Лекция #1. Введение в язык Kotlin

Одним из основных инструментов, который мы будем использовать для создания проектов на **Kotlin**, является **IntelliJ IDEA** от **JetBrains**. **JetBrains** также является компанией, стоящей у истоков самого языка, поэтому разработка на **Kotlin** тесно интегрирована в **IntelliJ IDEA**.

IntelliJ IDEA является **Интегрированной средой разработки**, или **IDE**. Она похожа на другие **IDE** вроде **Visual Studio**. **IntelliJ IDEA** предоставляет основу для многих других **IDE** от **JetBrains**, включая **Android Studio** (которую мы и будем использовать в дальнейшем) для разработки приложений на **Android**, **PyCharm** для программирования на **Python** и **CLion** для программирования на **C** и **C++**.

IDE используется для написания кода в редакторе, компиляции кода для запуска на компьютере, просмотра результатов программы, исправления ошибок кода и многого другого.

Установка и настройка IntelliJ IDEA

IntelliJ IDEA можно скачать с сайта **JetBrains**. В наличии есть версии Community и Ultimate. Для обучения подойдет версия Community, которая является бесплатной для скачивания.

Перейдите на сайт и скачайте **IntelliJ IDEA 2021.2.2** или более позднюю версию. Выберите свою систему — macOS, Windows или Linux. Установите **IntelliJ IDEA**, следуя инструкции.

Перед запуском **IntelliJ IDEA** требуется также установить **Java Development Kit**, или **JDK**, который нужен для запуска **Kotlin** кода на компьютере. Следуйте инструкциям при установке **IDE**.

Java и JDK

Kotlin позволяет программировать на нескольких различных платформах. Двумя наиболее популярными являются **Java Virtual Machine**, или **JVM** и **Android**.

По большому счету, изначально **Kotlin** задумывался как современная замена языку **Java**. **Java** был создан в девяностых, став ранней попыткой кроссплатформенного прикладного языка программирования. Подход “Write Once, Run Everywhere” звучал многообещающе.

Вместо компиляции в нативный машинный код на каждой платформе, программы на **Java** компилируются в формат, который называется байт-кодом. Байт-код выполняется внутри приложения на **Java Virtual Machine**. **JVM** можно рассматривать как слой над вашей реальной машиной. Запустив байт-код на виртуальной машине, можно совместно использовать **Java**-код и приложения на многих типах компьютерных систем.

Одной из целей языка **Kotlin** является 100% совместимость с языком Java. Это включает конвертацию **Kotlin**-кода в **Java**-совместимый код с помощью компилятора **Kotlin**, чтобы **Kotlin**-код мог запускаться на **JVM**.

Большая часть кода и проектов из данного курса предназначены для запуска в качестве **Kotlin**-проектов на **JVM**. Для этого наравне с **IntelliJ IDEA** требуется установить **JDK**. Проще всего это сделать на сайте **Oracle**. Лучше скачать и установить самую последнюю версию **JDK** — по крайней мере начиная с 8 версии. Инструменты программного обеспечения **Java** называются **«Java SE»**. Они включают **JDK** и **Java Runtime Environment**, или **JRE**.

