Введение в платформу Java

Hello, World!

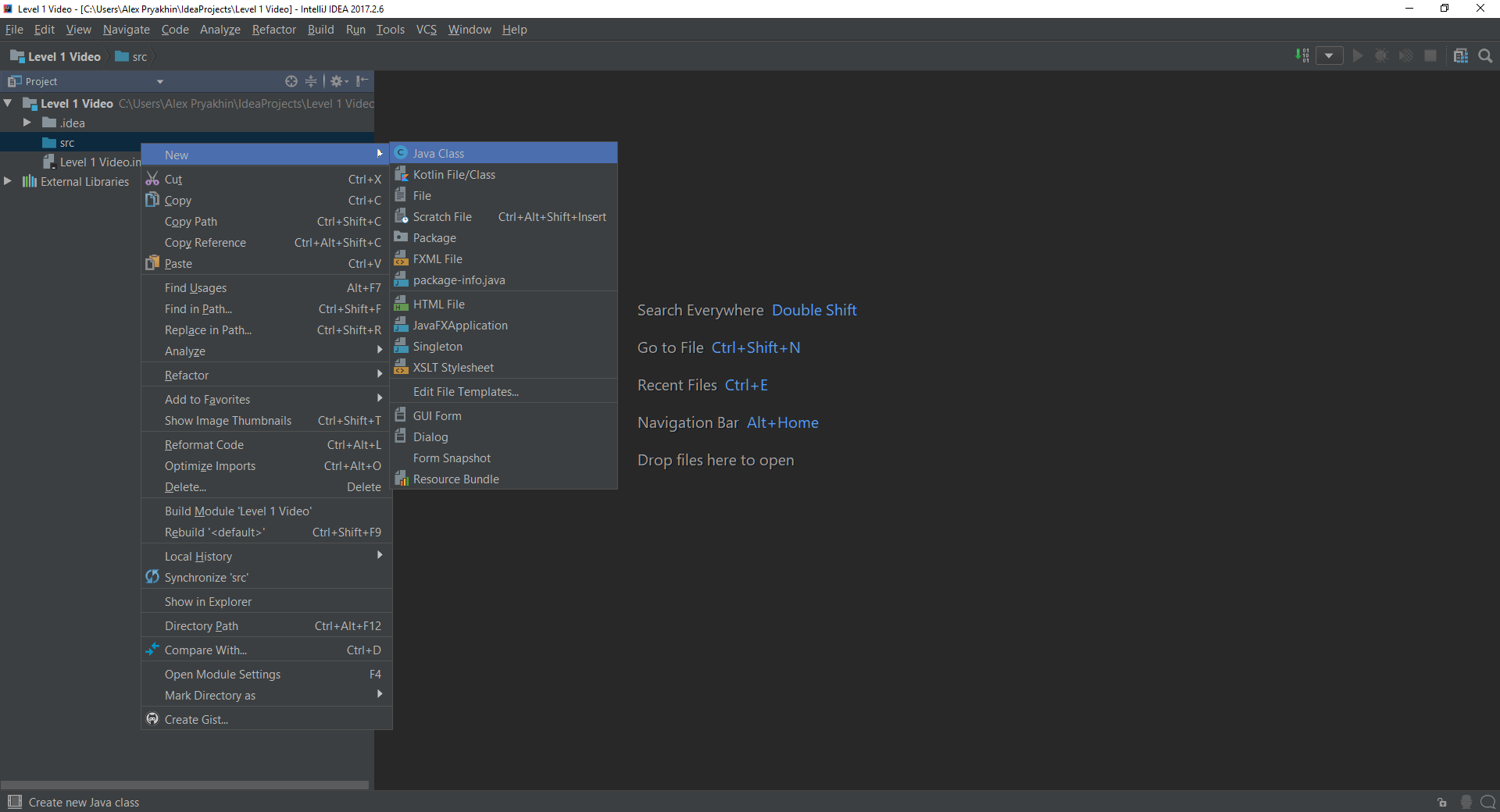
[Создаем приложение](#_2nrk766djf0c)

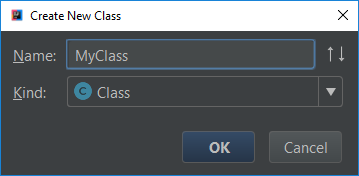
[Hello, World!](#_3tian8sn0psm)

# Создаем приложение

На прошлом занятии мы познакомились с языком Java и подготовили рабочее место, чтобы начать кодить. Приступим к созданию приложений.

