



GRID LAYOUT



Grid Layout

Michele Queiroz Ambrosio

Desenvolvedora front-end

@programi_

Sobre mim

(2012-2013)

Curso técnico em informática na ETEC;

(2014-2017)

Faculdade de Ciência da Computação na UNISO;

(2017-atualmente)

Desenvolvedora Front-end na Eduzz.

Sobre mim

Como você pode me encontrar nas redes sociais?

Instagram: @programi_

Twitch: twitch.tv/michele_ambrosio

LinkedIn: Michele Ambrosio

Github: @micheleambrosio



Objetivo geral

O objetivo geral deste curso é ensinar os principais conceitos, aplicações e propriedades que envolvem o grid layout.

Pré-requisitos

É importante que você já tenha uma base de HTML e assistido ao **Módulo 1 de CSS**, juntamente com o curso de **Posicionamentos e Displays com CSS** para acompanhar os exemplos e conseguir estilizar suas páginas.

Ferramentas utilizadas

- VSCode;
- Plugins para VSCode: Live Server e Emmet;
- Google Chrome.

Percurso

Etapa 1

Entendendo os conceitos do Grid Layout

Etapa 2

Nomenclaturas do Grid

Etapa 3

Iniciando com o CSS Grid

Etapa 4

Adicionando colunas e linhas ao Grid

Percorso

Etapa 5

Grid implícito e explícito

Etapa 6

Definindo o tamanho mínimo e máximo das faixas

Etapa 7

Alocando os itens do Grid nas posições específicas

Etapa 8

Áreas do Grid

Percurso

Etapa 9

Shorthand grid-template

Etapa 10

Definindo os espaçamentos dos elementos do Grid

Etapa 11

Shorthand grid

Etapa 12

Alinhando os itens com justify-items e align-items

Percurso

Etapa 13

Alinhando os itens da linha e coluna com `place-items`

Etapa 14

Alinhando o conteúdo do Grid com `justify-content` e `align-content`

Etapa 15

Alinhando o conteúdo do Grid horizontalmente e verticalmente com `place-content`

Etapa 16

Alinhando itens específicos com `justify-self` e `align-self`

Percurso

Etapa 17

Alinhando itens específicos horizontalmente e verticalmente com place-self

Etapa 18

Materiais de apoio

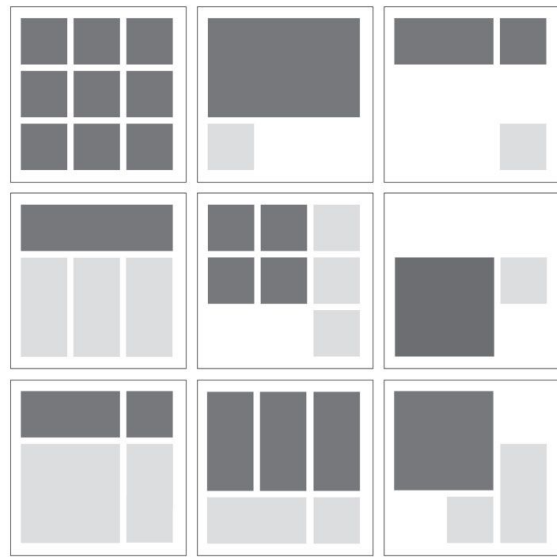


Etapa 1

Conceitos do grid layout

Entendendo os conceitos do grid

- Antigamente, era muito comum a construção de layouts usando tabelas, posicionamentos em bloco/linha e float;
- CSS Grid é um recurso que permite construir layouts baseados em grades bidimensionais, dividindo a página em regiões e definindo onde cada elemento deve ficar.

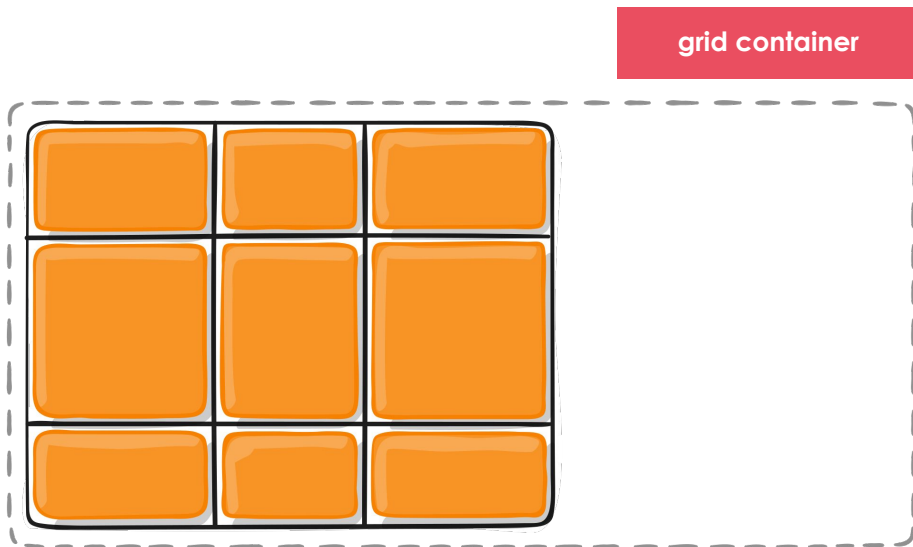


Etapa 2

Nomenclaturas do grid

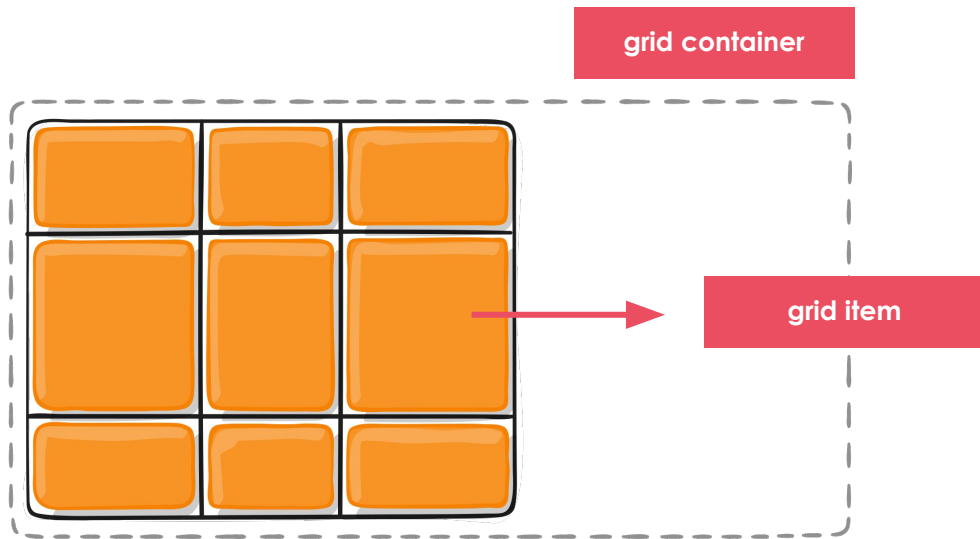
grid container

Elemento-pai que terá a propriedade **display: flex** especificada. Ele irá “envolver” todos os itens do grid.



grid item

Elementos que são filhos diretos do **grid container**.