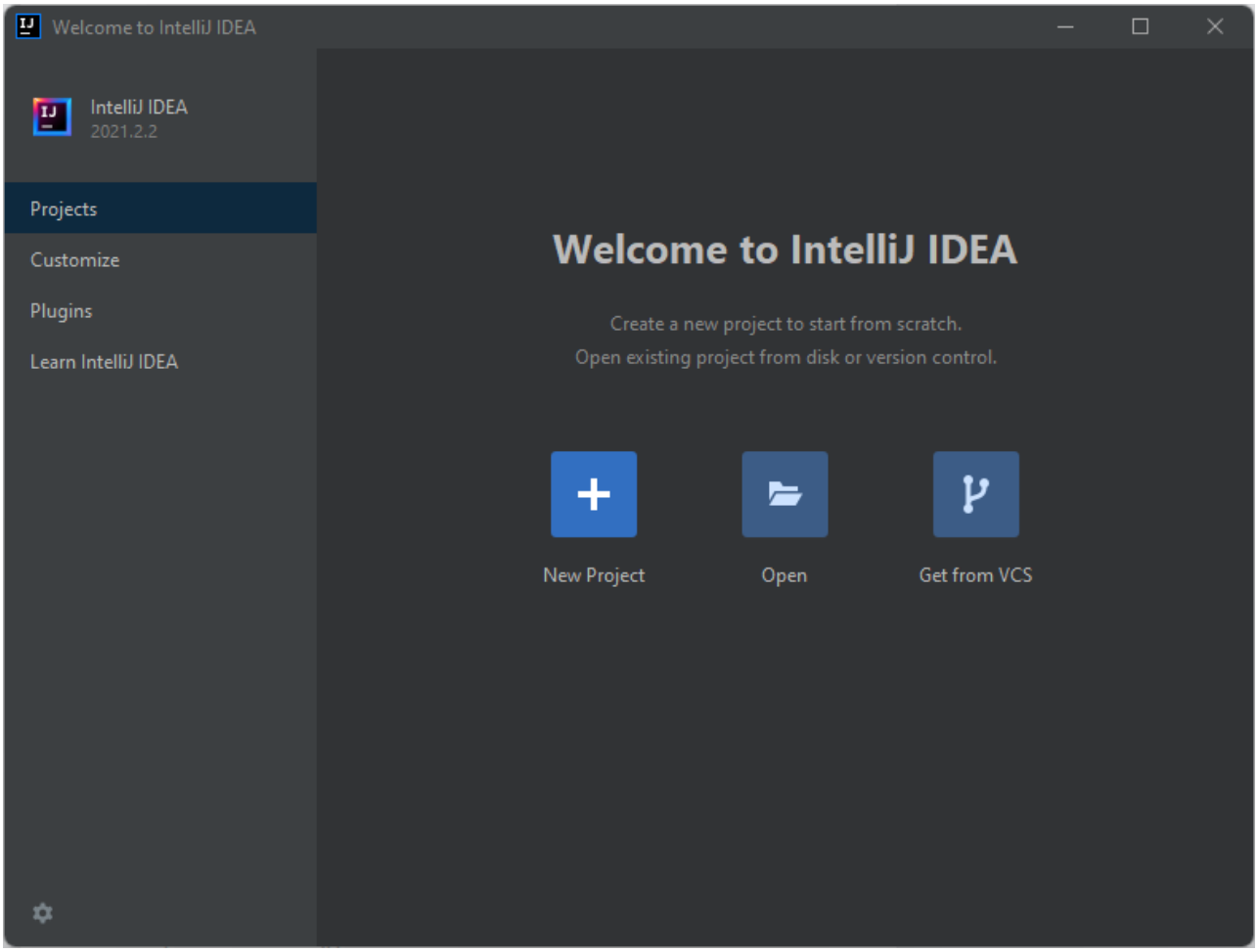
На заметку: Будьте внимательны, скачайте и установите **JDK**, а не только **JRE**, так как **JRE** позволяет запускать только **Java**-приложения и не включает инструменты для создания новых.

Запуск IntelliJ IDEA

После установки **IntelliJ IDEA** и **JDK** выполните обычный процесс запуска приложения **IntelliJ IDEA** на вашей платформе.

Если вы ранее устанавливали предыдущие версии **IntelliJ IDEA** на тот же компьютер, установщик, скорее всего, предложит импортировать настройки из предыдущей версии. В противном случае вам будет предложено выбрать цветовую тему и плагины для установки в **IDE**. Можете просто выбрать настройки по умолчанию и продолжить.

После этих действий вы увидите окно **Welcome to IntelliJ IDEA**.



Из приветственного окна можно создать новый проект, импортировать или открыть существующие проекты, извлечь код из системы контроля версий вроде **Git**, запустить инструменты настройки или получить справку по **IDE**.

