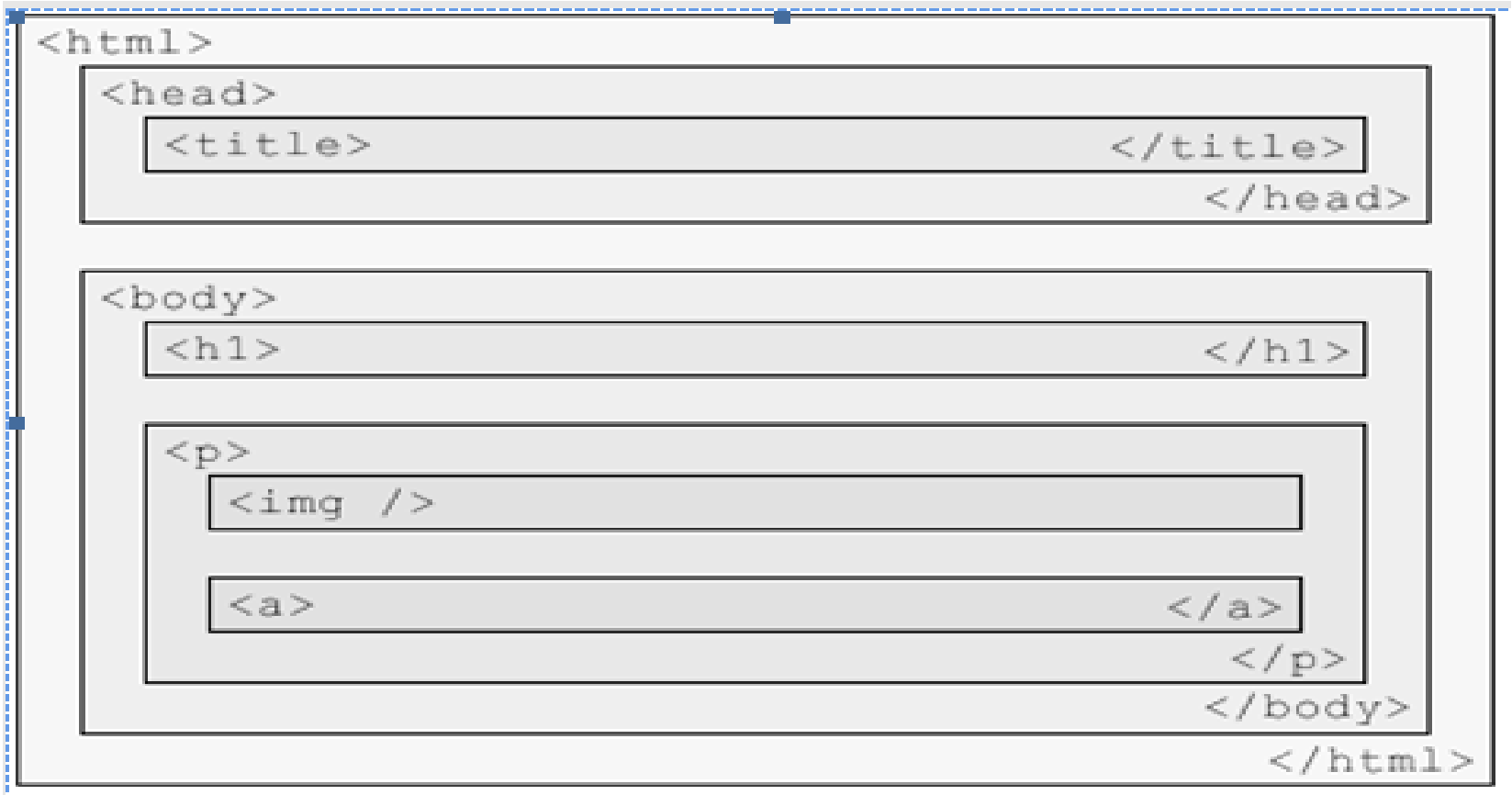
CSS

Style sheet

- Cascading Style Sheets
- A style sheet is a set of instructions each of which tells a browser how to draw a particular element on a page.
- Well-formed HTML documents are a collection of elements arranged in a kind of containment hierarchy.
- CSS handles the look and feel part of a web page.
- Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, as well as a variety of other effects.



The HTML containment hierarchy

Advantages of CSS

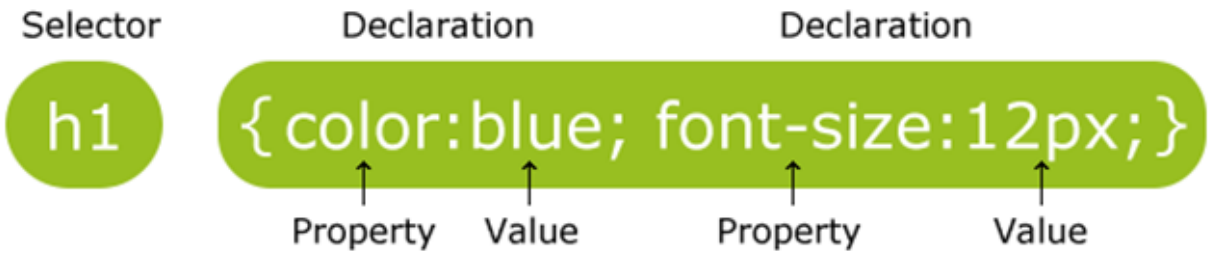
- **CSS saves time** - You can write CSS once and then reuse the same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many web pages as you want.
- **Pages load faster** - If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So, less code means faster download times.
- **Easy maintenance** - To make a global change, simply change the style, and all the elements in all the web pages will be updated automatically.
- **Superior styles to HTML** - CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Multiple Device Compatibility** - Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cellphones or for printing.
- **Global web standards** – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it’s a good idea to start using CSS in all the HTML pages to make them compatible with future browsers.


HTML
urvashiobsession
HTML+CSS

Introduction

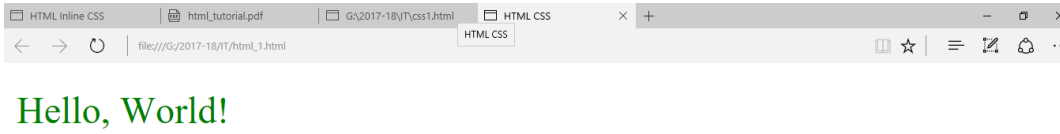
- We refer to the instructions in a style sheet as **statements**. There are a few different types of statement, but the one you'll use most is referred to as a **rule**.
- Rules have two parts: a **selector** and a **declaration**.
- The **selector** tells a browser which elements in a page will be affected by the rule. There are a number of different types of selector.
- The **declaration** tells the browser which set of **properties** to apply. There are many different properties.
- All these statements are contained in a Cascading Style Sheet. This is nothing more than a text file, with the suffix .css added to the name, something like core-style .css.
- Rules have a very simple form: the selector, followed by the set of properties, which are surrounded by curly braces (that is { and }).
- p {font-size: 1em} -- selects any <p> elements, or paragraphs, and makes their font 1em.

- A CSS rule has two main parts: a selector, and one or more declarations:
- The selector is normally the HTML element you want to style.
- Each declaration consists of a property and a value.
- The property is the style attribute you want to change. Each property has a value.



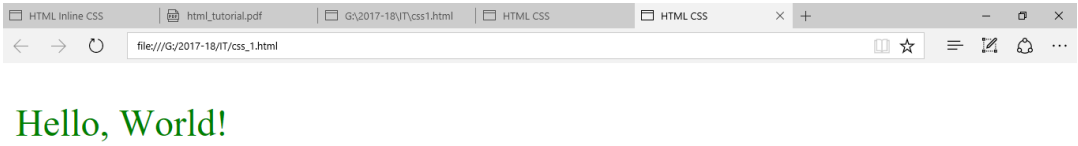
HTML tag example

```
<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p><font color="green" size="5">Hello, World!</font></p>
</body>
</html>
```



CSS Example

```
<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p style="color:green;font-size:24px;">Hello, World!</p>
</body>
</html>
```



CSS Comments

- Comments are used to explain your code, and may help you when you edit the source code at a later date. Comments are ignored by browsers.
- A CSS comment begins with "/*", and ends with "*/", like this:
- /*This is a comment*/

CSS selector

- CSS selectors are used to "find" (or select) HTML elements based on their element name, id, class, attribute, and more.
  - The element Selector
  - The id Selector
  - The class Selector
  - Grouping Selectors

Element Selector

- Example to give a color to all level 1 headings:

```
h1 {
   color: #ff0000;
}

H2
{
font-size:32;
}
```

The Type Selectors :

