UDACITY

NANODEGREE PROGRAM SYLLABUS

iOS Developer

Overview

The journey to becoming an iOS developer begins in your imagination—that moment when you first dream up a great idea for an app. This Nanodegree program will prepare you to publish your first iOS app, whether you're already programming or just beginning. As you master the Swift programming language and create a portfolio of apps to showcase your skills, you'll benefit from detailed code reviews, valuable career advice, and coaching from professional iOS developers.

IN COLLABORATION WITH

AT&T lyft Google

Estimated Time:
4 Months at
10hrs/week

Prerequisites:
Python & SQL

Flexible Learning:
Self-paced, so you can learn on the schedule that works best for you

Need Help?
udacity.com/advisor
Discuss this program with an enrollment advisor.

Course 1: Learn Swift Programming

You will complete a series of coding exercises to test your understanding of Swift. There will be exercises for variables, strings, if (else-if and else) statements, and functions.

	LEARNING OUTCOMES	
LESSON ONE	**Variables and Types**	• Declare variables and constant values with basic Swift types like Bool, Int, Double, and Float • Access and modify values from variables and constants • Debug compiler issues related to the incorrect use of variables and constants • Use escape characters and string interpolation to format variable and constant values within strings.
LESSON TWO	**Operators and expressions**	• Compute new values using existing variables and constants. • Use comparison operators to determine equality between two values. • Use boolean operators to build expressions that use truth values.
LESSON THREE	**Control Flow**	• Write boolean expressions that convey decision making logic. • Combine boolean expressions with logical operators. • Utilize boolean expressions alongside if, else-if, and else statements to control the flow of your code's execution. • Use switch statements to run code based on multiple values of a single variable. • Use for, while, and repeat while loops to control the flow of your code's execution.
LESSON FOUR	**Functions**	• Encapsulating existing code into reusable functions. • Properly define and call functions. • Specify function parameters and return types. • Differentiate between values that are in-scope and out-of-scope. • Correctly use local and external parameters. • Identify parameter types and return types.

LESSON FIVE	**Structures and Enum**	• Group multiple values together into structs. • Create instances of structs. • Add functions (known as methods) to structs. • Access properties and call methods of structs. • Define computed properties that calculate their value based on other values.. • Define enums and assign raw values to different cases. • Use enums in conjunction with switch statements.
LESSON SIX	**Optionals**	• Understand when a value can be nil and when to use an optional type. • Declare variables and constants as explicit or implicitly unwrapped optionals. • Unwrap optionals both safely and unsafely. • Use optional chaining and the nil coalescing operator to safely access optional values.
LESSON SEVEN	**Strings**	• Define and manipulate Strings using their built-in properties and methods • Perform common String operations like concatenation and finding substrings. • Perform common String manipulation such as adding, removing, and replacing substrings.
LESSON EIGHT	**Collections**	• Store unordered data of the same type using arrays. • Access and modify array contents. • Store pairs of keys and values using dictionaries. • Access and modify dictionary contents. • Store unordered data of the same type using sets.
LESSON NINE	**Object Oriented Programming**	• Understand the difference between value and reference types, and how this applies to structs and classes. • Make one class inherit the properties and methods of another class. • Understand polymorphism - how one type can be substituted for another type, and how this relates to inheritance • Write classes that conform to the same protocol. • Add additional functionality to classes using extensions.

Course 2: Intro to iOS App Development with Swift

Build your first app with Swift and Xcode, Apple's programming environment for app development. You'll learn how to use AutoLayout, UIButtons, and UILabels to create an interface, and how to react to touch events in an app using ViewController and multiple views. You'll also learn how to set up audio recording and playback in a voice recording app.

Course Project
Pitch Perfect

You will create an iPhone app that records audio and plays it back using various audio filters and modes including adjusted rate and pitch, echo, and reverb.