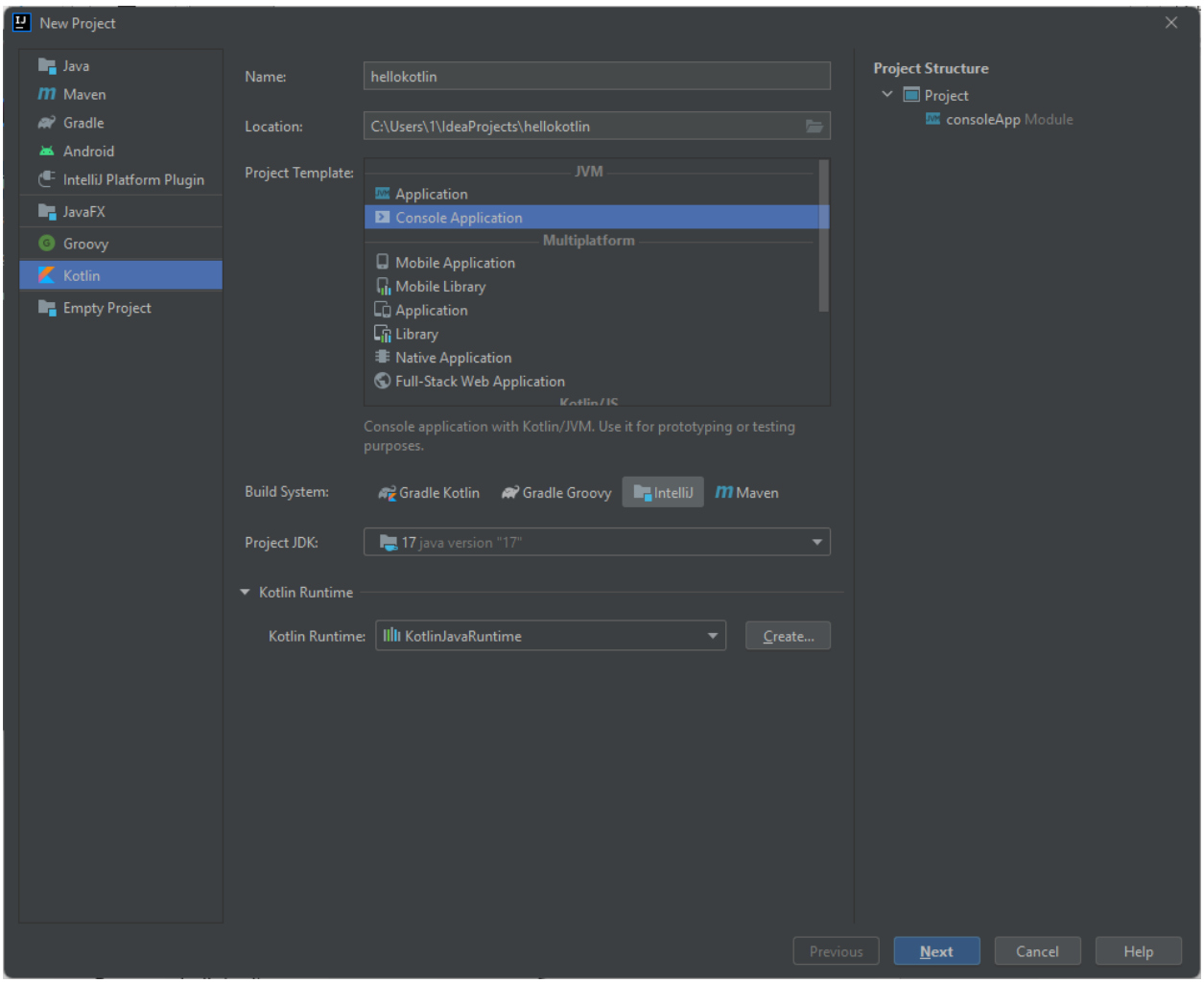
Создание простого проекта на Kotlin

В приветственном окне выберите пункт **Create New Project**. Вы увидите первый экран конфигурации.

