

Introduction to Android and Kotlin

PROGRAMMING
LANGUAGE

KOTLIN

What is Kotlin ?

Kotlin = Java + Extra updated new features.

- **Kotlin** is a **general-purpose**, **statically typed**, and **open-source** programming language.
- It runs on JVM and can be used anywhere Java is used today.
- It can be used to develop Android apps, server-side apps and much more.
- It has built world-class IDEs like IntelliJ IDEA, PhpStorm, Appcode, etc.

History of Kotlin

- It was first introduced by JetBrains in 2011 and a new language for the JVM.
- Kotlin is object-oriented language.
- Kotlin mainly targets the Java Virtual Machine (JVM), but also compiles to JavaScript.
- Kotlin is influenced by other popular programming languages such as Java, C#, JavaScript, Scala and Groovy.
- Kotlin is sponsored by Google, announced as one of the official languages for **Android Development** in 2017.

Features of Kotlin

- **Statically typed** – Statically typed is a programming language characteristic that means the type of every variable and expression is known at compile time.
- **Concise:** Kotlin reduces writing the extra codes.
- **Null safety:** Kotlin is null safety language. Kotlin aimed to eliminate the **NullPointerException (null reference)** from the **code.Interoperable**.
- **Interoperable:** We can easily access use java code from kotlin and kotlin code from java.
- **Smart cast:** It explicitly typecasts the immutable values and inserts the value in its safe cast automatically.
- **Compilation Time:** It has better performance and fast compilation time.
- **Tool-friendly:** Kotlin programs are build using the command line as well as any of Java IDE.
- **Functional and Object Oriented Capabilities** – Kotlin has rich set of many useful methods which includes higher-order functions, lambda expressions, operator overloading, lazy evaluation, operator overloading and much more.
- **Easy to Learn** – Basic is almost similar to java.If anybody worked in java then easily understand in no time.

Kotlin Drawbacks

- 1. Namespace declaration** – Kotlin allows developers to declare the functions at the top level. However, whenever the same function is declared in many places of your application, then it is hard to understand which function is being called.
- 2. No Static Declaration** – Kotlin does not have usual static handling modifier like Java, which can cause some problem to the conventional Java developer.
- 3 Fluctuation in Compilation** -In many cases such as performing incremental builds, Kotlin is faster than Java, there is no doubt about it. However, Java remains a clear winner when it comes to creating clean builds for Android apps.
- 4 Less Talent for Hire** - still less number of Kotlin developers available in the market compared with Java developers.
- 5 Limited Learning Resources** - Though the number of Android app developers switching to Kotlin is increasing almost every day, there is still limited number of resources available in the market to learn and master Kotlin.
- 6 Kotlin is Still Not Java** - While Kotlin and Java have a lot of similarities, making the switch from Java to Kotlin will take some time for developers to get familiar with how everything works in the Kotlin programming language.

Final Thoughts...

Having official support from a tech giant Google is definitely a clear sign that the future of Kotlin is bright.

Applications of Kotlin language

- You can use Kotlin to build Android Application.
- Kotlin can also compile to JavaScript, and making it available for the frontend.
- It is also designed to work well for web development and server-side development.
- Cross-platform Mobile applications
- Android Application Development
- Web Application Development
- Server Side Applications
- Desktop Application Development
- Data science based applications

Kotlin Jobs Opportunity

- Kotlin is very high in demand and all major companies are moving towards Kotlin to develop their web and mobile applications. Average annual salary for a Kotlin developer is around **₹ 3.6 Lakhs to ₹ 35.0 Lakhs** . Following are the great companies who are using Kotlin:
 - Google
 - Amazon
 - Netflix
 - Pinterest
 - Uber
 - Trello
 - Coursera
 - Basecamp
 - Corda
 - JetBrains
 - Many more...

Prerequisites to Learn Kotlin

- Before proceeding with this tutorial you should have a basic understanding of Java programming language.
- Or any programming environment and knowledge of basic concepts such as variables, commands, syntax, etc.
- We strongly recommend that you gain some basic knowledge of Java programming language before proceeding with Kotlin programming.

Installing Kotlin

- Like Java, Kotlin also runs on JVM therefore to install Kotlin on Windows directly and work with it using the command line You need to make sure you have JDK installed in your system.

