INSERTION OF IMAGES:

Image Tag: Using the <img> tag to add images to web pages.

Attributes: Understanding src for the image source, alt for alternative text, and other attributes like width and height.

LISTS:

Ordered Lists: Creating ordered lists with <ol> and list items with <li>.

Unordered Lists: Creating unordered lists with <ul> and list items with <li>.

FORMS:

Form Elements: Building basic forms with <form>, <input>, <textarea>, <button>, <select>, and <option> tags.

Form Attributes: Understanding attributes like action, method, and name.

Input Types: Exploring different input types such as text, password, email, submit, and more.

LINKING BETWEEN PAGES:

Anchor Tag: Using the <a> tag to create hyperlinks for navigation.

Attributes: Understanding the href attribute for specifying the URL and using target="_blank" for opening links in a new tab.

CONCLUSION

The first day provided a comprehensive overview of web semantics and HTML basics. These foundational skills are crucial for building well-structured, accessible, and performant web pages. The understanding of semantic web features and basic HTML elements sets the stage for more advanced topics in web development, ensuring a solid grasp of essential concept and best practices.

TRANING REPORT 102 DAY 2 REPORT

12 June, 2024

On Day 2, we focused on the fundamentals of CSS (Cascading Style Sheets) and how it is used to style HTML documents. CSS allows developers to create visually appealing and user-friendly websites by separating content (HTML) from presentation (CSS). The session covered the following key topics:

INTRODUCTION TO CSS:

Purpose: CSS is used to control the layout, color, fonts, and overall visual appearance of web pages. It allows for the separation of content from design, making it easier to manage and maintain website.

Types of CSS

We explored the three main types of CSS and their use cases:

Inline CSS:

Usage: Inline CSS involves applying styles directly within HTML elements using the style attribute. This method is quick for small changes but not ideal for large projects due to maintenance challenges.

Pros: Easy to implement for small, specific changes.

Cons: Makes the HTML code messy and harder to maintain.

Internal CSS:

Usage: Internal CSS is added within the <style> tag in the <head> section of an HTML document. This method is useful for single-page styles but can become cumbersome for larger projects.

Pros: Keeps styles within the same file for easy access when working on single-page applications.

Cons: Not efficient for styling multiple pages.

External CSS:

Usage: External CSS involves linking to an external stylesheet using the <link> tag in the HTML document. This method is ideal for maintaining consistency across multiple pages and is easier to manage and update.

Example:

```
html
<head>
  <link rel="stylesheet" type="text/css" href="styles.css">
</head>
```

Pros: Promotes reusability and maintainability of CSS code. Makes HTML cleaner and separates content from presentation.

Cons: Requires an additional HTTP request to fetch the CSS file, which might slightly impact load times.

LINKING STYLESHEETS

Linking External Stylesheets:

Syntax: Using the <link> tag to connect an HTML document to an external CSS file.

Example:

```
html
<head>
  <link rel="stylesheet" href="styles.css">
</head>
```

Attributes:

Rel ="stylesheet": Specifies the relationship between the current document and the linked file.

type ="text/css": (Optional) specifies the type of the linked file. The default value is text/css.

href ="styles.css": Specifies the path to the CSS file.

Advantages: Promotes reusability and maintainability of CSS code across multiple web pages. Allows for centralized control of the website's design.

PAGE DIVISIONS

**Using **

Divisions: The tag is used for block-level elements to create sections of a web page, while the tag is used for inline elements. These tags help organize and structure HTML documents.

Example:

```
html
<div class="container">
  <div class="header">Header Content</div>
  <div class="main-content">Main Content</div>
  <div class="footer">Footer Content</div>
</div>
```

Styling Divisions: Applying CSS styles to these divisions to layout and design web pages.

Example:

```
css
.container {
  width: 80%;
  margin: 0 auto;
}
.header, .footer {
  background-color: #f1f1f1;
  padding: 20px;
  text-align: center;
```

```
}
.main-content {
  padding: 20px;
```

CONCLUSION

The second day provided a solid understanding of CSS basics and different ways to apply styles to HTML documents. By learning the distinctions between inline, internal, and external CSS, as well as how to link stylesheets and create page divisions, we have laid the groundwork for more advanced styling techniques and responsive web design in future sessions. This foundation enables us to build visually appealing and well-structured web pages.