**Contents for Html Tutorial**

1. Introduction
2. Text Editor
3. Elements
4. Head
5. Body
6. Headings
7. Paragraph, line break
8. Bold, italic, strong and emphasis
9. Span, Hr and blockquote
10. Parent and Child Relationship
11. Comments
12. Lists
13. Attributes
14. Styling(CSS)
15. Colors
16. Font size and Family
17. Text styles
18. Margin
19. Anchor
20. Display
21. Buttons
22. Padding
23. Images
24. Size
25. Divs
26. Borders
27. Ids and Classes
28. Float
29. Positioning
30. Tables
31. Form
32. Semantics
33. Responsive
34. Charset
35. JavaScript
36. Iframe
37. Canvas
38. Svg
39. Pictures
40. Audios
41. Videos

**Introduction**

Html is a markup language used to create web pages.

Html can be used to build simple website for business and others, it can also be used to build complex websites.

It also has a way of connecting with other languages to have a better and more modern website.

Html is also easy to learn and doesn’t involve much brain processing since it just a markup language. Therefore you are encouraged to learn html even if you don’t want to be a programmer by profession or even if you don’t want to go into d programming business. Html is a skill everyone would like to learn even just the basics.

Creating a website using html involves writing out the code in text editor and viewing it in the browser

It also has a way of connecting with other languages to have a better and more modern website.

Html can add images and videos into your web content.

You can also create tables and even navigate through other website. You can create forms of different categories and aspects.

This tutorial will help you learn html at a grand scale. It will help you to become acquainted with tags and other processes you need to build websites.

There are examples and exercises that will help you in each tutorial to have a practical guide and to become more efficient with html. We hope you enjoy this tutorial.

**Text Editor**

To begin with Html you need a standard text editor. We recommend sublime text 3 because it has native supports for most programming languages. It helps in auto completion, syntax highlighting – coloring of code accordingly to help you differentiate and code folding.

So download sublime text 3 if you haven’t already done so and open it.

After opening it from your windows in your pc, create a new file using “ctrl + N” or the file menu and type some code into it.

Type “Helloworld” into your new file.

Then save the code using the index.html (note the “index” is the standard home page of every html file. You can save it with any other name).

Always remember to add the “.html” it helps the computer to know that your file is a html file.

Then right click and click the option open in browser to view your file.

After addition of new code in your file and saving it simply go to your browser and refresh the page.

Note: Simply adding helloworld is something you do not want to do in your real html file. The next chapters will deal with that

You are now good to go

‘

**Elements**

What are html elements?

Html elements are type of html that help you format the content of your web page.

Html elements consist of a beginning tag, contents and an end tag.

Tags are the keywords you write that formats your html contents. It is written with as such; <my tag> content </my tag>. Notice that it has a beginning and an end tag and that the end tag containers a / sign. That is to show that this is the end or closure of the tag.

Some elements might not have a closing tag just an opening one. So you can close them as such: <my tag />. But this is optional.

Html pages mostly consist of different elements. It is those elements that define or format your webpage.

Some elements also have attributes but that would be discussed in a later chapter.

On your editor the first thing to type after creating your new html file is the <!DOCTYPE html>. It specifies what version of the html it is going to use.

The next step is to write the <html> tag. The html tag is like a container for your code so all your code is going to be inside the <html> and at the end of the code, you close the tag.

The next chapter discusses more..

**Head**

The head is the part of the html page that defines the properties of the pages but doesn’t change the content. It serves a container for all the elements that provides details about the page.

To create it the head simply type in the head tag <head>. The head element include the title of your page- that is the title the browser shows when your page loads. So create the title by opening the <title> tag writing the title of your choice and closing it.

The <meta> tag is also found in the head of your html file. It specifies the character type of your html file. There are also stuffs like the meta name and others. More about that will be discussed in a later chapter. For now set the meta tag to <meta charset utf-8>. The meta tag is self-closing meaning it doesn’t have a closing tag so you can chose to leave it the way it is or self-close it as such: <meta charset utf-8 />, anyhow the code still runs.

Also external files are mostly linked in the head element. External files like your CSS and JavaScript, though we would come to that in a later chapter.

After that you close the head tag.

**Body**

Let’s go to the main code that formats the contents of your html page. Remember the head elements houses the meta code — the part that provides details for the page but doesn’t change the code. But the body element does the exact opposite, it houses the code that adds content to your html page.

To add the body element simply start with the opening of the body tag - <body>. All the code to format the page follows, after which you close the tag.

Html doesn’t take note of white spaces or indentations or new lines. But good coding ensures that you use white spaces, new lines and indentations properly. It helps to improve the readability of your code. So the content of any html that spans more than one line should be moved to the next line and indented using 2 spaces or 4 spaces depending on your choice. Also nested html elements should also be indented properly using two or 4 white spaces. It helps to identify elements opening and closing tags in case you have two same elements nested within each other.

