Intro to App Development with Swift Resource Guide

February 2017

iBooks File Edit View Go Store Window Help
Wed 9:41 AM
App Development with Swift
MacBook

Lesson 4

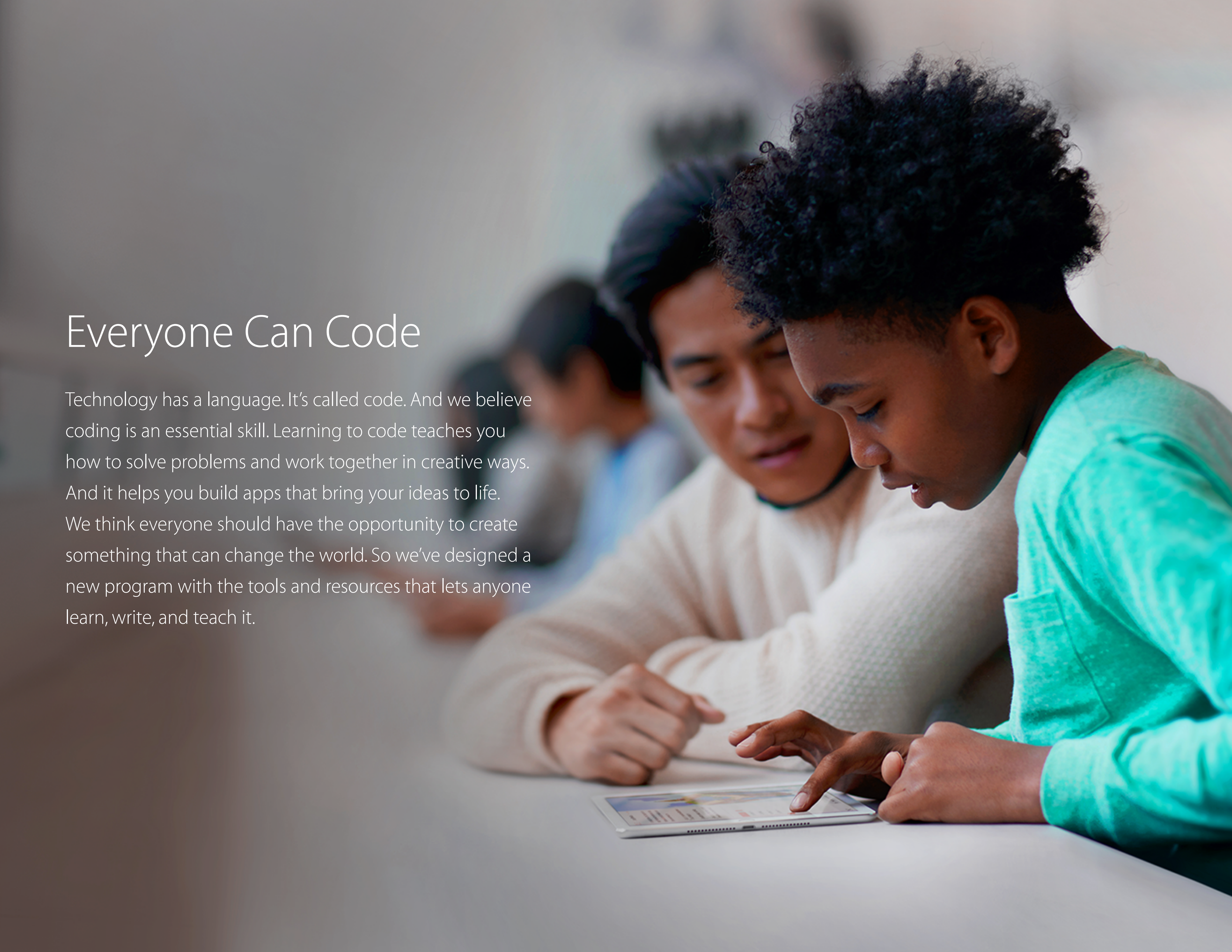
Hello, world!

WHAT YOU'LL DO
- Participate in a tradition that will welcome you into the ranks of programmers the world over

WHAT YOU'LL LEARN
- How programmers leave messages for each other
- How you can leave messages for yourself
- Where you can look for extra information about errors in your code
- How to identify bugs in code

WHAT YOU'LL WONDER
- How will I use this console outside of a playground?

