Final Project Planning

Web Development Essentials - Session 13

Session Overview

Learning Goals for Today:

- Understand the final project requirements.
- Learn how to plan and ideate your project.
- Create a basic structure for your project using HTML, CSS, and JavaScript.

Final Project Overview

What is the Final Project?

- The final project is a culmination of everything you've learned in this course. You will build a complete website combining **HTML**, **CSS**, and **JavaScript**.

Project Requirements:

- **Responsive Website**: The website must be responsive and mobile-friendly.
- **Minimum Pages**: At least three pages (e.g., Home, About, Contact).
- **Interactive Elements**: At least one form and one interactive JavaScript feature (e.g., a dynamic gallery, form validation, etc.).
- **Use of Semantic HTML**: Organize your content with semantic HTML5 elements.
- **Styling**: Use CSS to style the website, including flexbox or grid for layout.
- **JavaScript Interaction**: Implement basic JavaScript interactions (e.g., event handling, DOM manipulation).

Project Ideas and Inspiration

Brainstorming Project Ideas:

- Think of a simple, functional website that aligns with your interests or hobbies. Here are some example ideas:
  - **Portfolio Website**: Showcase your skills, projects, and contact information.
  - **Small Business Website**: Create a landing page for a business with product info, services, and contact details.
  - **Blog**: Develop a blog where users can read posts, subscribe, and leave comments.
  - **Event Website**: Build a website for an event with details, registration forms, and a gallery.

Questions to Consider:

- Who is your target audience?
- What is the main purpose of your website?
- What features or content will your users expect?

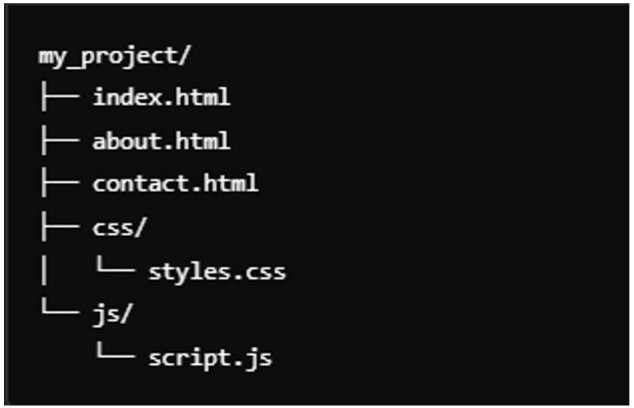
Project Planning Process

Steps to Plan Your Project:

1. **Define the Purpose**: Clearly identify the purpose of your website. Is it informational? Interactive? E-commerce?
2. **List Features and Requirements**: Make a list of essential pages, features, and functionalities.
3. **Create a Wireframe**: Sketch out a basic layout of your website. This helps visualize the structure and flow.
4. **Identify Tools and Technologies**: Decide which HTML elements, CSS properties, and JavaScript functionality you'll use.

Creating a Project Structure

Start by Building a Basic Folder Structure:



Why Structure is Important:

- A well-organized project structure helps you keep your files manageable and maintainable.
- Separating HTML, CSS, and JavaScript files allows for easier editing and debugging.

Wireframing Your Website

What is a Wireframe?

- A **wireframe** is a simple sketch or blueprint of your website's layout and structure, showing where elements like navigation, content, images, and forms will go.

Wireframe Example:

- **Home Page**:
  - Header (with logo and navigation)
  - Hero section (main headline and call to action)
  - Content sections (about info, latest news)
  - Footer (contact info, links)

Tools for Wireframing:

- Paper and pencil
- Online tools (e.g., Figma, Adobe XD, Balsamiq)

Building the Basic HTML Structure

Step 1: Create Your HTML Files:

- Start with your **index.html** file, which will be your homepage.

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My Project</title>
  <link rel="stylesheet" href="css/styles.css">
</head>
<body>
  <header>
    <h1>Welcome to My Website</h1>
    <nav>
      <a href="index.html">Home</a>
      <a href="about.html">About</a>
      <a href="contact.html">Contact</a>
    </nav>
  </header>
```

```html
  <main>
    <section>
      <h2>About Us</h2>
      <p>This is where you'll introduce your website.</p>
    </section>
  </main>

  <footer>
    <p>&copy; 2024 My Website</p>
  </footer>

  <script src="js/script.js"></script>
</body>
</html>
```

Adding Basic CSS Styles

Link Your CSS:

- Add your `styles.css` file to style the project.

```css
/* styles.css */
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
}

header {
  background-color: #333;
  color: white;
  padding: 1em;
  text-align: center;
}
```

```css
nav a {
  margin: 0 15px;
  color: white;
  text-decoration: none;
}

section {
  padding: 2em;
}

footer {
  background-color: #333;
  color: white;
  text-align: center;
  padding: 1em;
  position: fixed;
  bottom: 0;
  width: 100%;
}
```

JavaScript Interactions

Basic JavaScript Setup:

- Add interactive features using JavaScript, like a welcome message or form validation.

JavaScript Example: Welcome Message:

```
document.addEventListener("DOMContentLoaded", function() {
  let welcomeMessage = "Hello, welcome to my website!";
  console.log(welcomeMessage);
});
```

Hands-On Activity

Activity:

- Create the folder structure for your final project.
- Build your basic HTML pages (index.html, about.html, contact.html).
- Link your CSS file and add basic styles.

Goals:

- Have the foundation of your project ready by the end of the session.

Common Mistakes to Avoid

Starting Too Complex: Begin with a simple layout and gradually add features.

Poor Folder Structure: Keep files organized and separate concerns (HTML, CSS, JS).

Not Testing Responsiveness Early: Test your site at different screen sizes from the beginning.

Summary

What We Learned Today:

- How to approach the final project.
- The importance of planning and wireframing.
- Building the basic structure of your project using HTML, CSS, and JavaScript.

Questions?

Q&A Session

- Any questions before we wrap up?

Thank You & See You in the Next Class!