Web Development

Course Description

This comprehensive Web Development course is designed to equip teens and young adults with the skills needed to create modern, responsive websites from scratch. Starting with the fundamentals of HTML and CSS, you'll progress through interactive JavaScript, server-side programming, and even touch on emerging technologies in web development. By the end of this course, you'll have built several web projects, giving you a strong portfolio to showcase your skills. Whether you're looking to pursue a career in web development, enhance your digital skills, or bring your own web ideas to life, this course provides the perfect foundation for your journey into the world of web creation.

Learning Objectives

By the end of this course, learners will be able to:

1. Write semantic HTML to structure web content effectively

2. Style web pages using CSS, including responsive design principles

3. Create interactive web elements using JavaScript

4. Understand and implement basic server-side programming concepts

5. Use version control systems like Git for project management

6. Deploy websites to live servers

7. Implement basic SEO principles for web visibility

8. Understand and apply web accessibility standards

9. Create a portfolio of web projects demonstrating various skills

Requirements

To successfully complete this course, learners should have:

- A computer with internet access

- Basic computer literacy skills

- No prior programming experience required, but a willingness to learn and problem-solve is essential

- Text editor (will be guided on installation in the course)

Curriculum

Module 1: Introduction to Web Development

1. The Internet and How Websites Work

- Basic concepts of clients, servers, and protocols

- The role of web developers in the digital world

2. Setting Up Your Development Environment

- Choosing and installing a text editor

- Understanding file structures for web projects

Module 2: HTML Fundamentals

1. Structure and Syntax of HTML

- Tags, elements, and attributes

- Creating your first HTML page

2. Semantic HTML

- Using the right tags for the right purpose

- Improving accessibility and SEO with semantic markup

Module 3: Styling with CSS

1. CSS Basics

- Selectors, properties, and values

- Linking CSS to HTML

2. Layout and Positioning

- The box model

- Flexbox and CSS Grid for modern layouts

Module 4: Responsive Web Design

1. Mobile-First Design Principles

- Understanding viewport and media queries

- Creating fluid layouts

2. Responsive Images and Media

- Optimizing images for the web

- Using CSS for responsive media

Module 5: Introduction to JavaScript

1. JavaScript Fundamentals

- Variables, data types, and operators

- Control structures (if statements, loops)

2. DOM Manipulation

- Selecting and modifying HTML elements

- Creating interactive web elements

Module 6: Advanced JavaScript and Web APIs

1. Functions and Objects

- Writing reusable code with functions

- Understanding object-oriented programming concepts

2. Working with Web APIs

- Fetching data from APIs

- Local storage and other browser APIs

Module 7: Introduction to Server-Side Programming

1. Backend Basics

- Understanding server-side vs. client-side

- Introduction to Node.js and Express.js

2. Building a Simple Web Server

- Creating routes and handling requests

- Connecting to a database (e.g., MongoDB)

Module 8: Version Control and Deployment

1. Git and GitHub

- Basic Git commands and workflow

- Collaborating with others using GitHub

2. Deploying Your Website

- Choosing a hosting platform

- Deploying a static site vs. a dynamic application

Module 9: Web Accessibility and SEO

1. Making Websites Accessible

- WCAG guidelines

- Testing for accessibility

2. Search Engine Optimization Basics

- On-page SEO techniques

- Understanding meta tags and site structure for SEO

Module 10: Capstone Project

1. Planning Your Web Project

- Choosing a project that showcases learned skills

- Creating wireframes and planning site architecture

2. Building and Presenting Your Website

- Applying learned techniques to create a fully functional website

- Presenting your project and getting feedback

Resources

Videos

- "HTML & CSS Crash Course": A quick introduction to the basics of web structure and styling

- "JavaScript Essentials": Tutorial series covering core JavaScript concepts

PDFs

- "Web Developer's Handbook": A comprehensive guide to HTML, CSS, and JavaScript best practices

- "Responsive Design Cheat Sheet": Quick reference for creating mobile-friendly layouts

