**Язык Java – Лекция 2. Выполнение первой программы на Java**

**2.1. Запуск простейшей java-программы в консоли windows**

**Шаг 1. Установка jdk и работа с содержанием соответствующих каталогов**

Сайт для скачивания последней версии Java - <https://www.oracle.com/java/technologies/javase/jdk19-archive-downloads.html>

Папка, в которую устанавливается jdk:

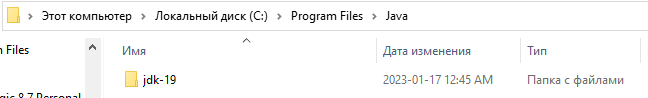


Рисунок 2.1

Раскрываем папку jdk, чтобы узнать её состав:

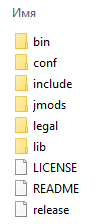


Рисунок 2.2. Совокупность папок первого уровня, содержащих файлы java

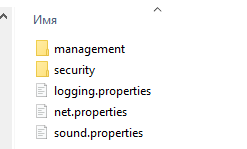


Рисунок 2.3. Содержимое папки conf

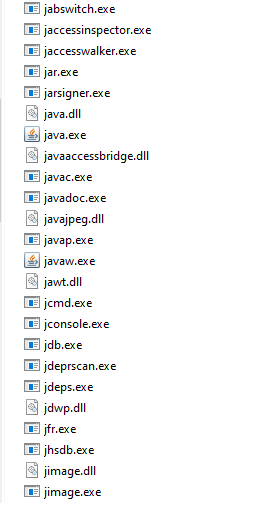


Рисунок **2.4. Содержимое папки bin.**

**Задание 2**

**Узнать о назначении базовых исполняемых файлов (утилит) с расширением .exe.**

**Шаг 2. Создание первого приложения на языке Java**

Что необходимо для создания программы на Java? Прежде всего нам надо написать код программы, и для этого нужен текстовый редактор. Можно использовать любой текстовый редактор, например, **Notepad++**.

**И чтобы превратить код программы в исполняемое приложение необходим компилятор. После установки JDK все файлы по умолчанию помещаются в каталог C:\Program Files\Java\jdk-[номер\_версии] (при использовании ОС Windows).**

Если мы откроем в нем подкаталог bin, то мы сможем увидеть в нем ряд утилит. Нас прежде всего интересует утилита компилятора **javac**.**exe**. Чтобы **скомпилировать** класс программы, нам надо передать ее код этому компилятору.

Также следует отметить другую утилиту из этой папки - **java.exe**, которая позволяет запускать скомпилированную программу.

Итак, создадим на жестком диске какой-нибудь каталог, в котором будут располагаться файлы с исходным кодом на языке Java.

Название каталога может быть любым. Создадим каталог для программ, исполняемых в консоли с названием Java\_Console: **C:/Java\_Console**. Затем в редакторе NotePad++ (или любом текстовом редакторе) текстовый файл (рисунок 1), который назовём как **Program.java** и сохраним его в созданном каталоге (рисунок 2).

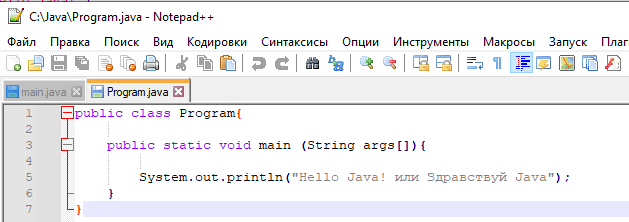


Рисунок 2.5. Код первой программы на Java в NotePad

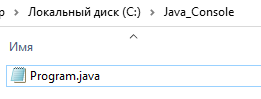


Рисунок 2.6. Вид программы в каталоге

***Учтите: имя файла должно совпадать с именем класса***!

**Шаг 3. Объяснение кода программы**

Java является объектно-ориентированным языком, поэтому вся программа представляется в виде набора взаимодействующих классов**. В данном случае определен один класс Program.**

При определении класса вначале идет модификатор доступа **public**, который указывает, что данный класс будет доступен всем, то есть мы сможем его запустить из командной строки. Далее идет ключевое слово class, а затем название класса. То есть класс называется Program. После названия в фигурных скобках расположено содержимое класса.

Класс может содержать различные переменные и методы. В данном случае у нас объявлен **один метод main**.

**Это главный метод в любой программе на Java, он является входной точкой программы и с него начинается все управление. Он обязательно должен присутствовать в программе**.

Метод main также имеет модификатор **public**. Слово **static** указывает, что метод main - статический, а слово **void** - что он не возвращает никакого значения. Позже мы подробнее разберем, что все это значит.

Далее в скобках у нас идут **параметры метода** - **String args[]** - это массив args, который хранит значения типа String, то есть строки.

В данном случае ни нам пока не нужны, но в реальной программе это те строковые параметры, которые передаются при запуске программы из командной строки.

После списка параметров в **фигурных скобках** идет **тело метода** - это те инструкции, которые и будет выполнять метод. В данном случае фактически определена только одна **инструкция** - вывод на консоль некоторой строки.

Для вывода на консоль используется **встроенный** метод System.out.println(). В этот метод передается выводимая строка. Каждая инструкция завершается точкой с запятой.

**Шаг 4. Компиляция и выполнение файла java через консоль**

**Первым делом** перейдем в каталог, где лежит наш файл с программой с помощью команды из каталога пользователя, который предлагается по умолчанию при открытии консоли командной строки.

Далее указать полный путь до утилиты javac.exe и «подложить» ей для компиляции нужный файл (рисунок 2.7):

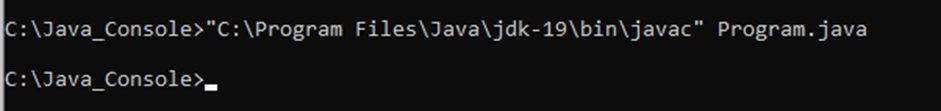


Рисунок 2.7.

Путь указывается в кавычках. Название файла для компиляции пишется через пробел. Работа компилятора вызывается нажатием клавиши Enter.

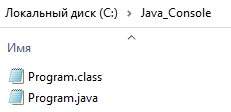


Рисунок 2.7. Содержимое папки после компиляции кода программы.

Компиляция приводит к появлению исполняемого файла Program.class, для выполнения которого надо вызвать утилиту java.exe (рисунок 2.8):



Рисунок 2.8. Команда на выполнение откомпилированной программы java

Расширение исполняемого файлы (.class) использовать уже не нужно.

Однако, выполнение программы происходит мгновенно исчезает с экрана незаметно для фиксации глазом.

Есть несколько способов обойти эту проблему. Подробное описание различных методов и особенностей их использования содержится в <https://javascopes.com/java-delay-code-execution-504111a0/>

**Но самый простой и несложный** заключается в том, что после выполнения команды вывода на консоль вставить команду ввода с консоли. Выполнение кода, дойдя до этой команды, будет ждать ввода. Нажатие на кнопку Enter приведёт к окончанию работы программы.

Подправим в блокноте первую программу и сохраним её в старой папке под именем Program1.java (рисунок 2.9):

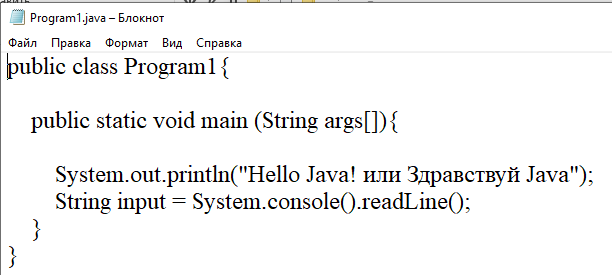


Рисунок 2.9. Программа с вариантом ввода с консоли и клавиатуры

Проделаем те же, что и с программой Program.java, манипуляции с компиляцией и запуском программыProgram1 (рисунок 2.10).

