# CSS MEDIA QUERIES

I

# A Dao of Web Design

https://alistapart.com/article/dao/

## Responsive Design

“The control which designers know in the print medium, and often desire in the web medium, is simply a function of the limitation of the printed page. We should embrace the fact that the web doesn’t have the same constraints, and design for this flexibility. But first, we must ‘accept the ebb and flow of things.’”

—John Allsopp, “A Dao of Web Design”

# Media Queries

## Responsive Design

Features you can include in a media query include :
width, height, device-width, device-height, orientation, aspect-ratio, device-aspect-ratio, color, color-index, monochrome, resolution, scan grid

Most of the above can be combined with min- and max- prefixes.

The most common media queries assess min-width and max-width.

Media queries can be used to load an alternate style sheet or, more commonly, to offer alternate styles within an existing style sheet.

# Media Query Syntax

CSS media queries use the @media rule followed by two optional values: “only” or “not”

“only” screens out older browsers from reading the rest of the query.

“not” negates the result: “not screen“ means everything except screen-based media.

A feature: value pair, enclosed by parentheses, comprises the essence of the media query.

Media features that can be assessed are predefined.

Multiple feature: value pairs can be combined with “and“.

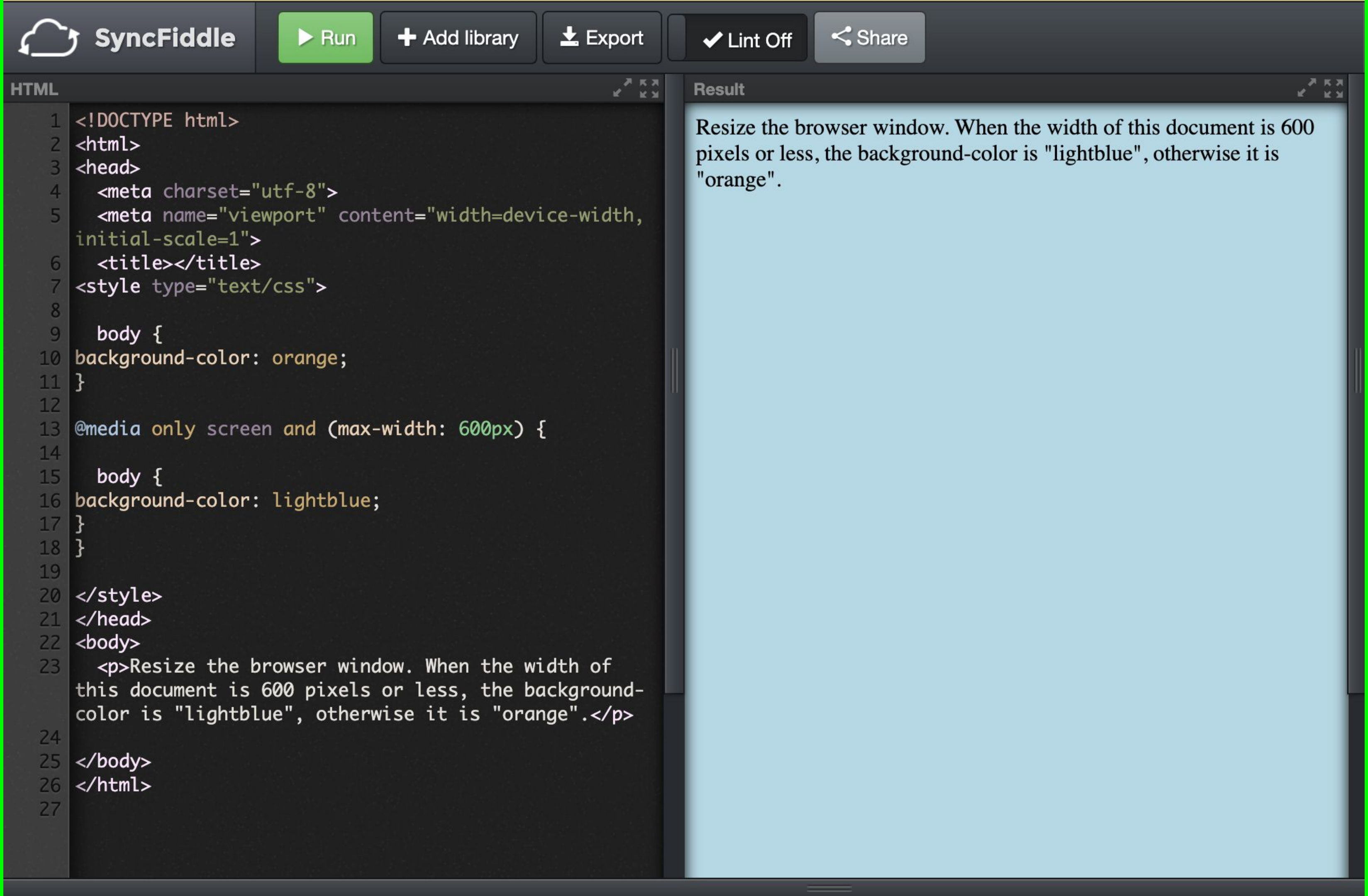
# Media Query Syntax

## CSS Rule Set

```
body {
        background-color: orange;
}
```

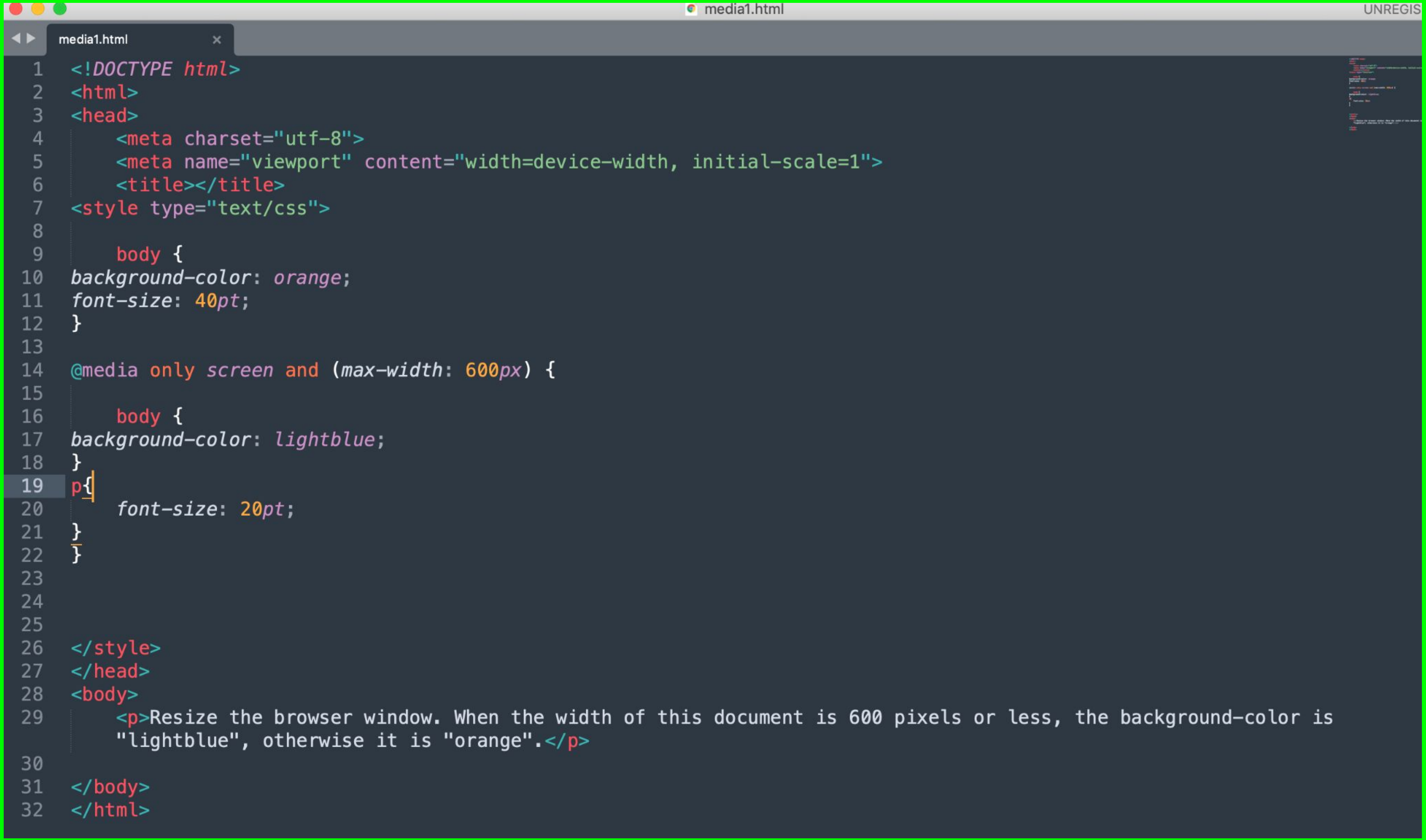
## CSS Rule Set with a Media Query

```
@media only screen and
        (min-width: 480px) {  body {
                background-color:
                orange;
        }
}
```