This is where the tutorial becomes serious, because this is where we start adding content to your html page so brace yourself for real action. You can also style your body element in different ways, changing the background color, margin and even adding padding with your css. So get ready …. The next chapters will focus on adding headings to your html pages..

**Headings**

Now in html we have six different types of headings that differs with size. Starting with the biggest to the smallest, the biggest heading is the <h1> tag, and the smallest is the <h6>. So we have the h1, h2, h3, h4, h5, h6 headings, each smaller than the previous.

Open the <h1> tag, write the content of your heading and close it as such:

After closing it reload your browser and refresh the page.

Do the same for the rest of the heading tags.

Take note that the h1 tag is not only bigger than the h2 but also more important. Means that when searching, the search engine takes note of the bigger headings before the smaller ones. The reason being that CSS can change the style of a smaller heading like the h6 and increase the font size to be very large. So the search engine looks for the h1 before the h6 whether it is styled larger or not.

The next chapter discusses about the paragraph tag.

**Paragraph, line breaks**

This is the part of html that enables us to add text to the page. It might not necessarily be a paragraph, just one line, one word, or whatever text you add is considered as a paragraph in html.

We use the <p> tag to write paragraph in html. Now open the <p> tag and write this text: this is my paragraph; and close it. Now refresh the browser and see what happens.

So the p tag formats the text of your web page. But the thing is, as we said earlier html does not recognize new lines, so if you press the enter button when typing your text content or you close the p tag and open it again, it still continues the text on the same line. To test this, type the following text in a new p tag:

This is my second paragraph

Now this is a new line;

Reload the browser and check it. Obviously it continued the two lines on the same line.

To solve this html provided what is known as the line-break element or the break element using the <br> tag. The <br> tag is self-closing – it doesn’t have a closure tag. So when typing and you want to move to a new line or insert a line break, simply add the <br> tag to your code and it moves to the next line.

**Bold, Italic, Strong and Emphasis**

Html also provides ways to add a little bit of styling to your text content.

For example if you want a certain part of your text to be bold, you simple use the <b> tag. Remember you can always restyle the font-size of the text using your css, but the bold element makes the text darker. Simply open the <b> tag at the beginning of the point you want to be bold and close it at the end. But remember you must have to use to <p> to ensure it is a paragraph.

The same can be done if you want to some part of your text to be italic. It is done using the <i> tag. It follows the same procedure of opening it at the start and closing it at the end of the part you want to make italic.

We also have the strong and emphasized text. The strong text looks like bold text and the emphasized looks like italicized text. But the thing is, they are not. The strong and emphasized text are phrase tags. They are used to make a phrase stand out and browsers traditionally interpret them as either being bold or being italic. Of course, their behavior can always be restyled in anyway using the css.

To add strong text use the <strong> tag, opening and closing it at the beginning and end of the phrase respectively.

To add emphasized tag to a phrase, you simply use the <em> tag at the beginning of the phrase, and close it at the end just as you do with the strong text.

**Span, Blockquotes and Hr**

In paragraphs tags, sometimes, we might want to add styling to some part of the paragraph, only some part and not all. Sometimes we might even want to program some part of a paragraph, using javascript, in a different way than the others.

That is where the element span comes in. It helps to differentiate a part of the paragraph from the rest.

To use it simply open and close the <span> tag and write the text content inside. You can then do whatever you want with the spanned text – styling, programming, etc.

Blockquote tags are element that helps to house quotes. Quotes from great men, proverbs, adages and other local quote can be the text content of blockquotes.

The irony of it is – blockquote doesn’t add any special feature to your quote. Like span, it just helps to differentiate the other text from the quote. But they are very useful in the readability of your program and good coding ensures that you use them to enclose your quotes.

Using the blockquote, you open it with the <blockquote> tag and after adding your code, close it.

The styling of the code can take place using the css.

Hr tags are used to insert horizontal line to your text. It helps for separation of paragraphs or like a horizontal border. Like the <br> tag, the <hr> doesn’t require a closing tag. You just insert it before or after the paragraph, where you want it to be, and see what it does.

**Parents-Child Relationships**

Parents and child relationships are very common in Html.

The most adult, oldest or the ancestor in html is the html element. This is because it is not indented at all. So everything else is a child of our html element. Though, you might chose to leave some of your html element un-indented which is not compulsory, but for good coding, element nested in other elements are meant to be indented.

So each nested elements, elements within elements, are children of the elements they are nested in. Means the head and body element are children of the html elements. And all other elements like the h1 and others are children of the body elements. When elements are nested in divs, they become children of that div, and span elements are children of the elements they are in.

