UDACITY
FOR ENTERPRISE

THE SCHOOL OF PROGRAMMING AND DEVELOPMENT

iOS Developer

NANODEGREE SYLLABUS

Overview

iOS Developer Nanodegree Program

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.

You will start by learning the basics of iOS app development using the Swift programming language and Xcode, Apple's development environment. You'll develop your first iOS apps using layouts, views, UIKit, and more.

Then, you'll progress to build more complex and advanced applications, using networking, and Apple's Grand Central Dispatch and Core Data, and will be ready to publish your capstone project to the App Store.

Program Information

ESTIMATED TIME
6 months
Study 10 hours/week

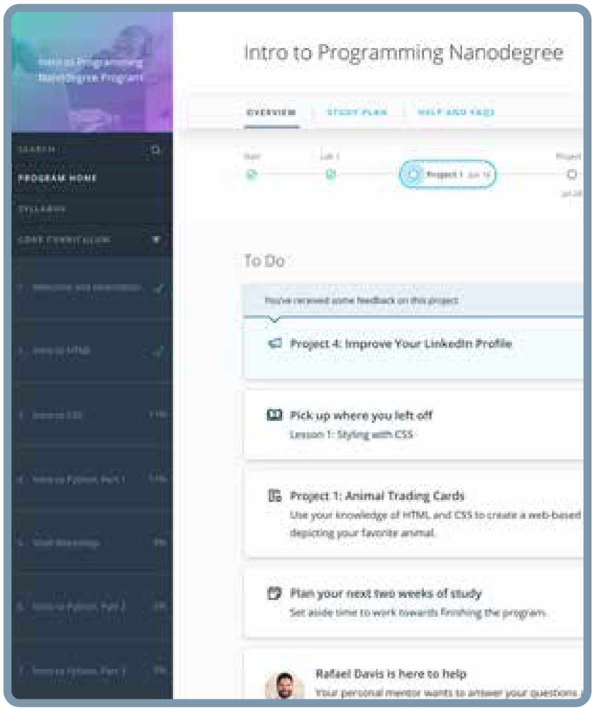
LEVEL
Foundational

PREREQUISITES
No programming experience is required, but if you'd like to try the Swift programming language, you can take the 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.

HARDWARE/SOFTWARE REQUIRED
Access to a Mac computer running macOS 10.14.3 or later

LEARN MORE ABOUT THIS NANODEGREE
Contact us at enterpriseNDs@udacity.com.

Our Classroom Experience



Intro to Programming Nanodegree

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

Project 1: Animal Trading Cards
Use your knowledge of HTML and CSS to create a web-based depicting your favorite animal.

Plan your next two weeks of study
Set aside time to work towards finishing the program.

Rafael Davis is here to help
Your personal mentor wants to answer your questions

REAL-WORLD PROJECTS

Learners build new skills through industry-relevant projects and receive personalized feedback from our network of 900+ project reviewers. Our simple user interface makes it easy to submit projects as often as needed and receive unlimited feedback.

KNOWLEDGE

Answers to most questions can be found with Knowledge, our proprietary wiki. Learners can search questions asked by others and discover in real-time how to solve challenges.

LEARNER HUB

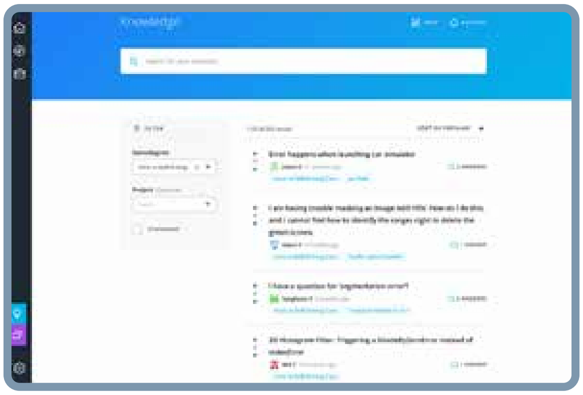
Learners leverage the power of community through a simple, yet powerful chat interface built within the classroom. Learner Hub connects learners with their technical mentor and fellow learners.

WORKSPACES

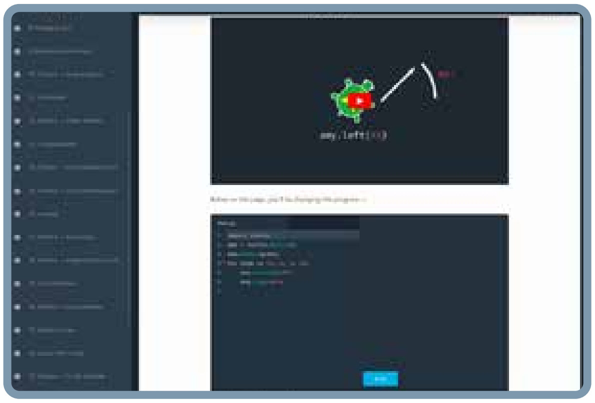
Learners can check the output and quality of their code by testing it on interactive workspaces that are integrated into the classroom.

QUIZZES

Understanding concepts learned during lessons is made simple with auto-graded quizzes. Learners can easily go back and brush up on concepts at anytime during the course.



