

Display

Ou comment mettre en forme nos pages web

La propriété CSS **display** est la plus importante pour contrôler la mise en page.

Chaque élément a une **valeur display par défaut** dépendante du type de l'élément.

Les valeurs par défaut de la plupart des éléments sont **block** ou **inline**.

Block

Un élément « block-level » commence sur une nouvelle ligne et s'étire autant qu'il peut sur la gauche et la droite.

Les éléments block-level courant par défaut :

<div>

<p>

<header>

<footer>

<section>

Inline

Un élément inline peut encadrer du texte dans un paragraphe sans interrompre le flux de ce paragraphe.

Les éléments block-level courant par défaut :

``

`<a>`

Ce que ça rend

display: inline:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. **HELLO WORLD!** Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

display: block:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit.

HELLO WORLD!

Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

Plusieurs blocks

display : block



display : inline-block



Flex

Pour aller plus loin

Flex

Mais qu'est ce donc ?

- Un module de mise en forme **uni-directionnel**
- A son propre flux (**flex flow**)
- Impact un **conteneur** et ses **enfants directs**



Propriétés du conteneur

display: **flex** | **inline-flex**;

Déclaration de l'élément en flexible

flex (type bloc)



inline-flex (type inline)



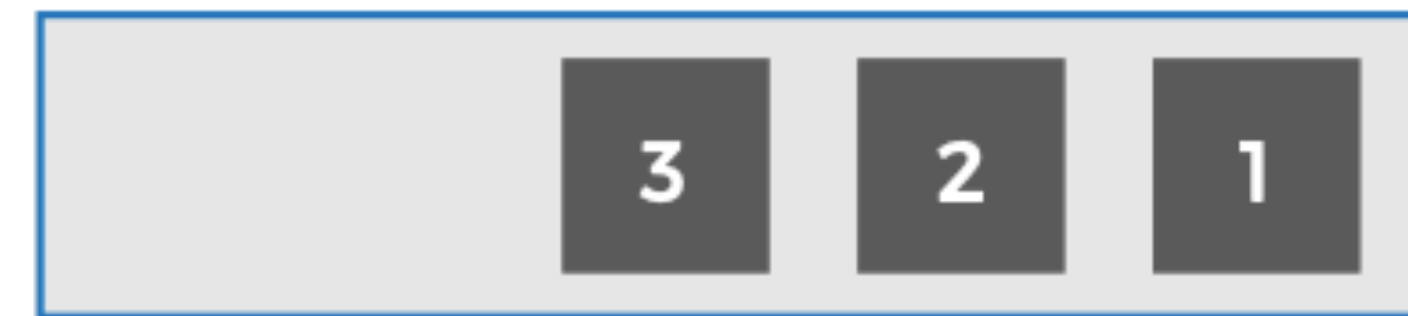
flex-direction

Défini l'axe principal (main-axis) et sa direction

row*



row-reverse



column



column-reverse

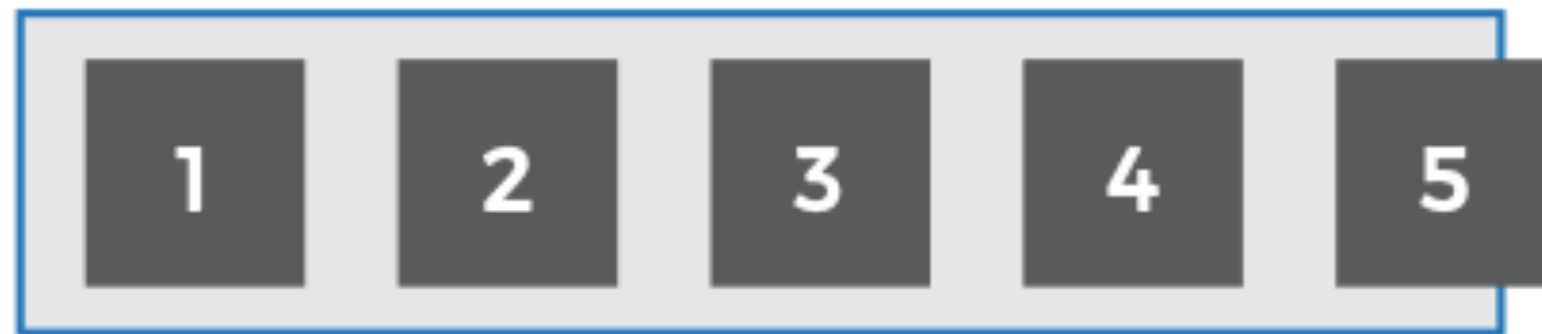


* valeur par défaut

flex-wrap

Les flex items restent sur une ligne ou non

nowrap*



wrap



wrap-reverse



flex-flow

Propriété raccourcie de flex-direction et flex-wrap

`flex-flow: <flex-direction> <flex-wrap>;`

justify-content

Répartition des espaces entre et autour des éléments flexibles sur l'axe principal (main-axis)

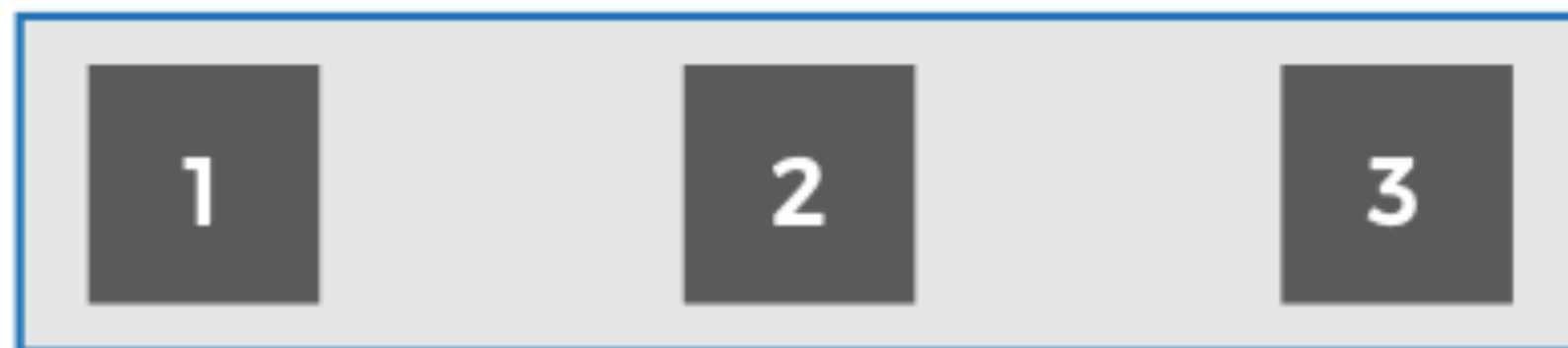
flex-start*



flex-end



space-between



space-around



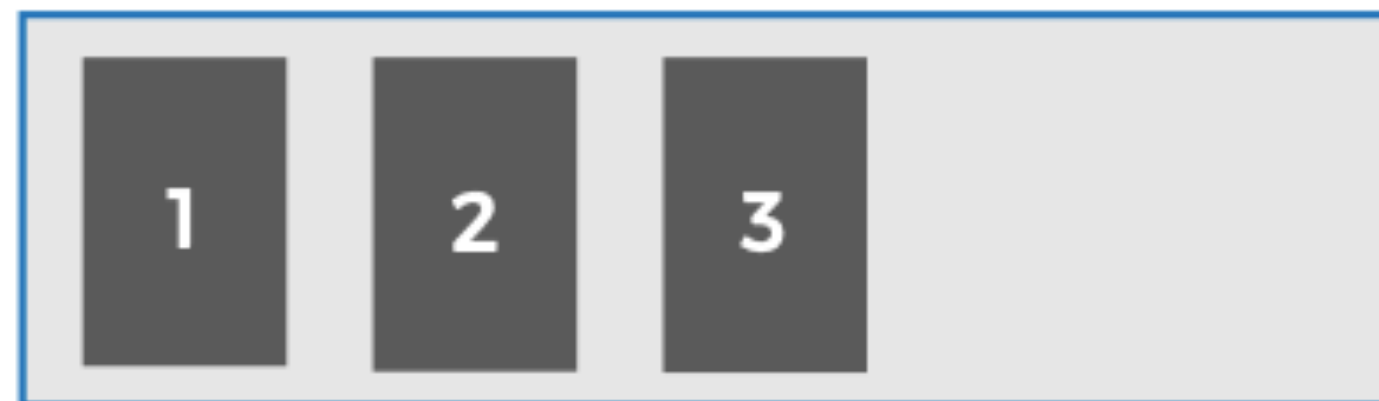
center



align-items

Répartition des espaces autour des éléments flexibles sur l'axe secondaire (cross-axis)

normal/stretch*



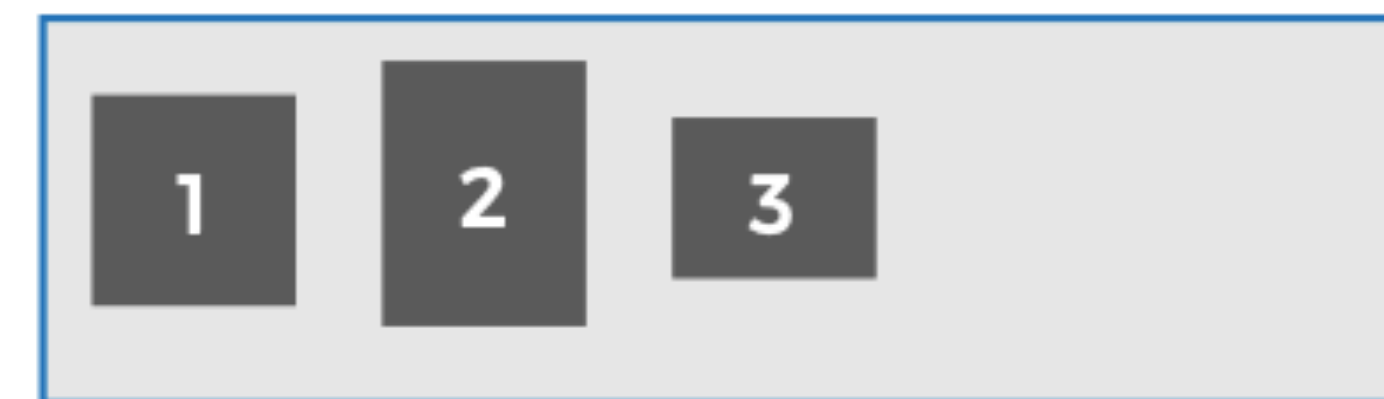
flex-start



flex-end



center



baseline



* valeur par défaut

Baseline

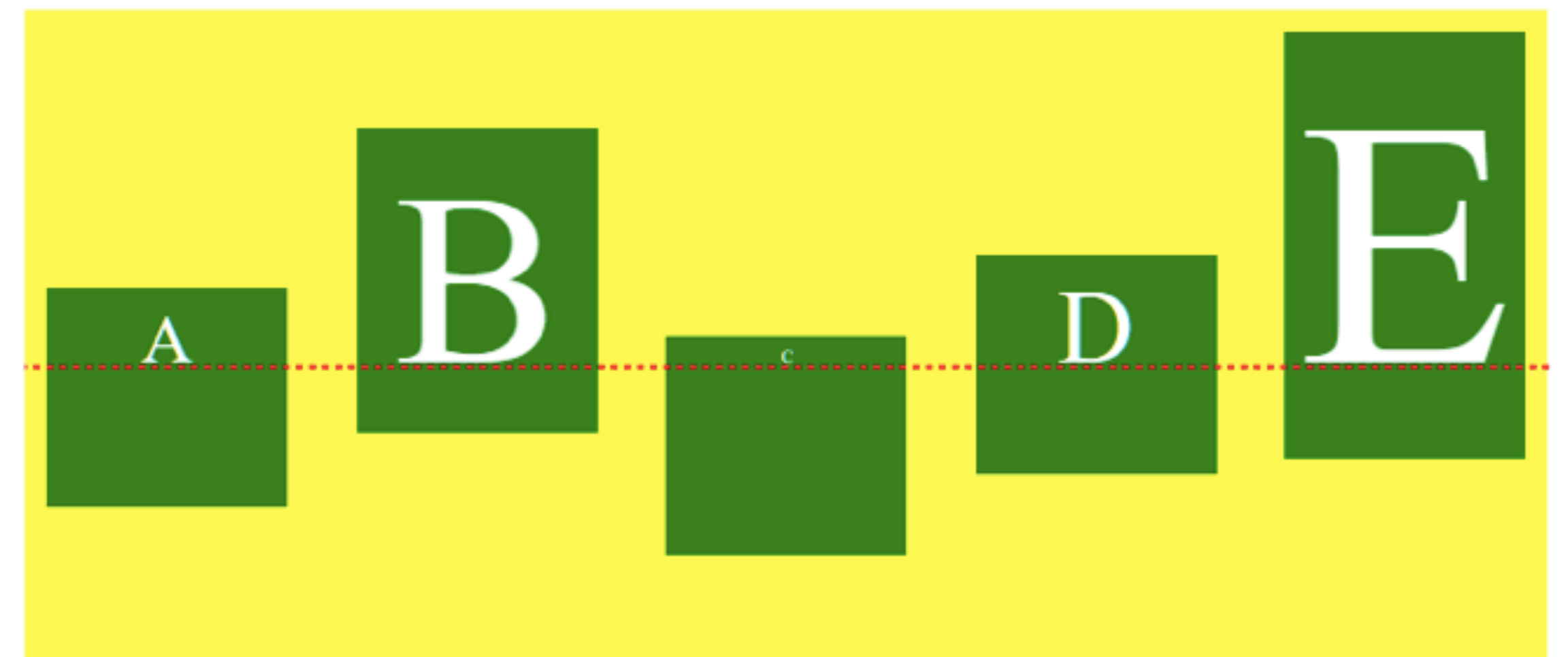
Quelle est la différence avec flex-start ?

- Si toutes les lettres ont la même taille, alors la valeur « baseline » est identique à « flex-start »
- Sinon les éléments seront alignés suivant la ligne de base d'écriture.

`align-items: flex-start`

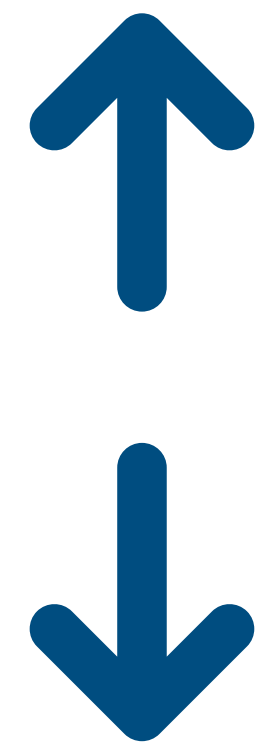


`align-items: baseline`

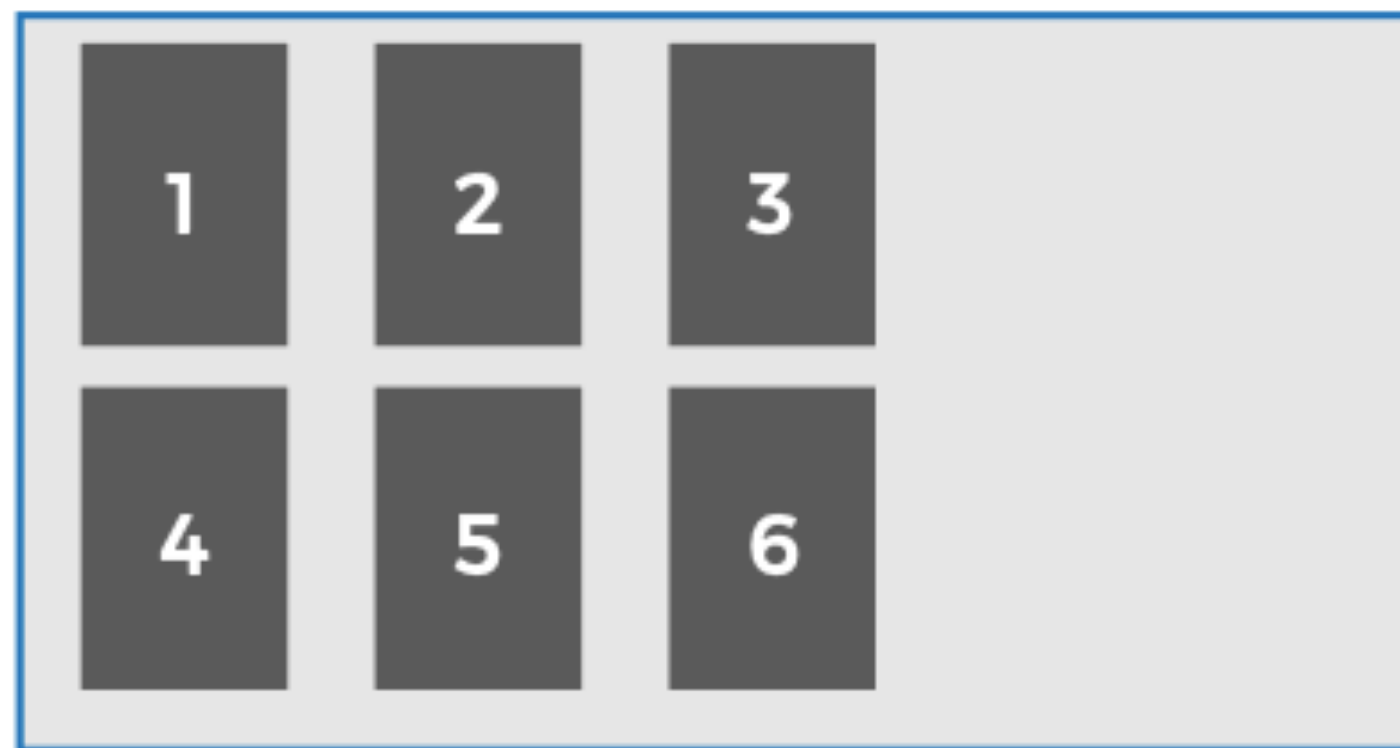


align-content

Répartition des espaces autour et entre des éléments flexibles sur l'axe secondaire (cross-axis)



