Лекция №2.
по курсу «Основы Java»

Москва 2020

Современные платформы для разработки программ

- **Java (*Sun Microsystems, 1995*) – платформа для разработки программ на объектно-ориентированном языке *Java*, программы на котором компилируются в Java байт-код (до сих пор имеет статус ведомственного стандарта Sun)**
- **.NET (*Microsoft, 2000*) – *многоязыковая* объектно-ориентированная платформа для разработки программ с общим промежуточным языком (CIL), общей инфтаструктурой языков (CLI) и единым представлением данных на основе XML (стандарты *ISO/ ECMA)*. Язык C# - наиболее удобный язык программирования для .NET*, но не единственный и не обязательный для использования***
- **Обе платформы уделяют особое внимание надежности и безопасности, на основе *исполнения управляемого кода* и *динамического контроля топов.* И Java, и .NET – наиболее безопасные платформы**

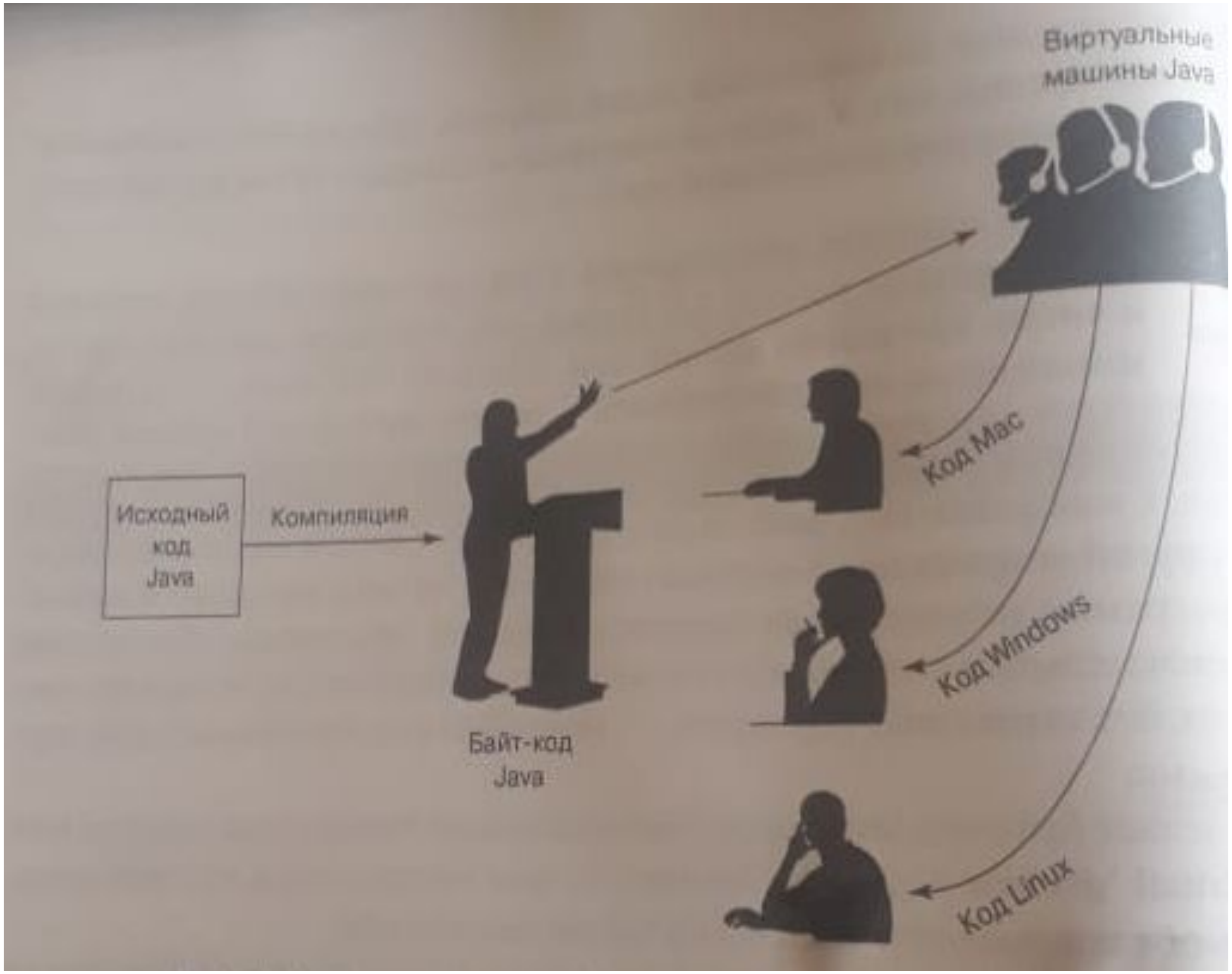
Виртуальная машина Java

Посредник между байт-кодом и конкретной платформой

Байт-код

Портируемая (переносимая) программа

Java Development Kit (инструмент комплект средств поддержки разработок)


Виртуальные
машины Java
Исходный
код
Java
Компиляция
Байт-код
Java
Код Mac
Код Windows
Код Linux

Работа с Java

```
import java.awt.*;
import java.awt.event.*;
class Party {
  public void buildInvite() {
    Frame f = new Frame();
    Label l = new Label("Вечеринка у Тима");
    Button b = new Button("Ваша ставка");
    Button c = new Button("Сбросить");
    Panel p = new Panel();
    p.add(l);
  } // Еще код...
}
```

Исходник

1

Наберите свой исходный код.

Сохраните его как ***Party.java***.

```
File Edit Window Help Plead
%javac Party.java
```

Компилятор

2

Скомпилируйте файл ***Party.java***, запустив утилиту javac (приложение-компилятор). Если все пройдет без ошибок, вы получите еще один файл с именем ***Party.class***.

Файл **Party.class**, сгенерированный компилятором, состоит из байт-кода.

```
Method Party()
  0 aload_0
  1 invokespecial #1 <Method java.lang.Object()>
  4 return
Method void buildInvite()
  0 new #2 <Class java.awt.Frame>
  3 dup
  4 invokespecial #3 <Method java.awt.Frame()>
```

Вывод (код)

3

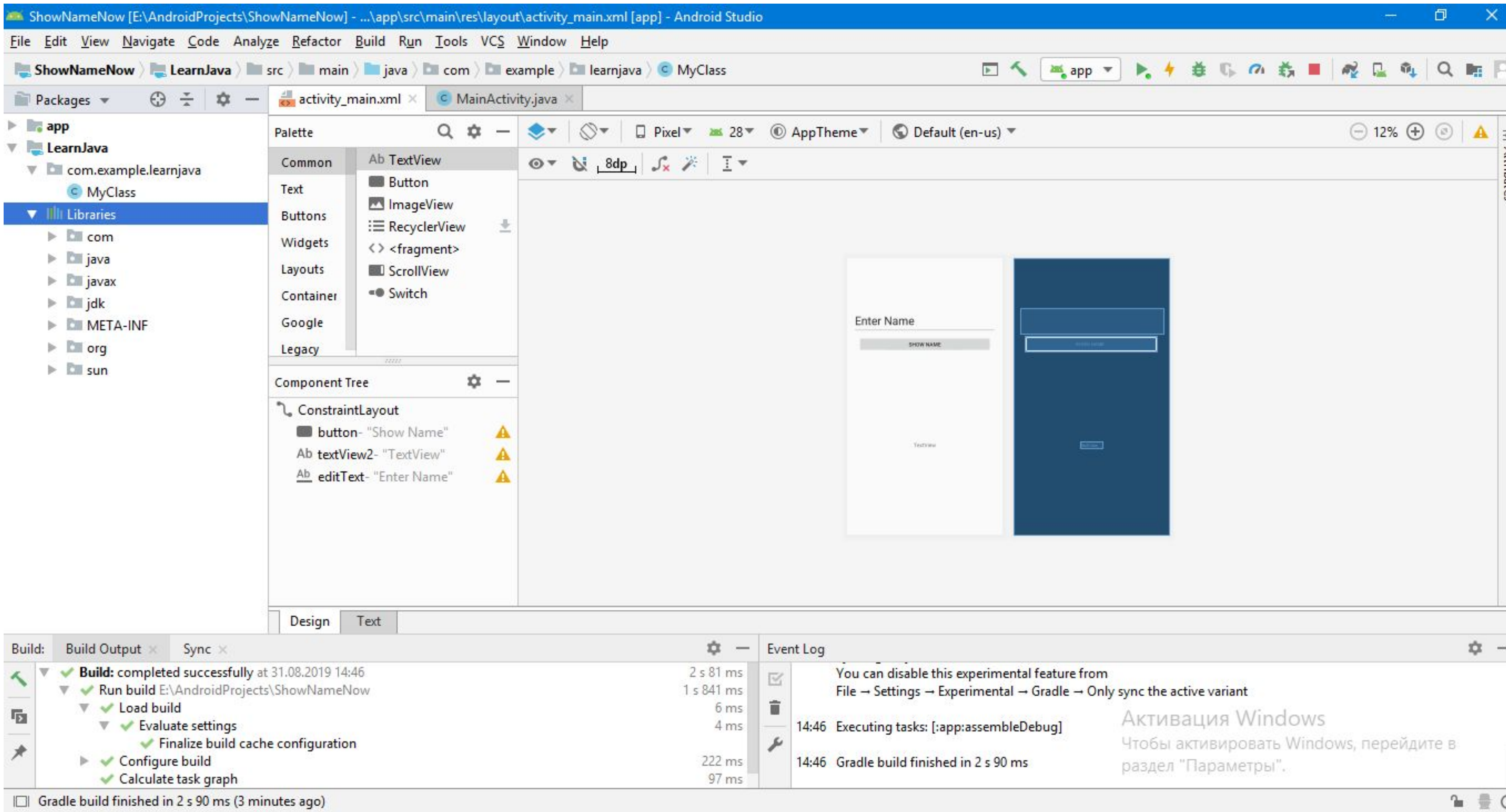
Скомпилированный код: файл ***Party.class***.

