

CSS (Cascading Style Sheets)

CSS stands for cascading style sheets. It was first developed in 1997, as a way for Web developers to define the **look and feel** of their Web pages. It was intended to allow developers to separate content from design and layout so that HTML could perform more of the function without worry about the design and layout. It is used to separate style from content. It is used to control the style of a web document in a simple and easy way.

Importance of CSS

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. The following are some of the key advantages of learning CSS:

- Create Stunning Web site - 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, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.
- Become a web designer - If you want to start a career as a professional web designer, HTML and CSS designing is a must skill.
- Control web - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.
- Learn other languages - Once you understand the basic of HTML and CSS then other related technologies like JavaScript, php, or angular are become easier to understand.

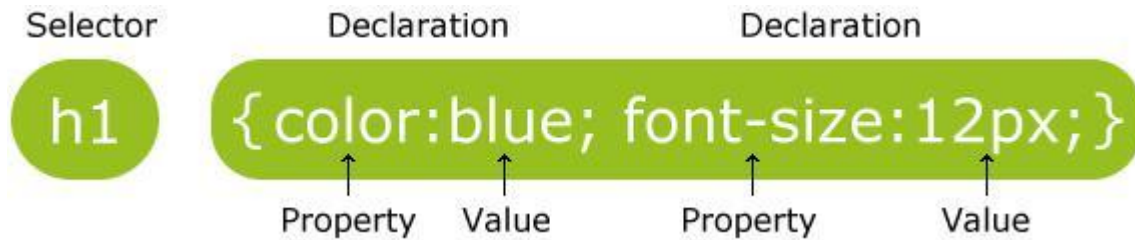
Applications of CSS

As mentioned before, CSS is one of the most widely used style language over the web. The following are the list few of them here:

- CSS saves time - You can write CSS once and then reuse 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 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 cell phones 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 to future browsers.

Syntax

A CSS rule has two main parts: a *selector* and one or more *declarations*. **Selector** is normally the HTML element you want to style and each **declaration** consists of a *property* and *value*. The **property** is the style attribute we want to use and each property has a **value** associated with it.



Example:

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

Here in this example p refers to the paragraph and the color red is assign to the paragraph with the text align center

Inserting CSS

We can use style sheets in three different ways in our HTML document. They are **external style sheet**, **internal style sheet** and **inline style**.

External Style Sheet

If we want to apply the same style to many pages, we use external style sheet. With an external style sheet, you can change the look of an entire Web site by changing one style sheet file. Each page must link to the style sheet using the **<link>** tag. The **<link>** tag goes inside the head section.

Example:

```
<html>
<head>
  <link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
</html>
```

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;
}
```

/*Note: Do not leave space between property value and units*/

```
body {
  background-image:url("images/back40.gif");
}
```

Internal Style Sheet

If you want a unique style to a single document, an internal style sheet should be used. You define internal styles in the head section of an HTML page, by using the `<style>` tag.

Example:

```
<head>
  <style type="text/css">
    hr {
      color:red;
    }
    p {
      margin-left:20px;
    }
    body {
      background-image:url("images/back40.gif");
    }
  </style>
</head>
```

Inline Styles

If you want a unique style to a single element, an inline style sheet should be used. An inline style loses many of the advantages of style sheets by mixing content with presentation. To use inline styles you use the **style attribute** in the relevant tag. The style attribute can contain any CSS property.

Example:

```
<p style="color:yellow;margin-left:20px">This is a paragraph.</p>
```

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 `*/`.

Example:

```
<style>
  /* body {
    background-image:url("images/back40.gif");
  } */
</style>
```

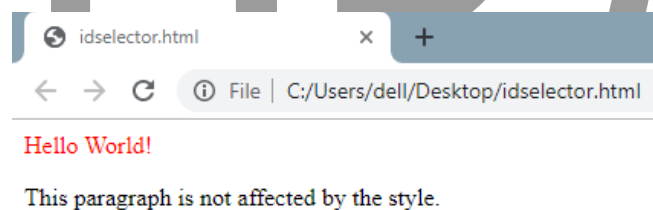
Id and Class Selectors

The **id** selector is used to specify a style for a single, unique element. The id selector uses id attribute of the HTML element and is defined with “#”.

Example:

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

Output:

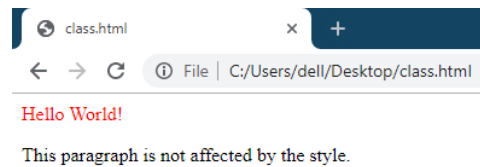


The **class** selector is used to specify a style for a group of elements. Unlike the id selector, the class selector is most often used on several elements. This allows you to set a particular style for any HTML elements with the same class. The class selector uses the HTML class attribute, and is defined with (dot symbol) ".".

Example:

```
<html>
<head>
  <style type="text/css">
    .para
    {
      text-align:left;
      color:red;
    }
  </style>
</head>
<body>
  <p class="para">Hello World!</p>
  <p>This paragraph is not affected by the style.</p>
</body>
</html>
```

Output:

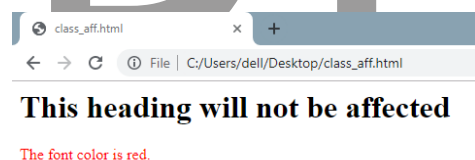


You can also specify that only specific HTML elements should be affected by a class.

Example:

```
<html>
  <head>
    <style type="text/css">
      p.center
      {
        color:red;
      }
    </style>
  </head>
  <body>
    <h1 class="center">This heading will not be affected</h1>
    <p class="center">The font color is red.</p>
  </body>
</html>
```

Output:



Multiple Styles Will Cascade into One

Styles can be specified:

- inside an HTML element
- inside the head section of an HTML page
- in an external CSS file

Cascading order

What style will be used when there is more than one style specified for an HTML element?

Generally speaking, we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number four has the highest priority:

- Browser default
- External style sheet
- Internal style sheet (in the head section)
- Inline style (inside an HTML element)

So, an inline style (inside an HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or in a browser (a default value).

Note: If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will override the internal style sheet!

DRAFT

CSS Background

Background properties are used to define the background effects of an HTML element. CSS properties used to define background effects are: **background-color**, **background-image**, **background-repeat**, **background-attachment**, and **background-position**.

Background Image

The background-image property specifies an image to use as the background of an element. By default, the image is repeated so it covers the entire element.

The background image for a page can be set like this:

```
body
{
  background-image:url('paper.gif');
}
```

Background Image - Repeat Horizontally or Vertically

By default, the background-image property repeats an image both horizontally and vertically. Some images should be repeated only horizontally or vertically, or they will look strange, like this:

Example:

```
body
{
  background-image:url('gradient2.png');
}
```

If the image is repeated only horizontally (repeat-x), the background will look better:

Example:

```
body
{
  background-image:url('gradient2.png');
  background-repeat:repeat-x;
}
```

Background Image - Set position and no-repeat

When using a background image, use an image that does not disturb the text. Showing the image only once is specified by the background-repeat property:

Example

```
body
{
  background-image:url('img_tree.png');
  background-repeat:no-repeat;
}
```

In the example above, the background image is shown in the same place as the text. We want to change the position of the image, so that it does not disturb the text too much.

The position of the image is specified by the background-position property:

Example:

```
body
{
  background-image:url('img_tree.png');
  background-repeat:no-repeat;
  background-position:right top;
}
```

Shorthand Property

To shorten the code, it is also possible to specify all the properties in one single property. This is called a shorthand property. The shorthand property for background is simply "background". When using the shorthand property, the order of the property values are: background-color, background-image, background-repeat, background-attachment, and background-position.

Example:

```
body
{
  background:#ffffff url('img_tree.png') no-repeat right top;
}
```

Grouping Selectors

In style sheets there are often elements with the same style.

```
h1
{
  color:green;
}
h2
{
  color:green;
}
p
{
  color:green;
}
```

To minimize the code, you can group selectors. Separate each selector with a comma. In the example below we have grouped the selectors from the code above:

Example

```
h1,h2,p
{
  color:green;
}
```


CSS Borders

The CSS border properties allows to specify how the border of the box representing an element should look. There are three properties of a border that can be changed:

- border-color: specifies the color of a border.
- border-style: specifies whether a border should be solid, dashed line, double line, or one of the other possible values.
- border-width: specifies the width of a border.

The border-color Property

The border-color property allows you to change the color of the border surrounding an element. You can individually change the color of the bottom, left, top and right sides of an element's border using the properties –

- border-bottom-color: changes the color of bottom border.
- border-top-color: changes the color of top border.
- border-left-color: changes the color of left border.
- border-right-color: changes the color of right border.

Example:

```
<html>
<head>
  <style type = "text/css">
    p.example1 {
      border:1px solid;
      border-bottom-color:#009900; /* Green */
      border-top-color:#FF0000; /* Red */
      border-left-color:#330000; /* Black */
      border-right-color:#0000CC; /* Blue */
    }
    p.example2 {
      border:1px solid;
      border-color:#009900; /* Green */
    }
  </style>
</head>
<body>
  <p class = "example1">
    This example is showing all borders in different colors.
  </p>
  <p class = "example2">
    This example is showing all borders in green color only.
  </p>
</body>
</html>
```

Output:

This example is showing all borders in different colors.

This example is showing all borders in green color only.

The border-style Property

The border-style property allows you to select one of the following styles of border:

- none: No border. (Equivalent of border-width:0;)
- solid: Border is a single solid line.
- dotted: Border is a series of dots.
- dashed: Border is a series of short lines.
- double: Border is two solid lines.
- groove: Border looks as though it is carved into the page.
- ridge: Border looks the opposite of groove.
- inset: Border makes the box look like it is embedded in the page.
- outset: Border makes the box look like it is coming out of the canvas.
- hidden: Same as none, except in terms of border-conflict resolution for table elements.

You can individually change the style of the bottom, left, top, and right borders of an element using the following properties:

- border-bottom-style: changes the style of bottom border.
- border-top-style: changes the style of top border.
- border-left-style: changes the style of left border.
- border-right-style: changes the style of right border.

Example:

```
<html>
<head>
  <title>Border Style</title>
</head>
<body>
  <p style = "border-width:1px; border-style:none;">
    I have no border
  </p>
  <p style = "border-width:1px; border-style:solid;">
    I have a solid border
  </p>
  <p style = "border-width:1px; border-style:dashed;">
    I have a dashed border.
  </p>
  <p style = "border-width:1px; border-style:double;">
    I have a double border.
  </p>
  <p style = "border-width:1px; border-style:groove;">
    I have a groove border.
  </p>
  <p style = "border-width:1px; border-style:ridge">
    I have a ridge border.
  </p>
  <p style = "border-width:1px; border-style:inset;">
```

```

    I have a inset border.
</p>
<p style = "border-width:1px; border-style:outset;">
    I have a outset border.
</p>
<p style = "border-width:1px; border-style:hidden;">
    I have a hidden border.
</p>
<p style = "border-width:4px;
    border-top-style:solid;
    border-bottom-style:dashed;
    border-left-style:groove;
    border-right-style:double;">
    I have four different styles of borders.
</p>
</body>
</html>

```

Output:

I have no border

I have a solid border

I have a dashed border.

I have a double border.

I have a groove border.

I have a ridge border.

I have a inset border.

I have a outset border.

I have a hidden border.

I have four different styles of borders.

The border-width Property

The border-width property allows you to set the width of an element borders. The value of this property could be either a length in px, pt or cm or it should be set to thin, medium or thick.

You can individually change the width of the bottom, top, left, and right borders of an element using the following properties –

- border-bottom-width changes the width of bottom border.
- border-top-width changes the width of top border.
- border-left-width changes the width of left border.
- border-right-width changes the width of right border.

Example:

```
<html>
<head>
  <title>Border Width</title>
</head>
<body>
  <p style = "border-width:4px; border-style:solid;">
    I have a border whose width is 4px.
  </p>
  <p style = "border-width:4pt; border-style:solid;">
    I have a border whose width is 4pt.
  </p>
  <p style = "border-width:thin; border-style:solid;">
    I have a border whose width is thin.
  </p>
  <p style = "border-width:medium; border-style:solid;">
    I have a border whose width is medium;
  </p>
  <p style = "border-width:thick; border-style:solid;">
    I have a border whose width is thick.
  </p>
  <p style = "border-bottom-width:4px;border-top-width:10px;
    border-left-width: 2px;border-right-width:15px;border-style:solid;">
    I have a border with four different width.
  </p>
</body>
</html>
```

Output:

I have a border whose width is 4px.

I have a border whose width is 4pt.

I have a border whose width is thin.

I have a border whose width is medium;

I have a border whose width is thick.

I have a border with four different width.

CSS Text

Text can be manipulate using CSS properties. The following are the text properties of an element:

- The **color** property is used to set the color of a text.
- The **direction** property is used to set the text direction.
- The **letter-spacing** property is used to add or subtract space between the letters that make up a word.
- The **word-spacing** property is used to add or subtract space between the words of a sentence.
- The **text-indent** property is used to indent the text of a paragraph.
- The **text-align** property is used to align the text of a document.
- The **text-decoration** property is used to underline, overline, and strikethrough text.
- The **text-transform** property is used to capitalize text or convert text to uppercase or lowercase letters.
- The **white-space** property is used to control the flow and formatting of text.
- The **text-shadow** property is used to set the text shadow around a text.

Some of the properties are listed in the example below:

```
<html>
<head>
</head>
<body>
  <p style = "color:red;">
    My color will be red
  </p>
  <p style = "text-align:right;">
    I will be right aligned.
  </p>
  <p style = "text-align:center;">
    I will be center aligned.
  </p>
  <p style = "text-align:left;">
    I will be left aligned.
  </p>
</body>
</html>
```

Output:

My color will be red

I will be right aligned.

I will be center aligned.

I will be left aligned.

CSS Fonts:

To set the fonts of a contents in HTML using CSS, the following font properties are used:

- The **font-family** property is used to change the face of a font.
- The **font-style** property is used to make a font italic or oblique.
- The **font-variant** property is used to create a small-caps effect.
- The **font-weight** property is used to increase or decrease how bold or light a font appears.
- The **font-size** property is used to increase or decrease the size of a font.
- The **font** property is used as shorthand to specify a number of other font properties.

Font-family:

The font family of a text is set with the font-family property. The font-family property should hold several font names as a "fallback" system. If the browser does not support the first font, it tries the next font, and so on. More than one font family is specified in a comma-separated list.

Example

```
p {  
  font-family: "Times New Roman", Times, serif;  
}
```

Font-style:

The font-style property is mostly used to specify italic text. This property has three values:

- normal - The text is shown normally
- italic - The text is shown in italics
- oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

Example:

```
<html>  
<head>  
  <style>  
    p.normal {  
      font-style: normal;  
    }  
    p.italic {  
      font-style: italic;  
    }  
    p.oblique {  
      font-style: oblique;  
    }  
  </style>  
</head>  
<body>  
  <p class="normal">I am in normal style.</p>  
  <p class="italic">I am in italic style.</p>  
  <p class="oblique">I am in oblique style.</p>  
</body>  
</html>
```

Output:



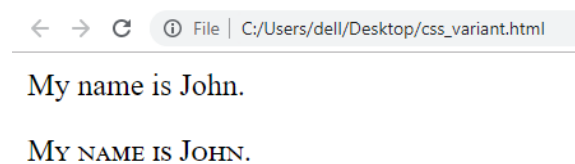
Font-variant:

The font-variant property specifies whether or not a text should be displayed in a small-caps font. In a small-caps font, all lowercase letters are converted to uppercase letters. However, the converted uppercase letters appear in a smaller font size than the original uppercase letters in the text.

Example:

```
<html>
<head>
  <style>
    p.normal {
      font-variant: normal;
    }
    p.small {
      font-variant: small-caps;
    }
  </style>
</head>
<body>
  <p class="normal">My name is John.</p>
  <p class="small">My name is John.</p>
</body>
</html>
```

Output:



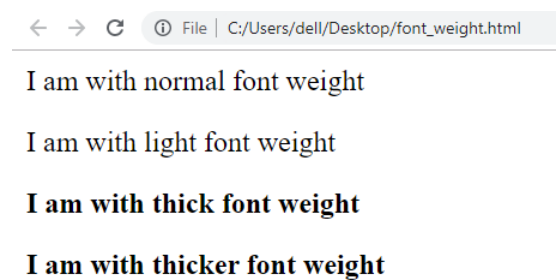
Font-weight:

The font-weight property specifies the weight of a font.

Example:

```
<html>
<head>
  <style>
    p.normal {
      font-weight: normal;
    }
    p.light {
      font-weight: lighter;
    }
    p.thick {
      font-weight: bold;
    }
    p.thicker {
      font-weight: 900;
    }
  </style>
</head>
<body>
  <p class="normal">I am with normal font weight</p>
  <p class="light">I am with light font weight</p>
  <p class="thick">I am with thick font weight</p>
  <p class="thicker">I am with thicker font weight</p>
</body>
</html>
```

Output:



← → ↻ ⓘ File | C:/Users/dell/Desktop/font_weight.html

I am with normal font weight

I am with light font weight

I am with thick font weight

I am with thicker font weight

Font-size:

The font-size property sets the size of the text. Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs. Always use the proper HTML tags, like <h1> - <h6> for headings and <p> for paragraphs. The font-size value can be an absolute, or relative size.

Absolute size:

- Sets the text to a specified size
- Does not allow a user to change the text size in all browsers (bad for accessibility reasons)
- Absolute size is useful when the physical size of the output is known

Relative size:

- Sets the size relative to surrounding elements
- Allows a user to change the text size in browsers

Font size can be set with pixels also that gives the full control over the text size

Example:

```
<html>
<head>
  <style>
    h1 {
      font-size: 40px;
    }
    h2 {
      font-size: 30px;
    }
    p {
      font-size: 14px;
    }
  </style>
</head>
<body>
  <h1>I am with font size 40px</h1>
  <h2>I am with font size 30px</h2>
  <p>I am with font size 14px</p>
  <p>I am with font size 14px</p>
</body>
</html>
```

Output:



CSS List

Lists are very helpful in conveying a set of either numbered or bullet points. This chapter teaches you how to control list type, position, style, etc., using CSS. The CSS list properties allows to:

- Set different list item markers for ordered lists
- Set different list item markers for unordered lists
- Set an image as the list item marker
- Add background colors to lists and list items

We have the following five CSS properties, which can be used to control lists:

- The **list-style-type** allows to control the shape or appearance of the marker.
- The **list-style-position** specifies whether a long point that wraps to a second line should align with the first line or start underneath the start of the marker.
- The **list-style-image** specifies an image for the marker rather than a bullet point or number.
- The **list-style** serves as shorthand for the preceding properties.
- The **marker-offset** specifies the distance between a marker and the text in the list.

The list-style-type property:

The list-style-type property allows to control the shape or style of bullet point (also known as a marker) in the case of unordered lists and the style of numbering characters in ordered lists.

The following are the values used for unordered list:

S.No	Value and Description
1	none
2	Disc(default): A filled in circle
3	Circle: An empty circle
4	Square: A filled in square

The following are some of the values used for ordered list:

Value	Description	Example
decimal	Number	1,2,3,4,5
lower-alpha	Lowercase alphanumeric characters	a, b, c, d, e
upper-alpha	Uppercase alphanumeric characters	A, B, C, D, E
lower-roman	Lowercase Roman numerals	i, ii, iii, iv, v
upper-roman	Uppercase Roman numerals	I, II, III, IV, V

Example:

```
<html>
<head>
</head>
<body>
  <ul style = "list-style-type:circle;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ul>
  <ul style = "list-style-type:square;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ul>
  <ol style = "list-style-type:decimal;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ol>
  <ol style = "list-style-type:lower-alpha;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ol>
  <ol style = "list-style-type:lower-roman;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ol>
</body>
</html>
```

Output:

- o Maths
- o Social Science
- o Physics
- Maths
- Social Science
- Physics
- 1. Maths
- 2. Social Science
- 3. Physics
- a. Maths
- b. Social Science
- c. Physics
- i. Maths
- ii. Social Science
- iii. Physics

The list-style-position Property

The list-style-position property indicates whether the marker should appear inside or outside of the box containing the bullet points. It can have one the two values:

Sr.No.	Value & Description
1	none
2	inside If the text goes onto a second line, the text will wrap underneath the marker. It will also appear indented to where the text would have started if the list had a value of outside.
3	outside If the text goes onto a second line, the text will be aligned with the start of the first line (to the right of the bullet).

Example:

```
<html>
<head>
</head>
<body>
  <ul style = "list-style-type:circle; list-stlye-position:outside;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ul>
  <ul style = "list-style-type:square;list-style-position:inside;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ul>
  <ol style = "list-style-type:decimal;list-stlye-position:outside;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ol>
  <ol style = "list-style-type:lower-alpha;list-style-position:inside;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ol>
</body>
</html>
```

Output:

- Maths
 - Social Science
 - Physics
-
- Maths
 - Social Science
 - Physics
-
1. Maths
 2. Social Science
 3. Physics
-
- a. Maths
 - b. Social Science
 - c. Physics

The list-style-image Property

The list-style-image allows you to specify an image so that you can use your own bullet style. The syntax is similar to the background-image property with the letters url starting the value of the property followed by the URL in brackets. If it does not find the given image, then default bullets are used.

Example:

```
<html>
<head>
</head>
<body>
<ul>
  <li style = "list-style-image: url(/images/bullet.gif);">Maths</li>
  <li>Social Science</li>
  <li>Physics</li>
</ul>
<ol>
  <li style = "list-style-image: url(/images/bullet.gif);">Maths</li>
  <li>Social Science</li>
  <li>Physics</li>
</ol>
</body>
</html>
```

Output:

- Maths
 - Social Science
 - Physics
-
- Maths
 2. Social Science
 3. Physics

DRAFT

The list-style Property

The list-style allows to specify all the list properties into a single expression. These properties can appear in any order.

Example:

```
<html>
<head>
</head>
<body>
  <ul style = "list-style: inside square;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ul>
  <ol style = "list-style: outside upper-alpha;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ol>
</body>
</html>
```

Output:

- Maths
- Social Science
- Physics

- A. Maths
- B. Social Science
- C. Physics

The marker-offset Property

The marker-offset property allows to specify the distance between the marker and the text relating to that marker. Its value should be a length as shown in the following example

Example:

```
<html>
<head>
</head>
<body>
  <ul style = "list-style: inside square; marker-offset:2em;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ul>
  <ol style = "list-style: outside upper-alpha; marker-offset:2cm;">
    <li>Maths</li>
    <li>Social Science</li>
    <li>Physics</li>
  </ol>
</body>
</html>
```

Output:

- Maths
- Social Science
- Physics

- A. Maths
- B. Social Science
- C. Physics

CSS Table

The look of HTML table can be improved with CSS. The following are the properties of table used in css:

- The **border-collapse** specifies whether the browser should control the appearance of the adjacent borders that touch each other or whether each cell should maintain its style.
- The **border-spacing** specifies the width that should appear between table cells.
- The **caption-side** captions are presented in the <caption> element. By default, these are rendered above the table in the document. You use the caption-side property to control the placement of the table caption.
- The **empty-cells** specifies whether the border should be shown if a cell is empty.
- The **table-layout** allows browsers to speed up layout of a table by using the first width properties it comes across for the rest of a column rather than having to load the whole table before rendering it.

The border-collapse

This property can have two values collapse and separate

Example:

```
<html>
<head>
  <style type = "text/css">
    table.one {border-collapse:collapse;}
    table.two {border-collapse:separate;}
    td.a {
      border-style:dotted;
      border-width:3px;
      border-color:#000000;
      padding: 10px;
    }
    td.b {
      border-style:solid;
      border-width:3px;
      border-color:#333333;
      padding:10px;
    }
  </style>
</head>
<body>
  <table class = "one">
    <caption>Collapse Border Example</caption>
    <tr><td class = "a"> Cell A Collapse Example</td></tr>
    <tr><td class = "b"> Cell B Collapse Example</td></tr>
  </table>
  <br />
  <table class = "two">
    <caption>Separate Border Example</caption>
    <tr><td class = "a"> Cell A Separate Example</td></tr>
    <tr><td class = "b"> Cell B Separate Example</td></tr>
  </table>
</body>
</html>
```

Output:

Collapse Border Example
Cell A Collapse Example
Cell B Collapse Example

Separate Border Example
Cell A Separate Example
Cell B Separate Example

The border-spacing Property

The border-spacing property specifies the distance that separates adjacent cells' borders. It can take either one or two values; these should be units of length. If there is only one value, it will apply to both vertical and horizontal borders. And if there are two values, in this case, the first refers to the horizontal spacing and the second to the vertical spacing:

Example:

```
<html>
<head>
  <style type = "text/css">
    table.one {
      border-collapse:separate;
      width:400px;
      border-spacing:10px;
    }
    table.two {
      border-collapse:separate;
      width:400px;
      border-spacing:10px 50px;
    }
  </style>
</head>
<body>
  <table class = "one" border = "1">
    <caption>Separate Border Example with border-spacing</caption>
    <tr><td> Cell A Collapse Example</td></tr>
    <tr><td> Cell B Collapse Example</td></tr>
  </table>
  <br />
  <table class = "two" border = "1">
    <caption>Separate Border Example with border-spacing</caption>
    <tr><td> Cell A Separate Example</td></tr>
    <tr><td> Cell B Separate Example</td></tr>
  </table>
</body>
</html>
```

Output:

Separate Border Example with border-spacing	
Cell A Collapse Example	
Cell B Collapse Example	

Separate Border Example with border-spacing	
Cell A Separate Example	
Cell B Separate Example	

The caption-side Property

The caption-side property allows you to specify where the content of a <caption> element should be placed in relationship to the table. The table that follows lists the possible values. This property can have one of the four values top, bottom, left or right. The following example uses each value.

The empty-cells Property

The empty-cells property indicates whether a cell without any content should have a border displayed. This property can have one of the three values - show, hide or inherit.

The table-layout Property

The table-layout property is supposed to help you control how a browser should render or lay out a table. This property can have one of the three values: fixed, auto or inherit. The following example shows the difference between these properties.

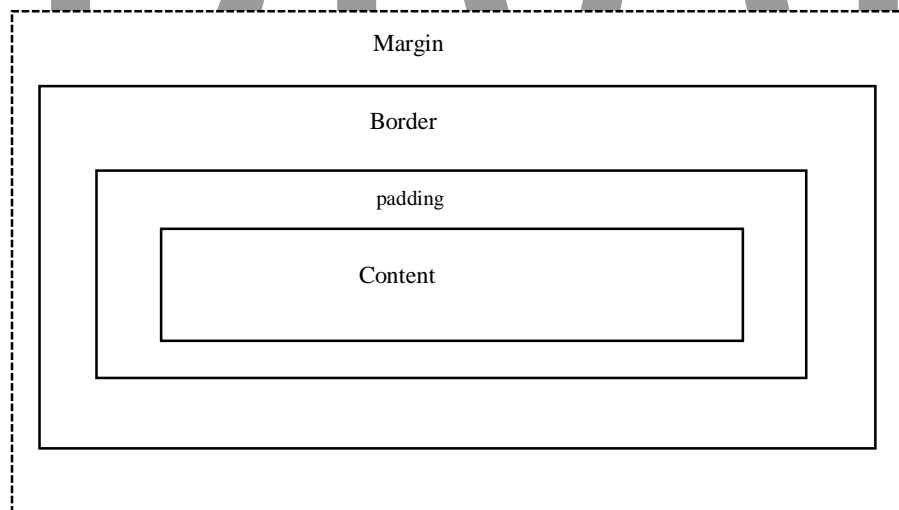
CSS Box Model

CSS box model is a container which contains multiple properties including borders, margin, padding and the content itself. It is used to create the design and layout of web pages. It can be used as a toolkit for customizing the layout of different elements. The web browser renders every element as a rectangular box according to the CSS box model.

Box-Model has multiple properties in CSS. Some of them are given below:

- borders
- margins
- padding
- Content

The following figure illustrates the box model.



Border Area: It is the area between the box's padding and margin. Its dimensions are given by the width and height of border.

Margin Area: This area consists of space between border and margin. The dimensions of Margin area are the margin-box width and the margin-box height. It is useful to separate the element from its neighbors.

Padding Area: It includes the element's padding. This area is actually the space around the content area and within the border box. Its dimensions are given by the width of the padding-box and the height of the padding-box.

Content Area: This area consists of content like text, image, or other media content. It is bounded by the content edge and its dimensions are given by content box width and height.

DRAFT

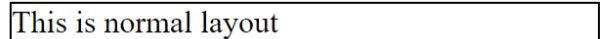
Box Layout

The width and height attribute are used to create the box layout in HTML using CSS which are illustrated in the example below

Example:

```
<html>
<head>
  <style>
    div {
      background-color: none;
      width: 300px;
      border: 1px solid ;
    }
  </style>
</head>
<body>
  <div>This is normal layout</div>
</body>
</html>
```

Output:

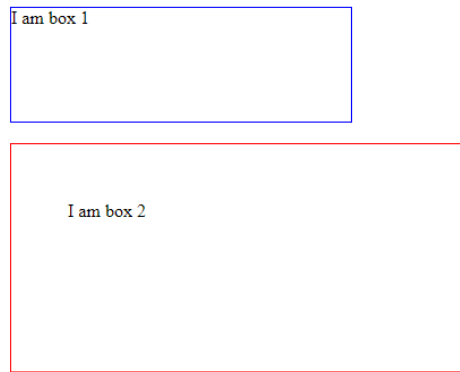


Example of box using padding :