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  color: #36CFFF;
}
H2
{
color:blue;
font-size:32;
}
</style>
</head>
<body>
<p >Hello World!</p>
<p>This paragraph is not affected by the style.</p>
<h1> This is heading 1</h1>
<h2> This is heading 2</h2>
</body>
</html>
```

Hello World!

This paragraph is not affected by the

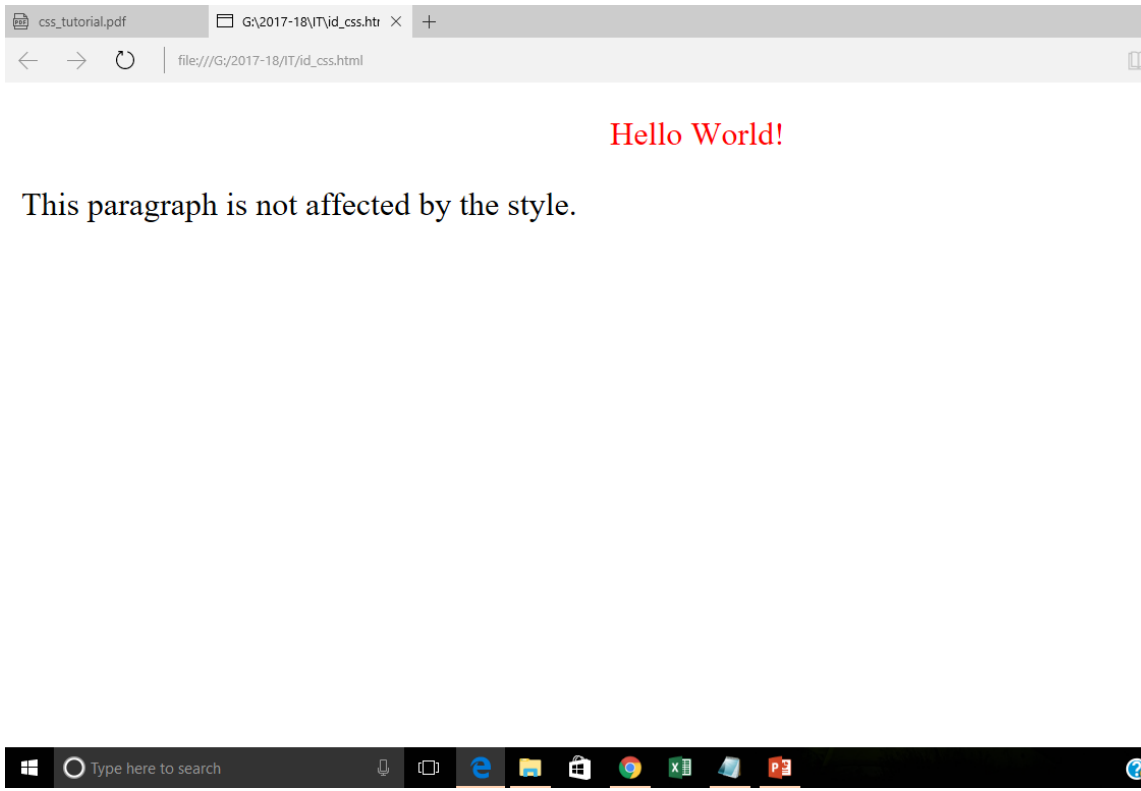
This is heading 1

This is heading 2

The id Selector

- The id selector is used to specify a style for a single, unique element.
- The id selector uses the id attribute of the HTML element, and is defined with a "#".
- The style rule below will be applied to the element with id="para1":

```
<!DOCTYPE html>
<html>
<head>
<style>
#para1
{
text-align:center;
color:red;
}
</style>
</head>
<body>
<p id="para1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>
</body>
```



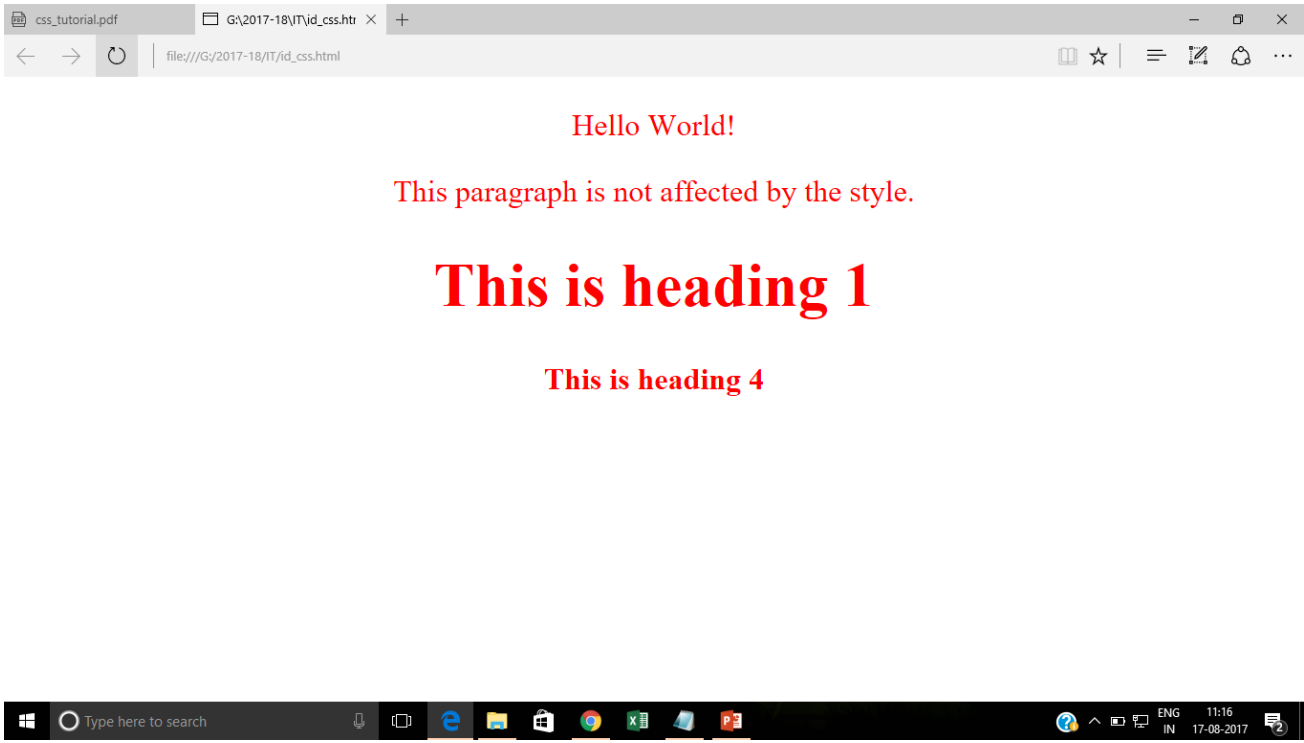
The Universal Selectors:

```
* {
  color: #000000;
}
```

- This rule renders the content of every element in our document in black.

The Universal Selectors:

```
<!DOCTYPE html>
<html>
<head>
<style>
*
{
text-align:center;
color:red;
}
</style>
</head>
<body>
<p id="para1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>
<h1> this is heading 1 </h1>
<h4> This is heading 4 </h4>
</body>
</html>
```



The Class Selectors

- You can define style rules based on the class attribute of the elements. All the elements having that class will be formatted according to the defined rule.

```
.black {
  color: #000000;
}
```

- This rule renders the content in black for every element with class attribute set to black in our document

```
h1.black {
  color: #000000;
}
```

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Internal CSS</title>
<style type="text/css">
.red{
  color: red;
}
.thick{
  font-size:20px;
}
.green{
  color:green;
}
</style>
</head>
<body>
<p class="red">This is red</p>
<p class="thick">This is thick</p>
<p class="green">This is green</p>
<p class="green red"> this is merged class style </p>
<p class="thick green">This is thick and green</p>
```

This is red

This is thick

This is green

This is thick and green

Multiple Style Rules

- You may need to define multiple style rules for a single element.
- You can define these rules to combine multiple properties and corresponding values into a single block as defined in the following example:

```
h1 {
color: #36C;
font-weight: normal;
letter-spacing: .4px;
margin-bottom: 1px;
text-transform: lowercase;
}
```

Grouping Selectors

- If you have elements with the same style definitions, better to group the selectors, to minimize the code.
- Just separate the selectors with a comma

```
h1 {
    text-align: center;
    color: red;
}
h2 {
    text-align: center;
    color: red;
}
p {
    text-align: center;
    color: red;
}
```

```
h1, h2, p {
    text-align: center;
    color: red;
}
```

Types of Style sheets

- You can use CSS in three ways in your HTML document:

1. External Style Sheet: Define style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.
2. Internal Style Sheet: Define style sheet rules in header section of the HTML document using <style> tag.
3. Inline Style Sheet: Define style sheet rules directly along-with the HTML elements using style attribute.

Inline Styles

- An inline style may be used to apply a unique style for a single element.
- To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.
- Rules defined inline with the element overrides the rules defined in an external CSS file as well as the rules defined in <style> element.
- The example below shows how to change the color and the font of a <p> element:

Inline Styles

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Inline CSS</title>
</head>
<body>
<p style="color:red;">This is red</p>