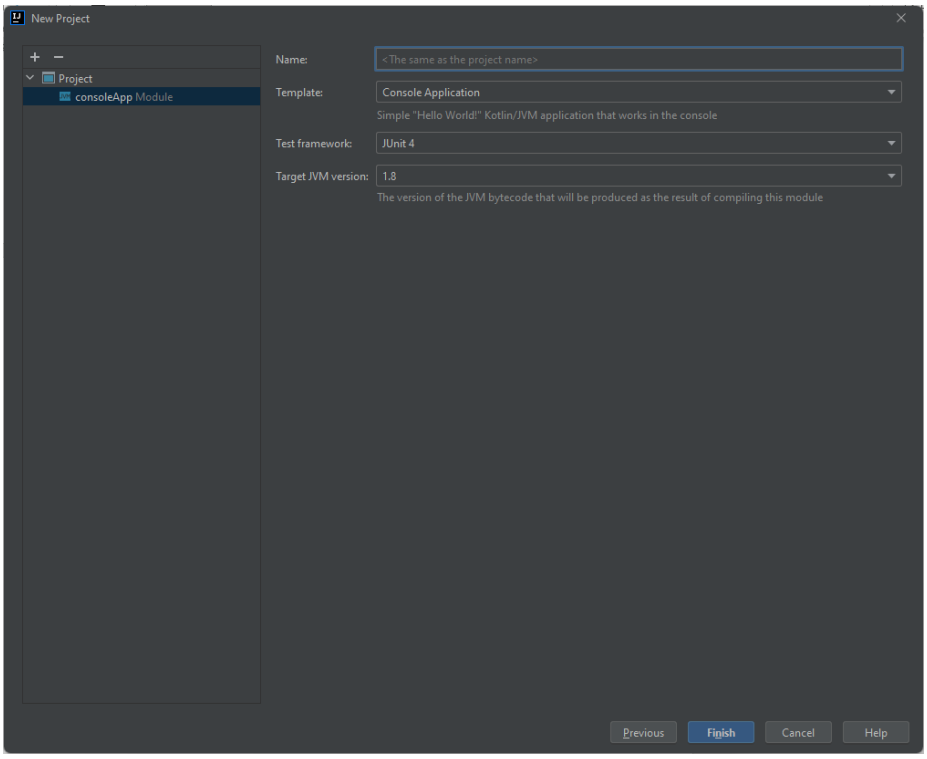
Выберите **Kotlin** из списка опций слева, **Console Application** в качестве типа проекта, **IntelliJ** в качестве системы сборки и нажмите **Next**.

Вы также увидите **Project SDK**, который должен быть установленной ранее версией **JDK** — или другой версией **JDK**, если у вас на ПК установлено более одной версии.

Вы должны увидеть следующее:



Затем вы увидите экран для проекта с названием и местом хранения файлов.

