HTML

Now let us first understand what HTML, CSS, and Javascript are for? For every website to be designed, HTML (**HyperText Markup Language**) is a must. This is the skeleton of a website. Without it, no website can run. CSS (**Cascading Style Sheets**) adds beauty to that website and JavaScript adds the brain to allow the functioning of that website. Therefore, CSS and JavaScript add beauty and brain to a particular website respectively.

Let us now take another example of a car. The HTML acts as the metallic body of a car and the CSS acts as the color and design of the car. Finally, the engine of a car is like the JavaScript on the website to add functionality.

Normally, a client or a user sends a request to the webserver of the website, he wants to visit. The web server that has its own IP address stores all the files in the backend which can be written in PHP, Python, or Node.js. The web server sends a response to the client in the form of HTML, CSS, and JavaScript.

Finally, to understand-

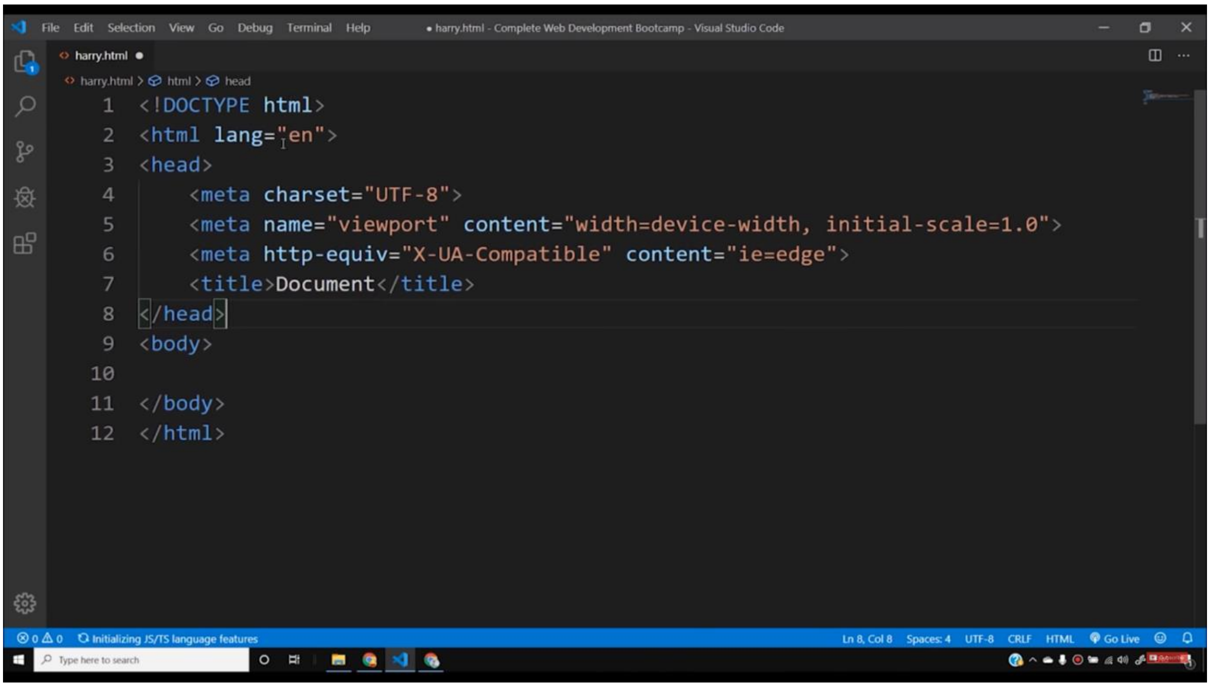
- **HTML** is used as a standard language for any website design. It acts as a static skeleton to a web application. It’s a well-standardized system.
- **CSS** is used to handle the presentation of the web page. It makes the website look attractive and beautiful.
- **JavaScript** allows scripting on your website and makes it completely dynamic in nature. It provides front end scripting for your website and is a high-level dynamic interpreted programming language.

The HTML code contains some sections such as ***DOCTYPE, meta tags, head***, and ***body.*** All the tags that we include in an HTML file, needs to be first opened and then closed by angular brackets. Let us now understand different sections one by one.

- **Doctype HTML**- Doctype HTML justifies that it is a HTML document. So, we are defining specifically for a browser to understand that it is an HTML document. To understand more about the type of documents, you can visit here.

- **<html lang= "en"> -** It is the opening part of the HTML tag that tells us the language of the document is in English.
- **<head> -** Head contains all the meta-tags in it which is used to describe the contents of a website. Meta means providing information about information. Therefore, meta tags are used to define the keywords and descriptions on our website. Head also contains the **title** of the website and all the external files like CSS and JavaScript that we link to it.
- **<body> -** Body contains all the contents of the webpage in it. However, in the beginning, our website may look a little uglier but after including stylesheets it will start looking attractive.

Sometimes we want to write a particular thing that we do not want the browser to take it as a part of code. So, we can include it in the comments by writing ***"<!-- Your text -->"*** in this format.



File Edit Selection View Go Debug Terminal Help
• harry.html - Complete Web Development Bootcamp - Visual Studio Code
harry.html ●
harry.html > html > head
Initializing JS/TS language features
Ln 8, Col 8 Spaces: 4 UTF-8 CRLF HTML Go Live
Type here to search

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

```html
<!DOCTYPE html>

<html lang="en">

<head>

   <meta charset="UTF-8">

   <meta name="viewport" content="width=device-width, initial-scale=1.0">

   <meta http-equiv="X-UA-Compatible" content="ie=edge">

   <meta name="description" content="This is description">

   <meta name="keywords" content="html, html tutorials, web development">

   <meta name="robots" content="INDEX, FOLLOW">

   <title>Document</title>


   <!-- This is how you include external css -->

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

```
    <!-- This is how you include external JavaScript -->
    <script src="harry.js"></script>
</head>
<body>



</body>
</html>
```

The Meta Tags are used to define the Meta data in an HTML. They are mainly used in SEO (Search Engine Optimization) techniques which help any particular website to rank better in Google or different search engines. It simply boosts the ranking of a webpage to get more traffic on any website.

So let us understand various meta tags.

- `<meta charset= “UTF-8”>`

Copy

- It simply means that the characters that are used will be of UTF-8. It declares the page’s character encoding. It should contain a standard IANA MIME name for character encodings. Moreover, authors are encouraged to use *UTF-8.*

```
<meta name= “viewport” content= “width=device-width, initial-scale=1.0">
```

Copy

- This tag is used to make your website responsive and adjust its width in such a way that it looks good in both PC or mobile. It helps in making the website mobile friendly also.

