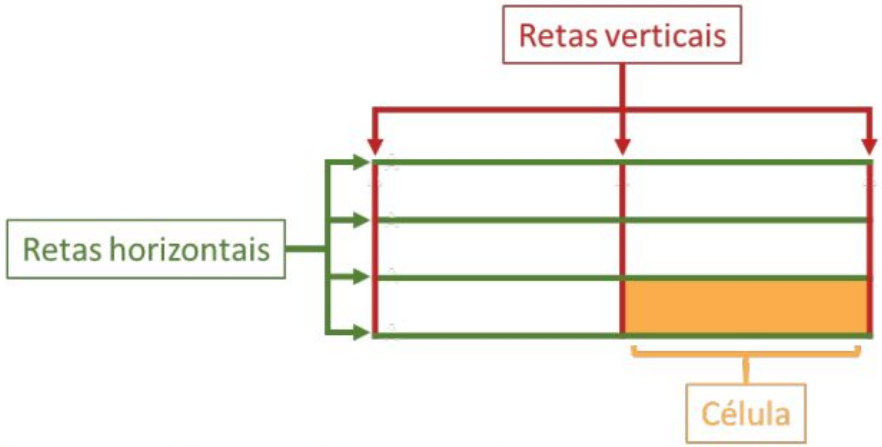
IOS – Instituto de Oportunidade Social

Grid CSS Layout

Conteúdo

- O que é Grid.
- Propriedades do elemento pai (Grid Container).
- Propriedades dos elementos filhos (Grid Items).

Grid CSS Layout



- Tabelas bidimensionais
- Vantagens sobre tabelas

Grid CSS Layout

Termos

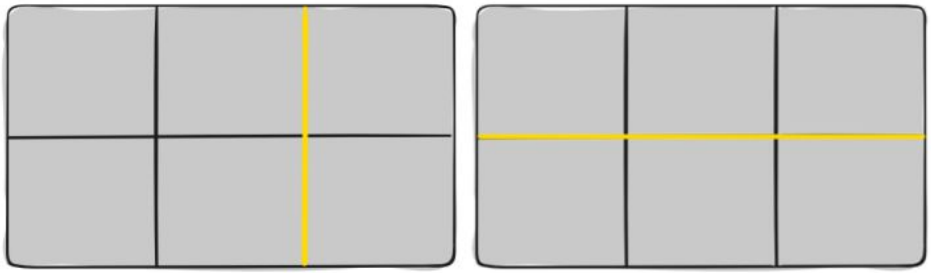
```
<section class="container">
    <div class="item item-1"> </div>
    <div class="item item-2"> </div>
    <div class="item item-3"> </div>
</section>
```

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

Grid CSS Layout

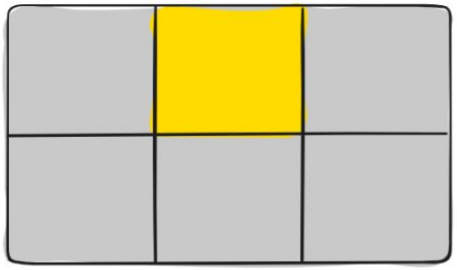
Grid Item: são os filhos do grid container.

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.



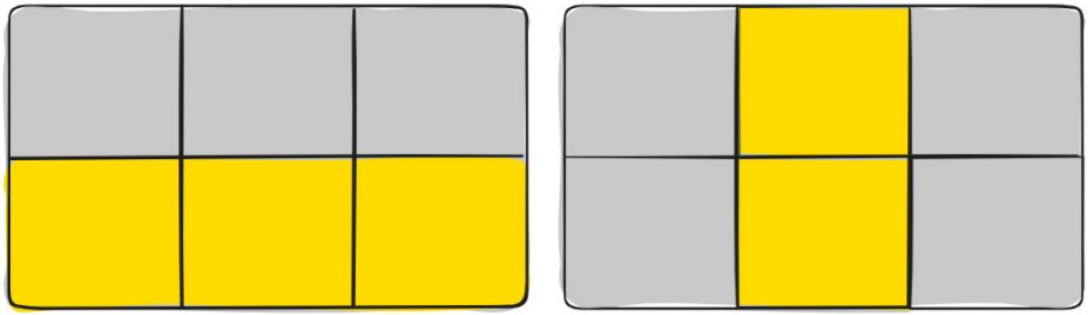
Grid CSS Layout

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.



