1. Welcome

This codelab is part of the [Kotlin Bootcamp for Programmers course](https://developer.android.com/courses/kotlin-bootcamp/overview). You'll get the most value out of this course if you work through the codelabs in sequence. Depending on your knowledge, you may be able to skim some sections. This course is geared towards programmers who know an object-oriented language, and want to learn [Kotlin](https://kotlinlang.org/).



Introduction

The Kotlin Bootcamp for Programmers course teaches you the Kotlin programming language. In this codelab, you learn about advantages of programming in the [Kotlin](https://kotlinlang.org/) programming language, and you install the IDE to get ready for the first lesson.

This course is geared towards programmers who know an object-oriented language and want to learn more about Kotlin. If you are familiar with C#, some of the features of Kotlin will be familiar. If you are familiar primarily with the Java programming language, you may be amazed at how much more concise and readable your code can be.

Since 2017, Google has officially supported Kotlin for developing Android apps. Read the announcement on the [Android Developers Blog](https://android-developers.googleblog.com/2017/05/android-announces-support-for-kotlin.html). This course content is a prerequisite of [Android Kotlin Fundamentals](https://codelabs.developers.google.com/android-kotlin-fundamentals/).

What you should already know

You should be familiar with:

* The basics of a modern, object-oriented, statically typed programming language such as Java or C#
* How to program with classes, methods, and exception handling in at least one language
* Using an IDE such as IntelliJ IDEA, Android Studio, Eclipse, or Visual Studio

What you'll learn

* How to work with the Kotlin REPL (Read-Eval-Print Loop) interactive shell
* The basic syntax of Kotlin code

What you'll do

* Install the Java Development Kit (JDK) and the IntelliJ IDEA, and become familiar with some Kotlin features.

## 2. Benefits of Kotlin

Kotlin is a new, modern programming language created by programmers, for programmers. It's focused on clarity, conciseness, and code safety.

## Robust code

The creators of Kotlin made various design decisions about the language to help programmers create robust code. For example, null-pointer exceptions in software have caused financial losses and spectacular computer crashes, and have resulted in countless hours of debugging. So Kotlin distinguishes between nullable and non-nullable data types, which helps catch more errors at compile time. Kotlin is strongly typed, and it does a lot to infer the types from your code. It has lambdas, coroutines, and properties, which allow you to write less code with fewer bugs.

## Mature platform

Kotlin has been around since 2011, and was released as open source in 2012. It reached version 1.0 in 2016, and since 2017 Kotlin has been an officially supported language for building Android apps. It's included with the IntelliJ IDEA as well as Android Studio 3.0 and later.

## Concise, readable code

Code written in Kotlin can be very concise, and the language is designed to eliminate boilerplate code such as getters and setters. For example, consider the following Java code:

public class Aquarium {

private int mTemperature;

public Aquarium() { }

public int getTemperature() {

return mTemperature;

}

public void setTemperature(int mTemperature) {

this.mTemperature = mTemperature;

}

@Override

public String toString() {

return "Aquarium{" +

"mTemperature=" + mTemperature +

'}';

}

}

It can be written concisely like this in Kotlin:

class Aquarium (var temperature: Int = 0)

Sometimes the goals of conciseness and readability are at odds with each other. Kotlin is designed to use "just enough boilerplate code" to ensure readability while keeping things concise.

## Interoperable with Java

Kotlin code compiles so that you can use Java and Kotlin code side-by-side, and continue to use your favorite Java libraries. You can add Kotlin code to an existing Java program, or if you want to migrate your program completely, IntelliJ IDEA and Android Studio both include tools to migrate existing Java code to Kotlin code.

3. Task: Install the Java Development Kit (JDK)

If you don't have the latest JDK already installed on your computer, follow the steps below. You need to have the JDK installed to run Kotlin programs.

To see which version of the JDK you have installed, if any, type javac -version in a terminal window.

javac -version

You can see what the latest version of the JDK is on the [Java SE Downloads](https://www.oracle.com/technetwork/java/javase/downloads) page. If you have the latest version, skip ahead to Install IntelliJ IDEA.

**Note:**We recommend that you install only the latest JDK and JRE.

**JRE or JDK?**

The JRE (Java Runtime Environment) is needed for running Java and Kotlin programs. The JDK (Java Development Kit), on the other hand, includes the JRE, plus the development tools that you need for writing and running Java programs. You need the JDK for writing Kotlin programs.