Виртуальные машины

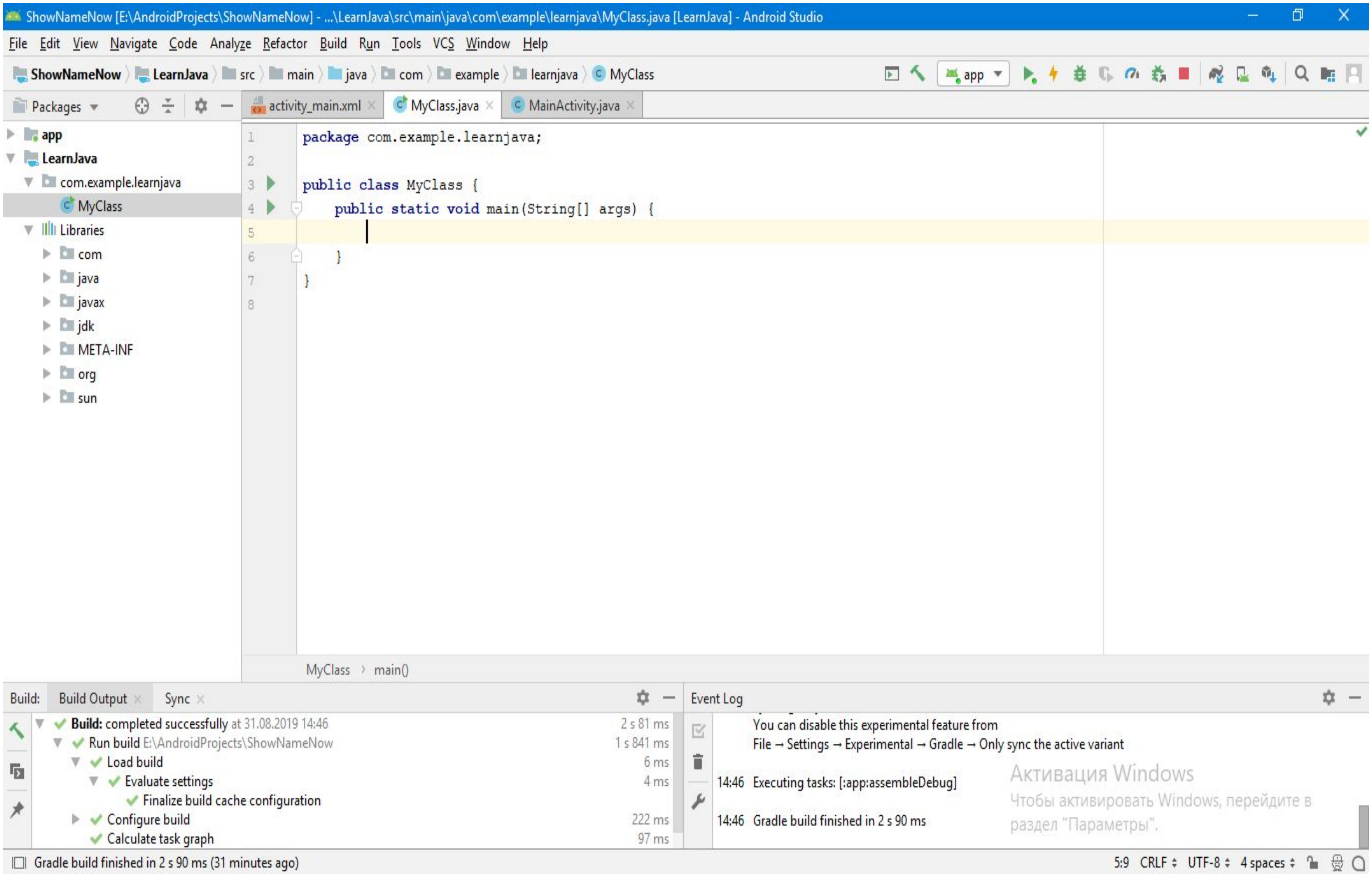
4

Выполните программу, запустив виртуальную машину Java (Java Virtual Machine, или JVM) с файлом **Party.class**. JVM транслирует байт-код в такой формат, который поймет целевая платформа, и запустит вашу программу.

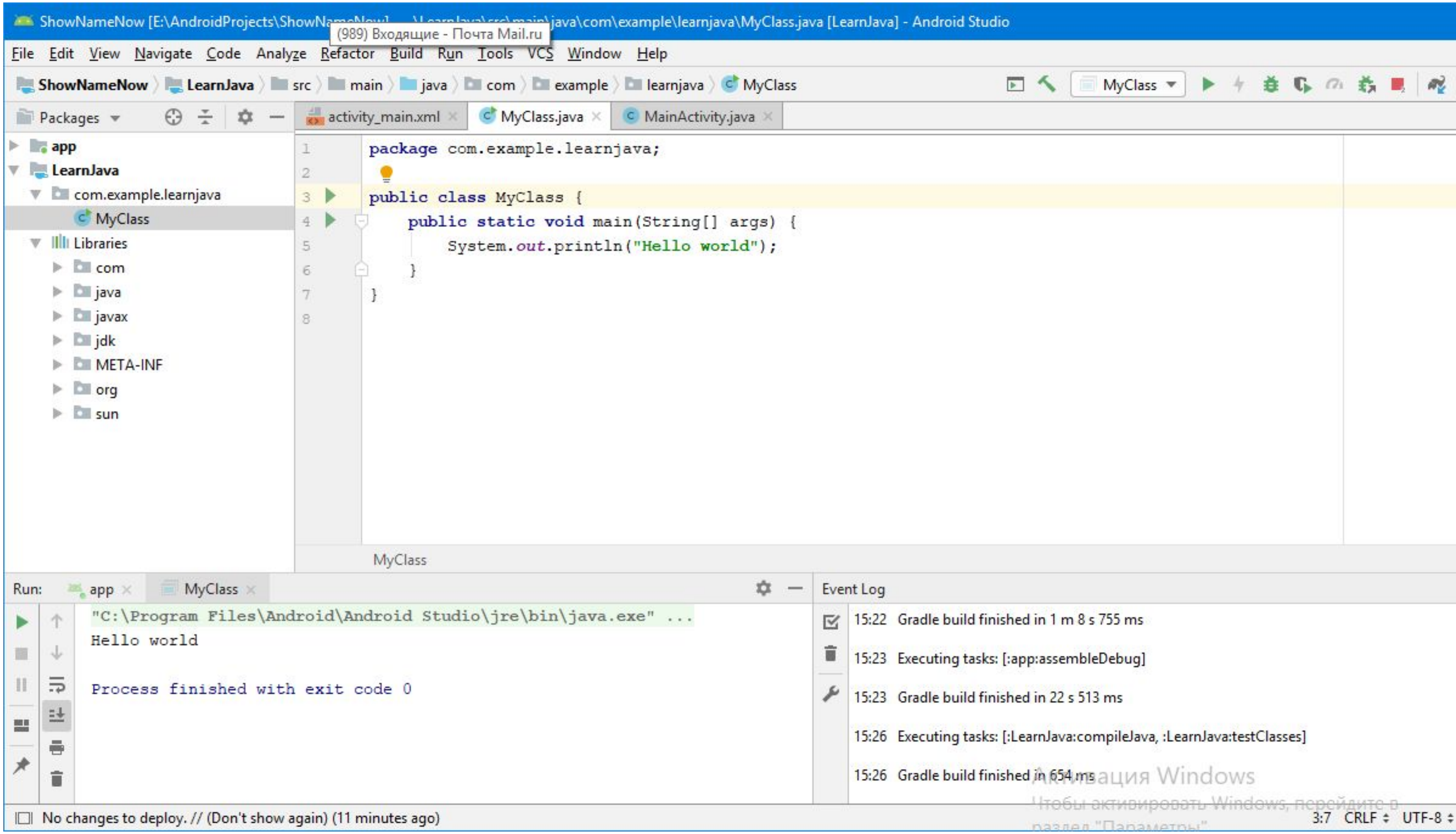
Создание модуля



Java code

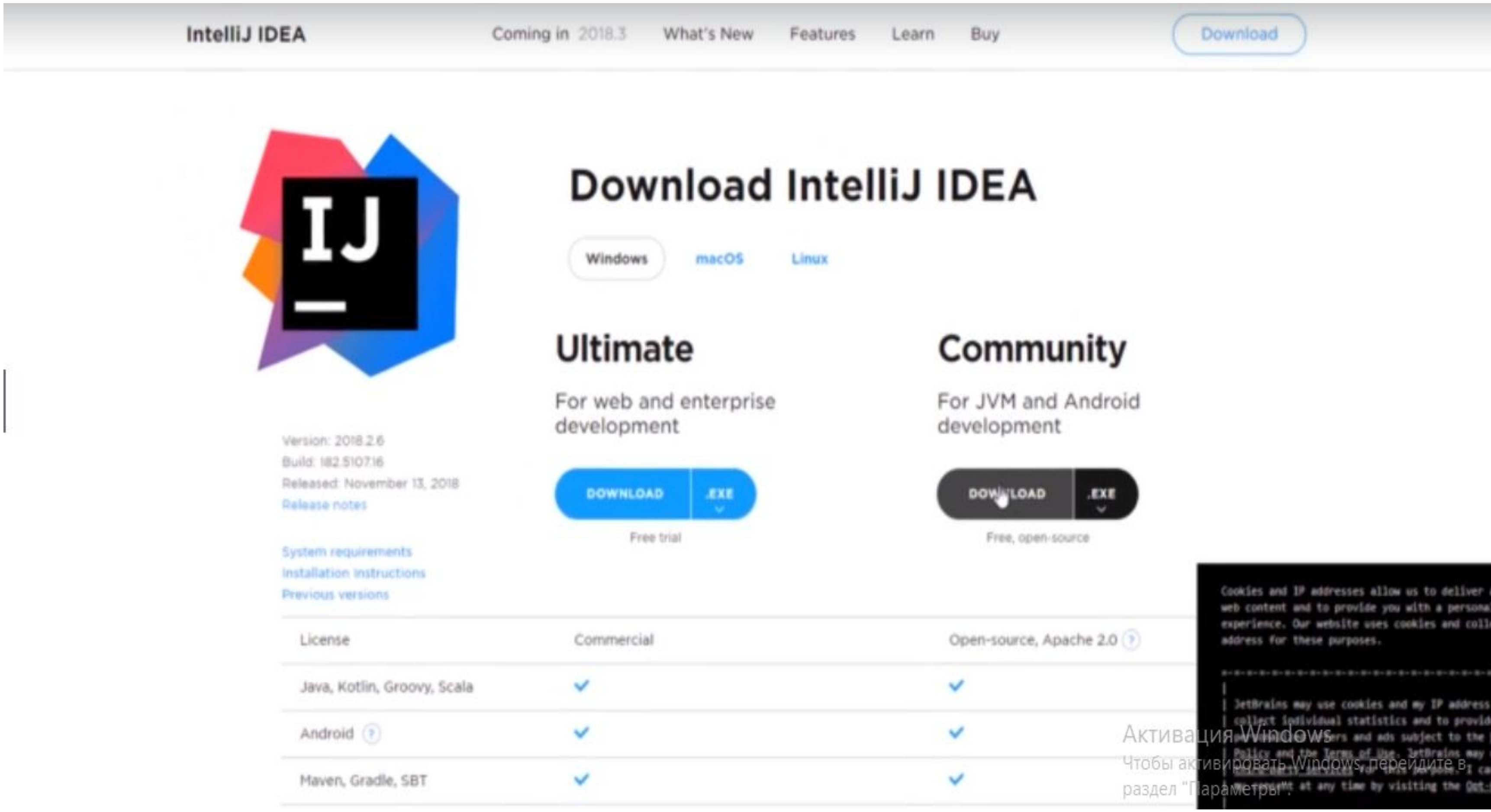

ShowNameNow [E:\AndroidProjects\ShowNameNow] - ...\LearnJava\src\main\java\com\example\learnjava\MyClass.java [LearnJava] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
ShowNameNow LearnJava src main java com example learnjava MyClass
app
Packages
activity_main.xml
MyClass.java
MainActivity.java
app
LearnJava
com.example.learnjava
MyClass
Libraries
com
java
javax
jdk
META-INF
org
sun
package com.example.learnjava;
public class MyClass {
public static void main(String[] args) {
}
}
MyClass › main()
Build: Build Output Sync
Build: completed successfully at 31.08.2019 14:46
Run build E:\AndroidProjects\ShowNameNow
Load build
Evaluate settings
Finalize build cache configuration
Configure build
Calculate task graph
2 s 81 ms
1 s 841 ms
6 ms
4 ms
222 ms
97 ms
Event Log
You can disable this experimental feature from
File → Settings → Experimental → Gradle → Only sync the active variant
14:46 Executing tasks: [:app:assembleDebug]
14:46 Gradle build finished in 2 s 90 ms
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".
Gradle build finished in 2 s 90 ms (31 minutes ago)
5:9 CRLF UTF-8 4 spaces