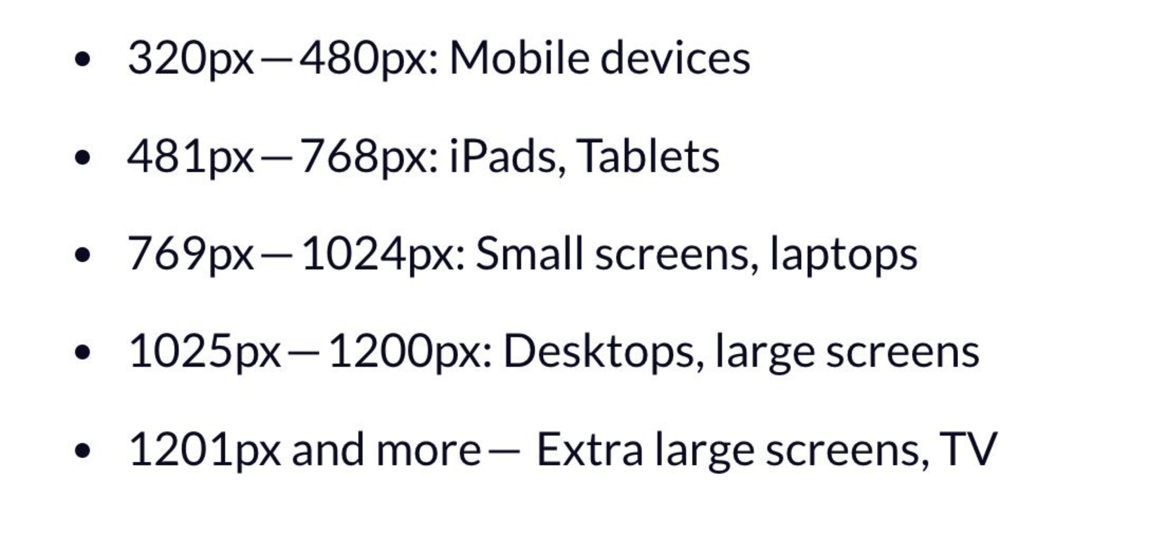
```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title></title>
<style type="text/css">

  body {
background-color: orange;
}

@media only screen and (max-width: 600px) {

  body {
background-color: lightblue;
}
}

</style>
</head>
<body>
  <p>Resize the browser window. When the width of this document is 600 pixels or less, the background-color is "lightblue", otherwise it is "orange".</p>

</body>
</html>
```

Resize the browser window. When the width of this document is 600 pixels or less, the background-color is "lightblue", otherwise it is "orange".



```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title></title>
<style type="text/css">

    body {
background-color: orange;
font-size: 40pt;
}

@media only screen and (max-width: 600px) {

    body {
background-color: lightblue;
}
p{
    font-size: 20pt;
}
}



</style>
</head>
<body>
    <p>Resize the browser window. When the width of this document is 600 pixels or less, the background-color is "lightblue", otherwise it is "orange".</p>

</body>
</html>
```

