HTML DOCUMENTATION

Create First HTML File

First make a new folder anywhere you want, and consider this as a project directory now open this folder into vs code , when your project folder is opened in vs code you can create a new file by **anyname.html** , we write .html to tell the computer that this file is an html file . Now whenever we open up this file , it is going to open up in our browser .HTML is a very different language so that it will never give us an error .

Basic Structure of HTML File

<!-- The <!DOCTYPE html> is written here to tell that this document is a type of html file -->

<!DOCTYPE html>

<!-- The below tag is saying that the whole html is written in between of <html></html> where as on attribute is also their lang='en' to tell that language used in the file is english -->

<html lang="en">

<!-- The below tag is head tag which is used to tell the browser of about description of the meta data

whereas the meta data is nothing but the information or data about data . -->

<head>

<meta charset="UTF-8">

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

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

<title>Document</title>

</head>

<!-- Anything you write in between of body tag is going to shown on the browser page -->

<body>

</body>

<!-- Now html is closed -->

</html>

Anything which is written in between **<>** is called tag . To write a tag **<html>Content</html>**

You have to open it with **<>** and close it with **</>**

Doctype

All HTML documents must start with a **<!DOCTYPE>** declaration.

The declaration is not an HTML tag. It is an "information" to the browser about what document type to expect.

In HTML 5, the declaration is simple **<!DOCTYPE html>**

In older documents (HTML 4 or XHTML), the declaration is more complicated because the declaration must refer to a DTD (Document Type Definition).

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

Elements and attribute

HTML **elements** are the building blocks of this language. Each element is an individual component on a particular web page . They are in every part of the page, The format of a tag is a start tag followed by the content and an end tag, eventually.

<tagname>CONTENT</tagname>

<!-- <tagname> = starting tag -->

<!-- </tagname> = closing tag -->

<!-- <tagname>CONTENT</tagname> = element -->

However, not all the HTML elements have closing words. They go by the names void, singleton, or empty elements.

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

The above tag is self closing tag

HTML **attribute** is what modifies an HTML element. It is usually in the form of unique words that one inserts inside the opening tag. They control the behavior of the element that follows.

```
<element attribute="value">element content</element>
```

Here in the image example src and alt is also an attribute .

Indentation

Spacing and indentation should be consistent throughout your code. Many developers choose to use 4-space or 2-space indentation. In HTML, each nested tag should be indented exactly once inside of its parent tag.

Comment

Single line - **<!-- single line -->**

Double line - **<!-- The <!DOCTYPE html> is written here to tell that this document is a type of html file -->**

Headings

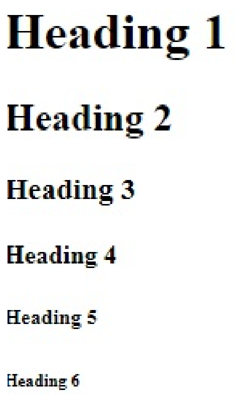
Search engine optimization uses the headings to index the structure and content of your web pages.Users often skip your page by its headings. It is important to use headings to show the document structure.

<h1> headings should be used for main headings, followed by <h2> headings, then the less important <h3>, and so on.

There are total 6 heading tags where as h1 is the biggest one and h6 is the smallest one not only in size but in the seo importance .

```
<h1>Heading 1</h1>

<h2>Heading 2</h2>

<h3>Heading 3</h3>

<h4>Heading 4</h4>

<h5>Heading 5</h5>

<h6>Heading 6</h6>
```

Output -


Heading 1
Heading 2
Heading 3
Heading 4
Heading 5
Heading 6

Paragraph and <pre></pre>

The p tag is an HTML tag used to define paragraphs of text. It is a block-level element, which means it creates a block of content that takes up the full width of its parent container. The p tag is used to group text together into logical sections and helps to structure the content of a web page.The basic usage of the p tag is to wrap text in a paragraph.

```
<p>This is a paragraph of text.</p>
```

In p tag there is a problem the browser automatically removes the extra line and spaces

So that why we use <pre> tag

- Formatting: The <p> tag does not preserve any formatting or spacing in the text, whereas the <pre> tag does.

- Line breaks: The <p> tag automatically adds a line break before and after the paragraph, whereas the <pre> tag preserves any line breaks in the text.
- Width: The <p> tag takes up the full width of its parent container, whereas the <pre> tag does not.

Formatting tag

HTML provides various formatting tags that allow you to change the appearance of text on web pages. These tags enable you to change the font, size, color, and style of text, as well as add emphasis

- **<b> and <strong> tags**

The <b> tag and the <strong> tag both indicate that the text should be displayed in bold. However, the <strong> tag is more semantically meaningful and is used to indicate that text is of greater importance or emphasis. The <b> tag should only be used for text that needs to be visually highlighted, but has no additional meaning.

Here is an example of using the <b> tag and the <strong> tag:

<b>This text is in bold.</b>

<strong>This text is more important than the surrounding text.</strong>

- **<i> and <em> tags**

The <i> tag and the <em> tag both indicate that the text should be displayed in italic. However, the <em> tag is more semantically meaningful and is used to indicate that text is emphasized, while the <i> tag is used for text that is simply styled in italic.

