**2020**

MANIKA GURUNG

6/11/2020



INTERNSHIP REPORT

**Acknowledgement**

First I would like to thank Ujwal Sir and Bhuwan sir for giving me the opportunity to do an internship within the organization, The main purpose of internship is to enhance a career and personal development. Internship focuses on providing practical approach to an individual about how the jobs are performed in real world. Internship binds us for learning and working environment with the professional employees. In this period we mostly learn how the works are performed professional way.

I would like to express my gratitude to sir,Ujwal Pradhan and sir, Bhuwan Gurung for providing proper guidance as well as encouraging and advising during my internship period. I was able to gain variant skill.

**Table of Contents**

Acknowledgement………………………………………………………………2

Summary………………………………………………………………………...3

1.Programming Languages learned…………………………………………… 4

2. HTML 5 Introduction……………………………………………..…………5-9

3.What is CSS ………………………………....................................................10-24

5.Conclusion……………………………………………………………………..25

**Programming Languages learned**

HTML 5 and CSS 3.HTML and CSS are not , strictly speaking programming languages, but they belong to what can be named descriptive markup languages.HTML (Hypertext Markup Language) consists of a set of tags that are used to structure the webpage. Since the creation of XHTML in 2000,it has to be written like an XML tree.

The CSS (Cascading Style Sheets) was introduced after the HTML to separate the content from the look .It consists of a list of properties that applied to the HTML tags.

# HTML5 Introduction

## What is New in HTML5?

The DOCTYPE declaration for HTML5 is very simple:

<!DOCTYPE html>

The character encoding (charset) declaration is also very simple:

<meta charset="UTF-8">

### HTML5 Example:

**<!DOCTYPE html>**<html>  
<head>  
<meta charset="UTF-8">  
<title>Title of the document</title>  
</head>  
  
<body>  
Content of the document......  
</body>  
  
</html>

## HTML Documents

All HTML documents must start with a document type declaration: **<!DOCTYPE html>**.

The HTML document itself begins with **<html>** and ends with **</html>**.

The visible part of the HTML document is between **<body>** and **</body>**.

## HTML Links

HTML links are defined with the **<a>** tag:

**Example:**

<a href="http://www.w3schools.com">This is a link</a>

The link's destination is specified in the **href attribute**.

Attributes are used to provide additional information about HTML elements.

## HTML Images

HTML images are defined with the **<img>** tag.

The source file (**src**), alternative text (**alt**), and size (**width** and **height**) are provided as **attributes**:

**Example:** <img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142">

## HTML Attributes

* All HTML elements can have **attributes**
* Attributes provide **additional information** about an element
* Attributes are always specified in **the start tag**
* Attributes usually come in name/value pairs like: **name="value"**

## Defining an HTML Table

An HTML table is defined with the **<table>** tag.

Each table row is defined with the **<tr>** tag. A table header is defined with the **<th>** tag. By default, table headings are bold and centered. A table data/cell is defined with the **<td>** tag.

# HTML Text Formatting

## HTML Formatting Elements

HTML uses elements like <b> and <i> for formatting output, like **bold** or *italic* text.

Formatting elements were designed to display special types of text:

* <b> - Bold text
* <strong> - Important text
* <i> - Italic text
* <em> - Emphasized text
* <mark> - Marked text
* <small> - Small text
* <del> - Deleted text
* <ins> - Inserted text
* <sub> - Subscript text
* <sup> - Superscript text

## HTML <b> and <strong> Elements

The HTML **<b>** element defines **bold** text, without any extra importance.

<b>This text is bold</b>

The HTML **<strong>** element defines **strong** text, with added semantic "strong" importance.

<strong>This text is strong</strong>

# HTML Lists

### HTML List Example

### An Unordered List:

* Item
* Item
* Item
* Item

### An Ordered List:

1. First item
2. Second item
3. Third item
4. Fourth item

# HTML Form

## The <form> Element

HTML forms are used to collect user input.

