



INTRODUCTION TO HTML & CSS

A Fundamental Guide to Webpage Development

Abstract

This documentation provides a structural / practical guide to creating a webpage using html and CSS. The guide contains illustrations and sample editable syntax.

Table of Contents

Introduction.....	2
HTML Syntax Elements	2
Structure of an HTML Document/Script	4
The HTML Structure	6
Adding a Title, Paragraph and Images.....	8
Adding Lists.....	10
Adding Tables.....	13
Adding Divs.....	15
Intro to CSS.....	20
Adding some style to our html [Elements and ID's]:	21
Working with Classes	23
Pseudo Styles	25
Positioning	26
External Style Sheets	28

Introduction

HTML is an abbreviation that means Hyper Text Markup Language, it is the basic language used in the design and formatting of content for webpages. It is usually used along with Cascading Style Sheets (CSS), and scripting languages such as JavaScript. For robustness and aesthetic, and proper of webpages, HTML usually incorporate CSS and JavaScript to improve rendering and functionality.

HTML codes can be written on a local PC with the aid of any text editors, but for ease of readability, clarity and editability, below are some recommended text editors for HTML scripting:

 [Visual Studio Code](#)

 [Notepad ++](#)

 [Apache NetBeans](#)

 [Eclipse](#)

 [Sublime](#)

For this guide, visual studio is going to be used, but you can download and use any of the text editors. The structure and language remain the same irrespective of which editor is being used.

HTML Syntax Elements

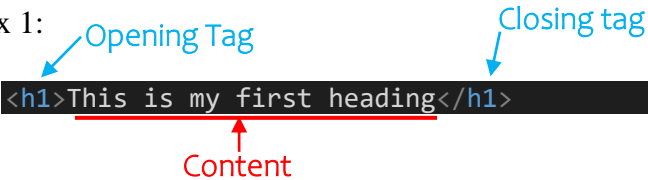
A basic HTML syntax has two elements:

- i. Tags
- ii. Content

More advanced syntax have:

1. Tags
2. Attributes (class/id)
3. Values (usually representing the attribute)
4. Content

Ex 1:



The diagram shows the HTML code `<h1>This is my first heading</h1>` on a black background. A blue arrow labeled 'Opening Tag' points to `<h1>`. A blue arrow labeled 'Closing tag' points to `</h1>`. A red arrow labeled 'Content' points to the text 'This is my first heading' between the tags. The text is underlined in red.

In the example above `<h1></h1>` represents the tags, while the text between the tags represents the content for that tagged element

The basic element of HTML are its **Tags** `< > </ >`. All HTML content must/should be written in tags, else it will not be recognised as HTML content

NB: `<h1></h1>` is not a default tag but is a **heading tag** used for creating/displaying headings. There are numerous tags, which we shall see shortly.

Ex 2:



The diagram shows the HTML code `<p id="about us">WebGIS Class Work</p>` on a black background. A blue arrow labeled 'Opening Tag' points to `<p`. A blue arrow labeled 'Attribute' points to `id="about us"`. An orange arrow labeled 'Value' points to the text 'about us' within the attribute. A blue arrow labeled 'Closing tag' points to `</p>`. A red arrow labeled 'Content' points to the text 'WebGIS Class Work' between the tags. The text is underlined in red.

Attributes and Values are used to identify and differentiate tags of the same group. They act more like a reference/index for a particular syntax block. Imagine if you had 50 paragraphs (`<p>50 of these</p>`) on a page, to easily identify a paragraph, it could be referenced with an attribute and value, so that it can be easily selected, styled, and edited.

Imagine saying look for paragraph `<p id= "21">` and saying read through all the paragraphs and look for the paragraph that says "the boy is struggling to identify a paragraph because he didn't give it an id" ...hope you understand it now.

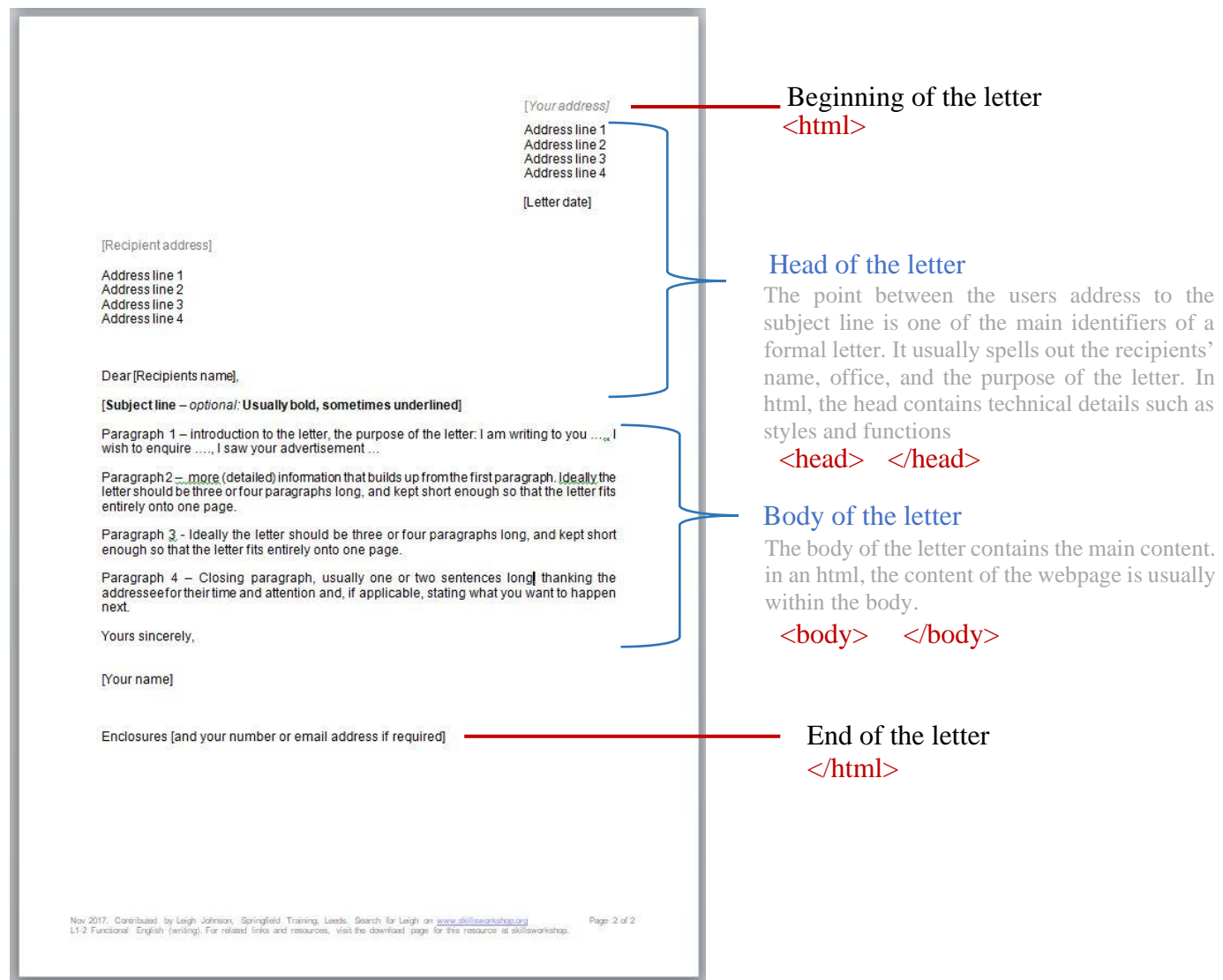
NB: Giving attributes and values is very important when it comes to styling using CSS.