However, remember that parent elements like div are still children of the body element and the body element is a child of the html element. So you can think of it as a family tree. So the div elements are descendants of the html element and the html element is its ancestor.

Also, descendant and ancestors are used in far relationships like the div and the html, but close relationships like the div and the element within it are parent to child relationships.

Other elements that are not nested in others but are both children of an element share this brother and sister - sibling’s relationship.

You may wonder why you need this or why it is vital for you to know this. At this stage, it is important for you to understand the parent –child relationships. But in the future, if you want to advance to JavaScript, parent-child relationship will be very required, especially when searching through your html page with the JavaScript.

So by yourself, look into some html code and try to answer which is a parent, child, ancestor, descendant and siblings so you can be familiar with it.

**Comments**

Comments is a very useful tool and is common to every programming language you come across. Comments are those part of the code that the browser or whatever runs your code ignores. They are not meant to be run, but instead serves the purpose of documenting.

Now that may seem like a less useful purpose, but believe me it is very useful. Should incase after a long time you want to get back to your code, comments helps you to remember immediately what your code does. It also helps others who want to use your code or who want to just take a look at it to understand what the code does – the main purpose of the code without you explain it.

So, in essence, good coding ensures that you use comment to document each and every detail of your work. It is not so stressful and it has much benefits.

In html, comment is done using the comment tag, it is written as such:

<!—this is my comment 🡪

If you typed that in your text editor, you noticed that the coloring is a bit deem than normal, to show that html choose to ignore that code.

Commenting can also span multiple lines as such:

<!—

Multiple

Lines

Comment

🡪

**Lists**

In html we have 3 types of list: the ordered list, the unordered list and the description list.

Ordered list:

The ordered list makes use of ordered numbering to number their list items – ie 1, 2, 3 ….

Writing the ordered list you first open the ordered list tag <ol> and then inside the list tag, you write the list item. Now each items are also written in the list item tag - <li>. After which you close the ordered list tag.

Unordered list:

The unordered list make use of unordered numbering like bullets, dashs, arrows etc to number each list item. Though by default it comes with bullet.

To create an unordered list, you simply open the unordered list tag <ul> and just as we did for the ordered list, create list items using the <li> tag which stores your list content, after which you close the <li> tags and the <ul> tag.

Description list:

This is a list of items followed by their descriptions. The description list doesn’t follow any numbering proceedures.

To create it you open the tag <dl>. Each list items is contained in the tag <dt> with a description under using the tags <dd>. Remember to close each tag. After which you close the <dt> tag.

**Attributes**

Attributes can be seen as a part, characteristic or feature of something. That’s how English sees it and its similar to how html sees it too.

Attributes are inherent features that are associated with certain elements or tags. Each element have distinct attributes that provide needed information about the elements.

Attributes are written in the beginning tag <tagname attribute> content </tagname>

Examples of attributes:

The src attribute for the img tag that specifies where the image can be found.

The href attribute for the a tag that specifies the link.

The width and height attributes that helps to set the size.

The style attributes that helps in styling our elements.

Attributes differ by the element and as we go further we will be dealing with more elements and attributes.

**Styling(CSS)**

You probably have been seeing the word CSS in our tutorial course and you might have been wondering what it really means. To start with CSS is an acronym for Cascading Style Sheet, and is a program that helps us style our html elements. To see a full tutorial on CSS you can check our CSS tutorials. But in the cause of this html tutorials, we are only going to be touching a few and basic part of CSS.

As we have earlier stated, CSS helps to add style to our html element. To do that we need to be able to write the styling down.

We can write our CSS in 3 different ways:

* As an inline styling
* As an internal styling
* As an external styling

As an inline styling:

We can write css code inline using the style attribute in the html element we want to style.

As an internal styling:

We can write our css code internally by using the <style> element in the <head> section of our html file.

The style element doesn’t have a closing tag so it can be self-closed. It also has an attribute type in which you can set the value to “text/css”

As an external styling:

We can write our css code in a different file, and link it to the html using the <link> tag in the <head> section. To do that you need to save your css file using the .css to specify that it is a css file.

The most recommended way is using an external css file. The link element has a href attribute that helps you to add your file path. You can also add the url of the file if it is a web page.

The rel attribute should also be given a value: “stylesheet”. And since the link doesn’t have a closing tag, it should be self-closed.

We can do so many things using the CSS, our next chapters will deal with that.

**Colors**

With cascading style sheets (CSS) it is possible to edit and change color of your html content. Text colors, background colors etc. can be formatted.

The syntax of CSS is different from that of html. In css, you write the name of the html tag you want to style. Then after that you add the styling in curl braces. As such:

P {

Color: blue

}

The styling follows the format of attribute: name and value, the name of the style is the color and the value is blue.

