Программирование
Тема 2.1 Введение в Java

- Java – это язык программирования и платформа (1995, Sun Microsystems, Джеймс Гослинг)
- Платфома Java:
  - Виртуальная машина Java (JVM);
  - Стандартная библиотека классов.
- Варианты распространения платформы Java:
  - JRE (Java Runtime Environment);
  - JDK (Java Development Kit).
- Версии JDK:
  - Java SE (Java Standard Edition);
  - Java EE (Java Enterprise Edition);
  - Java ME (Java Micro Edition).

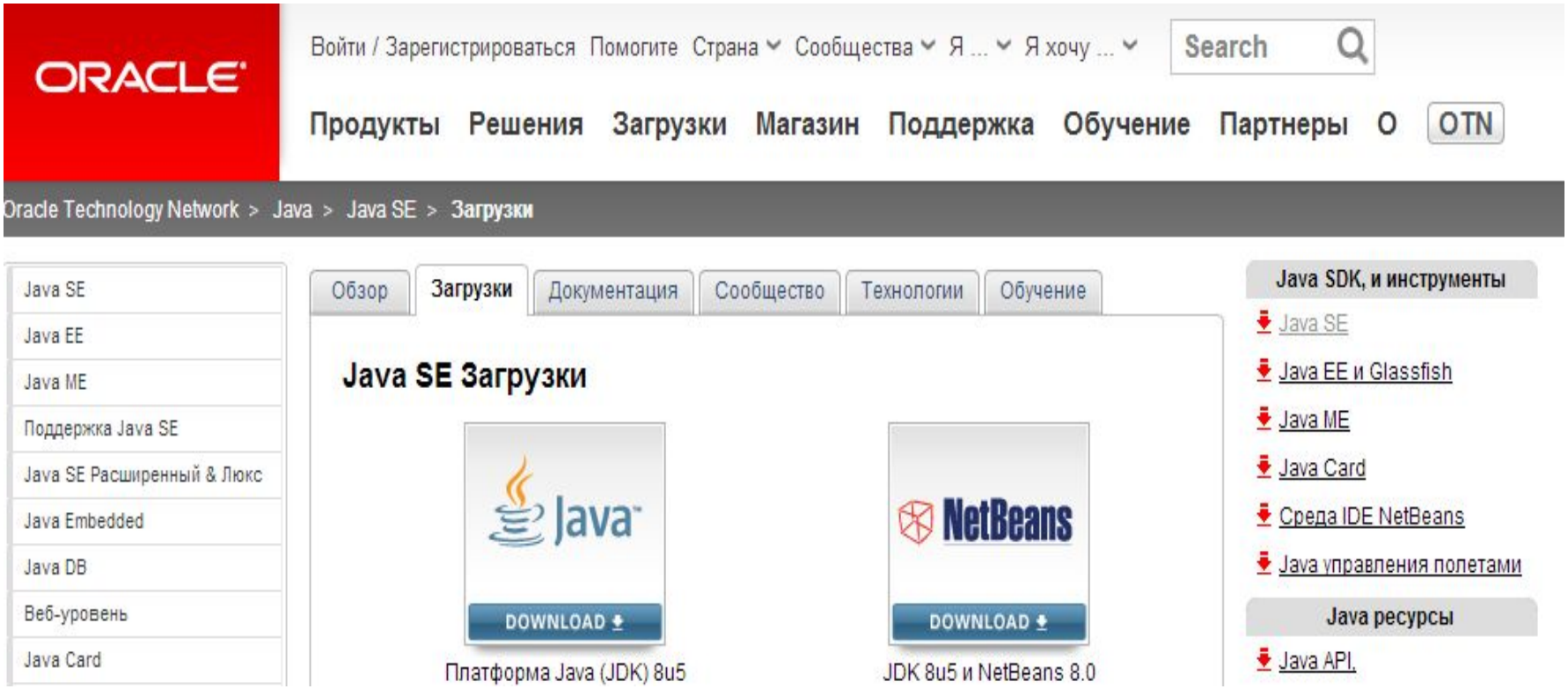
Платформа Java

- Платформа – это совокупность аппаратного и программного (ОС) обеспечения (MS Windows, Linux, Solaris OS, Mac OS).
- Платфома Java – это программное обеспечение, которое состоит из двух основных частей:
  - Виртуальная машина Java (JVM);
  - Стандартная библиотека классов.
- Платформа Java распространяется в двух вариантах – JRE (Java Runtime Environment) и JDK(Java Development Kit).
- JRE (среда исполнения Java) - это программное обеспечение, необходимая для исполнения Java-приложений, без компилятора и других средств разработки. Если пользователь хочет только запускать программы, это именно то, что ему нужно. JRE состоит из JVM, стандартных классов Java и вспомогательных файлов.
- JDK - средство разработчика, состоящее из JRE, утилит командной строки (например, javac, java, javadoc и др.), исходных кодов классов стандартной библиотеки и вспомогательных файлов. JDK является базовым средством разработки приложений. Оно не содержит никаких текстовых редакторов, а оперирует только с уже существующими java-файлами с помощью утилит командной строки.
- Версии JDK:
  - Java SE – комплект Java Standard Edition – для написания программного обеспечения для обычных компьютеров;
  - Java EE– комплект Java Enterprise Edition – для написания серверного программного обеспечения;
  - Java ME – комплект Java Micro Edition – для написания программного обеспечения для устройств, обладающих малыми ресурсами, например, сотовых телефонов, микроконтроллеров;

Платформа Java

- **Ссылка для скачивания JRE:**
  http://java.com/ru/download/index.jsp

- **Ссылка для скачивания JDK:**
  http://www.oracle.com/technetwork/java/javase/downloads/