```
<html>
  <head>
    <style>
      .div1 {
        width: 300px;
        height: 100px;
        border: 1px solid blue;
      }
      .div2 {
        width: 300px;
        height: 100px;
        padding: 50px;
        border: 1px solid red;
      }
    </style>
  </head>

  <body>
    <div class = "div1">I am box 1</div><br />
    <div class = "div2">I am box 2</div>
  </body>
</html>
```

Output:



Example of Box using margin, padding, float, width

```
<html>
<head><title>example</title>
<style type="text/css">
.top{
    width:100%;
    border-top:1px solid;
    border-right:1px solid;
    border-left:1px solid;
    border-bottom:1px solid;
}
.box1{
    padding:20px;
}
.mid{
    width:100%;
    border-top:1px solid;
}
.box2{
    padding:20px;
}
.middle{
    width:100%;
    border-top:1px solid;
}
.box3{
    width:80%;
    float:left;
    border-right:1px solid;
    padding:30px;
}
.box4{
    padding:30px;
}
.slastbox{
    width:100%;
    border-top:1px solid;
}
```

```

.box5{
    width:30%;
    float:left;
    border-right:1px solid;
    padding:20px;
}
.box6{
    padding:20px;
}
.lastbox{
    width:100%;
    border-top:1px solid;
}.box7{
    width:90%;
    float:left;
    border-right:1px solid;
    padding:20px;
}
.box8{
    padding:20px;
}
</style>
</head>
<body>
<div class="top">
  <div class="box1"></div>
<div class="mid">
  <div class="box2"></div>
  <div class="middle">
    <div class="box3"></div>
    <div class="box4"></div>
<div class="slastbox">
  <div class="box5"></div>
  <div class="box6"></div>
<div class="lastbox">
  <div class="box7"></div>
  <div class="box8"></div>
</body>
</html>

```

Output:

DRAFT

CSS Display Properties

The display property affects the most basic presentation of an element, effectively classing the element as a certain type of element. The rendering of the element may depend heavily on its display type, and certain properties will only work on elements that have specific display values.

The following are the possible values for display properties

- **inline** – This value causes an element to generate an inline-level box; for example, the HTML elements STRONG, CODE, or EM (among others). The element will generate one or more inline boxes when it is displayed.
- **block** – This value causes an element to generate a block-level box; for example, the HTML elements P, H1, or PRE (among others). The element will generate a block box when it is displayed.
- **list-item** – This value causes an element to generate both a block box and a list-item inline box. In HTML, the LI element is the only example of such an element.
- **run-in** – Under certain conditions, this value will cause the element to be inserted into the beginning of the following element. If an element A is set to display: run-in and is followed by a block-level element B, then A becomes the first inline-level box of B. If the element following A is not block-level, then A becomes a block-level box.
- **compact** – Under certain conditions, this value will cause the element to be placed to one side of the following element.
- **marker** – This value will set generated content to be a marker; thus, it should be used only in conjunction with the :before and :after pseudo- elements when they are set on block-level elements.
- **table** – This value causes an element to generate a block-level table box. This is analogous to the HTML element TABLE.
- **inline-table** – This value causes an element to generate an inline-level table box. While there is no analogue in HTML, it can be envisioned as a traditional HTML table which can appear in the middle of a line of text.
- **table-cell** – This value declares the element to be a table cell. This is analogous to the HTML element TD.
- **table-row** – This value declares the element to be a row of table cells. This is analogous to the HTML element TR.
- **table-row-group** – This value declares the element to be a group of table rows. This is analogous to the HTML element TBODY.
- **table-column** – This value declares the element to be a column of table cells. This is analogous to the HTML element COL.
- **table-column-group** – This value declares the element to be a group of table columns. This is analogous to the HTML element COLGROUP.
- **table-header-group** – This value declares the element to be a group of cells which is always visible at the top of the table, placed before any row or row-groups but after any top-aligned table captions. This is analogous to the HTML element THEAD.
- **table-footer-group** – This value declares the element to be a group of cells which is always visible at the bottom of the table, placed after any row or row-groups but before any bottom-aligned table captions. This is analogous to the HTML element TFOOT.
- **table-caption** – This value declares the element to be a caption for a table. This is analogous to the HTML element CAPTION.
- **none** – The element will generate no boxes at all, and thus will neither be displayed nor impact the layout of the document.

Example:

```
<html>
<head><title>display</title>
</head>
<body>
  <p style = "display:inline;">
    This paragraph will inline with the next paragraph
  </p>
  <p style = "display:inline;">
    and will make a single line.
  </p>
  <br />
  <br />
  <div style = "display:block;">
    This paragraph will be separate from the next paragraph
  </div>
  <div style = "display:block;">
    and this is second paragraph.
  </div>
</body>
</html>
```

Output:

```
This paragraph will inline with the next paragraph

and will make a single line.


This paragraph will be separate from the next paragraph

and this is second paragraph.
```

CSS Padding

The padding property allows you to specify how much space should appear between the content of an element and its border. The value of this attribute should be either a length, a percentage, or the word inherit. If the value is inherit, it will have the same padding as its parent element. If a percentage is used, the percentage is of the containing box.

The following CSS properties can be used to control lists. The different values for the padding on each side of the box can be set using the following properties:

- The **padding-bottom** specifies the bottom padding of an element.
- The **padding-top** specifies the top padding of an element.
- The **padding-left** specifies the left padding of an element.
- The **padding-right** specifies the right padding of an element.
- The **padding** serves as shorthand for the preceding properties.

Example:

```
<html>
  <head><title>Css paddidng</title>
</head>
  <body>
    <p style = "padding-bottom: 15px; border:1px solid black;">
      This is a paragraph with a specified bottom padding
    </p>
    <p style = "padding-left: 15px; border:1px solid black;">
      This is a paragraph with a specified left padding
    </p>
    <p style = "padding-right: 15px; border:1px solid black;">
      This is a paragraph with a specified right padding
    </p>
    <p style = "padding: 15px; border:1px solid black;">
      all four padding will be 15px
    </p>
    <p style = "padding:10px 2%; border:1px solid black;">
      top and bottom padding will be 10px, left and right
      padding will be 2% of the total width of the document.
    </p>
    <p style = "padding: 10px 2% 10px; border:1px solid black;">
      top padding will be 10px, left and right padding will
      be 2% of the total width of the document, bottom padding will be 10px
    </p>
    <p style = "padding: 10px 2% 10px 10px; border:1px solid black;">
      top padding will be 10px, right padding will be 2% of
      the total width of the document, bottom padding and top padding will be 10px
    </p>
  </body>
</html>
```

Output:

This is a paragraph with a specified bottom padding
This is a paragraph with a specified left padding
This is a paragraph with a specified right padding
all four padding will be 15px
top and bottom padding will be 10px, left and right padding will be 2% of the total width of the document.
top padding will be 10px, left and right padding will be 2% of the total width of the document, bottom padding will be 10px
top padding will be 10px, right padding will be 2% of the total width of the document, bottom padding and top padding will be 10px

CSS Margin

The margin property defines the space around an HTML element. It is possible to use negative values to overlap content.

The values of the margin property are not inherited by the child elements. Remember that the adjacent vertical margins (top and bottom margins) will collapse into each other so that the distance between the blocks is not the sum of the margins, but only the greater of the two margins or the same size as one margin if both are equal.

The following are the properties to set an element margin.

- The **margin** specifies a shorthand property for setting the margin properties in one declaration.
- The **margin-bottom** specifies the bottom margin of an element.
- The **margin-top** specifies the top margin of an element.
- The **margin-left** specifies the left margin of an element.
- The **margin-right** specifies the right margin of an element.

Example

```
<html>
<head>
<style>
p {
  background-color: green;
  color:white;
}
p.ex {
  margin-top: 50px;
  margin-bottom: 50px;
  margin-right: 100px;
  margin-left: 100px;
}
</style>
</head>
<body>
<p>This paragraph is not displayed with specified margin. </p>
<p class="ex">This paragraph is displayed with specified margin.</p>
</body>
</html>
```

Output:

This paragraph is not displayed with specified margin.

This paragraph is displayed with specified margin.

CSS Float:

With CSS float, an element can be pushed to the left or right, allowing other elements to wrap around it. Float is very often used for images, but it is also useful when working with layouts.

How Elements Float

Elements are floated horizontally; this means that an element can only be floated left or right, not up or down. A floated element will move as far to the left or right as it can. Usually this means all the way to the left or right of the containing element. The elements after the floating element will flow around it. The elements before the floating element will not be affected. If an image is floated to the right, a following text flows around it, to the left.

Example

```
img
{
float:right;
}
```

Floating Elements Next to Each Other

If you place several floating elements after each other, they will float next to each other if there is room. Here we have made an image gallery using the float property:

Example

```
.thumbnail
{
float:left;
width:110px;
height:90px;
margin:5px;
}
```

Turning off Float - Using Clear

Elements after the floating element will flow around it. To avoid this, use the clear property. The clear property specifies which sides of an element other floating elements are not allowed. Add a text line into the image gallery, using the clear property:

Example

```
.text_line
{
clear:both;
}
```

CSS Positioning

CSS helps to position the HTML element. The HTML element can be put at whatever location. You can specify whether you want the element positioned relative to its natural position in the page or absolute based on its parent element.

The following are the positioning related properties:

Relative Positioning

Relative positioning changes the position of the HTML element relative to where it normally appears. So "left:20" adds 20 pixels to the element's LEFT position.

You can use two values top and left along with the position property to move an HTML element anywhere in the HTML document.

- Move Left - Use a negative value for left.
- Move Right - Use a positive value for left.
- Move Up - Use a negative value for top.
- Move Down - Use a positive value for top.

Example:

```
<html>
  <head>
  </head>
  <body>
    <div style = "position:relative; left:80px; top:2px; background-
color:yellow;">
      I have relative positioning.
    </div>
  </body>
</html>
```

Output:

I have relative positioning.

Absolute Positioning

An element with position: absolute is positioned at the specified coordinates relative to your screen top-left corner. You can use two values top and left along with the position property to move an HTML element anywhere in the HTML document.

- Move Left - Use a negative value for left.
- Move Right - Use a positive value for left.
- Move Up - Use a negative value for top.
- Move Down - Use a positive value for top.

Example

```
<html>
  <head>
  </head>
  <body>
    <div style = "position:absolute; left:80px; top:20px; background-
color:yellow;">
      This div has absolute positioning.
    </div>
  </body>
</html>
```

Output

This div has absolute positioning.

Fixed Positioning

Fixed positioning allows you to fix the position of an element to a particular spot on the page, regardless of scrolling. Specified coordinates will be relative to the browser window.

You can use two values top and left along with the position property to move an HTML element anywhere in the HTML document.

- Move Left - Use a negative value for left.
- Move Right - Use a positive value for left.
- Move Up - Use a negative value for top.
- Move Down - Use a positive value for top

Example

```
<html>
  <head>
  </head>
  <body>
    <div style = "position:fixed; left:80px; top:20px; background-
color:yellow;">
      This div has fixed positioning.
    </div>
  </body>
</html>
```

Output:

This div has fixed positioning.

CSS Box Shadows

The box-shadow property attaches one or more shadows to an element.

To attach more than one shadow to an element, add a comma-separated list of shadows

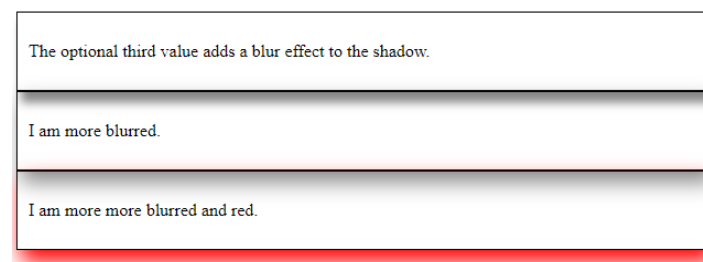
Syntax:

box-shadow: none|h-offset v-offset blur spread color |inset|initial|inherit;

Example:

```
<html>
<head>
<style>
#example1 {
  border: 1px solid;
  padding: 10px;
  box-shadow: 5px 10px 8px #888888;
}
#example2 {
  border: 1px solid;
  padding: 10px;
  box-shadow: 5px 10px 18px #888888;
}
#example3 {
  border: 1px solid;
  padding: 10px;
  box-shadow: 5px 10px 18px red;
}
</style>
</head>
<body>
<div id="example1">
  <p>The optional third value adds a blur effect to the shadow.</p>
</div>
<div id="example2">
  <p>I am more blurred.</p>
</div>
<div id="example3">
  <p>I am more more blurred and red.</p>
</div>
</body>
</html>
```

Output:



DRAFT

CSS3 Text Effects and Shadows

CSS3 supported to add shadow effects to text.

Example:

```
<html>
  <head>
    <style>
      h1 {
        text-shadow: 2px 2px;
      }
      h2 {
        text-shadow: 2px 2px red;
      }
      h3 {
        text-shadow: 2px 2px 5px red;
      }
      h4 {
        color: white;
        text-shadow: 2px 2px 4px #000000;
      }
      h5 {
        text-shadow: 0 0 3px #FF0000;
      }
      h6 {
        text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;
      }
      p {
        color: white;
        text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;
      }
    </style>
  </head>
  <body>
    <h1>CSS is fun</h1>
    <h2>CSS is fun</h2>
    <h3>CSS is fun</h3>
    <h4>CSS is fun</h4>
    <h5>CSS is fun</h5>
    <h6>CSS is fun</h6>
    <p>CSS is fun</p>
  </body>
</html>
```

Output

CSS is fun

CSS is fun

CSS is fun

CSS is fun

CSS is fun

CSS is fun

CSS is fun

DRAFT

Basics of Responsive Web Designs

Responsive web design provides an optimal experience, easy reading and easy navigation with a minimum of resizing on different devices such as desktops, mobiles and tabs).

Responsive Structure



Example

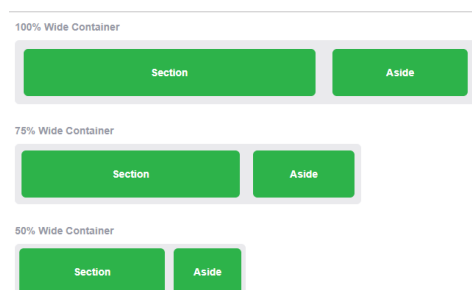
```
<html>  
  <head>  
    <style>  
      body {  
        font: 600 14px/24px "Open Sans",  
          "HelveticaNeue-Light",  
          "Helvetica Neue Light",  
          "Helvetica Neue",  
          Helvetica, Arial,  
          "Lucida Grande",  
          Sans-Serif;  
      }  
      h1 {  
        color: #9799a7;  
        font-size: 14px;  
        font-weight: bold;  
        margin-bottom: 6px;  
      }  
      .container:before, .container:after {  
        content: "";  
        display: table;  
      }  
      .container:after {  
        clear: both;  
      }  
      .container {  
        background: #eaeaed;  
        margin-bottom: 24px;  
        *zoom: 1;  
      }  
      .container-75 {  
        width: 75%;  
      }  
      .container-50 {  
        margin-bottom: 0;  
        width: 50%;  
      }
```

```

    .container, section, aside {
        border-radius: 6px;
    }
    section, aside {
        background: #2db34a;
        color: #fff;
        margin: 1.858736059%;
        padding: 20px 0;
        text-align: center;
    }
    section {
        float: left;
        width: 63.197026%;
    }
    aside {
        float: right;
        width: 29.3680297%;
    }
</style>
</head>
<body>
    <h1>100% Wide Container</h1>
    <div class = "container">
        <section>Section</section>
        <aside>Aside</aside>
    </div>
    <h1>75% Wide Container</h1>
    <div class = "container container-75">
        <section>Section</section>
        <aside>Aside</aside>
    </div>
    <h1>50% Wide Container</h1>
    <div class = "container container-50">
        <section>Section</section>
        <aside>Aside</aside>
    </div>
</body>
</html>

```

Output:



Media Queries

Media queries is for different style rules for different size devices such as mobiles, desktops, etc.
For media queries we use @media screen in css

Example

```
<html>
  <head>
    <style>
      body {
        background-color: lightpink;
      }
      @media screen and (max-width: 420px) {
        body {
          background-color: lightblue;
        }
      }
    </style>
  </head>
  <body>
    <p>
      If screen size is less than 420px, then it will show lightblue
      color, or else it will show light pink color
    </p>
  </body>
</html>
```

Output

If screen size is less than 420px, then it will show lightblue color, or else it will show light pink color

Introduction to Bootstrap

Bootstrap is a free and open-source framework for creating websites and web applications. It's the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web. Bootstrap can make things a whole lot easier. Bootstrap enables you to create responsive websites without you needing to do the "responsive" bit. Bootstrap takes care of that.

Advantages of Bootstrap

One of the main benefits of development frameworks like Bootstrap is that they can help speed up development times, while maintaining quality and consistency across the site. You no longer need to re-design every element. And you don't need to spend hours trying to get everything looking and working right across browsers, platforms, and devices. By using Bootstrap, all (most) of the hard work is done for you.

Given Bootstrap is the most popular frontend development framework on the web, this skillset could be a useful one to learn. Adding Bootstrap to your bag of tricks could help you in many ways from building websites faster, to landing your dream job.

Also, although Bootstrap comes with its own set of styles, these are easy to override. You're not locked into the "Bootstrap design". You are free to use whichever Bootstrap components you choose, while adding your own on top. There are thousands of websites out there that are built on Bootstrap, but with their own design.