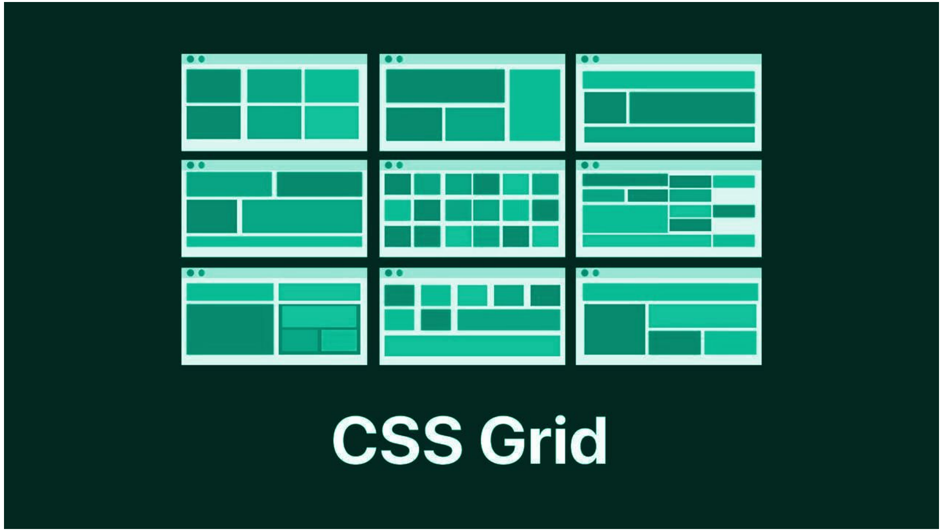
skill academy

CSS Grid



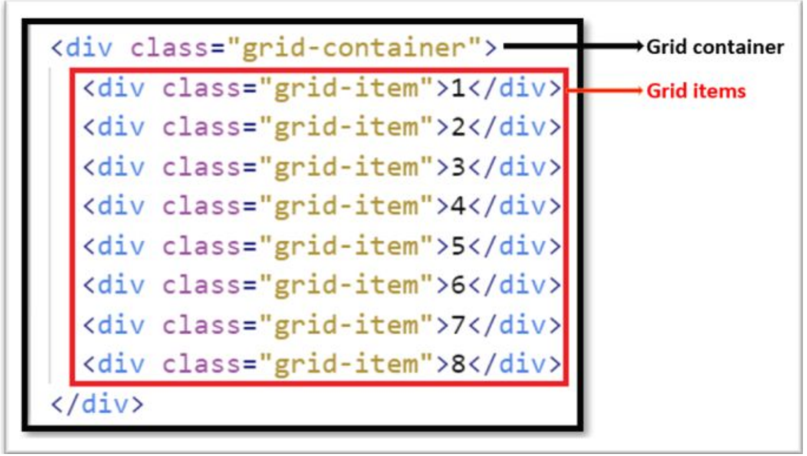
By Nitesh Gupta

CSS Grid

- CSS grid is a two-dimensional layout model with rows and columns to design web pages.

- CSS grid avoids using gloat and positioning.

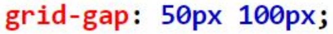
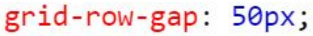
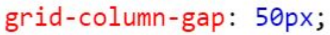
- The grid layout must have a **parent element(container)** with the **display** property “**grid**” or “**inline-grid**”. So that the child elements of the container automatically become **grid items**.



```
<div class="grid-container">
  <div class="grid-item">1</div>
  <div class="grid-item">2</div>
  <div class="grid-item">3</div>
  <div class="grid-item">4</div>
  <div class="grid-item">5</div>
  <div class="grid-item">6</div>
  <div class="grid-item">7</div>
  <div class="grid-item">8</div>
</div>
```