Verifying the Java installation

- To verify Java installation –
`C:\> javac version`

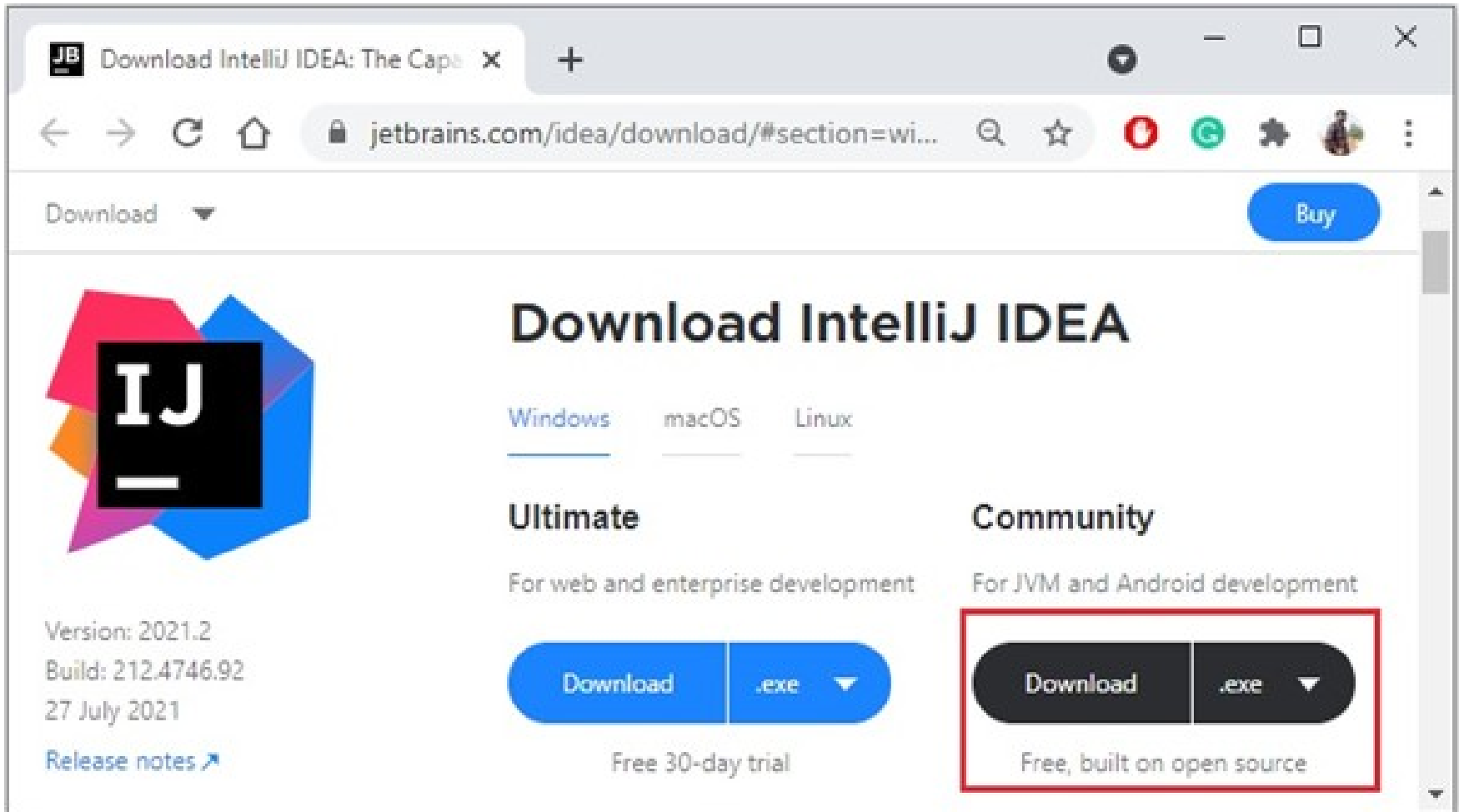
If Java is not intall on your system than You can install JDK
Java SE 8 or higher version

Install IDE for Kotlin

- There are various Java IDE available which supports Kotlin project development. We can choose these IDE according to our compatibility. The download links of these IDE's are given below.

IDE Name	Download links
IntelliJ IDEA	https://www.jetbrains.com/idea/download/
Android Studio	https://developer.android.com/studio/pr
Eclipse	view/index.html https://www.eclipse.org/downloads/

Setting Kotlin environment using IntelliJ IDEA




The screenshot shows the JetBrains website's download page for IntelliJ IDEA. The browser's address bar displays the URL `jetbrains.com/idea/download/#section=wi...`. The page features the IntelliJ logo on the left, which includes the version information: Version: 2021.2, Build: 212.4746.92, and the release date: 27 July 2021. Below the logo is a link to the release notes. The main heading is "Download IntelliJ IDEA". Underneath, there are tabs for "Windows", "macOS", and "Linux", with "Windows" being the active tab. The page is divided into two sections: "Ultimate" and "Community". The "Ultimate" section is described as "For web and enterprise development" and offers a "Free 30-day trial". The "Community" section is described as "For JVM and Android development" and is "Free, built on open source". Both sections have a "Download" button followed by a ".exe" button with a dropdown arrow. The "Community" download buttons are highlighted with a red rectangular box.

JB Download IntelliJ IDEA: The Cap... x +

jetbrains.com/idea/download/#section=wi...

Download ▾ Buy



Version: 2021.2
Build: 212.4746.92
27 July 2021
[Release notes ↗](#)

Download IntelliJ IDEA

[Windows](#) [macOS](#) [Linux](#)

Ultimate

For web and enterprise development

Download .exe ▾

Free 30-day trial

Community

For JVM and Android development

Download .exe ▾

Free, built on open source



IntelliJ IDEA Community Edition Setup



IJ

Welcome to IntelliJ IDEA Community Edition Setup

Setup will guide you through the installation of IntelliJ IDEA Community Edition.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Next >

Cancel



JETBRAINS COMMUNITY EDITION TERMS

IMPORTANT! READ CAREFULLY:

THESE TERMS APPLY TO THE JETBRAINS INTEGRATED DEVELOPMENT ENVIRONMENT TOOLS CALLED 'INTELLIJ IDEA COMMUNITY EDITION' AND 'PYCHARM COMMUNITY EDITION' (SUCH TOOLS, "COMMUNITY EDITION" PRODUCTS) WHICH CONSIST OF 1) OPEN SOURCE SOFTWARE SUBJECT TO THE APACHE 2.0 LICENSE (AVAILABLE HERE: <https://www.apache.org/licenses/LICENSE-2.0>), AND 2) JETBRAINS PROPRIETARY SOFTWARE PLUGINS PROVIDED IN FREE-OF-CHARGE VERSIONS WHICH ARE SUBJECT TO TERMS DETAILED



