Web Programming

Dr Hossein Nevisi

Department of Computer Science
Loughborough University

CSS (Cascading Style Sheet)

CSS is used to format the layout of a webpage

What is CSS?

- CSS is used to format the **layout of a webpage**
- For example, to alter the font, colour, background images, size, and spacing of your content, split it into multiple columns, or add animations and other decorative features
- CSS can be added to HTML documents in 3 ways:

1. Inline - by using the style attribute inside HTML elements
2. Internal - by using a <style> element inside the HTML document (it is recommended to place it in the <head> element)
3. External - by using a <link> element to link to an external CSS file ✓Most common way

1. Inline CSS

- is used to apply a unique style to a single HTML element by using the **style** attribute of the element

Example

property: value;

https://www.w3schools.com/colors/colors_picker.asp
https://www.w3schools.com/colors/colors_names.asp

- <body **style**=" background-color: skyblue; ">

  <!-- background-color CSS property sets the background color of an element -->

- <h3 **style** ="color: #706897; font-family: Arial, Helvetica, sans-serif; ">

  <!--The color property specifies the color of text, and the font-family property specifies the font for an element -->

  https://www.w3schools.com/css/css_font_websafe.asp

- <img **style** ="border: 1px solid rgb(0,0,255); border-radius: 10px; padding: 5px; width:100px;" src= " " >

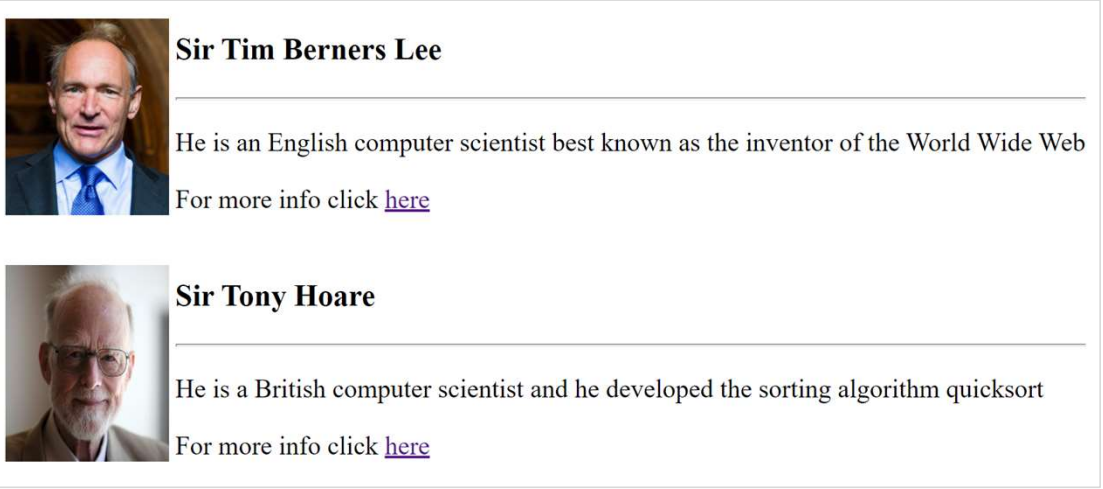
  <!--The CSS padding properties are used to generate space around an element's content, inside of any defined borders -->

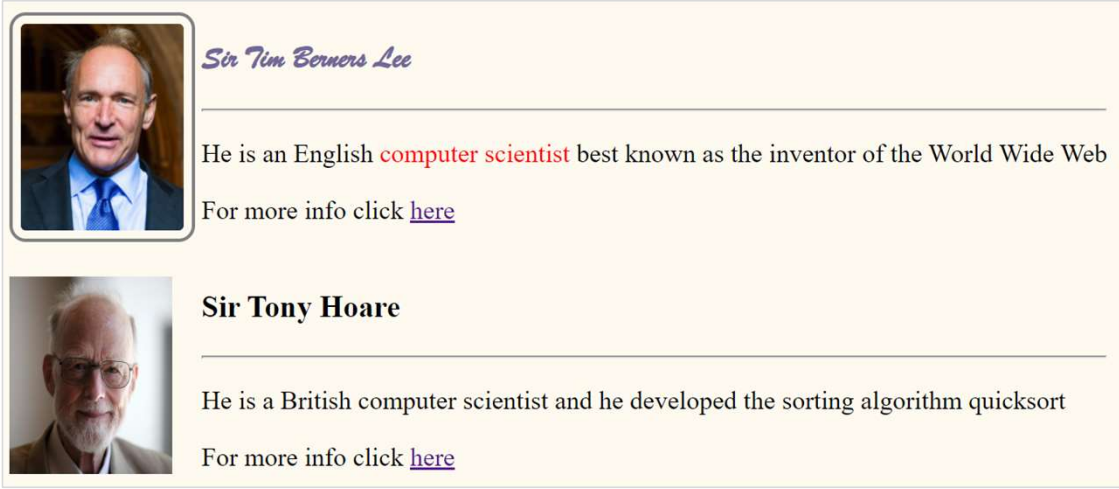
Example

```
<table>
   <tr>
    <td>
      <img src="images/Sir_Tim_Berners-Lee.jpg" alt="Sir Tim Berners Lee Picture" width="100" height="120">
    </td>
    <td>
      <h3>Sir Tim Berners Lee</h3>
      <hr>
      <p>He is an English <span>computer scientist</span> best known as the inventor of the World Wide Web </p>
      <p> For more info click <a href="https://en.wikipedia.org/wiki/Tim_Berners-Lee">here</a></p>
       </td>
   </tr>
   <tr>
    <td>
      <img src="images/Sir_Tony_Hoare.jpg" alt="Sir Tony Hoare Picture" width="100" height="120">
    </td>
    <td>
      <h3>Sir Tony Hoare</h3>
      <hr>
      <p>He is a British <span>computer scientist</span> and he developed the sorting algorithm quicksort </p>
      <p> For more info click <a href="https://en.wikipedia.org/wiki/Tony_Hoare">here</a></p>
    </td>
  </tr>
 </table>
```

An inline container mainly used to mark up a small part of a text

Example - Continued






```
<table>
<h3>Sir Tim Berners Lee</h3>
<span>computer scientist</span>
<img src="images/Sir_Tim_Berners-Lee.jpg" alt="Sir Tim Berners Lee Picture" width="100px" height="120">

<table style="background-color: FloralWhite;">
<h3 style ="color: #706897; font-family: Brush Script MT, Brush Script Std, cursive; ">
<span style ="color:red;">
<img style ="border: 2px solid gray; border-radius: 10px; padding: 5px; width:100px; "
src="images/Sir_Tim_Berners-Lee.jpg" alt="Sir Tim Berners Lee Picture" >
```

2. Internal CSS and CSS Selectors

- is used to define a style for a single HTML page
- is recommended to be defined in the <head> section of an HTML page, within a <style> element

```
<!DOCTYPE html>
<html lang ="en">
<head>
…
<style>
HTML-element {property: value;}   // p {text-align: center; color: red;}
.class-name {property: value;}    // .note {text-align: left; color: blue;}
#element-id {property: value;}    // #mainHeader {color: black;}
</style>
…
</head>

<body>
…
</body>
</html>
```

declaration block (contains one or more declarations separated by semicolons)

Selector

(The selector points to the HTML element/s you want to style)

CSS Selector	Example	Example description
element	p	Selects all <p> elements
.class	.intro	Selects **all** elements with class="intro"
#id	#firstname	Selects **the** element with id="firstname"
element.class	p.intro	Selects only <p> elements with class="intro"
*	*	Selects all elements
element,element,..	div, p	Selects all elements and all <p> elements

Example

```
<!DOCTYPE html>
<html lang ="en">
<head>
  <meta charset="utf-8">
  <title>Computer Scientists </title>
 <style>
   table {
       background-color: FloralWhite;
     }
   h3 {
       color: #706897;
       font-family: Brush Script MT, Brush Script Std, cursive;
     }
   img {
       border: 2px solid gray;
       border-radius: 10px;
       padding: 5px;
       width:100px;
     }
  </style>
  </head>
  <body>…</body>
  </html>
```

Sir Tim Berners Lee

He is an English computer scientist best known as the inventor of the World Wide Web

For more info click here

Sir Tony Hoare

He is a British computer scientist and he developed the sorting algorithm quicksort

For more info click here

Sir Tim Berners Lee

He is an English computer scientist best known as the inventor of the World Wide Web

For more info click here

Sir Tony Hoare

He is a British computer scientist and he developed the sorting algorithm quicksort

For more info click here

HTML id Attribute

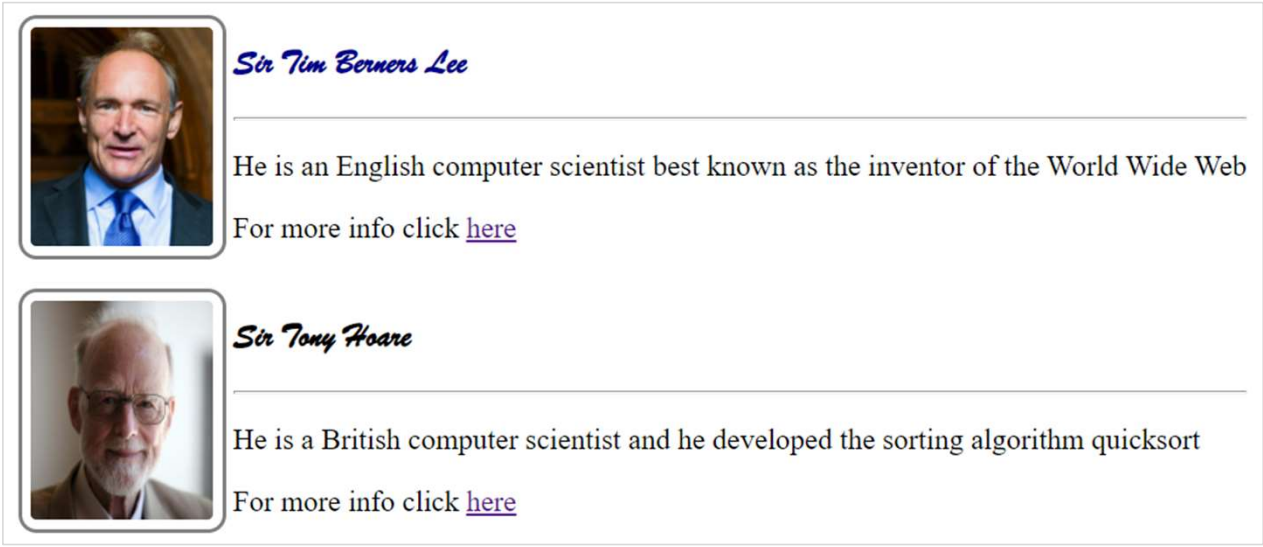
- The HTML id attribute is used to specify a unique id for an HTML element
- It is used to point to a specific style declaration in the CSS

Example

```
...
<h3>Sir Tim Berners Lee</h3>
...
<h3>Sir Tony Hoare</h3>
...
```

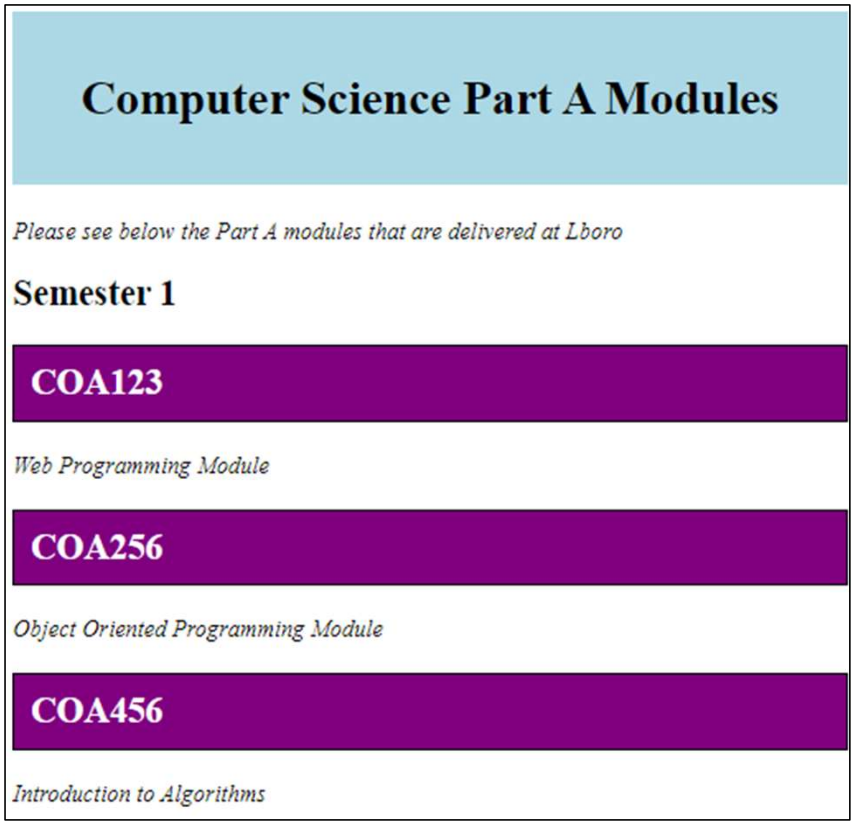
```
...
<h3 id= "first-heading">Sir Tim Berners Lee</h3>
...
<h3>Sir Tony Hoare</h3>
...
```

```
<style>
#first-heading  {
    color: darkblue;
  }
h3 {
    font-family: Brush Script MT, Brush Script Std, cursive;
  }
img {
    border: 2px solid gray;
    border-radius: 10px;
    padding: 5px;
    width:100px;
  }
</style>
```