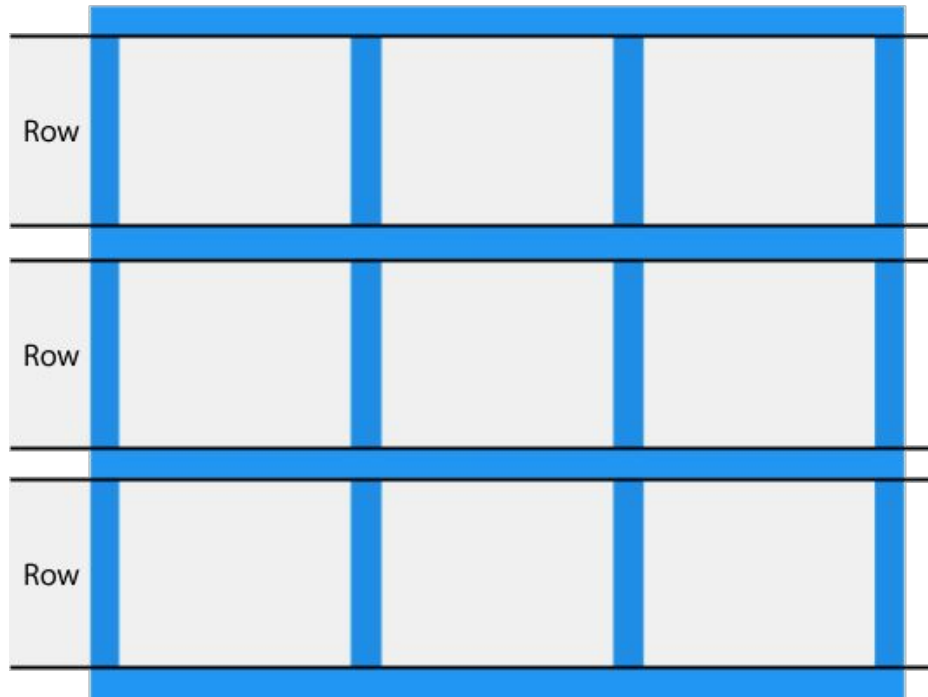
grid columns

As faixas verticais dos itens do grid, são chamados de colunas (*grid columns*)

Column	Column	Column

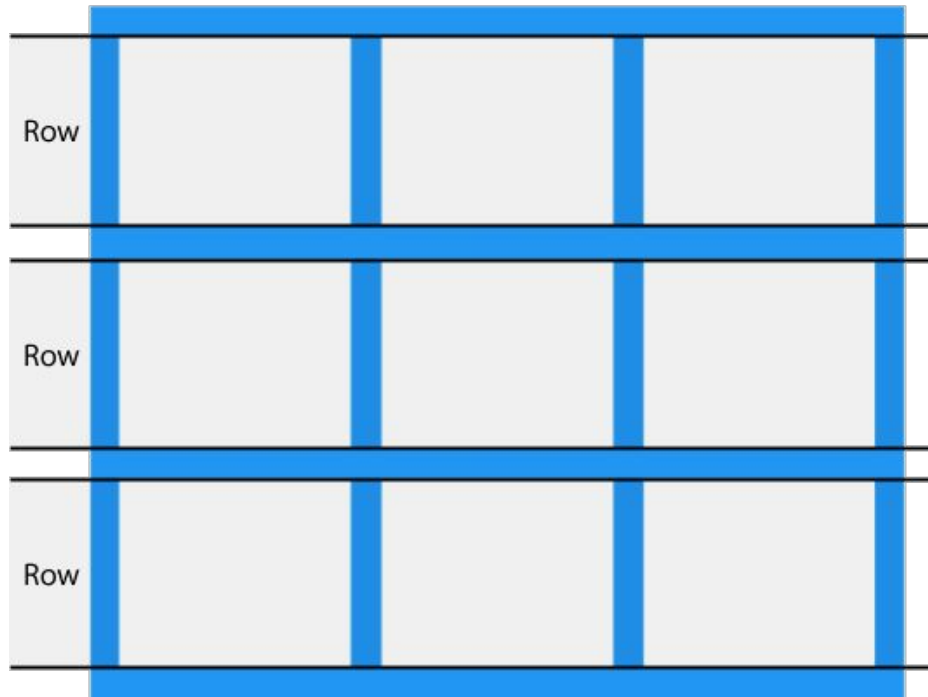
grid rows

As faixas horizontais dos itens do grid, são chamados de linhas (*grid rows*)



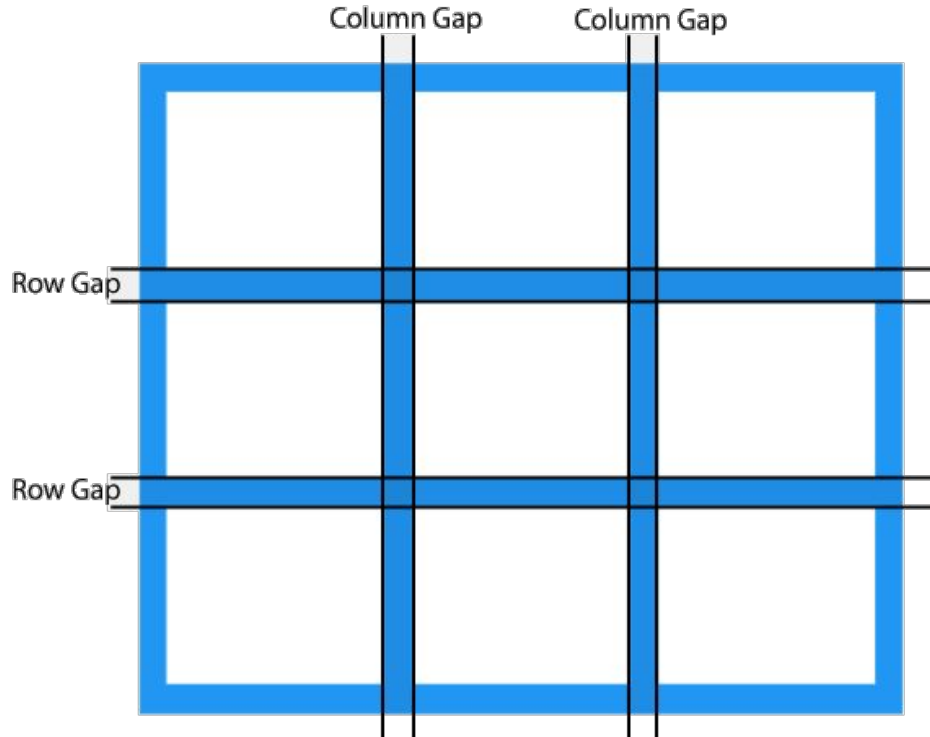
grid rows

As faixas horizontais dos itens do grid, são chamados de linhas (*grid rows*)