*the box below is a humour box, it is meant to be a refresher for your sore eyes and to break the long reads of syntax. You may/may not see a couple of them along the line. **Please feel free to skip it.***

"I hope you are not confused, and that you have not given up" ... the one phrase my tutor kept saying anytime he noticed we were confused. Well do not be, just go through the rest of the guide, I will try to explain as best as I can, as if I am trying to convince my lecturer or whoever is reading this to give me a 100% score for this guide 😊.

Structure of an HTML Document/Script

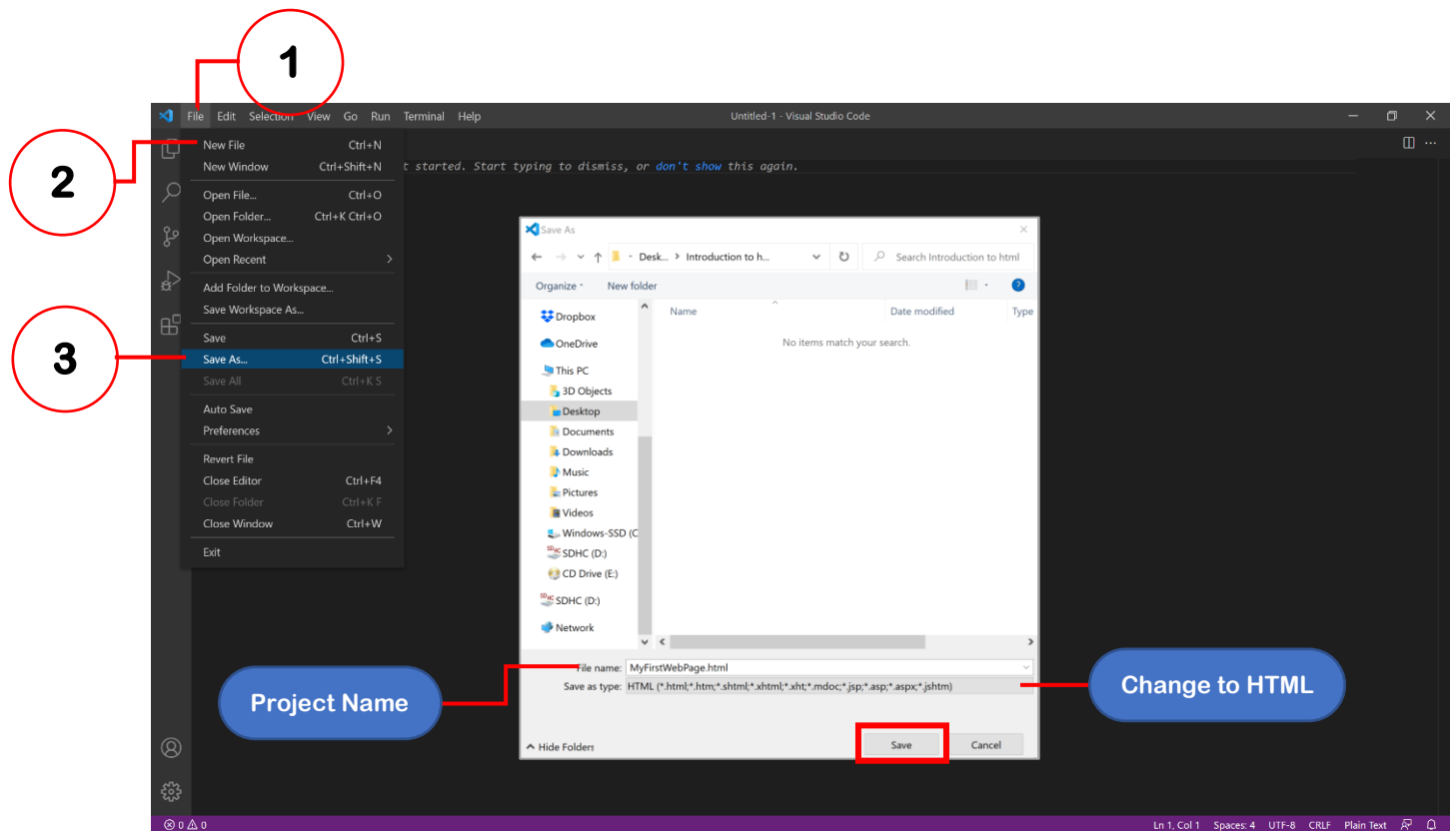
The Html script is more like a formal letter. It has specific features and feature locations that are similar to a formal letter. To explain, I will use the formal letter template below:



The above doctype is a formal letter. In the case of html the doctype is an html usually expressed as `<!doctype html>`

Now to get to creating our first html page, go to your installed text editor (in my case, Visual Studio Code). Go to:

1. File
2. New file (this is to create a new file)
3. Save As (.html)
4. Click save



Now we have created the html document (MyFirstWebPage.html) and will have to build the internal structure and content for the webpage

The HTML Structure

```
<!DOCTYPE html>
<html lang="en">
<head>

</head>
<body>

</body>
</html>
```

The html should have a **<!Doctype html>** this enables the browser and the text reader identify that the text inserted is a hyper text markup language.

The Structure Must also contain a **<html></html>**, all information must be within this tag. Just as we saw from the beginning and end of a letter, the first thing should be the users address, in this case html, and the last salutations and farewell should also be html. Thus, everything starts and ends with html.

In between the **<html>** and **</html>** is the head and body of the document.

The head contains all technical details of the webpage. This information is not usually seen on the webpage, except that they are styles, e.g., colours and borders.

The body contains the main content that appears within the page. For every paragraph, table, image, or list that has to be on the webpage, that content will have to be scripted within the body.

