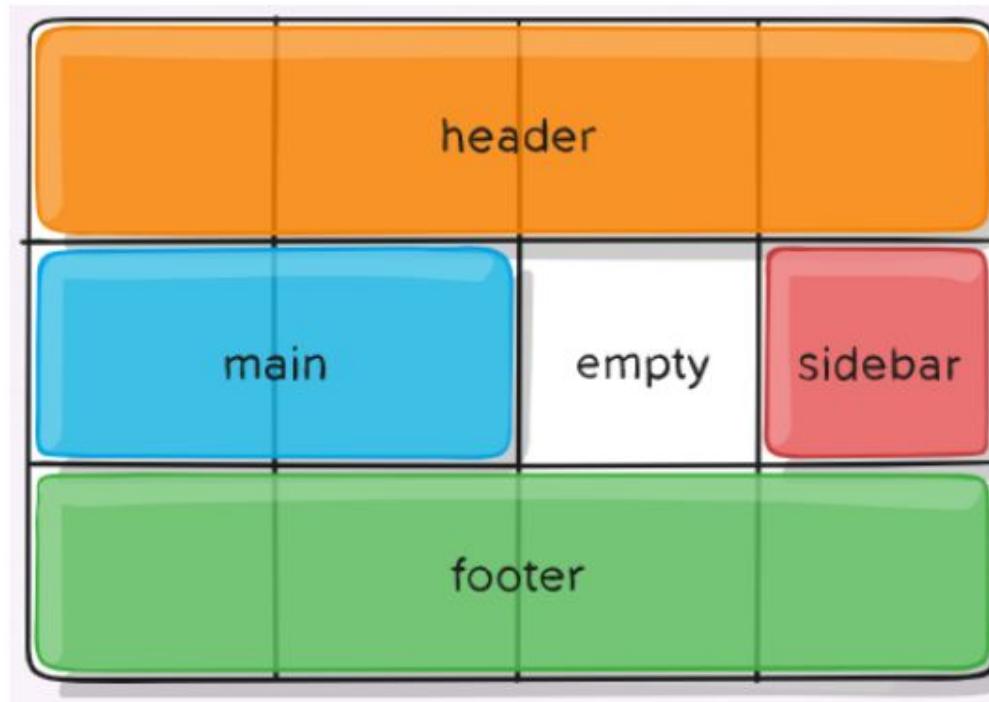
```
.item-a {  
    grid-area: header;  
}  
  
.item-b {  
    grid-area: main;  
}  
  
.item-c {  
    grid-area: sidebar;  
}  
  
.item-d {  
    grid-area: footer;  
}  
  
.container {  
    display: grid;  
    grid-template-columns: repeat(4, 1fr);  
    grid-template-rows: auto;  
    grid-template-  
    areas: "header header header header" "main main . sidebar" "footer footer footer footer"  
}
```

Elementos filhos (Grid Items)



Elementos filhos (Grid Items)

> grid-template-areas



Elementos filhos (Grid Items)



> grid-template

```
.container {  
    grid-template: [row1-start] "header header header" 25px [row1-end] [row2-  
start] "footer footer footer" 25px [row2-end] / auto 50px auto;  
}
```

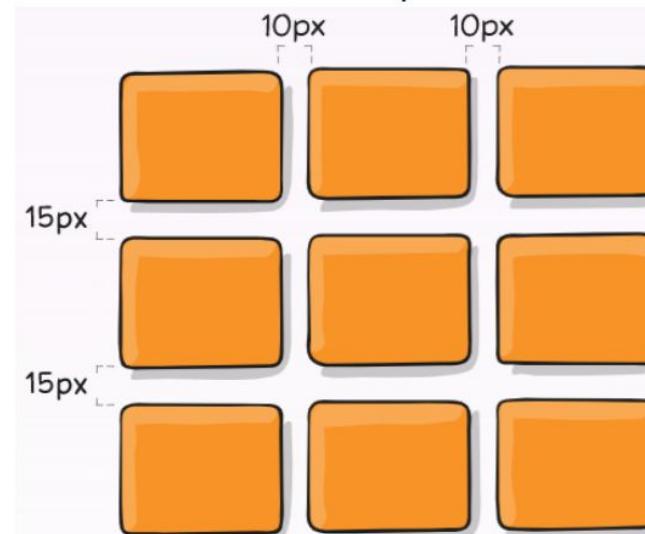
Esse exemplo seria equivalente a:

```
.container {  
    grid-template-rows: [row1-start] 25px [row1-end row2-start] 25px [row2-end];  
    grid-template-columns: auto 50px auto;  
    grid-template-areas: "header header header" "footer footer footer";  
}
```

Elementos filhos (Grid Items)

> column-gap / row-gap / grid-column-gap / grid-row-gap

```
.container {  
    grid-template-columns: repeat(3, 1fr);  
    grid-template-rows: 80px auto 80px;  
    column-gap: 10px;  
    row-gap: 15px;  
}
```



O resultado desse exemplo é mostrado na imagem seguinte:

Elementos filhos (Grid Items)



> gap ou grid-gap

A propriedade grid-gap é uma abreviação das propriedades row-gap e column-gap. Os valores de ser na ordem: primeiro e em segundo . Por exemplo:

```
.container {  
    grid-template-columns: repeat(3, 1fr);  
    grid-template-rows: 80px auto 80px;  
    grid-gap: 15px 10px;  
}
```