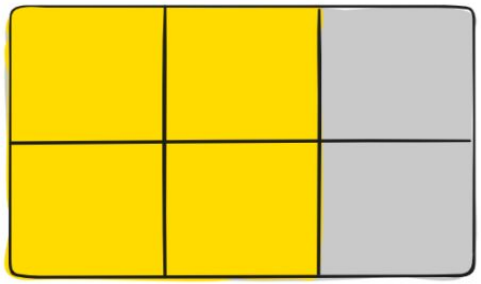
Grid CSS Layout

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



Grid CSS Layout

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)

Grid CSS Layout

Propriedades do 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

Grid CSS Layout

Propriedade display

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

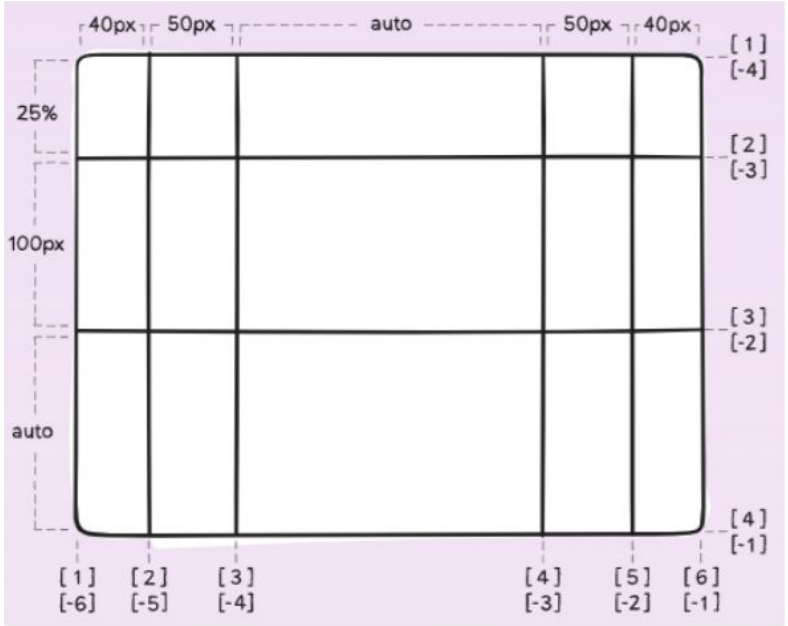
```
.container {
    display: grid;
}
```

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

Grid CSS Layout

grid-template-columns e grid-template-rows

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



Grid CSS Layout

grid-template-columns e 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];
}
```

Grid CSS Layout

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

.container {
    grid-template-columns: repeat(4, 1fr);
}
```

É equivalente a:

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

.container {
    grid-template-columns: 1fr 1fr 1fr 1fr;
}
```

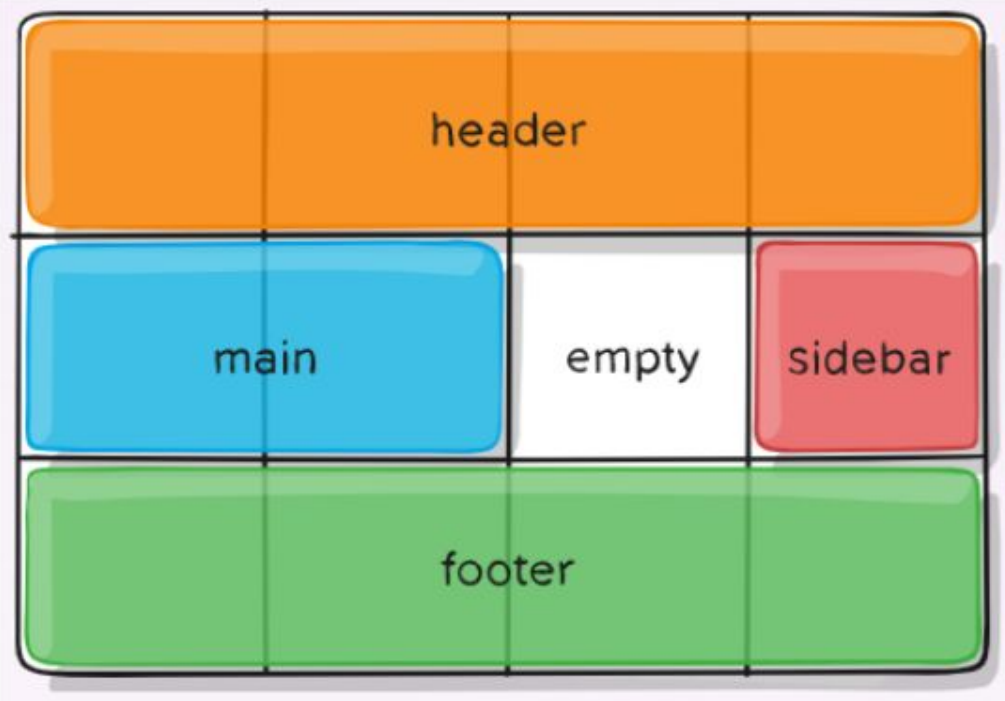
Grid CSS Layout

Propriedade 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"
```

Grid CSS Layout

Propriedade grid-template-areas



Propriedade 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:

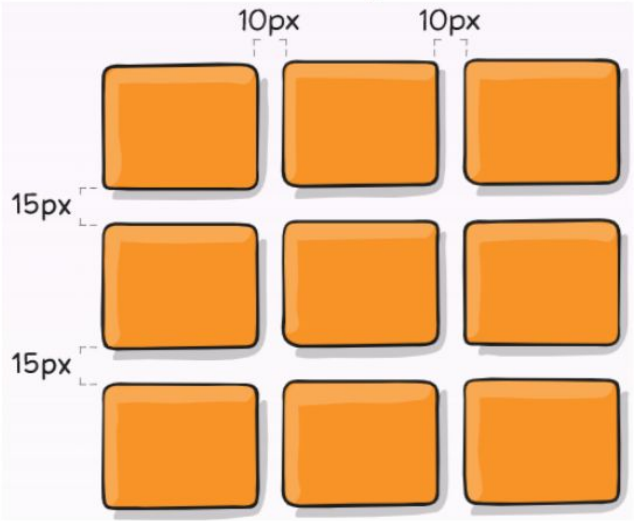
```
.container {
    grid-template-rows: [row1-start] 25px [row1-end row2-start] 25px [row2-end];
    grid-template-columns: auto 50px auto;
    grid-template-areas: "header header header" "footer footer footer";
}
```

Grid CSS Layout

Propriedades column-gap, row-gap, grid-column-gap e 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:



Propriedade gap ou grip-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

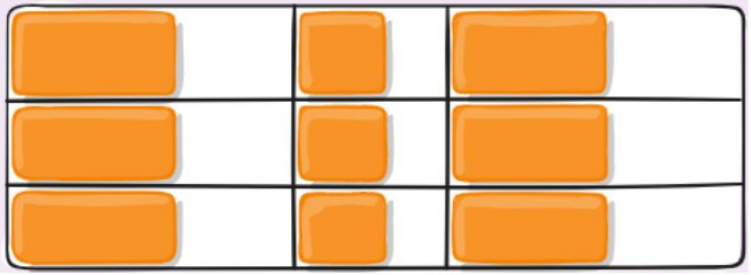
Propriedade justify-items

start

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

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

Resultado:

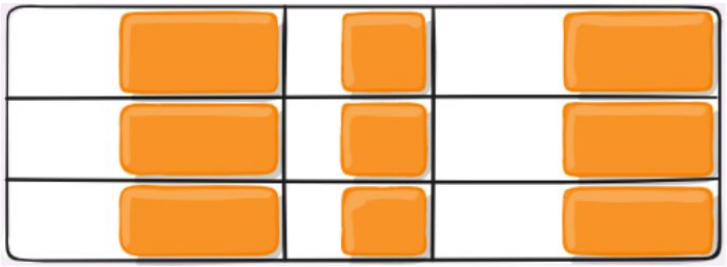


end

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

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

Resultado:

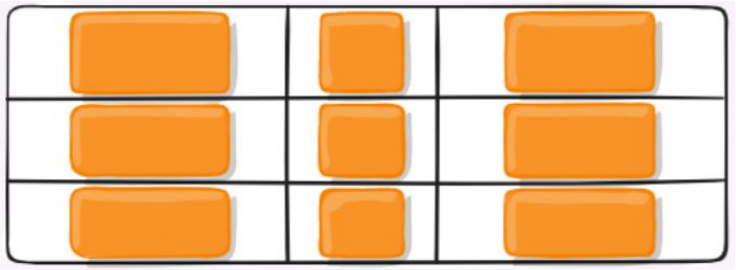


center

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

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

Resultado:

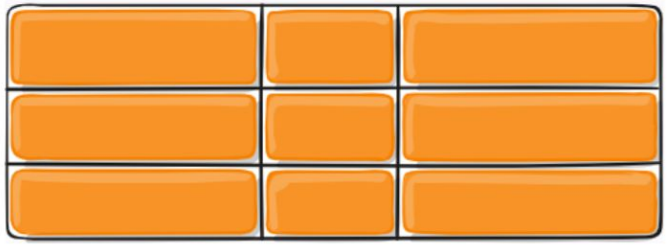


stretch

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

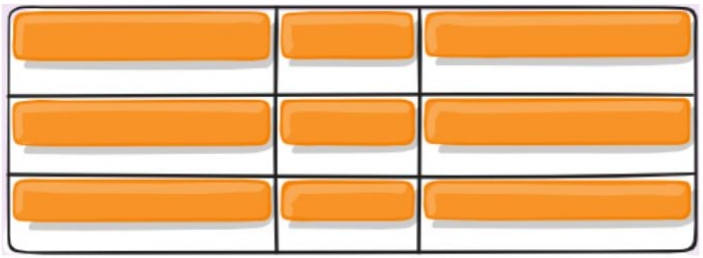
Propriedade align-items

start

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

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

Resultado:

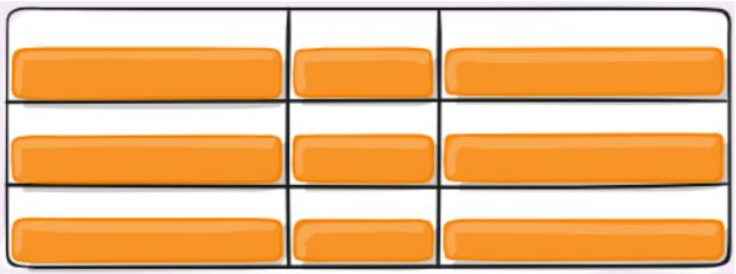


end

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

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

Resultado:



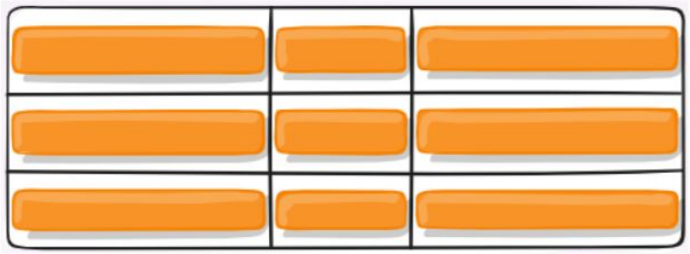
Grid CSS Layout

center

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

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

Resultado:

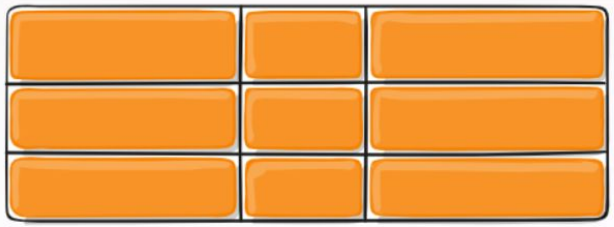


stretch

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

- 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

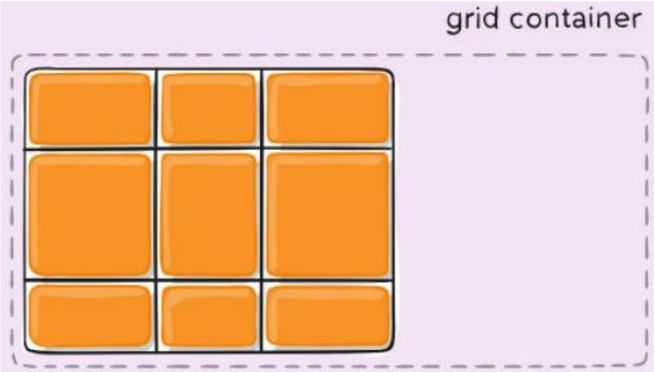
Grid CSS Layout

start

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

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

Resultado:

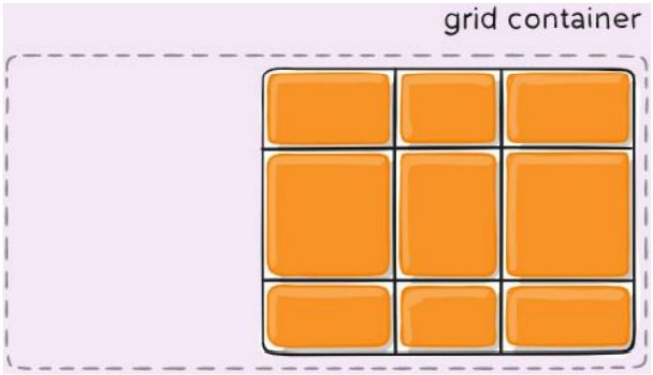


end

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

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

Resultado:

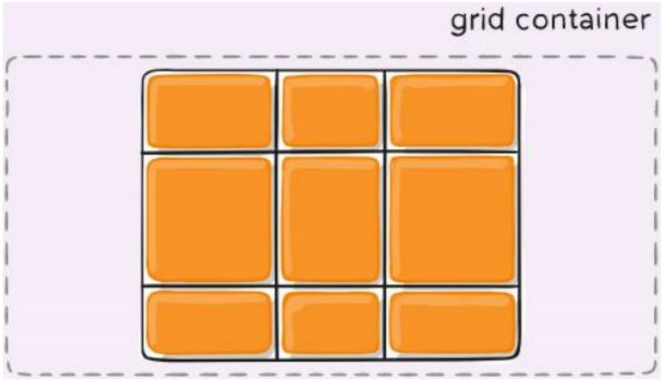


center

center: o grid fica centralizado horizontalmente no container.

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

Resultado:

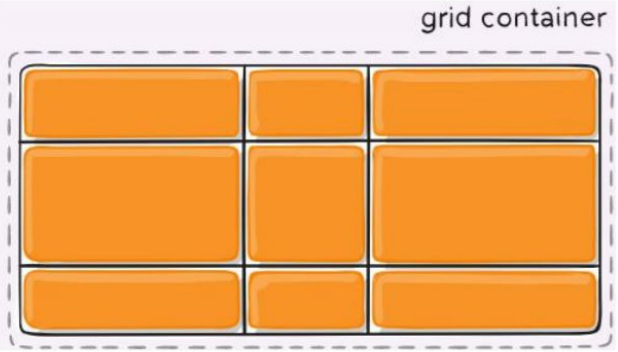


stretch

stretch: o grid preenche todo o espaço do container.

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

Resultado:

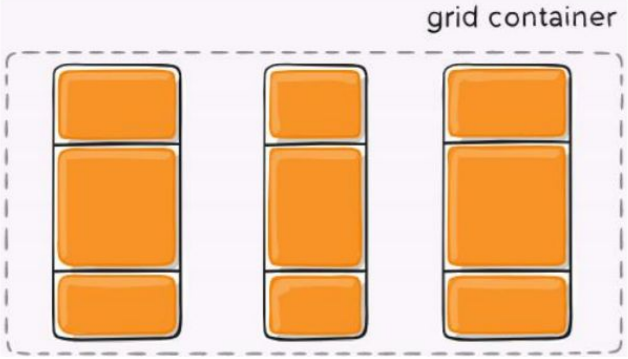


space-around

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:

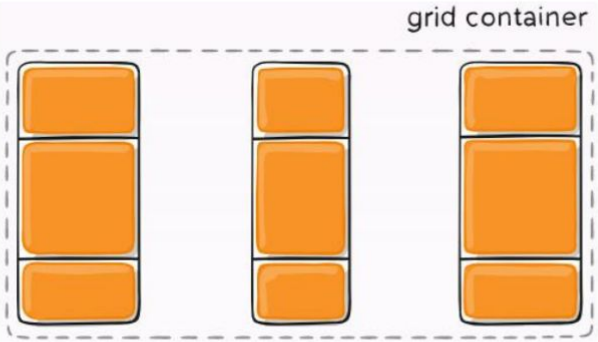


space-between

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;
}
```

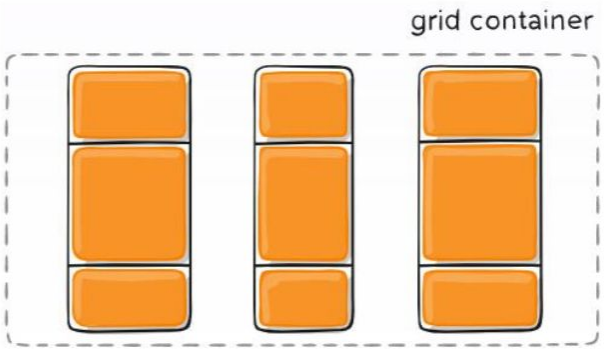
Resultado:



space-evenly

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

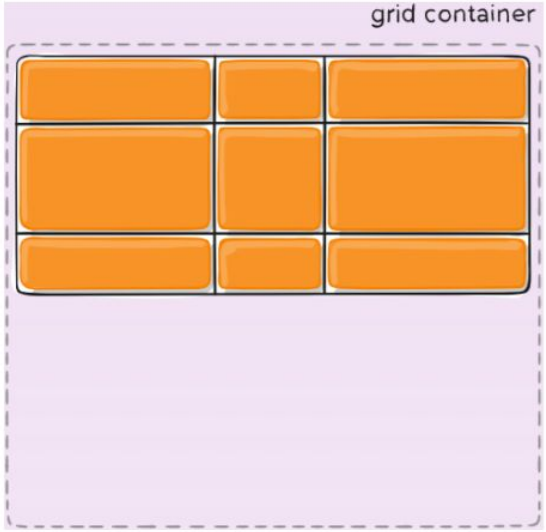
Grid CSS Layout

start

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

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

Resultado:



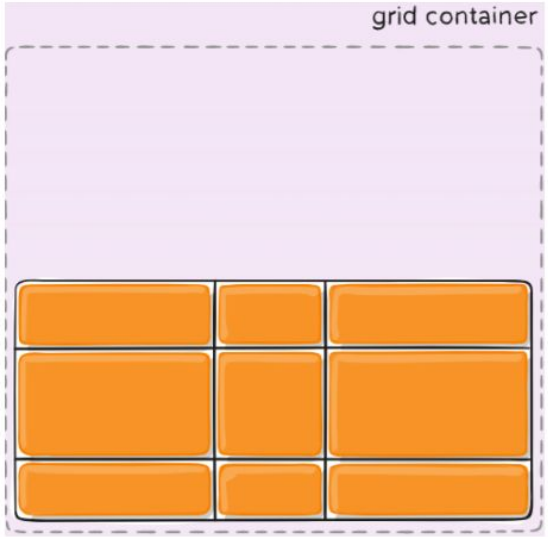
Grid CSS Layout

end

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

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

Resultado:



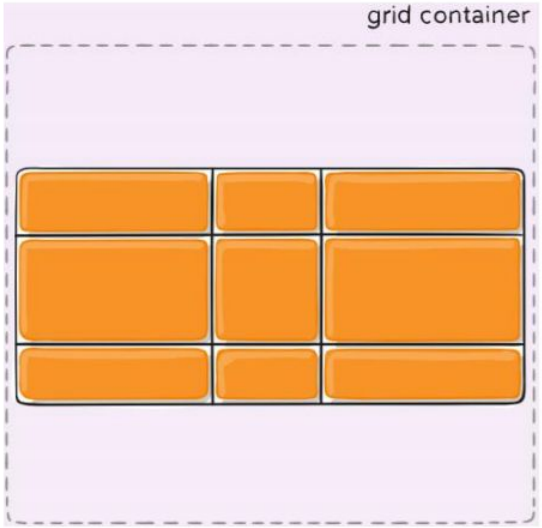
Grid CSS Layout

center

center: o grid fica centralizado verticalmente no container.

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

Resultado:



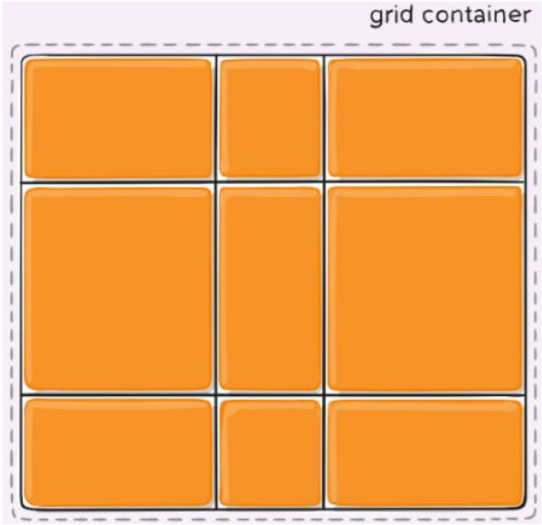
Grid CSS Layout

stretch

stretch: o grid preenche todo o espaço do container.

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

Resultado:



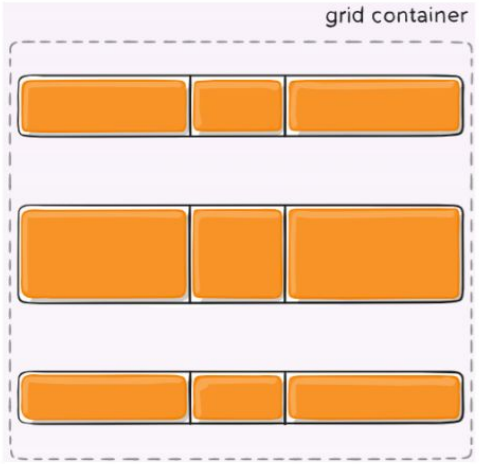