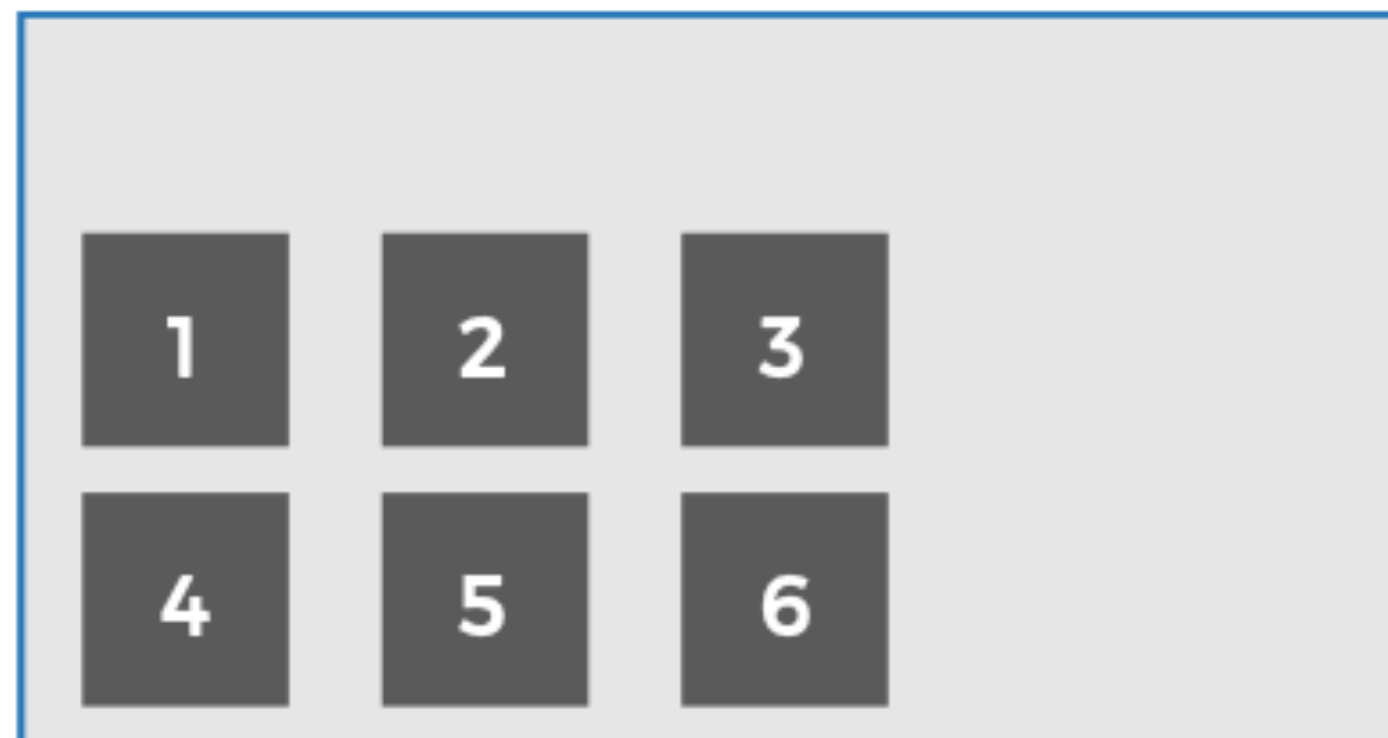
normal/stretch*



flex-start



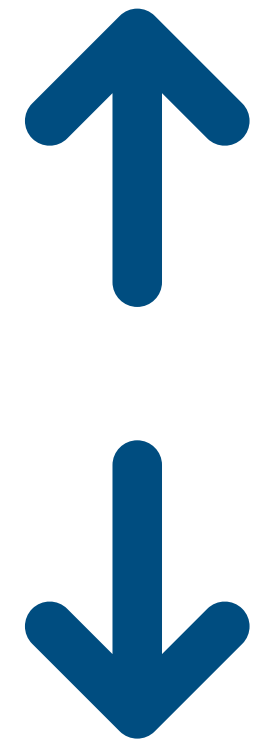
flex-end



* valeur par défaut

align-content

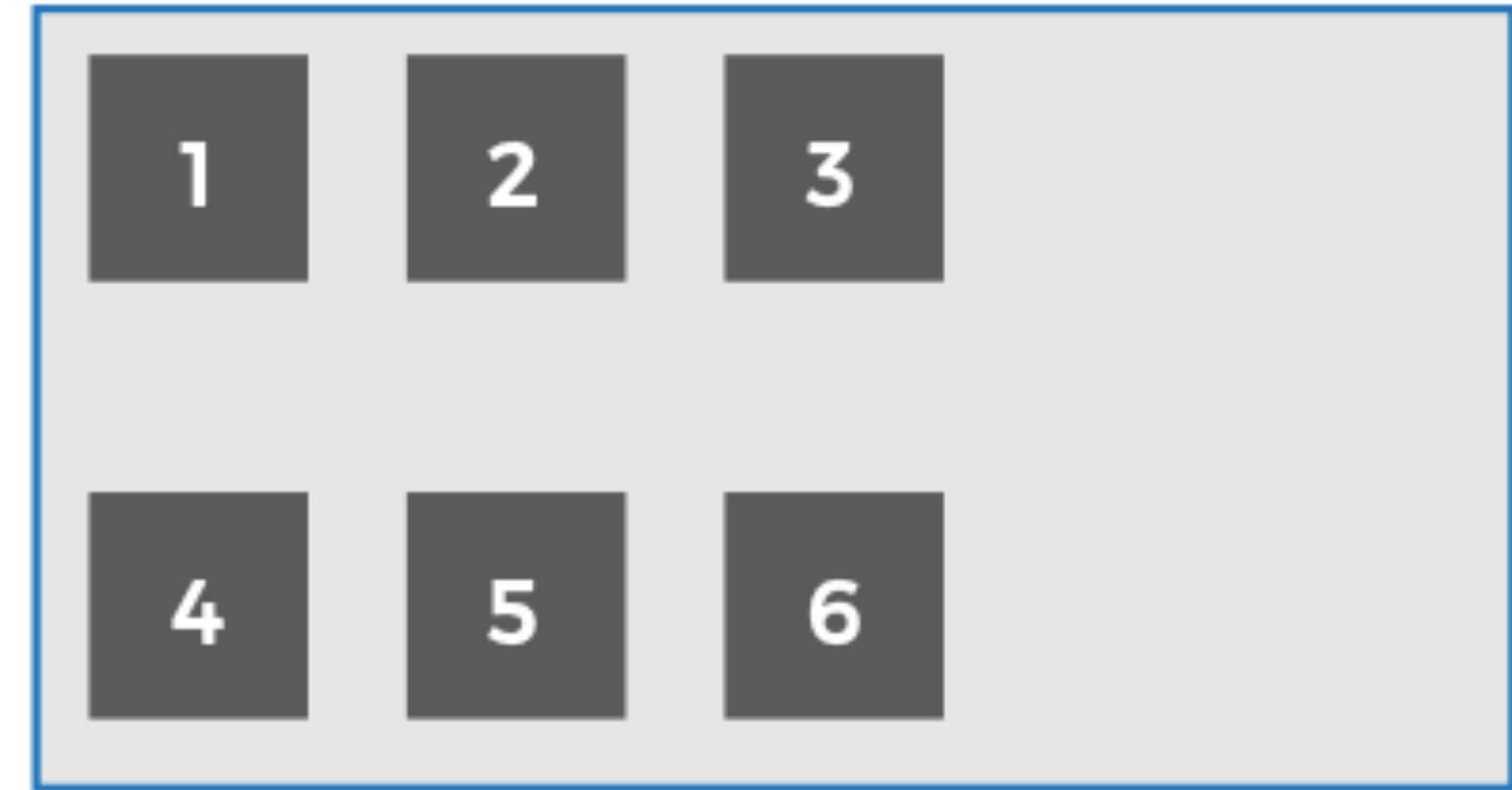
(Suite)



center



space-between



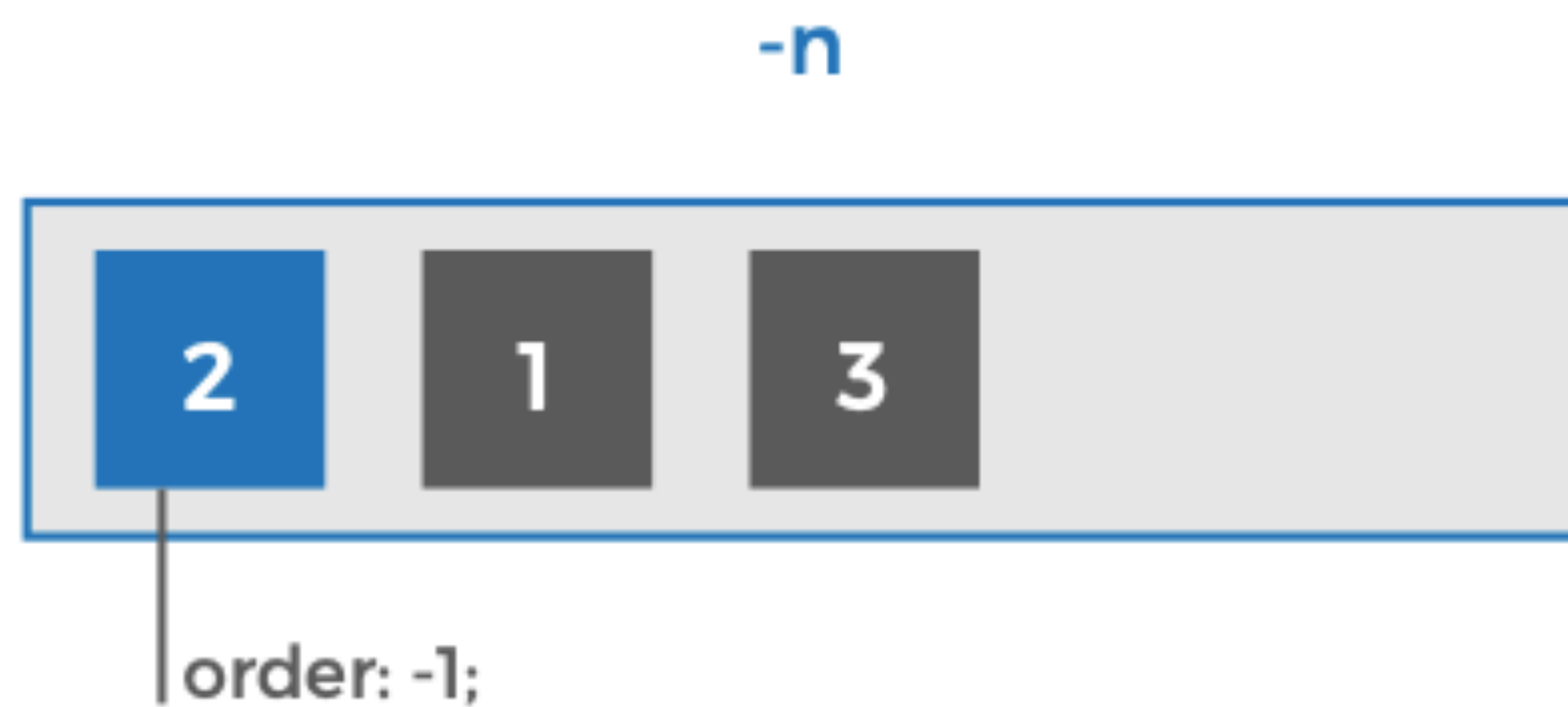
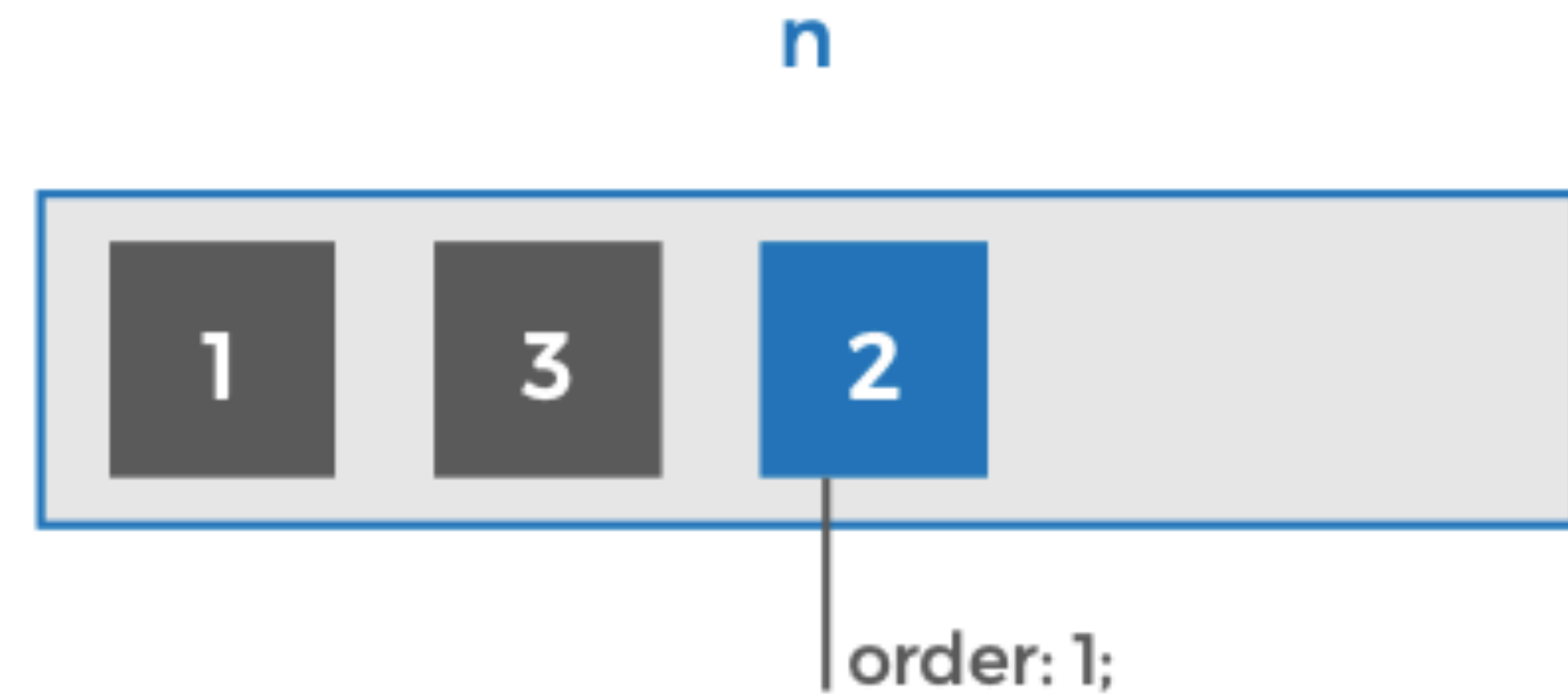
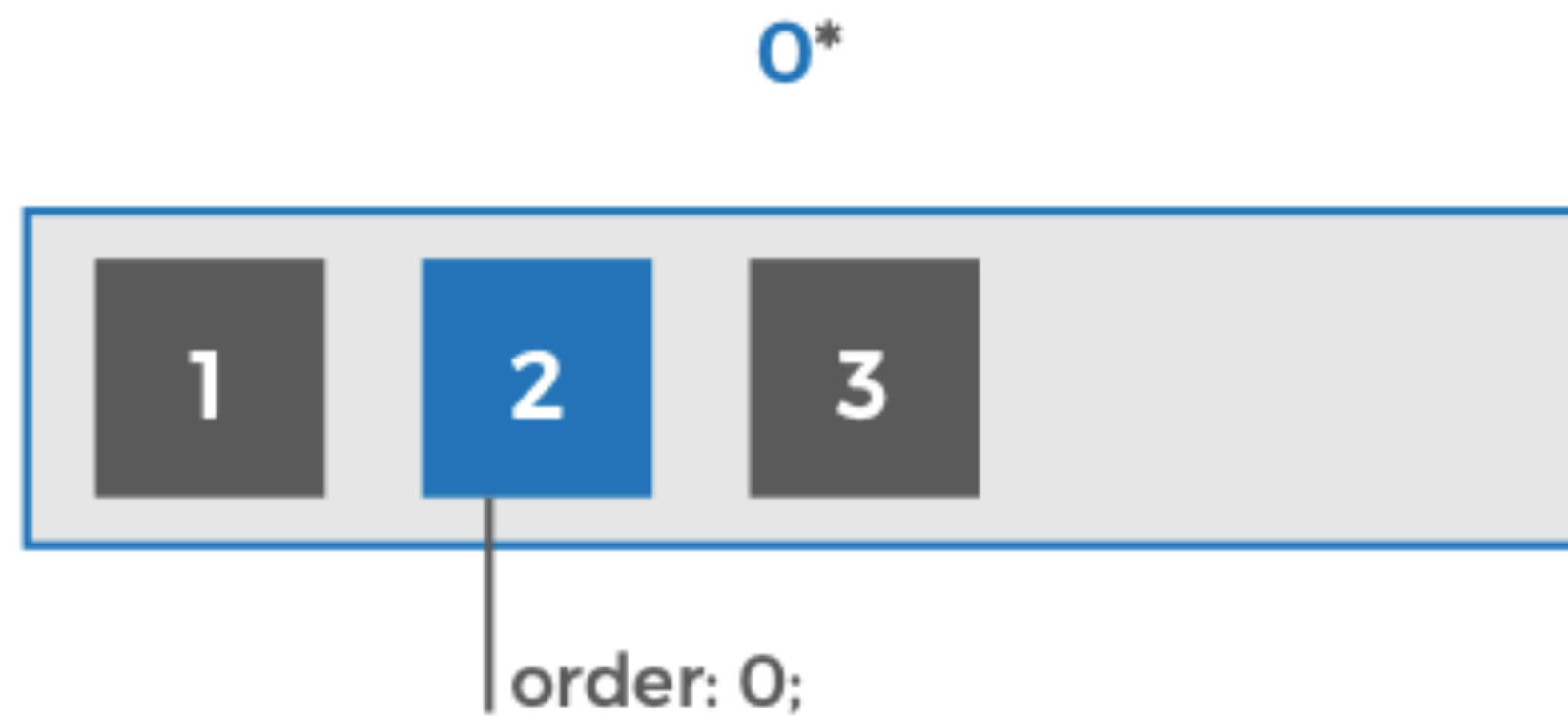
space-around



Propriétés des éléments flexibles

order

Défini l'ordre d'apparition des éléments (nombre entier)



flex-grow

Indique le facteur de grossissement d'un élément flexible (nombre positif)

1



flex-grow: 1;

2



flex-grow: 1;

flex-grow: 2;

flex-shrink

Indique le facteur de rétrécissement d'un élément flexible (nombre positif)



flex-basis

Détermine la taille de l'élément flexible sur l'axe principal (main axis)

auto*

(défini par la valeur de width ou height)

largeur

90% 90% du container

200px

10rem 1 rem = 16px

15vh vh = viewport height
15 % de la hauteur de la fenêtre

mots clés

content

min-content

max-content

Compatibilité ?

[Update compatibility data on GitHub](#)

	Desktop						Mobile					
	Chrome	Edge	Firefox	Internet Explorer	Opera	Safari	Android webview	Chrome for Android	Firefox for Android	Opera for Android	Safari on iOS	Samsung Internet
flex-basis	29 ▼	12 ▼	22 * ▼	11 * ▼	12.1 ▼	9 ▼	4.4 ▼	29 ▼	22 * ▼	12.1 ▼	9 ▼	2.0 ▼
auto	22	12	18	11	12.1	9 ▼	≤37	25	18	12.1	9 ▼	1.5
content	No	12 — 79	61	No	No	No	No	No	61	No	No	No
max-content	No	No	66 ▼	No	No	No	No	No	66 ▼	No	No	No
min-content	No	No	66 ▼	No	No	No	No	No	66 ▼	No	No	No

What are we missing?



Full support



No support

★

See implementation notes.



User must explicitly enable this feature.

-x-

Requires a vendor prefix or different name for use.

