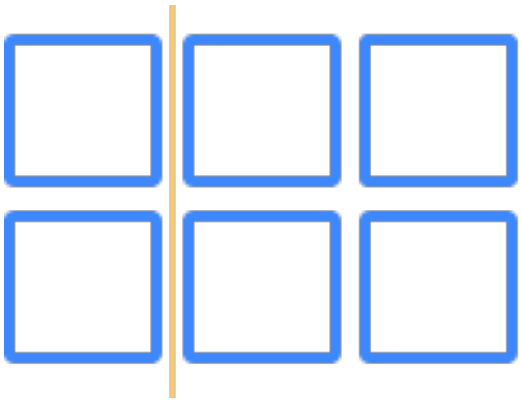
CSS Grid

La terminologie des grilles

Grid container : Le conteneur contenant l'ensemble de la grille CSS.

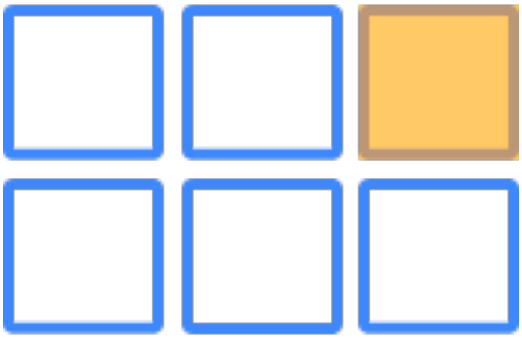
Il s'agira de l'élément qui possède la propriété :
display : grid ou **display : inline-grid**.

Grid Line



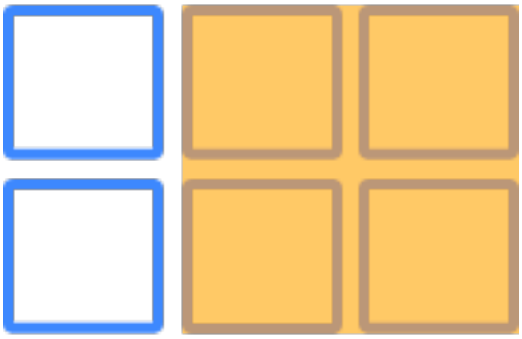
Des lignes verticales et horizontales divisent la grille
en colonnes et rangées.

Grid cell



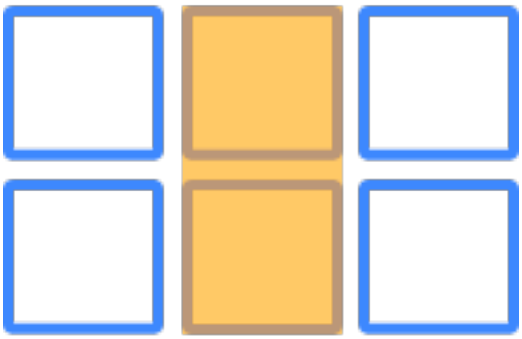
1 seule cellule de la grille

Grid area



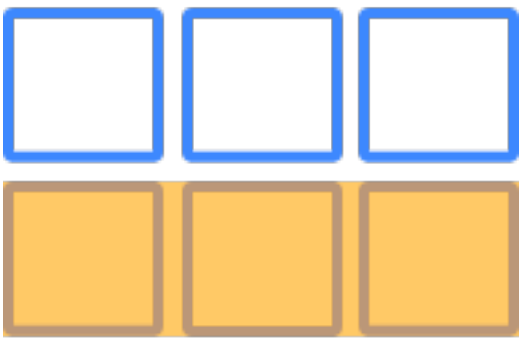
Espace rectangulaire entouré de quatre lignes de grille.
Une zone de grille peut contenir un nombre quelconque de cellules de grille.

Grid track



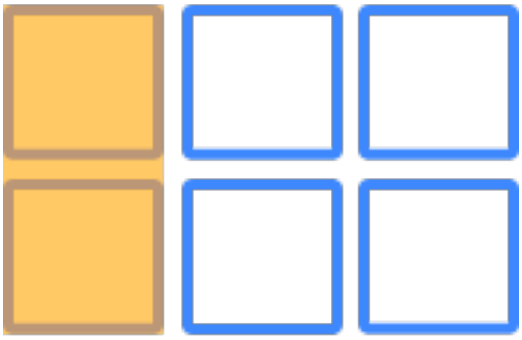
L'espace entre deux lignes de la grille.
Cet espace peut être horizontal ou vertical.

Grid row



Piste (track) horizontale d'une grille.

Grid column



Piste (track) verticale

Firefox Nightly
https://www.mozilla.org/en-US/firefox/channel/desktop/#nightly

Première grille

Créer une grille

La première chose à faire est de créer un conteneur de grille.
Pour ce faire, il suffit de déclarer display : grid sur l'élément conteneur.
Seuls les enfants directs seront des éléments de cette grille.

Définition des lignes et des colonnes

Il existe plusieurs façons de définir les lignes et les colonnes.

Pour notre première grille, nous utiliserons les propriétés grid-template-columns et grid-template-rows.

Ces propriétés nous permettent de définir la taille des lignes et des colonnes de notre grille.

Pour créer deux lignes de 150px de hauteur fixe et trois colonnes de 150px de largeur fixe, il suffit d'écrire :

```
grid-template-columns: 150px 150px 150px;
grid-template-rows: 150px 150px;
```

Et pour ajouter une quatrième colonne de 70px il suffit de :

```
grid-template-columns: 150px 150px 150px 70px;
```

Pour ajouter une gouttière (gap)

```
grid-gap: 1rem;
```

Cette simple ligne vous donne une gouttière de taille égale entre toutes les lignes et les colonnes.

Pour définir la taille de la gouttière pour les colonnes et les lignes individuellement, vous pouvez utiliser les propriétés grid-column-gap et grid-row-gap.

https://codepen.io/web-god/pen/KKvmqmm

L’unité fr

Dans notre première grille, nous avons créé des colonnes avec une largeur fixe en px.