Java code

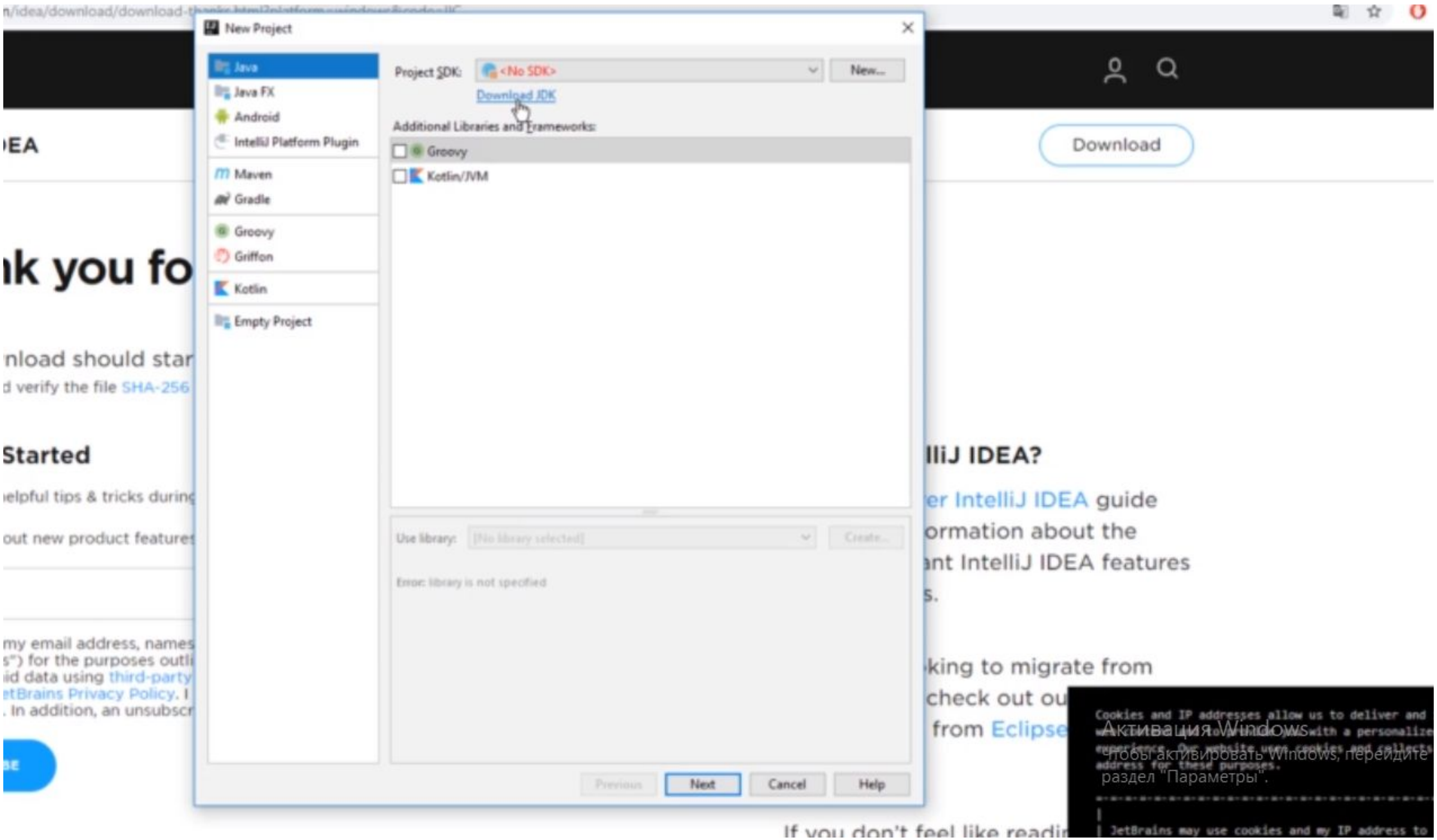


ShowNameNow [E:\AndroidProjects\ShowNameNow\...\LearnJava\src\main\java\com\example\learnjava\MyClass.java [LearnJava] - Android Studio
(989) Входящие - Почта Mail.ru
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
ShowNameNow LearnJava src main java com example learnjava MyClass
MyClass
Packages
activity_main.xml
MyClass.java
MainActivity.java
app
LearnJava
com.example.learnjava
MyClass
Libraries
com
java
javax
jdk
META-INF
org
sun
package com.example.learnjava;
public class MyClass {
public static void main(String[] args) {
System.out.println("Hello world");
}
}
MyClass
Run:
app
MyClass
"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...
Hello world
Process finished with exit code 0
Event Log
15:22 Gradle build finished in 1 m 8 s 755 ms
15:23 Executing tasks: [:app:assembleDebug]
15:23 Gradle build finished in 22 s 513 ms
15:26 Executing tasks: [:LearnJava:compileJava, :LearnJava:testClasses]
15:26 Gradle build finished in 654 ms
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".
No changes to deploy. // (Don't show again) (11 minutes ago)
3:7 CRLF UTF-8

