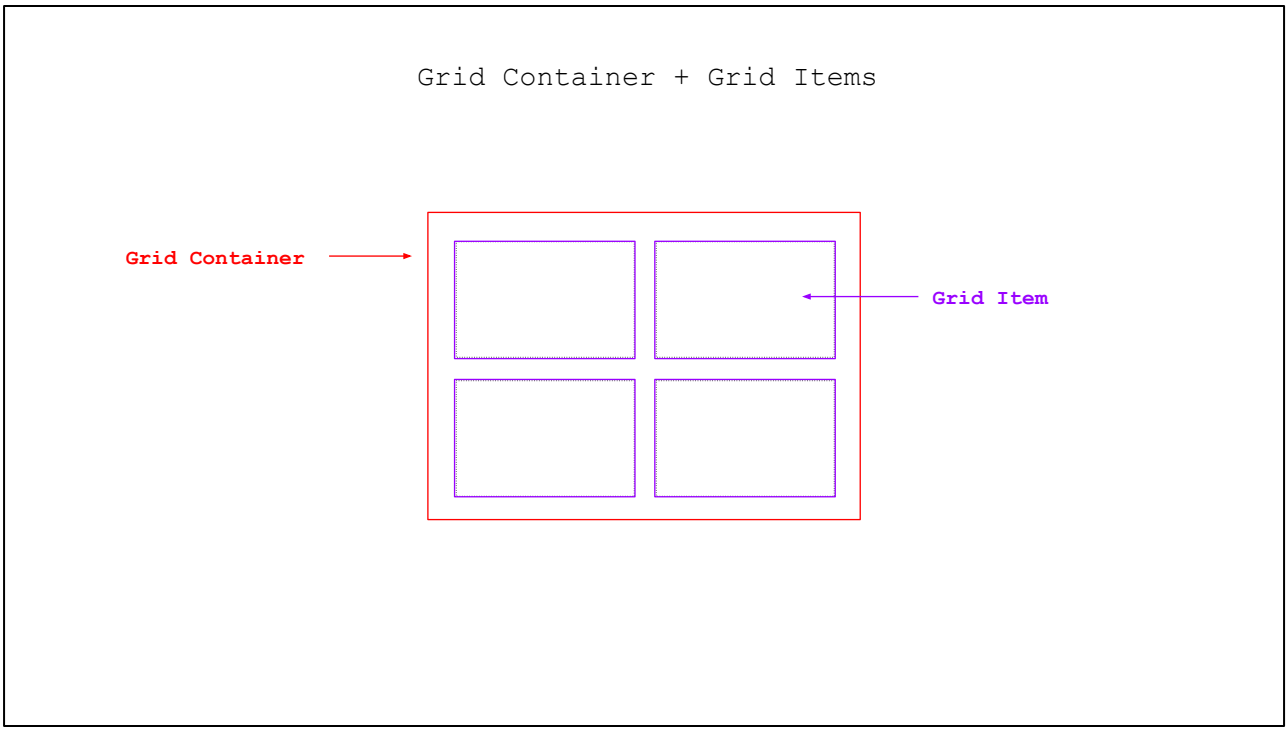
CSS Grid

Interactive 1

```
display: grid;
```

One of the more recent developments in CSS and HTML is the introduction of CSS grid. This enables you to easily change an html element into a grid.



To set up a grid, you need to have a ‘grid container’ and ‘grid items’ inside of it.

The ‘grid container’ will be a parent html element that contains ‘grid items’ as html children. In this scenario the parent is determining the grid structure those items fit into