In CSS the term color is mostly used to color text, when you want to format background color, you use the background-color name and then change the value. You can also change the color of a border.

The color names follow the same color value names we know blue, red and others. But there are also different type of color values: rgb, hex, hsl, rgba and hsla.

The rgb stands for red green blue and it specifies the mixture of the three colors. It is written as this rgb(250, 34, 45). The highest value is 255, and each color mixture gives different colors. But you don’t have to start cramming the mixtures, you can simply search for color mixtures online and copy and paste the values.

The hex stands for hexadecimal colors. It is used in this form #ee4590, where each stands for #rrggbb, are hexadecimal values that are displayed from 00 to ff. But you can always browse hexadecimal mixtures online.

Hsl values stand for hue, saturation and lightness. The hue is a degree of color from 0 to 360. 120 is red, 240 is green and 360 is blue, saturation is from 0 – 100 the thickness of the color, means 100 is the full color saturation and lightness is the brightness from 0 – 100.

In hsla and rgba, the a stand for opacity from 0 – 1. It is the degree of transparency in the color.

You can always get the color values online, but those are the types of colors value html supports.

**Font size and Family**

In html, you can specify the font family, style and size using css.

Font family can also be known as font styling. It changes the styles of the fonts. There are many acceptable font style in html. And there are two types of font family – generic family and font family. Both can be considered as font families.

You can change the font family in css using the font-family property and then the name of the font family you want to set.

Also font styling is to specify the style of the font. And we have 3 properties in font styling - font-style, font-weight and font-variant.

Font-style has 3 values: italic, oblique and normal.

Font-weight has 2: normal and bold.

And font-variant has 2: normal and small-caps; in small caps all the text are converted to uppercase letters but they would appear small in the browsers.

Finally we have the font-size.

The font-size property takes numbers as values.

It is measured in px-which stands for pixels, percentage (%) and vw or vh which stands for view width and view height respectively. The view width and view height is similar to percentage. Each element has a default font size so changing it would mean changing the font-size for that element only.

**Text styles**

Text styling can also be done using css. In css we have many text styling.

Text-decoration is used to underline or add line to text. If you don’t want line on your text you can set the text-decoration property to none. Otherwise there are 3 values: underline, overline and line-through.

Text-align is used to align the text properly. It can be aligned to center, left or right. You can also justify your text.

Text-transform property is used to transform your text to uppercase, lowercase, or capitalize.

You can also change the text color using the property color and the color name as such: color: red.

There are many other styling in text, some of which we will learn as we go further in this tutorial.

**Margin**

The margin of any html element can be set using the css. If you notice all element displayed in block style have similar margin and are displayed in a straight line. But css allows us to overwrite that and put our elements in any side of width of our webpage.

We can do that using the CSS property margin. The margin also has 4 different properties: the margin top, the margin bottom, the margin left and the margin write.

When you set the margin property with a number value, it tends to shift just margin of the left side of the element. Also the margin property can take 4 values as such: Margin: 5px 4px 5px 3px. The first value goes to the top margin. The second to the left. The third to the down and the forth to the right. Also if you give it two values, the first would be for the top and bottom, and the second for the left and right. But just one value, just as I stated earlier, will adjust the margin of the left.

The other margin properties adjust the margin distinctively according to their names, the margin-top for the top; the margin-bottom for the bottom; the margin-left for the left and the margin-right for the right side.

We can also set the margin to auto. This means that the element should be situated at the width-center of your webpage. In some cases, we might need to write two auto to specify that the element be situated at the vertical and the horizontal center, the very center of the webpage.

**Anchor**

The anchor tag or element is used to insert links in your webpage. You can link to another webpage using the file path, or even the url. You can also link to a part of your webpage using Ids, which we will discuss extensively as we go further.

To add the anchor tag to your webpage we used the <a> tag. Then we add an attribute to the opening tag “href” in the href we specify the file path, or the url of the file we want to go to. We can also make images, buttons and other elements clickable by nesting the <a> tag inside the button element or in the paragraph or heading or whatever element we want to use. For image element, we nest the image inside the anchor tag.

To link a part of the same webpage, we can use an Id or class. For Ids, we use the # sign before the Id name and for class we use the “.” Before it. We would go to that extensively later. And if we want the anchor tag to be unclick able we can just set the href attribute to #. Also if we want it to return to the root folder we set it to “/”.

We can also target our anchor tag to display in a new tab on the browser. That can be done by setting the target attribute to \_blank. When we do that our link opens in a new webpage.

