HTML WORLD

Learning HTML5

https://github.com/Prakash9596/HTML_World.git

https://prakash9596.github.io/HTML_World/

1. (Ctrl+K Ctrl+F) To Format the selected text
2. (Ctrl+K Ctrl+T) To Change the theme
3. (Ctrl+a Ctrl+/) To comment all the lines in visual Studio
4. (Ctrl+ [&& Ctrl+]) To move selected text left/right
5. (Ctrl+ alt+ up/down arrow) To use mutiple cursor, click anywhere to come back to one cursor

--> git commands:

1. git clone https://github.com/Prakash9596/HTML_World.git (1)
2. git status (2)
3. git add . or git add filename (3)
4. git status (4)
5. git restore --staged filename (jo file add kr diye hain staging area me usko wapus lane ke liye)
6. git status
7. git add filename (again add kr rhe staging area me)
8. git status
9. git diff (to show difference between local and staging area)
10. git commit -m "Commiting HTML Codes" (5)
11. git push (6)

12. git log (commit id dekhne ke liye)
13. git checkout -b new_branch(new_branch create krke usme switch kr jayega)
14. git checkout branch_name (to switch to any other branch)
15. git branch (to check branch available locally)
16. git fetch (will tell us the difference b/t local and central repo)
17. git merge (to merge the change)
18. git pull (to sync local with central repo --> this is comnination of fetch and merge)

--> merge conflicts command:

What is IDE?

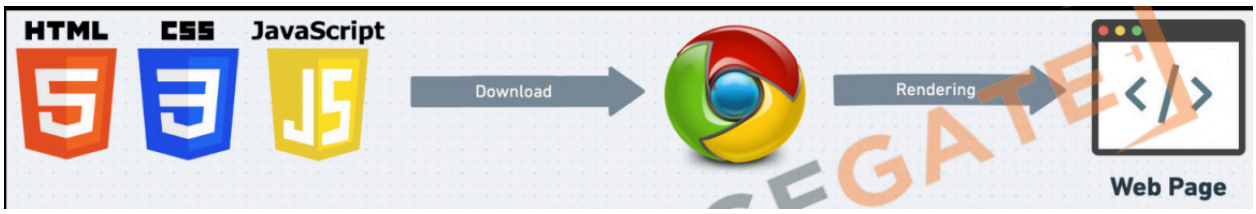
1. IDE stands for Integrated Development Environment.
2. Software suite that consolidates basic tools required for software development.
3. Central hub for coding, finding problems, and testing.
4. Designed to improve developer efficiency.

Need of an IDE. (VS Code)

1. Streamlines development.
2. Increases productivity.
3. Simplifies complex tasks.
4. Offers a unified workspace.
5. IDE Features 1. Code Autocomplete 2. Syntax Highlighting 3. Version Control 4. Error Checking

VsCode Extensions → 1. Live Server 2. Prettier

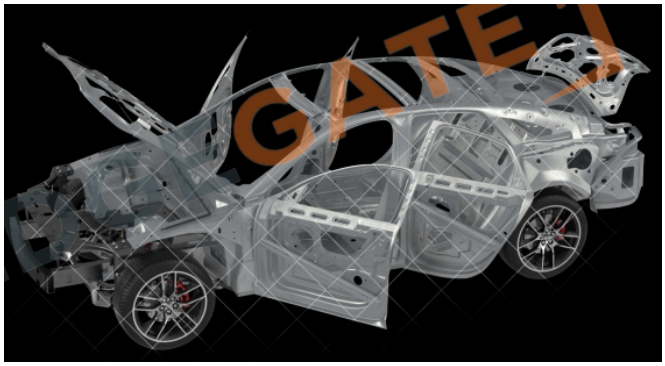
Role of Browser



1. Displays Web Page: Turns HTML code into what you see on screen.
2. User Clicks: Helps you interact with the web page.
3. Updates Content: Allows changes to the page using JavaScript.
4. Loads Files: Gets HTML, images, etc., from the server.

HTML (Hypertext Markup Language)

1. Structure: Sets up the layout.
2. Content: Adds text, images, links.
3. Tags: Uses elements like <p>, <a>
4. Hierarchy: Organizes elements in a tree.



CSS (Cascading Style Sheets)

1. Style: Sets the look and feel.
2. Colors & Fonts: Customizes text and background.
3. Layout: Controls position and size.
4. Selectors: Targets specific HTML elements.

JS (Java Script)

1. JavaScript has nothing to do with Java
2. Actions: Enables interactivity.
3. Updates: Alters page without reloading.
4. Events: Responds to user actions.
5. Data: Fetches and sends info to server.

Chapter-1

File Extension:

HTML
1. Most commonly used.
2. Works across all browsers.
3. Widely recognized and supported.
4. Typically saved as .html.

HTM
1. Less commonly used.
2. Originated for compatibility with older systems.
3. Works the same as .html.
4. Typically saved as .htm.

Importance of index.html
1. Default name of a website's homepage.
2. First page users see when visiting a website
3. Important for SEO (Search Engine Optimization)
4. Provides uniform starting point across servers
5. Serves as fallback when no file is specified in URL

What are Tags
1. Elements that are used to create a website are called HTML Tags.
2. Tags can contain content or other HTML tags.
3. Define elements like text, images, links

Using Emmet ! to generate code

1. Type ! and wait for suggestions.(boilerplate code)

MDN Documentation

1. Visit **https://developer.mozilla.org/**
2. Official resource for HTML
3. Offers comprehensive guides and tutorials
4. Includes examples for real-world use
5. Updated with latest HTML features
6. Trusted by developers worldwide

Comments

1. Used to add notes in HTML code
2. Not displayed on the web page
3. Syntax: <!-- Comment Here -->
4. Helpful for code organization
5. Can be multi-line or single-line