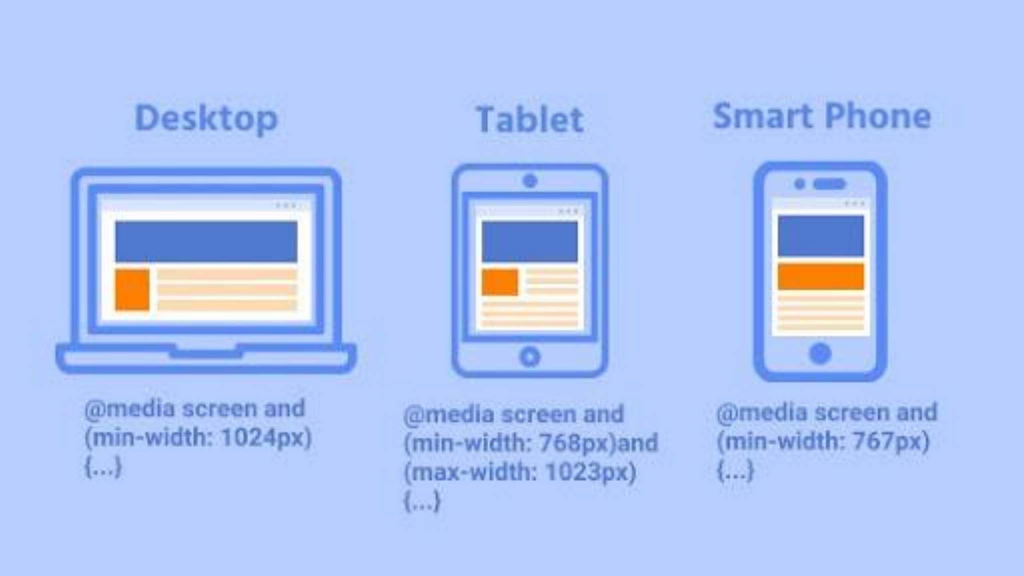
# Common device breakpoints



- 320px—480px: Mobile devices
- 481px—768px: iPads, Tablets
- 769px—1024px: Small screens, laptops
- 1025px—1200px: Desktops, large screens
- 1201px and more— Extra large screens, TV

- There is no standard list of breakpoints. There are a ton of devices on the market (w/new devices released every year) so we can't and we shouldn't define fixed breakpoints for each of them.

Some say it is better to define your breakpoints with your individual content in mind . . . rather than design for specific devices that may change.



```
@media screen and
(min-width: 1024px)
{...}
```

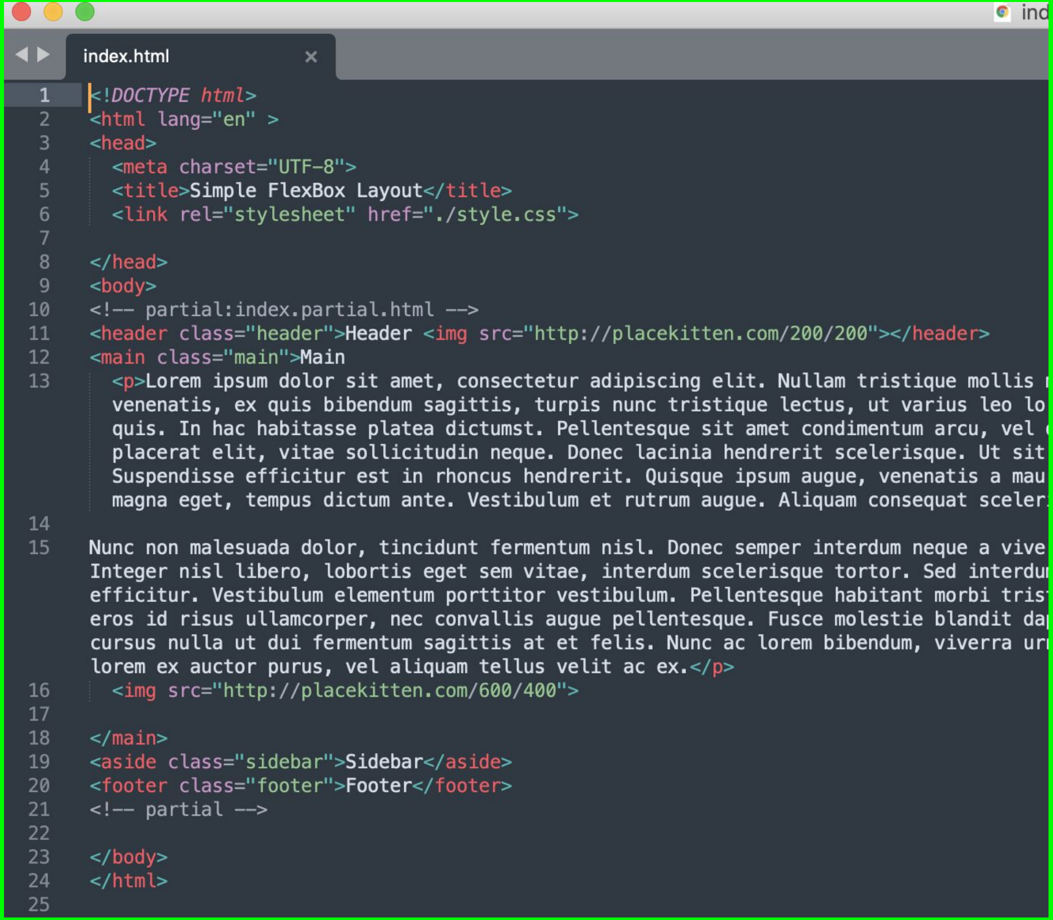
```
@media screen and
(min-width: 768px)and
(max-width: 1023px)
{...}
```

```
@media screen and
(min-width: 767px)
{...}
```