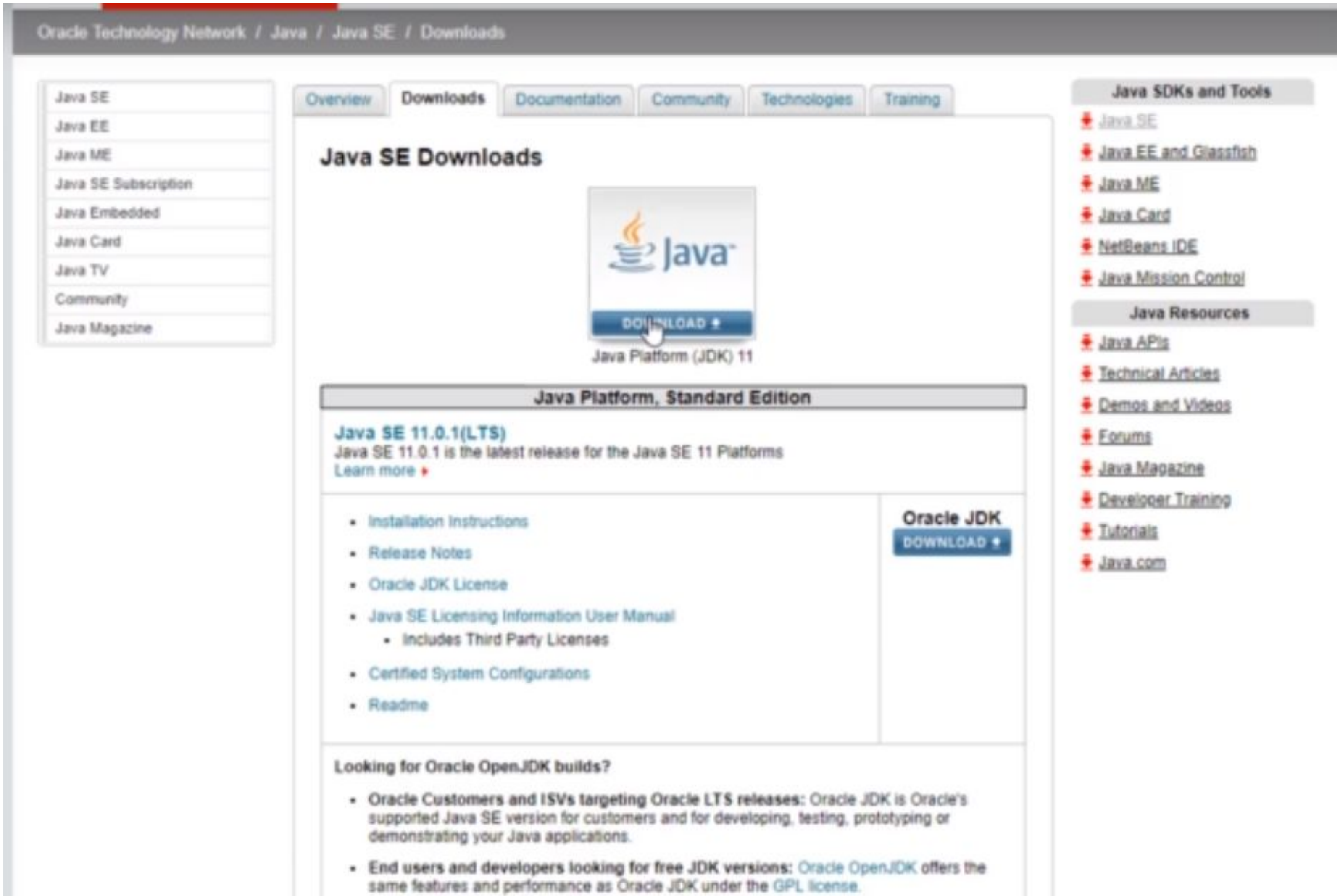
IntelliJ IDEA



Создание проекта IntelliJ IDEA



Установка Java development kit



Установка jdk

- Java Developer Day hands-on workshops (free) and other events
- Java Magazine

JDK 11.0.1 checksum

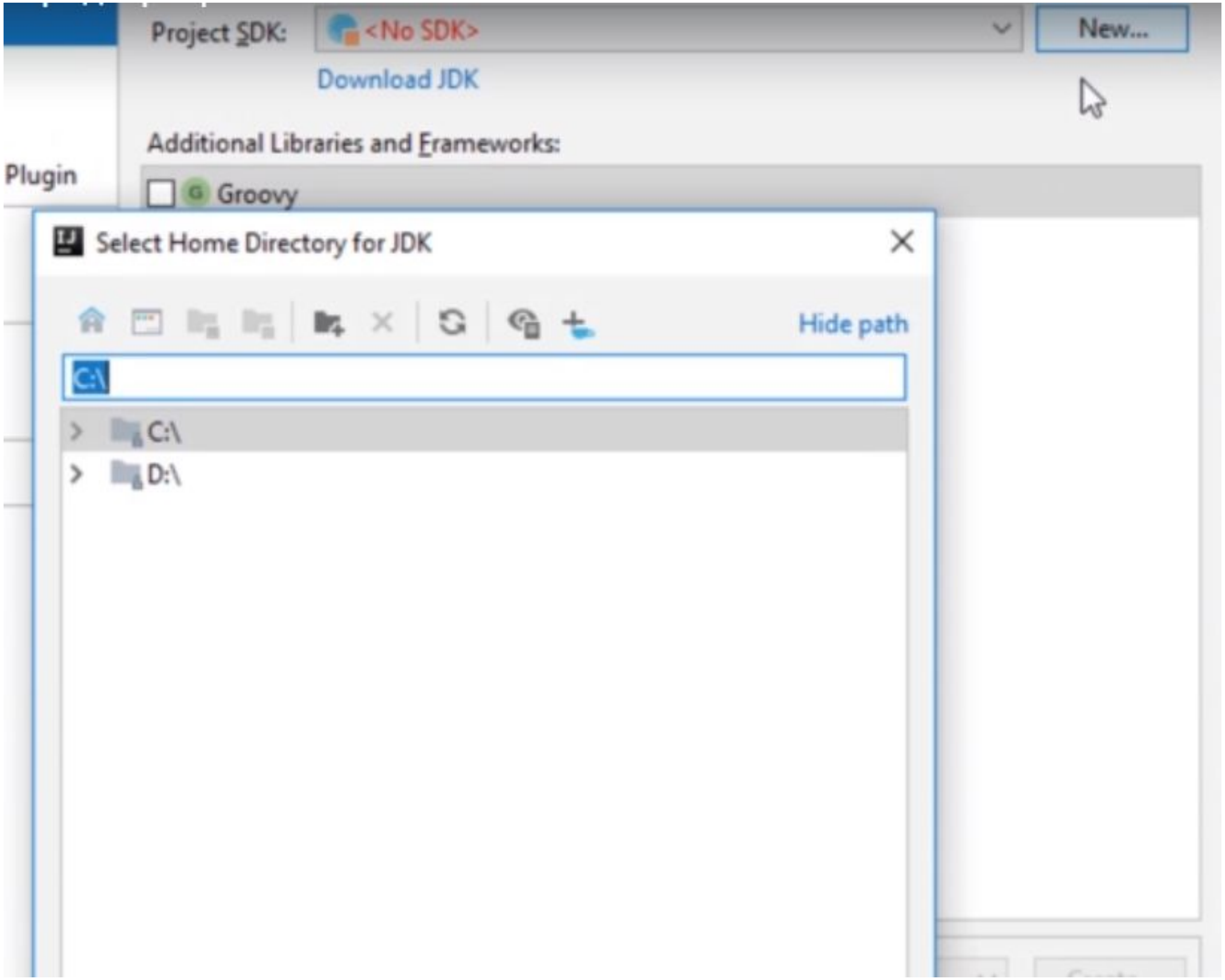
Java SE Development Kit 11.0.1

You must accept the Oracle Technology Network License Agreement for Oracle Java SE to download this software.

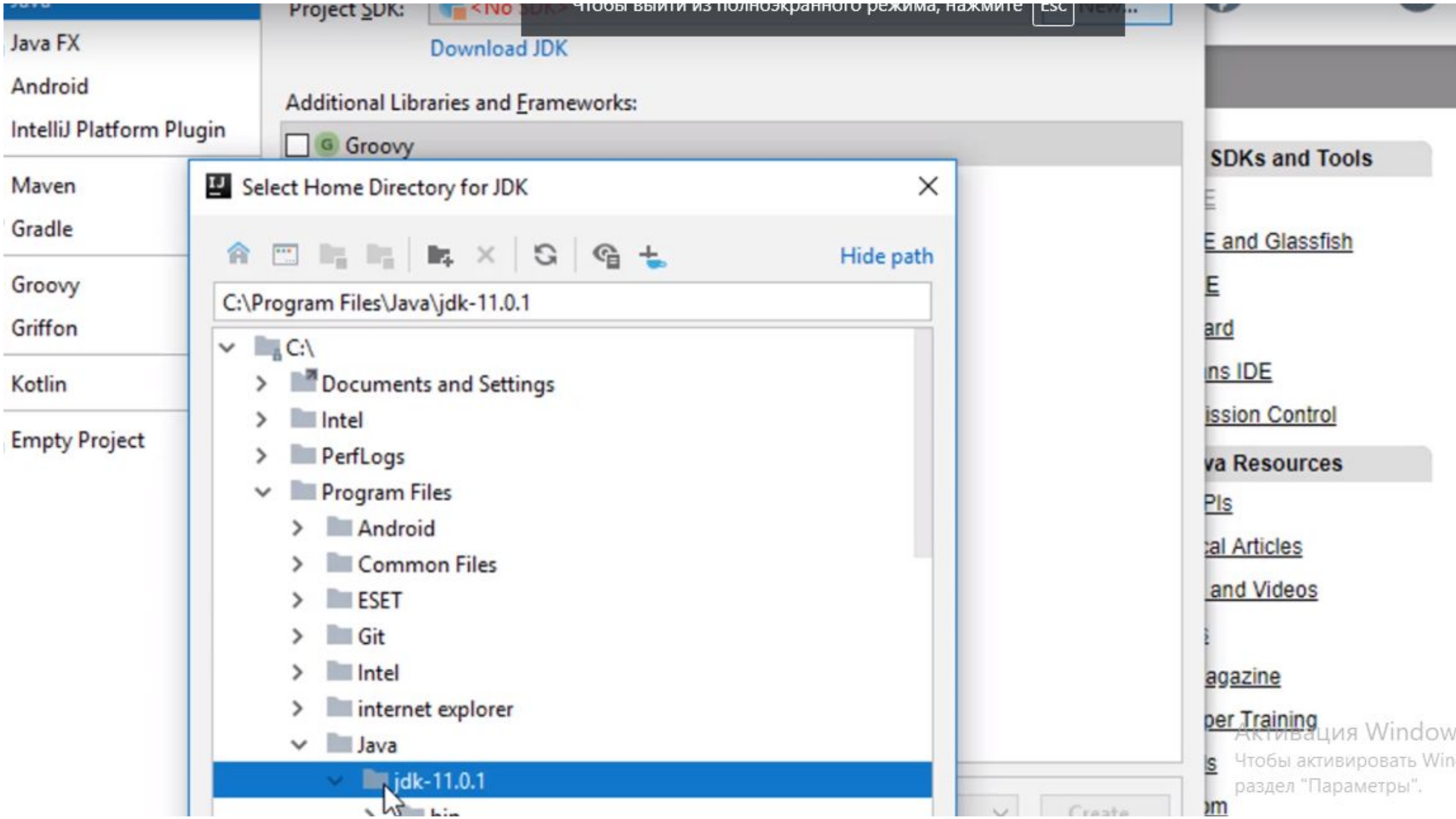
Thank you for accepting the Oracle Technology Network License Agreement for Oracle Java SE; you may now download this software.

Product / File Description	File Size	Download
Linux	147.4 MB	jdk-11.0.1_linux-x64_bin.deb
Linux	154.09 MB	jdk-11.0.1_linux-x64_bin.rpm
Linux	171.43 MB	jdk-11.0.1_linux-x64_bin.tar.gz
macOS	166.2 MB	jdk-11.0.1_osx-x64_bin.dmg
macOS	166.55 MB	jdk-11.0.1_osx-x64_bin.tar.gz
Solaris SPARC	186.8 MB	jdk-11.0.1_solaris-sparcv9_bin.tar.gz
Windows	150.98 MB	jdk-11.0.1_windows-x64_bin.exe
Windows	170.99 MB	jdk-11.0.1_windows-x64_bin.zip