Платформа Java

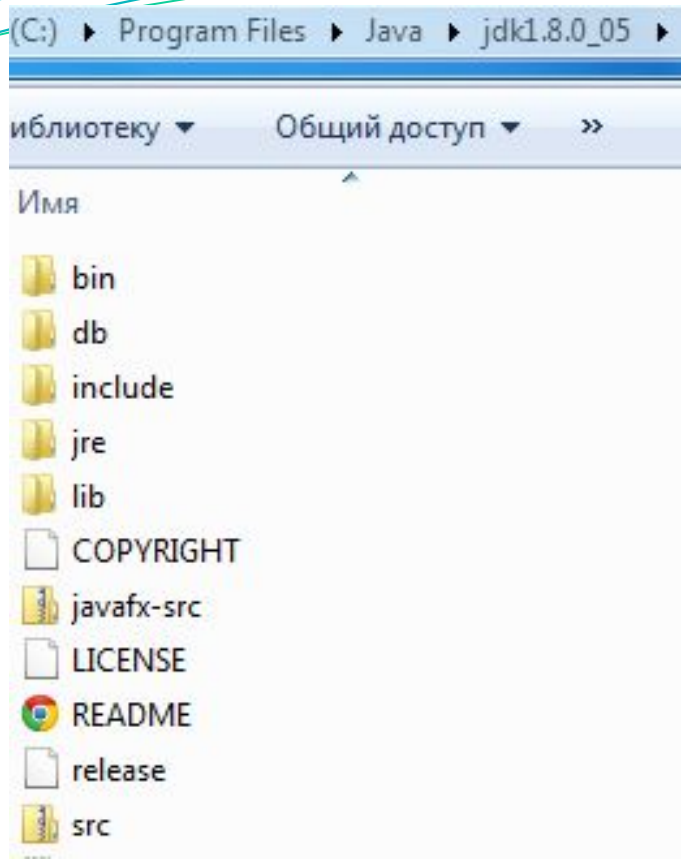
Java SE Development Kit 8u5

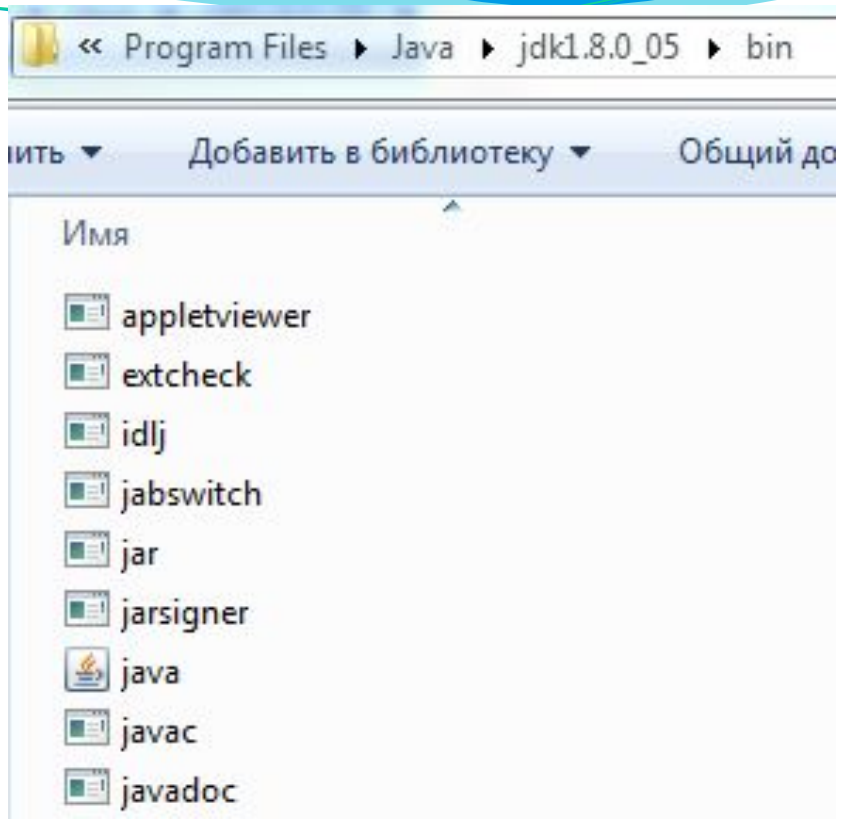
Вы должны принять лицензионное соглашение для Java SE Oracle двоичный код , чтобы загрузить это программное обеспечение.

Спасибо, что приняли лицензионное соглашение Oracle двоичного кода для Java SE; Вы можете сейчас скачать это программное обеспечение.

Продукт / файла Описание	Размер файла	Скачать
Linux x86	133,58 МБ	JDK-8u5-Linux-i586.rpm
Linux x86	152,5 MB	JDK-8u5-Linux-i586.tar.gz
Linux x64	133,87 МБ	JDK-8u5-Linux-x64.rpm
Linux x64	151,64 МБ	JDK-8u5-Linux-x64.tar.gz
Mac OS X x64	207,79 МБ	JDK-8u5-MacOSX-x64.dmg
Solaris SPARC 64-разрядный (SVR4 пакет)	135,68 МБ	JDK-8u5-Solaris-sparcv9.tar.Z
Solaris SPARC 64-бит	95,54 Мб	JDK-8u5-Solaris-sparcv9.tar.gz
Solaris x64 (SVR4 пакет)	135,9 MB	JDK-8u5-Solaris-x64.tar.Z
Solaris x64	93,19 Мб	JDK-8u5-Solaris-x64.tar.gz
Windows x86	151,71 МБ	JDK-8u5-окна-i586.exe
Windows x64	155,18 МБ	JDK-8u5-окна-x64.exe

Структура Java Development Kit (JDK)





- **Проверка версии установленной Java – платформы:**

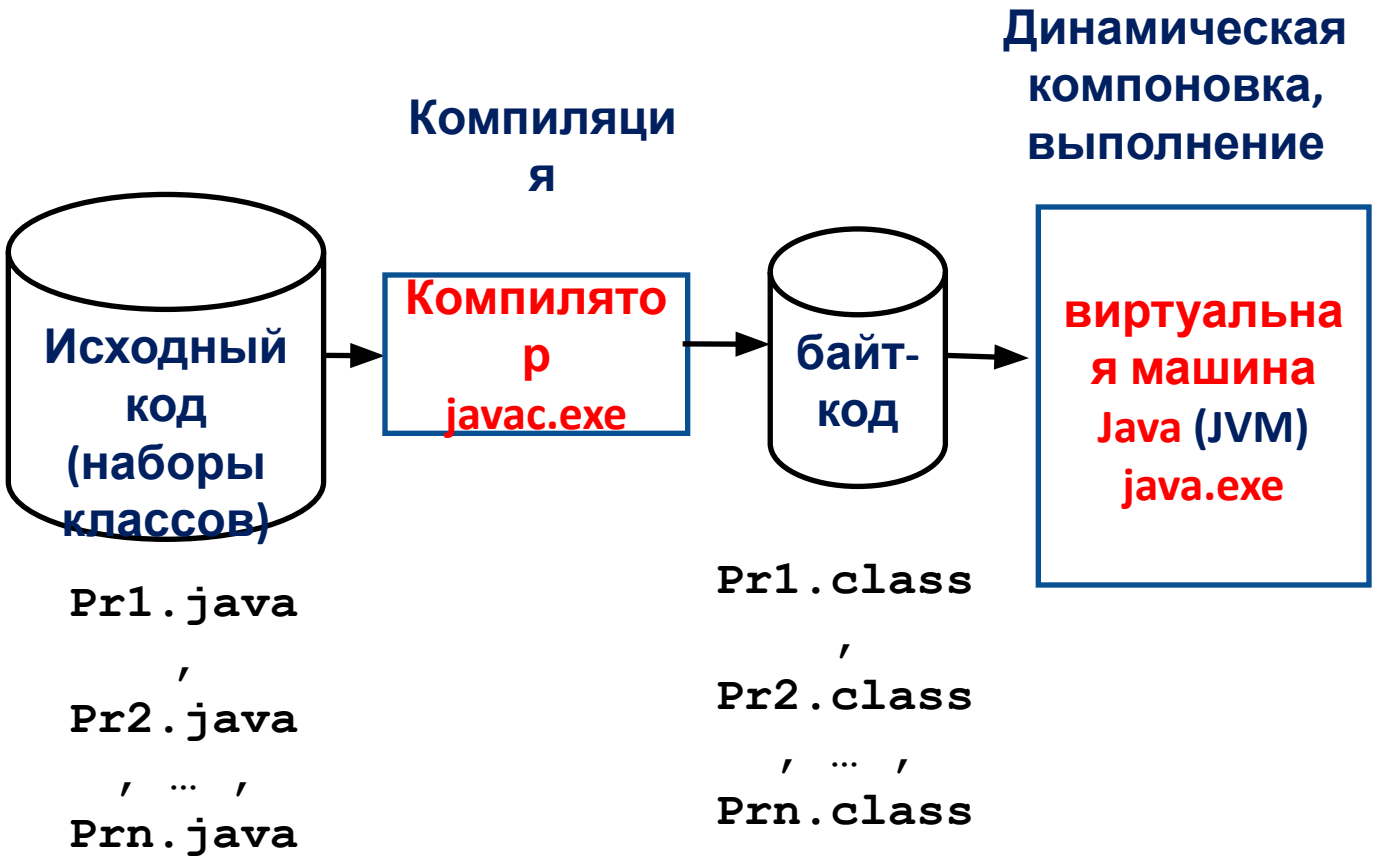
```
C:\Users\admin>java -version
java version "1.8.0_05"
Java(TM) SE Runtime Environment (build 1.8.0_05-b13)
Java HotSpot(TM) Client VM (build 25.5-b02, mixed mode)
```

- **Проверка доступности javac:**

```
C:\Users\admin>javac
Usage: javac <options> <source files>
where possible options include:
  -g                         Generate all debugging info
  -g:none                    Generate no debugging info
  -g:{lines,vars,source}     Generate only some debugging info
```

- **Если команда javac недоступна, то нужно добавить путь “ C:\Program Files\Java\jdk1.8.0_05\bin” к системной переменной PATH (Пуск->Панель управления->Система->Дополнительно -> Переменные среды)**

Структура, компиляция и выполнение Java-программы



Структура, компиляция и выполнение Java-программы

- Исходный код Java-программы – это текстовый файл, содержащий в себе одно или несколько описаний классов. Среди классов должен быть класс, содержащий метод main(). Именно с метода main() начинается выполнение программы. Текстовый файл с исходным кодом Java-программы должен иметь имя, совпадающее с именем класса, содержащим метод main() и расширение java.
- Хорошим стилем программирования является описание одного класса в одном java-файле, имя которого совпадает с именем этого класса.
- Исходные java-файлы компилируются с помощью компилятора javac.exe в файлы с расширением .class. Однако эти файлы не содержат машинный код, исполняемый непосредственно процессором, они содержат так называемый байт-код – код, обрабатываемый виртуальной машиной Java (JVM). Запуск программы в рамках JVM осуществляется с помощью утилиты java.exe.
- Поскольку существует реализация JVM для многих операционных систем, один и тот же файл .class может быть запущен на разных ОС. Этим достигается переносимость или кросс-платформенность Java-программ.
- Виртуальная машина Java (JVM) – часть среды исполнения Java (JRE), выполняющая интерпретацию байт кода в машинный код конкретного процессора.
- Виртуальная Java-машина не только исполняет байт-код (интерпретирует его, занимается JIT-компиляцией и исполняет JIT-компилированный код), но и выполняет ряд других функций. Например, взаимодействует с операционной системой, обеспечивая доступ к файлам или поддержку графики. А также обеспечивает автоматическое высвобождение памяти, занятой ненужными объектами – так называемую сборку мусора (garbage collection).
- Все методы стандартных классов, вызываемые в программе, подключаются к ней только на этапе выполнения, а не включаются в байт-коды, то есть происходит динамическая компоновка.

