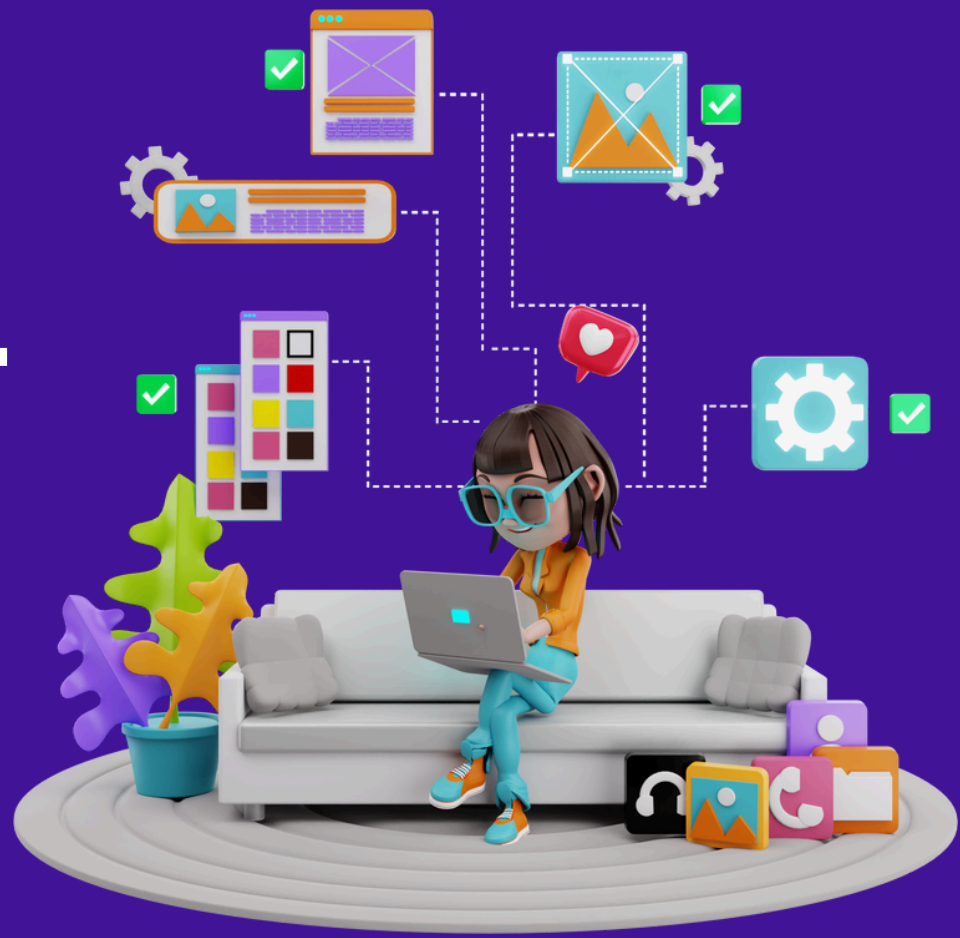
WEB DEVELOPMENT COURSE

LEARN FROM SCRATCH TO ADVANCED

CSS



CSS INTRODUCTION

CSS, or Cascading Style Sheets, is a styling language used to define the look and formatting of a document written in HTML. It allows developers to separate content from design, making it easier to maintain and update websites. With CSS, you can control layout, colors, fonts, and overall visual presentation.
CSS works by associating rules with HTML elements. These rules are applied through selectors, properties, and values. For example, you can use a selector to target all paragraphs (`p`) and set their text color to blue using `color: blue;`. CSS can be included directly in HTML files using `<style>` tags, linked as an external stylesheet using the `<link>` tag, or applied inline within HTML elements.

Key features of CSS include the box model, which defines the space around elements, and responsive design techniques like media queries, which allow the creation of adaptable layouts for different screen sizes. By mastering CSS, you can create visually appealing and user-friendly websites.

WHAT IS CSS?

CSS stands for Cascading Style Sheets. It is a stylesheet language that is used to describe the visual presentation of a web page written in HTML (Hypertext Markup Language).

CSS (Cascading Style Sheets) is a stylesheet language used to describe the presentation of a document written in HTML or XML. CSS controls the layout, colors, fonts, and overall visual appearance of web pages. By separating content (HTML) from design (CSS), it allows for more flexibility and maintainability in web development.

CSS can be applied inline within HTML elements, internally within a `<style>` tag in the HTML document, or externally by linking to a separate .css file. This separation enables developers to make global style changes across multiple pages by editing just one CSS file. CSS also supports responsive design, ensuring web pages look good on different devices and screen sizes.

WHAT IS CSS?

CSS stands for Cascading Style Sheets. It is a stylesheet language that is used to describe the visual presentation of a web page written in HTML (Hypertext Markup Language).

CSS (Cascading Style Sheets) is a stylesheet language used to describe the presentation of a document written in HTML or XML. CSS controls the layout, colors, fonts, and overall visual appearance of web pages. By separating content (HTML) from design (CSS), it allows for more flexibility and maintainability in web development.

CSS can be applied inline within HTML elements, internally within a `<style>` tag in the HTML document, or externally by linking to a separate .css file. This separation enables developers to make global style changes across multiple pages by editing just one CSS file. CSS also supports responsive design, ensuring web pages look good on different devices and screen sizes.

WHY THE WORD "CASCADE"?

The term "cascade" in CSS encapsulates the intricate process by which styles are systematically applied to HTML elements, prioritizing rules in a hierarchical manner.

This concept is pivotal because CSS allows multiple style rules to potentially target the same element, necessitating a methodical approach to resolve conflicts and determine the final appearance of the webpage.

At its core, the cascade operates on several key principles. First, specificity governs which rule holds precedence. CSS assigns a specificity value based on the type of selector used—inline styles, IDs, classes, elements—where more specific selectors override less specific ones.
This ensures that more targeted styling directives, such as those applying to particular classes or IDs, take precedence over general rules.