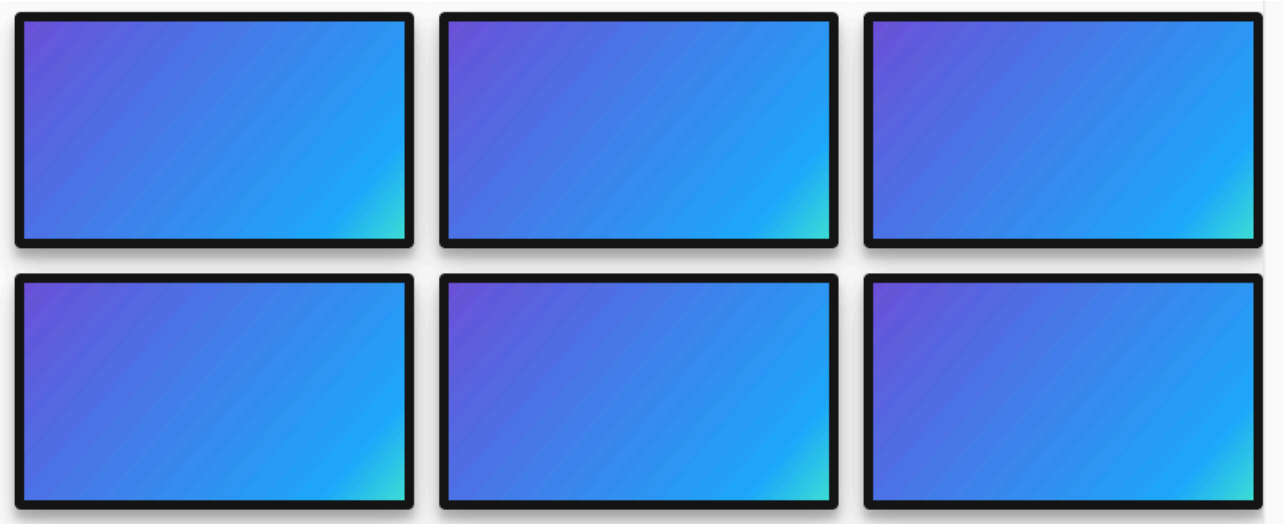
C'est très bien, mais ce n'est pas très flexible.

Heureusement, CSS Grid Layout introduit une nouvelle unité de longueur appelée fr, qui est l'abréviation de "fraction".

L'unité fr est une fraction de l'espace disponible dans la grille.

```
.container {
  display: grid;
  width: 800px;
  grid-template-columns: 1fr 1fr 1fr;
  grid-template-rows: 150px 150px;
  grid-gap: 1rem;
}
```

https://codepen.io/web-god/pen/WNEjjgb



Unités mixtes

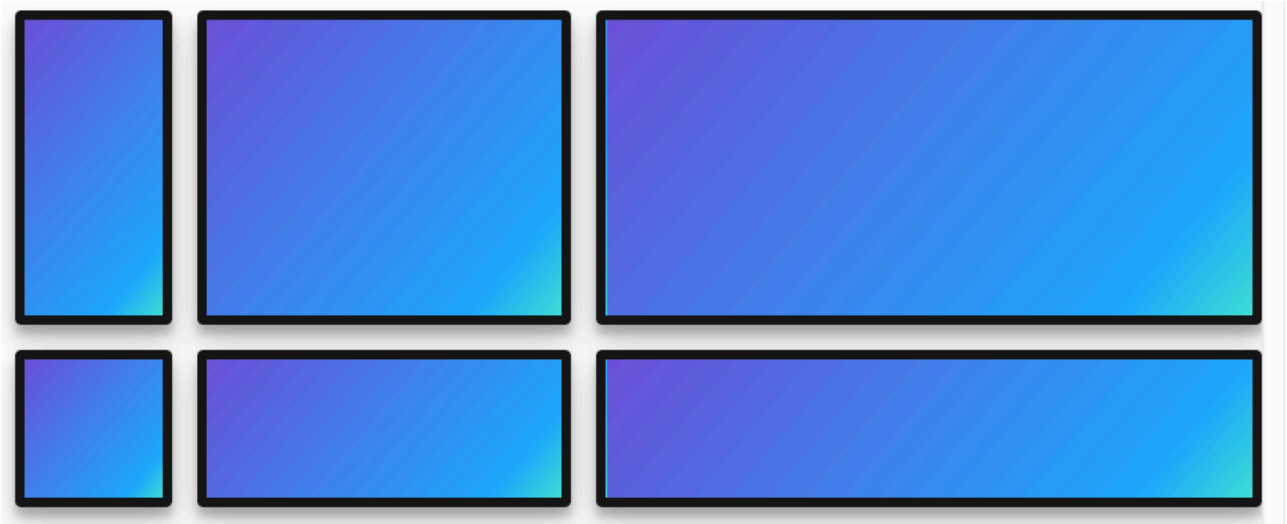
Lorsque vous déclarez les tailles de piste, vous pouvez utiliser des tailles fixes avec des unités telles que px et em. Vous pouvez également utiliser des tailles flexibles telles que les pourcentages ou l'unité fr. La véritable magie de CSS Grid Layout réside toutefois dans la possibilité de mélanger ces unités. La meilleure façon de comprendre est de prendre un exemple :

```
.container {
  width: 100%;
  display: grid;
  grid-template-columns: 100px 30% 1fr;
  grid-template-rows: 200px 100px;
  grid-gap: 1rem;
}
```

https://codepen.io/web-god/pen/PoKmjgb

Ici, nous avons déclaré une grille à trois colonnes.

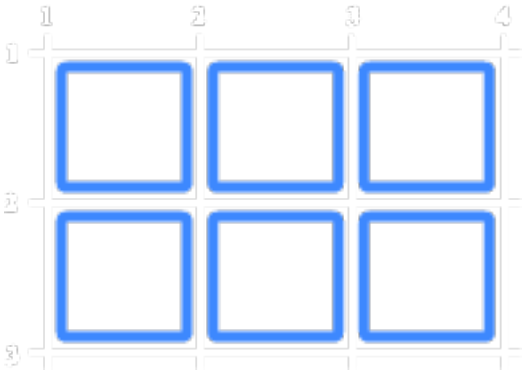
La première colonne a une largeur fixe de 100px.
La deuxième colonne occupera 30 % de l'espace disponible,
et la troisième colonne est 1fr, ce qui signifie qu'elle occupera une fraction de l'espace disponible. Dans ce cas, elle occupera tout l'espace restant (1/1).



Position des éléments

Comprendre les lignes de la grille

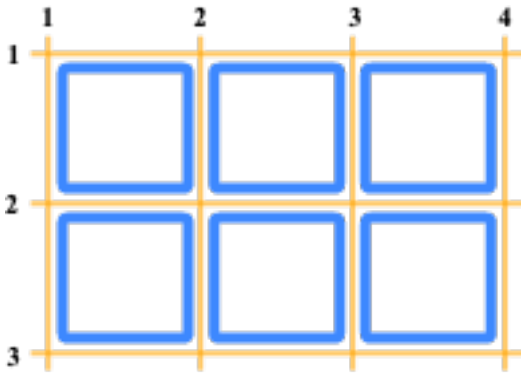
Il existe plusieurs façons de placer des éléments, mais nous allons commencer par un exemple de base. Considérons une grille simple de 3x2 :



Chaque élément de cette grille sera placé automatiquement dans l'ordre par défaut.