html file

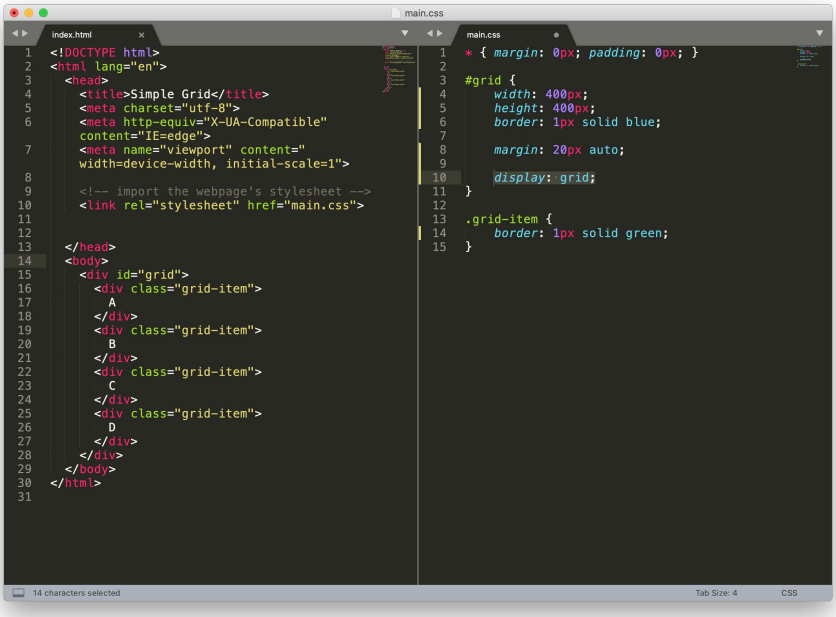
```
<div id="grid">
    <div class="grid-item">
        This is a grid cell
    </div>
    ...
</div>
```

css file

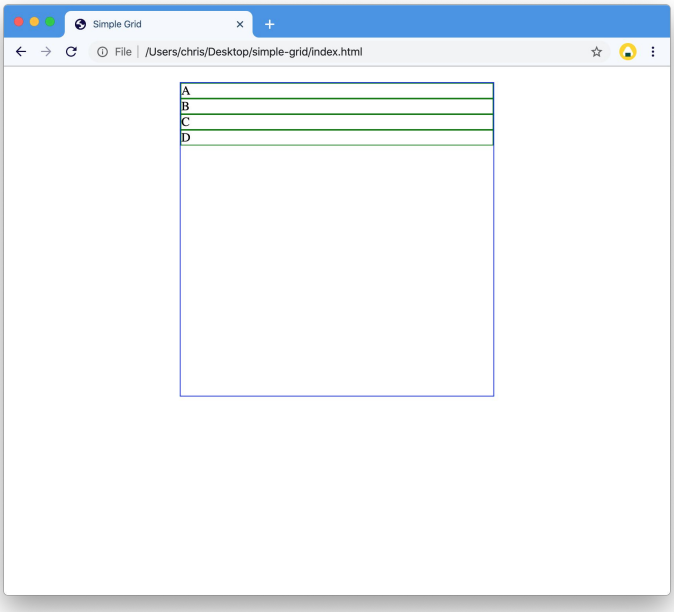
```
#grid {
    display: grid;
}
```

To set up a grid, you'd change the 'display' of the parent element to 'display: grid'.

By default the first level of children elements become cells in the grid



If you want to look at extended code samples, i'll place them in the slideshow



However, unless you define the columns and rows of the grid, simply changing an element to ‘display: grid’ won’t organize items as a grid

```
grid-template-columns: 100px 100px;
```

to do this you'll define a 'grid-template-columns' value for the grid parent container.

```
grid-template-columns: 100px 100px;
```

The number of columns are defined by the number of values passed to this property, each separated by a space.

In this scenario, i'd have 2 columns in my grid, each 100px wide

```
grid-template-columns: 100px 100px 100px;
```