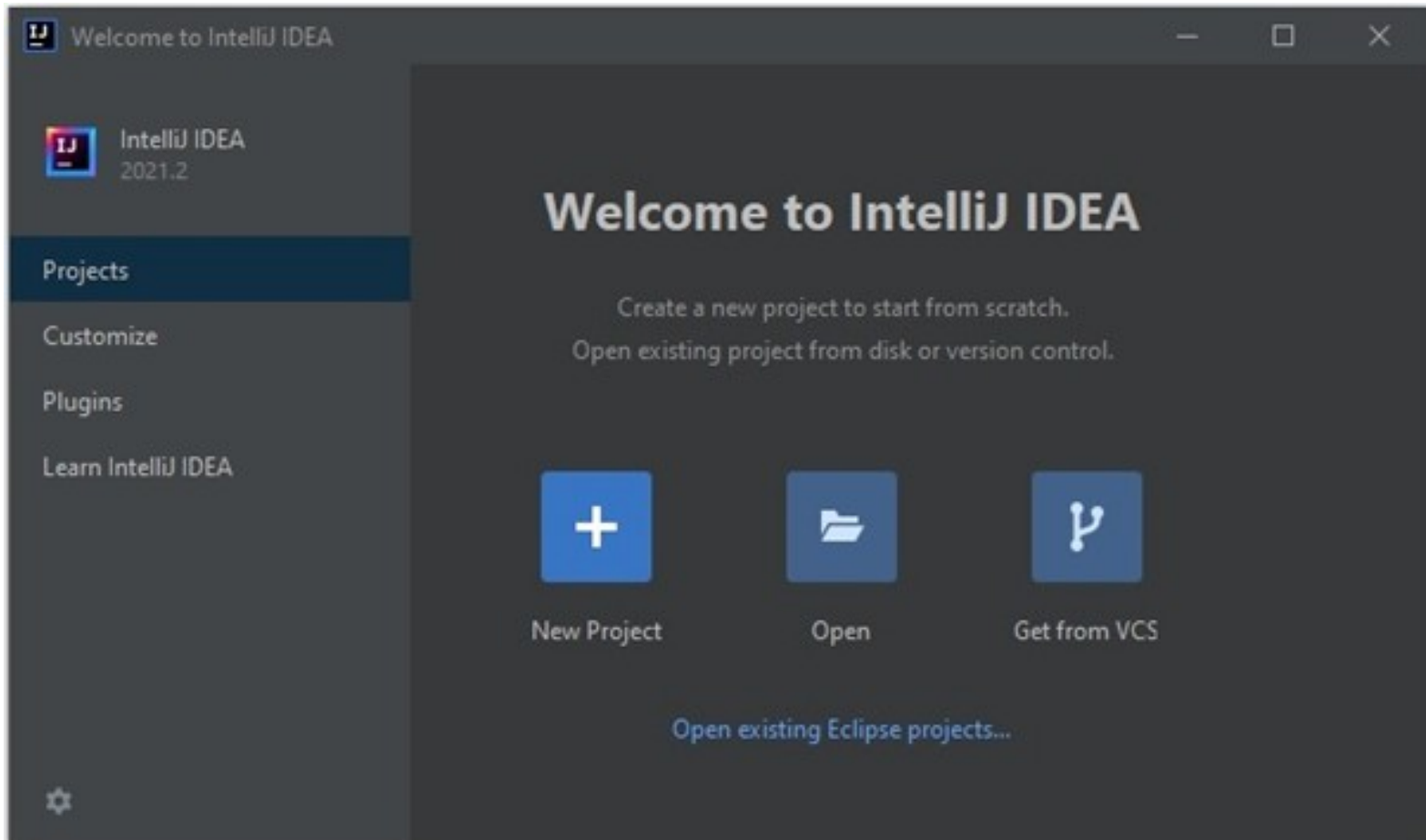
I confirm that I have read and accept the terms of this User Agreement