HTML class Attribute

- The class attribute specifies one or more class names for an element
- In contrast to values of id attributes, a class name is not unique and can be used by multiple HTML elements
- It is used to point to a specific style declaration in the CSS




```
<style>
  #main-header {
    background-color: lightblue;
    padding: 40px;
    text-align: center;
  }
  .module {
    background-color: purple;
    border: 2px solid black;
    color: white;
    padding: 10px;
  }
   p {font-style: italic;}
</style>
```

```
<h1 id="main-header"> Computer Science Part A Modules</h1>
<p> Please see below the Part A modules that are delivered at Lboro</p>
<h2> Semester 1</h2>
<h2 class="module"> COA123 </h2>
<p>Web Programming Module</p>
<h2 class="module"> COA256 </h2>
<p>Object Oriented Programming Module</p>
<h2 class="module"> COA456 </h2>
<p>Introduction to Algorithms</p>
```

CSS Selectors

Self-Study

- In CSS, there are several selectors (patterns) used to select the element(s) you want to style

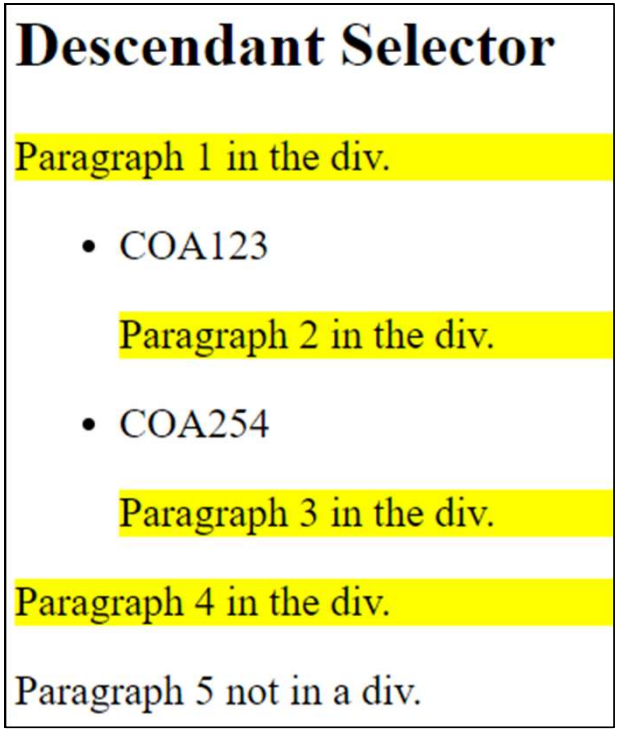
Selector	Example	Example description
element	P	Selects all <p> elements
.class	.intro	Selects all elements with class="intro"
#id	#firstname	Selects the element with id="firstname"
element.class	p.intro	Selects all <p> elements with class="intro"
element,element	div, p	Selects all elements and all <p> elements
element element	div p	Selects all <p> elements inside elements (all descendant p in the div)
element>element	div > p	Selects all <p> elements where the parent is a element
element+element	div + p	Selects the first <p> element that are placed immediately after elements
*	*	Selects all elements

- More information about CSS Selectors (full list):
  https://www.w3schools.com/cssref/css_selectors.asp

Example 1

Self-Study

```
<!DOCTYPE html>
<html>
<head>
<style>
div p {
  background-color: yellow;
}
</style>
</head>
<body>
<h2>Descendant Selector</h2>
<div>
  <p>Paragraph 1 in the div.</p>
  <ul>
   <li>COA123</li>
   <p> Paragraph 2 in the div. </p> <!- descendant of div -->
   <li>COA254</li>
   <p> Paragraph 3 in the div. </p> <!-- descendant of div -->
  </ul>
  <p>Paragraph 4 in the div.</p>
</div>
<p>Paragraph 5 not in a div.</p>
</body>
</html>
```

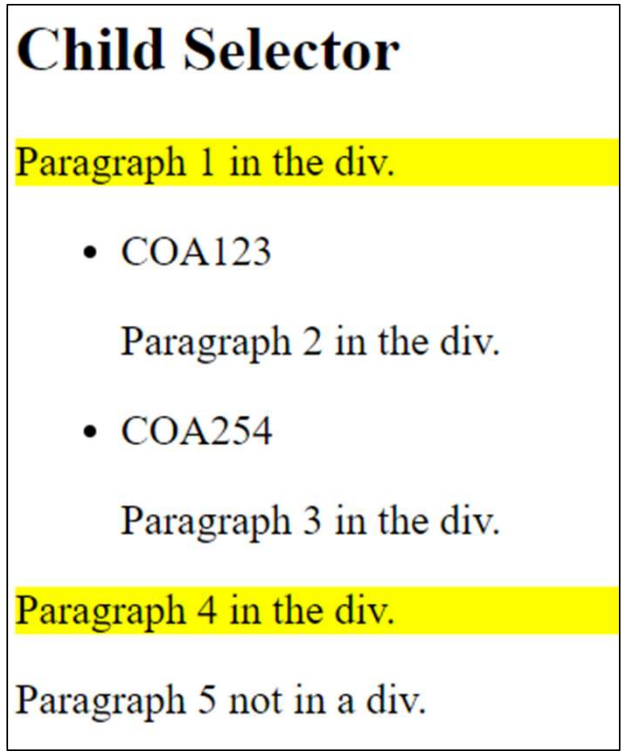


- The descendant selector matches all elements that are descendants (inside) of a specified element

Example 2

Self-Study

```
<!DOCTYPE html>
<html>
<head>
<style>
div > p {
  background-color: yellow;
}
</style>
</head>
<body>
<h2>Child Selector</h2>
<div>
  <p>Paragraph 1 in the div.</p>
  <ul>
     <li>COA123</li>
     <p> Paragraph 2 in the div. </p> <!-- descendant of div -->
      <li>COA254</li>
     <p> Paragraph 3 in the div. </p> <!-- descendant of div -->
  </ul>
  <p>Paragraph 4 in the div.</p>
<div>
<p>Paragraph 5 not in a div.</p>
</body>
</html>
```



- The child selector selects all elements that are the direct children of a specified element

Example 3

```
<html>
<head>
<style>
#header {
  background-image: url("images/background-header.jpg");
  background-size: cover;
  border: 5px solid black;
  text-align: center;
  color: white;
}

#header p {
  font-family: "Lucida Console", Courier, monospace;
  font-style: italic;
}
</style>
</head>

<body>
<div id="header">
  <h2>COA123</h2>
  <p>Computer Science - Web Programming Module</p>
</div>
<p>This is some text outside the div element.</p>
</body></html>
```

CSS Comments

```
<html>
<head>
<style>
#header {
  background-image: url("images/background-header.jpg");
  background-size: cover;
  border: 5px solid black;
  text-align: center;
  color: white;
}
/* Select all paragraph inside the element with id: header */
#header p {
  font-family: "Lucida Console", Courier, monospace;
  font-style: italic;
}
</style>
</head>
<body>
<!-- Header of the page -->
<div id="header">
  <h2>COA123</h2>
  <p>Computer Science - Web Programming Module</p>
</div>
<p>This is some text outside the div element.</p>
</body></html>
```

CSS comments

HTML comments

3. External CSS

- It enables us to change the look of an entire website by changing just one file (filename.css)
- Each HTML page must include a reference to the external style sheet file
- External styles are defined within the <**link**> element, inside the <head> section of an HTML page
- The <**link**> tag defines the relationship between the current document and an external resource

Example

```
<!DOCTYPE html>
<html lang ="en">
<head>
  <meta charset="utf-8">
  <title>Computer Science Modules</title>
  <link rel="stylesheet" href="mystyle.css">
</head>

<body>
...
</body>
</html>
```

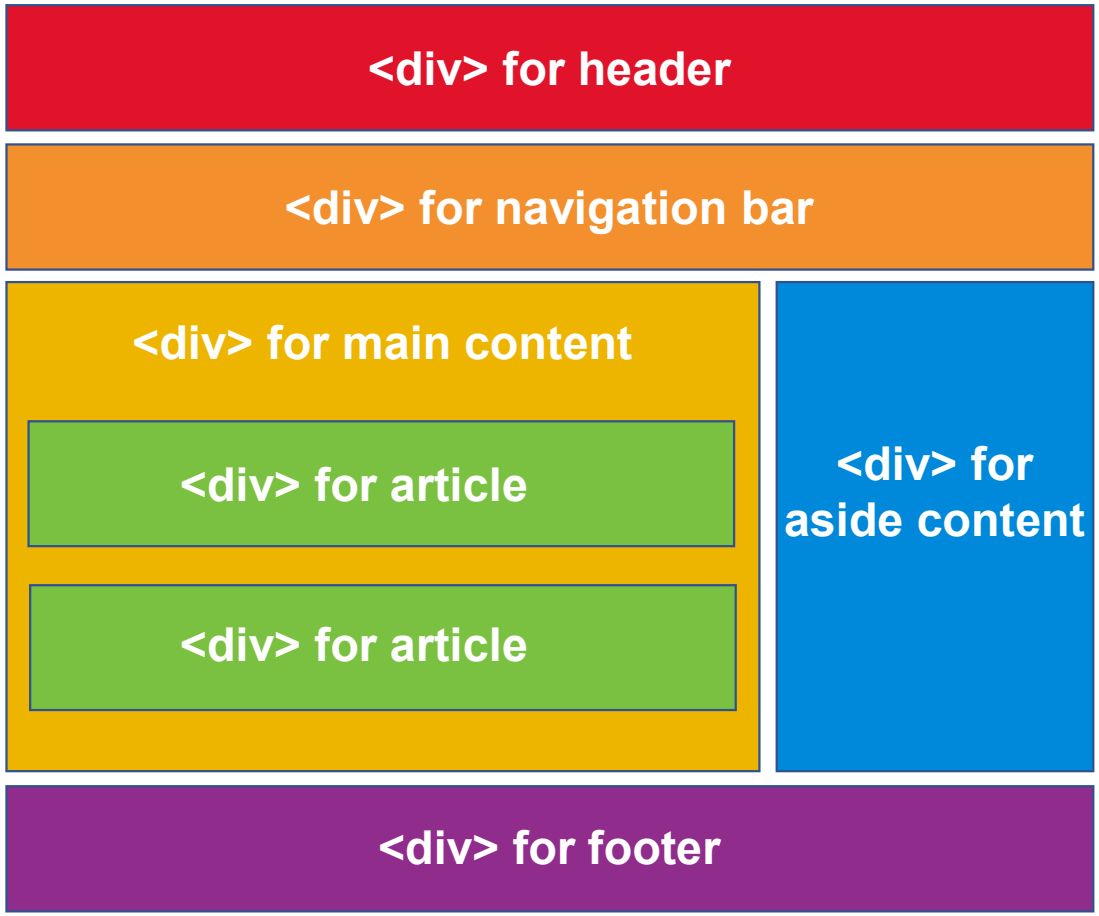
mystyle.css

```
#main-header {
  background-color: lightblue;
  padding: 30px;
  text-align: center;
}
.message {
  background-color: purple;
  text-align: center;
  border: 2px solid black;
  color: white;
  padding: 5px;
}
```

CSS Layout Methods

Using in Web Layouts

- We can use div as a container to structure a webpage into separate components for individual styling:

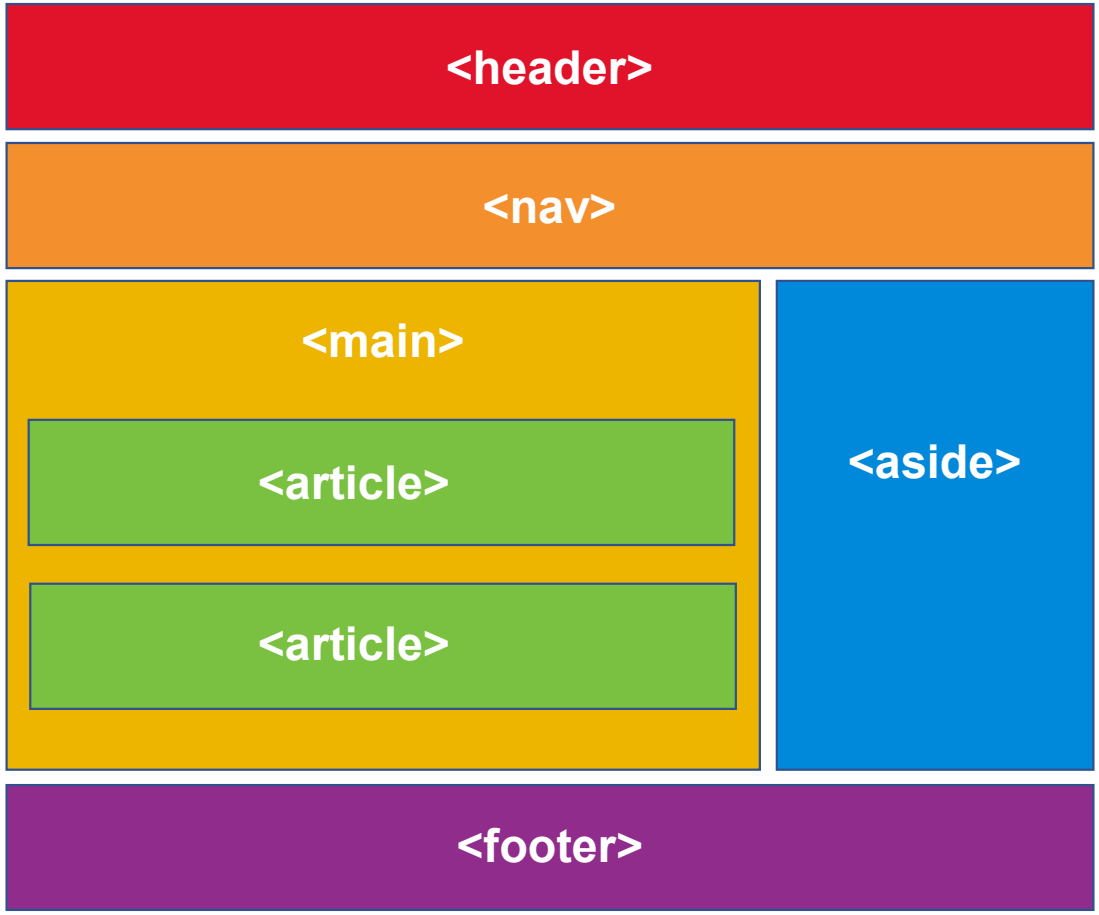


Semantic Elements in HTML5

- In HTML, there are some semantic elements that can be used to define different parts of a web page
- Many of them act more or less like (group other elements together)
- The semantic elements added in HTML5 are:

  - <article>
  - <aside>
  - <details>
  - <figcaption>
  - <figure>
  - <footer>
  - <header>
  - <main>
  - <mark>
  - <nav>
  - <section>
  - <summary>
  - <time>

Example:



Benefits of Using Semantic elements

- Advantages: clearly defines its content; easier to read; greater accessibility; more consistent code

non-semantic elements:

```
<div id="header"></div>
<div id="main">
        <div class="article">
                <div class="figure">
                        <img>
                        <div class="figcaption"></div>
                </div>
        </div>
</div>
<div id="footer"></div>
```

semantic elements

```
<header></header>
<main>
        <article>
                <figure>
                        <img>
                        <figcaption></figcaption>
                </figure>
        </article>
</main>
<footer></footer>
```

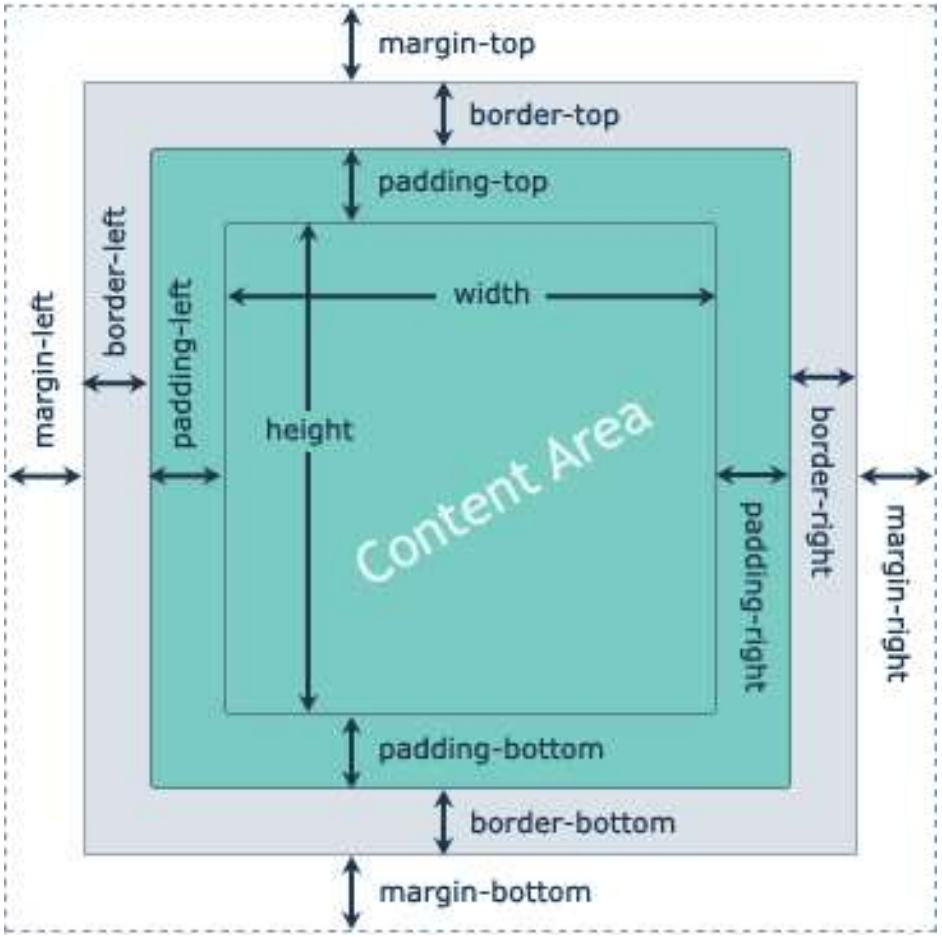
- More information: https://www.w3schools.com/html/html5_semantic_elements.asp

The CSS Box Model

- When we are talking about design and layout, we can assume that there is a box that wraps around every HTML element, and this box is called the **CSS box model**

- These boxes is key to being able to create layouts with CSS, or to align items with other items

- **CSS box model** is a box that can be assumed wraps around every HTML element



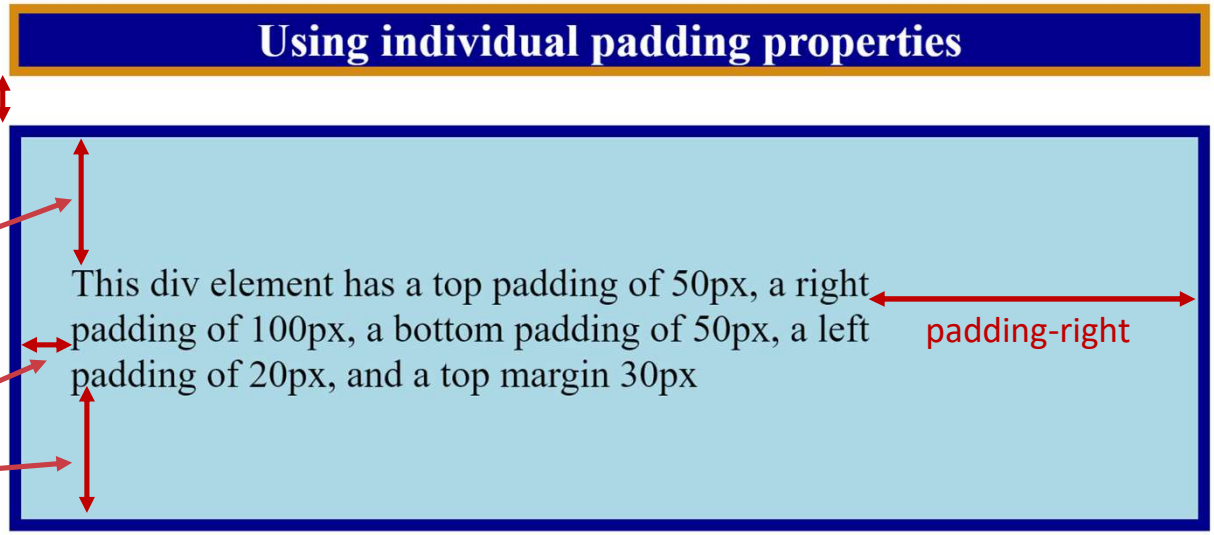
https://www.tutorialrepublic.com/css-tutorial/css-box-model.php

- Every box is composed of four areas:

  - **Content**: The content of the box is where text and images appear
  - **Padding**: Clears an area around the content. The padding is transparent
  - **Border**: A border that goes around the padding and content
  - **Margin**: Clears an area outside the border. The margin is transparent

Example

```
div {
  border: 5px solid darkblue;
  margin-top: 20px;
  background-color: lightblue;
  padding-top: 50px;
  padding-right: 100px;
  padding-bottom: 50px;
  padding-left: 20px;
<!-- padding: 50px 100px 50px 20px; -->
}

h3 {
 border: 4px solid #d68910;
 background-color: darkblue;
 color:white;
 text-align: center;
}
```



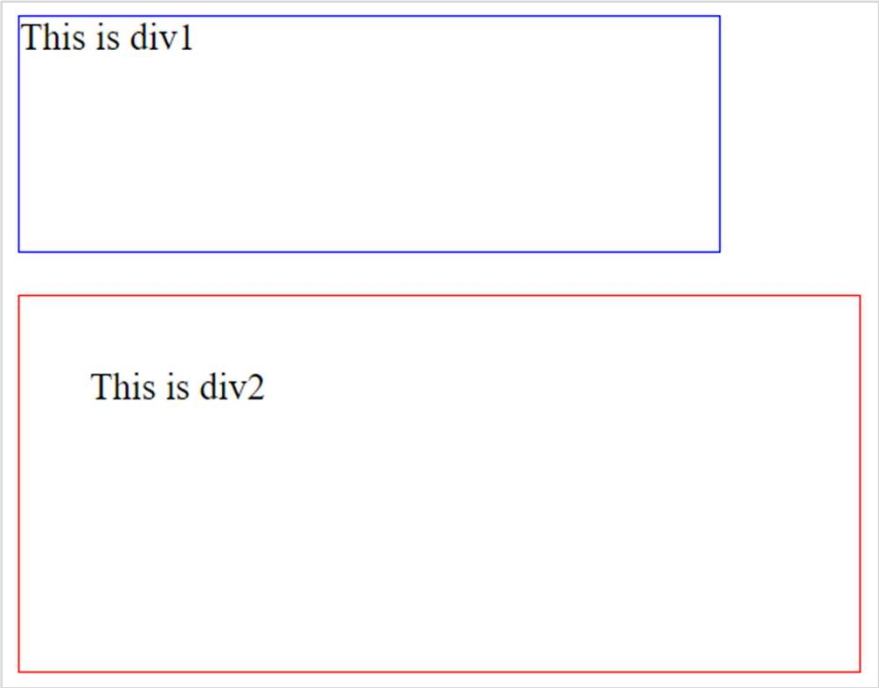

```
<h3>Using individual padding properties</h3>
<div>
This div element has a top padding of 50px, a right
padding of 100px, a bottom padding of 50px, a left
padding of 20px, and a top margin 30px
</div>
```

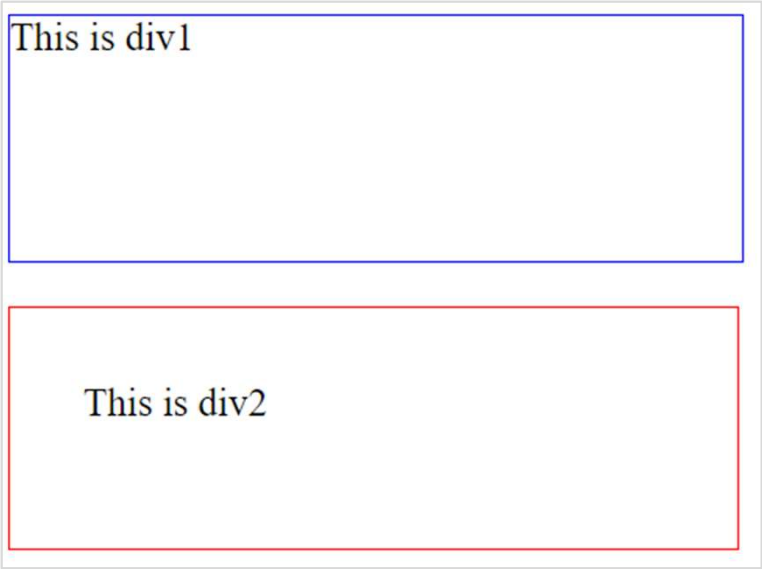
CSS Box Sizing

Self-Study

- The CSS box-sizing property allows us to include the padding and border in an element's total width and height.
- The box-sizing property can make building CSS layouts easier and a lot more intuitive.
- Example:

```
.div1 {
width: 300px;
height: 100px;
border: 1px solid blue;
}
.div2 {
width: 300px;
height: 100px;
padding: 30px;
border: 1px solid red;
}
```





By default:
width + padding + border = actual width of an element
height + padding + border = actual height of an element

```
<div class="div1">This is div1</div>
<br>
<div class="div2">This is div2</div>
```

```
.div1,.div2{box-sizing: border-box;}
```

padding and border are included in the width and height.
Both divs are the same size now!

