Lesson 1.1 - Get started

About this codelab

subjectLast updated Sep 21, 2023

account_circleWritten by Google Developers Training team

1. Benefits of Kotlin

Kotlin is a new, modern programming language that helps developers be more productive. It focuses 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:

```
data 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.

---

2. 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 page. If you have the latest version, skip ahead to Install IntelliJ IDEA.

Note: We recommend that you install only the latest Long Term Support (LTS) version of the JDK and JRE. 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.

For additional information on uninstalling older versions of the JRE, see How do I uninstall Java on my Mac? or How do I uninstall Java on my Windows computer?

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 on **JDK Download**.
2. Under **Downloads**, choose the link for the JDK for your operating system.
3. Accept the license agreement.
4. Click on the **Download** button.

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-14.0.1_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-14.0.1` directory, but it depends on the latest version.
2. Accept the defaults, and follow the on-screen instructions to install the JDK.

Step 4: Add the JDK installation directory to PATH (Windows only)

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

1. In **Settings** for Windows, search for `edit environment` in **Find a setting**.
2. Select **Edit environment variables for your account** in the list of matches.
3. In the **Environment Variables** dialog in the **User variables** section, select **Path** and click the **Edit...** button.
4. Add the path to the JDK's bin directory, for example, `C:\Program Files\Java\jdk-14.0.1\bin`, after any existing items.

Note: This was tested for JDK 14 installed on Windows 10. Other versions of the JDK may use different directories, and the steps may be different for other versions of Windows.

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
```

Note: If you receive an error or an unexpected version from either command, confirm you have the correct path for the latest JRE.

---

3. Install IntelliJ IDEA

Step 1: Download and install IntelliJ IDEA

Download IntelliJ IDEA for your operating system. The Community Edition of IntelliJ IDEA is free and open-source.

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.

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.

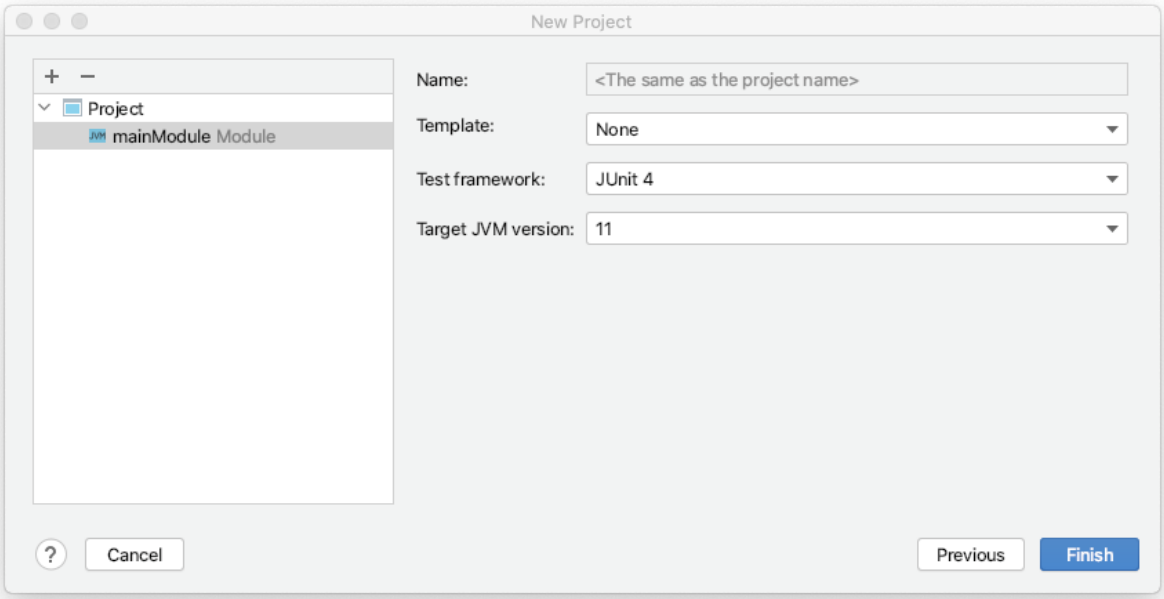
---

4. 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 **+ New Project.**
2. In the **New Project** pane, select **Kotlin** in the left-hand navigation.
3. Select **Kotlin/JVM** (**JVM | IDEA** in newer versions) in the right panel and click **Next**.
4. Name your project `HelloKotlin`.
5. In the "Project JDK" section, select the latest version of the JDK that you have installed.