	LEARNING OUTCOMES	
LESSON ONE	**Introduction and Xcode**	• Navigate the major components of the Xcode development environment including the Navigator, Debug Area, and Utilities • Create an Xcode project for a new iOS application • Express the goals and architecture of the Model View Controller (MVC) design pattern
LESSON TWO	**AutoLayout and Buttons**	• Use Storyboards, Xcode's visual editing tool, to position, size, and configure user interface objects. • Link user interface objects in a Storyboard to their corresponding controller using IBOutlets. • Specify callback functions called IBActions that are invoked as a result of user interaction • Create AutoLayout constraints to ensure UI elements are sized and positioned correctly regardless of device size and dimensions
LESSON THREE	**ViewController and Multiple Views**	• Configure application state at the appropriate customizations points in a view's lifecycle • Create and navigate multiple-view applications using a UINavigationController • Manipulate user interface objects by utilizing IBOutlets and IBActions

LESSON FOUR	**Delegation and Recording**	• Write protocols to express functionality that can be adopted by Swift classes. • Use protocols to delegate the responsibilities of a particular task or set of tasks to another object. • Create and interface with an AVAudioRecorder to capture and save audio with an iOS device's microphone. • Use segues to transition between views in an application
LESSON FIVE	**Playback and Effects**	• Create and configure StackViews which contain and automatically configure layout constraints for its subviews • Playback audio using objects defined in the AVFoundation framework • Apply audio playback effects using audio nodes exposed by a custom interface
LESSON SIX	**Suggested Electives**	• Version Control with Git. • GitHub & Collaboration.

Course 3: UIKit Fundamentals

You will create a first version of the MemeMe app that enables a user to take a picture, and add text at the top and bottom to form a meme. The user will be able to share the photo on Facebook and Twitter and also by SMS or email.

Course Project
MemeMe 1.0: The Meme Editor

You will create a first version of the MemeMe app that enables a user to take a picture, and add text at the top and bottom to form a meme. The user will be able to share the photo on Facebook and Twitter and also by SMS or email.

Course Project
MemeMe 2.0: The Final Product

You will create an app that enables a user to take a picture, and add text at the top and bottom to form a meme. The user will be able to share the photo on Facebook and Twitter and also by SMS or email. Memes will appear in a tab view with two tabs: a table view and a collection view.

	LEARNING OUTCOMES	
LESSON ONE	**Outlets and Actions**	• Understand how to connect outlets and actions using only code and graphically using storyboard. • Use core UIKit classes like UIButton, UILabel and UISwitch. • Practice debugging problems with IBOutlets and IBActions.
LESSON TWO	**View Presentations and Segues**	• See how Apple distinguishes between modal presentation and navigation. • Learn how to present views modally. • Use powerful UIKit classes like UIImagePickerController, UIAlertController and UIActivityViewController.
LESSON THREE	**The Delegate Pattern**	• Learn how delegates make important connections between the model, view, and controller • Implement UIKit components that make use of the delegate pattern, UITextField and UITextFieldDelegate • Demonstrate your understanding by building a series of challenge apps.

LESSON FOUR	**Table Views**	• Learn the essential UITableViewDelegate and UITableViewDatasource methods. • Explore the code for several apps with tables, and then implement your own UITableView. • Practice manipulating table cells
LESSON FIVE	**Navigation**	• Learn how iOS uses navigation stacks to manage multiple views in an app. • Create the navigation that enables a user to tap a row of a table and view the details of an item. • Learn navigation classes like UINavigationControll and UIBarButtonItem.
LESSON SIX	**Suggested Electives**	• AutoLayout

Course 4: Network Requests and GCD

Incorporate networking into your apps, and harness the power of APIs to display images and retrieve data. Use Apple's Grand Central Dispatch, or GCD, framework to create asynchronous apps, ensuring a smooth user experience, even while your apps run lengthy operations in the background.

Course Project
On the Map

You will create an app with a map that shows information posted by other students. The map will contain pins that show the location where other students have reported studying. By tapping on the pin users can see a URL for something the student finds interesting. The user will be able to add their own data by posting a string that can be reverse geocoded to a location, and a URL.