If i added another value, separated by a space, i’d then have 3 columns

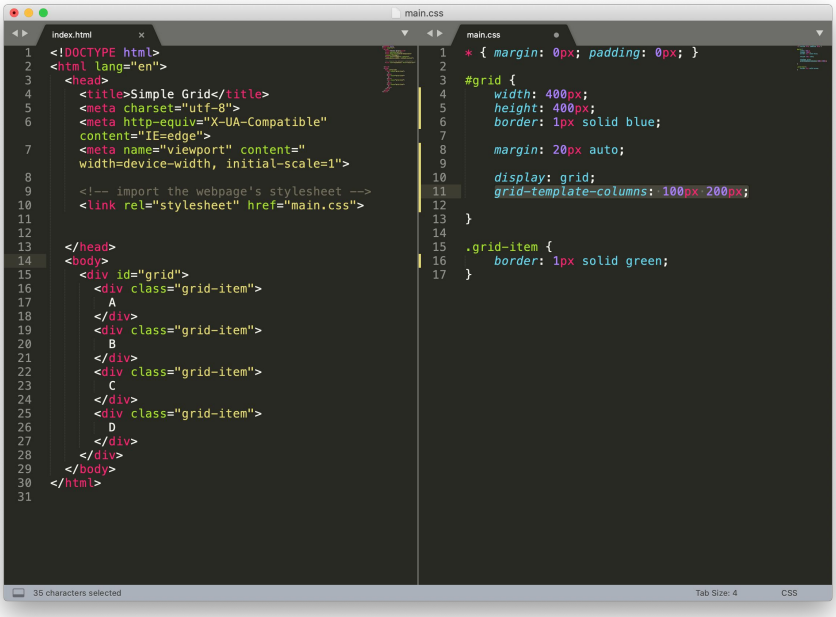
html file

```
<div id="grid>
    <div class="grid-item">
        This is a grid cell
    </div>
    ...
</div>
```

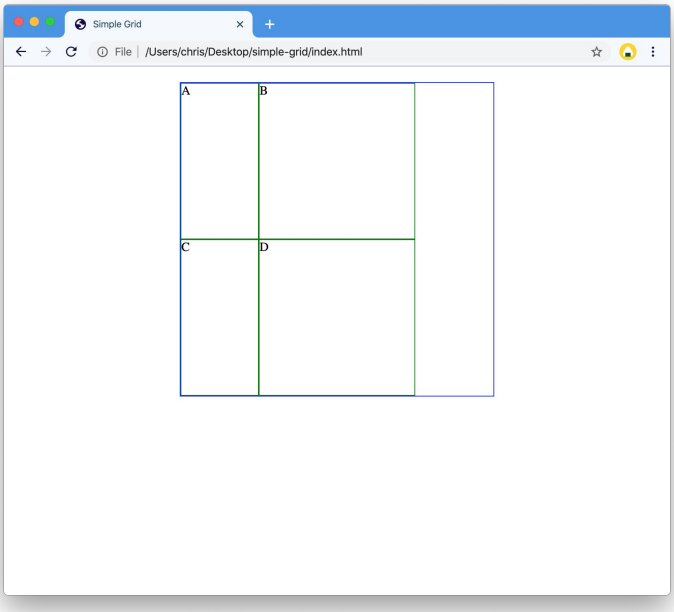
css file

```
#grid {
    display: grid;
    grid-template-columns: 100px 200px;
}
```

In this example, the first column would be 100px and the second 200px, so the values don't need to be consistent in your grid



Here’s the extended code i’m using in my example



And how it renders in code

```
grid-template-rows: 50px 100px 50px;
```

To add rows, it follows the same logic, just using the grid-template-rows property instead

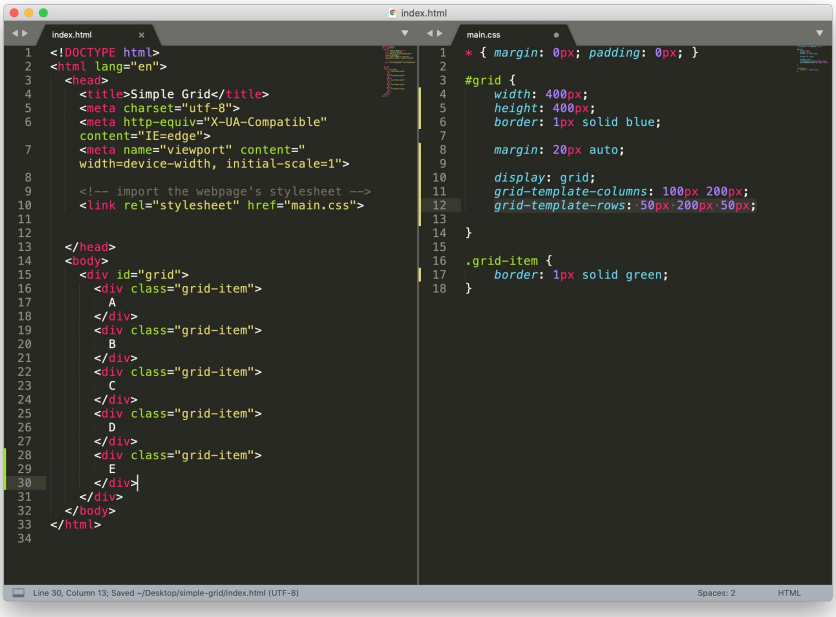
html file

```
<div id=”grid>
    <div class=”grid-item”>
        This is a grid cell
    </div>
    ...
</div>
```

