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 programme with the tools and resources that let anyone learn, write and teach it.
# Everyone Can Code Curriculum

The Everyone Can Code programme 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.

<table>
<thead>
<tr>
<th>Curriculum</th>
<th>Device</th>
<th>Audience</th>
<th>App</th>
<th>Prerequisites</th>
<th>Overview</th>
<th>Learning materials</th>
<th>Support resources</th>
<th>Number of lesson hours included</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Started with Code 1</td>
<td>☑️</td>
<td>Years 1 to 3</td>
<td>codeSpark Academy</td>
<td>None</td>
<td>Begin to think like coders with hands-on explorations of coding concepts using visual-based apps.</td>
<td>• codeSpark Academy app lessons</td>
<td>• Get Started with Code 1: Teacher Guide</td>
<td>30 hours, including Teacher Guide and app lessons</td>
</tr>
<tr>
<td>Get Started with Code 2</td>
<td>☑️</td>
<td>Years 4 to 6</td>
<td>Tynker Space Cadet</td>
<td>None</td>
<td>Explore fundamental coding concepts and practise thinking like coders using visual-based apps.</td>
<td>• Tynker Dragon Spells course</td>
<td>• Get Started with Code 2: Teacher Guide</td>
<td>36 hours, including Teacher Guide and app lessons</td>
</tr>
<tr>
<td>Learn to Code 1 &amp; 2</td>
<td>☑️</td>
<td>Year 7 and up</td>
<td>Swift Playgrounds</td>
<td>None</td>
<td>Learn fundamental coding concepts using real Swift code.</td>
<td>• Swift Playgrounds app</td>
<td>• Learn to Code 1 &amp; 2: Teacher Guide</td>
<td>Up to 85 hours, including Teacher Guide and Learn to Code 1 &amp; 2 lessons</td>
</tr>
<tr>
<td>Learn to Code 3</td>
<td>☑️</td>
<td>Year 7 and up</td>
<td>Learn to Code 3</td>
<td>Learn to Code 1 &amp; 2</td>
<td>Expand coding skills and start thinking more like an app developer.</td>
<td>• Swift Playgrounds app</td>
<td>• Learn to Code 3: Teacher Guide</td>
<td>Up to 45 hours, including Teacher Guide and Learn to Code 3 lessons</td>
</tr>
<tr>
<td>Intro to App Development with Swift</td>
<td>☑️</td>
<td>Year 10 and up</td>
<td>Intro to App Development with Swift</td>
<td>None</td>
<td>Get practical experience with the tools, techniques and concepts needed to build a basic iOS app from scratch.</td>
<td>• Intro to App Development with Swift book and project files</td>
<td>• Intro to App Development with Swift: Teacher Guide</td>
<td>90 hours</td>
</tr>
<tr>
<td>App Development with Swift</td>
<td>☑️</td>
<td>Year 10 and up</td>
<td>App Development with Swift</td>
<td>None</td>
<td>Build a foundation in Swift, UIKit and networking through hands-on labs and guided projects. Students can build an app of their own design by the end of the course.</td>
<td>• App Development with Swift book and project files</td>
<td>• App Development with Swift: Teacher Guide</td>
<td>180 hours</td>
</tr>
</tbody>
</table>
Overview

The Intro to App Development with Swift and App Development with Swift curriculums were designed to teach students in year 10 and above with little or no programming experience how to be app developers, capable of bringing their own ideas to life.

The Intro to App Development with Swift course introduces students to the world of app development and the basics of Swift and Xcode. The course culminates in a final project where they can choose one of two basic iOS apps to build.

App Development with Swift takes students further, whether they’re new to coding or want to expand their skills. If they’re already familiar with Swift, Xcode and iOS development, they can move through lessons quickly or go straight to the labs, where they’ll build mini projects and test their code in playgrounds. By the end of the course, they’ll be able to build a fully functioning app of their own design.
Key Features

**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.

**Step-by-step instructions.** Detailed instructions with images and videos guide students through all the steps of building an app in Xcode.

