Introduction to Programming Nanodegree Syllabus

Learn to Code

Before You Start

Thank you for your interest in the Introduction to Programming Nanodegree! 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.

Estimated Time Commitment

Depending on how quickly you work through the material, the amount of time required is variable. We have included an hourly estimation for each section of the program. If you spend about 10 hours per week working through the program, you should finish in 17 - 19 weeks, so approximately 4.5 months.

Project: Getting Started with HTML *Estimated Time: 10 hours*

For this project, you will submit your very first programming file containing HTML code. HTML is the coding language for building websites. We recommend taking notes from this section and using your notes as the content for your HTML file. This project is not graded. Our reviewers will provide you with helpful suggestions and advice for learning in this program.

Supporting Lesson Content: Intro to the Web and HTML Basics

Lesson Title	Learning Outcomes
Welcome to the Nanodegree	➔ Understanding on how to set up for the program on your personal device ➔ Introduction to the "Programmer Mindset" ➔ Successfully writing and rendering your first lines of HTML code with a text editor and browser
Nanodegree Orientation	➔ Understanding on how to submit projects ➔ Understanding on student support services offered for students ◆ Student Forums, Slack Community, 1:1 Video Appointments ➔ Habits of Successful Students
The World Wide Web	➔ High level overview on how the web works

	➔ Components of the web: browsers, HTTP requests, Servers, the Internet
HTML Basics	➔ HTML tags ➔ Adding Images ➔ HTML Syntax ➔ Whitespace ➔ Inline vs Block elements ➔ HTML Document Structure

Project: Make a Web Page *Estimated Time: 30 hours*

In this section, you'll learn both HTML and CSS - both languages for developing websites. For the project, you'll use HTML and CSS to make a stylish web page on any topic. You will apply your knowledge of HTML Document Structure to your html file and then create custom CSS styling based on your personal preferences. This project will demonstrate your understanding of linking CSS files in HTML files, implementing CSS classes to avoid repetition, as well create semantically organized HTML code.

Supporting Lesson Content: HTML Syntax & CSS

Lesson Title	Learning Outcomes
Creating a Structured Document	➔ Page Structure ➔ Visual Styling ➔ Designing with Boxes
Work Session: HTML Structure	➔ Apply what you've learned to your first project ➔ Share and Discuss on the Student Forums
Adding CSS for Style	➔ Understanding CSS ➔ Divs, Spans, and Classes ➔ Semantic Tags ➔ Using DevTools in the Browser ➔ Verifying HTML and CSS files ➔ Debugging HTML and CSS code
Work Session: CSS Practice	➔ Apply what you've learned to your own html and css files ➔ Continued Learning opportunities with pre-recording webcast sessions

Project: Code Your Own Quiz *Estimated Time: 60 hours*

In this section, you will learn the Python programming language. You will finish by building your own fill-in-the blank style quiz that can even be used as a study tool to help you remember important vocabulary.

Supporting Lesson Content: Python Programming

Lesson Title	Learning Outcomes
Getting Setup with Python	➔ Installing Python and learning Command Line Interface (CLI) basics
Introduction to Serious Programming	➔ Write your first lines of Python code ➔ Learn about language ambiguity:human language vs computer language ➔ Python expressions for math problems
Basic Debugging	➔ Explore syntax error messages and troubleshoot basic Python code
Variables & Strings	➔ Learn how to store values in **Variables** and work with text as **Strings** ➔ Selecting substrings with String Indexing
String Manipulation	➔ Use String methods: **slicing**, **concatenation**, **find,** and **replace**
Input → Function → Output	➔ Learn how to use **functions** to take an input and transform it into some output
Print vs Return	➔ Understand the difference between **print** and **return** statements
Control Flow & Loops	➔ Learn how to manage the flow of a computer program using **Boolean** values, **if** statements, and **While** loops
Debugging	➔ Get acquainted with five key debugging strategies to help you address problems in your code
Mad Libs Generator	➔ Use the skills you've learned so far to continue developing your Mad Libs generator
Structured Data: Lists & For Loops	➔ Use **Lists** to store more complex data ➔ Use **For** loops to programmatically access each item within a List
How to Solve Problems	➔ Practice problem-solving techniques by breaking down large problems into smaller ones.

Project: Create a Movie Website *Estimated Time: 40 hours*

For this project, you'll write code to store a list of your favorite movies, including box art imagery and a movie trailer URL. You will then use your code to generate a static web page allowing visitors to browse their movies and watch the trailer.

Supporting Lesson Content: Object-Oriented Programming

Lesson Title	Learning Outcomes
Mini-Project: Take a Break	➔ Using documentation to find useful libraries for problem-solving ➔ Create a program that uses Python modules: **time** and **webbrowser**
Mini-Project: Secret Message	➔ Learn how to open files with Python code ➔ Use documentation to learn how to rename files ➔ Decode a secret message with Python module: **os**
Mini-Project: Draw Turtles	➔ Use Python Turtle graphics to create shapes with a GUI (graphical user interface) ➔ Understanding of **Classes** and **Objects** in Python
Mini-Project: Send a Text	➔ Use the Twilio API to make a program that sends a text message to your phone
Mini-Project: Profanity Editor	➔ Create program that checks for profanity in a message and replaces curse words
Movie Website Creation	➔ Make a movie website using the concept of Classes in Python as well as your understanding of HTML & CSS
Advanced Class Making	➔ Learn about advanced ideas in **Object Oriented Programming** like class variables, **inheritance**, reusing methods, and method overloading

Discover Your Path *Estimated Time: 10 hours*

In this section, there is no project submission. Instead, you will explore a quick overview of the vast world of programming. After this section, you'll have a better understanding of different options you have as a programmer. This will help guide to in your final project for this program.

Supporting Lesson Content: Exploration of Five Programming Career Tracks

Lesson Title	Learning Outcomes
Front-End Programming	➔ Learn about front-end web developers who create intuitive and responsive websites
Back-End Programming	➔ Learn about back-end web programmers who write server-side code to build web apps that serve millions of people worldwide
Mobile Programming	➔ Learn about mobile programming and the differences between iOS and Android programming
Data Analysis Programming	➔ Learn about data analysts who analyze data to direct growth and make informed decisions
Reverse Engineer Project Compass	➔ Dissect a fully-functioning web app and identify the roles of various programmers in its creation

Project: Choose Your Path - Final Project *Estimated Time: 20 - 40 hours*

Choose one of the five paths (Front-End, Back-End, Android, Data Analyst, or iOS) and complete the associated course and project of your choosing. After you have met specifications on one of these project options (as well as the other required projects), you'll be eligible to graduate.

Supporting Lesson Content: Choice of content from five career-track programs.

Elective Title	Learning Outcomes
Front-End Developer Project: Memory Game	➔ This Project is all about demonstrating your mastery of HTML, CSS, and JavaScript. You'll build a complete browser-based card matching game (also known as Concentration). But this isn't just any memory game! It's a snazzy, well-designed, and feature-packed memory game!
Back-End Developer Project: Logs Analysis	➔ In this project, you'll practice your SQL skills by building a reporting tool that summarizes data from a large database.
Android Developer Project: Your First App	➔ In this project, you will build your first Android app - a design for a local business that could be used as a business card.
Data Analyst Project: Investigate a Dataset	➔ In this project, you will choose one of Udacity's curated datasets and investigate it using NumPy and pandas. You'll complete the entire data analysis process, starting by posing a question and finishing by sharing your findings.
iOS Developer Project: Pitch Perfect	➔ In this project, you will create an iPhone app that records a conversation with you and a friend and plays it back to make you sound like a chipmunk or Darth Vader!