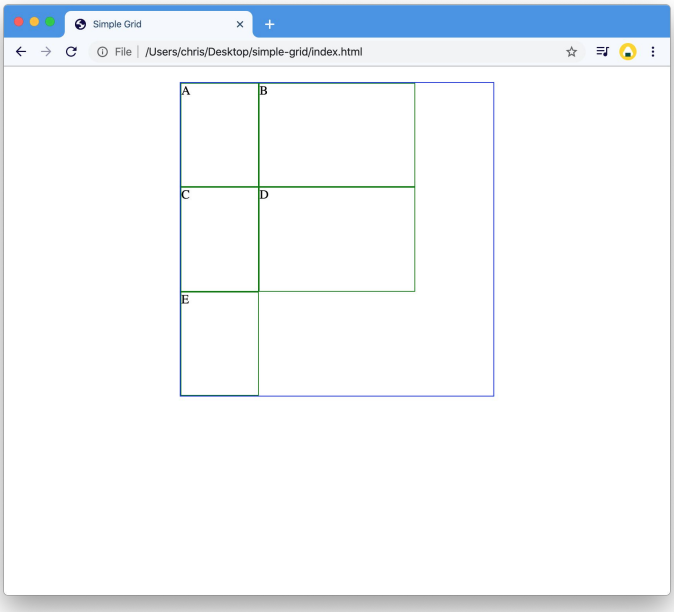
css file

```
#grid {
    display: grid;
    grid-template-columns: 100px 200px;
    grid-template-rows: 50px 100px 50px;
}
```

In this example, the first row would be 50px and the second 100px, and the third 50px



In my code example i’ve added more html elements since i want to show additional rows


Simple Grid
File
/Users/chris/Desktop/simple-grid/index.html
A
B
C
D
E

```
grid-template-columns: 1fr 1fr 1fr;
grid-template-rows: 1fr 2fr 1fr;
```

In addition to being able to set columns and rows with pixel or percentile values, you can abstract the units of measurements to fractions.

To do this you use the fr unit of measurement

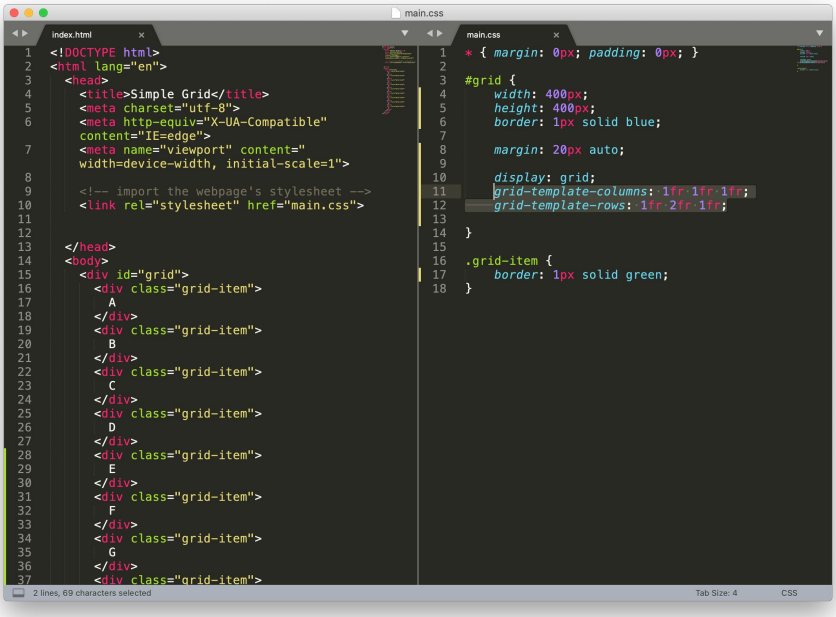
html file

```
<div id="grid>
    <div class="grid-item">
        This is a grid cell
    </div>
    ...
</div>
```

