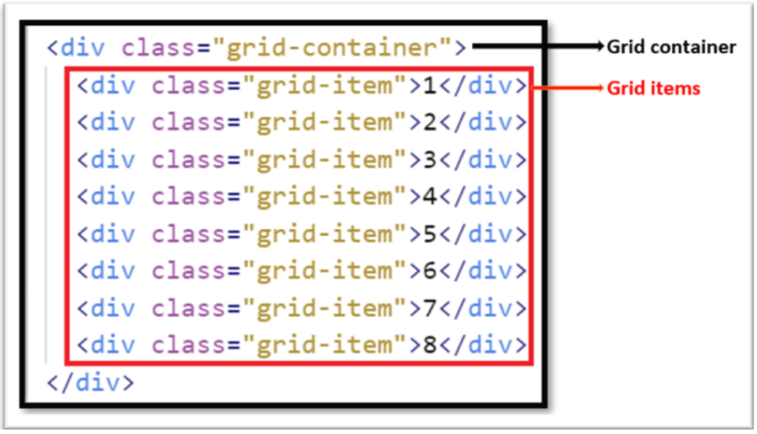
CSS Grid

Topics Covered:

1. What is CSS Grid?
2. Important terminologies.
3. Grid properties.
    a. Grid container properties.
    b. Grid item properties.
4. Difference between CSS Flex and CSS Grid.

CSS Grid:

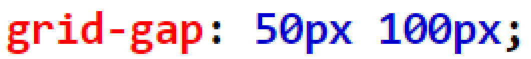
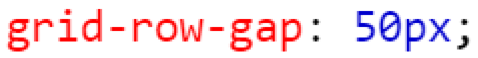
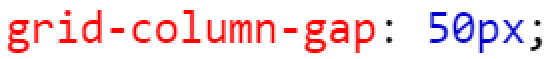
- CSS grid is a two-dimensional layout model with rows and columns to design web pages.
- CSS grid avoids using gloat and positioning.
- The grid layout must have a **parent element(container)** with the **display** property “**grid**” or “**inline-grid**”. So that the child elements of the container 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>
```

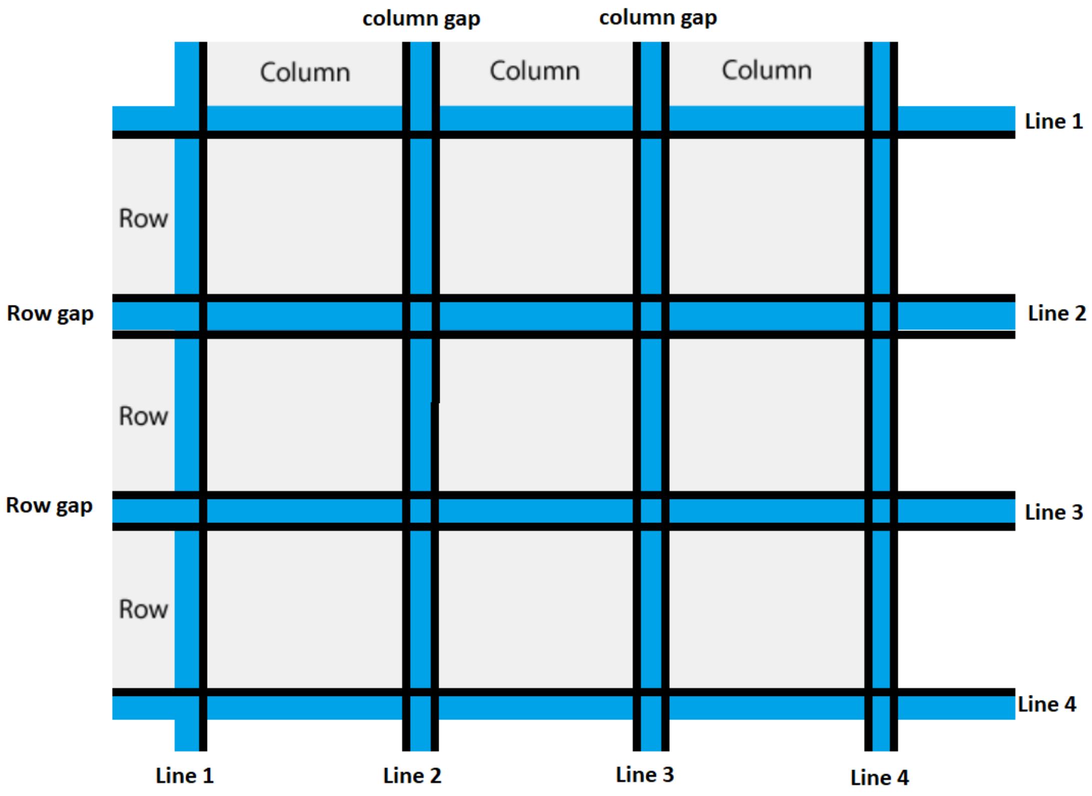
Grid terminologies:

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;`

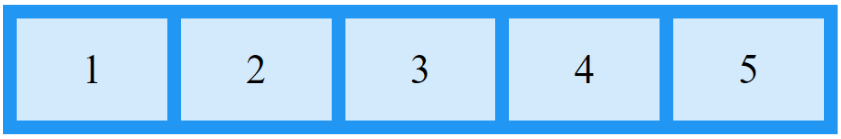
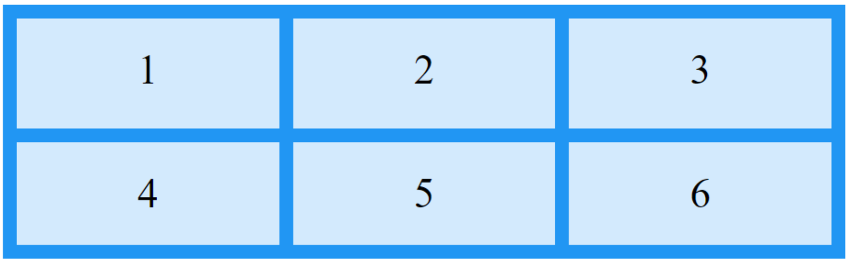
Terminologies	Description
grid lines	The line between each row or column. The line between each line is called the row line. The line between each line is called the column line. The lines are used to place the items in the container by using the following properties: ● grid-column-start ● grid-column-end ● grid-row-start ● grid-row-end

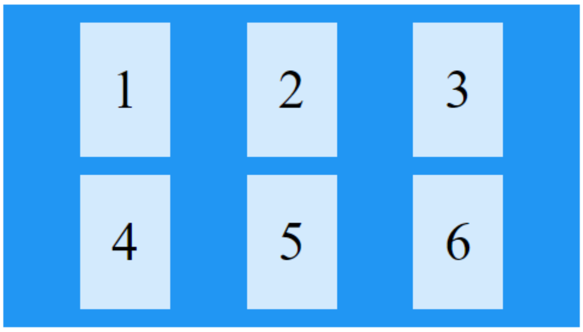
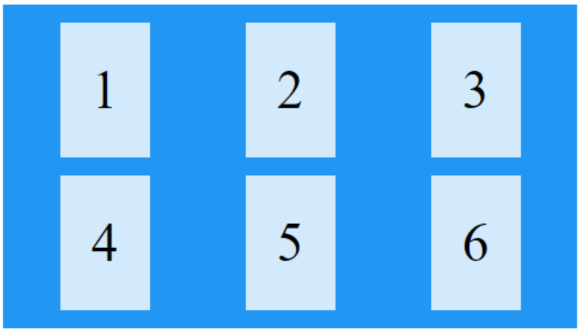
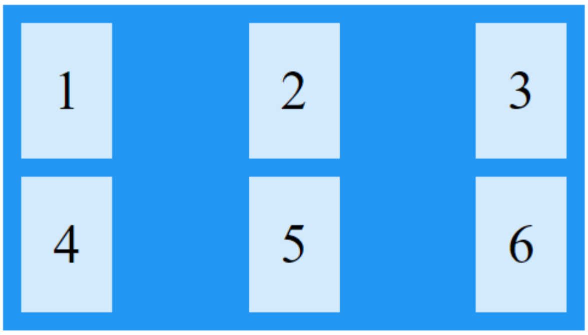
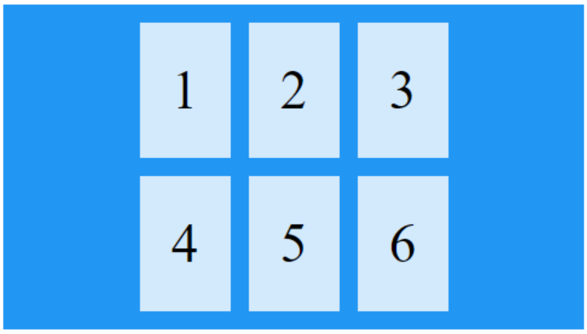
Grid terminologies can be easily understood with help of the following image:

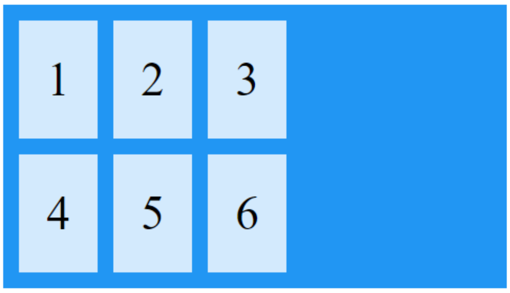
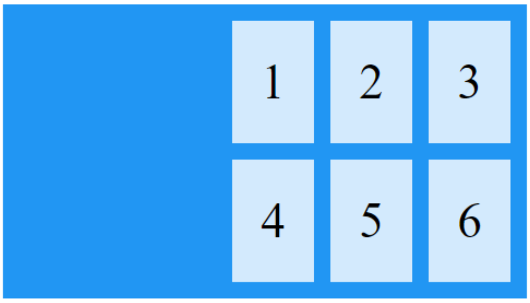


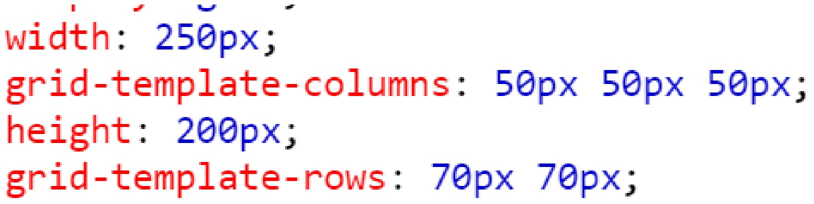