Customise the layout of pages

- There are different methods in CSS to customise the layout of pages
  - Tables
  - In-line blocks
  - Position
  - Float

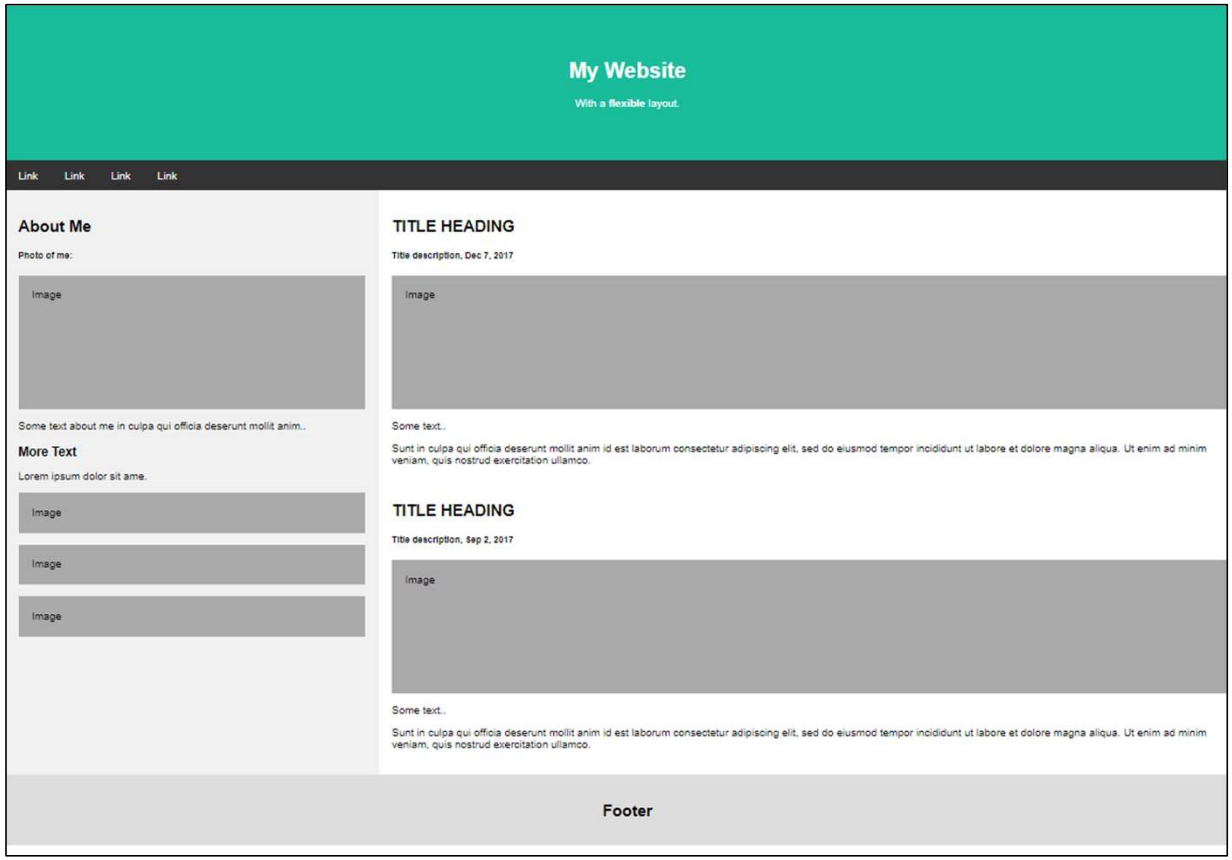
  Legacy Layout Methods

  - Flexbox
  - Grid

  Modern layout methods

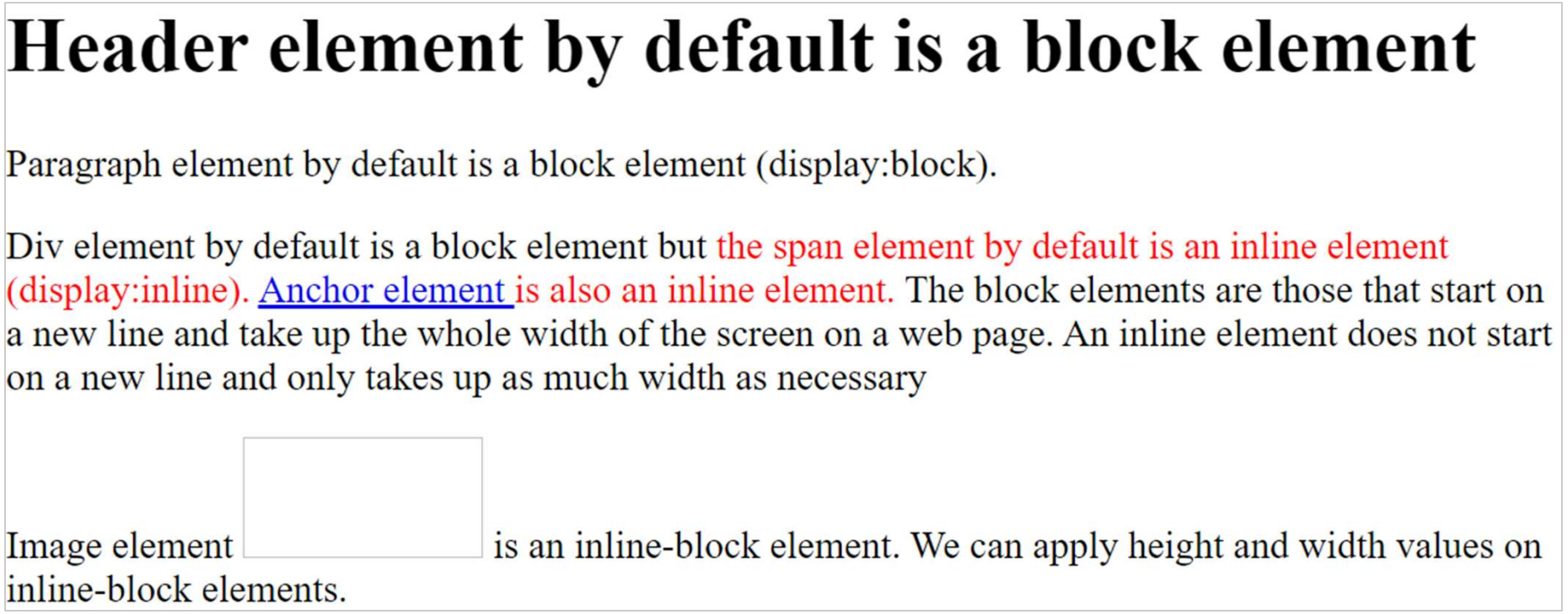
Make it easier to design **responsive layout**

➢ Responsive Web Design (RWD) makes your web page look good on all devices



CSS display Property

```
<h1>Header element by default is a block element</h1>
<p> Paragraph element by default is a block element (display:block).</p>
<div>Div element by default is a block element but <span style="color:red"> the span element
by default is an inline element (display:inline). <a href="www.lboro.ac.uk"> Anchor element
</a> is also an  inline element.</span> The block elements are those that start on a new line
and take up the whole width of the screen on a web page. An inline element does not start on
a new line and only takes up as much width as necessary</div>
<p> Image element <img scr=""  width="100px" height="100px" > is an inline-block element
(display:inline-block). We can apply height and width values on inline-block elements.</p>
```



CSS `display` Property (continued)

- The display property specifies if/how an element is displayed
- some different display values:
  - `{display: none;}`
  - `{display: inline;}`
  - `{display: block;}`
  - `{display: inline-block;}`
  - `{display: flex;}`
  - `{display: grid;}`

  } will be discussed later
- Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline
- A block-level element always starts on a new line and by default takes up the full width available (stretches out to the left and right as far as it can). Examples of block-level elements: , <h1> - <h6>, <p>, <form>, <header>, <footer>, <section>, <ul> and
- An inline element does not start on a new line and only takes up as much width as necessary. Examples of inline elements: , <a>
- Compared to “display: inline”, the major difference is that “display: inline-block” allows to set a width and height on the element. An example of inline-block element is <img>
- “display: none” removes the entire element from the page

CSS display Property - Example

- Override The Default Display Value

```
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
```

- Item 1
- Item 2
- Item 3

```
li {display:inline;}
```

Item 1 Item 2 Item 3

```
<div>
Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
</div>
<div>
Etiam semper diam at erat
pulvinar, at pulvinar felis
blandit.
</div>
```

```
div{display:inline-block;width:150px}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Etiam semper diam at erat pulvinar, at pulvinar felis blandit.

CSS Positioning

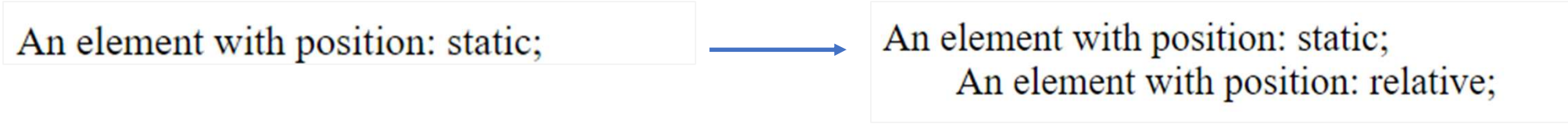
Self-Study

- The position property sets how an element is positioned in a document
- The top, right, bottom, and left properties determine the final location of positioned elements
- There are five different position values: static, relative, fixed, absolute, sticky

1. static. HTML elements are positioned static **by default**. it is always positioned according to the normal flow of the page.
2. relative. An element with "position: relative" is positioned relative to its normal position

```
<div>
An element with position: static;
</div>
<div class="container">
An element with position: relative;
</div>
```

```
div.container {
  position: relative;
  left: 30px;
}
```



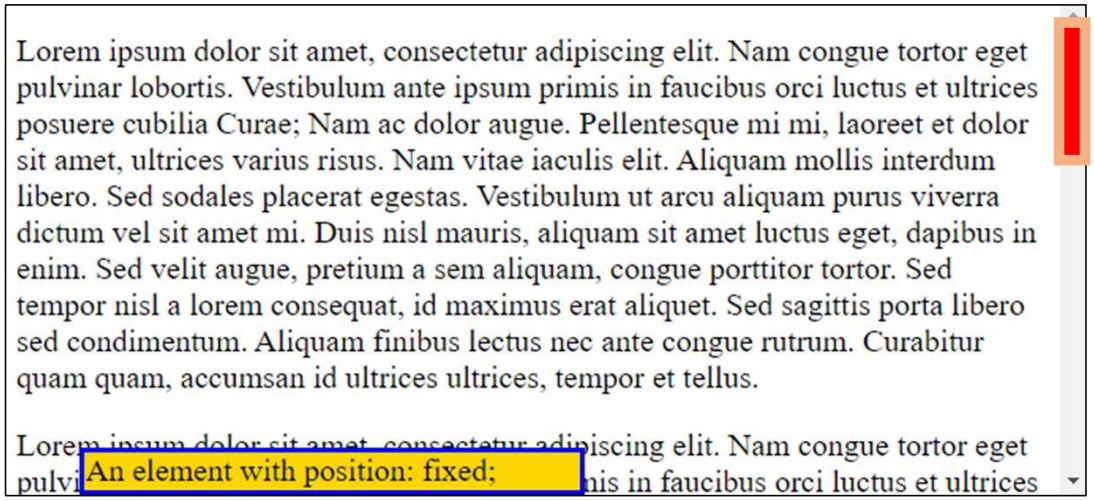
CSS Positioning (continued)

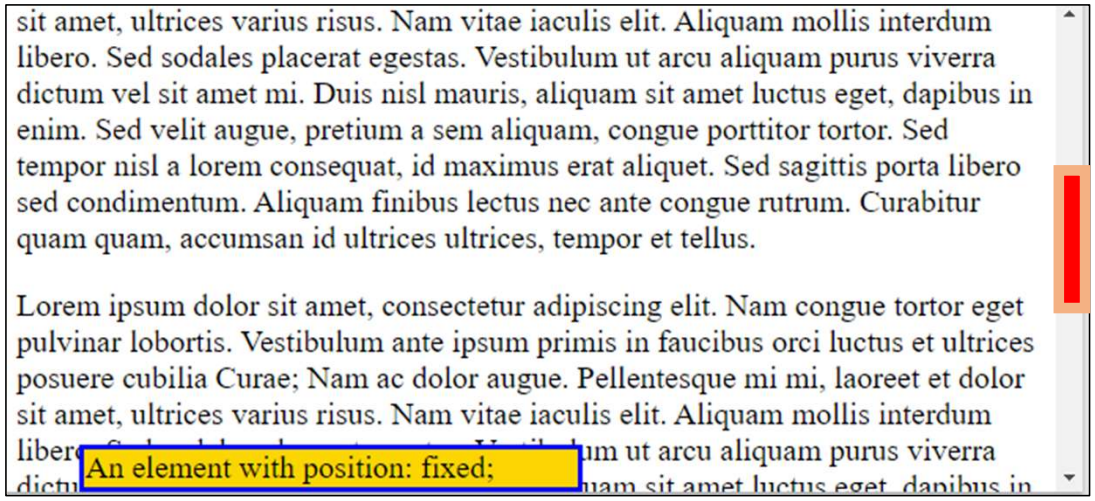
Self-Study

3. fixed. An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled

```
<p>
Lorem ipsum dolor sit amet, …
…
…
</p>
<div class="fixed-container">
An element with position: fixed;
</div>
```

```
div.fixed-container{
  position: fixed;
  left: 40px;
  bottom: 0px;
  width: 250px;
  border: 3px solid blue;
}
```





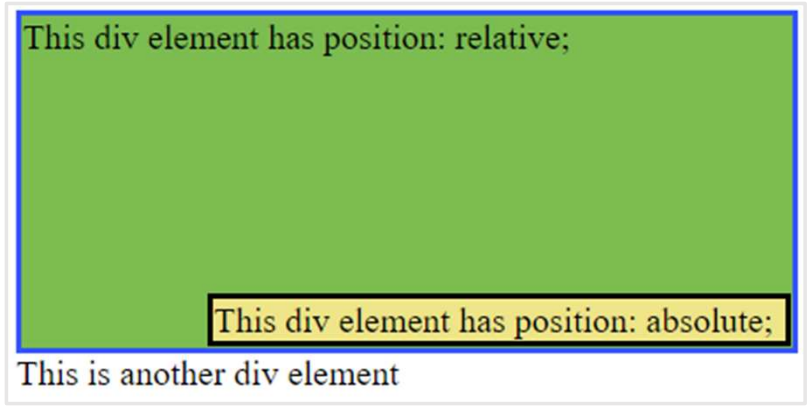
CSS Positioning (continued)

Self-Study

4. absolute. The absolutely positioned element is positioned relative to its nearest ancestor that is not static. If a positioned ancestor doesn't exist, it uses the document body

```
<div class="relative">This div element has position: relative;
    <div class="absolute">This div element has position: absolute;</div>
</div>
<div>This is another div element</div>
```

```
div.relative {
    position: relative;
    width: 350px;
    height: 150px;
    border: 3px solid #3353FF;
    background: 7CBC51;}
div.absolute {
    position: absolute;
    bottom: 0px;
    right: 0px;
    width: 260px;
    border: 3px solid black;
    background: #EFE589; }
```



5. sticky. An element with position: sticky; is positioned based on the user's scroll position

CSS float Property

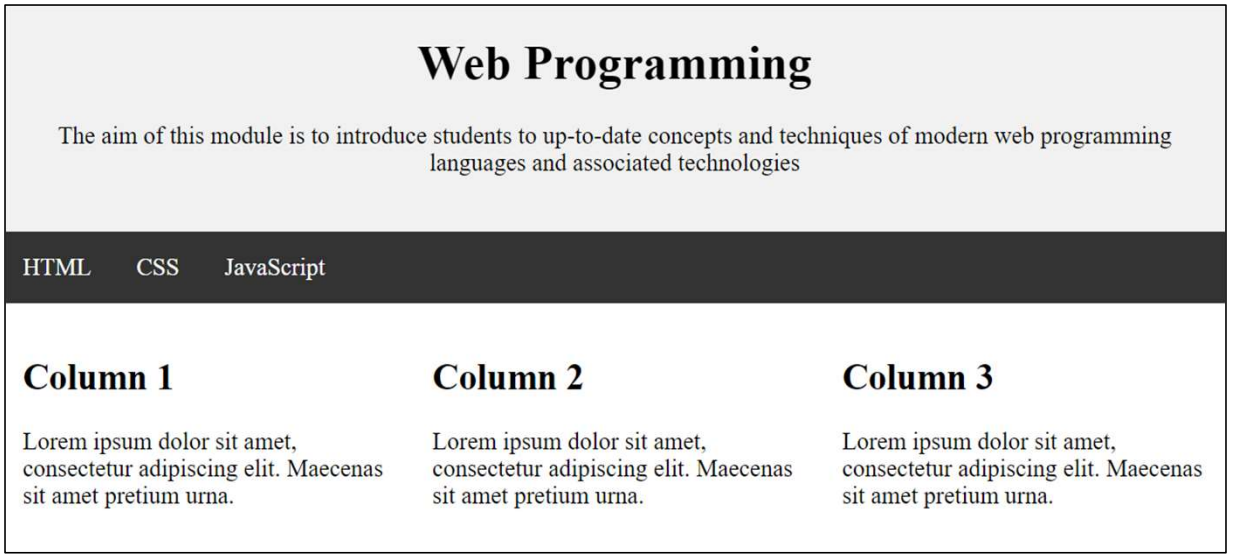
- The float CSS property places an element on the left or right side of its container
- Example

```
<html>
<head>
<style>
img{
width: 100px;
height: 100px;
float: right;
margin-left:5px;
}
</style>
</head>
<body>
<div>
<img src="images/logo.png" alt="Lboro logo" >
<p> Loughborough University (abbreviated as Lough or Lboro for post-nominals) is a public research university in the market town of Loughborough, Leicestershire, England. It has been a university since 1966, but it dates back to 1909, when Loughborough Technical Institute began with a focus on skills directly applicable in the wider world. In March 2013, the university announced it had bought the former broadcast centre at the Queen Elizabeth Olympic Park as a second campus. It belonged to the 1994 Group of smaller research universities until the group dissolved in November 2013. </p>
</div>
</body>
</html>
```

Using float property for laying out the page (not recommended anymore)

- Example

```
<section>
  <article>
    <h2>Column 1</h2>
    <p>Lorem ipsum dolor sit amet,… </p>
  </article>

  <article>
    <h2>Column 2</h2>
    <p>Lorem ipsum dolor sit amet,… </p>
  </article>

  <article>
    <h2>Column 3</h2>
    <p>Lorem ipsum dolor sit amet,… </p>
  </article>
</section>
```




```
/* Create three equal columns that floats next
to each other */
article{
  float: left;
  width: 33.33%;
  padding: 15px; }

/* Clear floats after the columns */
section::after {
   content: "";
   display: table;
   clear: both;
}
```

Note: Elements after a floating element will flow around it. To avoid this, use the **clearfix hack**

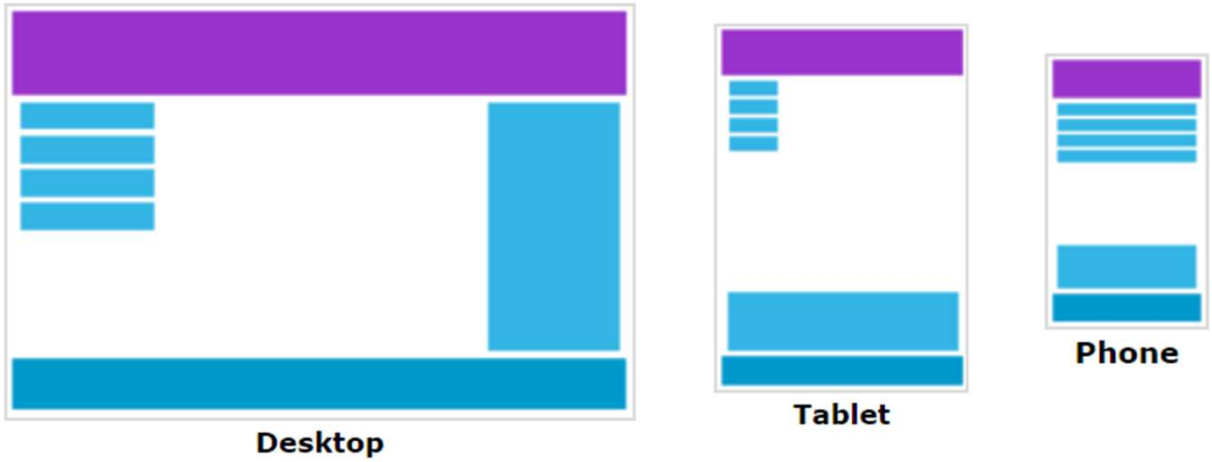
- Full source code:

https://www.w3schools.com/css/css_website_layout.asp

https://www.w3schools.com/howto/howto_css_clearfix.asp

Responsive Web Design (RWD)

- Responsive Web Design (RWD) makes your web page look good on all devices



- Quick fixes for having a more responsive page
- Using <meta> viewport element
- Using Media Query technique
- Set width property to a percentage

- Modern Layout
- Using modern layout methods such as flexbox and grid
- Using responsive framework such as Bootstrap or W3.CSS

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<!– helps the page is sized correctly on all devices. -->

        @media only screen and (max-width: 600px) {
          body {background-color: blue;  }
          article{ width:100%; }
        } <!-- If the browser window is 600px or smaller, the
        background color will be blue, and the article divs
        gets the whole width of the screen -->
img {
  max-width: 100%;
  height: auto;
} <!-- The image will be responsive and scale down if it has
to, but never scale up to be larger than its original size-->
```

Customise the layout of pages

- There are different methods in CSS to customise the layout of pages
  - Tables
  - In-line blocks
  - Position
  - Float

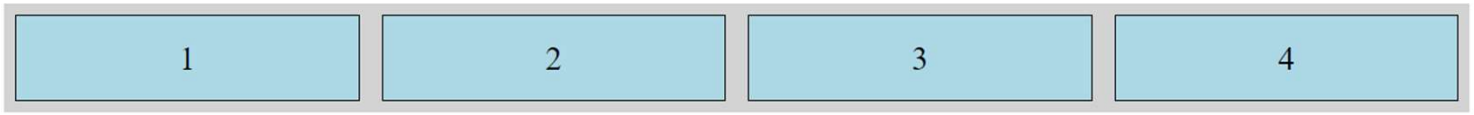
  Legacy Layout Methods

  - Flexbox. The Flexible Box Module, usually referred to as flexbox, is a one-dimensional layout method for arranging items in rows or columns. This layout allows responsive elements within a container to be automatically arranged depending upon screen size
  - Grid. The CSS Grid Layout Module offers a grid-based (two-dimensional) layout system which controls columns and rows together

