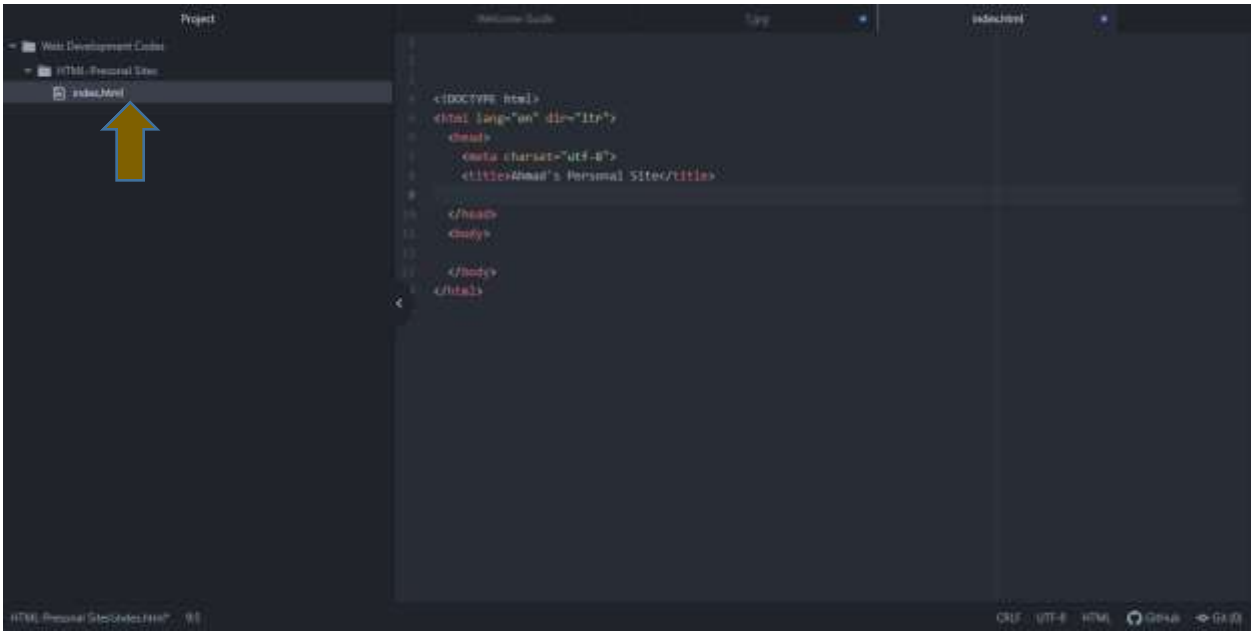
How to run html code:

Right click on file and copy full path or press **Ctrl+Shift+C**



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Ahmad's Personal Site</title>

  </head>
  <body>

  </body>
</html>
```

Then paste this path into the browser like chrome

<header>

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/header

The **<header>** HTML element represents introductory content, typically a group of introductory or navigational aids. It may contain some heading elements but also a logo, a search form, an author name, and other elements.

Usage notes

The <header> element is not sectioning content and therefore does not introduce a new section in the outline. That said, a <header> element is intended to usually contain the surrounding section's heading (an h1–h6 element), but this is **not** required.

Try it

```
<header class="page-header">

  <h1>Cute Puppies Express!</h1>

</header>


<main>
```

```
   <p>I love beagles <em>so</em> much! Like, really, a lot. They’re adorable and their ears are so, so snuggly soft!</p>

</main>
```

Code 2:

```
<h1>  The Adventures of Sherlock Holmes<h1>

<h4>by<h4>

  <h3>Arthur Conan Doyle<h3>
```

HTML Headings

HTML headings are titles or subtitles that you want to display on a webpage.

Example

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

HTML Headings

HTML headings are defined with the <h1> to <h6> tags.

<h1> defines the most important heading. <h6> defines the least important heading.

Example

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
```

```
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

Headings Are Important

Search engines use the headings to index the structure and content of your web pages.

Users often skim a 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.

Note: Use HTML headings for headings only. Don't use headings to make text **BIG** or **bold**.

<br>: The Line Break element

https://devdocs.io/html/element/br
The <br> HTML element produces a line break in text (carriage-return). It is useful for writing a poem or an address, where the division of lines is significant.
It doesn’t require open and close tags. So called self-closing tag.

