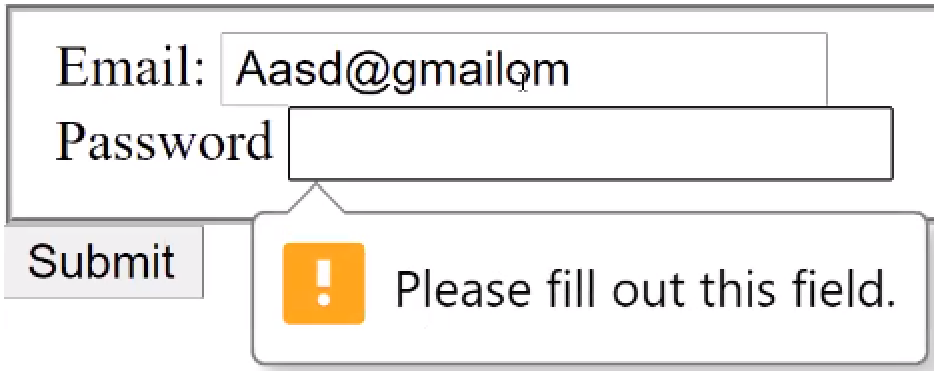
HTML

INPUT:

Required is a Boolean attribute. So it takes either true or false. If we have the required attribute in our input tag, when we click on the submit button, the browser it self would give a pop-up

```
<body>
  <form action="#">
    <fieldset>
      <div>
        <label for="emailID">Email</label>
        <input id="emailID" type="email" name="email" required="true">
     </div>
     <div>
         <label for="password">Password</label>
         <input id="password" type="password" name="pass" required="true">
     </div>
 </fieldset>
<input type="submit" name="">
</form>
</body>
```



Note: The required attribute works with the following input types: text, search, URL, tel, email, password, date pickers, number, checkbox, radio, and file.

If we fill both the input box then the URL will change from

file:///C:/Users/CLASSIC/Desktop/prac.html

file:///C:/Users/CLASSIC/Desktop/prac.html?email=name%40mail.com&pass=password#

The default behavior, <form> will get this action and put all the data into the key value pair in the URL. The entire <form> will re-direct the user to the URL.

Note: The required attribute works with the following input types: text, search, URL, tel, email, password, date pickers, number, checkbox, radio, and file.

Autocomplete:

Another attribute called autocomplete,

The autocomplete attribute specifies whether or not an input field should have autocomplete enabled. Autocomplete allows the browser to predict the value. When a user starts to type in a field, the browser should display options to fill in the field, based on earlier typed values.

Note: The autocomplete attribute works with the following input types: text, search, URL, tel, email, password, datepickers, range, and color.

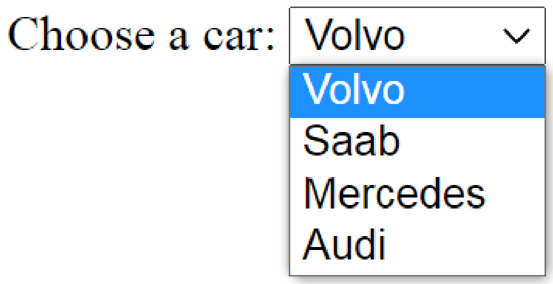
Select:

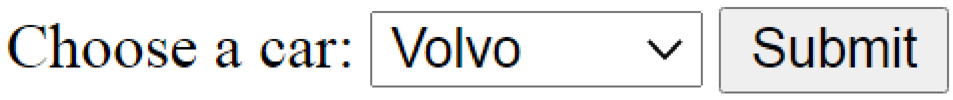
The <**select**> element is used to create a drop-down list. The <**select**> element is most often used in a form, to collect user input. The name attribute is needed to reference the form data after the form is submitted (if you omit the name attribute, no data from the drop-down list will be submitted). The id attribute is needed to associate the drop-down list with a label.

Input:

```
<label for="cars">Choose a car:</label>
<select name="cars" id="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="mercedes">Mercedes</option>
  <option value="audi">Audi</option>
</select>
```

Output:


Choose a car: Volvo
Volvo
Saab
Mercedes
Audi

When we select 

The URL would show the change as follows

file:///C:/Users/CLASSIC/Desktop/prac.html

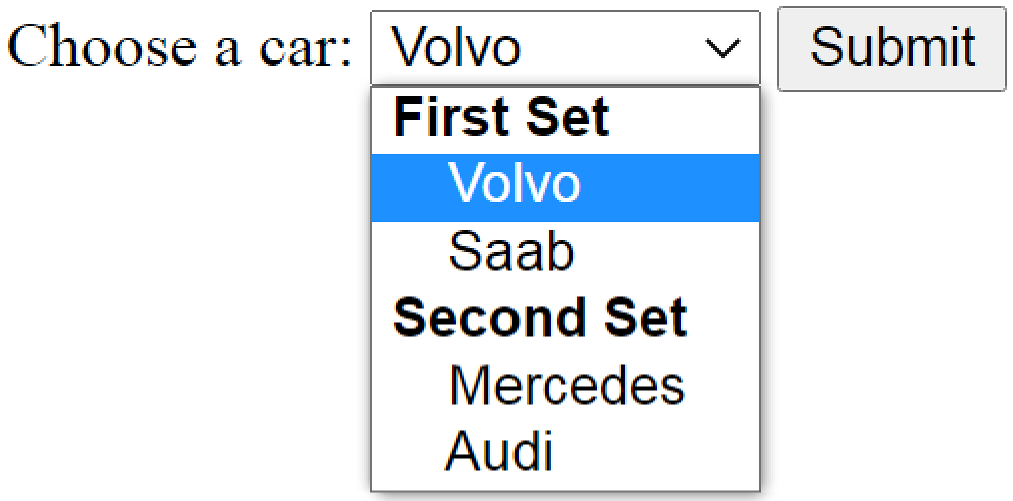
https://www.google.com/?cars=volvo

<optgroup>:

If we want to separate sets of colors, in the case we can group our options. We can use optgroup attribute for this purpose.

```
<label for="cars">Choose a car:</label>
<select name="cars" id="cars">
  <optgroup>
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
  </optgroup>
  <optgroup>
    <option value="mercedes">Mercedes</option>
    <option value="audi">Audi</option>
  </optgroup>
</select>
```

Output:



Audio & Video:

Early, few years back, any media content needed flash or silverlite (external 3rd part tool) should be installed to view it. This was before HTML5 came in. flash player was an external tool that was supposed to be installed into Chrome to play a mini-game or watch a video.

Now flash has been discontinued as it has become obsolete.

Video:

This is not a self-closing tag. In order to render the video, we need to provide a source (src).

<video src="https://d2ezlykacdqcnj.cloudfront.net/typing-macbook/typing-macbook.mp4"></video>

The video would not play. We need to add '**controls'** attribute

<video *controls* src="https://d2ezlykacdqcnj.cloudfront.net/typing-macbook/typing-macbook.mp4"></video>

We can put local videos using <href> tags.

The other attributes would be:

Auto play → this would run the video automatically.

Loop → the video keeps running unless you stop it. It keeps playing.

Muted → if you provide muted then the volume would be muted by default.

Width & Height → to set the width and height of the video size in the web page. We need to set the width and height ratio or else we will have empty space.

If our browser was not able to play a particular video extension, then it will ignore the file extension that is not being rendered and it will show the file that can be rendered.

Source tag can be general if you are not concerned about the formats. But if you want to go for multiple video extensions (not knowing if the browser will support a particular video type) then you have to give separate source code for every extension so that based on the format that the browser understands it will select the video extension.

Audio:

It would be a very similar approach. We will copy the URL and will share in the source attributes. The controls are similar to the video player options.

<audio *controls* src="https://www.soundhelix.com/examples/mp3/SoundHelix-Song-1.mp3"></audio>

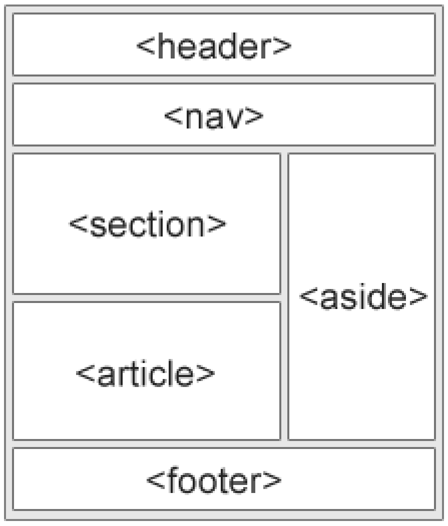
NOTE: If your video is not playing, then you have an option called poster attribute, which will accept the any image file.

<video controls poster="https://www.esa.int/var/esa/storage/images/esa_multimedia/images/2020/07/solar_orbiter_s_first_views_of_the_sun5/22136942-2-eng-GB/Solar_Orbiter_s_first_views_of_the_Sun_pillars.gif" width="500" src="https://d2ezlykacdqcnj.cloudfront.net/spinning-coin/spinning-coin.mp4"></video>

Semantic:

Semantic is the practice of giving content on the page meaning and structure by using the proper element. Semantic code describes the value of content on a page, regardless of the style or appearance of that content.

A semantic element clearly describes its meaning to both the browser and the developer.



Examples of non-semantic elements: <div> and <span> - Tells nothing about its content.

Examples of semantic elements: <form>, <table>, and <article> - Clearly defines its content.

Semantic Elements in HTML

Many web sites contain HTML code like: <div id="nav"> <div class="header"> <div id="footer"> to indicate navigation, header, and footer.

In HTML there are some semantic elements that can be used to define different parts of a web page:

Tag	Description
<article>	Defines independent, self-contained content
<aside>	Defines content aside from the page content
<details>	Defines additional details that the user can view or hide
<figcaption>	Defines a caption for a <figure> element
<figure>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
<footer>	Defines a footer for a document or section
<header>	Specifies a header for a document or section
<main>	Specifies the main content of a document
<mark>	Defines marked/highlighted text
<nav>	Defines navigation links
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time

ARIA – **A**ccessibility **R**ich **I**nternet **A**pplications

These are set of attributes that define ways to make web content and web applications (especially those developed with JavaScript) more accessible to people with disabilities.

<input type="submit" role="button">

<input type="reset" role="button">

https://www.w3.org/TR/html-aria/

https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA