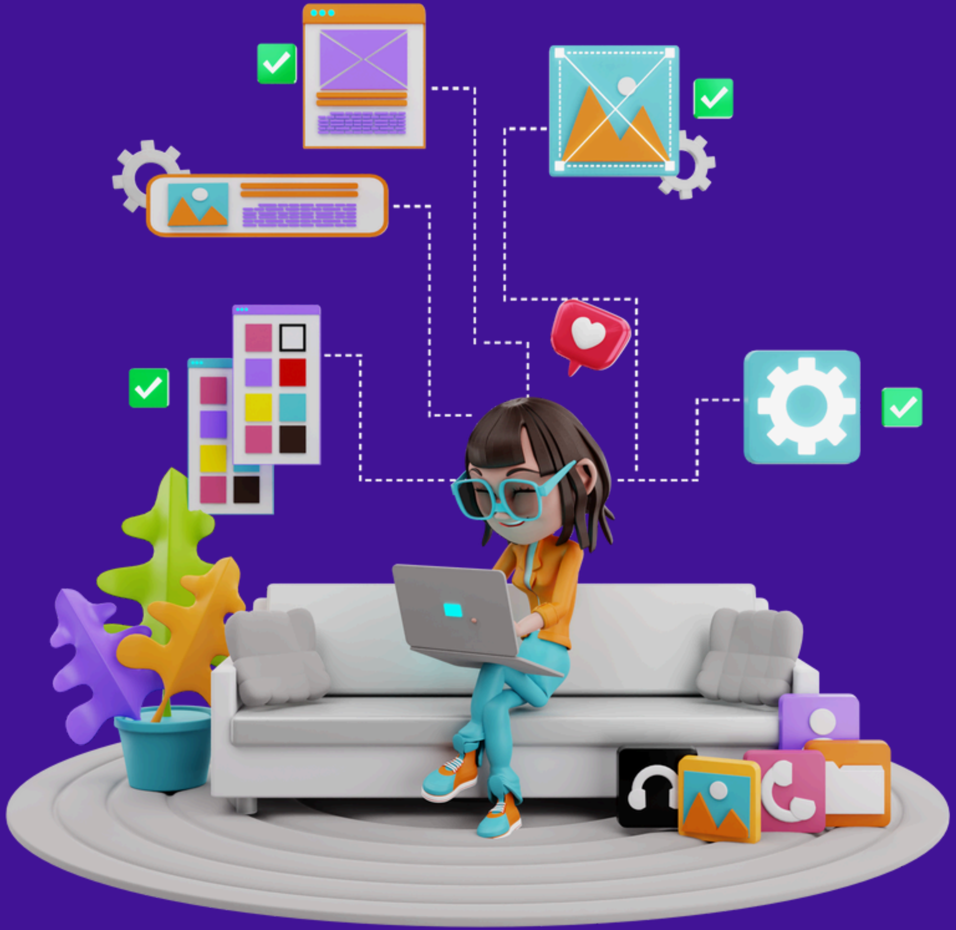
WEB DEVELOPMENT COURSE

LEARN FROM SCRATCH TO ADVANCED

HTML



HTML INTRODUCTION

Today, I'm writing this tutorial to create a resource that will help you learn HTML in less than 30 days. Here's a recommended timeline for learning HTML, based on your educational background:

- High School students and younger: Around 25 days
- Those beyond High School: Around 20 days
- College students and above: Around 10-20 days

You may be wondering why I'm discussing these timelines. It's important for me to set expectations before you start your journey of learning html with me.

HTML INTRODUCTION

Today, I'm writing this tutorial to create a resource that will help you learn HTML in less than 30 days. Here's a recommended timeline for learning HTML, based on your educational background:

- High School students and younger: Around 25 days
- Those beyond High School: Around 20 days
- College students and above: Around 10-20 days

You may be wondering why I'm discussing these timelines. It's important for me to set expectations before you start your journey of learning html with me.

What is HTML?

HTML (HYPERTEXT MARKUP LANGUAGE) WAS CREATED BY TIM BERNERS-LEE IN 1991 AS A STANDARD FOR CREATING WEB PAGES. IT'S A MARKUP LANGUAGE USED TO STRUCTURE CONTENT ON THE WEB, DEFINING ELEMENTS LIKE HEADINGS, PARAGRAPHS, LINKS, AND IMAGES. HTML FORMS THE BACKBONE OF WEB CONTENT. IN LAYMAN'S TERMS, HTML IS LIKE THE SKELETON OF A WEBSITE. IT'S A SET OF INSTRUCTIONS THAT TELLS A WEB BROWSER HOW TO DISPLAY TEXT, IMAGES, VIDEOS, AND OTHER ELEMENTS ON A WEBPAGE. THINK OF IT AS THE BUILDING BLOCKS THAT CREATE THE STRUCTURE AND LOOK OF A WEBSITE, SIMILAR TO HOW BRICKS AND MORTAR ARE USED TO BUILD A HOUSE.

IN A NUTSHELL:

- HTML IS THE LANGUAGE OF THE WEB, USED TO CREATE WEBSITES.
- HTML DEFINES THE BAREBONE STRUCTURE OR LAYOUT OF WEB PAGES THAT WE SEE ON THE INTERNET.
- HTML CONSISTS OF A SET OF TAGS CONTAINED WITHIN AN HTML DOCUMENT, AND THE ASSOCIATED FILES TYPICALLY HAVE EITHER A ".HTML" OR ".HTM" EXTENSION.
- THERE ARE SEVERAL VERSIONS OF HTML, WITH HTML5 BEING THE MOST RECENT VERSION.

FEATURES OF HTML

- EIT IS PLATFORM-INDEPENDENT. FOR EXAMPLE, CHROME DISPLAYS THE SAME PAGES IDENTICALLY ACROSS DIFFERENT OPERATING SYSTEMS SUCH AS MAC, LINUX, AND WINDOWS.
- IMAGES, VIDEOS, AND AUDIO CAN BE ADDED TO A WEB PAGE (FOR EXAMPLE - YOUTUBE SHOWS VIDEOS ON THEIR WEBSITE)

- HTML IS A MARKUP LANGUAGE AND NOT A PROGRAMMING LANGUAGE.
- IT CAN BE INTEGRATED WITH OTHER LANGUAGES LIKE CSS, JAVASCRIPT, ETC. TO SHOW INTERACTIVE (OR DYNAMIC) WEB PAGES

WHY THE TERM HYPERTEXT & MARKUP LANGUAGE?

THE TERM 'HYPERTEXT MARKUP LANGUAGE' IS COMPOSED OF TWO MAIN WORDS: 'HYPERTEXT' AND 'MARKUP LANGUAGE.' 'HYPERTEXT' REFERS TO THE LINKING OF TEXT WITH OTHER DOCUMENTS, WHILE 'MARKUP LANGUAGE' DENOTES A LANGUAGE THAT UTILIZES A SPECIFIC SET OF TAGS. THUS, HTML IS THE PRACTICE OF DISPLAYING TEXT, GRAPHICS, AUDIO, VIDEO ETC. IN A CERTAIN WAY USING SPECIAL TAGS.

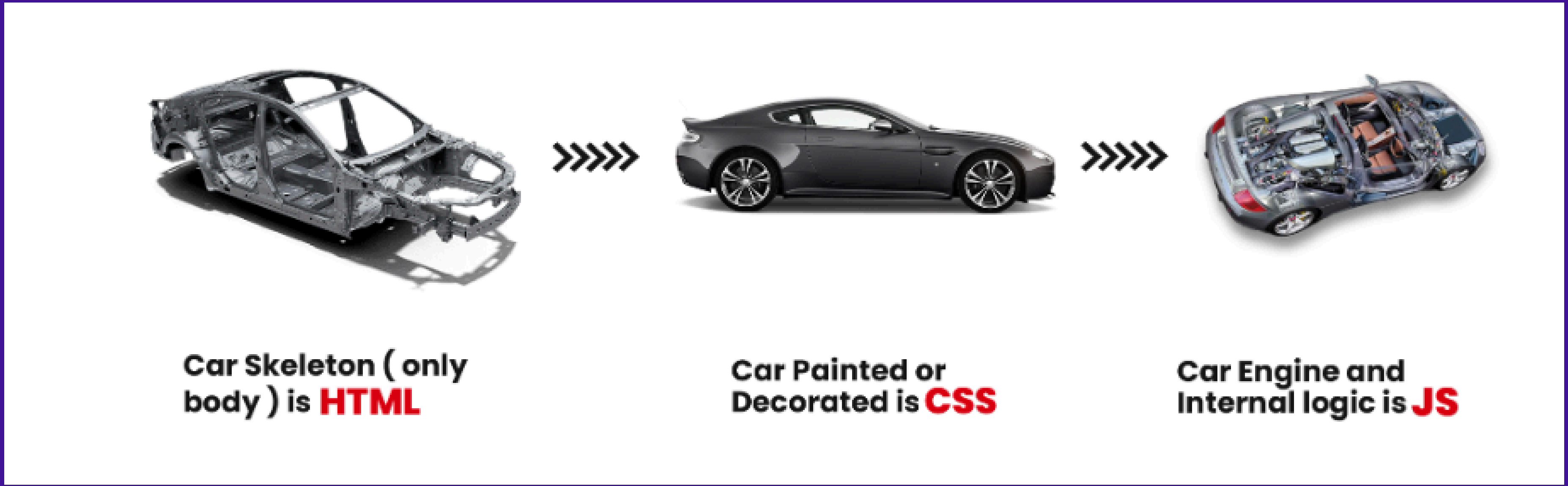
NOTE: TAGS ARE MEANINGFUL TEXTS ENCLOSED IN ANGLE BRACES, LIKE '<...>'. FOR EXAMPLE, THE '<HEAD>' TAG. EACH TAG HAS A UNIQUE MEANING AND SIGNIFICANCE IN BUILDING AN HTML PAGE, AND IT CAN INFLUENCE THE WEB PAGE IN VARIOUS WAYS.

QUICK EXERCISE:

OPEN A WEBPAGE OF YOUR CHOICE, RIGHT-CLICK ON THE BROWSER, AND SELECT 'VIEW PAGE SOURCE,' AND THEN YOU WILL SEE THE HTML CODE FOR THAT PAGE.

THIS IS THE CODE THAT THE SERVER SENT TO DISPLAY THE PAGE YOU'RE CURRENTLY VIEWING. IN THIS TUTORIAL, WE'LL LEARN HOW TO WRITE THIS TYPE OF CODE USING HTML.

A BEAUTIFUL ANALOGY TO UNDERSTAND HTML, CSS, AND JAVASCRIPT:


Car Skeleton (only body) is HTML
Car Painted or Decorated is CSS
Car Engine and Internal logic is JS

IN BUILDING A WEBPAGE, THINK OF HTML, CSS, AND JAVASCRIPT AS DIFFERENT PARTS OF A CAR. HTML IS LIKE THE CAR'S SKELETON, FORMING THE BASIC STRUCTURE AND FRAME. CSS ADDS THE PAINT AND FINISHING TOUCHES, MAKING THE CAR LOOK APPEALING WITH COLOR, STYLE, AND DESIGN. JAVASCRIPT IS SIMILAR TO THE ENGINE AND MECHANICAL PARTS, INFUSING THE CAR WITH FUNCTIONALITY, MOVEMENT, AND INTERACTIVE FEATURES. SIMILARLY, WHEN DEVELOPING A WEBSITE, HTML LAYS OUT THE STRUCTURE, CSS ENHANCES ITS VISUAL APPEAL, AND JAVASCRIPT PROVIDES INTERACTIVITY AND DYNAMIC CONTENT

HISTORY OF HTML:

- IN 1989, TIM BERNERS-LEE ESTABLISHED THE WORLD WIDE WEB (WWW), AND IN 1991, HE CREATED THE FIRST VERSION OF HTML.
- FROM 1995 TO 1997, FURTHER WORK WAS DONE TO DEVELOP AND REFINE DIFFERENT VERSIONS OF HTML.
- IN 1999, A COMMITTEE WAS ORGANIZED THAT STANDARDIZED HTML 4.0, A VERSION STILL USED BY MANY TODAY.
- THE LATEST AND MOST STABLE VERSION OF HTML IS 5, ALSO KNOWN AS HTML5.

FEEL FREE TO READ MORE HISTORY OF HTML HERE ON THE HTML WIKIPEDIA PAGE BUT I WILL MOVE AHEAD AND COVER IMPORTANT ASPECTS OF HTML.

IN BUILDING A WEBPAGE, THINK OF HTML, CSS, AND JAVASCRIPT AS DIFFERENT PARTS OF A CAR. HTML IS LIKE THE CAR'S SKELETON, FORMING THE BASIC STRUCTURE AND FRAME. CSS ADDS THE PAINT AND FINISHING TOUCHES, MAKING THE CAR LOOK APPEALING WITH COLOR, STYLE, AND DESIGN. JAVASCRIPT IS SIMILAR TO THE ENGINE AND MECHANICAL PARTS, INFUSING THE CAR WITH FUNCTIONALITY, MOVEMENT, AND INTERACTIVE FEATURES. SIMILARLY, WHEN DEVELOPING A WEBSITE, HTML LAYS OUT THE STRUCTURE, CSS ENHANCES ITS VISUAL APPEAL, AND JAVASCRIPT PROVIDES INTERACTIVITY AND DYNAMIC CONTENT

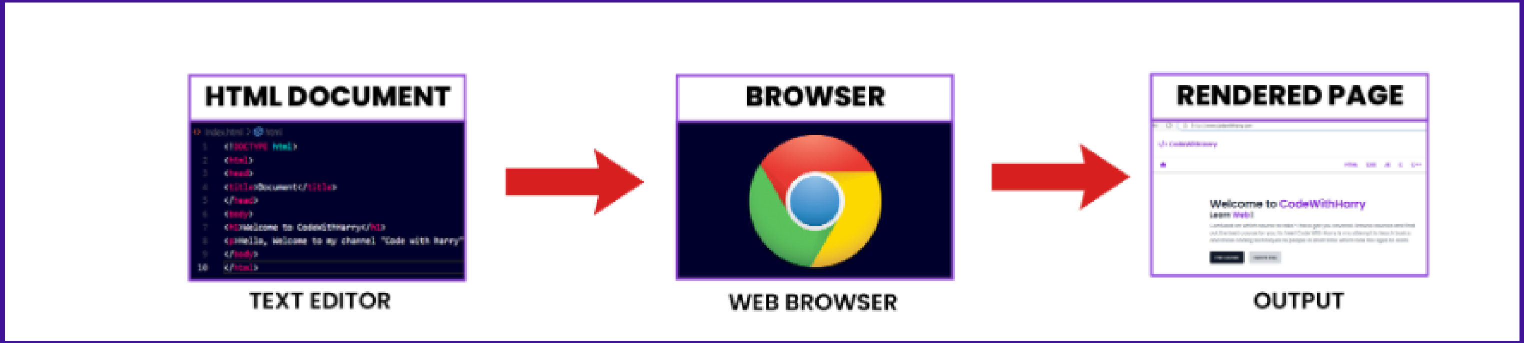
HTML WORKING

YOU MUST HAVE HEARD OF FRONTEND AND BACKEND. FRONTEND REFERS TO THE VISIBLE PART OF A WEBSITE OR APP THAT USERS INTERACT WITH, LIKE THE TABLES, IMAGES, AND BUTTONS. IT'S BUILT USING LANGUAGES LIKE HTML, CSS, AND JAVASCRIPT. THE BACKEND, ON THE OTHER HAND, HANDLES BEHIND-THE-SCENES OPERATIONS LIKE STORING AND PROCESSING DATA WHEN USERS INTERACT WITH THE FRONTEND. IT USES LANGUAGES LIKE PYTHON, RUBY, AND JAVA. IN ESSENCE, FRONTEND IS WHAT USERS SEE, WHILE BACKEND MANAGES THE FUNCTIONALITY.

HOW DO WEBSITES WORK?

WHEN WE WANT TO ACCESS ANY INFORMATION ON THE INTERNET, WE SEARCH FOR IT USING A WEB BROWSER. THE WEB BROWSER RETRIEVES THE CONTENT FROM WEB SERVERS, WHERE IT IS STORED IN THE FORM OF HTML DOCUMENTS.

AN HTML DOCUMENT IS CREATED BY WRITING CODE WITH SPECIFIC TAGS IN A CODE EDITOR OF YOUR CHOICE. THE DOCUMENT IS THEN SAVED WITH THE '.HTML' EXTENSION. ONCE SAVED, THE BROWSER INTERPRETS THE HTML DOCUMENT, READS IT, AND RENDERS THE WEB PAGE.



```
<!DOCTYPE html>
<html>
<head>
<title>Document</title>
</head>
<body>
<h1>Welcome to CodeWithHarry</h1>
<p>Hello, Welcome to my channel "Code with Harry"
</body>
</html>
```

WHAT IS A WEB BROWSER?

A WEB BROWSER IS A PROGRAM THAT UNDERSTANDS HTML TAGS AND RENDERS THEM IN A HUMAN-READABLE FORMAT THAT IS EASILY VIEWABLE BY PEOPLE VISITING THE WEBSITE. DEVELOPERS WRITE CODE IN HTML BECAUSE IT'S A STRAIGHTFORWARD WAY TO INSTRUCT THE WEB BROWSER ON WHAT TO DISPLAY. IN THE NEXT SECTION, I'LL SHOW YOU HOW TO SET UP VS CODE FOR WRITING YOUR OWN HTML CODE AND RENDERING IT IN A WEB BROWSER.

WHAT IS AN HTML DOCUMENT?

AN HTML DOCUMENT IS A TEXT DOCUMENT SAVED WITH THE '.HTML' OR '.HTM' EXTENSION, CONTAINING TEXT AND SPECIFIC TAGS ENCLOSED IN '< >'. THESE TAGS PROVIDE THE NECESSARY INSTRUCTIONS FOR CONFIGURING THE WEB PAGE. THE TAGS THEMSELVES ARE STANDARDIZED AND FIXED. THE STRUCTURE OF AN HTML DOCUMENT WILL BE EXPLAINED LATER IN THIS HTML TUTORIAL.

WHAT IS A RENDERED PAGE:

THE RENDERED PAGE IS THE OUTPUT SCREEN OF OUR HTML DOCUMENT WHICH IS THE PAGE DISPLAYED ON THE BROWSER.

HOW DOES A BASIC WEBSITE WORK?

- WEB BROWSER(CLIENT) REQUESTS WEBSITES LIKE WWWXYZ.COM FROM THE WEB SERVER.
- WEB SERVER IN RETURN SENDS HTML, CSS, AND JAVASCRIPT FILES TO IT.
- HTML, CSS, AND JAVASCRIPT FILES ARE PARSED BY A WEB BROWSER WHICH IS RESPONSIBLE FOR SHOWING YOU A BEAUTIFUL WEBSITE.

HOW DOES A BROWSER WORK?

A WEB BROWSER PLAYS A CRUCIAL ROLE IN PARSING AND RENDERING HTML CODE TO THE CLIENT. A WEB BROWSER IS A COMPLEX PROGRAM THAT PERFORMS MANY TASKS BEHIND THE SCENES. HERE'S A SUMMARY OF HOW IT WORKS:

- A BROWSER IS AN APPLICATION THAT READS HTML DOCUMENTS AND DISPLAYS THEM AS WEB PAGES. BROWSERS CAN'T ACCESS THE CONTENT DIRECTLY FROM WHERE IT'S STORED; THIS IS WHERE SERVERS COME INTO PLAY.

- A SERVER ACTS AS AN INTERMEDIARY, LISTENING TO BROWSER REQUESTS AND FULFILLING THEM BY DELIVERING THE HTML DOCUMENT TO THE BROWSER.
- WEB BROWSERS PERFORM TWO MAIN TASKS: PARSING AND RENDERING.
- DURING THE PARSING STAGE, THE BROWSER RECEIVES RAW BYTES, WHICH ARE CONVERTED INTO CHARACTERS. THESE CHARACTERS ARE THEN CONVERTED INTO TOKENS, WHICH IN TURN ARE TRANSFORMED INTO NODES. THESE NODES ARE ORGANIZED INTO A TREE-LIKE DATA STRUCTURE KNOWN AS THE DOM (DOCUMENT OBJECT MODEL).
- ONCE THE DOM TREE IS CONSTRUCTED, THE BROWSER MOVES ON TO THE RENDERING STAGE. AT THIS POINT, EACH NODE IN THE DOM TREE IS RENDERED AND DISPLAYED ON THE SCREEN.

DONT WORRY ABOUT HOW BROWSER EXACTLY WORKS JUST YET. RATHER FOCUS ON LEARNING HTML. IN THE NEXT TUOTRIAL WE WILL INSTALL VS CODE AND SOME EXTENSIONS FOR WRITING OUR HTML CODE.

HTML INSTALLATION

- LET'S GET OUR HANDS DIRTY AND START PREPARING TO WRITE SOME CODE. IN THIS TUTORIAL, WE WILL INSTALL VS CODE AND SOME RELATED EXTENSIONS FOR FASTER AND MORE EFFICIENT HTML DEVELOPMENT.

WHAT ARE THE PREREQUISITES TO LEARNING HTML?

I CAN SAFELY SAY THAT THERE ARE NO PREREQUISITES TO LEARNING HTML. HTML IS THE LANGUAGE OF THE WEB AND IS OFTEN THE FIRST STEP THAT WEB DEVELOPERS TAKE IN LEARNING TO CODE.

TOOLS NEEDED TO MAKE AN HTML PAGE:

1) **HTML EDITOR:** IT'S A STRAIGHTFORWARD TOOL WHERE EVERY PIECE OF HTML CONTENT MUST BE WRITTEN. YOU CAN USE ANY TEXT EDITOR OF YOUR CHOICE. IN THIS TUTORIAL, WE'RE USING VISUAL STUDIO CODE BECAUSE IT'S LIGHTWEIGHT AND OPEN-SOURCE.
POPULAR EDITORS FOR HTML DEVELOPMENT INCLUDE TEXT EDITORS LIKE **NOTEPAD++** AND **TEXTEDIT,** CODE EDITORS SUCH AS **SUBLIME TEXT** AND **VISUAL STUDIO CODE,** AND FULL-FLEDGED IDES LIKE **WEBSTORM** AND **ECLIPSE**. ONLINE PLATFORMS LIKE **CODEPEN** AND **JSFIDDLE** ARE ALSO COMMONLY USED FOR QUICK HTML EDITING AND TESTING.

NOTE: YOU CAN WRITE HTML EVEN IN A NOTEPAD. TEXT EDITORS LIKE VS CODE MAKE THESE THINGS EASIER.

2) **BROWSER:** HTML TAGS ARE NOT DISPLAYED BY BROWSERS; INSTEAD, THEY ARE READ AND INTERPRETED TO RENDER THE WEB PAGE. IN A WEB BROWSER, HTML STRUCTURES ARE DISPLAYED IN A STYLED AND VISUALLY APPEALING FORM. IN THIS TUTORIAL, WE ARE USING GOOGLE CHROME. OTHER COMMONLY USED BROWSERS INCLUDE CHROMIUM, FIREFOX, SAFARI ON MAC, AND MICROSOFT EDGE.

INSTALLATION & SETUP OF VISUAL STUDIO CODE FOR HTML:

WE WILL INSTALL AND SET UP HTML TO OPTIMIZE ITS UTILITY FOR CREATING WEB PAGES. ADDITIONALLY, WE'LL INSTALL EXTENSIONS IN VISUAL STUDIO CODE TO ENHANCE ITS FUNCTIONALITY. IF YOU'RE UNSURE ABOUT WHICH EDITOR TO USE, YOU CAN CONFIDENTLY START WITH VISUAL STUDIO CODE. YOU WON'T REGRET IT; IT'S ONE OF THE BEST FREE CODE EDITORS AVAILABLE IN THE MARKET.

- SEARCH FOR "VISUAL STUDIO CODE DOWNLOAD" ON GOOGLE
- DOWNLOAD **VISUAL STUDIO CODE** FOR YOUR OPERATING SYSTEM. I AM USING WINDOWS SO I WILL INSTALL IT FOR WINDOWS

HTML Execution

Your Journey to Creating Your First Website Begins Here!

Let's mark this as an important milestone: the creation of your first website!
And what's a better way to start than with the traditional "Hello, World!"?

Why "Hello, World!"?

In the programming world, "Hello, World!" is more than just a phrase. It's a tradition, an emotion, a simple program that teaches you the syntax and gets you started. And guess what? HTML is no different!
Our first website will display the text 'Hello World'

Let's Get Started: Setting Up Your VS Code

If you haven't already set up your environment, let's begin by opening Visual Studio Code (VS Code).

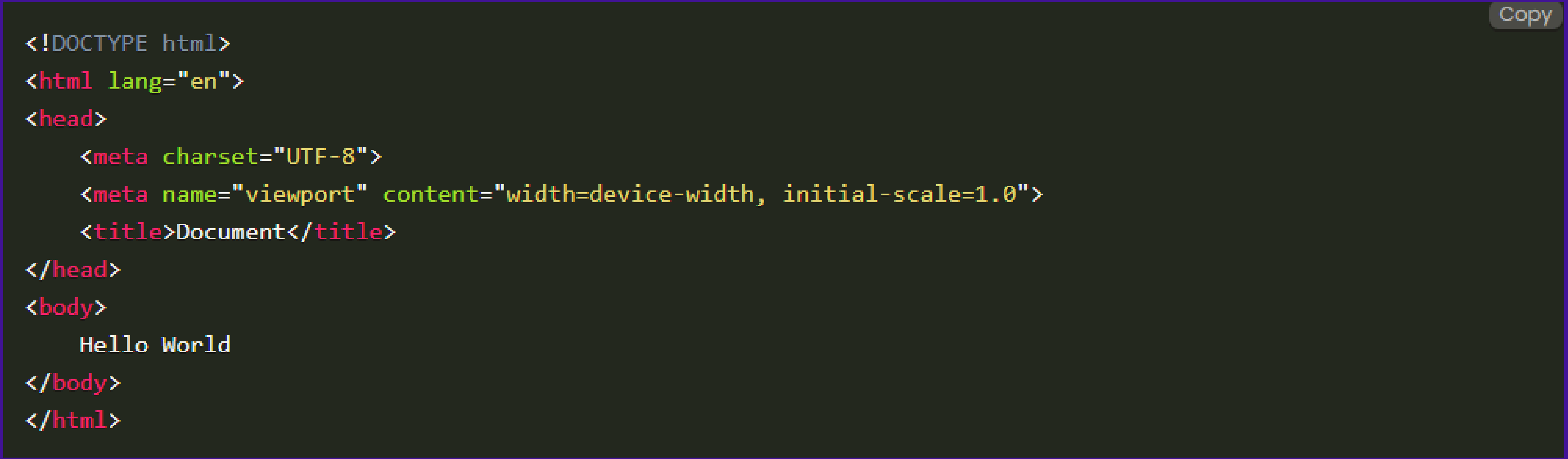
Creating a New File

Click on "Open Folder" and open a folder somewhere on your computer. I am opening a folder named html-tutorial
Once VS Code is open, you'll want to create a new file:

1. Click on the "New File" icon in VS Code.
2. Type the filename as "index.html" and hit Enter.

Pasting the Code

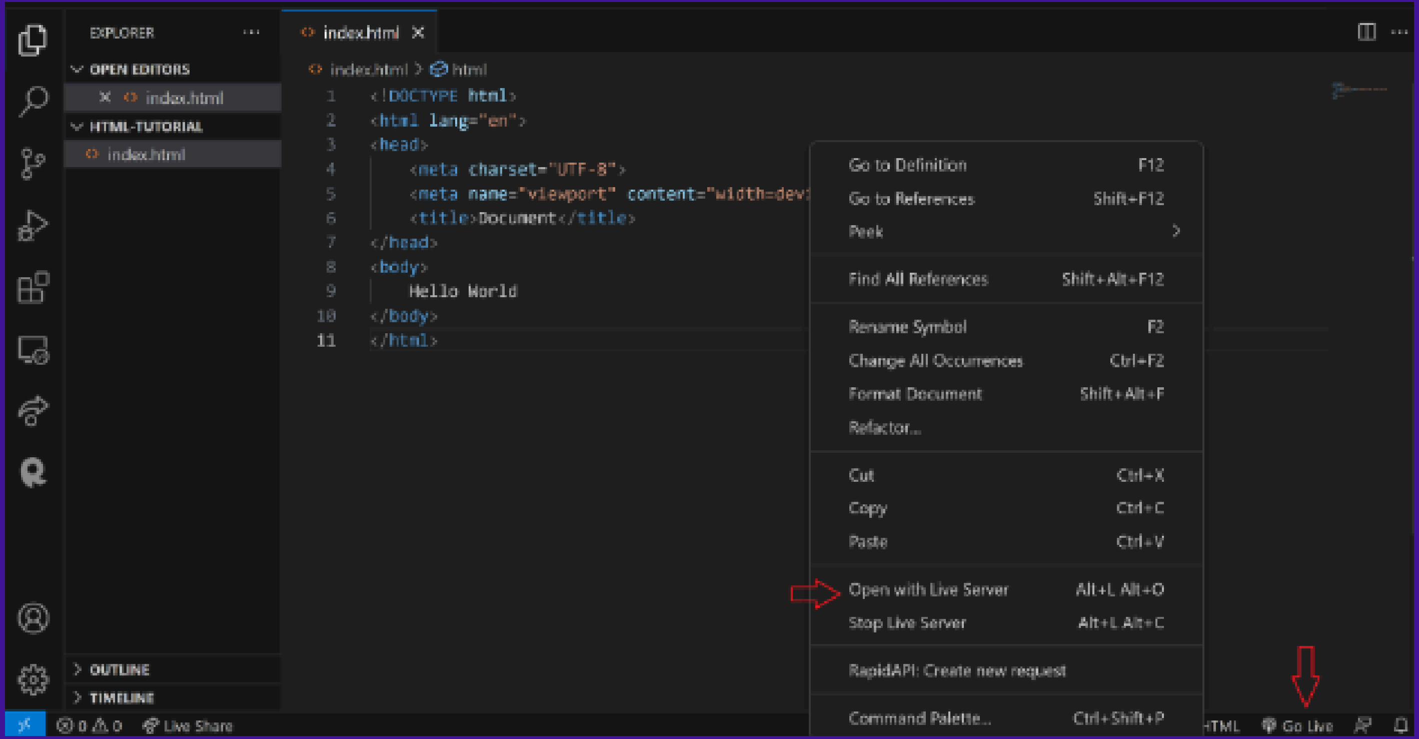
Now that your file is ready, copy the following code and paste it into your "index.html" file.



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    Hello World
</body>
</html>
```

Going Live using the "live server" extension

To see your webpage in action, locate the "Go Live" icon at the bottom-right corner of your VS Code window and click it. If you don't see this icon, you probably haven't installed the Live Server extension, which we discussed in a previous tutorial.

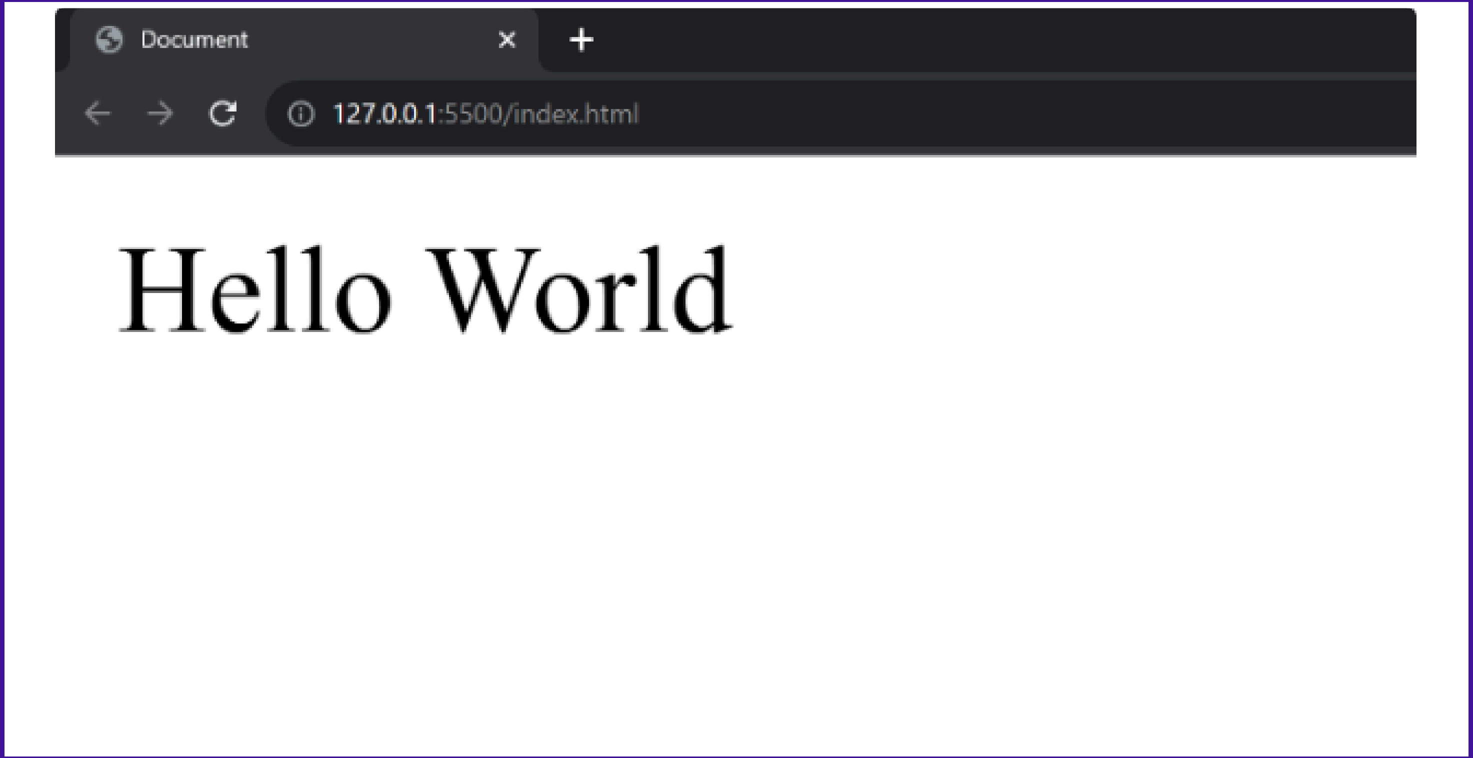


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=dev
    <title>Document</title>
</head>
<body>
    Hello World
</body>
</html>
```

EXPLORER
OPEN EDITORS
index.html
HTML-TUTORIAL
index.html
OUTLINE
TIMELINE
Live Share
Go to Definition F12
Go to References Shift+F12
Peek
Find All References Shift+Alt+F12
Rename Symbol F2
Change All Occurrences Ctrl+F2
Format Document Shift+Alt+F
Refactor...
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Open with Live Server Alt+L Alt+O
Stop Live Server Alt+L Alt+C
RapidAPI: Create new request
Command Palette... Ctrl+Shift+P
HTML
Go Live

Your First Website is Live!

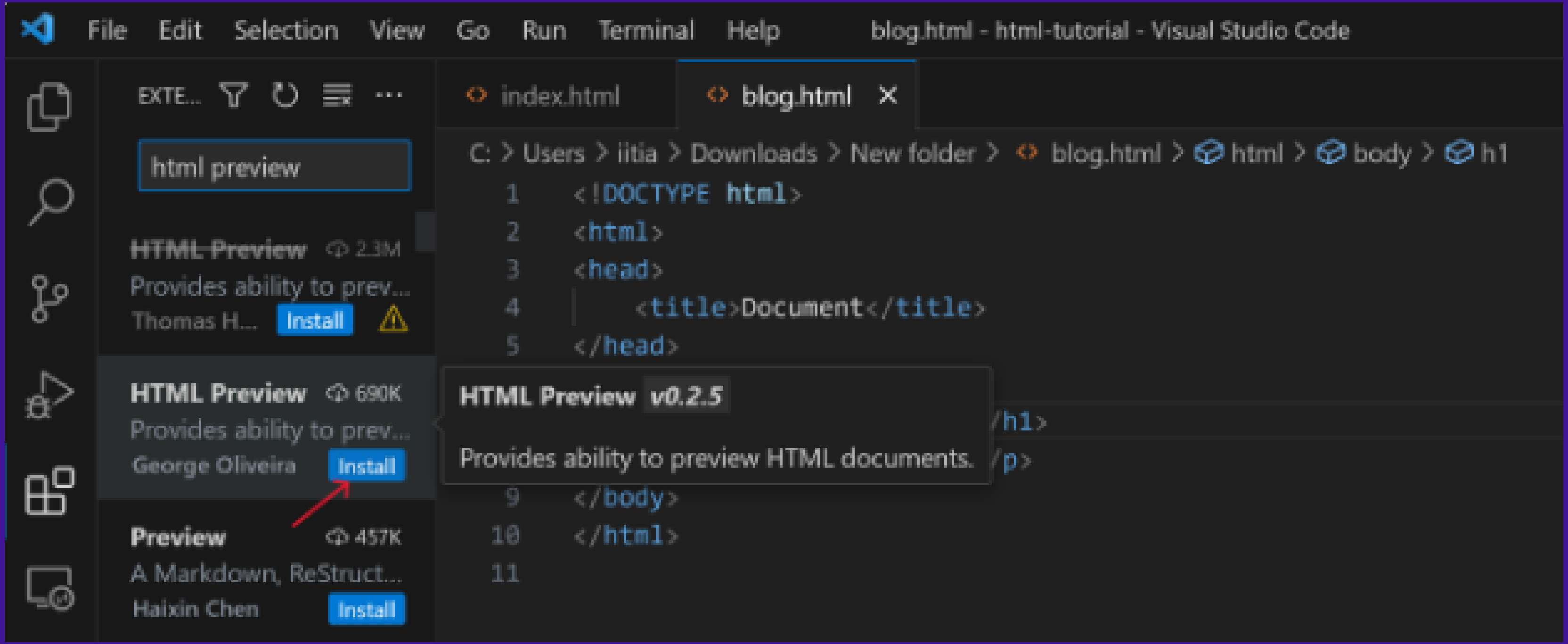
Congratulations! If you've followed along, you should now see your very first website displaying the classic "Hello, World!" message.



Hello World

Live Preview Extension

Another useful extension for working with HTML in VS Code is 'HTML Preview.' To install it, simply click on the extensions icon in VS Code and type 'HTML Preview' in the search bar. Install it!


File
Edit
Selection
View
Go
Run
Terminal
Help
blog.html - html-tutorial - Visual Studio Code
EXTE...
index.html
blog.html
C: > Users > iitia > Downloads > New folder > blog.html > html > body > h1
Install
Install
Install

html preview

~~HTML Preview~~ 2.3M
Provides ability to prev...
Thomas H...

HTML Preview 690K
Provides ability to prev...
George Oliveira

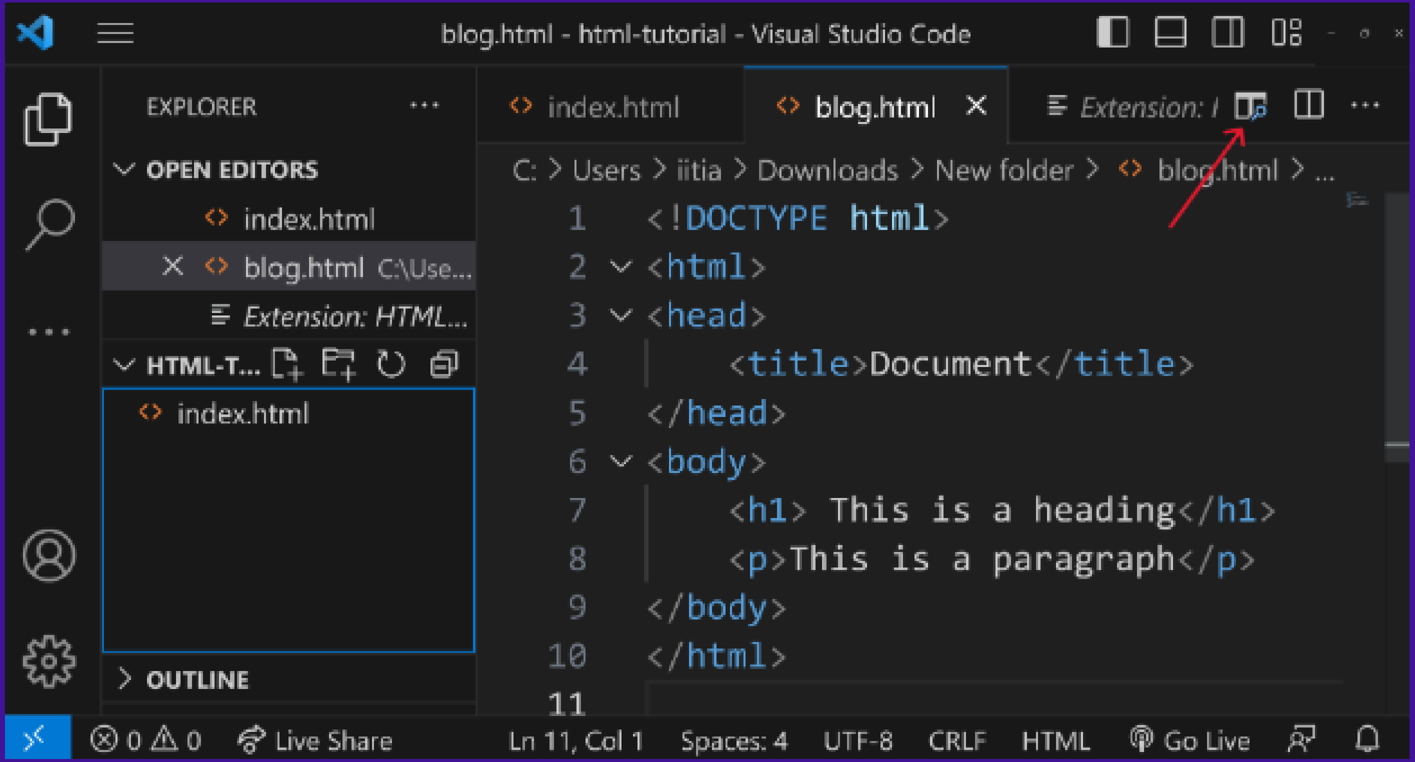
Preview 457K
A Markdown, ReStruct...
Haixin Chen

HTML Preview v0.2.5

Provides ability to preview HTML documents.

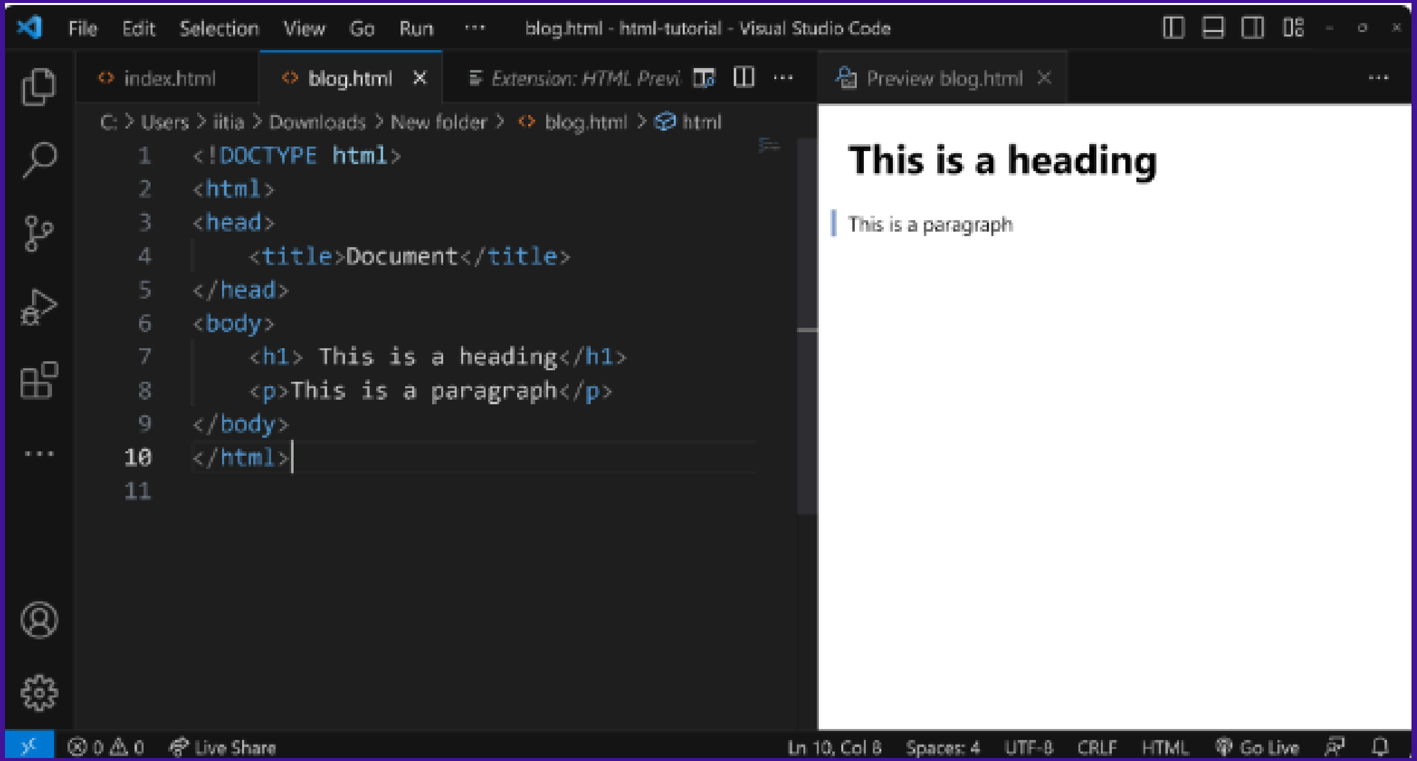
```
<!DOCTYPE html>
<html>
<head>
    <title>Document</title>
</head>
                                              /h1>
                                              /p>
</body>
</html>
```

Now, you will see a button within VS Code. Clicking on this button will allow you to preview your HTML right within the editor.


blog.html - html-tutorial - Visual Studio Code
EXPLORER
OPEN EDITORS
index.html
blog.html C:\Use...
Extension: HTML...
HTML-T...
index.html
OUTLINE
index.html
blog.html
Extension: I
C: > Users > iitia > Downloads > New folder > blog.html > ...
0 0 Live Share
Ln 11, Col 1 Spaces: 4 UTF-8 CRLF HTML Go Live

```html
<!DOCTYPE html>
<html>
<head>
    <title>Document</title>
</head>
<body>
    <h1> This is a heading</h1>
    <p>This is a paragraph</p>
</body>
</html>
```

Once you click the button, you'll see a live preview of your HTML directly within VS Code.


File Edit Selection View Go Run ···
blog.html - html-tutorial - Visual Studio Code
index.html
blog.html
Extension: HTML Previ
Preview blog.html
C: > Users > iitia > Downloads > New folder > blog.html > html
Ln 10, Col 8 Spaces: 4 UTF-8 CRLF HTML Go Live
Live Share

```html
<!DOCTYPE html>
<html>
<head>
    <title>Document</title>
</head>
<body>
    <h1> This is a heading</h1>
    <p>This is a paragraph</p>
</body>
</html>
```

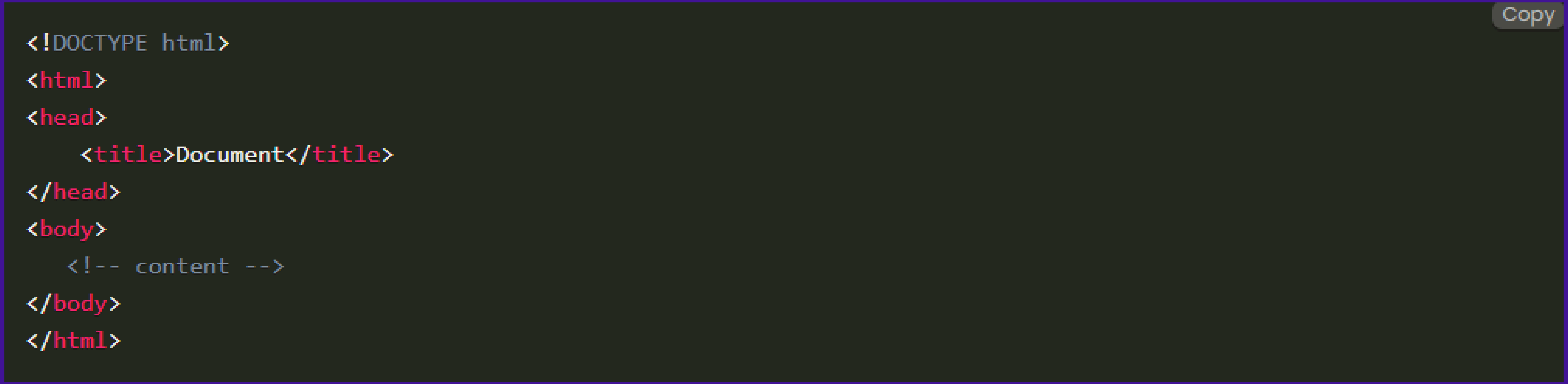
This is a heading

This is a paragraph

ou don't even need a browser to render plain HTML. This live preview feature in VS Code is perfect for this HTML tutorial, allowing you to build and preview your entire HTML website without ever leaving the editor.

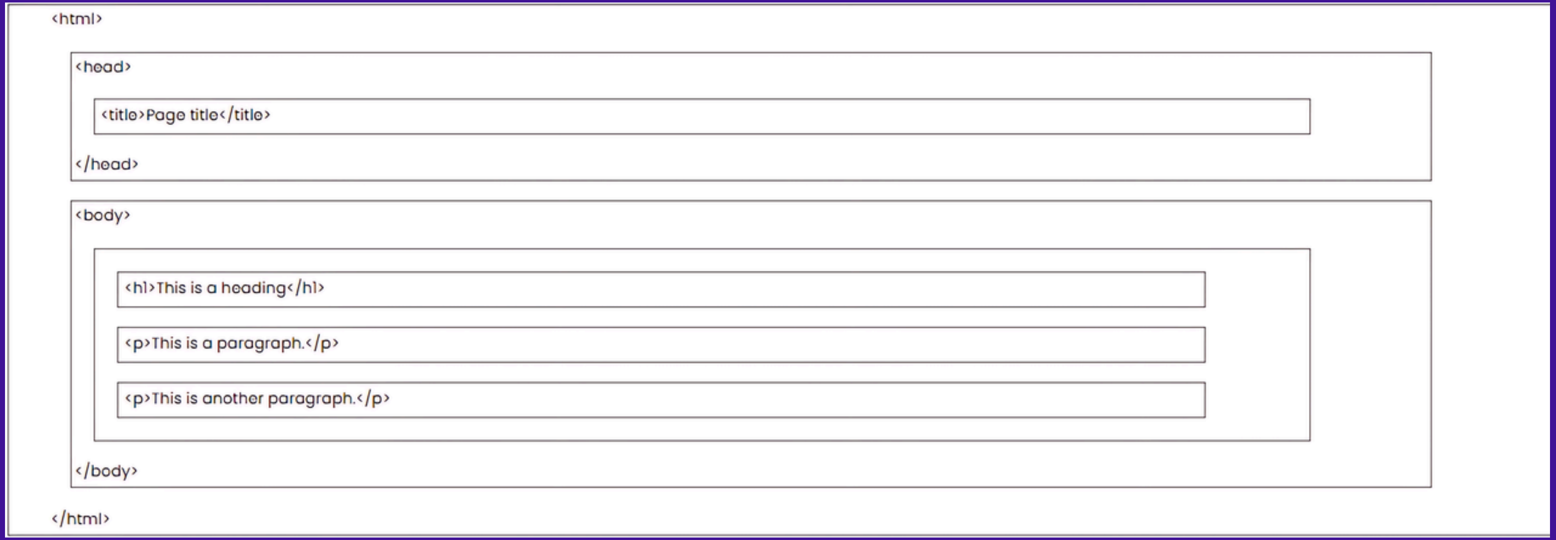
HTML Page Structure

An HTML document is structured using a set of nested tags. Each tag is enclosed within <…> angle brackets and acts as a container for content or other HTML tags. Let's take a look at a basic HTML document structure:



```
<!DOCTYPE html>
<html>
<head>
    <title>Document</title>
</head>
<body>
   <!-- content -->
</body>
</html>
```

This is how the title appears on an HTML page:



<html>
<head>
<title>Page title</title>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
</body>
</html>

Almost every website uses this structure. The main content goes inside the body tag. No worries if this looks complicated; let's break it down!

Note: These are the essential elements for a basic HTML document: <!DOCTYPE html>, <html>, <head>, <title>, </head>, <body>, </body>, </html>

DOCTYPE Declaration

```
<!DOCTYPE html>
```

The <!DOCTYPE html> declaration informs the web browser about the HTML version being used. The latest version is HTML5. But if this changes in the future (maybe 10 years down the line), the doctype declaration will be helpful!

HTML Root Element

```
<html>
```

The <html> tag is the root element that encapsulates all the content on the page.

```
</html>
```

The </html> tag marks the end of the <html> section.

Head Section

```
<head>
```

The <head> tag contains metadata and links to external resources like CSS and JavaScript files.

```
</head>
```

The </head> tag marks the end of the <head> section.

Title Tag

```
<title>Document</title>
```

The <title> tag sets the title of the web page, which is displayed in the browser's title bar or tab.

Body Tag

```html
<body>
```

The <body> tag contains the visible content of the web page. This is where text, images, and other elements go.

```html
</body>
```

The </body> tag marks the end of the visible content of the web page.

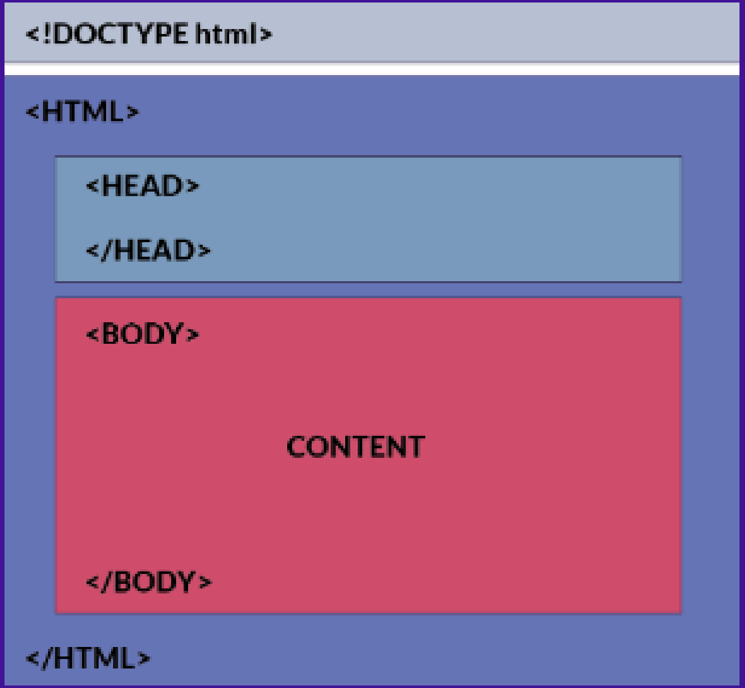
Every HTML page should include at least these essential elements to define the basic layout. In upcoming tutorials, we'll dive deeper into the fascinating world of HTML.

Summary

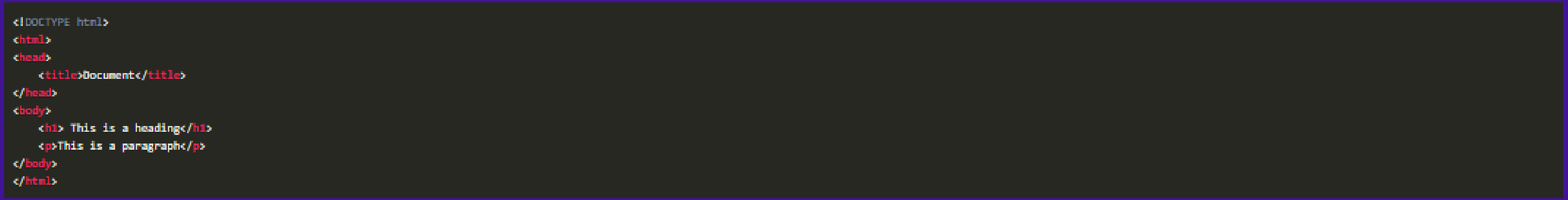
- The <!DOCTYPE html> tag specifies that the document is an HTML5 document.
- The <html lang="en"> tag defines the document to be in English.
- The <head> section contains metadata and the title of the webpage, which appears in the browser's title bar.
- The <body> section contains the content that will be displayed on the webpage.
- The h1 and p are two types of tags. We will learn about more tags in the later section

Visualization of an HTML Document:

The following image provides a visual representation of the HTML structure:

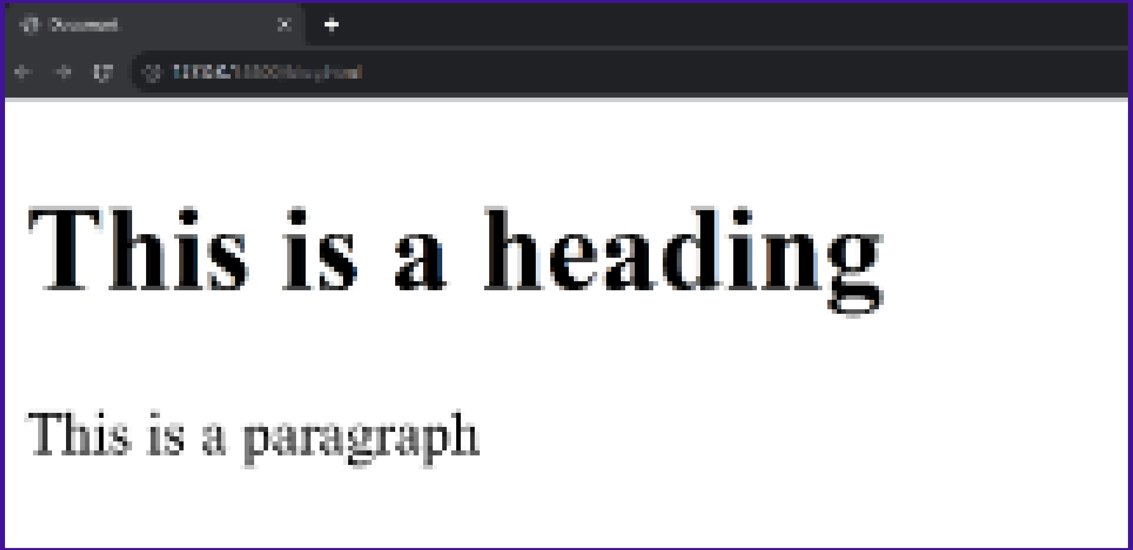


How This Content Appears in a Web Browser:



```html
<!DOCTYPE html>
<html>
<head>
    <title>Document</title>
</head>
<body>
    <h1> This is a heading</h1>
    <p>This is a paragraph</p>
</body>
</html>
```

Below is an image showing how this HTML document will be rendered in a web browser:


This is a heading

This is a paragraph

In the browser, the title bar will display the content from the <head> section, specifically the <title> tag. The main area of the browser window (usually a white background) will display the content inside the <body> tag.
In the upcoming sections, we will learn about html tags and elements.

HTML Tags

If you want to build a beautiful website, tags are essential elements that help you achieve that.
An HTML tag acts as a container for content or other HTML tags. Tags are words enclosed within < and > angle brackets.
They serve as keywords that instruct the web browser on how to format and display the content.

Commonly used tags in HTML

Here are some commonly used tags in HTML. These are the only tags used 70% of the time

Document Structure Tags

1. <!DOCTYPE html>: Specifies the document type.
2. <html>: Encloses the entire HTML document.
3. <head>: Contains meta-information and links to scripts and stylesheets.
4. <body>: Contains the content of the web page.

Metadata Tags

1. <title>: Sets the title of the web page.
2. <meta>: Provides metadata such as character set, author, and viewport settings.
3. <link>: Links external resources like stylesheets.

Text Formatting Tags

1. <p>: Paragraph.
2. <h1>, <h2>, <h3>, <h4>, <h5>, <h6>: Headings.
3. <strong>: Strong emphasis (typically bold).
4. <em>: Emphasis (typically italic).
5. <br>: Line break.
6. <hr>: Horizontal rule.

List Tags

1. <ul>: Unordered list.
2. <ol>: Ordered list.
3. <li>: List item.

Hyperlink and Media Tags

1. <a>: Anchor (used for links).
2. <img>: Image.
3. <audio>: Audio content.
4. <video>: Video content.

Form Tags

1. <form>: Form.
2. <input>: Input field.
3. <textarea>: Text area.
4. <button>: Button.
5. <select>: Dropdown list.
6. <option>: Options within a <select> or <datalist>.

Table Tags

1. <table>: Table.
2. <tr>: Table row.
3. <td>: Table data cell.
4. <th>: Table header cell.
5. <thead>: Table header group.
6. <tbody>: Table body group.
7. <tfoot>: Table footer group.

Semantic Tag

1. <header>: Header section.
2. <footer>: Footer section.
3. <article>: Article.
4. <section>: Section.
5. <nav>: Navigation.
6. <aside>: Sidebar content.

Paired and Unpaired HTML Tags

Well, that was a really long list. Don't worry we will study these in detail. In HTML, tags can be broadly categorized into two types:

1. Paired Tags (Container Tags)

These are tags that come in pairs, consisting of an opening tag and a corresponding closing tag. The content goes between these two tags.

- **Opening Tag:** The opening tag starts with < and ends with >. For example, <p>.
- **Closing Tag**: The closing tag also starts with < but includes a forward slash / before the tag name, and ends with >. For example, </p>.

Examples:

- Paragraphs: <p>This is a paragraph.</p>
- Headings: <h1>This is a heading.</h1>

2. Unpaired Tags (Self-Closing Tags or Stand-Alone Tags)

These are tags that don't require a closing tag. They are self-contained, encapsulating all the information within a single tag.

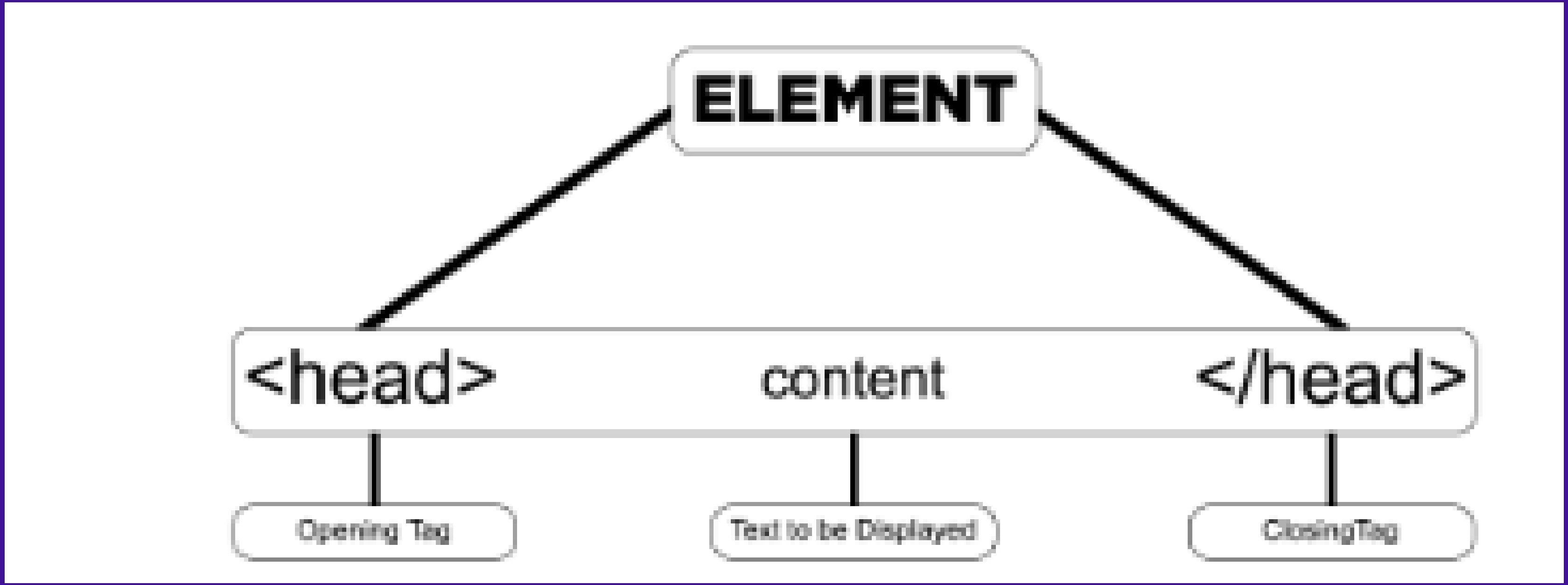
- **Self-Closing Tag:** A self-closing tag starts with < and ends with /> (though the / is optional in HTML5). For example, <img /> or <br>.
  Note: Later if you happen to use react or a framework like Next.js, you will have to close the tag like this <br/> <hr/>. So it is better to cultivate the habit!

Examples of self-closing tags:

- **Line Break: <br/>**
- **Horizontal Rule: <hr/>**
- **Image: <img src="image.jpg" alt="An example image"/>**

Pictorial Representation of Tags

The image below offers a visual representation of how tags are structured in HTML. As you can see, an element can contain other elements, which may also contain additional elements, forming a tree-like structure. This hierarchy can include self-closing tags as well as nested tags, as illustrated in the picture



HTML Elements

Beginners often get confused between HTML elements, nested elements, and tags. Let's clarify the difference by understanding each one step-by-step.

What is an HTML Element?

An HTML element is a complete set that consists of a start tag (or opening tag), content, and an end tag (or closing tag).
HTML Element = Start Tag + Content + End Tag
For example:

```
first heading</h1>
```

In this example, <h1> is the start tag, "This is our first heading" is the content, and </h1> is the end tag. Together, they form an HTML element.

What is a Nested HTML Element?

A nested HTML element is an HTML structure where one element is placed inside another element. The enclosing element is often referred to as the "parent," while the enclosed element is called the "child."

Nested HTML Element = One HTML Element Inside Another HTML Element

```
<h1><b>This is our first heading</b></h1>
```

In this example, the <b> element (child) is nested inside the <h1> element (parent).

What is an Empty HTML Element?

An empty HTML element is one that does not have a closing tag or content. These elements are also known as "void elements" or "self-closing elements."
Empty **HTML Element = Tags with No Content**

```
<br />
```

This is a break tag, which has no content and no closing tag. It's used to insert a line break within text or other inline elements. The <hr /> tag, used for horizontal rules, is another example of an empty or void element.

HTML Tags vs. Elements

HTML Tags

HTML tags are the markers that define the start and end of an element. They are wrapped in angle brackets, like <p> and </p>.

HTML Elements

An HTML element includes an opening tag, content, and a closing tag, forming a complete set. For example, <p>This is a paragraph.</p>.

Key Takeaways

- Tags set boundaries; elements include tags plus content.
- Tags come in pairs (most of the time), whereas elements may include nested elements.
- Self-closing or void elements like <br /> are still considered elements, even though they don't have a closing tag or content.
- Elements can be "parent" or "child" when nested, but tags cannot.

HTML Attributes

HTML attributes are used to define the characteristics of an HTML element. They are placed within the element's opening tag and consist of two parts: the name and the value.

- **Name:** Specifies the property for that element.
- **Value:** Sets the value of that property for the element.

Types of HTML Attributes

There are three main types of HTML attributes:

- **Core Attributes:** These are basic attributes that can be applied to most HTML elements. Examples include id, class, and style.
- **Internationalization Attributes:** These attributes help adapt the document to different languages and regions. Examples include lang and dir.
- **Generic Attributes:** These attributes provide additional information about the element but don't necessarily affect its appearance or behavior. Examples include data-* attributes for storing custom data private to the page or application.

Core attributes are some of the most widely used attributes in HTML. There are four main types:

- id
- class
- title
- style

ID Attribute

The ID attribute is used to assign a unique identifier to an HTML element. Each element with an ID has its own unique identity, similar to how each individual has a unique identity. Multiple elements cannot have the same ID.
Example:

```
"html">This is an HTML tutorial</p>
"python">This is a Python tutorial</p>
```

In this example, the ID attribute helps to distinguish between two paragraphs by having different values: "html" and "python".

Class Attribute

The class attribute is used to associate an HTML element with a particular class, typically for styling or JavaScript manipulation. Unlike the ID attribute, the class attribute is not unique, and multiple elements can share the same class.

Title Attribute

The title attribute provides additional information about an element and is often displayed as a tooltip when the mouse hovers over it.

```
<h4 title="hello, motto">Title attribute</h4>
```

Copy

Style Attribute

The style attribute allows for inline styling of HTML elements. It is used in conjunction with CSS properties to directly style individual elements within the HTML code.

Case Sensitivity

The HTML standard is flexible about the case of attribute names, allowing them to be written in either uppercase or lowercase, such as "title" or "TITLE." However, for best practices and compatibility with stricter document types like XHTML, the W3C recommends using lowercase attributes.

HTML Comments

Comments in HTML are like little notes you leave in your code for yourself or other people. These notes help make the code easier to understand but don't show up on the actual website. The web browser just skips over them!

Key Points About HTML Comments

- Comments are ignored by web browsers.
- They aid in code readability and documentation.
- HTML comments are denoted by <!-- content -->.
- The shortcut key for commenting out code is Ctrl + /.
- HTML supports both single-line and multi-line comments.

Types of Comments in HTML

HTML primarily supports two types of comments:

Single-line Comments

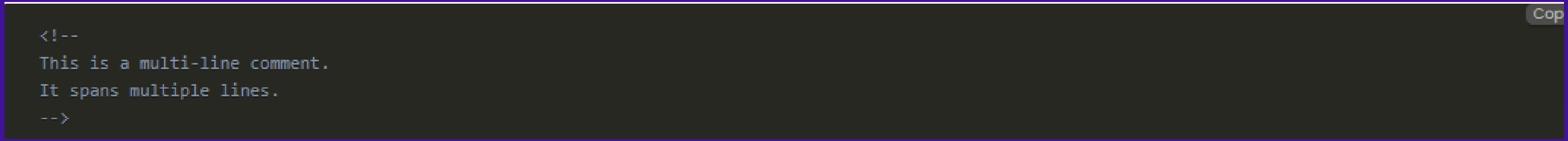
Single-line comments are contained within one line. They are useful for short annotations.

Example:

```html
<!-- This is a single-line comment -->
```

Multi-line Comments

Multi-line comments span across multiple lines, making them ideal for detailed explanations or temporarily disabling blocks of code.



```
<!--
This is a multi-line comment.
It spans multiple lines.
-->
```

HTML Id & Classes

Multi-line comments span across multiple lines, making them ideal for detailed explanations or temporarily disabling blocks of code.

What is an ID?

An ID is an attribute, a unique identifier assigned to only one HTML element within a page. It is often used for unique styling and JavaScript manipulations.

```
<div id="myUniqueID">This is a div with an ID.</div>
```

What are Classes?

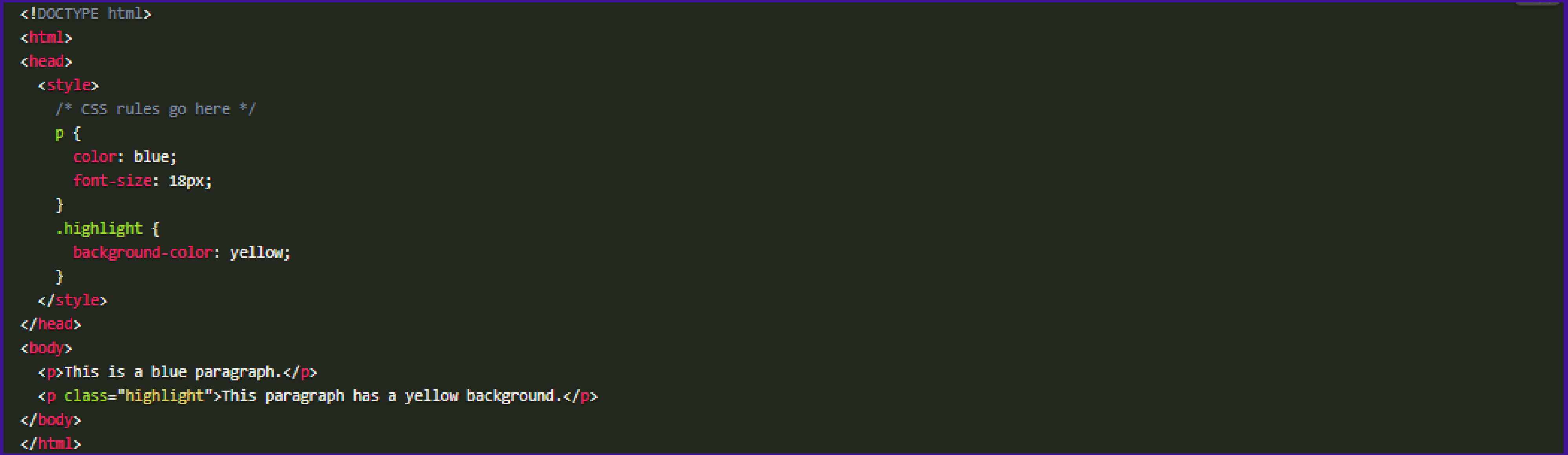
The class attribute lets you give the same name to multiple HTML elements. That way, you can easily change their look or behavior all at once. Classes are not unique and can be assigned to multiple elements. They are generally used for applying the same styles or behaviors to a group of elements.

```
<div class="myClass">This is a div with a class.</div>
<p class="myClass">This is a paragraph with the same class.</p>
```

The Style Tag

The style tag in HTML is used to include embedded CSS (Cascading Style Sheets) within an HTML document. It is commonly placed within the <head> section of the HTML file, although it can technically be used in the <body> as well. The style tag allows you to define the look and feel of various HTML elements on the page, including their colors, sizes, margins, and other visual styles.

Here's a simple example:



```
<!DOCTYPE html>
<html>
<head>
  <style>
    /* CSS rules go here */
    p {
      color: blue;
      font-size: 18px;
    }
    .highlight {
      background-color: yellow;
    }
  </style>
</head>
<body>
  <p>This is a blue paragraph.</p>
  <p class="highlight">This paragraph has a yellow background.</p>
</body>
</html>
```

In this example, we have targetted the second paragraph by its class name in CSS. The style tag is used to add CSS right into HTML. We will learn about CSS and selectors later in the CSS tutorial

Using IDs and Classes in CSS

In CSS, elements with IDs are selected using a hash (#) symbol before the ID, and elements with classes are selected using a dot (.) before the class name.

Differences Between IDs and Classes

- **Uniqueness:** IDs are unique, and classes can be reused.
- **JavaScript:** IDs are often used for JavaScript operations.
- **Styling:** Both can be used for styling, but IDs have higher specificity.

Conclusion

Understanding the difference between IDs and Classes is crucial for effective web development. While IDs are for unique elements, classes are for grouping elements.

HTML BASIC TAGS

Skeletal Tags

Let's discuss some basic HTML tags known as **"skeletal tags"**.

<html> Tag: "Root of an HTML Page"

Syntax:

```
<html>
  <!-- Content -->
</html>
```

The <html> tag is the root element that wraps all the content on the page. It generally contains two main sections: the header (<head>...</head>) and the body (<body>...</body>).

<head> Tag: "Header Part of an HTML Page"

Syntax:

```
<head>
  <!-- Header Content -->
</head>
```

The <head> tag contains meta information and the title of the document. While the title appears in the browser tab, meta information is often used for SEO purposes.

<title> Tag: "Title Part of an HTML Page"

Syntax:

```
<title>
  // Title Name
</title>
```

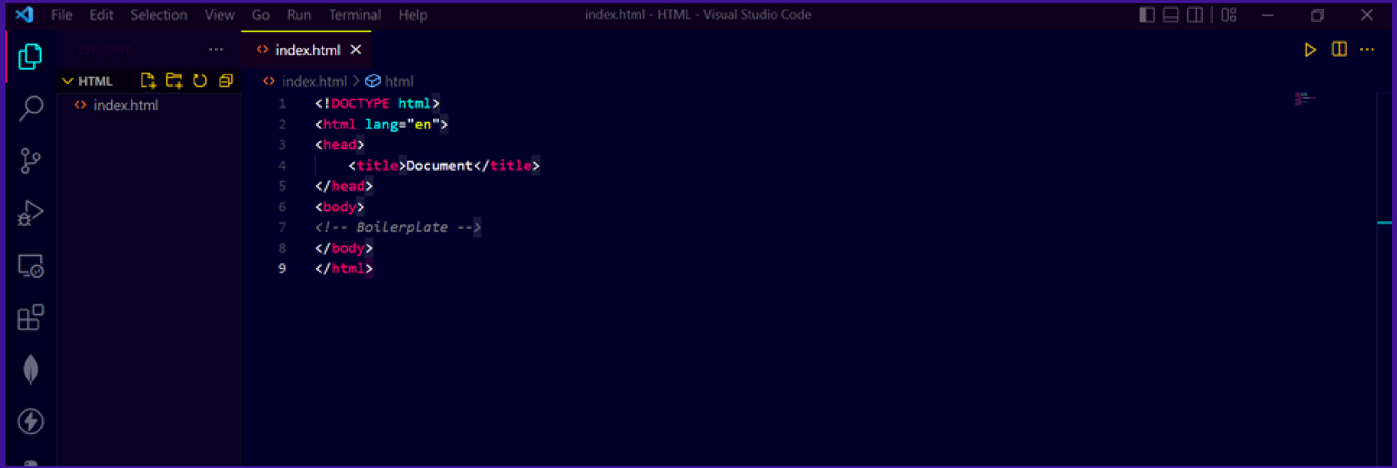
<body> Tag: "Body Part of an HTML Page"

Syntax:

```
<body>
  // Body Content
</body>
```

The <body> tag encloses the main content of the page, and everything within this tag is displayed in the browser.

The image below shows the skeletal tags and essential tags needed to define the layout of a webpage:


File Edit Selection View Go Run Terminal Help
index.html - HTML - Visual Studio Code
HTML
index.html
index.html > html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Document</title>
</head>
<body>
<!-- Boilerplate -->
</body>
</html>
```

Heading Tags

In HTML, heading tags ranging from <h1> to <h6> are used to define the structure and layout of text on a web page. These tags help create a hierarchical organization of content, making it easier for both users and search engines to understand the page's content.

The <h1> tag is generally used for the main title and is the largest and most prominent, while <h2> to <h6> tags are used for subheadings, further subheadings and so on... Proper use of heading tags not only improves the readability of a web page but also optimizes it for search engine ranking in Google

<h1> Tag: First-Level Heading

The <h1> tag defines the first-level heading and is typically the largest and boldest among all the heading tags. It is often used for the main title of the page.

```
<h1>
  <!-- Heading content -->
</h1>
```

<h2> Tag: Second-Level Heading

The <h2> tag is used for second-level headings and is slightly smaller than the <h1> tag. This is commonly used for section titles.

```
<h2>
  <!-- Heading content -->
</h2>
```

<h3> Tag: Third-Level Heading

The <h3> tag is used for third-level headings. These are smaller than <h2> tags and are often used for sub-sections within an <h2> section.

Copy

```
<h3>
  <!-- Heading content -->
</h3>
```

<h4> Tag: Fourth-Level Heading

The <h4> tag defines a fourth-level heading, which is smaller than the <h3> tag. It's often used for sub-sections within an <h3> section.

Copy

```
<h4>
  <!-- Heading content -->
</h4>
```

<h5> Tag: Fifth-Level Heading

The <h5> tag is used for fifth-level headings and is smaller than <h4> tags. These are rarely used but can be helpful for deeply nested sections.

Copy

```
<h5>
  <!-- Heading content -->
</h5>
```

<h6> Tag: Sixth-Level Heading

The <h6> tag defines the sixth-level heading and is the smallest among all the heading tags. It's rarely used but can serve specific formatting needs.

```
<h6>
  <!-- Heading content -->
</h6>
```

Summary

- **<h1> Tag:** Used for the main title of the page; largest and most prominent heading.
- **<h2> Tag:** Used for major section headings; smaller than <h1> but still quite prominent.
- **<h3> Tag:** Used for sub-sections within an <h2> section; smaller than <h2> but larger than <h4>.
- **<h4> Tag:** Often used for headings within an <h3> section; useful for further breaking down content.
- **<h5> Tag:** Rarely used; suitable for deeply nested sections or less important headings.
- **<h6> Tag:** The smallest heading tag; used for very specific or minor headings, rarely seen in general usage.

Paragraph Tag

To create well-structured text in your HTML document, the <p> tag is essential for defining paragraphs.

<p> Tag: Defining a Paragraph in HTML

The <p> tag is used to format text into distinct paragraphs. Each paragraph element is separated by automatic empty line spaces above and below the content, providing a clear visual separation. The tag must be closed with its corresponding </p> tag.

```
<p>
  <!-- Paragraph content -->
</p>
```

Attributes and Styling

While the <p> tag is straightforward, you can enhance its functionality using various attributes like class or id for CSS styling. You can also include inline styles using the style attribute.

```
<p class="example" style="color: blue;">
  This is a styled paragraph.
</p>
```

Best Practices

It's advisable to use the <p> tag for textual content and not for layout control. For layout purposes, consider using HTML5 semantic tags like <section>, <article>, or CSS techniques.

Horizontal Line Tag

To add a horizontal line in your HTML document, the <hr> tag comes in handy.

How to use the <hr> tag?

The syntax of the hr tag looks something like this.

```
<hr>
```

The <hr> tag is an empty or self-closing tag, meaning it doesn't require a closing tag. It serves as a visual separator, dividing different sections of your document with a horizontal line.

Line Break Tag

To insert a line break in your HTML document, you can utilize the <br> tag.
<br> tag is used to insert line breaks in text or paragraphs

The syntax for the <br> tag looks like this:

```html
<br>
```

The <br> tag is commonly referred to as an empty or self-closing tag, meaning it doesn't require a closing tag. This tag is responsible for breaking text lines or separating paragraphs. When implemented, it automatically moves the text following the tag to the next line.

It's particularly useful in formatting textual content where line breaks are essential for readability or visual layout. For instance, it can be used in addresses, poems, or song lyrics to preserve the original line structure.

Anchor Tag

Links are fundamental to navigating the web. In HTML, links are created using the <a> tag, also known as the Anchor tag.

Key Characteristics of HTML Links

- Specified by the <a> tag.
- Also known as hyperlinks.
- Used to link one document to another.
- Includes a closing tag </a>.

Syntax of HTML Links

```
<a href="Your specified path">
    content
</a>
```

Essential Attributes of the Anchor Tag

HTML links primarily use two attributes:

- href attribute: Defines the URL the link points to.
- target attribute: Specifies where to open the linked document.

Target Attribute Values

- **_blank**: Opens the linked document in a new window or tab.
- **_top:** Opens document in the full body of the window.
- **_self:** Opens document in the same window or tab (default behavior).
- **_parent:** Opens the linked document in the parent frame.

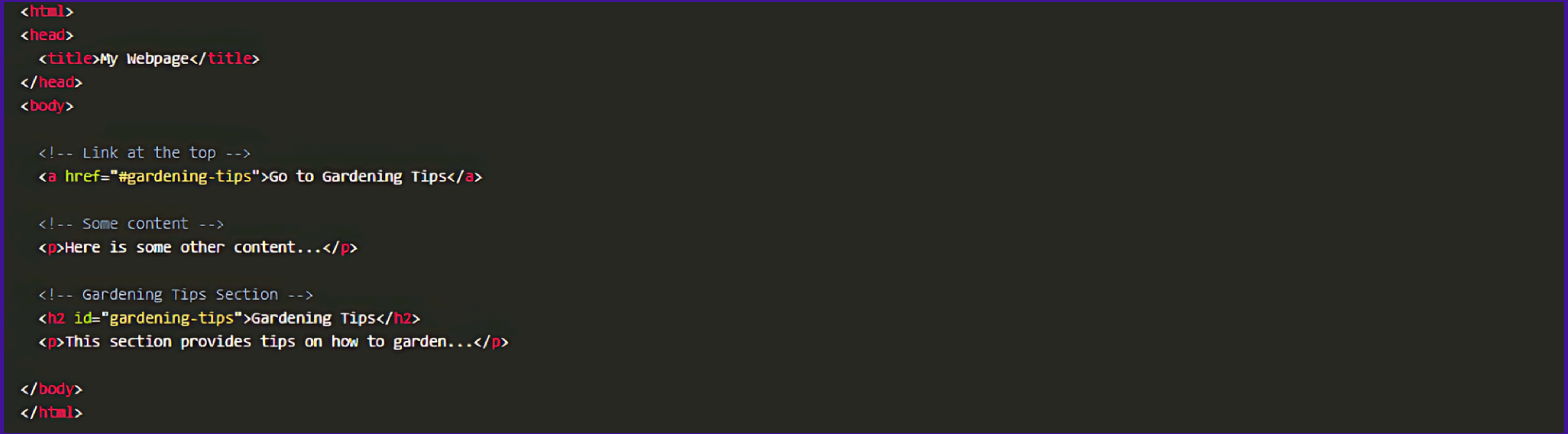
Linking to Specific Page Sections

To link to a specific section of a webpage, you can:

- Use the name or id attribute of the target section.
- Use a hyperlink with a hash (#) followed by the target id or name.

Example

Let's say you have a long webpage with multiple sections, and you want to create a link at the top that, when clicked, takes the user directly to a specific section further down the page. You can do this using HTML links that target specific sections.


```html
<html>
<head>
  <title>My Webpage</title>
</head>
<body>

  <!-- Link at the top -->
  <a href="#gardening-tips">Go to Gardening Tips</a>

  <!-- Some content -->
  <p>Here is some other content...</p>

  <!-- Gardening Tips Section -->
  <h2 id="gardening-tips">Gardening Tips</h2>
  <p>This section provides tips on how to garden...</p>

</body>
</html>
```

Link Colors

Links typically appear in different colors based on their state:

- Active: Displayed in red and underlined like this sentence
- Visited: Appears purple and underlined like this sentence
- Unvisited: Shown as blue and underlined like this sentence

You can customize these colors using CSS to better match the style of your website.

Image Tag

Images play a crucial role in enhancing web pages by providing a visual context that complements textual content. In HTML, the <img> tag is used to embed images into web pages.

Basic Syntax for Embedding Images

This is how the syntax to embed an image in html looks like:

```
<img src="image's path" />
```

Key Features of the <img> Tag

- It's a self-closing tag, meaning it doesn't require a corresponding closing tag.
- Commonly used attributes include the "alt" attribute for image descriptions and the "src" attribute for specifying the image location.
- Supports various image formats including PNG, JPEG, JPG, and GIF.

Setting Mandatory Attributes

- The "src" and "alt" attributes are essential for the proper functioning of the <img> tag.
- **src attribute:** Specifies the path to the image file.
- **alt attribute:** Provides a text description for the image.

```
<img src="images/profile_picture.jpg" alt="Profile Picture" />
```

Note: To find the image's location, right-click on the image, go to properties, and check the location field.

Setting Image Dimensions

Although dimensions can be set using the "width" and "height" attributes in the <img> tag, modern best practices recommend using CSS for this purpose.

```
<img src="image.png" alt="Description" width="300" height="100" />
```

Setting the width and height attributes for images in HTML can have a positive impact on Search Engine Optimization (SEO). Specifying these dimensions in the <img> tag allows browsers to allocate the correct amount of space on a web page even before the image is fully loaded. This prevents layout shifts, improving the Cumulative Layout Shift (CLS) score—a key metric in Google's Core Web Vitals. A better CLS score can lead to a higher page ranking in search engine results.

Note: Styling dimensions and other properties are now more commonly managed through CSS for better flexibility and maintainability.

Pre Tag

The <pre> tag serves as an indispensable tool in HTML for displaying preformatted text, such as code snippets in various programming languages.

What Does the <pre> Tag Do?

The <pre> tag preserves the original formatting of text, making it an excellent choice for displaying code where spacing and indentation are key.

```
<pre>
    <!-- code snippet in any programming language -->
</pre>
```

Key Features

- The <pre> tag maintains both spaces and line breaks, ensuring that text appears exactly as it was originally formatted.
- The <pre> tag has both an opening tag <pre> and a closing tag </pre>.
- Additional attributes can also be added for further customization.

When to Use the <pre> Tag?

The <pre> tag is most effective when you want the text to display on your HTML page exactly as it was typed, without any formatting changes. It is especially useful for displaying code snippets or preformatted text from data files.

Displaying a Simple Python Program and Its Output

In this section, we will use HTML to display a simple Python program that prints 'Hello, World!' to the console. Don't worry, you don't need to know Python; we're just showing how to display the program using the HTML <pre> tag.

Python Program

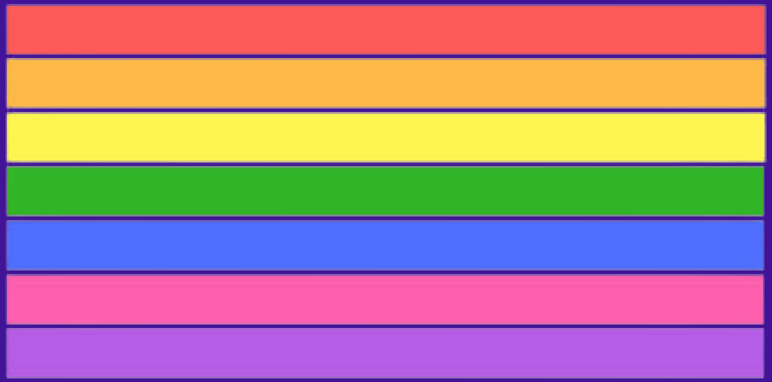
```
<pre>
  # A simple Python program to print "Hello, World!"
def main():
    print("Hello, World!")

if __name__ == "__main__":
    main()
</pre>
```

INLINE & BLOCK ELEMENTS

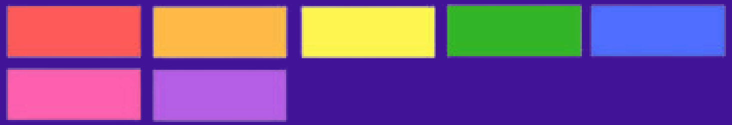
HTML In line Elements

Inline Elements don't start on a new line. It only takes the width required to cover the content. HTML elements are generally divided into two categories: **Block-level** and **Inline** elements.

Block- Level Element



INLINE ELEMENTS:



No matter what the width is, block elements will always start on a new line and take up the full available width of their container by default. This means they essentially claim all the horizontal space for themselves, pushing any content that comes after them to a new line. But the inline elements will fit snugly within the flow of other elements, taking up only as much width as their content requires.

What are Inline Elements?

Inline elements do not start on a new line and only take up as much width as necessary. They are part of the flow within other elements. Inline elements can contain other inline elements, but they generally should not contain block-level elements. For example, you could nest a <strong> (strong emphasis) element within a <span> (a generic inline container) element, like so:

```html
<span>This is <strong>important</strong> text.</span>
```

However, placing a block-level element like a <div> or <p> inside an inline element like <span> or <a> is typically considered incorrect HTML and could lead to unexpected behavior in terms of layout and styling.

```html
<!-- This is generally considered incorrect -->
<span>This is <div>not recommended</div> text.</span>
```

Common Inline Elements

- <span>: A generic inline container for text
- <a>: Defines a hyperlink
- <strong>: Defines important text
- <em>: Defines emphasized text
- <img>: Embeds an image
- <input>: Defines an input control

Examples

Here is an example of using inline elements within a paragraph.
This text contains a link, an important text, and an emphasized text.

Styling Inline Elements

You can use CSS to style inline elements. However, some properties like width and height may not apply.
Here is an exhaustive list of the most used Inline Elements:

- <a>
- <abbr>
- <acronym>
- <button>
- <br>
- <big>
- <bdo>
- <b>
- <cite>
- <code>
- <dfn>
- <i>
- <em>
- <img>

- <input>
- <kbd>
- <label>
- <map>
- <object>
- <output>
- <tt>
- <time>
- <samp>
- <script>
- <select>
- <small>
- <span>
- <strong>
- <sub>
- <sup>
- <textarea>

HTML Block Elements

You already know about HTML inline elements. All HTML tags have specific display behavior: they are either block-level elements or inline elements.

What are Block-level Elements?

Block-level elements are those that start on a new line and take up the entire width of their container by default. They essentially claim all the horizontal space for themselves, pushing any content that comes after them to a new line.

Characteristics of Block-level Elements:

- Always start on a new line.
- Take up the full width available.
- Width and height can be controlled via CSS.
- Can contain other block-level as well as inline elements.

Common Block-level Elements:

- <h1>,<h2>,<h3>,<h4>,<h5>,<h6> - Headings
- <p> - Paragraphs
- <hr> - Horizontal rule
- <address> - Address information
- <article> - Article content
- <aside> - Sidebar content
- <blockquote> - Block quotations
- <canvas> - Drawing area
- <dd> - Description in a description list

- <div> - Generic container
- <dl> - Description list
- <dt> - Term in a description list
- <fieldset> - Group of related form elements
- <figcaption> - Caption for a figure
- <figure> - Image or media with a caption
- <footer> - Footer of a section or page
- <form> - HTML form
- <header> - Header of a section or page
- <li> - List item
- <main> - Main content of a document
- <nav> - Navigation links
- <noscript> - Alternate content when JavaScript is not enabled
- <ol> - Ordered list
- <ul> - Unordered list
- <pre> - Preformatted text
- <section> - Standalone section in a document
- <table> - Table
- <video> - Video content

HTML LISTS

HTML Lists

Our day-to-day lives often involve the use of lists. For example, when we go shopping, the bill we receive includes a list of all the items we've purchased. In a similar manner, web developers use lists to neatly display data on websites.

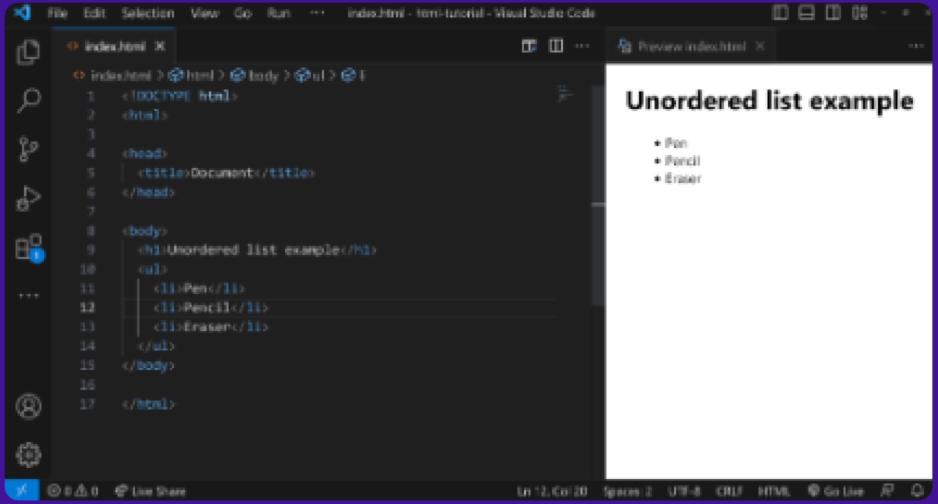
Types of HTML Lists

HTML provides different types of lists to display data in various forms. Each list contains one or more list items.

- **Unordered List:** Displays items using bullets.
- **Ordered List:** Displays items in a numerical sequence, and supports various numbering styles like Arabic numerals, Roman numerals, and so on.
- **Definition List:** Organizes items in a format similar to a dictionary, with terms and their corresponding definitions.

An Unordered List

An unordered list uses bullets to display items. It is suitable for listing items where the order doesn't matter. We will soon explore unordered lists in great detail


File Edit Selection View Go Run ··· index.html - html-tutorial - Visual Studio Code
index.html
Preview index.html
Ln 12, Col 20 Spaces: 2 UTF-8 CRLF HTML Go Live
Live Share

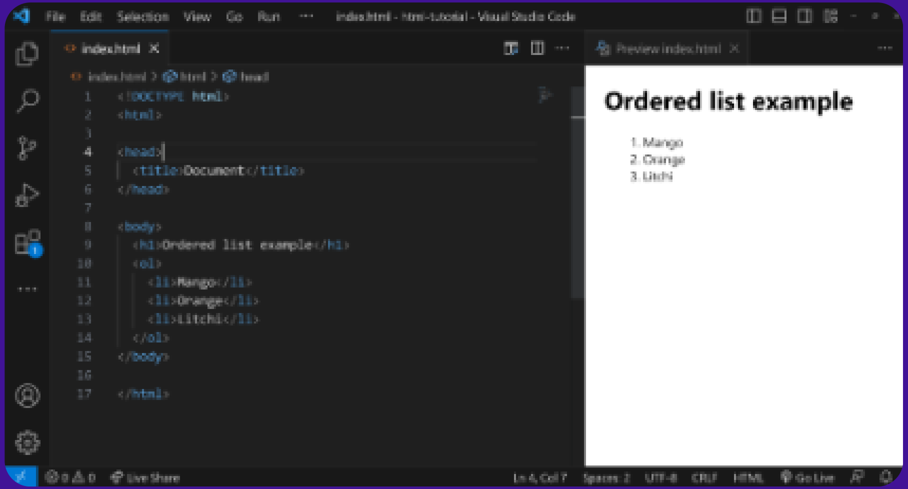
```
<!DOCTYPE html>
<html>

<head>
  <title>Document</title>
</head>

<body>
  <h1>Unordered list example</h1>
  <ul>
    <li>Pen</li>
    <li>Pencil</li>
    <li>Eraser</li>
  </ul>
</body>

</html>
```

Unordered list example

- Pen
- Pencil
- Eraser

An Ordered List

An ordered list uses numbers or other types of markers to indicate the sequence of items. It's ideal for listing steps in a process or ranking items in order of importance. We will soon explore ordered lists in great detail


File Edit Selection View Go Run ··· index.html - html-tutorial - Visual Studio Code
index.html
Preview index.html
Ln 4, Col 7 Spaces: 2 UTF-8 CRLF HTML Go Live
Live Share

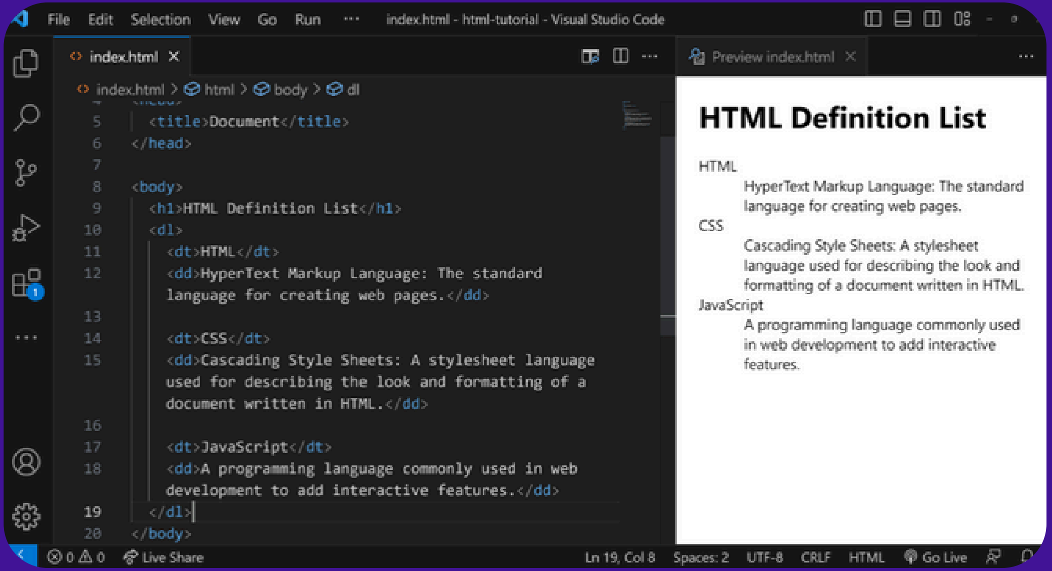
```
<!DOCTYPE html>
<html>

<head>
  <title>Document</title>
</head>

<body>
  <h1>Ordered list example</h1>
  <ol>
    <li>Mango</li>
    <li>Orange</li>
    <li>Litchi</li>
  </ol>
</body>

</html>
```

Ordered list example

1. Mango
2. Orange
3. Litchi

A Definition List

A definition list arranges items in a way similar to a dictionary, with a term followed by its definition. This is useful for glossaries or to display metadata.


File Edit Selection View Go Run ··· index.html - html-tutorial - Visual Studio Code
index.html
Preview index.html
index.html > html > body > dl
Ln 19, Col 8 Spaces: 2 UTF-8 CRLF HTML Go Live
Live Share

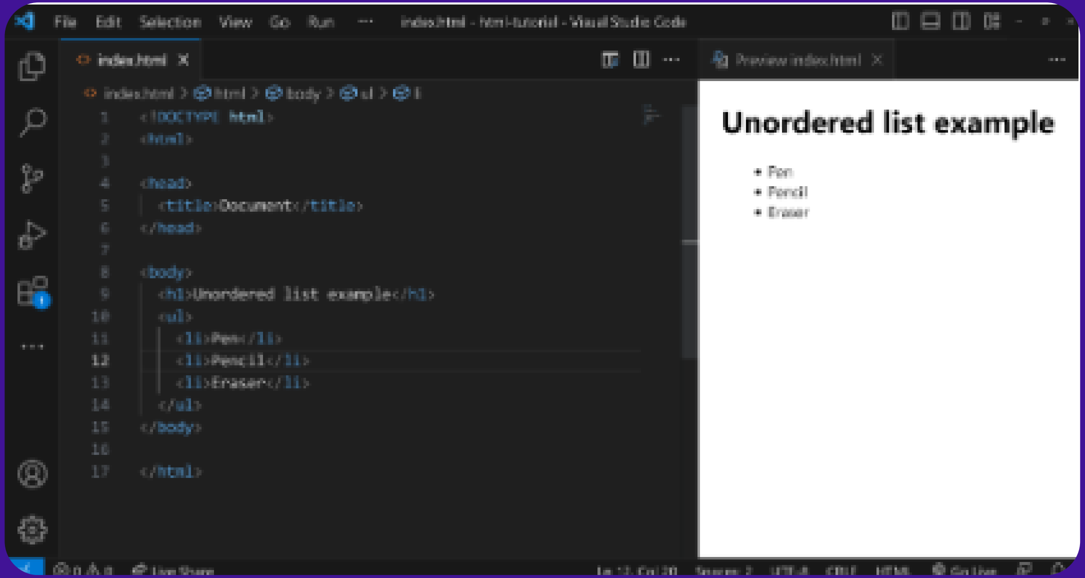
```html
    <title>Document</title>
  </head>

  <body>
    <h1>HTML Definition List</h1>
    <dl>
      <dt>HTML</dt>
      <dd>HyperText Markup Language: The standard language for creating web pages.</dd>

      <dt>CSS</dt>
      <dd>Cascading Style Sheets: A stylesheet language used for describing the look and formatting of a document written in HTML.</dd>

      <dt>JavaScript</dt>
      <dd>A programming language commonly used in web development to add interactive features.</dd>
    </dl>
  </body>
```

```
HTML Definition List

HTML
        HyperText Markup Language: The standard
        language for creating web pages.
CSS
        Cascading Style Sheets: A stylesheet
        language used for describing the look and
        formatting of a document written in HTML.
JavaScript
        A programming language commonly used
        in web development to add interactive
        features.
```

HTML Unordered List

An unordered list is a list of items that are not arranged in any specific, sequential order. Unlike ordered lists, the items in an unordered list are typically marked with bullet points, dashes, or other symbols to indicate list membership, but these markers do not imply any particular order.

Syntax for Creating Unordered Lists



```html
<!DOCTYPE html>
<html>

<head>
    <title>Document</title>
</head>

<body>
    <h1>Unordered list example</h1>
    <ul>
        <li>Pen</li>
        <li>Pencil</li>
        <li>Eraser</li>
    </ul>
</body>

</html>
```

Unordered list example

- Pen
- Pencil
- Eraser

Key Characteristics of Unordered Lists

- No specific sequence is required.
- Typically displayed as bullet points.
- Defined using the <ul> tag.
- Individual items use the <li> tag.

Basic Example

```
<ul>
  <li>Pen</li>
  <li>Pencil</li>
  <li>Eraser</li>
</ul>
```

Output:

- Pen
- Pencil
- Eraser

Customizing Bullet Points with 'type' Attribute

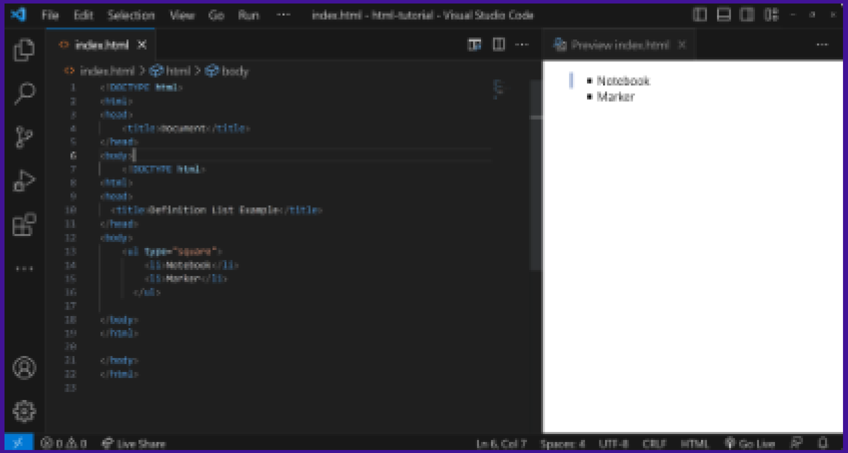
You can specify the style of bullet points using the type attribute. It supports three values:

- disc - default bullet style
- square
- circle

Example Using Square Bullets:

```
<ul type="square">
  <li>Notebook</li>
  <li>Marker</li>
</ul>
```

Output:



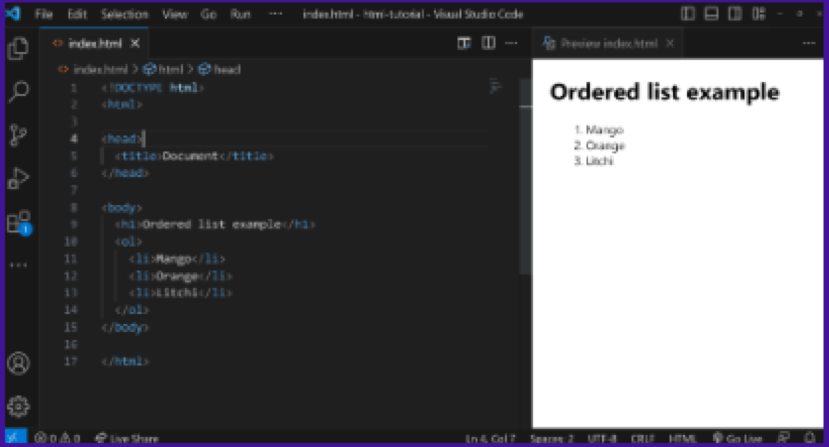
```html
<!DOCTYPE html>
<html>
<head>
    <title>Document</title>
</head>
<body>
    <!DOCTYPE html>
<html>
<head>
  <title>Definition List Example</title>
</head>
<body>
    <ul type="square">
        <li>Notebook</li>
        <li>Marker</li>
     </ul>

</body>
</html>

</body>
</html>
```

```
▪ Notebook
▪ Marker
```

HTML Ordered List

An ordered list is a list of items that are arranged in a specific, sequential order. Each item in the list is usually numbered to indicate its position in the sequence. Ordered lists are commonly used when the sequence of the items is important, such as in step-by-step instructions or rankings.

Syntax



```html
<!DOCTYPE html>
<html>

<head>
  <title>Document</title>
</head>

<body>
  <h1>Ordered list example</h1>
  <ol>
    <li>Mango</li>
    <li>Orange</li>
    <li>Litchi</li>
  </ol>
</body>

</html>
```

Ordered list example

1. Mango
2. Orange
3. Litchi

Key Points

- Ordered lists are used for items that follow a sequence.
- They are created using the <ol> (Ordered List) tag.
- List items are enclosed within <li> (List Item) tags.

Basic Example

```html
<ol>
  <li>Mango</li>
  <li>Orange</li>
  <li>Litchi</li>
</ol>
```

Output:

1. Mango
2. Orange
3. Litchi

Setting the 'type' Attribute

The type attribute specifies the style of numbering. You have several options:

1. **Uppercase Roman Numerals:** Use type="I"
2. **Lowercase Roman Numerals:** Use type="i"
3. **Arabic Numerals:** Use type="1" (This is the default if the type attribute is not specified)
4. **Lowercase Alphabetical Letters**: Use type="a"
5. **Uppercase Alphabetical Letters:** Use type="A"

Setting the 'start' Attribute

The start attribute specifies the starting number for the list.

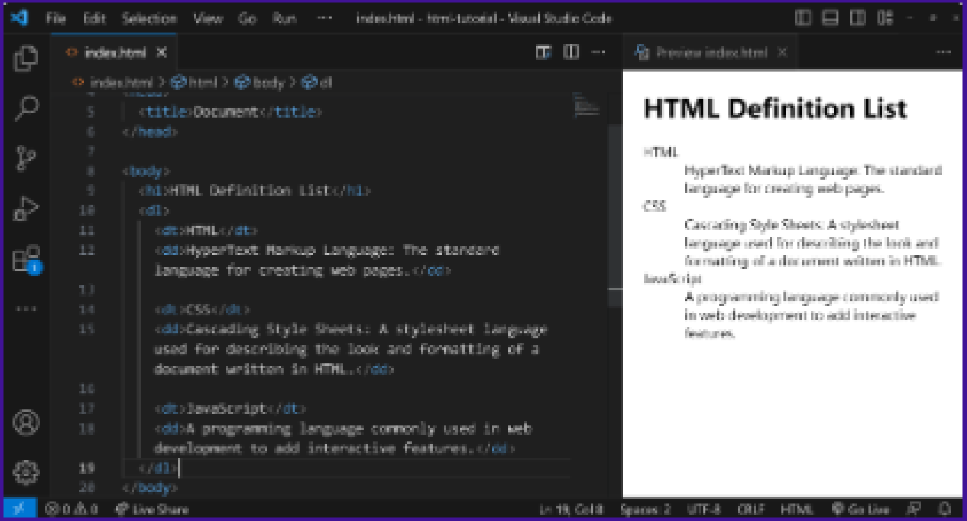
```
<ol type="A" start="3">
  <li>Pen</li>
  <li>Pencil</li>
</ol>
```

Output:

1. Pen
2. Pencil

HTML Definition Lists

A Definition List in HTML is used to represent a list of terms along with their corresponding descriptions or definitions. The Definition List is created using the <dl> (Definition List) element, which wraps around one or more pairs of <dt> (Definition Term) and <dd> (Definition Description) elements.



File Edit Selection View Go Run ··· index.html - html-tutorial - Visual Studio Code
index.html
Preview index.html
index.html > html > body > dl

```
  <title>Document</title>
</head>

<body>
  <h1>HTML Definition List</h1>
  <dl>
    <dt>HTML</dt>
    <dd>HyperText Markup Language: The standard
    language for creating web pages.</dd>

    <dt>CSS</dt>
    <dd>Cascading Style Sheets: A stylesheet language
    used for describing the look and formatting of a
    document written in HTML.</dd>

    <dt>JavaScript</dt>
    <dd>A programming language commonly used in web
    development to add interactive features.</dd>
  </dl>
</body>
```

HTML Definition List

HTML
    HyperText Markup Language: The standard language for creating web pages.
CSS
    Cascading Style Sheets: A stylesheet language used for describing the look and formatting of a document written in HTML.
JavaScript
    A programming language commonly used in web development to add interactive features.

Definition List Example

Here's a simple example to illustrate:

```
<h1>HTML Definition List</h1>
<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language: The standard language for creating web pages.</dd>

  <dt>CSS</dt>
  <dd>Cascading Style Sheets: A stylesheet language used for describing the look and formatting of a document written in HTML.</dd>

  <dt>JavaScript</dt>
  <dd>A programming language commonly used in web development to add interactive features.</dd>
</dl>
```

Understanding the example

In this example:

- <dl> is the container for the list.
- <dt> defines the terms that you want to explain.
- <dd> contains the definitions or explanations for the terms.

HTML TABLES

HTML Tables

HTML tables allow you to arrange data like text, images, and links in rows and columns. You use the <table> tag to start and end a table.

Syntax of HTML Table

```
<table>
  // table content
</table>
```

Key Elements of HTML Table

- <table>: Defines the table itself.
- <tr>: Used for table rows.
- <th>: Used for table headings.
- <td>: Used for table cells (data).

Basic Table Structure

```
<table>
  <tr>
    <th>Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Harry</td>
    <td>100</td>
  </tr>
</table>
```

rowspan and colspan Attributes

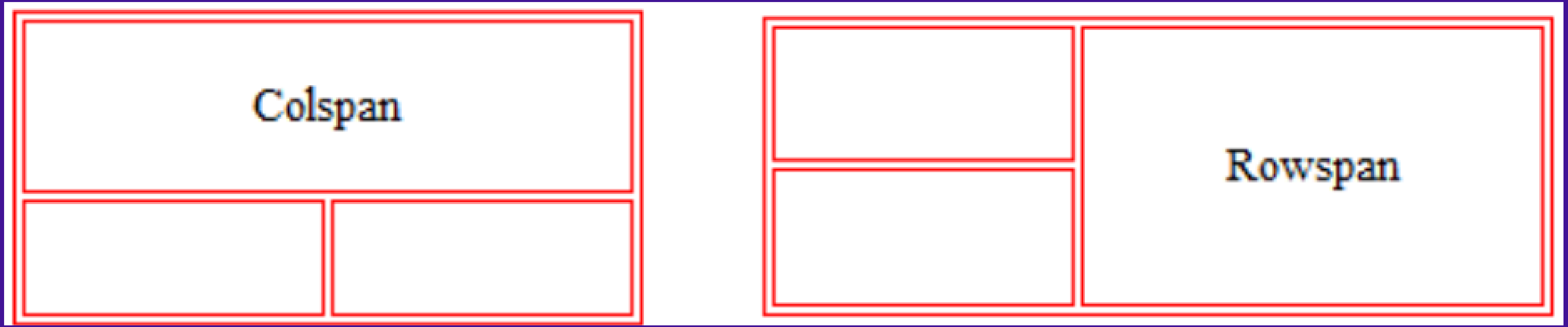
Rowspan: If you want a table cell to span multiple rows, you can use the rowspan attribute.

```
<td rowspan="value">
```

Colspan: If you want a table cell to span multiple columns, you can use the colspan attribute.

```
<td colspan="value">
```

Visual Representation of Rowspan and Colspan



Examples

Here are simple examples to demonstrate the use of rowspan and colspan in HTML tables.

Example for Colspan:

```html
<table border="1">
  <tr>
    <td colspan="2">Merged Columns</td>
  </tr>
  <tr>
    <td>Column 1</td>
    <td>Column 2</td>
  </tr>
</table>
```

Example for Rowspan:

```html
<table border="1">
  <tr>
    <td>Row 1, Column 1</td>
    <td rowspan="2">Merged Rows</td>
  </tr>
  <tr>
    <td>Row 2, Column 1</td>
  </tr>
</table>
```

More on Tables

Let's take a closer look at HTML tables and delve into some more aspects of using tables in HTML.

Adding a Caption

To add a title to your table, you can use the <caption> element. This element helps both in terms of SEO and accessibility.

```html
<table>
  <caption>Student Details</caption>
     <!-- Rest of the table here -->
</table>
```

Table Headers and Footers

Besides <th> for individual header cells, HTML tables allow you to group header or footer content using <thead> and <tfoot>.

```html
<table>
  <thead>
     <!--  header content -->
  </thead>
  <tfoot>
  <!-- footer content -->
  </tfoot>
  <tbody>
  <!-- body content -->
  </tbody>
</table>
```

Column Groups

You can use the <colgroup> and <col> elements to apply styles to an entire column in an HTML table.

```
<table>
  <colgroup>
    <col style="background-color:yellow">
  </colgroup>
  <!-- rest of the table -->
</table>
```

Accessibility in Tables

To make your tables more accessible, you can use the scope attribute in <th> elements to specify if they are headers for columns, rows, or groups of columns or rows.

```
<th scope="col">Name</th>
<th scope="col">Age</th>
```

Sample HTML Table

Here is an example HTML table with all the important elements:

```html
<table border="1">
  <!-- Caption -->
  <caption>Employee Information</caption>

  <!-- Table Header -->
  <thead>
    <tr>
      <th>ID</th>
      <th>Name</th>
      <th>Position</th>
      <th>Salary</th>
    </tr>
  </thead>

  <!-- Table Body -->
  <tbody>
    <tr>
      <td>1</td>
      <td>Alice</td>
      <td>Developer</td>
      <td>$80,000</td>
    </tr>
    <tr>
      <td>2</td>
      <td>Bob</td>
      <td>Designer</td>
      <td>$70,000</td>
    </tr>
    <tr>
```

```
 <td>3</td>
 <td>Carol</td>
 <td>Manager</td>
 <td>$90,000</td>
 </tr>
 </tbody>
 <!-- Table Footer -->
 <tfoot>
 <tr>
 <td colspan="3">Total Employees</td>
 <td>3</td>
 </tr>
 </tfoot>
</table>
```

Conclusion

We've covered some advanced topics related to HTML tables in this blog. By using these features, you can create tables that are not only visually appealing but also highly functional and accessible. Stay tuned for more insights into HTML and web development!

HTML FORMS

Introduction to HTML Forms

HTML forms are essential for collecting user input on web pages. Whether it's a search bar, a login screen, or a multi-field registration form, HTML forms play a key role in web interactions. They enable users to submit data, which can be processed, stored, or returned by a server.

Why Do We Use Forms?

Forms serve as the gateway between the user and the server, allowing for dynamic, interactive web experiences. They are crucial for tasks such as user authentication, data submission, feedback collection, and more. Simply put, forms make websites more engaging and functional.

HTML Forms Structure:

The fundamental structure of an HTML form is encapsulated within the <form> tags. Inside these tags, you'll place various form controls like text fields, checkboxes, radio buttons, and buttons for submitting the form.

```
<form action="/submit" method="post">
  <!-- Text input for username -->
  <label for="username">Username:</label>
  <input type="text" id="username" name="username" required>
  <br><br>

  <!-- Password input -->
  <label for="password">Password:</label>
  <input type="password" id="password" name="password" required>
  <br><br>
```

```html
<!-- Radio buttons for gender -->
 <label>Gender:</label>
 <input type="radio" id="male" name="gender" value="male">
 <label for="male">Male</label>
 <input type="radio" id="female" name="gender" value="female">
 <label for="female">Female</label>
 <br><br>
 <!-- Submit button -->
 <input type="submit" value="Submit">
 </form>
```

How to Use Form Controls?

The <input> tag is commonly used to create form controls. The attributes of this tag define the control's behavior.

```html
<input type="" name="" value="">
```

The **"type"** attribute specifies the type of input control (e.g., text, password, checkbox).
The **"name"** attribute is used for identifying the control, especially when the data is sent to the server.
The "**value"** attribute sets a default value for the control, which the user can overwrite.

HTML Input Types

Input types in HTML forms are the backbone of interactive web applications. They allow users to send information to web servers for various purposes like searching, logging in, or providing feedback. In this blog, we'll explore common HTML input types: text, password, radio, and checkbox.

Text Input

The text input type is the most basic form of input and is widely used for collecting simple text data.

```
<input type="text" name="username" placeholder="Enter your username">
```

In the above example, the placeholder attribute provides a hint to the user about what to enter.

Password Input

The password input type is similar to the text type but hides the characters entered by the user for security reasons.

```
<input type="password" name="password" placeholder="Enter your password">
```

Radio Buttons

Radio buttons are used when you want the user to select only one option from a set of choices.

```
<input type="radio" id="male" name="gender" value="male">
<label for="male">Male</label>
<input type="radio" id="female" name="gender" value="female">
<label for="female">Female</label>
```

Checkbox

Checkboxes allow the user to select multiple options from a set.

```html
<input type="checkbox" id="subscribe" name="subscribe" value="yes">
<label for="subscribe">Subscribe to newsletter</label>
```

More input types

Here is a comprehensive list of input types you can use in html

Input Type	**Description**
text	Allows the user to type a single line of text.
password	Allows the user to type a password.
submit	Represents a button that, when pressed, submits the form.
reset	Represents a button that, when pressed, resets all the form controls to their initial values.
radio	Represents an option in a set of options that are mutually exclusive with each other.
checkbox	Represents an option in a set that may be selected independently of other options.

button	Represents a clickable button.
color	Allows the user to select a color.
date	Allows the user to select a date.
datetime-local	Allows the user to select a date and time with no time zone.
email	Allows the user to enter an email address.
file	Allows the user to select one or more files from their device storage.
hidden	Represents a value that is not displayed but is submitted to the server.
image	Defines an image that acts as a submit button.
month	Allows the user to select a month and year.
number	Allows the user to enter a number.
range	Allows the user to select a number from a range.

search	Allows the user to enter a search query string.
tel	Allows the user to enter a telephone number.
time	Allows the user to select a time.
url	Allows the user to enter a URL.
week	Allows the user to select a week.

Conclusion

Understanding the different types of HTML input is crucial for creating interactive and user-friendly forms. Each input type serves a specific purpose, making it easier to collect, validate, and process user data.

Textarea & Select

In addition to the basic input types, HTML forms offer other controls like textarea and select for richer user interaction. These elements allow for more complex data collection and provide a better user experience. In this blog, we will dive into these form controls and provide examples.

The Textarea Element

The textarea element is used when you need multiline text input from the user. This is particularly useful for comments, reviews, or any other type of input where the length is unpredictable.

```html
<textarea name="comment" rows="4" cols="50">
      Enter your comment here...
</textarea>
```

The rows and cols attributes define the visible dimensions of the textarea.

The Select Element

```html
<select name="fruits">
      <option value="apple">Apple</option>
      <option value="banana">Banana</option>
      <option value="cherry">Cherry</option>
</select>
```

Each option inside the select tag represents an item in the dropdown list.

Combining Textarea and Select

You can combine textarea and select in the same form to capture varied types of user input.

```
<form action="/submit">
      <textarea name="comment" rows="4" cols="50">Enter your comment here...</textarea>
      <select name="fruits">
        <option value="apple">Apple</option>
        <option value="banana">Banana</option>
        <option value="cherry">Cherry</option>
      </select>
      <input type="submit" value="Submit">
</form>
```

Conclusion

The textarea and select elements add another layer of interactivity to HTML forms, allowing for more complex and useful data collection. Understanding how to use these elements effectively can greatly enhance your web application's user interface.

More on forms

HTML forms are the backbone of interactive websites. They allow users to submit data, which can be processed on the server. While we have covered basic input types in previous tutorials, this tutorial aims to delve deeper into form attributes, both common and new HTML5 additions. We'll also look at HTML5 validation attributes to ensure data integrity.

Common Attributes

action

The action attribute specifies the URL where the form data should be sent after submission.

```
<form action="/submit.php" method="POST">
</form>
```

The method attribute defines how data is sent. The two most common methods are GET and POST.

method

The method attribute defines how data is sent. The two most common methods are GET and POST.

```
<form action="/submit.php" method="POST">
</form>
```

name

The name attribute specifies the name for the form element, making it easier to reference in scripts or the server-side code.

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

New HTML5 Attributes

Placeholder

This attribute provides a hint to the user as to what can be entered in the field.

```
<input type="text" placeholder="Enter your username">
```

Required

The required attribute makes a field mandatory to fill out.

```
<input type="text" required>
```

autofocus

The autofocus attribute automatically focuses the cursor on the particular input when the page loads.

```
<input type="text" autofocus>
```

HTML5 Validation Attributes

required

As mentioned above, this attribute makes a field mandatory.

```
<input type="text" required>
```

pattern

The pattern attribute specifies a regular expression that the input must match to be valid.

```
<input type="text" pattern="[a-zA-Z0-9]+">
```

Conclusion

Understanding the different attributes available for HTML forms is crucial for building robust and user-friendly web applications. This tutorial covered both commonly used and new HTML5-specific attributes that enhance functionality and user interaction. Employing these attributes effectively will greatly enhance your web forms.

Miscellaneous Tags

HTML Code Tag

The HTML <code> tag is a powerful element for displaying code snippets within a webpage. It preserves both spaces and line breaks, making it ideal for showcasing code. In this blog post, we'll explore how to use the <code> tag effectively, especially in conjunction with Prism for code highlighting.

What is the <code> Tag?

The <code> tag is a semantic HTML tag that's used for displaying code snippets. It can be used both inline and within a block-level element like <pre>.

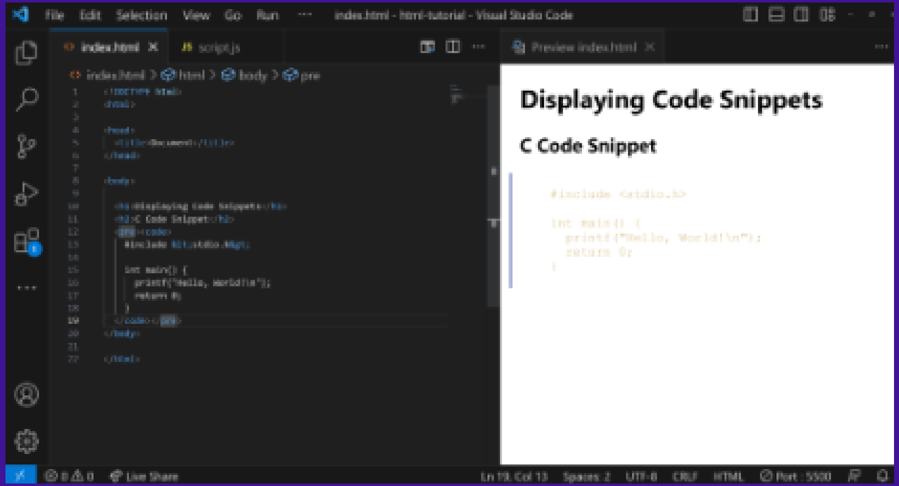
Why Use the <code> Tag?

- **Semantic Meaning:** Provides semantic value to the enclosed code.
- **Readability:** This makes it easier for both browsers and developers to understand that the text is code.
- **Styling:** Easier to style and highlight with CSS or JavaScript libraries like Prism.

Basic Usage

The most straightforward way to use the <code> tag is inline for short code snippets:

```
<code>Your code here</code>
```



```
<!DOCTYPE html>
<html>

<head>
  <title>Document</title>
</head>

<body>

  <h1>Displaying Code Snippets</h1>
  <h2>C Code Snippet</h2>
  <pre><code>
   #include &lt;stdio.h&gt;

   int main() {
    printf("Hello, World!\n");
    return 0;
   }
  </code></pre>
</body>

</html>
```

Displaying Code Snippets

C Code Snippet

```
#include <stdio.h>

int main() {
  printf("Hello, World!\n");
  return 0;
}
```

Using <code> with <pre>

For multiline code snippets, it's common to combine the <code> tag with the <pre> tag:

```
<pre><code>Your multiline code here</code></pre>
```

Conclusion

The HTML <code> tag is a simple yet powerful way to include code snippets in your webpage.

HTML Semantic Tags

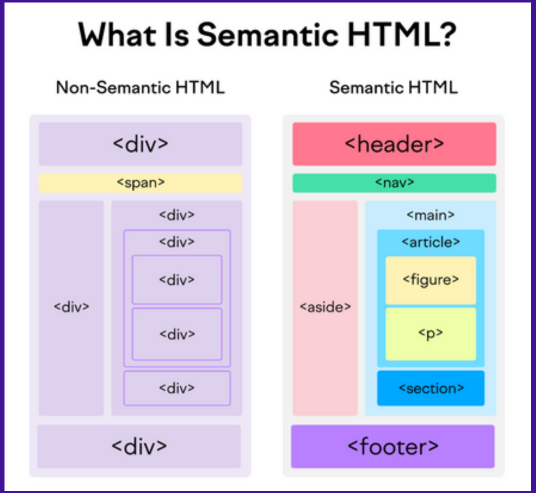
The HTML <code> tag is a simple yet powerful way to include code snippets in your webpage.

What are Semantic Tags?

Semantic tags add meaning to your HTML. They tell both the browser and the developer what kind of content is being presented.

Here are some of the key semantic tags you must know about:

- <header>: Used to represent the top section of a web page, often containing headings, logos, and navigation.
- <nav>: Signifies a navigation menu on a web page.
- <article>: Indicates a self-contained piece of content, such as a blog post or news article.
- <section>: Represents a thematic grouping of content on a web page.
- <aside>: Typically used for sidebars or content that is tangentially related to the main content.
- <footer>: Represents the footer of a web page, usually containing copyright information and contact details.
- <figure> and <figcaption>: Used for embedding images, diagrams, or charts, along with a caption.
- <main>: Signifies the main content area of a web page.
- <time>: Used to represent time-related information, like dates and times.



Why Use Semantic Tags?

They enhance SEO, improve accessibility, and make your code easier to read and maintain.

Commonly Used Semantic Tags

Here are some commonly used semantic tags in HTML:

- **header:** Contains introductory content.
- **footer:** Holds footer information.
- **article:** Encapsulates a self-contained composition.
- **section:** Represents a standalone section.
- **aside:** Contains content aside from the content it is placed in.
- **nav:** Holds navigation links.

Examples

Using the <header> tags and <footer> tags

```html
<header>
    <h1>My Website</h1>
    <nav>
        <ul>
            <li><a href="#">Home</a></li>
            <li><a href="#">About</a></li>
            <li><a href="#">Services</a></li>
            <li><a href="#">Contact</a></li>
        </ul>
    </nav>
</header>

<footer>
    <p>Copyright 2023</p>
</footer>
```

Using the <article> and <section> tags

```html
<article>
    <h2>Article Title</h2>
    <section>
      <p>Content here</p>
    </section>
</article>
```

Using the <aside> and <nav> tags

```html
<aside>
    <p>This is an aside content</p>
  </aside>
<nav>
    <ul>
      <li>Home</li>
      <li>About</li>
    </ul>
</nav>
```

Using the <figure> and <figcaption> tags

```html
<figure>
    <img src="image.jpg" alt="An example image">
    <figcaption>This is an example image.</figcaption>
</figure>
```

Conclusion

Using HTML's semantic tags can greatly benefit both your website's SEO and the maintainability of your code. They offer a way to structure your HTML in a meaningful manner, making your website more accessible and easier to understand.

HTML Canvas

The <canvas> element in HTML is a powerful feature for rendering graphics and shapes directly within web pages. Though it's often paired with JavaScript for dynamic rendering, the canvas itself is an HTML element. In this blog, we'll explore what you can do with the <canvas> element alone.

What is Canvas?

The <canvas> element serves as a container for graphics, which can be rendered via scripting. Essentially, it offers a drawing area for visual content.

Why Use Canvas?

Here are some reasons why you might use the <canvas> element:

- **Graphics:** For drawing shapes, graphs, and other visual representations.
- **Dynamic Content:** To dynamically update visual elements.
- **Interactivity:** Though this involves JavaScript, the canvas element is the foundation for interactive graphical content.

Basic Syntax

Here's how you can define a simple <canvas> element:

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

Attributes of Canvas

While the <canvas> element is simple, it does have a couple of important attributes:

- **width:** Specifies the width of the canvas.
- **height:** Specifies the height of the canvas.

Styling with CSS

You can also style the <canvas> element with CSS. For example, to add a border:

```
canvas {
    border: 1px solid black;
}
```

Conclusion

The HTML <canvas> element serves as a robust foundation for creating graphics and other visual elements on a web page. Even without involving JavaScript, understanding the canvas element and its basic attributes can be quite useful.

HTML Global Attributes

HTML Global Attributes play a crucial role in HTML development, providing a consistent set of attributes that can be applied to any HTML element. In this blog, we'll explore what these attributes are, their descriptions, and how to use them effectively.

What Are HTML Global Attributes?

The global attributes are a set of attributes that can be used with all HTML elements, making them incredibly versatile and essential for dynamic HTML coding.

List of Common Global Attributes

Attribute	**Description**
accesskey	Specifies a shortcut key to activate/focus an element
class	Specifies one or more classes for an element
contenteditable	Specifies whether the content is editable or not

Using Global Attributes

Now, let's look at some examples that illustrate the usage of some of these global attributes.

Using the class Attribute

```html
<div class="container">This is a container.</div>
```

Using the id Attribute

```html
<span id="unique">Unique Element</span>
```

Using the data-* Attribute for Custom Data

```html
<div data-user="123">Custom Data</div>
```

Conclusion

HTML Global Attributes offer a powerful and consistent way to control and manage HTML elements. Understanding these attributes can significantly improve your HTML coding efficiency and the dynamism of the web pages you create.

HTML Entities

HTML entities are a crucial part of HTML markup language. They enable you to display characters that are reserved in HTML or that aren't readily available on the keyboard. In this blog, we'll explore what HTML entities are, their types, and how to use them.

What Are HTML Entities?

HTML entities are used to represent special characters in a format that the browser can understand. They start with an ampersand (&) and end with a semicolon (;).

Why Use HTML Entities?

Here are some reasons:

- **Reserved Characters:** Characters like <, >, and & are reserved in HTML.
- **Special Symbols:** For symbols like ©, ®, or mathematical symbols.
- **Non-Breaking Spaces:** To create white spaces that won't break into a new line

Common HTML Entities

```
&lt;  for <
&gt;  for >
&amp; for &
&nbsp; for a non-breaking space
&copy; for ©
```

How to Use HTML Entities

Entities can be implemented easily within HTML code. Here are some examples:

Using Reserved Characters

```
<p>The price is 10 &lt; 20.</p>
```

Displaying Special Symbols

```
<p>Copyright &copy; 2023.</p>
```

Creating Non-Breaking Spaces

```
<p>This is an example&nbsp;text.</p>
```

Conclusion

HTML entities are essential for rendering special or reserved characters on a web page. Understanding how to use them effectively is key to creating web pages that display content as intended.

Character Sets

Character sets are an essential concept in HTML, influencing how textual content is displayed and interpreted by the browser. This blog aims to elucidate what character sets are, why they matter, and how to specify them in HTML.

What is a Character Set?

A character set is a set of symbols and characters that a computer uses to represent text. In HTML, specifying the correct character set ensures that text is displayed properly across different browsers and platforms.

Why is it Important?

Using the correct character set is crucial for:

- **Accurate Rendering:** To ensure that browsers correctly display your text.
- **Multi-language Support:** To display text in various languages and alphabets.
- **Data Integrity:** To make sure the data sent and received remains consistent.

Specifying Character Set in HTML

The character set is generally specified using the <meta> tag within the <head> section of an HTML document

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

Common Character Sets

Here are some commonly used character sets:

- **UTF-8**: Universal Character Set, 8-bit. It can represent any character in the Unicode standard.
- **ISO-8859-1:** Western Alphabet.
- **ASCII:** American Standard Code for Information Interchange.

Examples

Using UTF-8

```html
<meta charset="UTF-8">
```

Using ISO-8859-1

```html
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
```

Conclusion

Understanding and specifying the correct character set is crucial for creating web pages that render text accurately across different platforms and languages. UTF-8 is the most commonly used and recommended character set due to its wide applicability and support.