```
<h1>  The Adventures of Sherlock Holmes</h1>

<br>

<h4>by<h4>

  <br>

  <h3>Arthur Conan Doyle<h3>
```

<hr>: The Thematic Break (Horizontal Rule) element

The <hr> HTML element represents a thematic break between paragraph-level elements: for example, a change of scene in a story, or a shift of topic within a section.

Attributes

This element's attributes include the global attributes.

align Deprecated

Sets the alignment of the rule on the page. If no value is specified, the default value is left.

color Non-Standard

Sets the color of the rule through color name or hexadecimal value.

noshade Deprecated

Sets the rule to have no shading.

size Deprecated

Sets the height, in pixels, of the rule.

width Deprecated

Sets the length of the rule on the page through a pixel or percentage value.

<center>: The Centered Text element

The <center> HTML element is a block-level element that displays its block-level or inline contents centered horizontally within its containing element. The container is usually, but isn't required to be, <body>.
This tag has been deprecated in HTML 4 (and XHTML 1) in favor of the CSS text-align property, which can be applied to the <div> element or to an individual <p>. For centering blocks, use other CSS properties like margin-left and margin-right and set them to auto (or set margin to 0 auto).

DOM interface

This element implements the HTMLElement interface.

Example 1

```
<center>This text will be centered.

<p>So will this paragraph.</p></center>
```

Example 2:

```
<center>

<hr>

<h1>The Adventures of <br> Sherlock Holmes<h1>

  <br>

  <h4>by<h4>

<br>

  <h2>by Arthur Conan Doyle<h2>

  <hr>
```

```
 <centrer>
```

Example

```
<!--Headings-->
<h1>This is level 1 heading created using h1 tag</h1>
 <h1>Heading level 1<h1>
   <h2>Heading level 2<h2>
     <h3>Heading level 3<h3>
       <h4>Heading level 4<h4>
         <h5>Heading level 5<h5>
           <h6>Heading level 6<h6>
<!-- //Opening tag
        Comments
               //Closing tags   -->
```

<title>: The Document Title element

The <title> HTML element defines the document's title that is shown in a browser's title bar or a page's tab. It only contains text; tags within the element are ignored.

Attributes

This element only includes the global attributes.

Usage notes

The <title> element is always used within a page's <head> block.

HTML <meta> charset Attribute

Definition and Usage

The charset attribute specifies the character encoding for the HTML document.

The HTML5 specification encourages web developers to use the UTF-8 character set, which covers almost all of the characters and symbols in the world

Attribute Values

Value	Description
character_set	Specifies the character encoding for the HTML document. The HTML5 specification encourages web developers to use the UTF-8 character set!

```
<head>
  <meta charset="UTF-8">
</head>
```

Unicode Table:

Unicode is a computing standard for the consistent encoding symbols. It was created in 1991. It’s just a table, which shows glyphs position to encoding system.

https://unicode-table.com/en/

Copy any symbol (utf) then paste in in title and refresh browser.

```
<!DOCTYPE html>

<html lang="en" dir="ltr">

  <head>

    <meta charset="utf-8">

    <title>💖Ahmad's Personal Site</title>


  </head>

  <body>


  </body>

</html>
```

<!—W3Schools learn about given things-->

```
<head>
  <meta charset="UTF-8">
  <meta name="description" content="Free Web tutorials">
  <meta name="keywords" content="HTML, CSS, JavaScript">
  <meta name="author" content="John Doe">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

Learn more about Unicode

https://www.joelonsoftware.com/2019/12/05/so-hows-that-retirement-thing-going-anyway/

Mdn website

https://developer.mozilla.org/en-US/ (View page source to know about page backend)

Ctrl+F to find

Viewport meta tag

Search meta on mdn.

This article describes how to use the "viewport" <meta> tag to control the viewport's size and shape.

Background

The browser's viewport is the area of the window in which web content can be seen. This is often not the same size as the rendered page, in which case the browser provides scrollbars for the user to scroll around and access all the content.

Some mobile devices and other narrow screens render pages in a virtual window or viewport, which is usually wider than the screen, and then shrink the rendered result down so it can all be seen at once. Users can then pan and zoom to see different areas of the page. For example, if a mobile screen has a width of 640px, pages might be rendered with a virtual viewport of 980px, and then it will be shrunk down to fit into the 640px space.

