

Java

Курс «Основы программирования на Java»

Введение в Java

Автор: А. Е. Анисимов, УдГУ

План курса:

- ВВЕДЕНИЕ В JAVA
- ТИПЫ ДАННЫХ И ОПЕРАТОРЫ
- МАССИВЫ
- СТРОКИ
- ФАЙЛЫ. ПОТОКИ ВВОДА/ВЫВОДА
- КЛАССЫ. ИНКАПСУЛЯЦИЯ
- НАСЛЕДОВАНИЕ И ПОЛИМОРФИЗМ
- АБСТРАКТНЫЕ КЛАССЫ. ИНТЕРФЕЙСЫ
- ШАБЛОНЫ КЛАССОВ
- ИСКЛЮЧЕНИЯ И ОШИБКИ
- КОЛЛЕКЦИИ
- МНОГОПОТОЧНОСТЬ

Контакты, ссылки:

- **Анисимов Андрей Евгеньевич**, ст. преподаватель кафедры теоретических основ информатики УдГУ, ауд. 312 корп. VI. aae@udsu.ru
- <https://vk.com/imitifjava> - Группа ВК

Формы занятий:

- **Лекция** : Теория
- **Практика**: Разбор и решение задач
- **Приём**: Анализ преподавателем выполненных студентами заданий
- **Самостоятельная работа**: Изучение студентом материала и работа над заданием

Программные средства:

Название	Описание	Лого	Где взять
JDK Java Development Kit , ver >=10	Комплект приложений разработчика (JVM, JRE, javac)		http://www.oracle.com
Eclipse	Интегрированная среда разработки		http://www.eclipse.org
IntelliJ IDEA	Интегрированная среда разработки		https://www.jetbrains.com
MySQL Server ...	Сервер базы данных		http://www.mysql.com
Microsoft Workbench			

Рекомендуемая литература:

- Блинов И.Н., Романчик В.С. Java. Промышленное программирование : практ. пособие /– Минск : УниверсалПресс, 2007. <http://book.e-ran.by>
- Эккель Брюс. Философия Java. СПб.: Питер, 2009.
- Васильев А.Н. Java. Объектно-ориентированное программирование. СПб.: Питер, 2012.
- Хорстманн К., Корнелл Г. Java 2. Библиотека профессионала. М.: Изд. Дом «Вильямс», 2006. В 2 т.

Рекомендуемые источники:

- <http://www.java.com/> — официальный сайт
- <https://javarush.ru/> — онлайн-курс обучения программированию на Java
- <https://stepik.org> — онлайн-платформа обучения разным курсам, в т.ч. Java
- <https://geekbrains.ru/courses/111> курсы Java

Java

Тема: Введение в Java

Курс «Основы программирования
на Java»

- Тема: Введение в Java

A. Основные понятия ООП

B. Язык Java

C. Простое приложение

D. Классы и объекты

E. Сравнение объектов

F. Консоль

G. Обзор среды разработки Eclipse

А. Основные понятия ООП

- **Объектно-ориентированное программирование (ООП)** – методология программирования, основанная на представлении программного продукта в виде совокупности объектов, каждый из которых является экземпляром (представителем) конкретного определенного класса (типа).

- **Объект** – *сущность*, обладающая совокупностью свойств и проявляющая свое поведение.
 - физически размещается в памяти компьютера
 - как правило – именованная сущность
 - набор данных (*полей*) и функций (*методов*)
 - любой объект относится к определённому классу (типу) – является его *экземпляром*

- Пример 1. Объекты

Предмет_1

-Название: Чайник
-Цена: 1900 руб.
-Масса: 1000 г

+ Налить()
+ Вскипятить()



Предмет_2

-Название: Кастрюля
-Цена: 2500 руб.
-Масса: 1500 г

+ Налить()
+ Вскипятить()