```
<meta http-equiv= “X-UA-Compatible” content= “ie=edge”>
```

Copy

- It helps any particular website to open in the highest compatibility mode available. It is mostly for those who are still using Internet Explorer. Because there are still some people who have not upgraded their system and are still using the older versions.

Let us begin with the simple **heading** tags. There are basically six types of heading tags ranging from ***<h1>, <h2>, <h3>, <h4>, <h5>, and <h6>.*** These are the six heading tags from h1 being the largest font size and h6 being the smallest font size. There is an important you should know about **H1** tags. In every website there is only one **<h1>** tag, which is the main heading of the website. You should never write the normal paragraph text as headings just to make it bold. It is advised that using <h1> tag **only once** will help in SEO (Search Engine Optimization) of a website and even ranking it higher in different search engines.

Then comes the paragraph tag which is denoted by **<p>.** Whenever we want to add a paragraph on our website then we can simply use paragraph tag in the format-

```
<p>some random texts</p>
```

Copy

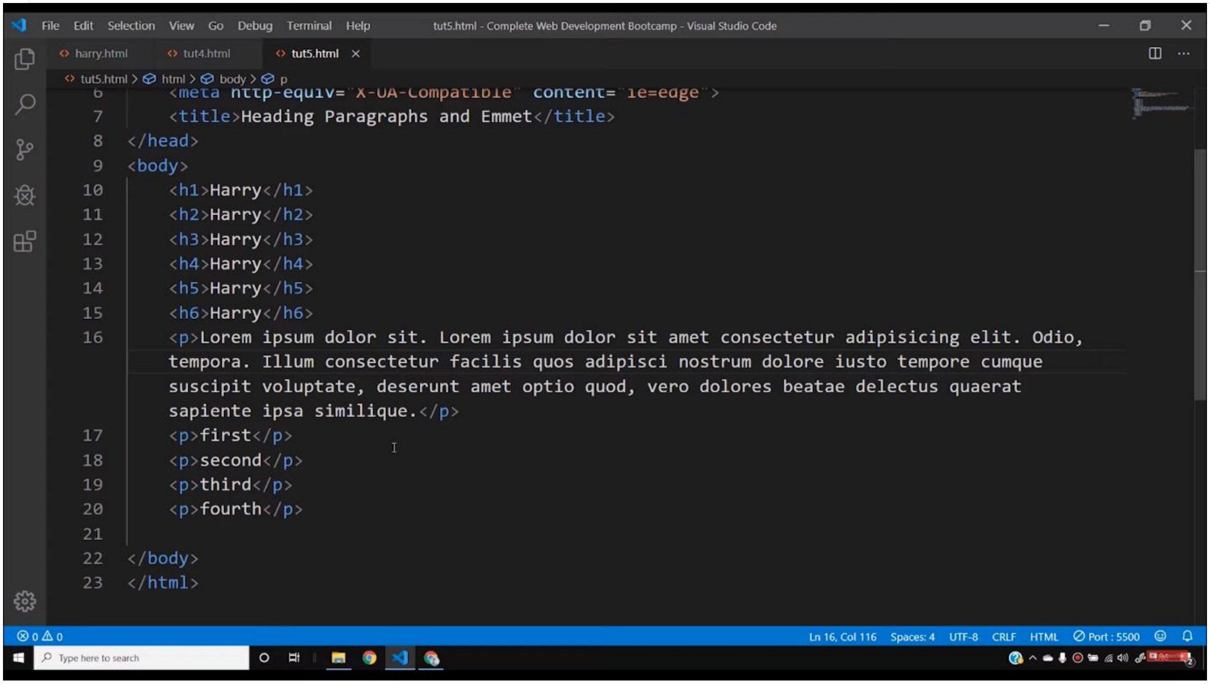
To write a new paragraph, simply jump on a new line and start writing the new paragraph in a paragraph tag. To get 4 different lines of paragrapgh tag, we can simply write ***p*4.***

Let us now discuss the main advantage of emmet abbreviation. If we want some random texts up to any number of words then we can either copy from any article from the Internet or just write random words, which of course will not look good. To avoid this, we can write ***lorem*** and the number of words we want in a paragraph. Let us understand with an example-

```
<p>Lorem40</p>
```

Copy

This code will give us a paragraph of 40 words.



File Edit Selection View Go Debug Terminal Help
tut5.html - Complete Web Development Bootcamp - Visual Studio Code
harry.html
tut4.html
tut5.html
tut5.html > html > body > p
Ln 16, Col 116 Spaces: 4 UTF-8 CRLF HTML Port : 5500
Type here to search

```
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Heading Paragraphs and Emmet</title>
</head>
<body>
    <h1>Harry</h1>
    <h2>Harry</h2>
    <h3>Harry</h3>
    <h4>Harry</h4>
    <h5>Harry</h5>
    <h6>Harry</h6>
    <p>Lorem ipsum dolor sit. Lorem ipsum dolor sit amet consectetur adipisicing elit. Odio,
    tempora. Illum consectetur facilis quos adipisci nostrum dolore iusto tempore cumque
    suscipit voluptate, deserunt amet optio quod, vero dolores beatae delectus quaerat
    sapiente ipsa similique.</p>
    <p>first</p>
    <p>second</p>
    <p>third</p>
    <p>fourth</p>

</body>
</html>
```

In the body tag, if you type “a", anchor tag will appear. Just hit the enter key or if you want you can manually write the whole tag. Refer to the declaration below:

```
<a href=""> </a>
```

Copy

Here href is the attribute of anchor tag where you have to write the URL of the website or Link that you want to open. Next, you have to write the Keyword on which you want the user to click so that he will be redirected to the linked website. Refer to the example below:

```
<a href="https://google.com">Go to Google</a>
```

Copy

Now, the problem arises here is that as soon as we click on the keyword, the website will open on the same webpage but if in case you want to open the website in a new tab then you have to add a new attribute in the anchor tag I.e. “target".

You might be confused between tag and attribute, then let me quickly tell you the difference between both.

Tag is like a container that allows you to handle an element whereas attribute is the property that enhances that container makes it more convenient to use.

To open a website in a new tab, you have to write something like this:

```
<a href="https://google.com" target = "_blank">Go to Google</a>
```

Copy

Through this anchor tag, you can also link internal webpages that are locally available in your directory. You just have to use

the address of the internal webpages including the file name with its extension.
For instance:

```
<a href="tut4.html" target = "_blank">Go to Google</a>
```

Copy

So, this was all in the anchor tag of HTML now let’s move to the next topic I.e. handling Images in HTML.
In order to insert an image in a webpage, you have to use the img tag.
An img tag generally have two basic attributes “src" and “alt".

