**HTML**

**Introduction**

* Hyper Text Markup Language
* Used for creating structure of the webpage
* Current Version: 5
* Case insensitive

**Basic Structure of HTML document**

<!DOCTYPE html>

<html>

<head>

<meta name=”viewport” content=”width=device-width, initial-scale=1” >

<title></title>

</head>

<body>

<h1></h1>

<p></p>

</body>

</html>

**Explanation**

<!DOCTYPE html>: used to specify document type.

<html>: comprises entire html document

<head>: head of the document

<body>: body of the document

<h1>: heading tag

<p>: paragraph tag

**HTML Element**

* Starting tag + content + ending tag
* Content can be text or other html element
* Example:

<p>paragraph</p>

<br/>

**Nested HTML Elements**

* Elements can be nested means one elements can contain other elements

**Empty HTML Elements**

* HTML elements with no content are called empty elements.
* Example: <br/>, <img/>

**HTML Attribute**

* Additional information about tag
* Always specified in starting tag
* Example:

<img src=”” alt=”” />

<a href=”” target=”” ></a>

**HTML Comments**

* Used to explain/describe the code
* <!-- Comment -->

**HTML Headings**

* There are six heading tags are available in html, <h1> to <h6>
* All headings have different size
* <h1> has largest size and <h6> has smallest size.
* Block level element

**HTML Paragraphs**

* HTML paragraphs are defined with the <p> tag.
* Block level element

**HTML Images**

* <img> tag is used to embed an image in a web page.
* Images are not technically inserted into a web page; images are linked to web pages.

**Syntax:**

<img src="" alt="">

* ‘src’ attribute specifies the path of image.
* ‘alt’ attribute provides an alternate.
* Other Attributes: width, height,

**<picture> Element**

* The HTML <picture> element allows you to display different pictures for different devices or screen sizes.
* The <picture> element contains one or more <source> elements, each referring to different images through the ‘srcset’ attribute. This way the browser can choose the image that best fits the current view and/or device.
* Example :

<picture>   
  <source media="(min-width:650px)" srcset="img\_food.jpg">  
  <source media="(min-width: 465px)" srcset="img\_car.jpg">  
  <img src="img\_girl.jpg"> <!--Default-->  
</picture>

**favicon**

* A favicon is a small image displayed next to the page title in the browser tab.
* Syntax:

<link rel="icon" type="image/x-icon” href="">

**HTML Links**

* Syntax:

<a href="url">link text</a>

**HTML Tables**

<table align=”center” >

<caption></caption>

<thead>

<tr>

<th></th>

</tr>

</thead>

<tbody>

<tr>

<td></td>

</tr>

</tbody>

<tfoot>

<tr>

<td></td>

</tr>

</tfoot>

</table>

**Explanation**

* <table>: define table
* <tr>: table row
* <th>: table head
* <td>: table data
* <thead>: set of row defining table head
* <tbody>: set of row defining table body
* <tfood>: set of row defining table footer

**Table Attributes**

* Border
* Colspan
* Rowspan
* Cellpadding
* Cellspacing
* align

**HTML List**

Three types of list

1. Unordered List
2. Ordered List
3. Definition list

**Unordered List**

* <ul> tag is used to define unordered list.
* <li> tag is used to define list items.
* ‘type‘ attribute used to change the marker points, It can have possible values disc, square, circle.

**Ordered List**

* <ul> tag is used to define unordered list.
* <li> tag is used to define list items.
* ‘type‘ attribute used to change the marker points, It can have possible values ‘1’, ‘A’, ‘a’, ‘i’, ‘I’.

Note: By default, an ordered list will start counting from 1. If you want to start counting from a specified number, you can use the start attribute:

**Description Lists**

* <dl> tag is used to define description list.
* <dt> tag is used to define list items.
* <dd> tag is used to define term description.

**Block and Inline Elements**

* A block-level element always starts on a new line, and the browsers automatically add some space (a margin) before and after the element.
* Example: <div>, <section>, <h1> to <h6>, <p>, <img>, <footer>
* An inline element does not start on a new line. An inline element only takes up as much width as necessary.
* Note: An inline element cannot contain a block-level element!
* Example: <a>, <span>, <b>, <i>, <u>, <strong>

**Iframes Tags**

* Inline frame.
* An HTML iframe is used to display a web page|document within a web page.
* Syntax:

<iframe src="url" title=""></iframe>

* Use the height and width attributes to specify the size of the iframe.

