

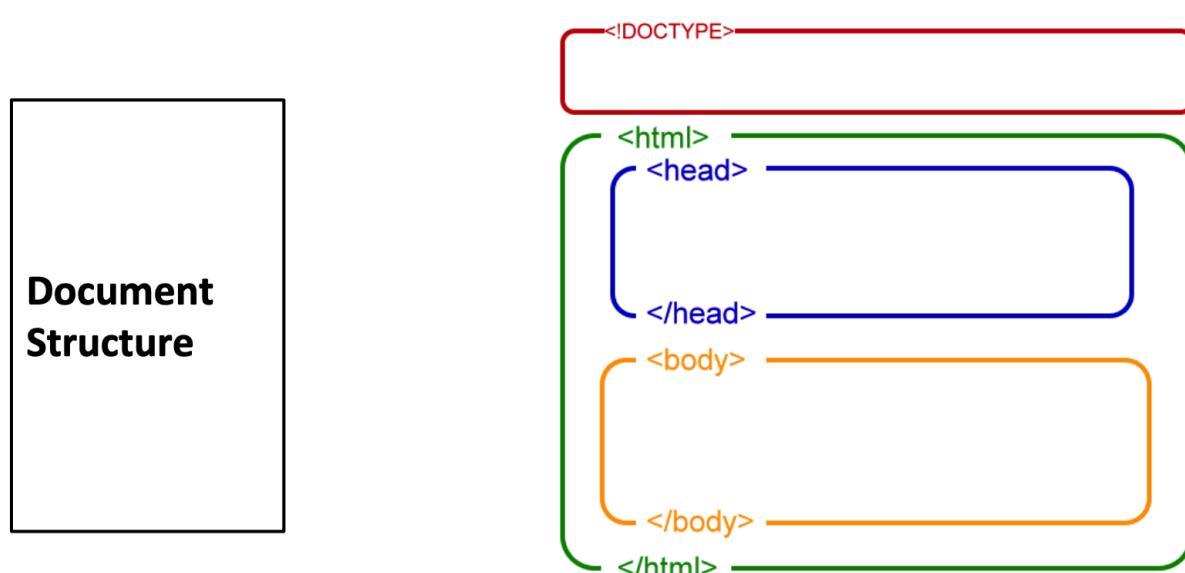
# HTML :

- Stands for **Hyper text markup language**. tells the broser how to display the content.
- It is a **markup language**.
- **markup language :**
  - It is a computer language that is used to annotate text to provide information about its structure and formatting.
  - Typically composed of **Markup-Tags** and **Content**.
  - **Markup-Tag** : is used to identify the begining and end of a piece of content, and they provide information about how that content should be displayed or processed.
  - **Content** : is the text, image or other data that is being marked up.

## HTML Layout Structure :

<|>esto

### HTML Layout Structure



•

```
<html>
  <head>

  </head>
  <body>
    All visible things are in body.
    <h1>Hello World!!</h1>
  </body>
</html>
```

# Html Basics :

---

- Elements
  - opening tag + content + closing tag = element.
  - the tags tell the information about the content.
  - eg :- <b> Hello!! </b>
- Tags
  - the opening and closing tags.
  - every tag has certain meaning to it.
  - eg :- <b></b> stands for bold.
- Attributes
  - They provide additional info about the contents of the elements.
  - 2 part : name & value. name = value
  - <a href="abc.com"> ABC </a>
- Semantic HTML
  - In programming, Semantics = "meaning of a piece of code".
  - eg:- what effect does running that line of JS code have? or what purpose/role that HTML element have? (rather than what does it look like).
  - semantic HTML introduces meaning to the webpage rather than just presentation.
  - eg :-
    1. <p> : it indicates enclosed text is paragraph. both semantic and presentational. people know what para are and browser know how to display them.
    2. <em>Hey</em> will tell the screen readers to put more emphasis while pronouncing the content.
- Why should you care about semantics?
  - impacts SEO highly.
  - It tells the browser the meaning of page and its content.
  - It helps in communicating with search engines and client computers very well.
  - eg :-
    1. If suppose we create a list using CSS and Div Element --> it might get displayed as list on page but there could be some keyboard features or something that won't come with it as in case of using an original list element.
    2. <em>Hey</em> will tell the screen readers to put more emphasis while pronouncing the content.
- Use semantic tags correctly