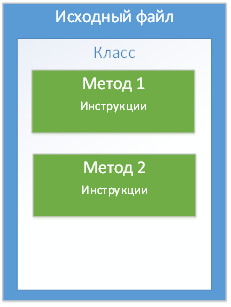
Создадим новый проект в IDE и разместим в его папке **src** файл с кодом. В Java у таких файлов расширение **.java**.





Получим файл **MyClass.java** и класс **MyClass**.

Разберемся, что будет происходить в механизмах **Java**. Схематическая диаграмма файла с кодом:



Этот файл содержит код класса. Сам класс — составляющая программы. Более подробно познакомимся с понятием класса позже, пока достаточно такого определения. Небольшие программы в Java вполне могут обходиться одним классом. Его код находится в фигурных скобках.

В начале работы JVM ищет класс с именем, которое ей передали при старте. В классе должен быть главный метод — **main**, без которого программа работать не будет. JVM выполнит все инструкции между фигурными скобками главного метода.

Обобщим: любая программа в Java содержит минимум один класс и метод main для него. Если классов больше, в них наличие main не обязательно.

# Hello, World!

|  |
| --- |
| /\*\*  \* Created by GeekBrains student  \*/  public class MyClass {  public static void main(String[] args) {  System.*out*.println("Hello, World!");  } } |

В первых строках кода видим комментарии: они могут быть однострочными (начинаются с символов //) и многострочными (блок текста, заключенный в /\* … \*/). Комментарии упрощают работу с кодом: разработчик может делать пометки, которые не будут влиять ни на размер программы, ни на скорость ее выполнения.

|  |
| --- |
| *// Однострочный комментарий*  */\* Блочный комментарий \*/* |

После комментариев идет объявление класса **MyClass** (названия класса и файла, в котором он создан, должны совпадать).

Далее в коде видим объявление метода **main()**. Выполнение любой Java-программы начинается с его вызова. Ключевое слово **void** сообщает компилятору, что **main()** не возвращает значений (позже покажем, что методы могут и возвращать конкретные значения). Следует иметь в виду, что в Java учитывается регистр символов, и **Main** не равнозначно **main**. О модификаторах доступа (**public**, **private**), ключевом слове **static** и **String[] args** поговорим в дальнейшем, в теме об основах ООП и массивах.

В следующей строке кода — **System.out.println("Hello, World!")**, которая находится в теле метода **main()**. В консоль выводится текстовая строка «Hello, World!», и каретка переводится на новую строку. Вывод текста на экран выполняется встроенным методом **println()**, который может выводить также числа, символы, значения переменных и другое.

В Java все операторы должны оканчиваться символом «;» . Точка с запятой отсутствует после закрытия кодовых блоков фигурной скобкой (}), так как формально фигурные скобки не являются операторами.

Запустим наш код. В IDE старт возможен при наличии метода **main** в классе. Можно запускать через меню **Run -> Run…** или комбинацию горячих клавиш **Alt + Shift + F10**. Если работа идет через консоль, нужно перейти в директорию проекта, найти файл **MyClass.java**. Это класс с кодом, и его надо превратить в байт-код для выполнения на JVM. Для этого применим команду:

|  |
| --- |
| javac MyClass.java |

По результатам работы компилятора получим файл, который можно запускать в рамках системы. В нашем случае — это **MyClass.class**. Чтобы запустить программу, введите команду:

|  |
| --- |
| java MyClass |

Заметьте: расширение указывать не нужно.

http://dl4.joxi.net/drive/2018/01/23/0001/0430/78254/54/f3866d27e5.jpg