Exit

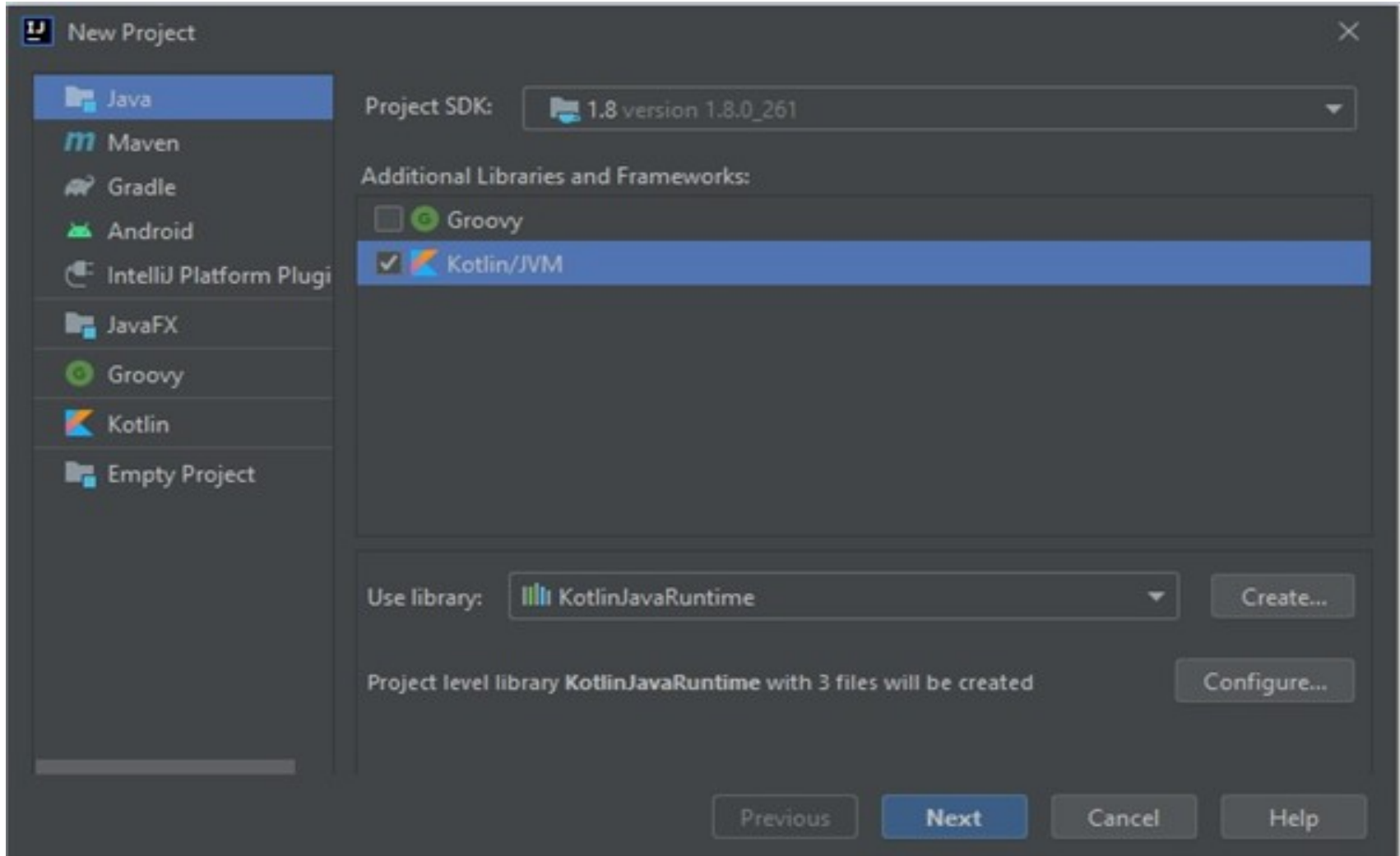
Continue

Creating first application

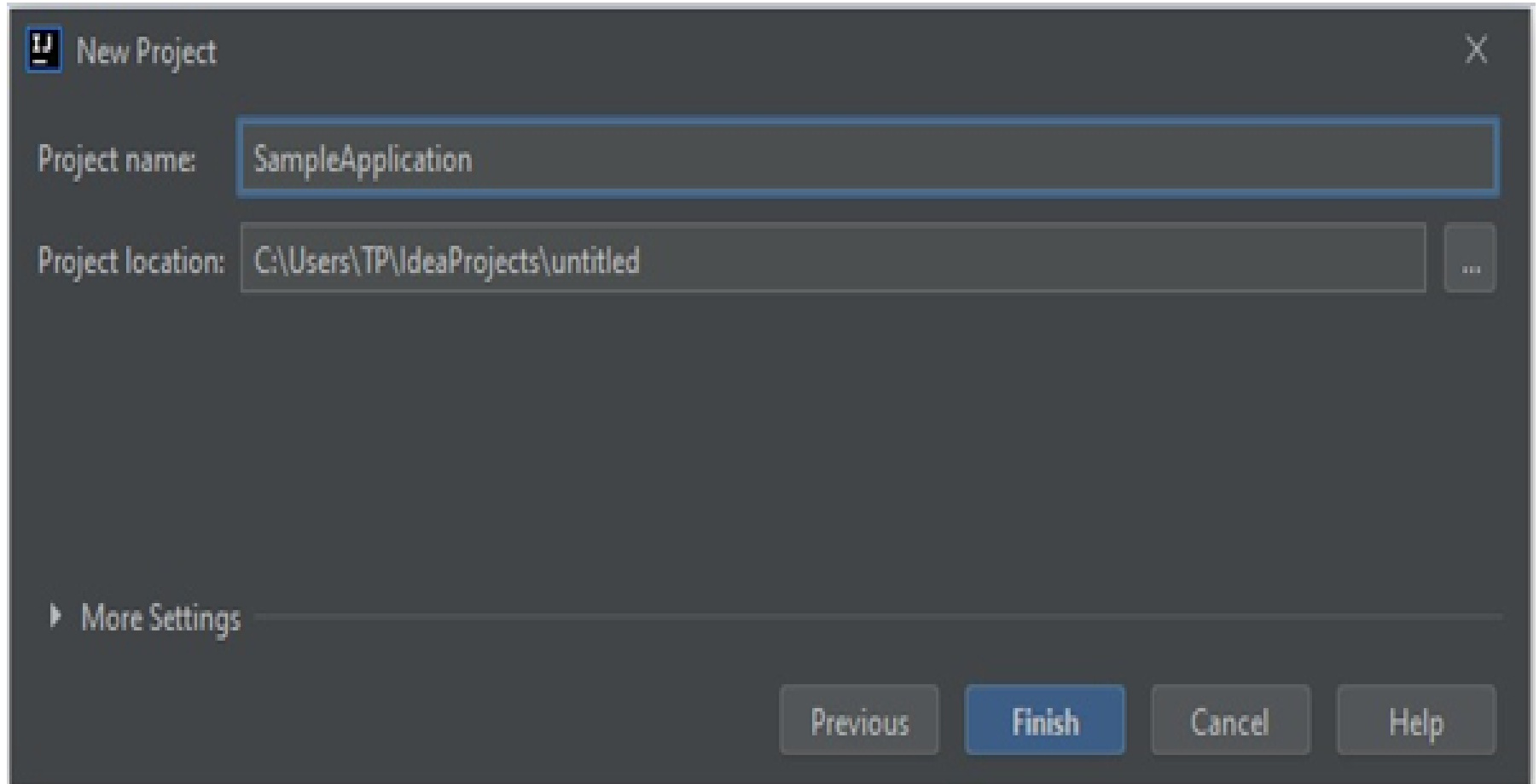
To create first application, click on **NewProject**.



Select **Kotlin/JVM** and click **Next**.



Name the project and select the desired location.



The image shows the 'New Project' dialog box in IntelliJ IDEA. The dialog has a dark theme. At the top, there is a title bar with the IntelliJ logo and the text 'New Project', and a close button (X) on the right. Below the title bar, there are two input fields. The first is labeled 'Project name:' and contains the text 'SampleApplication'. The second is labeled 'Project location:' and contains the text 'C:\Users\TP\IdeaProjects\untitled'. To the right of the 'Project location:' field is a button with three dots (...). At the bottom left, there is a section titled 'More Settings' with a right-pointing arrow. At the bottom right, there are four buttons: 'Previous', 'Finish' (which is highlighted in blue), 'Cancel', and 'Help'.