This is done because not all pages are optimized for mobile and break (or at least look bad) when rendered at a small viewport width. This virtual viewport is a way to make non-mobile-optimized sites in general look better on narrow screen devices.

However, this mechanism is not so good for pages that are optimized for narrow screens using media queries — if the virtual viewport is 980px for example, media queries that kick in at 640px or 480px or less will never be used, limiting the effectiveness of such responsive design techniques. The viewport meta tag mitigates this problem of virtual viewport on narrow screen devices.

HTML <head> Tag

Definition and Usage

The <head> element is a container for metadata (data about data) and is placed between the <html> tag and the <body> tag.

Metadata is data about the HTML document. Metadata is not displayed.

Metadata typically define the document title, character set, styles, scripts, and other meta information.

The following elements can go inside the <head> element:

- <title> (required in every HTML document)
- <style>
- <base>
- <link>
- <meta>
- <script>
- <noscript>

HTML Paragraph Tag <p>Paragraph is here</p>

The <p> tag defines a paragraph of text. It is a block-level element and always starts on a new line. Before and after each paragraph, browsers add margin automatically. You can modify the margins using the CSS margin property .

If you need to just to move text to a new line use the <br> tag.

Syntax

The <p> tag comes in pairs. The content is written between the opening (<p>) and closing (</p>) tags. If the closing tag is omitted, it is considered that the end of the paragraph matches with the start of the next block-level element.

Spaces between the opening <p> tag and its content are ignored by the browser. In order to set an indent, use the CSS text-indent property .

The <p> tag cannot contain tables and other block-level elements.

Example of the HTML <p> tag:

```
<!DOCTYPE html>

<html>

  <head>

    <title>Title of the document</title>

  </head>

  <body>

    <p>This is a paragraph</p>

  </body>

</html>
```

Italicize the text

The **<em>** HTML element marks text that has stress emphasis. The <em> element can be nested, with each level of nesting indicating a greater degree of emphasis.

1. **<em> </em>**
2. **<i> </i>**

These two tags are used to italicize the text.

The `<em>` element is for words that have a stressed emphasis compared to surrounding text, which is often limited to a word or words of a sentence and affects the meaning of the sentence itself.

Typically this element is displayed in italic type. However, it should not be used to apply italic styling; use the CSS `font-style` property for that purpose. Use the `<cite>` element to mark the title of a work (book, play, song, etc.). Use the `<i>` element to mark text that is in an alternate tone or mood, which covers many common situations for italics such as scientific names or words in other languages. Use the `<strong>` element to mark text that has greater importance than surrounding text.

<em> Vs. <i>

- <i> tag only italicize the text but <em> tag stressed the text inside it.
- It’s good to always use emphasis tag <em> instead of italic tag <i>.
- <em> contains more information and <i> is just a style

Some developers may be confused by how multiple elements seemingly produce similar visual results. `<em>` and `<i>` are a common example, since they both italicize text. What's the difference? Which should you use?

By default, the visual result is the same. However, the semantic meaning is different. The `<em>` element represents stress emphasis of its contents, while the `<i>` element represents text that is set off from the normal prose, such a foreign word, fictional character thoughts, or when the text refers to the definition of a word instead of representing its semantic meaning. (The title of a work, such as the name of a book or movie, should use `<cite>`.)

This means the right one to use depends on the situation. Neither is for purely decorative purposes, that's what CSS styling is for.

An example for `<em>` could be: "Just *do* it already!", or: "We *had* to do something about it". A person or software reading the text would pronounce the words in italics with an emphasis, using verbal stress.

An example for `<i>` could be: "The *Queen Mary* sailed last night". Here, there is no added emphasis or importance on the word "Queen Mary". It is merely indicated that the object in question is not a queen named Mary, but a ship named *Queen Mary*. Another example for `<i>` could be: "The word *the* is an article".

<b>: The Bring Attention To element

Bold Tag

The **`<b>`** HTML element is used to draw the reader's attention to the element's contents, which are not otherwise granted special importance. This was formerly known as the Boldface element, and most browsers still draw the text in boldface. However, you should not use `<b>` for styling text; instead, you should use the CSS `font-weight` property to create boldface text, or the `<strong>` element to indicate that text is of special importance.

Attributes

This element only includes the global attributes.