**Computer Code**

* <kbd>:  keyboard input
* <samp>: sample output from a computer program
* <code>: piece of computer code
* <var>: variable in programming or in a mathematical expression
* <pre>: preformatted text
* Example :

<code>

<var>int x;</var>

</code>

**Meta Tag:**

* Used to specify the additional information about document.
* Used in SEO.
* Example:

<meta charset="UTF-8">

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

<meta http-equiv="refresh" content="30">

**HTML Forms:**

* Form is used to collect user information.

**The <form> Element:**

* The HTML <form> element is used to create an HTML form for user input.
* The <form> element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.
* Syntax:

<form></form>

* From attributes:

**action**: Defines the action to be performed when the form is submitted

**method**: The method attribute specifies how to send form-data (the form-data is sent to the page specified in the action attribute)

**target**: Specify the target window or frame where the result of the script will be displayed.

**enctype**: We can use the enctype attribute to specify how the browser encodes

the data before it sends it to the server.

**Autocomplete:** attribute specifies whether a form should have autocomplete on or off.

**<Input> Element:**

* An <input> element can be displayed in many ways, depending on the **type** attribute.
* The <input type="text"> defines a single-line input field for text input.
* Example:

<form>  
  <input type="text" id="firstname" name="firstname"><br>  
  <input type="text" id="lastname" name="lastname">  
</form>

**<label> Element:**

* The <label> tag defines a label for many form elements.
* The for attribute of the <label> tag should be equal to the id attribute of the <input> element to bind them together.
* Example:

<form>

<label for="firstname">First name:</label><br/>

<input type="text" id="firstname"><br>

<label for="lastname">Last name:</label><br/>

<input type="text" id=" lastname" name="lname">

</form>

**Radio Button:**

* The <input type="radio"> defines a radio button.
* Radio buttons let a user select **ONE** of number of choices.
* Example:

<form>  
  <input type="radio" id="html"

name="**language**" value="HTML">

<label for="html">HTML</label><br>  
  <input type="radio" id="css"

name="**language**" value="CSS">

<label for="css">CSS</label><br>

</form>

**Checkbox Button:**

* The <input type="checkbox"> defines a checkbox.
* Checkboxes let a user select ZERO or MORE options of number of choices.
* Example:

<form>  
  <input type="checkbox" id="vehicle1"

name="vehicle" value="Bike">  
<label for="vehicle1"> I have a bike</label><br>  
<input type="checkbox" id="vehicle2"

name="vehicle" value="Car">  
<label for="vehicle2"> I have a car</label><br>  
<input type="checkbox" id="vehicle3"

name="vehicle" value="Boat">  
<label for="vehicle3"> I have a boat</label>

</form>

**Submit Button:**

* The <input type="submit"> defines a button for submitting the form data to a form-handler.
* The form-handler is specified in the form's action attribute.
* Example:

<form action=”http://example.com” >  
  <input type="checkbox" id="vehicle1"

name="vehicle" value="Bike">  
<label for="vehicle1"> I have a bike</label><br>  
<input type="checkbox" id="vehicle2"

name="vehicle" value="Car">  
<label for="vehicle2"> I have a car</label><br>  
<input type="checkbox" id="vehicle3"

name="vehicle" value="Boat">  
<label for="vehicle3"> I have a boat</label>

<input type="submit" value="Submit">

</form>

**name attribute of <input>:**

* Notice that each input field must have a name attribute to be submitted.
* If the name attribute is omitted, the value of the input field will not be sent at all.

**‘type’ attribute:**

* The type attribute in the <input> element defines the kind of input control that will be displayed.
* Possible values for the type attribute and their descriptions:

1. **text**: Creates a single-line text input field.
2. **password**: Similar to the text input, but the characters are hidden.
3. **email**: Provides a text input field specifically designed for email addresses.
4. **number**: Displays a text input field for numeric values.
5. **tel**: Used for inputting telephone numbers.
6. **url**: Used for inputting URLs (web addresses).
7. **search**: Creates a search input field.
8. **date**: Allows users to select a date from a date picker.
9. **time**: Allows users to select a time from a time picker.
10. **datetime-local**: Allows users to input a date and time (local time).
11. **month**: Allows users to select a month and year.
12. **week**: Allows users to select a week and year.
13. **color**: Presents a color picker for selecting colors.
14. **checkbox**: Creates a checkbox, allowing users to select multiple options.
15. **radio**: Creates a radio button, allowing users to select a single option from a group.
16. **file**: Creates a file upload input field.
17. **submit**: Creates a submit button for submitting forms.
18. **reset**: Creates a reset button to reset form fields.
19. **button**: Creates a generic button.
20. **hidden**: Hides the input element from the user while still being present in the HTML.