<https://developer.mozilla.org/en-US/docs/Web/CSS/flex-basis>

flex

Propriété raccourcie de flex-grow, flex-shrink et flex-basis

```
flex: <flex-grow> <flex-shrink> <flex-basis>;
```

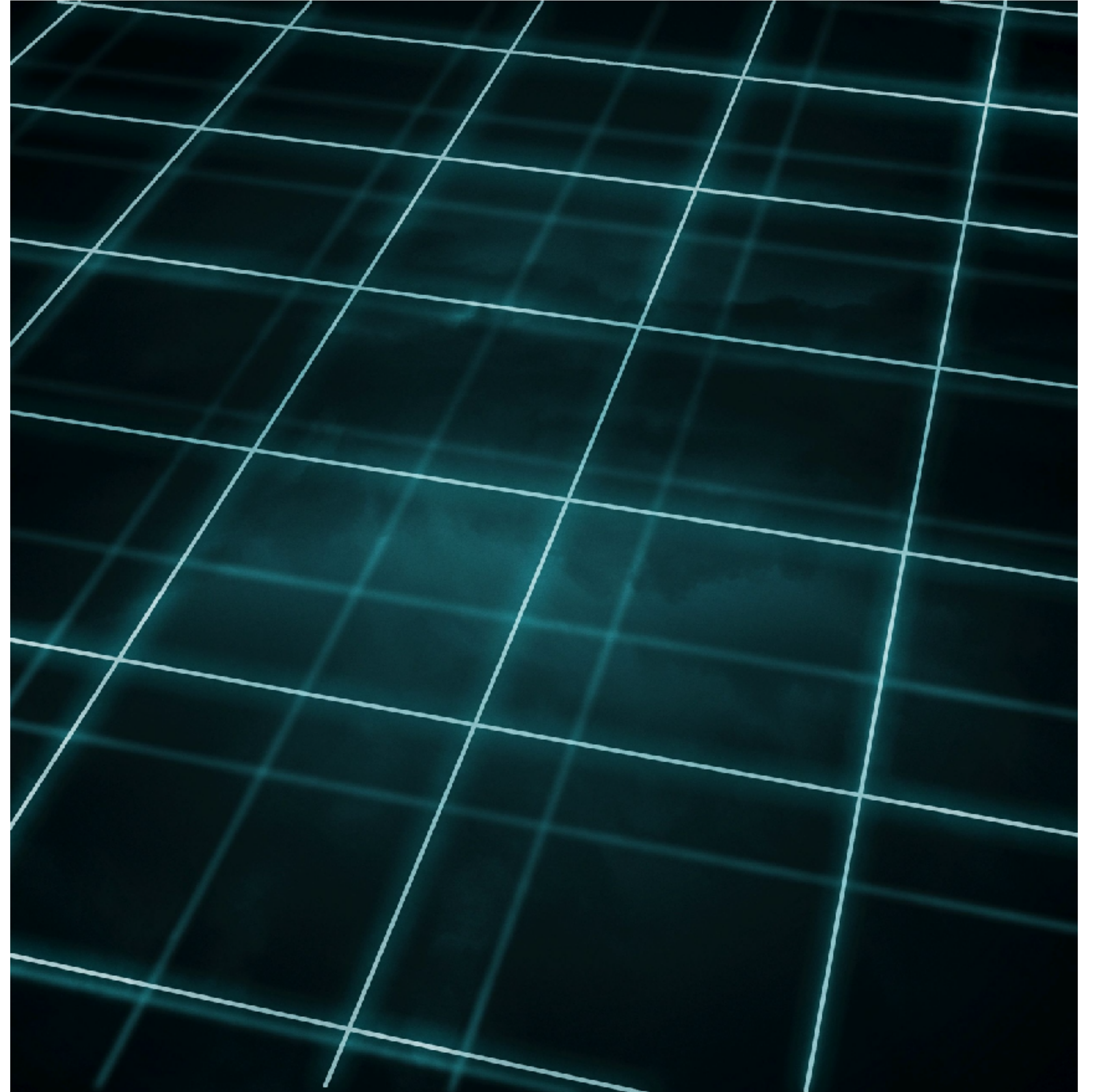
Grid

Pour compléter le flex

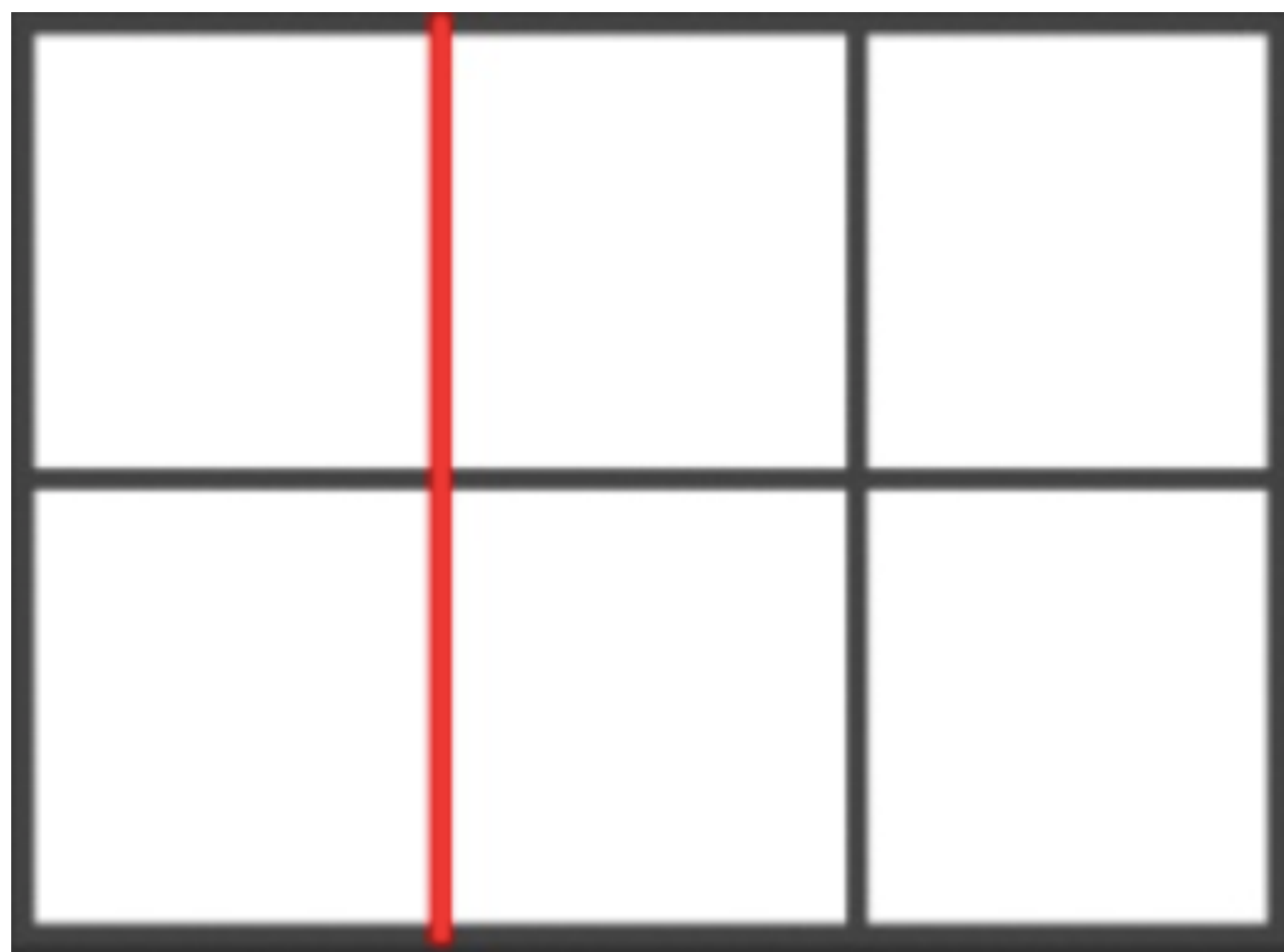
Grid

Mais qu'est ce donc ?

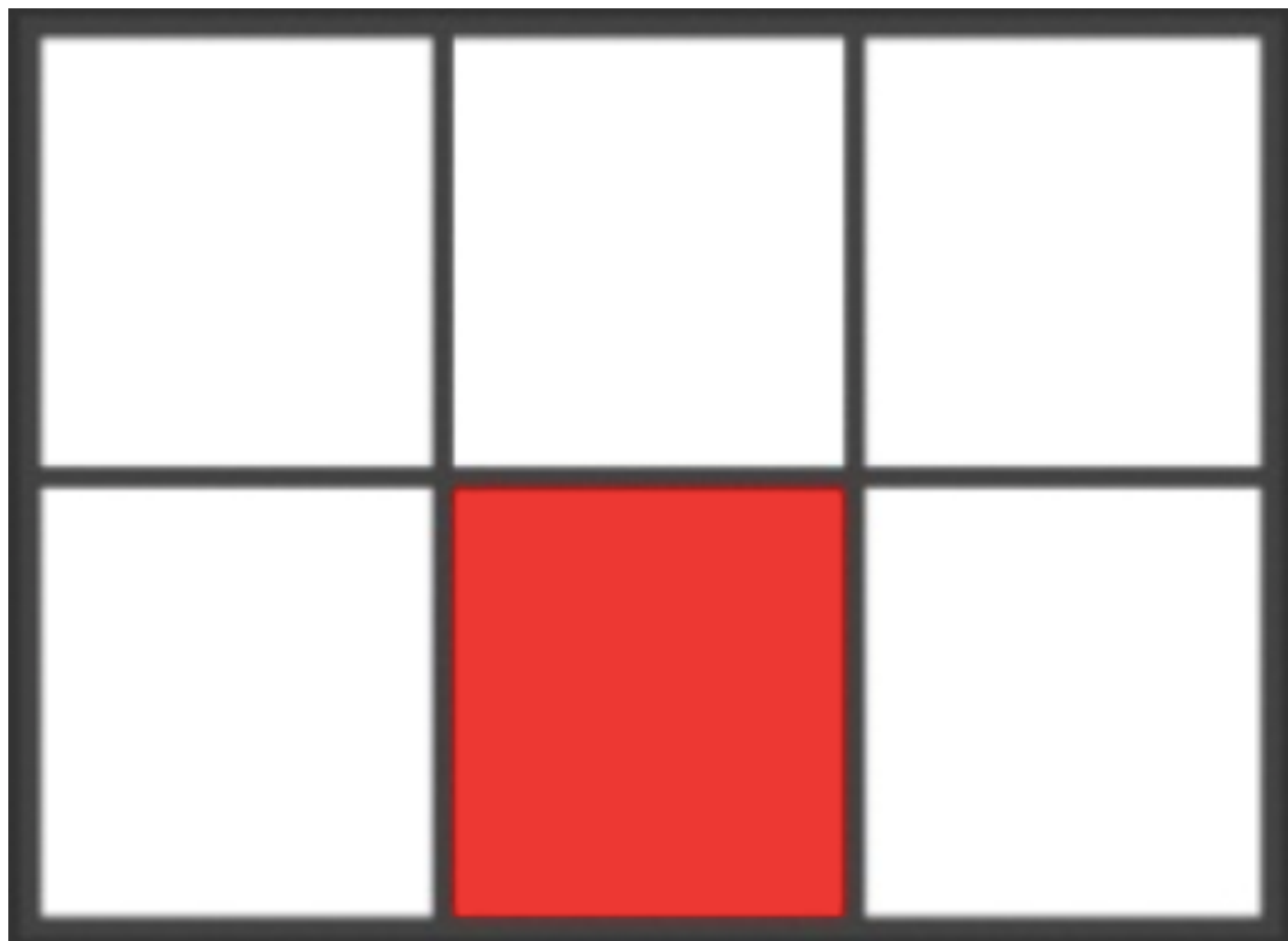
- Un module de mise en forme **multi-directionnel**
- Impact un **conteneur** et ses **enfants directs**



Grid terminology



Grid Line



Grid Cell



Grid Area

Propriétés de la grille

display: **grid** | **inline-grid**;

Déclaration de l'élément en grille

grid (type bloc)

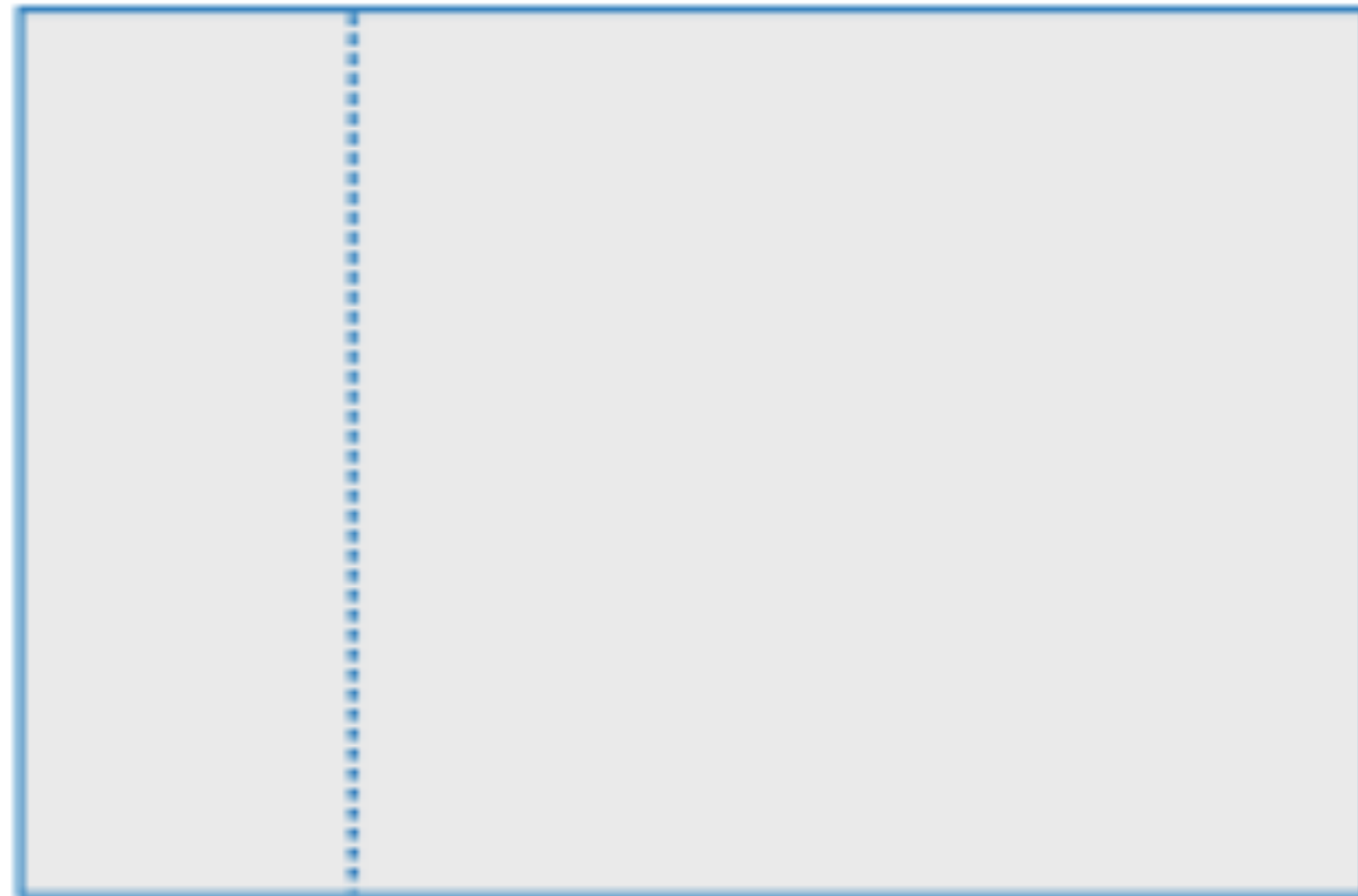


inline-grid (type inline)



grid-template-columns

Défini le nom et les tailles des colonnes de la grille



none* (pas de grille explicite)

`grid-template-columns: none;`

<longueur> (px, %, em, fr, ...)

`grid-template-columns: 100px;`

[nom]

`grid-template-columns: [start] 100px;`

repeat()

`grid-template-columns: repeat(3, 1fr);`

minmax()

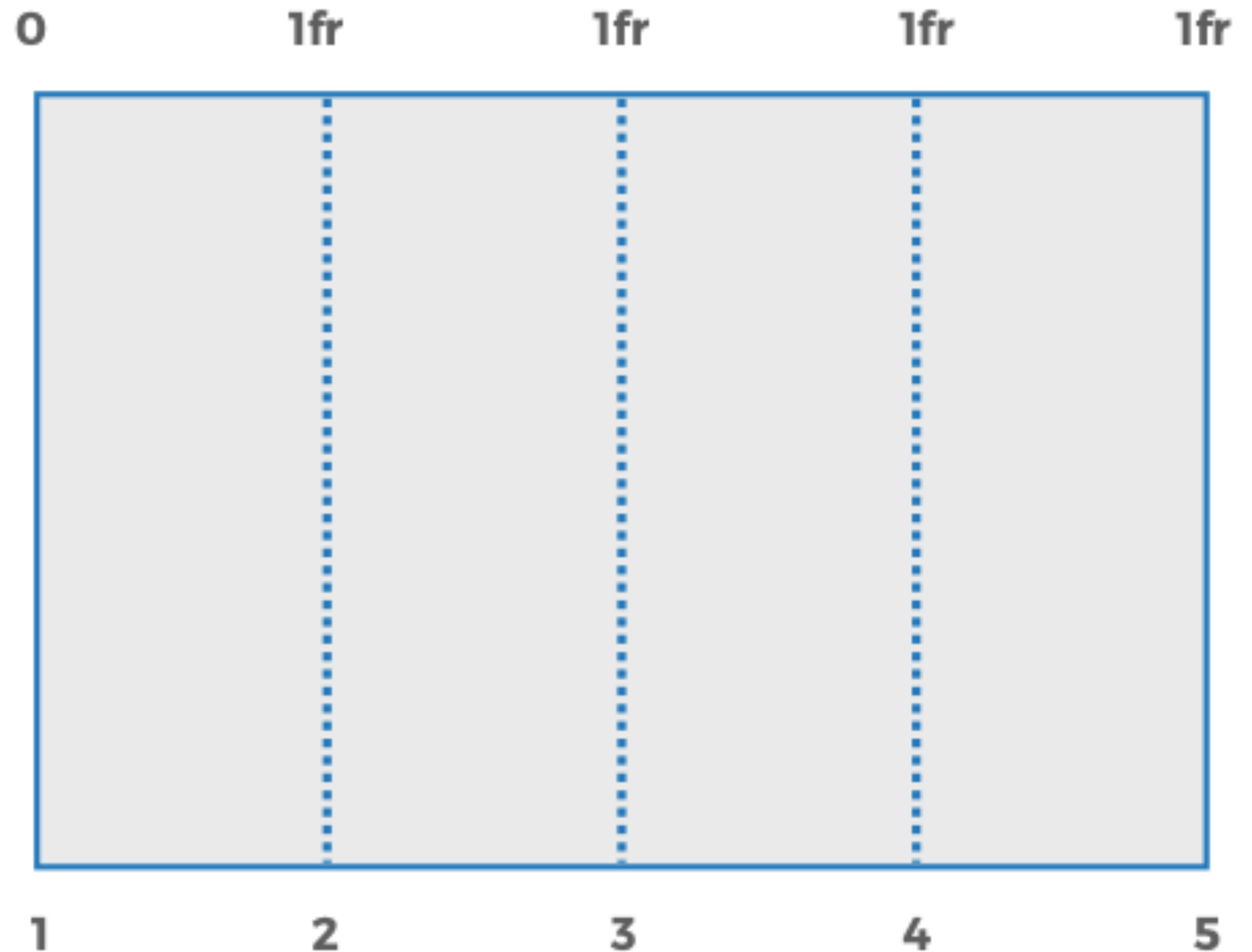
`grid-template-columns: minmax(100px, 30%);`

grid-template-columns

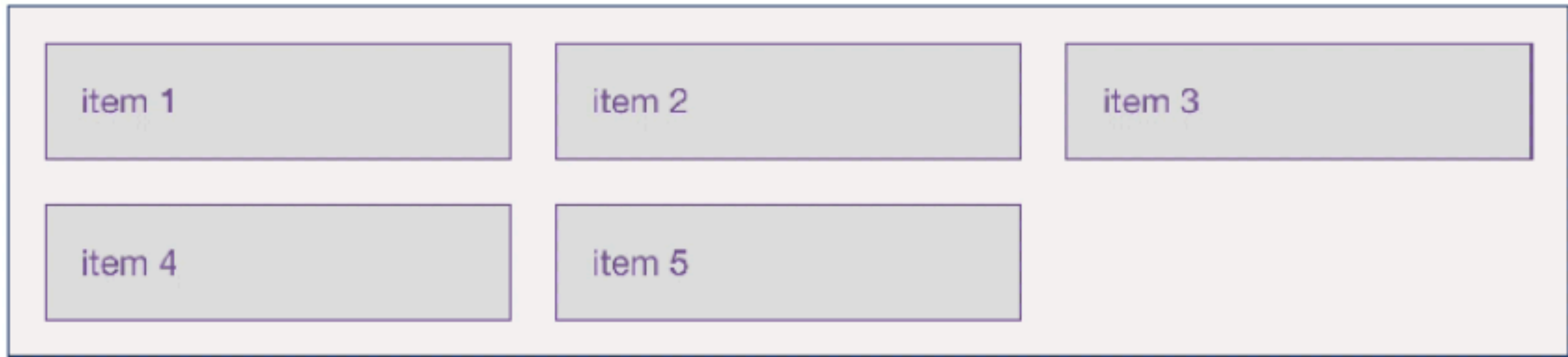
grid-template-columns: [col1-start] 1fr [col1-end col2] 1fr [col3] 1fr 1fr;

ou

grid-template-columns: repeat(4, 1fr);