Step 1: Uninstall any older versions of the JDK/JRE

Before you install the latest and greatest, remove all older versions of the JDK:

* For Windows, select **Control Panel** > **Add/Remove Programs**.
* For Mac instructions, see [Uninstalling the JDK](https://docs.oracle.com/javase/8/docs/technotes/guides/install/mac_jdk.html#A1096903).

For additional information on uninstalling older versions of the JRE, see [How do I uninstall Java on my Mac?](https://www.java.com/en/download/help/mac_uninstall_java.xml) or [How do I uninstall Java on my Windows computer?](https://www.java.com/en/download/help/uninstall_java.xml)

Step 2: Download the JDK

You can download the JDK for free here:  
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

1. Click the **Download** button under the JDK for the latest Java SE version.
2. Select **Accept License Agreement**.
3. Choose the JDK for your operating system.

Step 3: Install the JDK (for Mac)

From either the **Downloads** window of the browser, or from the file browser, double-click the .dmg file to launch the install file.

1. A **Finder** window appears with an icon of an open box and the name of the .pkg file.
2. Double-click the package icon to launch the installation app, and follow the prompts as they appear.
3. You might need to enter the administrator password to continue.
4. After the installation is complete, feel free to delete the .dmg file to save space.

Step 3: Install the JDK and JRE (for Windows)

1. Run the downloaded installer (for example, jdk-13.0.2\_windows-x64\_bin.exe), which installs both the JDK and the JRE. By default, the JDK is installed in the C:\Program Files\Java\jdk-13 directory.
2. Accept the defaults, and follow the on-screen instructions to install the JDK.

Step 4: Add the JDK and JRE installation directories to PATH (Windows only)

Windows searches the current directory and the directories listed in the PATH environment variable (system variable) for executable programs.

1. Open **Control Panel** > **System** > **Advanced system settings** > **Environment Variables**.
2. Under **System variables**, click **New** and add a variable named JAVA\_HOME with the JRE's directory for a value. For example, C:\Program Files\Java\jre1.8.0\_x, where *x* is the version number.
3. Under **System variables**, scroll down to select **Path**, then click **Edit**.
4. Add the JRE's bin directory to the start of the Path, followed by a semicolon: %JAVA\_HOME%\bin;
5. Append the JDK's bin directory to end of the Path, preceded by a semicolon. For example, ;C:\Program Files\Java\jdk-12\bin.

Step 5: Verify the JDK installation

1. To verify that the JDK was installed correctly, type the following commands in a terminal window:

java -version

javac -version

**Windows users:** If you receive an error from either command, confirm you've added the correct paths for the JRE (java) and the JDK (javac).

4. Task: Install IntelliJ IDEA

Step 1: Download and install IntelliJ IDEA

[Download IntelliJ IDEA](https://www.jetbrains.com/idea/download/index.html) for your operating system.

**Windows:**

1. Run the ideaIC.exe file that you downloaded.
2. Follow the instructions in the installation wizard.

**Mac:**

1. To mount the macOS disk image, double-click the ideaIC.dmg file that you downloaded.
2. Copy **IntelliJ IDEA** to the **Applications** folder.