IOS – Instituto de
Oportunidade Social

Propriedade justify-items



Propriedade justify-items



A propriedade `justify-items` alinha horizontalmente os itens do grid. Os valores possíveis para configurar essa propriedade são:

start: os itens ficam alinhados na reta esquerda da célula do grid.

```
.container {  
    justify-items: start;  
}
```

Resultado:

Propriedade justify-items

end: os itens ficam alinhados na reta direita da célula do grid.

```
.container {  
    justify-items: end;  
}
```

Resultado:

Propriedade justify-items



center: os itens ficam centralizados horizontalmente na célula do grid.

```
.container {  
    justify-items: center;  
}
```

Resultado:

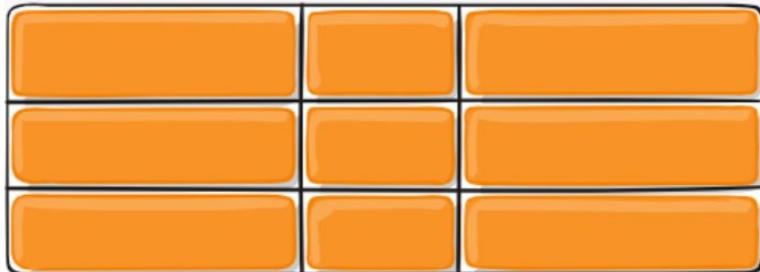
Propriedade justify-items



stretch: os itens preenchem todo o espaço da célula do grid.

```
.container {  
  justify-items: stretch;  
}
```

Resultado:



Esse mesmo comportamento para o alinhamento horizontal dos itens pode ser definido individualmente para cada item do grid com a propriedade **justify-self**.

IOS – Instituto de
Oportunidade Social

Propriedade align-items



Propriedade align-items

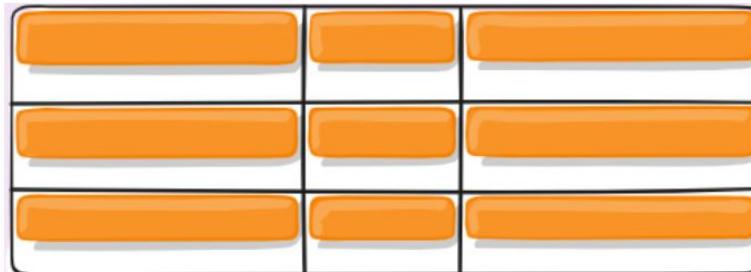


A propriedade align-items alinha verticalmente os itens do grid. Os valores possíveis para configurar essa propriedade são:

start: os itens ficam alinhados na reta superior da célula do grid.

```
.container {  
    align-items: start;  
}
```

Resultado:



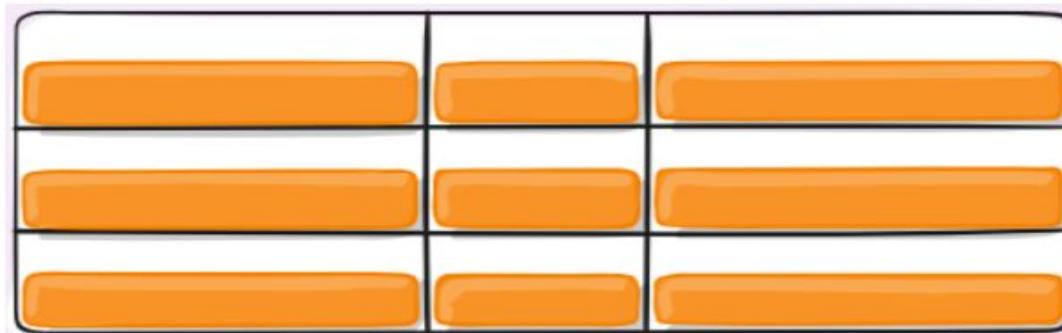
Propriedade align-items



end: os itens ficam alinhados na reta inferior da célula do grid.

```
.container {  
    align-items: end;  
}
```

Resultado:



Propriedade align-items



center: os itens ficam centralizados verticalmente na célula do grid.

```
.container {  
    align-items: center;  
}
```

Resultado:

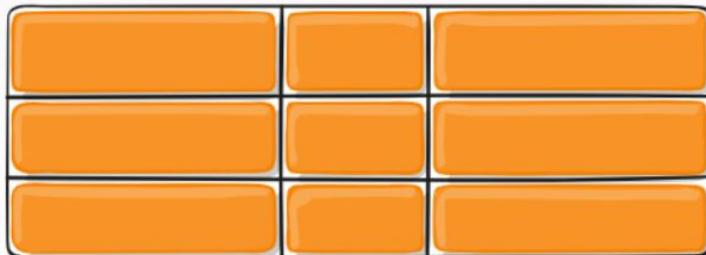
Propriedade align-items



stretch: os itens preenchem todo o espaço da célula do grid.

```
.container {  
    align-items: stretch;  
}
```

Resultado:



Esse mesmo comportamento para o alinhamento vertical dos itens pode ser definido individualmente para cada item do grid com a propriedade **align-self**.

IOS – Instituto de
Oportunidade Social

Propriedade place-items



Propriedade place-items



Propriedade place-items

- A propriedade **place-items** define os dois alinhamentos vertical (**align-items**) e horizontal (**justify-items**) com uma única declaração. Os valores possíveis para configurar essa propriedade são:
 - <align-items> / <justify-items> O primeiro valor define o align-items e o segundo o **justify-items**. Se o segundo valor for omitido, o primeiro valor é atribuído para ambos os alinhamentos.