grid gaps

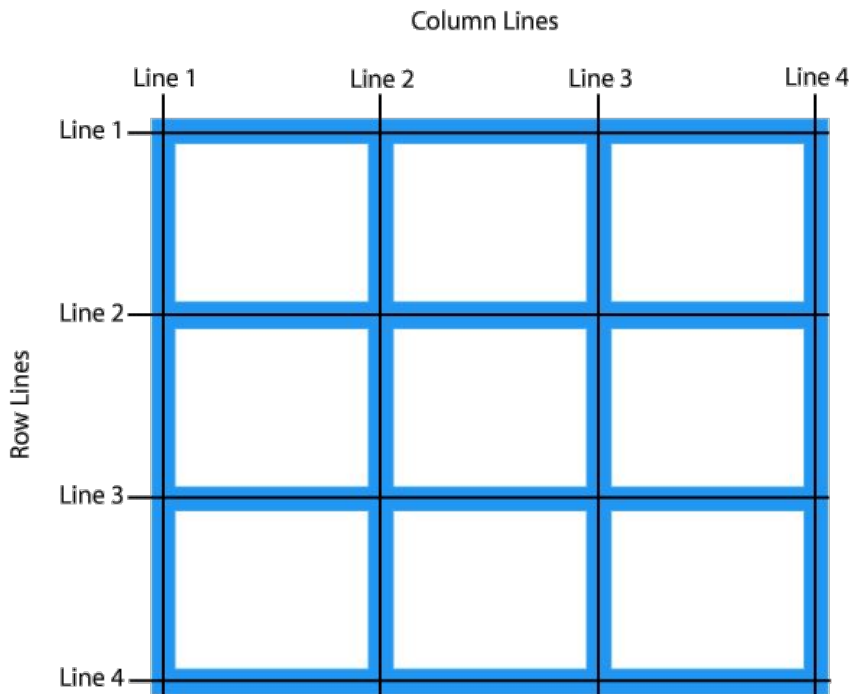
Espaços entre as linhas (*grid rows*) e coluna (*grid columns*) do grid.



grid line

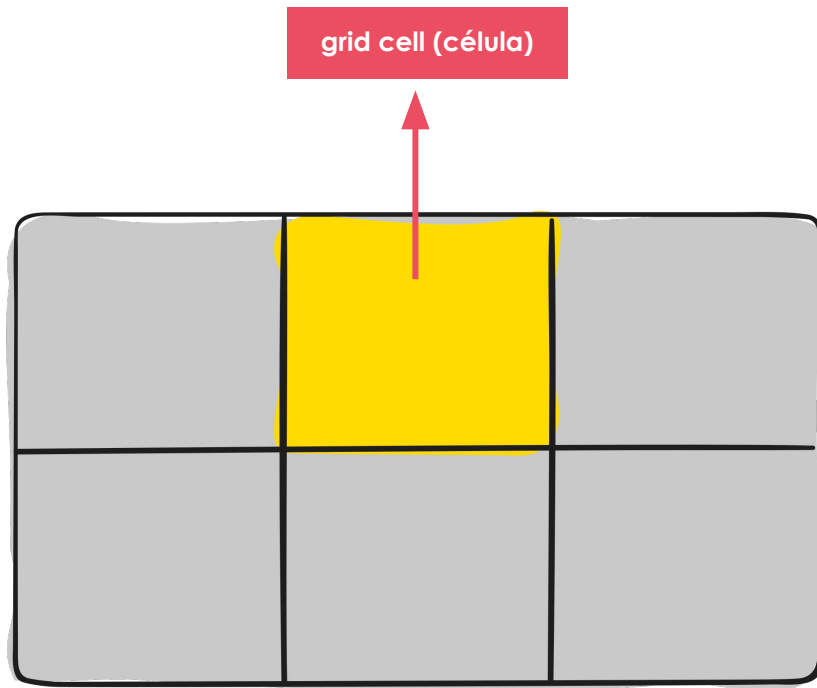
Divisórias que compõem a estrutura da grade.

As linhas entre as colunas são chamadas de *column lines* (linhas de coluna) e, as que estão entre as faixas horizontais, são chamadas de *row lines* (linhas de linha).



grid cell

São as células, ou seja, cada “quadrado” da grade, que vai entre uma *grid line* e outra.



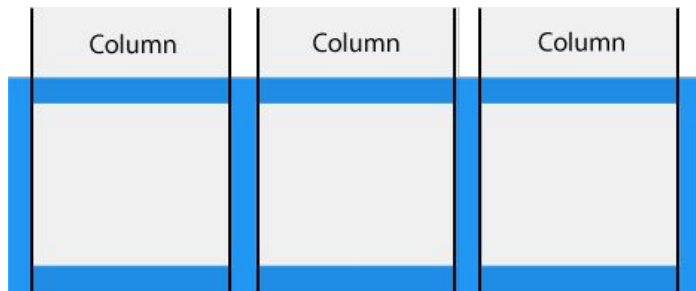
Etapa 4

Adicionando colunas e linhas ao grid

Linhas e colunas no grid

grid-template-columns

define as colunas (*columns*) do grid



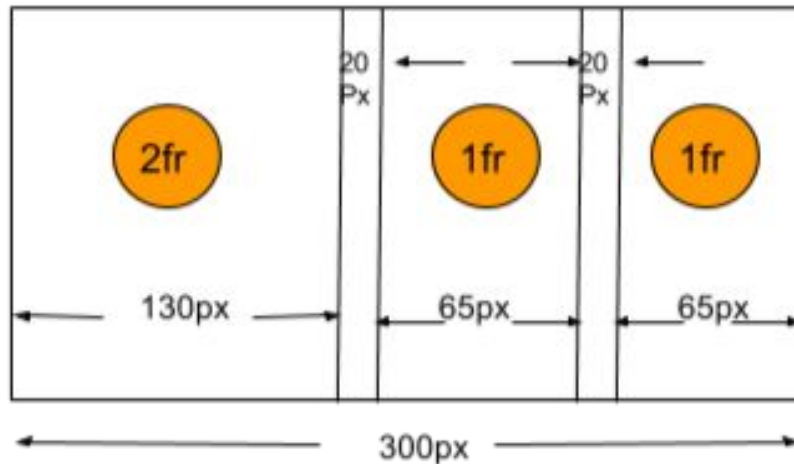
grid-template-rows

define as linhas (*rows*) do grid



Unidade de medida fr

Criada para o CSS Grid, a unidade de medida **fr** representa uma fração do espaço disponível no *container* do grid.



