**Day 5: Layouts with CSS**

Knowledge:

* **display Property:**
  + Learn about different values of the **display** property: **block**, **inline**, and **inline-block**.
  + Understand how these values affect the layout of elements.
* **Positioning Elements:**
  + Explore the **position** property and its values: **static**, **relative**, and **absolute**.
  + Understand how these values impact the positioning of elements on the page.

Task:

1. **display Property:**
   * Create a new HTML file with different types of elements (divs, spans, paragraphs).
   * Apply different **display** values (**block**, **inline**, **inline-block**) to these elements.
   * Observe and understand how each value affects the layout.
2. **Positioning Elements:**
   * Experiment with the **position** property on elements within your HTML file.
   * Apply different values (**static**, **relative**, **absolute**) and observe the changes in element positioning.
3. **Layout Design:**
   * Create a simple webpage layout with a header, navigation bar, main content area, and footer.
   * Use the **display** property to structure your layout.
   * Apply different positioning values to elements for a visually appealing design.

Example CSS Code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Layouts with CSS</title>

    <style>

        body {

            font-family: 'Arial', sans-serif;

            margin: 0;

            padding: 0;

            background-color: #f8f8f8;

            color: #333;

        }

        /\* Your CSS code for layout goes here \*/

        .container {

            max-width: 800px;

            margin: 0 auto;

            padding: 20px;

            background-color: #fff;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

            border-radius: 8px;

        }

        .block-element,

        .inline-element,

        .inline-block-element,

        .position-static,

        .position-relative,

        .position-absolute {

            margin-bottom: 20px;

            padding: 20px;

            border: 1px solid #ddd;

            border-radius: 8px;

            text-align: center;

        }

        .block-element {

            display: block;

            background-color: #4CAF50;

            color: #fff;

        }

        .inline-element {

            display: inline;

            background-color: #007bff;

            color: #fff;

        }

        .inline-block-element {

            display: inline-block;

            background-color: #ff9800;

            color: #fff;

        }

        .position-static,

        .position-relative,

        .position-absolute {

            background-color: #f44336;

            color: #fff;

            position: relative; /\* Added position:relative for better context \*/

        }

    </style>

</head>

<body>

    <!-- Professional Information Section -->

    <div class="container">

        <h1>Learn Layouts with CSS</h1>

        <p>Welcome to Day 5 of our web development journey! Today, we'll dive into the <code>display</code> and <code>position</code> properties to control the layout and positioning of elements on a webpage.</p>

        <p>Follow the examples below and observe how different values impact the layout and positioning of elements.</p>

    </div>

    <!-- Display Property Examples -->

    <div class="container">

        <h2>Display Property Examples</h2>

        <div class="block-element">Block Element</div>

        <span class="inline-element">Inline Element</span>

        <div class="inline-block-element">Inline-Block Element</div>

    </div>

    <!-- Positioning Examples -->

    <div class="container">

        <h2>Positioning Examples</h2>

        <div class="position-static">

            <h3>Position: Static</h3>

            <p>This is the default position for an element. Elements with a static position are positioned according to the normal flow of the document.</p>

        </div>

        <div class="position-relative">

            <h3>Position: Relative</h3>

            <p>Positioning an element relative to its normal position. Use the <code>top</code>, <code>right</code>, <code>bottom</code>, and <code>left</code> properties to adjust the position.</p>

        </div>

        <div class="position-absolute">

            <h3>Position: Absolute</h3>

            <p>Position an element absolutely to its nearest positioned ancestor. If there is none, it positions relative to the initial containing block (usually the document body).</p>

        </div>

    </div>

    <!-- Footer Section -->

    <footer>

        <p>&copy; 2023 Web Development Learning. All rights reserved.</p>

    </footer>

</body>

</html>

Additional Challenge (Optional):

* Implement a responsive design for your layout using media queries.
* Explore the **float** property for additional layout options.