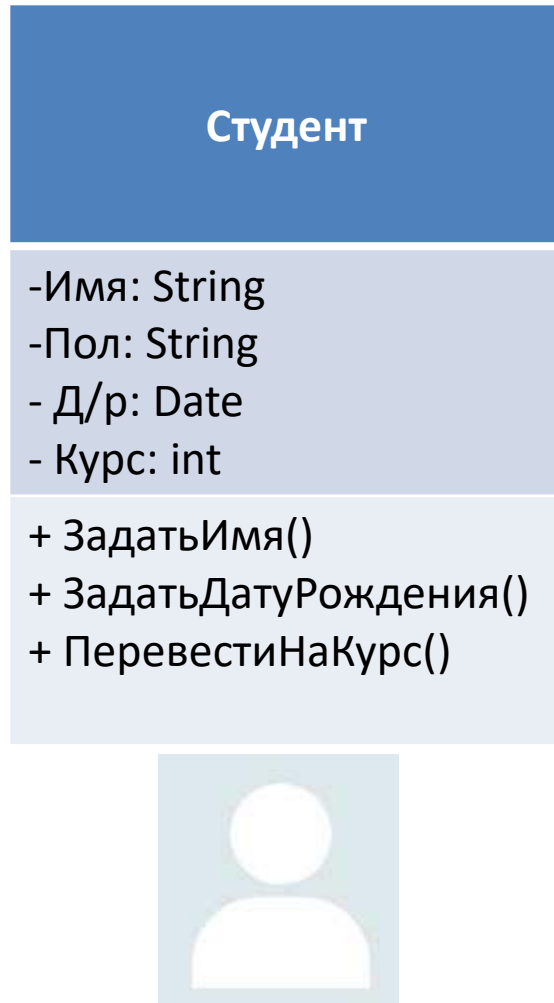


- Пример 2. Объекты

Иван	Мария
<ul style="list-style-type: none">-Имя: Иван-Пол: мужской- Д/р: 01.10.1994- Курс: 2	<ul style="list-style-type: none">-Имя: Мария-Пол: женский- Д/р: 27.11.1993- Курс: 3
<ul style="list-style-type: none">+ ЗадатьИмя()+ ЗадатьДатуРождения()+ ПеревестиНаКурс()	<ul style="list-style-type: none">+ ЗадатьИмя()+ ЗадатьДатуРождения()+ ПеревестиНаКурс()

- **Класс** – *тип данных*, определяющий внутреннее устройство родственных объектов.
 - описание данных (полей) и функций (методов)
 - обобщенное описание совокупности реально существующих сущностей (объектов)

- Пример 3. Класс



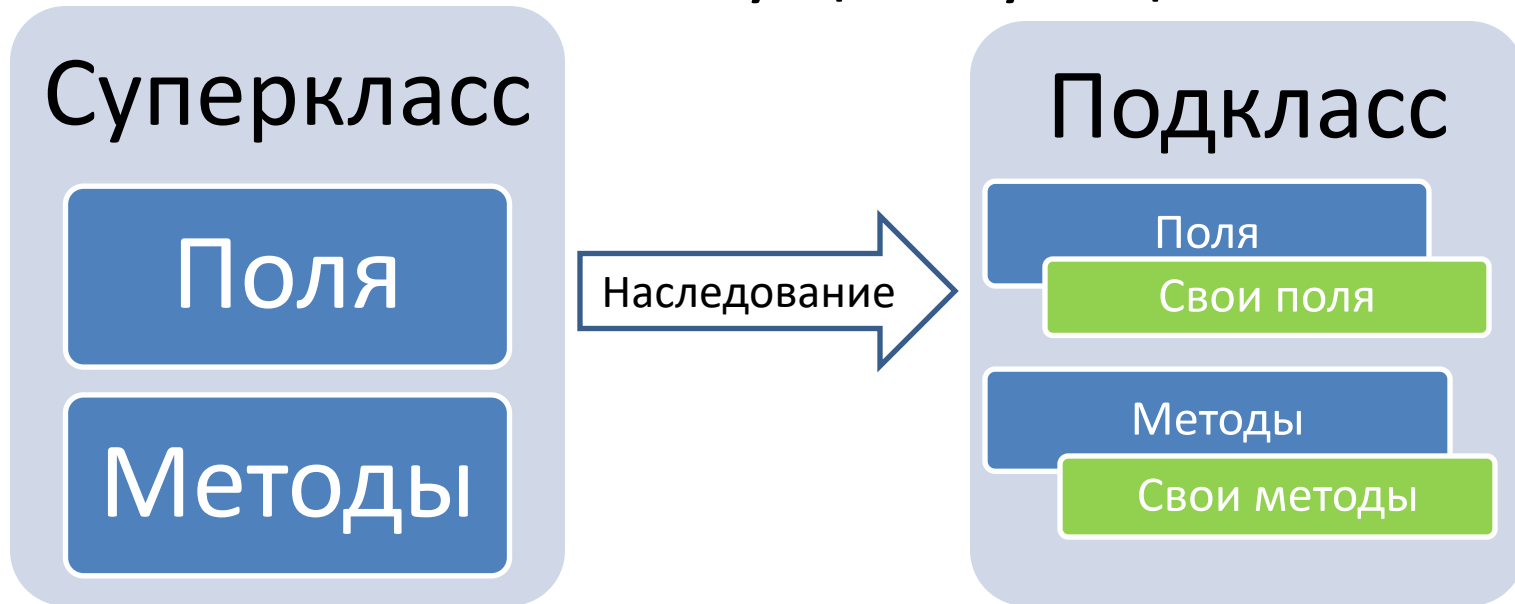
Основные принципы ООП:

1. Инкапсуляция
2. Наследование
3. Полиморфизм

Принцип 1: Инкапсуляция - принцип, объединяющий данные и манипулирующий этими данными код, а также защищающий данные от прямого внешнего доступа и неправильного использования.

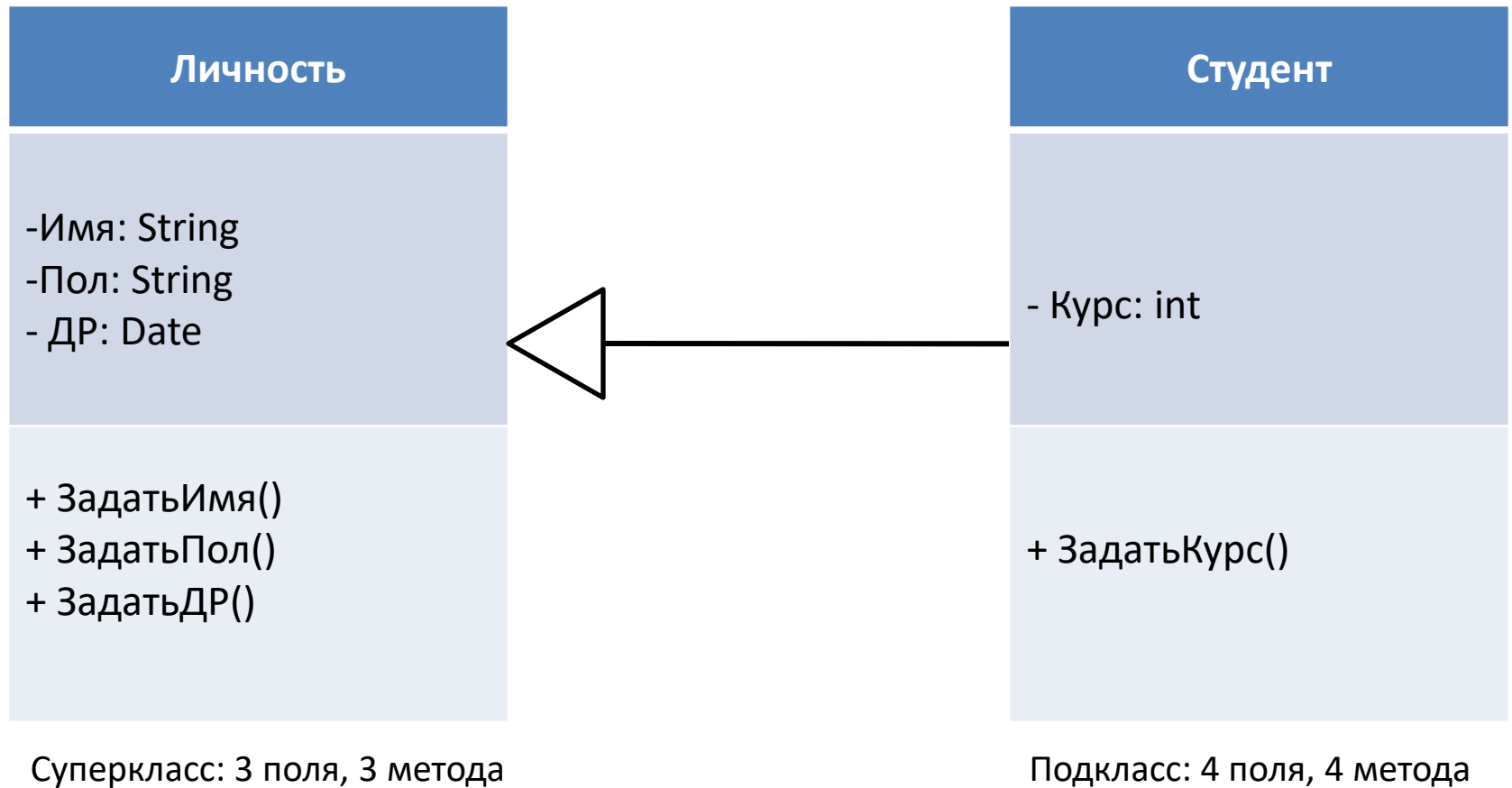