Here is an example of using the <i> tag and the <em> tag:

<i>This text is in italic.</i>

<em>This text is emphasized.</em>

- **<u> tag**

The <u> tag is used to underline text. However, it is generally not recommended to use this tag, as underlined text is often associated with hyperlinks.

Here is an example of using the <u> tag

<u>This text is underlined.</u>

- **<s> tag**

The <s> tag is used to strikethrough text. This tag is commonly used to indicate that text has been deleted or is no longer relevant.

Here is an example of using the <s> tag:

<s>This text has been strikethrough.</s>

- **<sup> and <sub> tags**

The <sup> tag is used to display superscript text, which is text that is smaller and higher up than the surrounding text. This tag is commonly used for footnotes and mathematical expressions.

The <sub> tag is used to display subscript text, which is text that is smaller and lower down than the surrounding text. This tag is commonly used for chemical formulas and mathematical expressions.

Here is an example of using the <sup> and <sub> tags:

E = mc<sup>2</sup> , H<sub>2</sub>O

- **<blockquote> tag**

The <blockquote> tag is used to create a block of quoted text. This tag is commonly used to quote external sources or to highlight particularly important text.

Here is an example of using the <blockquote> tag:

<blockquote>

"Be the change you wish to see in the world."

</blockquote>

HTML Links

HTML links are used to navigate between web pages or to link to other resources, Links are created using the <a> tag, which stands for "anchor."

Here is an example of using the <a> tag to create a link to a web page:

```
<a href="https://www.google.com">Click here to visit Example.com</a>
```

In the above example, the href attribute specifies the URL of the web page that the link points to. When the user clicks on the link, the browser will navigate to the specified URL.

- **Linking to Other Resources**

You can also use the <a> tag to link to other resources, such as images, audio files, or documents. Here is an example of using the <a> tag to create a link to an image:

```
<a href="images/cat.jpg">Click here to view the image of a cat</a>
```

In the above example, the href attribute specifies the path to the image file relative to the current HTML file. When the user clicks on the link, the browser will display the image.

- **Linking to Email Addresses**

You can use the <a> tag to create links that open the user's email client and create a new email message. Here is an example of using the <a> tag to create a link to an email address:

```
<a href="mailto:example@example.com">Click here to send an email</a>
```

In the above example, the href attribute specifies the email address that the new email message should be addressed to. When the user clicks on the link, the browser will open the user's email client with a new email message addressed to the specified email address.

- **Linking to Sections of a Web Page**

You can use the <a> tag to create links to specific sections of a web page. To do this, you need to use the id attribute to give the section a unique identifier, and then use that identifier in the link's href attribute. Here is an example:

```
<h2 id="section1">Section 1</h2>

<p>Content of section 1</p>

<h2 id="section2">Section 2</h2>

<p>Content of section 2</p>

<a href="#section1">Go to section 1</a>

<a href="#section2">Go to section 2</a>
```

In the above example, the <h2> tags have been given unique identifiers using the id attribute. The links to the sections use the href attribute with the value of the corresponding id attribute preceded by a "#" symbol. When the user clicks on the link, the browser will scroll to the specified section of the web page.

Images

The <img> tag in HTML is used to display images on web pages. It is a self-closing tag, meaning that it does not require a closing tag. The src attribute is the most important attribute of the <img> tag, as it specifies the URL of the image file that should be displayed.

Here is an example of using the <img> tag to display an image on a web page:

```
<img src="images/cat.jpg" alt="Picture of a cat">
```

In the above example, the src attribute specifies the path to the image file relative to the current HTML file. The alt attribute provides alternative text for the image, which is displayed if the image cannot be loaded or if the user is using a screen reader.

- **Adjusting the Size of an Image**

You can adjust the size of an image using the width and height attributes. These attributes specify the width and height of the image in pixels. Here is an example:

```
<img src="images/cat.jpg" alt="Picture of a cat" width="300" height="200">
```

In the above example, the width and height attributes have been set to 300 and 200 pixels, respectively.

- **Adding a Caption to an Image**

You can add a caption to an image by placing the <img> tag inside a <figure> element and adding a <figcaption> element. Here is an example

```
<figure>

<img src="images/cat.jpg" alt="Picture of a cat">

<figcaption>A picture of a cat</figcaption>
```

</figure>

In the above example, the <figure> element contains the <img> tag and the <figcaption> element, which provides a caption for the image.

Marquee

The <marquee> tag in HTML is used to create a scrolling text or image display. When the <marquee> tag is used, the text or image inside it moves across the screen horizontally or vertically, depending on how it is configured.

Here is an example of using the <marquee> tag to create a scrolling text display:

<marquee behavior="scroll" direction="left">

This text is scrolling from right to left!

</marquee>

In the above example, the behavior attribute is set to "scroll" to indicate that the text should scroll across the screen, and the direction attribute is set to "left" to indicate that the text should move from right to left.

You can also specify the speed at which the text scrolls by using the scrollamount attribute and the amount of space between the scrolling text and the edge of the marquee element by using the hspace and vspace attributes.