The **<form>** element defines an HTML form:

<form>  
.  
form elements  
.  
</form>

HTML forms contain **form elements**.

Form elements are different types of input elements, checkboxes, radio buttons, submit buttons, and more.

## The <input> Element

The **<input>** element is the most important **form element**.

The <input> element has many variations, depending on the **type** attribute.

Here are the types used in this chapter:

## Text Input

**<input type="text">** defines a one-line input field for **text input**:

## Radio Button Input

**<input type="radio"> defines a radio button.**

Radio buttons let a user select ONE of a limited number of choices

## The Submit Button

**<input type="submit">** defines a button for **submitting** a form to a **form-handler**.

The form-handler is typically a server page with a script for processing input data.

The form-handler is specified in the form's **action** attribute

## What is CSS?

* **CSS** stands for **C**ascading **S**tyle **S**heets
* CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
* CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
* External stylesheets are stored in **CSS files**

# CSS Syntax and Selectors

## CSS Syntax

A CSS rule-set consists of a selector and a declaration block:



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

## Three Ways to Insert CSS

There are three ways of inserting a style sheet:

* External style sheet
* Internal style sheet
* Inline style

# CSS Colors

Colors in CSS are most often specified by:

* a valid color name - like "red"
* an RGB value - like "rgb(255, 0, 0)"
* a HEX value - like "#ff0000"

# CSS Backgrounds

The CSS background properties are used to define the background effects for elements.

CSS background properties:

* background-color
* background-image
* background-repeat
* background-attachment
* background-position

## Background Color

The background-color property specifies the background color of an element

## 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:

### Example

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

## CSS Border Properties

The CSS border properties allow you to specify the style, width, and color of an element's border.

## Border Style

The border-style property specifies what kind of border to display.

The following values are allowed:

* dotted - Defines a dotted border
* dashed - Defines a dashed border
* solid - Defines a solid border
* double - Defines a double border
* groove - Defines a 3D grooved border. The effect depends on the border-color value
* ridge - Defines a 3D ridged border. The effect depends on the border-color value
* inset - Defines a 3D inset border. The effect depends on the border-color value
* outset - Defines a 3D outset border. The effect depends on the border-color value
* none - Defines no border
* hidden - Defines a hidden border

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

## CSS Margins

The CSS margin properties are used to generate space around elements.

The margin properties set the size of the white space outside the border.

With CSS, you have full control over the margins. There are CSS properties for setting the margin for each side of an element (top, right, bottom, and left).

## Margin - Individual Sides

**CSS has properties for specifying the margin for each side of an element:**

* margin-top
* margin-right
* margin-bottom
* margin-left

All the margin properties can have the following values:

* auto - the browser calculates the margin
* length - specifies a margin in px, pt, cm, etc.
* % - specifies a margin in % of the width of the containing element
* inherit - specifies that the margin should be inherited from the parent element

p {  
    margin-top: 100px;  
    margin-bottom: 100px;  
    margin-right: 150px;  
    margin-left: 80px;

* **px;**
  + top margin is 25px
  + right margin is 50px
  + bottom margin is 75px
  + left margin is 100px

If the margin property has three values:

* **margin: 25px 50px 75px;**
  + top margin is 25px
  + right and left margins are 50px
  + bottom margin is 75px

If the margin property has two values:

* **margin: 25px 50px;**
  + top and bottom margins are 25px
  + right and left margins are 50px

If the margin property has one value:

* **margin: 25px;**
  + all four margins are 25px

## The auto Value

You can set the margin property to auto to horizontally center the element within its container.

The element will then take up the specified width, and the remaining space will be split equally between the left and right margins:

### Example

div {  
    width: 300px;  
    margin: auto;  
    border: 1px so

## Border Width

The border-width property specifies the width of the four borders.

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.

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

5px border-width

# CSS Padding

The CSS padding properties are used to generate space around content.

The padding clears an area around the content (inside the border) of an element.

With CSS, you have full control over the padding. There are CSS properties for setting the padding for each side of an element (top, right, bottom, and left).

