Project 198

DP EDUCATION

Coding School


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel= "stylesheet"
    href= "imageOpacity.css">
    <title>Document</title>
</head>
<body>
    <div class="demo-wrap">
        <img
          class="img-bg"
          background-image src="scenery.jpg"
          alt=""
        >
        <div class="text">
          <h1>Hello World!</h1>
        </div>
```

html Glitch

See the webpage

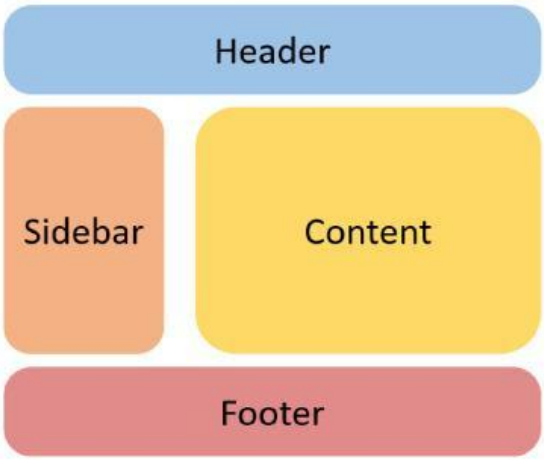
Glitch Start here

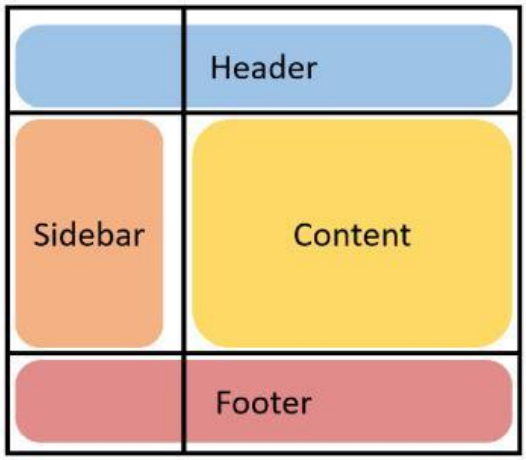
GRID

- Sometimes the layout of a content is in rows as well as in columns. Then we will use grids.
- A very common case of using a grid is to create a photo gallery.

There are images of each size in this photo gallery. But all these images are displayed in a grid. There are 3 rows and 3 columns in this photo gallery.

- Another example of using grids is the layout of many websites.





- Now let's see how to define a grid

- First we need a container class. The display property of the container class should be given as grid.
- Then the following two properties are used to give the number of rows and columns we need.
    - grid-template-rows
    - grid-template-columns
- Now code the index.html page as follows.

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet" href="style.css">
    <title>Grids</title>
  </head>
<body>
    <div class="container"></div>
  </body>
</html>
```

- Now code the style.css page as follows.

Here a grid of 3 rows and 2 columns will be created.

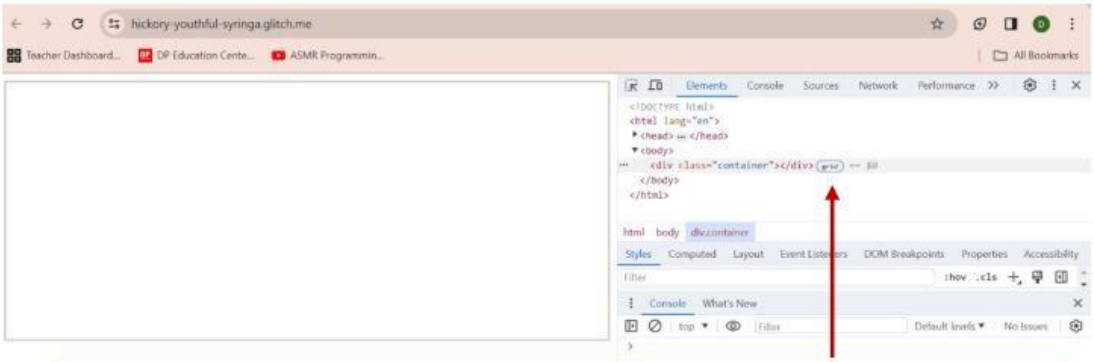
style.css ✨ PRETTIER

```css
.container {
    display: grid;
    /* 3 x 2 */
    grid-template-rows: 100px 100px 100px;
    grid-template-columns: 100px 100px;
    border: 3px solid lightgray;
}
```