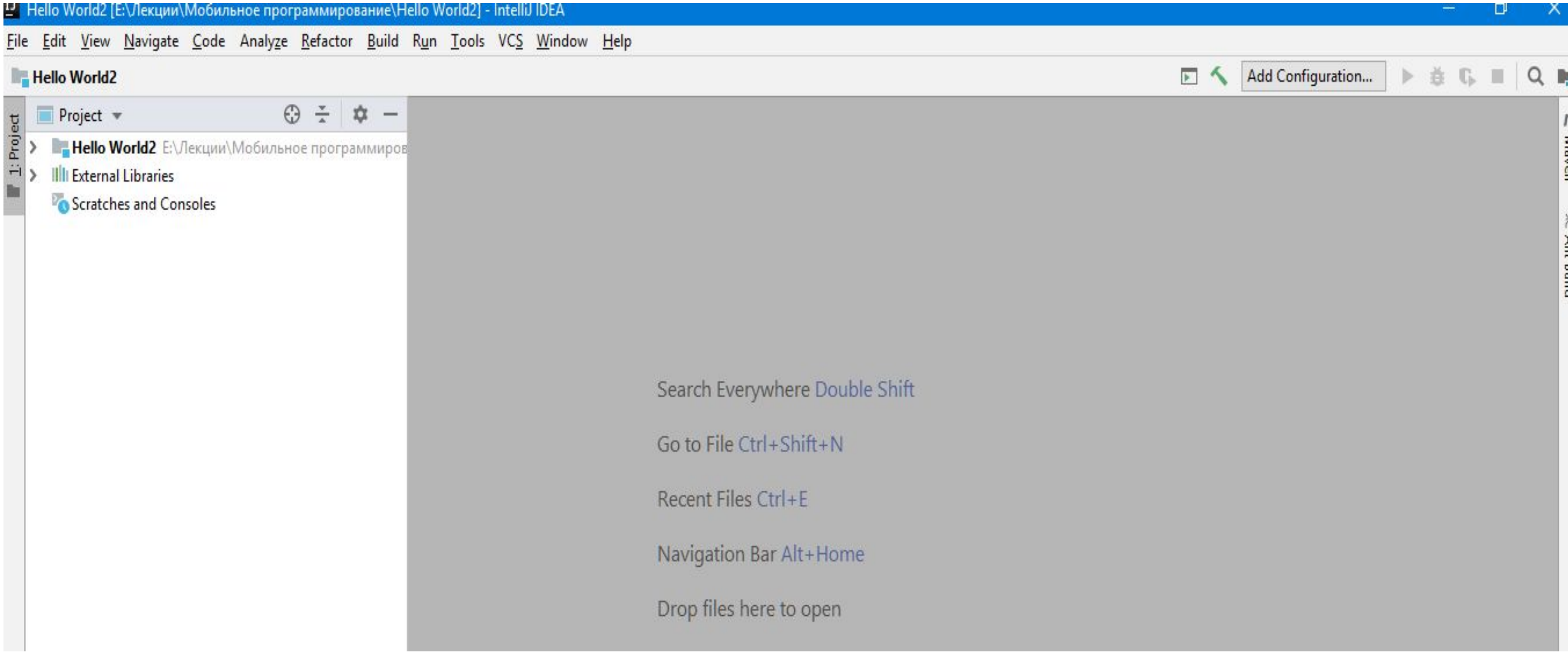
IntelliJ IDEA



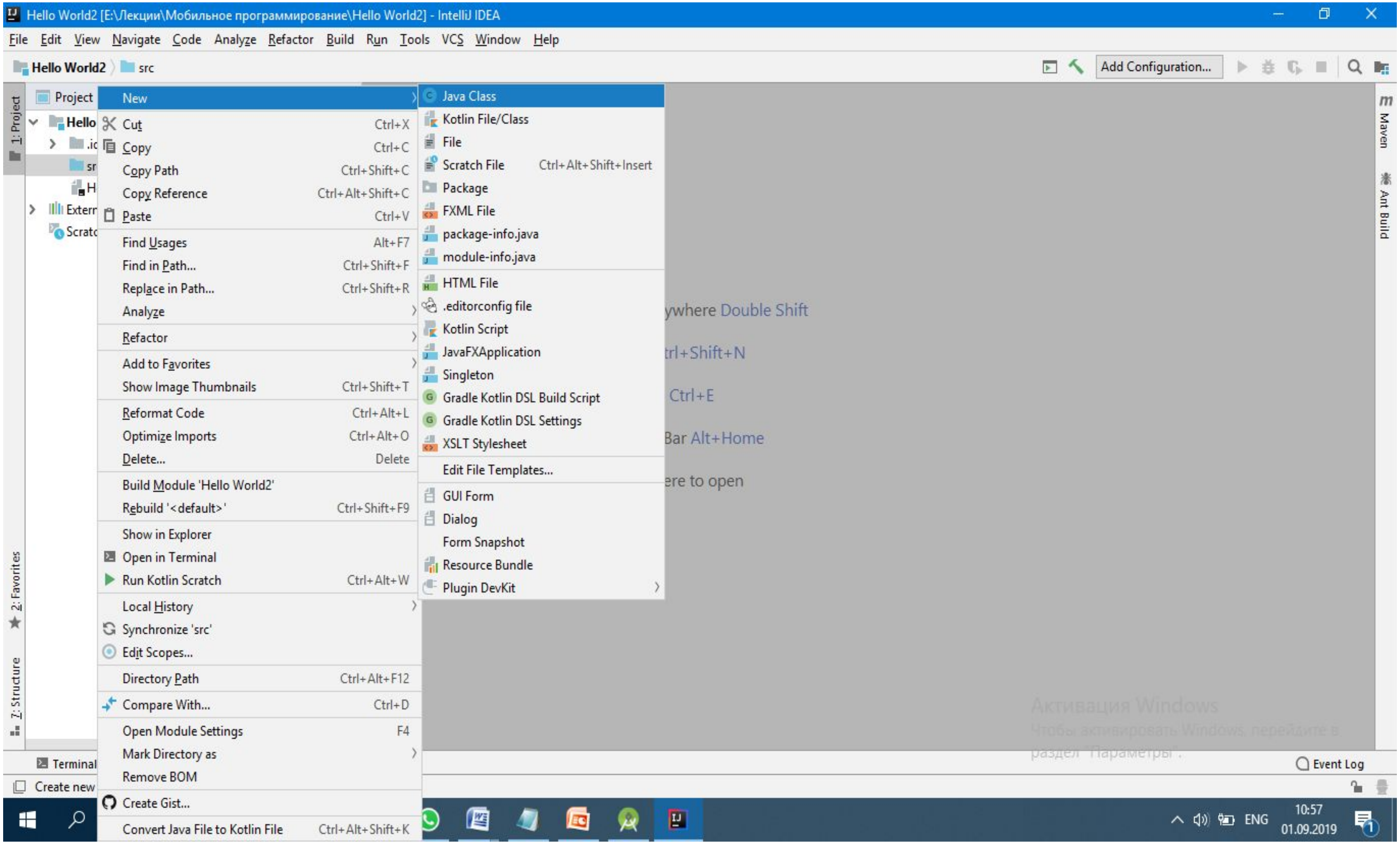
IntelliJ IDEA



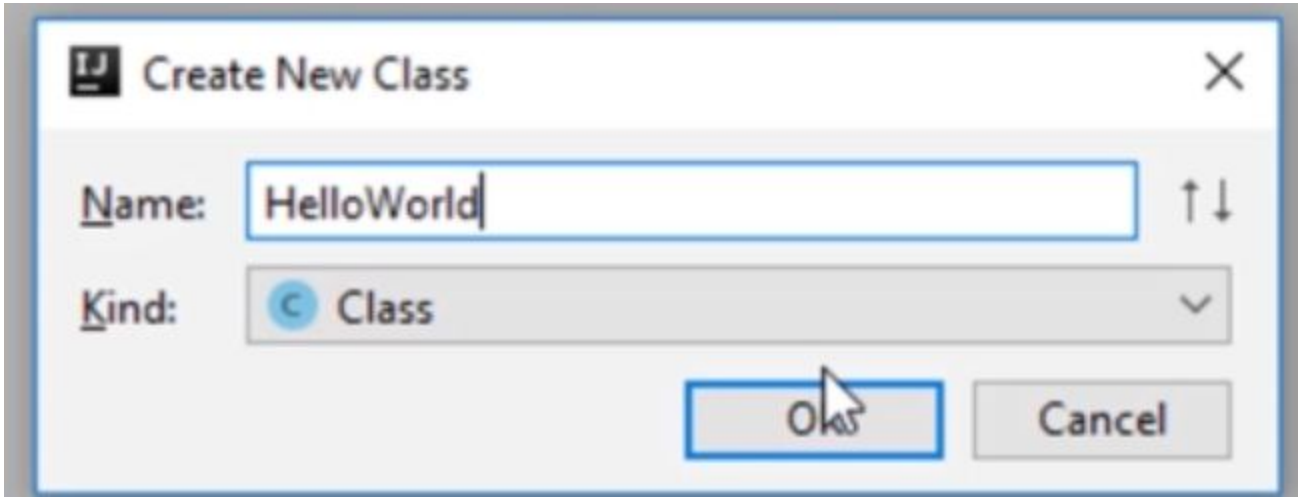
Первый проект в Intell IDEA



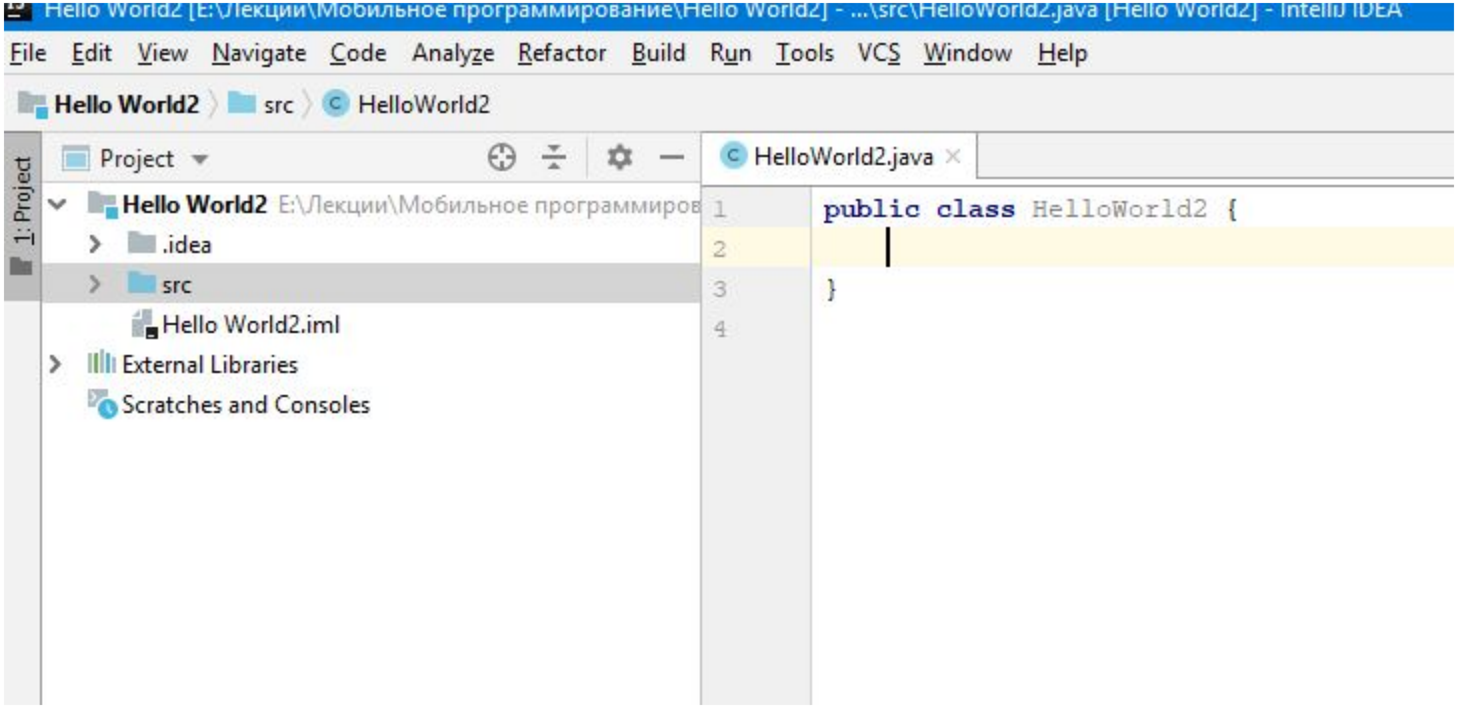
Первый проект в Intell IDEA



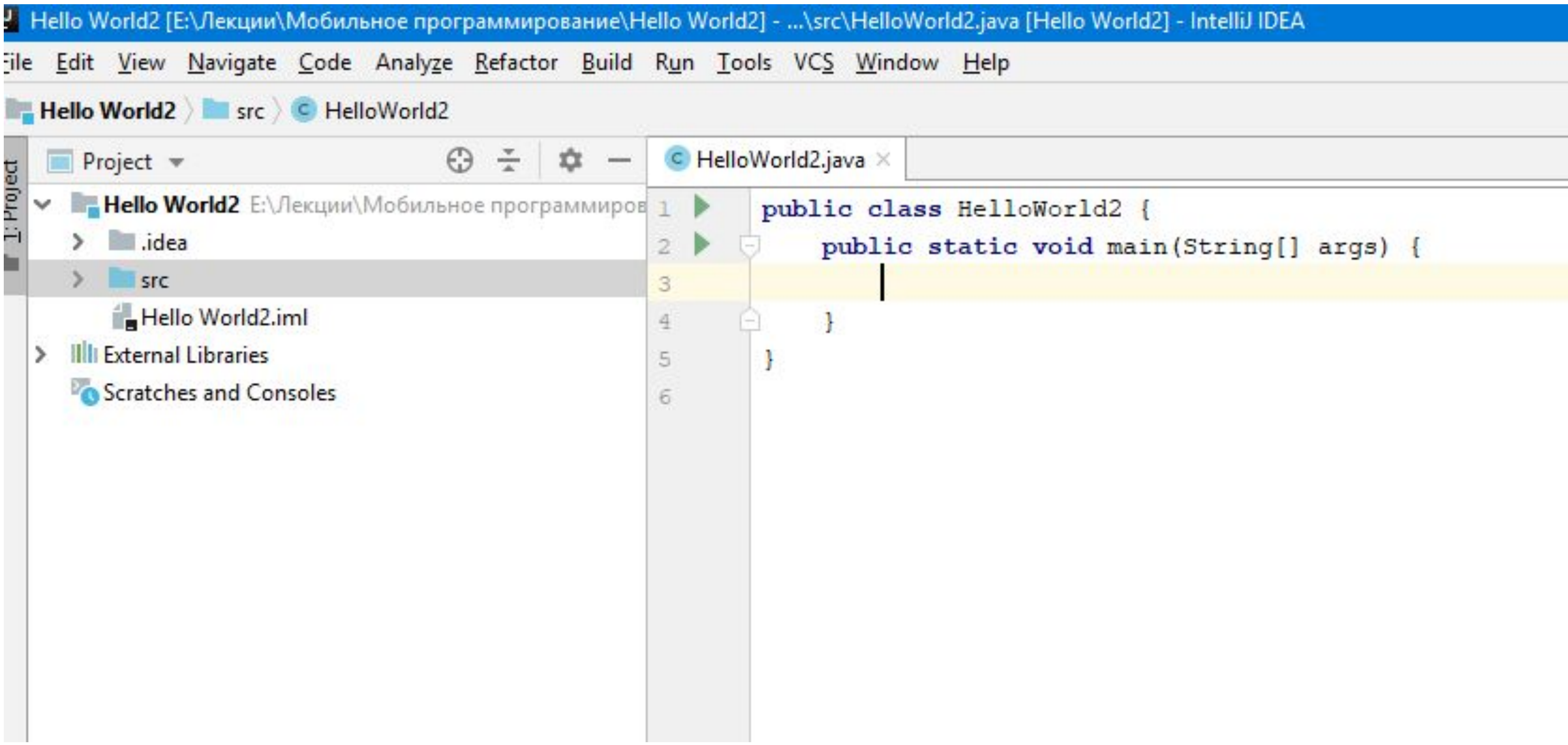
Разделы курса



Первый проект в Intell IDEA



Первый проект в Intell IDEA