Usage notes

- Use the `<b>` for cases like keywords in a summary, product names in a review, or other spans of text whose typical presentation would be boldfaced (but not including any special importance).
- Do not confuse the `<b>` element with the `<strong>`, `<em>`, or `<mark>` elements. The `<strong>` element represents text of certain *importance*, `<em>` puts some emphasis on the text and the `<mark>` element represents text of certain *relevance*. The `<b>` element doesn't convey such special semantic information; use it only when no others fit.
- Similarly, do not mark titles and headings using the `<b>` element. For this purpose, use the `<h1>` to `<h6>` tags. Further, stylesheets can change the default style of these elements, with the result that they are not *necessarily* displayed in bold.
- It is a good practice to use the `class` attribute on the `<b>` element in order to convey additional semantic information as needed (for example `<b class="lead">` for the first sentence in a paragraph). This makes it easier to manage multiple use cases of `<b>` if your stylistic needs change, without the need to change all of its uses in the HTML.
- Historically, the `<b>` element was meant to make text boldface. Styling information has been deprecated since HTML4, so the meaning of the `<b>` element has been changed.
- If there is no semantic purpose to using the `<b>` element, you should use the CSS `font-weight` property with the value `"bold"` instead in order to make text bold.

```
<!DOCTYPE html>

<html lang="en" dir="ltr">

<head>

  <meta charset="utf-8">

  <title>Ahmad's Personal Site</title>

</head>

<body>

  <h1>Muhammad Ahmad</h1>

<p><i>Web designer of WordPress and <b>HTML, CSS and JS</b>.<i></p>
```

```
</body>

</html>
```

<strong>: The Strong Importance element

The **`<strong>`** HTML element indicates that its contents have strong importance, seriousness, or urgency. Browsers typically render the contents in bold type.

<b> vs. <strong>

It is often confusing to new developers why there are so many ways to express the same thing on a rendered website. `<b>` and `<strong>` are perhaps one of the most common sources of confusion, causing developers to ask "Should I use `<b>` or `<strong>`? Don't they both do the same thing?"

Not exactly. The `<strong>` element is for content that is of greater importance, while the `<b>` element is used to draw attention to text without indicating that it's more important.

It may help to realize that both are valid and semantic elements in HTML5 and that it's a coincidence that they both have the same default styling (boldface) in most browsers (although some older browsers actually underline `<strong>`). Each element is meant to be used in certain types of scenarios, and if you want to bold text for decoration, you should instead actually use the CSS `font-weight` property.

The intended meaning or purpose of the enclosed text should be what determines which element you use. Communicating meaning is what semantics are all about.

<em> vs. <strong>

Adding to the confusion is the fact that while HTML 4 defined `<strong>` as indicating a stronger emphasis, HTML 5 defines `<strong>` as representing "strong importance for its contents." This is an important distinction to make.

While `<em>` is used to change the meaning of a sentence as spoken emphasis does ("I *love* carrots" vs. "I love *carrots*"), `<strong>` is used to give portions of a sentence added importance (e.g., "**Warning!** This is **very dangerous.**") Both `<strong>` and `<em>` can be nested to increase the relative degree of importance or stress emphasis, respectively.

Examples

Basic example

```
<p>Before proceeding, <strong>make sure you put on your safety
goggles</strong>.</p>
```

Copy to Clipboard

The resulting output:

Before proceeding, **make sure you put on your safety goggles**.

<ul>: The Unordered List element

The **`<ul>`** HTML element represents an **unordered list** of items, typically rendered as a bulleted list.

```
<ul>
  <li>Milk</li>
  <li>Cheese
    <ul>
      <li>Blue cheese</li>
      <li>Feta</li>
    </ul>
  </li>
</ul>
```

Usage notes

- The `<ul>` element is for grouping a collection of items that do not have a numerical ordering, and their order in the list is meaningless. Typically, unordered-list items are displayed with a bullet, which can be of several forms, like a dot, a circle, or a square. The bullet style is not defined in the HTML description of the page, but in its associated CSS, using the `list-style-type` property.
- The `<ul>` and `<ol>` elements may be nested as deeply as desired. Moreover, the nested lists may alternate between `<ol>` and `<ul>` without restriction.
- The `<ol>` and `<ul>` elements both represent a list of items. They differ in that, with the `<ol>` element, the order is meaningful. To determine which one to use, try changing the order of the list items; if the meaning is changed, the `<ol>` element should be used, otherwise you can use `<ul>`.