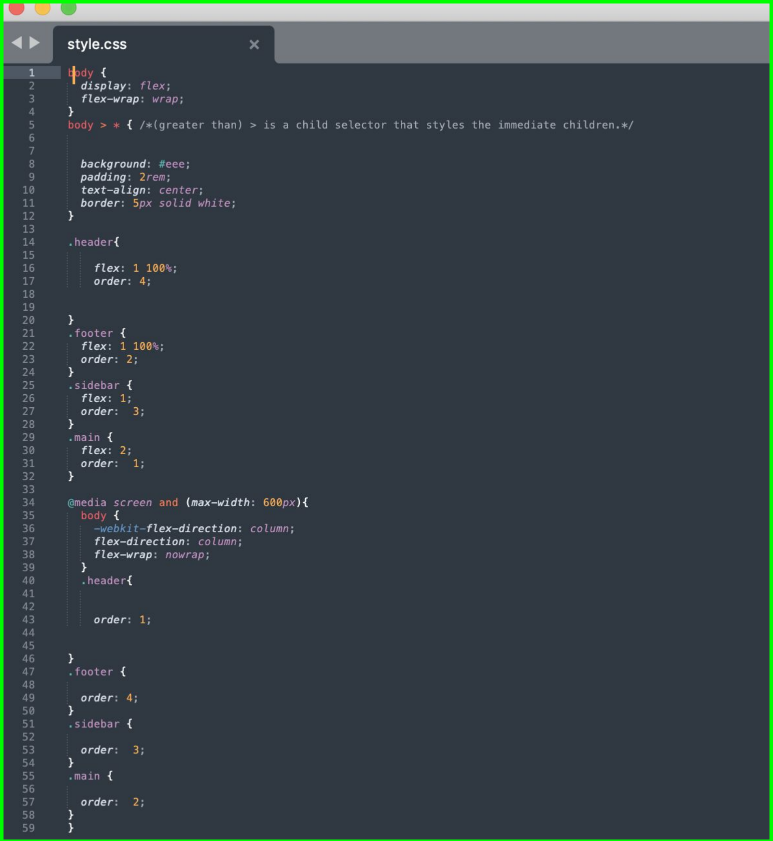
One of the utilities of libraries like Bootstrap is the breakpoints are built into the template and kept current: https://getbootstrap.com/docs/5.0/layout/breakpoints/

# LAYOUT APPLICATIONS

A simple flexbox example:  https://i6.cims.nyu.edu/~mr6465/media_queries/flex-media/

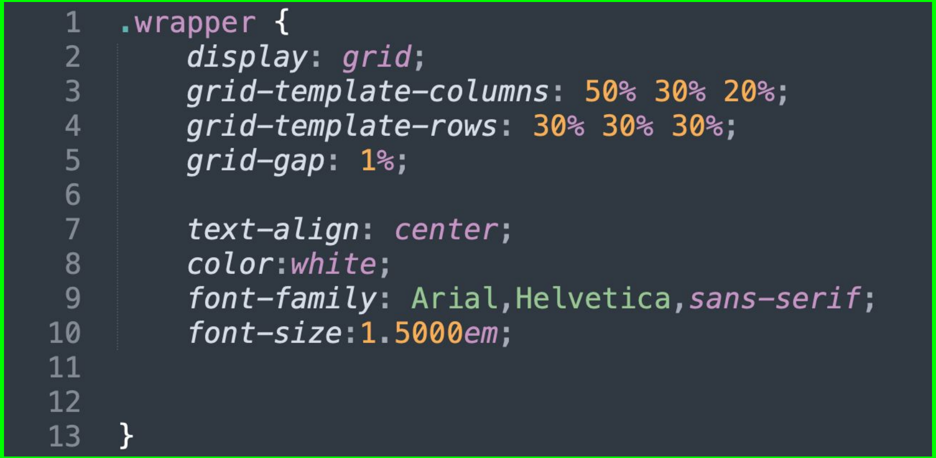


```html
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>Simple FlexBox Layout</title>
  <link rel="stylesheet" href="./style.css">

</head>
<body>
<!-- partial:index.partial.html -->
<header class="header">Header <img src="http://placekitten.com/200/200"></header>
<main class="main">Main
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam tristique mollis r
  venenatis, ex quis bibendum sagittis, turpis nunc tristique lectus, ut varius leo lo
  quis. In hac habitasse platea dictumst. Pellentesque sit amet condimentum arcu, vel
  placerat elit, vitae sollicitudin neque. Donec lacinia hendrerit scelerisque. Ut sit
  Suspendisse efficitur est in rhoncus hendrerit. Quisque ipsum augue, venenatis a mau
  magna eget, tempus dictum ante. Vestibulum et rutrum augue. Aliquam consequat sceler

Nunc non malesuada dolor, tincidunt fermentum nisl. Donec semper interdum neque a vive
Integer nisl libero, lobortis eget sem vitae, interdum scelerisque tortor. Sed interdu
efficitur. Vestibulum elementum porttitor vestibulum. Pellentesque habitant morbi tris
eros id risus ullamcorper, nec convallis augue pellentesque. Fusce molestie blandit da
cursus nulla ut dui fermentum sagittis at et felis. Nunc ac lorem bibendum, viverra ur
lorem ex auctor purus, vel aliquam tellus velit ac ex.</p>
  <img src="http://placekitten.com/600/400">

</main>
<aside class="sidebar">Sidebar</aside>
<footer class="footer">Footer</footer>
<!-- partial -->

</body>
</html>