Принцип 2: Наследование – создание нового класса на основе существующего.

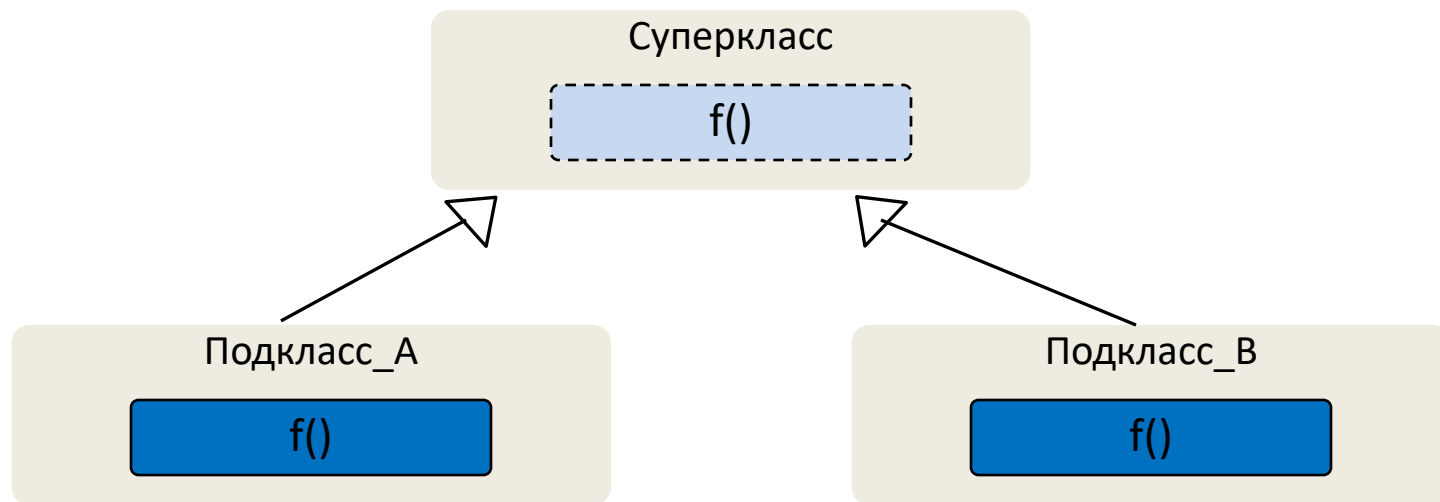


Подкласс может *наследовать* свойства суперкласса, а также может *добавить* свои свойства или *переопределить* (перекрыть) родительские.