Here is an example of using the <marquee> tag to create a scrolling image display:

<marquee behavior="scroll" direction="up">

<img src="images/cat.gif" alt="Picture of a cat">

</marquee>

In the above example, the <img> tag is used inside the <marquee> tag to display the image. The behavior attribute is set to "scroll" to indicate that the image should scroll across the screen, and the direction attribute is set to "up" to indicate that the image should move from bottom to top.

It's important to note that the <marquee> tag is not part of the HTML5 specification and is considered deprecated. It is not recommended to use the <marquee> tag in modern web development because it can be distracting to users and can cause accessibility issues. Instead, CSS animations or JavaScript can be used to create similar effects in a more controlled and accessible way.

In conclusion, the <marquee> tag is used to create scrolling text or image displays in HTML. While it can be effective for certain purposes, it is not recommended for modern web development due to its potential drawbacks.

Ordered List

In HTML, the <ol> tag is used to create an ordered list. An ordered list is a numbered list where each item is preceded by a number, and the order of the items is important.

Here is an example of using the <ol> tag to create a simple ordered list:

<ol>

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ol>

In the above example, the <ol> tag is used to create the ordered list, and each item in the list is contained within an <li> tag. The result would look something like this:

1. First item
2. Second item
3. Third item

By default, the <ol> tag uses decimal numbering for the items, starting at 1. However, you can also use other types of numbering by using the type attribute. Here are some examples:

<ol type="A">

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ol>

<ol type="I">

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ol>

In the above examples, the type attribute is used to change the numbering type to uppercase letters (A, B, C) and Roman numerals (I, II, III), respectively.

You can also use the start attribute to specify a starting number for the list. For example:

<ol start="5">

<li>Fifth item</li>

<li>Sixth item</li>

<li>Seventh item</li>

</ol>

In the above example, the start attribute is used to start the list at 5 instead of 1.

In conclusion, the <ol> tag is used to create ordered lists in HTML, and it can be customized using the type and start attributes.

14. Unordered List

In HTML, the <ul> tag is used to create an unordered list. An unordered list is a list where the order of the items is not important, and each item is preceded by a bullet point or another marker.

Here is an example of using the <ul> tag to create a simple unordered list:

<ul>

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ul>

In the above example, the <ul> tag is used to create the unordered list, and each item in the list is contained within an <li> tag. The result would look something like this:

- First item
- Second item
- Third item

By default, the bullet points used in an unordered list are solid circles. However, you can also use other types of markers by using the type attribute. Here are some examples:

<ul type="square">

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ul>

<ul type="disc">

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ul>

In the above examples, the type attribute is used to change the bullet point type to squares and discs, respectively.

You can also use nested <ul> and <li> tags to create sub-lists within a list. Here is an example:

```
<ul>

<li>First item</li>

<li>Second item

<ul>

<li>Sub-item 1</li>

<li>Sub-item 2</li>

</ul>

</li>

<li>Third item</li>

</ul
```

In the above example, the second item in the list contains a nested unordered list with two sub-items.

In conclusion, the <ul> tag is used to create unordered lists in HTML, and it can be customized using the type attribute. You can also create nested lists using nested <ul> and <li> tags.

15. Tables

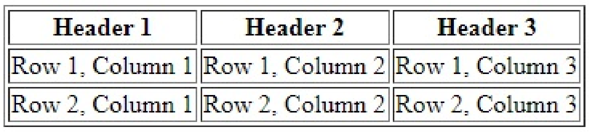
In HTML, the <table> tag is used to create tables. Tables are used to display data in rows and columns, and they can be customized using various attributes and styles.

Here is an example of using the <table> tag to create a simple table:

```
<table border ='1'>

<tr>

<th>Header 1</th>

<th>Header 2</th>

<th>Header 3</th>

</tr>

<tr>

<td>Row 1, Column 1</td>

<td>Row 1, Column 2</td>

<td>Row 1, Column 3</td>

</tr>

<tr>

<td>Row 2, Column 1</td>

<td>Row 2, Column 2</td>

<td>Row 2, Column 3</td>

</tr>

</table>
```

In the above example, the <table> tag is used to create the table, and each row is contained within a <tr> tag. The first row is a header row, and the headers are contained within <th> tags. The remaining rows are data rows, and the data is contained within <td> tags.

The result would look something like this:



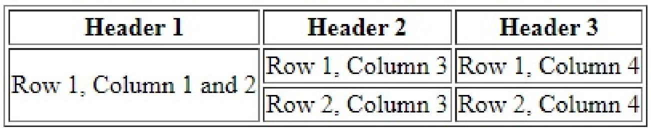
Header 1	Header 2	Header 3
Row 1, Column 1	Row 1, Column 2	Row 1, Column 3
Row 2, Column 1	Row 2, Column 2	Row 2, Column 3

Here the border attribute is giving border to the table if you remove it border will also remove

You can also use the colspan and rowspan attributes to merge cells in a table. Here is an example:

```
<table>

<tr>

<th>Header 1</th>

<th>Header 2</th>

<th>Header 3</th>

</tr>

<tr>

<td rowspan="2">Row 1, Column 1 and 2</td>

<td>Row 1, Column 3</td>

<td>Row 1, Column 4</td>

</tr>

<tr>

<td>Row 2, Column 3</td>

<td>Row 2, Column 4</td>

</tr>

<table>
```

In the above example, the first data cell in the first row spans two rows using the rowspan attribute. The result would look something like this:



Header 1	Header 2	Header 3
Row 1, Column 1 and 2	Row 1, Column 3	Row 1, Column 4
	Row 2, Column 3	Row 2, Column 4

You can also use various CSS styles to customize the appearance of tables, such as border, padding, and background color.