**Sample projects.** Using the included project files, students can try out certain parts of code without having to build an entire app from the beginning.

**Study tools.** Students can check their understanding and apply what they’ve learned with review questions, key vocabulary, links to documentation and more.
Course Outlines

Intro to App Development with Swift

This introductory one-term course is designed to help students build a solid foundation in programming fundamentals using Swift as the language. Students get practical experience with the tools, techniques and concepts needed to build a basic iOS app.

App Design lessons take students through the process of designing an app, including brainstorming, planning, prototyping and evaluating an app of their own. Even though they might not yet have the skills to build the app, the work they put into the prototype will set them up for future development.

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 how to customise their Xcode environment and to debug.

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

Lesson 6: Functions. Students discover 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 examining the type system, from the standard library in Swift to custom types.

Lesson 10: Parameters and Results. Students expand their knowledge of functions by finding out 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.

The Teacher Guide includes additional extension activities, discussion questions, and activities for the app journal that students maintain throughout the term.
Course Outlines (continued)

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

**Lesson 13: QuestionBot.** Students 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 discover how to create and work with arrays by adding and removing objects, and how for loops work with each object in an array.

**Lesson 15: Defining Structures.** Students recognise 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 examine 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 practise 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 find out how to build user interfaces using Interface Builder, and tie user interface elements into code via Outlets and Actions. They’ll practise 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 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 explore 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 grid-like system. In the process, they build the SimpleCenter, ElementQuiz and AnimalSounds apps.

**Lesson 19: Enumerations and Switch.** Students discover that enumerations, or enums, are a way to define a named list of options, what they’re 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 complete one or both final project options from scratch. The first option is a Rock/Paper/Scissors game and the second is a Meme generator. Students review a variety of concepts covered in the course and build the user interface, model data and controller objects that make up the entire app.

**Lesson 21: App Design.** Students go through a design cycle that focuses on prototyping, much like the process that professional app developers go through.

**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.
Course Outlines (continued)

App Development with Swift

This two-term course features 45 lessons, each designed to teach a specific skill related to either Swift or app development. Each type of lesson takes a different approach:

- **Swift lessons.** These lessons focus on specific concepts. The labs for each are presented in playgrounds so that students can experiment with code and see the results immediately. Playground files are provided.

- **App development lessons.** Focusing on building specific features for iOS apps, these lessons typically take students step by step through a mini project. The labs help students apply what they learned to a new scenario.

At the end of each of the first five units, students complete guided projects that include a description of user-centred features, a project plan and instructions for building a fully functioning app. Through these projects, students can create features that interest them, all while performing the type of work they can expect in an app development workplace. In the last unit, they’ll examine how to design, prototype and architect an app of their own design.

The Teacher Guide includes tips for extending or adapting lessons, increasing collaboration and supporting students who need additional assistance. It also includes downloadable Keynote presentations for each lesson, solution code for the labs and a rubric for evaluating student work.

**Unit 1: Getting Started with App Development.** Students find out about the basics of data, operators and control flow in Swift, as well as documentation, debugging, Xcode, building and running an app, and Interface Builder. They then apply this knowledge to the guided project, Light, in which they create a simple torch app.

**Unit 2: Introduction to UIKit.** Students explore Swift strings, functions, structures, collections and loops. They also learn about UIKit — the system views and controls that make up a user interface — and how to display data using Auto Layout and stack views. They put this knowledge to practice in the guided project, Apple Pie, where they build a word-guessing game app.

**Unit 3: Navigation and Workflows.** Students discover how to build simple workflows and navigation hierarchies using navigation controllers, tab bar controllers and segues. They also examine two powerful tools in Swift, optionals and enumerations. They put this knowledge into practice with the guided project, Personality Quiz, a personalised survey that reveals a fun response to the user.