Função repeat

Permite repetir para todas, ou para algumas das faixas, o mesmo valor para seu tamanho de coluna ou linha.

```
/* Antes */  
.container {  
  grid-template-columns: 1fr 1fr 1fr;  
}  
  
/* Depois */  
.container {  
  grid-template-columns: repeat(3, 1fr);  
}
```

Função repeat: outros valores

auto-fill

```
grid-template-columns: repeat(auto-fill, minmax(100px, 1fr));
```



auto-fit

```
grid-template-columns: repeat(auto-fit, minmax(100px, 1fr));
```



Etapa 5

Grid implícito e explícito

Unidade de medida fr

Quando define-se as colunas e linhas através das propriedades das propriedades **grid-template-columns** e **grid-template-rows**, estamos criando um grid explícito.

Mas, se os itens não couberem, o grid colocará um item fora desse grid que definimos, criando o que chamamos de grid implícito.

Propriedade `grid-auto-flow`

Controla como o algoritmo de reposicionamento automático irá se comportar.

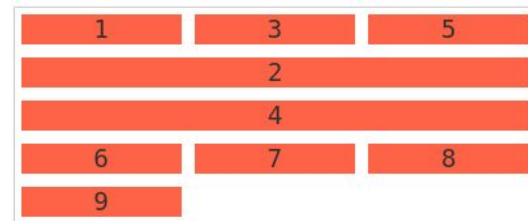
`grid-auto-flow: row;`



`grid-auto-flow: column;`



`grid-auto-flow: dense;`



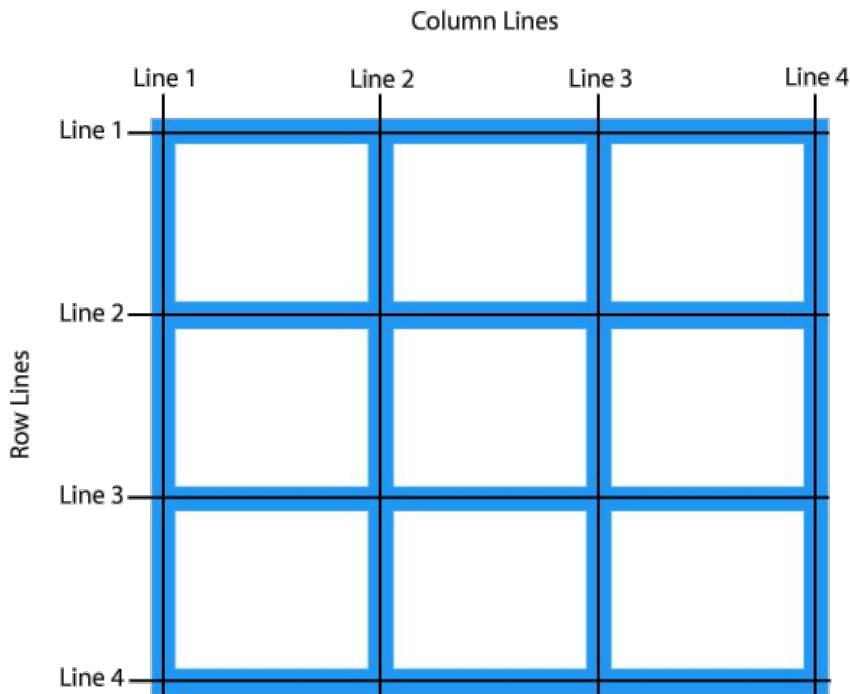
Etapa 7

**Alocando os itens do Grid nas
posições específicas**

Grid Lines

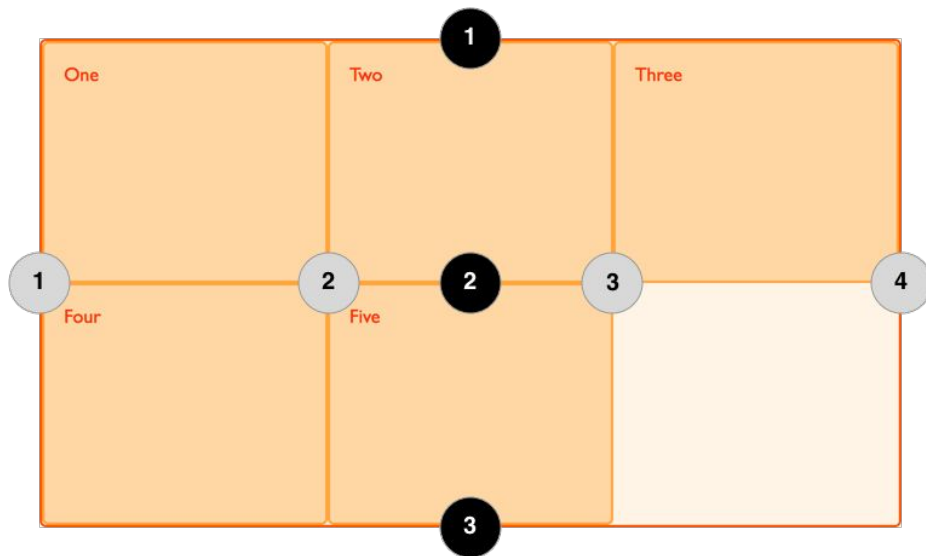
Divisórias que compõem a estrutura da grade.

As linhas entre as colunas são chamadas de *column lines* (linhas de coluna) e, as que estão entre as faixas horizontais, são chamadas de *row lines* (linhas de linha).

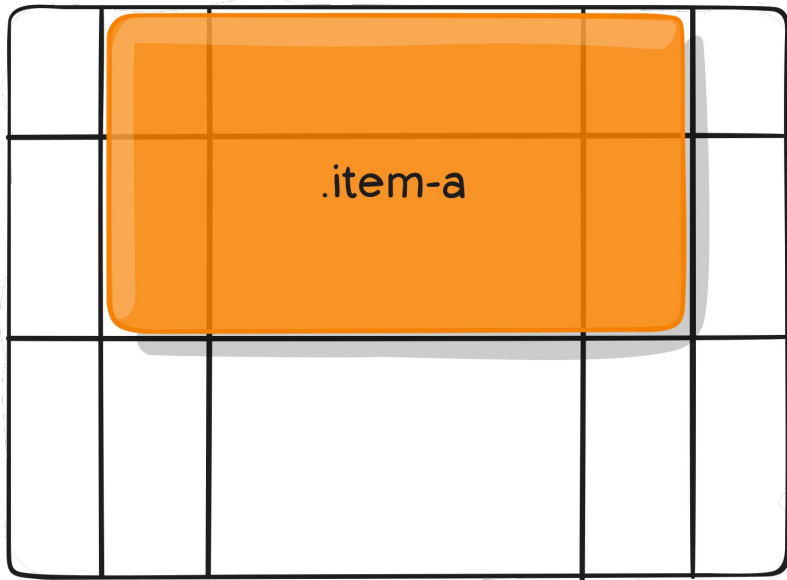


Grid Lines

Cada linha possui um número para que possamos usar de referência (exceto para grids implícitos).

