Week 1: HTML Fundamentals

Day 1: Setup Your Environment

What You Need:

1. **Text Editor:** Download and install VS Code from https://code.visualstudio.com/ or continue using Notepad++
2. **Web Browser:** Use Chrome or Firefox
3. **Folder Structure:** Create a folder on your computer called `path-1-project`

Learning Resources

Start Here:

- MDN HTML Basics: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics
- W3Schools HTML Tutorial: https://www.w3schools.com/html/

Week 1 Exercises

Exercise 1: Personal Profile Page Create a page about yourself with:

- Your name as h1
- A short bio paragraph
- A profile image (use any image)
- Links to your social media (or fake links)
- A list of your hobbies

Exercise 2: Program Information Form Create a form for broadcast programs with:

- Program name (text input)
- Description (textarea)
- Duration in minutes (number input)
- Time slot (time input)
- Channel (dropdown with options: SABC1, SABC2, SABC3, eTV)
- Is Live? (checkbox)
- Submit button

Exercise 3: Navigation Menu Create a simple nav bar with links to: Home, Programs, Schedule, Contact

Exercise 4: Blog Post Page Create a blog post with:

- Article header with title and date
- Multiple paragraphs
- An image
- Subheadings (h2, h3)
- Footer with author info

Validation: Use https://validator.w3.org/ to check your HTML is valid.

What Good Looks Like

Your HTML should:

- Be properly indented (VS Code does this automatically if you right-click → Format Document)
- Use semantic tags (not just divs everywhere)
- Have descriptive class/id names
- Pass the W3C validator without errors

Week 2: CSS Styling & Layout

Learning Resources

Start Here:

- MDN CSS Basics: https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps
- W3Schools CSS Tutorial: https://www.w3schools.com/css/
- CSS Tricks Flexbox Guide: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

Week 2 Exercises

Exercise 1: Style Your Profile Page Take Exercise 1 from Week 1 and style it:

- Choose colors and fonts
- Add spacing (margin/padding)
- Style the image (border-radius for circular?)
- Make links look nice
- Add hover effects

Exercise 2: Responsive Navigation Bar Create a navigation bar that:

- Is horizontal on desktop
- Has hover effects
- Collapses to vertical on mobile (use media query)

Exercise 3: Program Card Layout Create 6 program cards in a grid using Flexbox:

- Each card has image, title, duration, time
- Cards are in a 3-column grid on desktop
- Cards stack vertically on mobile
- Cards have hover effect (lift up, shadow)

Exercise 4: Form Styling Style your form from Week 1:

- Make inputs look better (padding, borders)
- Style the submit button (colors, hover effect)
- Add focus states (input:focus)
- Make it look professional

Week 3: JavaScript Basics & DOM Manipulation

Learning Resources

Start Here:

- JavaScript.info (Tutorial): https://javascript.info/
- MDN JavaScript Basics: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics
- W3Schools JavaScript Tutorial: https://www.w3schools.com/js/

Week 3 Exercises

Exercise 1: Color Changer Create a button that changes the page background color when clicked.

- HTML: button element
- JS: addEventListener, change document.body.style.backgroundColor

Exercise 2: Simple To-Do List Create a to-do list where you can:

- Add new items (input + button)
- Mark items as complete (click to toggle)
- Delete items (delete button per item)
- Store items in an array

Exercise 3: Form Validation Validate a form before submission:

- Check all required fields are filled
- Check email format (contains @)
- Show error messages
- Prevent form submission if invalid (event.preventDefault())

Exercise 4: Tabbed Interface Create tabs that switch content:

- 3 buttons: Tab 1, Tab 2, Tab 3
- 3 content sections
- Clicking a tab shows its content, hides others
- Active tab has different style

Exercise 5: Image Gallery Create a simple gallery:

- Array of image URLs
- Display one image at a time
- Previous/Next buttons to navigate
- Current image number (e.g., "3 of 6")

Week 4: Final Project - Program Manager

Project Requirements

Build a web application to manage broadcast programs. All data stored in **localStorage** (no backend needed yet).