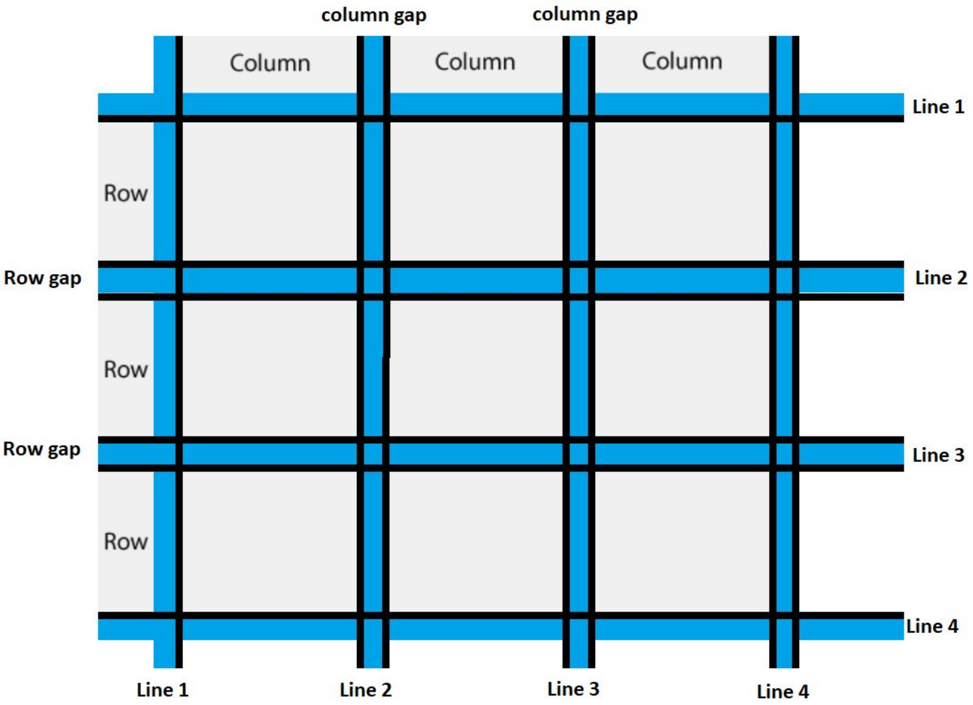
CSS Grid | Important Terminologies | Grid Container properties | Grid Item properties | Difference between Flex and Grid | Example

Important Terminologies

Terminologies	Description
grid columns	The vertical lines of the grid items
grid rows	The horizontal lines of the grid items
grid gaps	The space between each row or column.  `grid-gap: 50px 100px;` The space between each row is called the row gap.  `grid-row-gap: 50px;` The space between each column is called a column gap.  `grid-column-gap: 50px;`
grid lines	The line between each row or column. The line between each line is called the row line. The line between each line is called the column line. The lines are used to place the items in the container by using the following properties: ● grid-column-start ● grid-column-end ● grid-row-start ● grid-row-end

CSS Grid
Important Terminologies
Grid Container properties
Grid Item properties
Difference between Flex and Grid
Example

Important Terminologies



CSS Grid

Important Terminologies

Grid Container properties

Grid Item properties

Difference between Flex and Grid

Example

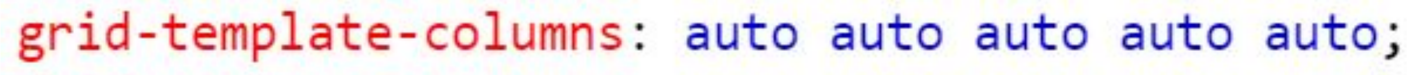
Grid Container Properties

Property: grid-template-columns

Value: Width of each column or auto.

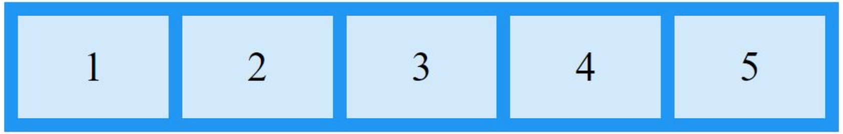
Description: Defines the number of columns and their width in the grid.

Code:



```
grid-template-columns: auto auto auto auto auto;
```

Output:



1	2	3	4	5

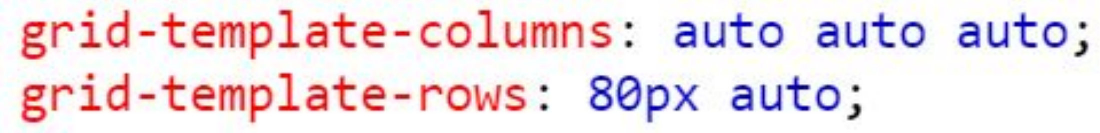
Grid Container Properties

Property: grid-template-rows

Value: Height of each row or auto.

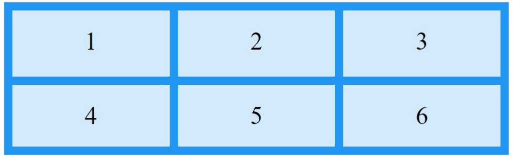
Description: Defines the number of rows and their height in the grid.

Code:



```
grid-template-columns: auto auto auto;
grid-template-rows: 80px auto;
```

Output:



1	2	3
4	5	6

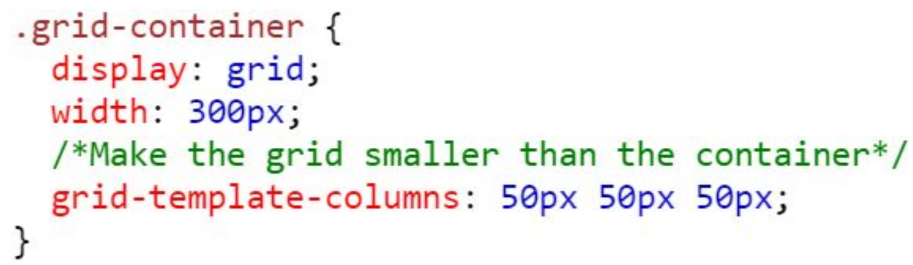
Grid Container Properties

Property: justify-container

Value: space-evenly, space-around, space-between, center, start, end.

Description: Used to align the whole grid inside the container. The total width of the grid must be less than the container’s width to add this property.

Code:



```
.grid-container {
  display: grid;
  width: 300px;
  /*Make the grid smaller than the container*/
  grid-template-columns: 50px 50px 50px;
}
```

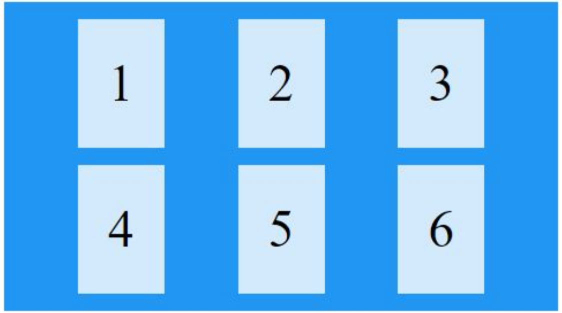
Grid Container Properties

Property: justify-container

Value: space-evenly

Description: Gives an equal amount of space between and around the columns.

Example:



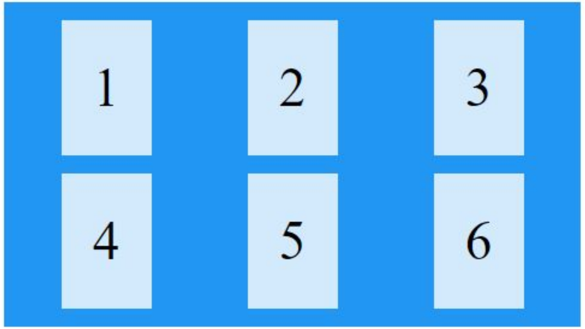
Grid Container Properties

Property: justify-container

Value: space-around

Description: Gives an equal amount of space around the columns.

Example:



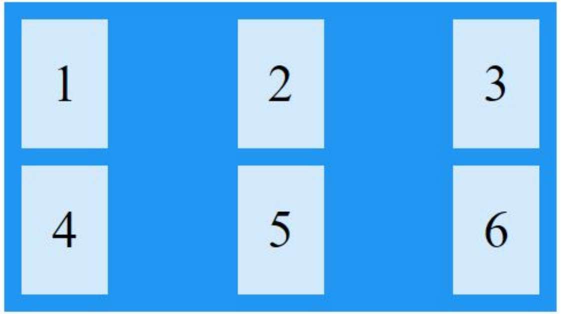
Grid Container Properties

Property: justify-container

Value: space-between.

Description: Gives an equal amount of space between the columns.

Example:



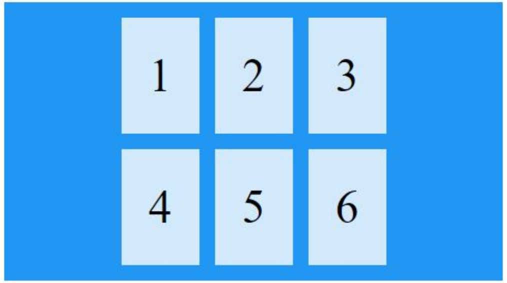
Grid Container Properties

Property: justify-container

Value: center.

Description: Aligns the grid in the middle of the container.

Example:



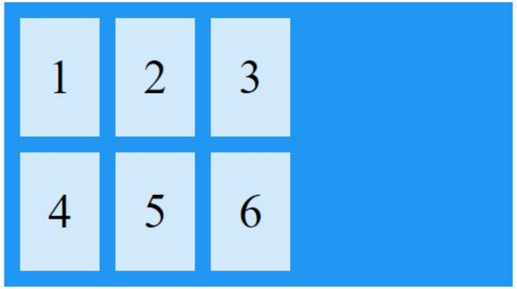
Grid Container Properties

Property: justify-container

Value: start.

Description: Aligns the grid at the beginning of the container.

Example:



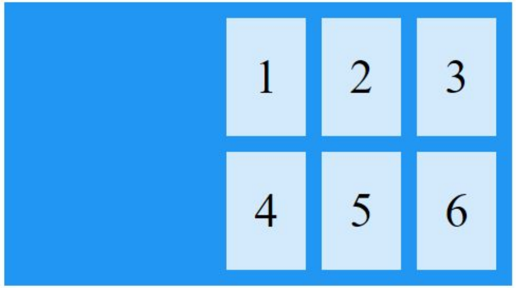
Grid Container Properties

Property: justify-container

Value: end.

Description: Aligns the grid at the end of the container.

