Learning Objectives: Intro to Programming - JavaScript

Last updated Spring 2026.

Course Objectives

By the end of this course, students will:

1. Demonstrate JavaScript language proficiency
    a. Master basic syntax, including variables, data types, strings, and conditional logic.
    b. Implement reusable functions and automate repetitive tasks using loops.
    c. Create, manipulate, and iterate over complex data structures such as arrays and objects.
    d. Distinguish between synchronous and asynchronous execution, utilize the Fetch API, and manage non-blocking operations with promises.
2. Master modern web development
    a. Build functional web pages using semantic HTML and style them with fundamental CSS concepts like the box model and Flexbox.
    b. Use the DOM API to dynamically select, create, and update web page content through JavaScript.
    c. Request and handle data from external sources (such as the GitHub API and Open Source APIs) to render dynamic information on a website.
3. Learn professional workflow & tools
    a. Establish a professional development environment using Git and GitHub, including branching, pushing, and managing pull requests.
    b. Define and apply logic through pseudocode and callback functions to solve technical challenges.
    c. Demonstrate proficiency by deploying a professional portfolio and a dynamic web application, supported by technical documentation and presentation.

Lesson-by-Lesson Objectives

Week	Lesson Name	*Students will be able to…*
1	JS Basics & Functions	Demonstrate a foundational understanding of JavaScript syntax by practicing problem-solving techniques,

		manipulating various data types and strings, implementing conditional logic, and encapsulating code within reusable functions.
2	JS Loops & Arrays	Manage complex data and automate repetitive tasks by implementing JavaScript arrays and loops, while establishing a local development environment through Git and GitHub version control workflows.
3	JS Objects	Create, manipulate, and iterate over JavaScript objects and date instances while demonstrating the ability to manage project versions using a local Git branching and pushing workflow.
4	JS Algorithms	Define and apply algorithmic thinking through pseudocode, implement flexible JavaScript logic using callback functions, and complete the collaborative Git workflow by creating and merging a PR.
5	HTML Basics	Construct a functional web page from scratch by writing semantic HTML boilerplate, organizing content with structural elements and attributes, and managing their code using professional development tools like VS Code and Git.
6	CSS Basics	Apply fundamental CSS concepts (including the box model, selectors, and Flexbox) to style a multi-section webpage, implement a navigational header, and manage their code changes using Git branching and pull requests.
7	The DOM API	Manipulate the Document Object Model (DOM) by selecting, creating, and appending HTML elements using JavaScript to dynamically update web page content and structure.
8	Async Programming & Promises	Distinguish between synchronous and asynchronous execution, understand the role of Promises in managing non-blocking operations, and implement a dynamic message form using DOM manipulation and event handling.
9	Fetch API	Use the Fetch API and asynchronous JavaScript to request data from the GitHub API, handle server responses and potential errors, and dynamically render that data as a styled list within the DOM of their portfolio website.
10	Open API	Demonstrate the ability to integrate external data into a web application by performing at least two distinct fetch calls to an Open Source API and dynamically displaying the

		retrieved information on a new, linked HTML page.
11	Final Project	Demonstrate their proficiency in front-end development by deploying a professional portfolio and a dynamic API-driven web application, documented through a video presentation that explains their technical challenges and project functionality.

Final Project Outcomes

Students demonstrate proficiency by completing two capstone projects:

Portfolio Project Students will design and deploy a professional portfolio website that showcases their technical abilities through:

- Semantic HTML structure with responsive CSS layouts (Flexbox/Grid)
- Dynamic JavaScript functionality (DOM manipulation, GitHub API integration, event handling)
- Professional development workflow (Git version control, GitHub hosting, technical documentation)

Open API Project Students will build a data-driven web application that demonstrates their ability to:

- Integrate external data sources by fetching from multiple API endpoints
- Implement user navigation between different data views
- Handle asynchronous operations and potential error cases with clean, readable code