```
<img src="" alt="">
```

Copy

Here “src" is the field where you have to insert the URL or address of the image and the “alt" attribute is a field that will display to users if their browser fails to load the image.
In the “alt" tag, we generally input a keyword of the image that can define the image in case the user can’t see the image.
For instance:

```
<img src="https://source.unsplash.com/random" alt= "Error loading
image">
```

Copy

Here a random image will be loaded because this URL stores multiple images and any one of them gets reloaded every time we refresh the webpage.
Apart from this, let's see another example where we can adjust the height and width of the image using the URL only. Well, this is not something that we can do in every URL. Here we can do it because the developers of the website have created this URL accordingly.
For instance:

```
<img src="https://source.unsplash.com/1600x900/?nature,water" alt=
"Error loading image">
```

Copy

If you want to manipulate the dimensions of the images, you can use the “height" and “width" attribute of img tag. Though it is not recommended to manipulate the dimension using these attributes, we will use CSS to alter the design accordingly. Refer to the illustration below:

```
<img src="https://source.unsplash.com/1600x900/?nature,water" alt=
"Error loading image" width = "233" height="34">
```

HTML Formatting Elements

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>` - Smaller 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.

Example

```
<b>This text is bold</b>
```

Tables and Lists in <title> tag.

The lists are basically of two types-

- ***Ordered lists (ol)***

```
<ol>

          <li>This is the first item of my unordered list</li>

</ol>
```

Copy

- ***Unordered lists (ul)***

```
<ul>

          <li>This is the first item if my unordered list</li>

</ul>
```

Copy

The difference between an ordered and an unordered list is that the ordered list displays the list in this format **-**

1.

2.

3.

....

On the other hand, the unordered lists display the list in the following format-

- .
- .

- Both the lists have more than one attribute which we can write using the *type* command. For example, if we write:
- `<ol type= “I”>`
- Copy
- Then we will get the lists as I, II, III, and so on. In the same format, we can also get the lists as A, B, C, and so on.
- This applies on unordered lists also. If we write
- `<ul type= “square”>`
- Copy
- Then we will get a bulleted square instead of a circle.

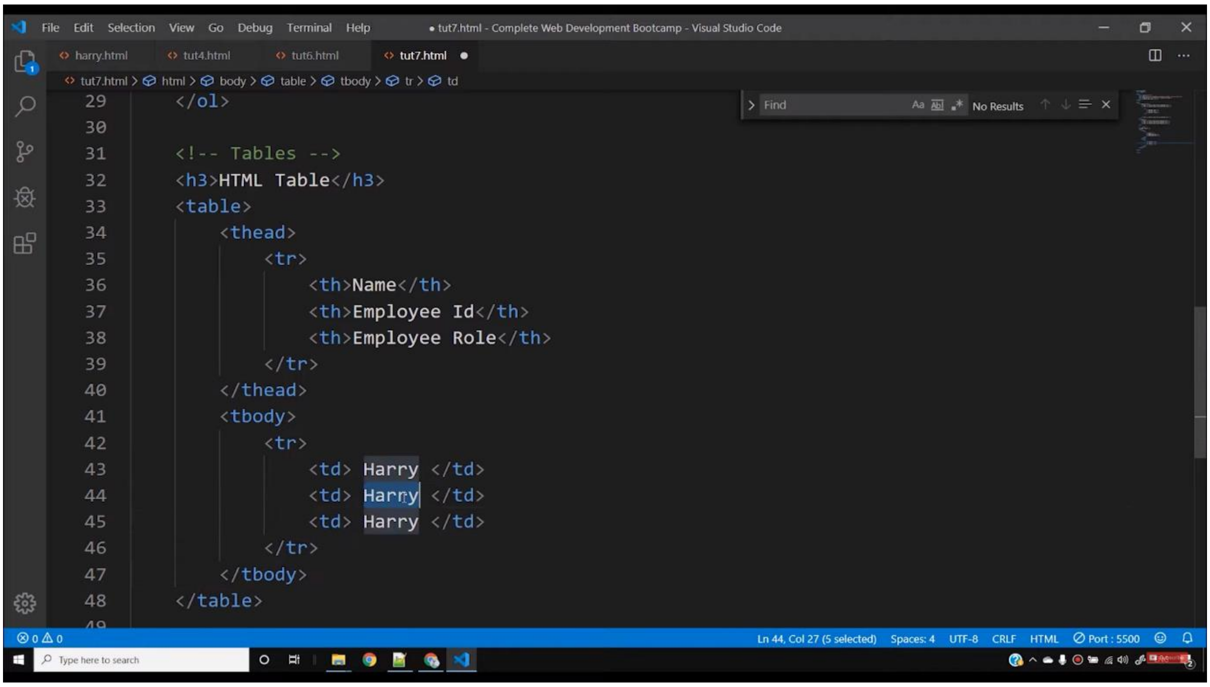
HTML also allows the nesting of lists. It simply means we can add a list into another list.

```
 <ul type="circle">
        <li>This is first item of my unordered list</li>
        <li>This is second item of my unordered list</li>

        <ul>
            <li>Another one</li>
            <li>Another two</li>
            <li>Another three</li>
 </ul>
```

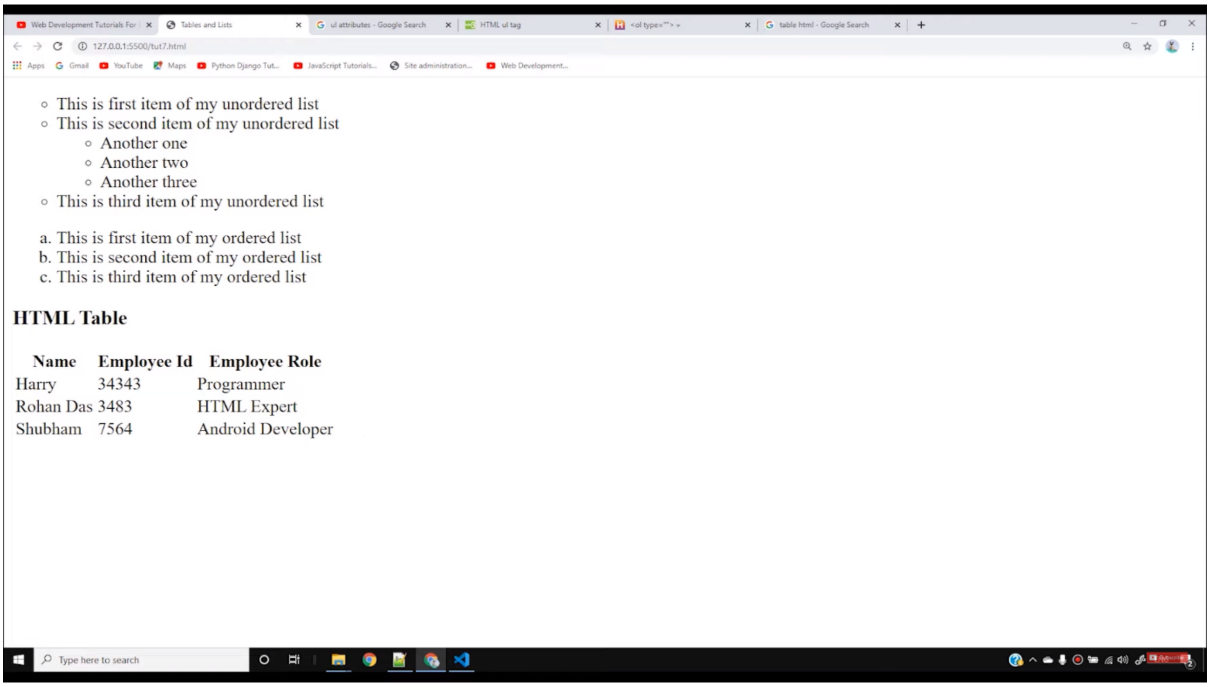
Let us now discuss the **Tables** in HTML. A table is just a combination of rows and columns on a webpage. The structure of the table looks like-

The main part is the table tag, and it consists of two parts: the table *head* and *table body.* The <thead> consists of the main head of the table and <tbody> consists of the body of the table. <tr> is used to justify that it is the part of a row. Inside the <tr> tag, we give the headings of a row under the <th> tag. The final structure of a table looks like-


File Edit Selection View Go Debug Terminal Help
• tut7.html - Complete Web Development Bootcamp - Visual Studio Code
harry.html
tut4.html
tut6.html
tut7.html
tut7.html > html > body > table > tbody > tr > td
Find
No Results
Ln 44, Col 27 (5 selected) Spaces: 4 UTF-8 CRLF HTML Port : 5500
Type here to search