KEY VOCABULARY
- Console
- Debugging
- Error
- Log
- Print

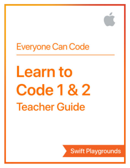
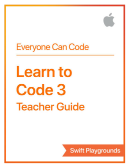


Everyone Can Code

Technology has a language. It's called code. And we believe coding is an essential skill. Learning to code teaches you how to solve problems and work together in creative ways. And it helps you build apps that bring your ideas to life. We think everyone should have the opportunity to create something that can change the world. So we've designed a new program with the tools and resources that lets anyone learn, write, and teach it.

Everyone Can Code Program

The Everyone Can Code program includes a range of resources that take students all the way from no coding experience to building their first apps. The table below provides an overview of all the free teaching and learning resources available.

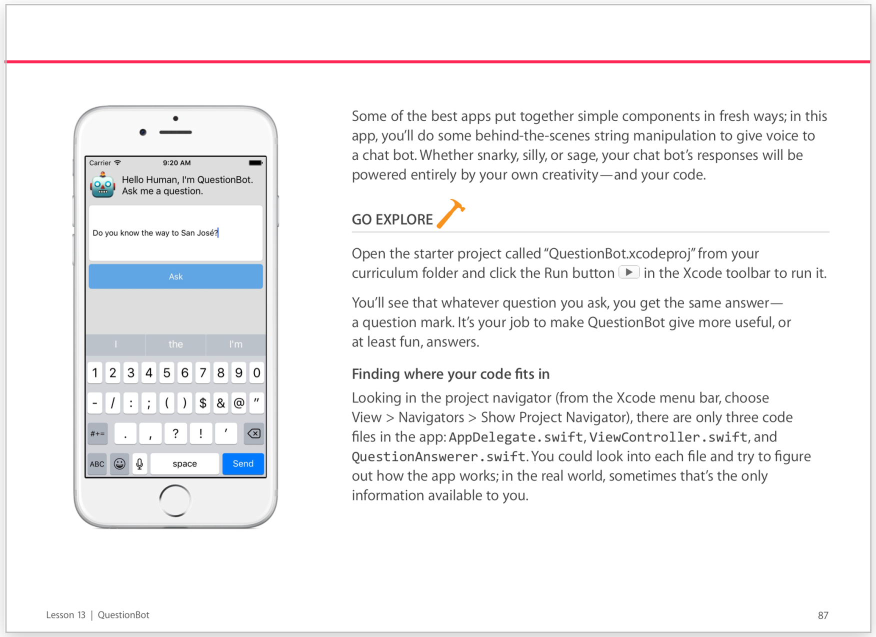
Resource	Device	App	Audience	Prerequisites	Overview	Learning materials	Support resources	Number of lesson hours included
 Everyone Can Code Learn to Code 1 & 2 Teacher Guide Swift Playgrounds			Middle school and up	None	Learn fundamental coding concepts using real Swift code.	• Swift Playgrounds app • Learn to Code 1 & 2 lessons • iTunes U course	• Learn to Code 1 & 2: Teacher Guide	45 hours, including Teacher Guide and Learn to Code 1 & 2 lessons
 Everyone Can Code Learn to Code 3 Teacher Guide Swift Playgrounds			Middle school and up	Learn to Code 1 & 2	Expand coding skills and start thinking more like an app developer.	• Swift Playgrounds app • Learn to Code 3 lessons	• Learn to Code 3: Teacher Guide	20 hours, including Teacher Guide and Learn to Code 3 lessons
Everyone Can Code Intro to App Development with Swift Xcode			High school and college	None	Get practical experience with the tools, techniques, and concepts needed to build a basic iOS app from scratch.	Intro to App Development with Swift book and Xcode project files	• Intro to App Development with Swift: Teacher Guide • Professional learning workshops	90 hours

Overview

Intro to App Development with Swift is designed to help students new to coding build a solid foundation in programming fundamentals using Swift as the language. Throughout this course, students get practical experience with the tools, techniques, and concepts needed to build a basic iOS app from scratch. Students will also learn user interface design principles, which are fundamental to programming and making great apps.