## Padding - Individual Sides

CSS has properties for specifying the padding for each side of an element:

* padding-top
* padding-right
* padding-bottom
* padding-left

All the padding properties can have the following values:

* length - specifies a padding in px, pt, cm, etc.
* % - specifies a padding in % of the width of the containing element
* inherit - specifies that the padding should be inherited from the parent element

The following example sets different padding for all four sides of a <p> element:

p {  
    padding-top: 50px;  
    padding-right: 30px;  
    padding-bottom: 50px;  
    padding-left: 80px;  
}

## Padding - Shorthand Property

To shorten the code, it is possible to specify all the padding properties in one property.

The padding property is a shorthand property for the following individual padding properties:

* padding-top
* padding-right
* padding-bottom
* padding-left

If the padding property has four values:

* **padding: 25px 50px 75px 100px;**
  + top padding is 25px
  + right padding is 50px
  + bottom padding is 75px
  + left padding is 100px

If the padding property has three values:

* **padding: 25px 50px 75px;**
  + top padding is 25px
  + right and left paddings are 50px
  + bottom padding is 75px

If the padding property has two values:

* **padding: 25px 50px;**
  + top and bottom paddings are 25px
  + right and left paddings are 50px

If the padding property has one value:

* **padding: 25px;**
  + all four paddings are 25px

|  |
| --- |
|  |

# CSS Height and Width

The height and width properties are used to set the height and width of an element.

The height and width can be set to auto (this is default. Means that the browser calculates the height and width), or be specified in length values, like px, cm, etc., or in percent (%) of the containing block.

 The height and width properties do not include padding, borders, or margins; they set the height/width of the area inside the padding, border, and margin of the element!

The max-width property is used to set the maximum width of an element.

**CSS Outline**

The CSS outline properties specify the style, color, and width of an outline.

An outline is a line that is drawn around elements (outside the borders) to make the element "stand out".

However, the outline property is different from the border property - The outline is NOT a part of an element's dimensions; the element's total width and height is not affected by the width of the outline.

This element has a thin black border and a double outline that is 10px wide and green.

# CSS Text

# TEXT FORMATTING

## Text Color

The color property is used to set the color of the text.

With CSS, a color is most often specified by:

* a color name - like "red"
* a HEX value - like "#ff0000"
* an RGB value - like "rgb(255,0,0)"

Look at [CSS Color Values](http://www.w3schools.com/cssref/css_colors_legal.asp) for a complete list of possible color values.

The default text color for a page is defined in the body selector.

## Text Alignment

The text-align property is used to set the horizontal alignment of a text.

A text can be left or right aligned, centered, or justified.

## Text Decoration

The text-decoration property is used to set or remove decorations from text.

## Text Transformation

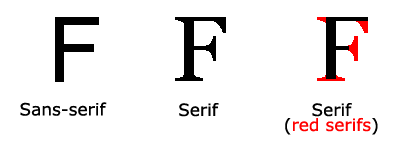
The text-transform property is used to specify uppercase and lowercase letters in a text.

It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word:

# CSS Font

The CSS font properties define the font family, boldness, size, and the style of a text.

## Difference Between Serif and Sans-serif Fonts



## 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)

## Font Size

The font-size property sets the size of the text.

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 browser

## Font Weight

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

## Background Color

The background-color property can be used to specify a background color for links:

### Example:

a:link {  
    background-color: yellow;  
}  
  
a:visited {  
    background-color: cyan;  
}  
  
a:hover {  
    background-color: lightgreen;  
}  
  
a:active {  
    background-color: hotpink;  
}

|  |  |  |
| --- | --- | --- |
|  |  |  |

The display property specifies if/how an element is displayed.

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline.

# CSS Layout - The position Property

The position property specifies the type of positioning method used for an element (static, relative, fixed or absolute).

## The position Property

The position property specifies the type of positioning method used for an element.

There are four different position values:

* static
* relative
* fixed
* absolute

Elements are then 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 position value.