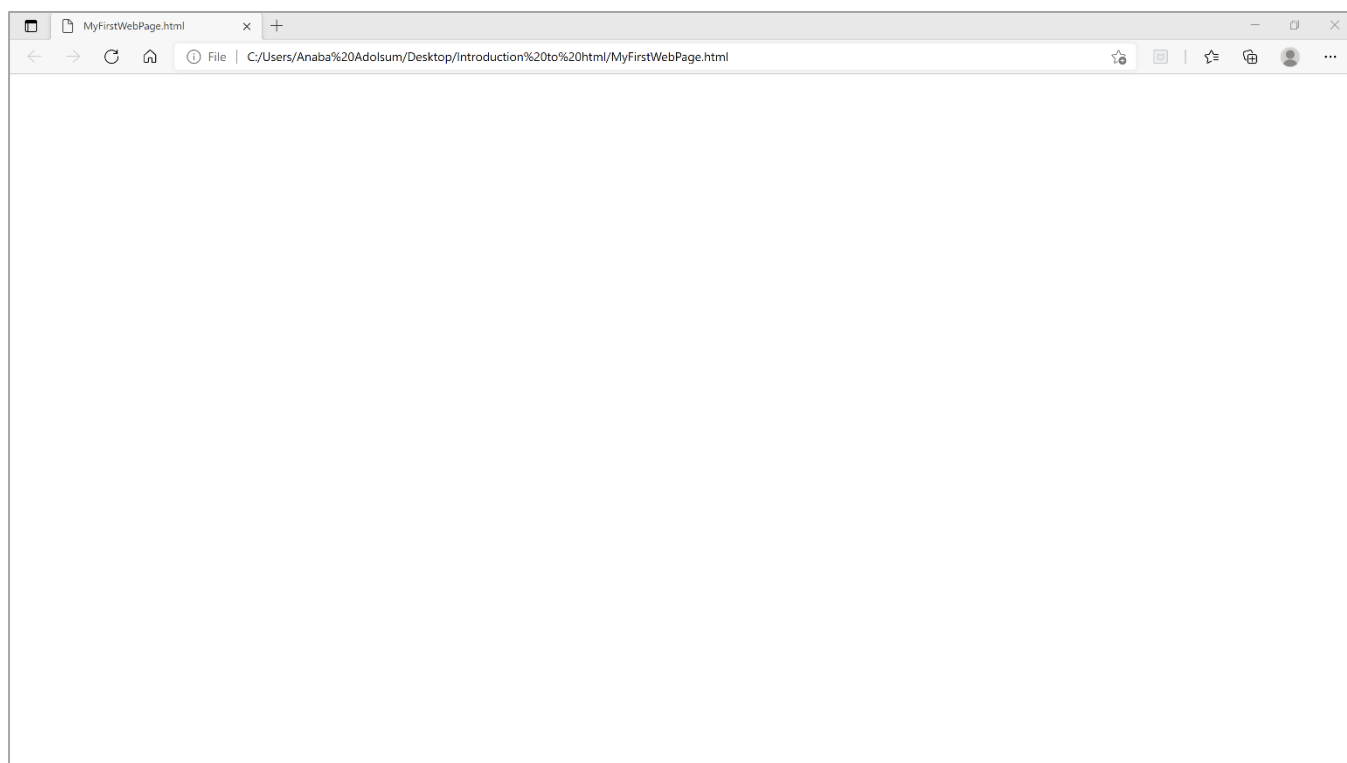
I hope you are beginning to understand the concept now, if you haven't, don't worry. We are now going to start adding the content.

it was at this point I opened my eyes after a long blink 🙄 (my colleagues claimed I was asleep 🤪), but they were wrong. I was meditating on the awesomeness I was going to put on my page 😊.

Everything in an html document must be written between a set of tags, within the head or body of the html document.

The above rule is the fundamental to making sure all intended inputs typed within the page works and displays as expected. There are over 121 different tags that can be used in an html script, this link [HTML Reference \(w3schools.com\)](https://www.w3schools.com/html/html_tags.asp) references a list of tags that would be needed for a beautiful HTML document.

Now let's find the HTML document we saved on our desktops. It should have the symbol of your default internet browser. Click to open it.



the opened web file would have nothing in it. Now we are going to add some content, with the use of tags.

Adding a Title, Paragraph and Images

```
<
!DOCTYPE html>
<html lang="en">
<head>
  <title>First Webpage</title>

</head>
<body>
<h1>IT'S ALL ABOUT HTML: The Basics</h1>

<img src= "html wordle.png">
<hr>

<p>welcome to my first html page. this page is an introduction to HTML, CSS and JavaScript. since I love working with maps, i might end up making this guide about WebGIS and incorporate some map elements to it. but for now, the focus would be on html. and building a beautiful webpage with basic html and css. Hope this website and the guide you are reading it from is helpful.</p>

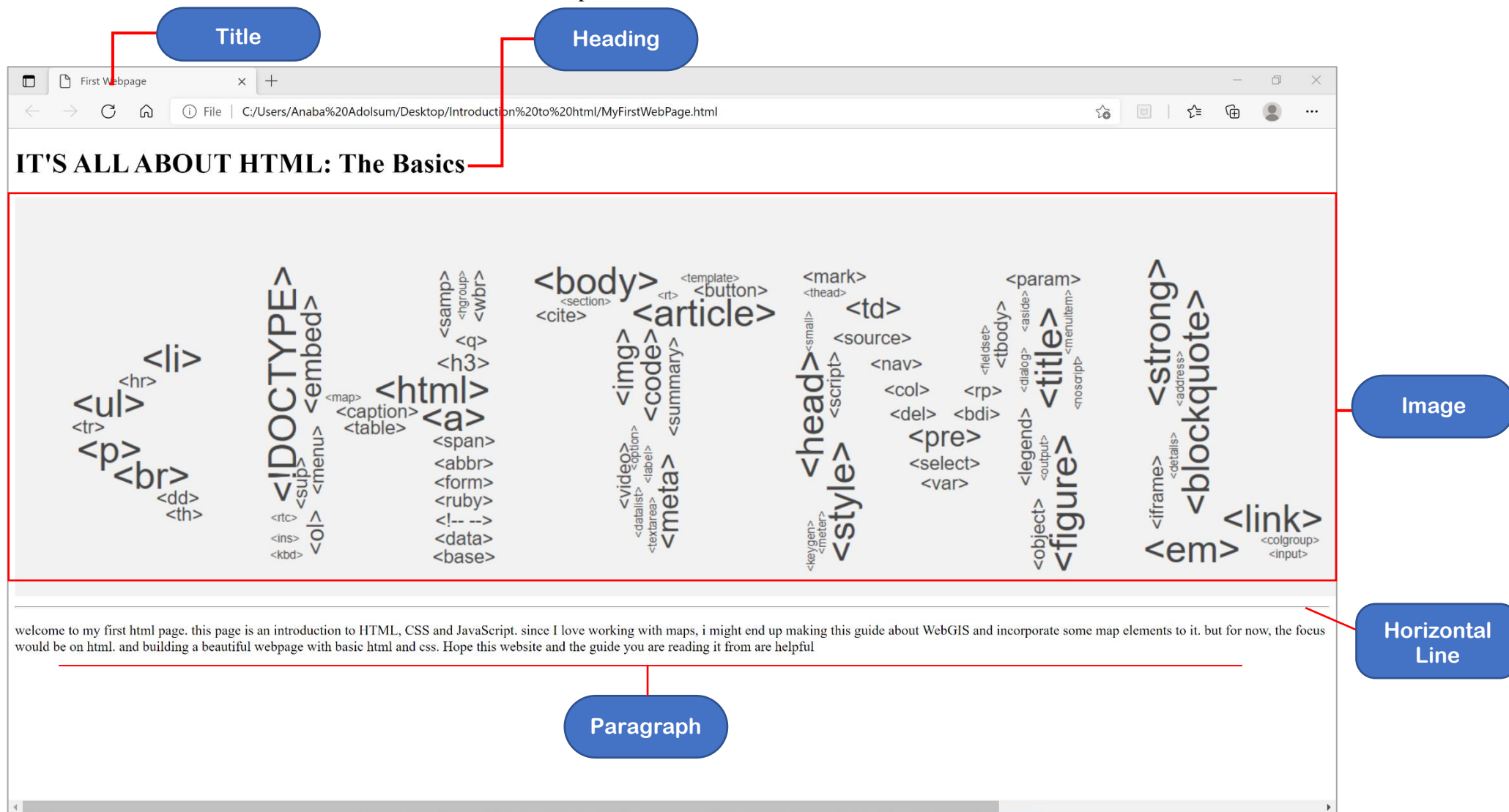
</body>
</html>
```

Building up on the structure of the html page we initially created, we are now adding some content. in the html doc above, I have added:

1. A title tag `<title> </title>`
2. A heading tag `<h1> </h1>`
3. An image link tag ``
4. A horizontal line tag `<hr>`
5. A paragraph tag `<p> </p>`

NB: be sure to save any additions to your html script before refreshing the browser.