CSS also helps us set the font color of our link into anything we want. We can also set the text-decoration to none to remove the underline. We can also do more with CSS like setting it so that when the mouse hovers on it the style changes. We can do that by writing:

a: hover {

color: black;

text-decoration: underline;

}

Also we can set it so that the active link has a different styling using the a:active. We also have the a:visited and the a:link. The visited makes the visited page to have a different styling. All this can be done with css using color and text-decoration.

**Display**

Some html element as you might have noticed so far, display vertically, under one another. But others like the anchor tag display horizontally, beside one another. The vertical display is called the block display, while the horizontal display is called the inline display.

This is known as html layout and it can be formatted using CSS.

We can set a display manually to block or inline. We can do that using the CSS property: display. We can either set the display to block, or inline. This is useful especially when we are creating navigation links using anchor tags and maybe we nest the anchor tag in a list. Of course, we know list has a block display. We can change it in CSS into inline display to make the navigation like the normal horizontal navigations we see in websites.

There is also a third type of display, the flex display…

**Buttons**

The button tag is used to add buttons to our website. We’ve seen buttons on webpage especially used to navigate through webpage or to perform other options.

To add the buttons, we use the <button> tag and inside it we write the text content that will be displayed in the button after which you close the tag. Buttons are also inline element, they display horizontally, and we can change that if we want to, using the css. Also our button text content can also contain anchor tags. To navigate through webpages.

We can also style our button using css. We can change the text color using the color property, we can also change the background color using the background-color property in CSS.

Some buttons, however, need to perform distinct functions, which is where JavaScript comes into play. We can program our button to do anything we want using javascript. For that purpose, the button has an attribute onclick. That attribute helps to run whatever function we defined for the button to run when it is clicked. We will see more of that in our subsequent chapters.

Also we can format the border color and add padding around to our button with CSS, but that we will be seen in the next chapters.

**Padding**

As we were saying previously, we can add padding to our html elements. The CSS has made provision for that. The padding property in CSS, takes numeric values to show how big we want our padding to be.

Also just like the margin property, the padding property can also take four values. As such: padding: 4px 4px 0px 8px; the first value specifies the padding for the top area of the element. The second value for the left. The third for the bottom and the fourth for the right. Also if it takes two values the first is for the top and the bottom while the second goes to the left and right. But if it takes one value, the CSS adds padding all around the element.

Also each side has a distinct property namely: padding-top, which adds padding to the top side; padding-bottom, which add padding to the bottom; padding-left and padding-right which adds padding to the left and right respectively.

Take note that padding is not to be confused with margin. Margin specifies where the elements should start from or stop, that is, its limit. While padding swells up the element with the amount of pixels you specify. For text it might not necessarily increase the font size, but the area around it would be increased.

**Images**

Images can also be added into our webpage using the <img> tag. Mind you, the img tag doesn’t have a closing tag, so we self-close it.

The img tag has the attribute src which helps us to input the src of the image file, whether it is a local path, or a url.

It also has another important attribute, the alt attribute. The value given to that attribute is text type. The text is displayed if there is any little problem and the image cannot be displayed, maybe a wrong file path or maybe the image has been removed, the text would be displayed instead. Therefore the text has to be something that describes the image.

We can also resize the image using our css or the style attribute anyone we chose. We can use the width and the height property in pixels to set the image size.

Also images can be used as our background color. That can be done with CSS. We use the background-image property in CSS for that; and we add the image file path or url. Also make sure you set the repeat to none, in CSS under the property background-repeat, because sometimes if the image is smaller than our background, it tends to repeat to fill the space.

**Size**

As we’ve earlier seen we can resize our html element using CSS,

We can format our text size in CSS using the font-size property, and then we can set the value in pixels (px), in percentages % or view width (vw) and view height (vh).

Percentages is used with the screen size similar to the view width and the view height. While pixels are some small dots that makes up the screen. So our size can be measured in those.

Also we can size other elements using the width property and the height property in CSS. They both take numbers as values and are measured in pixels, percentages and view width and height.

**Divs**

Divs, can be described as containers that we put other elements. They don’t naturally display anything, but can be seen as containers for other code.

To use them we open our <div> tag. After that, you put all the html tags inside and close it. Initially divs accept the size of the elements in it. But u can reset the size using css properties width and height.

You can also create an empty div, set the size and the background color and display it on your webpage. Divs can be used for functionality and also to beautify your work.

Divs can also help in grouping some elements together and formatting them together, maybe aligning them differently or adding a different background color from the one the body has.

It helps you to create different sections on your webpage very efficiently.

The usefulness of divs cannot be overemphasized, but be careful not to create too much divs in your webpage.

**Borders**

Borders can be formatted in your webpage using the CSS. CSS has a property border that helps you to add border remove border or format border.