Первый проект в Intell IDEA

Весь код на Java пишется внутри классов { между фигурными скобками; }

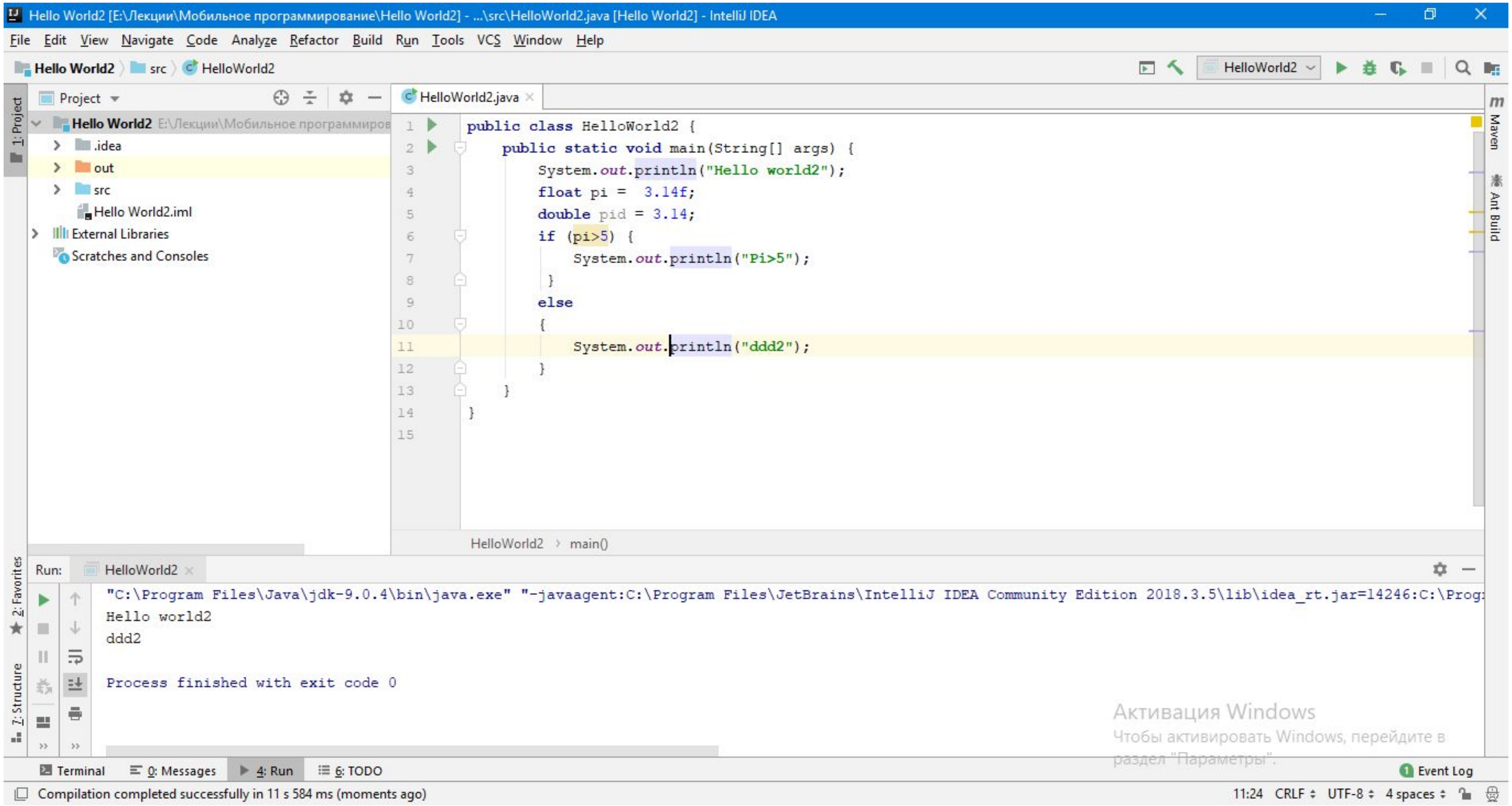
Точка входа в программу:
public static void main(String... args){}

Имена классов с заглавной буквы (HellowWorld)

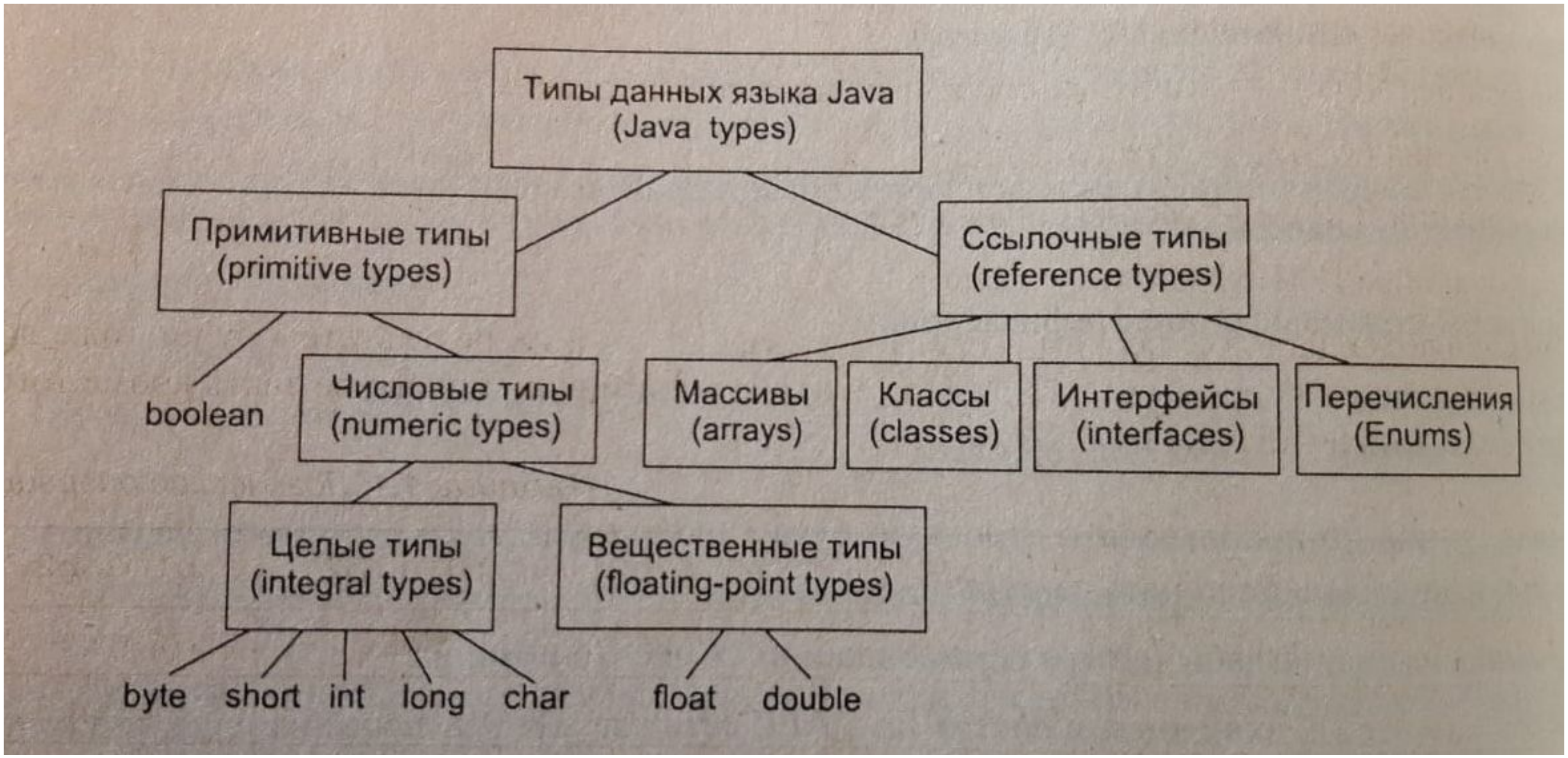
Имена методов с прописной буквы (main())

Вывести строку в консоль System.out.println("Строка");