Examples

Simple example

```
<ul>
  <li>first item</li>
  <li>second item</li>
  <li>third item</li>
</ul>
```

Copy to Clipboard

The above HTML will output:

- first item
- second item
- third item

<ol>: The Ordered List element

The `<ol>` HTML element represents an ordered list of items — typically rendered as a numbered list.

Attributes

This element also accepts the global attributes.

`reversed`

This Boolean attribute specifies that the list's items are in reverse order. Items will be numbered from high to low.

`start`

An integer to start counting from for the list items. Always an Arabic numeral (1, 2, 3, etc.), even when the numbering `type` is letters or Roman numerals. For example, to start numbering elements from the letter "d" or the Roman numeral "iv," use `start="4"`.

`type`

Sets the numbering type:

- `a` for lowercase letters
- `A` for uppercase letters
- `i` for lowercase Roman numerals
- `I` for uppercase Roman numerals
- `1` for numbers (default)

The specified type is used for the entire list unless a different `type` attribute is used on an enclosed `<li>` element.

Note: Unless the type of the list number matters (like legal or technical documents where items are referenced by their number/letter), use the CSS `list-style-type` property instead.

Usage notes

Typically, ordered list items display with a preceding marker, such as a number or letter.

The `<ol>` and `<ul>` elements may nest as deeply as desired, alternating between `<ol>` and `<ul>` however you like.

The `<ol>` and `<ul>` elements both represent a list of items. The difference is with the `<ol>` element, the order is meaningful. For example:

- Steps in a recipe
- Turn-by-turn directions
- The list of ingredients in decreasing proportion on nutrition information labels

To determine which list to use, try changing the order of the list items; if the meaning changes, use the `<ol>` element — otherwise you can use `<ul>`.

Examples

Simple example

```
<ol>
  <li>Fee</li>
  <li>Fi</li>
  <li>Fo</li>
  <li>Fum</li>
</ol>
```

Copy to Clipboard

The above HTML will output:

Using Roman Numeral type

```
<ol type="i">
  <li>Introduction</li>
  <li>List of Grievances</li>
  <li>Conclusion</li>
</ol>
```

Copy to Clipboard

The above HTML will output:

Using the start attribute

```
<p>Finishing places of contestants not in the winners' circle:</p>

<ol start="4">
  <li>Speedwalk Stu</li>
  <li>Saunterin' Sam</li>
  <li>Slowpoke Rodriguez</li>
</ol>
```

Copy to Clipboard

The above HTML will output:

Nesting lists

```
<ol>
  <li>first item</li>
  <li>second item  <!-- closing </li> tag not here! -->
    <ol>
      <li>second item first subitem</li>
      <li>second item second subitem</li>
      <li>second item third subitem</li>
    </ol>
  </li>            <!-- Here's the closing </li> tag -->
  <li>third item</li>
</ol>
```

Copy to Clipboard

The above HTML will output:

Unordered list inside ordered list

```
<ol>
  <li>first item</li>
  <li>second item  <!-- closing </li> tag not here! -->
    <ul>
      <li>second item first subitem</li>
      <li>second item second subitem</li>
      <li>second item third subitem</li>
    </ul>
  </li>            <!-- Here's the closing </li> tag -->
  <li>third item</li>
</ol>
```

Copy to Clipboard

The above HTML will output:

<li>: The List Item element

The **`<li>`** HTML element is used to represent an item in a list. It must be contained in a parent element: an ordered list (`<ol>`), an unordered list (`<ul>`), or a menu (`<menu>`). In menus and unordered lists, list items are usually displayed using bullet points. In ordered lists, they are usually displayed with an ascending counter on the left, such as a number or letter.

<a>: The Anchor element

The **<a>** HTML element (or *anchor* element), with its `href` attribute, creates a hyperlink to web pages, files, email addresses, locations in the same page, or anything else a URL can address.

Content within each `<a>` **should** indicate the link's destination. If the `href` attribute is present, pressing the enter key while focused on the `<a>` element will activate it.

```
<p>You can reach Michael at:</p>

<ul>

  <li><a href="https://example.com">Website</a></li>

  <li><a href="mailto:m.bluth@example.com">Email</a></li>

  <li><a href="tel:+123456789">Phone</a></li>

</ul>
```

Note: If we want to open link of website or video etc. then it will be written as:

HTML from

```
 <a href="https://youtu.be/l5jUaLBXEg8" target="#"> Angela Yu</a>
```

It means that target is blank

`href`

The URL that the hyperlink points to. Links are not restricted to HTTP-based URLs — they can use any URL scheme supported by browsers:

- Sections of a page with fragment URLs
- Pieces of media files with media fragments
- Telephone numbers with `tel:` URLs
- Email addresses with `mailto:` URLs
- While web browsers may not support other URL schemes, web sites can with `registerProtocolHandler()`

`hreflang`

Hints at the human language of the linked URL. No built-in functionality. Allowed values are the same as the global `lang` attribute.