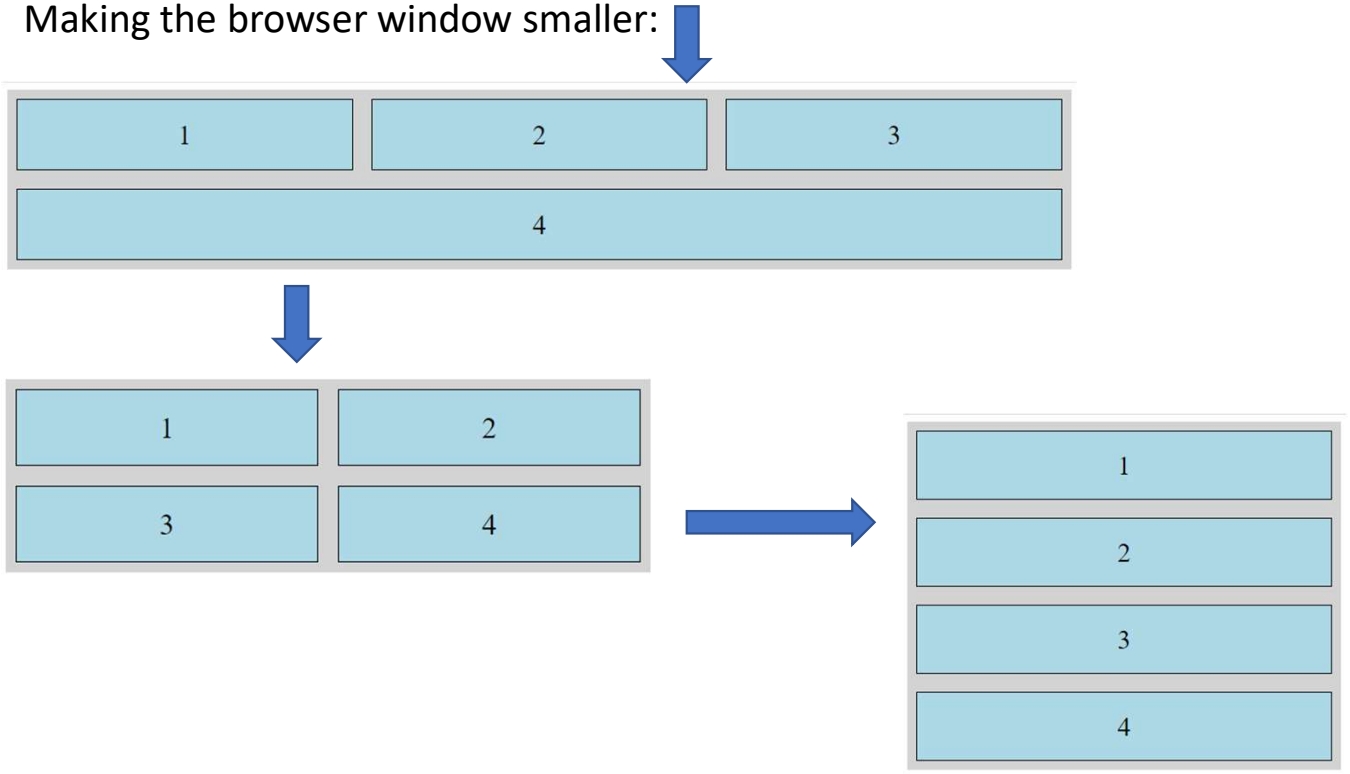
Modern layout methods → make it easier to design **responsive layout**

CSS Flexbox

- We first need to select which elements are to be laid out as flexible boxes. To do this, we set a special value of display property (`display:flex;`) on the parent element of the elements you want to affect



Making the browser window smaller:




```
<style>
.flex-container {
  display: flex;
  flex-wrap: wrap;
  background-color: lightgray;
/* by default:flex-direction:row; */}
.flex-container > div {
  flex: 300px;
  background-color: lightblue;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
  border: 1px solid black;
  }
</style>
```

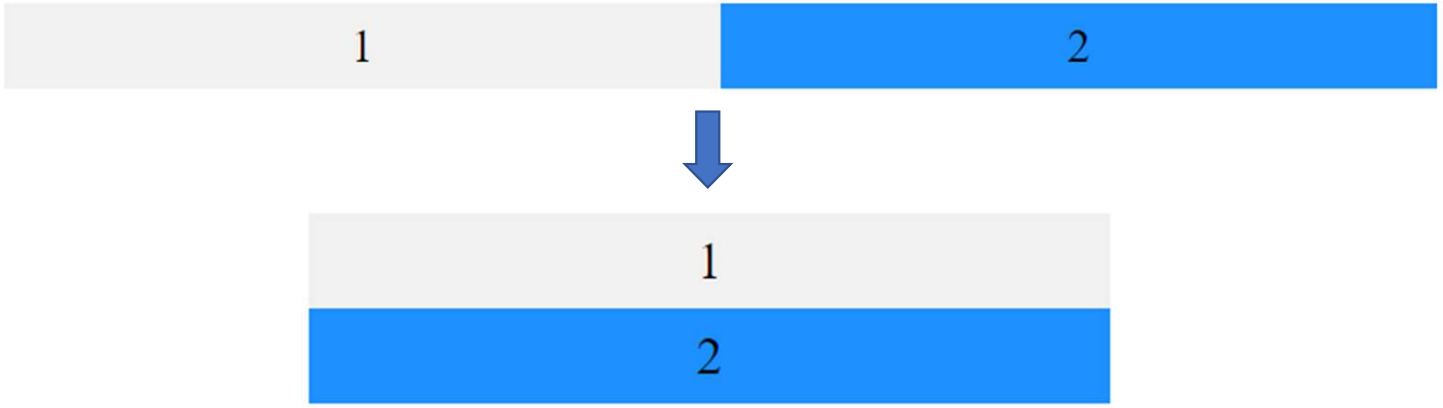
```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
</div>
```

in <body>:

CSS Flexbox (continued)

- To make it more responsive, we can use percentage values as well as Media Query

```
<div class="flex-container">
  <div class="flex-item-left">1</div>
  <div class="flex-item-right">2</div>
</div>
```




```
.flex-container {
    display: flex;
    flex-direction: row;
    font-size: 30px;
    text-align: center;
 }
 .flex-item-left {
    background-color: #f1f1f1;
    padding: 10px;
    flex: 50%;
 }
 .flex-item-right {
    background-color: dodgerblue;
    padding: 10px;
    flex: 50%;
 }

 …
```

```
@media (max-width: 800px) {
  .flex-item-right, .flex-item-left {
    flex: 100%;
    }
}
```

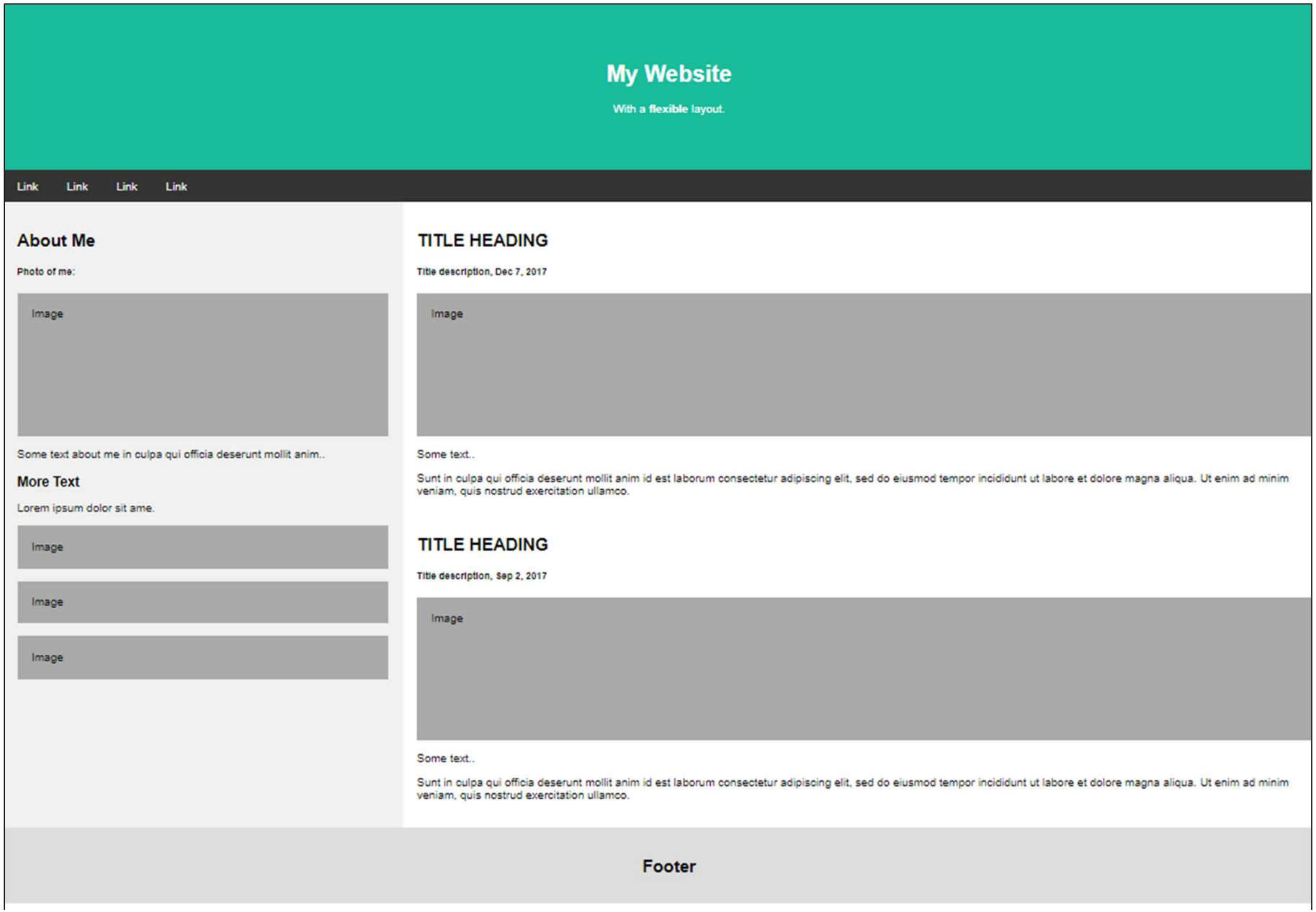
Or:

```
@media (max-width: 800px) {
   .flex-container {
     flex-direction: column;
   }
 }
```

CSS Flexbox

- Example:

https://www.w3schools.com/css/tryit.asp?filename=trycss3_flexbox_website2



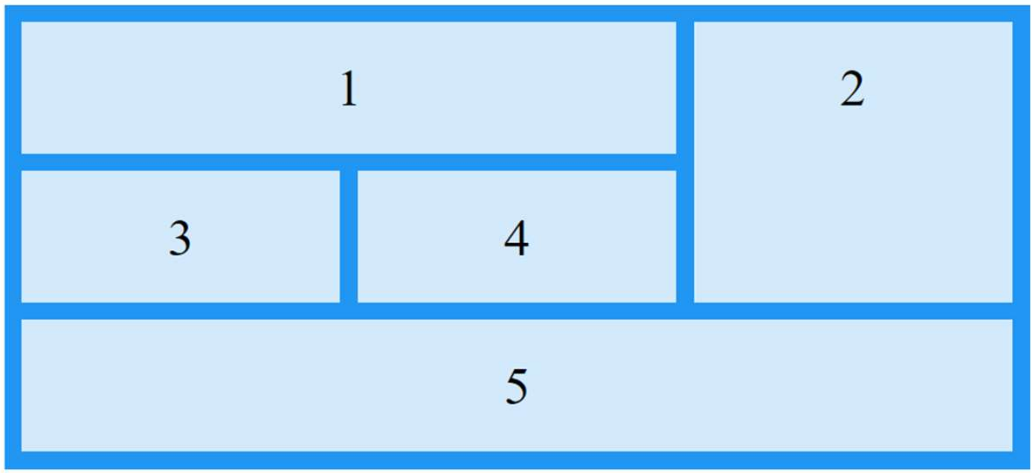
CSS Grid Layout

Beyond Scope

- Offers a grid-based layout system, with rows and columns
- An element becomes a **grid container** when its display property is set to grid: `display: grid;`

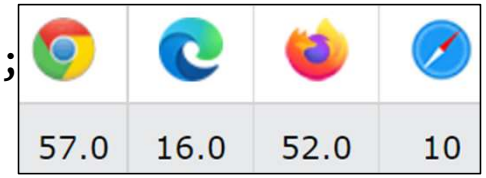
- Example:

```
<main>
  <div class="grid-item item1">1</div>
  <div class="grid-item item2">2</div>
  <div class="grid-item item3">3</div>
  <div class="grid-item item4">4</div>
  <div class="grid-item item5">5</div>
</main>
```



https://www.w3schools.com/css/css_grid_item.asp

```
main {
    display: grid;
    grid-gap: 10px;
    background-color: #2196F3;
    padding: 10px; }
.grid-item {
    background-color: #D3EAFD;
    text-align: center;
    padding: 20px;
    font-size: 30px;  }
.item1 {
    grid-column: 1 / span 2;
    grid-row: 1; }
.item2 {
    grid-column: 3;
    grid-row: 1 / span 2; }
.item5 {
    grid-column: 1 / span 3;
    grid-row: 3; }
```