Specifically, the cascade in CSS works as follows:

- **Specificity**: This metric quantifies the relevance of a style rule based on the specificity of its selectors. Higher specificity grants a rule greater influence over lower-specificity rules targeting the same element.

- **Source Order**: When selectors possess identical specificity, the order in which they appear in the stylesheet or document plays a decisive role. Styles declared later or applied inline within the HTML typically override previous declarations for the same element.

- **Importance**: Using !important within a style rule conveys utmost precedence, superseding all other declarations, irrespective of specificity or order. This feature is reserved for exceptional cases due to its potential to complicate maintenance and troubleshooting.

WHY USE CSS?

CSS, or Cascading Style Sheets, is fundamental to modern web development for several compelling reasons. Firstly, CSS enables the separation of content and presentation within web pages written in HTML. This separation enhances the maintainability and flexibility of websites by allowing developers to alter the visual appearance of a site without needing to modify its underlying structure. Moreover, CSS promotes consistency across a website by facilitating the application of consistent styles to multiple pages from a single stylesheet. This not only saves development time but also ensures a cohesive user experience.

Furthermore, CSS supports responsive web design, enabling websites to adapt smoothly to different devices and screen sizes. By employing CSS media queries and flexible layout techniques, developers can create designs that are not only visually appealing but also functional across a range of devices, from desktops to smartphones. Additionally, CSS provides extensive customization options, allowing developers to style elements precisely as desired, from fonts and colors to layout and animations.

Analogy to Understand CSS

Imagine reading a book with only plain text. It's quite dull, isn't it? Now picture a book enriched with different colors, fonts, and styles. That's what CSS does to a webpage.

How does CSS work?

CSS operates by selecting HTML elements and applying styles to them. Styles dictate the appearance of elements on a webpage. You can target HTML elements, classes, or IDs, defining properties like colors, fonts, margins, etc.

```
/* Example of a CSS rule */
selector {
    property: value;
}
```

QUICK FOLLOW-UP TASK:

- Right-click on the screen and select "Inspect" or press (Ctrl + Shift + C).
- Click on the arrow icon in the top-left corner of the inspection pane.
- Select elements on the page to toggle the CSS rules.

KEY FEATURES OF CSS:

- Styles and layouts of web pages.
- Works alongside HTML and XML documents.
- Enables responsive design for different screen sizes.
- Supports interactive effects like hover states and animations.
- CSS is now modularized, with ongoing updates rather than version numbers.
- Allows reusability of the same rules across multiple HTML documents.

YOUR FIRST CSS WEBSITE

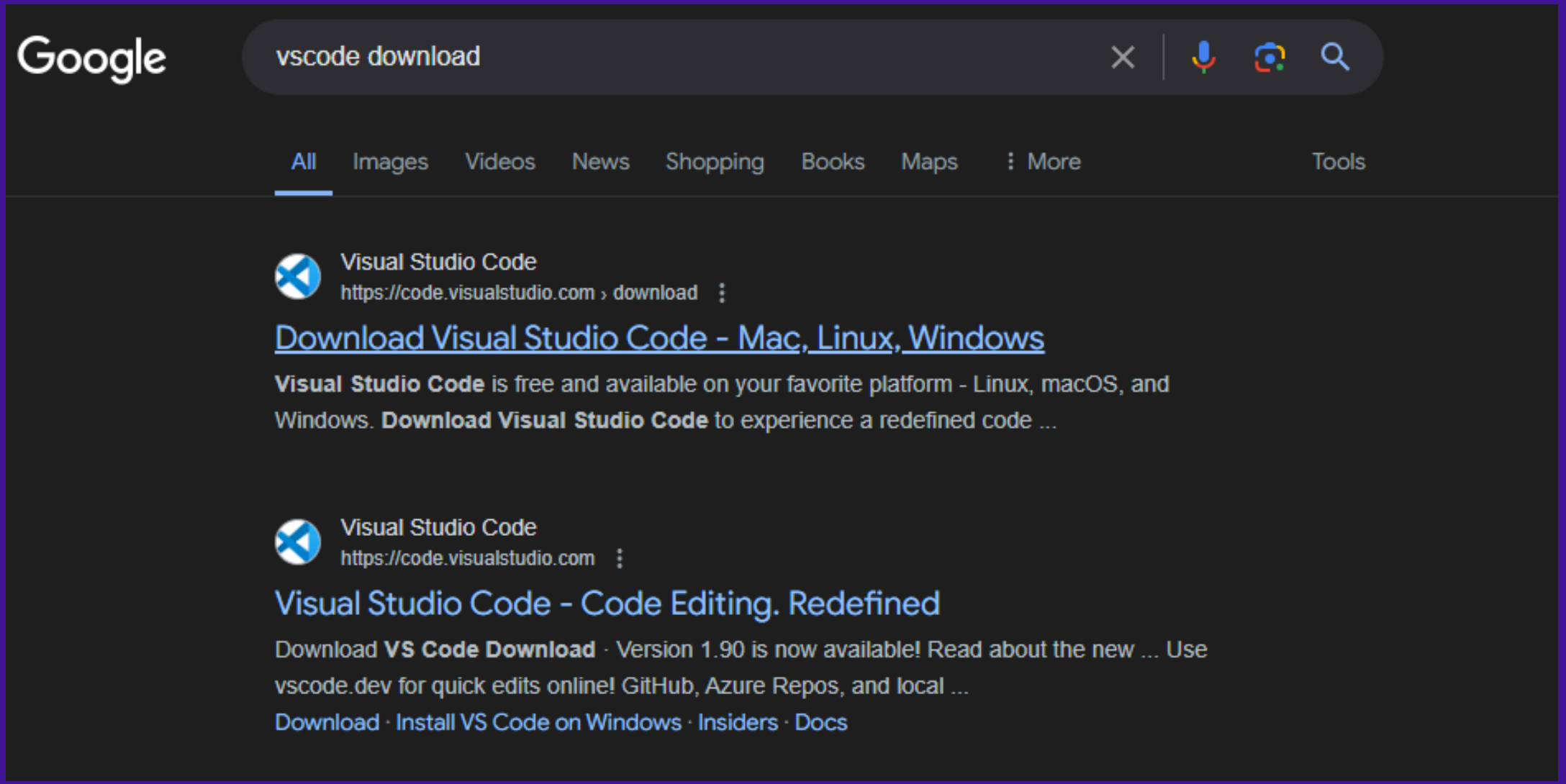
In this course, we will focus on creating our first CSS website.
We will divide the tutorial into the following parts:

- Setting up the environment.
- Creating an HTML page.
- Adding CSS to HTML.