Additional Resources

- Mozilla Developer Network (MDN): Extensive web development documentation (https://developer.mozilla.org/)

- CodePen: Platform for testing and showcasing HTML, CSS, and JavaScript code (<https://codepen.io/>)

First Video Script

# Expanded Web Development Course Script - Week 1

## Day 1: The Internet and How Websites Work

### Introduction (15 minutes)

#### Welcome Speech:

"Welcome to our Web Development course! Over the next few weeks, we'll embark on an exciting journey into the world of web creation. You'll learn how to build websites from scratch, understand the technologies that power the internet, and develop skills that are in high demand in today's digital world. Whether you're dreaming of a career in tech or just want to bring your own ideas to life online, this course will give you the foundation you need."

#### Course Overview:

- Briefly outline the course structure: HTML, CSS, JavaScript, server-side basics, and project work.

- Explain the balance between theory and practical work.

- Mention the final project: building a complete website.

#### Ice-breaker Activity:

"Let's start by getting to know each other. Please introduce yourself and share one website you admire and why. This will give us all some inspiration and help us understand what makes a great website from a user's perspective."

### Lecture: Basic Concepts (30 minutes)

#### What is the Internet?

"The Internet is a global network of interconnected computers that communicate with each other using standardized protocols. It's like a vast digital highway system that allows information to travel between devices all over the world."

- Analogy: Compare the Internet to a postal system, where data packets are like letters being sent between addresses.

#### The Client-Server Model:

"Websites operate on what we call a client-server model. Let's break this down:

- The client is typically a web browser on your computer or phone.

- The server is a computer that stores and serves website files.

- When you type a web address, your client sends a request to the server.

- The server then sends back the requested files, which your browser interprets and displays as a web page."

- Visual Aid: Draw a simple diagram showing a computer (client) sending a request to a server and receiving a response.

#### Key Protocols:

"Communication on the Internet is governed by protocols. The two most important for web development are:

1. HTTP (Hypertext Transfer Protocol):

- This is the foundation of data communication on the web.

- It defines how messages are formatted and transmitted between web browsers and servers.

2. HTTPS (HTTP Secure):

- This is the secure version of HTTP.

- It encrypts the data sent between the client and server, providing security for sensitive information like passwords or credit card numbers."

- Analogy: Compare HTTP to sending a postcard (anyone can read it) and HTTPS to sending a sealed letter (only the intended recipient can read it).

### Activity: "Human Web Server" (20 minutes)

#### Setup:

- Divide the class into two groups: "Clients" and "Servers".

- Give "Servers" cards with simple web page content (e.g., "Welcome to My Homepage", "About Me", "My Hobbies").

- Give "Clients" cards with web addresses (e.g., "www.myhomepage.com", "www.aboutme.com").

#### Instructions:

1. "Clients" approach "Servers" and request a page by showing their web address card.

2. "Servers" check if they have a matching page and, if so, show the content to the "Client".

3. If a "Server" doesn't have the requested page, they respond with a "404 - Page Not Found" message.

#### Debrief:

Discuss how this mimics real web interactions, including successful requests, errors, and the need for standardized communication.

### Discussion: The Role of Web Developers (15 minutes)

#### Prompt Question:

"Based on what we've learned so far, what do you think web developers do?"

#### Key Points to Cover:

1. Front-end Developers:

- Focus on what users see and interact with in the browser.

- Work with HTML, CSS, and JavaScript.

- Ensure websites are responsive and work across different devices.

2. Back-end Developers:

- Work on server-side logic and databases.

- Use languages like Python, Ruby, Java, or PHP.

- Handle data storage, security, and server configuration.

3. Full-stack Developers:

- Comfortable working on both front-end and back-end.

- Have a broad skill set covering all aspects of web development.

#### Real-world Perspective:

"Web developers are digital architects and builders. They create everything from simple landing pages to complex web applications like social media platforms or e-commerce sites."