Стандартная библиотека классов Java

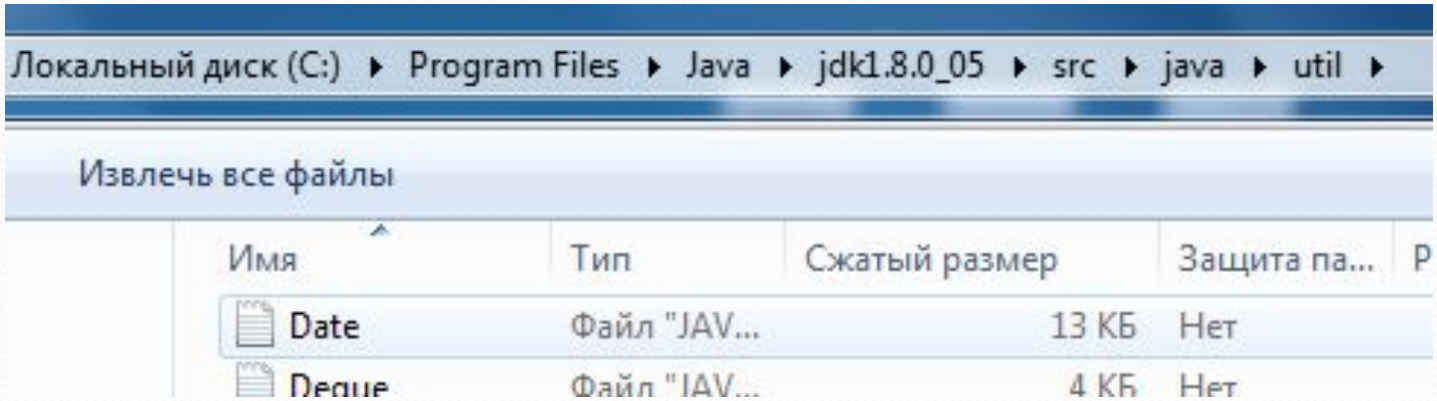
- **Наиболее часто используемые пакеты библиотеки классов:**

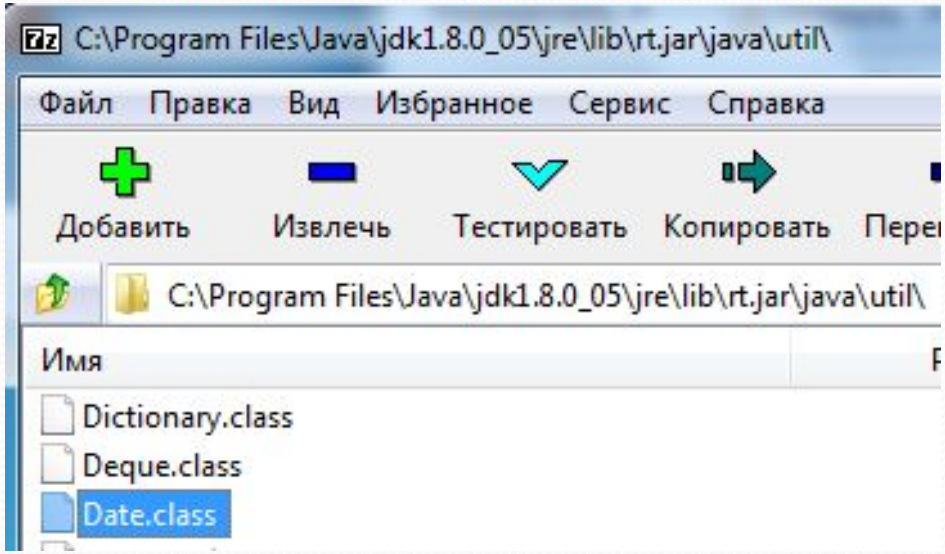
- **java.lang - базовые классы, необходимые для работы любого приложения (название - сокращение от language);**
- **java.util - многие полезные вспомогательные классы;**
- **java.awt, java.swing - библиотека для создания графического интерфейса пользователя(GUI);**
- **java.io - работа с потоками данных (streams) и с файлами.**

- **Импортирование классов пакетов в программу:**
- `import java.lang.*;`
- `import java.util.Date;`

Стандартная библиотека классов Java

- **Пример.** **Размещение класса java.util.Date**



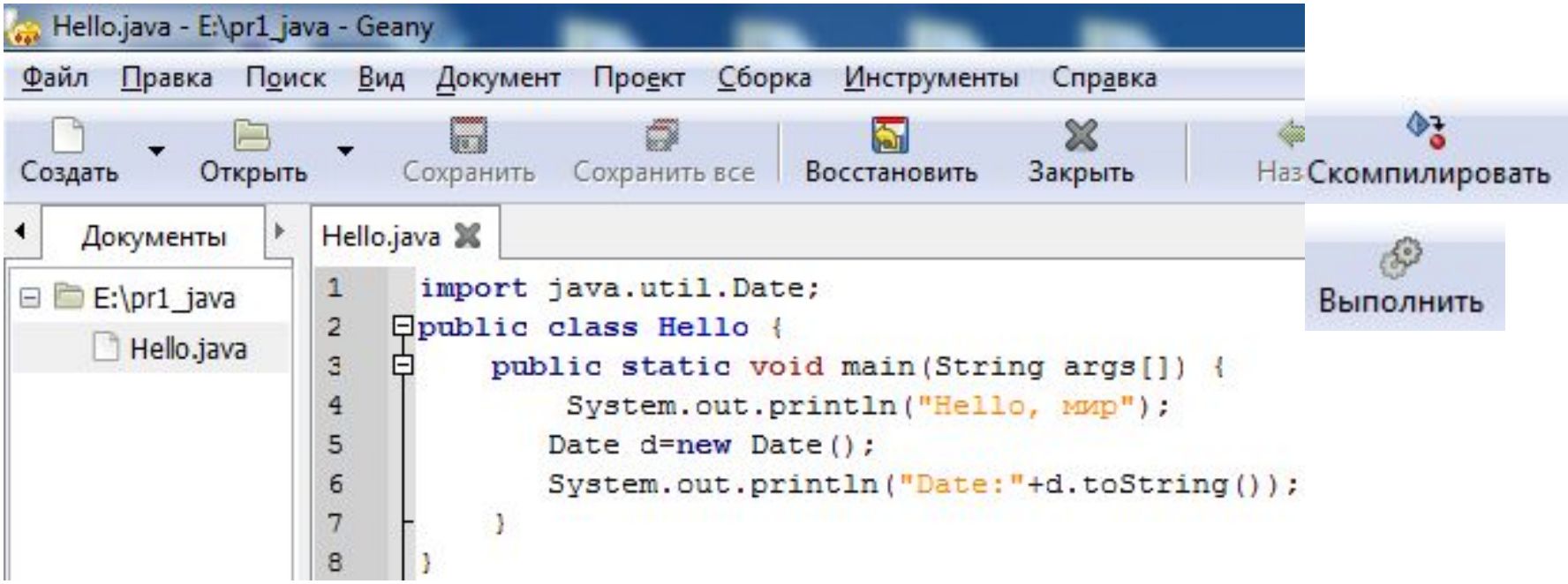


Стандартная библиотека классов Java

- Стандартная библиотека классов позволяет существенно упростить процесс разработки программ на языке программирования Java.
- Классы стандартной библиотеки классов размещены в пакете с именем java и его подпакетах. Например, базовые классы, необходимые для работы любой java-программы, находятся в подпакете lang пакета java (java.lang). Пакет java.lang по умолчанию импортируется в каждый модуль компиляции, поэтому команду import java.lang.*; писать в программе не обязательно.
- Пакет (package) - это некий контейнер для классов, который задает для классов пространство имен, каталог для размещения и пакетный уровень доступа.
- Например, класс Date стандартной библиотеки java размещается в пакете java.util. На слайде номер 11 показаны каталоги, в которых находится файлы Date.java и Date.class.