Now to have a look at what we inserted in our script.



NB: it is advisable to keep images in the same folder as the project html document. This makes tagging/ referencing them in the script easy. In the case where the image is within the html document folder, you only need to reference the image name, and image extension.

```
Ex: <img src= "html wordle.png">
```

“html wordle” =I mage name,
.png= extension type

In the event where the image is in a different folder, then you will provide the full directory, as depicted in the syntax below

```
Ex2:<img src= "C:\Users\Anaba Adolsum\Pictures\html wordle.png">
```

Images can also be linked from online sources. All you need to do is replace the image directory with the image url. As depicted below:

```
Ex3:
```

Adding Lists

There are two types of lists in html, i.e., an ordered list and an unordered list. The difference between these lists is that whiles one is numbered, the other is in bullet points.

How to add lists

Unordered List	Ordered List
<pre> item item </pre>	<pre> item item </pre>

So currently, in our body tag we have:

```

<body>
<h1>IT'S ALL ABOUT HTML: The Basics</h1>

<img src= "html wordle.png">
<hr>

<p>welcome to my first html page. this page is an introduction to HTML, CSS and JavaScript. since I love working with maps,
    i might end up making this guide about WebGIS and incorporate some map elements to it. but for now, the focus would be on
    html. and building a beautiful webpage with basic html and css. Hope this website and the guide you are reading it from is helpful.</p>



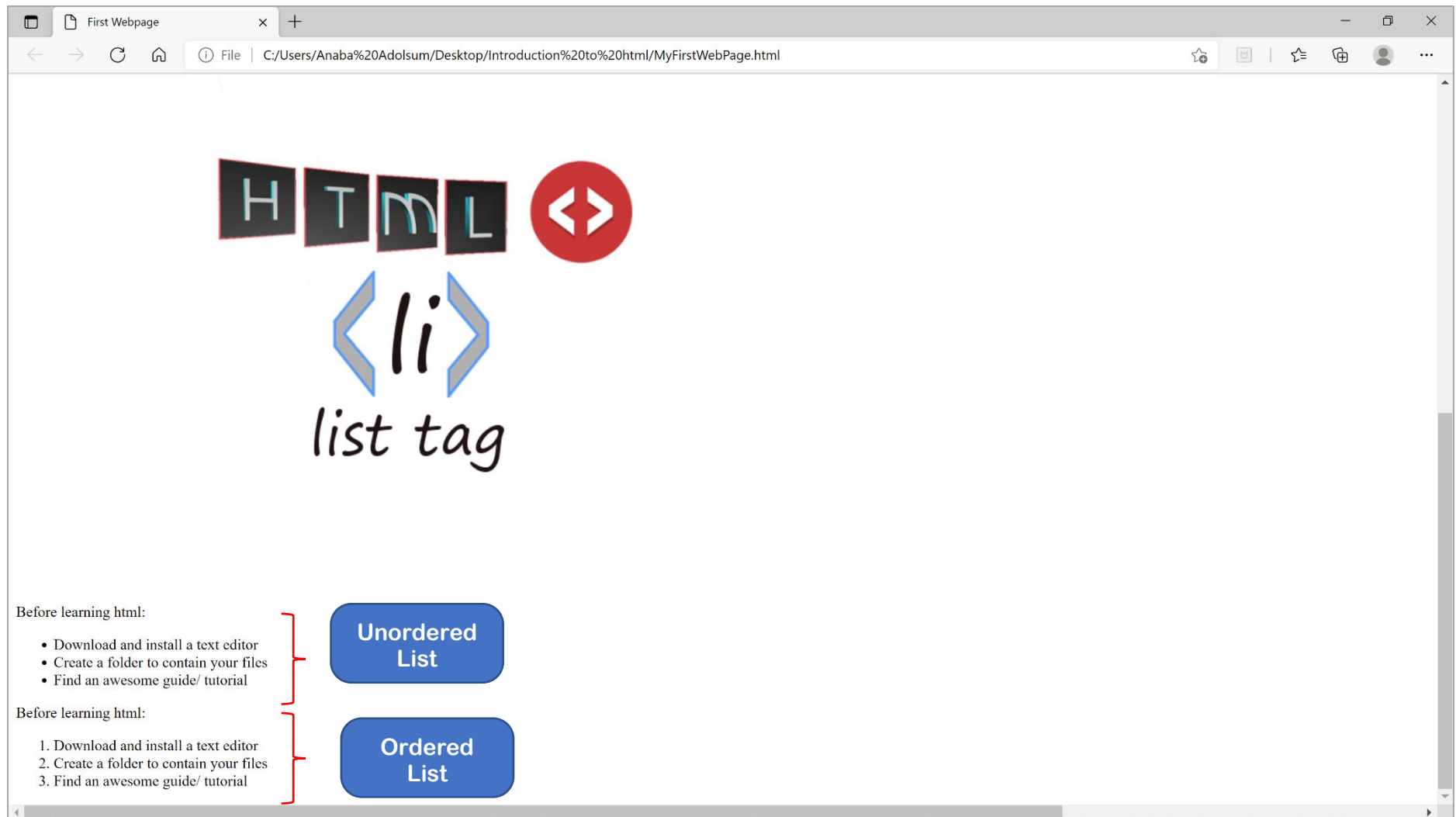
<p>Before learning html:</p>
<ul>
    <li>Download and install a text editor</li>
    <li>Create a folder to contain your files</li>
    <li>Find an awesome guide/ tutorial</li>
</ul>
} Unordered List

<p>Before learning html:</p>
<ol>
    <li>Download and install a text editor</li>
    <li>Create a folder to contain your files</li>
    <li>Find an awesome guide/ tutorial</li>
</ol>
} Ordered List

</body>
</html>

```

😊 hope you are able to add some content at this point, and also comprehend the use of tags to add information to the webpage. If you do not still understand at this point, kindly send me your address.... I shall come over and take you through it personally 🤖 😁 😎.