Example:



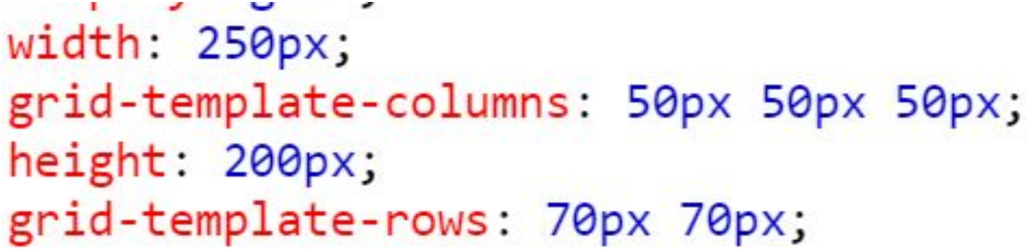
Grid Container Properties

Property: align-content

Value: space-evenly, space-around, space-between, center, start, end.

Description: Used to align the grid items vertically inside the container. The total height of the grid items must be less than the container's height to add this property.

Code:



```
width: 250px;
grid-template-columns: 50px 50px 50px;
height: 200px;
grid-template-rows: 70px 70px;
```

CSS Grid

Important Terminologies

Grid Container properties

Grid Item properties

Difference between Flex and Grid

Example

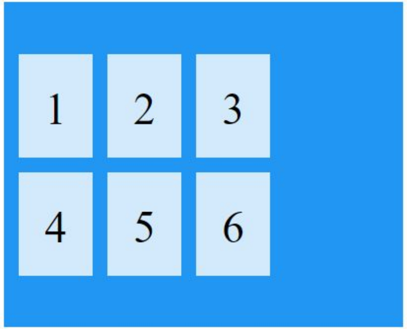
Grid Container Properties

Property: align-content.

Value: center.

Description: Aligns the rows in the center of the container.

Example:



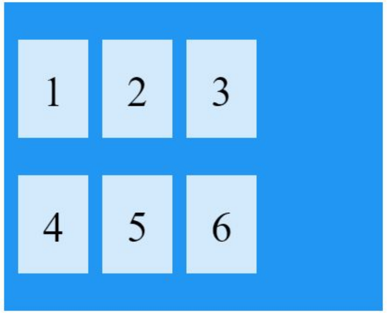
Grid Container Properties

Property: align-content.

Value: space-evenly.

Description: Gives an equal amount of space between and around the rows.

Example:



Grid Container Properties

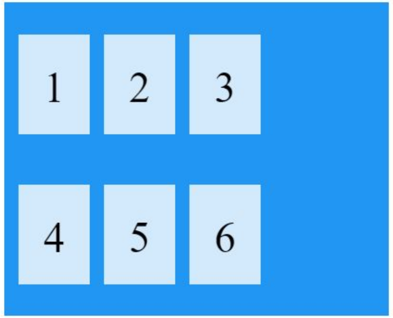
skill academy

Property: align-content.

Value: space-around.

Description: Gives an equal amount of space around the rows.

Example:



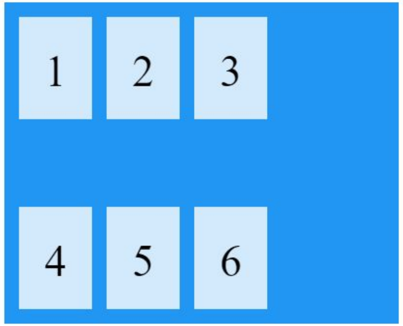
Grid Container Properties

Property: align-content.

Value: space-between.

Description: Gives an equal amount of space between the rows.

Example:



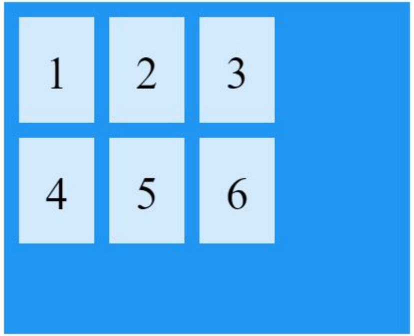
Grid Container Properties

Property: align-content.

Value: start.

Description: Aligns the row at the beginning of the container.

Example:



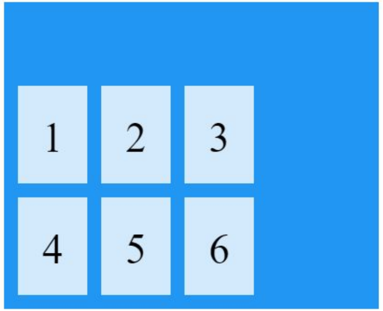
Grid Container Properties

Property: align-content.

Value: end.

Description: Aligns the row at the end of the container.