SETTING UP THE ENVIRONMENT

Installing VS Code

- Go to Google using your preferred browser.
- Type Visual Studio Code download in the search bar.
- Click on download and follow the installation instructions for your OS.
- Here's a quick video showing the VS Code installation process:

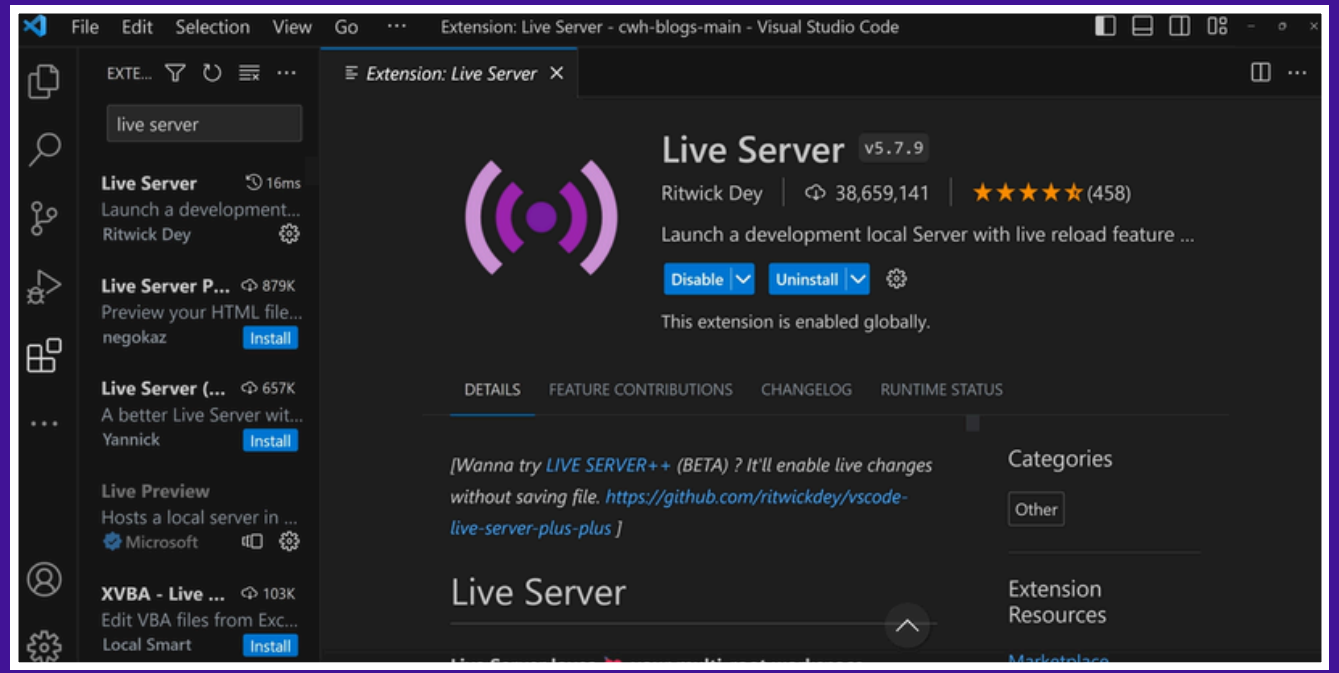

Google
vscode download
All
Images
Videos
News
Shopping
Books
Maps
More
Tools
Visual Studio Code
https://code.visualstudio.com › download
Download Visual Studio Code - Mac, Linux, Windows
Visual Studio Code is free and available on your favorite platform - Linux, macOS, and Windows. Download Visual Studio Code to experience a redefined code ...
Visual Studio Code
https://code.visualstudio.com
Visual Studio Code - Code Editing. Redefined
Download VS Code Download · Version 1.90 is now available! Read about the new ... Use vscode.dev for quick edits online! GitHub, Azure Repos, and local ...
Download · Install VS Code on Windows · Insiders · Docs

INSTALLING THE LIVE SERVER EXTENSION

We will also install the Live Server extension in Visual Studio Code to enable live reloading of pages.

The Live Server extension launches a local development server and reloads the page live as you make changes.

Here's a quick video showing how to install the Live Server extension:

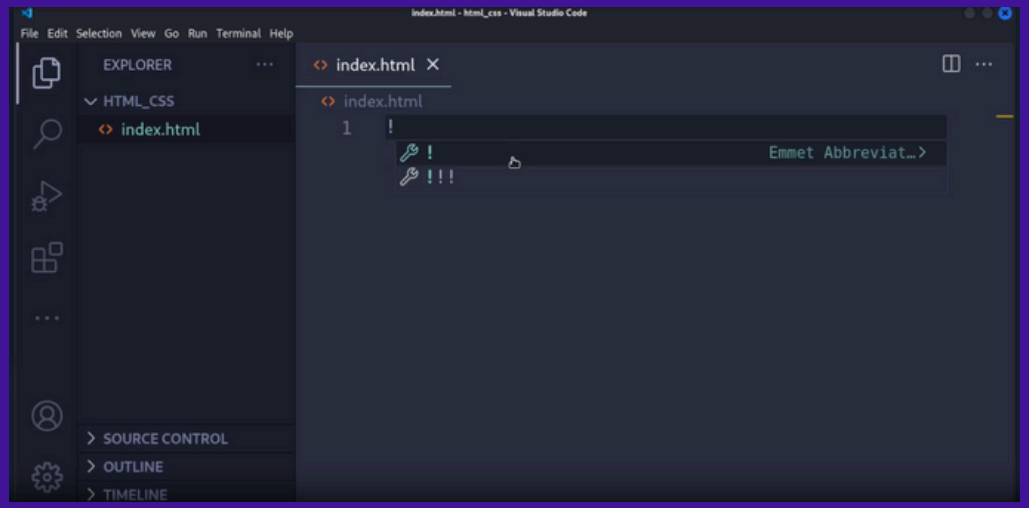


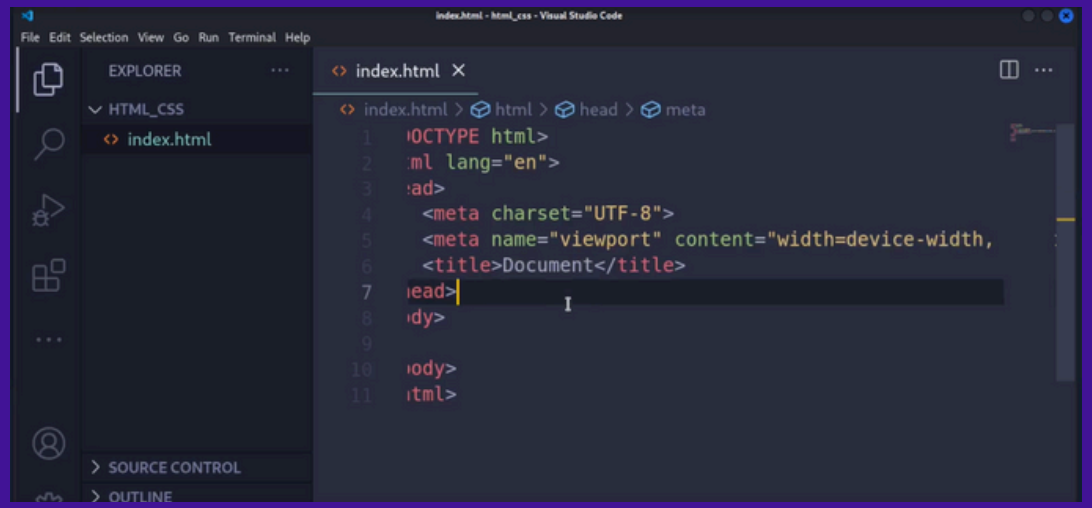
CREATING AN HTML PAGE

If you're not comfortable with HTML, you can start by following this HTML Tutorial. Assuming you have some knowledge of HTML, open VS Code and create a new file named `index.html`.

After creating the HTML file, type "!" and press enter.

This function simplifies the process by automatically generating the default HTML boilerplate code. It streamlines the initial setup of a new web page, ensuring you have the essential structure in place. This includes necessary elements such as the <!DOCTYPE> declaration, <html>, <head>, and <body> tags, saving you time and effort in setting up the foundational framework for your website.





The code needs updates such as revising the title and adding more content to the body section. These changes aim to create a structure where the title meets specific needs, and the body includes additional relevant information. The final code should demonstrate a clear and relevant format that matches the updated title and enriched body text, improving the overall clarity and relevance of the output.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>First CSS Website</title>
</head>
<body>
    I'm learning CSS with my favorite CodeWithHarry.
</body>
</html>
```

Adding CSS to HTML

Now that we've set up the environment and created an HTML page, it's time to add some CSS to enhance its appearance. To do this, insert the following code within the head tag of your HTML document. This code will provide styling instructions that define how the various elements on your page should look, making your web page more visually appealing and user-friendly.