The property takes 3 values. The first value defines the size of the border in pixels or percentages. Most divs or buttons or tables need varying size of borders. So you need to specify the size or how thick you want yours to be.

The second value is the style of border; we have many styles of border in the html; the solid, dotted, dashed, double, groove: for 3d, ridge, inset and outset border. You have to specify which you want to use. The most commonly used border is the solid for a straight solid border.

The third value is for the border color. You can specify the color in plain text, hex, hsl values or rgb values.

CSS also allows us to use this properties individually in case we just want to add a border-color, border-width or border-style only, each has its respective property.

Also, you can format each sides individually, using the border-right, border-left, border-top and border-bottom. You can also format the style, width and color for each of them individually too.

The border-radius property helps us to create rounded borders. It can take one value, a number, to specify how round the border should be. It can also take 4 values, each of them being numbers specified in pixels or percentages. The amount of pixels you indicate will specify how round the corner would be. The first value goes to the top-left corner, the second to the top-right corner, the third to the bottom-left and the fourth to the bottom-right.

**Ids and Classes**

Ids and classes are very useful attributes in web development. They help to give elements a name or identification. They are also used for navigation, take for example, you want to link to a part of your website. You simply give that element an id name or class name and specify it in the anchor tag.

So the both attributes are very similar in different ways, but they also have differences, while Id is mainly used for navigations and identifications, class are used to format elements. Means that id is mainly used for one element while class names can be used for many elements. Though they can be used interchangeably as html would allow it, but that is their main functions.

To add an id or class simply add the id or class attribute to the opening tag of the html elements and specify the name you want to give it in quotes.

To target them using css, anchor tags etc, you use #idname - # sign before the Idname; and a .classname – . before the classname.

Remember and Id name and class name must not have space in between if not the browser would take it as two different ids or classes.

You can format the element(s) with the id or classes in CSS using the specified way mentioned above.

Also, you can program the element with the specified Id or class in JavaScript using the document.getElementById and document.getElementByClassname method. We would see more to that in future chapters.

**Floats**

So far, you must have noticed that when adding html elements they always appear on the left side of our webpage. Though you can use margin to move them, it is stressful to add other elements beside it, especially big elements like divs, even with the inline property.

That is where the CSS property floats comes in. The floats property helps us to float elements around the webpage. We can set it to right, moving the elements to the right; we can also set it to left, moving it to the left; and to the top and the bottom.

Floating somehow destabilize our websites so we have to be very careful how we use it. We can also set the margin property to make sure it is floated properly.

Also there is a CSS property known as overflow, it helps us to discontinue the floating. When we are done floating the elements we want to float, we may discover that, the next elements we add are still floated. We can then set the overflow property of the next element after the float to hidden. In that way it returns to normal again.

**Positioning**

The position property in CSS helps specify the type of positioning to be used for our html element.

We have 5 different types of positioning, we have: relative, absolute, fixed, static and sticky.

After the type of positioning are specified, we can then use the top, right, bottom and left properties. This properties move the element respectively.

The first type of positioning is the static and it is the default type. This means that it doesn’t use the top, left, right and bottom properties. Instead it is positioned according to the normal flow of positioning in the page.

The second type which is the relative positioning, and it is positioned relative to its normal position. The top, right, bottom and left properties can then be used to move the element not to, they move it from. Meaning that when you set the top properties, the amount of pixels you use is the amount of pixels it will move from the top not to the top.

The third type is the fixed. It fixes the element to the page, and a fixed element cannot be moved even when the page is scrolled. The top, right, bottom and left property can be used to position it accordingly.

The fourth type is the sticky positioning. It is in-between the fixed and relative in the sense that at some point is it relative when the element is hanging loose and can move when scrolled. But when it reaches the edge of the viewport, it doesn’t move it stays there, like it sticks to the edge and becomes fixed.

Finally, we have the absolute positioning. It positioned relative to its nearest parents instead of to the viewport. However, if it has no positioned parents it is then positioned relative to the viewport and it moves along with the page scrolling.

**Tables**

Tables can be added into our webpage, using the table tag. The table tag also has other elements nested into it.

The first is the <tr> tag. It specifies the table rows. Each tr tag specifies a row. It also has tags nested into it.

The <th> tag is one of them. It specifies the table heading and it is used inside the first tr tag. Each th element contains the heading table cell and its content.

The next is the <td> which is the table data. It is used for the rest of the tr and each td tag contains table data cell and its content.

Tables can be styled using CSS. By default, if you’ve noticed so far, the table comes with no border lining at all. It is your duty then, to add it using the CSS border property. You can specify the border size, style and color. You can also make it rounded and so on.

Also, CSS helps you to add color to you table content and background color to the table itself. It depends on you to choose the outlook of your table and then style it with the CSS.