Example:

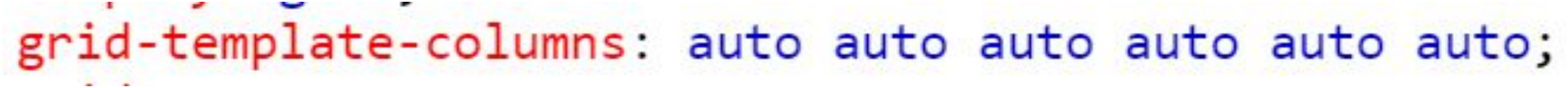


Grid Item Properties

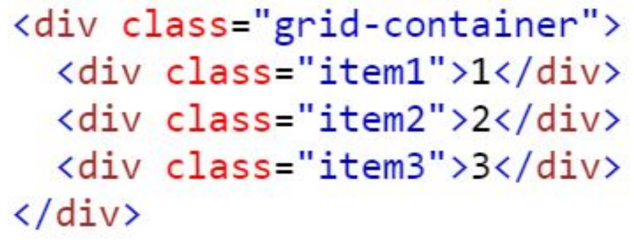
Property: grid-column.   **Value:** column line numbers.

Description: Define where the item will start and end vertically. Shorthand property of grid-column-start and grid-column-end.

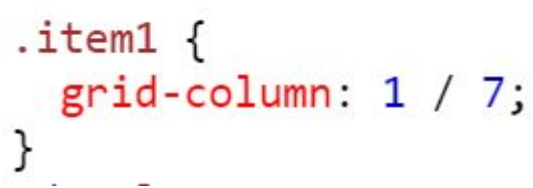
Code:



```
grid-template-columns: auto auto auto auto auto auto;
```



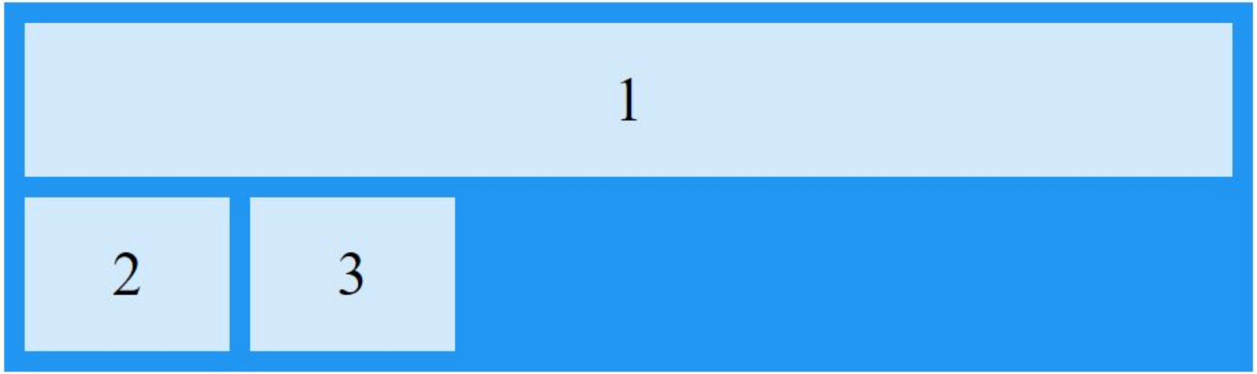
```
<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
</div>
```



```
.item1 {
  grid-column: 1 / 7;
}
```

Grid Item Properties

Output:

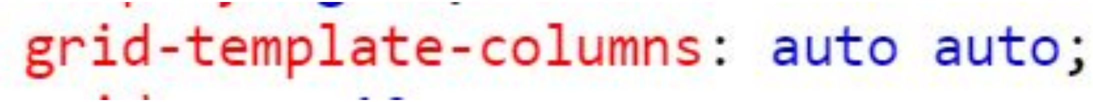


Grid Item Properties

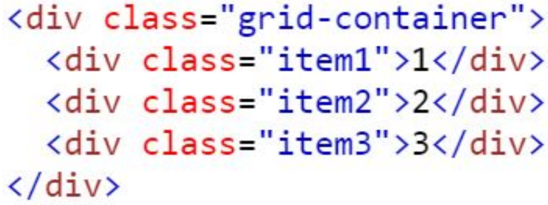
Property: grid-row.   **Value:** rowline numbers.

Description: Define where the item will start and end horizontally. Shorthand property of grid-row-start and grid-row-end.

