Front-End Web Developer 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 Web Developer 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 Intro to HTML & CSS course.

The Front-End Web Developer Nanodegree is composed of 9 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: Developer Mindset

In this project, you'll learn about the history of the languages you'll be studying, the Web, and our place today as Web Developers in an evolving story. You'll also learn how to get feedback from Udacity Project Reviewers. Later on in the Nanodegree program you'll receive code review in a similar way.

Supporting Lesson Content: Establishing Developer Mindset

Lesson Title	Learning Outcomes
Establishing Developer Mindset	➔ List the three main languages of Front-End Web Development ➔ Explain the purpose of HTML ➔ Explain how CSS is used in web development ➔ Give examples of the purpose of JavaScript ➔ Recognize that the web is constantly evolving

Project: Mockup to Article

In this project, you'll be given a design mockup that you will convert into a website built with HTML. You'll need to carefully examine the mockup to determine the specific tags to use to achieve the correct visual and structural result.

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

Project: 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: 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: 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: HTML, CSS, and Responsive Web Design

Lesson Title	Learning Outcomes
Sizing Elements	➔ Identify issues surrounding the default sizing for the box model ➔ Size elements using height, width, padding, border, and margin (properties of the box model) ➔ Use semantic elements to develop structurally sound markup ➔ Recognize the difference between inline and block elements as it relates to sizing elements
Intro to HTML and CSS	➔ Leverage the cascading behavior of CSS to write reusable styles ➔ Positioning elements using a grid-based design ➔ Use DevTools to aid in website development ➔ Use CSS frameworks to improve workflow and quickly assemble beautiful, responsive websites ➔ Use Twitter Bootstrap to build a responsive portfolio site
Responsive Web Design Fundamentals	➔ Identify the importance of designing websites for different resolutions (mobile, tablet, desktop) ➔ Build media queries to respond to different screen resolutions ➔ Determine appropriate breakpoints for responsive design ➔ Use developer tools for device emulation to test responsive designs ➔ Use responsive patterns, properties, and units for flexible sizing
Responsive Images	➔ Determine the appropriate image type to use on the web given a specific situation ➔ Optimize images to improve website performance ➔ Use `srcset` to let the browser determine the best image to use given the user's screen resolution ➔ Use `calc` to dynamically size images

Project: Online Resume

In this project, you'll write JavaScript code to power your own online resume. You will build a resume by writing JavaScript code that combines your personal information with pre-written HTML and CSS templates to generate your resume.

Supporting Lesson Content: JavaScript & jQuery

Lesson Title	Learning Outcomes
JavaScript Basics	➔ Use developer tools to interact with a website ➔ Use console.log to debug information
Data Types	➔ Add content to a webpage using JavaScript ➔ Replace text with the `.replace()` function ➔ Determine what JavaScript values are truthy/falsy ➔ Use Objects and Arrays to store data ➔ Explain the purpose of JSON ➔ Write valid JSON
Flow Control	➔ Control the flow of code with `if` statements ➔ Loop over data with a `for` loop ➔ Write self-contained blocks of code using functions
jQuery Basics: the DOM, $, and Selectors	➔ Explain the purpose of jQuery ➔ Use jQuery to select elements from the page ➔ Use jQuery methods to filter the list of selected items
The Tricks: DOM Manipulation	➔ Modify the classes on an element ➔ Change an element's attributes ➔ Add/remove DOM elements ➔ Run a block of code over each item in a set
Event Listeners with jQuery	➔ Listen for browser events and run code in response ➔ Identify different event types ➔ Monitor types of events ➔ Use event delegation to minimize number of event listeners

Project: 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: Object Oriented JavaScript, HTML5 Canvas, and READMEs