**Input field attributes:**

1. **checked**: Specifies that an input field should be pre-selected when the page loads (for type="checkbox" or type="radio")
2. **disabled:** Specifies that an input field should be disabled.
3. **max:** Specifies the maximum value for an input field.
4. **maxlength**: Specifies the maximum number of character for an input field
5. **min:** Specifies the minimum value for an input field
6. **minlength:** specifies the minimum number of characters required in an input field.
7. **pattern**: Specifies a regular expression to check the input value against
8. **readonly**: Specifies that an input field is read only (cannot be changed)
9. **required**: Specifies that an input field is required (must be filled out)
10. **size**: Specifies the width (in characters) of an input field
11. **step**: Specifies the legal number intervals for an input field
12. **value**: Specifies the default value for an input field
13. **from:** Specify that the input field belongs to which form.
14. **Form action:** Specifies the URL of the file that will process the input when the form is submitted.
15. **Form method:** Defines the HTTP method for sending form-data to the action URL.

**readonly vs disabled attributes:**

* The value of a **read-only** input field will be sent when submitting the form while the value of **disabled** input field is not send when submitting the form.
* **read-only** input field allows some user interaction like copy the value but **disabled** input filed not allowed any user interaction.

**<select> and <option> element:**

* The <select> element defines a drop-down list:
* The <option> element defines an option that can be selected.
* By default, the first item in the drop-down list is selected.
* To define a pre-selected option, add the **selected** **attribute** to the option:
* Use the **multiple** attribute to allow the user to select more than one value:
* Example:

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

**<textarea> element:**

* The <textarea> element defines a multi-line input field (a text area).
* The rows attribute specifies the visible number of lines in a text area.
* The cols attribute specifies the visible width of a text area.
* Example:

<textarea name="message" rows="10" cols="30">

The cat was playing in the garden.

</textarea>

**<button> element:**

* The <button> element defines a clickable button:
* Example:

<button type="button"

onclick="alert('Hello World!')">

Click Me!

</button>

**<fieldset> and <legend> Elements:**

* The <fieldset> element is used to group related data in a form.
* The <legend> element defines a caption for the <fieldset> element.
* Example:

<fieldset>

<legend>First Name</legend>

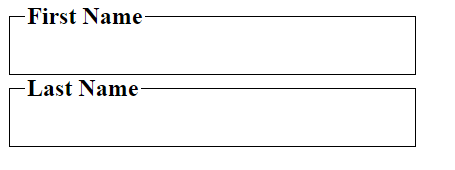
<input type="text" name="firstname" id="firstname" placeholder=”Enter first name” />

</fieldset>

<fieldset>

<legend>Last Name</legend>

<input type="text" name="lastname" id="lastname" placeholder=”Enter last name” /> </fieldset>



**<datalist> Element:**

* The <datalist> element specifies a **list** of pre-defined options for an <input> element.
* The **list** attribute of the <input> element, must refer to the **id** attribute of the <datalist> element.
* Example:

<form>  
  <input list="**browsers**">  
  <datalist id="**browsers**">  
    <option value="Chrome">

<option value="edge">

<option value="Firefox">  
  </datalist>  
</form>

**HTML Canvas Graphics**

* The HTML <canvas> element is used to draw graphics, using JavaScript.
* The <canvas> element is only a container for graphics. We must use JavaScript to actually draw the graphics.
* Canvas by default inline-block element.
* Canvas has several methods for drawing paths, boxes, circles, text, and adding images.
* A canvas is a rectangular area on an HTML page. By default, a canvas has no border and no content.
* Example:

        <canvas

            id="my-canvas"

            width="100"

            height="10"

            style="border: 1px solid #000000"

        ></canvas>

* After creating the rectangular canvas area, you must add a JavaScript to do the drawing.