Si nous souhaitons avoir un plus grand contrôle, nous pouvons positionner les éléments sur la grille en utilisant les numéros des lignes de la grille.

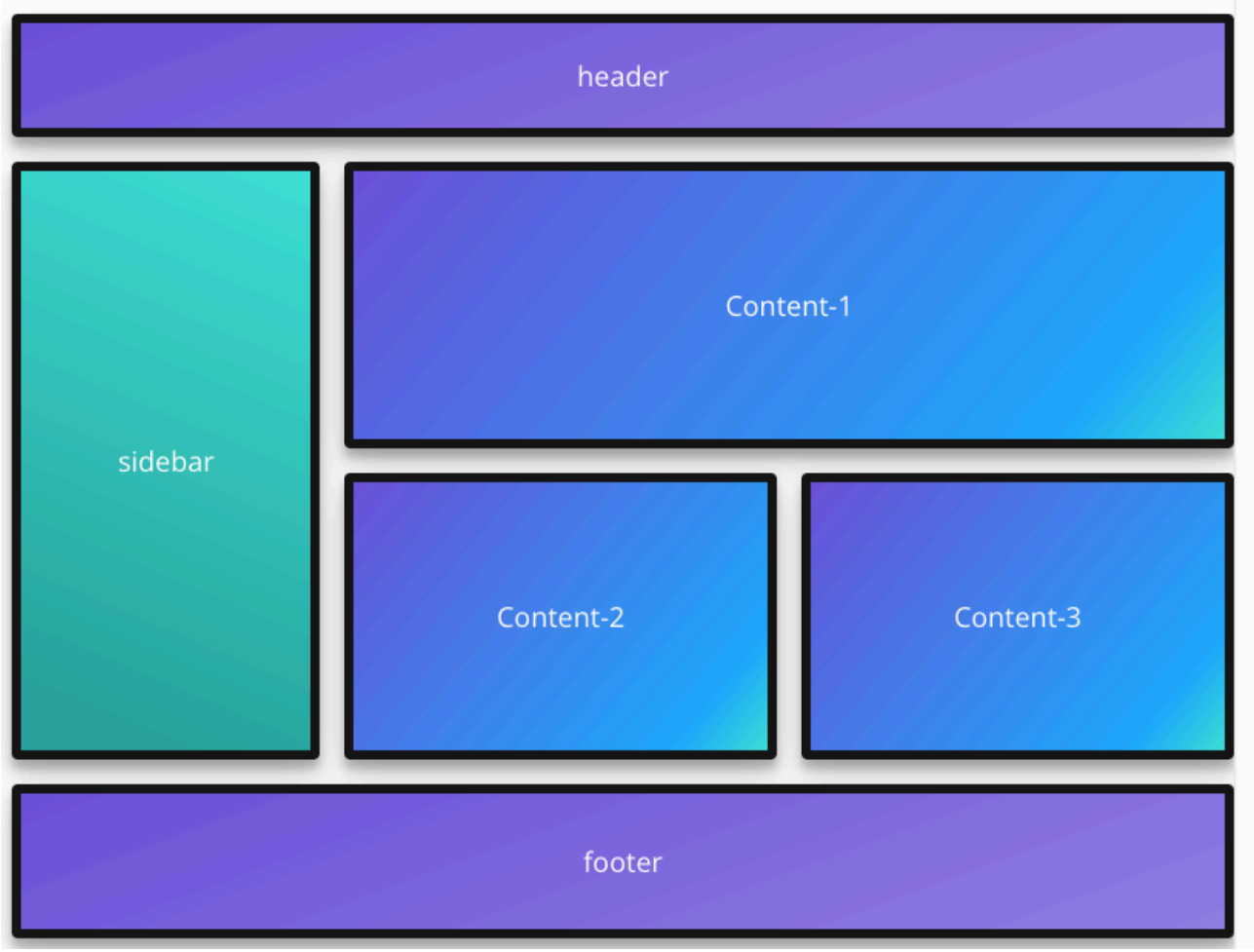
L'exemple ci-dessus serait numéroté comme suit :



https://codepen.io/web-god/pen/ZEJKJzq

Mise en page basique

Nous utiliserons simplement les propriétés raccourcies grid-row et grid-column pour placer manuellement des éléments tels qu'un en-tête, un pied de page, etc.



https://codepen.io/web-god/pen/XWaRaKQ

Template Areas

Une autre méthode de positionnement des éléments consiste à utiliser des zones de grille nommées avec les propriétés grid-template-areas et grid-area.
Une grid-template-areas est composée de plusieurs cellules

Recréons la grille de notre exemple précédent avec la propriété grid-template-areas :

https://codepen.io/web-god/pen/ZEJKJZz

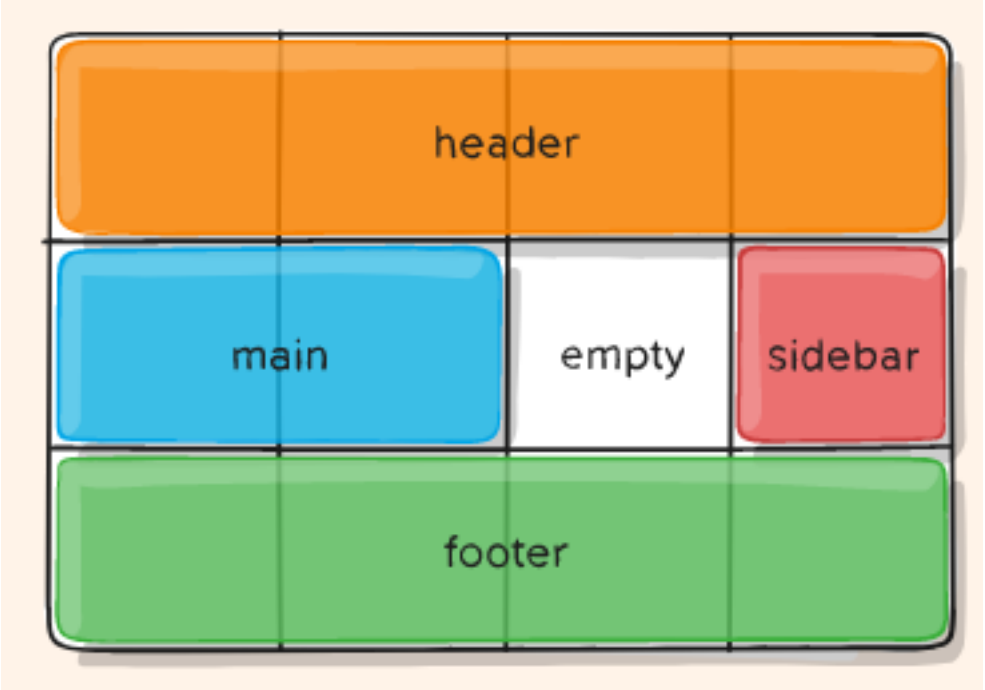
```
<div class="container">
  <div class="header">header</div>
  <div class="sidebar">sidebar</div>
  <div class="content-1">Content-1</div>
  <div class="content-2">Content-2</div>
  <div class="content-3">Content-3</div>
  <div class="footer">footer</div>
</div>
.container {
  display: grid;
  width: 100%;
  height: 600px;
  grid-template-columns: 200px 1fr 1fr;
  grid-template-rows: 80px 1fr 1fr 100px;
  grid-gap: 1rem;
  grid-template-areas:
      "header header header"
      "sidebar content-1 content-1"
      "sidebar content-2 content-3"
      "footer footer footer";
}
.header {
  grid-area: header;
}
.sidebar {
  grid-area: sidebar;
}
.content-1 {
  grid-area: content-1;
}
.content-2 {
  grid-area: content-2;
}
.content-3 {
  grid-area: content-3;
}
.footer {
  grid-area: footer;}
```