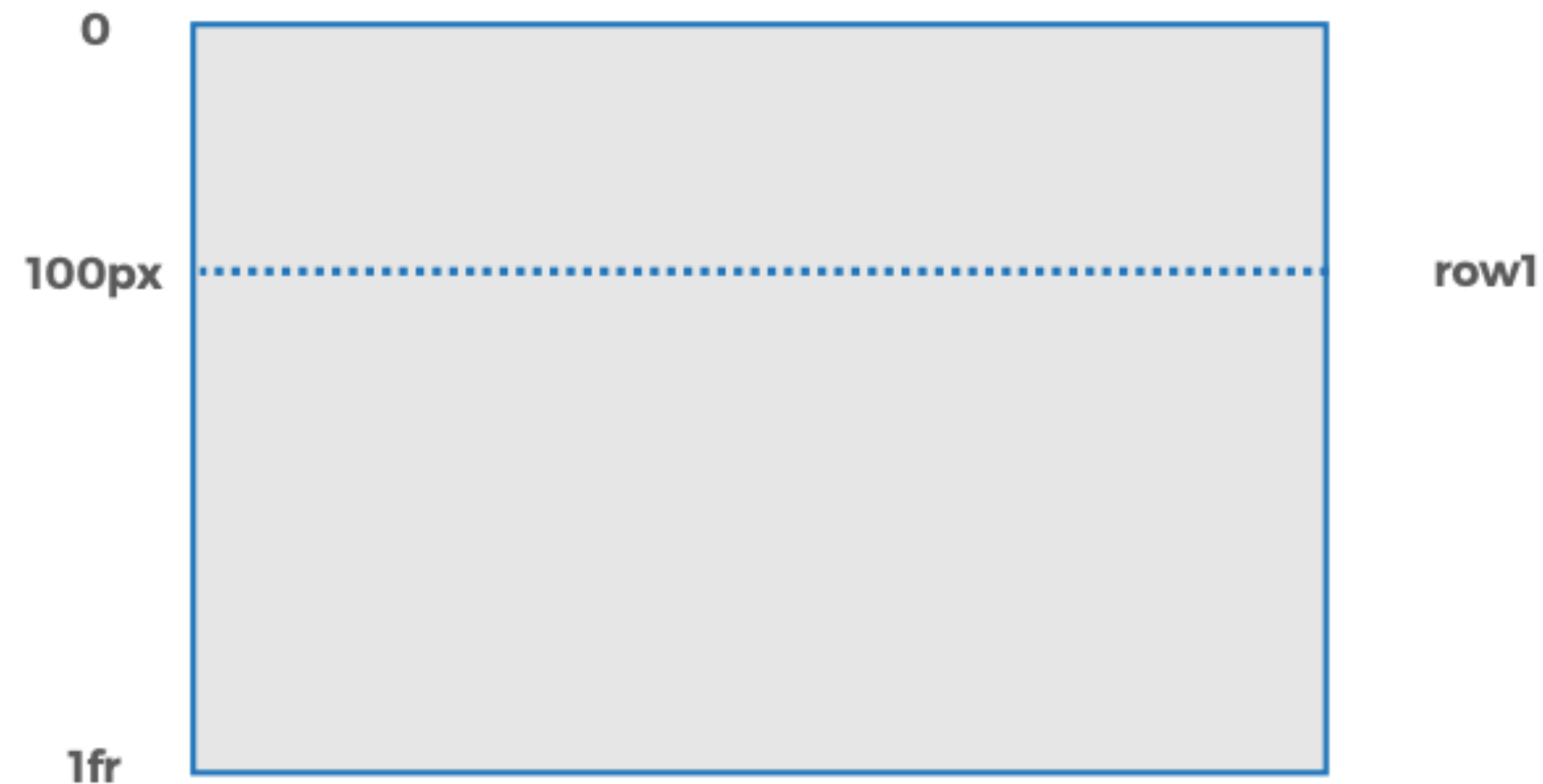
grid-template-columns



grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

grid-template-rows

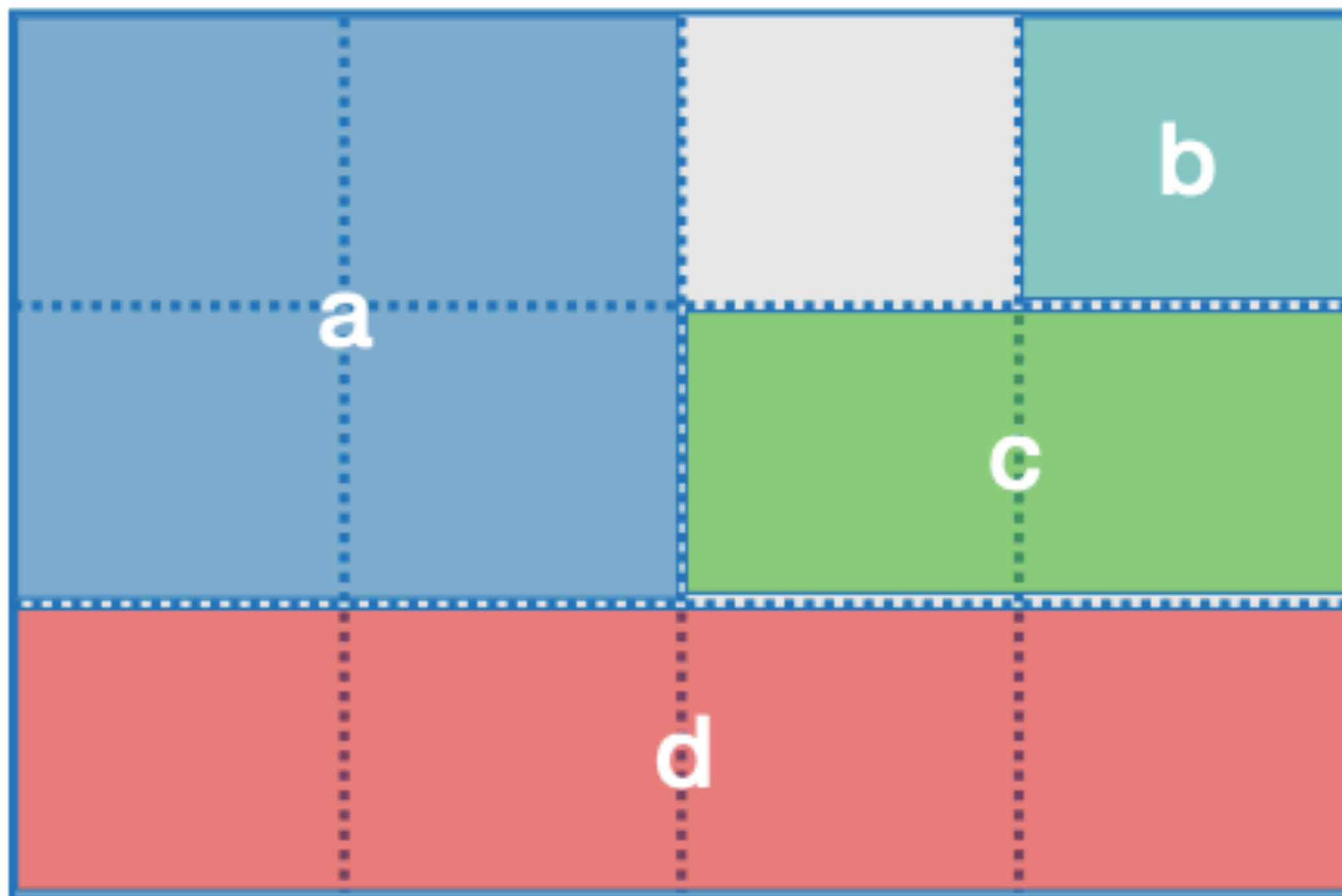
Défini le nom et les tailles des lignes de la grille



```
grid-template-rows: [row1] 100px minmax(100px, 1fr);
```

grid-template-areas

Nomme des zones de grilles



```
grid-template-columns: repeat(4, 1fr);
```

```
grid-template-rows: repeat(3, 1fr);
```

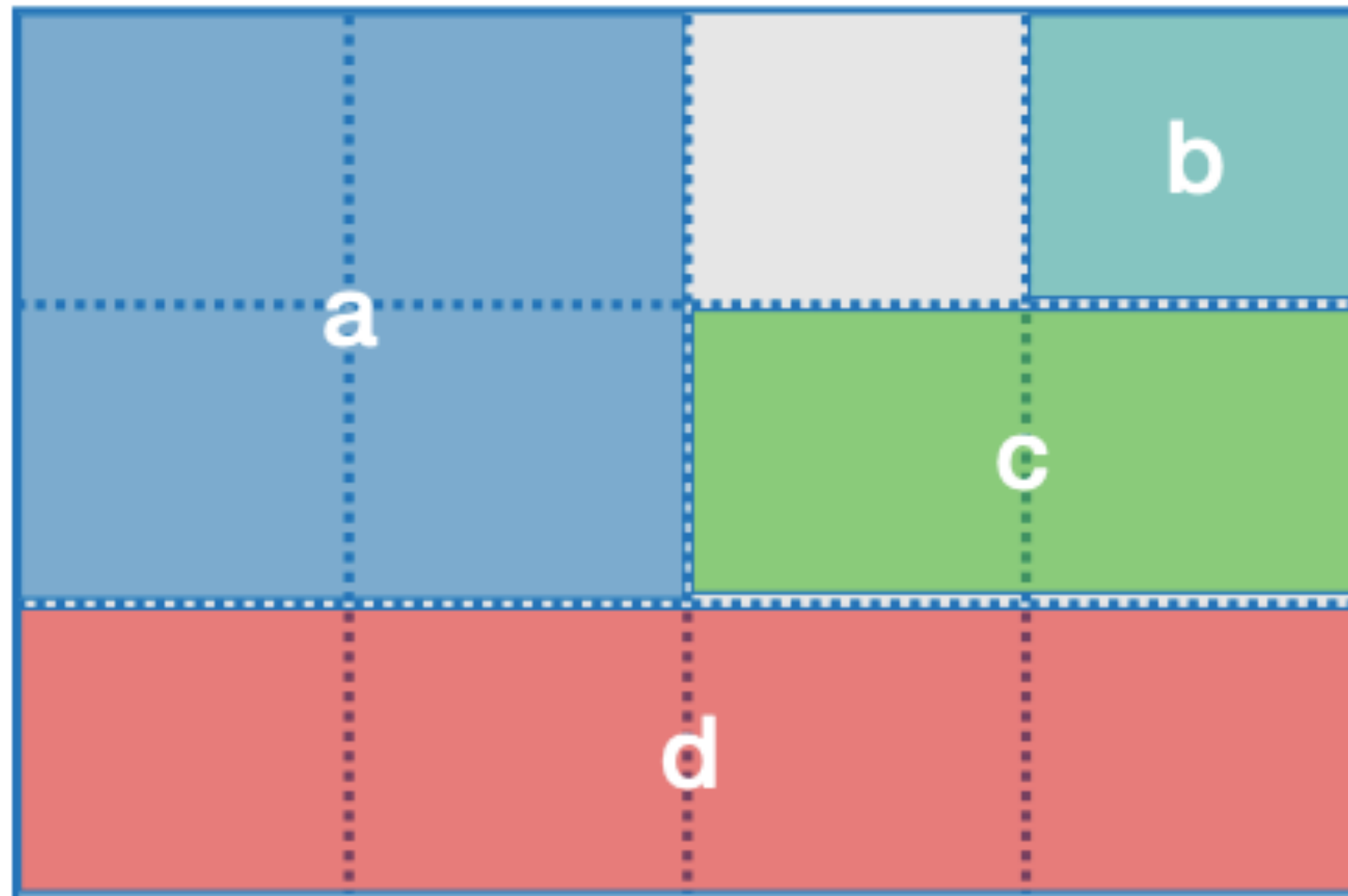
```
grid-template-areas: "a a . b"
```

```
"a a c c"
```

```
"d d d d";
```

grid-template

Propriété raccourcie de grid-template-columns, grid-template-rows et grid-template-areas



grid-template: "a a . b" 1fr

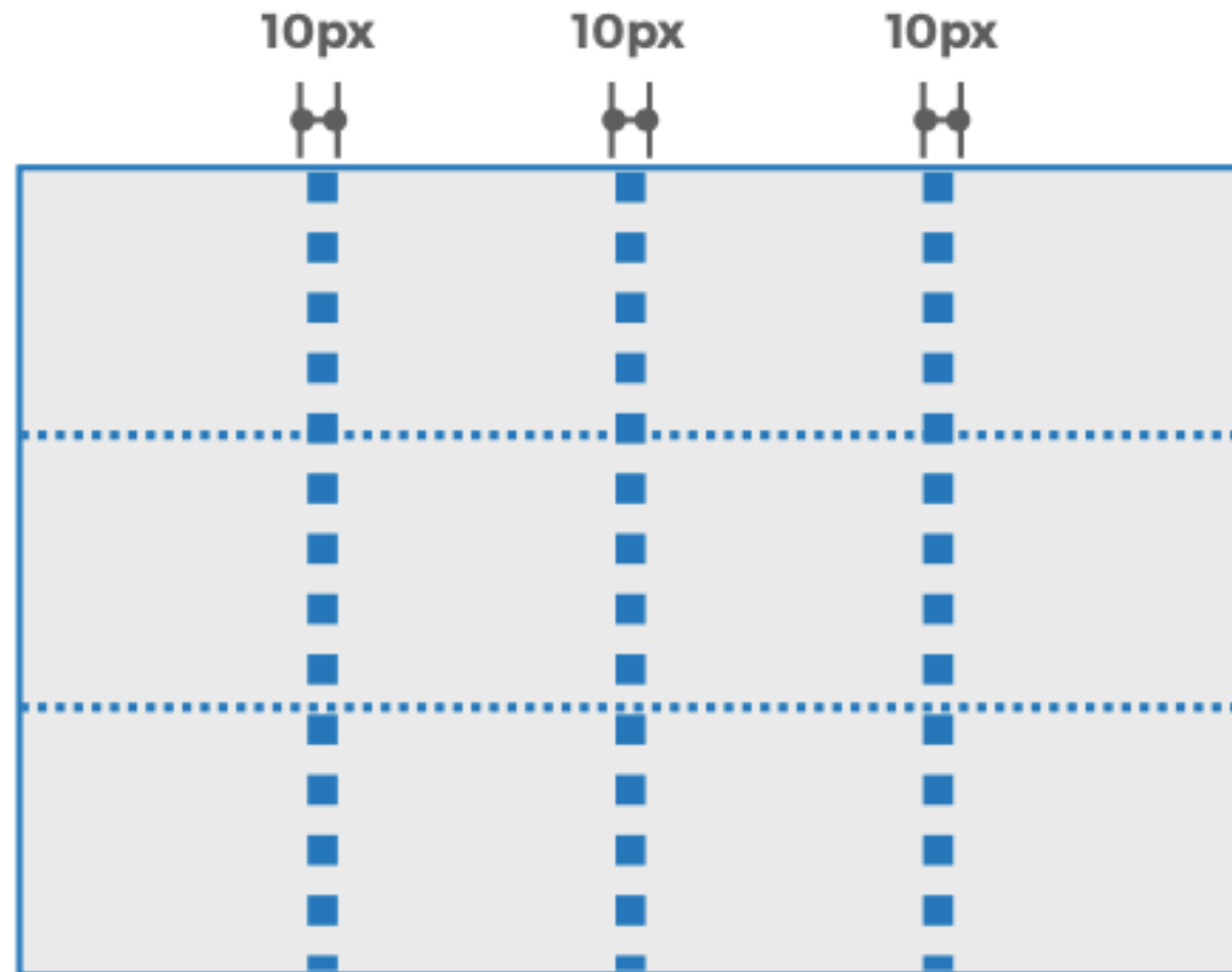
"a a c c" 1fr

"d d d d" 1fr

/ repeat(4, 1fr);

Grid-column-gap

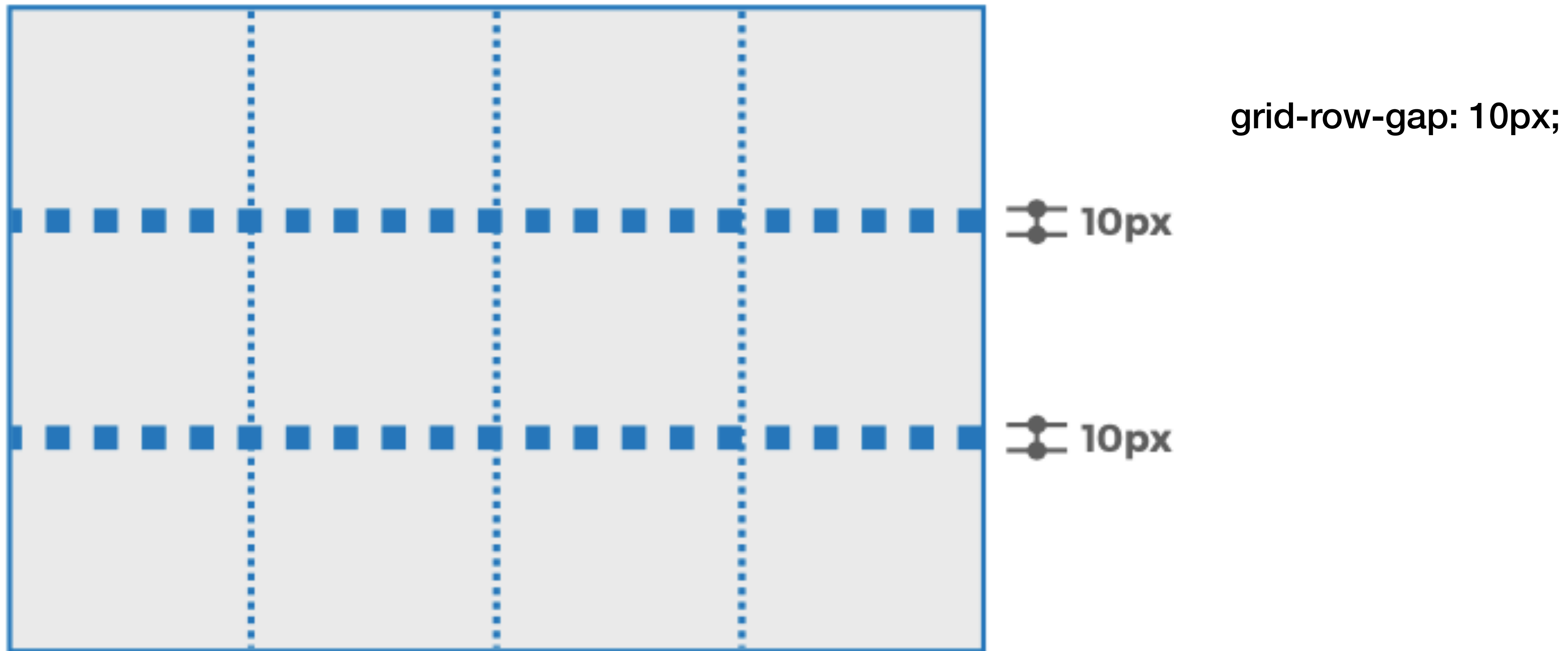
Défini l'espacement entre les colonnes de la grille



`grid-column-gap: 10px;`

grid-row-gap

Défini l'espacement entre les lignes de la grille



grid-gap

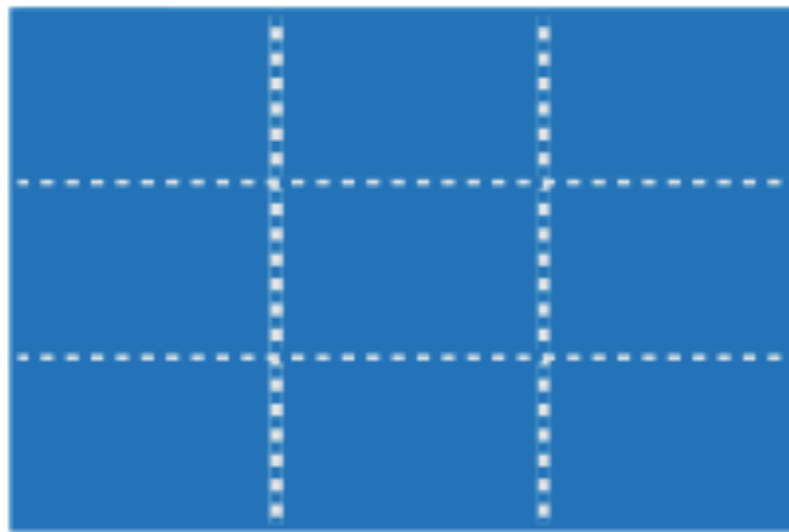
Propriété raccourcie de grid-column-gap et grid-row-gap

```
grid-gap: <grid-column-gap> <grid-row-gap>;
```