Swift is the powerful and intuitive programming language created by Apple for building apps. It makes programming easier, more flexible, and more fun. Swift is not only great for getting you started with coding, it's also super powerful. It's designed to scale from writing the simplest program, like "Hello, world!" to the world's most advanced software.

Xcode is the Mac app used to build every other Mac app and every iOS app, too. It has all the tools you need to create an amazing app experience. And it's available as a free download from the Mac App Store.


Carrier
9:20 AM
Hello Human, I'm QuestionBot. Ask me a question.
Do you know the way to San José?
Ask
space
Send

Some of the best apps put together simple components in fresh ways; in this app, you'll do some behind-the-scenes string manipulation to give voice to a chat bot. Whether snarky, silly, or sage, your chat bot's responses will be powered entirely by your own creativity—and your code.

GO EXPLORE

Open the starter project called "QuestionBot.xcodeproj" from your curriculum folder and click the Run button ▶ in the Xcode toolbar to run it.

You'll see that whatever question you ask, you get the same answer—a question mark. It's your job to make QuestionBot give more useful, or at least fun, answers.

Finding where your code fits in

Looking in the project navigator (from the Xcode menu bar, choose View > Navigators > Show Project Navigator), there are only three code files in the app: `AppDelegate.swift`, `ViewController.swift`, and `QuestionAnswerer.swift`. You could look into each file and try to figure out how the app works; in the real world, sometimes that's the only information available to you.

Lesson 13 | QuestionBot 87

In the classroom

Intro to App Development with Swift is designed to use with high school and college students new to programming. It can be used in a stand-alone class or as part of any app development or computer science program. The Teacher Guide will help you bring Intro to App Development with Swift into the classroom. The lessons were designed to both highlight key coding concepts and to demonstrate how coding is a way of thinking that can be applied and understood through everyday life scenarios. Correlation maps are included in the appendix, which provide alignment of the lessons to the British Columbia Applied Design, Skills and Technologies grades 9 and 10 standards, the British Columbia Computer Programming grades 11 and 12 standards, British Columbia Math grades 11 and 12 standards, as well as to the US Computer Science Teachers Association's Interim Computer Science Standards for Level 3A.

What's Included

Playgrounds. Students learn programming concepts as they write code in playgrounds—an interactive coding environment that lets them experiment with code and see the results immediately. Playground files are provided.

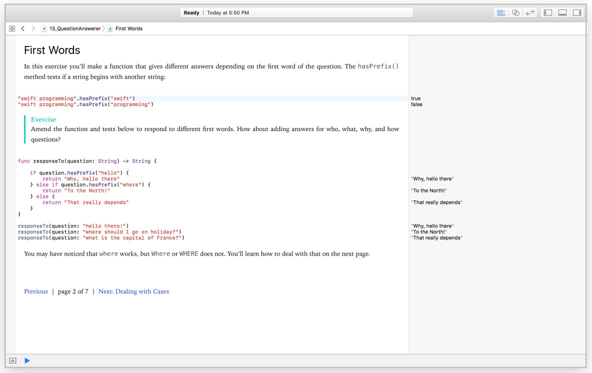
Step-by-step instructions. Students are guided through all the steps of building an app in Xcode with detailed instructions that include videos.

Sample Xcode projects. Xcode project files are included so that students can experiment with certain parts of code without having to build an entire app from scratch.

Build apps. The course takes students through the steps of building a variety of apps, including a chat bot that responds to different questions, a colour-picker app, a chemical elements quiz, and more.

App journal activities. These activities take students through the beginning of the app design process, from thinking about the purpose of an app to market research and early user testing.

Review and reflect. Students can answer questions that help them check their understanding and apply what they've learned.



First Words

In this exercise you'll make a function that gives different answers depending on the first word of the question. The `hasPrefix()` method tests if a string begins with another string:

```
"swift programming".hasPrefix("swift")                          true
"swift programming".hasPrefix("programming")                    false
```

Exercise
Amend the function and tests below to respond to different first words. How about adding answers for who, what, why, and how questions?

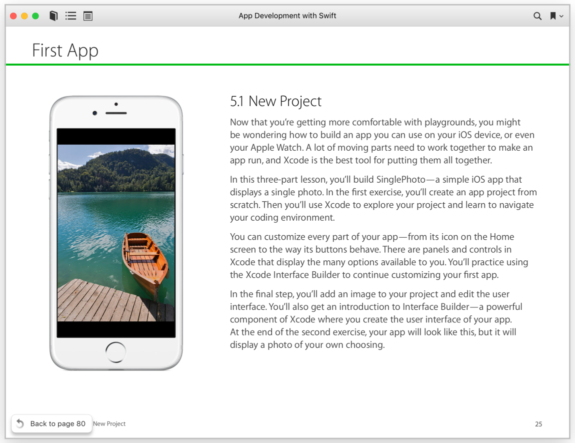
```
func responseTo(question: String) -> String {

    if question.hasPrefix("hello") {
        return "Why, hello there"                               "Why, hello there"
    } else if question.hasPrefix("where") {
        return "To the North!"                                  "To the North!"
    } else {
        return "That really depends"                            "That really depends"
    }
}

responseTo(question: "hello there!")                            "Why, hello there"
responseTo(question: "where should I go on holiday?")           "To the North!"
responseTo(question: "what is the capital of France?")          "That really depends"
```

You may have noticed that `where` works, but `Where` or `WHERE` does not. You'll learn how to deal with that on the next page.

Previous | page 2 of 7 | Next: Dealing with Cases



First App

5.1 New Project

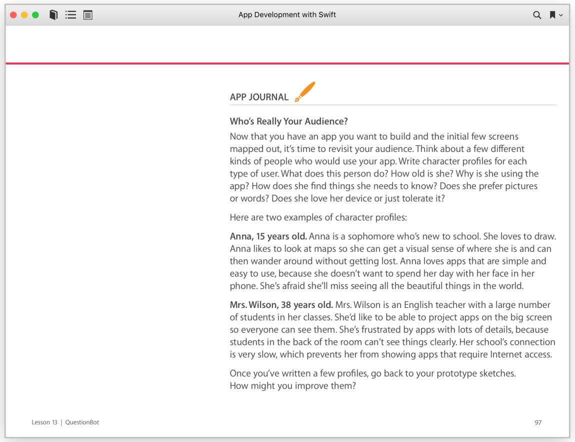
Now that you're getting more comfortable with playgrounds, you might be wondering how to build an app you can use on your iOS device, or even your Apple Watch. A lot of moving parts need to work together to make an app run, and Xcode is the best tool for putting them all together.

In this three-part lesson, you'll build SinglePhoto—a simple iOS app that displays a single photo. In the first exercise, you'll create an app project from scratch. Then you'll use Xcode to explore your project and learn to navigate your coding environment.

You can customize every part of your app—from its icon on the Home screen to the way its buttons behave. There are panels and controls in Xcode that display the many options available to you. You'll practice using the Xcode Interface Builder to continue customizing your first app.

In the final step, you'll add an image to your project and edit the user interface. You'll also get an introduction to Interface Builder—a powerful component of Xcode where you create the user interface of your app. At the end of the second exercise, your app will look like this, but it will display a photo of your own choosing.

Back to page 80 New Project 25



APP JOURNAL

Who's Really Your Audience?

Now that you have an app you want to build and the initial few screens mapped out, it's time to revisit your audience. Think about a few different kinds of people who would use your app. Write character profiles for each type of user. What does this person do? How old is she? Why is she using the app? How does she find things she needs to know? Does she prefer pictures or words? Does she love her device or just tolerate it?

Here are two examples of character profiles:

Anna, 15 years old. Anna is a sophomore who's new to school. She loves to draw. Anna likes to look at maps so she can get a visual sense of where she is and can then wander around without getting lost. Anna loves apps that are simple and easy to use, because she doesn't want to spend her day with her face in her phone. She's afraid she'll miss seeing all the beautiful things in the world.

Mrs. Wilson, 38 years old. Mrs. Wilson is an English teacher with a large number of students in her classes. She'd like to be able to project apps on the big screen so everyone can see them. She's frustrated by apps with lots of details, because students in the back of the room can't see things clearly. Her school's connection is very slow, which prevents her from showing apps that require Internet access.

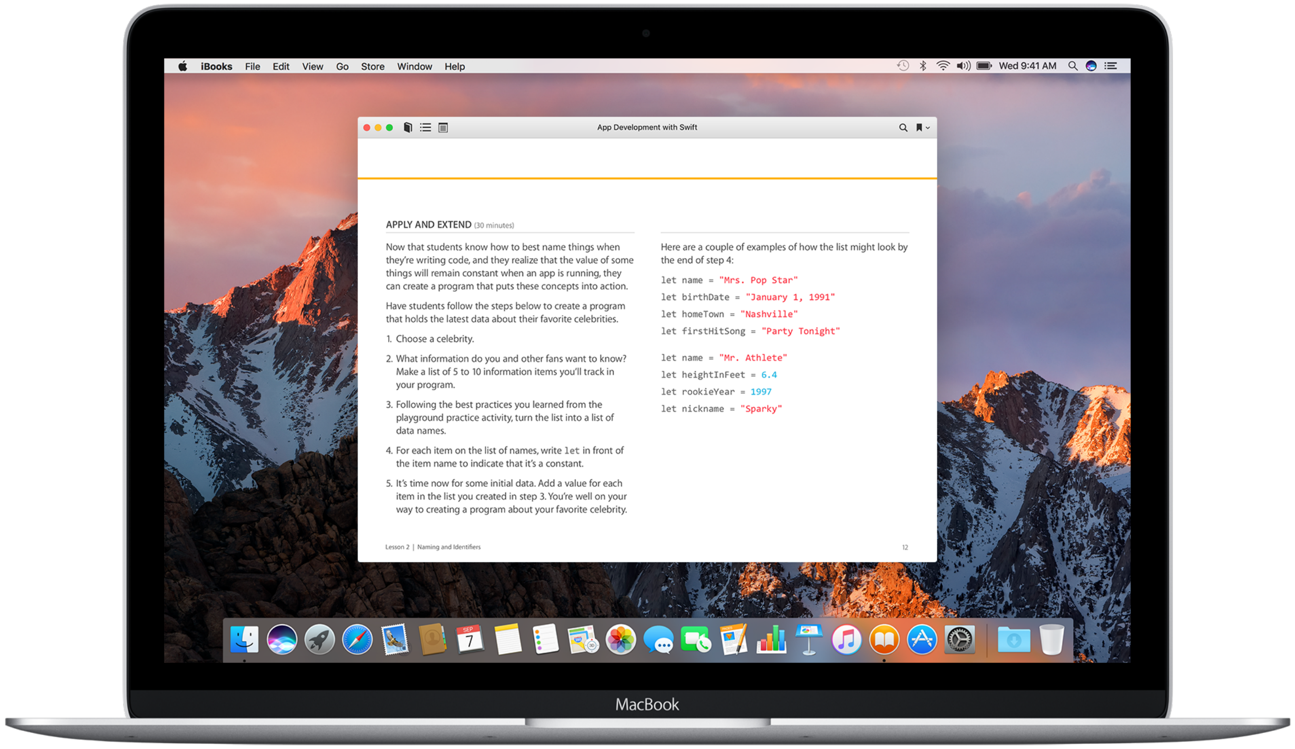
Once you've written a few profiles, go back to your prototype sketches. How might you improve them?

Lesson 13 | QuestionBot 97

Support Resources

Intro to App Development with Swift: Teacher Guide

The Teacher Guide includes extension and application activities, discussion questions, and activities for an app journal. It's designed to cover one semester of instruction for high school and up. Correlation maps are included in the appendix, which provide alignment of the lessons to the British Columbia Applied Design, Skills and Technologies grades 9 and 10 standards, the British Columbia Computer Programming grades 11 and 12 standards, British Columbia Math grades 11 and 12 standards, as well as to the US Computer Science Teachers Association's Interim Computer Science Standards for Level 3A.


iBooks File Edit View Go Store Window Help
Wed 9:41 AM
App Development with Swift
MacBook

APPLY AND EXTEND (30 minutes)

Now that students know how to best name things when they're writing code, and they realize that the value of some things will remain constant when an app is running, they can create a program that puts these concepts into action.

Have students follow the steps below to create a program that holds the latest data about their favorite celebrities.

1. Choose a celebrity.
2. What information do you and other fans want to know? Make a list of 5 to 10 information items you'll track in your program.
3. Following the best practices you learned from the playground practice activity, turn the list into a list of data names.
4. For each item on the list of names, write `let` in front of the item name to indicate that it's a constant.
5. It's time now for some initial data. Add a value for each item in the list you created in step 3. You're well on your way to creating a program about your favorite celebrity.

Here are a couple of examples of how the list might look by the end of step 4:

```
let name = "Mrs. Pop Star"
let birthDate = "January 1, 1991"
let homeTown = "Nashville"
let firstHitSong = "Party Tonight"

let name = "Mr. Athlete"
let heightInFeet = 6.4
let rookieYear = 1997
let nickname = "Sparky"
```

Lesson 2 | Naming and Identifiers   12

Course Outline

Intro to App Development with Swift

Lesson 1: Playground Basics. Students gain familiarity with the interactive playground environment.

Lesson 2: Naming and Identifiers. Students explore the fundamentals of solving problems by using good names and identifiers.

Lesson 3: Strings. Students are introduced to the concept of strings and string interpolation.

Lesson 4: Hello, world! Students are welcomed to the tradition of programming, learning to customize their Xcode environment and how to debug.

Lesson 5: First App. Students create their first app using Xcode, displaying their work in an iOS simulator.

Lesson 6: Functions. Students learn what makes functions so powerful as they combine detailed steps into a definition they can use again and again.

Lesson 7: BoogieBot. Students put their knowledge of functions to work by controlling an animated dancing robot within the playground.

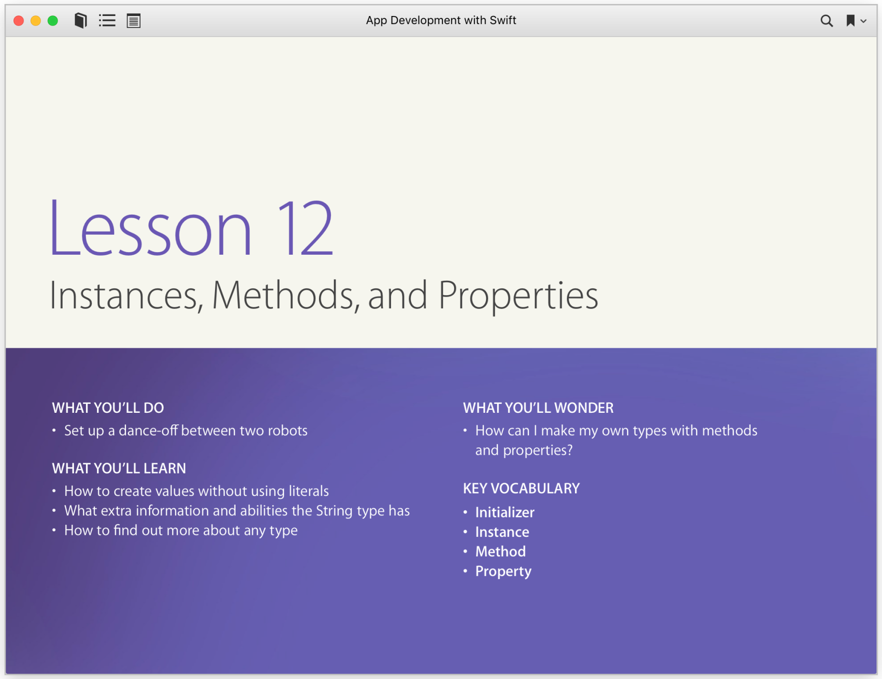
Lesson 8: Constants and Variables. Students expand their understanding of naming as they're formally introduced to the concepts of constants and variables.

Lesson 9: Types. Students become more familiar with the underpinnings of Swift by learning about the type system, from the standard library in Swift to custom types.

Lesson 10: Parameters and Results. Students expand their knowledge of functions by learning about parameters and return values to make functions more flexible and powerful.

Lesson 11: Making Decisions. Students learn how to make decisions in code using conditional if/else statements, true or false Bool values, and comparison operators.

Lesson 12: Instances, Methods, and Properties. Students build on their knowledge of types by learning about the methods and properties that make up an instance of that type.



