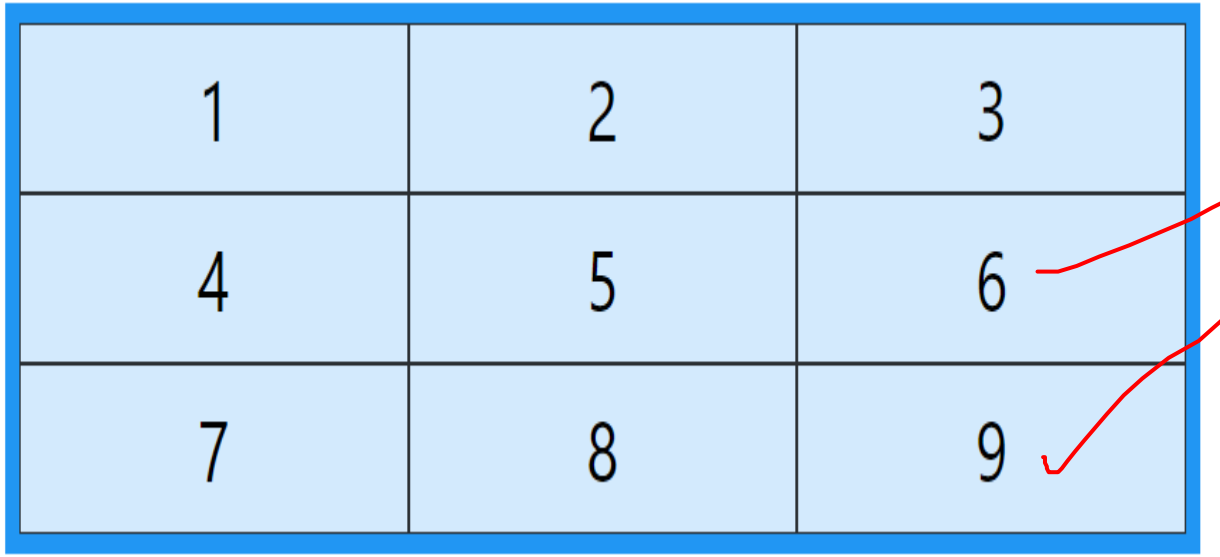
CSS Grid

- A grid can be defined as an **intersecting set of horizontal lines and vertical lines**.
- CSS Grid layout **divides a page into major regions.**
- CSS grid is a **two-dimensional layout system** that allows you to create **rows and columns.**
- You can define the grid using the **grid-template-columns and grid-template-rows properties**.
- These properties allow you to **specify the size** and **number of columns and ro**ws in your grid



Grid Container

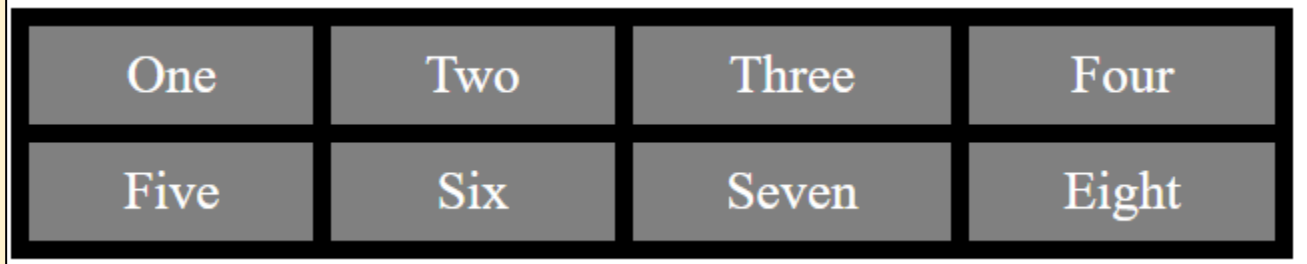
- We can define the **grid container** by setting the **display property to grid** or **inline-grid** on an element.
- Ex. a **grid container with four rows and four columns**, where the **size of the rows and columns is determined automatically** by the content within them.

```
<html> <head>  <style>
        .main {
            display: grid;
            grid: auto auto / auto auto auto auto;
            grid-gap: 10px;
            background-color: black;
            padding: 10px;
        }
        div{
            background-color: grey;
            text-align: center;
            color: white;
            padding: 10px 10px;
            font-size: 30px;
        }   </style>
```

```
</head>

<body>
    <div class="main">
        <div>One</div>
        <div>Two</div>
        <div>Three</div>
        <div>Four</div>
        <div>Five</div>
        <div>Six</div>
        <div>Seven</div>
        <div>Eight</div>
    </div>

</body>
</html>
```



CSS “grid-template Property”

- The **grid-template** CSS property is a shorthand property for defining **grid columns, grid rows, and grid areas.**

The grid-template property is a shorthand property for the following properties:

1) **grid-template-areas** - used to **specify the grid layout** by using the named items.