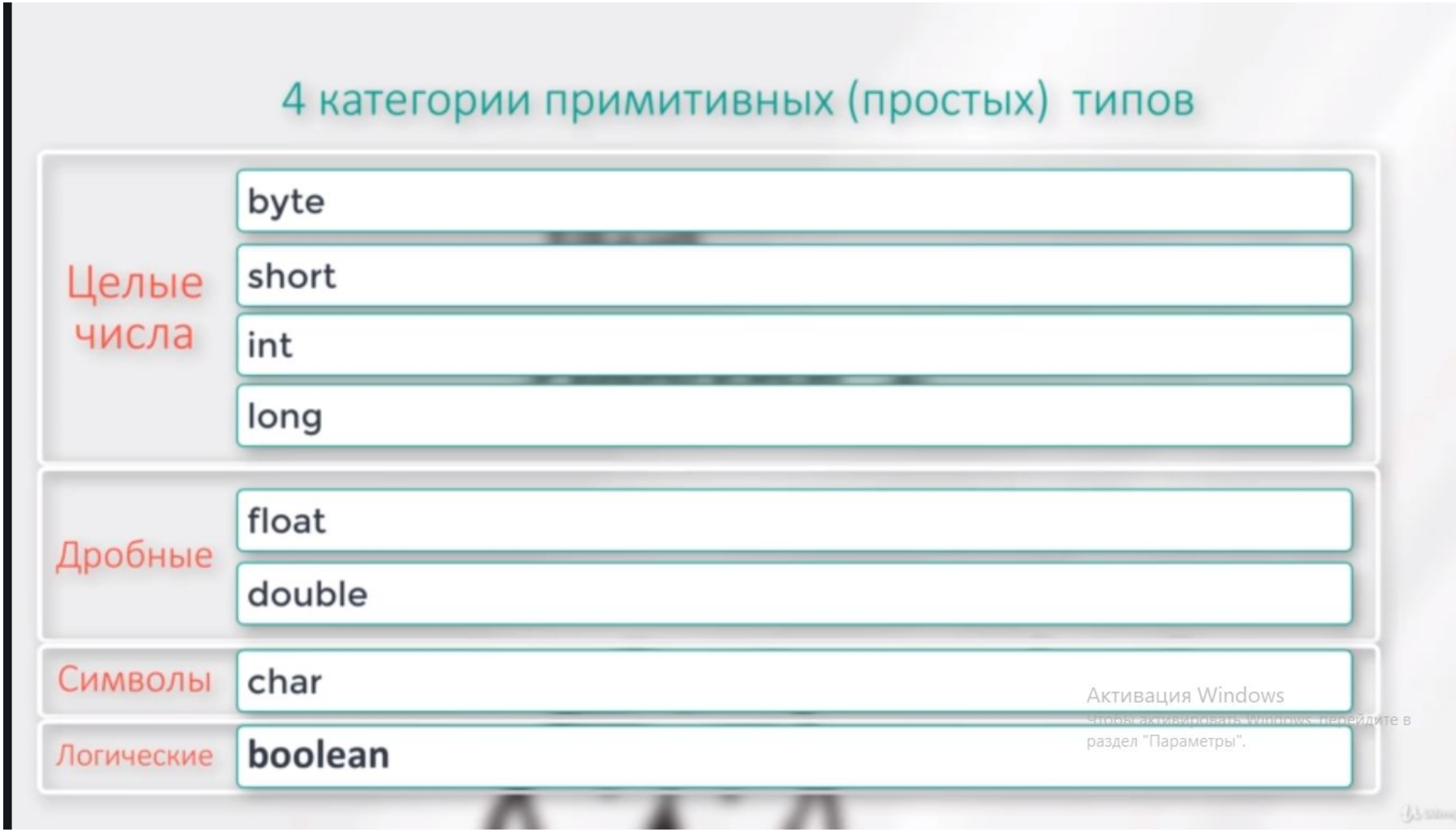
Условный оператор



Типы данных


Типы данных языка Java
(Java types)
Примитивные типы
(primitive types)
Ссылочные типы
(reference types)
boolean
Числовые типы
(numeric types)
Массивы
(arrays)
Классы
(classes)
Интерфейсы
(interfaces)
Перечисления
(Enums)
Целые типы
(integral types)
Вещественные типы
(floating-point types)
byte
short
int
long
char
float
double

Типы целых чисел



Типы целых чисел

4 категории примитивных (простых) типов

Целые числа

byte	**1 байт**	**от -128 до 127**
short	**2 байта**	**от -32768 до 32767**
int	**4 байта**	**от -2147483648 до 2147483647**
long	**8 байт**	**от -9223372036854775808 до 9223372036854775807**

byte number; short number = 3500; int number = 950000; long number = 999999999999;

number = 120;

Задача. Дано количество дней 1000, скорость света равна 300 000 км/сек, какое расстояние пройдет свет за это время?

Условный оператор

If (условие)
{
тело условия
}
Elseif …
Else ..

&& - логическое И
|| - логическое или

Циклы

Циклы с фиксированным количеством итераций

```
for (int i =0; i<10; i++)
{

}
```

While (лог выр) оператор

do оператор while (лог выражение)

```
for (int i:localArr)
{
Тело цикла
}
```

Циклы

Вывод числа от 1 до 1000.

```
public class Loop {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 1000) {
            System.out.println(i);
            i++;
        }
    }
}
```

Цикл с фиксированным количеством итераций

```
public class Loop {
    public static void main(String[] args) {
        for(int i = 0; i < 100; i++) {
            System.out.println(i);
        }
    }
}
```

Цикл с пост условием.

```
public class Loop {
    public static void main(String[] args) {
        int a = 0;
        do {
            System.out.println("Привет");
        } while (a > 0);
    }
}
```

break – прерывание цикла

Задача на циклы

Вывести все четырехзначные числа типа ABCD, где: A + B = C + D

```
for (int a = 1; a <= 9; a++)
    {
      for (int b = 0; b <= 9; b++)
      {
        for (int c = 0; c <= 9; c++)
        {
          for (int d = 0; d <= 9; d++)
          {
            if ((a + b) == (c + d))
            {
              System.out.println(
                  " " + a + " " + b + " " + c + " " + d);
            }
          }
        }
      }
    }
  }
```

Задача на циклы

Вывести все четырехзначные числа типа ABCD, где: A + B = C + D

```
import java.util.Random;
import java.util.Scanner;

public class LectionTwo {
    public static void main(String[] args) {


        Scanner keyboard = new Scanner(System.in);
        int randomNumber = new Random().nextInt(10) + 1;
        while (true) {
            System.out.println("Введите число");
            int number = keyboard.nextInt();
            if (number < randomNumber) System.out.println("<");
            if (number > randomNumber) System.out.println(">");
            if (number == randomNumber) {
                System.out.println("Pobeda!");
                break;
            }

        }
    }
}
```

Циклы

a++ инкремент => a = a + 1

a-- декремент => a = a -1

a +=5 => a = a + 5

a -=5 => a = a -5

Цикл while (условие)

```
public class Loop {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 1000) {
            System.out.println(i);
            i++;
        }
    }
}
```

МАССИВЫ

№	Объявление массива, Java-синтаксис	Примеры	Комментарий

1.	`dataType[] arrayName;`	`int[] myArray;`
		
		`Object[] arrayOfObjects;`
			Желательно объявлять массив именно таким способом, это Java-стиль

2.	`dataType arrayName[];`	`int myArray[];`
		
		`Object arrayOfObjects[];`
			Унаследованный от C/C++ способ объявления массивов, который работает и в Java

Как и любой другой объект, создать массив Java, то есть зарезервировать под него место в памяти, можно с помощью оператора `new`. Делается это так:

```
new typeOfArray [length];
```

Где `typeOfArray` — это тип массива, а `length` — его длина (то есть, количество ячеек), выраженная в целых числах (`int`). Однако здесь мы только выделили память под массив, но не связали созданный массив ни с какой объявленной ранее переменной. Обычно массив сначала объявляют, а потом создают, например:

```
int[] myArray; // объявление массива
myArray = new int[10]; // создание, то есть, выделение памяти для массива на 10 элементов типа int
```

МАССИВЫ

int[] myArray = new int[10];
получаем массив из десяти целых чисел, и, пока это не измениться в ходе программы, в каждой ячейке записан 0.
массив с данными ссылочного типа, то по умолчанию в каждой ячейке записаны null

```
String[] seasons  = new String[4]; /* объявили и создали массив. Java выделила память под массив из 4

seasons[0] = "Winter"; /* в первую ячейку, то есть, в ячейку с нулевым номером мы записали строку Wint
seasons[1] = "Spring"; // проделываем ту же процедуру с ячейкой номер 1 (второй)
seasons[2] = "Summer"; // ...номер 2
seasons[3] = "Autumn"; // и с последней, номер 3
```

Теперь во всех четырёх ячейках нашего массива записаны названия сезонов. Инициализацию также можно провести по-другому, совместив с инициализацией и объявлением:

```
String[] seasons  = new String[] {"Winter", "Spring", "Summer", "Autumn"};
```

Более того, оператор `new` можно опустить:

```
String[] seasons  = {"Winter", "Spring", "Summer", "Autumn"};
```

Массивы

Тип данных [] arr = new Тип данных [размерность]

int[] a = new int[50]

или

Int[ ] arr = { 1,2,3 }

Двумерные:
int [ ] [ ] d = new int [3][4]
int [ ] [ ] arr2 = { {1,2,3}, {4,5,6} }
Элементами массива – являются массивы

МНОГОМЕРНЫЕ МАССИВЫ

Многомерный массив объявляется и создается следующим образом:

```
Int[][] myTwoDimentionalArray = new int [8][8];
```

Для работы с массивами в Java есть класс java.util.Arrays (arrays на английском и означает “массивы”). В целом с массивами чаще всего проделывают следующие операции: заполнение элементами (инициализация), извлечение элемента (по номеру), сортировка и поиск.

```
==
!=

if (!str1.equals(“x”))
{

}
```

Циклы

```
Scanner keyboard = new Scanner(System.in);
    int randomNumber = new Random().nextInt(10) + 1;
    while (true) {
      System.out.println("Введите число");
      int number = keyboard.nextInt();
      if (number < randomNumber) System.out.println("<");
      if (number > randomNumber) System.out.println(">");
      if (number == randomNumber) {
        System.out.println("Pobeda!");
        break;
      }
```