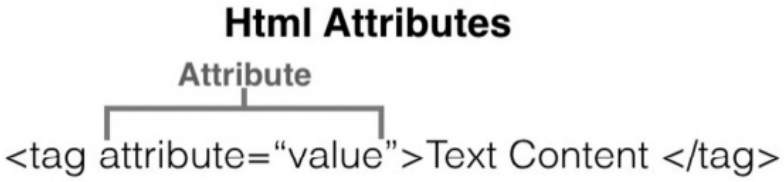
Case Sensitivity

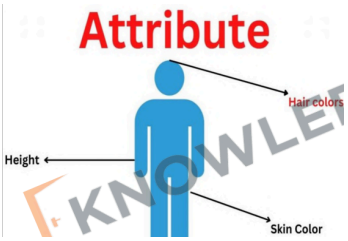
1. HTML is case-insensitive for tag names
2. Attribute names are also be case-insensitive
3. Best practice: use lowercase for consistency

Chapter-2

HTML Attributes:

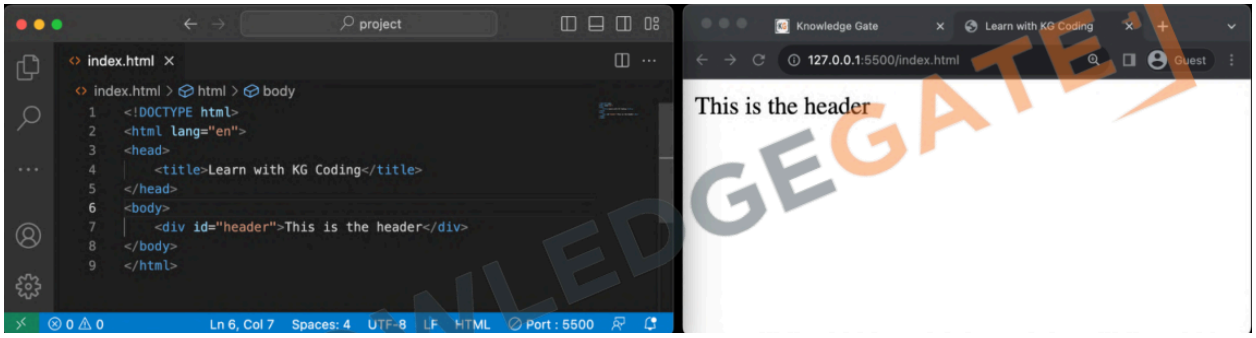
1. Provides additional information about elements
2. Placed within opening tags
3. Common examples: href, src, alt
4. Use name=value format
5. Can be single or multiple per element





id property:

- Unique Identifier: Each id should be unique within a page.
- Anchoring: Allows for direct links to sections using the #id syntax in URLs.
- CSS & JavaScript: Used for selecting elements for styling or scripting.



HTML Tags

Heading Tag:

1. Defines headings in a document
2. Ranges from <h1> to <h6>
3. <h1> is most important and <h6> is least.
4. Important for SEO 5. Helps in structuring content.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Headings</title>
</head>
<body>
    <!-- Heading tag (h1-h6) shows its (high-low) importance not their
(high-low) size-->
    <h1>heading 1</h1>
    <h2>heading 2</h2>
    <h3>heading 3</h3>
    <h4>heading 4</h4>
    <h5>heading 5</h5>
    <h6>heading 6</h6>
</body>
</html>
```

Paragraph Tag:

1. Used for defining paragraphs
2. Enclosed within and tags
3. Adds automatic spacing before and after
4. Text wraps to next line inside tag
5. Common in text-heavy content

```
<body>
    <!-- Use lorem50 to generate random paragraph of 50 characters-->
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Dolorum,
       dolor! </p>
</body>
```

**
 Tag:**

1.adds a line break within text
2.is empty, no closing tag needed
3.
 and </br> are both valid

```
<body>
    <p>Lorem ipsum, dolor<br> sit amet consectetur adipisicing elit. Iste,
    <br> mollitia! Facilis, eos possimus officia nemo eveniet vitae?
     <br><br><br>adipisci numquam.</p>
</body>
```

<hr> tag:

1. <hr> creates a horizontal rule or line
2. <hr> also empty, acts as a divider

```
<body>
    <h1>testing HR Tag</h1>
    <hr>
    <hr>
    <p>this is the testing of HR tag</p>
</body>
```

Image Tag:

1. Used to embed images
2. alt attribute for alternative text
3. Can be resized using width and height
4. Self-closing, doesn't require an end tag

```
<body>
    <!--height="150px"-->
    <image src="/images/Prakash.jpeg" alt="this is my passport size photo"
      width="150px"/>
    <hr>
    <image
src="https://th.bing.com/th/id/R.ab7861c8f52e2a71c5401f80b4869183?rik=72zA
1PYP9ptQgg&riu=http%3a%2f%2fapple.gop%2fmedia%2fuploads%2fzinnia%2fshiny-r
ed-apples.jpg&ehk=1OqkLNXKlIZEiFEiMfL%2bjWmShEoe5N412%2bXM1iioX5s%3d&risl=
&pid=ImgRaw&r=0" alt="this is an image of Apple" width="300px"/>
</body>
```

Video Tag:

1. Embeds video files on a page
2. Uses src attribute for video URL
3. Supports multiple formats like MP4, WebM
4. Allows for built-in controls via attributes like autoplay, controls, loop

```
<body>
    <h1>here is your video</h1>
    <video src="/assets/Timer.mp4" alt="countdown timer"
    height="300px" loop controls autoplay muted />
</body>
```

Anchor Tag:

1. Used for creating hyperlinks
2. Requires href attribute for URL
3. Can link to external sites or internal pages
4. Supports target attribute to control link behavior

```
<body>
    <!-- target="_blank" means open it in new tab-->
    <a href="https://www.youtube.com/">Youtube</a><br>
    <a href="https://www.google.com/" target="_blank">Google</a><br>
    <a href="https://www.instagram.com/">Instagram</a>
</body>
```

Bold/Italic/Underline/Strikethrough Tag:

1. <b> makes text bold
2. <i> makes text italic
3. <u> underlines text
4. <s> or <strike> applies strikethrough
5. Primarily used for text styling and emphasis

```
<body>
    <b>KnowledgeGate</b><hr>
    <i>Website</i><hr>
    <u>YouTube</u><hr>
    <s>course</s>
</body>
```

<pre> tag:

1. Preserves text formatting
2. Maintains whitespace and line breaks
3. Useful for displaying code
4. Enclosed within <pre> and</pre> tags

```
<body>
    <!-- to maintain whitespaces-->
    <p>
        Dear Prakash,

        Jha.