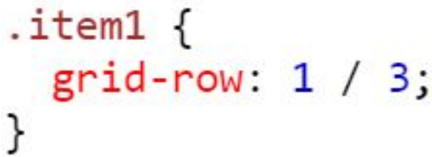
Code:



```
grid-template-columns: auto auto;
```



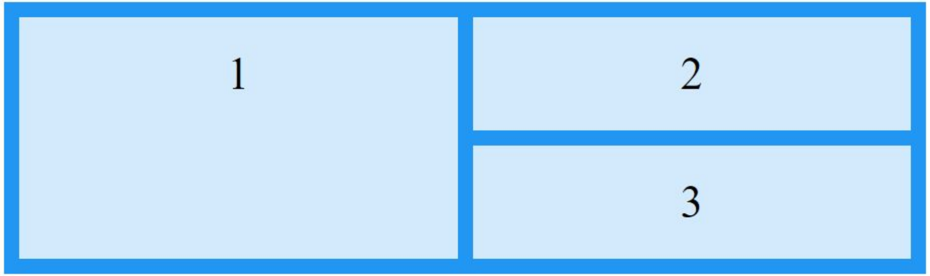
```
<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
</div>
```



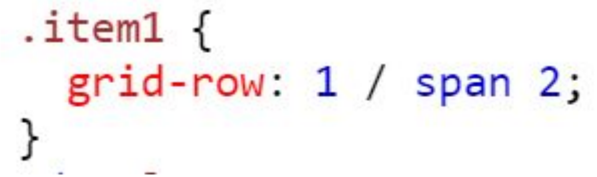
```
.item1 {
  grid-row: 1 / 3;
}
```

Grid Item Properties

Output:



Span can be used instead of line numbers. Which gives same output



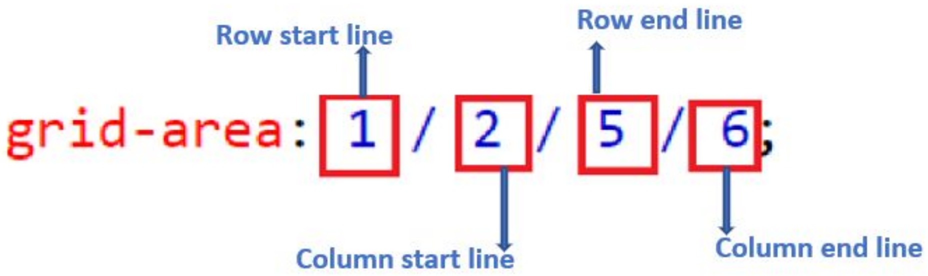
```
.item1 {
  grid-row: 1 / span 2;
}
```

Grid Item Properties

Property: grid-area.

Description: Shorthand property of grid-column-start, grid-column-end, grid-row-start, and grid-row-end.

Code:



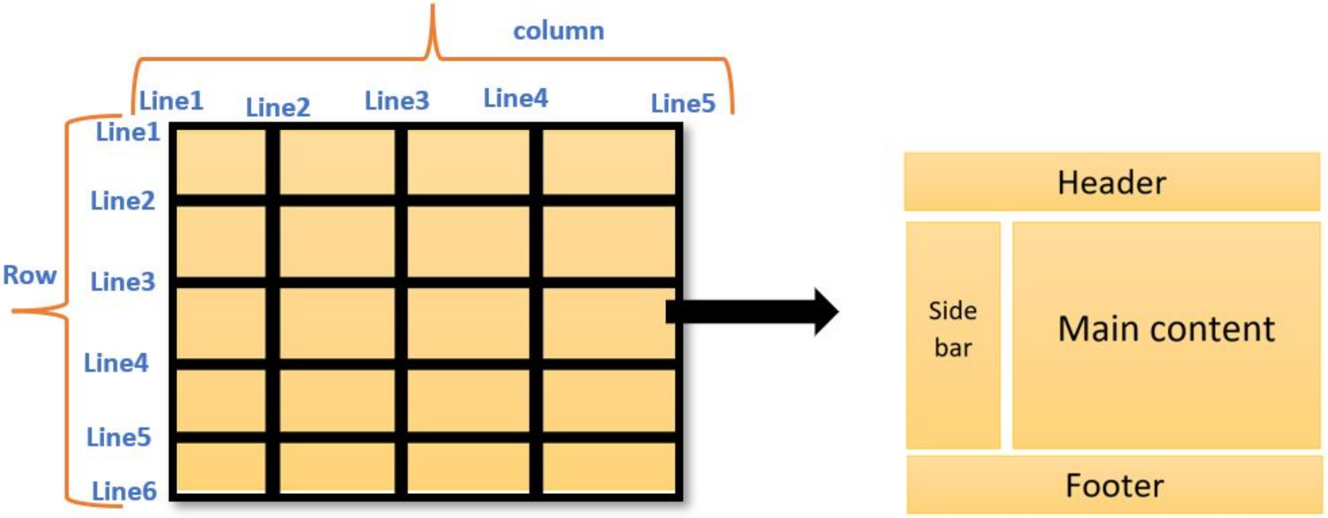
```
grid-area: 1 / 2 / 5 / 6;
```

Difference between Flex and Grid

CSS Flex	CSS Grid
One-dimensional layout can work on either row or column at a time.	Two-dimensional layout can work on rows and columns at a time.
Best suitable for application components and small-scale layouts.	Designed for large-scale layouts that are non-linear in design.
Helps to align contents.	Helps to design the outer layout of the web page.
Content first.	Layout first.