## position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

This <div> element has position: static;

Here is the CSS that is used:

## Position: relative

An element with position: relative; is positioned relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

## Position: fixed;

An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

Notice the fixed element in the lower-right corner of the page. Here is the CSS that is used:

## Position: absolute;

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

 A "positioned" element is one whose position is anything except static.

Here is a simple example:

This <div> element has position: relative;

This <div> element has position: absolute;

Here is the CSS that is used:

# CSS3 Flexible Box

Flexible boxes, or flex box, are a new layout mode in CSS3.

Use of flex box ensures that elements behave predictably when the page layout must accommodate different screen sizes and different display devices.

For many applications, the flexible box model provides an improvement over the block model in that it does not use floats, nor do the flex container's margins collapse with the margins of its contents.

**Flex Direction**

The flex-direction property specifies the direction of the flexible items inside the flex container. The default value of flex-direction is row (left-to-right, top-to-bottom).

The other values are as follows:

* row-reverse - If the writing-mode (direction) is left to right, the flex items will be laid out right to left
* column - If the writing system is horizontal, the flex items will be laid out vertically
* Column-reverse - Same as column, but reversed.

**The justify-content Property**

The justify-content property horizontally aligns the flexible container's items when the items do not use all available space on the main-axis.

The possible values are as follows:

* flex-start - Default value. Items are positioned at the beginning of the container
* flex-end - Items are positioned at the end of the container
* center - Items are positioned at the center of the container
* space-between - Items are positioned with space between the lines
* space-around - Items are positioned with space before, between, and after the lines

**The align-items Property**

The align-items property vertically aligns the flexible container's items when the items do not use all available space on the cross-axis.

The possible values are as follows:

* Stretch - Default value. Items are stretched to fit the container
* flex-start - Items are positioned at the top of the container
* flex-end - Items are positioned at the bottom of the container
* center - Items are positioned at the center of the container (vertically)
* baseline - Items are positioned at the baseline of the container

**The flex-wrap Property**

The flex-wrap property specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line.

The possible values are as follows:

* no wrap - Default value. The flexible items will not wrap
* wrap - The flexible items will wrap if necessary
* wrap-reverse - The flexible items will wrap, if necessary, in reverse order

**The align-content Property**

The align-content property modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines.

The possible values are as follows:

* Stretch - Default value. Lines stretch to take up the remaining space
* flex-start - Lines are packed toward the start of the flex container
* flex-end - Lines are packed toward the end of the flex container
* center - Lines are packed toward the center of the flex container
* space-between - Lines are evenly distributed in the flex container
* Space-around - Lines are evenly distributed in the flex container, with half-size spaces on either end.

**What is a Grid-View?**

Many web pages are based on a grid-view, which means that the page is divided into columns:

# CSS3 Animations

CSS3 animations allows animation of most HTML elements without using JavaScript or Flash!

**What are CSS3 Animations?**

An animation lets an element gradually change from one style to another.

You can change as many CSS properties you want, as many times you want.

To use CSS3 animation, you must first specify some keyframes for the animation.

Keyframes hold what styles the element will have at certain times.

**The @keyframes Rule**

When we specify CSS styles inside the @keyframes rule, the animation will gradually change from the current style to the new style at certain times.

To get an animation to work, we must bind the animation to an element.

The following example binds the "example" animation to the <div> element. The animation will lasts for 4 seconds, and it will gradually change the background-color of the <div> element from "red" to "yellow":

**CSS3 Animation Properties**

* animation-name
* animation-duration
* animation-timing-function
* animation-delay
* animation-iteration-count
* animation-direction
* animation-fill-mode
* animation-play-state

**Conclusion**

Overall internship is a really good program. It helps to enhance and develop my skills, abilities ,and knowledge. I have gained experience and could further update my learning on HTML and CSS in a better way. Madmix Studio is a better place to do an internship since it provide numerous benefits and advantages to the practical trainees. They provide us all the facilities needed like computer systems,wi-fi, etc.