        Thank You
    </p>
    <pre>
     Dear Prakash,

     Jha.

     Thank You
    </pre>
</body>
```

Big/Small Tag:

1. <big> increases text size
2. <small> decreases text size
3. Less common due to CSS alternatives

```
<body>
    <big>This is big text</big><hr>
    <small>this is small text</small>
</body>
```

Superscript/Subscript Tag:

1.<sup> makes text superscript
2. <sub> makes text subscript
3. Used for mathematical equations, footnotes
4. Does not change font size, just position

```
<body>
    <p>(a+b)2 = a2 + b2 + 2ab </p>
    <p>(a+b)<sup>2</sup> = a<sup>2</sup> + b<sup>2</sup> + 2ab</p>
    <p> CH4 + O2 = H20 + CO2</p>
    <p>CH<sub>4</sub> + O<sub>2</sub> = H<sub>2</sub>0 + CO<sub>2</sub>
    </p>
</body>
```

Character Entity Reference

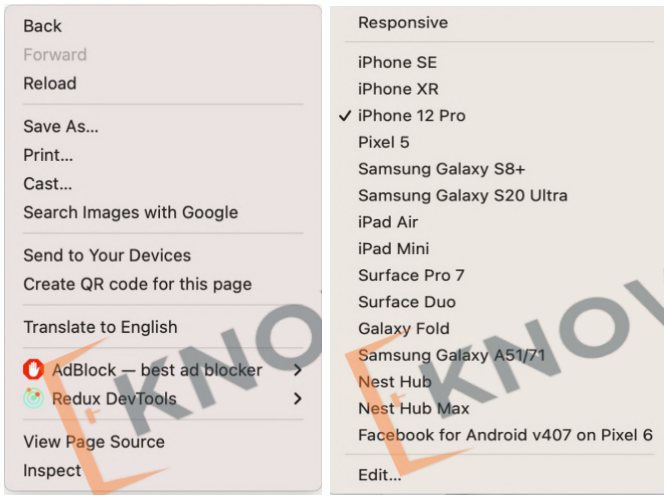
1. Used to display reserved or special characters
2. Syntax often starts with & and ends with ; (e.g., &)

		–	–	−	−	°	°	Δ	Δ	α	&al
€	€	—	—	±	±	º	º	Λ	Λ	β	&be
¢	¢	…	…	√	√	ª	ª	Θ	Θ	γ	&ga
£	£	§	§	∞	∞	¹	¹	Ξ	Ξ	δ	&de
¥	¥	¶	¶	∝	∝	²	²	Π	Π	ε	&ep
¤	¤	†	†	×	×	³	³	Σ	Σ	ζ	&ze
ƒ	ƒ	‡	‡	÷	÷	¼	¼	Φ	Φ	η	&et
©	©	¡	¡	~	∼	½	½	Ψ	Ψ	θ	&th
®	®	¿	¿	≈	≈	¾	¾	Ω	Ω	ι	&io
™	™	‰	‰	≅	≅	∴	∴	∇	∇	κ	&ka

Chapter-3

Inspect Element:

1. Allows real-time editing of HTML/CSS
2. Useful for debugging and testing
3. Shows element hierarchy and layout
4. Includes console for JavaScript
5. Highlights selected elements on page



Responsive Design : Different Screen Sizes

1. Adapts layout for different screen sizes
2. Flexible layouts
3. Optimizes images and assets
4. Enhances user experience on mobile and desktop

Live edit HTML / CSS / JavaScript:

1. Changes made are temporary
2. Affect only the current session
3. Not saved to the server
4. Reset upon page reload
5. Useful for testing, not permanent fixes

Like: If you change the question in your question paper that has no effect on the actual exam.

Validating WebPages: Using validator.w3.org

1. Ensures HTML adheres to standards
2. Minimizes cross-browser issues
3. Helps in achieving better SEO results
4. Easier to debug and maintain
5. Optimizes performance by reducing parsing errors

Chapter-4

Semantic/Non-Semantic Tags

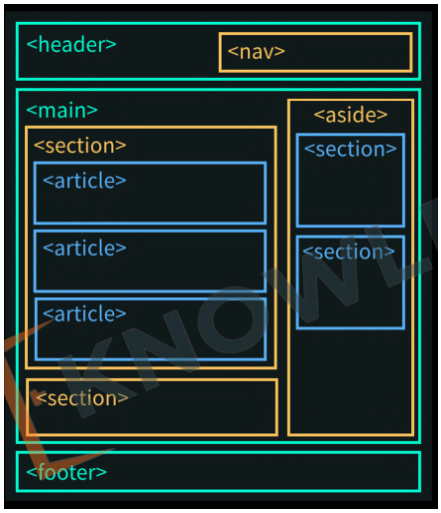
Semantic Tags

- Meaningful: Describe content.
- SEO: Good for search engines.
- Accessibility: Useful for screen readers.
- Examples: <header>, <footer>,<article>, <section>, <nav>.

Non-Semantic Tags

- Generic: No specific meaning.
- For Styling: Used for layout.
- No SEO: Not SEO-friendly.
- Examples: , , <i>, <b>

Body Tags:



Header Tag:

1. Purpose: Used to contain introductory content or navigation links.
2. Semantic: It's a semantic tag, providing meaning to the enclosed content.
3. Location: Commonly found at the top of web pages, but can also appear within or tags.
4. Multiple Instances: Can be used more than once on a page within different sections.

```
<body>
    <header>This is my header</header>
</body>
```

Main Tag:

1. Purpose: Encloses the primary content of a webpage.
2. Semantic: Adds meaning, indicating the main content area.
3. Unique: Should appear only once per page.
4. Accessibility: Helps screen readers identify key content.
5. Not for Sidebars: Excludes content repeated across multiple pages like site navigation or footer.