Grid properties:

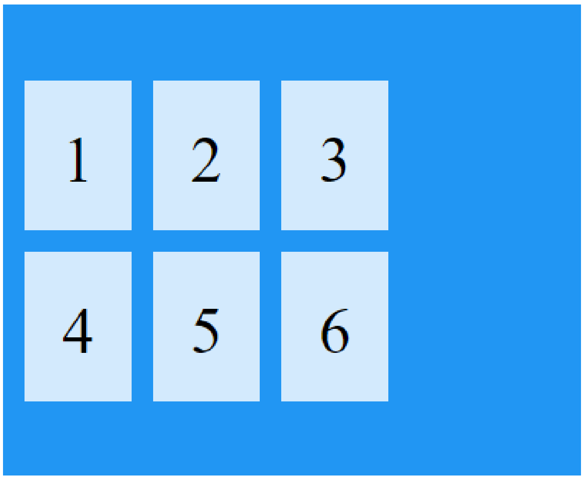
Grid container properties:

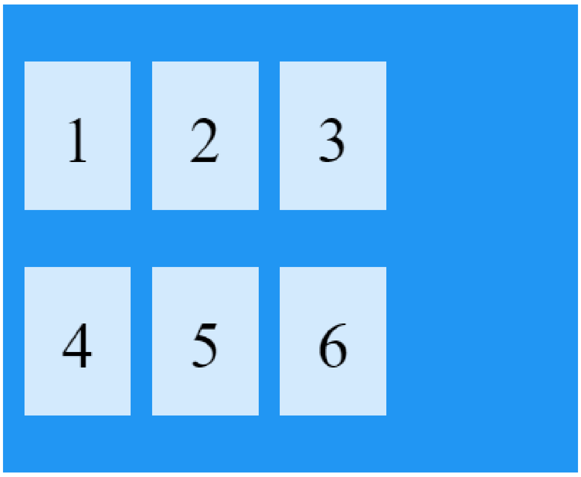
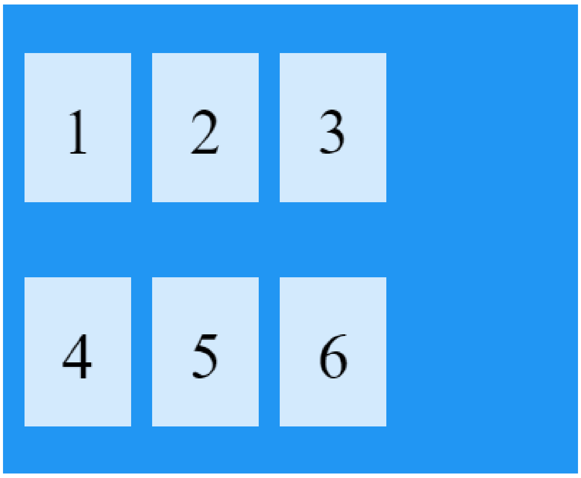
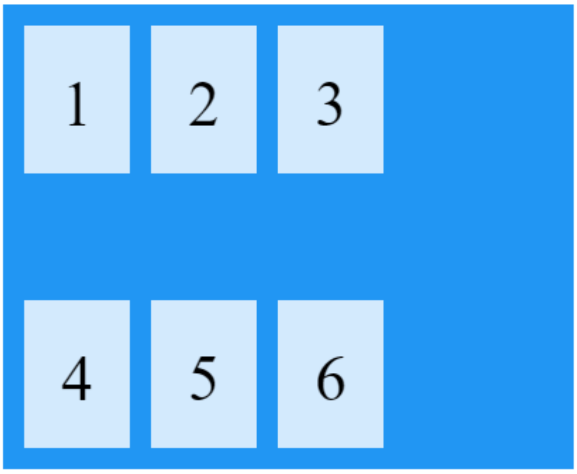
<table>
<tr><th>Property</th><th>Value</th><th>Description</th></tr>
<tr><td rowspan="2"><b>grid-template-columns</b></td><td>Width of each column or auto.</td><td>Defines the number of columns and their width in the grid.</td></tr>
<tr><td colspan="2">Code:
<pre><code>grid-template-columns: auto auto auto auto auto;</code></pre>
Output:

1 2 3 4 5</td></tr>
<tr><td rowspan="2"><b>grid-template-rows</b></td><td>Height of each row or auto.</td><td>Defines the number of rows and their heights in the grid</td></tr>
<tr><td colspan="2">Code:
<pre><code>grid-template-columns: auto auto auto;