Empty cell

Nous pouvons indiquer une cellule vide dans la grille avec un point.

Chaque ligne de votre déclaration doit avoir le même nombre de cellules.

```
.container {
  display: grid;
  grid-template-columns: 50px 50px 50px 50px;
  grid-template-rows: auto;
  grid-template-areas:
    "header header header header"
    "main main . sidebar"
    "footer footer footer footer";
}
```



Justify-items

Aligne les éléments de la grille sur l'axe des rangées (rows).

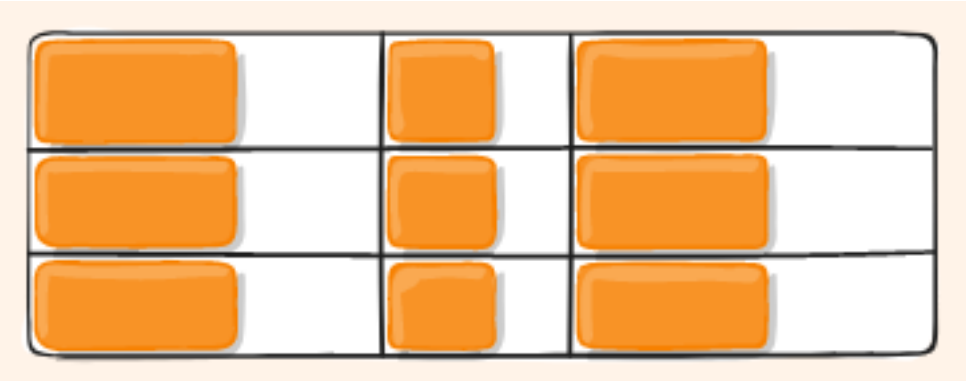
start - aligne les éléments de manière à ce qu'ils soient alignés avec le bord de début de leur cellule.

end - aligne les éléments de manière à ce qu'ils soient alignés avec le bord de fin de leur cellule

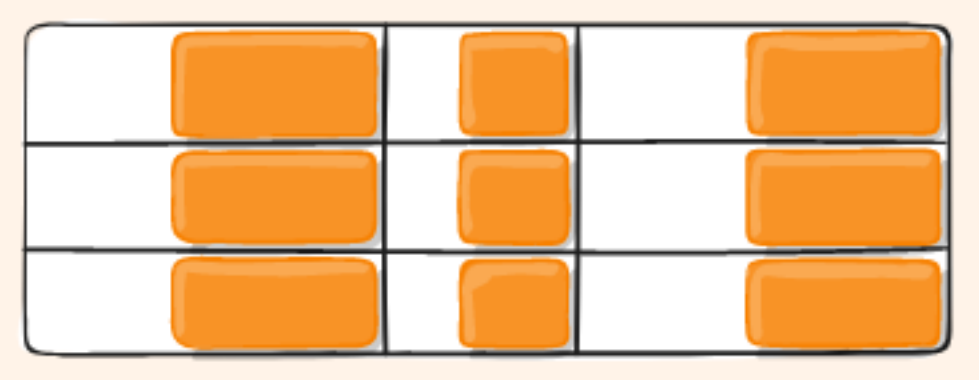
center - aligne les éléments au centre de leur cellule

stretch - remplit toute la largeur de la cellule (valeur par défaut).

justify-items: start;

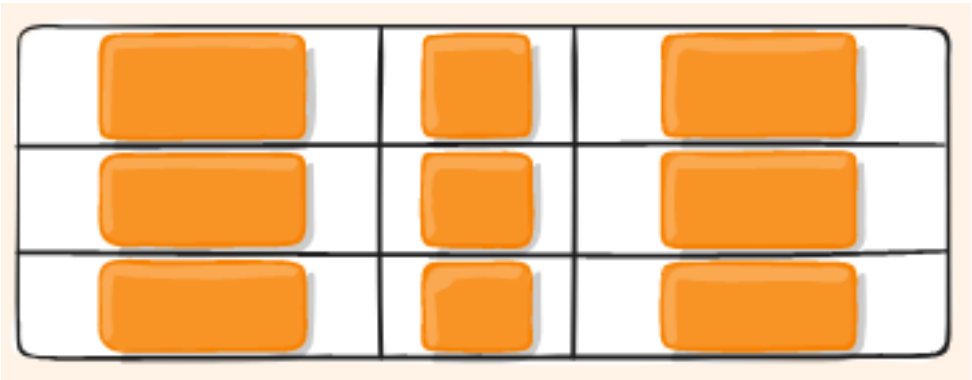


justify-items: end;

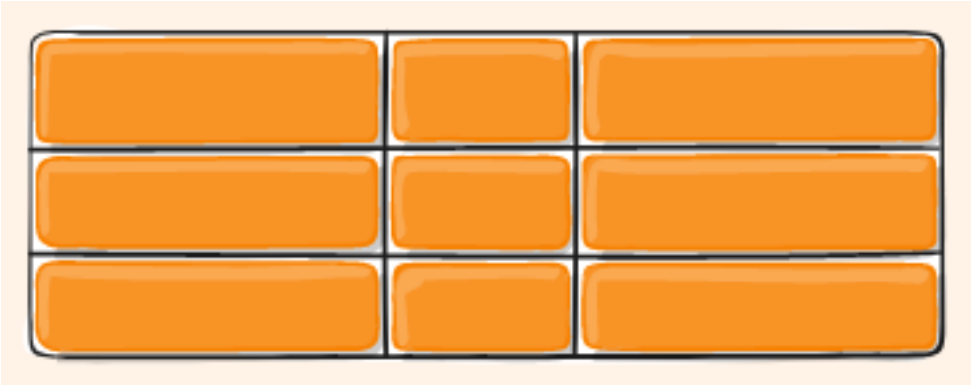


justify-items: center;

justify-items: center;



justify-items: stretch;



align-items

Aligne les éléments de la grille sur l'axe des colonnes (par opposition à justify-items qui s'aligne sur l'axe des lignes).