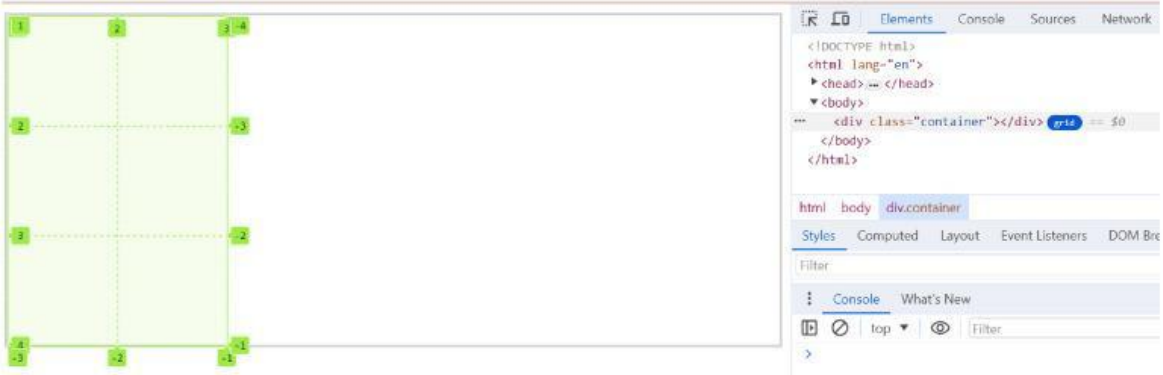
This property is used to create 3 rows and the size for each row is given as follows.

This property is used to create 2 Columns and the size for each Column is given as follows.

❖ preview now



Right click on the webpage and click on inspect. Then the elements can be taken care of as above. Now click on the grid here. Then you can view the created grid as follows.



❖ The above css can be given as shorthand as below.

```
.container {
    display: grid;
    /* 3 x 2 */
    grid-template-rows: repeat(3, 100px);
    grid-template-columns: repeat(2, 100px);
    border: 3px solid lightgray;
}
```

It can be given as above using the repeat keyword. Here, the number of rows or columns are given in the first signs in the brackets and its size is given afterwards.

❖ Now code as below.

```html
<div class="container">
    <div class="box">A</div>
    <div class="box">B</div>
    <div class="box">C</div>
    <div class="box">D</div>
</div>
```

```css
.container {
    display: grid;
    /* 3 x 2 */
    grid-template: repeat(3, 100px) / repeat(2, 100px);
    border: 3px solid lightgray;
}
.box {
    width: 5rem;
    height: 5rem;
    background-color: gold;
}
```

This is also a short method to create a grid using the repeat keyword. Here, the part before the / is for the rows of the grid and the part after the / is for the columns.

❖ Each box created is in each column.

❖ Although this view is suitable for desktop, it may not be suitable for mobile view. Therefore, we have to move this box as needed.

❖ The following properties are used for that.

- justify-items (along the **horizontal** axis)
- align-items (along the **vertical** axis)

❖ Now code as below.

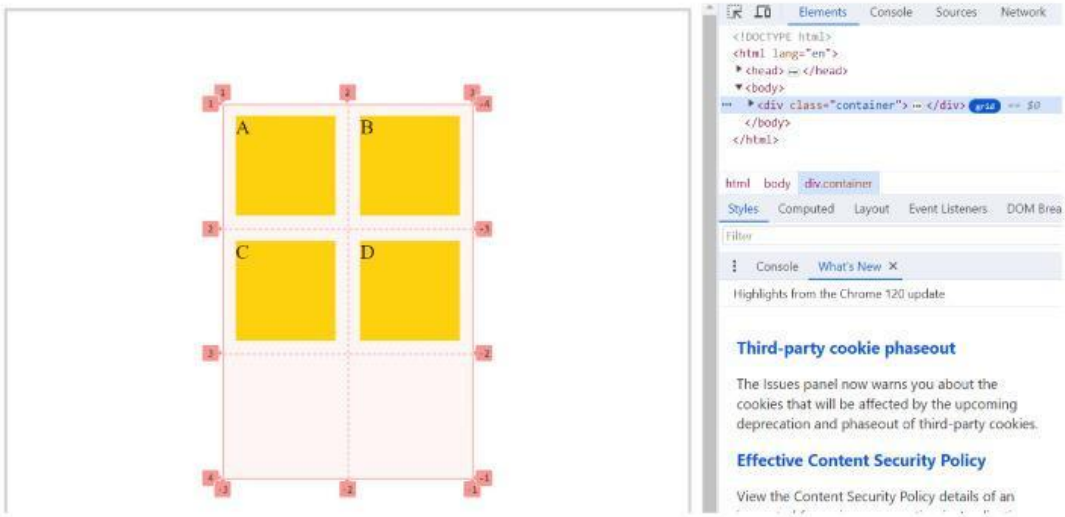
```
.container {
    display: grid;
    /* 3 x 2 */
    grid-template: repeat(3, 100px) / repeat(2, 100px);
    border: 3px solid lightgray;
    justify-items: center;
    align-items: center;
    justify-content: center;
    align-content: center;
    height: 100vh;
}
```

This will center the items in the grid relative to the horizontal line of the grid.