Структура, компиляция и выполнение Java-программы

- **Пример 1:**



- **Кодировка:**

Кириллица (WINDOWS-1251)

```
C:\Users\admin>E:

E:\>cd pr1_java

E:\pr1_java>javac Hello.java

E:\pr1_java>java Hello
Hello, мир
Date:Tue Aug 05 14:03:34 GMT+07:00 2014
```

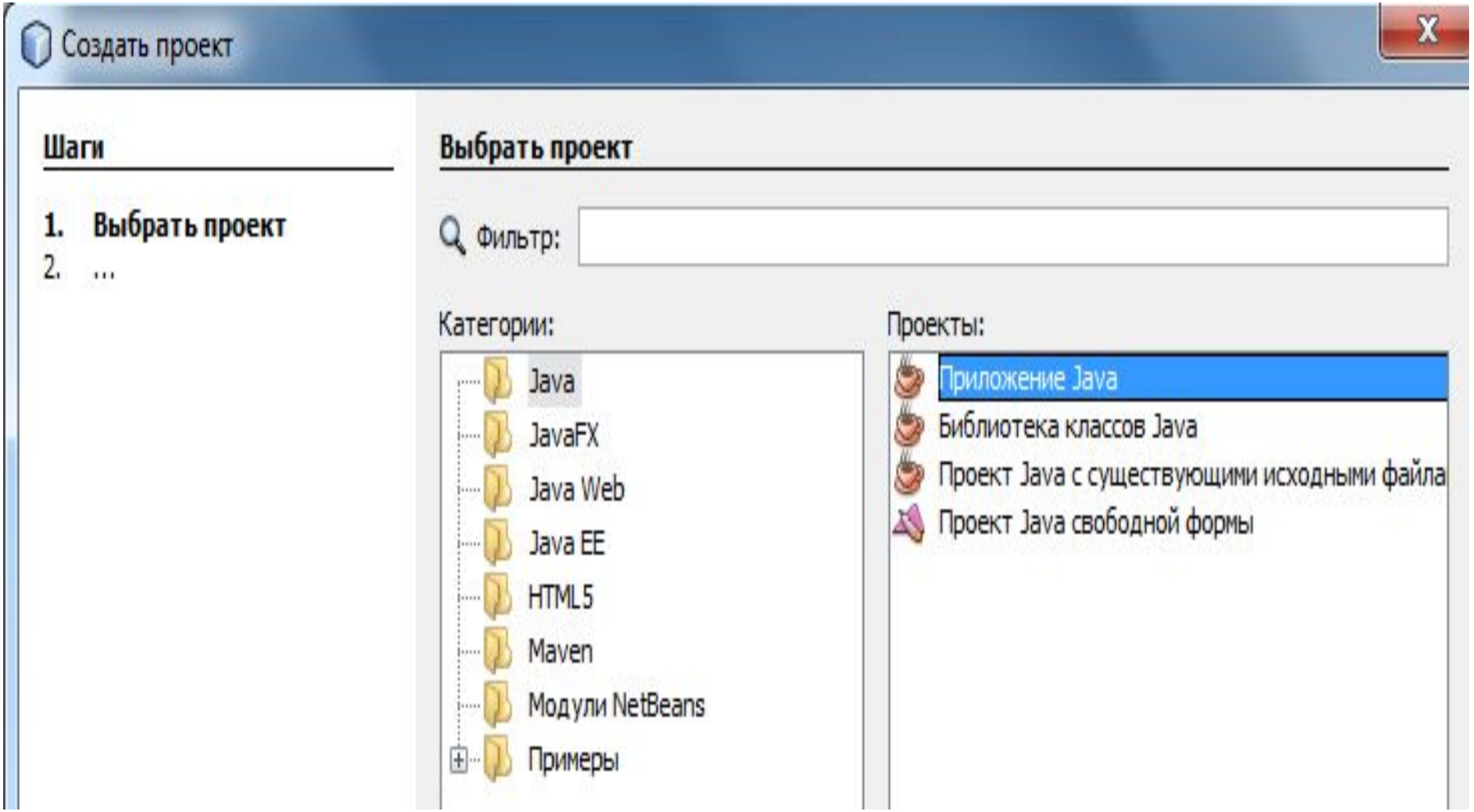
```
E:\pr1_java>dir

Содержимое папки E:\pr1_java

.
..
Hello.class
Hello.java
```

Структура, компиляция и выполнение Java-программы.

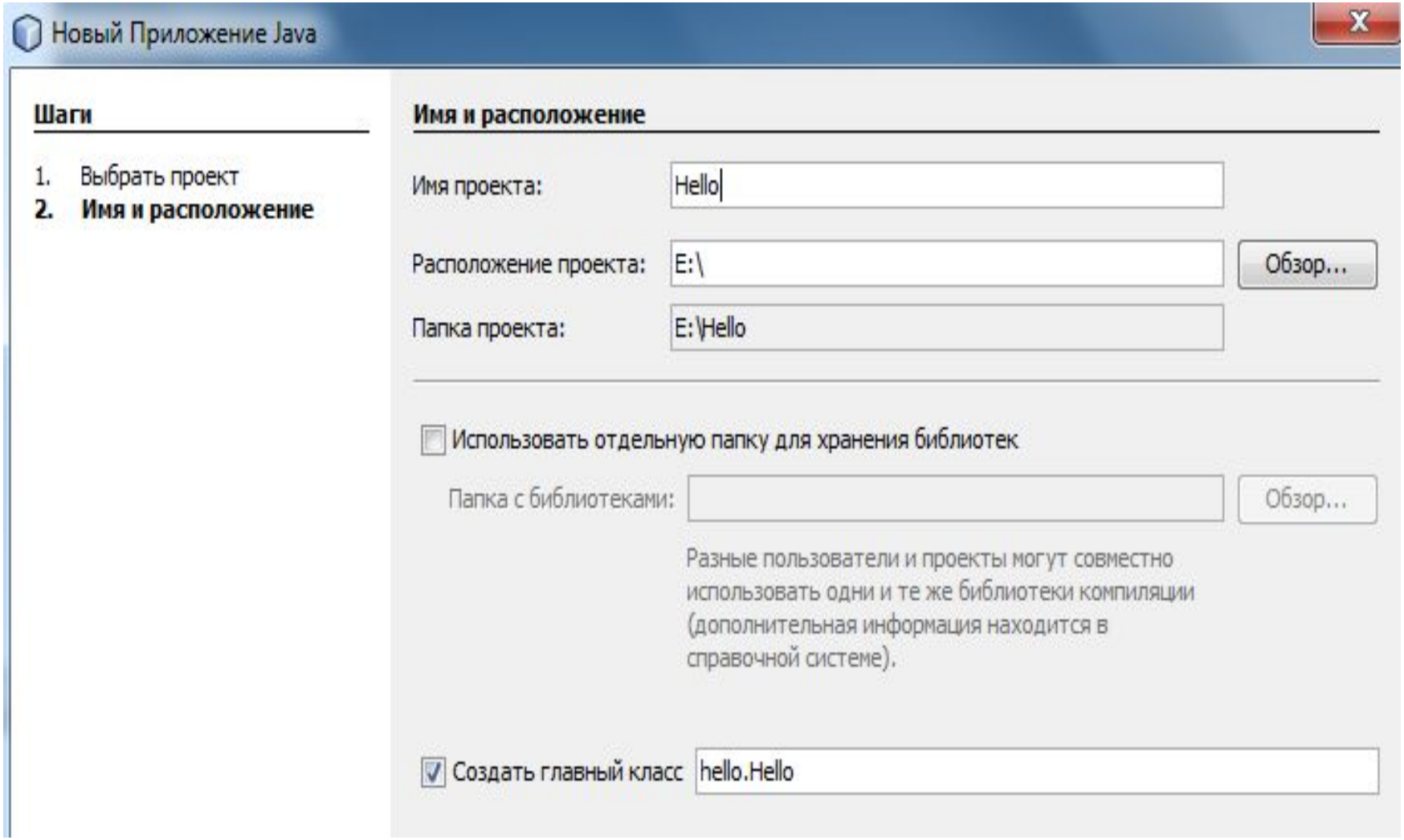
- В примере 1 файл с программой имеет имя Hello и расширение .java (Hello.java). Файл с программой набран и сохранен с помощью программы Geany.
- В командной строке запустим компилятор javac Hello.java и получим файл Hello.class, который можно запустить на выполнение при помощи команды java Hello.
- Если установлен JDK и добавлен путь “ C:\Program Files\Java\jdk1.8.0_05\bin” к системной переменной PATH, то в Geany можно не только набрать программу на Java, но также ее можно компилировать и выполнять.
- Рассмотрим исходный код программы примера 1. Вся программа состоит из одного класса с именем Hello. Класс Hello находится в пакете по умолчанию, которому соответствуют папка pr1_java, в которой находится класс Hello. У этого класса имеется единственный метод main. С метода main начинается выполнение программы. Все приложения Java должны иметь один метод main(). Давайте расшифруем каждое слово в коде.
- Ключевое слово **import** указывает пакет, в котором находится стандартный класс Date.
- Ключевое слово **public** - это спецификатор доступа. Когда члену класса предшествует public, то к этому члену возможен доступ из кода, внешнего по отношению к классу, в котором описан данный метод. В данном случае, метод main объявлен как public для того, чтобы JVM могла обратиться к этому методу.
- Ключевое слово **static** позволяет методу main() вызываться без потребности создавать объект класса Hello. Это важно, потому что JVM вызывает этот метод в первую очередь. Следовательно этот метод должен быть как static и не должен зависеть от экземпляров любого создаваемого класса.Через переменную-массив args типа String методу main() могут передаются параметры командной строки.
- Оператор System.out.println(“Hello, мир") посылает строку текста в стандартный поток вывода, т. е. на экран. Оператор Date d = new Date() создает объект d класса Date и инициализирует его текущей датой и временем.
- Оператор System.out.println("Date:"+d.toString()) посылает сформированную строку текста на экран.