Error happens when launching car simulator



CUSTOM STUDY PLANS

Mentors create a custom study plan tailored to learners' needs. This plan keeps track of progress toward learner goals.

PROGRESS TRACKER

Personalized milestone reminders help learners stay on track and focused as they work to complete their Nanodegree program.

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.

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.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
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.
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-ofscope. • Correctly use local and external parameters. • Identify parameter types and return types.

Nanodegree Program Overview

Course 1: Learn Swift Programming, cont.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
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.
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.
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 recordingand playback in a voice recording app.

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.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
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.

Course 2: Intro to iOS App Development with Swift, cont.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
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.

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.

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.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
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.

Course 3: UIKit Fundamentals, cont.

LESSON TITLE	LEARNING OUTCOMES
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
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.
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.

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.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
PROBLEM SET: JSON PARSING	• Extract values from JSON objects and arrays. • Access data from a locally defined JSON file.

Course 4: Network Requests and GCD, cont.

LESSON TITLE	LEARNING OUTCOMES
CHAINING ASYNCHRONOUS REQUESTS	• Perform multiple network requests in sequence using callbacks and closures.
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.
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.
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.
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.
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.
SUGGESTED ELECTIVES	• iOS Debugging.

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.

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.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
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.
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.

Course 5: Data Persistence, cont.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
SELECTIVE ELECTIVES	• Objective-C for Swift Developers. » Project 0-C: Interoperability Problem Set (Optional). • Firebase in a Weekend. • Firebase Analytics.

Course 6: Final Project

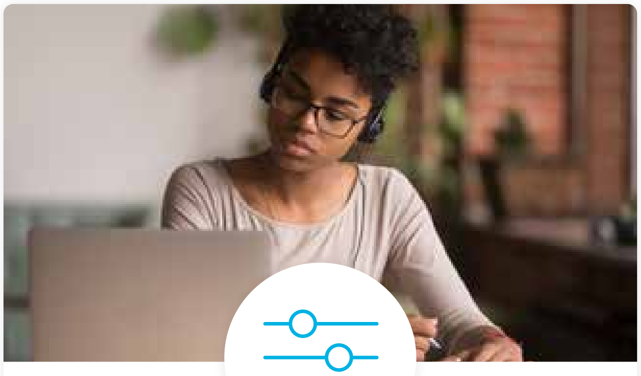
Project

Capstone 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.

LESSON TITLE	LEARNING OUTCOMES
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.
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.
REFLECT	• Reflect on development, what has been learned, and what should change for future development. • Monitor App Store feedback.
SELECTIVE ELECTIVES	• Technical Interview Prep. • Mobile Design Patterns.

Our Nanodegree Programs Include:



Pre-Assessments

Our in-depth workforce assessments identify your team's current level of knowledge in key areas. Results are used to generate custom learning paths designed to equip your workforce with the most applicable skill sets.

Dashboard & Progress Reports

Our interactive dashboard (enterprise management console) allows administrators to manage employee onboarding, track course progress, perform bulk enrollments and more.

Industry Validation & Reviews

Learners' progress and subject knowledge is tested and validated by industry experts and leaders from our advisory board. These in-depth reviews ensure your teams have achieved competency.

Real World Hands-on Projects

Through a series of rigorous, real-world projects, your employees learn and apply new techniques, analyze results, and produce actionable insights. Project portfolios demonstrate learners' growing proficiency and subject mastery.

Our Review Process

Real-life Reviewers for Real-life Projects

Real-world projects are at the core of our Nanodegree programs because hands-on learning is the best way to master a new skill. Receiving relevant feedback from an industry expert is a critical part of that learning process, and infinitely more useful than that from peers or automated grading systems. Udacity has a network of over 900 experienced project reviewers who provide personalized and timely feedback to help all learners succeed.

Vaibhav

UDACITY LEARNER

"I never felt overwhelmed while pursuing the Nanodegree program due to the valuable support of the reviewers, and now I am more confident in converting my ideas to reality."

now at

CODING VISIONS INFOTECH

All Learners Benefit From:

Line-by-line feedback for coding projects

Industry tips and best practices

Advice on additional resources to research

Unlimited submissions and feedback loops

How it Works

Real-world projects are integrated within the classroom experience, making for a seamless review process flow.

- Go through the lessons and work on the projects that follow
- Get help from your technical mentor, if needed
- Submit your project work
- Receive personalized feedback from the reviewer
- If the submission is not satisfactory, resubmit your project
- Continue submitting and receiving feedback from the reviewer until you successfully complete your project

About our Project Reviewers

Our expert project reviewers are evaluated against the highest standards and graded based on learners' progress. Here's how they measure up to ensure your success.

900+

Expert Project Reviewers

Are hand-picked to provide detailed feedback on your project submissions.

1.8M

Projects Reviewed

Our reviewers have extensive experience in guiding learners through their course projects.

3

Hours Average Turnaround

You can resubmit your project on the same day for additional feedback.

4.85 /5

Average Reviewer Rating

Our learners love the quality of the feedback they receive from our experienced reviewers.

2440 W El Camino Real, #101
Mountain View, CA 94040, USA - HQ

For more information visit: **www.udacity.com/enterprise**