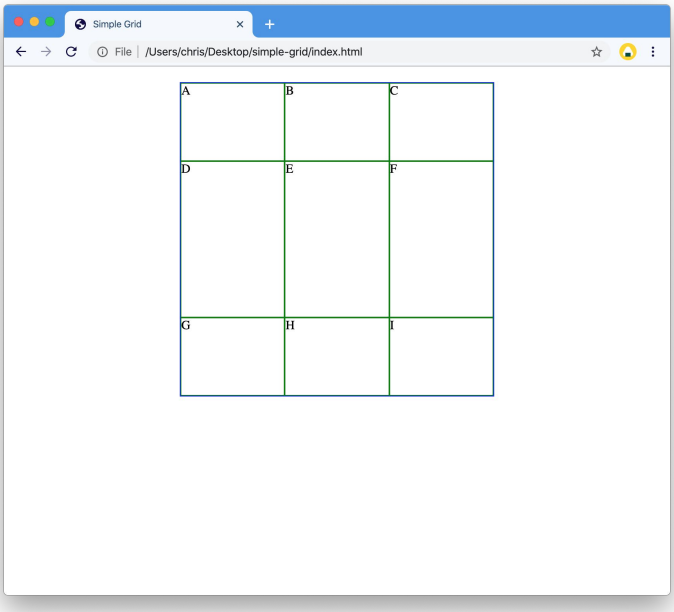
css file

```
#grid {
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
    grid-template-rows: 1fr 2fr 1fr;
}
```

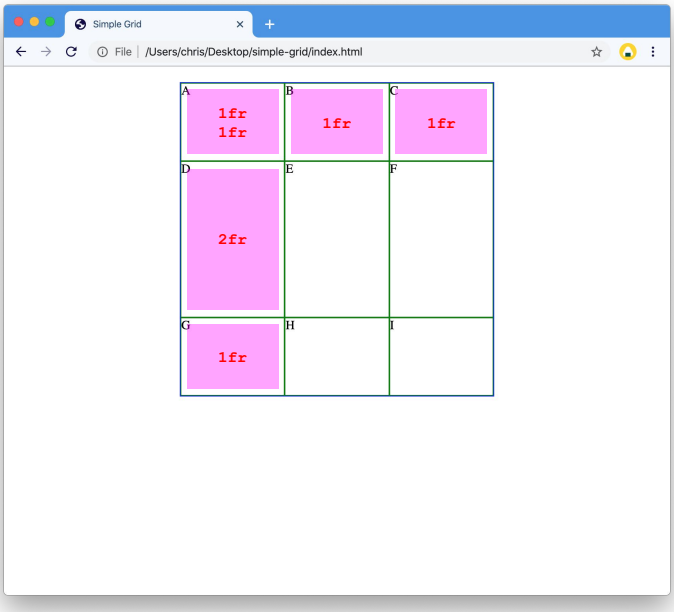
In this example, the 3 columns would be the same width, and the second row would be 2x in height to the other rows



Now with this code



The grid items fills the entire space of the parent container based on the fractions each row and column have been assigned



Just to reiterate, the rows and columns are each being defined in fractions, and could be visualized in this fashion

```
grid-gap: 10px;
```

The last thing i'll mention, is how to add spaces between grid items. To do this is quite simple, and the spaces are added by the grid-gap property