Создание программы на Java в NetBeans IDE 8.0

Файл -> Создать проект

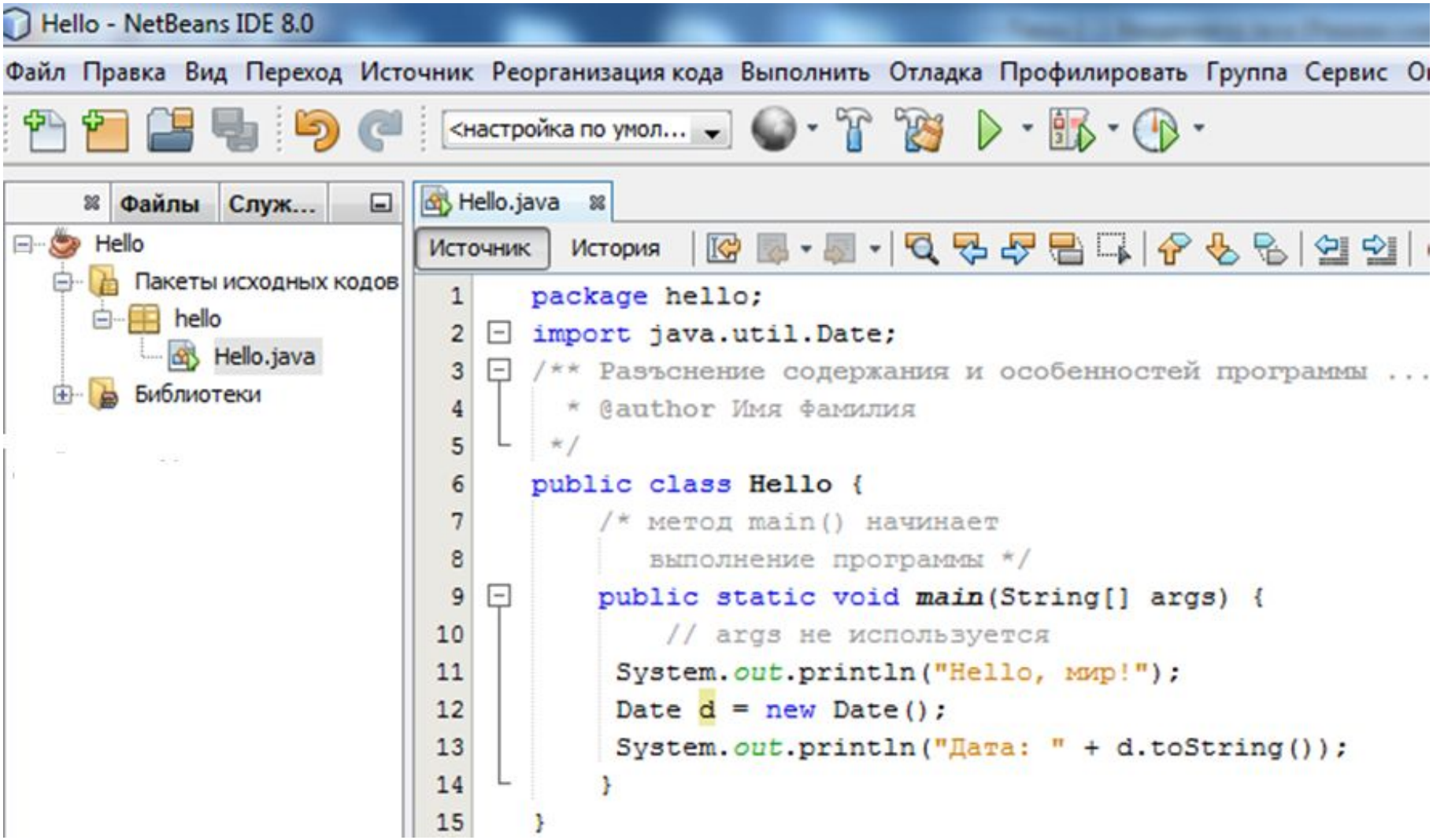


Создание программы на Java в NetBeans IDE 8.0



Создание программы на Java в NetBeans IDE 8.0

Пример программы с комментариями



Создание программы на Java в NetBeans IDE 8.0

- в Java как и в C++ можно использовать однострочные и многострочные комментарии:

  // комментарий до конца строки;

  /* комментарий, который может
  занимать несколько строк */ .

- В Java введены комментарии третьего типа:

  /**начинается комментарий, который может занимать несколько строк
  до звездочки и наклонной черты */

- Утилита javadoc из JDK извлекает эти комментарии в отдельные файлы формата HTML и создает гиперссылки между ними. В такой комментарий можно вставить указания программе javadoc , которые начинаются с символа @.

- **Типы данных Java:**
  - простые;
  - ссылочные

Простые типы данных

Тип	Содержание	Разрядность (бит)	Значение по умолчанию
boolean	Логическое значение true или false	-	false
byte	Целое со знаком	8	0
char	Unicode-символ	16	'\u0000'
short	Целое со знаком	16	0
int	Целое со знаком	32	0
long	Целое со знаком	64	0L
float	Число с плавающей точкой	32	0.0F
double	Число с плавающей точкой	64	0.0D

- **Классы-оболочки для простых типов:**
  - Boolean, Byte, Character, Short, Integer, Long, Float, Double

```
int i = Integer.parseInt("123");
boolean = Boolean.parseBoolean("true");
```

Типы данных Java

- В Java все типы делятся на простые и ссылочные.
- Простые типы данных – это типы данных, для которых в языке Java определен размер занимаемой памяти, диапазон допустимых значений и набор допустимых операций.
- В Java определены 8 простых типов данных: boolean, byte, char, short, int, long, float, double.
- **boolean:** хранит значение true или false.
- **byte**: хранит целое число от -128 до 127 и занимает 1 байт (8 бит).
- **char**: хранит одиночный символ в кодировке Unicode и занимает 2 байта (16 бит), поэтому диапазон хранимых значений от 0 до 65536.
- **short:** хранит целое число от -32768 до 32767 и занимает 2 байта (16 бит).
- **int**: хранит целое число от -2147483648 до 2147483647 и занимает 4 байта (32 бита).
- **long**: хранит целое число от –9 223 372 036 854 775 808 до 9 223 372 036 854 775 807 и занимает 8 байт (64 бита).
- **float:** хранит число с плавающей точкой от 3.4e-38 до 3.4e+38 и занимает 4 байта (32 бита).
- **double**: хранит число с плавающей точкой от 1.7e-308 до 1.7e+308 и занимает 8 байт (64 бита).
- *Замечания:*
  - В Java, в отличие от C++, нет беззнаковых целых типов.
  - Тип char используется для хранения символов в кодировке Unicode, которая позволяет работать с алфавитами различных языков. Первые 256 символов соответствуют стандарту ASCII.
- Классы-оболочки для простых типов содержатся в пакете java.lang и имеют очевидные имена: Integer, Float, Double, Short, Byte, Character, Boolean, Long.
- В этих классах определены полезные методы преобразования типов, например, методы преобразования строк, состоящих из цифр, в числа. В классе Integer, в частности, есть метод ***static int parseInt(String s)*** преобразующий строку из цифр в целое число.
- Как правило, классы оболочки используются в типизации коллекций (например, List <Integer>). Нужно везде, где это возможно, стараться использовать простые типы.