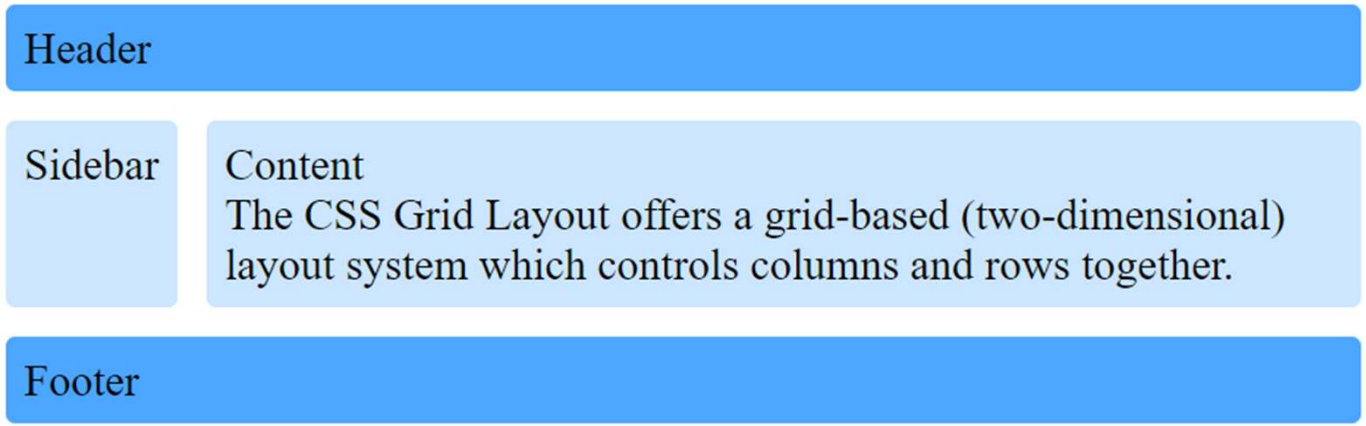


CSS Grid Layout (continued)

- The **grid-area** property can be used to assign names to grid items
- Named grid items can be referred to by the **grid-template-areas** property of the grid container

- Example:

```
<div class="grid-container">
  <div class="grid-item header">Header</div>
  <div class="grid-item sidebar">Sidebar</div>
  <div class="grid-item content">Content
    <br /> The CSS Grid Layout offers a grid-based
(two-dimensional) layout system.</div>
  <div class="grid-item footer">Footer</div>
</div>
```



Beyond Scope

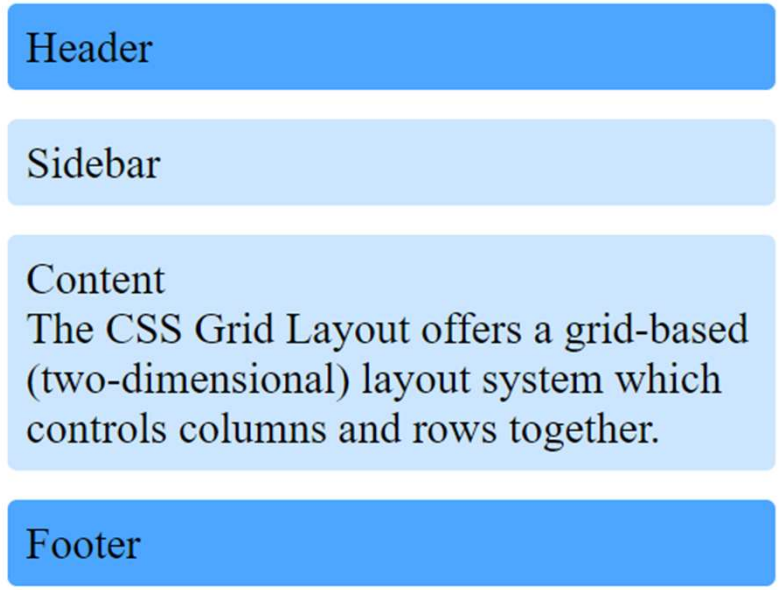
```
.header {
    grid-area: myHeader;
  }
.sidebar {
    grid-area: mySidebar;
  }
.content {
    grid-area: myContent;
  }
.footer {
    grid-area: myFooter;
  }
.grid-container {
    display: grid;
    grid-gap: 16px;
    grid-template-areas:
    "myHeader   myHeader"
    "mySidebar  myContent"
    "myFooter   myFooter";
}

…
```

CSS Grid Layout (continued)

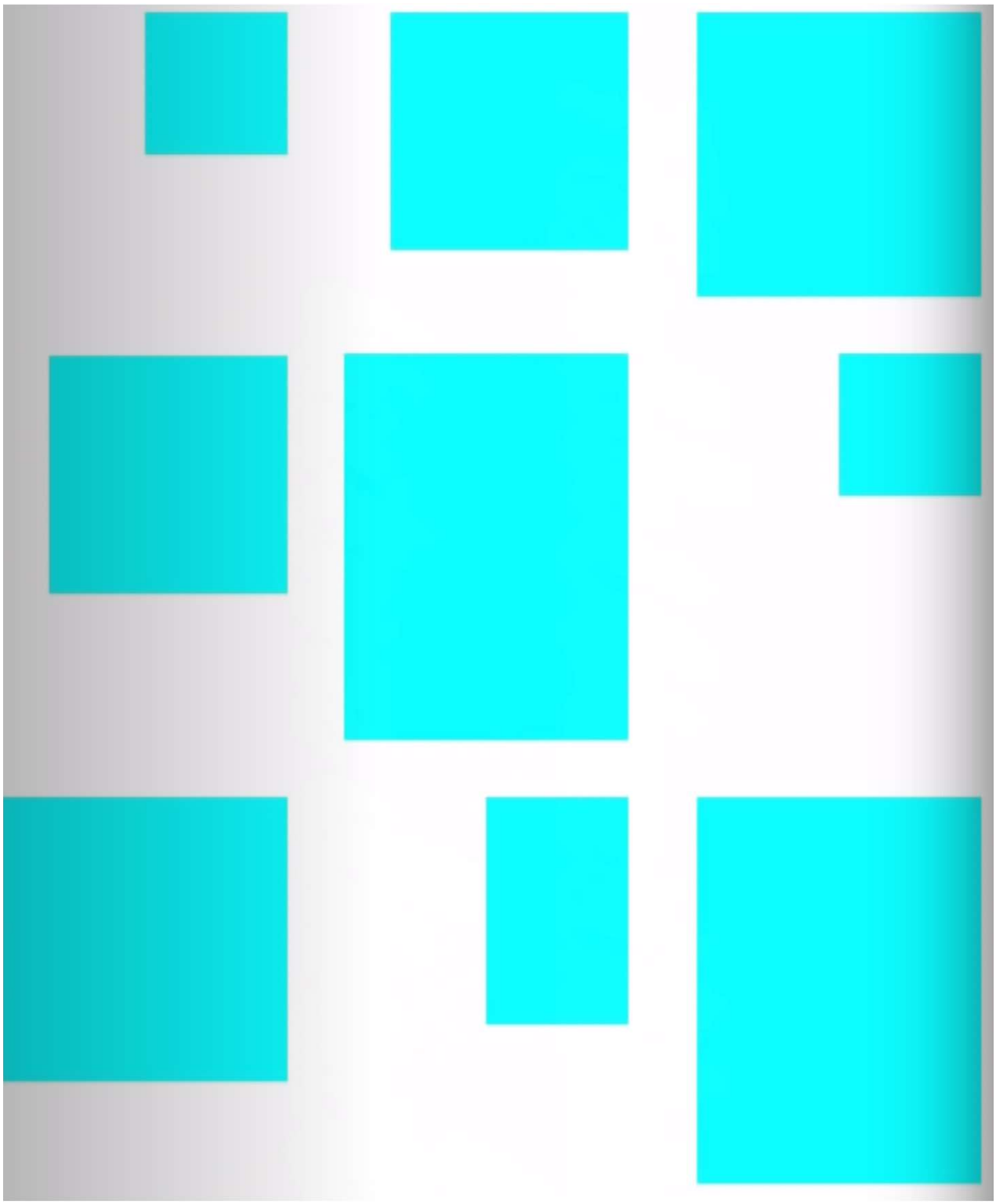
Beyond Scope

```
.header {
    grid-area: myHeader;
  }
.sidebar {
    grid-area: mySidebar;
  }
.content {
    grid-area: myContent;
  }
.footer {
    grid-area: myFooter;
  }
.grid-container {
    display: grid;
    grid-gap: 1em;
    grid-template-areas:
    "myHeader   myHeader"
    "mySidebar  myContent"
    "myFooter   myFooter";
  }
…
```

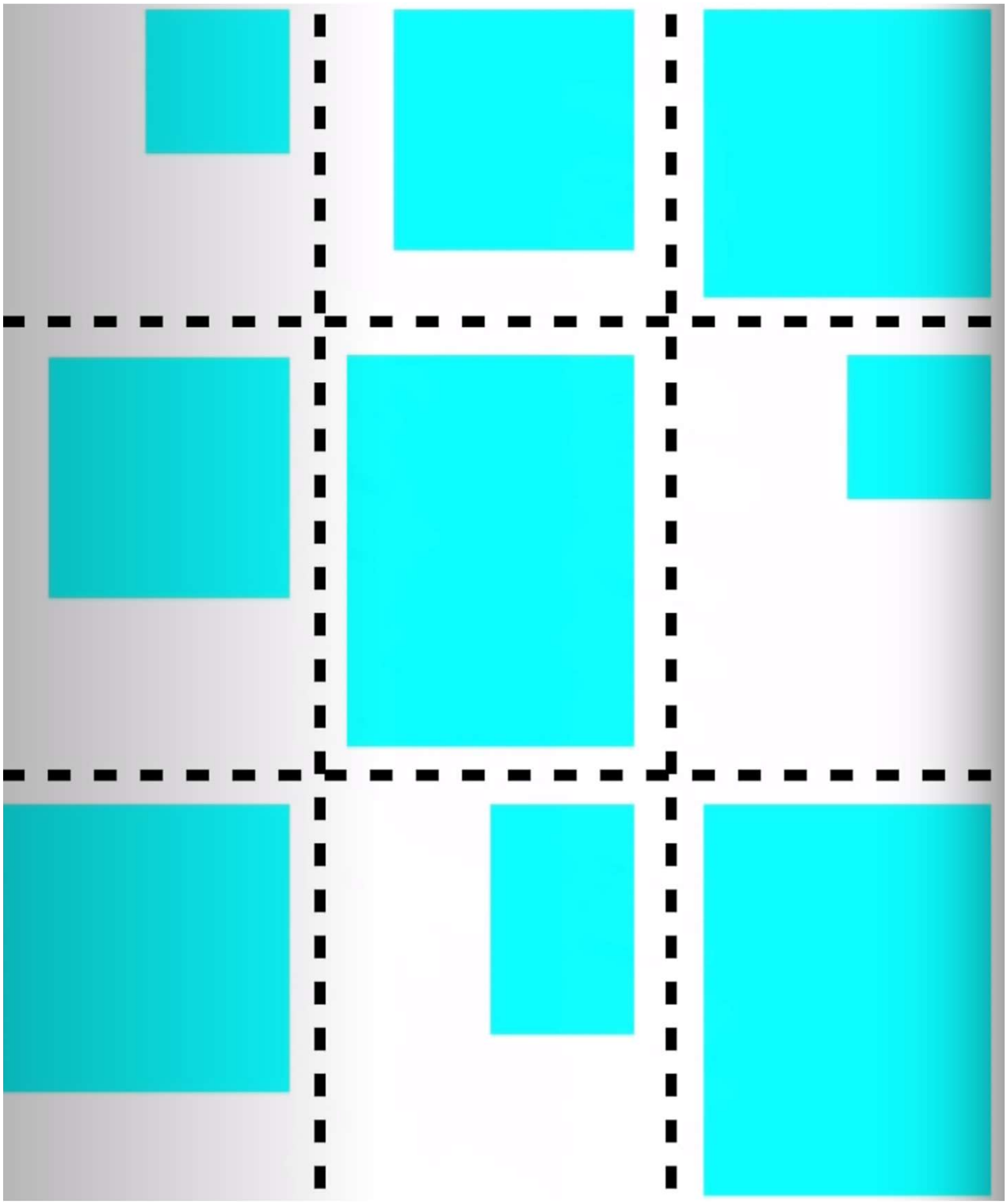
```
@media only screen and (max-width: 600px)  {
.grid-container {
  grid-template-areas:
  "myHeader   myHeader"
  "mySidebar  mySidebar"
  "myContent  myContent"
  "myFooter   myFooter";
   }
}
```