In conclusion, the <table> tag is used to create tables in HTML, and tables can be customized using various attributes and styles. Tables are a useful way to display data in rows and columns.

```
<table>

<caption>This is a table caption</caption>

<table>
```

This will give caption to table

16. Div vs Span

In HTML, <div> and <span> are two commonly used container elements that are used to group and format content on a web page.

The <div> element is a block-level container that is used to group larger sections of content together. It is often used to create a layout and divide a web page into different sections. The <div> element can contain other HTML elements such as headings, paragraphs, images, forms, and even other <div> elements. The <div> element can also have its own CSS styles applied to it to format its content.

Here's an example of using the <div> element to group content:

```
<div>
```

<h1>This is a heading</h1>

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

</div>

On the other hand, the <span> element is an inline container that is used to group smaller pieces of content together. It is often used to apply styles to specific parts of text or to group inline elements such as links or icons. The <span> element can be used inside other HTML elements such as headings, paragraphs, and lists. It can also have its own CSS styles applied to it.

Here's an example of using the <span> element to apply styles to a specific part of text:

<p>This is a <span style="color: red;">red</span> text.</p>

In the above example, the <span> element is used to apply the color red to the word "red" inside a paragraph.

In summary, while both <div> and <span> elements are used to group and format content, they differ in their use cases. The <div> element is used to group larger sections of content, while the <span> element is used to group smaller pieces of content and apply styles to them.

17. Block Vs inline

In HTML, elements can be classified as either block-level or inline elements based on how they behave on a web page.

Block-level elements are those that create a block of content on a web page, taking up the full width available to them. They start on a new line and can contain other block-level or inline elements within them. Some common examples of block-level elements are headings (<h1> to <h6>), paragraphs (<p>), lists (<ul>, <ol>, <dl>), and <div> elements. Block-level elements are often used for larger pieces of content like sections, articles, and navigation menus.

Here's an example of using a block-level element to create a paragraph:

<p>This is a paragraph of text. It takes up the full width of the container and starts on a new line.</p>

On the other hand, inline elements are those that are part of a block-level element and are typically used to style or format smaller pieces of content within a block. They do not create a new line and only take up as much space as they need to display their content. Some common examples of inline elements are <a>, <span>, <img>, and <strong>. Inline elements are often used to style text within a block-level element.

Here's an example of using an inline element to create a link within a paragraph:

<p>This is a paragraph of text with a <a href="https://example.com">link</a> to another web page.</p>

In summary, block-level elements create blocks of content that take up the full width available to them and start on a new line, while inline elements are used within block-level elements to style or format smaller pieces of content and do not create new lines.

18. Iframe

The <iframe> tag in HTML is used to embed a web page or other types of media content within an HTML document. It stands for "inline frame" and allows you to display another HTML page, a PDF file, a video, or any other type of content from another source within your web page.

The <iframe> tag requires a source attribute to specify the URL of the content you want to embed. You can also set other attributes to customize the appearance and behavior of the iframe, such as width, height, scrolling, frameborder, and allowfullscreen.

Here's an example of using the <iframe> tag to embed a YouTube video within an HTML document:

<iframe width="560" height="315" src="https://www.youtube.com/embed/dQw4w9WgXcQ" frameborder="0" allowfullscreen></iframe>

In this example, the width and height attributes set the dimensions of the iframe, and the src attribute specifies the URL of the YouTube video. The frameborder attribute sets whether or not the iframe has a border, and the allowfullscreen attribute enables the video to be played in fullscreen mode.

The <iframe> tag can be useful for embedding content from other websites, displaying dynamic content within your web page, or providing an interactive user experience. However, it's important to use it responsibly and securely, as it can also be used for malicious purposes such as phishing attacks or cross-site scripting (XSS) vulnerabilities.

19. HTML Audio

The <audio> tag in HTML is used to embed audio content within an HTML document. It provides a way to play audio files, such as music or sound effects, directly in the browser without requiring a separate plugin or application.

The <audio> tag requires a source attribute to specify the URL of the audio file, as well as other attributes to customize the appearance and behavior of the player, such as controls, autoplay, loop, preload, and volume.

Here's an example of using the <audio> tag to embed an MP3 file within an HTML document:

<audio controls>

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

```
Your browser does not support the audio element.

</audio>
```

In this example, the controls attribute adds a player with playback controls, such as play/pause, volume, and seek bar. The source element specifies the URL of the audio file and its MIME type. The content inside the <audio> tag is used as a fallback for browsers that do not support the <audio> tag or the specified audio format.