IOS – Instituto de
Oportunidade Social

Propriedade justify-content



Propriedade justify-content

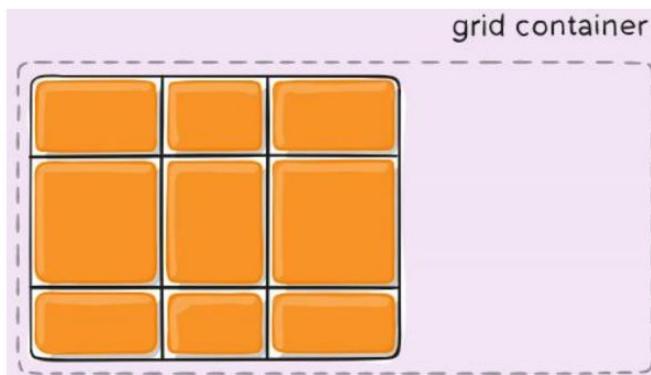


A propriedade `justify-content` alinha o grid completo ao longo do eixo horizontal do container do grid. Os valores possíveis para configurar essa propriedade são:

start: o grid fica alinhado na reta esquerda do container.

```
.container {  
  justify-content: start;  
}
```

Resultado:

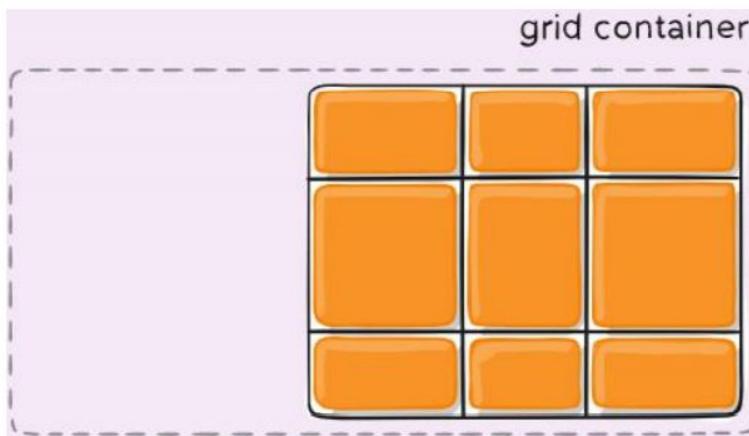


Propriedade justify-content

end: o grid fica alinhado na reta direita do container.

```
.container {  
    justify-content: end;  
}
```

Resultado:

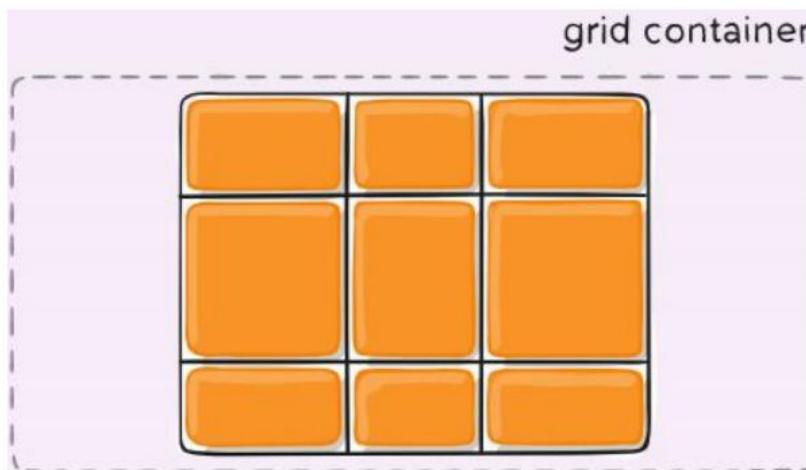


Propriedade justify-content

center: o grid fica centralizado horizontalmente no container.

```
.container {  
    justify-content: center;  
}
```

Resultado:



Propriedade justify-content

stretch: o grid preenche todo o espaço do container.

```
.container {  
    justify-content: stretch;  
}
```

Resultado:



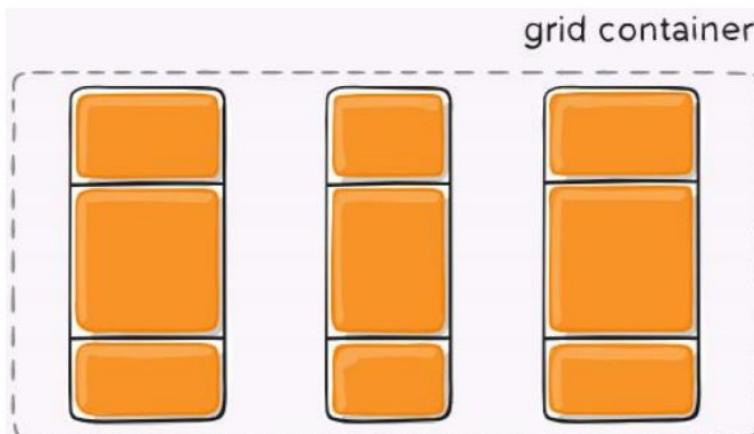
Propriedade justify-content



space-around: coloca uma quantidade uniforme de espaço entre cada item do grid e metade do tamanho do espaço nas extremidades da esquerda e da direita.

```
.container {  
  justify-content: space-around;  
}
```