stretch - remplit toute la hauteur de la cellule (valeur par défaut)

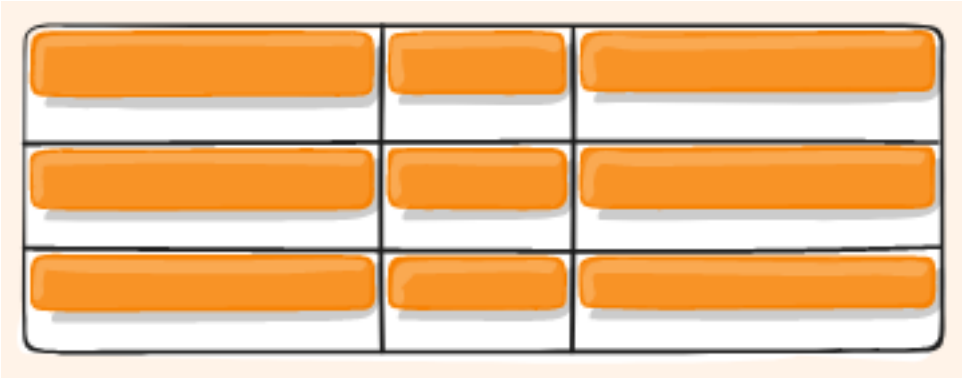
start - aligne les éléments de manière à ce qu'ils soient alignés avec le bord de début de leur cellule

end - aligne les éléments de manière à ce qu'ils soient alignés sur le bord final de leur cellule

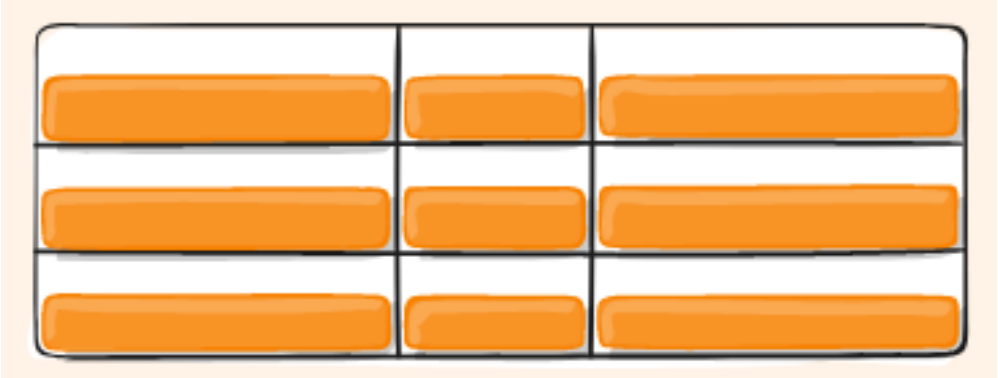
center - aligne les éléments au centre de la cellule.

baseline - aligne les éléments sur la ligne de base du texte.

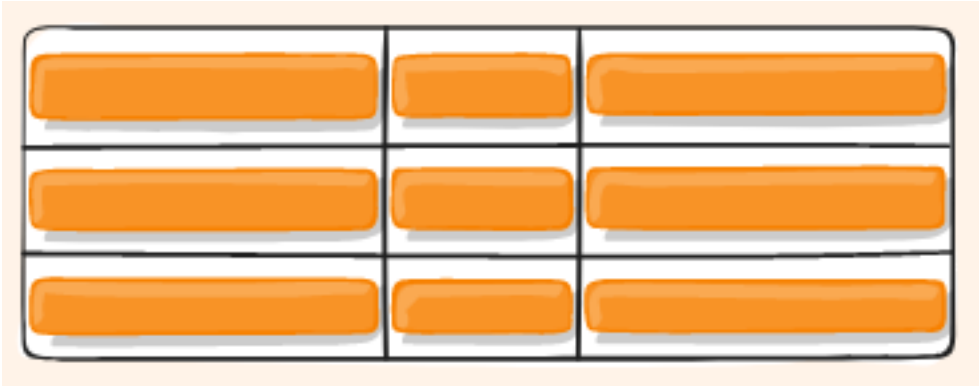
align-items: start;



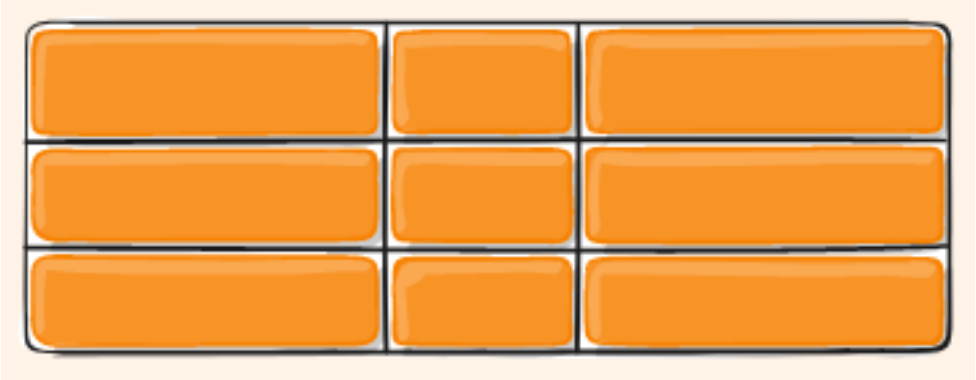
align-items: end;



align-items: center;



align-items: stretch;



justify-content

Parfois, la taille totale de votre grille peut être inférieure à la taille de son conteneur. Cela peut se produire si tous les éléments de votre grille sont dimensionnés avec des unités non flexibles comme les px.

Dans ce cas, vous pouvez définir l'alignement de la grille dans le conteneur de la grille avec justify-content.

Cette propriété aligne la grille sur l'axe des lignes (par opposition à align-content qui aligne la grille sur l'axe des colonnes).

start - aligne la grille sur le bord de départ du conteneur de la grille.

end - aligne la grille de manière à ce qu'elle affleure le bord d'extrémité du conteneur de la grille.

center - Aligne la grille au centre du conteneur de la grille.