## Use Semantic Tags Correctly

Some of the most commonly misused semantic tags include:

- **blockquote** - Some people use the <blockquote> tag for indenting text that is not a quotation.
  - **p** - Some web editors use <p>&nbsp;</p> (a non-breaking space contained in a paragraph) to add extra space between page elements, rather than defining actual paragraphs for the text of that page.
  - **ul** - Like blockquote, enclosing text inside a <ul> tag indents that text in most browsers.
  - **h1-h6** - The heading tags can be used to make fonts bigger and bolder, but if the text is not a heading, it should not be inside a heading tag.
- 

- What's the difference b/w <b> and <strong>, <i> and <em> ?
  - <b> : is a style.
  - <strong> : is a indication of how something should be understood.
  - <em> : same as <strong> just adding more emphasis/attention to this element.
- Some of the benefits from writing semantic markup

## Some of the benefits from writing semantic markup are as follows:

- Search engines will consider its contents as important keywords to influence the page's search rankings (see SEO)
  - Screen readers can use it as a signpost to help visually impaired users navigate a page
  - Finding blocks of meaningful code is significantly easier than searching though endless divs with or without semantic or namespaced classes
  - Suggests to the developer the type of data that will be populated
  - Semantic naming mirrors proper custom element/component naming
- 

## Deep dive into <HEAD> :

---

- Head is basically a container for **metadata**.

&lt;|&gt;esto

## HTML Head

is a container for metadata

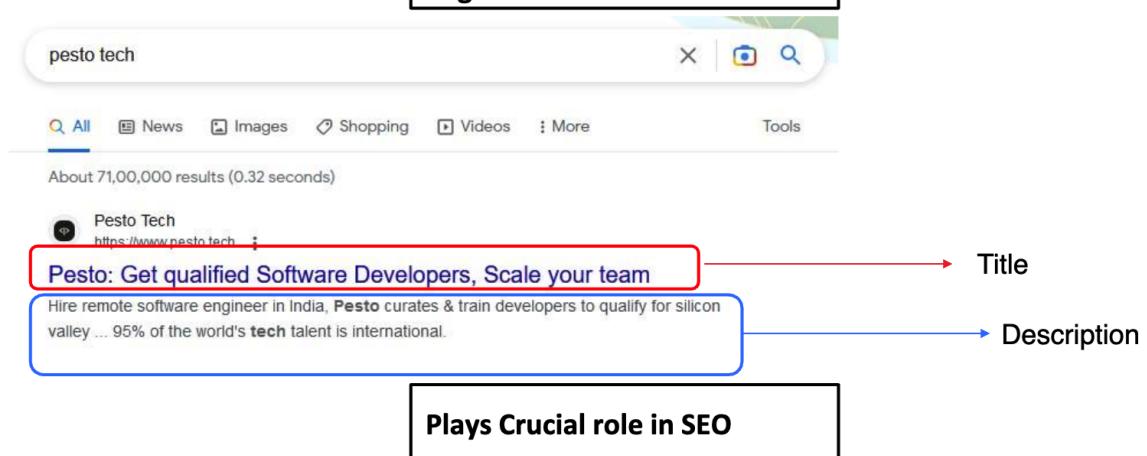


- 

&lt;|&gt;esto

## Meta Tags

Meta tags are guide to search engines



- 

- List of all meta tags: <https://www.metatags.org/all-meta-tags-overview/>

## 3 - ways to add CSS in your webpage :

### 1. inline :

- highest priority, but not recommended for big applications as it becomes difficult to maintain.
- `<h1 style="color: red">Hey there!!</h1>`

### 2. internal :

- recommended to add in the `<head></head>` section of html.

```
<style>color: red;</style>
```

### 3. external:

- o <link rel="stylesheet" type="text/css" href="style.css">
- o Can we insert a `style.scss` file in Html page using `<link>` tag?
  - No, you cannot directly insert a `.scss` file using a `<link>` tag in an HTML document.
  - The `<link>` tag is used to include external CSS files, not Sass (`.scss`) files. Browsers do not understand Sass syntax, as it requires preprocessing to be converted into standard CSS.
  - Sass is a preprocessor that extends CSS with features like variables, nesting, mixins, and more, which are not part of the standard CSS specification.
  - To use Sass in your projects, you need to follow the steps mentioned earlier.
    1. Write your styles in a `.scss` file.
    2. Compile the `.scss` file into a regular `.css` file using the Sass compiler.
      - command : `sass styles.scss styles.css`
    3. Link the compiled `.css` file in your HTML using the `<link>` tag.