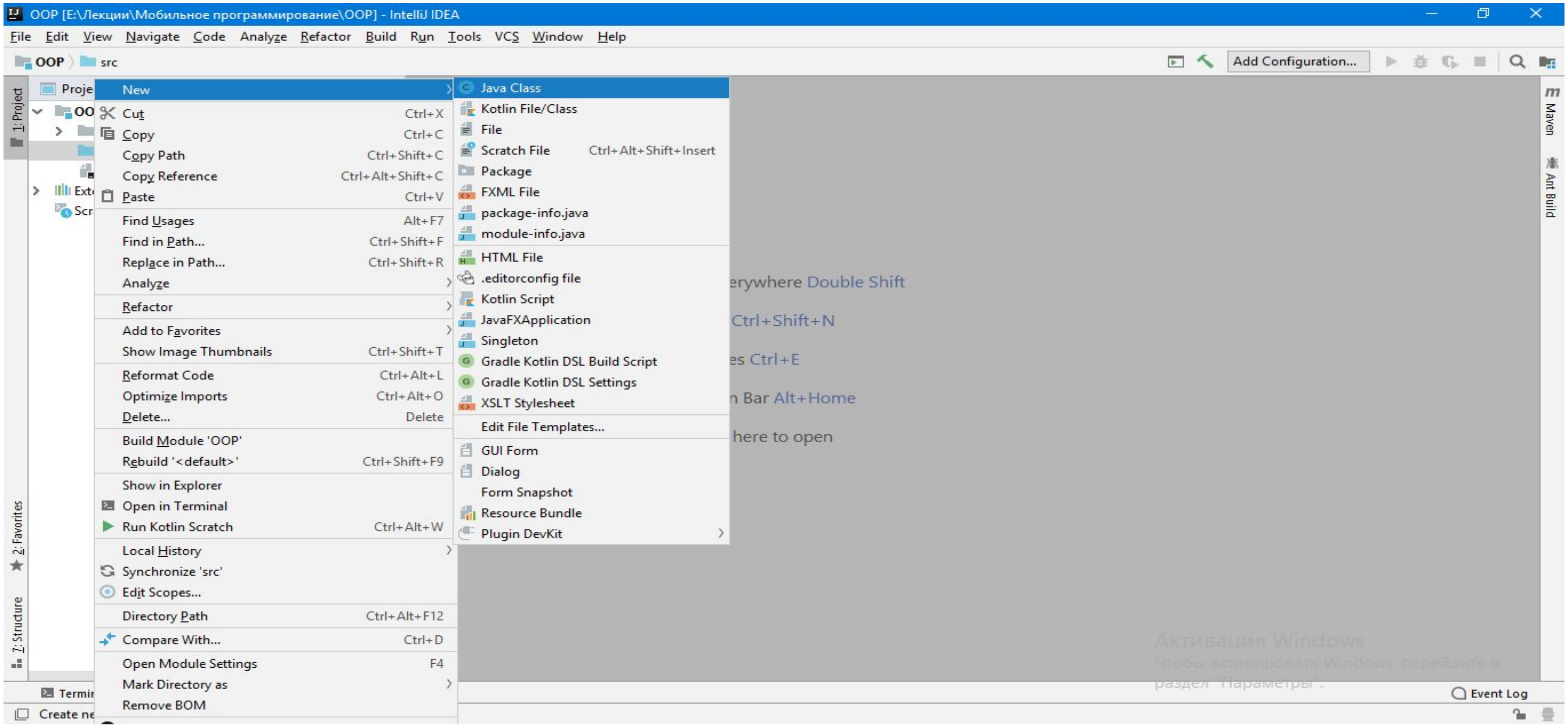
Классы

- *Классы* позволяют проводить конструирование из полезных компонентов, обладающих простыми инструментами, что позволяет абстрагироваться от деталей реализации.
- Данные и операции над ними образуют определенную сущность, и они не разносятся по всей программе, как нередко бывает в случае процедурного программирования, а описываются вместе. Локализация кода и данных улучшает наглядность и удобство сопровождения программного обеспечения.
- *Инкапсуляция* позволяет привнести свойство *модульности*, что облегчает распараллеливание выполнения задачи между несколькими исполнителями и обновление версий отдельных компонентов.

Полиморфизм оказывается полезным преимущественно в следующих ситуациях.

- Обработка разнородных структур данных. Программы могут работать, не различая вида *объектов*, что существенно упрощает код. Новые виды могут быть добавлены в любой момент.
- Изменение *поведения* во время исполнения. На этапе исполнения один *объект* может быть заменен другим, что позволяет легко, без изменения кода, адаптировать алгоритм в зависимости от того, какой используется *объект*.
- Реализация работы с наследниками. Алгоритмы можно обобщить настолько, что они уже смогут работать более чем с одним видом *объектов*.
- Создание "каркаса" (framework). Независимые от приложения части предметной области могут быть реализованы в виде набора универсальных *классов*, или каркаса (framework), и в дальнейшем расширены за счет добавления частей, специфичных для конкретного приложения.

Классы


OOP [E:\Лекции\Мобильное программирование\OOP] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
OOP src
Add Configuration...
New
Cut Ctrl+X
Copy Ctrl+C
Copy Path Ctrl+Shift+C
Copy Reference Ctrl+Alt+Shift+C
Paste Ctrl+V
Find Usages Alt+F7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Show Image Thumbnails Ctrl+Shift+T
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Delete... Delete
Build Module 'OOP'
Rebuild '<default>' Ctrl+Shift+F9
Show in Explorer
Open in Terminal
Run Kotlin Scratch Ctrl+Alt+W
Local History
Synchronize 'src'
Edit Scopes...
Directory Path Ctrl+Alt+F12
Compare With... Ctrl+D
Open Module Settings F4
Mark Directory as
Remove BOM
Java Class
Kotlin File/Class
File
Scratch File Ctrl+Alt+Shift+Insert
Package
FXML File
package-info.java
module-info.java
HTML File
.editorconfig file
Kotlin Script
JavaFXApplication
Singleton
Gradle Kotlin DSL Build Script
Gradle Kotlin DSL Settings
XSLT Stylesheet
Edit File Templates...
GUI Form
Dialog
Form Snapshot
Resource Bundle
Plugin DevKit
erywhere Double Shift
Ctrl+Shift+N
es Ctrl+E
n Bar Alt+Home
here to open
Event Log

Классы

```
class ИмяКласса {

    int переменная1;
    String переменная2;
    ...

    void метод1();
    int метод2();
    ...

}
```

Main.java Box.java

```java
public class Box {
    double width;
    double height;
    double length;
}
```

```java
public class Main {
    public static void main(String[] args) {
        Box myBox = new Box();
        myBox.height = 55;
        Box box4 = myBox;

    }
}
```

void - не возвращает никаких данных

```
тип имяМетода(список_параметров) {
(int, boolean, Box...)        тело_метода
                                 ...
                              }
```

Конструктор

```java
public class Box {
    double width;
    double height;
    double length;

    Box() {
        width = 10;
        height = 10;
        length = 10;
    }
```

Модификаторы доступа

По умолчанию все модификаторы доступа public.
'Generate...' and then 'Getter and Setter'.
Default – это доступ по умолчанию , доступность внутри пакета

```java
public class Box {
    private double width;
    double height;
    double length;

    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        this.width = width;
    }
}
```

public | private | protected | default

Пакеты

Элементами *пакета* являются содержащиеся в нем классы и интерфейсы, а также вложенные *пакеты*. Чтобы получить *составное имя пакета*, необходимо к полному имени *пакета*, в котором он располагается, добавить точку, а затем его собственное *простое имя*. Например, *составное имя* основного *пакета* языка Java – `java.lang` (то есть *простое имя* этого пакета `lang`, и он находится в объемлющем пакете `java`). Внутри него есть вложенный пакет, предназначенный для *типов* технологии reflection, которая упоминалась в предыдущих главах. Простое название пакета `reflect`, а значит, составное – `java.lang.reflect`.

Простое имя классов и интерфейсов дается при объявлении, например, `Object`, `String`, `Point`. Чтобы получить *составное имя* таких *типов*, надо к *составному имени пакета*, в котором находится *тип*, через точку добавить *простое имя типа*. Например, `java.lang.Object`, `java.lang.reflect.Method` или `com.myfirm.MainClass`. Смысл последнего выражения таков: сначала идет обращение к *пакету* com, затем к его *элементу* – вложенному *пакету* `myfirm`, а затем к *элементу пакета* `myfirm` – классу `MainClass`. Здесь `com.myfirm` – *составное имя пакета*, где лежит класс `MainClass`, а `MainClass` — *простое имя*. Составляем их и разделяем точкой – получается полное имя класса `com.myfirm.MainClass`.

Пакеты

Программа на *Java* представляет собой набор *пакетов* (packages). Каждый *пакет* может включать вложенные *пакеты*, то есть они образуют иерархическую систему.

Кроме того, *пакеты* могут содержать классы и интерфейсы и таким образом группируют *типы*. Это необходимо сразу для нескольких целей. Во-первых, чисто физически невозможно работать с большим количеством классов, если они "свалены в кучу". Во-вторых, модульная *декомпозиция* облегчает проектирование системы. К тому же, как будет показано ниже, существует специальный уровень доступа, позволяющий *типам* из одного *пакета* более тесно взаимодействовать друг с другом, чем с классами из других *пакетов*. Таким образом, с помощью *пакетов* производится логическая группировка *типов*. Из *ООП* известно, что большая *связность* системы, то есть среднее количество классов, с которыми взаимодействует каждый *класс*, заметно усложняет развитие и поддержку такой системы. Используя *пакеты*, гораздо проще организовать эффективное взаимодействие подсистем друг с другом.

Наконец, каждый *пакет* имеет свое *пространство* имен, что позволяет создавать одноименные классы в различных *пакетах*. Таким образом, разработчикам не приходится тратить время на разрешение конфликта имен.

Пакеты

Исходный код располагается в файлах с расширением .java, а бинарный – с расширением .class

исходный код классов

```
space.sunsystem.Moon
space.sunsystem.Sun
space.sunsystem.Test
```

хранится в файлах

```
space\sunsystem\Moon.java
space\sunsystem\Sun.java
space\sunsystem\Test.java
```

Используйте == для сравнения два примитива, или посмотреть,
если два ссылки относятся к тот же объект.
Используйте equals чтобы увидеть если два объекта равны.

```
Integer.parseInt("3")
```

```
for (int cell : locationCells)
```

Не принято, чтобы классы находились не внутри пакетов

Пакеты

Package – указывает в каком пакете находится данный класс

```
package Main;


import box.Box;

public class Main {
    public static void main(String[] args) {
        Box myBox = new Box();


        Box box4 = myBox;

    }
}
```

Отношения между классами

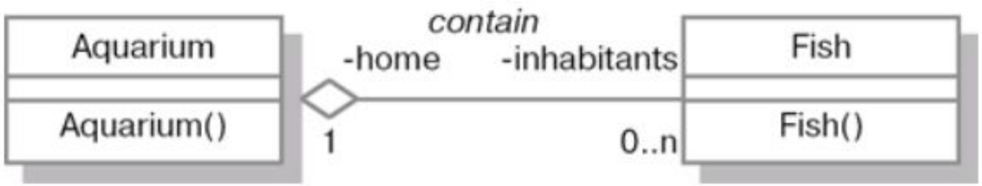
агрегация (Aggregation);
ассоциация (Association);
наследование (Inheritance);
метаклассы (Metaclass).

Приняты также обозначения:

- " 1..n " - от единицы до бесконечности;
- " 0 " - ноль;
- " 1 " - один;
- " n " - фиксированное количество;
- " 0..1 " - ноль или один.

Агрегация

Отношение между *классами* типа "содержит" (contain) или "состоит из" называется агрегацией, или включением. Например, если аквариум наполнен водой и в нем плавают рыбки, то можно сказать, что аквариум агрегирует в себе воду и рыбок.




```
// определение класса Fish
public class Fish {
   // определения поля home
   // (ссылка на объект Aquarium)
   private Aquarium home;

   public Fish() {
   }
}
```

```
// определение класса Aquarium
public class Aquarium {
   // определения поля inhabitants
   // (массив ссылок на объекты Fish)
   private Fish inhabitants[];
   public Aquarium() {
   }
}
```

Задача

Найти наименьшее число имеющие n десятичных разрядов, кратное числам 2,3,5,7.
Указание n – может быть очень большим, например, n = 1000.

10^k = 10*10*….*10

(10^k) mod p

(A*B) mod p = (A mod p) * (B mod p)
5 * 7 mod 2 = 1 = 1 * 1 = 1