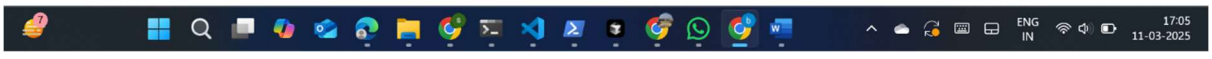
1. Tag Structure and Nesting

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Nested HTML Elements</title>
  <style>
    /* Styling for the span element to show inline styling */
    .custom-text {
      color: blue;
      font-weight: bold;
    }
  </style>
</head>
<body>

  <!-- Heading with nested <em> and <strong> tags -->
  <h1>Understanding <em>Nested</em> <strong>HTML Elements</strong></h1>

  <!-- Explanation -->
  <!--
    The <h1> heading contains:
    - <em>: Italicized emphasis.
    - <strong>: Bold text to indicate strong emphasis.
  -->

  <!-- Paragraph with various inline elements -->
  <p>
    Learning HTML is essential for web development. Here's an important concept:
    <mark>Always write clean and readable code</mark> for better
maintainability.
    In programming, we use <code>document.getElementById()</code> to access
elements in JavaScript.
    The <abbr title="World Wide Web">WWW</abbr> is the foundation of the
internet.
    Some old HTML tags like <strike>this text is deprecated</strike> should be
avoided.
```

You can also use a styled inline span to apply specific formatting.
</p>

<!-- Explanation -->
<!--
The paragraph includes:
- <mark>: Highlights important text.
- <code>: Displays a code snippet.
- <abbr>: Defines an abbreviation with a title attribute for clarification.
- <strike>: Marks text as deleted (deprecated but still used in some cases).
- : Allows inline styling (blue bold text applied via CSS).
-->

</body>
</html>

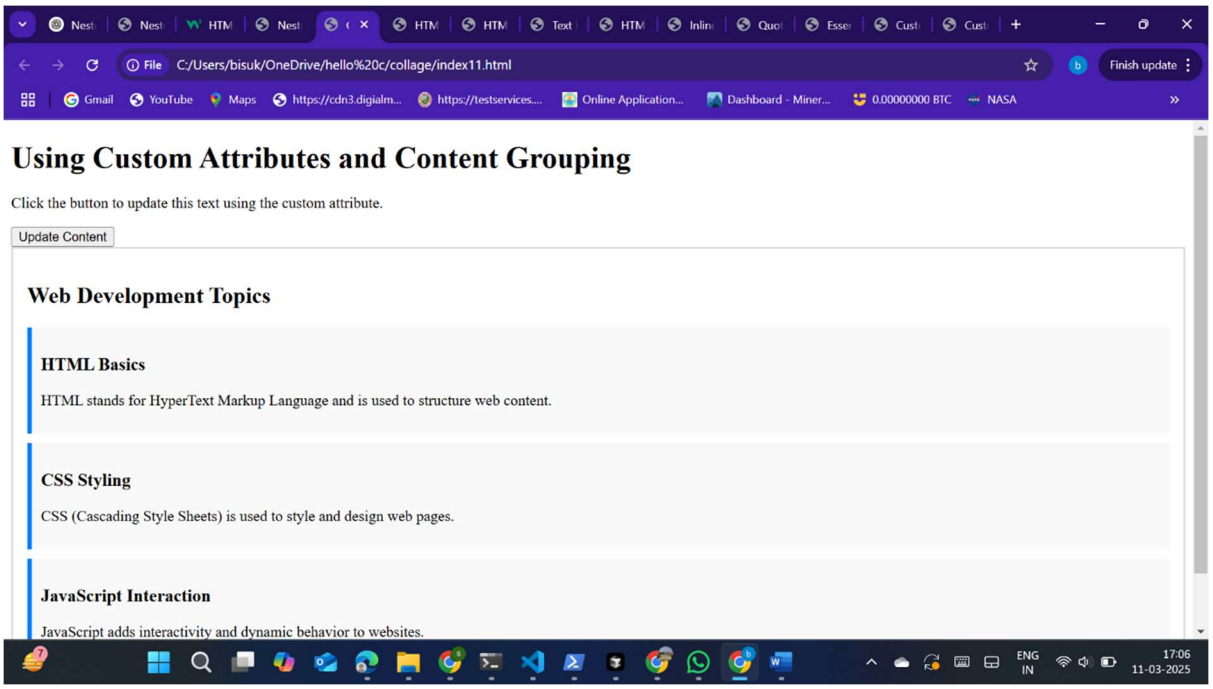


2. Non-Breaking Spaces and Preserve Formatting

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Non-Breaking Spaces & Preserved Formatting</title>
    <style>
        /* Custom HR Style */
        hr {
            border: none;
            height: 4px;
            background-color: #007BFF;
            margin: 20px 0;
        }

        /* Styling for better readability */
        body {
            font-family: Arial, sans-serif;
            padding: 20px;
        }

        .highlight {
            color: red;
            font-weight: bold;
        }
    </style>
</head>
<body>

    <!-- Non-Breaking Spaces Example -->
    <h1>Using Non-Breaking Spaces</h1>
    <p>
        The phrase
        <span class="highlight">Web Development</span>
        remains on the same line because of non-breaking spaces.
    </p>

    <!-- Explanation -->
    <!--
          (non-breaking space) ensures that "Web Development" stays together and doesn't break onto a new line.
    -->

    <hr> <!-- Styled Horizontal Rule -->

    <!-- Using <pre> to Preserve Formatting -->
    <h2>Poem with Preserved Formatting</h2>
    <pre>
        The moon shines bright,
        The stars align,
        A whisper in the night,
```

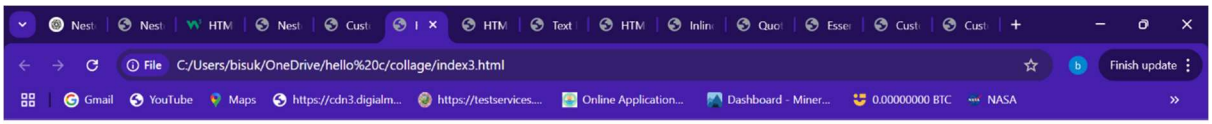
```
        A love divine.

        Soft waves crash,
        Upon the shore,
        A gentle hush,
        Forevermore.
    </pre>

    <!-- Explanation -->
    <!--
        The <pre> tag preserves the exact spaces and line breaks, ensuring the poem is displayed as written.
    -->

</body>
</html>

```



3. Meta Tags for SEO Optimization

```
<!DOCTYPE html>

<html lang="en">

<head>

    <!-- Specify the character encoding as UTF-8 -->

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

```
    <!-- Define the author of the document -->
    <meta name="author" content="Your Name">

    <!-- Set a refresh interval of 5 seconds -->
    <meta http-equiv="refresh" content="5">

    <!-- Provide a brief description of the page with relevant SEO keywords -->
    <meta name="description" content="Learn about HTML meta tags, including character encoding, author details, auto-refresh, and SEO descriptions.">

    <title>HTML Meta Tags Example</title>
</head>
<body>

    <h1>Welcome to My HTML Meta Tags Example</h1>
    <p>This page demonstrates the correct use of meta tags in an HTML document.</p>

</body>
</html>
```



Welcome to My HTML Meta Tags Example

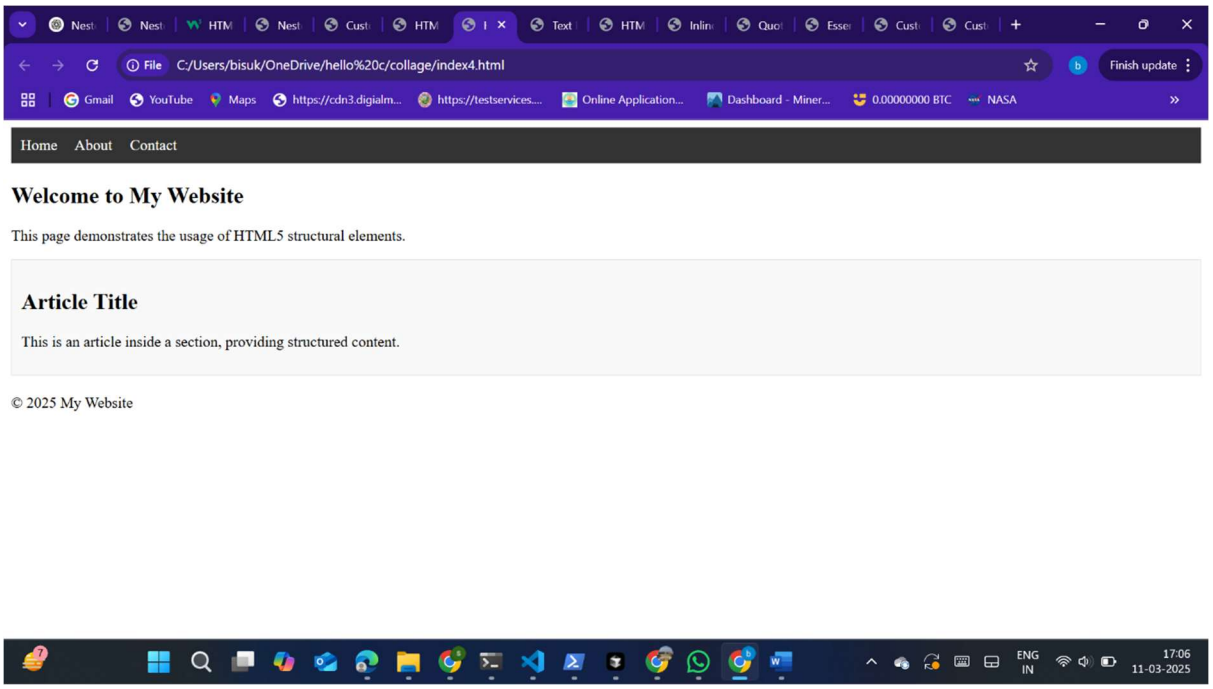
This page demonstrates the correct use of meta tags in an HTML document.

4. Conditional Comments for Browser Compatibility

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>HTML5 Compatibility Example</title>

  <style>
    /* Basic styling for HTML5 elements */
    nav {
      background-color: #333;
      padding: 10px;
    }
    nav a {
      color: white;
      text-decoration: none;
      margin-right: 15px;
    }
```

```
    section {
      margin: 20px 0;
    }
    article {
      padding: 10px;
      border: 1px solid #ddd;
      background-color: #f9f9f9;
    }
  </style>

  <!--[if lt IE 9]>
    <!-- Fallback for older versions of IE that don't support HTML5 elements -->
    <script>
      document.createElement("nav");
      document.createElement("section");
      document.createElement("article");
      document.createElement("aside");
      document.createElement("footer");
    </script>
    <style>
      nav, section, article, aside, footer {
        display: block;
      }
    </style>
  <![endif]-->
</head>
<body>

  <!-- Navigation section -->
  <nav>
    <a href="#">Home</a>
```

```
      <a href="#">About</a>
      <a href="#">Contact</a>
    </nav>

    <!-- Main content using HTML5 structure -->
    <section>
      <h1>Welcome to My Website</h1>
      <p>This page demonstrates the usage of HTML5 structural elements.</p>

      <article>
        <h2>Article Title</h2>
        <p>This is an article inside a section, providing structured content.</p>
      </article>
    </section>

    <footer>
      <p>&copy; 2025 My Website</p>
    </footer>

</body>
</html>
```



5. Working with Heading Tags and Text Formatting

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

   <!-- Main heading with bold and italicized text -->
   <h1>
      Welcome to <b>HTML</b> <i>Formatting</i> Demo
   </h1>

   <!-- Subheading with underlined text and smaller text -->
   <h2>
```

Understanding <u>Text Formatting</u> in HTML <small>(Using various tags)</small>

</h2>

<!-- Paragraph with superscript and subscript text -->

** **

In mathematics, we often use ^{superscript} for exponents and _{subscript} for molecular formulas.

For example, water is represented as H₂O, and Einstein's famous equation is E = mc².

** **

<!-- Displaying the polynomial equation using superscript -->

** **

Here is a polynomial equation formatted properly using HTML:

3x⁷ + 11x⁵ + 2x⁴ − 5x^{-2a} − 23x^{a−3b}

** **

<!-- Comments explaining the usage of different formatting tags -->

<!--

<b>: Makes text bold.

<i>: Italicizes text.

<u>: Underlines text.

<small>: Reduces the font size.

<sup>: Raises text to superscript.

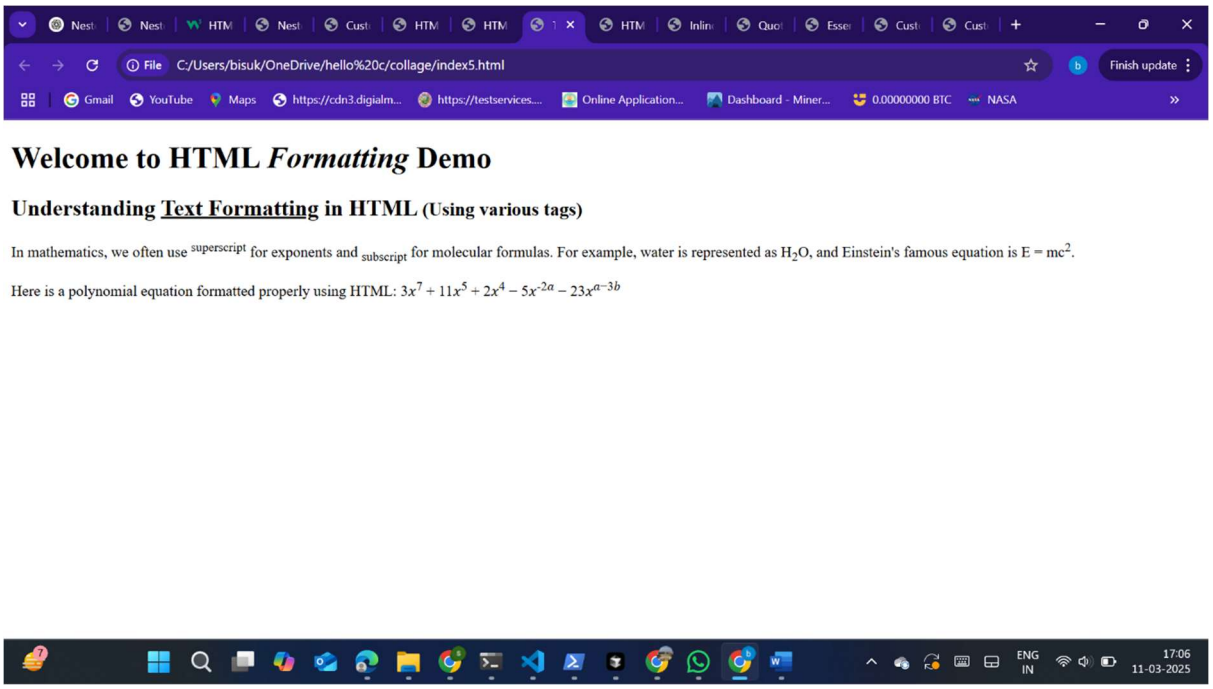
<sub>: Lowers text to subscript.

These tags help in structuring and presenting content effectively.

-->

</body>

</html>



6. Using the and *Tags*

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>HTML Code and Variable Formatting</title>
</head>
<body>

   <!-- Main heading -->
   <h1>Demonstrating <code> and <var> Tags in HTML</h1>

   <!-- Paragraph explaining the purpose of <code> and <var> -->
   <p>
      In HTML, the <code>&lt;code&gt;</code> tag is used to display programming code in a monospaced font,
```

```
    while the <code>&lt;pre&gt;</code> tag preserves the formatting, including spaces and line breaks.

    The <code>&lt;var&gt;</code> tag is used to represent variables in mathematical or programming contexts.

  </p>


  <!-- Code block with <pre> and <code> to maintain indentation -->

  <h2>Example: Displaying Raw Code in HTML</h2>

  <pre>

<code>

// This is a simple JavaScript function to add two numbers

function addNumbers(a, b) {

  return a + b;

}


// Calling the function

let result = addNumbers(5, 10);

console.log("Sum:", result);

</code>

  </pre>


  <!-- Explanation of variables using the <var> tag -->

  <h2>Understanding Variables</h2>

  <p>

    In programming, a <var>variable</var> is a symbolic name for a value that can change.

    For example, in JavaScript, we can define a variable like this:

    <code>let <var>x</var> = 10;</code>.

    Here, <var>x</var> is the variable holding the value <code>10</code>.

  </p>


  <!-- Comments explaining the HTML structure -->

  <!--

    <pre>: Preserves whitespace and line breaks for displaying raw code.
```

<code>: Displays text in a monospaced font, useful for programming code.

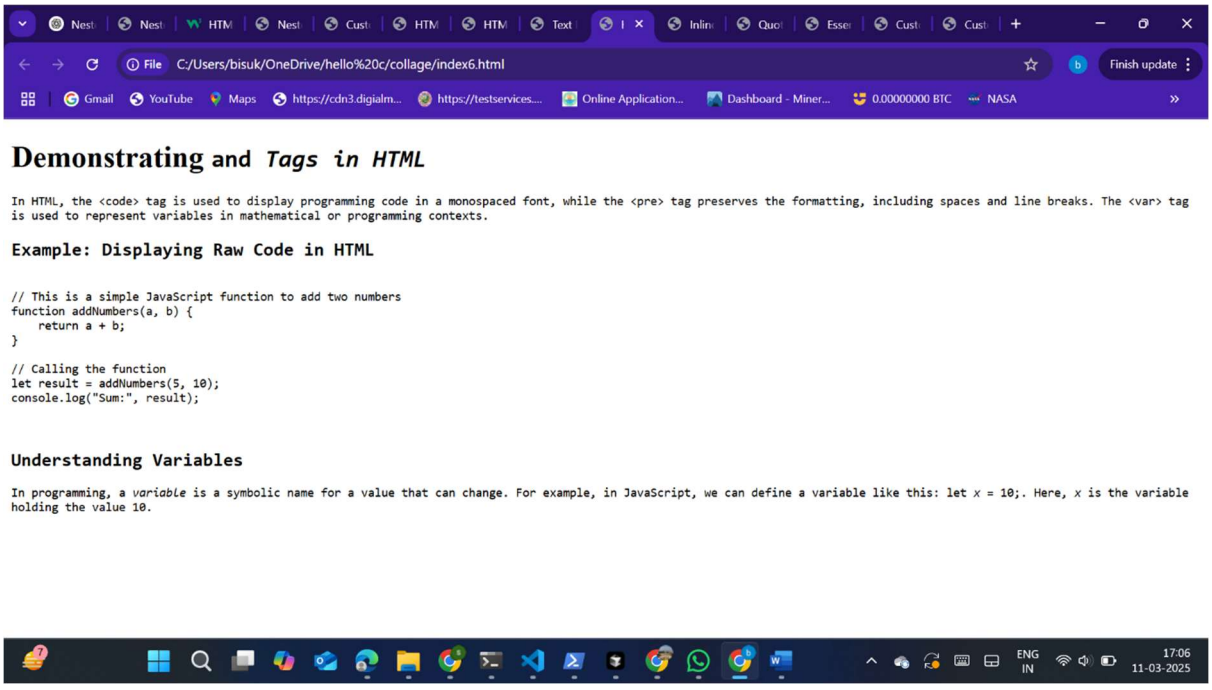
<var>: Represents variables in mathematical or programming contexts.

This structure ensures proper readability and formatting of code snippets and variable names.

-->

</body>

</html>



7. Working with Inline and Block-level Elements

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

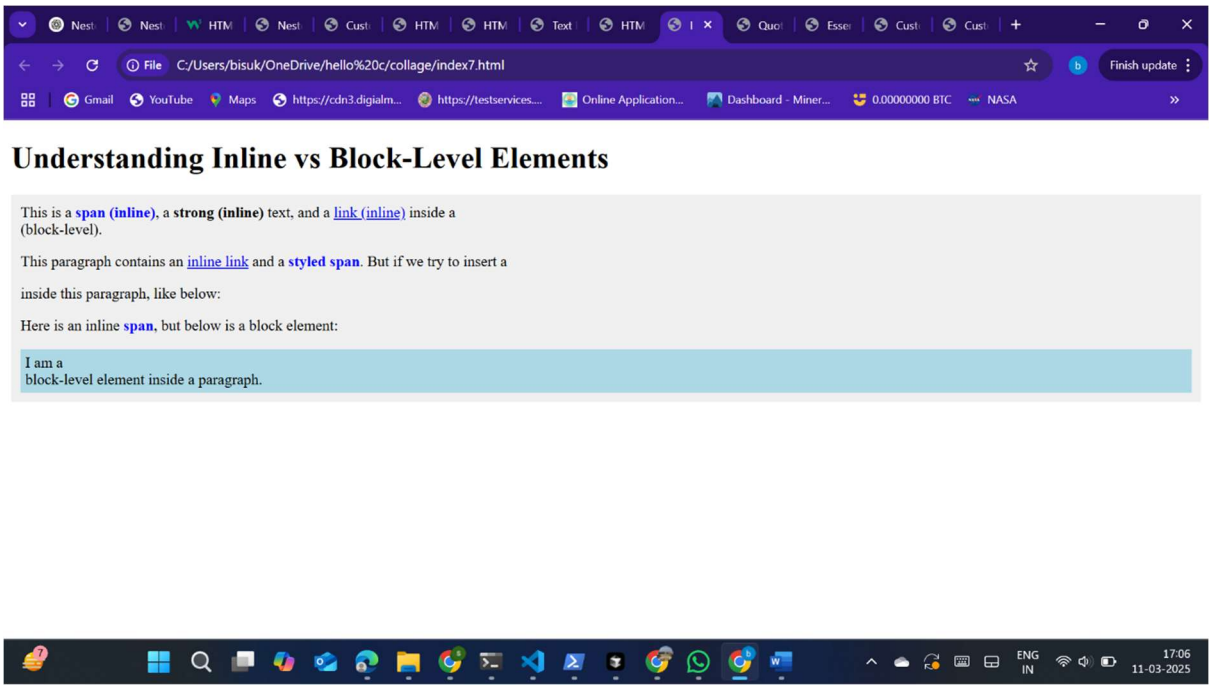
<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Inline vs Block-Level Elements</title>

<style>

```
    /* Styling for visualization */
    .block-container {
      background-color: #f0f0f0;
      padding: 10px;
      margin-bottom: 15px;
    }
    .inline-element {
      color: blue;
      font-weight: bold;
    }
    .block-example {
      background-color: lightblue;
      padding: 5px;
      margin-top: 10px;
    }
  </style>
</head>
<body>

  <h1>Understanding Inline vs Block-Level Elements</h1>

  <!-- Block-level element <div> containing multiple inline elements -->
  <div class="block-container">
    This is a <span class="inline-element">span (inline)</span>,
    a <strong>strong (inline)</strong> text,
    and a <a href="#">link (inline)</a> inside a <div> (block-level).
  </div>

  <!-- Explanation -->
  <!--
    The <div> is a block-level element, meaning it takes up the full width of its parent by default.
```

```
    Inside it, we placed <span>, <strong>, and <a> elements, which are inline elements.
    Inline elements do not start on a new line and only take up as much width as necessary.
  -->

  <!-- Mixing inline and block-level elements -->
  <p>
    This paragraph contains an <a href="#">inline link</a> and a <span class="inline-element">styled span</span>.
    But if we try to insert a <div> inside this paragraph, like below:
  </p>

  <p>
    Here is an inline <span class="inline-element">span</span>,
    but below is a block element:
  </p>

  <div class="block-example">
    I am a <div> block-level element inside a paragraph.
  </div>

  <!-- Explanation -->
  <!--
    Normally, block-level elements like <div> should not be placed inside inline elements like <p>.
    If you try to do so, the browser will automatically break the flow and treat the <div> separately.
    Unlike inline elements, block elements start on a new line and take up the full width.
  -->

</body>
</html>
```

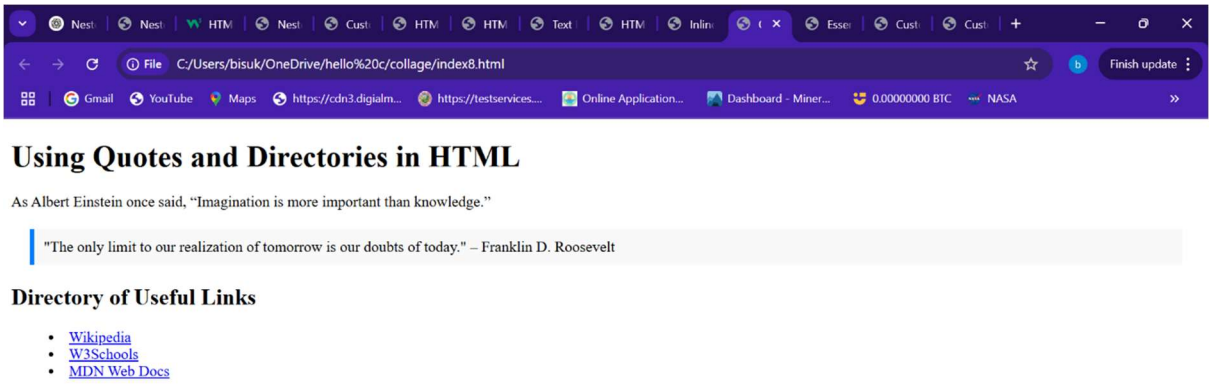


8. The Use of Direction and Quoting Elements

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Quotations and Directory Example</title>
  <style>
    /* Styling blockquote for better visibility */
    blockquote {
      margin: 20px;
      padding: 10px;
      border-left: 5px solid #007BFF;
      background-color: #f9f9f9;
    }
  </style>
```

```
</head>
<body>

  <h1>Using Quotes and Directories in HTML</h1>

  <!-- Short quote using <q> -->
  <p>
    As Albert Einstein once said, <q>Imagination is more important than knowledge.</q>
  </p>

  <!-- Explanation -->
  <!--
    The <q> tag is used for short quotes. It automatically adds quotation marks around the text.
  -->

  <!-- Long quote using <blockquote> with CSS styling -->
  <blockquote>
    "The only limit to our realization of tomorrow is our doubts of today."
    – Franklin D. Roosevelt
  </blockquote>

  <!-- Explanation -->
  <!--
    The <blockquote> tag is used for longer quotes, usually displayed as a separate block.
    CSS is applied to add margins, padding, and a left border for better readability.
  -->

  <!-- Deprecated <dir> tag to represent a directory of links -->
  <h2>Directory of Useful Links</h2>
  <dir>
    <li><a href="https://www.wikipedia.org/">Wikipedia</a></li>
```

```
    <li><a href="https://www.w3schools.com/">W3Schools</a></li>
    <li><a href="https://developer.mozilla.org/">MDN Web Docs</a></li>
  </dir>

  <!-- Explanation -->
  <!--
    The <dir> tag was originally used to define a directory-style list of items.
    However, it is now deprecated, and <ul> (unordered list) should be used instead.
  -->

</body>
</html>
```



9. Creating Accessible Content with Meta Tags

```
<!DOCTYPE html>
<html lang="en">
<head>
```

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

    <!-- Meta tag for responsive design -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <!-- Meta tag to refresh the page every 30 seconds -->
    <meta http-equiv="refresh" content="30">

    <!-- Meta tag for SEO and social media description -->
    <meta name="description" content="Learn about essential HTML meta tags for mobile responsiveness, SEO, and accessibility.">

    <!-- Open Graph meta tag for better sharing on social media platforms -->
    <meta property="og:title" content="Essential HTML Meta Tags">
    <meta property="og:description" content="Understand how to use HTML meta tags for SEO, mobile responsiveness, and social media sharing.">
    <meta property="og:image" content="https://example.com/meta-image.jpg">
    <meta property="og:url" content="https://example.com">

    <title>Essential HTML Meta Tags</title>
</head>
<body>

    <h1>Understanding Meta Tags in HTML</h1>

    <p>
        This page demonstrates the importance of using meta tags for **SEO, responsiveness, and social media sharing**.
    </p>

    <!-- Explanation of the meta tags -->
    <!--
        1. <meta name="viewport">
```

- Ensures the page is mobile-friendly by adjusting the width and scaling.

- Essential for accessibility and responsive design.

2. <meta http-equiv="refresh">

- Automatically refreshes the page every 30 seconds.

- Useful for live updates but should be used sparingly to avoid usability issues.

3. <meta name="description">

- Provides a summary of the page, improving search engine ranking.

- This description is often displayed in search results.

4. Open Graph (<meta property="og:...">)

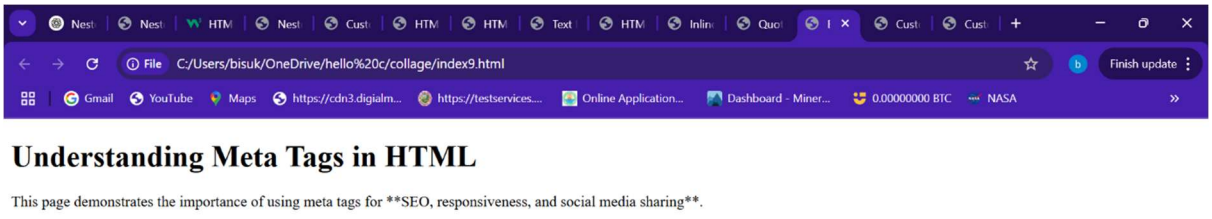
- Enhances the page's appearance when shared on social media.

- Helps platforms like Facebook and Twitter display a title, image, and description correctly.

-->

</body>

</html>

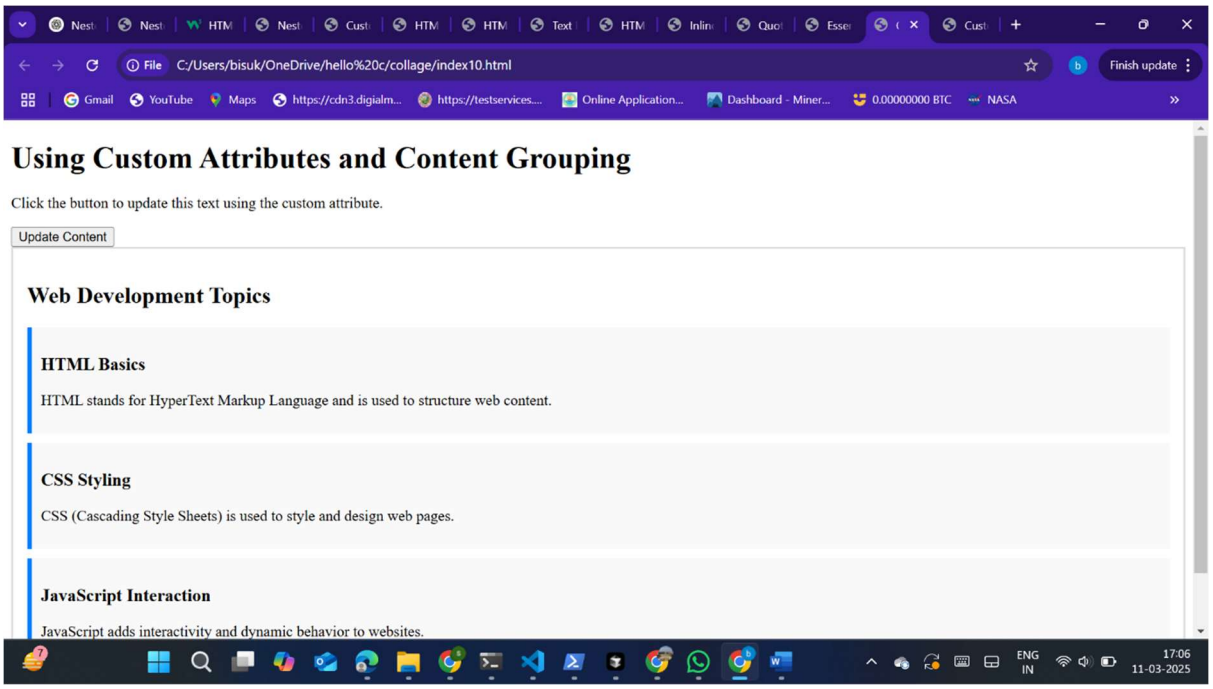


10. Custom Attributes and Grouping Content

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Custom Attributes and Content Grouping</title>
  <style>
    /* Basic styling for better readability */
    section {
      border: 2px solid #ddd;
      padding: 15px;
      margin-bottom: 20px;
    }
    article {
      background-color: #f9f9f9;
      padding: 10px;
      margin-top: 10px;
      border-left: 5px solid #007BFF;
    }
  </style>
</head>
<body>

  <h1>Using Custom Attributes and Content Grouping</h1>

  <!-- A div with a custom data attribute -->
  <div id="infoBox" data-info="This is dynamically updated content!">
    <p id="dynamicText">Click the button to update this text using the custom attribute.</p>
    <button onclick="updateText()">Update Content</button>
```

```
</div>

<!-- Explanation of the data attribute -->
<!--
   The <div> has a custom data attribute (data-info) that stores additional information.
   JavaScript will use this attribute to update the paragraph text dynamically.
-->

<!-- Grouping related content using <section> and <article> -->
<section>
   <h2>Web Development Topics</h2>

   <article>
      <h3>HTML Basics</h3>
      <p>HTML stands for HyperText Markup Language and is used to structure web content.</p>
   </article>

   <article>
      <h3>CSS Styling</h3>
      <p>CSS (Cascading Style Sheets) is used to style and design web pages.</p>
   </article>

   <article>
      <h3>JavaScript Interaction</h3>
      <p>JavaScript adds interactivity and dynamic behavior to websites.</p>
   </article>
</section>

<!-- Explanation of the structure -->
<!--
   The <section> groups related content under "Web Development Topics".
```

```
    Each <article> represents a different topic with a heading and a paragraph.
    This structure improves readability, organization, and SEO.
  -->

  <script>
    function updateText() {
      // Get the div element with the custom data attribute
      let infoBox = document.getElementById("infoBox");

      // Retrieve the custom data attribute value
      let newText = infoBox.getAttribute("data-info");

      // Update the paragraph content
      document.getElementById("dynamicText").innerText = newText;
    }
  </script>

</body>
</html>
```



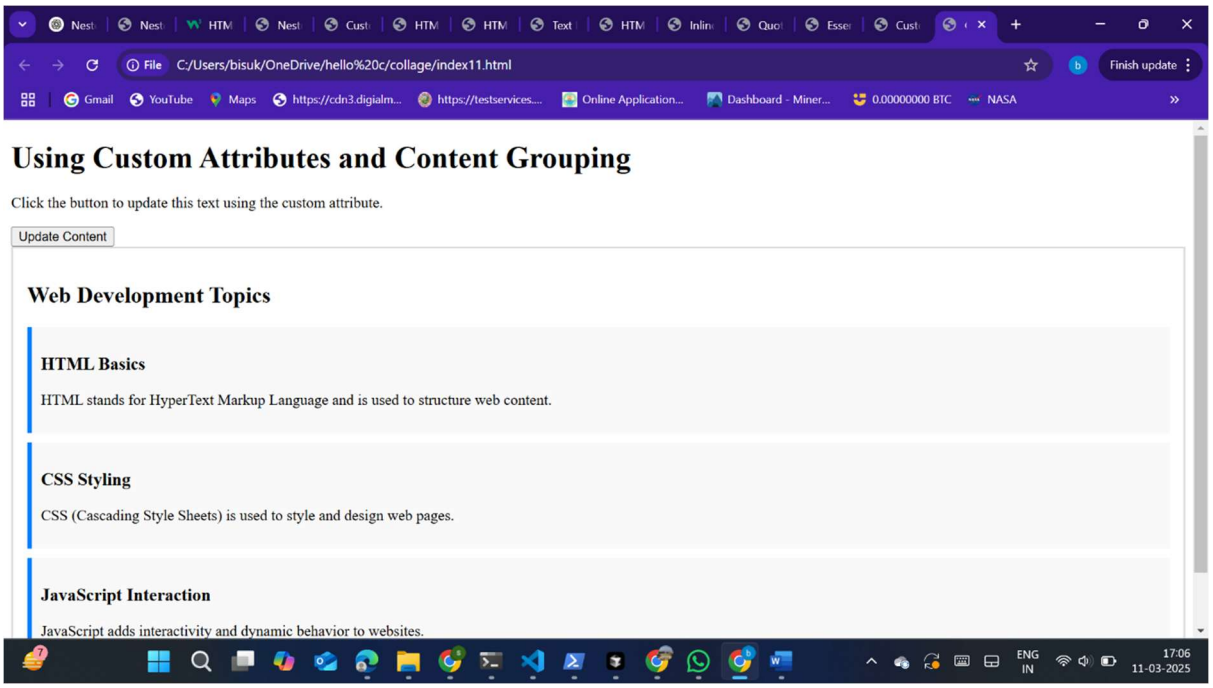
11. Creating a Responsive Layout with Grouped Content

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Custom Attributes and Content Grouping</title>
  <style>
    /* Basic styling for better readability */
    section {
      border: 2px solid #ddd;
      padding: 15px;
      margin-bottom: 20px;
    }
    article {
      background-color: #f9f9f9;
      padding: 10px;
```

```
        margin-top: 10px;
        border-left: 5px solid #007BFF;
    }
  </style>
</head>
<body>

  <h1>Using Custom Attributes and Content Grouping</h1>

  <!-- A div with a custom data attribute -->
  <div id="infoBox" data-info="This is dynamically updated content!">
    <p id="dynamicText">Click the button to update this text using the custom attribute.</p>
    <button onclick="updateText()">Update Content</button>
  </div>

  <!-- Explanation of the data attribute -->
  <!--
    The <div> has a custom data attribute (data-info) that stores additional information.
    JavaScript will use this attribute to update the paragraph text dynamically.
  -->

  <!-- Grouping related content using <section> and <article> -->
  <section>
    <h2>Web Development Topics</h2>

    <article>
      <h3>HTML Basics</h3>
      <p>HTML stands for HyperText Markup Language and is used to structure web content.</p>
    </article>

    <article>
```

```
        <h3>CSS Styling</h3>
        <p>CSS (Cascading Style Sheets) is used to style and design web pages.</p>
    </article>

    <article>
        <h3>JavaScript Interaction</h3>
        <p>JavaScript adds interactivity and dynamic behavior to websites.</p>
    </article>
</section>

<!-- Explanation of the structure -->
<!--
    The <section> groups related content under "Web Development Topics".
    Each <article> represents a different topic with a heading and a paragraph.
    This structure improves readability, organization, and SEO.
-->

<script>
    function updateText() {
        // Get the div element with the custom data attribute
        let infoBox = document.getElementById("infoBox");

        // Retrieve the custom data attribute value
        let newText = infoBox.getAttribute("data-info");

        // Update the paragraph content
        document.getElementById("dynamicText").innerText = newText;
    }
</script>

</body>
```

```
</html>
```


File C:/Users/bisuk/OneDrive/hello%20c/collage/index11.html
Finish update
Gmail
YouTube
Maps
Online Application...
Dashboard - Miner...
0.00000000 BTC
NASA
Using Custom Attributes and Content Grouping
Click the button to update this text using the custom attribute.
Update Content
Web Development Topics
HTML Basics
HTML stands for HyperText Markup Language and is used to structure web content.
CSS Styling
CSS (Cascading Style Sheets) is used to style and design web pages.
JavaScript Interaction
JavaScript adds interactivity and dynamic behavior to websites.
ENG
IN
17:06
11-03-2025