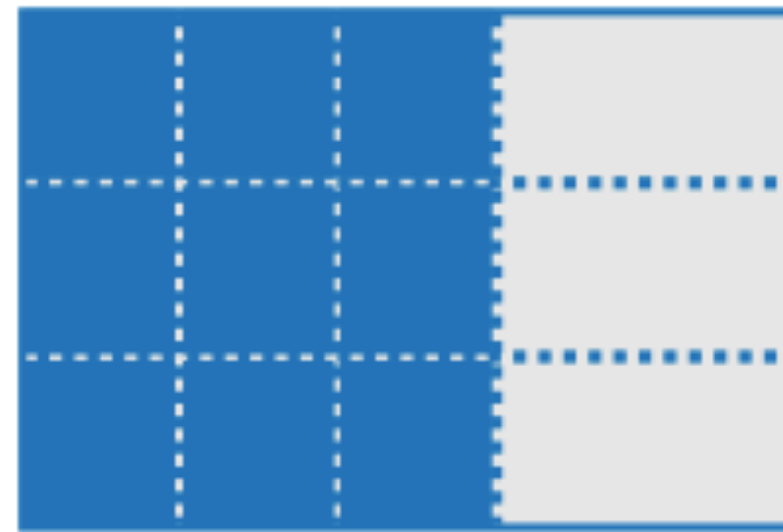
justify-content

Alignement des colonnes de la grille dans le conteneur

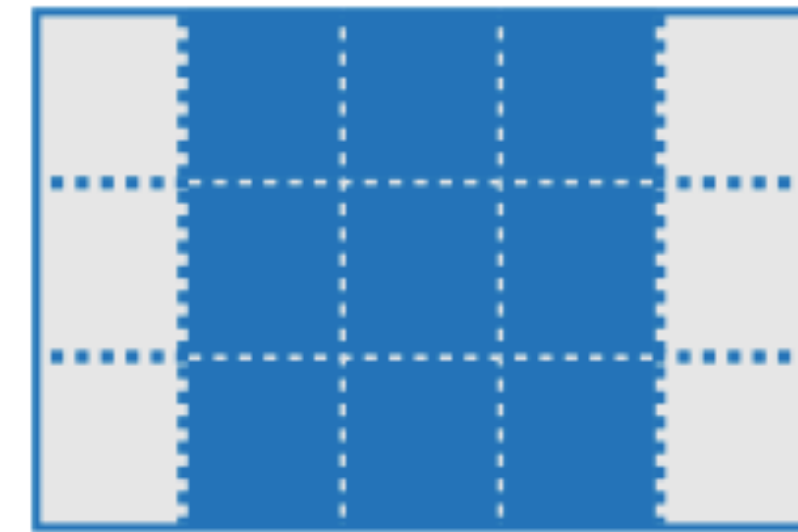
stretch



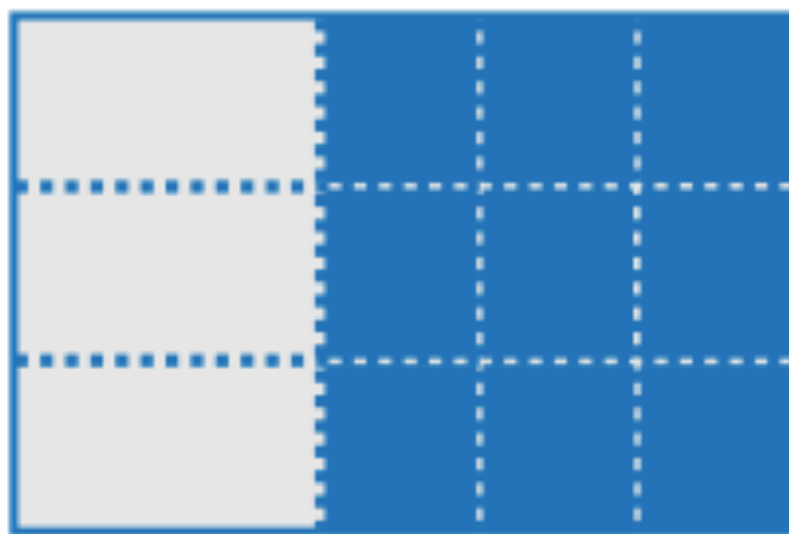
start



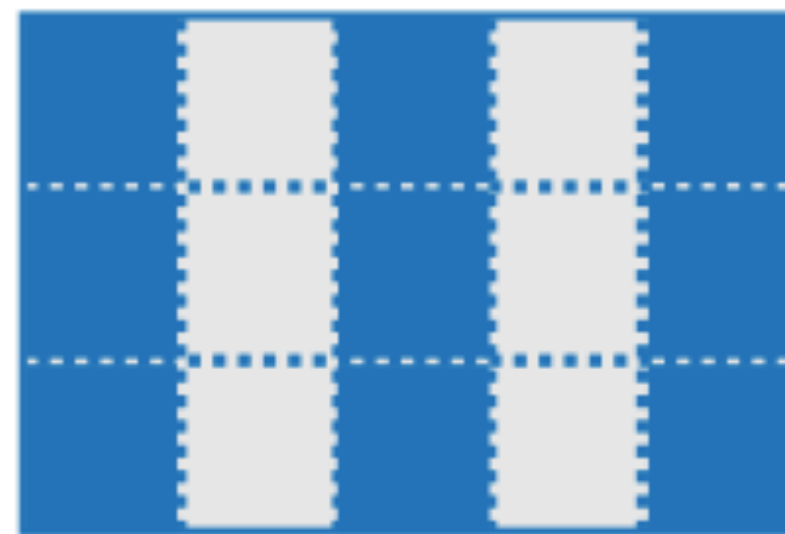
center



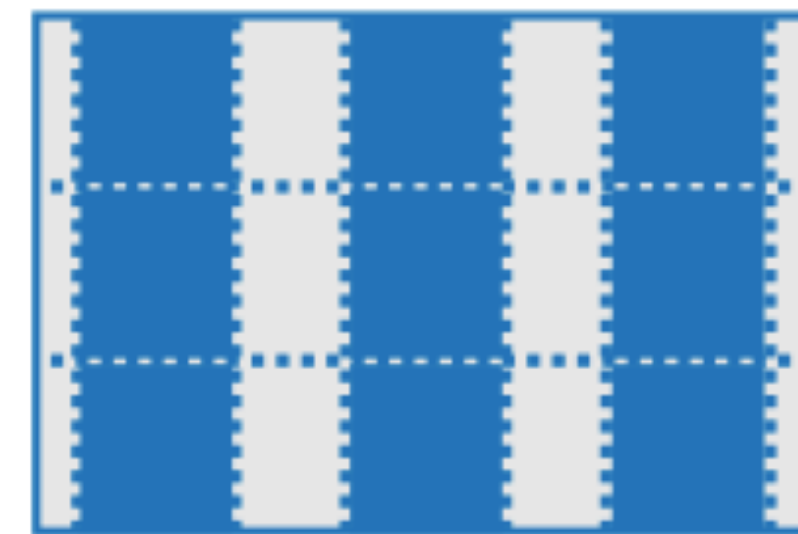
end



space-between



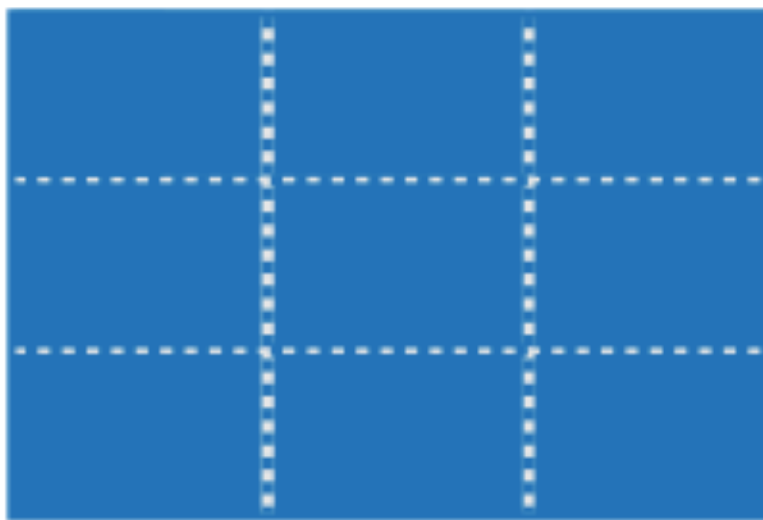
space-around/
space-evenly



align-content

Alignement des lignes de la grille dans le conteneur

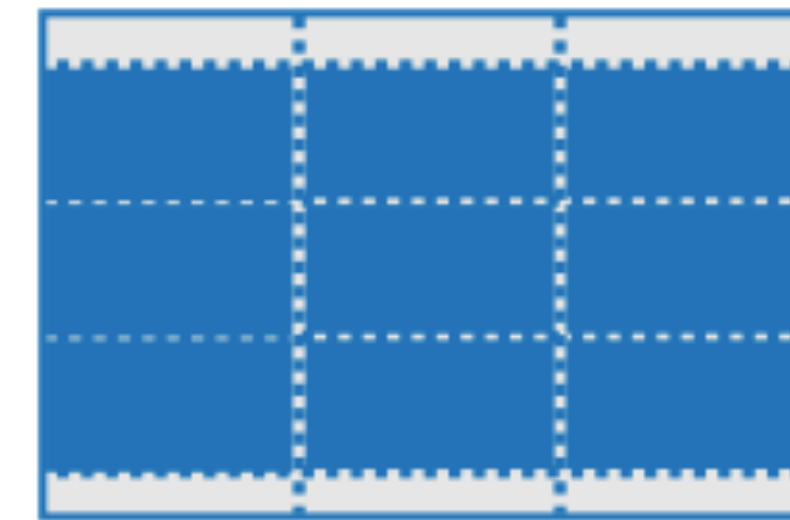
stretch



start



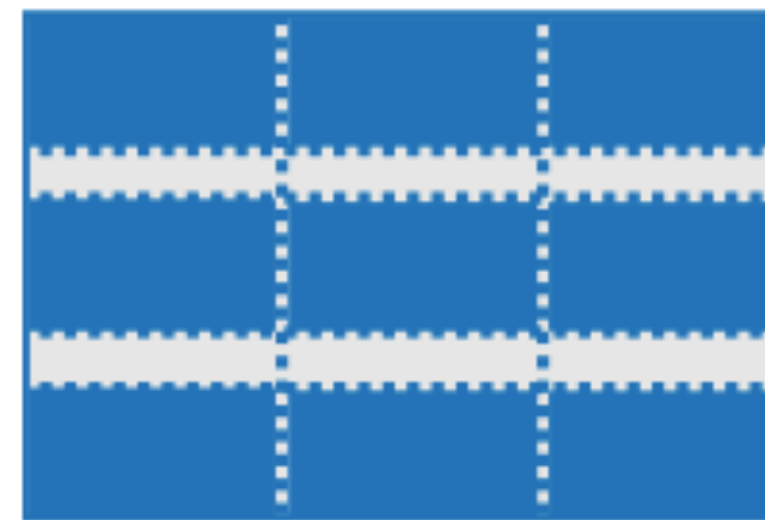
center



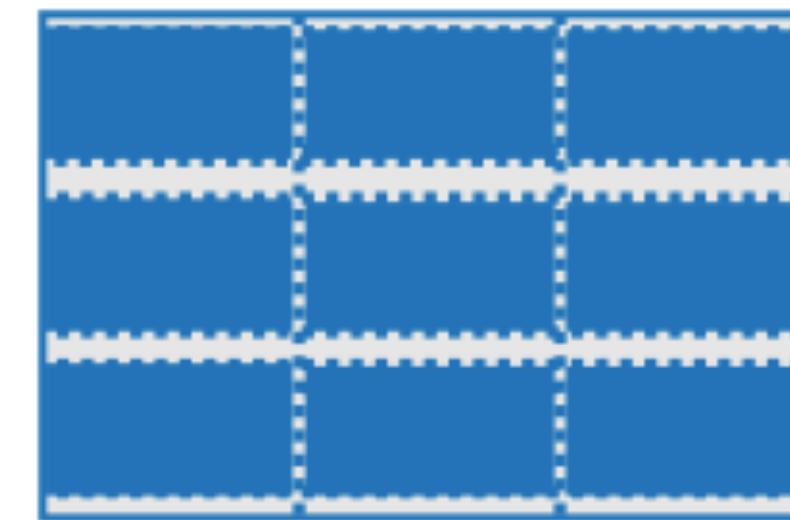
end



space-between



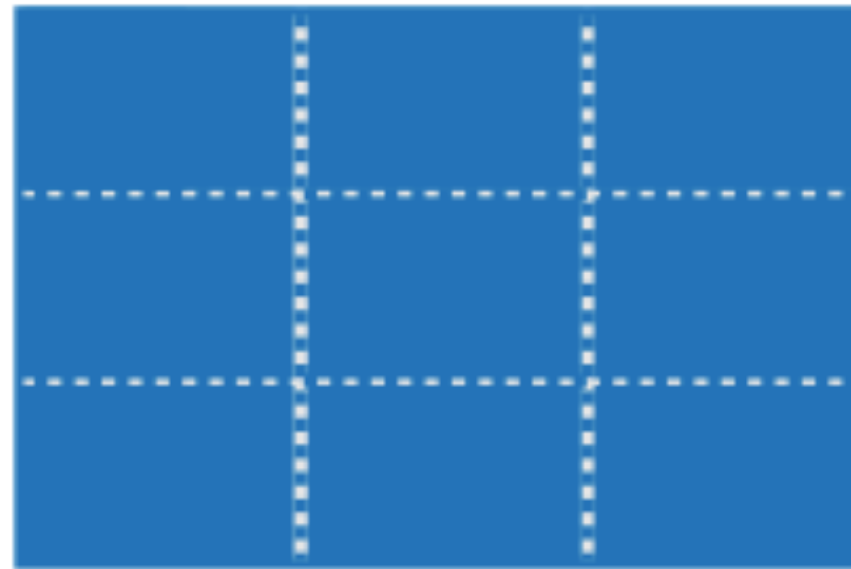
space-around/
space-evenly



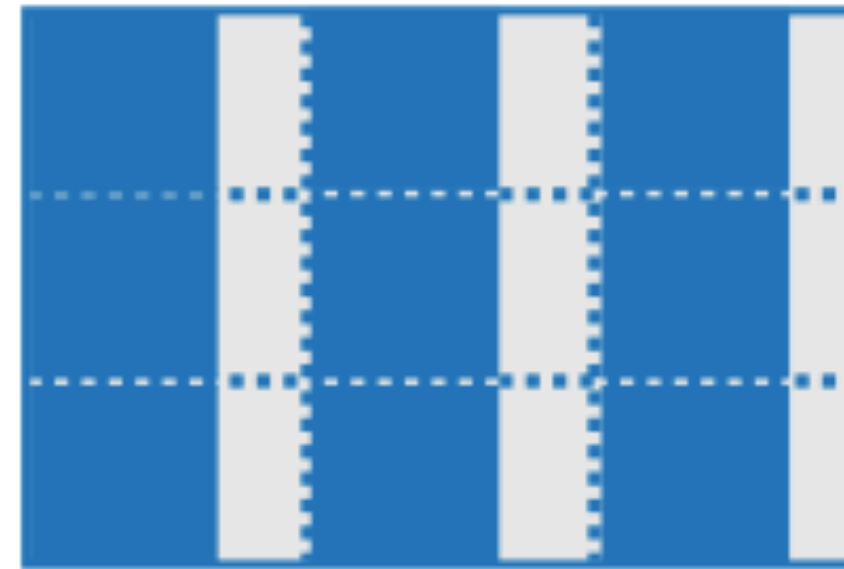
justify-items

Alignement des contenus sur les colonnes de la grille

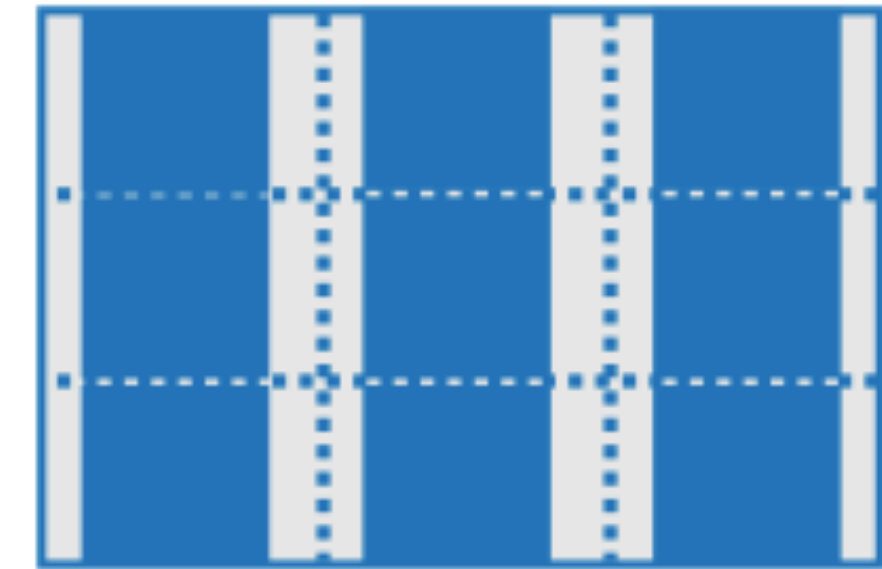
stretch*



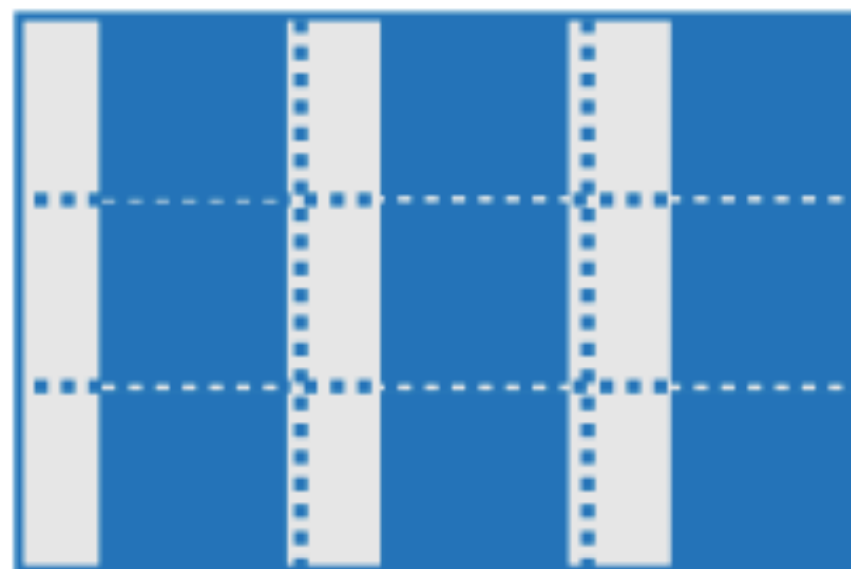
start



center



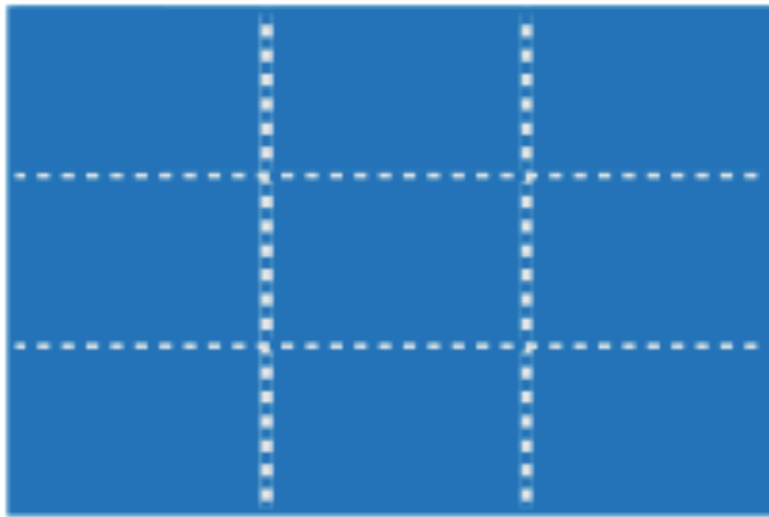
end



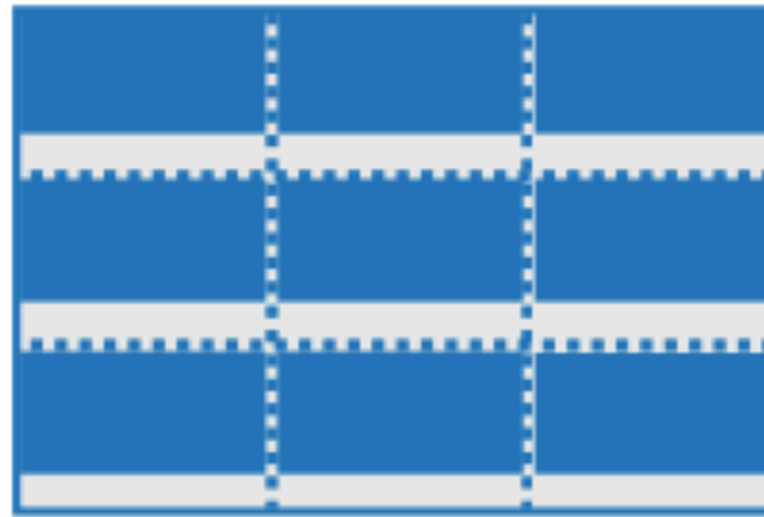
align-items