You can also use JavaScript to control the playback of the audio element, such as playing, pausing, or changing the volume. The <audio> tag supports a wide range of audio formats, including MP3, WAV, OGG, and AAC.

The <audio> tag can be useful for adding audio content to your web page, such as background music, podcasts, or audio feedback for user actions. However, it's important to consider the accessibility and usability of the audio content, as well as its impact on page load times and bandwidth usage.

20. HTML Video

The <video> tag in HTML is used to embed video content within an HTML document. It provides a way to play video files directly in the browser without requiring a separate plugin or application.

The <video> tag requires a source attribute to specify the URL of the video file, as well as other attributes to customize the appearance and behavior of the player, such as controls, autoplay, loop, preload, and poster.

Here's an example of using the <video> tag to embed an MP4 file within an HTML document:

```
<video controls>

<source src="video/movie.mp4" type="video/mp4">

Your browser does not support the video tag.

</video>
```

In this example, the controls attribute adds a player with playback controls, such as play/pause, volume, and seek bar. The source element specifies the URL of the video file and its MIME type. The content inside the <video> tag is used as a fallback for browsers that do not support the <video> tag or the specified video format.

You can also use JavaScript to control the playback of the video element, such as playing, pausing, or changing the volume. The <video> tag supports a wide range of video formats, including MP4, WebM, and Ogg.

The <video> tag can be useful for adding video content to your web page, such as tutorials, product demos, or promotional videos. However, it's important to consider the accessibility and usability of the video content, as well as its impact on page load times and bandwidth usage.

21. Embed PDF

To embed a PDF file in an HTML document, you can use the <embed> tag. The <embed> tag specifies an external application or interactive content within an HTML document, such as a PDF document, an audio or video file, or a Flash animation.

Here's an example of using the <embed> tag to embed a PDF file within an HTML document:

```
<embed src="document.pdf" width="500" height="600" type="application/pdf">
```

In this example, the src attribute specifies the URL of the PDF file to be embedded. The width and height attributes define the dimensions of the embedded content in pixels. The type attribute specifies the MIME type of the embedded content, in this case, application/pdf.

You can also use the <object> tag to embed a PDF file in HTML. The <object> tag allows you to specify alternate content, such as text or images, to be displayed if the browser does not support the embedded content. Here's an example of using the <object> tag to embed a PDF file:

```
<object data="document.pdf" type="application/pdf" width="500" height="600">

<p>Alternative text for the PDF file goes here.</p>

</object>
```

In this example, the data attribute specifies the URL of the PDF file to be embedded. The type attribute specifies the MIME type of the embedded content. The width and height attributes define the dimensions of the embedded content in pixels. The content inside the <object> tag is used as a fallback for browsers that do not support the <embed> tag or the specified MIME type.

Keep in mind that embedding large PDF files in HTML can significantly increase the page load time and bandwidth usage. It's also important to ensure that the PDF file is accessible and readable to users with disabilities.

22. Embed Youtube Video

To embed a YouTube video in an HTML document, you can use the <iframe> tag. Here's an example of how to embed a YouTube video:

<iframe width="560" height="315" src="https://www.youtube.com/embed/VIDEO_ID" frameborder="0" allowfullscreen></iframe>

In this example, replace VIDEO_ID with the unique ID of the YouTube video you want to embed. You can find the video ID in the video's URL after the "v=" parameter.

The width and height attributes define the dimensions of the embedded video in pixels. The frameborder attribute sets whether the video should have a border or not, and the allowfullscreen attribute allows the video to be played in full-screen mode.

You can also customize the appearance and behavior of the embedded video using various YouTube player parameters. For example, you can auto-play the video, hide the controls, or start the video at a specific time. You can find more information about the available player parameters on the YouTube Embedded Players and Player Parameters documentation.

Keep in mind that embedding videos from external sources like YouTube can affect your website's performance and page load times. It's also important to ensure that the video is relevant to your website's content and that you have the appropriate rights to use the video on your website.

23. Embed Google Maps

To embed a Google Map in an HTML document, you can use the <iframe> tag. Here's an example of how to embed a Google Map

<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d387315.0585970057!2d-
122.4462318770306!3d37.758679300433726!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x8085807dfe46646f%3A0xd8372b19d83c1c90!2sG
width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe>

In this example, replace the URL with the Google Maps URL for the location you want to embed. You can customize the map's appearance and behavior using various parameters, such as zoom level and map type. You can find more information about the available parameters on the Google Maps Embed API documentation.

The width and height attributes define the dimensions of the embedded map in pixels. The frameborder attribute sets whether the map should have a border or not, and the allowfullscreen attribute allows the map to be viewed in full-screen mode.

It's important to ensure that the embedded map is relevant to your website's content and that you have the appropriate permissions to use the map on your website. Additionally, embedding maps from external sources can affect your website's performance and page load times.

24. HTML Entities

HTML entities are codes that represent characters or symbols that have a special meaning in HTML. These entities are used to display reserved characters and symbols that would otherwise be interpreted as HTML code, rather than displayed on the webpage.