Example

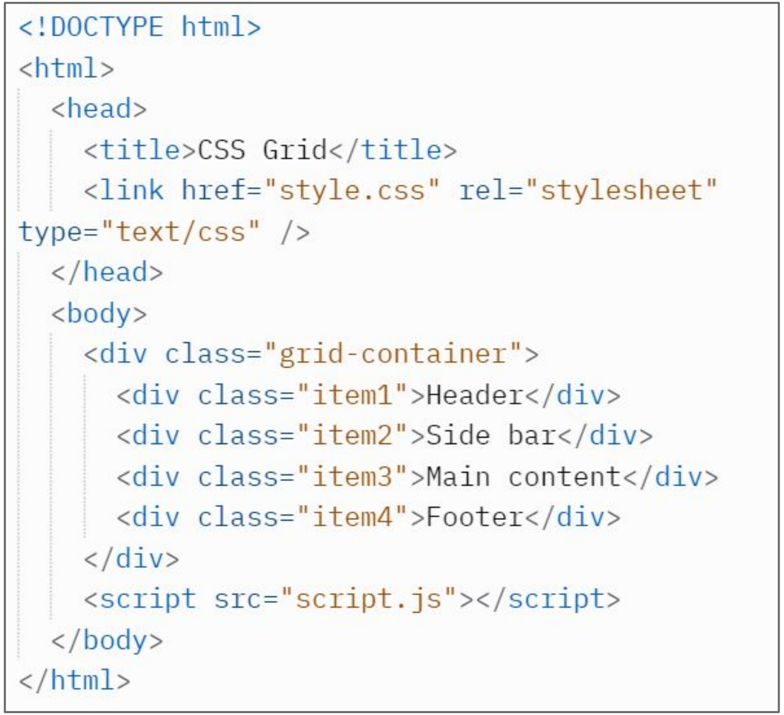
Outline:



- CSS Grid
- Important Terminologies
- Grid Container properties
- Grid Item properties
- Difference between Flex and Grid
- **Example**

Example

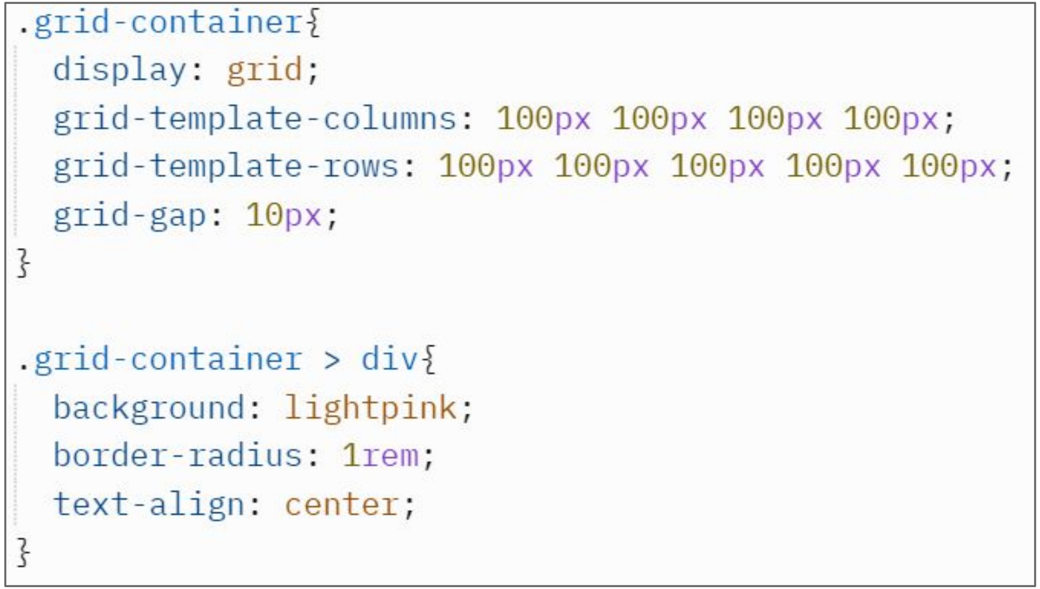
HTML code:



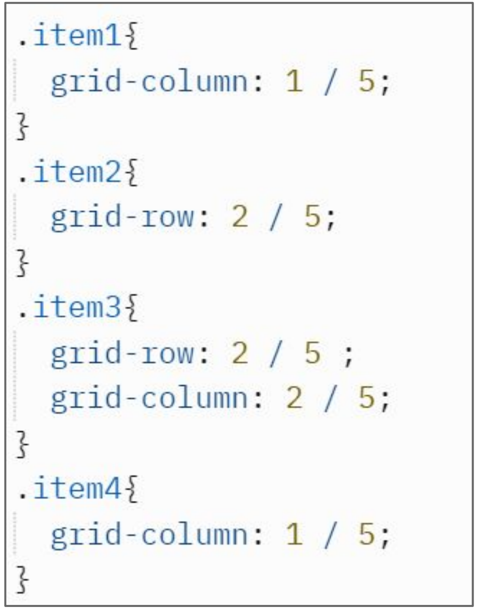
```
<!DOCTYPE html>
<html>
  <head>
    <title>CSS Grid</title>
    <link href="style.css" rel="stylesheet"
type="text/css" />
  </head>
  <body>
    <div class="grid-container">
      <div class="item1">Header</div>
      <div class="item2">Side bar</div>
      <div class="item3">Main content</div>
      <div class="item4">Footer</div>
    </div>
    <script src="script.js"></script>
  </body>
</html>
```

Example

CSS:



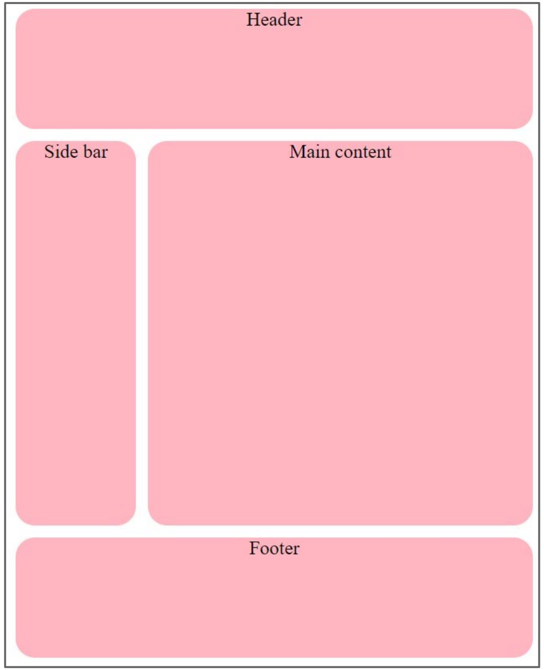
```
.grid-container{
  display: grid;
  grid-template-columns: 100px 100px 100px 100px;
  grid-template-rows: 100px 100px 100px 100px 100px;
  grid-gap: 10px;
}

.grid-container > div{
  background: lightpink;
  border-radius: 1rem;
  text-align: center;
}
```



```
.item1{
  grid-column: 1 / 5;
}
.item2{
  grid-row: 2 / 5;
}
.item3{
  grid-row: 2 / 5 ;
  grid-column: 2 / 5;
}
.item4{
  grid-column: 1 / 5;
}
```

Example

Output:



Header

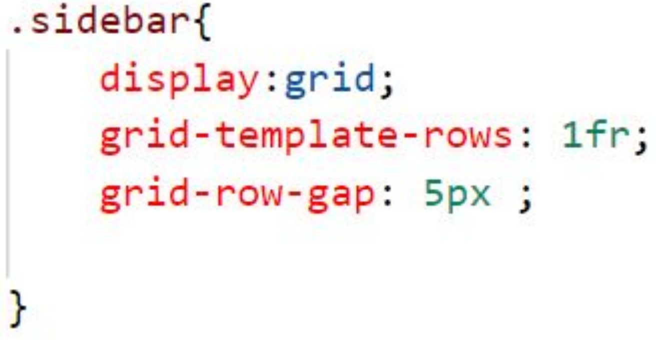
Side bar

Main content

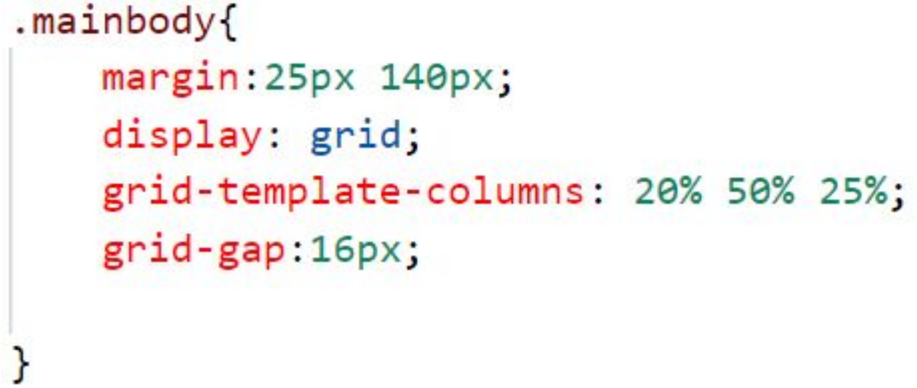
Footer

Example

Grid properties used in our LinkedIn page project



```
.sidebar{
        display:grid;
        grid-template-rows: 1fr;
        grid-row-gap: 5px ;

}
```



```
.mainbody{
        margin:25px 140px;
        display: grid;
        grid-template-columns: 20% 50% 25%;
        grid-gap:16px;

}
```