**Linux:**

1. See Install-Linux-tar.txt in the downloaded .tar.gz file.

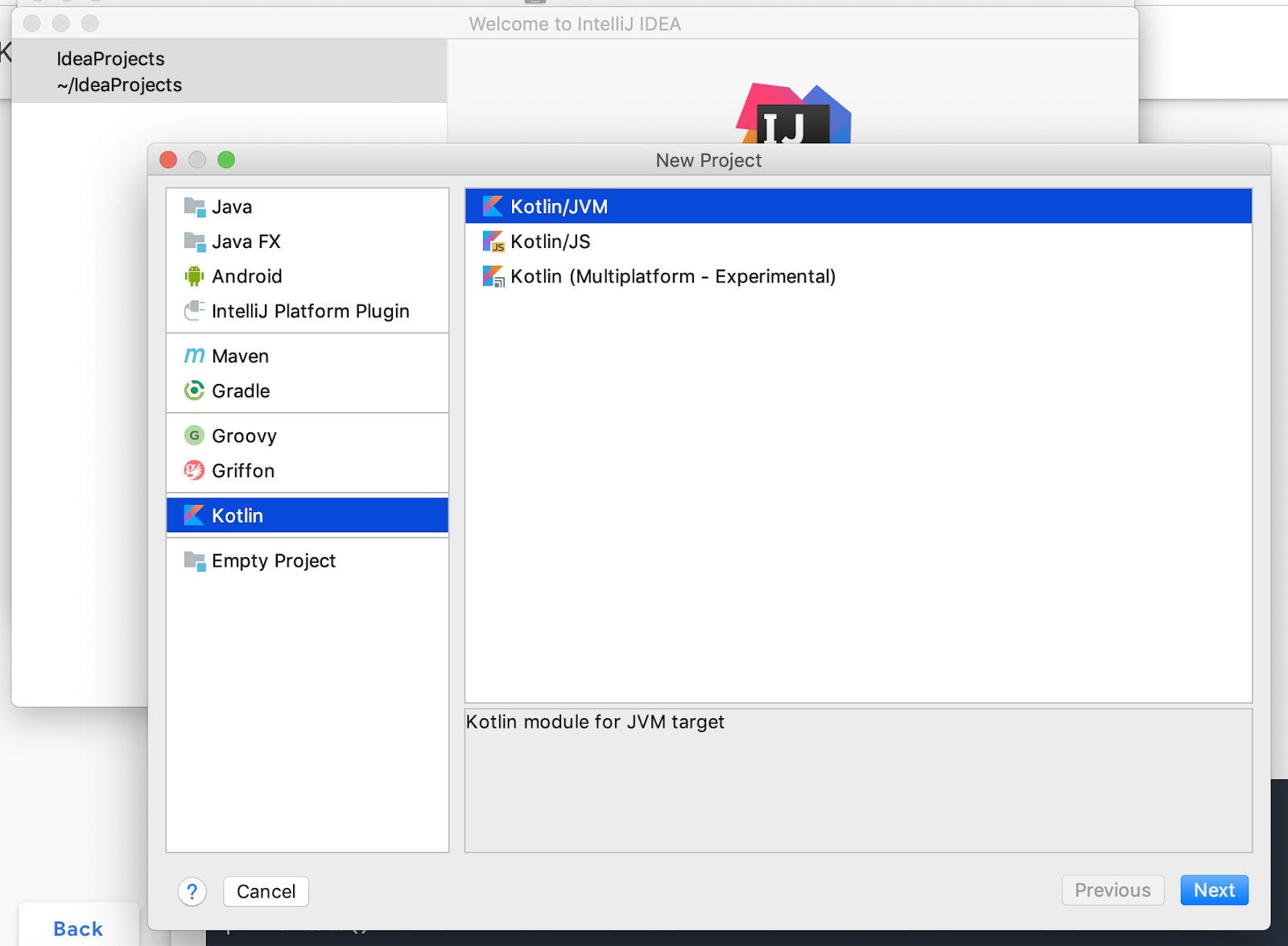
For more information on how to install and set up IntelliJ IDEA, check out [Install IntelliJ IDEA](https://www.jetbrains.com/help/idea/install-and-set-up-product.html).

Step 2: Verify your IntelliJ IDEA installation

1. Start IntelliJ IDEA.
2. Install any updates and additional content you are prompted for.
3. Select**Configure** > **Check for Updates** until there are no more updates available.

5. Task: Create Hello Kotlin

Create a Kotlin project so IntelliJ IDEA knows you're working in Kotlin.

1. In the **Welcome to IntelliJ IDEA** window, click **Create New Project.**
2. In the **New Project** pane, select **Kotlin** in the left-hand navigation.
3. Select **Kotlin/JVM** in the right panel and click **Next**.  
   
4. Name your project Hello Kotlin.
5. Click **Finish.**

Now you can access the [REPL](https://en.wikipedia.org/wiki/Read%E2%80%93eval%E2%80%93print_loop) (Read-Eval-Print Loop), Kotlin's interactive shell. Commands that you type into the REPL are interpreted as soon as you press Control+Enter (Command+Enter on a Mac).

1. Select **Tools** > **Kotlin** > **Kotlin REPL** to open the REPL.

The first time you run IntelliJ IDEA after installing, it may take a few moments before the **Kotlin** menu appears under **Tools**.