**Canvas element Javascript Methods:**

* The **getContext** method in HTML's <canvas> element is used to obtain a rendering context and create graphics using various drawing and manipulation methods.
* The most commonly used context type is "2d" for 2-dimensional graphics.
* Example:

let c = document.getElementById("my-canvas");

let context = c.getContext("2d");

Here are the methods available on the 2D rendering context obtained through **getContext("2d")**:

1. **Paths:**
   * **moveTo(x, y)**: Moves the pen to the specified point.
   * **lineTo(x, y)**: Draws a line from the current point to the specified point.
   * **arc(x, y, radius, startAngle, endAngle, anticlockwise)**: Draws an arc.
   * **rect(x, y, width, height)**: Adds a rectangle to the path.
2. **Transformation:**
   * **scale(scaleX, scaleY)**: Scales the current transformation.
   * **rotate(angle)**: Rotates the current transformation.
   * **translate(x, y)**: Translates the current transformation.

1. **Text:**
   * **fillText(text, x, y, [maxWidth])**: Fills text at the specified position.
   * **strokeText(text, x, y, [maxWidth])**: Strokes text at the specified position.
2. **Styles and Colors:**
   * **font**: Sets the font style used for text rendering.
   * **fillStyle**: Sets the fill color or pattern.
   * **strokeStyle**: Sets the stroke color or pattern.
   * **globalAlpha**: Sets the global transparency level.

**Drawing Line**

<canvas

id="my-canvas"

width="500"

height="500"

style="border: 1px solid #d3d3d3"

>

Browser does not support canvas tag.

</canvas>

        <script>

            let c = document.getElementById("my-canvas");

            let ctx = c.getContext("2d");

            ctx.moveTo(0, 0);

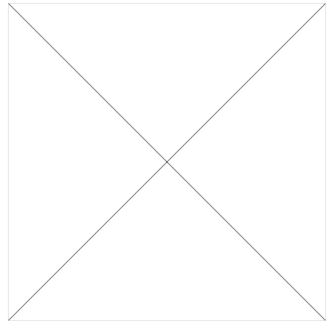
            ctx.lineTo(500, 500);

            ctx.moveTo(500, 0);

            ctx.lineTo(0, 500);

            ctx.stroke();

        </script>



**Drawing Circle**

<script>

let c=document.getElementById("my-canvas");

let ctx = c.getContext("2d");

ctx.beginPath();

ctx.arc(95,50,40,0,2\*Math.PI);

ctx.stroke();

</script>

**Draw a text:**

<script>

let c = document.getElementById("my-canvas");

let ctx = c.getContext("2d");

ctx.font = "30px Arial";

ctx.fillText("Hello World", 10, 50);

</script>

**Stroke text:**

<script>

let c = document.getElementById("my-canvas");

let ctx = c.getContext("2d");

ctx.font = "30px Arial";

ctx.strokeText("Hello World", 10, 50);

</script>

**Draw Liner gradient:**

<script>

let c = document.getElementById("my-canvas");

let ctx = c.getContext("2d");

// Create gradient

let grd = ctx.createLinearGradient(0, 0, 200, 0);

grd.addColorStop(0, "red");

grd.addColorStop(1, "white");

// Fill with gradient

ctx.fillStyle = grd;

ctx.fillRect(10, 10, 150, 80);

</script>

**Svg**

* SVG stands for Scalable Vector Graphics.
* The HTML <svg> element is a container for SVG graphics.
* Here are some of the ways to draw paths, shapes, text, and images using SVG.

1. **Basic Shapes:** SVG supports several basic shapes that can be drawn using simple elements:
   * **<rect x="x" y="y" width="width" height="height" />**: Draws a rectangle.
   * **<circle cx="cx" cy="cy" r="radius" />**: Draws a circle.
   * **<ellipse cx="cx" cy="cy" rx="x-radius" ry="y-radius" />**: Draws an ellipse.
   * **<line x1="x1" y1="y1" x2="x2" y2="y2" />**: Draws a line.
   * **<polyline points="x1,y1 x2,y2 ..." />**: Draws a connected series of line segments.
   * **<polygon points="x1,y1 x2,y2 ..." />**: Draws a closed polygon.
2. **Text:**
   * **<text x="x" y="y">Text content</text>**: Renders text at the specified position.