```
    </ol>

    <!-- Tables -->
    <h3>HTML Table</h3>
    <table>
        <thead>
            <tr>
                <th>Name</th>
                <th>Employee Id</th>
                <th>Employee Role</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td> Harry </td>
                <td> Harry </td>
                <td> Harry </td>
            </tr>
        </tbody>
    </table>
```



- This is first item of my unordered list
- This is second item of my unordered list
  - Another one
  - Another two
  - Another three
- This is third item of my unordered list

a. This is first item of my ordered list
b. This is second item of my ordered list
c. This is third item of my ordered list

HTML Table

Name	Employee Id	Employee Role
Harry	34343	Programmer
Rohan Das	3483	HTML Expert
Shubham	7564	Android Developer

Whenever we add a <form> tag in the HTML, it is going to ask for some action for submitting that particular form in the backend for future reference. So, for now, we will write it as *backend.php.* All the data submitted in a form will be stored automatically in the backend "backend.php" after submitting it. The template will look like-

```
<form action= "backend.php">
```

Copy

Then comes the <input>tags which are present inside the form, where the user provides the input. These inputs can be of any type whether text, button, checkbox, date, time, etc. Input Tags are used to create interactive controls for web-based forms in order to accept data from the user.

The **<span>** is an in-line element and **<div>** is a block element. Which means, if we use two separate div tags for different inputs, then all the inputs will come on different lines. We will learn about the span and div in detail in the upcoming tutorial. Till then we will use the <br> tags for breaking a line.

- To get the input as *text*, the syntax is-

```
<input type= “text”>
```

Copy

- To get the input type as an *email* in the form, the syntax is-

```
<div>
            Email: <input type="email" name="myEmail">
</div>
```

Copy

The *name* here is used so that the backend can recognize the tag that we are using.

- To get the *submit* button in the form, the syntax is-

```
<div>
          <input type= “submit” value= “submit now”>
</div>
```

Copy

- We can also add date and time in the form. To add these, the general syntax is-

```
<div>
```

```
         <input type= “date” name= ”myDate” id= “”>
</div>
```

Copy

It will give the complete date form in the format of “dd/mm/yyyy”.\

- To add any numeric text in the HTML form, the syntax is-

```
<div>
             Number: <input type= “number” name “myNumber”>
</div>
```

Copy

While filling several online forms, you must have seen the radio buttons and checkboxes in the form. Radio buttons are such buttons that allows to select any one of the following options amongst all. For example, while selecting the gender, we can only select either male or female. Whereas the checkbox allows selecting the multiple options available. The example of both the formats are as follows-

- For checkbox-

```
 <div>
            Are you eligible?: <input type="checkbox"
name="myEligibility" checked>
 </div>
```

Copy

- For Radio buttons-

```
<div>
            Gender: Male <input type="radio" name="myGender"> Female
<input type="radio" name="myGender">
```

```
            Other <input type="radio" name="myGender">
 </div>
```

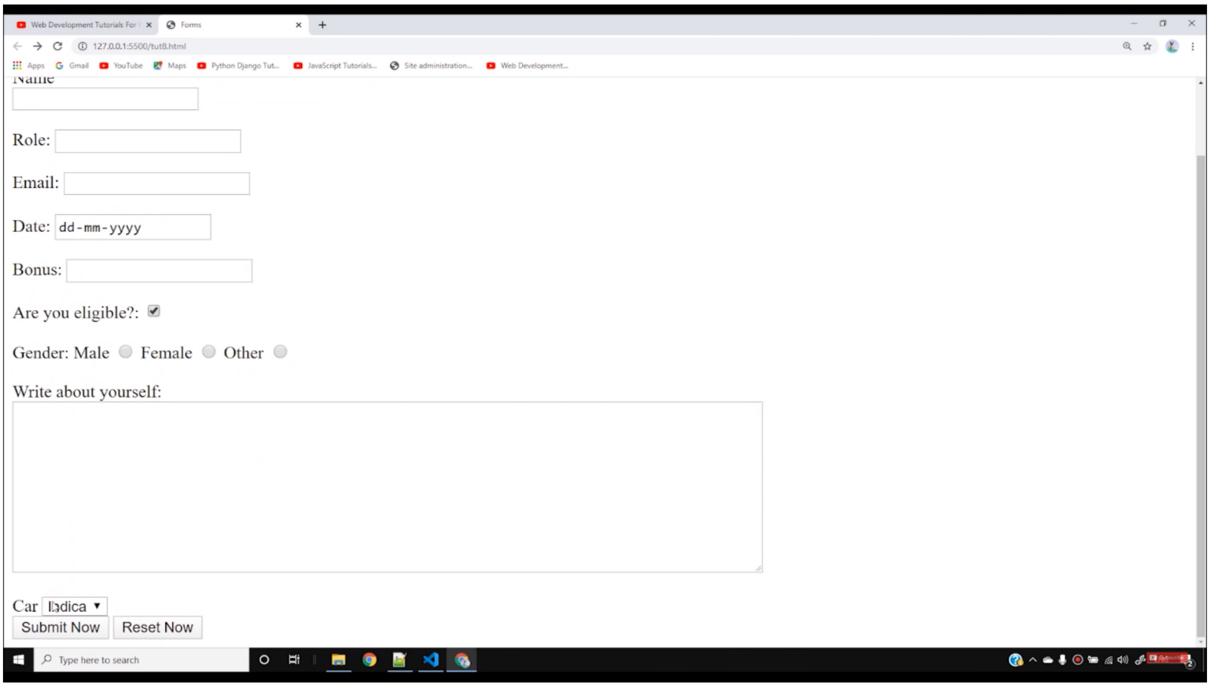
Copy

- To reset all the information, entered in the form, we take the help of a ***reset*** button. To get the reset button, we have to write-