New Project

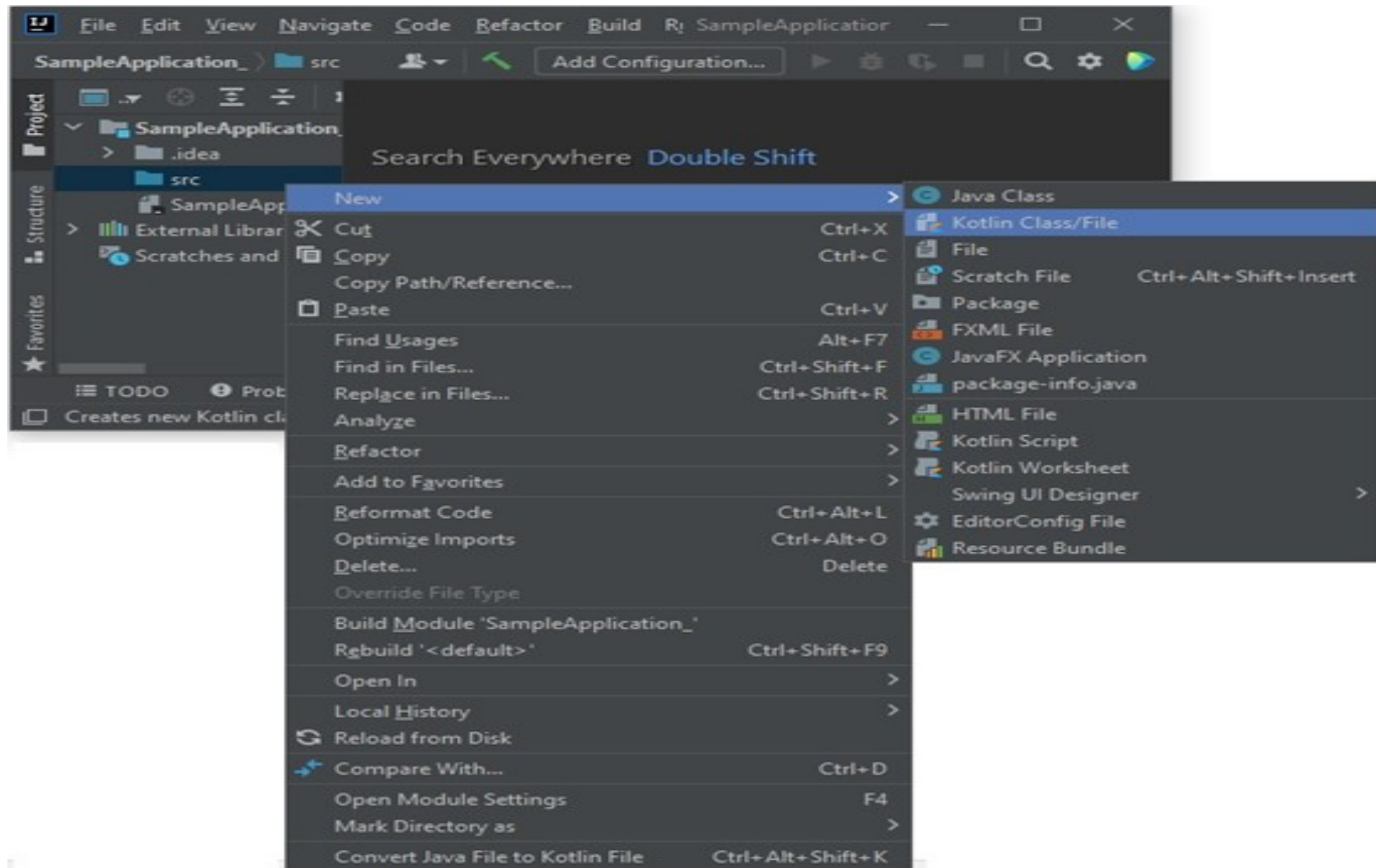
Project name: SampleApplication

Project location: C:\Users\TP\IdeaProjects\untitled

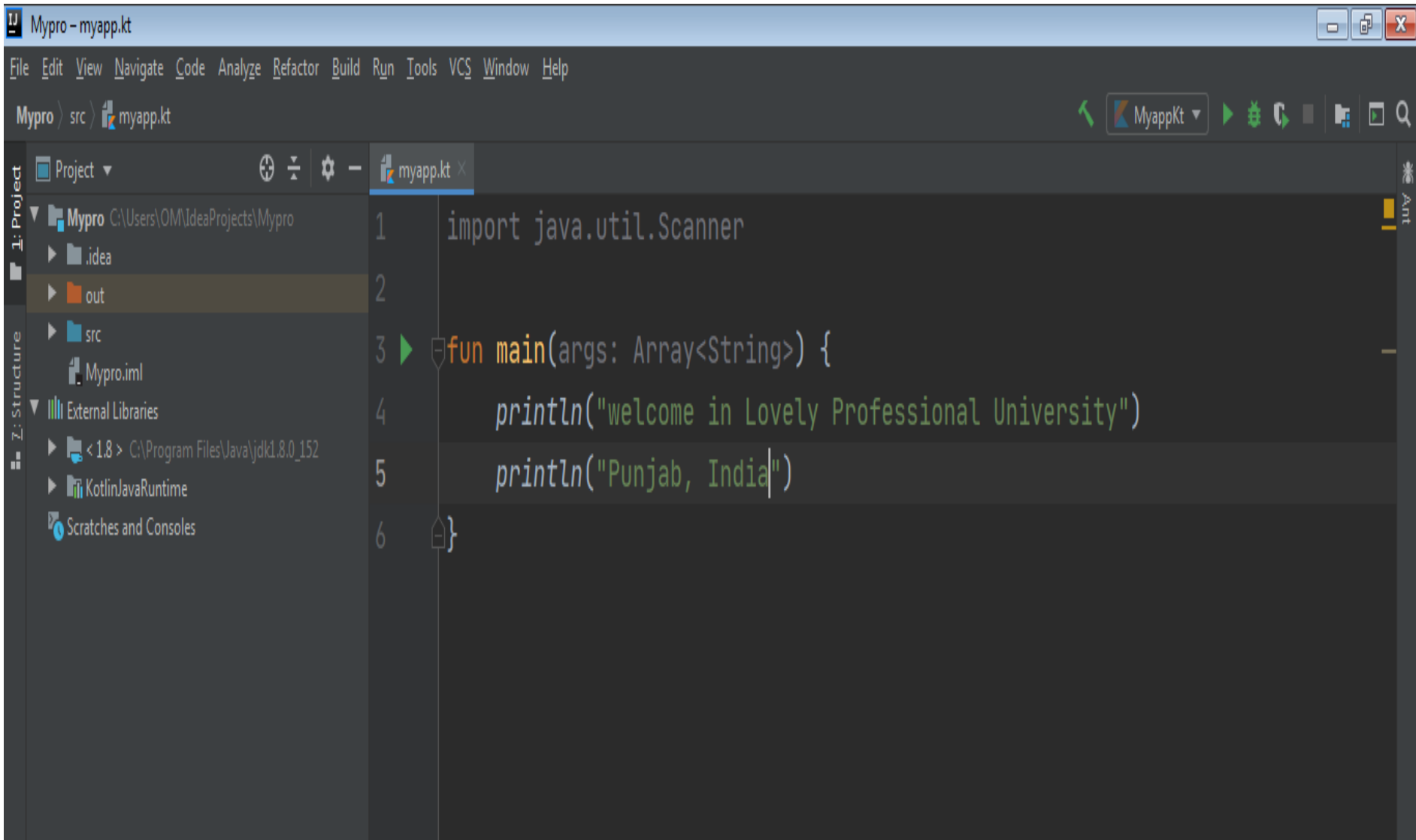
More Settings

Previous Finish Cancel Help