grid-template-rows: 80px auto;</code></pre>
Output:

1 2 3
4 5 6</td></tr>
<tr><td><b>justify-container</b></td><td colspan="2">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.
<pre><code>.grid-container {
  display: grid;
  width: 300px;
  /*Make the grid smaller than the container*/
  grid-template-columns: 50px 50px 50px;
}</code></pre></td></tr>
</table>

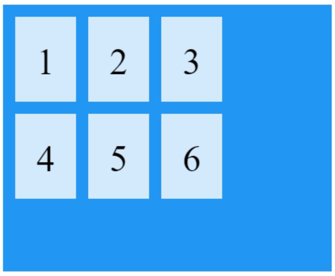
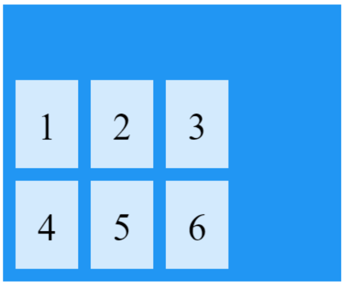
Property	Value	Description
justify-container	space-evenly	Gives an equal amount of space between and around the columns. 
justify-container	space-around	Gives an equal amount of space around the columns. 
justify-container	space-between	Gives an equal amount of space between the columns. 
justify-container	center	Aligns the grid in the middle of the container. 

