IOS – Instituto de Oportunidade Social

CSS 13 - CSS Grid Layout Parte 02

Objetivos

- Compreender a criação de diferentes layouts com o uso do **Grid CSS**;

- Conhecer as diversas **propriedades** do Grid CSS;

- Aplicar os **recursos** do Grid CSS nas folhas de estilo.

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**.

Por exemplo, os dois trechos de código definem a mesma configuração:

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

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

IOS – Instituto de Oportunidade Social

Grid Items

Grid Items

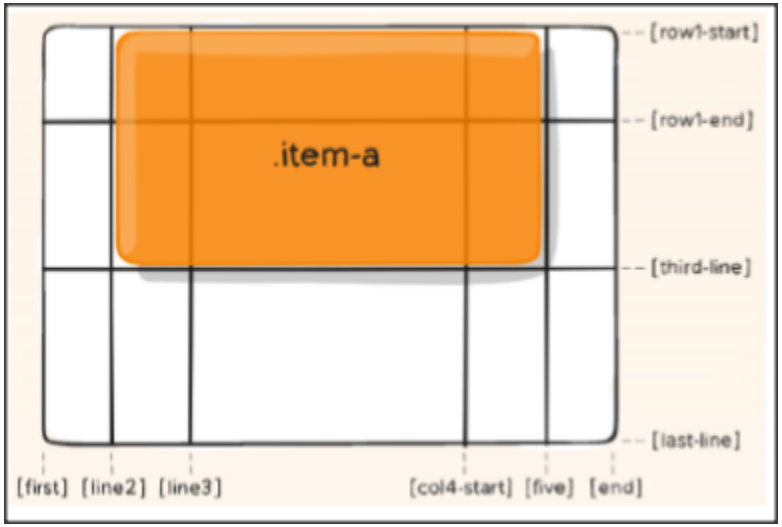
Essas propriedades determinam a localização de um item de acordo com uma reta específica no grid. As propriedades **grid-column-start** e **grid-row-start** definem a reta onde a localização do item irá começar e as propriedades **grid-column-end** e **grid-row-end** onde a localização do item irá terminar. Os valores possíveis são:

- **<reta>**: pode ser o **número da reta** no grid ou o nome dela.
- **span <number>**: o item irá ocupar a **quantidade de trilhas** que for especificado no parâmetro <number>.
- **span <name>**: o item irá ocupar a quantidade de trilhas até a reta com o **nome especificada** no parâmetro <name>.
- **auto:** indica auto posicionamento.

Grid Items

Exemplo:

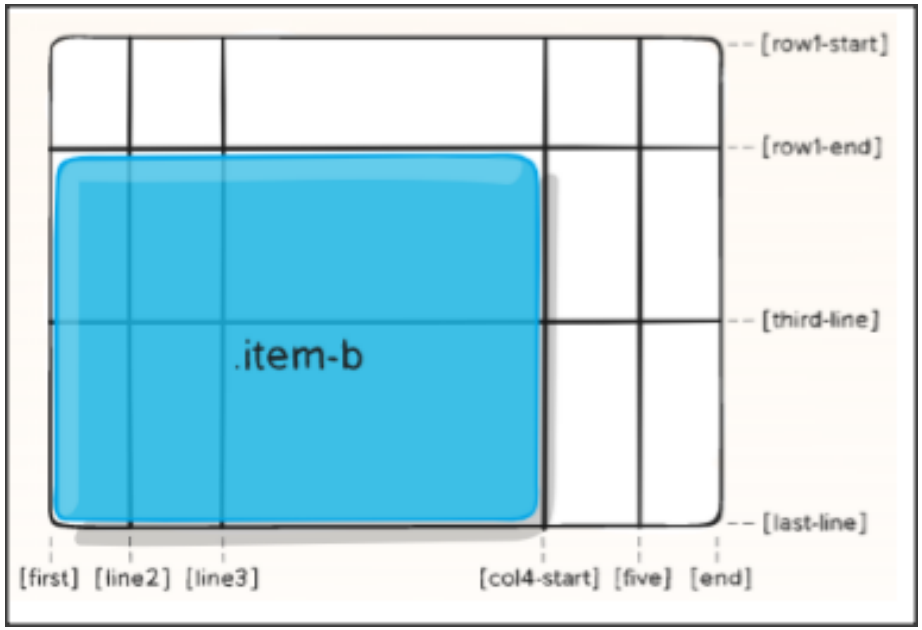
```
.item-a {
        grid-column-start: 2;
        grid-column-end: five;
        grid-row-start: row1-start;
        grid-row-end: 3; }
```