Take note that table contents can be anything ranging from text to buttons and even divs.

**Forms**

With Html we can create forms on our webpages. Forms gives us avenue to interact with our user, collect information and use that information to perform an action.

The html has a form element which allows us to create forms. The form element has two important attributes, the action attribute and the method attribute.

The action attribute has to do with what you want to use the information given to do. It usually a link to a file where your info will go to possibly another webpage. Most use php which is a server website for that.

The second attribute is the method attribute and it has to do with how you want your information to go. It has two main values: the get and the post values. The get values is used when you have a specific action you want to use the information for, maybe to change the website, create data or something. But the post is just used for no action information.

Html provides different type of form elements. Input, text area, button, and select.

The input element has different types. We have the text, submit, button, checkbox and radiobutton.

Every form element needs a label. So to create forms, we first open the form tag and list out the above mentioned attributes in it. After that we create a label for whatever input type we want. The label has an attribute for. The value of that attribute is the name of the input element. Each input element has an attribute name, so, the name you give it is the value of the for element of its label.

Also the input we can specify what type of input we want.

For text we simply write text and a text field comes out.

For checkbox, the input attribute, type should be set to checkbox, then you write each line you want to check with it label differently, either before or after the checkbox, some chose to write it without label, just simply writing the label after the input tag, but that is not good coding practice

For radiobuttons you simply create one label for them, and after each input tag with the type radio, you write the words after the button. Remember users can chose only one button.

The button type is just like normal buttons but with specific functions.

And the submit type is the button that submits your information.

There ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

**Semantics**

Semantic elements are those element that defines its content to the browser and the developer. Its name is self-explanatory to the developers. Non-Semantic elements don’t describe it contents.

Examples of non-semantic elements are divs.

While example of semantic elements are tables, forms, articles, navs, section, aside, header and footer.

The header element is defined using the opening and closing of the <header> tag. They house the headers of a page or a section.

The section element is defined using the <section> tags. It defines a section or part in the page. That section houses elements.

The article element, which is defined using the <article> tag defines an independent part of the page. It contents can mostly stand on its own.

The <nav> tag defines the navigation element and is used for navigations around your webpage. Most navigations are found at the top of the page after the header and they consist of links both to inside your webpage and outside of it.

The aside element with the <aside> tag defines content that are not part of the main webpage. So as the name implies they are pushed aside. But they are related to your webpage and can be likened to a side bar.

The footer element is defined using the opening and closing of the <footer> tag. They house the footers of a page or a section of a page.

**Responsive**

Responsive design is about creating web pages that look good on all type of devices. It automatically changes form according to the screen size of the any device.

To design responsive websites, the first thing to do is to set your meta tag in the head element to name=viewport, content= “width=device-width, initial-scale=1.0”. The above code sets the items on your webpages, according to the dimensions.

Also when setting the size of images or divs instead of pixels, it is better to use percentages. And when setting font size use view width (vw).

You can also use CSS to make your website responsive. After styling your elements for large screens, you can use the @media screen and (max-width: 800px){myelements {property: value}}.

The above code sets a different styling for screen width less than 800px.

Remember it takes time to build responsive website so take it slow. Also, we recommend using Bootsrap – a CSS library base that helps build responsive website.

**Charset**

The charset attribute is used to specify the character encoding for the HTML webpage. It is mostly used in the meta element with the character encoding utf-8.

The character encoding is the way your html webpage displays your text or characters. And to display your webpage correctly, the web browser must know what character set you used in your page.

There are other standard character sets like the ASCII, the ISO-8859-1 and the ANSI.

All of them differ from each other, and the default character set for html five is the utf-8.

The character encoding has to do with what comes out when you press a particular key on your keyboard, that is what the browser recognizes the key as.

Each character are all assigned to numbers from 0 to 255. The utf-8 continues the value from 256 with more than 10,000 different characters. This includes spacing modifiers, greek and Coptic characters, general punctuation, currency symbols letter-like symbols, box drawings, geometric shapes and block elements, most of which other character encoding lacks.

You can look up character encoding and read more about it on google.

**JavaScript**

Javascript is the programming language we use to program our webpages. It helps make our webpages more dynamic and interactive. Javascript is a separate language on its own, with its own syntax so you have to link it with your html page.

To do that we have the script tag. Its content is javascript syntax and you can write your javascript code in it. The script tag also has an attribute known as the src attribute. It helps you to link your external javascript file into the webpage. You can link more than one external javascript in your webpage, it doesn’t matter.

Also javascript helps you to target certain html elements for programming. You can do that using the document.getElementById(“demo”). You can then change the text content by adding: document.getElementById(“demo”).innerHtml = “content”.