Property	Value	Description
justify-container	start	Aligns the grid at the beginning of the container. 
justify-container	end	Aligns the grid at the end of the container. 

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

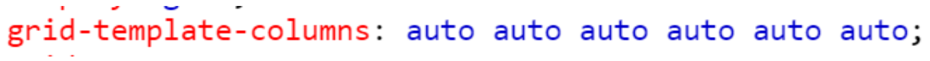
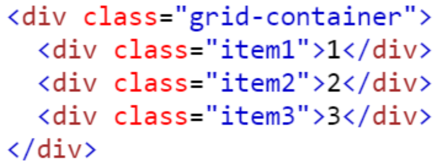
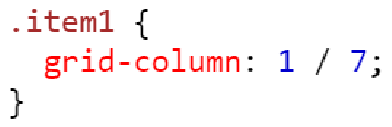
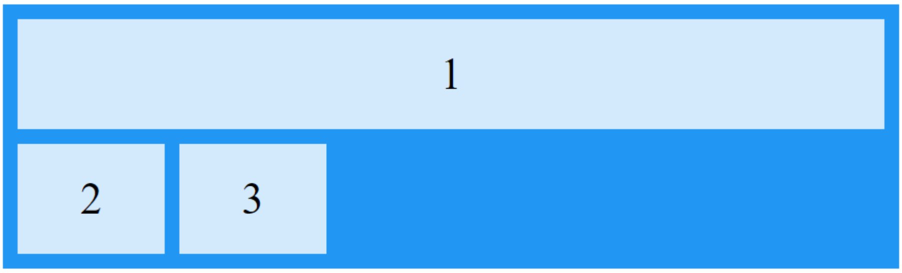
align-content	center	Aligns the rows in the center of the container. 

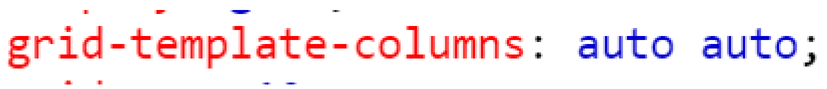
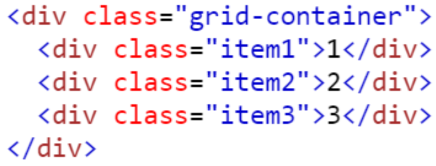
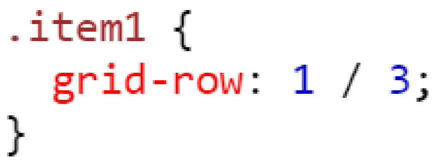
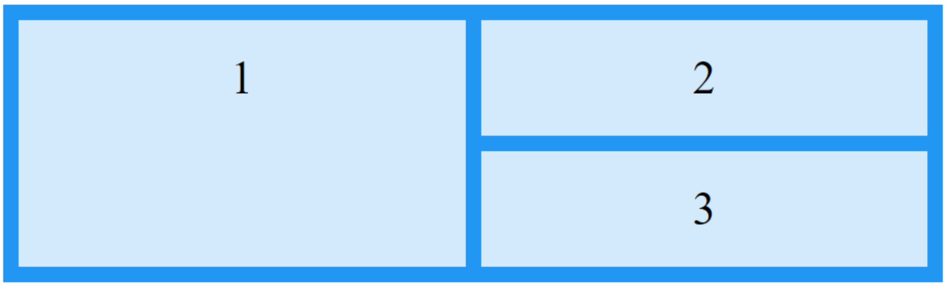
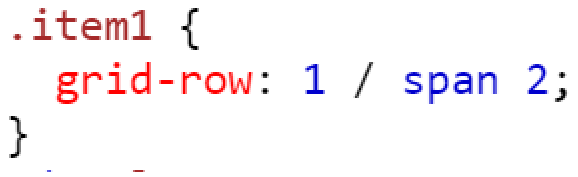
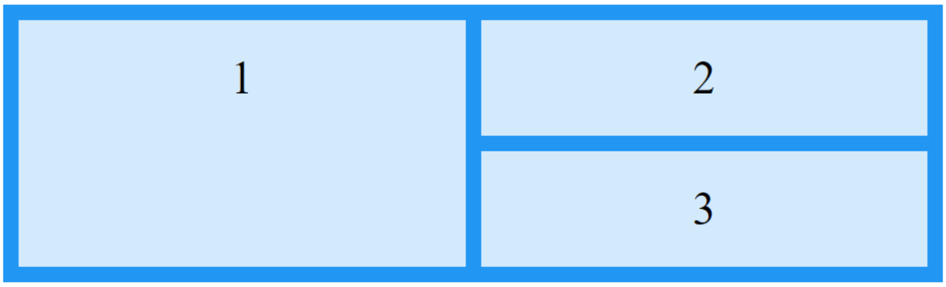
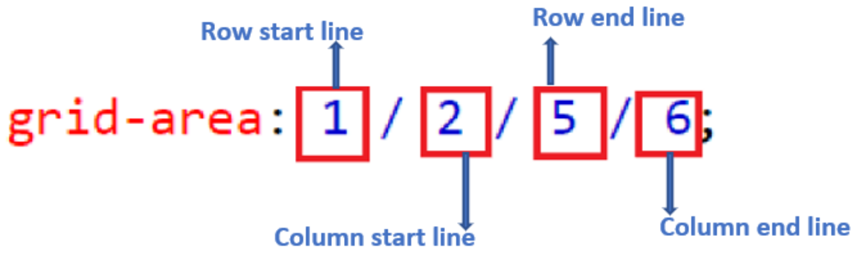
Property	Value	Description
align-content	space-evenly	Gives an equal amount of space between and around the rows. 
align-content	space-around	Gives an equal amount of space around the rows. 
align-content	space-between	Gives an equal amount of space between the rows. 

Property	Value	Description
align-content	start	Aligns the row at the beginning of the container. 
align-content	end	Aligns the row at the end of the container. 

Gird items property:

Property	Value	Description