Alignement des contenus sur les lignes de la grille

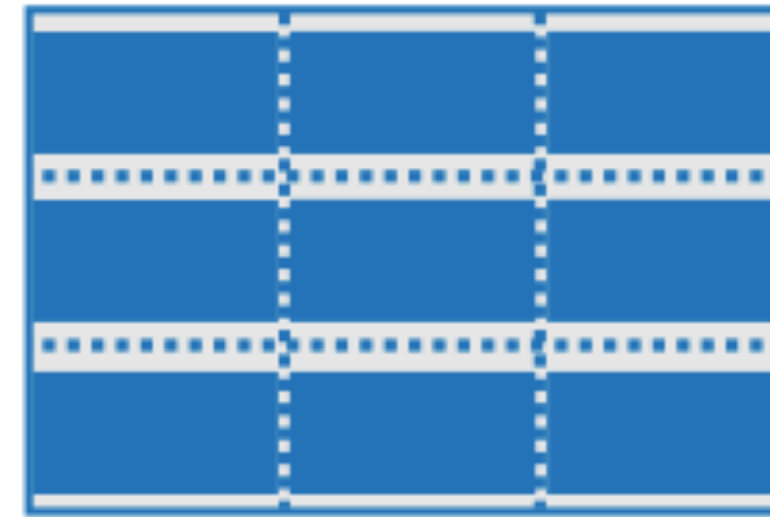
stretch*



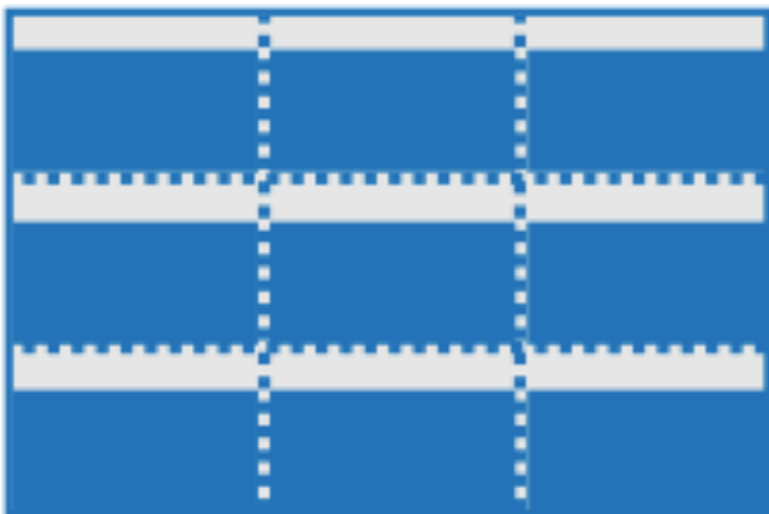
start



center



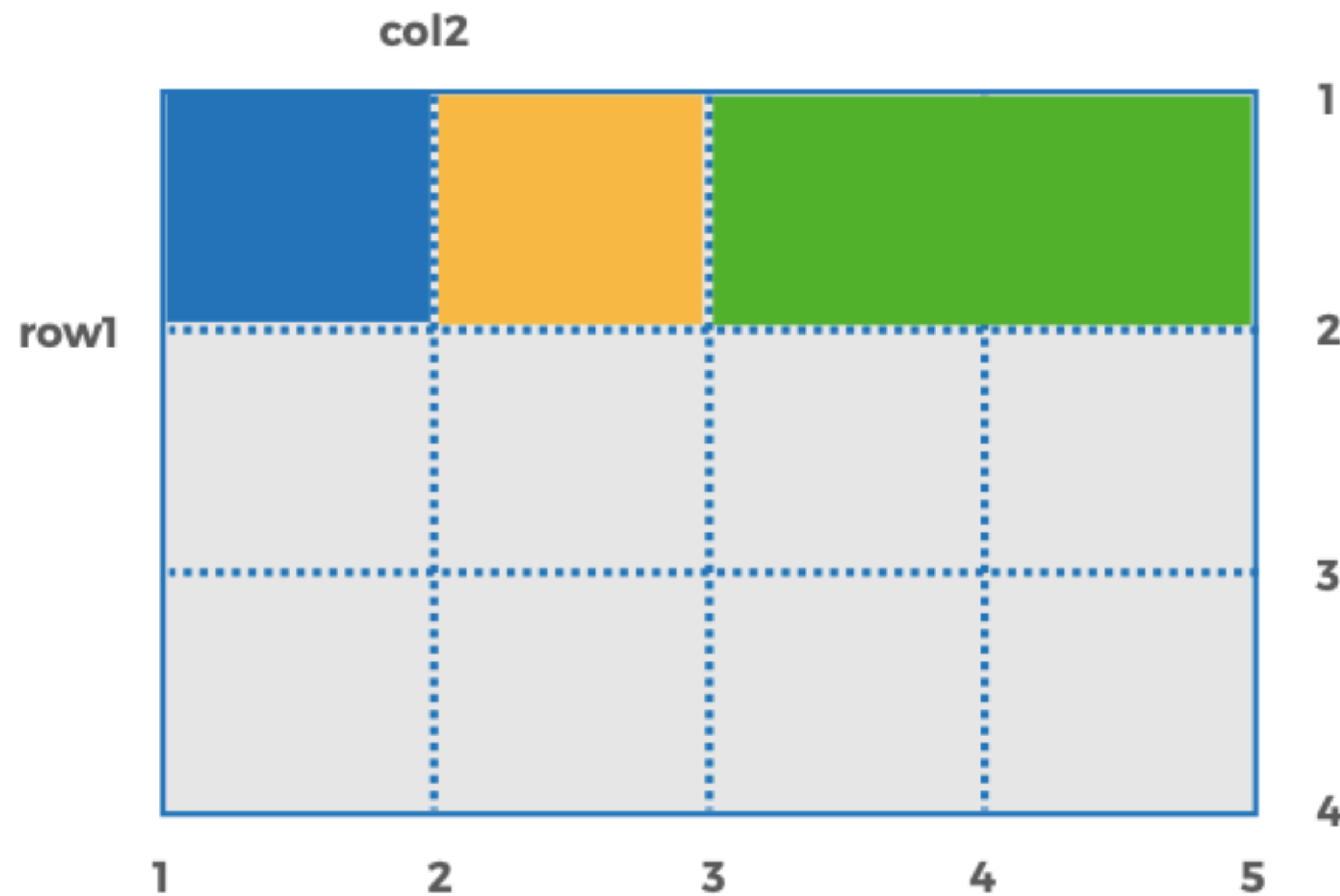
end



Propriétés des éléments de la grille

grid-column-start

Départ de l'item sur l'axe des colonnes



`grid-column-start: 1;`

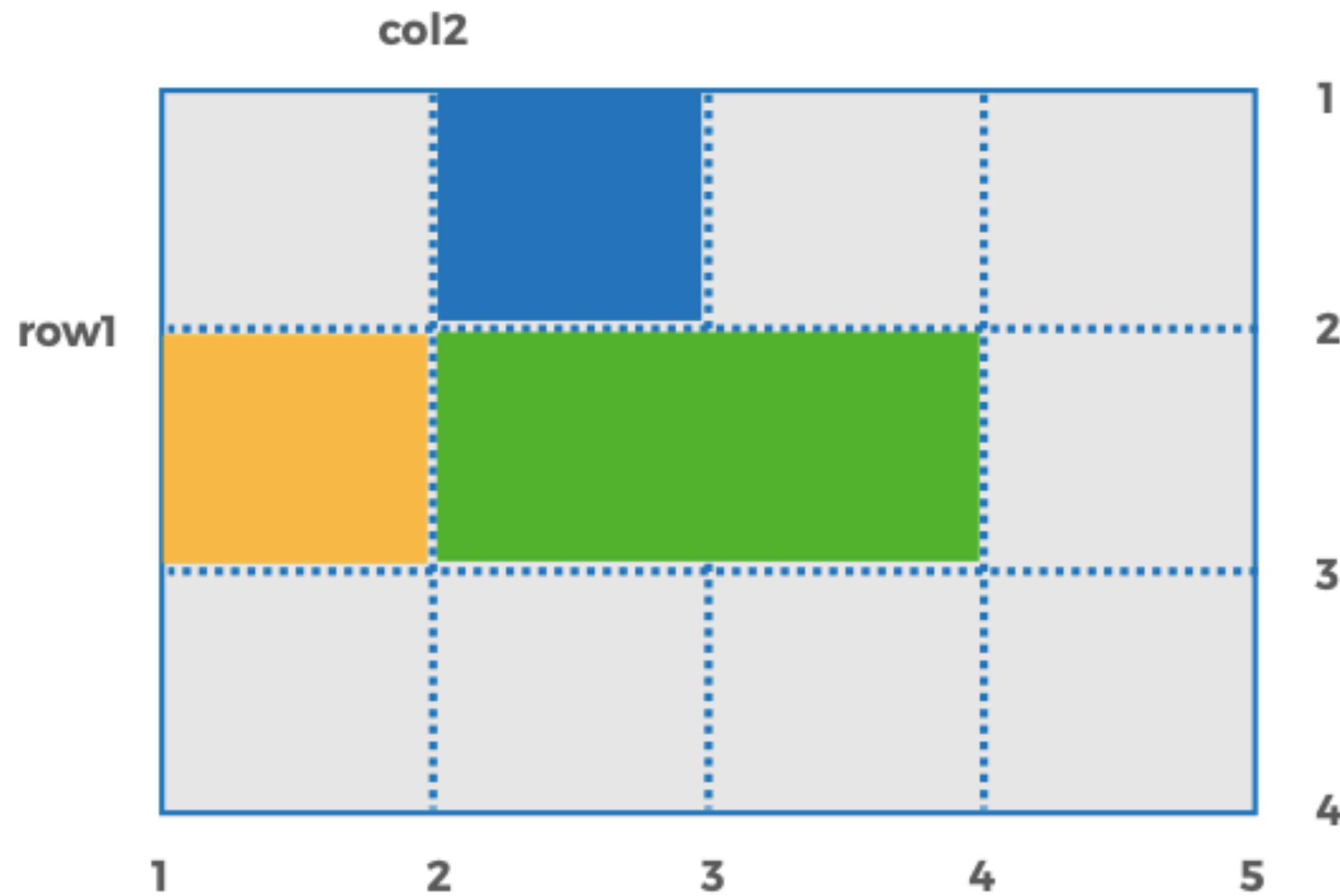
`grid-column-start: col2;`

`grid-column-start: span 2;`

| s'étend de ...

grid-column-end

Arrivée de l'item sur l'axe des colonnes



grid-column-end: 3;

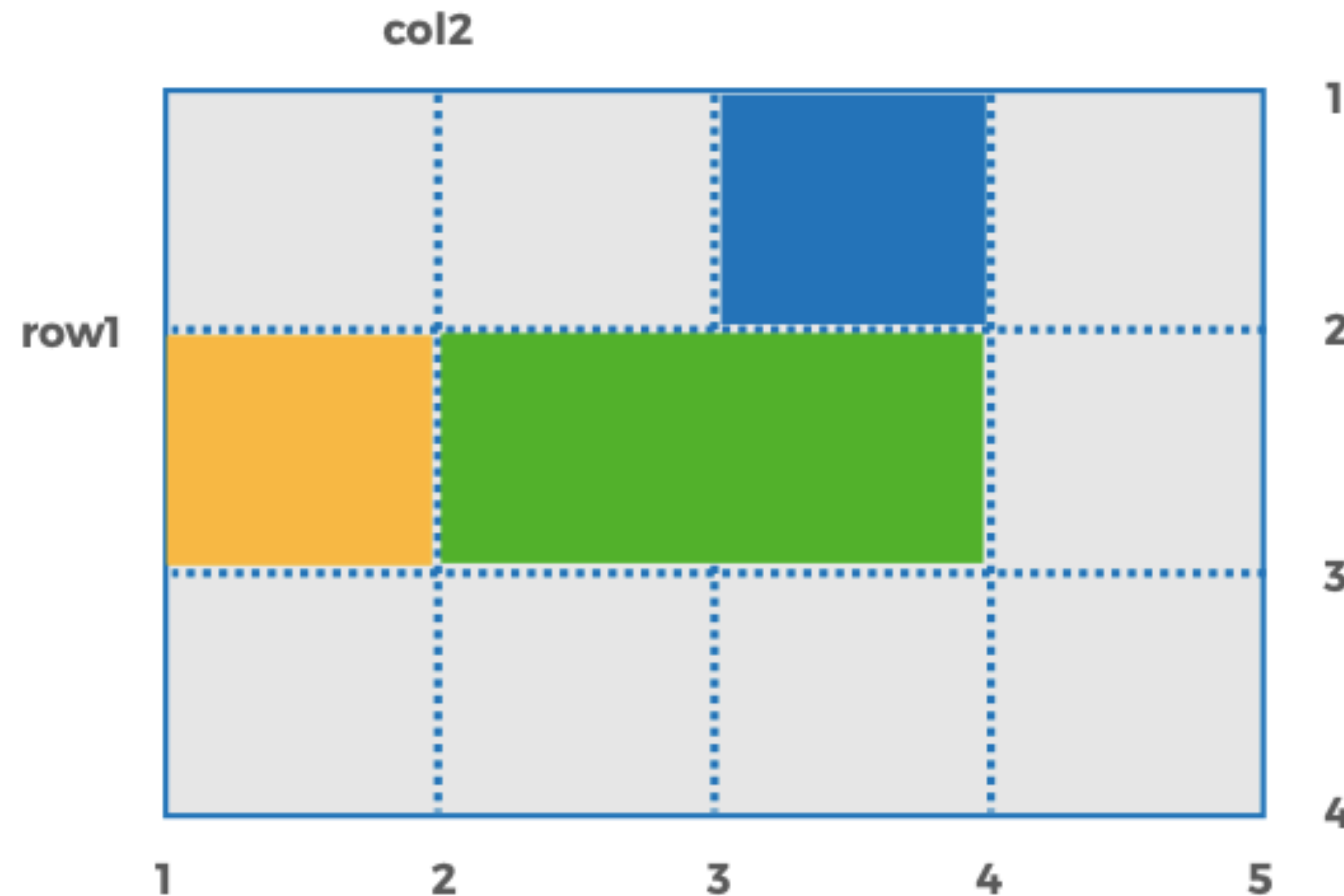
grid-column-end: col2;

grid-column-end: span 2;

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

Grid-column

Taille et emplacement de l'item sur l'axe des colonnes



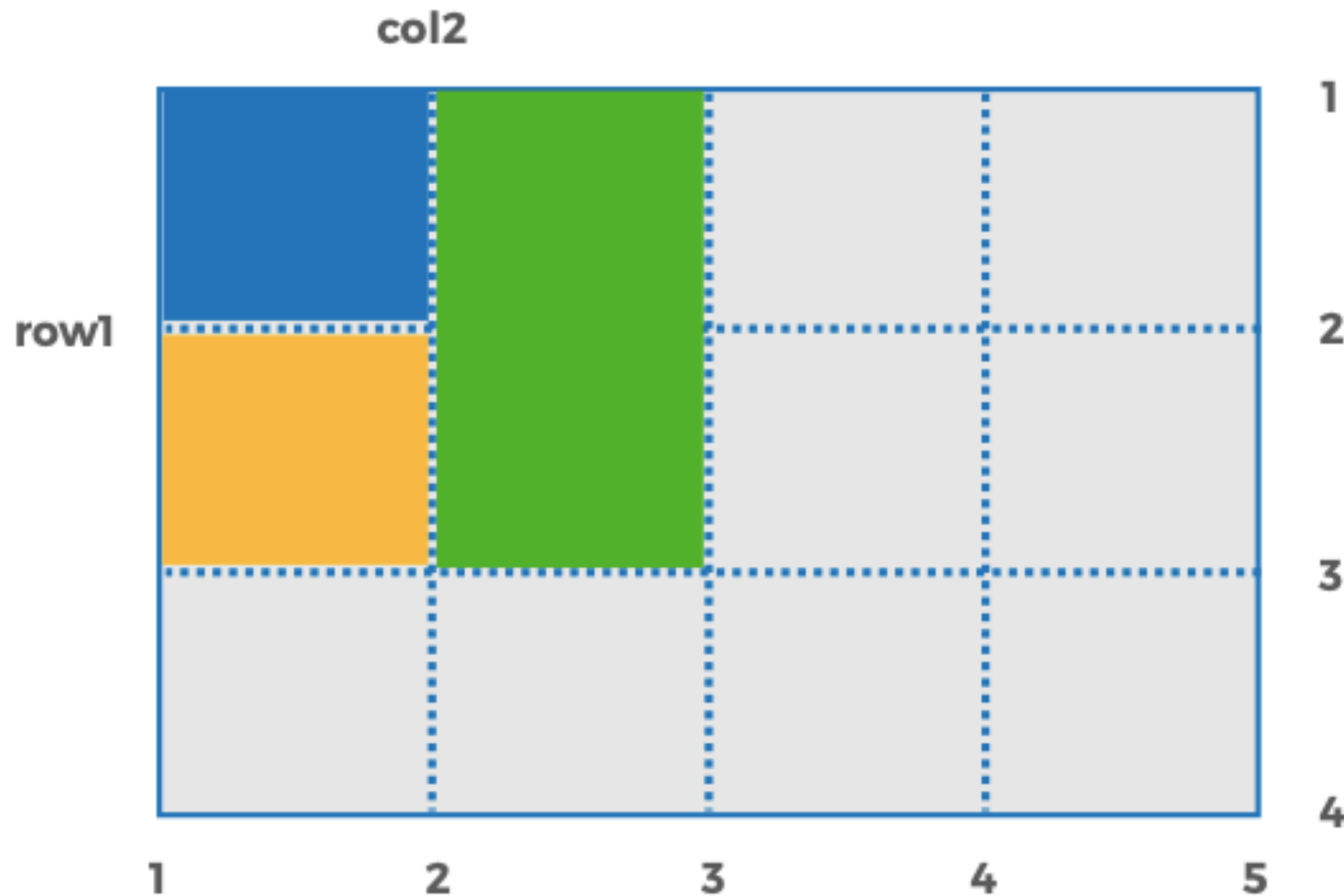
grid-column: 3;

grid-column: auto / col2;

grid-column: 2 / span 2;

grid-row-start

Départ de l'item sur l'axe des lignes



`grid-row-start: 1;`

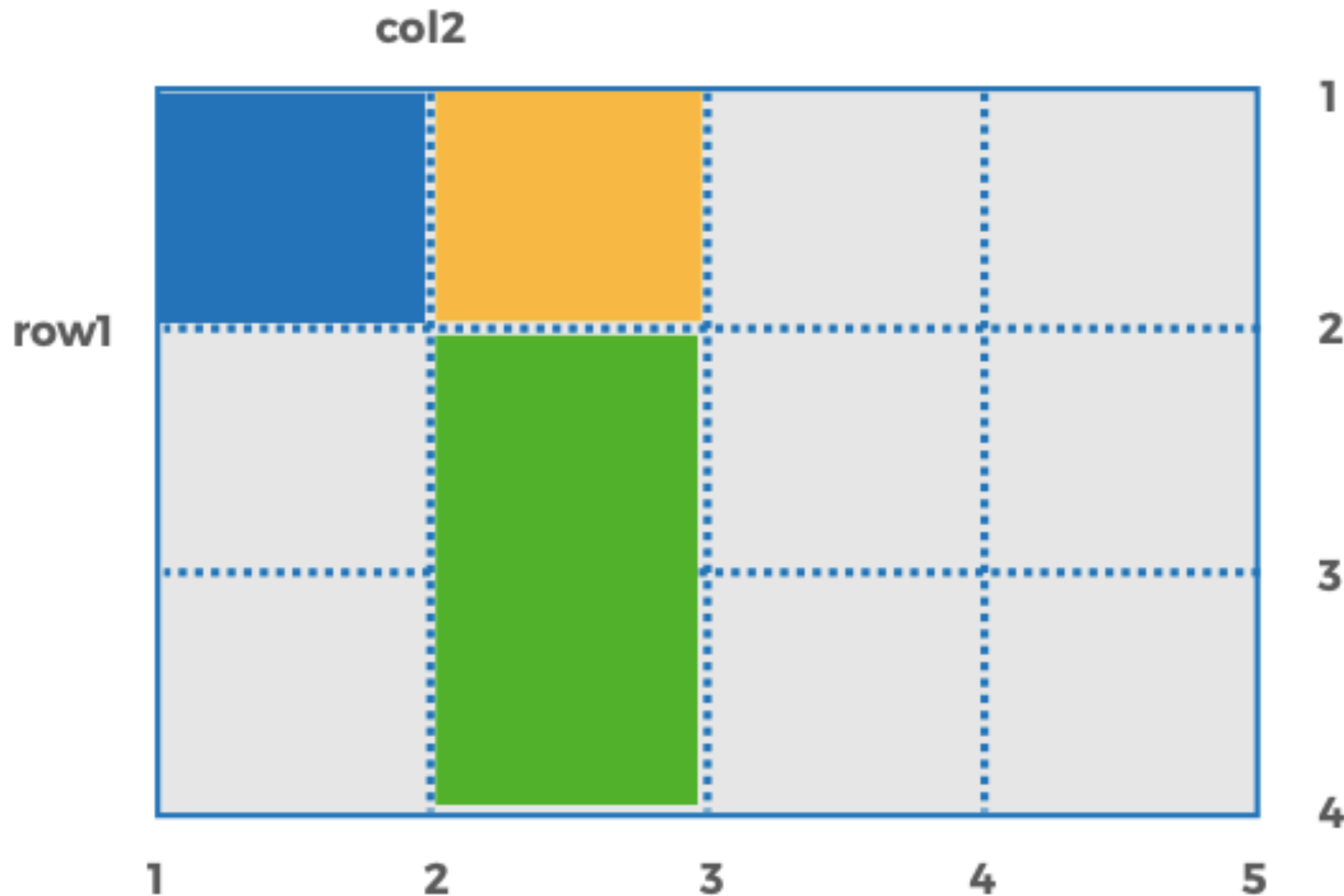
`grid-row-start: row1;`

`grid-row-start: span 2;`

s'étend de ...