```
<body>
    <!-- There can be multiple instances of header(semantic)-->
    <!-- There can be only 1 instance of Main(semantic)-->
    <!-- There will be automatic 1 line space between header & main-->
    <header>header</header>
    <main>
        <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit.</p>
    </main>
</body>
```

Section Tag:

1. Purpose: Groups related content in a distinct section.
2. Semantic: Adds structure and meaning.
3. Headers: Often used with a heading <h1> to <h6> > to indicate section topic.
4. Nested: Can be nested within other <section> or <article> tags

```
<body>
    <header>header</header>
    <main>
        <section>
            <h2>Admissions</h2>
            admissions will start soon
        </section>
        <section>
            <h2>Placements</h2>
        </section>
    </main>
```

Article Tag:

1. Purpose: content should make sense even if taken out of this page.
2. Semantic: Provides contextual meaning.
3. Independence: Content should make sense even if taken out of the page context.
4. Multiple Instances: Can be used multiple times on the same page.

```
<body>
    <header>header</header>
    <main>
        <article>
            Lorem ipsum dolor sit amet consectetur, adipisicing elit. Voluptas commodi ut hic eligendi quisquam possimus alias.
        </article>
    </main>
</body>
```

Aside Tag:

1. Purpose: Contains sidebar or supplementary content.
2. Semantic: Indicates content tangentially related to the main content.
3. Not Crucial: Content is not essential to understanding the main content.
4. Examples: Could hold widgets, quotes, or ads.

```
<body>
   <!-- aside(semantic): supplementary content, not crucial, ex: ads-->
    <header>header</header>
    <main>
        <aside>
            This is not related.
        </aside>
    </main>
</body>
```

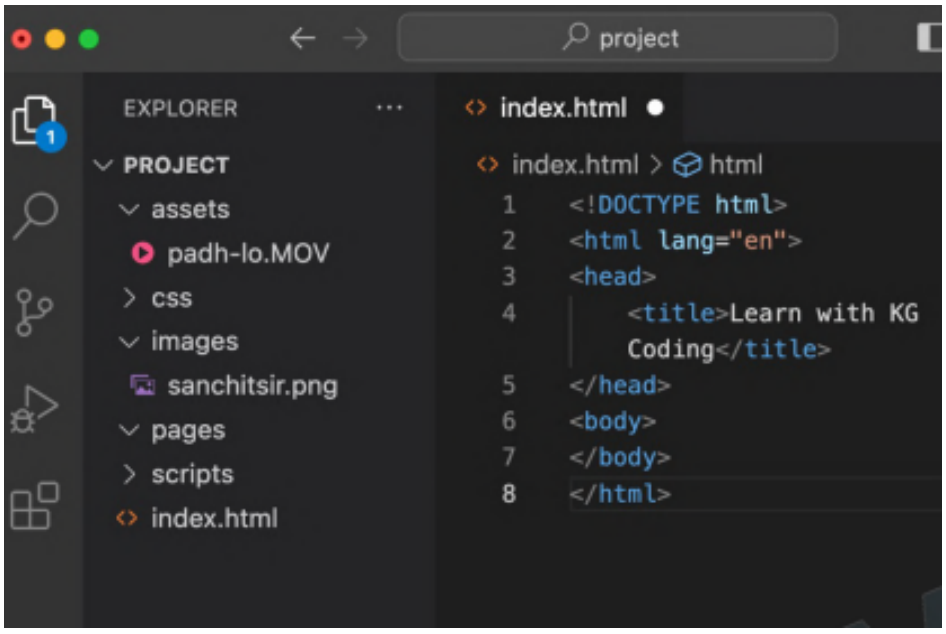
Footer Tag:

1. Purpose: For footer content like extra info or links.
2. Semantic: Provides meaning to enclosed content.
3. Location: Typically at the bottom of pages or sections.
4. Content: Includes copyrights, contact info, and social links.
5. Multiple Instances: Can be used more than once on a page.

```
<body>
    <header>header</header>
    <main>
        <p>Lorem ipsum </p>
    </main>
    <hr>
    <footer>
        - First link <br>
        - Second link <br>
        - Third link
    </footer>
</body>
```

Recommended Folder Structure:

1. Root Directory: Main folder containing all website files.
2. HTML Files: Store main .html files at the root level for easy access.
3. CSS Folder: Create a css/ folder for all Cascading Style Sheets.
4. JS Folder: Use a scripts/ folder for JavaScript files.
5. Images Folder: Store images in an images/ or images/ folder.
6. Assets: Other assets like fonts can go in an assets/ folder.
7. Sub-directories: For multi-page websites, use sub-folders to categorize content.



Navigation Tags:

1. Purpose: Encloses navigation links or menus.
2. Semantic: Signals that the content is meant for navigating the site.
3. Common Content: Usually contains lists <ul>, <ol>, of links <a>.
4. Accessibility: Aids screen readers in identifying site navigation.

```
<body>
    <!-- header and navigation should be at the top of the body-->
    <header> This is heading <br>
        <nav>
            <a href="#content">Content</a> <br>
            <a href="#about">About us</a> <br>
            <a href="#contact">contact us</a> <br>
            <a href="#feedback">Your Feedback</a>
        </nav>
    </header>
    <hr>
    <main>
        <p id = "content">Lorem ipsum </p><br>
        <p id="about">My name is Prakash Jha</p><br>
        <p id="contact">My Contact: 7903433712</p><br>
        <p id="feedback">Please provide your feedback here...</p>
    </main>
</body>
```

Block / Inline Elements:

Block Elements

- New Line: Start on a new line.
- Full Width: Take up all horizontal space.
- Styling: Can have margins and padding.
- Size: Width and height can be set.
- Examples: , <p>, <h1>, <ul>,

Inline Elements

- Flow: Stay in line with text. • Width: Just as wide as the content.
- No Break: No new line between elements.
- Limited Styling: Can't set size easily.
- Examples:, <a>, ,, <img>

```
<body>
    <p id="para">This is a paragraph1</p>
    <p id="para">This is a paragraph2</p>
    <a href="#para">Go To Paragraph1</a>
    <a href="#para">Go To Paragraph2</a>
</body>
```

Div Tags:

1. Purpose: Acts as a container for other HTML elements.
2. Non-Semantic: Doesn't provide inherent meaning to enclosed content.
3. Styling: Commonly used for layout and styling via CSS.
4. Flexibility: Highly versatile and can be customized using classes or IDs.