```
<div>
            Input type= “reset” value= “Reset Now”
</div>
```

Copy

After inserting all these input tags, our form will look like-
These are some of the examples of basic input tags used inside the form in the HTML. Apart from these, there are numerous more input tags available, but you need not to learn all these at once. You can always take help of references available. And one more advice, never try to grasp all things at once. Practice makes a man perfect.



Name

Role:

Email:

Date: dd-mm-yyyy

Bonus:

Are you eligible?: ☑

Gender: Male ○ Female ○ Other ○

Write about yourself:

Car Indica ▾

Submit Now Reset Now

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

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Forms</title>
</head>

<body>
```

```
<h2>This is HTML forms tutorial</h2>
<form action="backend.php">
  <label for="name"> Name</label>
  <div>
    <input type="text" name="myName" id="name">
  </div>
  <br>
  <div>
    Role: <input type="text" name="myRole">
  </div>
  <br>
  <div>
    Email: <input type="email" name="myEmail">
  </div>

  <br>
  <div>
    Date: <input type="date" name="myDate">
  </div>
  <br>
  <div>
    Bonus: <input type="number" name="myBonus">
  </div>
```

```
    <br>

    <div>

      Are you eligible?: <input type="checkbox"
name="myEligibility" checked>

    </div>

    <br>

    <div>

      Gender: Male <input type="radio" name="myGender">
Female <input type="radio" name="myGender">

      Other <input type="radio" name="myGender">

    </div>


    <br>

    <div>

      Write about yourself: <br><textarea name="myText"
cols="90" rows="10"></textarea>

    </div>


    <br>

    <div>

      <label for="car">Car</label>

      <select name="myCar" id="car">

        <option value="ind">Indica</option>

        <option value="swf" selected>Swift</option>
      </select>
    </div>
    <br>

    <div>
      <input type="submit" value="Submit Now">
      <input type="reset" value="Reset Now">
    </div>
  </form>
</body>

</html>
```

Inline elements are those elements which only occupy the space bounded by the tags defining the element, instead of breaking the flow of element. On the other hand, block-level elements take up the entire space of its parent element. Let us understand this with an example-
If we write any text in the paragraph tag like this-

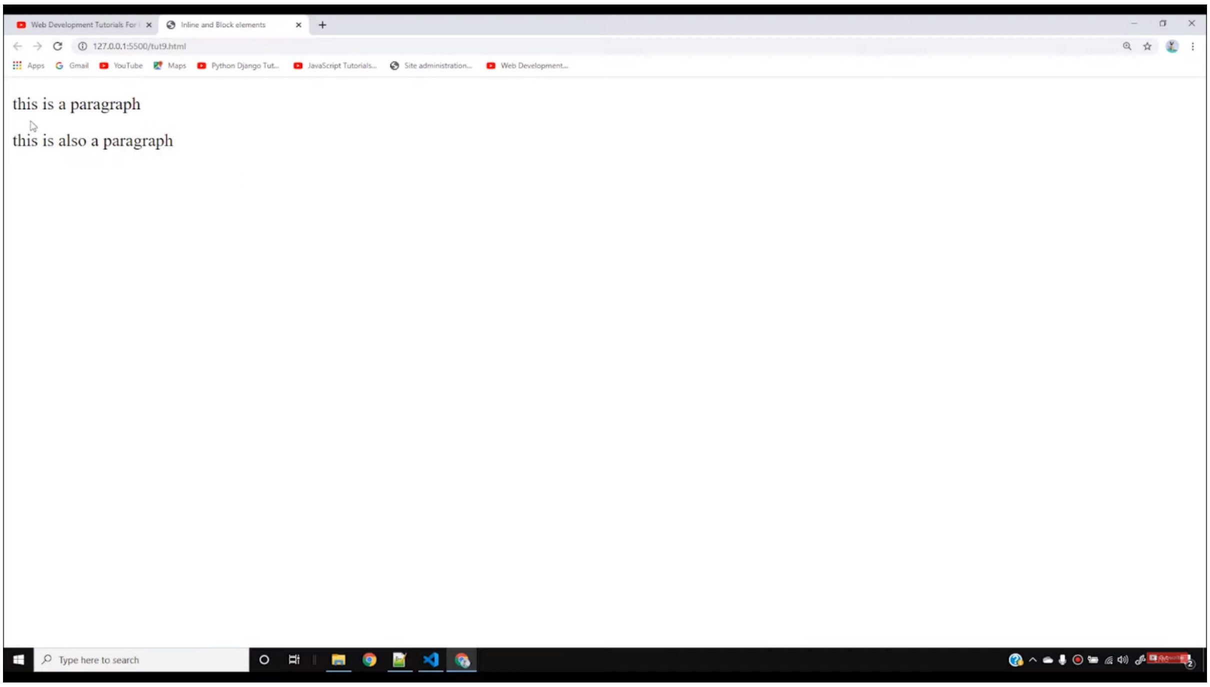
```
<p>This is a paragraph</p>

<p>This is also a paragraph</p>
<p>This is also a paragraph</p>
```

Copy

OR

```
<p>This is a paragraph</p> <p>This is also a paragraph</p>
```



this is a paragraph

this is also a paragraph

In both the above examples, we will see the output in both different lines, not in the same line. We want both the texts in the same line but it is not so. Can you think why?

It is because the paragraph tag is a **block element.** The Block element means that it will take the full width of a single line and does not allow any other content to fit in it. But, if we write both the texts between the <span> tags like-

```
<span>This is a paragraph</span> <span>This is also a
paragraph</span>
```

