Front End Web Developer Nanodegree Program Syllabus

Build Stunning User Experiences for the Web

Program Overview

The goal of the Front End Web Developer Nanodegree program is to equip learners with the unique skills they need to build and develop a variety of websites and applications. A graduate of this Nanodegree program will be able to:

- Construct responsive websites using CSS, Flexbox and CSS Grid
- Develop interactive websites and UI (User Interface) applications using JavaScript and HTML
- Connect a web application to backend server data using JavaScript
- Automate application build and deployment using Webpack
- Improve offline performance of websites using Service Worker

This program includes 4 courses and 5 projects. Each project you build will be an opportunity to demonstrate your growing web development skills and will help you demonstrate those skills to potential employers or customers.

Estimated Length of Program: 4 months
Frequency of Classes: Self-paced
Prerequisite Knowledge: A well-prepared learner is able to:

- Layout a simple webpage using HTML
- Style a website element using CSS
- Write and test software with JavaScript
- Inspect websites using Developer Tools on a modern web browser (Chrome, Firefox, or Edge)
- Debug and troubleshoot errors and failures in JavaScript programs

Project 1: Styled Blog Website

For your first project, you'll create a multi-page blog website, using best practices for content and page styling with HTML and CSS. You'll practice using responsive layouts, Flexbox, and CSS Grid to create the structure and design for your own blog.

Supporting Course Content: CSS & Website Layout

Lesson	Learning Outcomes
Introduction to HTML	➔ Create a programming project with a code editor ➔ Construct nested websites with HTML tags and elements ➔ Troubleshoot and debug HTML errors and bugs
Introduction to CSS	➔ Style website components by ID, class and type ➔ Connect CSS to a website ➔ Position and display website elements ➔ Modify and control website typography ➔ Troubleshoot and debug issues with stylesheets
Introducing Flexbox	➔ Control web elements orientation and layout with Flexbox ➔ Control ordering of web elements with Flexbox ➔ Align and justify web elements with Flexbox ➔ Transform and resize web elements with Flexbox
CSS Grid	➔ Compare and contrast the use cases for CSS Grid and Flexbox ➔ Structure the layout of a web page using grid columns and rows
Creating Responsive Layouts	➔ Define custom styles for different screen sizes using media queries ➔ Observe and create breakpoints in a website to change layout and styling as a page is resized

Project 2: Dynamic Landing Page for Marketing Content

In this project, you will build a multi-section landing page. Often times, you won't know how much content will be added to a page through a Content Management System (CSM) or an API. To handle this problem,

you will dynamically add content to a web page. You'll be building a landing page that combines your skills with JavaScript, HTML, and CSS to update and control the page and create a dynamic user experience.

Supporting Course Content: JavaScript and the DOM

Lesson	Learning Outcomes
Syntax	➔ Declare block-scoped variables using *let* and *const* ➔ Format JavaScript strings using template literals ➔ Manage arrays and objects using JavaScript destructuring syntax ➔ Iterate over arrays and objects using JavaScript for...of syntax
The Document Object Model	➔ Describe and explain the Document Object Model for web browsers ➔ Access page elements by ID, class, and type using JavaScript
Creating Content with JavaScript	➔ Modify HTML content with JavaScript ➔ Create HTML content and elements with JavaScript ➔ Remove HTML content with JavaScript ➔ Style HTML content with JavaScript and CSS
Working with Browser Events	➔ Describe and explain the phases of browser events ➔ Create event listeners that handle browser events by writing code that runs when an event is triggered ➔ Describe and explain the events that are fired as a web page loads
Performance	➔ Manage website performance by controlling content creation efficiently ➔ Describe what happens when a webpage has to be redrawn ➔ Describe and explain the JavaScript call stack ➔ Describe and explain the JavaScript event loop ➔ Write efficient code by analyzing the call stack and event loop ➔ Delay code execution with setTimeout

Project 3: Weather Journal

In this project, you'll apply your new skills to combine data from the OpenWeatherMap API and client side (browser) HTML forms to create a web app that records a weather journal for users. This project requires you to create an asynchronous web app that uses Web API and user data to dynamically update the UI for a Weather Journal App.

Supporting Course Content: Web APIs and Asynchronous Applications

Lesson	Learning Outcomes
Node & Express Environment	➔ Set up a Node and Express environment to develop a web application on your local machine ➔ Install JavaScript packages using npm (Node Package Manager) ➔ Setup and run a local development server ➔ Manage web application file structure and website assets using Express
HTTP Requests & Routes	➔ Handle requests to an Express with routes ➔ Describe and explain the differences between GET and POST requests ➔ Build a web server and use it to serve data and responses to web requests
Asynchronous JavaScript	➔ Manage asynchronous JavaScript control flow with Promises ➔ Request data from a server using JavaScript Fetch ➔ Update and modify website elements dynamically using asynchronously retrieved data

Project 4: Article Analysis Website

In this project, you'll get a taste of some common production environments and tools that you will likely come across in a front end developer role. You will be building a web tool that allows users to run Natural Language Processing (NLP) on articles or blogs found on other websites. Using an exciting new API called Aylien, you can build a simple web interface to interact with their NLP system. This tool will give back pertinent information about the article: whether the content is subjective (opinion) or objective (fact-based) and whether it is positive, neutral, or negative in tone.

For this project, you will use:

- Node
- Express
- Aylien API
- Webpack
- Service Worker

Supporting Course Content: Build Tools, Webpack, and Service Worker

Lesson	Learning Outcomes
Intro to Build Tools	➔ Describe and explain the problems solved by using automated build tools ➔ Inspect and analyze the activity that occurs when loading a web page in a browser ➔ Describe and explain what Webpack is and how it can be used
Basics of Webpack	➔ Install Webpack to a computer

	➔ Configure webpack for automating build tasks ➔ Define an entry point for a webpack bundle ➔ Install and configure middleware for building an application ➔ Extend Webpack functionality by installing plugins
Sass and Webpack	➔ Describe and explain the benefits and use cases for Sass ➔ Create CSS variables with Sass ➔ Extend and nest CSS sheets and classes with Sass ➔ Configure Webpack to use Sass controlled stylesheets
Final Touches	➔ Control variable and function scope with JavaScript IIFEs (Immediately Invoked Function Expressions) ➔ Optimize an application build pipeline with Webpack ➔ Cache server data and websites functionality using Service Worker

Project 5: Front End Web Developer Nanodegree Program Capstone

In the final project, you'll combine all of the skills you've developed throughout the Nanodegree program to build an online travel app. You'll work with data sources from multiple APIs to create a dynamic travel weather planning application that helps people plan trips by generating weather forecasts for the places they're visiting.

You'll pull together all of the JavaScript, HTML, CSS, and build tool skills and knowledge you've gained to create this application. The design is up to you, and you'll have the flexibility to include and combine other APIs (even your own!) to build this final project.