### Wrap-up (10 minutes)

#### Key Takeaways:

1. The Internet is a global network enabling communication between computers.

2. Websites work on a client-server model, with browsers requesting data from servers.

3. HTTP and HTTPS are crucial protocols for web communication.

4. Web developers create and maintain the websites and applications we use daily.

#### Preview Next Class:

"Tomorrow, we'll set up our development environment. We'll install the tools we need to start building websites and learn about how to organize our web projects."

#### Optional Homework:

"If you're eager to get started, try researching different text editors used for web development. Come prepared to share what you've learned!"

## Day 2: Setting Up Your Development Environment

### Review (10 minutes)

#### Quick Quiz:

1. What does HTTP stand for?

2. In the client-server model, what is the client typically?

3. Name one difference between front-end and back-end development.

#### Addressing Questions:

Open the floor for any questions about yesterday's material.

### Lecture: Development Tools (20 minutes)

#### What is a Text Editor?

"A text editor is a type of program used for editing plain text files. For web development, we use specialized text editors that offer features to make coding easier and more efficient."

#### Why Not Use Word Processors?

"Word processors like Microsoft Word are designed for formatted documents, not code. They can add hidden characters that break your code. Text editors are built specifically for writing code."

#### Popular Text Editors:

1. Visual Studio Code (VS Code):

- Free, open-source, created by Microsoft.

- Highly customizable with a large extension ecosystem.

- Great for beginners and professionals alike.

2. Sublime Text:

- Fast and lightweight.

- Not free, but has an unlimited free trial.

3. Atom:

- Free, open-source, created by GitHub.

- Highly customizable, good for beginners.

#### Integrated Development Environments (IDEs):

"IDEs are more comprehensive tools that include a text editor along with other features like debugging tools, compilation capabilities, and more. They're often used for more complex projects or specific programming languages."

### Guided Practice: Installation (30 minutes)

#### Installing VS Code:

1. Go to the official VS Code website (https://code.visualstudio.com/).

2. Download the appropriate version for your operating system.

3. Run the installer and follow the prompts.

4. Open VS Code once installed.

#### Tour of VS Code:

- Show the main areas: file explorer, text editor, integrated terminal.

- Demonstrate how to create a new file and save it.

- Show some basic features like syntax highlighting and auto-completion.

### Lecture: File Structures (20 minutes)

#### Basic Web Project Structure:

```

my-website/

│

├── index.html

├── css/

│ └── styles.css

├── js/

│ └── script.js

└── images/

└── logo.png

```

#### Explanation:

- `index.html`: The main HTML file, usually the homepage of your website.

- `css/`: Folder for CSS files that control the styling of your web pages.

- `js/`: Folder for JavaScript files that add interactivity to your website.

- `images/`: Folder for storing image files used in your website.

#### File Types:

- HTML (.html): Structure and content of web pages.

- CSS (.css): Styling and layout of web pages.

- JavaScript (.js): Adds interactivity and dynamic behavior to web pages.

- Image files (.jpg, .png, .gif): Visual elements of your website.

### Activity: Create Your First Project Structure (20 minutes)

#### Instructions:

1. Open VS Code.

2. Create a new folder called "my-first-website".

3. Inside this folder, create:

- An `index.html` file

- A `css` folder with a `styles.css` file inside

- A `js` folder with a `script.js` file inside

- An `images` folder

#### In `index.html`, add basic HTML structure:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>My First Website</title>

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

</head>

<body>

<h1>Welcome to My First Website!</h1>

<script src="js/script.js"></script>

</body>

</html>

```

### Wrap-up (10 minutes)

#### Key Takeaways:

1. Text editors are essential tools for web development.

2. A well-organized file structure is crucial for managing web projects.

3. Different file types serve different purposes in web development.

#### Homework:

"Explore your text editor. Try to find features like:

- Changing the color theme

- Opening and closing the file explorer

- Using multiple cursors (hint: try holding Ctrl or Cmd and clicking in multiple places)

Come prepared to share one cool feature you discovered!"

#### Preview Next Class:

"In our next session, we'll dive into HTML. You'll learn how to create the structure of a web page and write your first HTML code!"