	LEARNING OUTCOMES	
LESSON ONE	**Making a Network Request**	• Express the flow of data from a client to a server when a client makes an HTTP request. • Create a network request in Swift and receive and consume a data response. • Switch execution from a background thread to a (main) foreground thread to avoid blocking an app's UI • Abide by Apple's App Transport Security protocol to ensure user safety when access data over a network. • Download and display an image using a simple network request
LESSON TWO	**Using Web Services and APIs**	• Make requests to a web service (API) using documented endpoints and parameters. • Make a GET request to access data stored on a remote server. • Use a web service to download JSON data. • Convert raw byte data into JSON-like data that can be consumed by an app.
LESSON THREE	**Problem Set: JSON Parsing**	• Extract values from JSON objects and arrays • Access data from a locally defined JSON file

LESSON FOUR	**Chaining Asynchronous Requests**	• Perform multiple network requests in sequence using callbacks and closures.
LESSON FIVE	**Authenticating Requests**	• Perform an authorization flow that mimics OAuth. • Authenticate a network request using tokens. • Secure network requests by ensuring the use of HTTPS • Make a HTTP POST request to modify data stored by a remote server
LESSON SIX	**Improving Networking with MVC**	• Refactor an existing application to separate network functionality into its correct role within the MVC design pattern. • Create a usable interface that controllers can use to make network requests
LESSON SEVEN	**Closures Reloaded**	• Create closures by assigning functions to a constant or variable • Specify closure (function) types for use as values and parameters • Define functions which accept closure parameters • Use type aliasing to simplify the use of complex types • Define and use functions within functions
LESSON EIGHT	**GCD and Queues**	• Define and utilize queues for grouping related processes • Run code asynchronously using Grand Central Dispatch • Avoid common pitfalls by ensuring the use of the main thread for situations involving UIKit and CoreData
LESSON NINE	**Backgrounding Lengthy Tasks**	• Download large files from the network synchronously. • Download large files from the network asynchronously. • Use completion handlers to update the user interface after a network request.
LESSON TEN	**Suggested Electives**	• iOS Debugging.

UDACITY

Course 5: Data Persistence

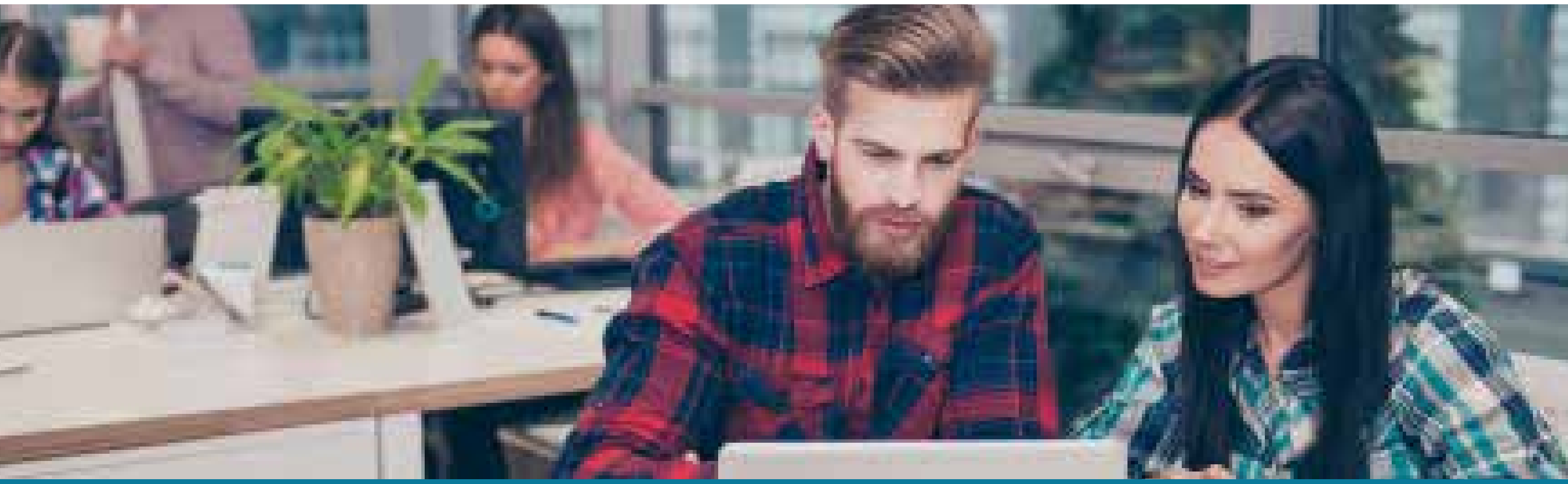
Learn about simple persistence, the iOS File System, and the "sandbox." Set up the classes we need to get Core Data up and running so that we can create, save, and delete model objects. Enable user interfaces to reactively update whenever the model changes, and safely migrate user data between versions.