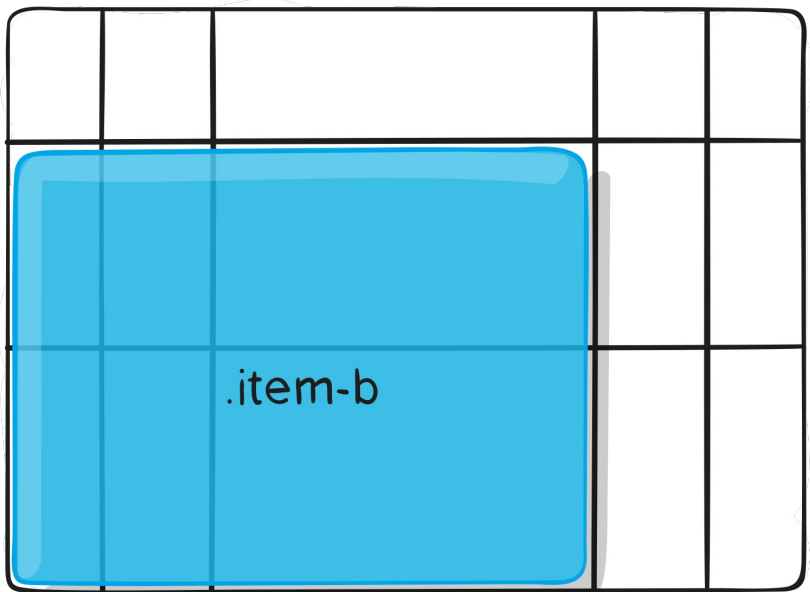


Posicionando os itens do grid



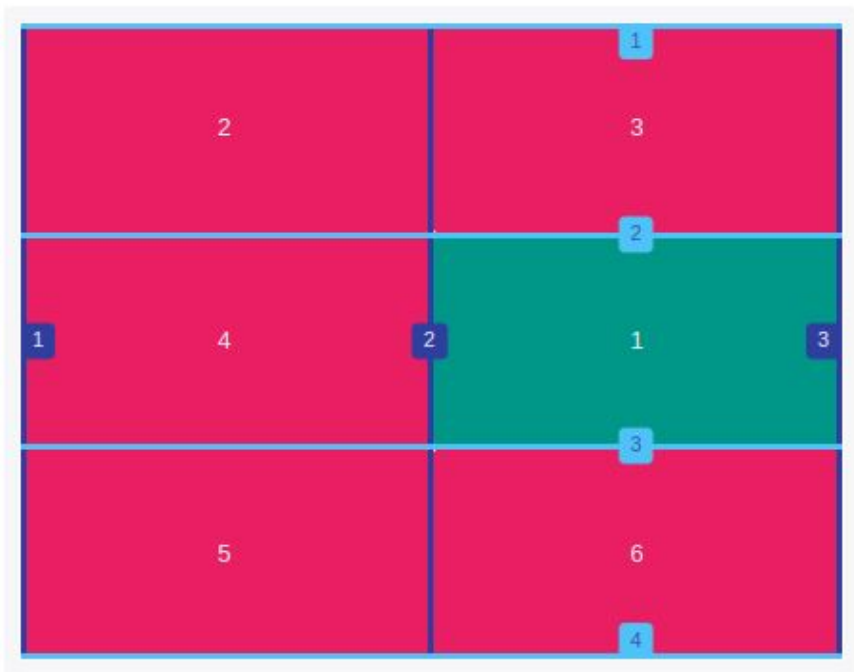
```
.item {  
  grid-column-start: 2;  
  grid-column-end: 5;  
}
```

Posicionando os itens do grid



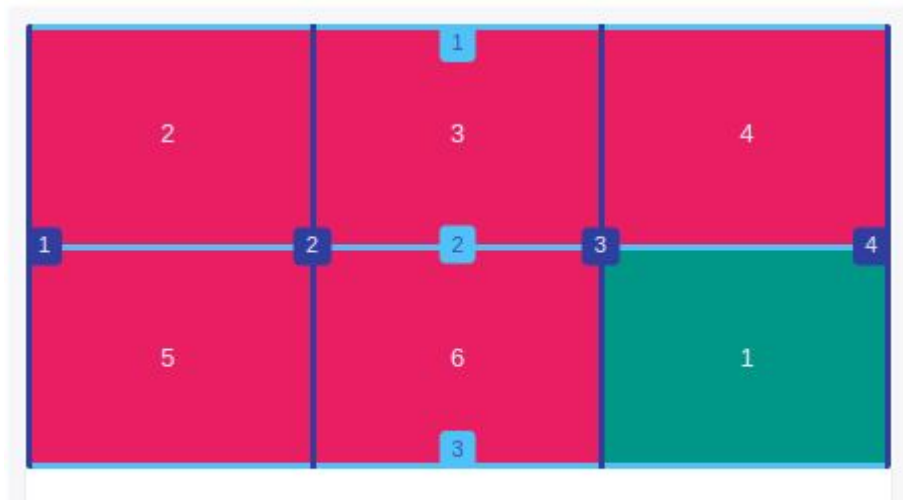
```
.item {  
  grid-column-start: 1;  
  grid-column-end: span 3;  
  grid-row-start: 2;  
  grid-row-end: span 2;  
}
```

Posicionando os itens do grid



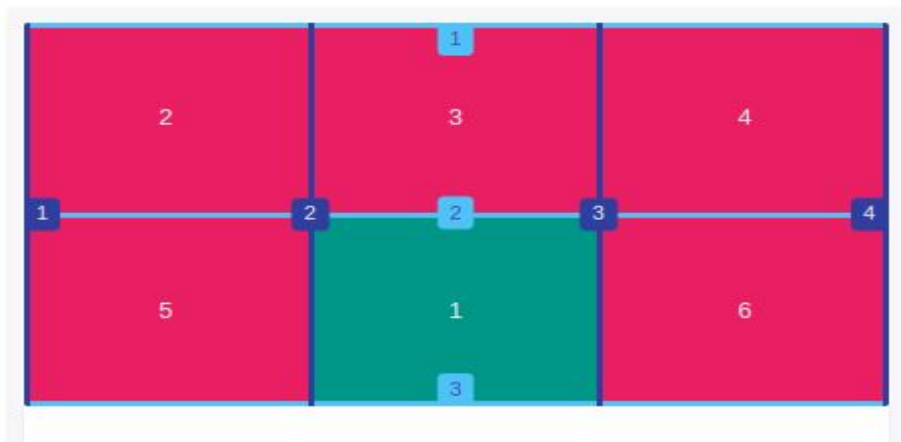
```
.item-1 {  
  grid-row-start: 2;  
  grid-row-end: 3;  
  grid-column-start: 2;  
  grid-column-end: 3;  
}
```