Adding Tables

Heading	Heading	Heading	← Row 1
<data>	<data>	<data>	← Row 2
<data>	<data>	<data>	← Row 3
<data>	<data>	<data>	← Row 4

Tables are literally a stack of rows and columns containing data. Unlike MS Word or Excel where we can click and insert tables, in html, we would have to create each stack of rows with the necessary data within it. Thus, you would have to create the table row with the headings and the table rows with the data.

```
<table>
  <tr>
    <th> put heading here</th>
    <th> put heading here</th>
    <th> put heading here</th>
  </tr>
  <tr>
    <td> put data here</td>
    <td> put data here</td>
    <td> put data here</td>
  </tr>
  <tr>
    <td> put data here</td>
    <td> put data here</td>
    <td> put data here</td>
  </tr>
</table>
```

Row 1, with Column headings

Row 2, with data/ information

Row 3, with data/ information

So now to create a table in our webpage to contain our information

```
<table>
  <tr>
    <th>ID</th>
    <th> Name </th>
    <th>contact</th>
    <th>email</th>

  </tr>
  <tr>
    <td>1293857</td>
    <td>Valentine Anaba</td>
    <td>0540997791</td>
    <td>contact@invclick.com</td>
  </tr>
  <tr>
    <td>1293857</td>
    <td>Adolsum Anaba</td>
    <td>0540997791</td>
    <td>valentine.anaba@invclick.com</td>
  </tr>
</table>
```

In the syntax above, I have created a table with 3 rows and 4 columns.



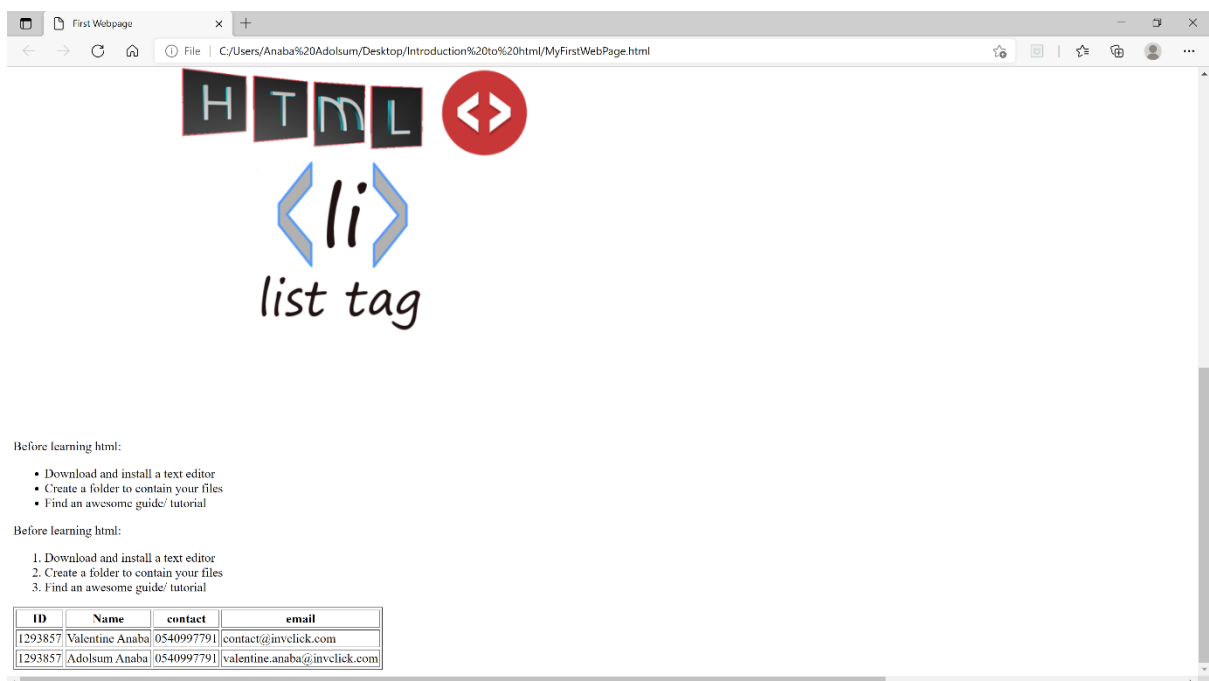
As I mentioned earlier, html provides data in raw text. But with the help of CSS, we can style and update the table to show the borders, add some space, and make the table look beautiful.

Before we proceed to CSS, let me show you how you can add an in-text CSS to the script.

In the opening tag of the table, edit or replace with the syntax below:

```
<table border="1", cellspacing="3",cell padding="20">
```

The above syntax adds a border, with a cell spacing of 3, and a cell padding of 20. Notice how the new table has borders now 😊.



Adding Divs

The div tag is a very important and useful tag, used in dividing and segmenting the webpage elements. It is used to group html tags, more like a container for flow content. for instance, in our script, I could put our list tags withing a div tag, so as to create a section.

Divs are used in the body to contain the elements that belong to a particular session of the page to make styling earlier.

Before you frown your face and go “what the heck is a div?”. and start throwing a tantrum, and talking to the text on your screen 🤔, kindly see the next page. Just try it out

We are going to edit the tags we already set-up with div tags `<div></div>` with a little in-line css, so that we can see, appreciate and understand div.

```
<div style="background-color: rgb(206, 233, 233);" >
<p>Before learning html:</p>
<ul>
  <li>Download and install a text editor</li>
  <li>Create a folder to contain your files</li>
  <li>Find an awesome guide/ tutorial</li>
</ul>

<p>Before learning html:</p>
<ol>
  <li>Download and install a text editor</li>
  <li>Create a folder to contain your files</li>
  <li>Find an awesome guide/ tutorial</li>
</ol>
</div>
```

Notice how we put the opening div tag `<div>`, above the paragraph `<p>` and the closing tag after the closing tag of the ordered list ``. What it basically did was to put the paragraphs and list within a div, to create a *division* or create a section. The `style="background-color: rgb(206, 233, 233);"` is a CSS style. (We shall get to that shortly)



See how in the webpage, all the elements that were captured within the div have a coloured background 😊. The edge of the coloured background is the border of the division.

It is important to include div's on a webpage to make it easy to apply cascading styles to elements in the div more easily.

I was going to add one more thing, but I think we can skip to CSS now... I know no one is thinking “what was he going to add?” ... well just so you know, after I learnt to this point, I started scouting for web design gigs. And yes, I updated my CV to include webpage development 🤖🤖🤖.

Before moving to CSS, let's see our script so far:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>First Webpage</title>

</head>
<body>
<h1>IT'S ALL ABOUT HTML: The Basics</h1>

<img src= "html wordle.png">
<hr>

<p>welcome to my first html page. this page is an introduction to HTML, CSS and JavaScript. since I love working with maps,
  i might end up making this guide about WebGIS and incorporate some map elements to it. but for now, the focus would be on
  html. and building a beautiful webpage with basic html and css. Hope this website and the guide you are reading it from is helpful.</p>


```

```

<div style="background-color: rgb(206, 233, 233);" >
<p>Before learning html:</p>
<ul>
  <li>Download and install a text editor</li>
  <li>Create a folder to contain your files</li>
  <li>Find an awesome guide/ tutorial</li>
</ul>

<p>Before learning html:</p>
<ol>
  <li>Download and install a text editor</li>
  <li>Create a folder to contain your files</li>
  <li>Find an awesome guide/ tutorial</li>
</ol>
</div>

<table border="1", cellspacing="3",cell padding="20">
  <tr>
    <th>ID</th>
    <th> Name </th>
    <th>contact</th>
    <th>email</th>