Course Project
Virtual Tourist

You will create an app that downloads and stores images from Flickr. The app will allow users to drop pins on a map, as if they were stops on a tour. Users will then be able to download pictures for the location and persist both the pictures, and the association of the pictures with the pin.

	LEARNING OUTCOMES	
LESSON ONE	**Simple Persistence**	• Learn about simple persistence and how to save small pieces of data. • How to set user preferences, using NSUserDefaults. • Practice setting simple preferences to an existing app.
LESSON TWO	**iOS File System and Sandboxing**	• Learn about the iOS File System, the "sandbox" • See how to access these files using NSFileManager. • Use the file manager to save and read a file.
LESSON THREE	**Introducing Core Data**	• Meet Core Data, Apple's framework for managing the data layer. • Explore what a data layer is. • Convert a non-Core Data note-taking app to have a Core Data model.
LESSON FOUR	**The Core Data Stack**	• Set up the classes we need to get Core Data up and running. • Use the stack to manage model object creation and deletion. • Persist changes so that data stays put when you restart the app or device.

LESSON FIVE	**Simpler Code with Core Data**	• Enable user interfaces to reactively update whenever the model changes. • Set up an NSFetchedResultsController to observe data changes and notify the UI. • Modify a table view to work with a fetched results controller as its data source. • Turn on caching to reduce how often apps ask the store for data.
LESSON SIX	**Rounding Out Core Data**	• Update the data model and safely migrate user data between versions. • Work with multiple managed object contexts for different types of tasks. • Keep the user interface responsive by sending lengthy tasks to a background queue.
LESSON SEVEN	**Selective Electives**	• Objective-C for Swift Developers • Project 0-C: Interoperability Problem Set (Optional) • Firebase in a Weekend • Firebase Analytics



Course 6: Final Project

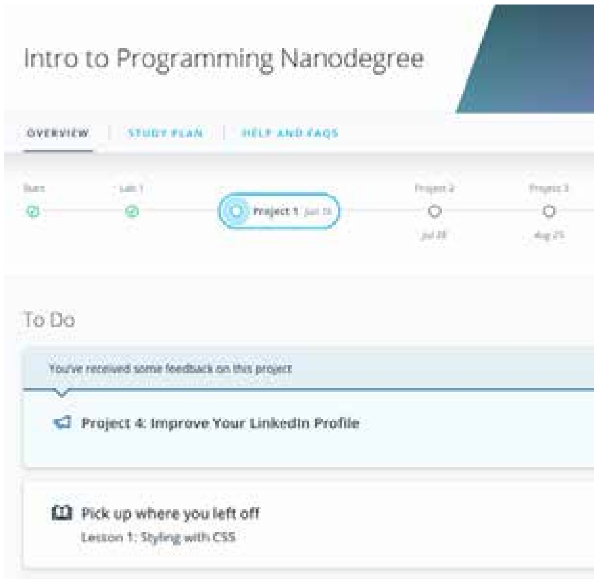
This is your chance to let your iOS Developer skills shine! For this final project, you'll design and build your own iOS app, taking the design from the drawing board to the App Store.

Course Project
You Decide! (Capstone Project)

This is your chance to let your iOS Developer skills shine! For this final project, you'll design your own iOS app, taking the design from drawing board to App Store.

	LEARNING OUTCOMES	
LESSON ONE	**Research**	• Brainstorm app ideas and decide on an app and feature list that is realistic and exciting. • Sketch UI storyboards and outline expected app use cases and flows. • Research and experiment with APIs, web services, and libraries that could be useful for an app idea.
LESSON TWO	**Build**	• Adhere to a proven development process to create quality iPhone and iPad apps. • Build an app and collect user feedback. • Fix crashes and bugs to improve the quality of an app.
LESSON THREE	**Reflect**	• Reflect on development, what has been learned, and what should change for future development. • Monitor App Store feedback.
LESSON FOUR	**Selective Electives**	• Technical Interview Prep. • Mobile Design Patterns.