Code For Learning (Optional):

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

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

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

**<title>Day 5: Mastering Layouts with CSS</title>**

**<style>**

**body {**

**font-family: 'Arial', sans-serif;**

**margin: 0;**

**padding: 0;**

**background-color: #f8f8f8;**

**color: #333;**

**}**

**.container {**

**max-width: 800px;**

**margin: 0 auto;**

**padding: 20px;**

**background-color: #fff;**

**box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);**

**border-radius: 8px;**

**margin-bottom: 20px;**

**}**

**h1, h2, h3 {**

**color: #4CAF50;**

**}**

**p {**

**line-height: 1.6;**

**}**

**.example-container {**

**margin-bottom: 40px;**

**}**

**.code-container {**

**background-color: #f0f0f0;**

**padding: 15px;**

**border-radius: 5px;**

**margin-top: 20px;**

**overflow-x: auto;**

**}**

**</style>**

**</head>**

**<body>**

**<div class="container">**

**<h1>Day 5: Mastering Layouts with CSS</h1>**

**<p>Welcome to Day 5 of our web development journey! Today, we'll explore the <code>display</code> and <code>position</code> properties in CSS to gain better control over the layout and positioning of elements on a webpage.</p>**

**</div>**

**<!-- Display Property Explanation -->**

**<div class="container">**

**<h2>The <code>display</code> Property</h2>**

**<p>The <code>display</code> property determines how an element is rendered on the page. There are three commonly used values: <code>block</code>, <code>inline</code>, and <code>inline-block</code>.</p>**

**<div class="example-container">**

**<h3>Example: Block Element</h3>**

**<div class="code-container">**

**<p>&lt;div class="block-element"&gt;&lt;/div&gt;</p>**

**</div>**

**<p>The block element takes up the full width available and starts on a new line.</p>**

**</div>**

**<div class="example-container">**

**<h3>Example: Inline Element</h3>**

**<div class="code-container">**

**<p>&lt;span class="inline-element"&gt;&lt;/span&gt;</p>**

**</div>**

**<p>The inline element only takes up as much width as necessary and does not start on a new line.</p>**

**</div>**

**<div class="example-container">**

**<h3>Example: Inline-Block Element</h3>**

**<div class="code-container">**

**<p>&lt;div class="inline-block-element"&gt;&lt;/div&gt;</p>**

**</div>**

**<p>The inline-block element takes up only as much width as necessary, but it allows for setting a height and width.</p>**

**</div>**

**</div>**

**<!-- Position Property Explanation -->**

**<div class="container">**

**<h2>The <code>position</code> Property</h2>**

**<p>The <code>position</code> property is used to specify the positioning method of an element. The values include <code>static</code>, <code>relative</code>, and <code>absolute</code>.</p>**

**<div class="example-container">**

**<h3>Example: Position Static</h3>**

**<div class="code-container">**

**<p>&lt;div class="position-static"&gt;&lt;/div&gt;</p>**

**</div>**

**<p>The static position is the default positioning. Elements are positioned according to the normal flow of the document.</p>**

**</div>**

**<div class="example-container">**

**<h3>Example: Position Relative</h3>**

**<div class="code-container">**

**<p>&lt;div class="position-relative"&gt;&lt;/div&gt;</p>**

**</div>**

**<p>The relative position allows you to move an element relative to its normal position without affecting other elements.</p>**

**</div>**

**<div class="example-container">**

**<h3>Example: Position Absolute</h3>**

**<div class="code-container">**

**<p>&lt;div class="position-absolute"&gt;&lt;/div&gt;</p>**

**</div>**

**<p>The absolute position positions an element absolutely to its nearest positioned ancestor, or to the initial containing block if there is none.</p>**

**</div>**

**</div>**

**<!-- Task for Students -->**

**<div class="container">**

**<h2>Task for Students</h2>**

**<p>Now it's time to practice what you've learned! Create an HTML document and apply the <code>display</code> and <code>position</code> properties to various elements. Observe how each property affects the layout and positioning.</p>**

**</div>**

**<!-- Footer Section -->**

**<footer>**

**<p>&copy; 2023 Web Development Learning. All rights reserved.</p>**

**</footer>**

**</body>**

**</html>**