  </tr>
  <tr>
    <td>1293857</td>
    <td>Valentine Anaba</td>
    <td>0540997791</td>
    <td>contact@invclick.com</td>
  </tr>
  <tr>
    <td>1293857</td>
    <td>Adolsun Anaba</td>
    <td>0540997791</td>
    <td>valentine.anaba@invclick.com</td>
  </tr>
</table>
</body>
</html>

```

CSS-

Cascading **S**tyl**e** **S**heet**s**

Intro to CSS

CSS basically acts as a style/symbology for HTML. Whereas Html is more of the building block of the webpage, CSS acts as the design aspect of the webpage.

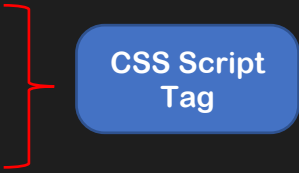
There are 2 ways to use CSS

1. Create style tag `<style></style>` within the head of the html, where all CSS syntax will be placed.
2. Link an external CSS file to the html page.

Continuing with our initial html script, we are going to use the first approach for using CSS.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>First Webpage</title>

  <style>
  </style>
</head>
```

A diagram consisting of a red curly bracket on the left side of the code block, spanning the lines containing the `<style>` and `</style>` tags. To the right of the bracket is a blue rounded rectangle with the text "CSS Script Tag" in white.

Now that we have our style tags, all CSS syntax will be typed/ inputted within those tags.

The components of a cascading should include:

1. **The Element:** the element refers to whatever is usually in the `< >`, e.g., `h1`, `p`, `hr` etc.
2. **The property:** this refers to what you would want to change e.g. font, colour, placement etc.
3. **The Value:** to what you want to change it to.

Element Value

Ex: `p {font-family: Arial;`

Property

Adding some style to our html [Elements and ID's]:

```
<head>
  <title>First Webpage</title>

  <style>
p {font-family: Arial;
  color: chocolate;
  font-size: larger}

</style>

</head>
```

In the style tags, in the head of the html, we have defined the font type, the colour of the font, and the size of the font.

NB: *the element stated was the p, representing all paragraphs in the body of the html. Thus, if we apply this style, it will affect all content tagged as Paragraphs <p> in the body of the html.*

To apply and effect changes to specific elements, it is necessary to provide IDs, so that they can be easily referenced. For instance, I would like to make my very first paragraph black, with a different font size. What I would need to do is to provide that paragraph with an ID. So, in the body of my script, I can provide the id, as shown below:

```
<p id="first">
welcome to my first html page. this page is an introduction to HTML, CSS and J
avaScript. since I love working with maps,
  i might end up making this guide about WebGIS and incorporate some map ele
ments to it. but for now, the focus would be on
html. and building a beautiful webpage with basic html and css. Hope this webs
ite and the guide you are reading it from is helpful.</p>
```

then in the head, I will reference the id.

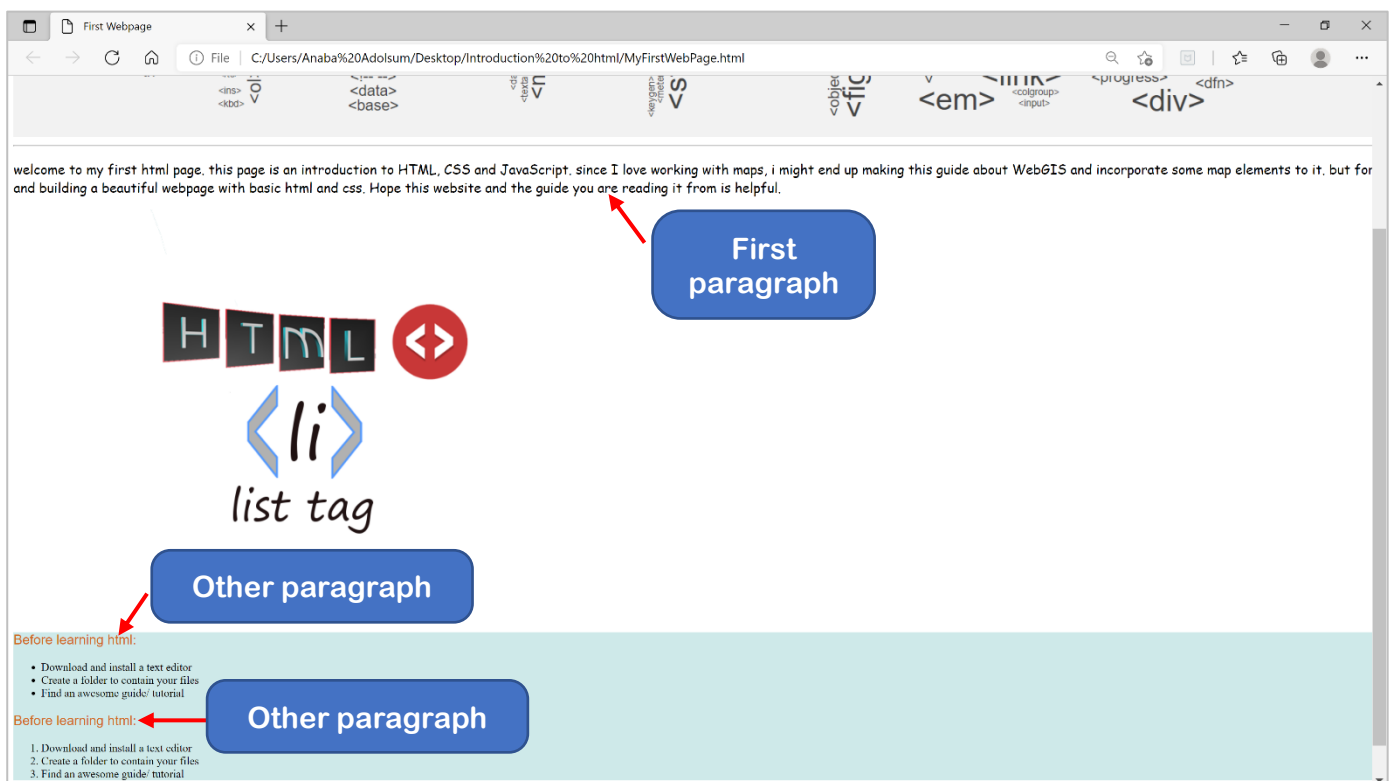
```
<style>

p {font-family: Arial;
  color: chocolate;
  font-size: larger}

#first {font-family: cursive;
  color:black;
  font-size: larger;};

</style>
```

The id is referenced with **#** then the name of the id, **first**. Now when this script is run the first paragraph will be black, but the others will change to chocolate.



Working with Classes

Classes and ID's are just like they sound. Not in the sense that you have to go get your school ID, but in the sense that, a class is relatively a collection of IDs for a particular element.

Let's say I am a student in the Computer Science class, with an id of 1000000. I am not the only one in the class, there are also some other 30 students with similar ID's. now the HoD has some jerseys to share on campus. He tells the facility manager to send all gold jerseys to the computer science class. Guess what happens.... We all get gold jerseys.

Now during the distribution of the jerseys, the HoD comes with the class grades. He has an extra gift for the best student. So, whiles we are in class he says, 'the best student, who gets the golden pen is student ID 1000000'. In this case only the student with that ID gets the golden pen. Everyone else in the class is stuck with just a jersey.