```
<body>
    <!-- iske andar kuchh v likh skte jaise paragraph, big, link-->
    <!-- commonly used for layout and styling via css-->
    <!-- act as a container for other HTML element-->
    <div>
        <p>this is a para</p>
        <big>this is big</big>
    </div>
    <div>
        <a href="#dummy">Dummy</a>
    </div>
</body>
```

Span Tags:

1. Purpose: Used for inline elements to style or manipulate a portion of text.
2. Non-Semantic: Doesn't add specific meaning to the enclosed text.
3. Styling: Commonly used for changing color, font, or adding effects via CSS.
4. Inline Nature: Doesn't break text flow or create a new block-level element.

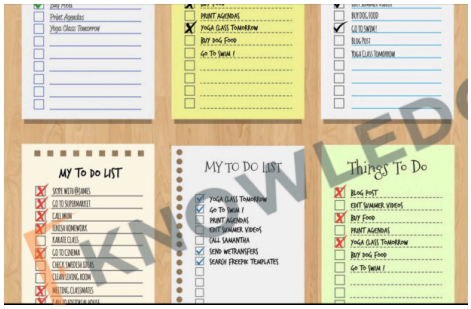
```
<body>
    <p>
        Lorem ipsum <span id="dolor">dolor</span> sit amet consectetur
    </p>
</body>
```

Chapter-5

List Tag:

Ordered Lists:

1. Purpose: Used for creating lists with items that have a specific order.
2. Default: Items are automatically numbered.
3. Nesting: Can be nested within other lists.



1. Wake up
2. Brush teeth
3. Take a shower
4. Have breakfast
5. Go to work

A. Apple
B. Banana
C. Cherry
D. Dragonfruit

a. Apple
b. Banana
c. Cherry
d. Dragonfruit

I. Apple
II. Banana
III. Cherry
IV. Dragonfruit

i. Apple
ii. Banana
iii. Cherry
iv. Dragonfruit

```
<body>
    <!-- Ordered Lists Types
        • Numeric: Default type, (1, 2, 3, ...)
                   Attribute: type="1"
        • Uppercase Letters: (A, B, C, ...)
                   Attribute: type="A"
        • Lowercase Letters: (a, b, c, ...)
                   Attribute: type="a"
        • Uppercase Roman: (I, II, III, ...)
                   Attribute: type="I"
        • Lowercase Roman: (i, ii, iii, ...)
                   Attribute: type="i"-->
    <ol type = "A">
        <li>HTML</li>
        <ol type="I">
            <li>Level a</li>
            <li>Level b</li>
        </ol>
        <li>C</li>
        <li>C++</li>
        <li>Java</li>
    </ol>
</body>
```

Unordered Lists:

1. Purpose: Used for lists where the order of items doesn't matter.
2. Default: Items are usually bulleted.
3. Nesting: Can be nested within other lists.


```
<body>
    <ul>
        <li>Cycling</li>
        <li>Cricket</li>
        <li>YouTube</li>
    </ul>
</body>
```

List, Tables & Forms:

Table Tag: <tr>, <th>, <td>

1. <tr> Table Row : Used to define a row in an HTML table.
2. <th> Table Header : Used for header cells within a row. Text is bold and centered by default.
3. <td> Table Data : This Holds the actual data.

Captions:

1. Purpose: Provides a title or description for a table.
2. Placement: Must be inserted immediately after the <table> opening tag.
3. Alignment: Centered above the table by default.
4. Accessibility: Helps screen readers understand the table's purpose.

Col Spans:

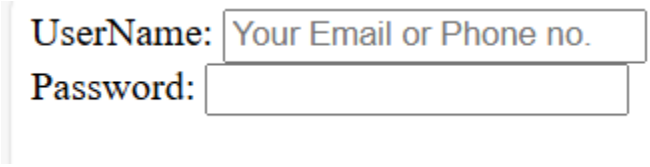
1. Attribute: Uses the colspan attribute in <td> or <th> tags.
2. Purpose: Allows a cell to span multiple columns horizontally.
3. Alignment: Takes the space of the specified number of columns.
4. Layout: Useful for combining cells to create complex table layouts.

```
<body>
    <!-- <table> To create table -->
    <table border="20">

        <caption>Technologies</caption>

        <tr>
            <th colspan="2">Tech I am good at</th>
        </tr>

        <tr>
            <th>Language</th>
            <th>Competency</th>
            <th>Years</th>
        </tr>

        <tr>
            <td>C</td>
            <td>5/10</td>
            <td>10 years</td>
        </tr>

        <tr>
            <td>Java</td>
            <td>8/10</td>
            <td>9 years</td>
        </tr>