## Javascript in html : `async` vs `defer`

---

- NOTE : without using `async` or `defer` JS loading will not happen in parallel with HTML parsing.
- NOTE : Using `async` and `defer` together is not recommended, but if used together JS will give higher priority to `async`.

### 1. Adding javascript in `<head>` :

- o If the script size is heavy, then load time increases when added in head.
- o Hence not recommended for heavy scripts.
- o the loading of JS happens when the execution engine reaches that line and at that moment the parsing is paused.
- o parsing resumes when both the loading and execution of JS code is complete.

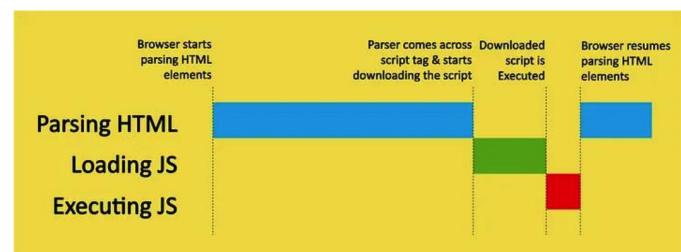
`<presto`

### HTML Head- Script execution

**Async & defer**

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Script in the head tag</title>
    <script src="index.js"></script>
  </head>
  <body>
    <!-- All the HTML content here -->
  </body>
</html>
```

If the script size is heavy, then load time increases



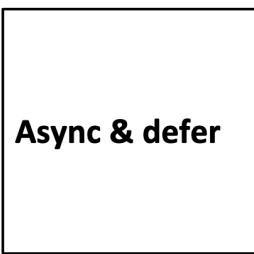
- o

### 2. Adding javascript at bottom of `<body>` :

- o we add javascript at bottom of body only if we dont want the script to load at the first paint.
- o the loading of JS happens after the parsing of entire HTML is complete.

&lt;desto

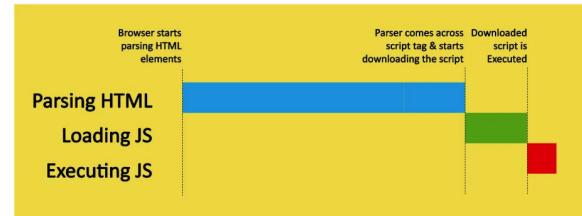
## HTML Head



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>Script at the end of page</title>
</head>
<body>

  <!-- All the HTML content here -->
  <script src="index.js"></script>
</body>
</html>
```

Add script at  
the bottom if it  
not required  
for first paint



- o

### 3. Using **async** :

- o It loads the JS parallel to the HTML parsing.
- o If executes the JS only when loading is completed, at that time the parsing is paused until the JS execution is finished.
- o we use **async** if our script contains some code that is necessary for our page to load at the very start.
- o eg :- if we want to load a image that comes from database at the main page of our website/app. like on amazon website the images show up the moment the js completes its loading.
- o eg :- sometimes websites like flipkat/amazon just load a lower resolution or pixalated image at the first page load and when we spend sometime on that page the it replaces the those images with high resolution ones. in such cases as well we will make the low resolution images as **async** so that it shows up as soon as it loads and since it is of low resolution it will load very fast if loading happens parallelly.

&lt;|&gt;esto

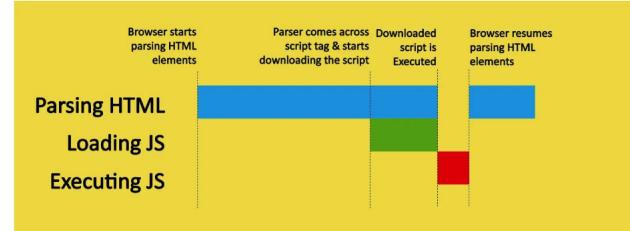
## HTML Head

### Async & defer

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Script with async attribute</title>
    <script async src="index.js"></script>
  </head>
  <body>

