Front End Nanodegree Syllabus

Build Stunning User Experiences

Before You Start

You've taken the first step toward becoming a web developer by choosing the Front End Nanodegree program. In order to succeed, we recommend having experience using the web, being able to perform a search on Google, and (most importantly) the determination to keep pushing forward! Prior programming experience is not required, but if you'd like to prepare for this Nanodegree, check out our HTML and CSS Syntax course.

The Front-End Web Developer Nanodegree is composed of 8 projects. With each project, you'll create something to demonstrate your mastery of in-demand skills. Projects range in complexity and each builds upon the last. In the end, you will have built a portfolio of projects, including a select set that are resume worthy.

Project 1: Animal Trading Cards

In this project, you'll be creating a trading card for your favorite animal. You will use your knowledge of HTML to create the structure for your trading card. Then you will use CSS styling to design your trading card.

Supporting Lesson Content: HTML Syntax

Lesson Title	Learning Outcomes
HTML Syntax	➔ Identify the parts that make up an HTML tag ➔ Determine when to use specific HTML tags ➔ Correctly structure nested HTML content ➔ Decide between a variety of text editors for writing code

Supporting Lesson Content: CSS Syntax

Lesson Title	Learning Outcomes
CSS Syntax	➔ Identify the benefit of separating style from content ➔ Use CSS to style a website ➔ Test styles by manipulating CSS properties ➔ Use CSS references to lookup standard CSS properties and values

How to Write Code Faster	➔ Use keyboard shortcuts to write code faster ➔ Use code editor packages and themes to improve workflow and write code more efficiently

Project: 2 Build a Portfolio Site

For this project, you'll be building a portfolio website. You will be provided a design mockup as a PDF-file, and you must replicate that design in HTML and CSS. You will develop a responsive website that will display images, descriptions and links to each of the portfolio projects you will complete through the course of your Nanodegree program on any size of screen.

Supporting Lesson Content: Responsive Web Design

Lesson Title	Learning Outcomes
Why Responsive	➔ Create your own responsive web page that works well on any device: phone, tablet, desktop or anything in between. ➔ Explore what makes a site responsive and how some common responsive design patterns work across different devices. ➔ Create your own responsive layout using the \`viewport\` tag and CSS media queries. ➔ Experiment with major and minor breakpoints ➔ Optimize text for reading.
Starting Small	➔ Build HTML elements for any screen size. ➔ Use the browser viewport to create consistent user experiences.
Building Up	➔ Use media queries and breakpoints to create responsive web page designs ➔ Create flexible HTML elements with an introduction to Flexbox

Supporting Lesson Content: Writing READMEs

Lesson Title	Learning Outcomes
Writing READMEs	➔ Identify Markdown syntax ➔ Explain the importance of documentation ➔ Write Markdown to document project instructions and information

Project 3: Memory Game

In this project, you'll demonstrate your mastery of HTML, CSS, and JavaScript by building a complete browser-based card matching game (also known as Concentration). From building a grid of cards, adding functionality to handle user input, and implementing gameplay logic -- you'll combine all your web development skills to create a fully interactive experience for your users.

Supporting Lesson Content: Intro to JavaScript

Lesson Title	Learning Outcomes
What Is JavaScript	➔ Gain insight on history of JavaScript. ➔ Begin writing code immediately using the JavaScript console.
Data Types & Variables	➔ Represent real-world data using JavaScript variables. ➔ Recognize distinctions between different data types.
Conditionals	➔ Use conditional statements to add logic and control flow into JavaScript programs.
Loops	➔ Reduce code duplication and automate repetitive tasks by leveraging JavaScript loops.
Functions	➔ Harness the power of functions to streamline and organize your programs.
Arrays	➔ Leverage, arrays to store complex data in JavaScript programs.
Objects	➔ Alongside arrays, use objects to store complex data.

Supporting Lesson Content: Intro to ES6

Lesson Title	Learning Outcomes
ES6 Syntax	➔ Utilize recent syntax improvements that have been made to the JavaScript language.

Supporting Lesson Content: Shell Workshop

Lesson Title	Learning Outcomes
Shell Workshop	➔ The Unix shell is a powerful tool for developers of all sorts. You'll get a quick introduction to the very basics of using it on your own computer.

Supporting Lesson Content: Version Control with Git & GitHub

Lesson Title	Learning Outcomes
What is Version Control	➔ You'll learn about the benefits of version control and install the version control tool Git!
Create A Git Repo	➔ Create a new repository from scratch ➔ Cloning an existing repository.
Review A Repo's History	➔ Review an existing Git repository's history of commits. ➔ Change how Git Log displays information. ➔ View files that have been modified. ➔ View changes that have been made.
Add Commits To A Repo	➔ Make commits that are saved to the repository. ➔ Write descriptive commit messages. ➔ Verify the changes you're about to save to the repository.
Tagging, Branching, and Merging	➔ Add special markers called tags to commits. ➔ Work on isolated development tracks by making use of Git's branches. ➔ Combine branches together.
Undoing Changes	➔ Modify or undo changes that have been saved to a repository.
Working With Remotes	➔ Create remote repositories on GitHub. ➔ Get and send changes to a remote repository.
Working On Another Developer's Repository	➔ Create copies of a project by forking another developer's repository. ➔ Collaborate with other developers by contributing to a public project.
Staying In Sync With A Remote Repository	➔ Leverage pull requests to send suggested changes to another developer. ➔ Move or combine commits with `git rebase`.

Supporting Lesson Content: JavaScript & the DOM

Lesson Title	Learning Outcomes
The Document Object Model	➔ Learn how the DOM is constructed ➔ Use DOM methods to select page elements ➔ Figure out where an Element's properties come from
Creating Content with JavaScript	➔ Use DOM and JavaScript to add new content to the page ➔ Learn DOM and JavaScript to remove page content ➔ Use DOM and JavaScript to style page elements

Working with Browser Events	➔ Discover the hidden world of browser events ➔ Use DOM and JavaScript to respond to specific events ➔ Learn when the web page is ready to be modified and controlled
Performance	➔ Learn how to measure the speed of your DOM and JavaScript code ➔ Identify code that causes Reflow and Repaint issues ➔ Explain how the JavaScript Event Loop works

Project 4: Classic Arcade Game Clone

In this project, you'll recreate the classic arcade game Frogger. You will be provided visual assets and a game loop engine; using these tools you must add a number of entities to the game including the player characters and enemies.

Supporting Lesson Content: Web Accessibility

Lesson Title	Learning Outcomes
Accessibility Overview	➔ Explore the diversity of different users experience with websites and applications. Learn about using screen readers practically and recognize the challenge of building web experiences for all users.
Focus	➔ Learn how important focus is to maintain an accessible site. Maintain focus using the Tabindex, Keyboard Design Patterns, and Offscreen Content.
Semantics Basics	➔ Dive into the differences between visual UI and semantically designed accessible UI. Add semantic elements to HTML to create a user interface that works for everyone.
Navigating Content	➔ Implement effective semantic navigation using headings, link text and landmarks.
ARIA	➔ Sometimes an HTML element may not have a role or value assigned semantically. In this lesson, you'll use ARIA attributes to provide context for screen readers.
Style	➔ Incorporate CSS styling into your accessible web design and use accessible color schemes to improve accessibility.

Supporting Lesson Content: Object-Oriented JavaScript

Lesson Title	Learning Outcomes
Objects in Depth	➔ Access an object's properties ➔ Create objects using *object literal notation* ➔ Add properties to objects ➔ Remove properties from objects using the *delete* operator ➔ Write methods to access an object with the *this* keyword ➔ Compare an object with another object ➔ Identify global variables as properties of the *window* object ➔ Identify the risks of using global variables ➔ Extract properties and values from an object
Functions at Runtime	➔ Analyze why JavaScript functions are *first-class* functions

	➔ *Callback*: pass a function as an argument into another function ➔ *Runtime scope*: identify variables available for a function to use ➔ Analyze how the JavaScript interpreter accesses variables through the *scope chain* ➔ Utilize a *closure* to pass arguments implicitly, and to store a snapshot of state at function declaration ➔ Write an *immediately-invoked function expression* (IIFE) to create private state
Classes and Objects	➔ Model real-world "things" using object-oriented programming ➔ Write a *constructor function* to instantiate objects ➔ Identify various ways a function can be invoked, including each approach's effect on the value of *this* ➔ Leverage *call*, *apply*, and *bind* to manually set the value of *this* ➔ Access and add properties to an object's *prototype* ➔ Implement *prototypal inheritance* to base an object on another object

Supporting Lesson Content: ES6

Lesson Title	Learning Outcomes
ES6 Functions	➔ With ES6, functions are getting some much-needed improvements. Learn a number of new things including arrow functions and classes.
ES6 Built-ins	➔ The JavaScript environment provides you with a number of features by default. You'll learn about Sets, Maps, Proxies, Generators, how iteration works, and more!
ES6 Professional Developer-fu	➔ With this massive improvement, not all browsers are able to support this new version of JavaScript. You'll learn about using polyfills and transpiling your ES6 JavaScript code to ES5.

Project 5: Feed Reader Testing

In this project, you'll be learning about testing with Javascript. Testing is an important part of the development process and many organizations practice a standard known as "test-driven development" or TDD. This is when developers write tests first, before they ever start developing their application. Whether you work in an organization that writes tests extensively to inform product development or one that uses tests to encourage iteration, testing has become an essential skill in modern web development!

Supporting Lesson Content: Web Tooling & Automation

Lesson Title	Learning Outcomes
Introduction	→ Learn the foundations of what web tooling is and how to prevent over-optimization.
Productive Editing	→ Get your text editor setup, learn all of its powerful features and keyboard shortcuts.
Powerful Builds	→ Start exploring the Gulp build system and automate many of the processes you perform multiple times throughout the course of your work.
Expressive Live Editing	→ Setup LiveReload to automatically reload your browser every time you make a change in your code.
How to Prevent Disasters	→ Learn how to prevent cross-browser issues in your CSS, prevent JavaScript errors, and more - all with your tool pipeline!
Awesome Optimizations	→ Learn how to concatenate, minimize, transpile, and more!