Lesson 12

Instances, Methods, and Properties

WHAT YOU'LL DO
- Set up a dance-off between two robots

WHAT YOU'LL LEARN
- How to create values without using literals
- What extra information and abilities the String type has
- How to find out more about any type

WHAT YOU'LL WONDER
- How can I make my own types with methods and properties?

KEY VOCABULARY
- **Initializer**
- **Instance**
- **Method**
- **Property**

Course Outline (continued)

Lesson 13: QuestionBot. Students will get experience modifying an existing Xcode project by writing new logic for an app bot that responds to different questions.

Lesson 14: Arrays and Loops. Students learn how to create and work with arrays by adding and removing objects, and they learn how for-loops work with each object in an array.

Lesson 15: Defining Structures. Students will recognize that it's often useful to group related information and functionality into a custom type.

Lesson 16: QuestionBot 2. Students expand on the QuestionBot app by building ChatBot, an app that displays the history of the conversation. They'll learn about the data source pattern, and build a simple data source object to provide information on Message objects to display in the message list view. Students practice appending to an array to store messages on the data source object to maintain a history of the conversation.

Lesson 17: Actions and Outlets. Students will learn how to build user interfaces using Interface Builder, and tie user interface elements into code via Outlets and Actions. They'll practice creating Outlets to access properties of a user interface view, and Actions to respond to user interaction with buttons and other controls.

Lesson 18: Adaptive User Interfaces. Students will learn a repeatable process to create a user interface on the smallest iPhone device size that scales up to all iPhone device sizes and orientations. They'll learn about auto layout, the system for laying out constraints that set the location and size of user interface elements. And they'll use stack views, a special object designed to automatically set auto layout constraints based on simpler settings and a gridlike system. In the process, they build the SimpleCenter, ElementQuiz, and AnimalSounds apps.

Lesson 19: Enumerations and Switch. Enumerations, or enums, are a way to define a named list of options. Students will learn what enums are used for, how to define them, and common ways to work with them. They'll also learn to use the switch statement to conditionally run specific code based on any option that an enum defines.

Lesson 20: Final Project. Students will complete one or both final project options from scratch. The first option is a Rock/Paper/Scissors game. The second option is a Meme generator. Students will review a variety of concepts covered in the course, build the user interface, the model data, and the controller objects that make up the entire app.

Lesson 21: What's Next? Students explore a wide range of app development resources, from the Apple Developer home page to videos from the Apple Worldwide Developers Conference on the latest frameworks and tools for building apps for all Apple platforms.

Requirements

Students will need the following to complete the lessons in the guide:

- A Mac running macOS Sierra or El Capitan
- Xcode 8
- Project files for the course, which are available via a download link in the book

Curriculum Alignment

Here is the alignment of the lessons of Intro to App Dev with Swift to the British Columbia Computer Programming grades 11 and 12 standards.

British Columbia Ministry of Education Intro to App Development with Swift CORRELATION MAP	Applied Design, Skills, and Technologies – Computer Programming (Grade 11)								
	Understand: Big Ideas			Do: Curricular Competencies			Know: Content Learning Standards		
Students are expected to be able to do the following:	Understand context ⓘ	Defining ⓘ	Ideating ⓘ	Prototyping ⓘ	Testing ⓘ	Making ⓘ	Sharing ⓘ	Applied Skills ⓘ	Applied Technology ⓘ
Overall Alignment	●	●	●	●	●	◔	◕	◔	◔
Constants and Variables ⓘ			◕		◑	◑	◔		◔
Types ⓘ	●	◑		◔	◕				
Parameters and Results ⓘ	◕	◑							
Making Decisions ⓘ	◕	◕		◔					
Instances, Methods and Properties ⓘ	◔	◑	◔	●		◔			
Question Bot ⓘ	◔	◕	◑	◔		◔			
Arrays and Loops ⓘ		◕		◕	◕		◕		
Defining Structures ⓘ			◔		◑		◑		
Question Bot 2 ⓘ			◔	●		◔	◔		
Actions and Outlets ⓘ				●	●	◔	◔		
Adaptive User Interfaces ⓘ	◔		◔		◑	◔	◑		
Enumerations and Switch ⓘ	◑	◑	◑	◔		◔			◔
Final Project ⓘ	●	◔	◑	●	◑		◕	◔	

KEY ⓘ Information ● 100% coverage ◕ 75% coverage ◑ 50% coverage ◔ 25% coverage ○ Further Notes

Curriculum Alignment

Here is the alignment of the lessons of Intro to App Dev with Swift to the British Columbia Computer Programming grades 11 and 12 standards.