Grid CSS Layout

space-around

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:

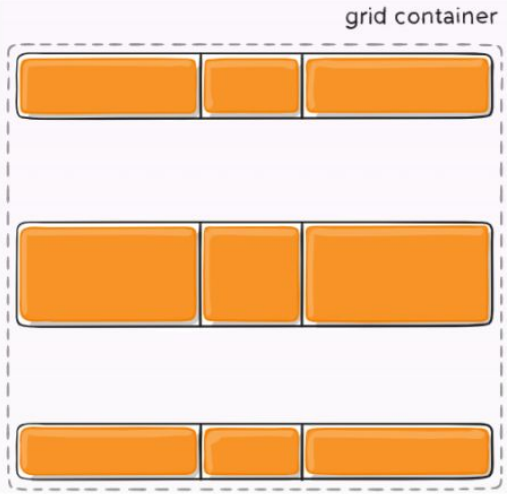


space-between

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:

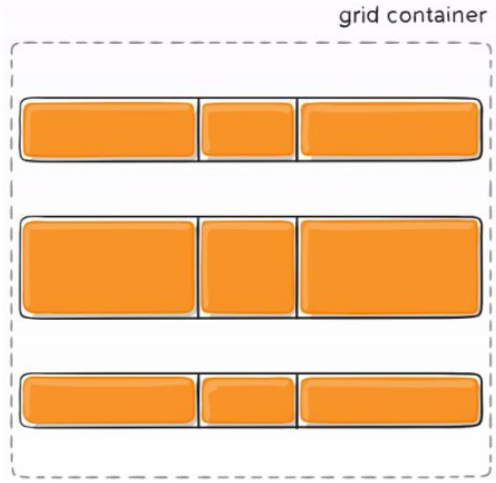


space-evenly

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

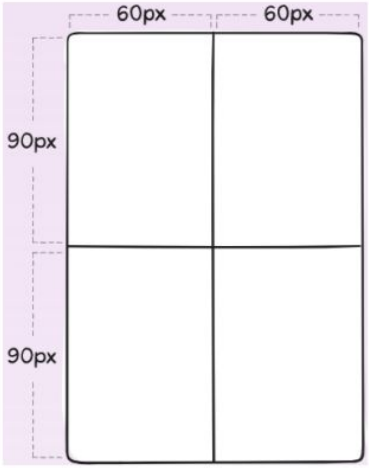
Grid CSS Layout

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

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

Resultado:

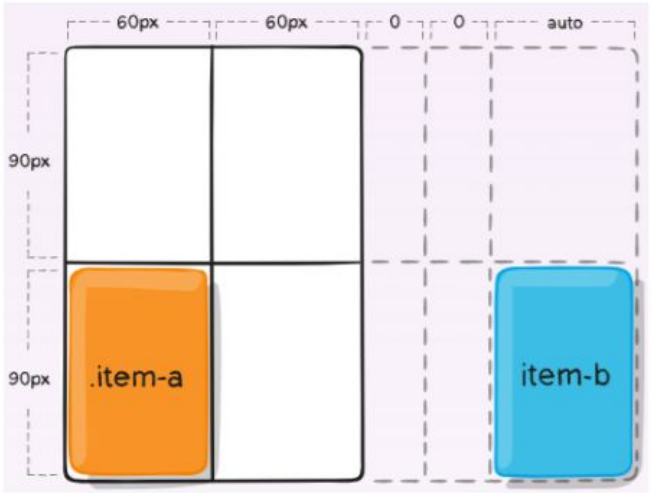


Grid CSS Layout

grid-auto-columns e grid-auto-rows

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

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

Resultado:



Grid CSS Layout

grid-auto-columns e grid-auto-rows

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

IOS – Instituto de Oportunidade Social

Propriedade grid-auto-flow

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.

- 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.

Grid CSS Layout

Exemplo