This will center the items in the grid relative to the vertical line of the grid.

This will center the grid relative to the horizontal line of the container.

This will center the grid relative to the vertical line of the container.



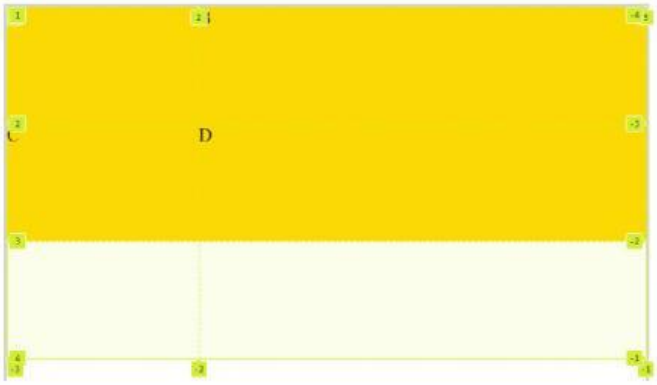
Here, items will style the items in the grid, and content will style the grid.

❖ The size of rows or columns can also be given by the percentage.

❖ Now code as below.

style.css PRETTIER

```css
.container {
    display: grid;
    /* 3 x 2 */
    grid-template: repeat(3, 100px) / 30% 70%;
    border: 3px solid lightgray;
    height: 100vh;
}
.box {
    background-color: gold;
}
```



❖ Now let's give one column with a fixed value for the columns and the percentage of the other 2 columns.

```css
.container {
    display: grid;
    /* 3 x 2 */
    grid-template: repeat(3, 100px) / 25px 30% 70%;
    border: 3px solid lightgray;
    height: 100vh;
}
.box {
    background-color: gold;
}
```

Then a 25px fixed column and the remaining width of the 2 columns given by the percentage will not be adjusted and the total width will be the same.

❖ Otherwise, to use only the remaining width, values can be given as follows.

style.css PRETTIER

```css
.container {
    display: grid;
    /* 3 x 2 */
    grid-template: repeat(3, 100px) / 25px 30fr 70fr;
    border: 3px solid lightgray;
    height: 100vh;
}
.box {
    background-color: gold;
}
```

❖ The following properties are used to create a space between the columns and rows of the grid.

- row-gap :- This leaves a gap between the rows.
- column-gap :- This leaves a gap between the columns.
- gap :- This leaves a gap between rows and columns.

❖ The following properties are used to change the location of the items in the grid.

- grid-row
- grid-column
- grid-area

❖ Now code as below.

When the value of the gap between columns and rows is equal, the value can be given in the following way. Otherwise, the value of the gap between the rows and the value of the gap between the columns should be given first.

Example:

```
gap: 20px 10px;
```

```
<div class="container">
    <div class="box box-one">A</div>
    <div class="box">B</div>
    <div class="box">C</div>
    <div class="box">D</div>
</div>
```

```
.container {
    display: grid;
    /* 3 x 2 */
    grid-template: 100px auto 100px / 30fr 70fr;
    gap: 10px;
    border: 3px solid lightgray;
    height: 100vh;
}
.box {
    background-color: gold;
}

.box-one {
  grid-column: 2;
}
```

The class is box-one and the box is given for the second column of the grid.

❖ Now code as below.

```html
<div class="container">
    <div class="box box-one">A</div>
    <div class="box">B</div>
    <div class="box">C</div>
    <div class="box box-four">D</div>
</div>
```

style.css ✦ PRETTIER

```css
.container {
    display: grid;
    /* 3 x 2 */
    grid-template: 100px auto 100px / 30fr 70fr;
    grid-template-areas:
      "header  header"
      "sidebar main"
      ".       footer";
    gap: 20px 10px;
    border: 3px solid lightgray;
    height: 100vh;
}
.box {
    background-color: gold;
}

.box-one {
  grid-area: header;
}

.box-four{
  grid-area: footer;
}
```

This property is used to set the position of the items in the grid.

The lines below show each row and the words in that row show each column.

The Class for the first and second columns is given as header to get the box which is box-one.

A sidebar is provided to get the box for the third column. After that it is given as main to get a box with a gap.

The fifth column is for the header. kept

Then it is given as footer for the sixth column.