Приведение типов данных в Java

- **Неявное приведение простых типов:**
  - byte, short, char -> int -> long -> float -> double;
- **Пример 2:**

```
1  public class Test2 {
2    public static void main(String args[])
3      { byte b1 = 50, b2 = -99;
4        short k = b1 + b2; //Ошибка!
5       // short k = (short)(b1 + b2); //Верно!
6        System.out.println("k = " + k);
7    }
8  }
```

```
Hello.java:4: error: incompatible types: possible lossy conversion from int to short
    short k = b1 + b2; //     !
                  ^
1 error
```

- **Пример 3:**

```
1  public class Test3 {
2    public static void main(String args[])
3      { float x = 2.5f; int i = 1;
4          i = i + x; //Ошибка!
5       // i = (int)(i + x); //Верно!
6       // i += x; //Верно!
7        System.out.println("i = " + i);
8    }
9  }
```

```
Hello.java:4: error: incompatible types: possible lossy conversion from float to int
    i = i + x; //     !
          ^
1 error
```

Приведение типов данных в Java

- Приведение типов в Java осуществляется автоматически (неявно) и с помощью операции явного приведения типов.
- Синтаксис объявления переменных в Java такой же как и в C++. В методах все локальные переменные простых типов перед использованием обязательно надо инициализировать. Попытка использования значения неинициализированной переменной приводит к ошибке во время компиляции. Над данными простых типов можно производить массу операций. Их набор восходит к языку C, он оказался удобным и кочует из языка в язык почти без изменений.
- Перед выполнением арифметической операции всегда происходит *повышение* типов byte, short, char. Они преобразуются в тип int, а может быть, и в тип long, если другой операнд типа long.
- Это правило приводит иногда к неожиданным результатам. Попытка откомпилировать простую программу, представленную на слайде (пример 2), приведет к сообщениям компилятора. Эти сообщения означают, что в файле HelloWorld.java, в строке 4, обнаружена возможная потеря точности (possible loss of precision). Затем приводятся обнаруженный (found) и нужный (required) типы, выводится строка, в которой обнаружена ошибка, и отмечается символ, при разборе которого найдена ошибка. Затем указано общее количество обнаруженных ошибок (1 error). В таких случаях следует выполнить явное приведение типа. В данном случае это будет *сужение* типа int до типа short . Оно осуществляется операцией явного приведения, которая записывается перед приводимым значением в виде имени типа в скобках. Определение short k = (short)(b1 + b2); будет верным.
- Операнд типа int повышается до типа long, если другой операнд типа long.
- Операнд типа long повышается до типа float если другой операнд типа float.
- Операнд типа float повышается до типа double если другой операнд типа double.
- Если при простом присваивании происходит понижение типа, то в Java всегда нужно использовать операцию явного приведения типа (пример 3).
- Нельзя выполнять приведение целого типа к типу boolean и наоборот. В Java значение 0 не является эквивалентом значения false, а ненулевая величина - значения true, в отличие от C++.

- **Ссылочные типы:**
  - классы;
  - интерфейсы;
  - массивы.
- **Пример 4:**

```
import java.text.SimpleDateFormat;
import java.util.Date;
public class Test4 {
 public static void main(String args[])
  { Date d = new Date();
    SimpleDateFormat format1 = new SimpleDateFormat("dd.MM.yyyy hh:mm");
    SimpleDateFormat format2 =
          new SimpleDateFormat("День dd Месяц MM Год yyyy Время hh:mm");
    System.out.println(format1.format(d));
    System.out.println(format2.format(d));
  }
}
```

```
10.08.2014 04:29
День 10 Месяц 08 Год 2014 Время 04:29
```

- **Пример 5:**

```
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.GregorianCalendar;
public class Test {
 public static void main(String arg[])
  { Calendar c = new GregorianCalendar();//текущая дата
    Calendar c2 = new GregorianCalendar(2013, Calendar.NOVEMBER, 25);
    c2.add(Calendar.DAY_OF_YEAR, 1); //увеличиваем дату на 1 день
    System.out.println(c2.getTime());// 26.11.2013
    c2.add(Calendar.DAY_OF_YEAR, -1); //уменьшаем дату на 1 день
    SimpleDateFormat format1 = new SimpleDateFormat("dd.MM.yyyy");
    System.out.println(format1.format(c2.getTime()));
    System.out.println(c2.get(Calendar.DAY_OF_MONTH));//25
    System.out.println(c2.get(Calendar.MONTH)+1);//11
    System.out.println(c2.get(Calendar.YEAR));//2013
  }
}
```

```
Tue Nov 26 00:00:00 GMT+07:00 2013
25.11.2013
25
11
2013
```

Типы данных Java

- Переменная, имеющая ссылочный тип, содержит ссылку на объект. Ссылка является своего рода указателем, но с ней нельзя выполнять операции, допустимые в языках C и C++.
- Объявление переменной ссылочного типа не создаёт соответствующий объект. Для создания объекта следует использовать оператор new.
- В Java для обозначения так называемой нулевой ссылки используется ключевое слово null .
- Пример 4 и пример 5 демонстрируют работу со ссылочными типами.
- В примере 4 показана работа с классами Date и SimpleDataFormat. Класс Date находится в пакете java.util, а класс и SimpleDataFormat – в пакете java.text.
- Класс Date хранит время в миллисекундах начиная с 1 января 1970 года. Данный класс имеет конструктор по умолчанию, который возвращает текущее время. Кроме этого можно создать объект Date используя конструктор, который принимает количество миллисекунд начиная с 1 января 1970 года. Для получения этого внутреннего времени используется метод getTime(). Кроме этого уже после создания можно изменить время с помощью setTime(long date).
- Для того, чтобы отображать дату и время в удобном формате используется класс SimpleDataFormat. При создании шаблона для отображения даты dd - означает день, MM - месяц, yyyy - год, hh - часы и mm - минуты. В шаблоне могут присутствовать не все единицы, кроме того в качестве разделителя можно использовать любой текст.
- Абстрактный класс Calendar позволяет работать с датой в рамках календаря. Единственной реализацией его является класс GregorianCalendar, также как и у даты конструктор по умолчанию возвращает календарь на текущий день, но вы можете задать его явно указав все параметры (пример 5).
- Календарь достаточно мощный класс, который позволяет увеличивать или уменьшать различные параметры даты, а также получать их, при этом учитывать високосные годы.

- **Вывод данных:** `out` – объект класса **`PrintStream`**, определенный в классе **`System`** пакета **`java.lang`**.

Пример 5:

```
import java.text.NumberFormat;
public class Test5 {
 public static void main(String args[])
  { System.out.println("Привет, мир!");
    System.out.print("Привет, мир! \n");
    int x=5, y = 6;
    System.out.println("x = " + x + "; y = " + y);
    System.out.printf("x = %d; y = %d \n", x, y);
    String name = "Иван";
    int age = 30;  float h = 1.7678f;
    NumberFormat nf = NumberFormat.getInstance();
    nf.setMinimumFractionDigits(2);
    nf.setMaximumFractionDigits(2);
    System.out.println("Рост: " + nf.format(h) + " метров");
    System.out.printf("Имя: %s Возраст: %d лет Рост: %.2f метров\n",
                      name, age, h);
  }
}
```

```
Привет, мир!
Привет, мир!
x = 5; y = 6
x = 5; y = 6
Рост: 1,77 метров
Имя: Иван Возраст: 30 лет Рост: 1,77 метров
```

Консольный ввод-вывод

- Для взаимодействия с консолью нам необходим класс System. Этот класс располагается в пакете java.lang, который автоматически подключается в программу.
- Для создания потока вывода в классе System определен статический объект out класса PrintStream. В классе PrintStream определен метод println, который позволяет вывести на консоль некоторое значение с последующим переводом курсора на следующую строку (пример 5).
- В метод println передается любое значение, как правило, строка, которое надо вывести на консоль. При необходимости можно и не переводить курсор на следующую строку. В этом случае можно использовать метод System.out.print(), который аналогичен println за тем исключением, что не осуществляет перевода на следующую строку.
- Но с помощью метода System.out.print также можно осуществить перевод курсора на следующую строку. Для этого надо использовать символ ’\n’ (пример 5).
- С помощью метода System.out.println можно выполнить форматированный вывод, если воспользоваться методами класса NumberFormat из пакета java.text (пример 5).
- в Java для форматированного вывода также можно использовать функцию System.out.printf() (пример 5).