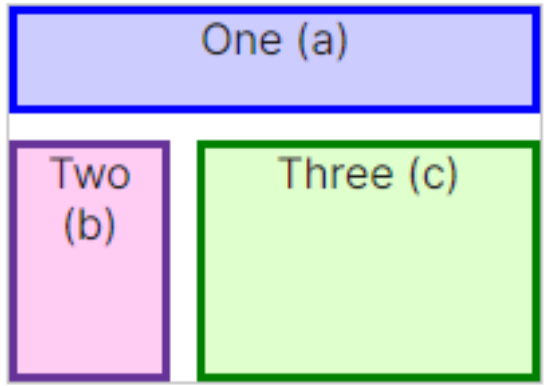
2) **grid-template-rows** - used to specify the **row size**.

3) **grid-template-columns** - used to specify the **size of the columns.**

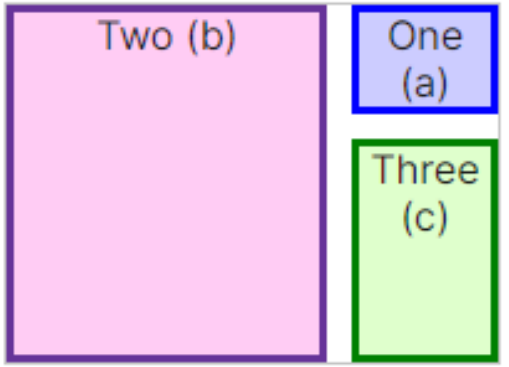
1) grid-template-areas

The **grid-template-areas** CSS property specifies named **grid areas,** establishing the **cells** in the **grid and assigning them names.**

```
grid-template-areas:
"a a a"
"b c c"
"b c c";
```




```
grid-template-areas:
"b b a“
 "b b c"
 "b b c";
```

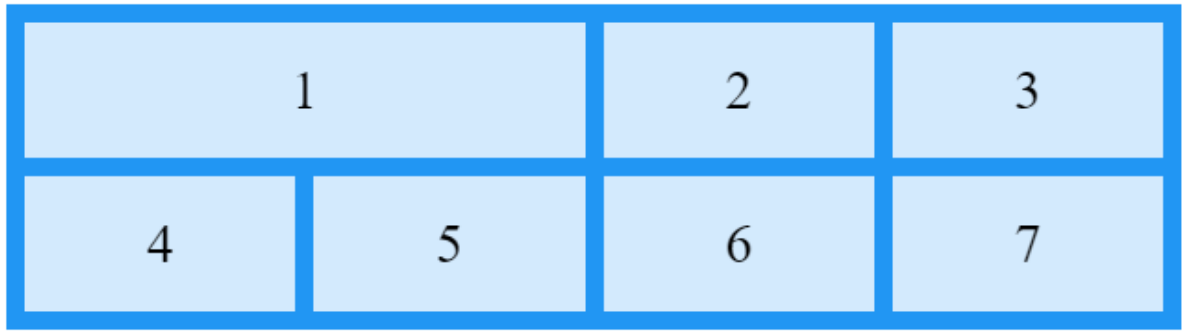


CSS “grid-template Property”

```
<html>
<head>
<style>
.item1 {
  grid-area: myArea;
}
.grid-container {
  display: grid;
  grid-template-areas: ' myArea myArea  . . ';
  grid-gap: 10px;
  background-color: #2196F3;
  padding: 10px;
}
div {
  background-color: cyan
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
}
</style>
</head>
```

```
<body>
<h1>The grid-template-areas Property</h1>
<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
  <div class="item7">7</div>
</div>
</body>
</html>
```

The grid-template-areas Property



CSS grid-template-columns Property

- The **grid-template-columns** property specifies the **number (and the widths) of columns** in a grid layout.

Ex. Given below sets up a **grid container with four equally sized columns**, where **each column's width will be determined automatically** based on its content.

```
<html><head> <style>
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto;
  grid-gap: 10px;
  background-color: #2196F3;
  padding: 10px;}
.item{
    background-color: lightgrey;
            text-align: center;
            padding: 20px 10px;
            font-size: 30px;
} </style></head>
```

```
<body>
<h1>The grid-template-rows Property</h1>
<div class="grid-container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
  <div class="item">6</div>
  <div class="item">7</div>
  <div class="item">8</div>
</div>
</body>
</html>
```

The grid-template-rows Property

1	2	3	4
5	6	7	8

CSS grid-template-rows Property

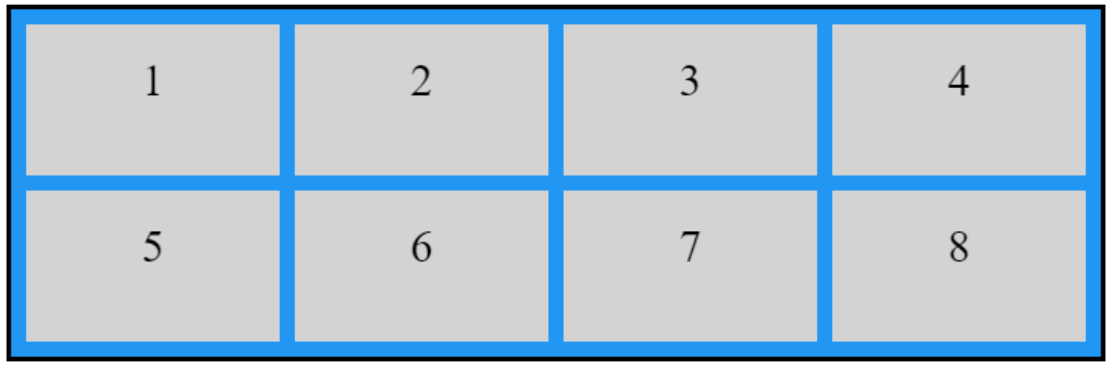
- The grid-template-rows property specifies the **number (and the heights) of the rows** in a grid layout.