Lesson Title	Learning Outcomes
Scope	➔ Explain the importance of scope in JavaScript ➔ Identify the execution context of functions ➔ Create functions that run outside of their lexical scope
Closure	➔ Explain how functions and function scope create closure ➔ Trace identifier lookup during function execution
The 'this' Keyword	➔ Identify how the `this` keyword is bound ➔ Determine the `this` value in a function ➔ Discover pitfalls when the `this` value loses its context
Prototype Chains	➔ Describe what a prototype chain is ➔ Explain how a prototype method is accessed by an instance ➔ Use a "constructor" function to create similar objects
Object Decorator Pattern	➔ Use the Decorator Pattern to add data and functionality to an object ➔ Remove duplication by using inheritance
Functional Classes	➔ Identify the purpose of a Constructor ➔ Use the `this` value to alter an object in a Constructor ➔ Add data to a constructed object ➔ Look up data on a constructed object
Prototypal Classes	➔ Identify the purpose of a Constructor's .prototype property ➔ Ddd methods to a Constructor's .prototype property
Pseudoclassical Patterns	➔ Identify the pseudoclassical pattern
Superclass and Subclasses	➔ Remove duplication by using Superclasses and Subclasses
Pseudoclassical Subclasses	➔ Use the Pseudoclassical pattern to create instances of objects ➔ Explain how .call works with functions ➔ Use .call to link functions together ➔ Use Object.create() to create objects that delegate functionality
HTML5 Canvas Basics	➔ Create a canvas

	➔ Use JavaScript to draw shapes and lines on the canvas ➔ Change the color of items on the canvas ➔ Write text on the canvas
From Pixels to Animation	➔ Implement filters to add effects to a canvas ➔ Use requestAnimationFrame for efficiently drawing on the canvas
Writing READMEs	➔ Identify Markdown syntax ➔ Explain importance of documentation ➔ Write Markdown to document project instructions and information

Project: Website Optimization

In this project, you'll optimize a provided website with a number of optimization and performance-related issues so that it achieves a target PageSpeed score and runs at 60 frames per second. You will learn about the critical rendering path, the process by which the browser receives HTML, CSS and JavaScript and the required processing to turn them into rendered pixels.

Supporting Lesson Content: Website Performance and Rendering Optimization

Lesson Title	Learning Outcomes
Website Performance Optimization	➔ Enable debugging on a mobile android device ➔ Remotely inspect page elements on a mobile device ➔ Record a timeline trace in Chrome DevTools
The Critical Rendering Path	➔ Identify the steps of the Critical Rendering Path ➔ Explain how HTML is converted to the DOM ➔ Run a Timeline trace of a website and navigate the Timeline interface ➔ Save and load Timeline traces ➔ Explain why CSS is render-blocking ➔ Explain how the DOM and CSSOM combine to form the Render Tree ➔ Calculate the size and layout of page elements ➔ Analyze how site elements are painted to the page
Optimizing the CRP	➔ Apply CSS optimizations to increase the Critical Rendering Path ➔ Identify that JavaScript is render-blocking ➔ Construct JavaScript tags so they don't block the Critical Rendering Path ➔ Calculate the Critical Rendering Path metrics
The Critical Rendering Path	➔ Explore what goes into a single frame ➔ Identify how the DOM, CSSOM, and Render Tree combine to display a finished website
App Lifecycles	➔ Explore the parts of RAIL ➔ List the time frame codes necessary for a webpage to perform smoothly
Weapons of Jank Destruction	➔ Navigate the Timeline pane in DevTools ➔ Record a website's frames using the Timeline pane ➔ Identify the causes of Jank ➔ Change how data is displayed in the Timeline pane
JavaScript	➔ Optimize JavaScript for animation ➔ Explain why requestAnimationFrame should be used for animations ➔ Use the JavaScript Profiler to diagnose long-running JavaScript code

	➔ Create Web Workers to run code separately from the main thread
Styles and Layout	➔ Identify and solve performance issues from style calculations ➔ Explain the performance implications of style changes ➔ Diagnose causes of layout thrashing
Compositing and Painting	➔ Use the Paint Profiler to determine what areas of the page are being painted ➔ Promote page elements to their own layers for increased rendering speed

Project: Neighborhood Map

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: Intro to AJAX

Lesson Title	Learning Outcomes
Requests and APIs	➔ Connect to external web APIs to power asynchronous browser updates
Building the Move Planner App	➔ Use the jQuery Javascript library to build Ajax requests and handle API responses ➔ Handle error responses with Ajax

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
Using An Organizational Library	➔ Build a reactive front end application using an organizational library, knockout.js ➔ Implement knockout models and observable elements in an application
Learning A New Codebase	➔ Use proven strategies to adapt to a new and unfamiliar codebase

Project: 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: 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