Grid Items

Exemplo:

```
.item-b {
        grid-column-start: 1;
        grid-column-end: span col4-start;
        grid-row-start: 2;
        grid-row-end: span 2;
```



Grid Items

Propriedades **grid-column** e **grid-row**

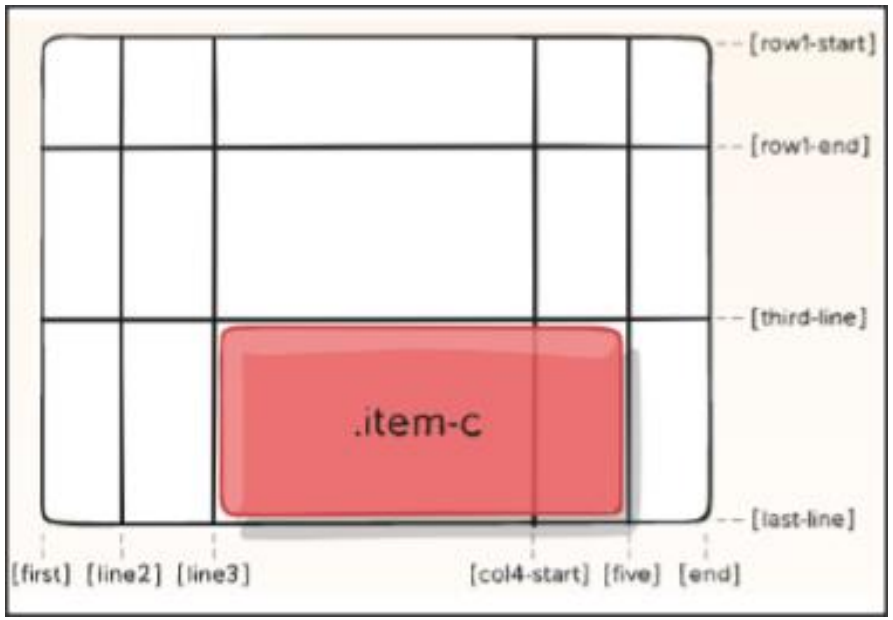
Essas propriedades são abreviações para a **grid-column-start + grid-column-end e grid-rowstart e grid-row-end** repectivamente. Os valores possíveis são **<start-line> / <end-line>** (reta incial / reta final).

Vejamos o exemplo e seu respectivo resultado. No exemplo a seguir, o item-c tem ocupa verticalmente dois espaços a partir da reta vertical dois (**grid-column: 2 / span 2**) e horizontalmente vai da terceira reta horizontal até a reta quatro (**grid-row: third-line / 4**).

Grid Items

Exemplo:

```
.item-c {
        grid-column: 2 / span 2;
        grid-row: third-line / 4; }
```