grid-column	Column line numbers	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;`
	`}`
	● 6 columns, 3 items, 7 column lines, and 3 row lines.
	Output:
		

<table>
<tr><th>Property</th><th>Value</th><th>Description</th></tr>
<tr><td rowspan="3"><b>grid-row</b></td><td>Row line numbers</td><td>Define where the item will start and end horizontally. Shorthand property of grid-row-start and grid-row-end.</td></tr>
<tr><td colspan="2"><b>Code</b>:

grid-template-columns: auto auto;

&lt;div class="grid-container"&gt;
  &lt;div class="item1"&gt;1&lt;/div&gt;
  &lt;div class="item2"&gt;2&lt;/div&gt;
  &lt;div class="item3"&gt;3&lt;/div&gt;
&lt;/div&gt;

.item1 {
  grid-row: 1 / 3;
}
- 2 columns, 3 items, 3 column lines, and 3 row lines
<b>Output</b>:
 </td></tr>
<tr><td colspan="2">Span can be used instead of line numbers.

.item1 {
  grid-row: 1 / span 2;
}
 </td></tr>
<tr><td><b>grid-area</b></td><td colspan="2">Shorthand property of grid-column-start, grid-column-end, grid-row-start, and grid-row-end.

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

Row start line
Row end line
Column start line
Column end line

</td></tr>
</table>

Difference between CSS Flex and CSS 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.