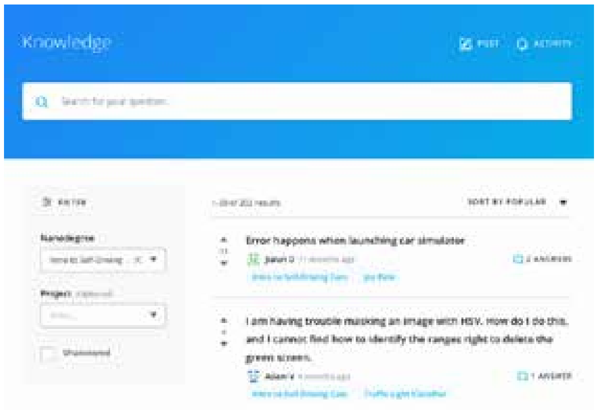
Our Classroom Experience


Intro to Programming Nanodegree
OVERVIEW
STUDY PLAN
HELP AND FAQS
Project 1
To Do
You've received some feedback on this project
Project 4: Improve Your LinkedIn Profile
Pick up where you left off
Lesson 1: Styling with CSS

REAL-WORLD PROJECTS
Build your skills through industry-relevant projects. Get personalized feedback from our network of 900+ project reviewers. Our simple interface makes it easy to submit your projects as often as you need and receive unlimited feedback on your work.

KNOWLEDGE
Find answers to your questions with Knowledge, our proprietary wiki. Search questions asked by other students, connect with technical mentors, and discover in real-time how to solve the challenges that you encounter.

STUDENT HUB
Leverage the power of community through a simple, yet powerful chat interface built within the classroom. Use Student Hub to connect with fellow students in your program as you support and learn from each other.

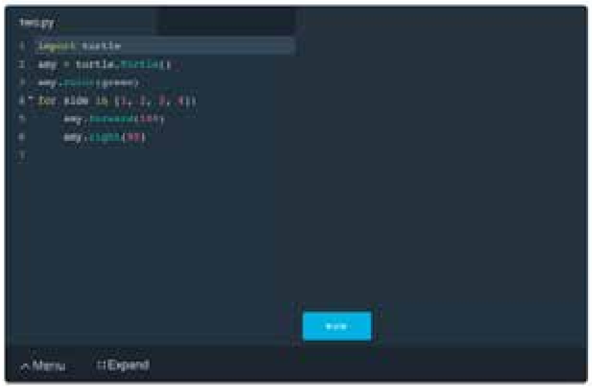


Error happens when launching car simulator

I am having trouble masking an image with HSV. How do I do this, and I cannot find how to identify the ranges right to delete the green screen.

WORKSPACES
See your code in action. Check the output and quality of your code by running them on workspaces that are a part of our classroom.

QUIZZES
Check your understanding of concepts learned in the program by answering simple and auto-graded quizzes. Easily go back to the lessons to brush up on concepts anytime you get an answer wrong.



```
import turtle
amy = turtle.Turtle()
amy.color('green')
for side in [1, 2, 3, 4]:
    amy.forward(100)
    amy.right(90)
```

CUSTOM STUDY PLANS
Preschedule your study times and save them to your personal calendar to create a custom study plan. Program regular reminders to keep track of your progress toward your goals and completion of your program.

PROGRESS TRACKER
Stay on track to complete your Nanodegree program with useful milestone reminders.

Learn with the Best

Jarrod Parkes

INSTRUCTOR

Jarrod is an experienced iOS developer with a passion for reinventing how students learn. He holds a BS in Computer Science from the University of Alabama.

Gabrielle Miller-Messner

INSTRUCTOR

Gabrielle earned her Ph.D. in Population Biology from UC Davis, where she discovered the joys of programming while analyzing DNA sequences. She has a background in teaching, and worked as an iOS Engineer before joining Udacity.