grid-row-end

Arrivée de l'item sur l'axe des lignes



grid-row: 1;

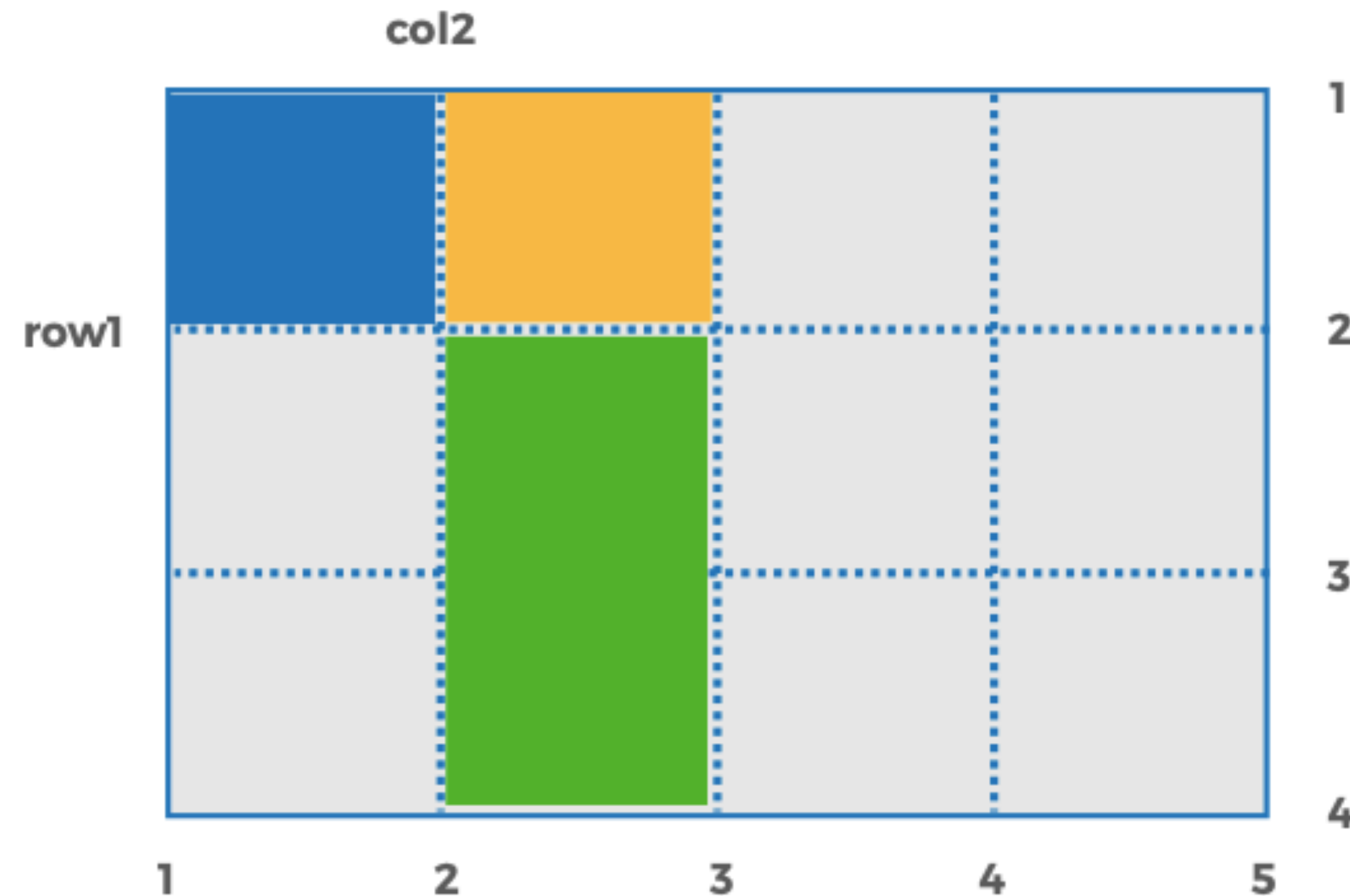
grid-row: auto / row1;

grid-row: 2 / span 2;

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

grid-row

Taille et emplacement de l'item sur l'axe des lignes



grid-row: 1;

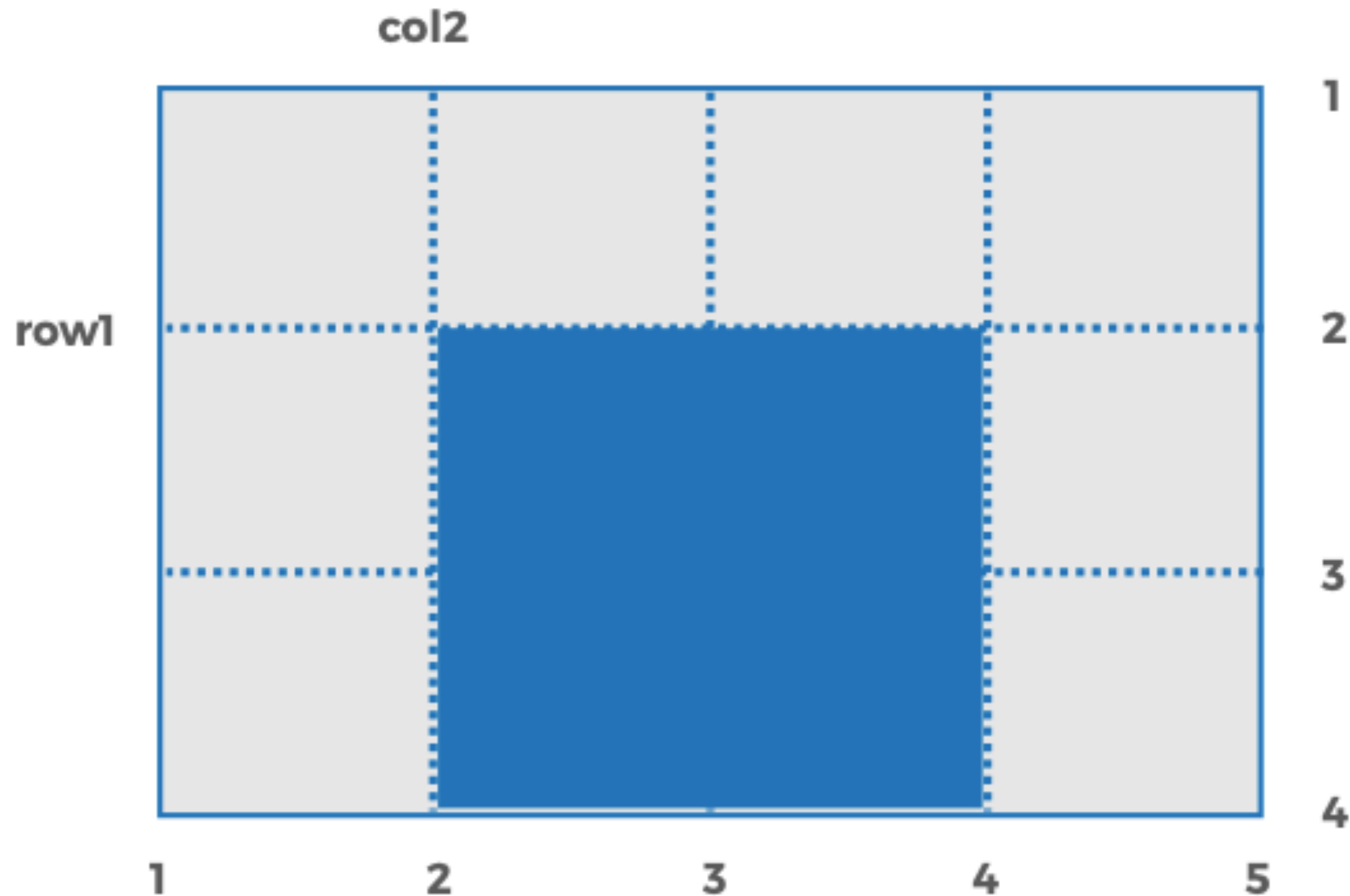
grid-row: auto / row1;

grid-row: 2 / span 2;

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

grid-area

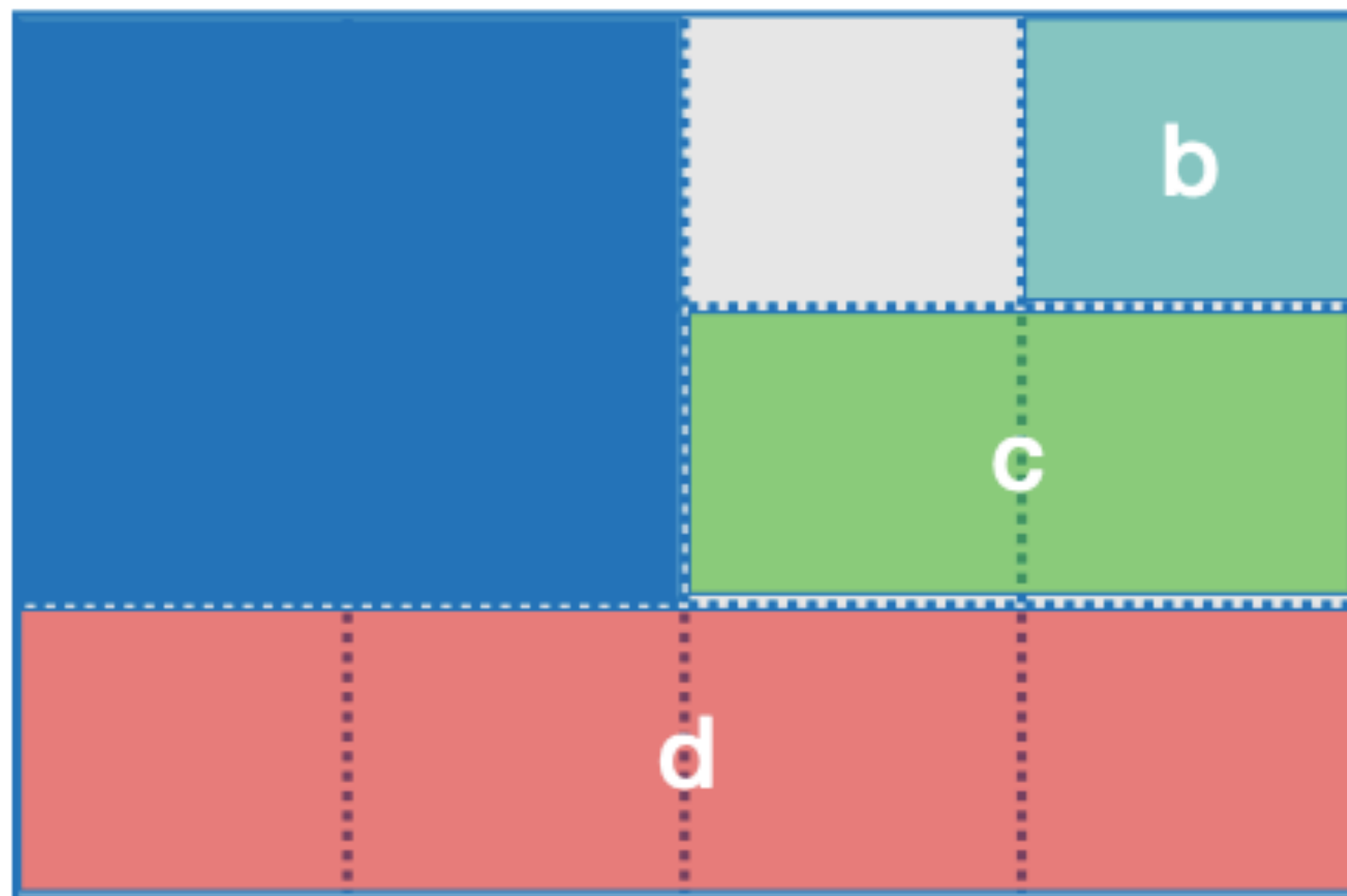
propriété raccourcie de :



grid-area: row1 / col2 / 4 / span 2;

grid-area

Example

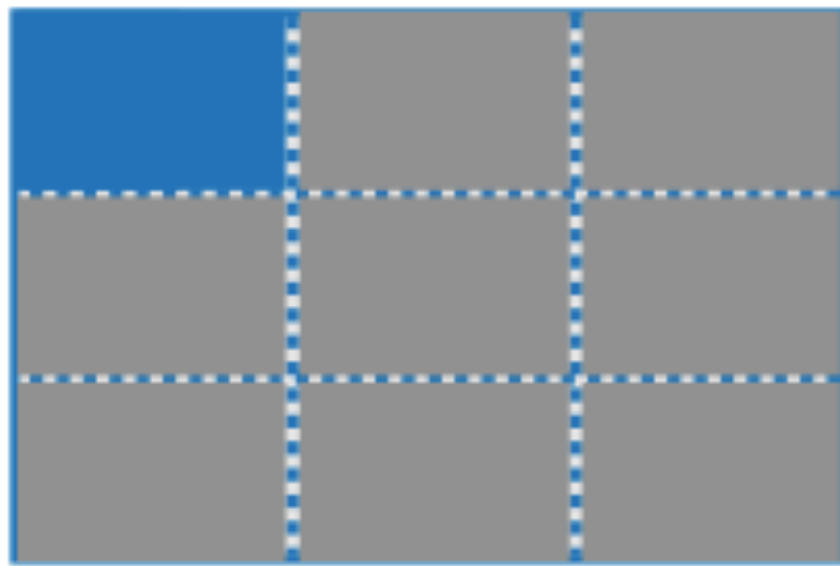


```
.container {  
  grid-template: "a a . b" 1fr  
                 "a a c c" 1fr  
                 "d d d d" 1fr  
                 / repeat(4, 1fr);  
}  
  
.element {  
  grid-area: a;  
}
```

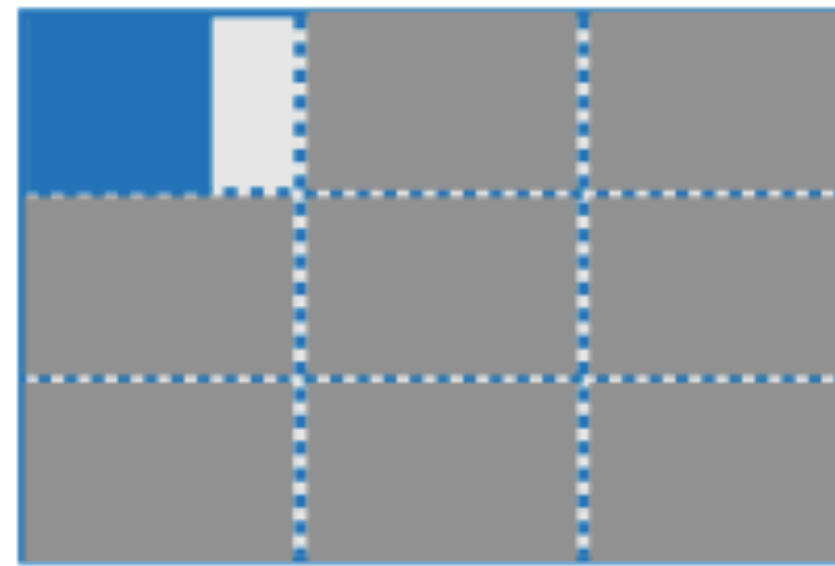
justify-self

Alignement de l'item sur les colonnes de la grille

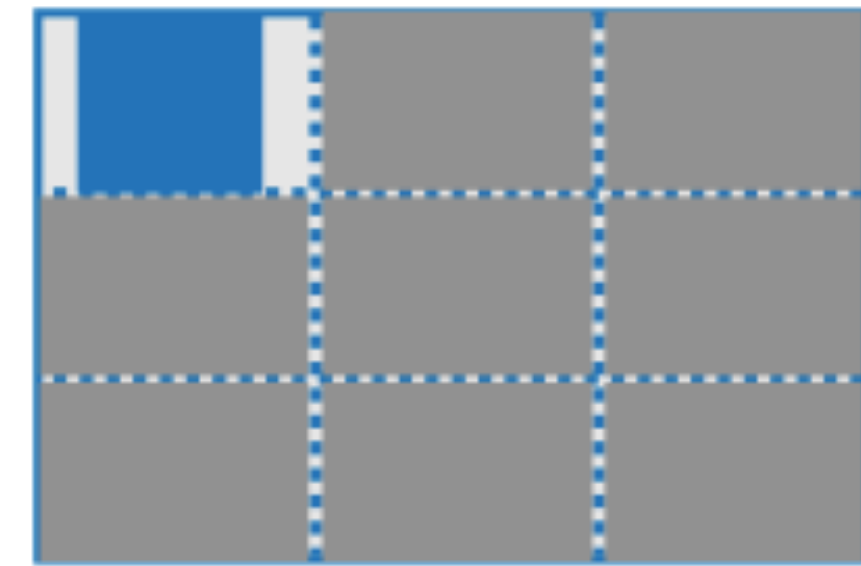
stretch*



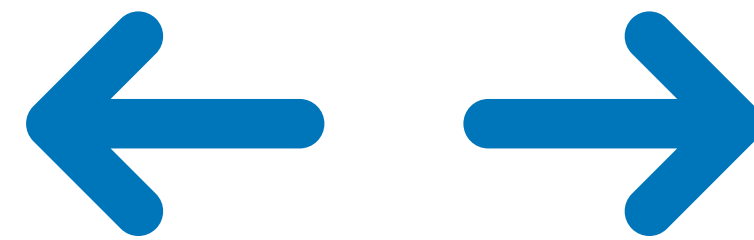
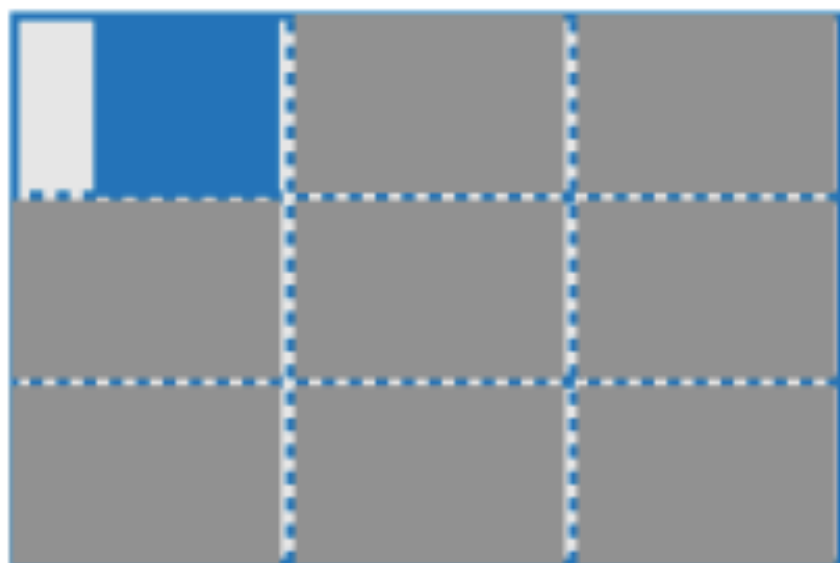
start