<p style="font-size:20px;">This is thick</p>

<p style="color:green;">This is green</p>

<p style="color:green;font-size:20px;">This is thick and green</p>
</body>
</html>
```

This is red

This is thick

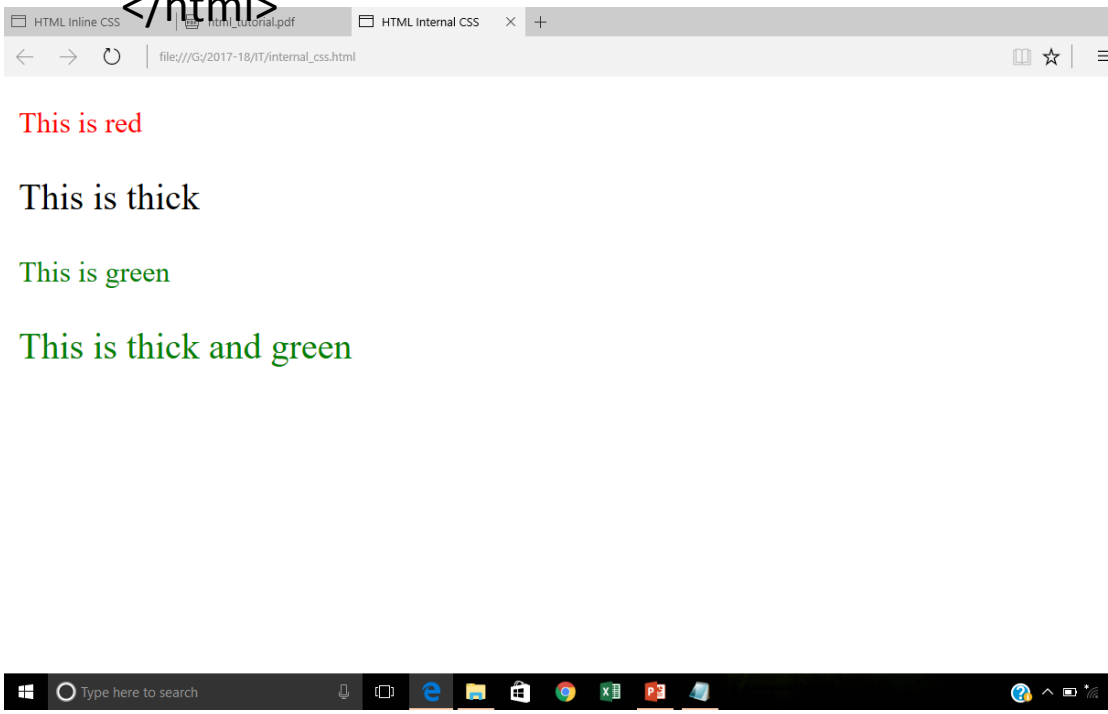
This is green

This is thick and green

Internal Styles

- Internal Style Sheet
- An internal style sheet may be used if one single page has a unique style.
- Internal styles are defined within the <style> element, inside the <head> section of an HTML page:
- If multiple style affect the same element only the last one is used.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Internal CSS</title>
<style >
.red{
  color: red;
}
.thick{
  font-size:20px;
}
.green{
  color:green;
}
</style>
</head>
<body>
<p class="red">This is red</p>
<p class="thick">This is thick</p>
<p class="green">This is green</p>
<p class="thick green">This is thick and green</p>
</body>
</html>
```



External Styles

- An external style sheet is ideal when the style is applied to many pages.
- With an external style sheet, you can change the look of an entire Web site by changing one file.
- Each page must link to the style sheet using the <link> tag.
- The <link> tag goes inside the head section:

External Styles

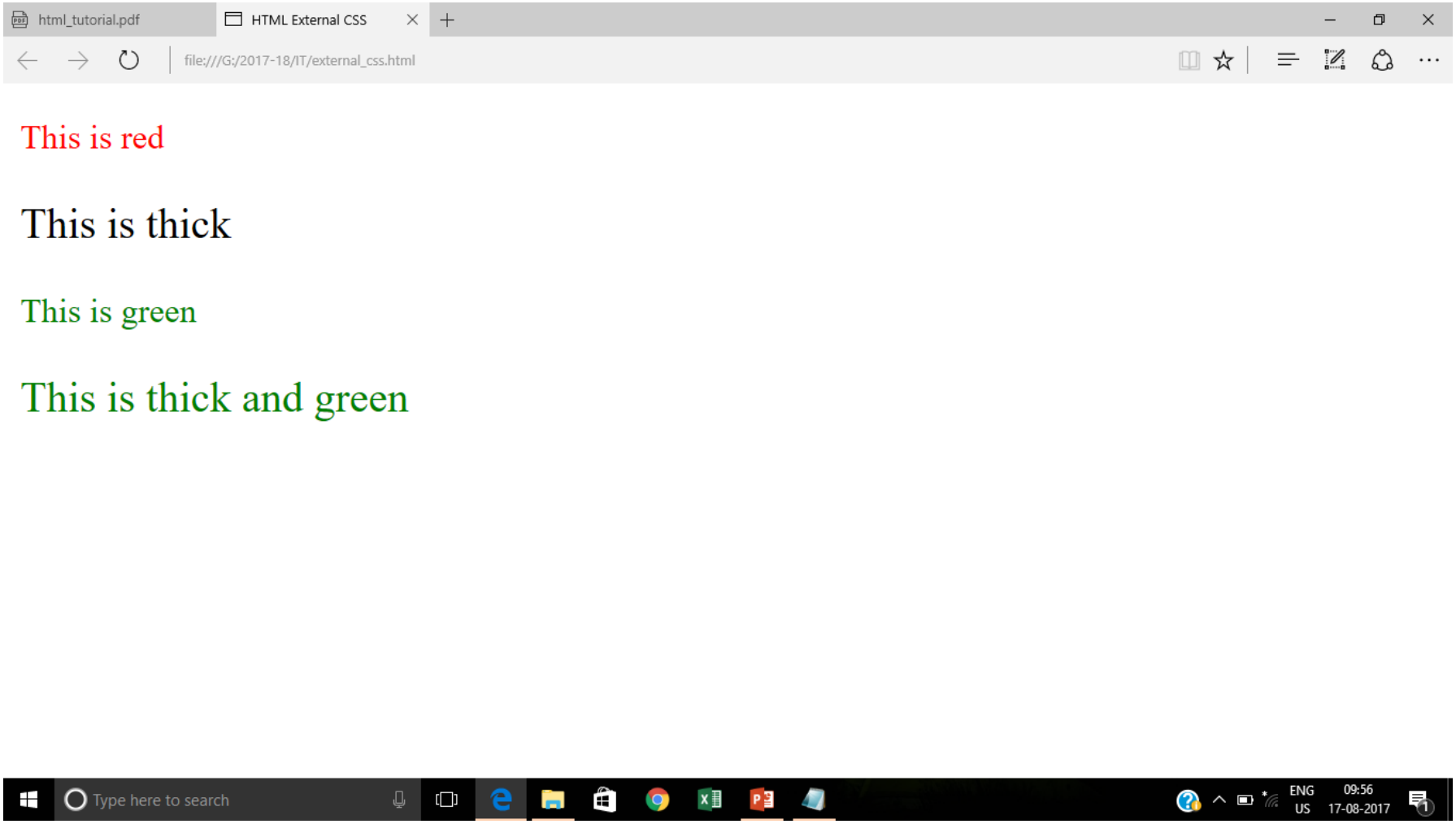
- <head>
  <link rel="stylesheet" type="text/css" href="mystyle.css">
  </head>
- An external style sheet can be written in any text editor. The file should not contain any html tags. Your style sheet should be saved with a .css extension.
- An example of a style sheet file is shown below:
- hr {color:sienna;}
  p {margin-left:20px;}
  body {background-image:url("images/back40.gif");}

External_css.html

style.css

```
.red{
  color: red;
}
.thick{
  font-size:20px;
}
.green{
  color:green;
}
```

```
<!DOCTYPE html>
<html>
<head>
<title>HTML External CSS</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<p class="red">This is red</p>
<p class="thick">This is thick</p>
<p class="green">This is green</p>
<p class="thick green">This is thick and green</p>
</body>
</html>
```

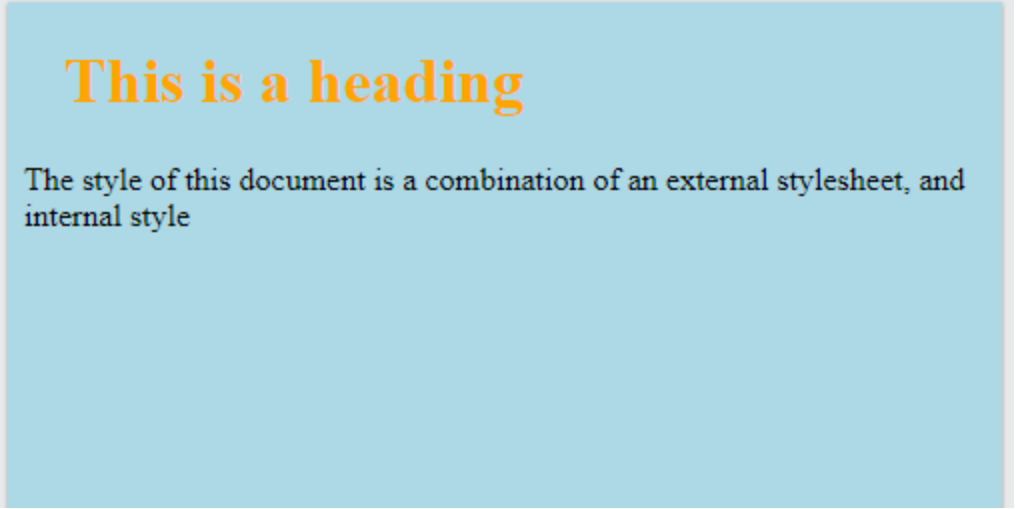

html_tutorial.pdf
HTML External CSS
file:///G:/2017-18/IT/external_css.html
This is red
This is thick
This is green
This is thick and green
Type here to search
ENG
US
09:56
17-08-2017

CSS Rules Overriding

- Any inline style sheet takes the highest priority. So, it will override any rule defined in <style>...</style> tags or the rules defined in any external style sheet file.
- Any rule defined in <style>...</style> tags will override the rules defined in any external style sheet file.
- Any rule defined in the external style sheet file takes the lowest priority, and the rules defined in this file will be applied only when the above two rules are not applicable.