1. **Styles and Colors:** SVG allows you to define styles using attributes:
   * **fill**: Specifies the fill color.
   * **stroke**: Specifies the stroke color.
   * **stroke-width**: Specifies the stroke width.
   * **opacity**: Specifies the transparency.
2. **Gradients:** SVG supports linear and radial gradients:
   * **<linearGradient id="gradientId" x1="x1" y1="y1" x2="x2" y2="y2">...</linearGradient>**: Defines a linear gradient.
   * **<radialGradient id="gradientId" cx="cx" cy="cy" r="radius">...</radialGradient>**: Defines a radial gradient.

**SVG Circle**

<svg width="500" height="500"

style="border: 1px solid #d3d3d3">

<circle

cx="250"

cy="250"

r="100"

stroke="black"

stroke-width="3"

fill="red"

/>

</svg>

**SVG Rectangle**

<svg width="500" height="500"

style="border: 1px solid #d3d3d3">

<rect

x="100"

y="100"

width="300"

height="300"

stroke="black"

stroke-width="3"

fill="red"

/>

</svg>

**SVG Polygon**

<svg width="500" height="500"

style="border: 1px solid #d3d3d3">

<polygon

points="250,100 325,400 100,200 400,200 175,400"

stroke="black"

stroke-width="3"

fill="red"

/>

</svg>

**Gradient**

<svg width="500" height="500"

style="border: 1px solid #d3d3d3">

<defs>

<linearGradient

id="grad1" x1="0%" y1="0%"

x2="100%" y2="0%" >

<stop

offset="0%"

style="stop-color: rgb(255, 255, 0);

stop-opacity: 1;" />

<stop

offset="100%"

style="stop-color: rgb(255, 0, 0);

stop-opacity: 1" />

</linearGradient>

</defs>

<rect

x="100" y="100"

width="300" height="300"

stroke="black"

stroke-width="3"

fill="url(#grad1)" />

</svg>

**Figure tag:**

* The figure tag in HTML is used to define a self-contained group of related content, such as an image, diagram, or table.
* The <figcaption> element is used to add a caption for the <figure> element.
* Example:

<figure>

<img src="./1.jpeg" alt="Iron Man" />

<figcaption>Iron Man</figcaption>

</figure>

**Video tag:**

* The HTML <video> element is used to show a video on a web page.

**Method 1: Using <source> Element**

* The <source> element within the <video> element to specify multiple video sources in different formats.
* The browser will choose the **first** compatible format and play it.
* Example:

<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>

**Method 2: Using ‘src’ Attribute**

* In this method, we use the src attribute directly within the <video> element to specify a single video source.
* Example:

<video controls src="video.mp4" type="video/mp4">

Your browser does not support the video tag.

</video>

**Video tag Attribute:**

1. **autoplay**: This attribute automatically starts playing the video as soon as it's loaded.
2. **controls**: When included, this attribute adds default playback controls (play, pause, volume, etc.) to the video player.
3. **loop**: If this attribute is present, the video will automatically restart from the beginning after it reaches the end.
4. **muted**: This attribute mutes the audio of the video when set to **true**.
5. **preload**: It indicates how the browser should preload the video. Possible values are:
   * **"auto"**: The browser should load the entire video when the page loads.
   * **"metadata"**: Only metadata (like duration and dimensions) should be loaded when the page loads.
   * **"none"**: The browser should not preload the video at all.
6. **poster**: Specifies an image to be displayed as a placeholder before the video is loaded or played.
7. **src**: This attribute specifies the URL of the video file to be played.
8. **width**: Sets the width of the video player in pixels or percentage.
9. **height**: Sets the height of the video player in pixels or percentage.
10. **type**: Specifies the MIME type of the video source.

**Audio tag:**

* The HTML <audio> element is used to play an audio file on a web page.

**Method 1: Using <source> Element**

* We can use the <source> element within the <audio> element to specify multiple audio sources in different formats.
* The browser will choose the first compatible format and play it.
* Example:

<audio controls>

<source src="audio.mp3" type="audio/mpeg">

<source src="audio.ogg" type="audio/ogg">

Your browser does not support the audio tag.

</audio>

**Method 2: Using src Attribute**

* We can use the ‘src’ attribute directly within the <audio> element to specify a single audio source.
* Example:

<audio controls src="audio.mp3" type="audio/mpeg">

Your browser does not support the audio tag.