```
<style>
  body {
    text-align: center;
    color: white;
    background-color: purple;
  }
</style>
```

You've successfully begun your journey into the world of CSS! As you follow these tutorials, you'll gain a better understanding of various CSS properties.

How CSS works?

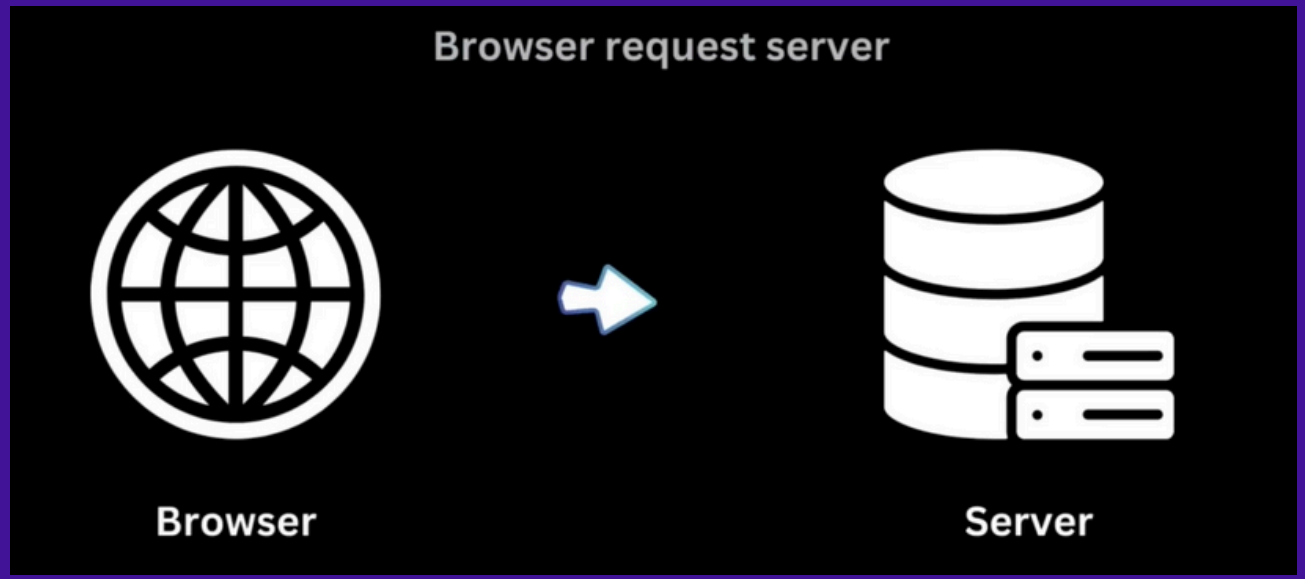
We wrote our first CSS style but still, things wouldn't be very much clear. Let's look at how CSS works on the DOM model.

How CSS Works?

The following steps will help us understand more about CSS working :

- The user types the URL and clicks enter.
- The browser makes a fetch request to the server.
- HTML is fetched from the server.
- HTML is converted into a DOM. In the DOM, each tag is considered a node.
- The browser fetches all the related files and assets that are linked to that HTML, such as external CSS, fonts, images, etc.

- The browser then parses the CSS and groups it based on the selectors, which can be tags.
- Each CSS is attached to its respective node. In this phase, CSS gets attached to its respective node. This is called a render tree.
- The render tree is the well-structured, well-arranged DOM node that will appear on the screen.
- The well-structured, custom-designed website is presented on the screen. This is called painting.



What is a DOM?

A DOM (Document Object Model) is a tree structure representing all the tags and elements on a web page. Every component, such as headings, paragraphs, images, and buttons, forms part of this tree. It's like a blueprint that web browsers use to interpret and display web content.

Here's an example:

The tags are converted into nodes. Each node establishes a parent-child relationship between each other. To be precise, Document Object Model(DOM) is a sort of API that represents and interacts with HTML documents.

Syntax of CSS

CSS follows a rule-based structure. Each rule consists of a selector and a declaration block. Selectors pick the HTML elements, while declaration blocks contain pairs of properties and values.

The general syntax for writing CSS.

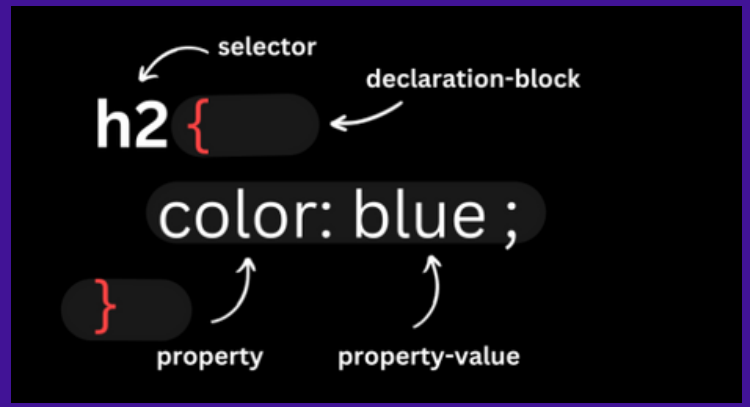
```
selector {
    property: value;
}
```

Note: Semi-colon (;) at the end of each new property and property value is IMPORTANT.
For example:

```
h2{
    color: blue;
}
```

In the example:

- h2: h2 is the selector.
- color: It's the property.
- blue: The property value.



Within the declaration block, there can be multiple pairs of properties and values.
Consider the example:

```
button{
    color: white;
    background-color: black;
    border: transparent;
    border-radius: 5px;
}
```

Ways to add CSS

There are three different ways to add CSS to an HTML page, which are:

1. Inline CSS
2. Internal CSS
3. External CSS

- **Inline CSS**

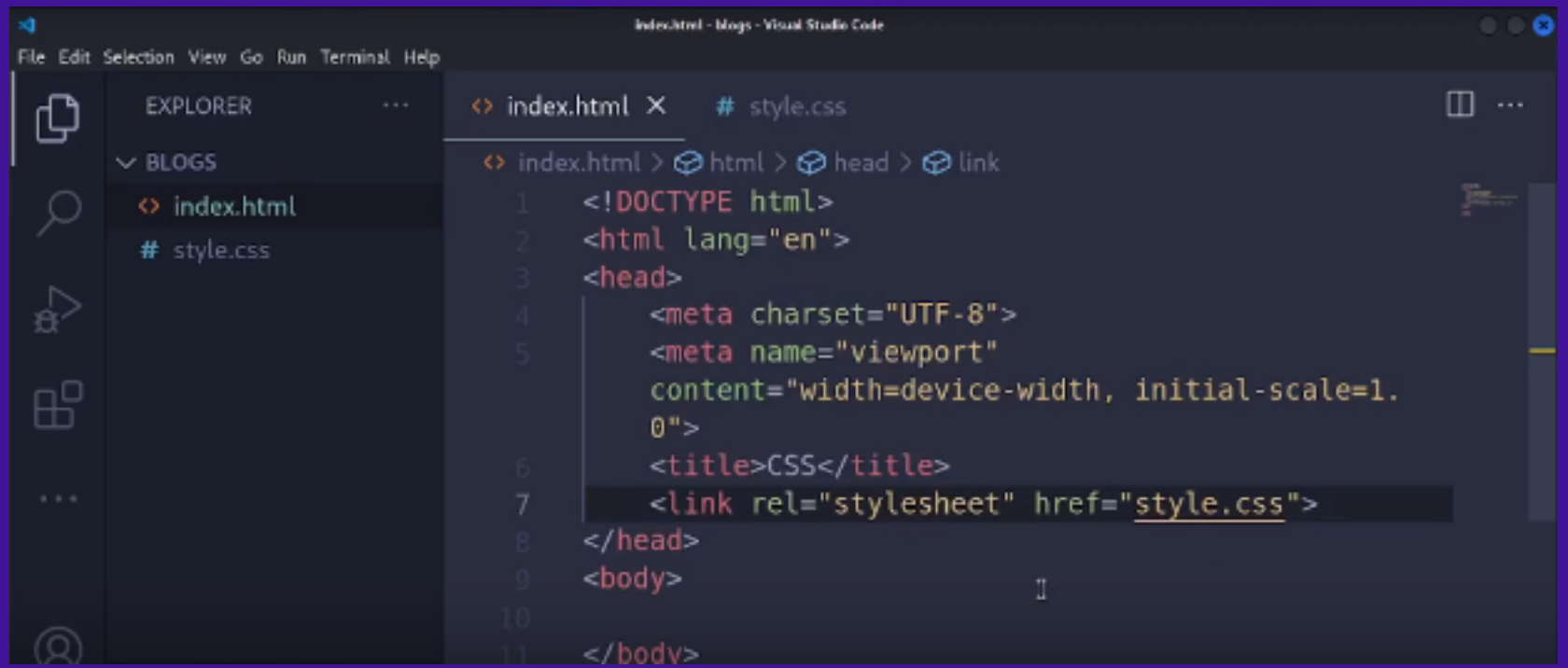
  Inline CSS is used to add custom properties to specific elements. The added style will only reflect on that particular element only.
  To use inline CSS, Insert the "style" attribute within the HTML element's opening tag.
  Consider the code snippet:

```
<h1 style="color: purple;">I'm harry</h1>
<h2>I'm CodeWithHarry</h2>
```

- **External CSS**

  External CSS works similarly to internal CSS but with a twist. Instead of adding the styles within the HTML file, we create a separate file with .css extension. This file will hold all the styling details. Then, we link this file to the HTML page, giving it the instructions on how to look.

  The following video will explain, how to link a CSS file to HTML.



There is a new <link> tag in the head section, and this link tag has rel and href properties.
The following points will explain each keyword's meaning:

1. <link>: This tag is used to create links between different resources, like stylesheets, fonts, and more. In our case, we are using a link tag to link the CSS file with the HTML file.
2. rel="stylesheet": rel stands for relationship, this defines the type of relationship between the HTML document and the linked resource. When set to "stylesheet", it specifies that the linked resource is a stylesheet that will be used to style the HTML content.
3. href="style.css" : The href attribute stands for "hypertext reference." It specifies the path or URL to the external resource we want to link. In this case, it's the path to the external CSS file called "style.css".

Consider the code snippets:

```
<head>
    <title>CodeWithHarry</title>
    <link rel="stylesheet" href="style.css">
</head>

<body>
    <p>I'm harry, from CodeWithHarry</p>
    <p>I'm a Developer and founder of CodeWithHarry.com</p>
</body>

</html>
```

- **CSS Selectors**

What are CSS Selectors?

CSS selectors are patterns used to select and style HTML elements. They specify which elements a CSS rule applies to. Common types include element selectors (e.g., `p` for paragraphs), class selectors (e.g., `.className` for elements with a specific class), ID selectors (e.g., `#idName` for a unique element), and attribute selectors (e.g., `[type="text"]` for elements with a specific attribute). Selectors can be combined and nested to target elements precisely, enabling complex and flexible styling.

There are different types of CSS selectors, which are as follows:

1. Universal Selector
2. Element Selector
3. Id Selector
4. Class Selector
5. Group Selector

- **Universal Selector**

Universal selector represented by "*" targets all the HTML elements on the page. The syntax of Universal Selector is as follows:

```
* {
    property : value;
}
```

```
<html>

<head>
  <style>
    * {
      color: purple;
      text-align: center;
    }
  </style>
</head>

<body>
  <p>Welcome to </p>
  <h1>CodeWithHarry</h1>
</body>

</html>
```

- **Element Selector (Type Selector)**

The element selector selects the target element based on the specific type. Suppose you want to underline all the <p> tags; in this case, the element selector will be the best choice.
The syntax of Element Selector is as follows:

```
p {
    property : value;
}
```

A selector can be any HTML tag. Here, we have considered the p tag.
Consider the cod snippet:

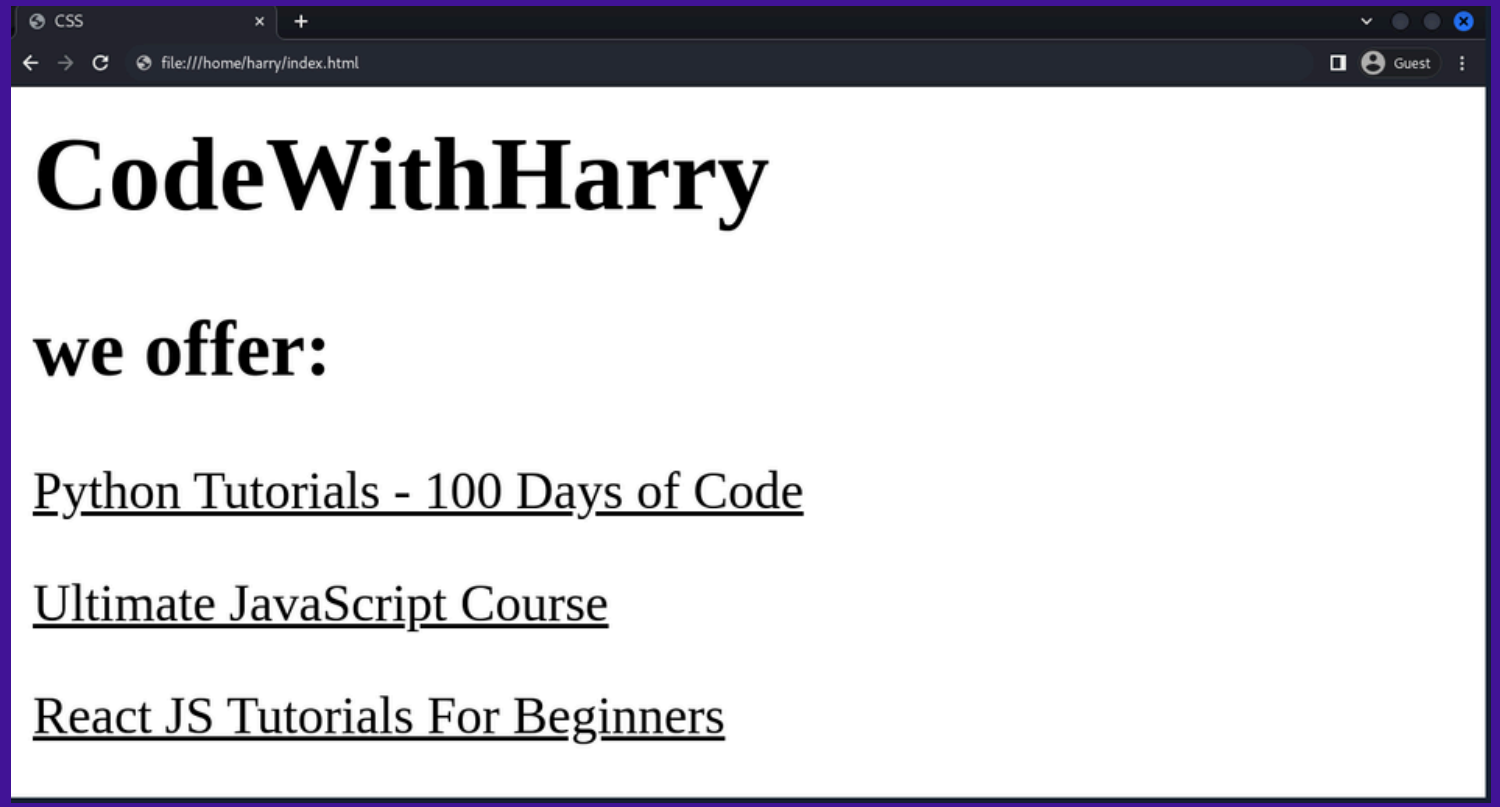
```
<html>

<head>
 <title>CSS</title>
 <style>
 p{
 text-decoration: underline;
 }
 </style>
</head>

<body>
 <h1>CodeWithHarry</h1>
 <h2>we offer: </h2>
 <p>Python Tutorials - 100 Days of Code</p>
 <p>Ultimate JavaScript Course</p>
 <p>React JS Tutorials For Beginners</p>
</body>

</html>
```

Output:



Note: Element selector is not recommended as the same tag can be used multiple times in the document. So, overlapping rules can occur in the stylesheet.

- **ID Selector**

The ID selector targets the elements based on the specific ID. It is written with the hash “#” character followed by the ID name in the style sheet.
The syntax of ID Selector is as follows:

```
#ID {
    property : value;
}
```

Consider the code snippet:

```
<html>

<head>
   <style>
      #title {
         text-align: center;
         color: red;
      }
   </style>
</head>

<body>
   <h1 id="title">CodeWithHarry</h1>
   <p>I'm a Developer and the founder of CodeWithHarry.com</p>
</body>

</html>
```

In the style block, the selector #title, will only target the HTML element having an ID of "title". Consider the output of the above code:

Notice, how the property color: red is only applied to <h1> tag.

- # CSS Comments

Comments help with documentation and are helpful for the future users who read that code, so that they can understand it easily.
Comments are ignored by the browser and Comments don't affect the styling or layout.
CSS Comments are enclosed between /* and */.
There are two types of comments in CSS:

- **Single-line comment:**

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

- **Syntax**

```
selector{
    /* property: value */
}
```

for example:

```
/* This is a single line commnent */
p{
    /* color: red */
}
```

here, the comment is between /* and */

- **Multi-line comments**

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

- **Syntax**

```
selector{
    /* property1: value1
    property1: value1
    property1: value1 */
}
```

These are similar to single-line comments, but this helps to comment large descriptions, notes, etc.

```
/* This is a
multi line
commment */
p {
    /* color: red;
    background-color: purple; */
    color: purple;
    background-color: red;
}
```

CSS Specificity

CSS Specificity helps determine what style will be applied to the HTML element(s) when there are overlapping or multiple CSS rules.
It is a value or weight assigned to a CSS selector. The higher the specificity, the more precedence the selector has.
Let's consider the following code

```
<html>
<head>
  <style>
    *{
      color: gray;
    }
    #title{
      color: red;
    }
    .h1{
      color: blue;
    }
    h1{
      color: pink;
    }
  </style>

</head>
<body>
  <h1 id="title" class="h1" style="color:purple">CodeWithHarry</h1>
</body>
</html>
```

Here the question is, what color will h1 be assigned to? This is calculated based on the selector's components

- **The Cascade Algorithm**

CSS stands for Cascading Stylesheets. The cascade is the algorithm for solving conflicts where multiple CSS rules apply to an HTML element. It's the reason that the text of the button styled with the above CSS will be purple.

Understanding the cascade algorithm helps you understand how the browser resolves conflicts like this. The cascade algorithm has 4 distinct stages.

1. Position and order of appearance: the order in which your CSS rules appear
2. Specificity: an algorithm that determines which CSS selector has the strongest match
3. Origin: the order in which CSS appears and where it comes from, whether that is a browser style, CSS from a browser extension, or your authored CSS
4. Importance: some CSS rules are weighted more heavily than others, especially with the important rule type

- **Effect of Position**

  If there are two rules that have selectors of identical specificity, the last one to be declared won. In an HTML page, you can add styles in different ways: through a <link> tag, a <style> tag, or directly in the element's style attribute. If you have one <link> tag at the top of your page and another at the bottom, the styles from the bottom one will be used. The same goes for <style> tags; the ones lower down on the page take priority.

  An inline style attribute with CSS declared in it will override all other CSS, regardless of its position, unless a declaration has !important defined.

  If the browser doesn't support a property, it is ignored

  Specificity
  CSS specificity determines which style rules get applied to an element when there are conflicts. Higher specificity means the style will be used. It's calculated based on a point system involving inline styles, IDs, classes, and tag names.

- **Inline Styles**

  Inline styles have the highest specificity and will always override styles if other selector(s) are also defined