    <!-- All the HTML content here -->

  </body>
</html>
```



o

#### 4. Using **defer** :

- o It loads the JS parallel to the HTML parsing.
- o It executes the JS only when loading and the parsing of HTML is completed or finished.
- o eg :- in amazon website we don't want the menu click event to actually load at the very beginnig but after some images have loaded so we add the **defer** keyword for such cases.

&lt;|&gt;esto

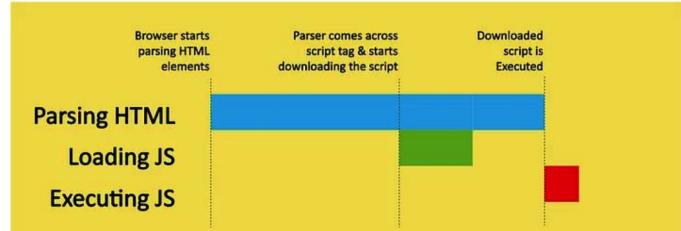
## HTML Head

### Async & defer

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Script with defer attribute</title>
    <script defer src="index.js"></script>
  </head>
  <body>

    <!-- All the HTML content here -->

  </body>
</html>
```



o

**link tag in HTML** `<a href="..." target="..."> content </a>`:

- `<a target="_blank|_self|_parent|_top|framename">`

Value	Description
_blank	Opens the linked document in a new window or tab.
_self	Opens the linked document in the same frame as it was clicked ( <b>this is default</b> ).
_parent	Opens the linked document in the parent frame ( <b>immediate parent</b> ).
_top	Opens the linked document in the full body of the window ( <b>ie. the topmost parent</b> ).
framename	Opens the linked document in the named iframe.

### Links- Mostly used tag

The **target** attribute specifies where to open the linked document.



```
<a href="https://pesto.tech" target="_blank">Opens link in a new window/tab</a>
<a href="https://pesto.tech" target="_self">Opens link in the same window/tab</a>
<a href="https://pesto.tech" target="_parent">Opens link in the parent frame or window</a>
<a href="https://pesto.tech" target="_top">Opens link in the full body of the window</a>
```

## SEO friendly tags :

1. <address> :
2. <blockquote> :
3. <abbr> :
4. <q> :
5. <cite> :
6. <bdo> :

## Symbols and Emojis :

```
<!DOCTYPE html>
<html lang="en">
  <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>My Website</title>
</head>
<body>
    <h1>My First Emoji</h1>
    <p>Happy face emoji is : &#128512;</p>
</body>
</html>

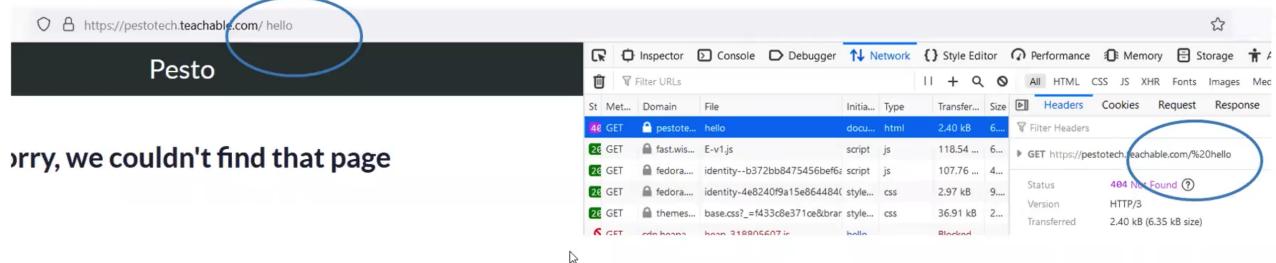
```

## URL encoding :

- in below image we can see no-ascii char " " has been replaced by "%20" ascii value.

### URL Encoding

**URL can be sent using ASCII char only**  
**URL encoding replaces non-ASCII characters with a "%" followed by hexadecimal digits.**



## XHTML :

- Html displays content even if it has errors.
- XHTML is more stricter and formatted.
- Use cases of HTML : Compliant web applications.
- Use case of XHTML : Legacy & Error-prone applications like financing systems.

### HTML

Tags aren't extensible

Tags are not case sensitive.

Possible to leave off any ending tag like </body>

Overlapping tags

### XHTML

Tags are extensible

Only lowercase tags are allowed

Tags must appear in pairs

## HTML-4 vs HTML-5 :