Grid Items

Propriedade grid-area

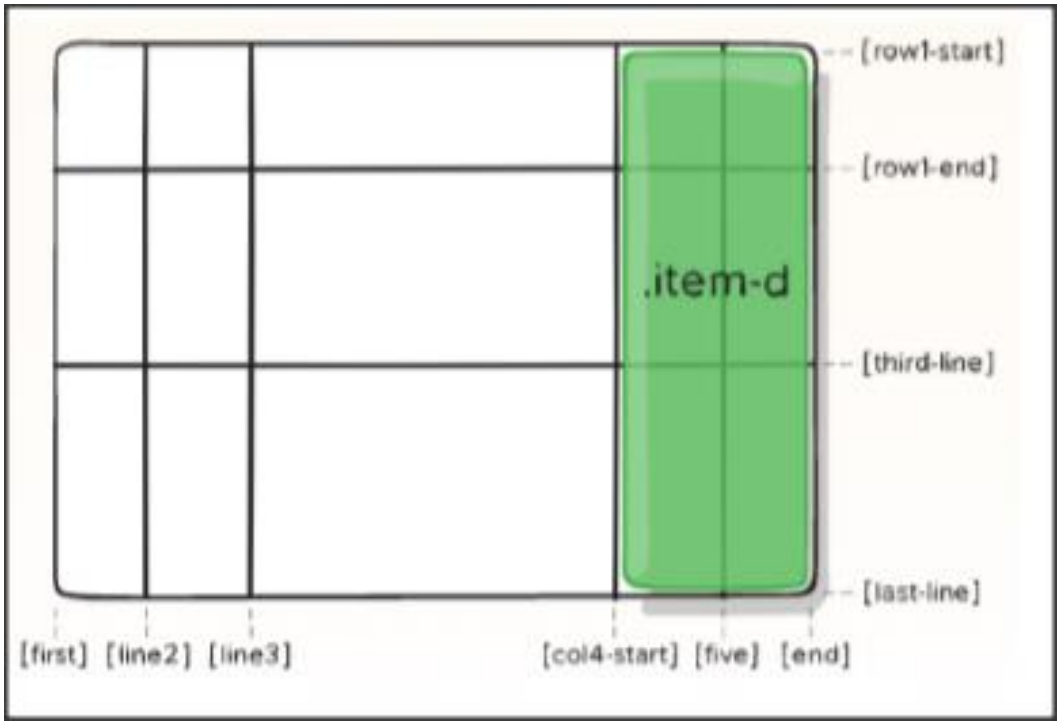
A propriedade **grid-area** define um nome/rótulo para o item do grid para que ele possa ser usado com a propriedade **grid-template-area**. Essa propriedade pode também ser usada como uma abreviação para configura **grid-row-start + grid-column-start + grid-row-end + grid-column-end** em uma única declaração. Os valores possíveis são:

- **<name>**: nome que você escolher para o item.
- **<row-start> / <column-start>**: retas horizontal e vertical iniciais. Você pode usar o nome ou número das retas.
- **<row-end> / <column-end>**: retas horizontal e vertical finais. Você pode usar o nome ou número das retas.

Grid Items

Exemplo:

.item-d { grid-area: 1 / col4-start / last-line / 6; }



IOS – Instituto de Oportunidade Social

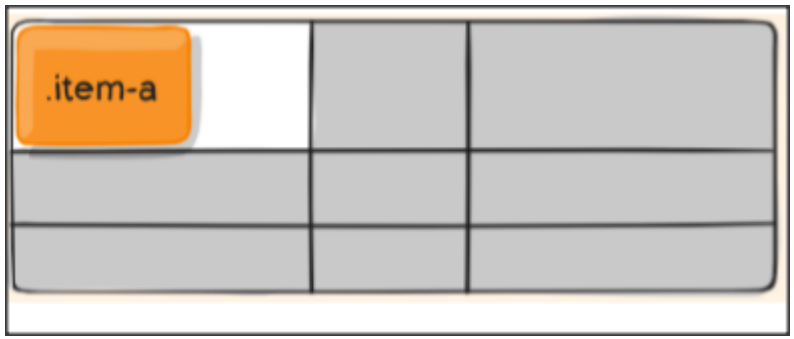
Propriedade justify-self

Propriedade justify-self

A propriedade **justify-self** alinha o item horizontalmente na célula. A diferença estre a justify-self e a **justify-items** é que a primeira configura o alinhamento horizontal de um item e a segunda de todos os itens do grid. Os valores possíveis para configurar essa propriedade são:

- **start**: o item fica alinhado na reta esquerda da célula do grid.

```
.item-a {    justify-self: start; }
```



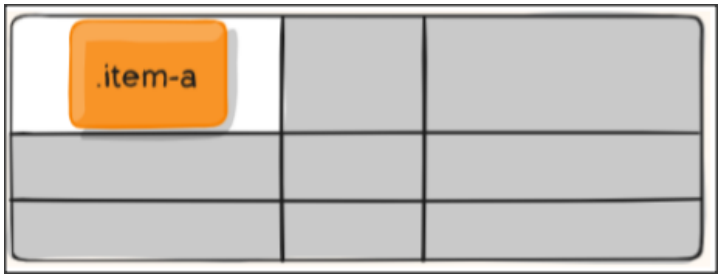
Propriedade justify-self

- **end**: o item fica alinhado na reta direita da célula do grid.

```
.item-a {    justify-self: end; }
```