`download`

Causes the browser to treat the linked URL as a download. Can be used with or without a value:

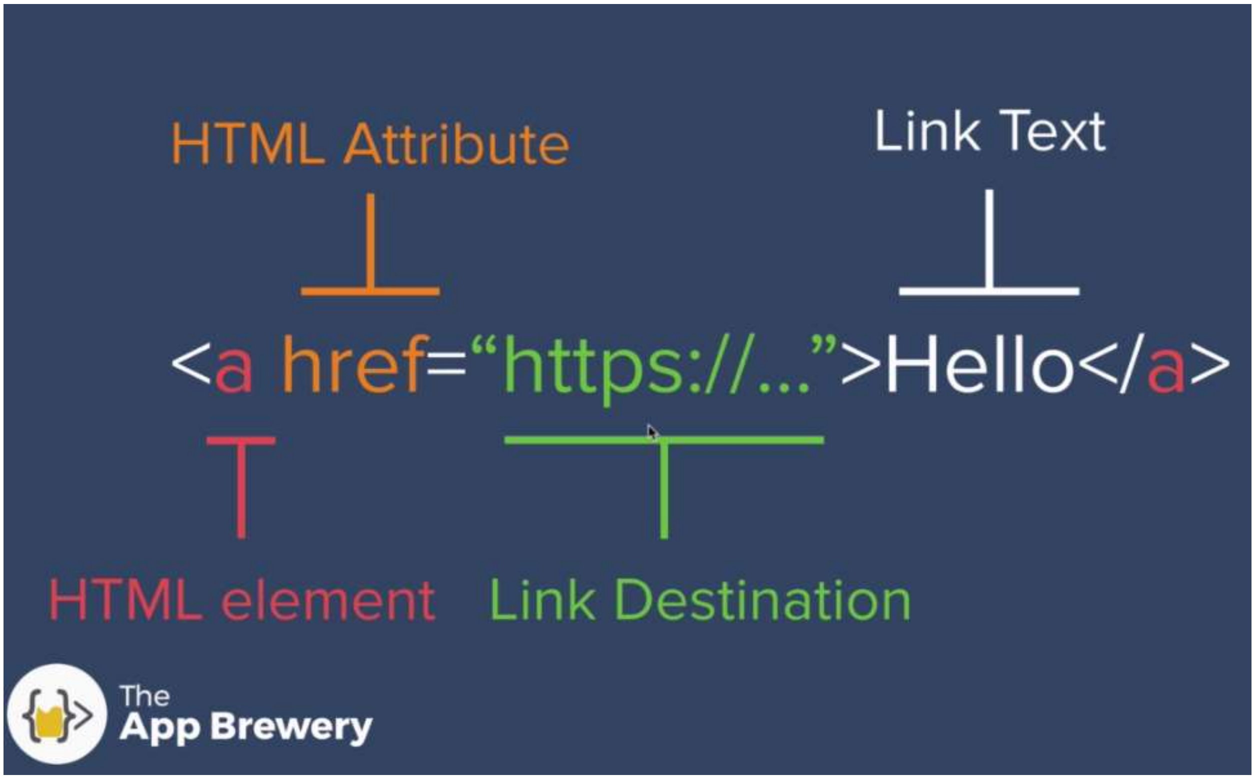
- Without a value, the browser will suggest a filename/extension, generated from various sources:
  - The `Content-Disposition` HTTP header
  - The final segment in the URL path
  - The media type (from the `Content-Type` header, the start of a `data:` URL, or `Blob.type` for a `blob:` URL)
- Defining a value suggests it as the filename. `/` and `\` characters are converted to underscores (`_`). Filesystems may forbid other characters in filenames, so browsers will adjust the suggested name if necessary.