```html
<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

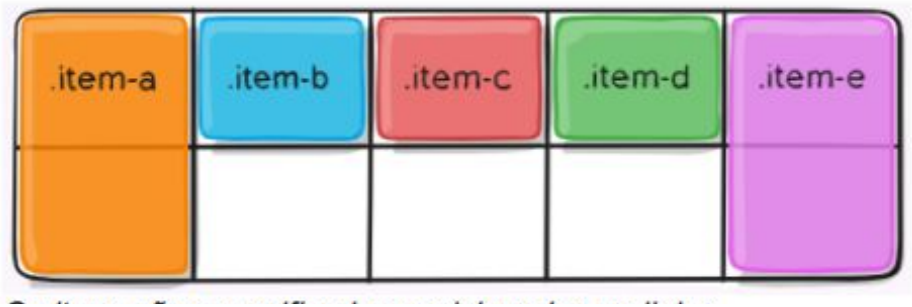
CSS

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

Grid CSS Layout

itens

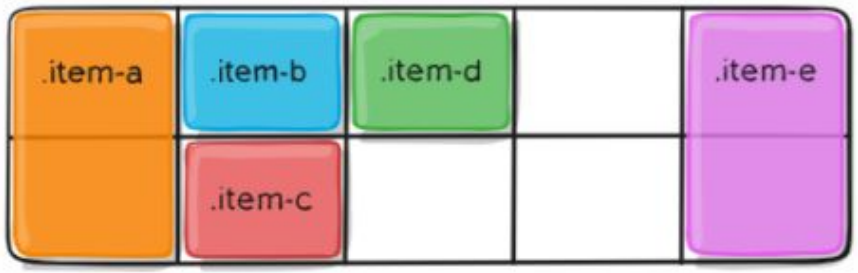
```
.item-a {
    grid-column: 1;
    grid-row: 1 / 3;
}

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



Grid CSS Layout

Propriedade gap ou grip-gap

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



IOS – Instituto de Oportunidade Social

Exercício

Exercício

-