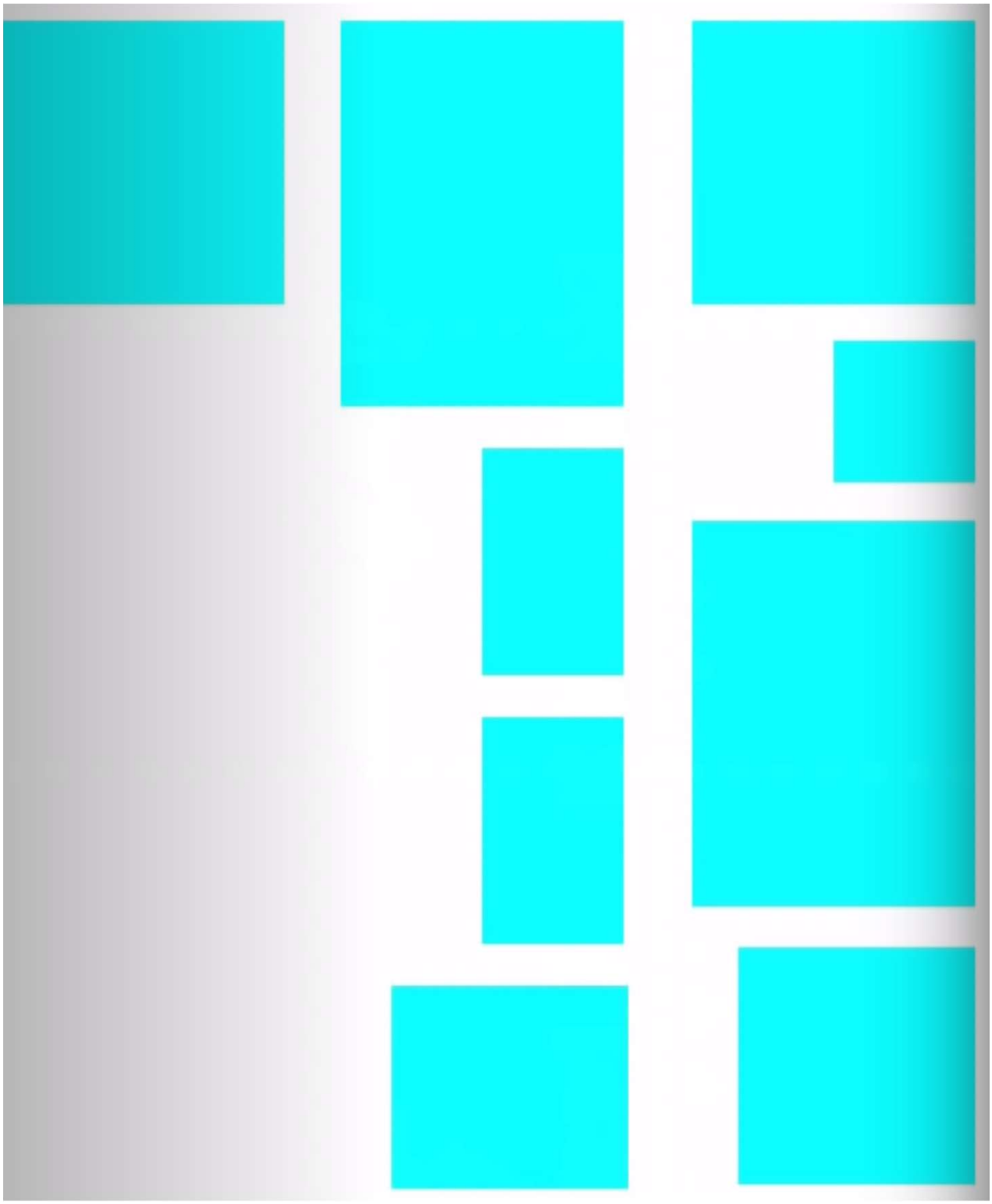
Using Grid Layout



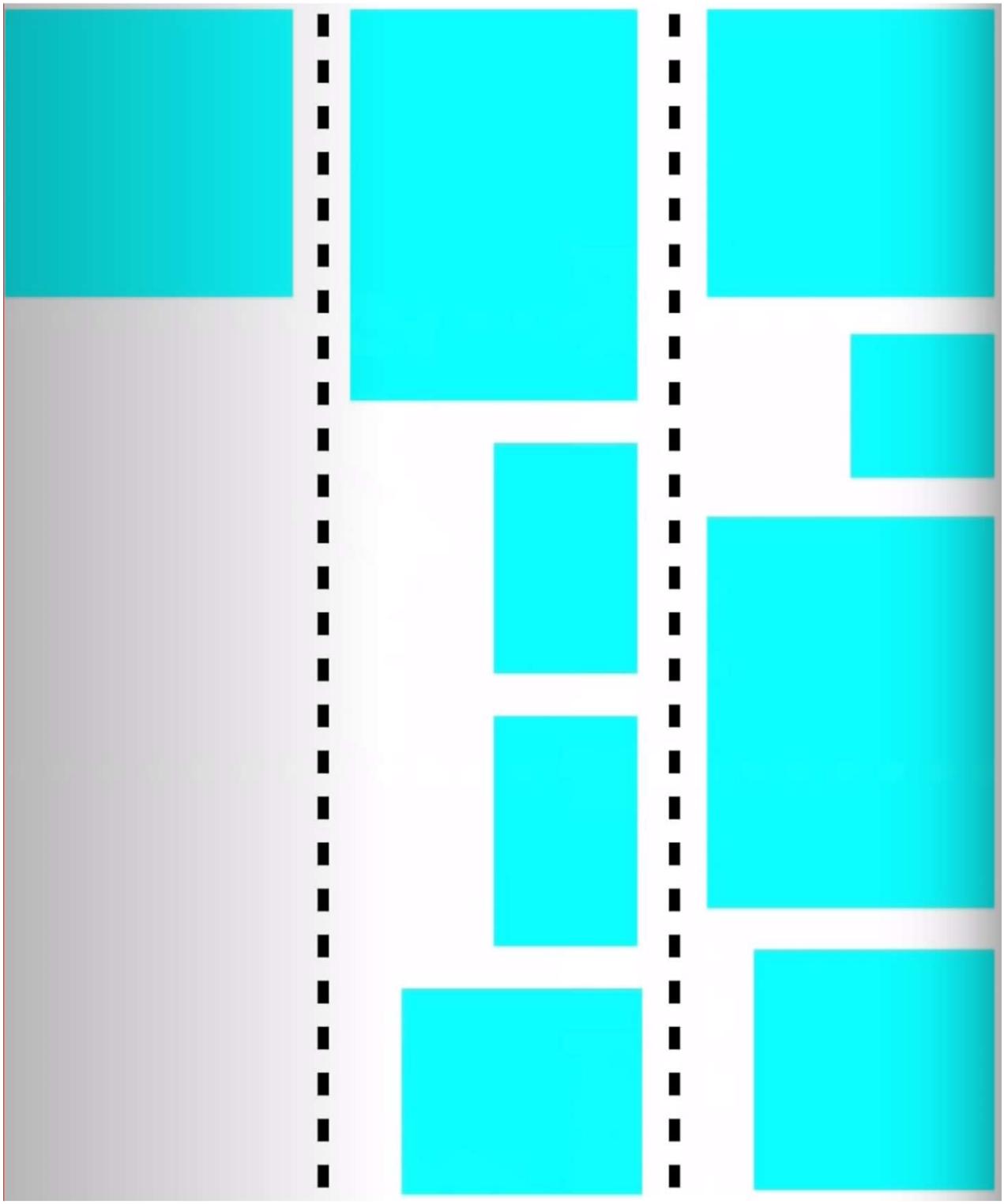
Using Grid Layout



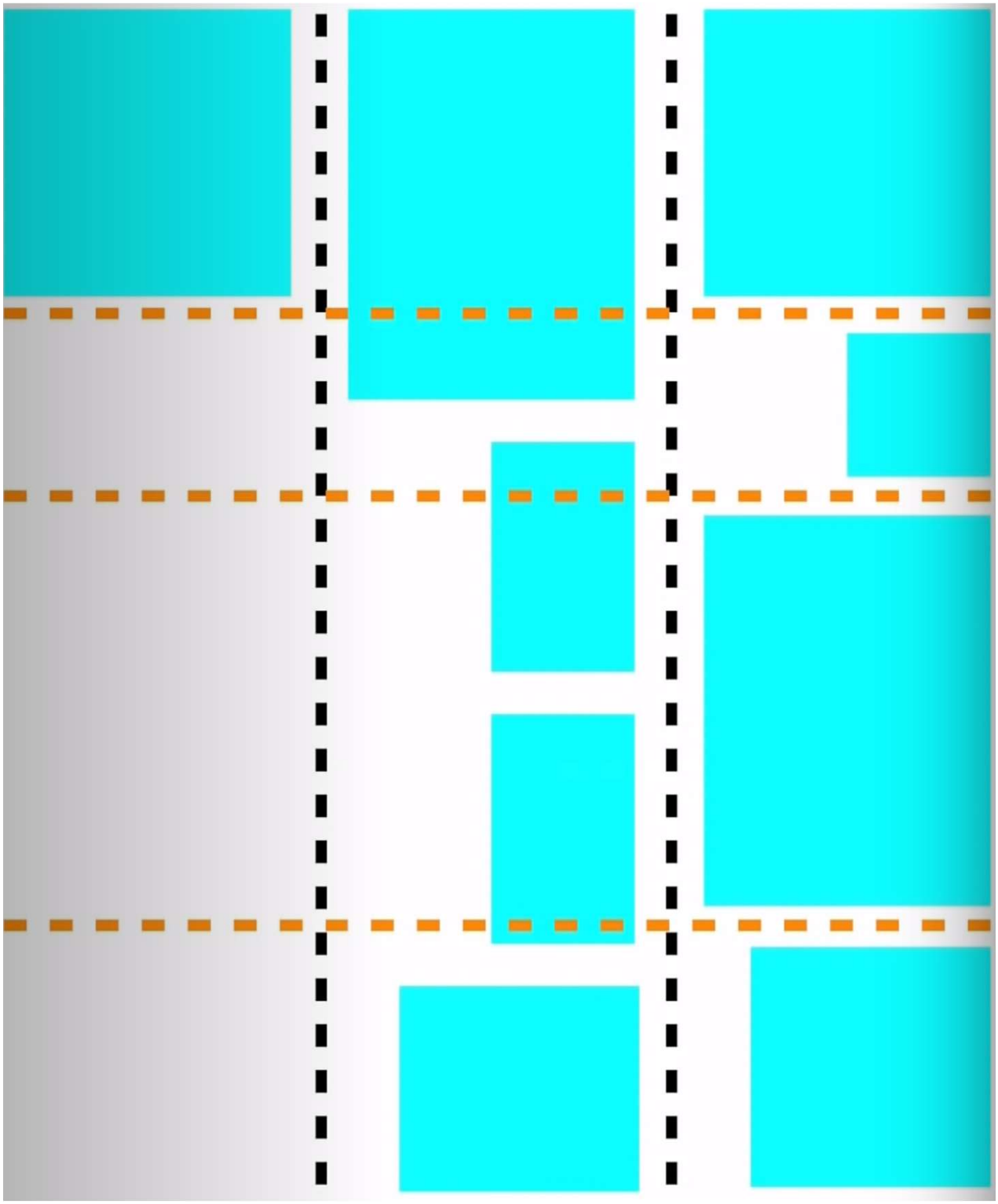
Using Flex Layout



Using Flex Layout



Using Flex Layout



CSS Units

- CSS has several different units for expressing a length such as px, em, and %

- px:
  - Pixels are an **absolute** unit of measurement
  - 1px is a single dot on a screen.
  - The size specified in pixels is fixed and won't change based on the size of the screen or the surrounding elements

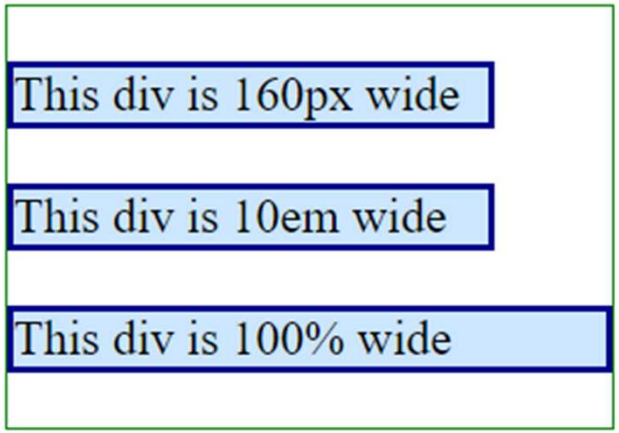
- em:
  - The ‘em’ unit is **relative**
  - When you set an element's size in em, it is calculated based on the font size of its nearest parent element
  - Useful for creating scalable and flexible layouts

- percentage (‘%’):
  - The ‘%’ unit is **relative**
  - It is often used for specifying widths and heights in relation to the parent element
  - Suitable for elements that need to adapt to different screen sizes

CSS Units

- CSS has several different units for expressing a length such as px, em, and %

```
<style>
  * {box-sizing: border-box;}
  .wrapper {width:200px; border:1px solid green; font-size: 16px;}
  .box {border:2px solid darkblue; background-color: #cce6ff;}
  .px {width: 160px;}
  .em {width: 10em;}
  .percentage{width: 100%;}
</style>

<div class="wrapper">
  <div class="box px">This div is 160px wide</div>
  <div class="box em">This div is 10em wide</div>
  <div class="box percentage">This div is 100% wide</div>
</div>
```



Reference:
https://www.w3schools.com/cssref/css_units.php
https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Values_and_units