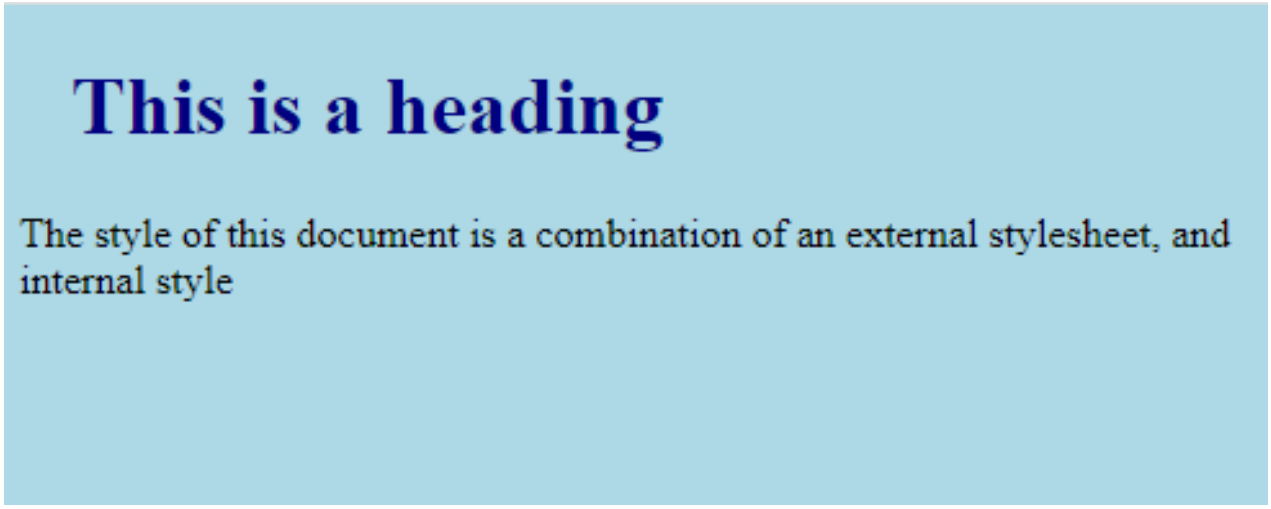
- If the internal style is defined **after** the link to the external style sheet, the <h1> elements will be "orange":

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.c
ss">
<style>
h1 {
  color: orange;
}
</style>
</head>
```



- However, if the internal style is defined **before** the link to the external style sheet, the <h1> elements will be "navy":

```
<head>
<style>
h1 {
  color: orange;
}
</style>
<link rel="stylesheet" type="text/css" href="mystyl
e.css">
</head>
```



Text formatting properties

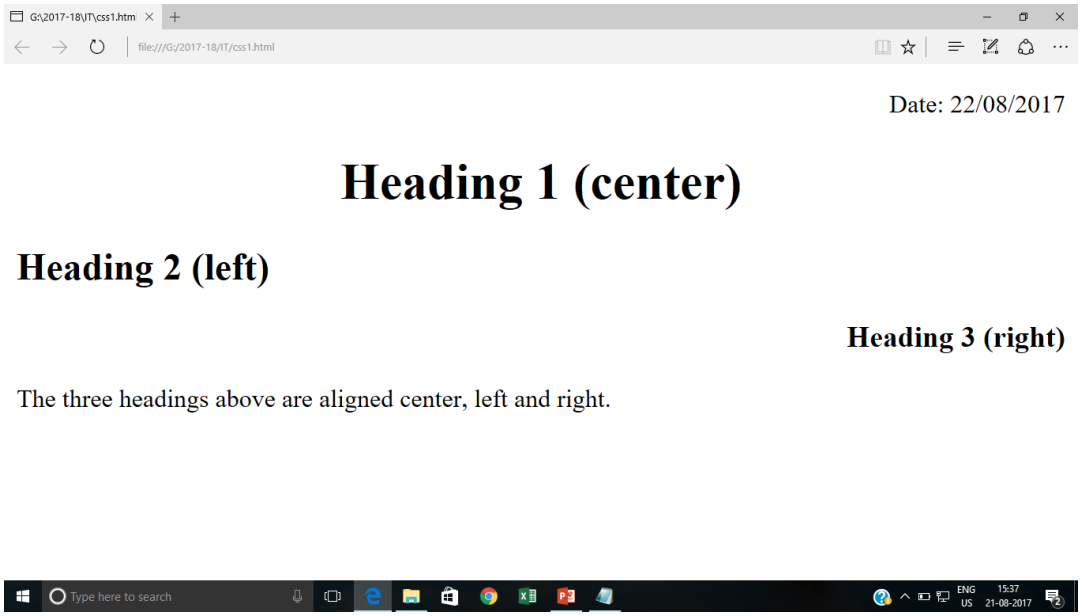
- The text can be styled with some of the text formatting properties.
- The heading uses the text-align, text-transform, and color properties.
- The paragraph is indented, aligned, and the space between characters is specified.
- **Text Color**
- The color property is used to set the color of the text.
- With CSS, a color is most often specified by:
- HEX value - like "#ff0000"
- RGB value - like "rgb(255,0,0)"
- Color name - like "red”
- The default color for a page is defined in the body selector.
- **body {color:blue;}**
  h1 {color:#00ff00;}
  h2 {color:rgb(255,0,0);}

Text Alignment

- The **text-align** property is used to set the horizontal alignment of a text.
- Text can be centered, or aligned to the left or right, or justified.
- When text-align is set to "justify", each line is stretched so that every line has equal width, and the left and right margins are straight (like in magazines and newspapers).

```
h1 {text-align:center;}
p.date {text-align:right;}
p.main {text-align:justify;}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
    text-align: center;
}
h2 {     text-align: left; }
h3 {
    text-align: right;
}
p.date {text-align:right;}
</style>
</head>
<body>
<p class="date"> Date: 22/08/2017 </p>
<h1>Heading 1 (center)</h1>
<h2>Heading 2 (left)</h2>
<h3>Heading 3 (right)</h3>
<p>The three headings above are aligned center, left and right.</p>
</body>
```

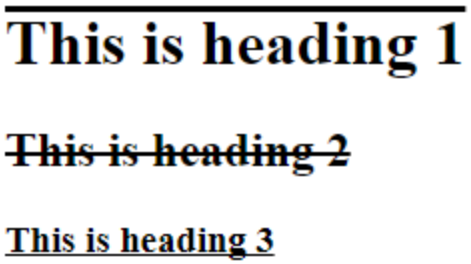


Text Decoration

- The **text-decoration** property is used to set or remove decorations from text.
- The value text-decoration: none;is mostly used to remove underlines from links for design purposes

```
a {text-decoration:none;}

h1 {text-decoration:overline;}
h2 {text-decoration:line-through;}
h3 {text-decoration:underline;}
```

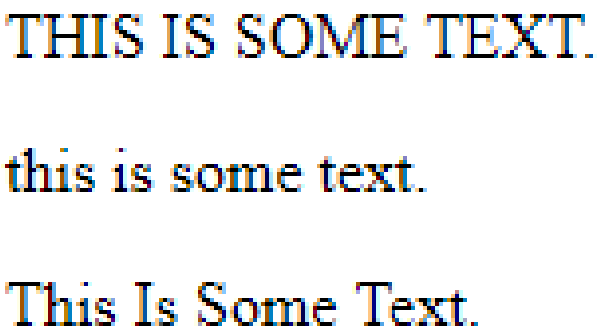


Text Transformation

- The t**ext-transform** property is used to specify uppercase and lowercase letters in a text.
- It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word.

- ```
  p.uppercase {
      text-transform: uppercase;
  }
  p.lowercase {
      text-transform: lowercase;
  }
  p.capitalize {
      text-transform: capitalize;
  }
  ```
  ```

```
<!DOCTYPE html>
<html>
<head>
<style>
p.uppercase {
    text-transform: uppercase;
}
p.lowercase {
    text-transform: lowercase;
}
p.capitalize {
    text-transform: capitalize;
}
</style>
</head>
<body>
<p class="uppercase">This is some text.</p>
<p class="lowercase">This is some text.</p>
<p class="capitalize">this is some text.</p>
</body>
</html>
```



Text Indentation

- The **text-indent** property is used to specify the indentation of the first line of a text.

Ex: p {
    text-indent: 50px;
}

- The l**etter-spacing** property is used to specify the space between the characters in a text.

Ex: h1 {
    letter-spacing: 3px;
}

- The **Text Direction** property is used to change the text direction of an element:

Ex:
```
p {
   direction: rtl;
}
```

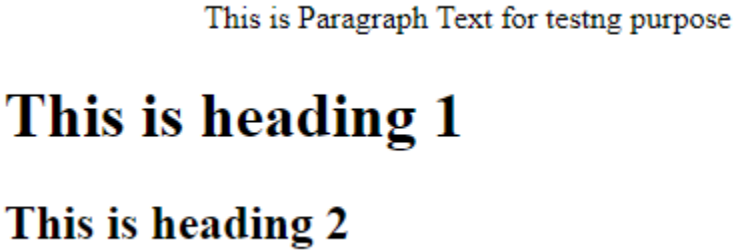
- The **Line Height** property is used to specify the space between lines:

Ex:
```
p.small {
   line-height: 0.8;
}

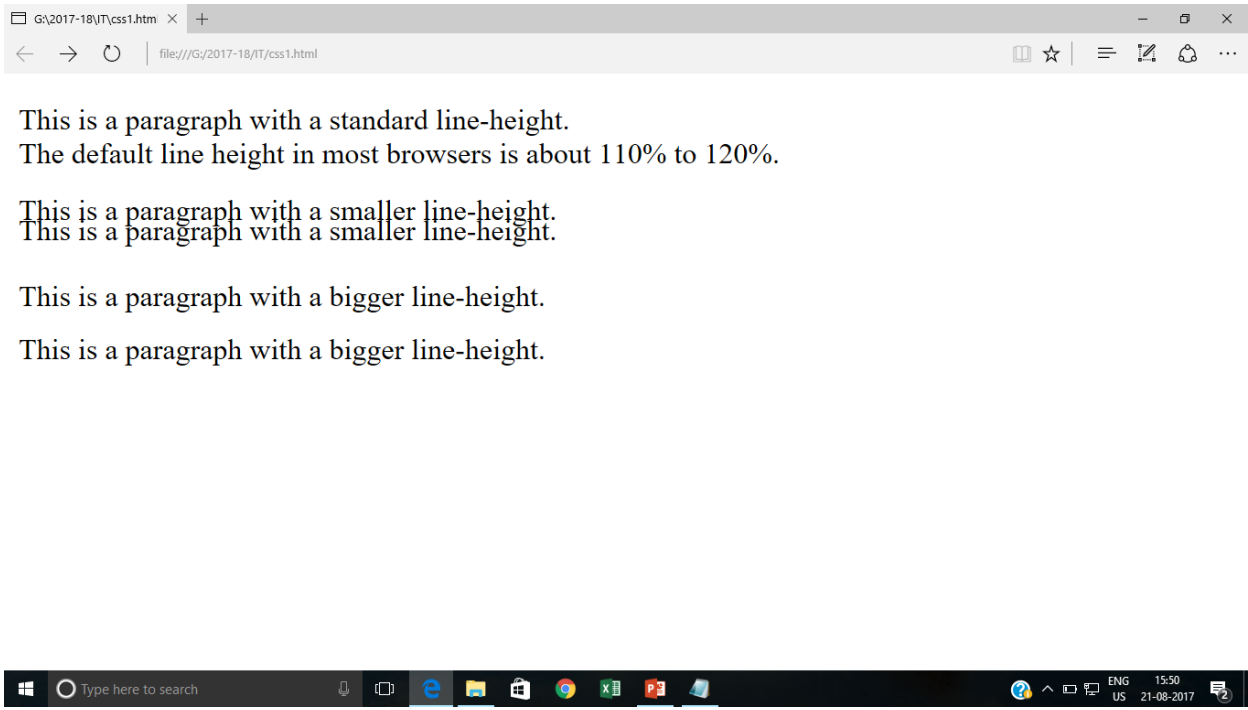
p.big {
   line-height: 1.8;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
    text-indent:100px;
}
h1 {
    letter-spacing: 3px;
}

h2 {
    letter-spacing: -3px;
}
</style>
</head>
<body>
<p>This is Paragraph Text for testng purpose</p>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
</body>
</html>
```



```
<!DOCTYPE html>
<html>
<head>
<style>
p.small {
   line-height: 0.7;
}
p.big {
   line-height: 1.8;
}
</style>  </head> <body>
<p> This is a paragraph with a standard line-height.<br>
The default line height in most browsers is about 110% to 120%.<br> </p>
<p class="small"> This is a paragraph with a smaller line-height.<br>
This is a paragraph with a smaller line-height.<br> </p>
<p class="big"> This is a paragraph with a bigger line-height.<br>
This is a paragraph with a bigger line-height.<br> </p>
</body>
</html>
```



CSS Border

- The CSS border properties allow you to specify the style and color of an element's border.
- Border Style
- The border-style property specifies what kind of border to display.
- None of the border properties will have ANY effect unless the **border-style** property is set!

The following values are allowed:

•dotted - Defines a dotted border
•dashed - Defines a dashed border
•solid - Border is a single solid line.
•double - Border is two solid lines.
•groove - Defines a 3D grooved border. The effect depends on the border-color value
Border looks as though it is carved into the page.
•ridge - Defines a 3D ridged border. The effect depends on the border-color value
order looks the opposite of groove.
•inset - Defines a 3D inset border. The effect depends on the border-color value
Border makes the box look like it is embedded in the page.
•outset - Defines a 3D outset border. The effect depends on the border-color value
Border makes the box look like it is coming out of the canvas.
•none - Defines no border
•hidden - Defines a hidden border

- border-style property can have from one to four values (for the top border, right border, bottom border, and the left border).

CSS Border

```
<html>
<head>
<style>
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {border-style: solid;}
p.double {border-style: double;}
p.groove {border-style: groove;}
p.ridge {border-style: ridge;}
p.inset {border-style: inset;}
p.outset {border-style: outset;}
p.none {border-style: none;}
p.hidden {border-style: hidden;}
p.mix {border-style: dotted dashed solid double;}
</style>
</head>
<body>
<h2>The border-style Property</h2>
<p>This property specifies what kind of borde
to display:</p>

<p class="dotted">A dotted border.</p>
<p class="dashed">A dashed border.</p>
<p class="solid">A solid border.</p>
<p class="double">A double border.</p>
<p class="groove">A groove border.</p>
<p class="ridge">A ridge border.</p>
<p class="inset">An inset border.</p>
<p class="outset">An outset border.</p>
<p class="none">No border.</p>
<p class="hidden">A hidden border.</p>
<p class="mix">A mixed border.</p>
</body>
</html>
```

The border-style Property

This property specifies what kind of border to display:

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border.

A ridge border.

An inset border.

An outset border.

No border.

A hidden border.

A mixed border.

Border Width

- The **border-width** property is used to set the width of the border.
- The width can be set as a specific size (in px, pt, cm, em, etc) or by using one of the three pre-defined values: thin, medium, or thick.
- **Note:** The "border-width" property does not work if it is used alone. Use the "border-style" property to set the borders first.
- p.one
  {
  border-style:solid;
  border-width:5px;
  }
  p.two
  {
  border-style:solid;
  border-width:medium;
  }

Border Color

- The border-color property is used to set the color of the border. The color can be set by:name ,RGB,Hex
- You can also set the border color to "transparent".
- **Note:** The "border-color" property does not work if it is used alone. Use the "border-style" property to set the borders first.
- border-color property can have from one to four values (for the top border, right border, bottom border, and the left border).
- p.one
  {
  border-style:solid;
  border-color:red;
  }
  p.two
  {
  border-style:solid;
  border-color:#98bf21;
  }

Border - Individual sides

- In CSS, there are also properties for specifying each of the borders (top, right, bottom, and left):

- ```
  p {
      border-top-style: dotted;
      border-right-style: solid;
      border-bottom-style: dotted;
      border-left-style: solid;
  }
  ```

- ```
  p {
      border-style: dotted solid;
  }
  ```

- The border-style property can have from one to four values.
- **border-style:dotted solid double dashed;**
  - top border is dotted
  - right border is solid
  - bottom border is double
  - left border is dashed

- **border-style:dotted solid double;**
  - top border is dotted
  - right and left borders are solid
  - bottom border is double
- **border-style:dotted solid;**
  - top and bottom borders are dotted
  - right and left borders are solid

- **border-style:dotted;**
  - all four borders are dotted

Border - Shorthand property

- The border property is a shorthand for the following individual border properties:
- **border-width**
- **border-style (required)**
- **border-color**

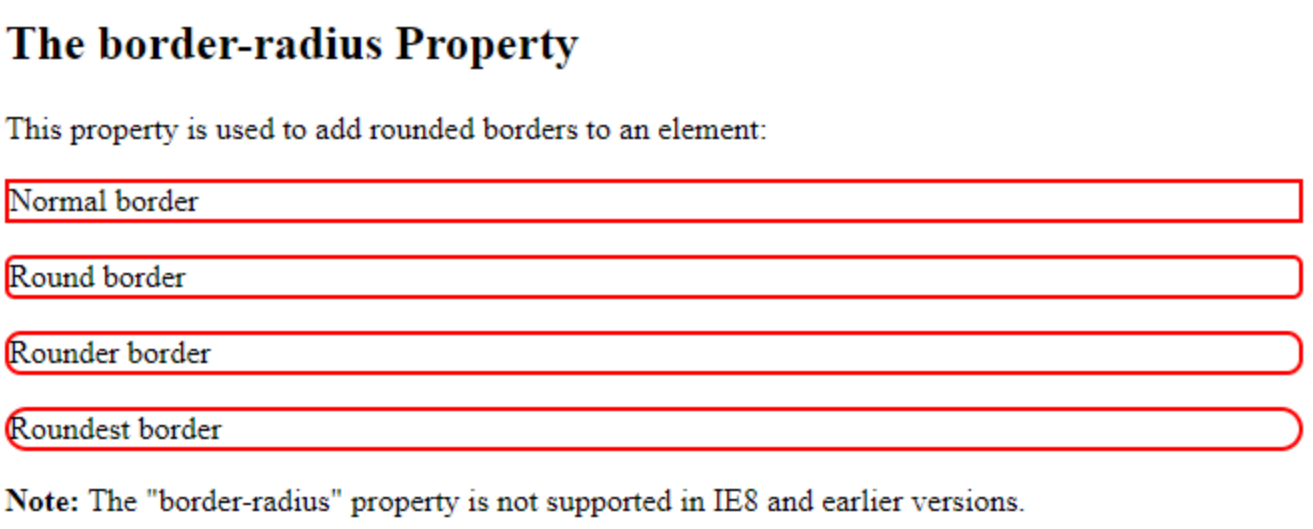
- border:5px solid red;

Rounded Borders

Rounded Borders property is used to add rounded borders to an element:

```
<html>    <head>
<style>
p.normal {
    border: 2px solid red;
}
p.round1 {
    border: 2px solid red;
    border-radius: 5px;
}
p.round2 {
    border: 2px solid red;
    border-radius: 8px;
}
p.round3 {
    border: 2px solid red;
    border-radius: 12px;
}
</style>    </head>     <body>
```

```
<body>
<h2>The border-radius Property</h2>
<p>This property is used to add rounded borders to an element:</p>
<p class="normal">Normal border</p>
<p class="round1">Round border</p>
<p class="round2">Rounder border</p>
<p class="round3">Roundest border</p>
<p><b>Note:</b> The "border-radius" property is not supported in IE8 and earlier versions.</p>
</body>
```