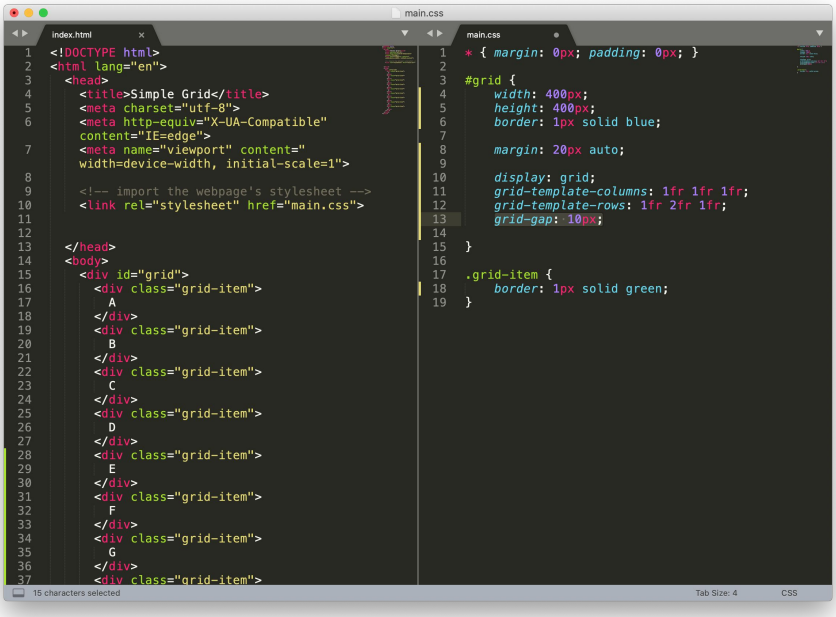
html file

```
<div id="grid>
    <div class="grid-item">
        This is a grid cell
    </div>
    ...
</div>
```

css file

```
#grid {
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
    grid-template-rows: 1fr 2fr 1fr;
    grid-gap: 10px;
}
```

In this example we're adding a space of 10px between all grid cells


main.css
index.html
<!DOCTYPE html>
<html lang="en">
<head>
<title>Simple Grid</title>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible"
content="IE=edge">
<meta name="viewport" content="
width=device-width, initial-scale=1">
<!-- import the webpage's stylesheet -->
<link rel="stylesheet" href="main.css">
</head>
<body>

A

B

C

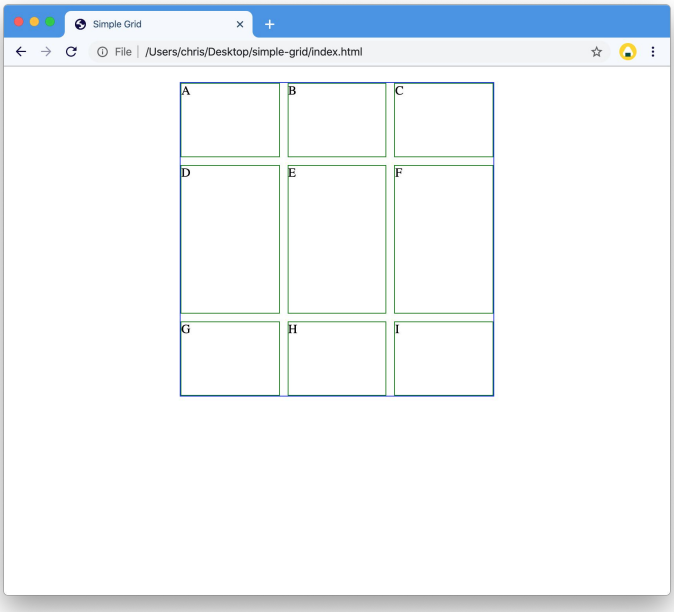
D

E

F

G

main.css
* { margin: 0px; padding: 0px; }
#grid {
width: 400px;
height: 400px;
border: 1px solid blue;
margin: 20px auto;
display: grid;
grid-template-columns: 1fr 1fr 1fr;
grid-template-rows: 1fr 2fr 1fr;
grid-gap: 10px;
}
.grid-item {
border: 1px solid green;
}
15 characters selected
Tab Size: 4
CSS


Simple Grid
File
/Users/chris/Desktop/simple-grid/index.html
A
B
C
D
E
F
G
H
I

Recommended Tutorial:

CSS Grid on Codecademy

These are the very basics of making a css grid, but there are many other things you can explore. For instance manipulating the grid items to cover multiple grid cells, where those cells start in the grid, or making the grid more dynamic. To get more into this, i'd strongly recommend doing the CSS Grid tutorial on codecademy

Using one of the sample grid letters code:

- translate one of your designs into html+css

Reference:

CSS Grid on Codecademy
https://www.codecademy.com/courses/learn-css/lessons/css-grid-i/exercises/grid-intro

CSS Grid Layout, W3C
https://www.w3schools.com/css/css_grid.asp

A Complete Guide to Grid, CSS Tricks
https://css-tricks.com/snippets/css/complete-guide-grid/