Kate Rotondo

INSTRUCTOR

Kate is an iOS developer, speaker, author, and teacher who has spoken at conferences across the globe from AltConf in San Francisco to Mobile Central Europe in Poland. She also has hosted a podcast on work-life integration for parents in tech.

Owen LaRosa

INSTRUCTOR

Owen is an iOS and Android app developer, and is the Student Experience Lead for iOS programs at Udacity. He graduated from the iOS Developer Nanodegree program in 2015.

UDACITY

All Our Nanodegree Programs Include:

EXPERIENCED PROJECT REVIEWERS

REVIEWER SERVICES

- Personalized feedback & line by line code reviews
- 1600+ Reviewers with a 4.85/5 average rating
- 3 hour average project review turnaround time
- Unlimited submissions and feedback loops
- Practical tips and industry best practices
- Additional suggested resources to improve

TECHNICAL MENTOR SUPPORT

MENTORSHIP SERVICES

- Questions answered quickly by our team of technical mentors
- 1000+ Mentors with a 4.7/5 average rating
- Support for all your technical questions

PERSONAL CAREER SERVICES

CAREER SUPPORT

- Resume support
- Github portfolio review
- LinkedIn profile optimization

Frequently Asked Questions

PROGRAM OVERVIEW

WHY SHOULD I ENROLL?

This Nanodegree program will prepare you to publish your first iOS app, whether you're already a developer or relatively new to programming.

In this program, you'll not only learn how to build iOS apps, you'll also learn best practices in mobile development, and gain mastery of Swift, an open-sourced object-oriented programming language. Through 6 hands-on, reviewed projects, you'll gain the skills you need to become an iOS Developer.

According to the 2017 Stack Overflow Job Trends Report, iOS Developers are among the Top-3 most in-demand developer positions in the job market. Enroll in this program today, and start building your future as an iOS Developer.

HOW DO I KNOW IF THIS PROGRAM IS RIGHT FOR ME?

This program is designed to prepare you for a job as a professional, junior-level iOS Developer within a wide range of organizations and environments: from large corporations where you'd likely be part of a development team, to entrepreneurial start-ups and contract projects where you could be working independently to deliver an application.

ENROLLMENT AND ADMISSION

DO I NEED TO APPLY? WHAT ARE THE ADMISSION CRITERIA?

No. This Nanodegree program accepts all applicants regardless of experience and specific background.

WHAT ARE THE PREREQUISITES FOR ENROLLMENT?

In order to succeed in this program, we recommend having the following experience:

- You are self-driven and motivated to learn. Participation in this program requires consistently meeting deadlines and devoting at least 10 hours per week to your work.
- Collaboration with peers and interactive feedback are critical to the success of the program. You must be a committed and contributing participant of the community.

Technical Requirements:

- Access to a Mac computer running macOS 10.14.3 or later

FAQs Continued

IF I DO NOT MEET THE REQUIREMENTS TO ENROLL, WHAT SHOULD I DO?
No programming experience is required, but if you'd like to try the Swift programming language, you may enjoy our free course, **Swift for Beginners**. This Nanodegree program includes coursework on using git and GitHub, but if you'd like exposure to git and GitHub before enrolling, you may wish to take our free course, **How to Use Git and GitHub**.

TUITION AND TERM OF PROGRAM

HOW IS THIS NANODEGREE PROGRAM STRUCTURED?
The iOS Nanodegree program is comprised of content and curriculum to support six (6)projects. We estimate that students can complete the program in six (6) months, working 10 hours per week.

Each project will be reviewed by the Udacity reviewer network. Feedback will be provided and if you do not pass the project, you will be asked to resubmit the project until it passes

HOW LONG IS THIS NANODEGREE PROGRAM?
Access to this Nanodegree program runs for the length of time specified in the payment card above. If you do not graduate within that time period, you will continue learning with month to month payments. See the **Terms of Use** and **FAQs** for other policies regarding the terms of access to our Nanodegree programs.

SOFTWARE AND HARDWARE

WHAT SOFTWARE AND VERSIONS WILL I NEED IN THIS PROGRAM?
Access to a Mac computer running macOS 10.14.3 or later