6. Click **Next.**
7. In the next dialog box, select the Target JVM version. Choose the latest JDK version that you have installed.



8. Click **Finish.**

Now you can access the REPL (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.

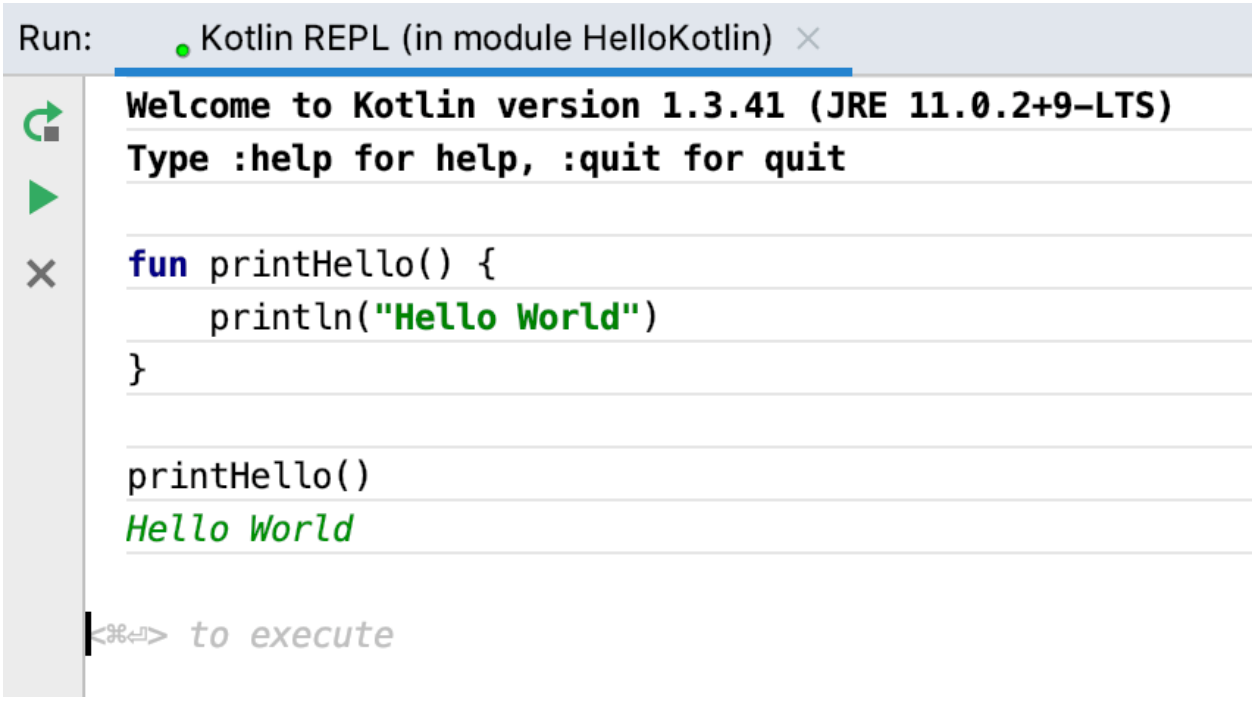
Note: The first time you run IntelliJ IDEA after installing, it may take a few moments before the **Kotlin** menu appears under **Tools**. You may even see the "Tip of the Day" to learn something while you wait!

2. You might see a small dialog box "Choose context module..." Choose the module name without any file extension.
3. Type or paste the code below into the REPL shell window.

```
fun printHello() {
   println("Hello World")
}

printHello()
```

4. Press `Control+Enter` (`Command+Enter` on a Mac). You should see `Hello World`, as shown below.



5. Take a quick look at this Kotlin code. The `fun` keyword designates a function, followed by the name, in this case `printHello`. As with other programming languages, the parentheses are for function parameters, if there are any, and the curly braces frame the code for the function (formally, they provide a scope). There is no return type because this function doesn't return anything. Also note Kotlin does not require 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.

---

5. 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()
```