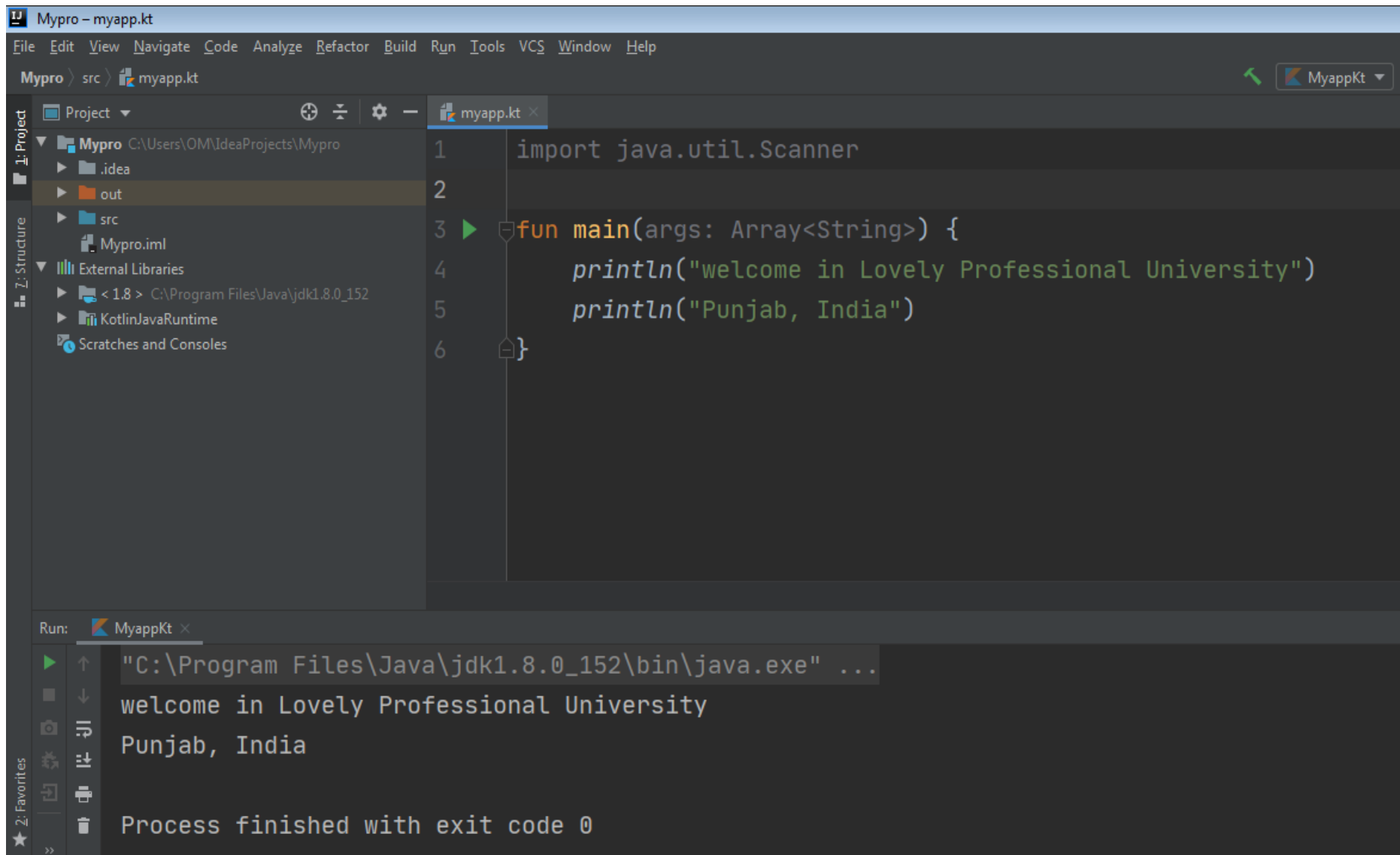
Now, create a new Kotlin file under the source(src) folder and let's name it



You can create a sample function as shown below. You can run this by pressing **Ctrl + Shift + F10**.