Supporting Lesson Content: JavaScript Testing

Lesson Title	Learning Outcomes
Rethinking Testing	→ Explain the benefits of Test-Driven Development → Use tests to identify expectations of code functionality
Writing Test Suites	→ Use the Jasmine testing framework → Identify the key functions that make up the Jasmine framework → Explain the Red-Green-Refactor life cycle of testing → Write Jasmine tests to validate asynchronous code

Project 6: Restaurant Reviews

For this project, you will convert a static webpage to a mobile-ready web application. You will take a static design that lacks accessibility and convert the design to be responsive on different sized displays and accessible for screen reader use. You will also begin converting this to a Progressive Web Application by caching some assets for offline use.

Supporting Lesson Content: Javascript Design Patterns

Lesson Title	Learning Outcomes
Changing Expectations	→ React to changing product specifications and developer expectations → Explore the Model-View-Controller design pattern → Analyze an existing application for MVC structure
Refactoring With Separation Of Concerns	→ Write code with discrete areas of responsibility in an MVC application → Refactor an existing application to make use of modern code design practices

Supporting Lesson Content: JavaScript Promises

Lesson Title	Learning Outcomes
Creating Promises	→ Learn what a promise is, how it makes writing asynchronous JavaScript simpler and how to handle errors.
Chaining Promises	→ Create sequences of asynchronous work by chaining Promises together and dive into more advanced error handling.

Supporting Lesson Content: Asynchronous JavaScript

Lesson Title	Learning Outcomes
Ajax with XHR	→ Connect to external web APIs to power asynchronous browser updates
Ajax with jQuery	→ Use the jQuery Javascript library to build Ajax requests and handle API responses → Handle error responses with Ajax
Ajax with Fetch	→ Use the new Fetch API to make asynchronous requests and handle the returned data

Supporting Lesson Content: Front-end Frameworks

Lesson Title	Learning Outcomes
Features of Single Page Apps	➔ Learn about the features of a single page web application.
Examine a Framework's Source	➔ Dig around in the Backbone framework to discover how many of its most popular features work.
Angular	➔ Learn how to build a single page application in the Angular framework.
Ember	➔ Learn how to build a single page application in the Ember framework.

Supporting Lesson Content: Offline Web Apps

Lesson Title	Learning Outcomes
The Benefits of Offline First	➔ Discover the differences between a standard web app and an offline-first application and get an introduction to new APIs.
Introducing the Service Worker	➔ Recognize the differences between good, poor, intermittent, and missing connectivity for your users, and master how to make applications that navigate these conditions with ease.
IndexedDB and Caching	➔ Use the IndexedDB API, along with Service Workers, to write caching solutions that will make your applications more performant.

Project 7: MyReads

In this project, you will create a React application from scratch and utilize React components to manage the user interface. You'll create a virtual bookcase to store your books and track what you're reading. Using the provided Books API, you'll search for books and add them to a bookshelf as a React component. Finally, you'll use React's *setState* method to build the functionality to move books from one shelf to another.

Supporting Lesson Content: React Fundamentals

Lesson Title	Learning Outcomes
Why React	➔ Identify why React was built ➔ Use *composition* to build complex functions from simple ones ➔ Leverage *declarative code* to express logic without control flow ➔ Recognize that React is just JavaScript
Rendering UI with React	➔ Use *create-react-app* to create a new React application ➔ Create reusable, focused *Class components* with composition ➔ Leverage *JSX* to describe UI
State Management	➔ Manage state in applications ➔ Use *props* to pass data into a component ➔ Create *functional components* focused on UI rather than behavior ➔ Add *state* to components to represent mutable internal data ➔ Use the *this* keyword to access component data and properties ➔ Update state with *setState()* ➔ Use *PropTypes* to typecheck and debug components ➔ Use *controlled components* to manage input form elements
Render UI with External Data	➔ Conceptualize the *lifecycle* of a component ➔ Use React's *componentDidMount* lifecycle hook for HTTP requests
Manage App Location with React Router	➔ Use React Router to add different routes to applications ➔ Use state to dynamically render a different "page" ➔ Use React Router's *Route* component ➔ Use React Router's *Link* component

Project 8: Neighborhood Map (React)

In this project, you will develop a single-page application featuring a map of your neighborhood or a neighborhood you would like to visit. You will then add additional functionality to this application, including: map markers to identify popular locations or places you'd like to visit, a search function to easily discover these locations, and a listview to support simple browsing of all locations. You will then research and implement third-party APIs that provide additional information about each of these locations (such as StreetView images, Wikipedia articles, Yelp reviews, etc).

Supporting Lesson Content: Google Maps API

Lesson Title	Learning Outcomes
Getting Started with the APIs	→ Set up your developer credentials and get started with the Google Maps APIs.
Understanding API Services	→ Explore the location services available in the Google Maps APIs, including the Geocoding, Elevation, and Directions APIs.
Using the APIs in Practice	→ Learn the practical details you need to know to use the Google Maps APIs in the real world.