**Unit 4: Tables and Persistence.** Students find out about scroll views, table views and building complex input screens. They also explore how to save data, share data to other apps and work with images in the user’s photo library. They use their new skills in the guided project, List, a task-tracking app that allows the user to add, edit and delete items in a familiar table-based interface. Students can customise the app to keep track of any type of information, such as a collection, tasks or playlists.

**Unit 5: Working with the Web.** Students learn about animations, concurrency and working with the web. They apply what they’ve learned in the guided project, Restaurant, a customisable menu app that displays the available dishes from a restaurant and allows the user to submit an order. This app uses a web service that allows students to set up the menu with their own menu items and photos.

**Unit 6: Prototyping and Project Planning.** Students learn how to design, prototype and architect a project of their own design. Given enough time, they should be able to build this project independently.
Additional Information

Download the Get Started with Code resources
- Tynker
- codeSpark Academy
- Get Started with Code 1
- Get Started with Code 2

Download the Swift Playgrounds resources
- Learn to Code 1 & 2: iTunes U Course
- Learn to Code 1 & 2: Teacher Guide
- Learn to Code 3: Teacher Guide
- Swift Playgrounds app

Download the App Development with Swift guides
- Intro to App Development with Swift
- Intro to App Development with Swift: Teacher Guide
- App Development with Swift
- App Development with Swift: Teacher Guide

Additional resources
- Learn more about the Everyone Can Code programme.
- Connect with other educators in the Apple Developer Forums.

About Swift
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 incredibly powerful. It’s designed to scale from writing the simplest program, like “Hello, world!”, to some of the world’s most advanced software.
Learn more about Swift.

About Xcode
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.
Learn more about Xcode.

Professional learning
MobileMakersEdu is an Apple Professional Learning Provider that offers training and support services so teachers can bring app development with Swift curriculums into their classrooms. For more information, visit their website.
Here’s the preliminary alignment of Intro to App Development with Swift with the Interim 2016 Computer Science Teachers Association (CSTA) K–12 Computer Science Standards for Level 3A. Once the new standards are finalised, the guide will undergo CSTA's formal crosswalk review. The alignment covers the algorithms and programming concepts within the Interim 2016 CSTA K–12 Computer Science Standards.

<table>
<thead>
<tr>
<th>CSTA K–12 Computer Science Standards Level 3A for Grades 9–10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Playground Basics</td>
</tr>
<tr>
<td>Naming and Identifiers</td>
</tr>
<tr>
<td>Strings</td>
</tr>
<tr>
<td>Hello, world!</td>
</tr>
<tr>
<td>First App</td>
</tr>
<tr>
<td>Functions</td>
</tr>
<tr>
<td>Boogie Bot</td>
</tr>
<tr>
<td>Constants and Variables</td>
</tr>
<tr>
<td>Types</td>
</tr>
<tr>
<td>Parameters and Results</td>
</tr>
<tr>
<td>Making Decisions</td>
</tr>
<tr>
<td>Instances, Methods and Properties</td>
</tr>
<tr>
<td>Question Bot</td>
</tr>
<tr>
<td>Arrays and Loops</td>
</tr>
<tr>
<td>Defining Structures</td>
</tr>
<tr>
<td>Question Bot 2</td>
</tr>
<tr>
<td>Actions and Outlets</td>
</tr>
<tr>
<td>Adaptive User Interfaces</td>
</tr>
<tr>
<td>Enumerations and Switch</td>
</tr>
<tr>
<td>Final Project</td>
</tr>
<tr>
<td>App Design</td>
</tr>
</tbody>
</table>
Curriculum Alignment: App Development with Swift

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

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit 1: Getting Started with App Development</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Unit 2: Introduction to UIKit</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Unit 3: Navigation and Workflows</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Unit 4: Tables and Persistence</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Unit 5: Working with the Web</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Unit 6: Prototyping and Project Planning</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>