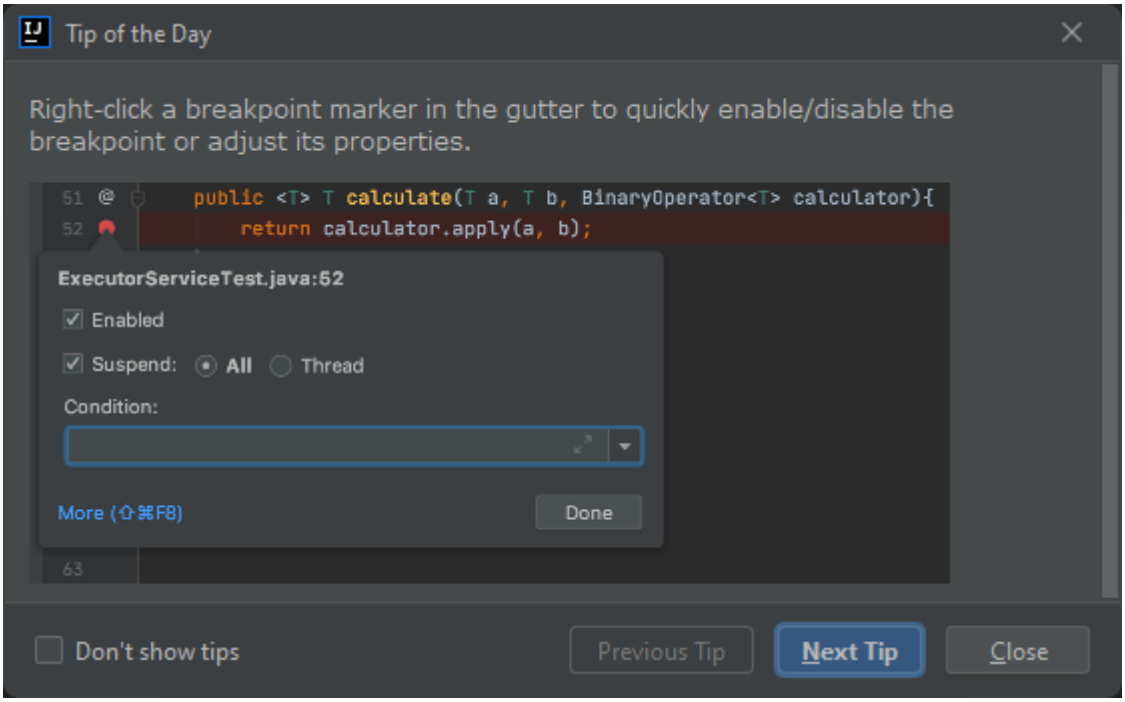


Введите hellokotlin для названия проекта, выберите место для проекта или просто оставьте значения по умолчанию. Нажмите **Finish**.

IntelliJ IDEA создает и конфигурирует ваш проект.

По завершении вы попадете в окно **Tip of the Day**, или Совет дня, в котором каждый раз при открытии приложения показываются полезные советы по **IntelliJ IDEA**.

Вы должны увидеть следующее:



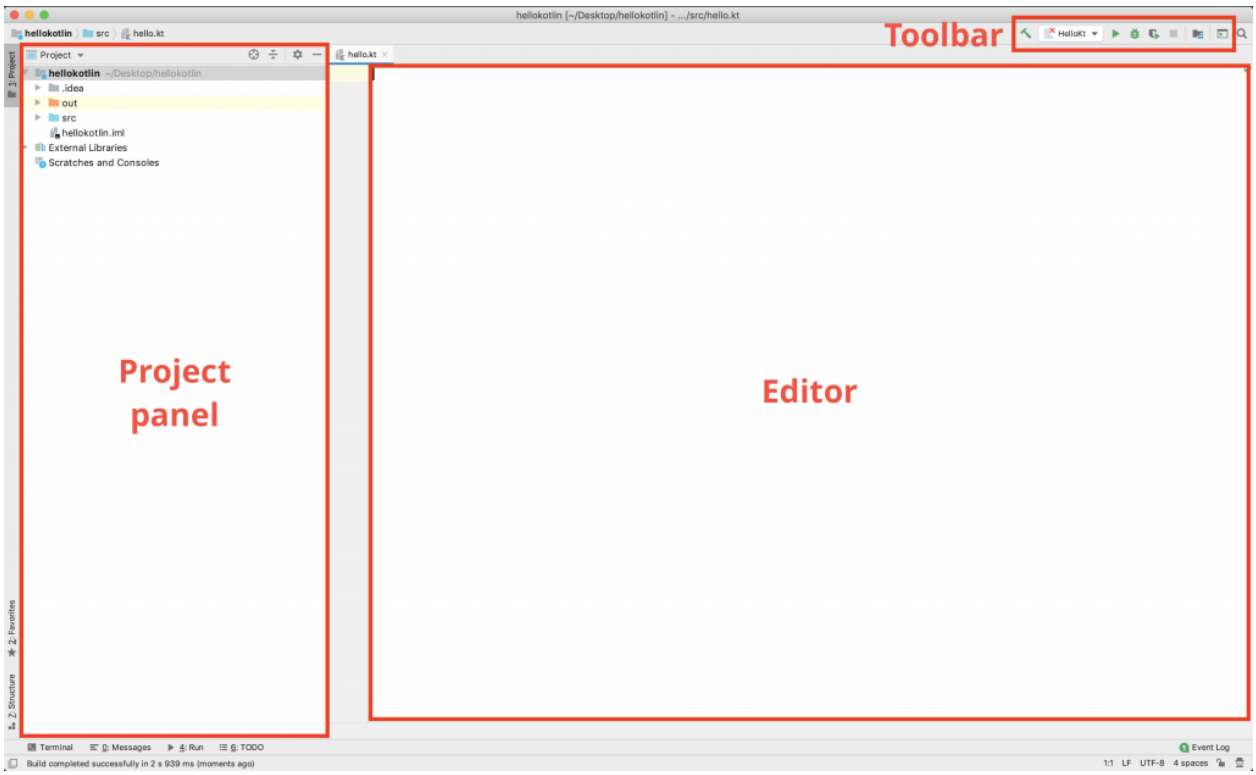
Закройте окно подсказки и проверьте панель **Project** слева от главного окна **IntelliJ IDEA**. На панели **Project** можно управлять всеми файлами, связанными с проектом, например файлами исходного кода **Kotlin**, которые имеют расширение **.kt**.