Posicionando os itens do grid



```
.item-1 {  
  grid-row-start: 2;  
  grid-row-end: 3;  
  grid-column-start: 2;  
  grid-column-end: 3;  
}
```

Posicionando os itens do grid



```
.item-1 {  
  grid-area: 2 / 2 / 3 / 3;  
}
```

Etapa 8

Áreas do grid

Shorthand de outras propriedades

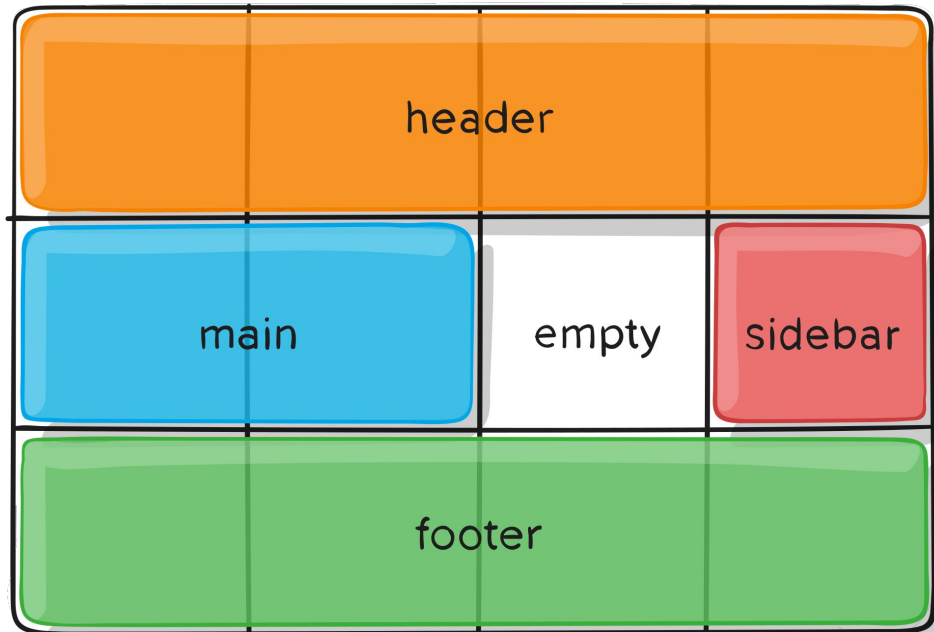
Através da propriedade **grid-area**, nós conseguimos definir os valores das propriedades **grid-row-start**, **grid-row-end**, **grid-column-start** e **grid-column-end** de uma só vez.

```
/* Antes */  
.item {  
  grid-column-start: segunda-linha-col;  
  grid-column-end: span 4;  
  
  grid-row-start: linha-dois;  
  grid-row-end: span 2;  
}  
  
/* Depois */  
.item {  
  grid-area: linha-dois / segunda-linha-col / span 2 / span 4;  
}|
```

grid-area: <grid-row-start>/<grid-column-start>/<grid-row-end>/<grid-column-end>

Nomeando as áreas

É possível nomear as áreas do grid e posicionar os itens dentro das áreas específicas, as referenciando através do nome.