Output



The screenshot displays an IDE window titled "Mypro - myapp.kt". The menu bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, and Help. The breadcrumb navigation shows "Mypro > src > myapp.kt". The left sidebar contains a Project view and a Structure view. The main editor shows the following Kotlin code:

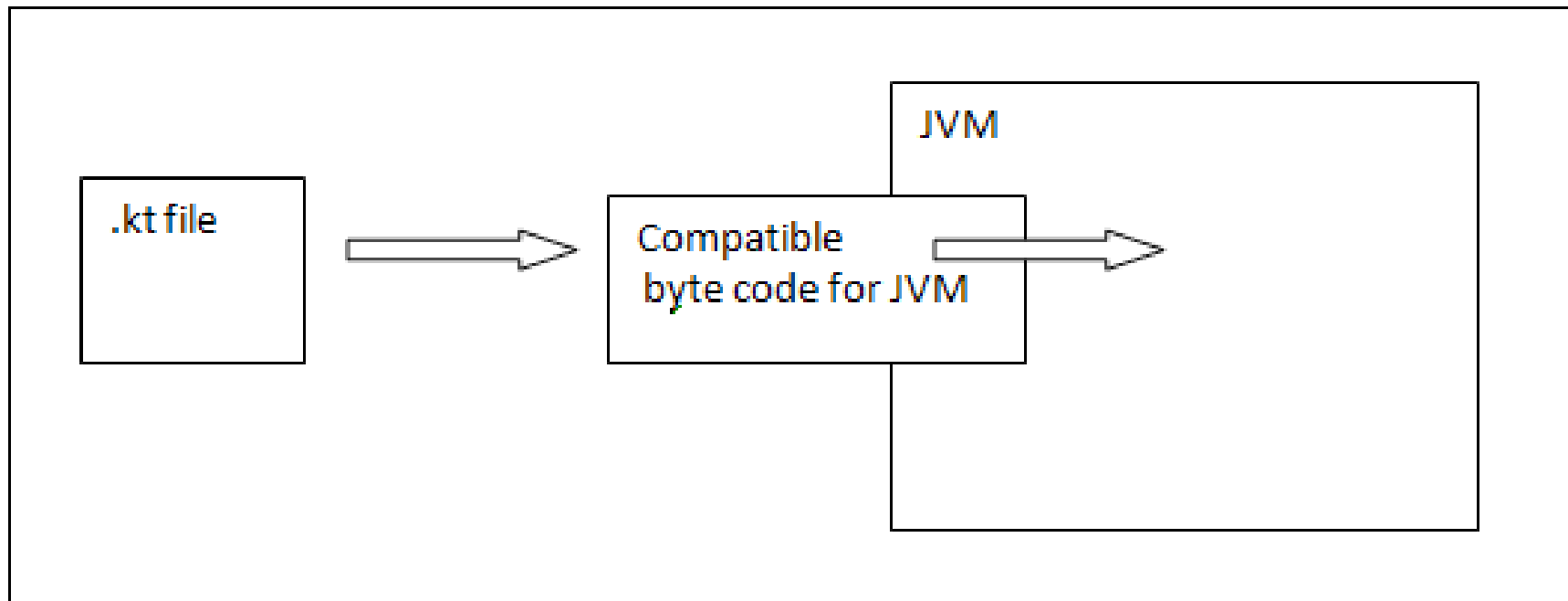
```
1 import java.util.Scanner
2
3 fun main(args: Array<String>) {
4     println("welcome in Lovely Professional University")
5     println("Punjab, India")
6 }
```

Below the editor is the Run tab, which shows the execution command and output:

```
Run: "C:\Program Files\Java\jdk1.8.0_152\bin\java.exe" ...
welcome in Lovely Professional University
Punjab, India
Process finished with exit code 0
```

Kotlin - Architecture

- Kotlin is a programming language and has its own architecture to allocate memory and produce a quality output to the end user.
- Following are the different scenarios where Kotlin compiler will work differently.
- Compile Kotlin into bytecode which can run on JVM. This bytecode is exactly equal to the byte code generated by the Java **.class** file.
- Whenever Kotlin targets JavaScript, the Kotlin compiler converts the **.kt** file into ES5.1(**JavaScript ES5**) and generates a compatible code for JavaScript.
- Kotlin compiler is capable of creating platform basis compatible codes via LLVM(Low Level VM).
- Kotlin Multiplatform Mobile (KMM) is used to create multiplatform mobile applications with code shared between Android and iOS.



Kotlin - Basic Syntax

- Program Entry Point - An entry point of a Kotlin application is the **main()** function. A function can be defined as a block of code designed to perform a particular task.

fun main()

- Entry Point with Parameters- Another form of **main()** function accepts a variable number of String arguments as follows:

fun main(args: Array<String>)

print() vs println()

- The **print()** is a function in Kotlin which prints its argument to the standard output, similar way the **println()** is another function which prints its argument on the standard output but it also adds a line break in the output.

Kotlin - Comments

- Kotlin supports single-line (or end-of-line) and multi-line (block) comments.
 1. `// This is single-line comment`
 2. `/* This is a
multi-line
comment */`

Variables

- They are the names you give to computer memory locations which are used to store values in a computer program and later you use those names to retrieve the stored values and use them in your program.
- Kotlin variables are created using either **var** or **val** keywords and then an equal sign = is used to assign a value to those created variables.

var name = "Shyam"

var age = 19

var height = 5.2

- **Kotlin Mutable Variables** means that the variable can be reassigned to a different value after initial assignment.

Kotlin Read-only Variables

- A read-only variable can be declared using **val** (instead of **var**) and once a value is assigned, it can not be re-assigned.

```
val subject = "Android"
```

```
subject = "Kotlin"
```

```
//Val cannot be reassigned
```

Kotlin Variable Naming Rules

- Kotlin variable names can contain letters, digits, underscores, and dollar signs.
- Kotlin variable names should start with a letter, \$ or underscores
- Kotlin variables are case sensitive which means **LPU** and **lpu** are two different variables.
- Kotlin variable can not have any white space or other control characters.
- Kotlin variable can not have names like var, val, String, Int because they are reserved keywords in Kotlin.