```
<html>
<head>
  <style>
    *{
      color: gray;
    }
    #title{
      color: red;
    }
    .h1{
      color: blue;
    }
    h1{
      color: pink;
    }
  </style>

</head>
<body>
  <h1 id="title" class="h1" style="color:purple">CodeWithHarry</h1>
</body>
</html>
```

- **ID Selector**

  The ID selector also has high specificity but comes after the inline Style specificity. So, if we have an ID and other selectors except the inline style, then the element will take the ID selector properties and values

```
<head>
  <style>
    *{
      color: gray;
    }
    #title{
      color: red;
    }
    .h1{
      color: blue;
    }
    h1{
      color: pink;
    }
  </style>
</head>
<body>
  <h1 id="title" class="h1">CodeWithHarry</h1>
</body>
</html>
```

- **Class and Attribute Selectors**

  Class selectors and attribute selectors have moderate specificity. Suppose the element has a class or attribute selector and not an inline style or ID selector, then the element will take properties and values from the class or attribute selector.

```
<head>
  <style>
    *{
      color: gray;

    .h1{
      color: pink;
    }
  </style>

</head>
<body>
  <h1 class="h1">CodeWithHarry</h1>
</body>
</html>
```

- **Element Selector:**

Element selectors like <p>, , <a>, etc. have low specificity.

```
<head>
  <style>
    *{
      color: gray;
    }
    h1{
      color: tomato;
    }
  </style>

</head>
<body>
  <h1 class="h1">CodeWithHarry</h1>
</body>
</html>
```

Here, you can see color:tomato is applied to the h1 tag.

- **Universal Selector:**

  Universal selectors (*) and combining selectors like not, first-child, and last-child have the lowest specificity

```
<head>
  <style>
    *{
      color: gray;
    }
  </style>

</head>
<body>
  <h1 class="h1">CodeWithHarry</h1>
</body>
</html>
```

Here, you can see color:gray is applied to the h1 tag.
So, the order of specificity is:
Inline Style > ID Selector > Class or Attribute Selector > Element Selector > Universal Selector

- **Specificity Calculation**

  To calculate specificity, assign a value to each part of the selector:
  - Universal Selector: 0
  - Element selectors and pseudo-elements: 1
  - Class selectors, attribute selectors, and pseudo-classes: 10
  - ID selectors: 100
  - Inline styles: 1000

  Then, add up the values of all the parts in the selector.
  Here is an example

```
<h1 id="title" class="h1">CodeWithHarry</h1>
```

  Here, the specificity value will be 111 because ID has a specificity of 100, the class has a specificity of 10, and the h1 element has a specificity of 1.
  In the case of a specificity tie, the rule that appears last in the CSS is applied.

- **Importance**

The !important flag in CSS is used to give a particular style rule the highest level of importance, overriding any other competing styles. When you add !important to a CSS rule, it will take precedence over other rules affecting the same element, regardless of their specificity. For example, if you have:

```
p {
  color: red !important;
}

p {
  color: blue;
}
```

The text in the <p> element will be red, not blue, because the !important flag makes that rule more important.
an !important at the end of a CSS value gets a specificity score of 10,000 points. This is the highest specificity that one individual item can get.
However, it's generally best to use !important sparingly, as it can make debugging and maintaining your stylesheets more complicated. It can override styles in ways that are hard to trace, leading to unexpected results.

CSS Properties

- **CSS Colors**

Syntax:

The color property of CSS helps to set the color of the HTML element(s). This helps to set the foreground color of text, text decorations, and borders.

```
/* Syntax
selector {
   color: value
}
*/
selector {
   /* colorname can be any colour, such as red, blue, yellow, purple, green, etc. */
   color: colorname
}
```

Note: In CSS we use color, not colour.

- **Example:**

```
<head>
  <style>
    p{
      color: purple;
    }
  </style>
</head>

<body>
  <p>Hello World</p>
  <p>AVENGERS</p>
</body>

</html>
```

- **Output:**

There are many ways to set the property-value of color, with some of the most common listed below.

- **Hexadecimal notation:**

The hex code consists of a hash(#) symbol followed by six characters. These six characters are arranged into a set of three pairs (RR, GG, and BB).
Each character pair defines the intensity level of the colour, where R stands for red, G stands for green, and B stands for blue.
The intensity value lies between 00 (no intensity) and ff (maximum intensity).
Breaking the Character Set (RRGGBB):

- RR: RR defines the intensity of color red, ranging from 00 (no red) to FF (maximum red).
- GG: GG defines the intensity of color Green, with values from 00 (no green) to FF (full green).
- BB: GG defines the intensity of color Blue, varying from 00 (no blue) to FF (full blue).

- **Syntax:**

```
seletor {
    color: #RRGGBB;
}
```

- **Example:**

```
<head>
  <style>
    h1 {
       color: #FF0000;
       /*Pure Red*/
    }
    h2 {
       color: #00FF00;
       /* Pure Green */
    }
    h3 {
       color: #0000FF;
       /* Pure Blue */
    }
    h4 {
       color: #b700ff;
       /* Custom Color */
    }
  </style>
</head>
<body>
  <h1>CodeWithHarry</h1>
  <h2>A Developer</h2>
  <h3>Founder CodeWithHarry.com</h3>
  <h4>Hello World</h4>
</body>
</html>
```

- **RGB**

  RGB stands for “Red, Green, Blue,” and it defines the colour value by taking three (red, green, blue) arguments.
  Each argument value lies between 0 and 255.