center



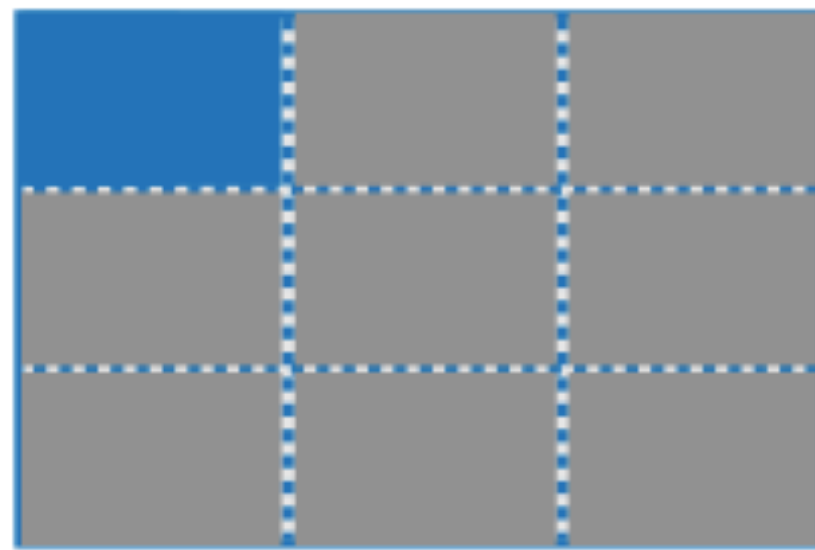
end



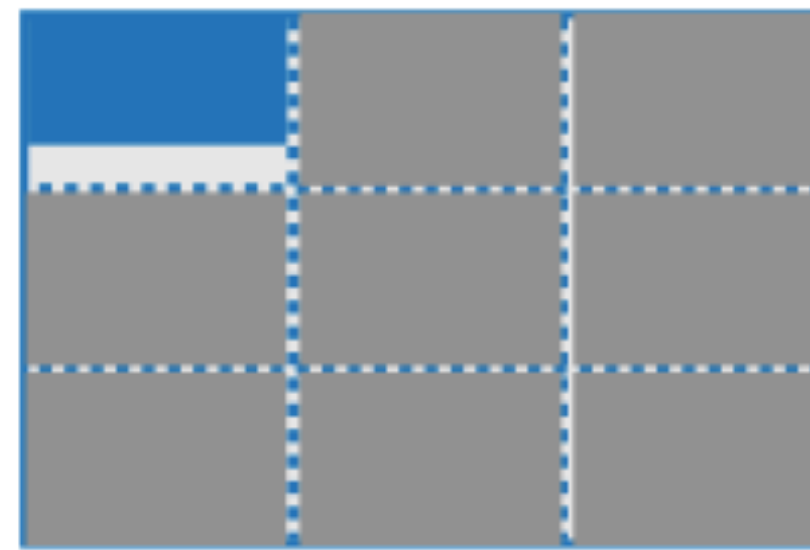
align-self

Alignement de l'item sur les lignes de la grille

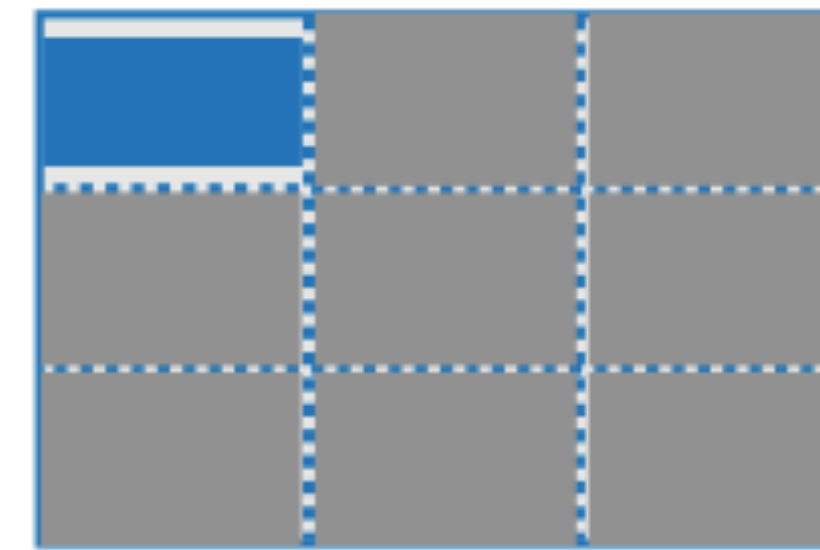
stretch*



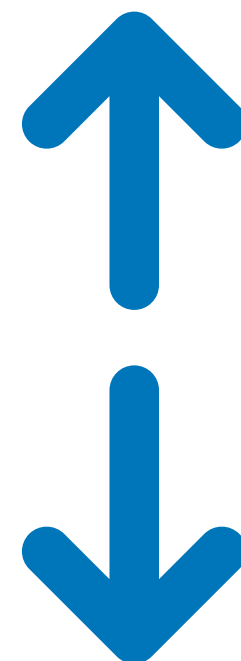
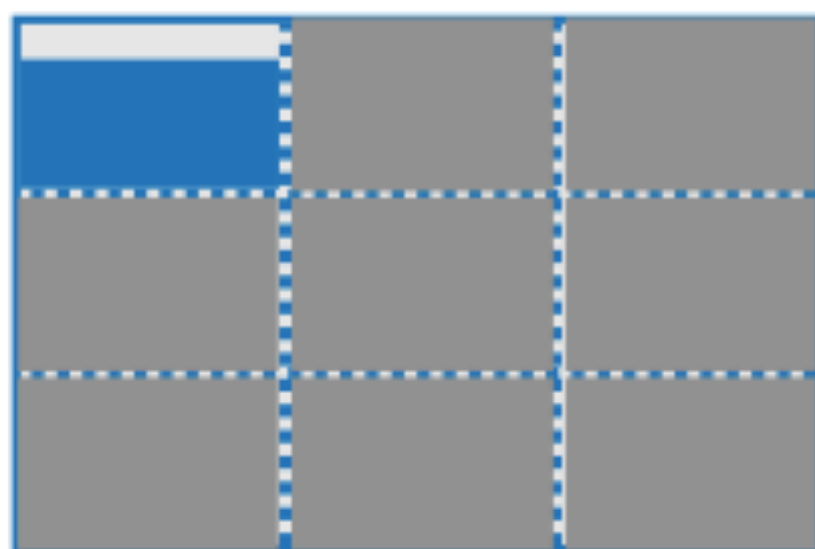
start



center

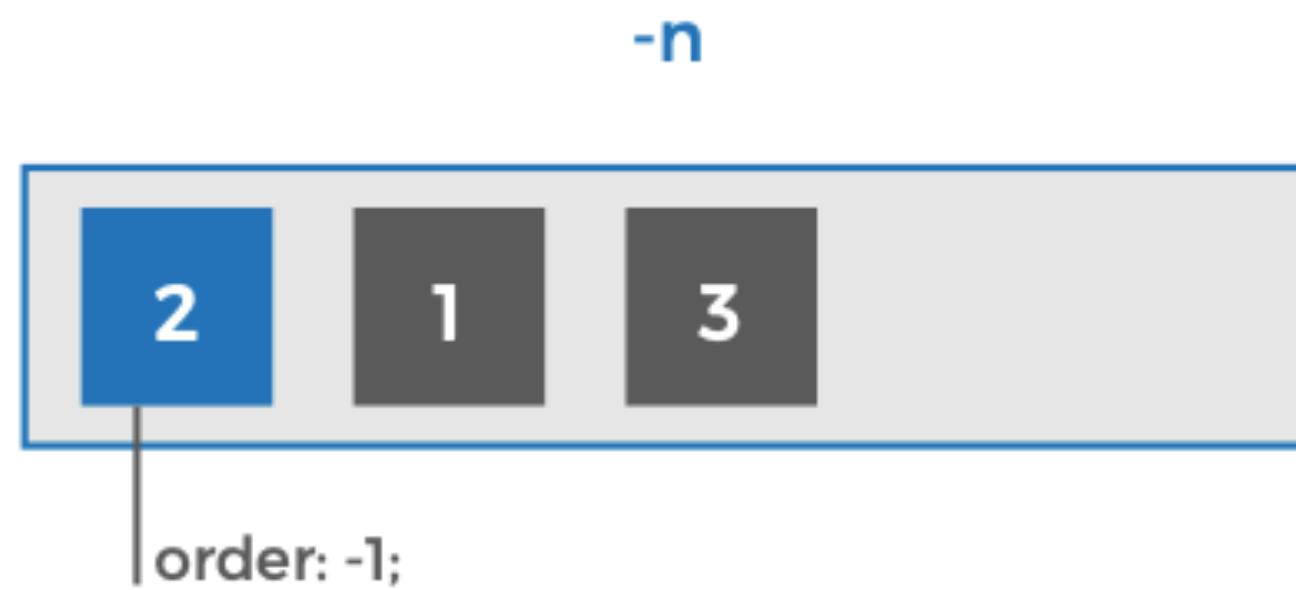
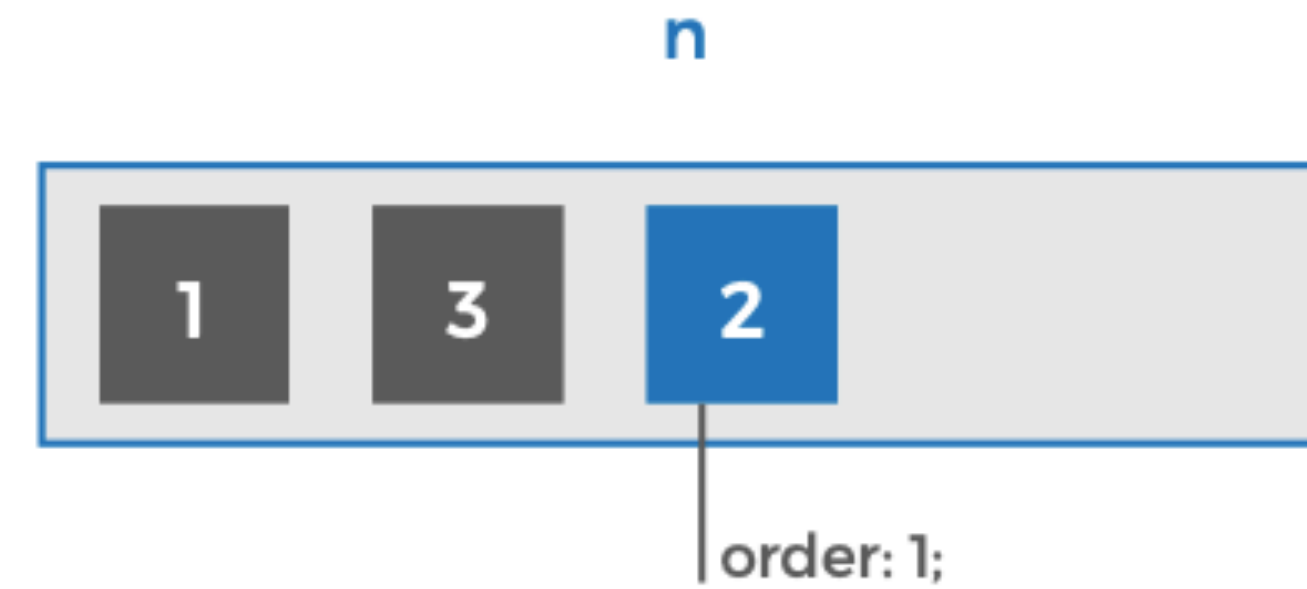
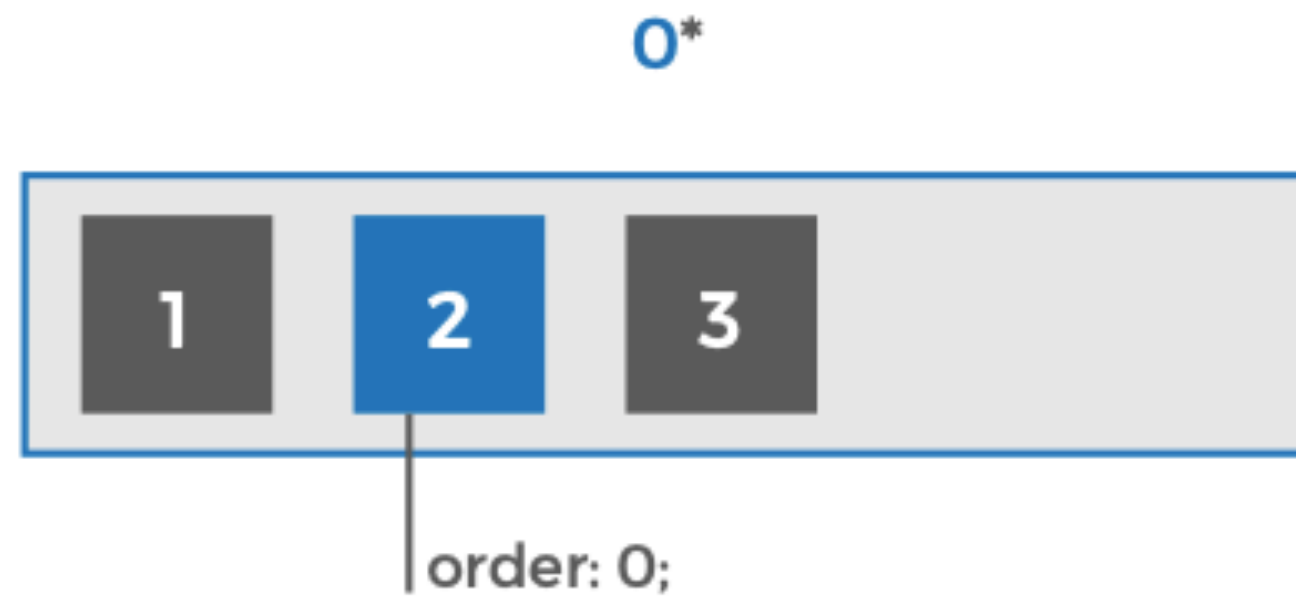


end



order

Défini l'ordre d'apparition des éléments (nombre entier)



Attention à l'accessibilité des éléments

* valeur par défaut

Autres

z-index

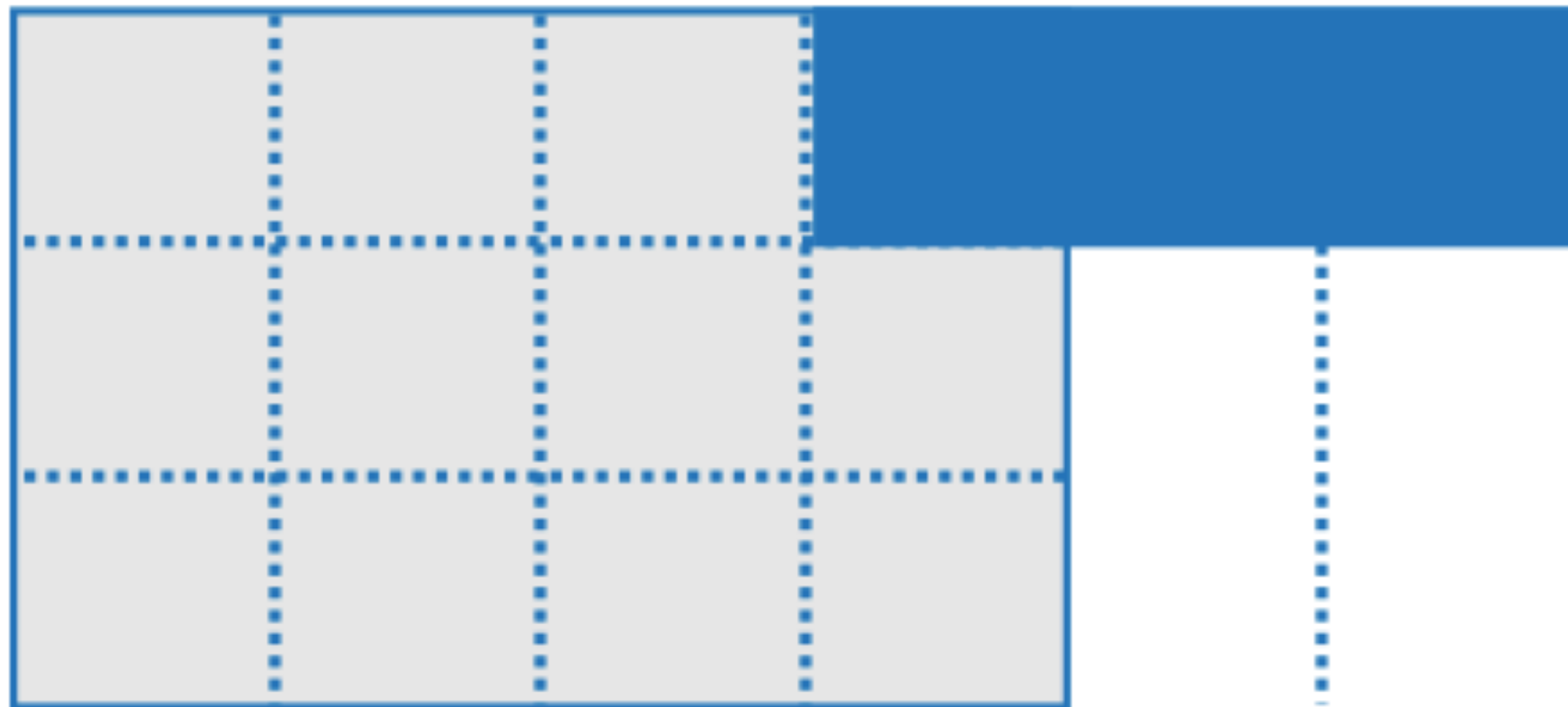
Les grid items peuvent se chevaucher.



```
.container {  
  grid-template: repeat(3, 1fr) / repeat(4, 1fr);  
}  
  
.blue {  
  grid-area: 1 / 1 / span 2 / span 3;  
  z-index: 1;  
}  
  
.green {  
  grid-area: 2 / 2 / span 2 / span 3;  
}
```


grid-auto-columns & grid-auto-rows

Taille des colonnes (ou lignes) implicites



```
.container {  
  display: grid;  
  grid-template: repeat(3,100px) /  
  repeat(4, 100px);  
  grid-auto-columns: 100px;  
}  
.element {  
  grid-column: 3 / 6;  
}
```

grid-auto-flow

Placement automatique les items non placés de façon explicites

row*

1	2	3

row dense

2	3	4
1		

column

	2	
	3	
1	4	


column dense

2	4	
3		
1		

* valeur par défaut

grid

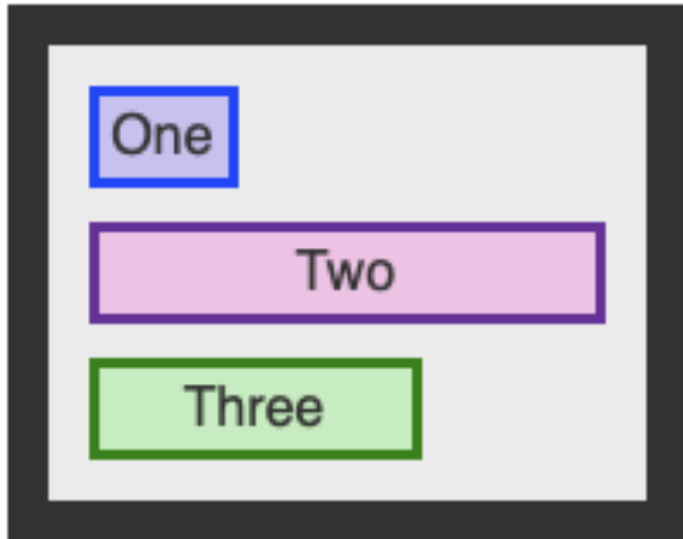
Une propriété pour les résumés toutes

 CSS Demo: grid Reset

```
grid: auto-flow / 1fr 1fr 1fr;
```

```
grid: auto-flow dense / 40px 40px 1fr;
```

```
grid: repeat(3, 80px) / auto-flow;
```



<https://developer.mozilla.org/en-US/docs/Web/CSS/grid>

Que choisir ?

Le composant
n'utilise-t-il qu'une
direction ?

Le composant
s'inscrit-il dans
une grille ?

Le composant
contient-il des
éléments flexibles ?