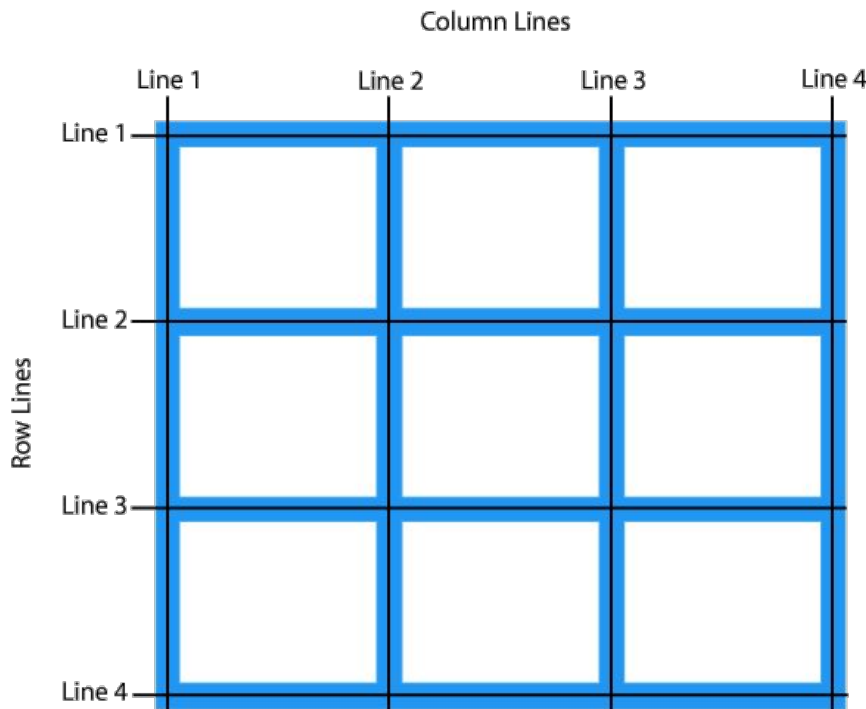


Etapa 10

**Definindo os espaçamentos
dos elementos do grid**

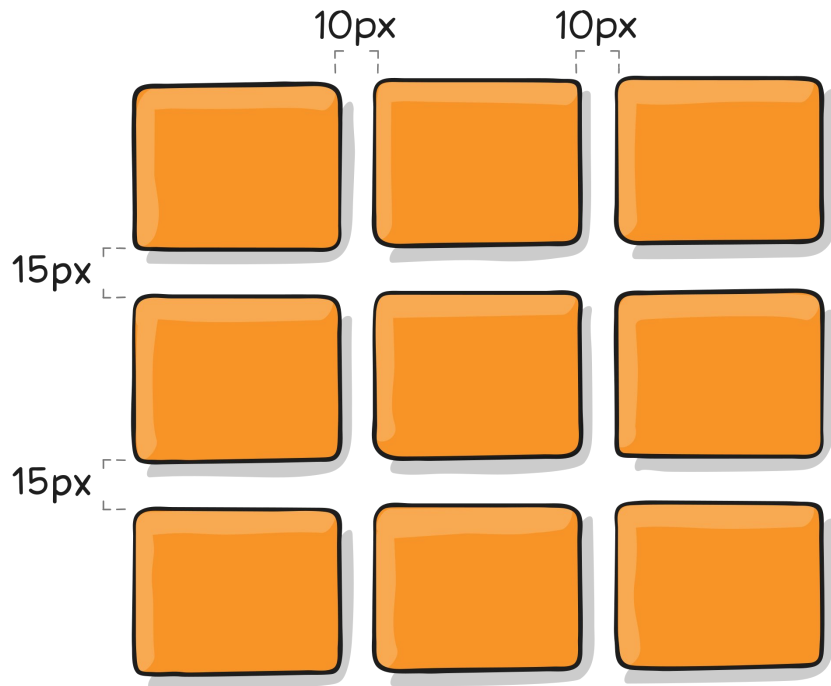
Espaçamento entre as faixas do grid

Tamanho das linhas do grid (*grid lines*) através das propriedades **column-gap** e **row-gap** (ou através da *shorthand gap*)



Espaçamento entre as faixas do grid

Os espaçamentos são adicionados apenas **entre** uma linha/coluna e outra.



Etapa 11

Shorthand “grid”

```
/* Antes */
```

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

```
/* Depois */
```

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



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

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

```
/* Antes */
```

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

```
/* Depois */
```

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

```
/* Antes */
```

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

```
/* Depois */
```

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

```
/* Antes */  
.container {  
  grid-template-areas:  
    "header header header"  
    "footer footer footer";  
  grid-template-rows: 1fr 25px;  
  grid-template-columns: auto 50px auto;  
}
```

```
/* Depois */  
.container {  
  grid: "header header header" 1fr  
        "footer footer footer" 25px  
        / auto 50px auto;  
}
```

```
/* Gera uma linha com 100px de altura e 2 colunas com 1fr. */  
grid: 100px / 1fr 1fr
```

```
/* Gera uma linha com 100px de altura. O grid-auto-flow é definido como column  
(pois está logo antes da definição das colunas). Ele também define o  
grid-auto-columns com 100px 50px*/
```










```
grid: 100px / auto-flow 100px 50px
```

Etapa 12










**Alinhando os itens com
justify-items e align-items**

Valores de justify-items

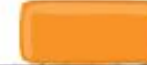
justify-items: start;










justify-items: center;

justify-items: end;

justify-items: stretch;

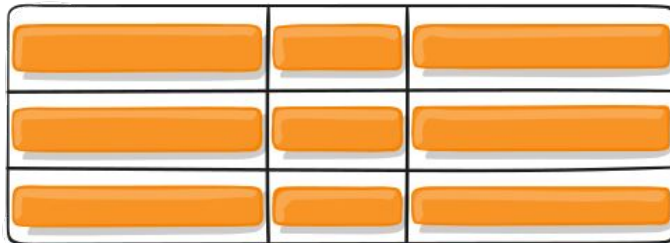
		
		
		

Valores de align-items

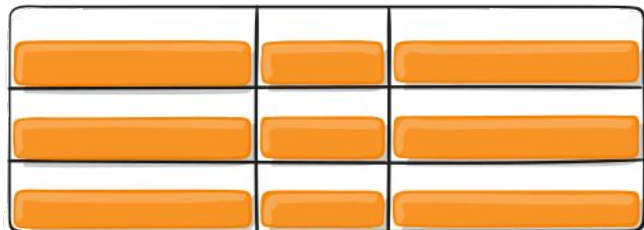
align-items: start;



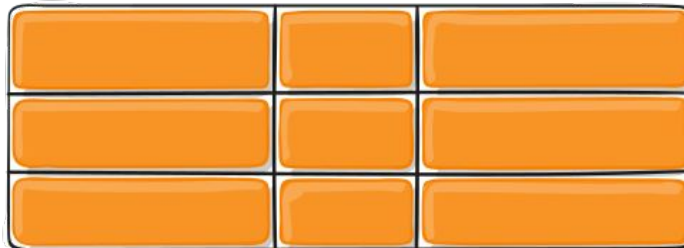
align-items: center;



align-items: end;



align-items: stretch;

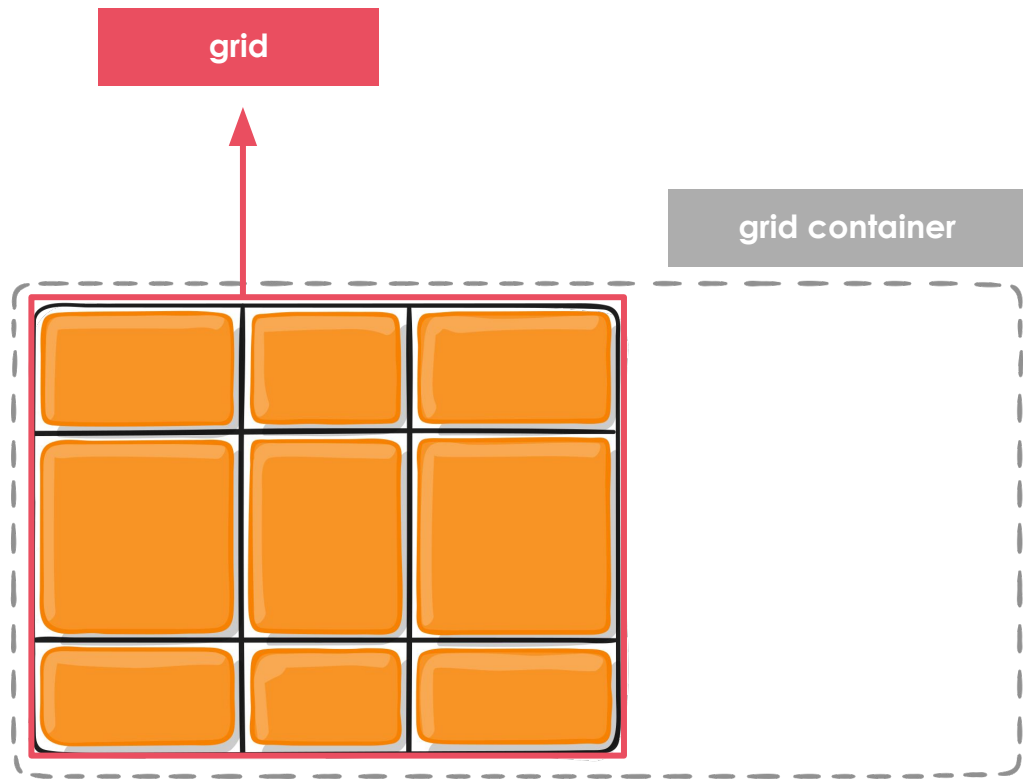


Etapa 14

**Alinhando o conteúdo do grid
com justify-content e
align-content**

Container do grid

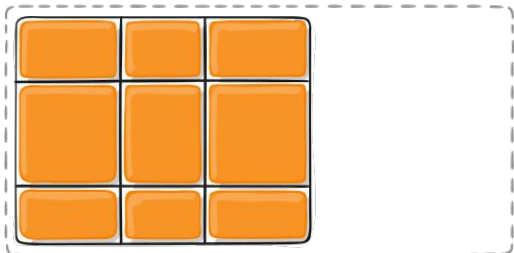
Container é o elemento que envolve a grade na qual estamos criando e, muitas vezes, ela acaba sendo maior do que o grid.