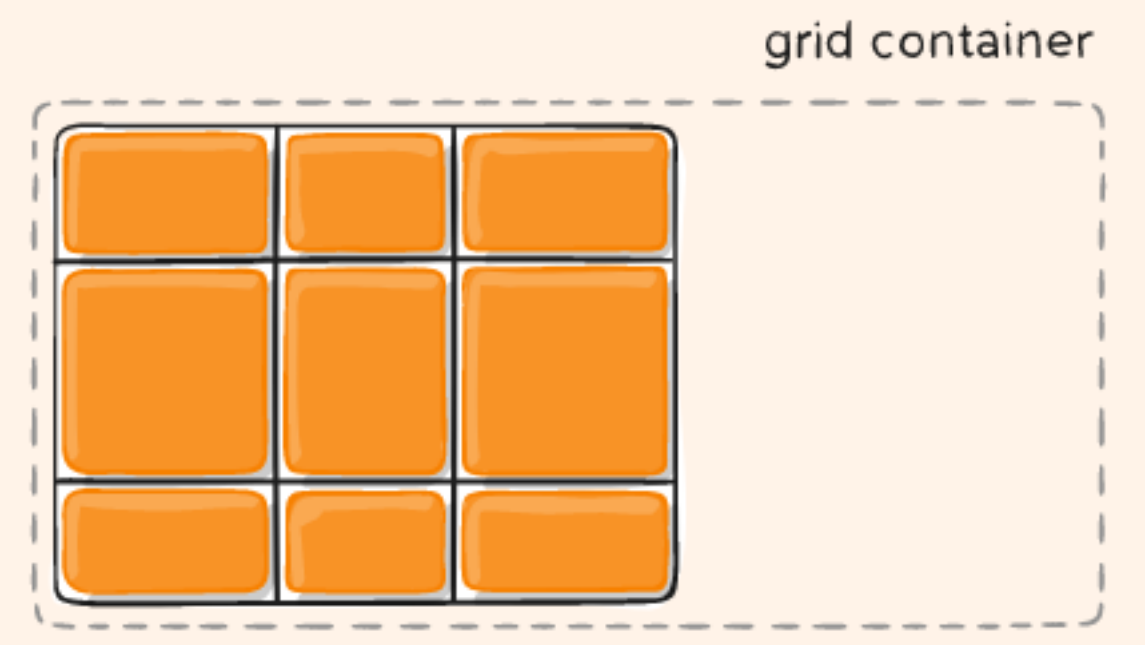
stretch - redimensionne les éléments de la grille pour qu'elle occupe toute la largeur du conteneur.

space-around (espace autour) - place une quantité égale d'espace entre chaque élément de la grille, avec des espaces de taille réduite de moitié sur les extrémités.

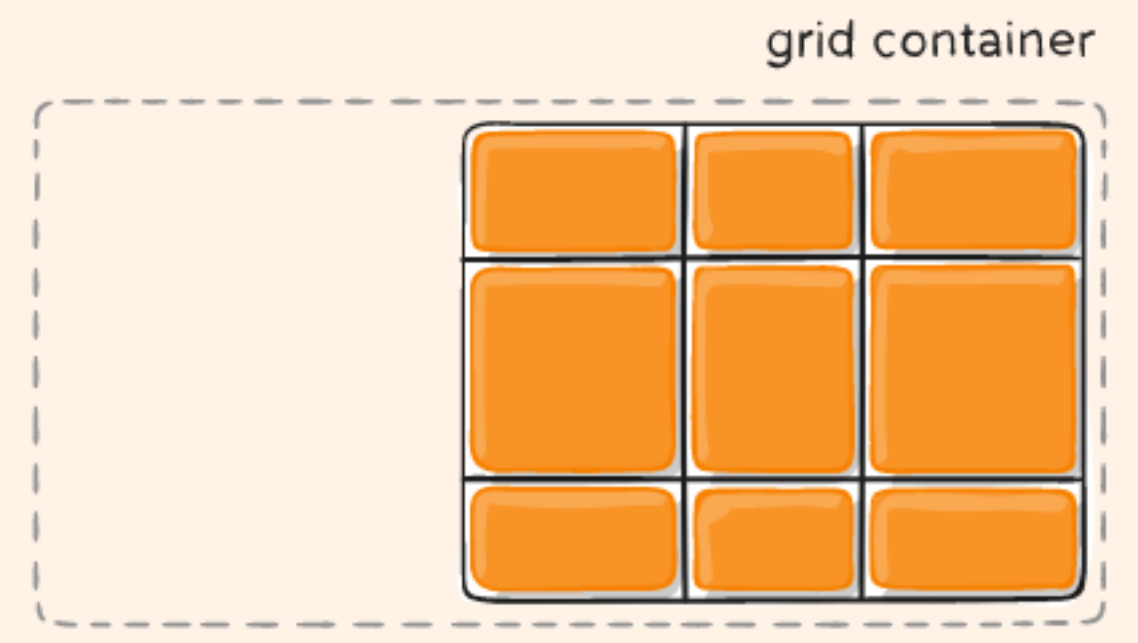
space-between - place une quantité égale d'espace entre chaque élément de la grille, sans espace aux extrémités.

space-evenly - place une quantité égale d'espace entre chaque élément de la grille, y compris les extrémités.

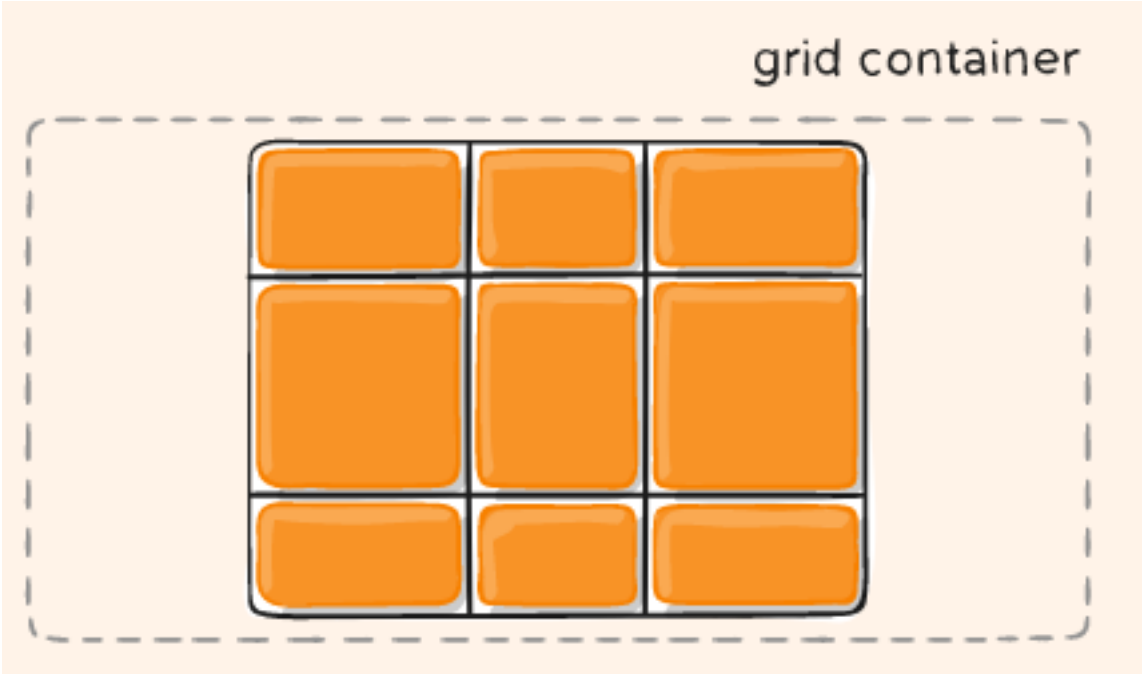
justify-content: start;



justify-content: end;



justify-content: center;



align-content

Cette propriété aligne la grille le long de l'axe des colonnes.