Кликните на стрелку рядом с hellokotlin, чтобы открыть его содержимое, и вы увидите папку **src** для проекта. Щелкните правой кнопкой мыши на папку src и выберите **New >> Kotlin File/Class**.

Должно открыться новое диалоговое окно **New Kotlin File/Class**. Введите **hello** и кликните OK.

Затем файл hello.kt откроется редакторе **IntelliJ IDEA**.

Базовый макет окна **IntelliJ IDEA** содержит панель **Project** слева, панель **Editor** посередине и **Toolbar** в верхней правой части, который можно использовать для запуска кода.



Пример программы на Kotlin

Мы напишем в редакторе код на **Kotlin** и запустим его, для чего необязательно разбирать все его части. Каждый элемент кода будет подробнее разобран в дальнейшем. Если у вас есть опыт программирования в **Java** или **C#** вы быстро вникните в ход дела.

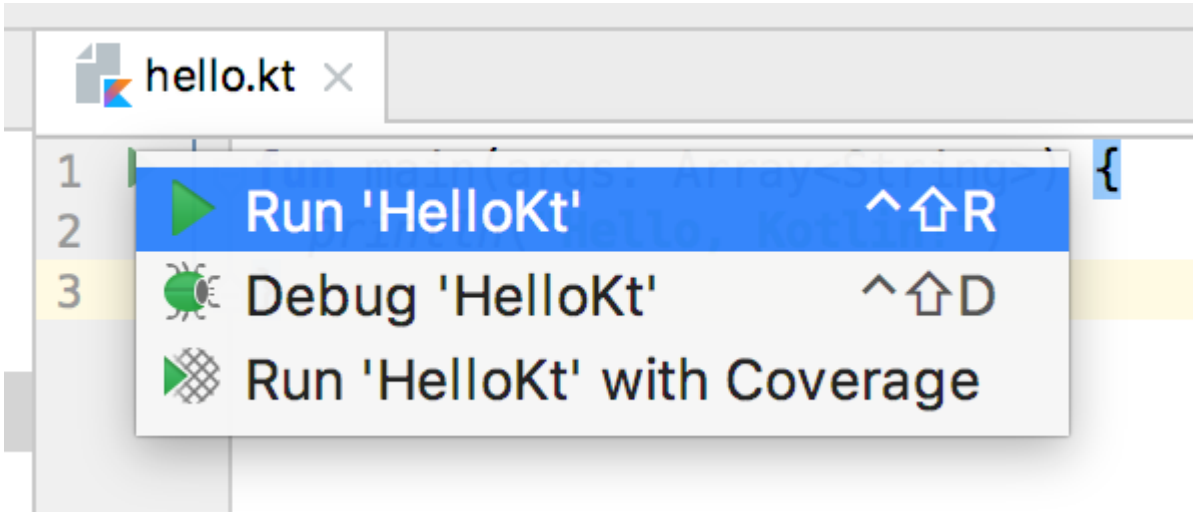
В панели редактора **Editor** файла **hello.kt** введите следующий код в точности, как он записан далее:

```
1  fun main() {
2    println("Hello, Kotlin!")
3  }
```

Мы написали функцию **Kotlin** под названием **main()** и добавили одну строку кода к функции внутри фигурных скобок, которая затем вызывает функцию под названием **println()**. Программа выводит на экран текст “Hello, Kotlin!”. Подробнее о том, как работают функции будем рассматривать в будущем.

Есть несколько способов запустить этот код, в том числе с помощью меню в **IntelliJ IDEA**, панель инструментов или через определенные комбинации клавиш.