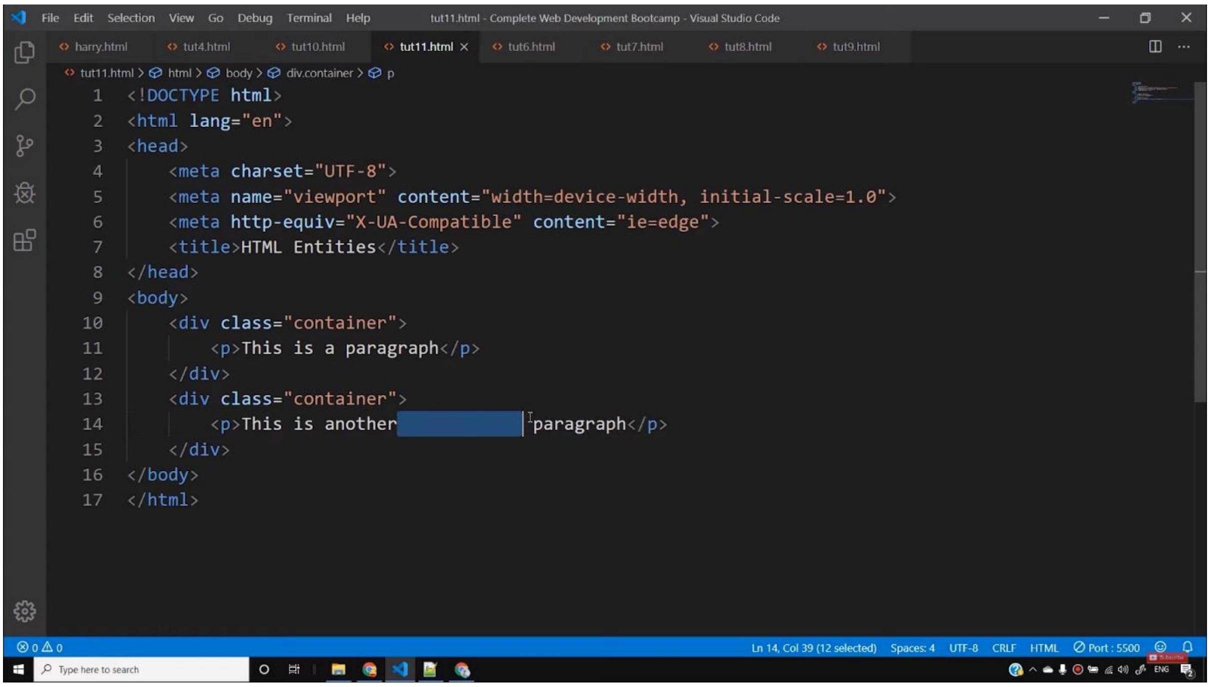
Try to understand with the below example. If we write the code as

```
<div class= “container”>
            <p>This is a Paragraph</p>
</div>
```

```
<div class= “container”>
            <p>This is another Paragraph</p>
</div>
```

Copy

If we add some spaces in between these texts then what will you expect?


File Edit Selection View Go Debug Terminal Help
tut11.html - Complete Web Development Bootcamp - Visual Studio Code
harry.html
tut4.html
tut10.html
tut11.html
tut6.html
tut7.html
tut8.html
tut9.html
tut11.html > html > body > div.container > p
Ln 14, Col 39 (12 selected) Spaces: 4 UTF-8 CRLF HTML Port : 5500
Type here to search

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>HTML Entities</title>
</head>
<body>
    <div class="container">
        <p>This is a paragraph</p>
    </div>
    <div class="container">
        <p>This is another             paragraph</p>
    </div>
</body>
</html>
```

You will normally say that all the extra spaces will reflect back on the webpage. But it is not so. Because HTML treats all the extra spaces as a single space only and automatically removes all of them. Therefore, if you want to use extra spaces or any special characters, then you have to use HTML entities. To get extra space we can use ***&nbsp*** (non-breaking space) after that particular text. For example-

```
<div class= “container”>
            <p>This is another &nbsp Paragraph</p>
</div>
```

Copy

By writing this way, we can create 1 extra space after the word *another.* By adding five *&nbsp,* we can get 5 extra spaces. However, I do not recommend using this method because it looks unprofessional. We will learn to create extra spaces in CSS with the help of margin, padding, or selectors that looks more professional.

Entities are also used to write some special characters that you cannot write from keyboards and also those words that are reserved in HTML. For example, if we want ***<p>*** to appear in the result, then it is not possible without the help of entities.

```
<div class= “container”>
          <p>This is another Paragraph<p></p>
</div>
```

There is a list of different reserved characters and hundreds of special characters that you cannot write without entities. There is no need to learn all those entities available. You are always free to take the help of various references available.

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

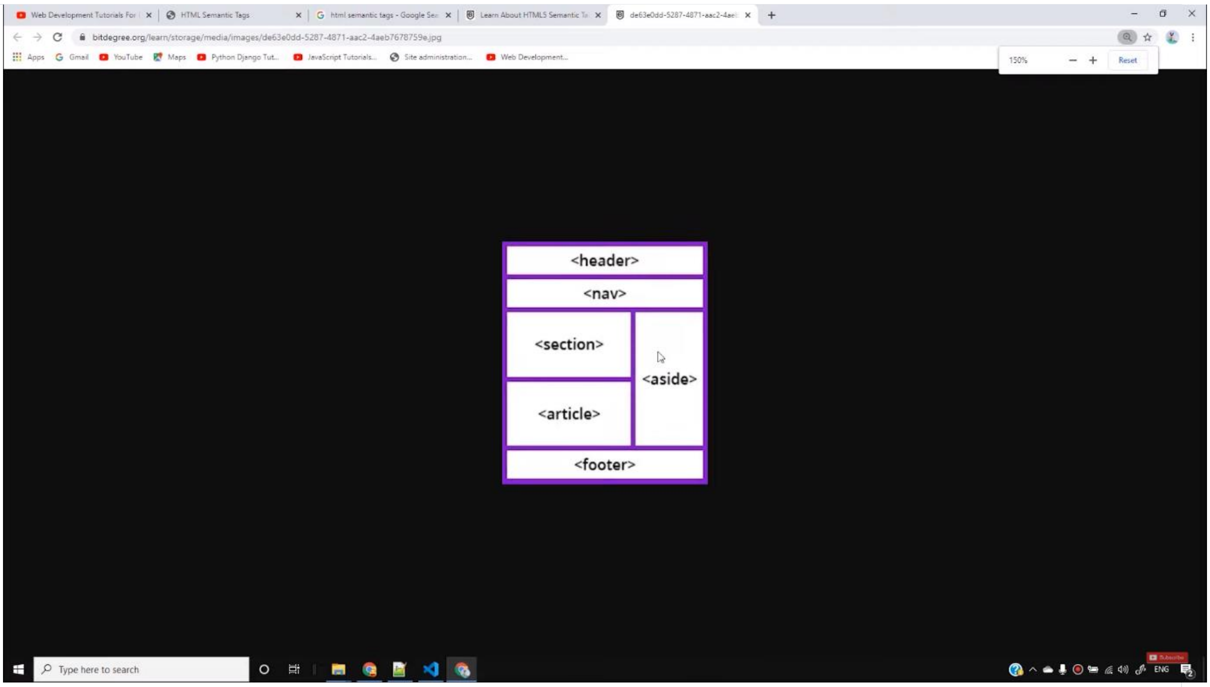
```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>HTML Entities</title>
</head>
<body>
    <div class="container">
        <p>This is a paragraph</p>
    </div>
    <div class="container">
        <p>This is another     paragraph with two spaces</p>
        <p>Paragraph is written like this <p> </p>
        <p>Pound is written like this £ </p>
        <p>Copyright is written like this © </p>
        <p>Another character is  ⇛ </p>
        <p>Another character is  ¼ </p>
        <p>Empty character is written like this  </p>
    </div>
</body>
</html>
```

Semantics are used to give proper meaning to the websites and help search engines to analyze it properly. Also, we can get help in return from different search engines in recognizing the original tags and their uses. For example, there are different tags used for the headers, navigation bars, body, and footers. So to analyze each and every tag properly, we have to use Semantics. The better the use of Semantics in a website, the more is the number of chances that it is being crawled by the search engines. It will also help in the better ranking of a website.

In very simple language, Semantic means to provide meaning to any word. If we talk in about a web page, then earlier most of the websites are made only using *divs* including different classes and IDs in them. But that was not the correct way of telling the search engines that if the elements are header, footer, or any other body.

There are many examples of Semantics. You can take the reference of the internet to view all. Here are some of the important semantic elements-

- **<header>**
- **<nav>**
- **<section>**
- **<article>**
- **<footer>**


bitdegree.org/learn/storage/media/images/de63e0dd-5287-4871-aac2-4aeb7678759e.jpg
150%
Reset
<header>
<nav>
<section>
<aside>
<article>
<footer>
Type here to search

With these examples, the search engines can easily differentiate between headers, navigation bars, sections, bodies, and footers. Also, when we share the link of this website in social media, then the viewers will easily understand headings, sections, and body of a website.

But now the question arises, is it necessary to use all these things? The straight answer is **No**. It is not necessary to use the semantics but including it will help in the SEO part and increase the probability of ranking of your website. If you are working on a blog, then you must use this technique to improve its ranking. On the other hand, non-semantic elements do not justify their meaning. For example, div and span don't tell much what they do apart from the division between texts.

The semantics part in the HTML is not so difficult, therefore without wasting much time we will directly jump on the CSS section where we will learn to make some good looking and real websites. It is always better to learn HTML, CSS, and JavaScript each 75% rather than learning only 100% HTML. You should first try to build the basic foundation of web development and then keep learning step by step through practising. Because you will never be able to learn 100% of web development. Therefore, try to improve skills by practising more.

Code as described/written in the video

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>HTML Semantic Tags</title>
</head>
<body>
    <h3>Semantic Elements</h3>
    <details>
```

```
        <summary>I have keys but no doors. I have space but no room.
You can enter but can’t leave. What am I?
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.
        A keyboard.</summary>
        lorem34
    </details>

</body>
</html>
```