start - aligne la grille sur le bord de départ du conteneur de la grille.

end - aligne la grille de manière à ce qu'elle affleure le bord d'extrémité du conteneur de la grille.

center - Aligne la grille au centre du conteneur de la grille.

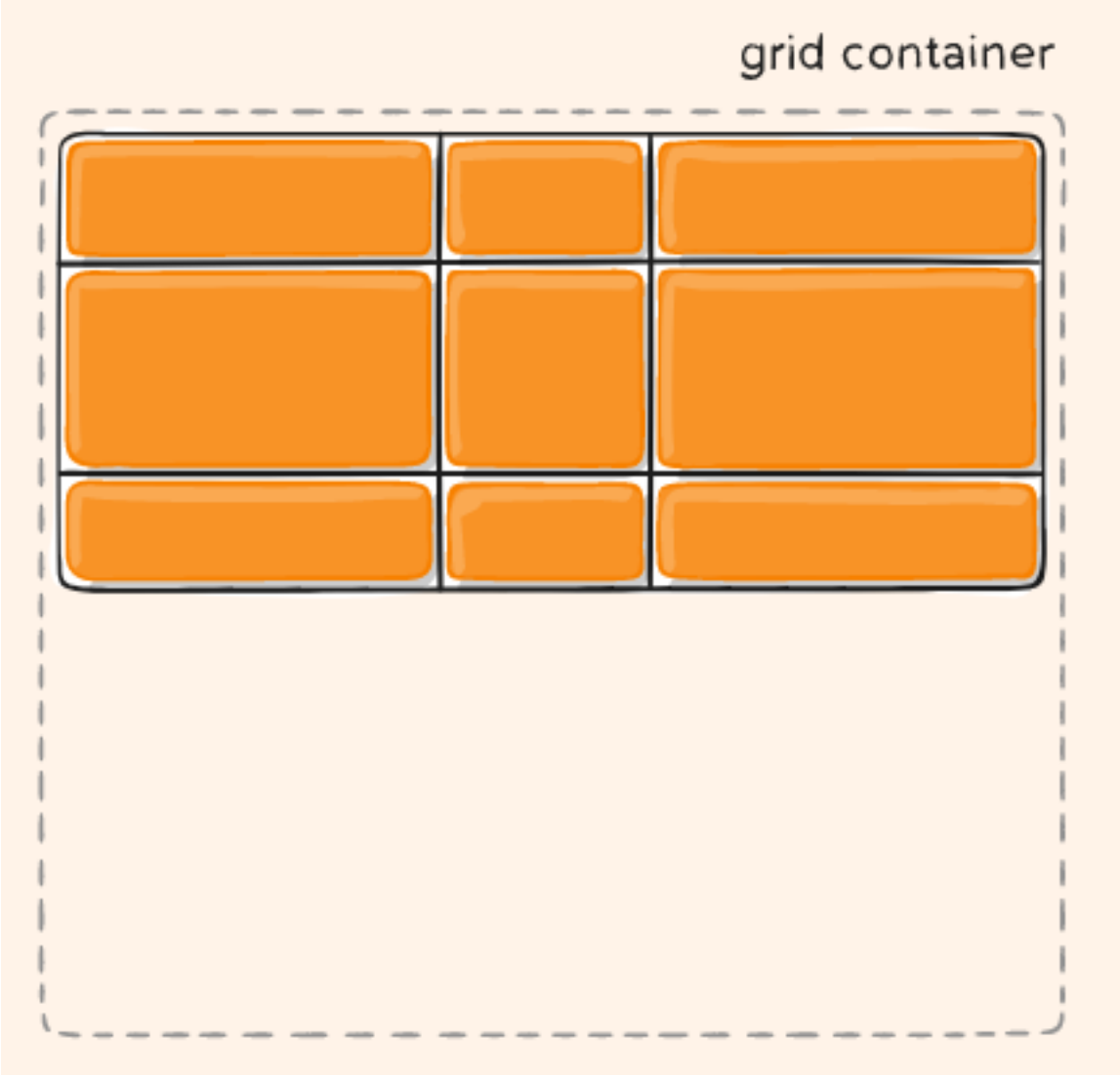
stretch - redimensionne les éléments de la grille pour qu'elle occupe toute la largeur du conteneur.

space-around (espace autour) - place une quantité égale d'espace entre chaque élément de la grille, avec des espaces de taille réduite de moitié sur les extrémités.

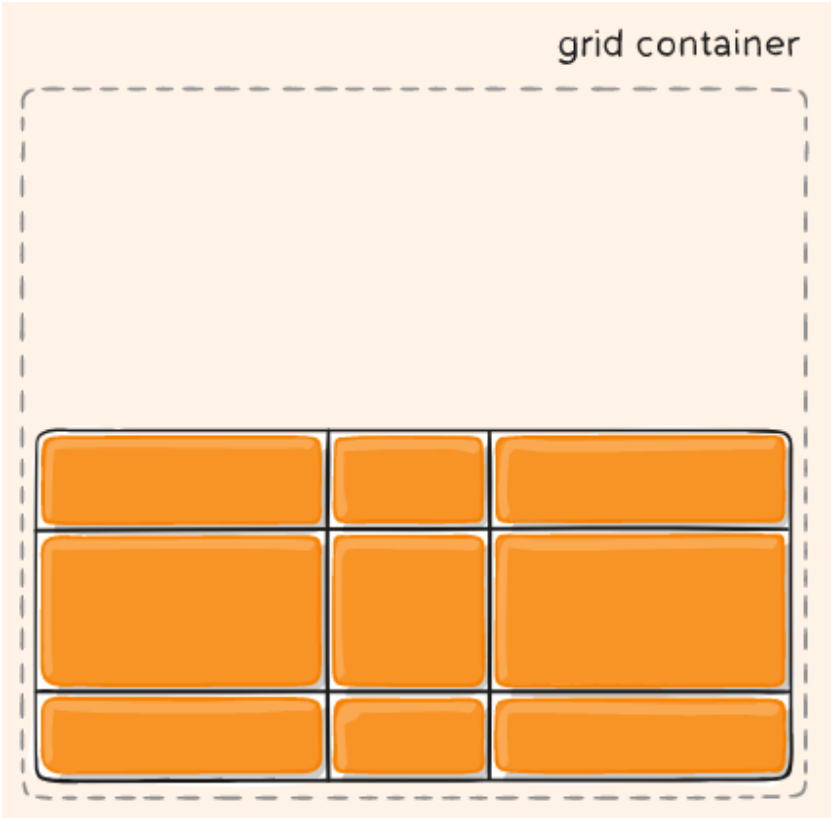
space-between - place une quantité égale d'espace entre chaque élément de la grille, sans espace aux extrémités.

space-evenly - place une quantité égale d'espace entre chaque élément de la grille, y compris les extrémités.