For example, the less than symbol (<) is used to begin an HTML tag, so if you want to display a less than symbol on a webpage, you would use the HTML entity code &lt; instead of typing the symbol directly. Similarly, the greater than symbol (>) is used to end an HTML tag, so you would use the entity code &gt; to display it.

There are many HTML entities that can be used to display a wide range of characters and symbols, including accented letters, mathematical symbols, and currency symbols. Here are some examples:

- &nbsp; - non-breaking space
- &copy; - copyright symbol
- &reg; - registered trademark symbol
- &frac12; - fraction half (½)
- &euro; - Euro currency symbol

To use an HTML entity in your code, simply replace the character or symbol with its corresponding entity code, and include it in your HTML document. It's important to note that some characters may require a specific encoding to display correctly, such as UTF-8, and you should always ensure that your webpage is properly encoded to avoid display issues.

HTML entities are an important tool for displaying special characters and symbols on a webpage, and can help ensure that your content is displayed correctly across different browsers and devices.

25. Semantics Element

Semantic elements in HTML refer to specific tags that provide meaning and structure to the content on a webpage. These tags describe the content within them and help search engines and screen readers to understand the purpose and context of the content.

Here are some common examples of semantic elements in HTML:

1. header: This tag defines a header for a document or section.
2. nav: This tag defines a section of navigation links.
3. main: This tag defines the main content of a document or section.
4. article: This tag defines a self-contained piece of content, such as a blog post or article.
5. section: This tag defines a generic section of content.
6. aside: This tag defines content that is tangentially related to the main content, such as a sidebar or callout box.
7. footer: This tag defines a footer for a document or section.

By using these semantic elements in HTML, web developers can create more meaningful and accessible web pages. They also help to improve the website's SEO by providing search engines with more information about the content on the page.

26. HTML Forms

HTML forms are used to collect user input and send it to a web server for processing. Forms can include a variety of input types, such as text fields, radio buttons, checkboxes, drop-down menus, and more.

Here's an example of a basic HTML form:

```
<form action="submit-form.php" method="post">

<label for="name">Name:</label>

<input type="text" id="name" name="name" required>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<label for="subject">Subject:</label>

<input type="text" id="subject" name="subject" required>

<label for="message">Message:</label>

<textarea id="message" name="message" rows="5" required></textarea>

<input type="submit" value="Submit">

</form>
```

In this example, we have a form element with four input fields and a submit button. The action attribute specifies where the form data should be submitted, and the method attribute specifies the HTTP method to use for the submission (in this case, post).

Each input field has a name attribute, which is used to identify the field on the server side when the form is submitted. The label element provides a text label for each input field, and the for attribute of the label is set to the id of the corresponding input field. This helps to make the form more accessible to users with disabilities.

The input element is used for text input fields, and the type attribute specifies the type of input field. In this example, we have used text for the name and subject fields, and email for the email field. The required attribute specifies that the field must be filled in before the form can be submitted.

The textarea element is used for multi-line text input, such as for the message field in this example. The rows attribute specifies the height of the textarea in rows.

Finally, we have the submit button, which is an input element with type="submit". When the user clicks this button, the form data is submitted to the server specified in the action attribute.

That's a basic overview of HTML forms with an example. I hope this helps!

27. Form Input Types

HTML forms allow users to input data that can be submitted to a server for processing. There are several different types of form input elements that can be used to collect different types of data. In this article, we will explore the most commonly used form input types in HTML.

Text input:

1. The most basic form input type is the text input. This input type allows users to enter text data into a form. The input can be single or multi line depending on the "type" attribute value.

```
<label for="username">Username:</label>

<input type="text" id="username" name="username">
```

Password input:

1. Password input is used when a user is required to enter a secure password. The text entered in this input type is masked with asterisks or dots.

```
<label for="password">Password:</label>

<input type="password" id="password" name="password">
```

Checkbox input:

1. Checkbox input is used when there is more than one option available, and a user can select more than one option.

```
<label for="fruits">Select your favourite fruits:</label>

<input type="checkbox" id="apple" name="fruit" value="apple">

<label for="apple">Apple</label><br>

<input type="checkbox" id="banana" name="fruit" value="banana">

<label for="banana">Banana</label><br>

<input type="checkbox" id="orange" name="fruit" value="orange">

<label for="orange">Orange</label><br>
```

Radio input:

1. Radio input is used when there is more than one option available, but only one option can be selected.

```
<label for="gender">Select your gender:</label>

<input type="radio" id="male" name="gender" value="male">

<label for="male">Male</label><br>

<input type="radio" id="female" name="gender" value="female">

<label for="female">Female</label><br>

<input type="radio" id="other" name="gender" value="other">

<label for="other">Other</label><br>
```

Select input:

1. Select input is used when there is a list of options available, and the user can only select one option.

```
<label for="country">Select your country:</label>

<select id="country" name="country">

<option value="usa">USA</option>

<option value="canada">Canada</option>

<option value="mexico">Mexico</option>

<option value="uk">UK</option>

<option value="france">France</option>

</select>
```

File input:

1. File input is used when the user is required to upload a file.

```
<label for="myfile">Select a file:</label>

<input type="file" id="myfile" name="myfile">
```

Submit input:

:

1. Submit input is used to submit the form data to the server for processing.

```
<input type="submit" value="Submit">
```

28. Label, Fieldset and legends

In HTML forms, the <label>, <fieldset>, and <legend> elements are important for improving accessibility and providing structure and organization to forms.