- Условное обозначение наследования



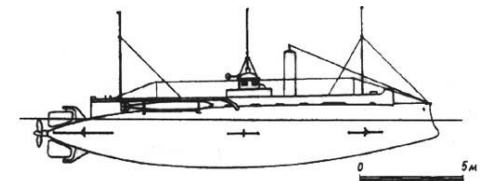
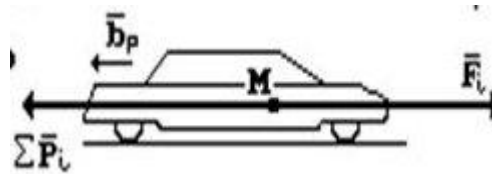
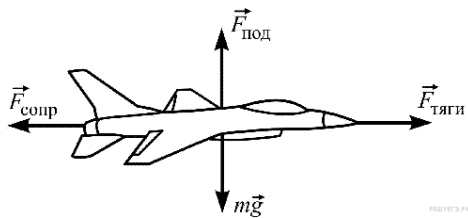
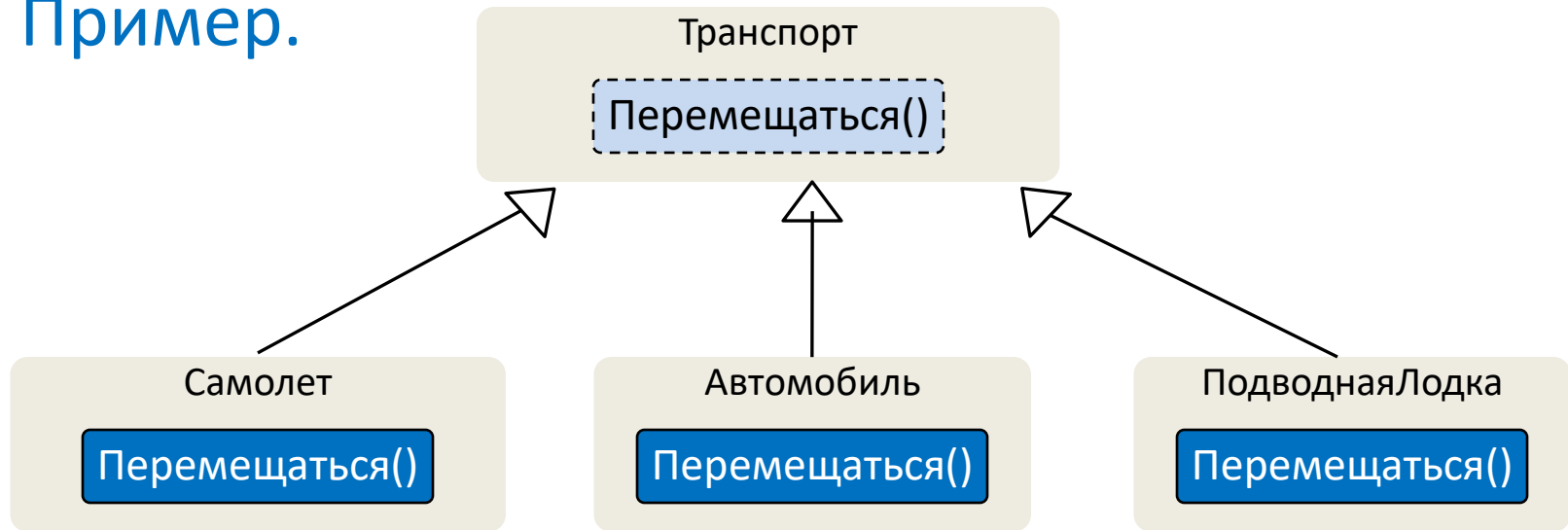
Принцип 3: Полиморфизм – принцип, согласно которому объекты могут содержать различную реализацию одних и тех же действий.



Пусть есть объект x. Вызывается x.f() . Который?

Определение того, метод какого именно класса должен быть вызван происходит во время выполнения и зависит от фактического типа (класса) объекта. Это называется *позднее связывание*.

Пример.



В. Язык Java

История

- Изначально был разработан для управления бытовой электроникой
- Sun Microsystems (с января 2010 - Oracle)

Версия	Год	Примечание
Начало	1990	James Gosling (OAK)
Java 1.0	1996	
Java 1.2 (Java 2)	1998	Swing, Unicode, CORBA, коллекции и др.
Java 1.4	2002	NIO (ввод-вывод), раб. с XML
Java 5.0 (Java 1.5)	2004	enum, аннотации, generic, autoboxing, foreach, Javadoc
Java 6 (Java 1.6)	2006	Развитие коллекций, упр. кэшем и др.
Java 7	2011	OpenJDK, динамически-ориент. типизация
Java 8	2014	λ-выражения, default в интерфейсах ...
Java 9, 10, 11...	...	

Java - это *объектно-ориентированный, платформенно-независимый* язык программирования, используемый для разработки информационных систем, работающих в сети *Internet*.

Объектно-ориентированный язык Java, разработанный в Sun Microsystems, предназначен для создания *переносимых* на различные платформы и операционные системы *программ*.