Консольный ввод-вывод

- **Ввод данных:** **`in`** – объект класса **`InputStream`**, определенный в классе **`System`** пакета **`java.lang;`**
  Классы **`BufferedReader`** и **`InputStreamReader`** из пакета **`java.io;`**
  Классы-оболочки для простых типов данных и класс

```
import java.io.*;
public class Test6 {
 public static void main(String args[]) throws IOException
  { BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    String str, name;  int age;  float h;
    System.out.print("Имя :"); name = br.readLine();
    System.out.print("Возраст :"); str = br.readLine();
    age = Integer.parseInt(str);
    System.out.print("Рост :"); str = br.readLine();
    h = Float.parseFloat(str);
    System.out.printf("Имя: %s Возраст: %d лет Рост: %.2f метров\n",
                      name, age, h);
  }
}
```

```
Имя :Ivan
Возраст :30
Рост :1.7
Имя: Ivan Возраст: 30 лет Рост: 1,70 метров
```

Консольный ввод-вывод

- В примере 6 ввод данных с клавиатуры производится в два этапа:
  - устанавливается связь с вводимыми с клавиатуры данными;
  - преобразовывается вводимые пользователем данные в данные соответствующего типа.
- Для того, чтобы установить связь с вводимыми с клавиатуры данными используются три различных класса (**BufferedReader, InputStreamRreader, System**), которые взаимодействуют между собой .
- Для получения введенной пользователем с клавиатуры строки используется метод **readLine()** класса **BufferedReader** .
- Преобразование строки, которая содержит некое число, в соответствующее численное значение производится с помощью специальных методов классов оболочек, которые существуют для каждого простого типа данных. Необходимым условием при вызове данных методов является то, что строка должна соответствовать подразумеваемому виду. Если метод получает строку **"Hello"**, то при всем своем желании компилятор не сможет преобразовать ее в число!
- Для более удобного ввода данных в Java существует класс Scanner пакета java.util (пример 7).
- При запуске программы в **консоли Windows**, часто бывает проблема с отображением русских букв. Например, если ввести в примере 6 и примере 7 значение переменной name русскими буквами, то возникнет проблема с их отображением. Для устранения этой проблемы можно использовать класс j**ava.io.Concole**. Помимо правильного определения кодировки, этот класс обладает удобными методами для чтения данных из консоли (пример 8).

- **Ввод данных:** **`in`** – объект класса **`InputStream`**, определенный в классе **`System`** пакета **`java.lang`**;
  Класс **`Scannner`** из пакета **`java.util`**;

- **Пример 7:**

```
import java.util.Scanner;
public class Test7 {
 public static void main(String args[])
  { Scanner in = new Scanner(System.in);
    String name;  int age;  float h = 1.7f;
    System.out.print("Имя : "); name = in.nextLine();
    System.out.print("Возраст : ");
    if (in.hasNextInt())
     { age = in.nextInt();
       System.out.print("Рост : ");
       if (in.hasNextFloat())
        { h = in.nextFloat();
          System.out.printf("Имя: %s Возраст: %d лет Рост: %.2f метров\n",
                      name, age, h);
        } else  System.out.println("Вы ввели не число!");
      } else  System.out.println("Вы ввели не целое число!");
  }
}
```

```
Имя : Ivan
Возраст : 30
Рост : 1,9
Имя: Ivan Возраст: 30 лет Рост: 1,90 метров
```

- **Ввод данных:** Класс **`Console`** из пакета **`java.io;`**
  Классы-оболочки для простых типов данных и класс **`String`** из пакета **`java.lang`**.
- **Пример 8:**

```
import java.io.Console;
public class Test8 {
 public static void main(String args[])
  { Console con = System.console();
    String str, name;  int age;  float h;
    name = con.readLine("Имя :");
    str = con.readLine("Возраст :"); age = Integer.parseInt(str);
    str = con.readLine("Рост :"); h = Float.parseFloat(str);
    System.out.printf("Имя: %s Возраст: %d лет Рост: %.2f метров\n",
                      name, age, h);
  }
}
```

```
Имя :Иван
Возраст :30
Рост :1.9
Имя: Иван Возраст: 30 лет Рост: 1,90 метров
```

Условные и циклические операторы

- **Синтаксис оператора if :**

```
if  (условие)  оператор1; [else оператор2;]
```

- **Синтаксис оператора for:**

```
for (инициализация; усл. завершения; изменение_парам)
   оператор;
```

- **Синтаксис оператора while:**

```
[инициализация;]
while (усл. завершения)
 { операторы; [изменение_парам;]
  }
```

- **Синтаксис оператора while:**

```
[инициализация;]
do
 { операторы;
   [изменение_парам;]
  } while(усл. завершения)
```

Условные и циклические операторы

- Результатом вычисления условия в операторе if и условия завершения в операторах for, while, do while должно быть значение логического типа (boolean): true или false.
- Операции сравнения в Java (==, !=, >, <, >=, <=) возвращают результат логического типа.
- Логические операции в Java (&&, ||, ! и другие) возвращают результат логического типа.
- В Java нельзя написать: int k = 10;

```
while(k)
  { … }.
```

Класс Math.

- **Статические методы класса Math:**
  - `Math.abs(n),`
  - `Math.round(n)`
  - `Math.ceil(n)  (например, Math.ceil(3.4) вернёт 4.0)`
  - `Math.cos(n), Math.sin(n), Math.tan(n)`
  - `Math.acos(n), Math.asin(n), Math.atan(n)`
  - `Math.toDegrees(n), Math.toRadians(n)`
  - `Math.sqrt(n),  Math.pow(n, b)`
  - `Math.log(n), Math.log10(n)`
- **Статические константы класса Math:**
  - `Math.PI, Math.E`
- **Примеры объявления именованных констант:**
  - `final int n = 10;`
  - `static final int k = 20;`

Класс Math

- Math.abs(n) — возвращает модуль числа n.
- Math.round(n) — возвращает целое число, ближайшее к вещественному числу n (округляет n).
- Math.ceil(n) — возвращает ближайшее к числу n справа число с нулевой дробной частью (например, Math.ceil(3.4) в результате вернёт 4.0).
- Math.cos(n), Math.sin(n), Math.tan(n) — тригонометрические функции sin, cos и tg от аргумента n, указанного в радианах.
- Math.acos(n), Math.asin(n), Math.atan(n) — обратные тригонометрические функции, возвращают угол в радианах.
- Math.toDegrees(n) — возвращает градусную меру угла в n радианов.
- Math.toRadians(n) — возвращает радианную меру угла в n градусов.
- Math.sqrt(n) — возвращает квадратный корень из n.
- Math.pow(n, b) — возвращает значение степенной функции n в степени b, основание и показатель степени могут быть вещественными.
- Math.log(n) — возвращает значение натурального логарифма числа n.
- Math.log10(n) — возвращает значение десятичного логарифма числа n.
- Все перечисленные функции принимают вещественные аргументы, а тип возвращаемого значения зависит от типа аргумента и от самой функции.
- В языке Java для объявления константы используется ключевое слово final. При объявлении констант после ключевого слова final указывается ее тип.
- В языке Java часто возникает необходимость в константах, доступных нескольким методом внутри одного класса обычно они называются константами класса. Константы класса объявляются с помощью ключевых слов static final вне методов класса,

- **Пример 9. Случайные числа** Math.random():

```
public class Test {
 public static void main(String[] args) {
   System.out.println(Math.random()); //вещ .число из [0;1)
   System.out.println(Math.random()+3); //вещ. число из [3;4)
   System.out.println(Math.random()*5); //вещ. число из [0;5)
   System.out.println((int)(Math.random()*5)); // целое
                                               // число из [0;4]
   System.out.println(Math.random()*5+3); // вещ. число
                                          // из[3;8)
   System.out.println((int)(Math.random()*5+3)); // целое
                                                 // число из [3;7]
   System.out.println((int)(Math.random()*11) - 5); // целое
                                                    // число из [-5;5]
  }
}
```

```
0.7848597195214644
3.4252065227557145
2.655615360588765
1
7.398510011361545
4
2
```