Note:

- `download` only works for same-origin URLs, or the `blob:` and `data:` schemes.
- How browsers treat downloads varies by browser, user settings, and other factors. The user may be prompted before a download starts, or the file may be saved automatically, or it may open automatically, either in an external application or in the browser itself.
- If the `Content-Disposition` header has different information from the `download` attribute, resulting behavior may differ:
  - If the header specifies a `filename`, it takes priority over a filename specified in the `download` attribute.
  - If the header specifies a disposition of `inline`, Chrome and Firefox prioritize the attribute and treat it as a download. Old Firefox versions (before 82) prioritize the header and will display the content inline.

`ping`

A space-separated list of URLs. When the link is followed, the browser will send `POST` requests with the body `PING` to the URLs. Typically for tracking.



<img>: The Image Embed element

The **<img>** HTML element embeds an image into the document.

The above example shows usage of the `<img>` element:

- The `src` attribute is **required**, and contains the path to the image you want to embed.
- The `alt` attribute holds a text description of the image, which isn't mandatory but is **incredibly useful** for accessibility — screen readers read this description out to their users so they know what the image means. Alt text is also displayed on the page if the image can't be loaded for some reason: for example, network errors, content blocking, or linkrot.

There are many other attributes to achieve various purposes:

- Referrer/CORS control for security and privacy: see `crossorigin` and `referrerpolicy`.
- Use both `width` and `height` to set the intrinsic size of the image, allowing it to take up space before it loads, to mitigate content layout shifts.
- Responsive image hints with `sizes` and `srcset` (see also the `<picture>` element and our Responsive images tutorial).

Supported image formats

The HTML standard doesn't list what image formats to support, so user agents may support different formats.

Note: The Image file type and format guide provides comprehensive information about image formats and their web browser support. This section is just a summary!

The image file formats that are most commonly used on the web are:

- APNG (Animated Portable Network Graphics) — Good choice for lossless animation sequences (GIF is less performant)
- AVIF (AV1 Image File Format) — Good choice for both images and animated images due to high performance.
- GIF (Graphics Interchange Format) — Good choice for *simple* images and animations.
- JPEG (Joint Photographic Expert Group image) — Good choice for lossy compression of still images (currently the most popular).
- PNG (Portable Network Graphics) — Good choice for lossy compression of still images (slightly better quality than JPEG).
- SVG (Scalable Vector Graphics) — Vector image format. Use for images that must be drawn accurately at different sizes.
- WebP (Web Picture format) — Excellent choice for both images and animated images

Formats like WebP and AVIF are recommended as they perform much better than PNG, JPEG, GIF for both still and animated images. WebP is widely supported while AVIF lacks support in Safari.

SVG remains the recommended format for images that must be drawn accurately at different sizes.

Image loading errors

If an error occurs while loading or rendering an image, and an `onerror` event handler has been set on the `error` event, that event handler will get called. This can happen in a number of situations, including:

- The `src` attribute is empty (`""`) or `null`.
- The `src` URL is the same as the URL of the page the user is currently on.
- The image is corrupted in some way that prevents it from being loaded.
- The image's metadata is corrupted in such a way that it's impossible to retrieve its dimensions, and no dimensions were specified in the `<img>` element's attributes.
- The image is in a format not supported by the user agent.

Attributes

This element includes the global attributes.

- `alt`
    - : Defines an alternative text description of the image.

How to add images in HTML?

- Insert tag **<img src="image location/image name.type" alt="Any text like my profile">**

How TO - Rounded Images

Learn how to create rounded and circular images with CSS.

How To Create Rounded Images

Step 1) Add HTML:

Example

```
<img src="img_avatar.png" alt="Avatar">
```

Step 2) Add CSS:

Use the `border-radius` property to add rounded corners to an image. 50% will make the image circular:

Example

```
img {
  border-radius: 50%;
}
```

<u>: The Unarticulated Annotation (Underline) element

The **`<u>`** HTML element represents a span of inline text which should be rendered in a way that indicates that it has a non-textual annotation. This is rendered by default as a simple solid underline, but may be altered using CSS.

```
<!DOCTYPE html>

<html lang="en" dir="ltr">

  <head>

    <meta charset="utf-8">

    <title>Contact Me</title>

  </head>

  <body>

    <h1>My Contact Details</h1>
```

```
<p>My Fictional Address: <strong>Main City Kasur</strong></p>
<p>Mobile: <b>03225932678</b></p>
<p>Email: <strong><u>ahmad481988@gmail.com</u></strong></p>

  </body>
</html>
```