British Columbia Ministry of Education Intro to App Development with Swift CORRELATION MAP		Applied Design, Skills, and Technologies – Computer Programming (Grade 11)			
		Understand: Big Ideas	Do: Curricular Competencies	Know: Content Learning Standards	
Students are expected to be able to know the following:					
structures within existing code	ways to modify existing code to meet a particular purpose	strategies to predict effects of code modification	programming language constructs to support input/output, logic, decision structure, and loops	requirements of a problem statement	ways to transform requirements into algorithms
●	●	◕	●	○	●
translation of design specifications into source code	tools to aid in the development process	pre-built libraries and their documentation	inline commenting to document source code	use of test cases to detect logical or semantical errors	
◑	○	●	●	●	

KEY ⓘ Information ● 100% coverage ◕ 75% coverage ◑ 50% coverage ◔ 25% coverage ○ Further Notes

Curriculum Alignment

Here is the preliminary alignment of Intro to App Development with Swift with the Computer Science Teachers Association (CSTA) Interim Computer Science Standards for Level 3A. Once the new standards are finalized, the guide will undergo CSTA's formal crosswalk review. The alignment covers the algorithms and programming concepts within the Computer Science Interim 2016 CSTA K–12 Standards.

CSTA K–12 Computer Science Standards Level 3A for Grades 9–10												
CSTA Standard	3A-A-2-1 Design Artifact	3A-A-2-2 Collaborating	3A-A-5-4 Licensing	3A-A-7-4 Respond to Event	3A-A-5-5 Research	3A-A-5-6 Mathematical Concepts	3A-A-4-7 Hierarchy & Abstraction	3A-A-4-8 Deconstruct Problem	3A-A-4-9 Abstraction	3A-A-3-10 Design Algorithms	3A-A-3-11 Modelling & Simulation	3A-A-6-12 Debugging
Playground Basics	●	●	●	●	●	●	●	●	●	●	●	●
Naming and Identifiers	●	●	●	●	●	●	●	●	●	●	●	●
Strings	●	●	●	●	●	●	●	●	●	●	●	●
Hello, world!	●	●	●	●	●	●	●	●	●	●	●	●
First App	●	●	●	●	●	●	●	●	●	●	●	●
Functions	●	●	●	●	●	●	●	●	●	●	●	●
BoogieBot	●	●	●	●	●	●	●	●	●	●	●	●
Constants and Variables	●	●	●	●	●	●	●	●	●	●	●	●
Types	●	●	●	●	●	●	●	●	●	●	●	●
Parameters and Results	●	●	●	●	●	●	●	●	●	●	●	●
Making Decisions	●	●	●	●	●	●	●	●	●	●	●	●
Instances, Methods, and Properties	●	●	●	●	●	●	●	●	●	●	●	●
QuestionBot	●	●	●	●	●	●	●	●	●	●	●	●
Arrays and Loops	●	●	●	●	●	●	●	●	●	●	●	●
Defining Structures	●	●	●	●	●	●	●	●	●	●	●	●
QuestionBot 2	●	●	●	●	●	●	●	●	●	●	●	●
Actions and Outlets	●	●	●	●	●	●	●	●	●	●	●	●
Adaptive User Interfaces	●	●	●	●	●	●	●	●	●	●	●	●
Enumerations and Switch	●	●	●	●	●	●	●	●	●	●	●	●
Final Project	●	●	●	●	●	●	●	●	●	●	●	●

Key: ● Aligns to standard

Additional Information

Download the Swift Playgrounds resources

- Learn to Code 1 & 2: Teacher Guide
- Learn to Code 3: Teacher Guide
- Swift Playgrounds app

Download the Intro App Development with Swift guides

- Intro to App Development with Swift: Student Guide
- Intro to App Development with Swift: Teacher Guide

Additional resources

- Learn more about the Everyone Can Code program.
- Learn more about Swift.
- Learn more about Xcode.
- Connect with other educators in the Apple Developer Forums.