The <label> element is used to associate a text label with a form input element. This is important for accessibility because it allows screen readers to read the label when the input element receives focus. It also makes it easier for users to click on the label to activate the input element.

Here's an example of using the <label> element:

```
<label for="name">Name:</label>

<input type="text" id="name" name="name">
```

In this example, the for attribute of the <label> element is set to the id attribute of the input element. This associates the label with the input element.

The <fieldset> element is used to group related form input elements together. This is useful for providing visual organization and accessibility for users who may need to navigate through the form using the keyboard.

Here's an example of using the <fieldset> element:

```
<fieldset>

<legend>Contact Information</legend>

<label for="name">Name:</label>

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

<label for="email">Email:</label>

<input type="email" id="email" name="email">

<label for="phone">Phone:</label>

<input type="tel" id="phone" name="phone">

</fieldset>
```

In this example, the <fieldset> element groups the three form input elements related to contact information together. The <legend> element is used to provide a title for the fieldset.

The <legend> element is used to provide a title or caption for a <fieldset> element. This is important for accessibility because it helps screen reader users understand the purpose of the fieldset.

Here's an example of using the <legend> element:

```
<fieldset>

<legend>Shipping Information</legend>

...

</fieldset>
```

In this example, the <legend> element provides a title for the fieldset that indicates it's related to shipping information.

29.SEO tags

SEO (Search Engine Optimization) tags are HTML elements used to provide information to search engines about the content and structure of a web page. These tags can help search engines understand the relevance and importance of specific content on a page, and can also improve the visibility and ranking of a website in search engine results pages (SERPs).

Some common SEO tags include:

1. Title Tag: The title tag appears in the head section of an HTML document and provides a brief and accurate description of the content on the page. It is usually displayed as the clickable headline in search engine results.
2. Meta Description Tag: The meta description tag provides a brief summary of the content on a page, and can be seen under the title in search engine results. It should be a concise and compelling description that encourages users to click through to the page.
3. Header Tags (H1, H2, H3, etc.): Header tags are used to indicate the structure and hierarchy of the content on a page. The H1 tag is usually reserved for the main headline or title of the page, while H2 and H3 tags are used for subheadings and supporting content.
4. Image Alt Tags: Alt tags provide a text description of images on a page, and are used by search engines to understand the content of an image. Including relevant alt tags can improve the accessibility and SEO of a website.
5. Canonical Tag: The canonical tag is used to indicate the preferred URL for a page when multiple URLs contain identical or very similar content. This helps to avoid duplicate content issues and improves the accuracy of search engine results.
6. Robots Meta Tag: The robots meta tag is used to provide instructions to search engine crawlers on how to index and follow links on a page. It can be used to prevent search engines from indexing specific pages or sections of a site, or to instruct them to follow or no follow links on a page.
7. Schema Markup: Schema markup is a structured data vocabulary that can be added to a page to provide additional information about the content on the page. This can include information about events, products, reviews, and more, and can improve the visibility and appearance of a website in search engine results.

By using these and other SEO tags correctly and consistently, website owners can help search engines understand and rank their content more effectively, leading to increased visibility, traffic, and engagement.

30. Favicon

A favicon, short for "favorite icon," is a small icon or logo that appears in a web browser's tab, bookmarks, and other places to represent a website. It is usually a square image with a size of 16x16 pixels or 32x32 pixels and saved in the ICO file format.

Favicons are an important aspect of a website's branding and can help visitors easily recognize and remember a website. They also improve the user experience by making it easy for users to identify and switch between multiple tabs.

To add a favicon to a website, the ICO file needs to be saved in the root directory of the website and added to the HTML code using the following code:

```
<link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
```

It is also recommended to add the following code to the head section of the HTML code to ensure compatibility with different web browsers:

```
<link rel="icon" href="favicon.ico" type="image/x-icon">

<link rel="icon" type="image/png" sizes="32x32" href="favicon-32x32.png">

<link rel="icon" type="image/png" sizes="16x16" href="favicon-16x16.png">
```

These additional lines of code specify the favicon's size and type in different formats, ensuring that it appears correctly on different devices and browsers.

In addition to the standard favicon, some websites also use animated or dynamic favicons that change based on certain events or user interactions. These can be created using HTML, CSS, and JavaScript and add an extra layer of interactivity to a website's branding.

31. Style in head Tag

In HTML, the <head> tag is used to contain metadata about the document, such as the document title, links to external resources, and style information. The <style> tag is used to define styles for HTML elements.