The ID is an identifier which must be unique in the whole HTML document. It is used to find an element while linking, scripting, or styling. Whereas, Classes allow CSS and JavaScript to select and access specific elements

Here, I'll try to explain with the vert basic example. When a new child is born, we urge to give him a name or his identity by which he will be known further. Or if you are having a pet, you must have given him some name to call. In the same way, IDs refer to giving a name to any particular element for its identity. It simply refers giving an identity to an element. We know, no two names can be given to any of the two members of the family. In the same way, one ID can be given to only one element on a website. Therefore, in the below example, the id ***mainBox*** cannot be given to any other element.

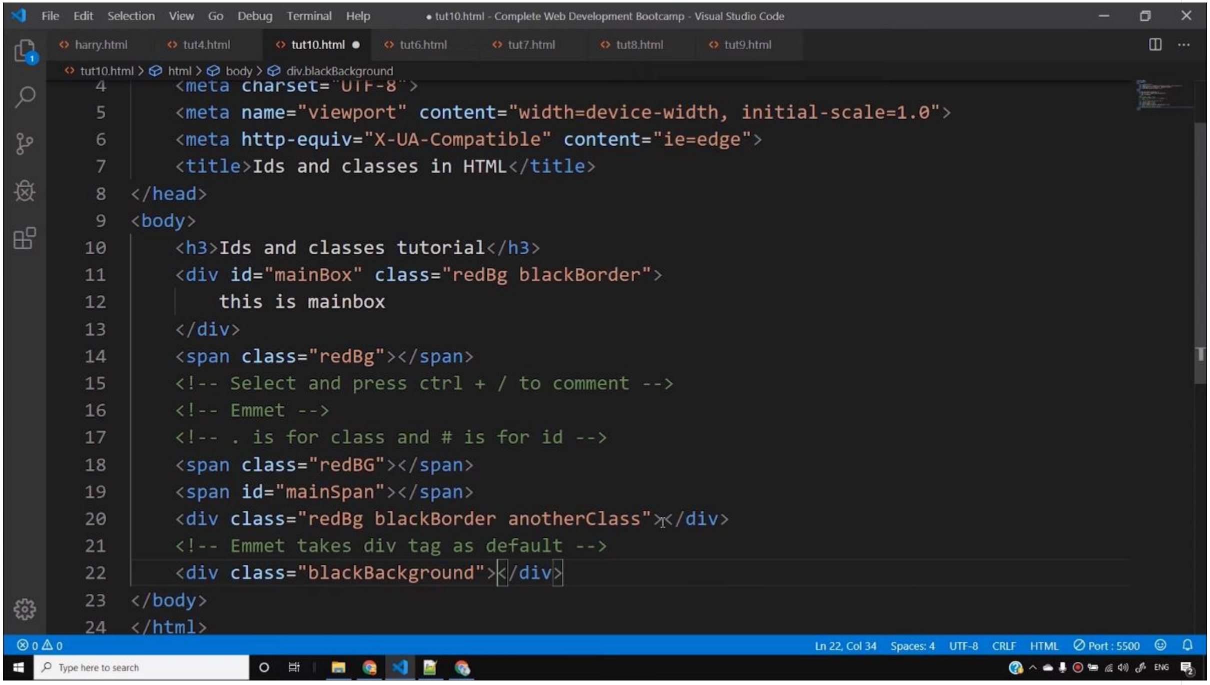
```
<div id= “mainBox” class= “redBG”>
```

Copy

Now the question arises what is the need for an ID in HTML? The answer is, while using JavaScript or CSS, we can target one full element and can make the necessary changes in it. In the same way, we can grab the full element and change the border or width or many more things through CSS.

Let us now understand what are classes with an example. Assume that I am having 100 elements in my HTML and I want to give a red background to all the 100 elements. To do this, we have two options. Either we have to select each element and assign a red background to it or we can create a class ***redBG*** and assign a red background to it. Then we can give this class to the elements in which we want a red background color. To avoid confusion, I am assuming that the class redBG is already defined.