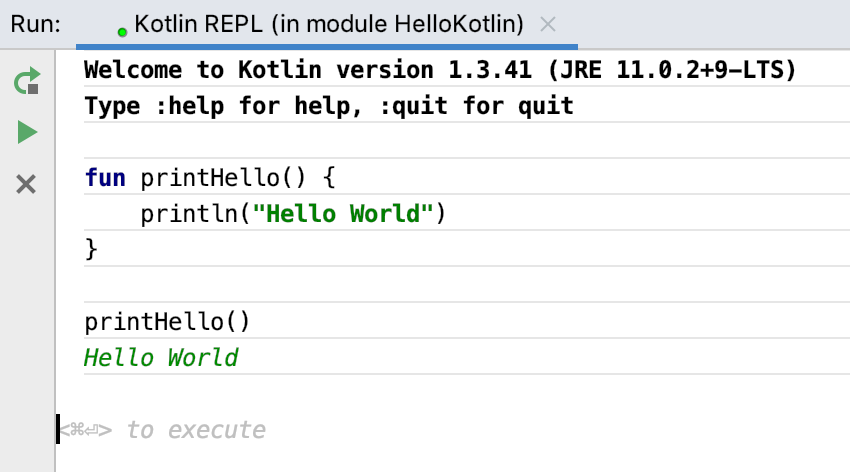
1. Type or paste the code below into the REPL.

fun printHello() {

println("Hello World")

}

printHello()

1. Press Control+Enter (Command+Enter on a Mac). You should see Hello World, as shown below.  
   
2. Take a quick look at this Kotlin code. The fun keyword designates a function, followed by the name. As with other programming languages, the parentheses are for function arguments, if any, and the curly braces frame the code for the function. There is no return type because the function doesn't return anything. Also note that there are no semicolons at the ends of lines.

**Note:** If you're used to putting semicolons at the end of lines, that's OK—Kotlin doesn't mind.

Congratulations! You've written your first Kotlin program.

6. Summary

* Kotlin is similar to other object-oriented programming languages.
* Install the latest JDK for your operating system to use Kotlin.
* Install the IntelliJ IDEA to work with Kotlin.
* In IntelliJ IDEA, start the Kotlin REPL (**Tools > Kotlin > Kotlin REPL**) to practice in an interactive shell.
* Enter code followed by Control+Enter (Command+Enter on a Mac) to run it.
* Here is "Hello World" in Kotlin:

fun printHello() {

println ("Hello World")

}

printHello()

7. Learn more

Kotlin documentation

If you want more information on any topic in this course, or if you get stuck, [https://kotlinlang.org](https://kotlinlang.org/) is your best starting point.

* [Kotlin coding conventions](https://kotlinlang.org/docs/reference/coding-conventions.html)
* [Kotlin idioms](https://kotlinlang.org/docs/reference/idioms.html)

Kotlin tutorials

The [https://try.kotlinlang.org](https://try.kotlinlang.org/) website includes rich tutorials called Kotlin Koans, a [web-based interpreter](https://try.kotlinlang.org/#/Examples/Hello,%20world!/Simplest%20version/Simplest%20version.kt), and a complete set of reference documentation with examples.

Udacity course

To view the Udacity course on this topic, see [Kotlin Bootcamp for Programmers](https://classroom.udacity.com/courses/ud9011).

IntelliJ IDEA

[Documentation for the IntelliJ IDEA](https://www.jetbrains.com/help/idea/) can be found on the JetBrains website.

## 8. Homework

This section lists possible homework assignments for students who are working through this codelab as part of a course led by an instructor. It's up to the instructor to do the following:

* Assign homework if required.
* Communicate to students how to submit homework assignments.
* Grade the homework assignments.

Instructors can use these suggestions as little or as much as they want, and should feel free to assign any other homework they feel is appropriate.

If you're working through this codelab on your own, feel free to use these homework assignments to test your knowledge.

## ****Answer these questions****

### **Question 1**

Which of the following is NOT a benefit of using the Kotlin language?

▢ Kotlin distinguishes between nullable and non-nullable data types.

▢ Kotlin is a supported language for building Android apps.

▢ Kotlin is designed so you can write less code with fewer bugs.

▢ Your code compiles faster in Kotlin.

### **Question 2**

How do you start the Kotlin REPL?

▢ Type repl on the command line.

▢ Create a Kotlin project in IntelliJ IDEA, then select **Run > Kotlin REPL**.

▢ Open IntelliJ IDEA, then select **File > Kotlin REPL**.

▢ Create a Kotlin project in IntelliJ IDEA, then select **Tools > Kotlin > Kotlin REPL**.

### **Question 3**

Which of the following is NOT true about Kotlin and Java code?

▢ Kotlin code and Java code can run side-by-side.

▢ You can add Kotlin code to an existing Java program.

▢ You can migrate existing Java code to Kotlin.

▢ Kotlin code will run faster than Java code.