The <style> tag is usually included in the <head> section of an HTML document. It is used to define CSS rules that can be applied to the HTML elements of the document. The syntax for the <style> tag is as follows:

```
<head>

<style>

/* CSS rules */

</style>

</head>
```

Within the <style> tag, you can write CSS code to style various HTML elements. For example, to set the background color of the body element to blue, you can write:

```
<head>

<style>

body {

background-color: blue;

}

</style>

</head>
```

The <style> tag can also be used to import an external CSS file. For example, if you have a file called styles.css in the same directory as your HTML file, you can include it in your document using the following code:

```
<head>

<link rel="stylesheet" type="text/css" href="styles.css">

</head>
```

This will link to the external CSS file and apply the styles to the HTML elements in your document.

32. Script and no script

In HTML, the <script> and <noscript> tags are used to specify JavaScript code and content to be displayed when JavaScript is disabled in the browser, respectively.

The <script> tag is used to define a client-side script that is executed when the page is loaded or when an event occurs. It can be used to add interactivity and functionality to the HTML document. The src attribute can be used to specify an external script file to be loaded.

For example, to add a script to an HTML document, you can use the following code:

```
<html>
<head>
<title>My Web Page</title>
<script>
function sayHello() {
alert("Hello World!");
}
</script>
</head>
<body>
<button onclick="sayHello()">Click me</button>
</body>
</html>
```

In this example, a function named sayHello() is defined in the <script> tag in the <head> section of the document. The function is then called when the button is clicked using the onclick attribute.

The <noscript> tag is used to provide alternative content to be displayed when the browser does not support JavaScript or when it is disabled. This can be useful for providing non-JavaScript users with an alternative means of accessing content or functionality.

For example, to display a message when JavaScript is disabled, you can use the following code:

```
<html>
<head>
<title>My Web Page</title>
<script>
document.write("JavaScript is enabled.");
</script>
<noscript>
<p>JavaScript is disabled. Please enable JavaScript to view this page.</p>
</noscript>
</head>
<body>
<!-- HTML content here -->
</body>
</html>
```

In this example, the <script> tag writes "JavaScript is enabled." to the document. If JavaScript is disabled, the <noscript> tag is used to display a message to the user instead.

33. Viewport tag

The viewport meta tag is an important tag in HTML that helps control the layout of a web page on mobile devices. When a website is opened on a mobile device, the device's screen size and resolution are smaller than that of a desktop computer. This can result in the website appearing zoomed in or out, with some parts of the page being cut off or requiring horizontal scrolling to view the entire page. The viewport meta tag helps to solve this problem by providing the browser with information about how the page should be scaled to fit the screen of the device.

To use the viewport meta tag, it must be placed in the head section of the HTML document, between the head tags. The tag includes several attributes that control the behavior of the page on mobile devices. Here is an example of how to use the viewport meta tag:

```
<head>

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

</head>
```

In this example, the name attribute is set to "viewport" to indicate that this tag is used to control the viewport. The content attribute specifies two values: width=device-width sets the width of the viewport to the width of the device, and initial-scale=1.0 sets the initial zoom level to 100%, which means that the page will be displayed at its original size on the device.

There are other attributes that can be used with the viewport meta tag to further control the layout of the page on mobile devices. These include minimum-scale, maximum-scale, and user-scalable. Here is an example that includes all the available attributes:

```
<head>

<meta name="viewport" content="width=device-width, initial-scale=1.0, minimum-scale=1.0, maximum-scale=2.0, user-scalable=yes">

</head>
```

In this example, the minimum-scale attribute sets the minimum zoom level to 100%, which prevents the user from zooming out beyond the original size of the page. The maximum-scale attribute sets the maximum zoom level to 200%, which allows the user to zoom in up to twice the original size of the page. The user-scalable attribute controls whether the user is allowed to zoom in or out of the page by pinching or spreading their fingers on the screen. By setting this attribute to "yes", the user is allowed to zoom in and out of the page.

In conclusion, the viewport meta tag is an important tool for creating mobile-friendly web pages that are optimized for viewing on mobile devices. By using this tag and its attributes, web developers can control the layout and scaling of their pages to ensure that they are accessible and easy to use on any device.

34. Right to left

A Right-to-Left (RTL) website is a website that is designed to cater to languages that are read from right to left. This includes languages such as Arabic, Hebrew, and Urdu, among others.

To create an RTL website, there are a few key considerations to keep in mind:

1. Text direction: The text on the website should be set to start from the right side of the page and flow to the left.
2. Alignment: All elements on the website should be aligned to the right side of the page. This includes headings, images, and form fields.
3. Navigation: The navigation menu should also be aligned to the right side of the page.
4. Fonts: Select fonts that are suitable for the language and can be displayed correctly in all browsers.
5. Images and graphics: Images and graphics should also be aligned to the right side of the page, and any text should be written in the appropriate language.

To create an RTL website, you can use a combination of CSS and HTML to ensure that all elements on the page are displayed correctly. You can also use tools such as Bootstrap, a popular CSS framework that includes RTL support, to make the process easier.

In addition, you can use tools such as Google's "Lingua Franca" to translate your content into the appropriate language and ensure that it is displayed correctly on the page.

Overall, creating an RTL website requires careful consideration of the language, layout, and design elements. By taking the time to plan and implement these elements correctly, you can create a website that is both functional and visually appealing for your target audience.