```

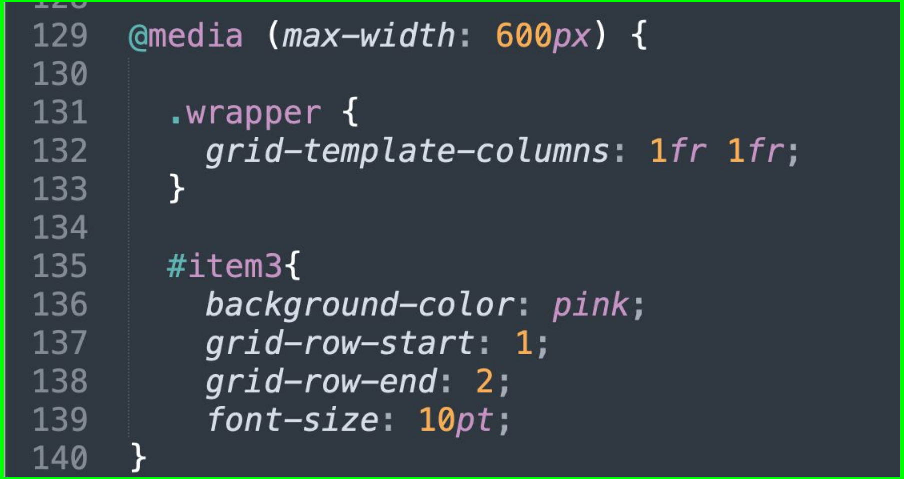


```css
body {
  display: flex;
  flex-wrap: wrap;
}
body > * { /*(greater than) > is a child selector that styles the immediate children.*/


  background: #eee;
  padding: 2rem;
  text-align: center;
  border: 5px solid white;
}

.header{

    flex: 1 100%;
    order: 4;


}
.footer {
  flex: 1 100%;
  order: 2;
}
.sidebar {
  flex: 1;
  order:  3;
}
.main {
  flex: 2;
  order:  1;
}

@media screen and (max-width: 600px){
  body {
    -webkit-flex-direction: column;
    flex-direction: column;
    flex-wrap: nowrap;
  }
  .header{


    order: 1;


}
.footer {

  order: 4;
}
.sidebar {

  order:  3;
}
.main {

  order:  2;
}
}
```

# USING BREAKPOINTS WITH GRID ENABLES RESPONSIVE MULTI-COLUMN LAYOUTS



```css
.wrapper {
    display: grid;
    grid-template-columns: 50% 30% 20%;
    grid-template-rows: 30% 30% 30%;
    grid-gap: 1%;

    text-align: center;
    color:white;
    font-family: Arial,Helvetica,sans-serif;
    font-size:1.5000em;


}
```



```css
@media (max-width: 600px) {

  .wrapper {
    grid-template-columns: 1fr 1fr;
  }

  #item3{
    background-color: pink;
    grid-row-start: 1;
    grid-row-end: 2;
    font-size: 10pt;
}
```

Default liquid layout with 3x3 grid

Smaller screen width: switch to 2 column grid!

# Examples:

- https://i6.cims.nyu.edu/~mr6465/media_queries/kpop/index_breakpoints.html
- https://i6.cims.nyu.edu/~mr6465/web_design/responsive/seasons/
- https://i6.cims.nyu.edu/~mr6465/media_queries/grid/