One point to note here is we can assign only one ID to a particular element but it is not so in the case of classes. An element can have more than one class in itself. The more classes we add in an element, the more property will get added to it. Classes are denoted by a dot '.' and ID is denoted by hash '#'. For example, to get a redBG class in an element we can simply write that element name followed by .redBG. The below picture shows everything, you have learned till now-


File Edit Selection View Go Debug Terminal Help
• tut10.html - Complete Web Development Bootcamp - Visual Studio Code
harry.html
tut4.html
tut10.html ●
tut6.html
tut7.html
tut8.html
tut9.html
tut10.html > html > body > div.blackBackground
⊗ 0 ⚠ 0
Ln 22, Col 34
Spaces: 4
UTF-8
CRLF
HTML
Port : 5500
Type here to search
ENG

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Ids and classes in HTML</title>
</head>
<body>
    <h3>Ids and classes tutorial</h3>
    <div id="mainBox" class="redBg blackBorder">
        this is mainbox
    </div>
    <span class="redBg"></span>
    <!-- Select and press ctrl + / to comment -->
    <!-- Emmet -->
    <!-- . is for class and # is for id -->
    <span class="redBG"></span>
    <span id="mainSpan"></span>
    <div class="redBg blackBorder anotherClass"></div>
    <!-- Emmet takes div tag as default -->
    <div class="blackBackground"></div>
</body>
</html>
```

These are the major differences between classes and IDs. You can take the help of the references provided and practice more to understand more about them. However, we will learn more in detail about them in CSS and JavaScripts.

```
<!DOCTYPE html>
```

```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Ids and classes in HTML</title>
</head>
<body>
    <h3>Ids and classes tutorial</h3>
    <div id="mainBox" class="redBg blackBorder">
        this is mainbox
    </div>
    <span class="redBg"></span>
    <!-- Select and press ctrl + / to comment -->
    <!-- Emmet -->
    <!-- . is for class and # is for id -->
    <span class="redBg"></span>
    <span id="mainSpan"></span>
    <div class="redBg blackBorder anotherClass"></div>

    <!-- Emmet takes div tag as default -->
    <div class="blackBackground"></div>

    <!-- Creating multiple elements using Emmet -->
    <!-- span.myClass.myClass2.myClass3*4 + <Tab> to print 4 similar elements using Emmet -->
    <span class="myClass myClass2 myClass3">First</span>
    <span class="myClass myClass2 myClass3">Second</span>
    <span class="myClass myClass2 myClass3">Third</span>
    <span class="myClass myClass2 myClass3">Fourth</span>

</body>
</html>
```

Emoji Characters

Emojis are also characters from the UTF-8 alphabet:

- 😄 is 128516
- 😍 is 128525
- 💗 is 128151

Example

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
</head>
<body>

<h1>My First Emoji</h1>

<p>&#128512;</p>

</body>
</html>
```

Some Emoji Symbols in UTF-8

Emoji	Value
🗻	&#128507;
🗼	&#128508;
🗽	&#128509;

🗾	&#128510;
🗿	&#128511;
😀	&#128512;
😁	&#128513;
😂	&#128514;
😃	&#128515;
😄	&#128516;
😅	&#128517;

What is HTML Canvas?

The HTML `<canvas>` element is used to draw graphics, on the fly, via JavaScript.

The `<canvas>` element is only a container for graphics. You must use JavaScript to actually draw the graphics.

Canvas has several methods for drawing paths, boxes, circles, text, and adding images.

Browser Support

The numbers in the table specify the first browser version that fully supports the `<canvas>` element.

Element					
<canvas>	4.0	9.0	2.0	3.1	9.0

Canvas Examples

A canvas is a rectangular area on an HTML page. By default, a canvas has no border and no content.

The markup looks like this:

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

Note: Always specify an `id` attribute (to be referred to in a script), and a `width` and `height` attribute to define the size of the canvas. To add a border, use the `style` attribute.

Here is an example of a basic, empty canvas:

Example

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid
#000000;">
</canvas>
```

SVG defines vector-based graphics in XML format.

What is SVG?

- SVG stands for Scalable Vector Graphics

- SVG is used to define graphics for the Web
- SVG is a W3C recommendation

The HTML <svg> Element

The HTML `<svg>` element is a container for SVG graphics.

SVG has several methods for drawing paths, boxes, circles, text, and graphic images.

Browser Support

The numbers in the table specify the first browser version that fully supports the `<svg>` element.

Element				
<svg>	4.0	9.0	3.0	3.2

SVG Circle

Example

```
<!DOCTYPE html>
<html>
<body>

<svg width="100" height="100">
  <circle cx="50" cy="50" r="40" stroke="green" stroke-
width="4" fill="yellow" />
</svg>

</body>
</html>
```

The HTML <video> Element

To show a video in HTML, use the `<video>` element:

Example

```
      <!DOCTYPE html>

      <html>

      <body>



<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
</video>

<p>

      Video courtesy of

      <a href="https://www.bigbuckbunny.org/"
target="_blank">Big Buck Bunny</a>.

      </p>


      </body>

      </html>
```

The HTML <audio> Element

To play an audio file in HTML, use the `<audio>` element:

Example

```
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio>
```

Try it Yourself »

HTML Audio - How It Works

The `controls` attribute adds audio controls, like play, pause, and volume.

The `<source>` element allows you to specify alternative audio files which the browser may choose from. The browser will use the first recognized format.

The text between the `<audio>` and `</audio>` tags will only be displayed in browsers that do not support the `<audio>` element.

HTML <audio> Autoplay

To start an audio file automatically, use the `autoplay` attribute:

Example

```
<audio controls autoplay>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio>
```

```
<!DOCTYPE html>
<html>
<body>

<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio>

</body>
</html>
```

The easiest way to play videos in HTML, is to use YouTube.

Struggling with Video Formats?

Converting videos to different formats can be difficult and time-consuming.

An easier solution is to let YouTube play the videos in your web page.

YouTube Video Id

YouTube will display an id (like tgbNymZ7vqY), when you save (or play) a video.

You can use this id, and refer to your video in the HTML code.

Playing a YouTube Video in HTML

To play your video on a web page, do the following:

- Upload the video to YouTube
- Take a note of the video id
- Define an `<iframe>` element in your web page
- Let the `src` attribute point to the video URL
- Use the `width` and `height` attributes to specify the dimension of the player
- Add any other parameters to the URL (see below)

Example

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY">
</iframe>
```

Try it Yourself »

YouTube Autoplay + Mute

You can let your video start playing automatically when a user visits the page, by adding `autoplay=1` to the YouTube URL. However, automatically starting a video is annoying for your visitors!

Note: Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add `mute=1` after `autoplay=1` to let your video start playing automatically (but muted).

YouTube - Autoplay + Muted

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY?autoplay=1&mute=1">
</iframe>
```

Try it Yourself »

YouTube Playlist

A comma separated list of videos to play (in addition to the original URL).

YouTube Loop

Add `loop=1` to let your video loop forever.

Value 0 (default): The video will play only once.

Value 1: The video will loop (forever).

YouTube - Loop

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY?playlist=tgbNymZ7vqY&loop=1">
</iframe>
```

Try it Yourself »

YouTube Controls

Add `controls=0` to not display controls in the video player.

Value 0: Player controls does not display.

Value 1 (default): Player controls display.

YouTube - Controls

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY?controls=0">
</iframe>
```

```
<!DOCTYPE html>
<html>
<body>

<iframe width="420" height="345" src="https://www.youtube.com/embed/tgbNymZ7vqY">
</iframe>

</body>
</html>
```