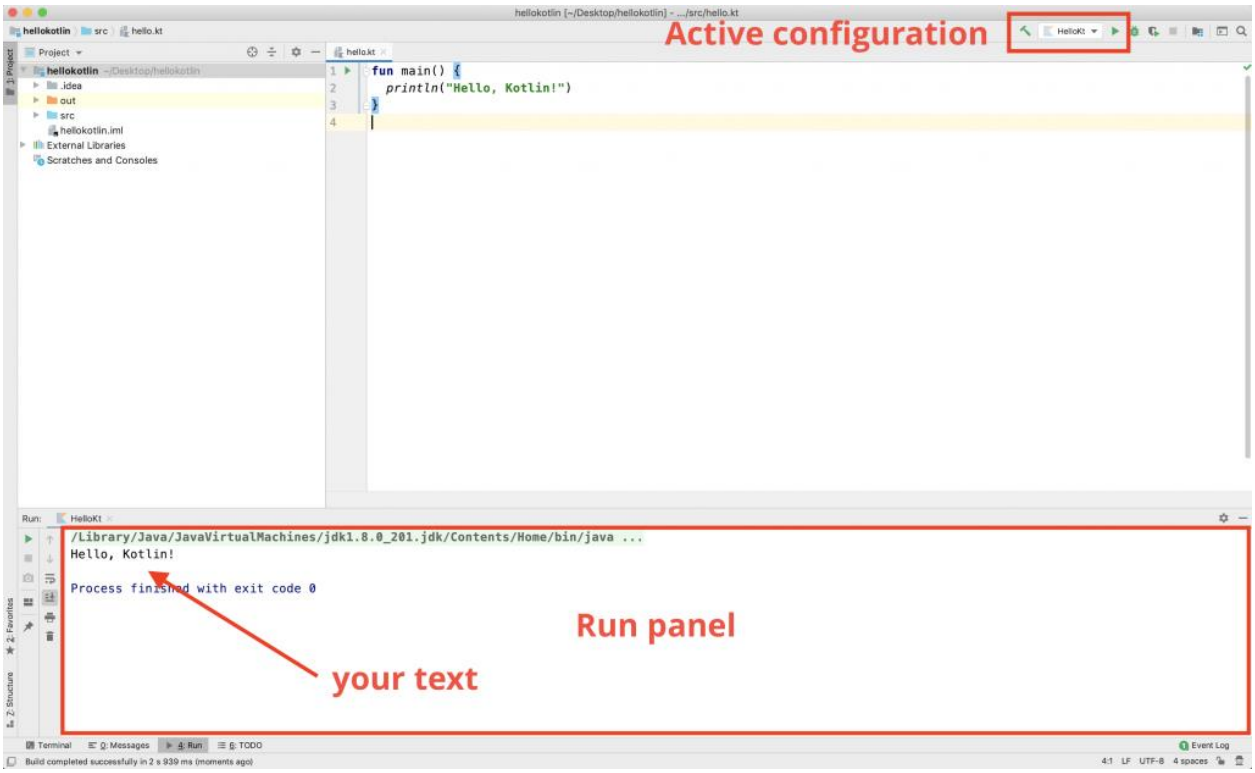
Самый простой способ запустить код — нажать маленькую зеленую кнопку **Run/Play** слева от функции **main()** на панели редактора.



После этого компилятор **Kotlin** сделает парсинг кода, преобразует его в байт-код и затем запустит его на локальной **JVM**.

После этого в нижней части окна **IntelliJ IDEA** откроется панель для запуска **Run**, которую иногда называются консолью.

Вы увидите результат выполнения программы на панели **Run** — в данном случае это текст, который нужно отобразить на экране.



После первого запуска кода на панели инструментов появится активная конфигурация, и вы можете запустить код, нажав на зеленую кнопку **Run** на панели инструментов.

Автоматическая сборка проектов на Gradle

В будущем при изучении **Kotlin** мы будем использовать **Gradle**, чтобы мы могли добавить внешние библиотеки (зависимости) в наш проект.

Gradle является популярным инструментом в экосистеме **Java** для сборки и управления зависимостями. Это чрезвычайно мощный и универсальный инструмент для сборки, и его возможности выходят далеко за рамки данного курса.

Gradle используется в качестве системы сборки приложений на **Android** созданных с помощью **Android Studio**, которая, как упоминалось ранее, основана на **IntelliJ IDEA**.

Чтобы открывать **Gradle** проекты, выполните те же действия, что и для проектов на **JVM Kotlin**. Выберите **File >> Open**, затем переходите к корневой папке проекта и выбираете ее.

IntelliJ IDEA обнаружит, что проект основан на **Gradle**, а затем откроет и настроит проект соответствующим образом.