        <tr>
            <td colspan="3"> Hello Prakash Jha</td>
        </tr>

    </table>
</body>
```

Forms:

1. Purpose: Used within a <form> element to collect user input.
2. Self-Closing: The tag is self-closing; doesn't require a closing tag.
3. Attributes: Common attributes are name, value, placeholder, and required.

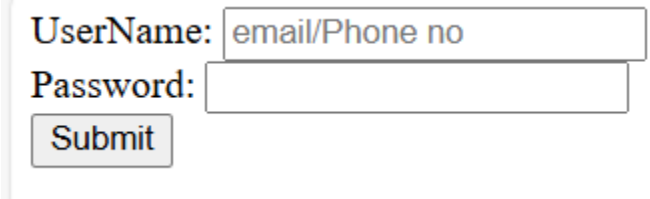
```
<form>
    UserName: <input type="text" placeholder="Your Email or Phone no." required="true"> <br>
    Password: <input type="password">
</form>
```



Action attribute:

1. Purpose: Specifies the URL to which the form data should be sent when submitted.
2. Default: If not specified, the form will be submitted to the current page's URL.
3. Server-Side: Usually points to a server-side script (like PHP, Python, etc.) that processes the form data.

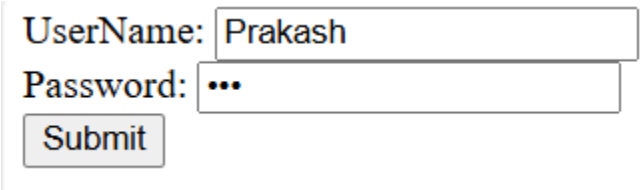
```
<!-- action: to move all these data to some function where we are collecting the data -->
<form action="/collecting-data.java">
    UserName: <input type="text" placeholder="email/Phone no" required="true"> <br>
    Password: <input type="password"> <br>
              <input type="submit">
</form>
```



Name and Value property:

- `name` Property: ○ ID for Data: Identifies form elements when submitting. ○ Unique: Should be unique to each element for clarity.
- `value` Property: ○ Default Data: Sets initial value for input elements. ○ Sent to Server: This is the data sent when form is submitted.

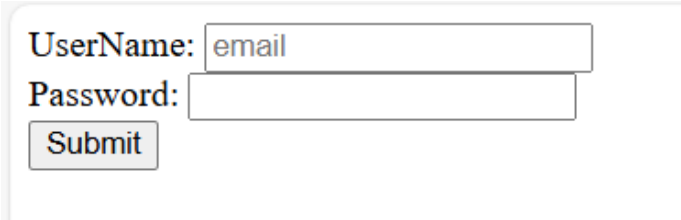
```
<form action="/collecting-data.java">
    UserName: <input type="text" placeholder="email/Phone no"
required="true" name="UserName" value="Prakash"> <br>
    Password: <input type="password" name="Password" value="xyz"> <br>
              <input type="submit">  </form>
```



Label Tag:

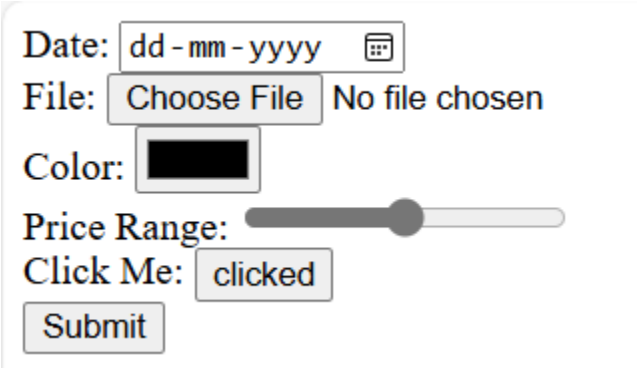
- Purpose: Adds a text description to form elements.
- for Attribute: Connects the label to a specific form element using the element's id.
- Accessibility: Makes the form more accessible.
- Readability: Enhances form readability and usability.

```
<body>
    <!-- label se hm linkage provide kr rhe, ki konsa input kiske liye h
(best practice)and either click on UserName or box its same -->
    <form action="/contact-me">
        <label for="email">UserName: </label>
        <input type="text" id="email" placeholder="email" required="true">
        <label for="user_password">Password: </label>
        <input id="user_password" type="password"> <br>
        <input type="submit">
    </form>
</body>
```

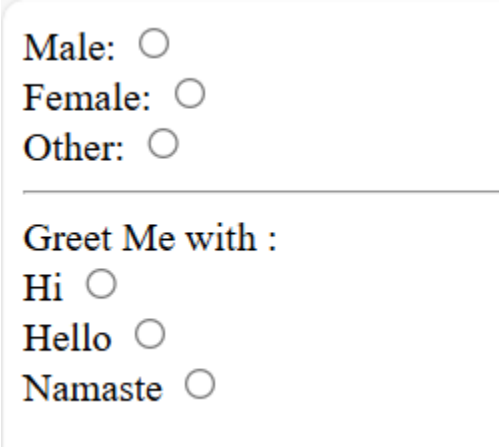


Input type: Date/File/Color/Range/Button/Submit

```
<body>
    <form action="/contact-me">
        Date: <input type="date"> <br>
        File: <input type="file"> <br>
        Color: <input type="color"> <br>
        Price Range: <input type="range"> <br>
        Click Me: <input type="button" value="clicked"> <br>
        <input type="submit">
    </form>
</body>
```



Radio:



```
<body>
    <!-- radio: For single choice-->
    <!-- name will be same for all (single choice hai n isliye)-->
    <form>
        <label for="male">Male:</label>
        <input type="radio" id="male" name="gender" value="male"> <br>

        <label for="female">Female:</label>
        <input type="radio" id="female"name="gender" value="female"> <br>

        <label for="other">Other:</label>
        <input type="radio" id="other" name="gender" value="other"> <br>

        <hr>
        <!-- multiple select ho rha, kyunki name alag alag rakh diye-->
        <p2> Greet Me with :</p2> <br>
        Hi      <input type="radio"  name="greeting1" value="Hi"> <br>
        Hello   <input type="radio"  name="greeting2" value="Hello"> <br>
        Namaste <input type="radio"  name="greeting3" value="Namaste">
<br>
    </form>
</body>
```

Checkbox:

```
<body>
    <!-- yahan bhi name same h but values multiple select ho skte hain
kyunki type checkbox hai-->
    <form>
        <label for="c">C</label>
        <input type="checkbox" name="lang" id="c" value="c"> <br>
        <label for="java">Java</label>
        <input type="checkbox" name="lang" id="java" value="java"><br>
        <label for="python">Python</label>
        <input type="checkbox" name="lang" id="python" value="python">
    </form>
</body>
```

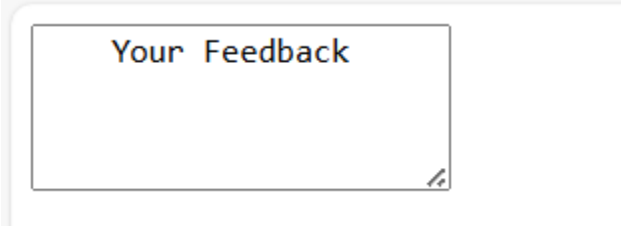
C ☐
Java ☐
Python ☐

Select:

```
<body>
    <form>
        <label for="country"> Select your Country</label>
        <select name="country" id="country">
            <option value="India">India</option>
            <option value="US">US</option>
            <option value="UK">UK</option>
            <option value="Other">Other</option>
        </select>
    </form>
</body>
```

TextArea:

```
<body>
    <form>
        <textarea name="Feedback" id="Feedback" cols="20" rows="4">
Your Feedback</textarea>
    </form>
</body>
```



iFrame Tag:

1. Embedded Content: Allows you to embed another webpage or multimedia content within a webpage.
2. src Attribute: Specifies the URL of the content to be embedded.
3. Dimensions: Width and height can be set using width and height attributes

```
<body>
    <!-- kisi aur ke website ko apne webpage pe use krne ke liye-->
    <iframe width="300" height="200"
src="https://www.youtube.com/embed/EbMtJ_Rl_fA?si=Sqru4bRDoinrBHLL"
title="YouTube video player" frameborder="0" allow="accelerometer;
autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture;
web-share" allowfullscreen></iframe>

    <iframe
src="https://en.wikipedia.org/wiki/Sinking_of_SS_Princess_Alice"
width="300" height="300"></iframe>
</body>
```

Github Pages & CodeSpace

What is Version Control

1. Definition: A system to track changes in files over time.
2. Types: Centralized (like SVN) and Distributed (like Git).
3. Purpose: Helps in teamwork and fixes mistakes.
4. Snapshots: Each 'commit' saves a file version.
5. Branching: Lets you work on different tasks separately.
6. Merge: Combines changes from different people.
7. Undo: Easy to revert to older file versions.

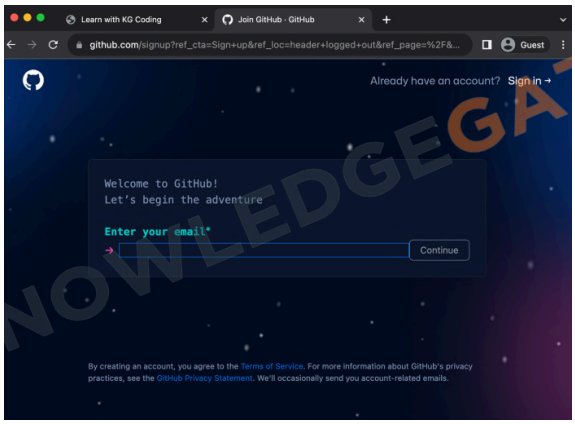
What is Git?

- Definition: A software tool that tracks changes in code, enabling collaboration and version control.
- Commit: Records a snapshot of file changes.