Core Features:

1. Display list of programs
2. Add new program (form with validation)
3. Edit existing program
4. Delete program (with confirmation)
5. Search/filter programs by name

Technical Requirements:

- Pure HTML, CSS, JavaScript (no frameworks)
- Responsive design (works on mobile)
- localStorage for data persistence
- Clean, readable code with comments
- No console errors

Project Structure

```
path-1-project/
├── index.html          (main page)
├── style.css           (all your styles)
├── script.js           (all your JavaScript)
└── README.md           (setup instructions)
```

Getting Started

1. Plan your data structure:

Think about how to store programs. Example:

```
// Array of program objects
[
    {
        id: 1,
        name: "Morning News",
        description: "Daily news briefing",
        duration: 60,
        timeSlot: "06:00",
        channel: "SABC1"
    },
    // more programs...
]
```

2. localStorage basics:

```
// Save data to localStorage
localStorage.setItem('programs', JSON.stringify(programsArray));

// Load data from localStorage
const savedData = localStorage.getItem('programs');
const programsArray = JSON.parse(savedData) || [];
```

Validation Requirements

Form must validate:

- All fields required (can't be empty)
- Duration must be a number
- Duration must be between 1 and 300 minutes
- Show error messages for each validation failure
- Disable submit button if form invalid

Example validation logic:

```
if (programName === '') {
    // show error: "Program name is required"
}
if (duration < 1 || duration > 300) {
    // show error: "Duration must be between 1 and 300 minutes"
}
```

Features Breakdown

Display Programs:

- Loop through programs array
- Create HTML elements for each program
- Insert into DOM
- If no programs, show message "No programs yet"

Add Program:

- Get form values
- Validate all fields
- Create program object with unique ID
- Add to programs array
- Save to localStorage
- Update display
- Clear form
- Show success message

Edit Program:

- Add "Edit" button to each program
- Click Edit → populate form with program data
- Change "Add" button to "Update"
- On submit, find program by ID and update it
- Save to localStorage
- Update display

Delete Program:

- Add "Delete" button to each program
- Click Delete → show confirmation dialog
- If confirmed, remove from array
- Save to localStorage
- Update display

Search:

- Add search input above program list
- On input change, filter programs array
- Show only programs where name includes search term
- Update display with filtered results

Testing Checklist

Before submitting, test:

- ☐ Can add a program (all fields)
- ☐ Can edit a program
- ☐ Can delete a program (with confirmation)
- ☐ Can search programs
- ☐ Form validation works (try empty fields)
- ☐ Data persists after page refresh
- ☐ Works on mobile (use Chrome DevTools device emulator)
- ☐ No errors in console
- ☐ Code is properly commented
- ☐ README has setup instructions

Deliverables

1. Working Application

- All features functional
- Responsive design
- No bugs or console errors

2. README.md Create a file called README.md with:

```
# Program Manager

## How to Run
1. Download all files
2. Open index.html in a web browser
3. Start adding programs!

## Features
- Add new broadcast programs
- Edit existing programs
- Delete programs
- Search programs by name
- Data saved in browser (localStorage)

## Technologies Used
- HTML5
- CSS3
- JavaScript (ES6)
```

3. BookStack Documentation Document on BookStack:

- What you learned each week
- Challenges you faced and how you solved them
- Code snippets with explanations
- Screenshots of your final project
- Tips for the next intern

Getting Help

When Stuck:

1. Read the error message in console
2. Google the error message
3. Check MDN documentation
4. Try console.log() to debug

Good Questions:

- "I'm trying to add a program to localStorage but getting 'undefined'. Here's my code: [code]. What am I missing?"

Bad Questions:

- "It doesn't work, help!"
- "How do I do everything?"

Learning Resources Summary

Documentation (Your Best Friends)

- **MDN Web Docs:** https://developer.mozilla.org/ (Most reliable, detailed)
- **W3Schools:** https://www.w3schools.com/ (Quick reference, examples)
- **JavaScript.info:** https://javascript.info/ (Great tutorials)

Video Tutorials (Optional)

- **freeCodeCamp:** YouTube channel with full courses
- **Traversy Media:** YouTube channel with practical tutorials
- **The Net Ninja:** YouTube channel with playlists

Practice Sites (Optional)

- **freeCodeCamp:** https://www.freecodecamp.org/ (Interactive lessons)
- **Codecademy:** Free courses on HTML, CSS, JavaScript

Tools

- **Can I Use:** https://caniuse.com/ (Check browser compatibility)
- **CSS Tricks:** https://css-tricks.com/ (CSS guides and tutorials)
- **Stack Overflow:** Google your errors, likely someone asked before

---

Final Checklist

Before moving to Path 2, you should be able to:

- ☐ Write valid HTML5 documents
- ☐ Style pages with CSS (colors, fonts, spacing, layouts)
- ☐ Use Flexbox for layouts
- ☐ Write JavaScript functions
- ☐ Manipulate the DOM (select, create, modify, remove elements)
- ☐ Handle events (clicks, form submissions)
- ☐ Work with arrays and objects
- ☐ Validate forms
- ☐ Use localStorage
- ☐ Debug using console and DevTools
- ☐ Build a complete functioning web app
- ☐ Read documentation and Google effectively

If you can do all of the above, you're ready for Month 2: Git!