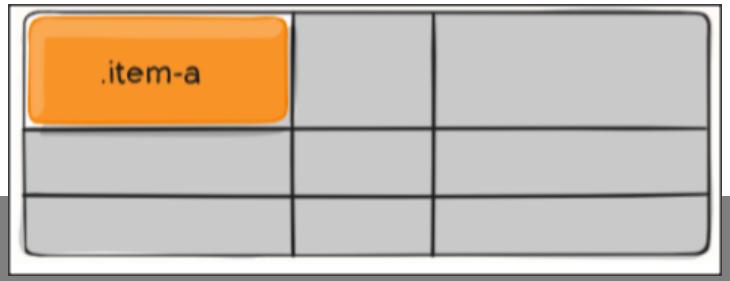
- **center**: o item fica centralizado horizontalmente na célula do grid.

```
.item-a {    justify-self: center; }
```



- **stretch**: o item preenche todo o espaço da célula do grid.

```
.item-a {    justify-self: stretch; }
```



IOS – Instituto de Oportunidade Social

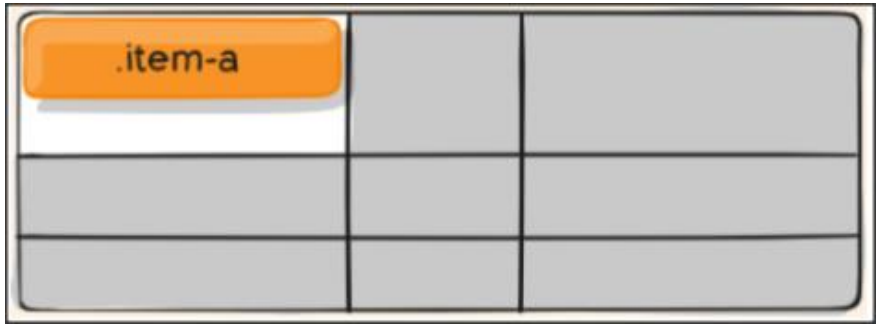
Propriedade align-self

Propriedade align-self

A propriedade **align-self** alinha o item verticalmente na célula. A diferença estre a align-self e a **align-items** é que a primeira configura o alinhamento vertical de um item e a segunda de todos os itens do grid. Os valores possíveis para configurar essa propriedade são:

- **start**: o item fica alinhado na reta superior da célula do grid.

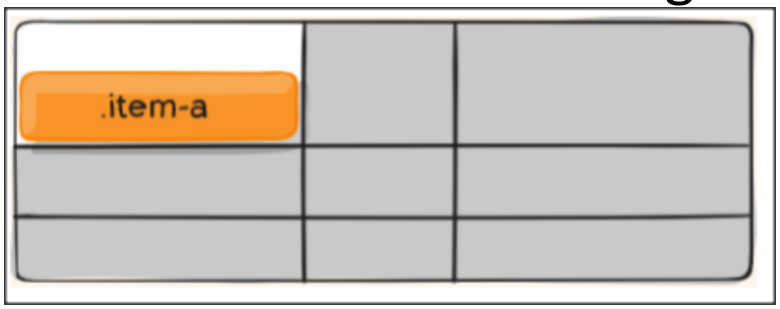
```
.item-a {    align-self: start; }
```



Propriedade align-self

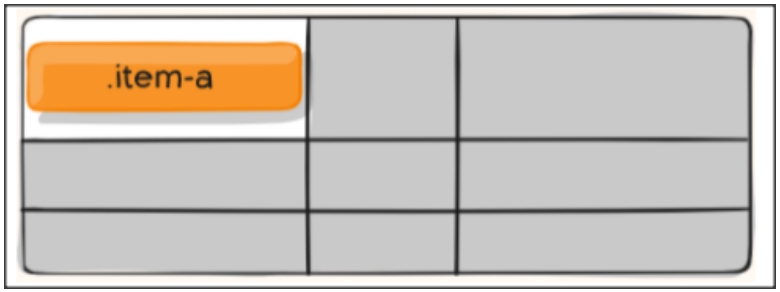
- **end**: o item fica alinhado na reta inferior da célula do grid.

```
.item-a {    align-self: end; }
```