- Branch: Allows separate paths of development.
- Merge: Combines changes from different branches.

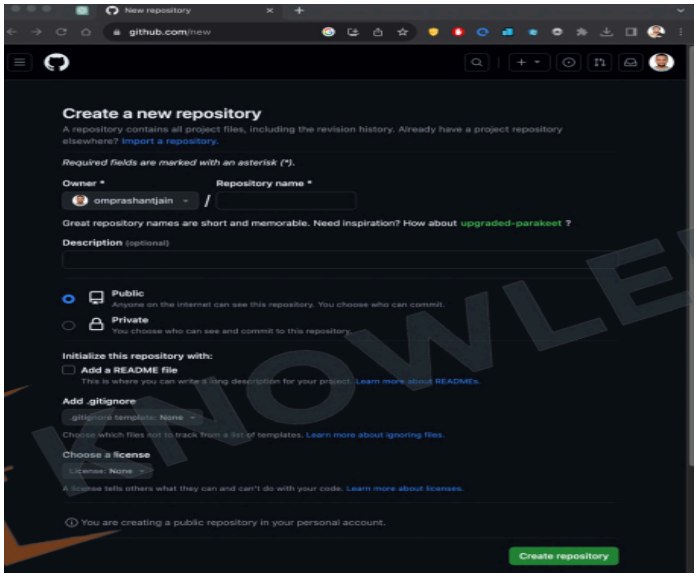
What is GitHub?

- Definition: A web service for hosting and collaborating on Git repositories.
- Fork: Creates a personal copy of another user's repository.
- Pull Request: A way to propose changes to existing code.
- Issues: Used for tracking bugs and feature ideas.

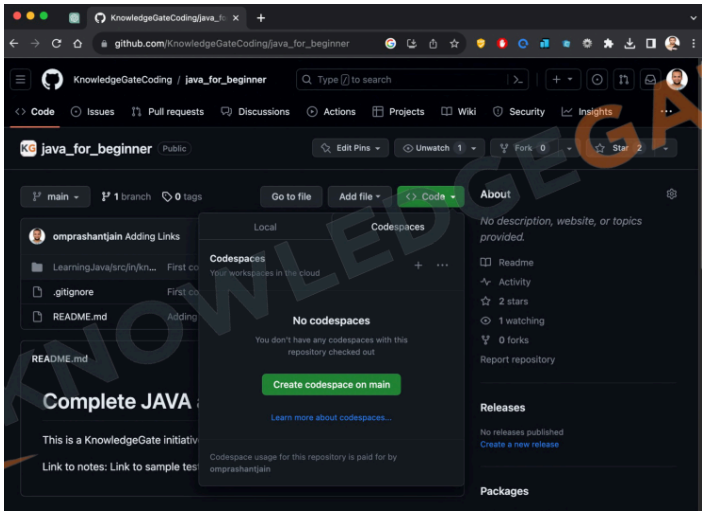
Account Creation:



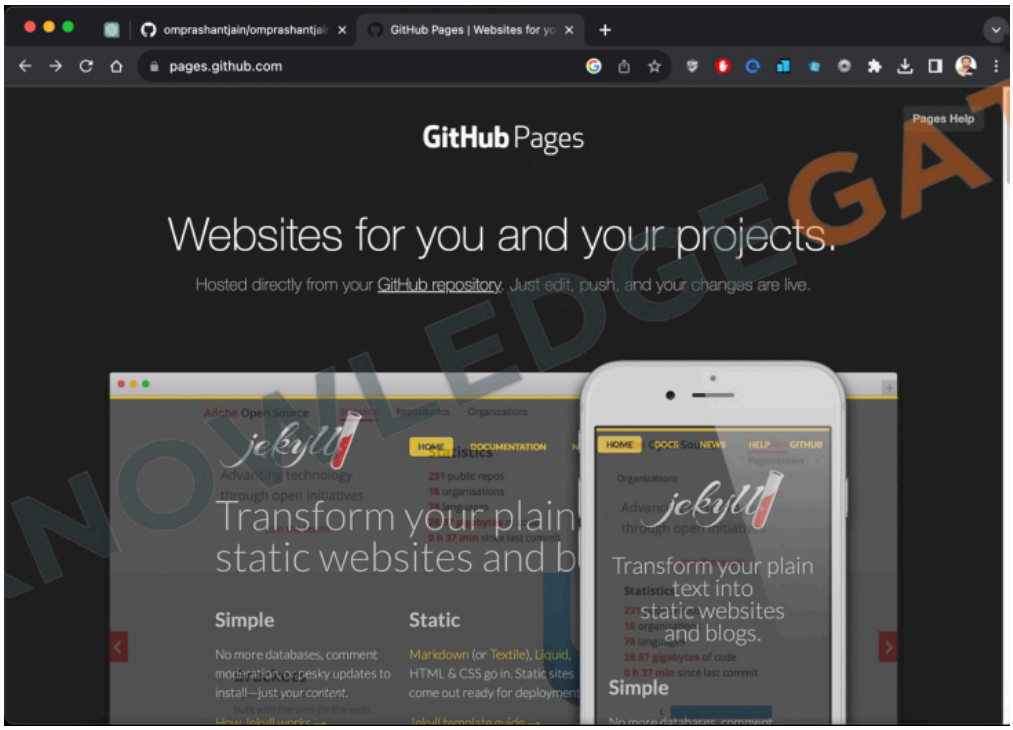
Creating a Repo:



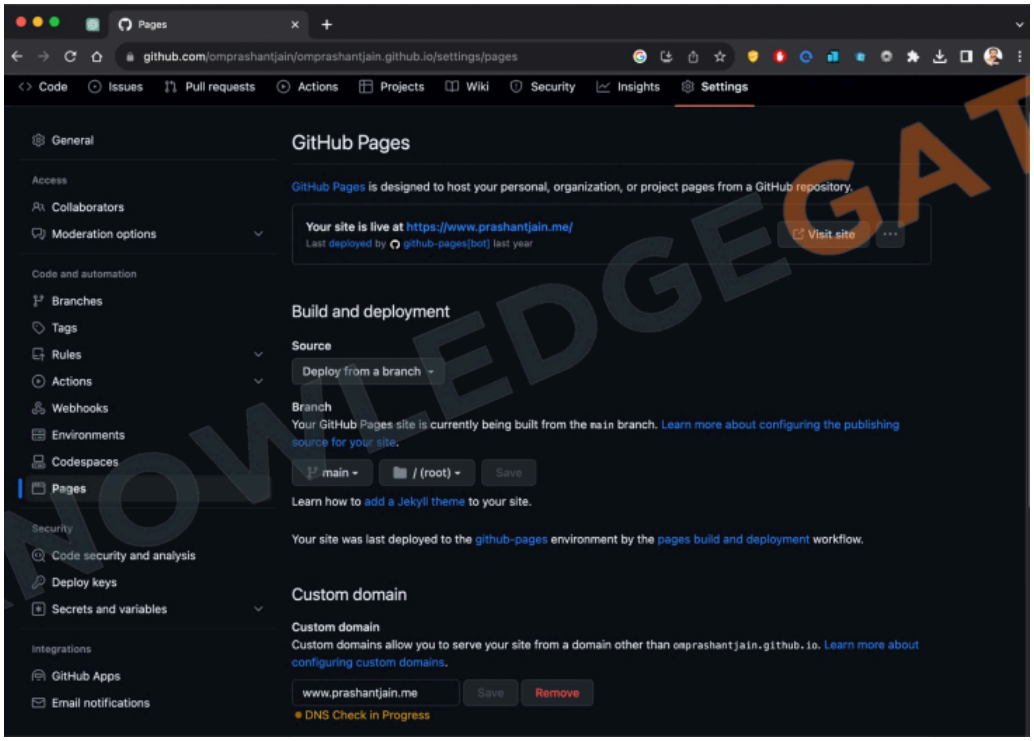
Creating a CodeSpace:



Creating a Github Page:



Publishing our Project:



Frameworks:

ReactJS

1. Definition: A tool for making websites interactive.
2. Components: Reusable pieces for building a webpage.
3. Virtual DOM: Makes websites faster by updating only what's needed.
4. JSX: A special way to write code that looks like HTML.
5. State: Keeps track of changes on the webpage.
6. Props: Shares information between different parts of a webpage.

AngularJS

1. Definition: A framework for building web applications, developed by Google.
2. Two-Way Data Binding: Updates both the view and the model simultaneously.
3. Directives: Custom HTML tags for added functionality.
4. Dependency Injection: Automatically manages how parts of the app work together.
5. Controllers: Manages the data for a specific part of the webpage.
6. SPA Support: Good for Single Page Applications where the page doesn't reload.

VueJS

1. Definition: A JavaScript framework for creating web interfaces.
2. Components: Small, reusable parts for building a website.
3. Reactivity: Automatically updates the webpage when data changes.
4. Directives: Special tokens in HTML for added functionality.
5. Vuex: Helps manage shared data across the site.
6. Single-File Components: Keeps template, script, and style in one file.