Just as expressed above, classes and ID's are used to reference and apply styles to specific elements (ID's) or group of elements (class).

When creating styles, the effect of the style affects all selected elements. For instance, I have 8 paragraphs in my webpage and want to change the colour of the text in all the paragraphs, I could simply type:

```
p {color: red;}
```

and all paragraph text will change to red.

Now let's assume, the content of 3 of the paragraphs were about html, another 3 were for JavaScript, and the remaining 2 where about CSS, I could simply put each of these paragraphs in classes. In the body tag, I could reference the paragraphs as:

```
<p class="html">this is about html and its basics</p>
<p class="java">this is about java and its fundamentals</p>
<p class="css">this is about css and its essentials</p>
```

Notice how each paragraph has a different class.


```
<p class="html">this is about html and its basics</p>
<p class="java">this is about java and its fundamentals</p>
<p class="css">this is about css and its essentials</p>
<p class="html">this is about html and its terminologies</p>
<p class="java">this is about java and its intrisit parts</p>
<p class="css">this is about css and its essentials II</p>
<p class="html">Advance html..for masters only</p>
<p class="java">Applied java for dumies</p>
```

Above are my 8 paragraphs, each containing different information, but some having similar classes.

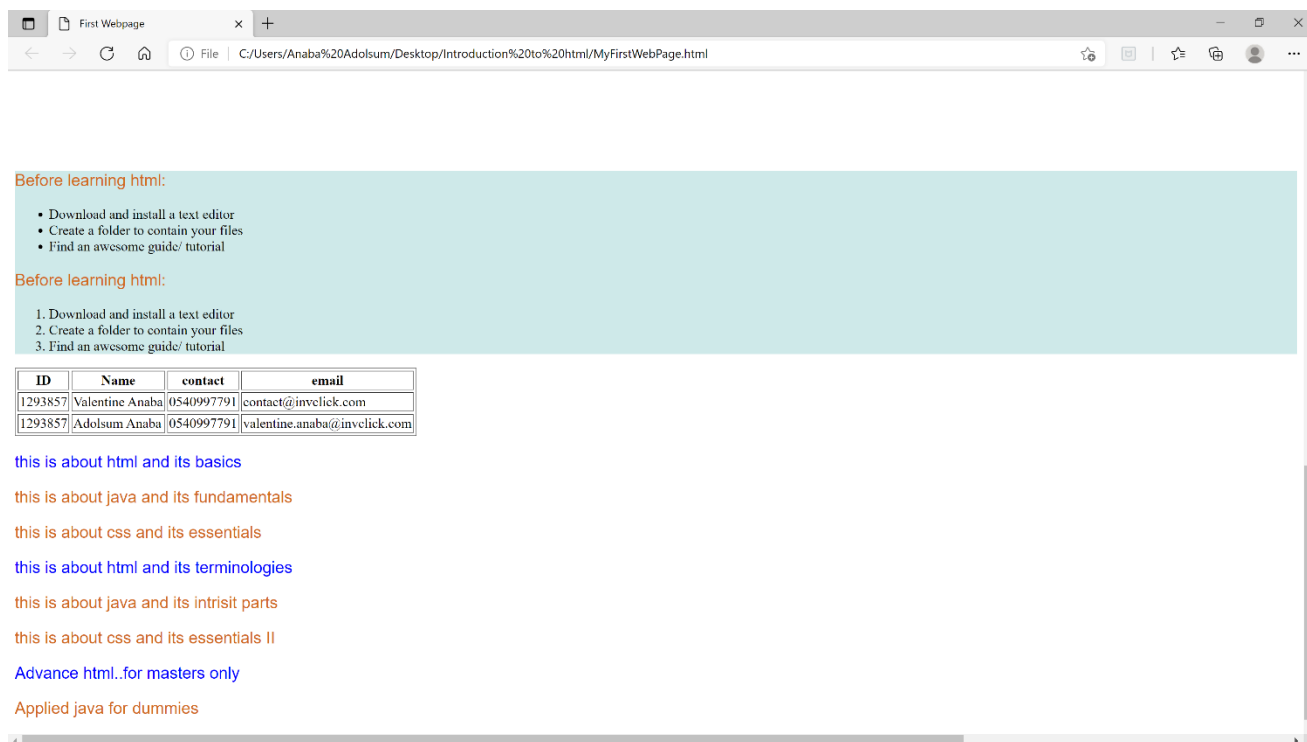
NB: *the section displayed above is in the body of the html script and not the head.*

Now to apply styles to elements of a class, all I need to do is to call out the class in the head of the html tag, in between the styles tag with the use of a full stop sign (.).

```
.html{color: blue;}
```

In the syntax above, the html class will be given a colour value of blue.

NB: *do not add the period/full stop (.) in the body.*



The instance of ID's is similar to that of class, but instead of referencing it with a period sign, it is referenced with the pound sign (#).

ID's are more specific. Each member of a class can have independent ids to affect changes to only individual elements of the same class. Kindly see the earlier examples on how Id's work.

Pseudo Styles

CSS Pseudo-elements are defined as a special selector that adds special styles to an element without changing any existing document. Pseudo-styles are uniquely applied to certain parts of a script e.g., the first letter in a paragraph, the last line, the first line etc. Pseudo styles usually add cool aesthetics to a webpage.

Below is a list of commonly used pseudo style elements.