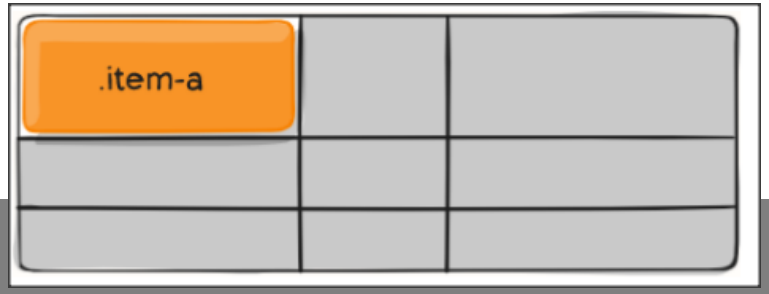
- **center**: o item fica centralizado verticalmente na célula do grid.

```
.item-a {    align-self: center; }
```



- **stretch**: o item preenche todo o espaço da célula do grid.

```
.item-a {    align-self: stretch; }
```



IOS – Instituto de Oportunidade Social

Propriedade place-self

Propriedade place-self

A propriedade **place-self** configura ambas as propriedades align-self e justify-self em uma única declaração. Os valores possíveis são:

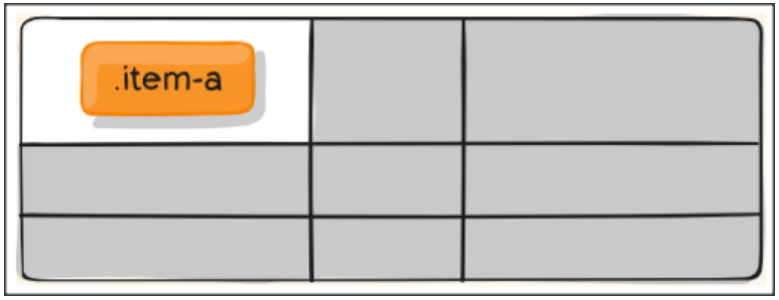
- **auto**: usa o alinhamento padrão para o modelo de layout.

- **<align-self> / <justify-self>**: o **primeiro valor** é para o **alinhamento vertical** e **segundo valor** é para o **alinhamento horizontal** dos itens. Se o segundo valor for omitido, o primeiro valor será atribuído para ambos os alinhamentos.

Propriedade place-self

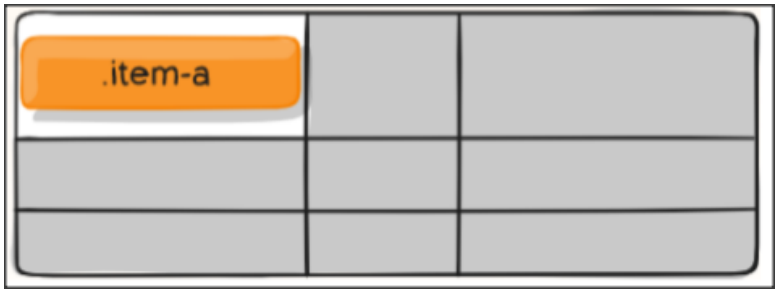
Vejamos um primeiro exemplo para **centralizar o item verticalmente e horizontalmente** na célula.

```
.item-a {     place-self: center; }
```



O próximo exemplo **centraliza verticalmente o item** na célula e o **faz ocupar todo o espaço horizontal**.

```
.item-a {     place-self: center stretch; }
```



IOS – Instituto de Oportunidade Social

Vamos Praticar

Vamos Praticar

Apostila de CSS:

- CSS Grid (1, 2 e 3)

Páginas 165 a 179

OBS: Acompanhar o passo a passo com o instrutor

IOS – Instituto de Oportunidade Social

Exercícios

Exercícios

Montar uma página HTML **cssGridLayoutP2.html** com seu respectivo par CSS.

- Criar 6 templates de Grid com a propriedade **grid**:

1) Utilizar **grid-column-start, grid-row-start, grid-column-end e grid-row-end**
2) Utilizar **grid-column e grid-row**
3) Utilizar **grid-area**
4) Utilizar **justify-self**
5) Utilizar **align-self**
6) Utilizar **place-self**