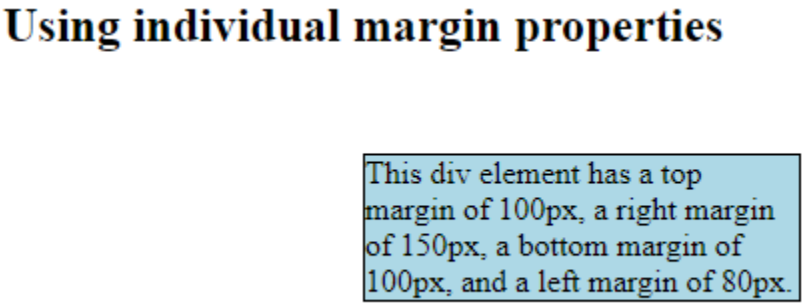
Margins

- CSS Margins properties are used to generate space around elements.
- Margin properties set the size of the white space outside the border.
- There are CSS properties for setting the margin for each side of an element (top, right, bottom, and left).
- CSS has properties for specifying the margin for each side of an element:
  - `margin-top`
  - `margin-right`
  - `margin-bottom`
  - `margin-left`

```
<html>
<head>
<style>
div {
   border: 1px solid black;
   margin-top: 50px;
   margin-bottom: 100px;
   margin-right: 250px;
   margin-left: 180px;
   background-color: lightblue;
}
</style>
</head>
<body>
<h2>Using individual margin properties</h2>
<div>This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.</div>
</body>
</html>
```

1.margin: 50px 100px 150px 200px;

Margin: 50px 100px



```
<html>
<head>
<style>
ul.a {
  list-style-type: circle;
}

ul.b {
  list-style-type: square;
}

ol.c {
  list-style-type: upper-roman;
}

ol.d {
  list-style-type: lower-alpha;
}
</style>
</head>
<body>

<h2>The list-style-type Property</h2>

<p>Example of unordered lists:</p>
<ul class="a">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>

<ul class="b">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>
```

```
<p>Example of ordered lists:</p>
<ol class="c">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>

<ol class="d">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>

</body>
</html>
```

The list-style-type Property

Example of unordered lists:

- Coffee
- Tea
- Coca Cola

- Coffee
- Tea
- Coca Cola

Example of ordered lists:

I. Coffee
II. Tea
III. Coca Cola

a. Coffee
b. Tea
c. Coca Cola

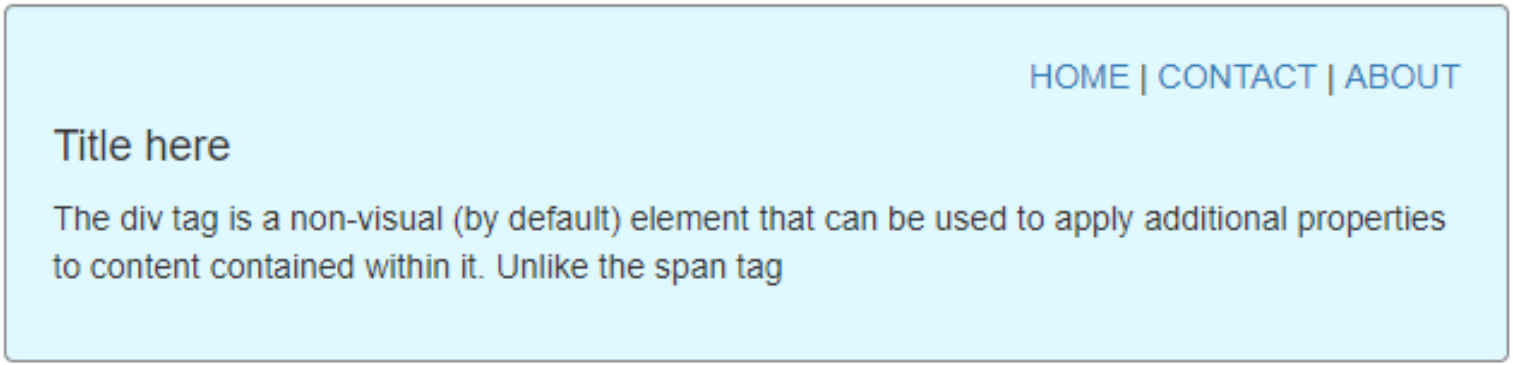
Div Tag

- The tag defines a division or a section in an HTML document.
- The tag is used to group block-elements to format them with CSS.

- **Tip:** The element is very often used together with CSS, to layout a web page.
- **Note:** By default, browsers always place a line break before and after the element. However, this can be changed with CSS.
- The tag is nothing more than a container unit that encapsulates other page elements and divides the HTML document into sections.
- Web developers use elements to group together HTML elements and apply CSS styles to many elements at once.
- For instance, by wrapping a set of paragraph elements into a element, the developer can take advantage of CSS styles and apply a font to all paragraphs at once by applying a font style to the tag instead of coding the same style for each paragraph element.

- The div tag is not more than a container for other tags. Here are some of the div's attributes:
- **id**
- **class**
- **title**
- **style**
- **height**
- **width**

```
<div id="my_menu" align="right">
 <a href="#">HOME</a> | <a href="#">CONTACT</a> | <a href="#">ABOUT</a> </div>
 <div id="my_content" align="left”>
<h4>Title here</h4>
<p>The div tag is a non-visual (by default) element that can be used to apply additional properties to content contained within it. Unlike the span tag</p>
</div>
```



Span Tag

- The HTML tag is used for grouping and applying styles to inline elements.
- Difference between the span tag and the div tag.

div tag	span tag
HTML div is a **block** element.	HTML span is an **inline** element
HTML div element is used to **wrap large sections of elements**.	HTML span element is used to **wrap small portion of texts, image** etc.

```
<!DOCTYPE html>
<html>
 <head>
  <title>HTML span Tag</title>
 </head>
 <body>
  <p>This is a paragraph <span style = "color:#FF0000;">       This is a paragraph</span>This is a paragraph</p>
   <p><span style = "color:#8866ff;">       This is another paragraph</span></p>
 </body>
</html>
```

This is a paragraph This is a paragraphThis is a paragraph

This is another paragraph

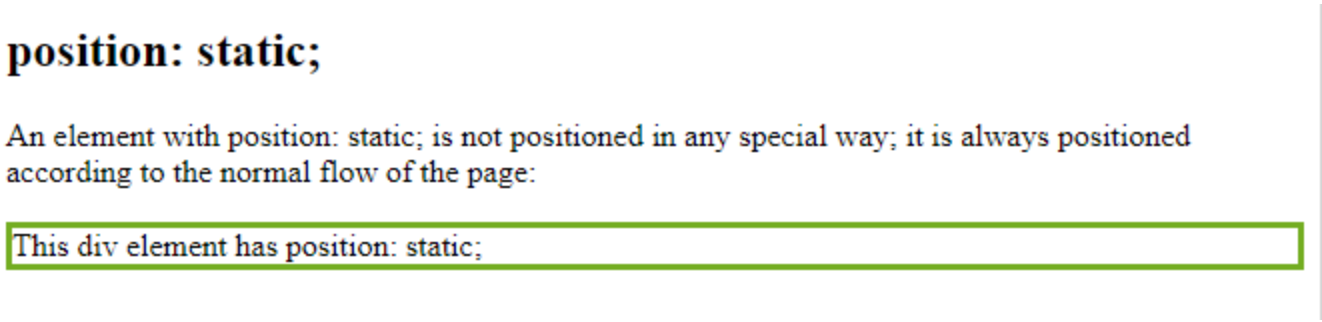
Positioning

- The **CSS positioning** properties allow you to position an element.
- It can also place an element behind another, and specify what should happen when an element's content is too big.
- useful for scripted animation effect.
- Elements can be positioned using the top, bottom, left, and right properties.
- However, these properties will not work unless the position property is set first. They also work differently depending on the positioning method.
- There are four different positioning methods.
  - static
  - relative
  - fixed
  - absolute

Positioning

- **Static Positioning**
  - HTML elements are positioned static by default.
  - A static positioned element is always positioned according to the normal flow of the page.
  - Static positioned elements are not affected by the top, bottom, left, and right properties.
  - Ex:

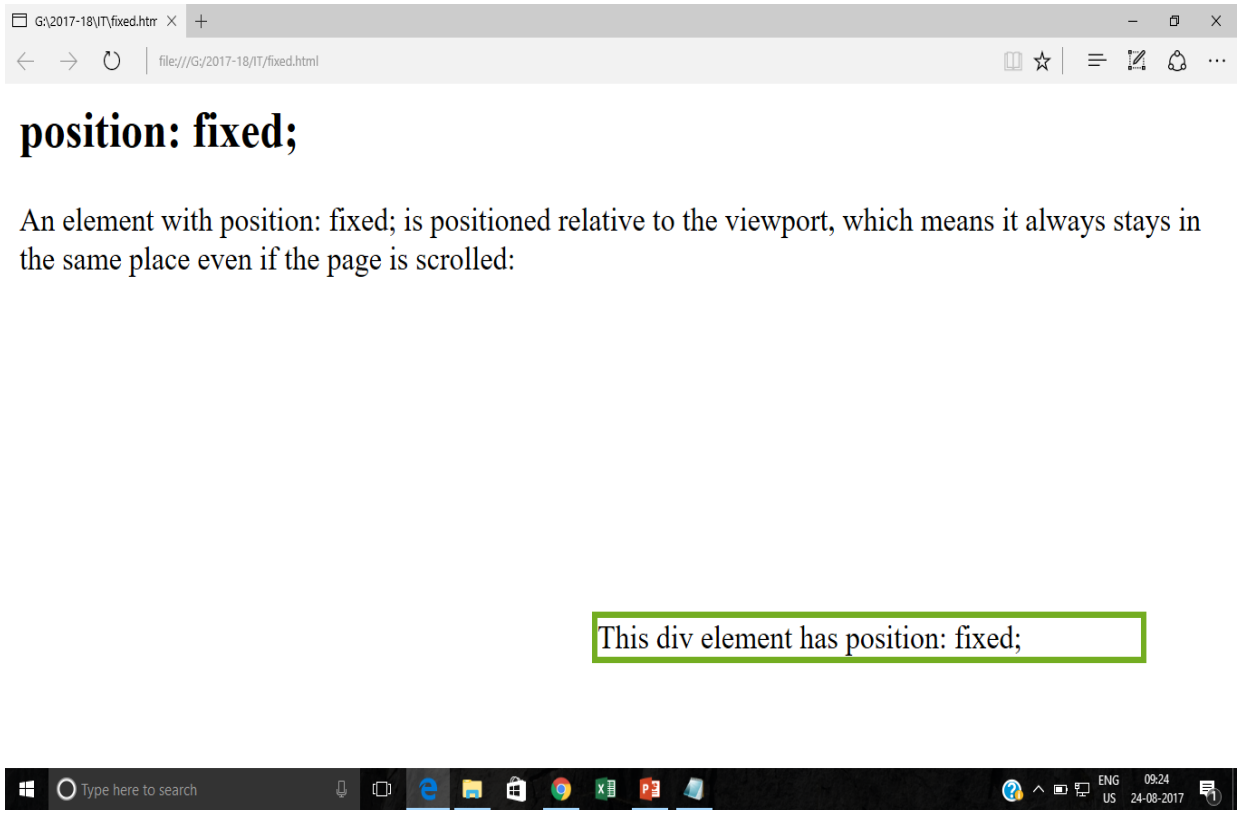
```
div.static {
position: static;
border: 3px solid #73AD21;
}
```




```
<html>
<head>
<style>
div.static {
   position: static;
   border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: static;</h2>
<p>An element with position: static; is not positioned in any special way; it is
always positioned according to the normal flow of the page:</p>
<div class="static">
This div element has position: static;
</div>
</body>
</html>
```

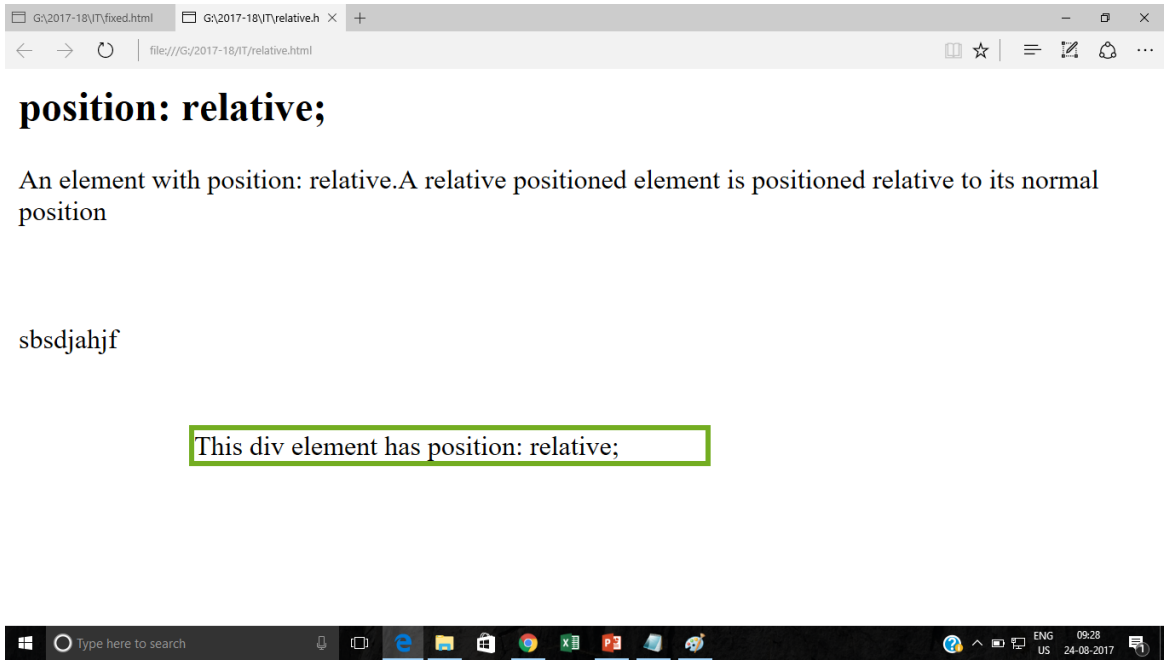
- **Fixed Positioning**
  - An element with fixed position is positioned relative to the browser window.
  - It will not move even if the window is scrolled.
  - The top, right, bottom, and left properties are used to position the element.
  - Fixed positioned elements are removed from the normal flow. The document and other elements behave like the fixed positioned element does not exist.
  - Fixed positioned elements can overlap other elements.
  - div.fixed {
    position: fixed;
    bottom: 0;
    right: 0;
    width: 300px;
    border: 3px solid #73AD21;
    }

```
<html>
<head>
<style>
div.fixed {
    position: fixed;
    bottom: 50;
    right: 50;
    width: 300px;
    border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: fixed;</h2>
<p>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>
<div class="fixed">
This div element has position: fixed;
</div>
</body>
```



- **Relative Positioning**
  - A relative positioned element is positioned relative to its normal position.
  - The content of relatively positioned elements can be moved and overlap other elements, but the reserved space for the element is still preserved in the normal flow.
  - div.relative {
    position: relative;
    left: 30px;
    border: 3px solid #73AD21;
    }
  - You should not use top and bottom together. If you do, the top overrides the bottom.
  - Similarly, you should not use left and right together. The left overrides right.

```
<html>
<head>
<style>
div.relative {
   position: relative;
   top:100;
left:100;
   width: 300px;
   border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: relative;</h2>
<p>An element with position: relative. A relative positioned element is positioned relative to its normal position</p>
<div class="relative">
This div element has position: relative;
</div>
<p> sbsdjahjf</p>
</body>
</html>
```

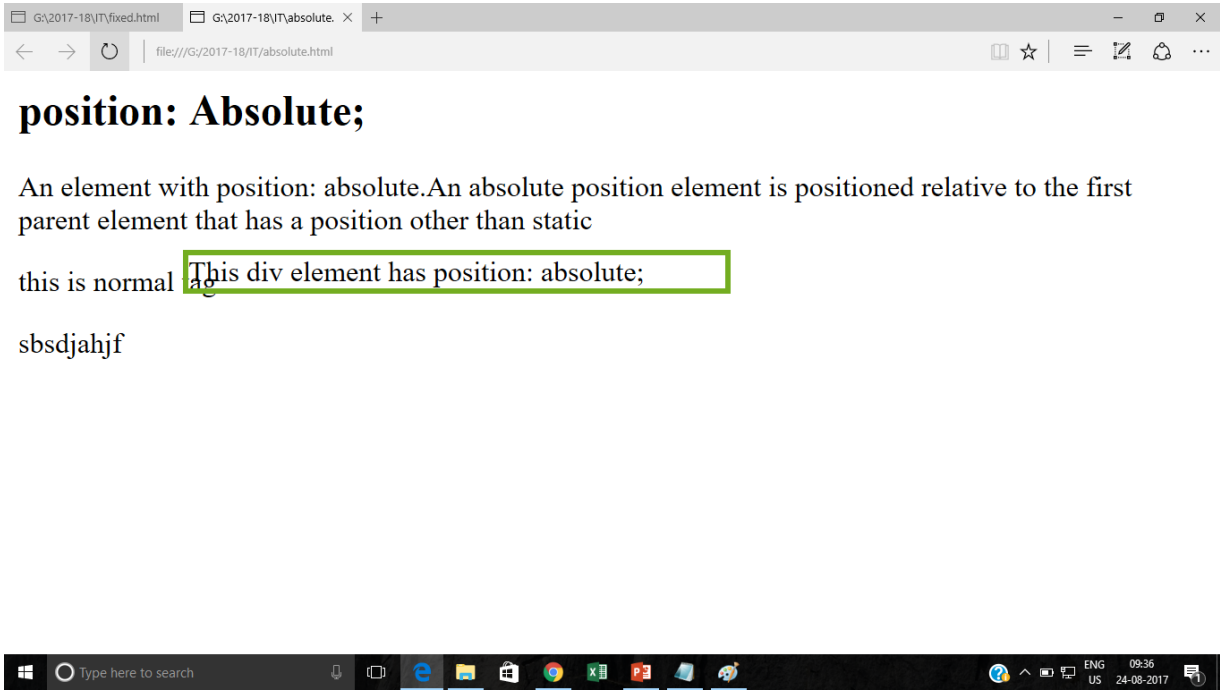


- **Absolute Positioning**
- position:absolute is like position:fixed, both makes the element go into its own layer.
- position:fixed's offset is relative to the window.
- position:absolute's offset is relative to its “containing block”.
- A “containing block” is effectively the first parent element that has a position value other than “static”.
- When no parent has any of “position” spec, then it is relative to the <html> element.

```
div.absolute {
    position: absolute;
    top: 80px;
    right: 0;
    width: 200px;
    height: 100px;
    border: 3px solid #73AD21;
}
```