Resultado:

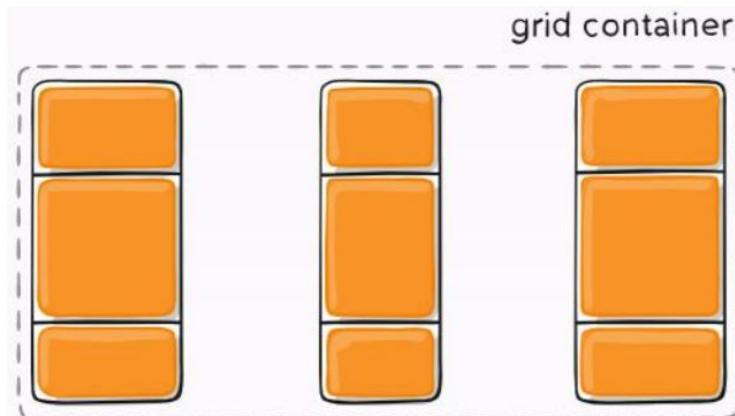


Propriedade justify-content

space-between: coloca uma quantidade uniforme de espaço entre cada item do grid e sem espaço nas extremidades da esquerda e da direita.

```
.container {  
    justify-content: space-between;  
}
```

Resultado:

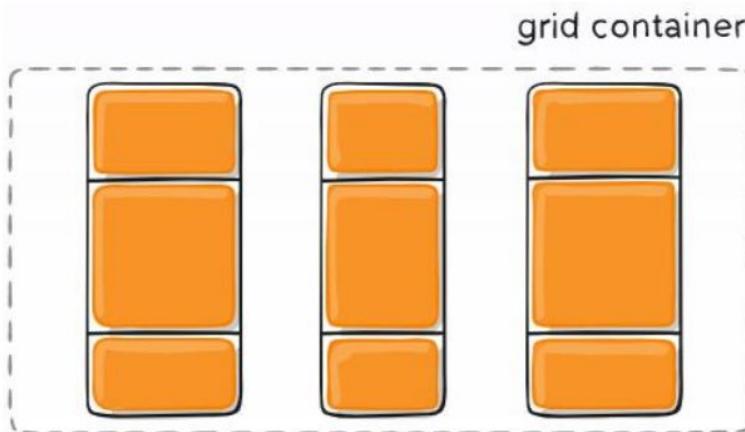


Propriedade justify-content



space-evenly: coloca uma quantidade uniforme de espaço entre cada item do grid, incluindo as extremidades da esquerda e da direita com mesmo tamanho de espaço.

```
.container {  
    justify-content: space-evenly;  
}
```



IOS – Instituto de
Oportunidade Social

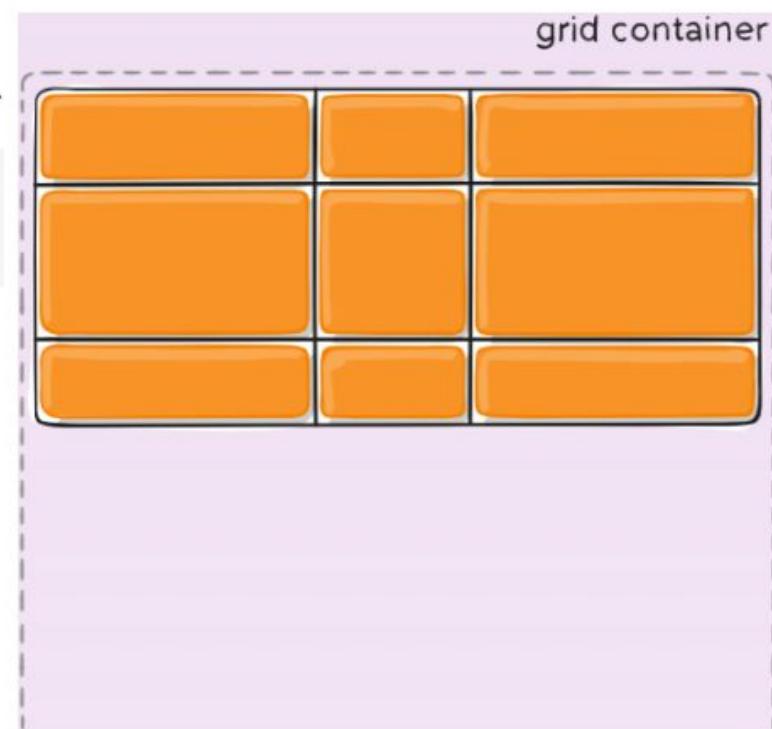
Propriedade align-content



Propriedade align-content

A propriedade align-content alinha o grid completo ao longo do eixo vertical do container do grid. Os valores possíveis para configurar essa propriedade são:

Resultado:



start: o grid fica alinhado na borda superior do container.

```
.container {  
  align-content: start;  
}
```

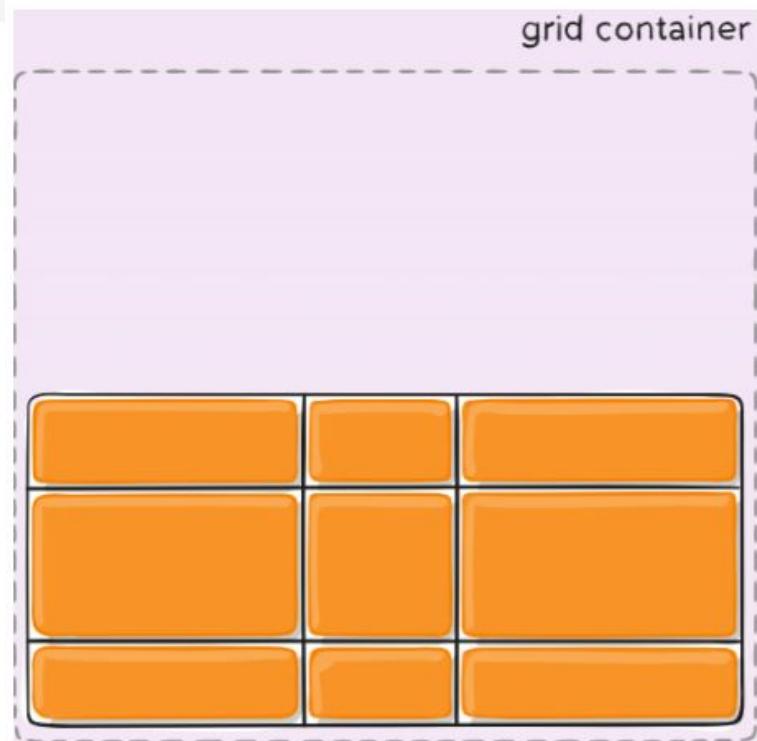
Propriedade align-content



end: o grid fica alinhado na borda inferior do container.

```
.container {  
    align-content: end;  
}
```

Resultado:

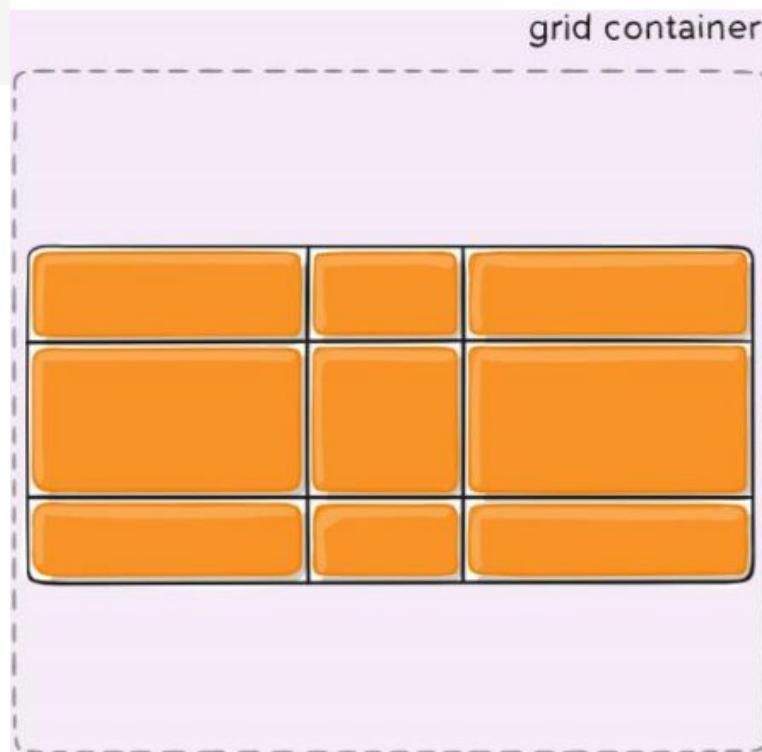


Propriedade align-content

center: o grid fica centralizado verticalmente no container.

```
.container {  
    align-content: center;  
}
```

Resultado:



Propriedade align-content

stretch: o grid preenche todo o espaço do container.

```
.container {  
    align-content: stretch;  
}
```

Resultado:



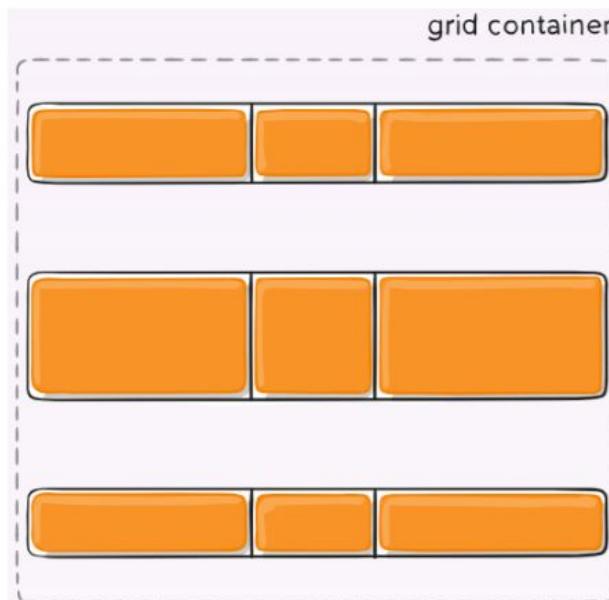
Propriedade align-content



space-around: coloca uma quantidade uniforme de espaço entre cada item do grid e metade do tamanho do espaço nas extremidades superior e inferior.

```
.container {  
    align-content: space-around;  
}
```

Resultado:



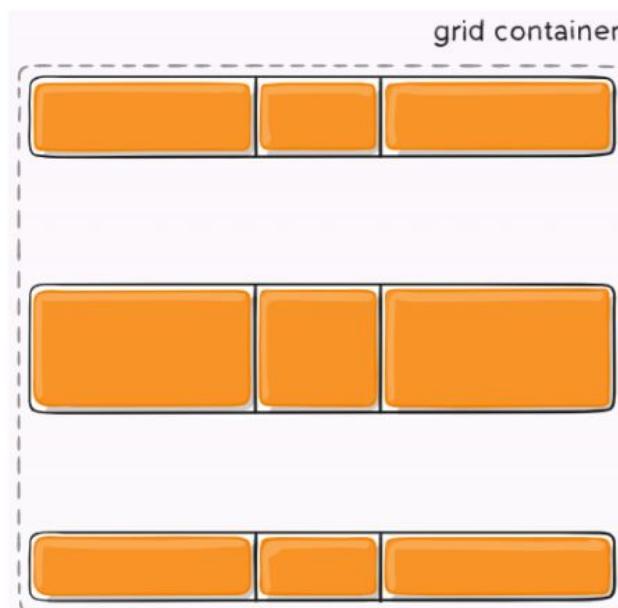
Propriedade align-content



space-between: coloca uma quantidade uniforme de espaço entre cada item do grid e sem espaço nas extremidades superior e inferior.

```
.container {  
    align-content: space-between;  
}
```

Resultados:



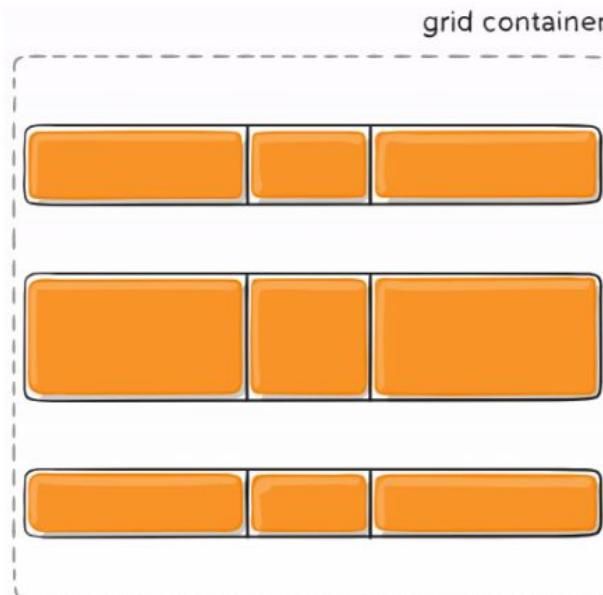
Propriedade align-content



space-evenly: coloca uma quantidade uniforme de espaço entre cada item do grid, incluindo as extremidades superior e inferior com mesmo tamanho de espaço.

```
.container {  
    align-content: space-evenly;  
}
```

Resultado:



IOS – Instituto de
Oportunidade Social

Propriedade place-content



Propriedade place-content



- A propriedade **place-content** define os dois alinhamentos **align-content** e **justify-content** com uma única declaração. Os valores possíveis para configurar essa propriedade são:
- <align-content> / <justify-content>: O primeiro valor define o **align-content** e o segundo o **justify-content**. Se o segundo valor for omitido, o primeiro valor é atribuído para ambos os alinhamentos.

IOS – Instituto de
Oportunidade Social

grid-auto-columns e grid-auto-rows



grid-auto-columns e grid-auto-rows



As propriedades **grid-auto-columns** e **grid-auto-rows** especificam o tamanho de quaisquer trilhas do grids geradas automaticamente, que podem também ser chamadas de trilha implícitas do grid. Trilhas implícitas são criadas quando há mais itens no grid do que células disponíveis ou quando um item do grid é colocado fora da grade explícita. Os valores possíveis para configurar essa propriedade são:

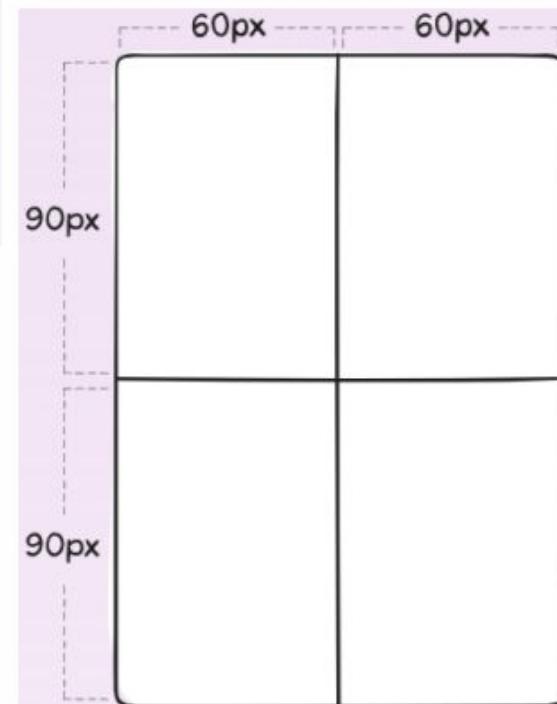
grid-auto-columns e grid-auto-rows



<track-size> : que pode ser um tamanho, uma porcentagem ou uma fração (usando a unidade **fr**) do espaço livre no grid

Resultado:

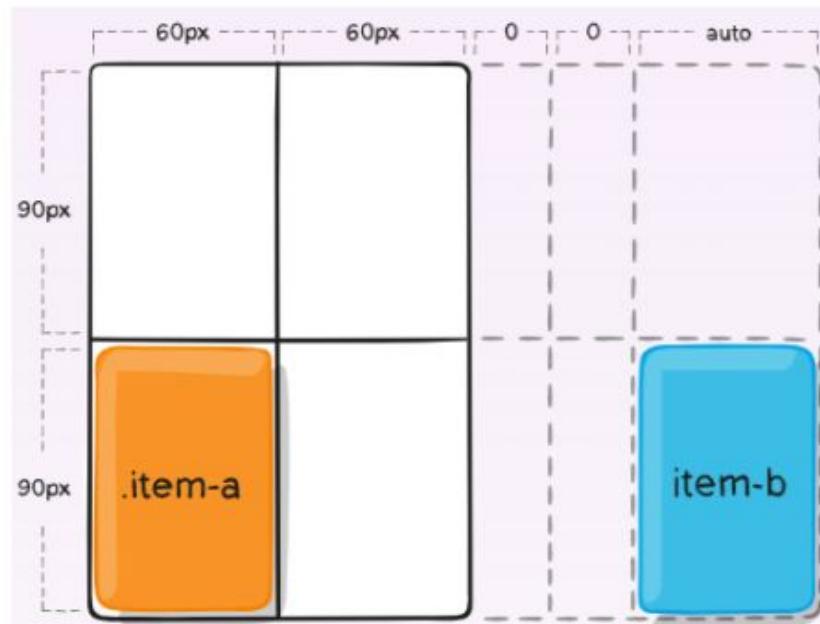
```
.container {  
    grid-template-columns: 60px 60px;  
    grid-template-rows: 90px 90px;  
}
```



grid-auto-columns e grid-auto-rows

Resultado:

```
.item-a {  
  grid-column: 1 / 2;  
  grid-row: 2 / 3;  
}  
  
.item-b {  
  grid-column: 5 / 6;  
  grid-row: 2 / 3;  
}
```



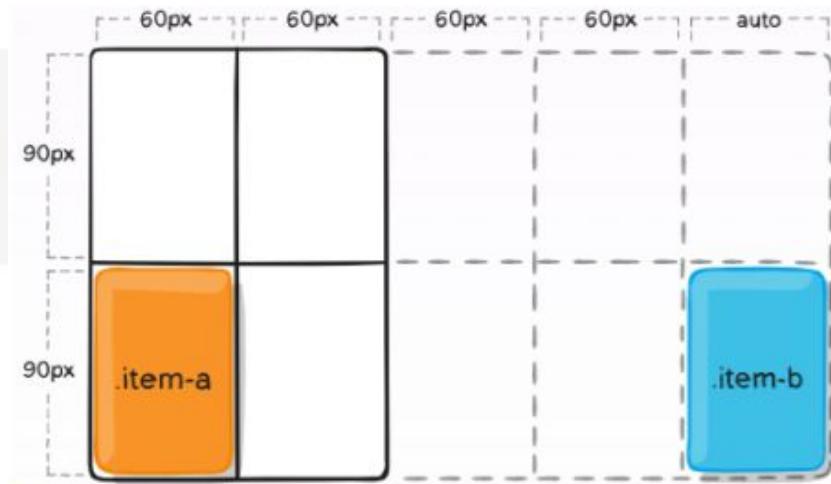
Nesse caso, o item-b começa na reta da coluna 5 e termina na reta coluna 6, mas nós não definimos anteriormente essa parte (nossa grid é 2x2).

grid-auto-columns e grid-auto-rows



Caso quisermos, nós podemos mudar a primeira declaração, quando criamos o grid 2x2 na classe .container para que sejam criadas trilhas implícitas com tamanho especificado, isso é o que as propriedades grid-auto-columns e grid-auto-rows fazem. Por exemplo:

```
.container {  
    grid-auto-columns: 60px;  
}
```



IOS – Instituto de
Oportunidade Social

Propriedade grid-auto-flow



Propriedade grid-auto-flow



Se você tiver itens do grid que não foram colocados explicitamente no grid, o algoritmo de auto posicionamento entra em ação para colocar os itens automaticamente. A propriedade **grid-autoflow** controla como o algoritmo funciona. Os valores possíveis para configurar essa propriedade são:

Valores para Propriedade grid-auto-flow

- **row** (default): define que o algoritmo de auto posicionamento irá preencher cada linha por vez, adicionando novas linhas se necessário.

Propriedade grid-auto-flow



- **end**: define que o algoritmo de auto posicionamento irá preencher cada coluna por vez, adicionando novas colunas se necessário.
- **dense**: define que o algoritmo de auto posicionamento use um formato de compactação "denso", ou seja, ele irá tentar preencher primeiro as lacunas no grid se itens menores surgirem.