You an change styles of your html webpage using the document.getElementById("demo").style.color = "blue"; you can also target html using the query selector method.

You can change attributes in html using javascript, style your webpage and even create web applications and functions for your webpage to run.

Javascript makes your webpage dynamic and interactive we recommend you learn more about javascript in our javascript tutorial.

**Iframe**

The iframe element is used to nest another file in your html webpage. It is written with the iframe tag <iframe> and it doesn’t have a closing tag. Its required attributes are the src and the title. The src is used to specify the file path or the url for the nested file while the title is like a description that won’t be seen. It is used for screen readers to know the description of your file.

You can also set the width and the height of your inline document using the width and height attributes, or you can use the style attributes and the width and height properties and values.

By default, iframes comes with a border around them. You can remove the border using css border property, setting it to none. Or you can change the border width, style or color using the css border property.

Also, you can also make a link to display as an iframe. You can do this by setting the target attribute of the anchor tag to the name of the iframe. Of course you have to give your iframe a name before then and then afterward set the target to the name.

**Canvas**

The html canvas element, using the canvas tag, is used to draw pictures in an area. The canvas element allows us to use javascript to draw on your webpage. So you add the canvas tag in your webpage.

By default, the canvas element is blank. But you can set the width and the height attribute with values in pixels. Then you can also add border using the css border property.

Then you can define the size of the box, the content is left to javascript.

You can do many things with canvas ranging from using colors to paint, to drawing shapes and writing text and styling it. All of this can be done using javascript.

You can check our javascript tutorial for more,…..

**SVG**

The acronym svg stands for scalable vector graphics. It is a html element that is used as a container for svg graphics.

It is opened with the svg tag and closed. The important properties in the svg are the width and the height.

Svg can be used to draw circles, rectangles and other numbers of sided polygons.

To do this after opening the svg, you define the elements you want to draw.

For circles you create a circle tag <circle> which is self-closing. Styling can be used to define the color of the circle. In styling you use the fill property to add color. You can also create stroke like borders, using the stroke-width property which takes a numbered value, and the stroke property which takes a color value.

For rectangle you use the rect tag. You can define the height and the width using the respective attributes. You can also use styling for coloring, using the fill property of the style attribute and the stroke-width and stroke to define the size of the stroke and it color. You can also make rounded rectangles.

This can be done by setting, the rx and ry attributes. They both take numbered values. Lets test that out shall we.

To draw polygons, we use the polygon tag, defined in the svg tag. It takes an attribute points which allow you to define numbered comma separated values, depending on the polygon you want to draw.

SVG is a very useful tool like the canvas for graphics.

**Pictures**

Pictures can be added to your html page using the picture tag. It allows us to display pictures of different size for different screens.

It is mostly useful when you want to display image in a different format. Some browsers might not support some image format, but by using picture tag, you can specify different formats and the browser would chose the one it supports.

Also, if you have a small screen size and you want to load a large image file, you can specify different pictures for the different screen size.

You add picture by opening the picture tag and closing it when done. It has a nested element called the source element, defined by a self-closing source tag. The element takes an attribute of media which has a property of min-width and a value of a screen-size in which a picture should be displayed. It still has another attribute srcset which shows the source of the image or the file path for the particular screen size. A picture can have more than one source element for different screen size.

At the end always specify an <img> element before closing the picture tag. The img element is used by the browser that do not support the picture element or if none of the source element matches your screen size.

**Audio**

The audio element is used to play an audio file on your webpage.

To add an audio file to your webpage, you simply open the audio tag and close it when done. The audio tag has an attribute controls. The controls attribute doesn’t take any value, instead it is used to add control like play, pause, skip, volume, etc.

It also has a nested element in it known as the source element. The source element has an attribute src like the img attribute that specifies the audio file path. It also has an attribute type that specifies the audio type. It takes value audio/mpeg for mp3.

Also you can add more than one audio file with different file types should incase some browsers doesn’t support the audio file type specified.

If you write text in between the audio element, remember that your text wouldn’t be displayed.

**Video**

The video element is used to show video on your webpage.

You can add the video by opening the <video> tag and closing it when done. The video element has some important attribute in it.

The width and the height attribute of the video element helps to specify the size of the video window. It takes as value numbers that specify the height and the width of the video.

Also the controls attribute is used to add control options in the video display like the play, pause, volume, skip, etc.

The video element has a nested element known as the source element. It is self-closing and takes as attribute the src attribute to specify the file name and path, and the type attribute to specify the video type whether mp4, ogg etc.

You can add more than one source tag in the video element, in case a browser doesn’t support a video type, you can add another type of the same file. The browser would play the one it supports.

The video tag also has another important attribute called autoplay. It automatically plays the video without the user asking it to. But the autoplay doesn’t work on mobile devices.