- You should not use top and bottom together. If you do, the top overrides the bottom. Similarly, you should not use left and right together. The left overrides right.

```
<html>
<head>
<style>
div.abs {
   position:absolute;
   top:100;
   left:100;
   width: 300px;
   border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: Absolute;</h2>
<p>An element with position: absolute.An absolute position element is positioned relative to the first parent element that has a position other than static</p>
<p> this is normal tag</p>
<div class="abs">
This div element has position: absolute;
</div>
<p> sbsdjahjf</p>
</body>
```



Summery

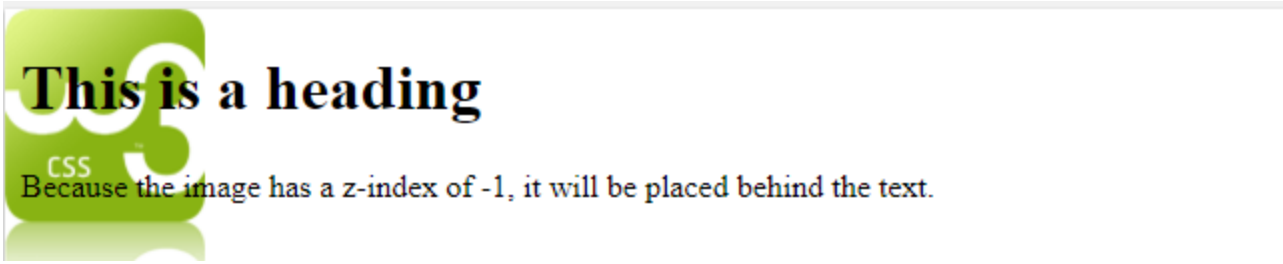
position Value	Own Layer	Relative To
static	no	(Normal Flow)
relative	no	Relative to its normal position
fixed	yes	Relative to window
absolute	yes	Relative to its containing block. That is, the first parent that's not positioned static. If none found, then it's <html> tag.

- **Overlapping Elements**
  - When elements are positioned outside the normal flow, they can overlap other elements.
  - The z-index property specifies the stack order of an element (which element should be placed in front of, or behind, the others).
  - An element can have a positive or negative stack order,0 is default value of z-index
  - The larger the value, the more it is on top.

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
    position: absolute;
    left: 0px;
    top: 0px;
    z-index: -1;
}
</style>
</head>
<body>

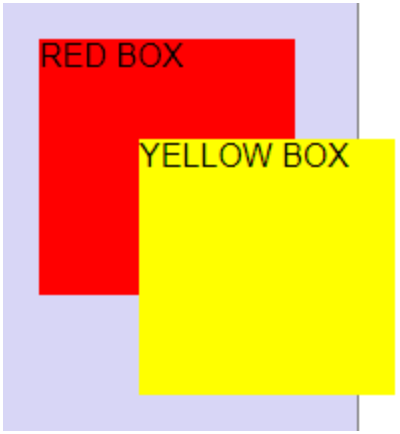
<h1>This is a heading</h1>
<img src="w3css.gif" width="100" height="140">
<p>Because the image has a z-index of -1, it will be placed behind the text.</p>

</body>
</html>
```



```
<div style="
width:128px;
height:128px;
position:fixed;
bottom:350px;
right:350px;
z-index:5;
background-color:red
">
RED BOX
</div>

<div style="
width:128px;
height:128px;
position:fixed;
bottom:300px;
right:300px;
z-index:6;
background-color:yellow
">
YELLOW BOX
</div>
```



- In CSS , we use color values for specifying the color. We can also use this property for the border-color and other decorative effects.
- We can define the color of an element by using the following ways:
- RGB format.
- RGBA format.
- Hexadecimal notation.
- HSL.
- HSLA.
- Built-in color.

Built-in Color

- As its name implies, built-in color means the collection of previously defined colors that are used by using a name such as red, blue, green, etc.
- **Syntax**: color: color-name;

RGB Format

- RGB format is the short form of '**RED GREEN** and **BLUE**' that is used for defining the color of an HTML
- element simply by specifying the values of R, G, B that are in the range of 0 to 255.
- The color values in this format are specified by using the **rgb()** property. This property allows three values that can either be in percentage or integer (range from 0 to 255).
- This property is not supported in all browsers; that's why it is not recommended to use it.
- **Syntax**

color: rgb(R, G, B);

RGBA color

- RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity /transparency for a color.
- An RGBA color value is specified with: rgba(red, green, blue, alpha). The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque/non transparent).
- Syntax : color:rgba(R, G, B, A);
- ```
  #p1 {background-color: rgba(255, 0, 0, 0.3);}  /* red with opacity */
  #p2 {background-color: rgba(0, 255, 0, 0.3);}  /* green with opacity */
  #p3 {background-color: rgba(0, 0, 255, 0.3);}  /* blue with opacity */
  ```
  ```

Hexadecimal notation

- Hexadecimal can be defined as a six-digit color representation.
- This notation starts with the # **symbol** followed by six characters ranges from **0 to F**.
- In hexadecimal notation, the first two digits represent the **red (RR)** color value, the next two digits represent the **green (GG)** color value, and the last two digits represent the **blue (BB)** color value.
- The black color notation in hexadecimal is #000000, and the white color notation in hexadecimal is #FFFFFF.
- Some of the codes in hexadecimal notation are #FF0000, #00FF00, #0000FF, #FFFF00, and many more.

Short Hex codes

- It is a short form of hexadecimal notation in which every digit is recreated to arrive at an equivalent hexadecimal value.
- For example, #7B6 becomes #77BB66 in hexadecimal.
- The black color notation in short hex is #000, and the white color notation in short hex is #FFF.
- Some of the codes in short hex are #F00, #0F0, #0FF, #FF0, and many more.

HSL Colors

- HSL stands for Hue, Saturation and Lightness.
- An HSL color value is specified with: hsl(hue, saturation, lightness).

1. Hue is a degree on the color wheel (from 0 to 360):
    1. 0 (or 360) is red
    2. 120 is green
    3. 240 is blue
2. Saturation is a percentage value: 100% is the full color.
    1. It takes value in percentage in which 100% represents fully saturated, i.e., no shades of gray, 50% represent 50% gray, but the color is still visible, and 0% represents fully unsaturated, i.e., completely gray, and the color is invisible.
3. Lightness is also a percentage; 0% is dark (black) and 100% is white.
    1. The lightness of the color can be defined as the light that we want to provide the color in which 0% represents black (there is no light), 50% represents neither dark nor light, and 100% represents white (full lightness).
4.
```
#p1 {background-color: hsl(120, 100%, 50%);}  /* green */
#p2 {background-color: hsl(120, 100%, 75%);}  /* light green */
```

HSLA Colors

- HSLA color values are an extension of HSL color values with an alpha channel - which specifies the opacity for a color.
- An HSLA color value is specified with: hsla(hue, saturation, lightness, alpha), where the alpha parameter defines the opacity. The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).
-
```
#p1 {background-color: hsla(120, 100%, 50%, 0.3);}  /* green with opacity */
#p2 {background-color: hsla(120, 100%, 75%, 0.3);}  /* light green with opacity */
```

Opacity

- The CSS opacity property sets the opacity for the whole element (both background color and text will be opaque/transparent).

- The opacity property value must be a number between 0.0 (fully transparent) and 1.0 (fully opaque)

```
#p1 {background-color:rgb(255,0,0);opacity:0.6;}  /* red with opacity */
#p2 {background-color:rgb(0,255,0);opacity:0.6;}  /* green with opacity */
#p3 {background-color:rgb(0,0,255);opacity:0.6;}  /* blue with opacity */
```

The transparent Keyword

- The transparent keyword is used to make a color transparent. This is often used to make a transparent background color for an element.

```
<html>
<head>
<style>
body {
  background-image: url("paper.gif");
}
div.ex1 {
  background-color: lightgreen;
  border: 2px solid black;
  padding: 15px;
}
div.ex2 {
  background-color: transparent;
  border: 2px solid black;
  padding: 15px;
}
</style>
</head>
<body>
```

Units

- There are 2 types of length units: “relative” and “absolute”.
- **Relative Units**
- **Relative units** → a length relative to some element's length. (for example, with respect to inherited value or browser default).
- 5rem
- 5em
- 5ex
- 5ch
- 5%
- The default font size in browsers is 16px. That is, by default, 1rem is 16px.

unit	meaning
em	Length equal to the current font size. (see detail below) (Note: it is NOT the width of “M”.)
ex	A font's “x-height”, usually the height of the letter x. Typically about half of em.
rem(CSS3)	Length equal to the computed value of font-size on the root element. When specified on the font-size property of the root element, the rem units refer to the property's initial value.
ch(CSS3)	Length roughly equal to the width of the 0 (ZERO) character in the current font. Roughly half of em.

Absolute Units

Absolute units → fixed in relation to each other.

1cm

10mm

2.54cm

1in

72pt

6pc

96px

There are 2 types of absolute units.

- ① "physical" units {in, cm, mm, pt, pc}.
- ② px

unit	meaning	conversion
in	Inch. 6 pc. 72 pt. 96 px.	2.54 cm
pc	1/6 inch. 12 pt. 16 px. “pc” stands for “pica”.	~0.423 cm
pt	1/72 inch. (~0.0138 inch) ~1.33 px. “pt” stands for “point”.	0.3527 mm

unit	meaning	conversion
cm	centimeter. 10 mm	~0.3937 inch; ~37.795 px
mm	millimeter. 0.1 cm	~3.78 px

unit	meaning	conversion
px	A “reference pixel”. Equivalent to 0.75pt. (or, think of 96dpi screen.)	0.2645 mm