Data Types

- Kotlin treats everything as an object which means that we can call member functions and properties on any variable.
- Kotlin is a statically typed language, which means that the data type of every expression should be known at compile time.
 - a) Number
 - b) Character
 - c) String
 - d) Boolean
 - e) Array

a) Kotlin Number Data Types

- Numeric values and they are divided into two groups:

(a) **Integer types** store whole numbers, +ve or -ve **val a: Int = 10000**

(b) **Floating point** numbers with a fractional part, containing one or more decimals. **val f: Float = 100.00f**

Data Type	Size (bits)	Data Range
Byte	8 bit	-128 to 127
Short	16 bit	-32768 to 32767
Int	32 bit	-2,147,483,648 to 2,147,483,647
Long	64 bit	-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807
Float	32 bit	1.40129846432481707e-45 to 3.40282346638528860e+38
Double	64 bit	4.94065645841246544e-324 to 1.79769313486231570e+308

b) Kotlin Character Data Type

- Kotlin character data type is used to store a single character and they are represented by the type **Char** keyword. A Char value must be surrounded by single quotes, like 'A' or '1'.

val letter: Char = 'A'

c) Kotlin String Data Type

- The String data type is used to store a sequence of characters.
- String values must be surrounded by double quotes (" ") or triple quote ("\"\"\" \"\"\"").
- We have two kinds of string available in Kotlin
 1. **Escaped String** is declared within double quote (" ") and may contain escape characters like '\n', '\t', '\b' etc.
 2. **Raw string** is declared within triple quote ("\"\"\" \"\"\"") and may contain multiple lines of text without any escape characters.

- `val x : String = "I am escaped String!\n"`
- `var y :String = """"This is going to be a
multi-line string and will
not have any escape sequence""";`

d) Kotlin Boolean Data Type

Boolean is very simple like other programming languages. We have only two values for Boolean data type - either **true** or **false**

```
val A: Boolean = true
```

```
// defining a variable with true value
```

e) Kotlin Array Data Type

- Kotlin arrays are a collection of homogeneous data. Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

```
val numbers: IntArray = intArrayOf(1, 2, 3, 4, 5)
```

Kotlin Data Type Conversion

- Type conversion is a process in which the value of one data type is converted into another type. Kotlin does not support direct conversion of one numeric data type to another. It has predefined **TYPE CHECKING** option.
- To convert a numeric data type to another type, Kotlin provides a set of functions:

- toByte()
- toShort()
- toInt()
- toLong()
- toFloat()
- toDouble()
- toChar()

```
val x: Int = 100
```

```
val y: Long = x.toLong()
```

Operator

- An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations on operands (values or variable). Kotlin is rich in built-in operators and provide the following types of operators:
 - Arithmetic operator
 - Relation operator
 - Assignment operator (Compound Assignments)
 - Unary operator
 - Logical operator
 - Bitwise operator

Arithmetic Operators

Operator	Description	Expression	Translate to
+	Addition	$a + b$	<code>a.plus(b)</code>
-	Subtraction	$a - b$	<code>a.minus(b)</code>
*	Multiply	$a * b$	<code>a.times(b)</code>
/	Division	a / b	<code>a.div(b)</code>
%	Modulus	$a \% b$	<code>a.rem(b)</code>

Relational Operators

Operators	Meaning	Expression	Translate to
>	greater than	<code>a > b</code>	<code>a.compareTo(b) > 0</code>
<	less than	<code>a < b</code>	<code>a.compareTo(b) < 0</code>
>=	greater than or equal to	<code>a >= b</code>	<code>a.compareTo(b) >= 0</code>
<=	less than or equal to	<code>a <= b</code>	<code>a.compareTo(b) <= 0</code>
==	is equal to	<code>a == b</code>	<code>a?.equals(b) ?: (b === null)</code>
!=	not equal to	<code>a != b</code>	<code>!(a?.equals(b) ?: (b === null)) > 0</code>

Note: **?** means it could be NULL possible

Assignment Operators

Operators	Expression	Translate to
+=	a = a + b	a.plusAssign(b) > 0
-=	a = a - b	a.minusAssign(b) < 0
*=	a = a * b	a.timesAssign(b) >= 0
/=	a = a / b	a.divAssign(b) <= 0
%=	a = a % b	a.remAssign(b)

Unary Operators

Operator	Description	Expression	Convert to
+	unary plus	+a	a.unaryPlus()
-	unary minus	-a	a.unaryMinus()
++	increment by 1	++a	a.inc()
--	decrement by 1	--a	a.dec()
!	not	!a	a.not()

Logical Operators

Operator	Name	Description	Example
&&	Logical and	Returns true if both operands are true	x && y
	Logical or	Returns true if either of the operands is true	x y
!	Logical not	Reverse the result, returns false if the operand is true	!x

Bitwise Operators

Function	Description	Example
shl (bits)	signed shift left	x.shl(y)
shr (bits)	signed shift right	x.shr(y)
ushr (bits)	unsigned shift right	x.ushr(y)
and (bits)	bitwise and	x.and(y)
or (bits)	bitwise or	x.or(y)
xor (bits)	bitwise xor	x.xor(y)
inv()	bitwise inverse	x.inv()

Standard Input / Output

- Kotlin standard I/O operations are performed to flow sequence of bytes or byte streams from input device such as Keyboard to the main memory of the system and from main memory to output device such as Monitor.

Standard Output

- Kotlin standard output is the basic operation performed to flow byte streams from main memory to the output device.
- You can output any of the data types integer, float and any patterns or strings on the screen of the system.

You can use any one of the following function to display output on the screen.

print() function

println() function