</audio>

**Audio tag Attribute:**

1. **autoplay**: This attribute automatically starts playing the video as soon as it's loaded.
2. **controls**: When included, this attribute adds default playback controls (play, pause, volume, etc.) to the video player.
3. **loop**: If this attribute is present, the video will automatically restart from the beginning after it reaches the end.
4. **muted**: This attribute mutes the audio of the video when set to **true**.
5. **preload**: It indicates how the browser should preload the video. Possible values are:
   * **"auto"**: The browser should load the entire video when the page loads.
   * **"metadata"**: Only metadata (like duration and dimensions) should be loaded when the page loads.
   * **"none"**: The browser should not preload the video at all.
6. **src**: This attribute specifies the URL of the video file to be played.
7. **type**: Specifies the MIME type of the video source.

**Embed tag:**

* The <embed> tag used to embed external content at the in the HTML document.
* Example:

<embed width="100%" height="500px" src=" 1.13-videoTag.html">

* Attributes used in the <embed> tag:

**src:** This attribute specifies the URL or path to the content you want to embed.

**width:** This attribute sets the width of the embedded content.

**height:** This attribute sets the height of the embedded content.

**type:** This attribute indicates the MIME type of the content you are embedding. It's essential for the browser to understand the type of content being embedded.

**Phrase Tags:**

* **<em>:** Represents text with emphasized. (italic)
* **<mark>:** Highlights or marks a specific portion of text for reference or distinction.
* **<strong>:** Indicates text with strong importance, typically displayed in a bold font.
* **<abbr>:** Defines an abbreviation or acronym and provides an optional explanation for it.
* **<bdo>:** Overrides the text direction of its parent element for bidirectional text support.
* **<dfn>:** Represents a definition term, often used within a glossary or to mark a term being defined.
* **<blockquote>:** Represents a block of quoted text from another source.
* **<q>:** Defines a short inline quotation, typically enclosed within quotation marks.
* **<cite>:** Indicates the title of a creative work or the name of the author or source.
* **<address>:** Represents contact information for the author or owner of the surrounding content.

**Progress Tags:**

* The <progress> tag represents the completion progress of a task.
* Example:

<label for="file">Downloading progress:</label>

<progress id="file" value="80" max="100"></progress>



* Attributes:

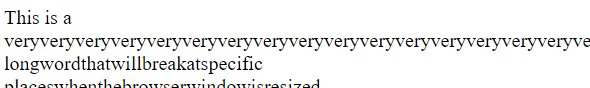
**Max:** Specifies how much work the task requires in total. Default value is 1.

**Value:** Specifies how much of the task has been completed.

**Wbr Tag:**

* The <wbr> (Word Break Opportunity) tag specifies where in a text it would be ok to add a line-break.
* <wbr> tag is an inline element
* When a word is too long, the browser might break it at the wrong place. We can use the <wbr> element to add word break opportunities.
* Example:

<p>This is a veryveryveryveryveryveryveryveryveryveryveryveryveryveryveryveryveryvery <wbr> longwordthatwillbreakatspecific <wbr> placeswhenthebrowserwindowisresized.</p>



**HTML Global Attributes**

* The global attributes are attributes that can be used with all HTML elements.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| [accesskey](https://www.w3schools.com/tags/att_global_accesskey.asp) | Specifies a shortcut key to activate/focus an element |
| [class](https://www.w3schools.com/tags/att_global_class.asp) | Specifies one or more classnames for an element (refers to a class in a style sheet) |
| [contenteditable](https://www.w3schools.com/tags/att_global_contenteditable.asp) | Specifies whether the content of an element is editable or not |
| [data-\*](https://www.w3schools.com/tags/att_global_data.asp) | Used to store custom data private to the page or application |
| [dir](https://www.w3schools.com/tags/att_global_dir.asp) | Specifies the text direction for the content in an element |
| [hidden](https://www.w3schools.com/tags/att_global_hidden.asp) | Specifies that an element is not yet, or is no longer, relevant |
| [id](https://www.w3schools.com/tags/att_global_id.asp) | Specifies a unique id for an element |
| [lang](https://www.w3schools.com/tags/att_global_lang.asp) | Specifies the language of the element's content |
| [style](https://www.w3schools.com/tags/att_global_style.asp) | Specifies an inline CSS style for an element |
| [title](https://www.w3schools.com/tags/att_global_title.asp) | Specifies extra information about an element |