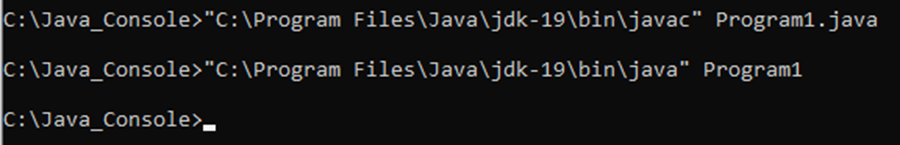


Рисунок 2.10

После компиляции в папке Java\_Console увидим (рисунок 2.11) :

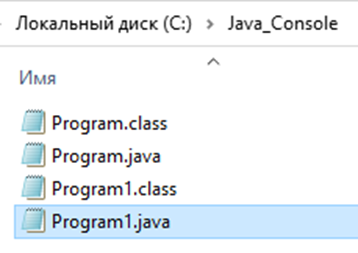


Рисунок 2.11. Содержимое папки после компиляции программы Program1.

При выполнении программы увидим консоль (рисунок 2.12), которая исчезнет после нажатия клавиши Enter:

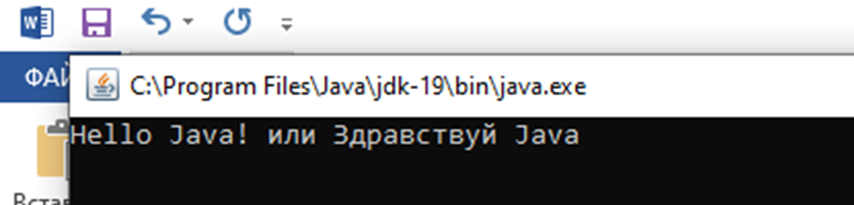


Рисунок 2.12. Результаты работы программы Program1.

Другим способом является использование задержки выполнения скрипта на заданное время использованием механизма прерывания по исключительным ситуациям. Этот приём обеспечивает минимальную задержку на 0.001 секунды. Ниже на рисунке 2.13 приведён текст программы, реализующей задержку на 10 секунд.

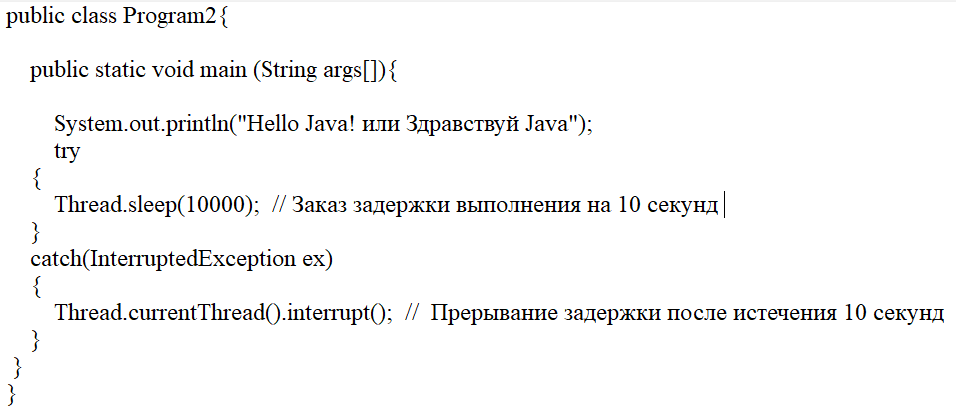


Рисунок 2.13. Текст программы, реализующей задержку выполнения на 10 секунд.

Об использовании кириллицы в Java надо сказать, что, начиная с 18-ой версии, в Java используется (назначена по умолчанию) кодировка utf-8 (<https://ru.wikipedia.org/wiki/UTF-8> ). Эта кодировка понимает русские буквы!

**Задание 3. Проверить версии программ 1 и 2.**

**2.2 Запуск программы в IntelliJ Idea**