Ex. Given below sets up a grid container with two rows, both with a **fixed height** of **100 pixels**

```
<html>
<head> <style>
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto;
  grid-template-rows: 100px 100px;
  grid-gap: 10px;
  background-color: #2196F3;
  padding: 10px;}
.item{
    background-color: lightgrey;
            text-align: center;
            padding: 20px 10px;
            font-size: 30px;
} </style></head>
```

```
<body>
<h1>The grid-template-rows Property</h1>
<div class="grid-container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
  <div class="item">6</div>
  <div class="item">7</div>
  <div class="item">8</div>
</div>
</body>
</html>
```

The grid-template-rows Property



CSS Grid Item

- A grid container contains **grid items**.
- By default, a container has **one grid item for each column, in each row**, but you can style the grid items so that they will **span multiple columns** and/or **rows.**
- **grid-column: 1 / 2 → grid item will start at the first column and end at the 2nd column, spanning a total of TWO columns.**

```
<html><head> <style>
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto;
  grid-template-rows: 100px 100px;
  grid-gap: 10px;
  background-color: #2196F3;
  padding: 10px;}
.div{
    background-color: lightgrey;
            text-align: center;
            padding: 20px 10px;
            font-size: 30px;
  } </style></head>
```

```
.item1 {
  grid-column: 1 / 2;
}
</style>
</head>
<body>
<h1>The grid-item with </h1>
<div class="grid-container">
  <div class="item1">1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
</body>
</html>
```

The grid-item with

1		2
3	4	5
6	7	8

CSS fraction (fr)

- With CSS Grid Layout, we get a new **flexible unit**: the **Fr unit.**
- Fr is a **fractional unit** and **1fr** is for **1 part of the available space.**

```
<html>
<head>
<style>
.grid-container {
  display: grid;
  grid-template-columns: 1fr 1fr 2fr;
  gap: 10px;
  background-color: #2196F3;
  padding: 10px;
}
div {
  background-color: rgba(255, 255, 255, 0.8);
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
} </style></head>
```

```
<body>
<h1>The grid-item with fraction</h1>
<div class="grid-container">
  <div >1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>

</div>
</body>
</html>
```

The grid-item with fraction

1	2	3
4	5	6

CSS repeat()

- **repeat()** is a notation that you can use with the **grid-template-columns** and **grid-template-rows properties** to **make your rules more concise and easier** to understand when creating a large

```
<html>
<head>
<style>
.grid-container {
  display: grid;
  grid-template-columns: repeat(3,  1fr);
  grid-template-rows: repeat(2,80px);
  background-color: #2196F3;
  padding: 10px;
}
div {
  background-color: rgba(255, 255, 255, 0.8);
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
} </style></head>
```

```
<body>
<h1>The grid-item with fraction</h1>
<div class="grid-container">
  <div >1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>

</div>
</body>
</html>
```

minmax()

- The **minmax()** CSS function defines a **size range greater than or equal to min and less than or equal to max**.

```
<html>
  <head>
      <style>
      .grid-container {
        display: grid;
        grid-template-columns: minmax(100px, 1fr) minmax(200px, 2fr) minmax(150px, 1fr);
        grid-gap: 10px;
      }

      .grid-item {
        background-color: lightblue;
        border: 1px solid gray;
        padding: 10px;
        text-align: center;
      }
    </style>   </head>
```

```
<body>
    <div class="grid-container">
      <div class="grid-item">Item 1</div>
      <div class="grid-item">Item 2</div>
      <div class="grid-item">Item 3</div>
    </div>
  </body>
</html>
```

CSS grid-gap Property

- The **grid-gap** property **defines the size of the gap** between the **rows and columns** in a grid layout, and is a shorthand property for the following properties
- grid-row-gap ;grid-column-gap

```
<html>
  <head>
    <title>minmax() example</title>
    <style>
      .grid-container {
        display: grid;
        grid-template-columns:auto auto auto;
        grid-gap: 10px;
      }

      .grid-item {
        background-color: lightblue;
        border: 1px solid gray;
        padding: 10px;
        text-align: center;
      }
    </style>   </head>
```

```
<body>
    <div class="grid-container">
      <div class="grid-item">Item 1</div>
      <div class="grid-item">Item 2</div>
      <div class="grid-item">Item 3</div>
      <div class="grid-item">Item 4</div>
      <div class="grid-item">Item 5</div>
      <div class="grid-item">Item 6</div>
    </div>
  </body>
</html>
```