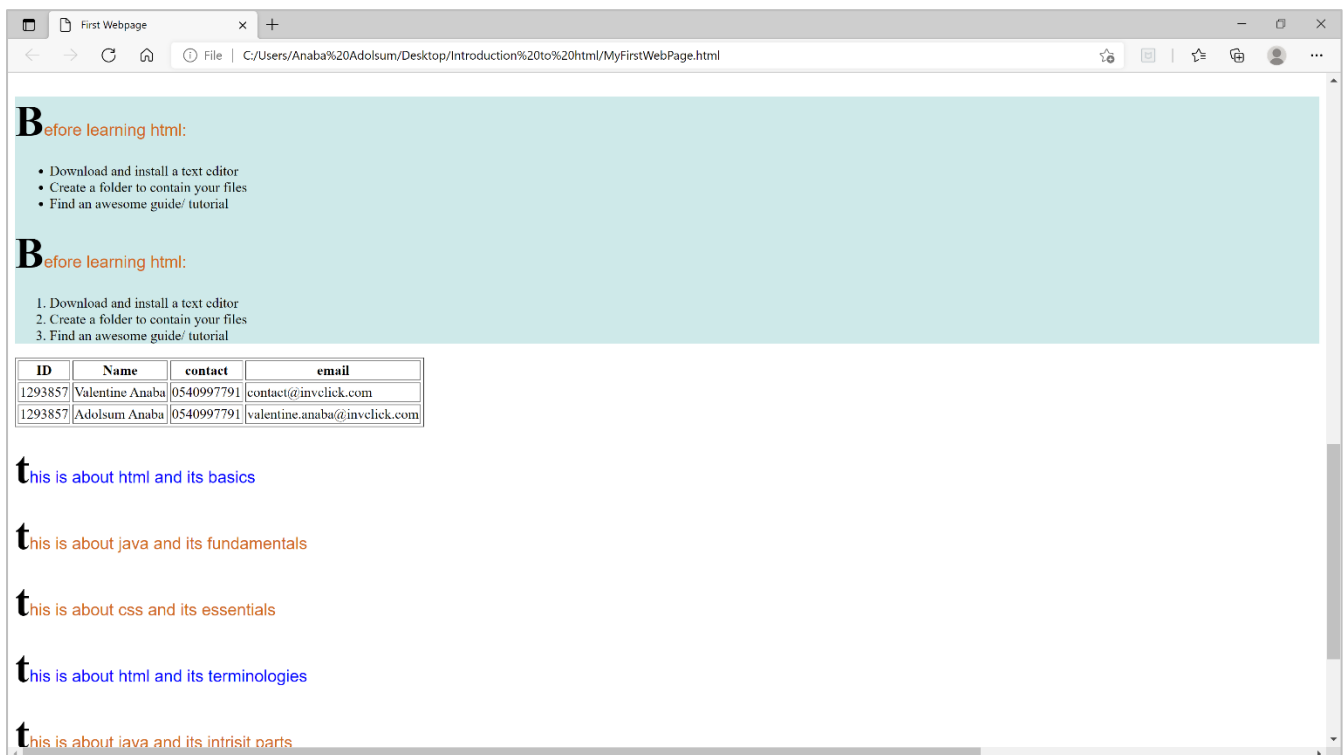
Value	Description
: before	This element is used to add a few contents before an element with the need for HTML.
: after	This element is used to add a few contents after an element.
: First-letter	This element is used to insert special kinds of styles to the first letter of the text in a box specified in a file.
:first-line	This element is used to insert special kinds of styles to the initial line of the text.
: Selection	It Highlights the Selected Portion of the Content.

Let's take an example. I want to bolden the first letter of every paragraph and make them a bit bigger than the other text.

In this case I will use the: **first letter** pseudo element, to apply this element to all paragraphs

```
p:first-letter{font-weight: bolder;  
font-size: 50px;  
color:black;  
font-family: 'Times New Roman';}
```

in the syntax above, the first letter of each paragraph will have a boldened font, with an increased size of up to 50px.



Positioning

Elements in html are usually positioned as static by default. Static positions are not affected by top, bottom, left or right properties, but positioned according to the normal flow of the page. Noticed how the information we entered in our script, followed the sequence in which they were typed, including the images.

There are 5 different position values:

1. **Static:** An element with **position: static;** is not positioned in any special way; it is always positioned according to the normal flow of the page
2. **Relative:** Moves an attribute from its current position to a prescribed position based on pixel. Elements can be moved to the left, right, top or bottom of its original position.

3. **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.
4. **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.
5. **Sticky:** positioned based on the user's scroll position. A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position: fixed).

Example/Sample Codes for Positioning

```
div.relative {  
  position: relative;  
  width: 400px;  
  height: 200px;  
  border: 3px solid #73AD21;  
}
```

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

Positions can also be stated as

```
P {position: relative;  
Top: 0px;  
Width: 100px;  
}
```

External Style Sheets

External Style Sheets are (.css) sheets that contain certain rules/styles that can be applied to a webpage. These styles are pre-stated outside the html document but can be integrated and applied to a particular site.

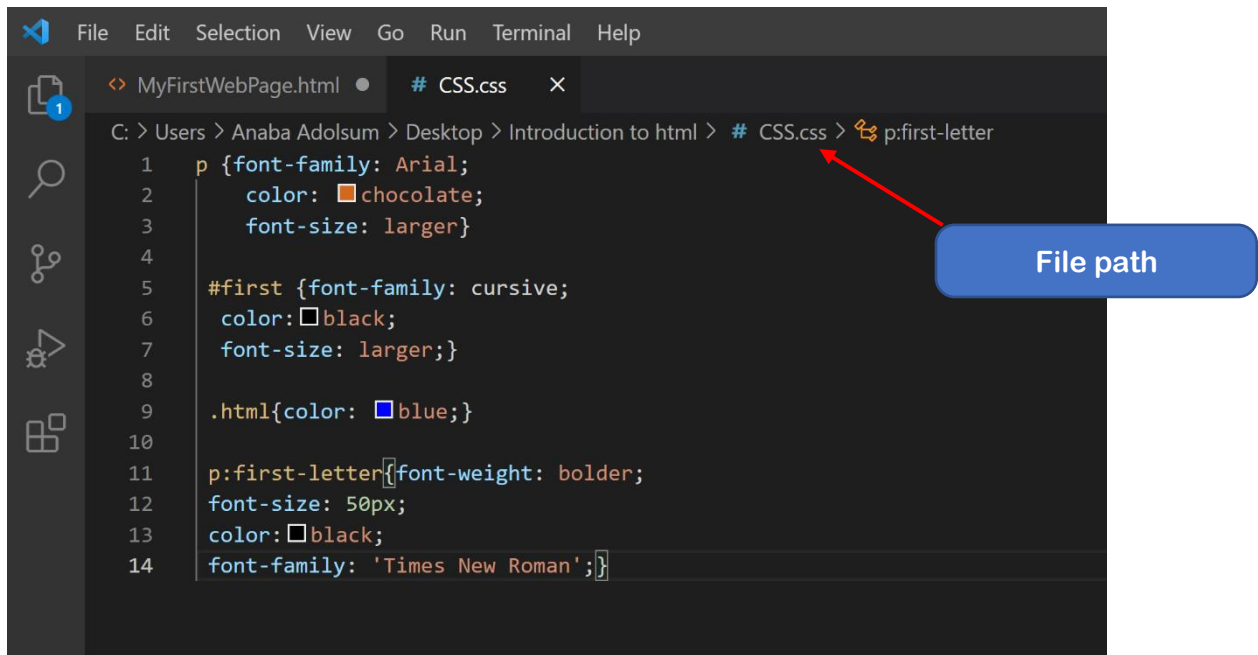
For instance, you just finished working on the body of your webpage but came across a beautiful style sheet with nice fonts and background colours that you would wish to apply to your page. Instead of having to type each specific code in the head of your html script, you can easily link the CSS file to the html, to adapt the styles.

If the file is already in CSS, then you can proceed to link it in the html file, else you would have to create a new sheet > copy and paste the rules, then save the file as a css.

Once that is done, reference the file in the head of your html:

```
Ex: <link rel="stylesheet" href="name of file.css">
```

To clearly depict the example, I am going to copy and paste the styles we created in CSS. Then link the style sheet in the html script



Now that the CSS is created, I am going to link it to my html.

NB: Codes in the CSS file do not have to be in any style tags. The CSS file should be saved as a **.css and not .html**

In the example above, the CSS file is within my HTML work folder, so all I have to do is specify the name of the CSS file. As depicted below

```
<link rel="stylesheet" href="CSS.css">
```

In the case, the CSS file was in another folder, I would have to specify the file extension, as shown below.

```
<link rel="stylesheet" href="C:\Users\Anaba Adolsum\Desktop\Introduction to  
html\CSS.css">
```

NB: Any changes made in the CSS file will reflect in all the sheets it has been linked to.