Valores de justify-content

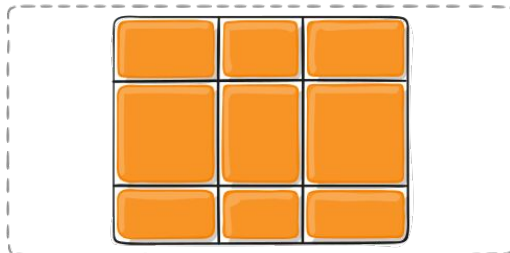
justify-content: start;

grid container



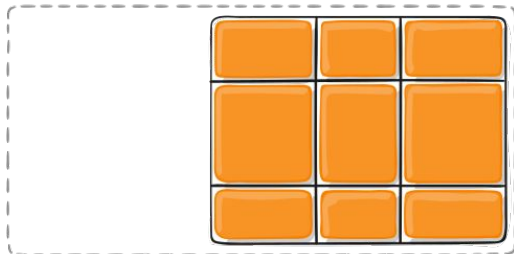
justify-content: center;

grid container



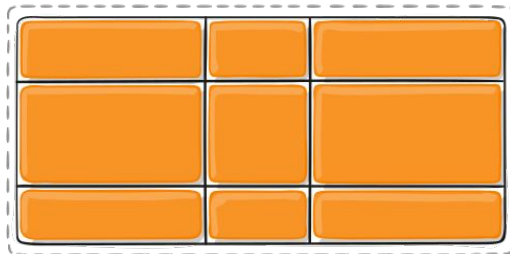
justify-content: end;

grid container



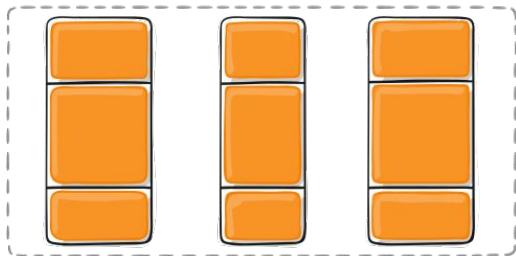
justify-content: stretch;

grid container

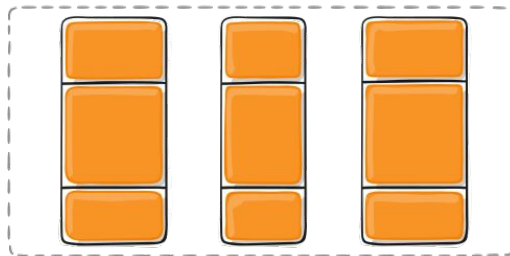


Valores de justify-content

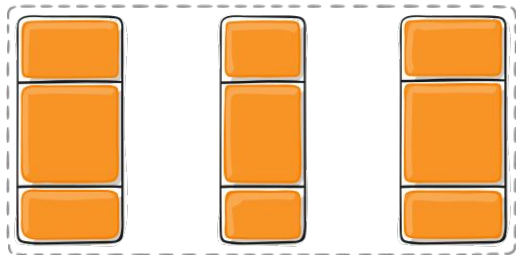
justify-content: space-around;
grid container



justify-content: space-evenly;
grid container



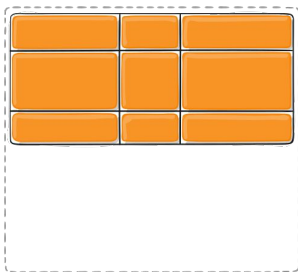
justify-content: space-between;
grid container



Valores de align-content

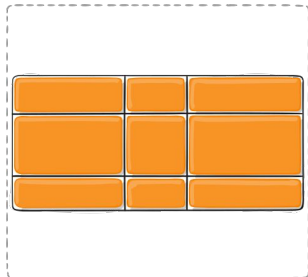
align-content: start;

grid container



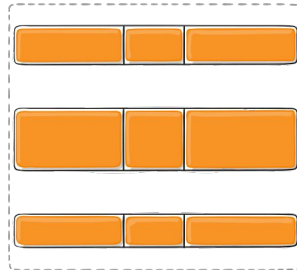
align-content: center;

grid container



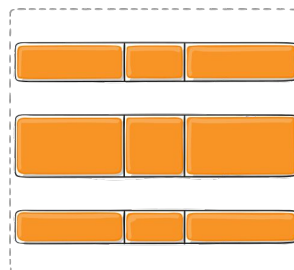
align-content: space-around;

grid container



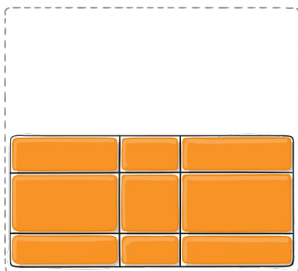
align-content: space-evenly;

grid container



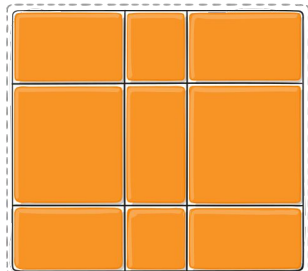
align-content: end;

grid container



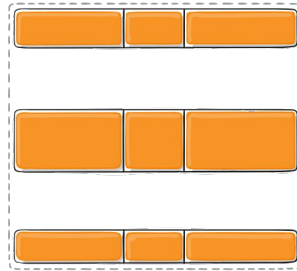
align-content: stretch;

grid container



align-content: space-between;

grid container



Etapa 16

**Alinhando itens específicos
com justify-self e align-self**

Valores de justify-self

justify-self: start;

.item-a		

justify-self: center;

.item-a		

justify-self: end;

	.item-a	

justify-self: stretch;

.item-a		

Valores de align-self

align-self: start;

.item-a		

align-self: center;

.item-a		

align-self: end;

.item-a		

align-self: stretch;

.item-a		

Materiais de apoio

- 🔗 [Grid Garden](#)
- 🔗 [CSS Grid Layout Module - W3Schools](#)
- 🔗 [A Complete Guide to Grid](#)
- 🔗 [CSS Grid Layout](#)
- 🔗 [Grid Guide \(Origamid\)](#)