Propriedade grid-auto-flow



Exemplo

```
<section class="container">
  <div class="item-a">item-a</div>
  <div class="item-b">item-b</div>
  <div class="item-c">item-c</div>
  <div class="item-d">item-d</div>
  <div class="item-e">item-e</div>
</section>
```

HTML

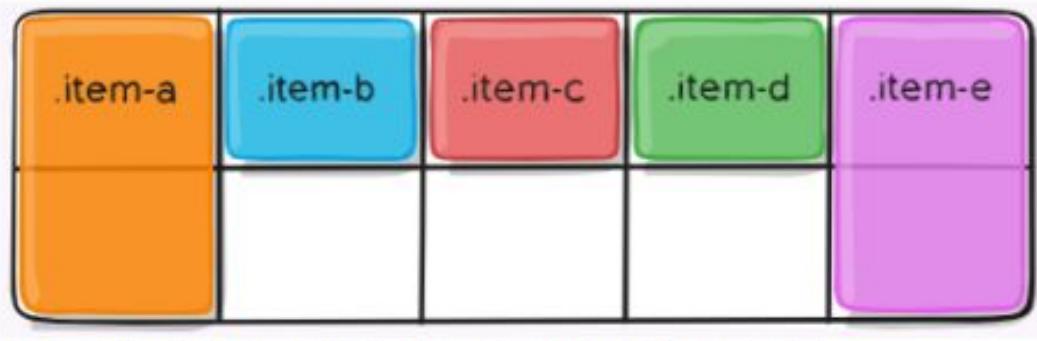
CSS

```
.container {
  display: grid;
  grid-template-columns: 60px 60px 60px 60px 60px;
  grid-template-rows: 30px 30px;
  grid-auto-flow: row;
```

Propriedade grid-auto-flow

Como a propriedade **grid-auto-flow** foi configurada com o valor row, o nosso grid será exibido com os três itens (item-b, item-c e item-d) no grid, que nós não especificamos, posicionados pela linha.

```
.item-a {  
    grid-column: 1;  
    grid-row: 1 / 3;  
}  
  
.item-e {  
    grid-column: 5;  
    grid-row: 1 / 3;  
}
```

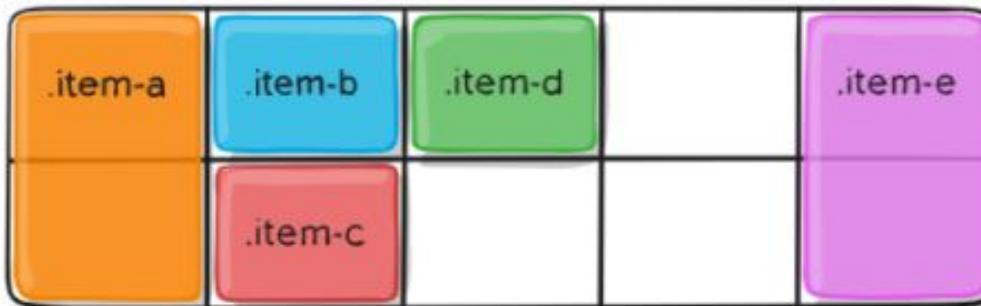


Propriedade grid-auto-flow



Se ao invés do valor **row** colocássemos o valor **column**, os itens seriam posicionados nas colunas.

```
.container {  
    display: grid;  
    grid-template-columns: 60px 60px 60px 60px 60px;  
    grid-template-rows: 30px 30px;  
    grid-auto-flow: column;  
}
```



IOS – Instituto de
Oportunidade Social

Propriedade grid



Propriedade grid



A propriedade **grid** é uma abreviação para configurar todas as seguintes propriedades usando uma única declaração:

grid-template-rows, **grid-template-columns**, **grid-template-areas**, **grid-auto-rows**, **grid-auto-columns** e **grid-auto-flow**.

```
.container {  
    grid: 100px 300px / 3fr 1fr;  
}
```

```
.container {  
    grid-template-rows: 100px 300px;  
    grid-template-columns: 3fr 1fr;  
}
```

Outro exemplo equivalente é:

```
.container {  
    grid: auto-flow / 200px 1fr;  
}
```

```
.container {  
    grid-auto-flow: row;  
    grid-template-columns: 200px 1fr;  
}
```

Propriedade grid



Mais um exemplo de trecho de código equivalente:

```
.container {  
    grid: auto-flow dense 100px / 1fr 2fr;  
}
```

```
.container {  
    grid-auto-flow: row dense;  
    grid-auto-rows: 100px;  
    grid-template-columns: 1fr 2fr;  
}
```

Propriedade grid



Vamos para um último exemplo de trecho equivalentes:

```
.container {  
    grid: 100px 300px / auto-flow 200px;  
}
```

```
.container {  
    grid-template-rows: 100px 300px;  
    grid-auto-flow: column;  
    grid-auto-columns: 200px;  
}
```

IOS – Instituto de
Oportunidade Social

Exercício



Exercício



A intenção do exercício é criar uma página sobre você utilizando containers

- Crie um container com 9 itens;
- No primeiro item deverá conter uma foto sua com o seu nome, idade e sua profissão;
- No segundo container deverá conter um resumo sobre você com curiosidades;
- No terceiro deverá conter uma breve descrição sobre a sua carreira profissional;

Exercício



- No quarto deverá conter uma lista com 5 coisa que você mais gosta em sua vida;
- No quinto Deverá conter uma lista com 5 qualidades
- No sexto, sétimo, oitavo e nono item deverá conter em cada item uma foto, pode ser fotos da sua família, amigos coisa que você ama e que são importantes para vocês;

Observação: O seu grid deverá conter 4 colunas e 6 linhas, lembrando que nenhum item pode sobressair o outro e as informações têm que estar bem separadas e legivel.

Algumas configurações do CSS e Grid

Configuração padrão para toda a pagina *;

> Margin de 0px e Padding de 0px;

Você pode utilizar um conjunto de paletas de cores deste site.

<https://colors.co/palettes/trending>