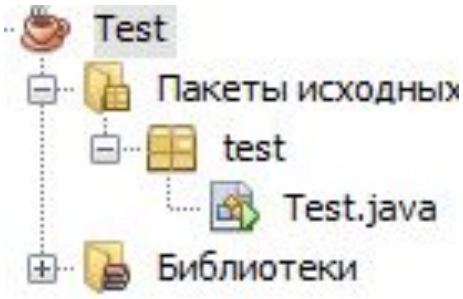
Класс java.util.Random

- **Пример 10. Случайные числа:**

```
true
-1139614796
0.19497605734770518
291
143
```

```
import java.util.Random;
import java.util.Date;
class Test {
  public static void main(String[] args) {
  // Random random =new Random(100);
     Random random =new Random();
  // Random random =new Random(System.currentTimeMillis());
  // Random random = new Random(new Date().getTime());
  // случайное true или false
    System.out.println(random.nextBoolean());
 // случайное число от Integer.MIN_VALUE до Integer.MAX_VALUE
    System.out.println(random.nextInt());
    // случайное число от -1 до 1
    System.out.println(random.nextDouble());
    // случайное число от 0 до 500
    System.out.println(random.nextInt(500));
    // случайное число от 100 до 239
    System.out.println(random.nextInt(239 - 100) + 100);
    // аналогично с float и long
  }
}
```

- **Пример 11. Проверка знания таблицы умножения:**



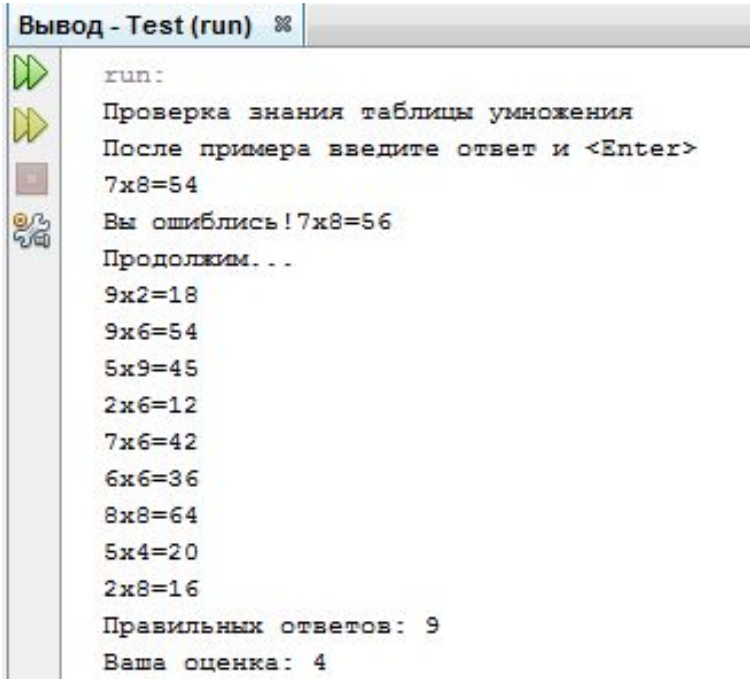

```
package test;
import java.util.Random;
import java.util.Scanner;
class Test {
 public static void main(String[] args) {
  int numb1, numb2; //сомножители
  int res;          //произведение
  int otv;          //ответ испытуемого
  int kol = 0;  //кол-во правильных ответов
  System.out.println("Проверка знания таблицы
  умножения");
  System.out.println("После примера введите ответ и
  <Enter>");
  final int n = 10; // 10 примеров
  Random random =new Random();
  Scanner in = new Scanner(System.in);
```

Пример 11 (продолжение)

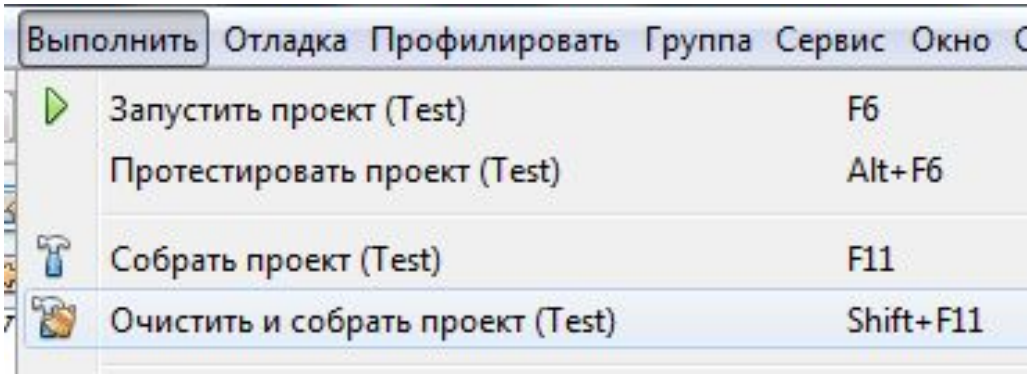
```
for (int i=1;i<=n;i++)
 { numb1 = random.nextInt(10 - 2) + 2;
   numb2 = random.nextInt(10 - 2) + 2;
   res = numb1*numb2;
   System.out.print(numb1 + "x" + numb2 + "=");
   otv = in.nextInt();
   if (otv == res) kol++;
    else
     { System.out.print("Вы ошиблись!");
  System.out.println(numb1 + "x" + numb2 + "=" +res);
          System.out.println("Продолжим...");
        }
    }
 System.out.println("Правильных ответов: "+ kol);
```

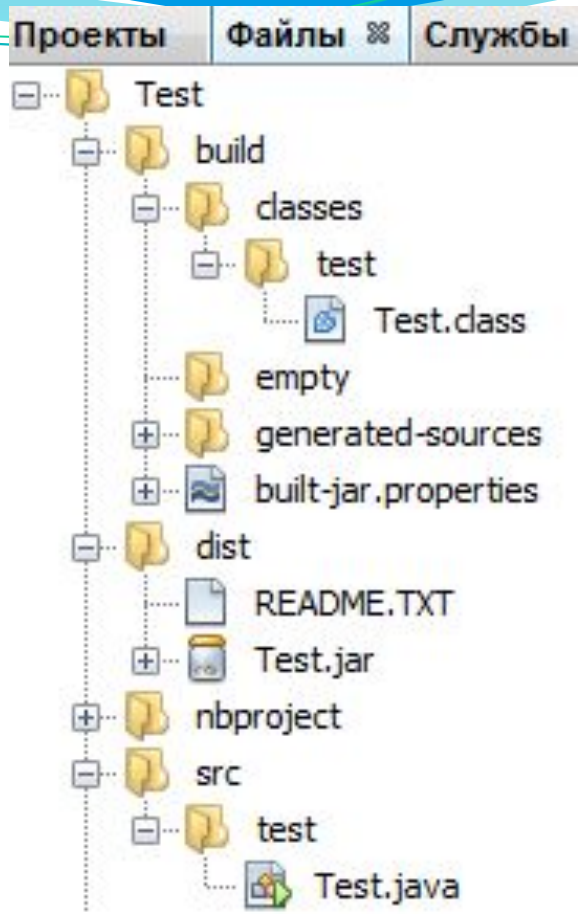
Пример 11 (продолжение)

```
System.out.print("Ваша оценка: ");
  switch (kol)
  { case 10 : System.out.print("5"); break;
    case 9 : System.out.println("4"); break;
    case 8 : System.out.println("4"); break;
    case 7 : System.out.println("3"); break;
    default: System.out.println("2"); break;
  }
  }
}
```



Создание Test.jar в Netbeans






```
Created dir: E:\Test\dist


Building jar: E:\Test\dist\Test.jar
To run this application from the command line without Ant, try:
java -jar "E:\Test\dist\Test.jar"
```

Контрольные вопросы

1. Понятие Java. Понятие и назначение JRE и JDK. Версии JDK.
2. Структура Java - программы, примеры. Компиляция и запуск программы из командной строки. Понятие и назначение JVM.
3. Простые типы данных Java: понятие, примеры. Преобразование простых типов данных в Java: правила и примеры.
4. Ссылочные типы данных Java: понятие, примеры. Классы Date и GregorianCalendar: назначение и примеры использования.
5. Консольный ввод-вывод данных в Java: характеристика и примеры.