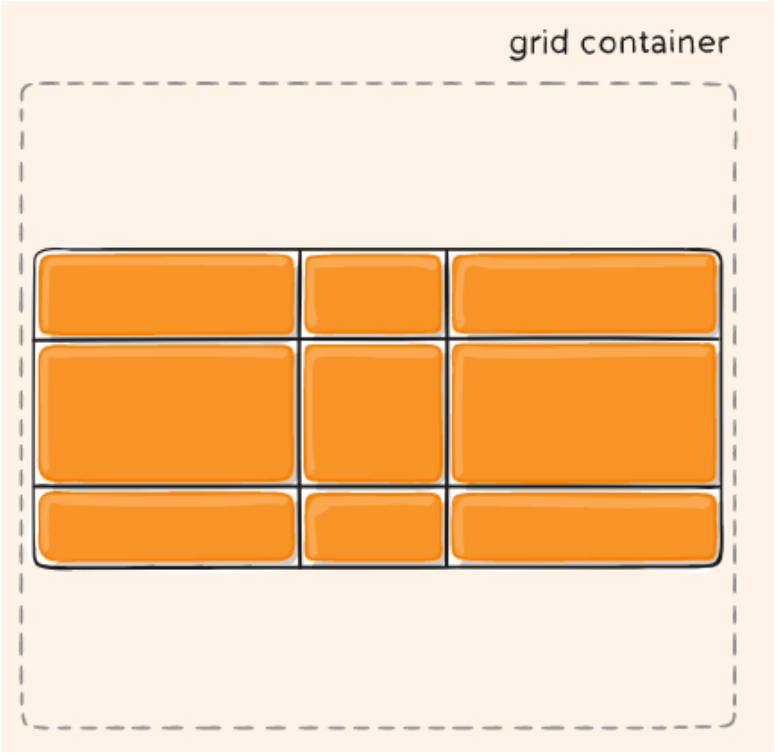
align-content: start;



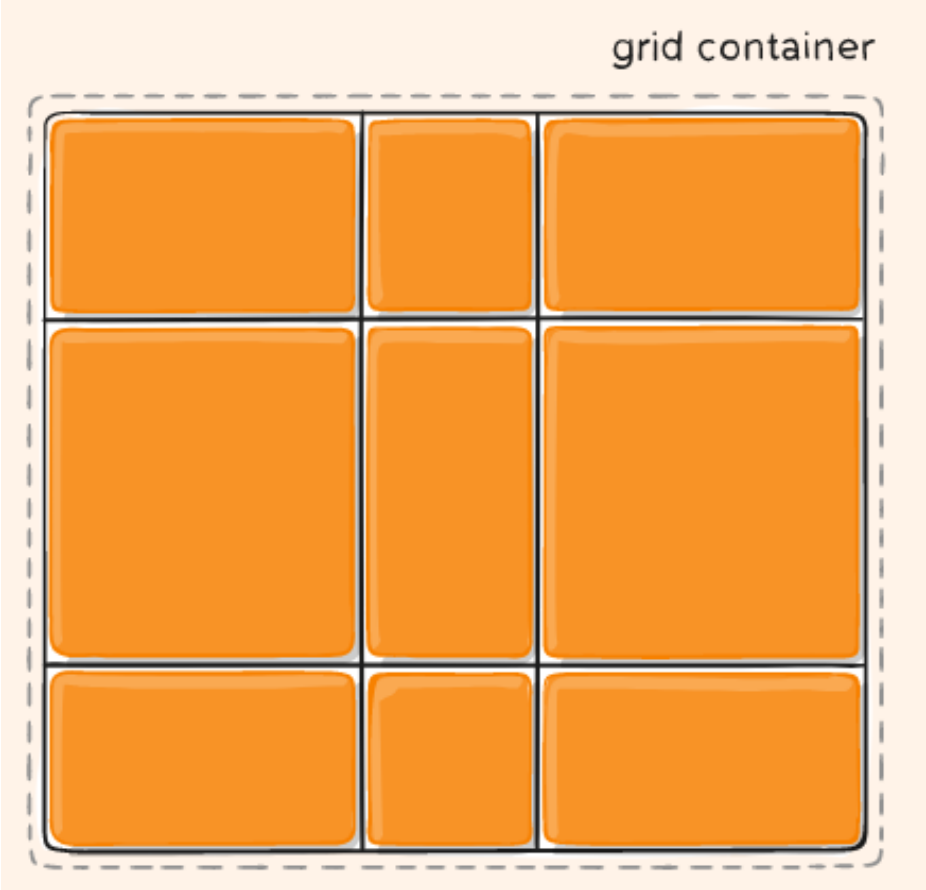
align-content: end;



align-content: center;



align-content: stretch;



Les grid responsive sans media-queries

minmax et auto-fit remplace les breakpoints des media-queries

Exemple :

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

Agrandir un item x 2

Dans les media-queries en mobile first on peut utiliser le span

```
@media (min-width:550px){
    .two-times{
      grid-column: auto / span 2;
  }
}
```

https://codepen.io/web-god/pen/oNeyqKN?editors=1100

Grid
https://mozilladevelopers.github.io/playground/css-grid/
https://gridbyexample.com/examples/

https://www.quackit.com/css/grid/examples/

https://grid.layoutit.com/ - generator

Grid Masonry
https://drafts.csswg.org/css-grid-3/#masonry-layout

Flexbox
https://progressivered.com/fla/?d=3&v=5&h=0&s=0&i=010&a=000

Sprite
https://www.alsacreations.com/tuto/lire/1068-sprites-css-background-position.html

Beaucoup de vidéos sur différents sujets pour développeur
https://www.youtube.com/c/GoogleChromeDevelopers

Mozilla dev tools
https://www.youtube.com/c/MozillaDeveloper/videos