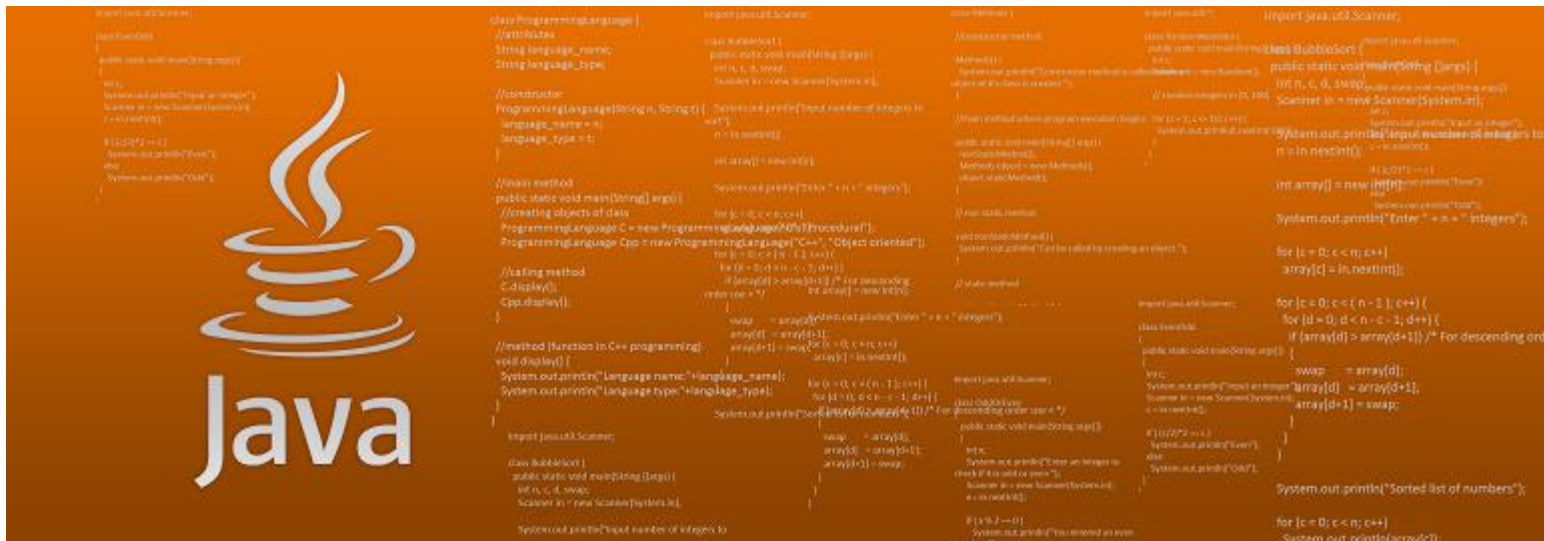
Через несколько лет после своего создания язык практически покинул клиентские страницы и перебрался на сервера. На стороне клиента его место занял язык JavaScript.

Платформы (редакции) Java

Название	Полное название	Описание
Java SE	Java Standard Edition	Уровень настольных систем
Java EE	Java Enterprise Edition	Уровень корпоративных приложений, содержит Java SE
Java ME	Java Micro Edition	Уровень мобильных устройств
Java Card		Уровень приложений, работающих на смарт-картах и других устройствах с очень ограниченным объемом памяти и возможностями обработки.

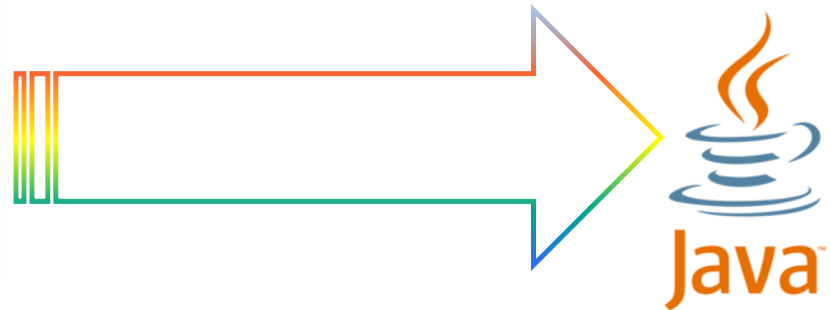
Свойства Java

- Чистый объектно-ориентированный язык
- Интерпретируемый язык (Java Virtual Machine)
- Кросс-платформенность
- Высокая надежность, интерактивность, безопасность, простота и читабельность, многопоточность, богатые и мощные библиотеки и др.

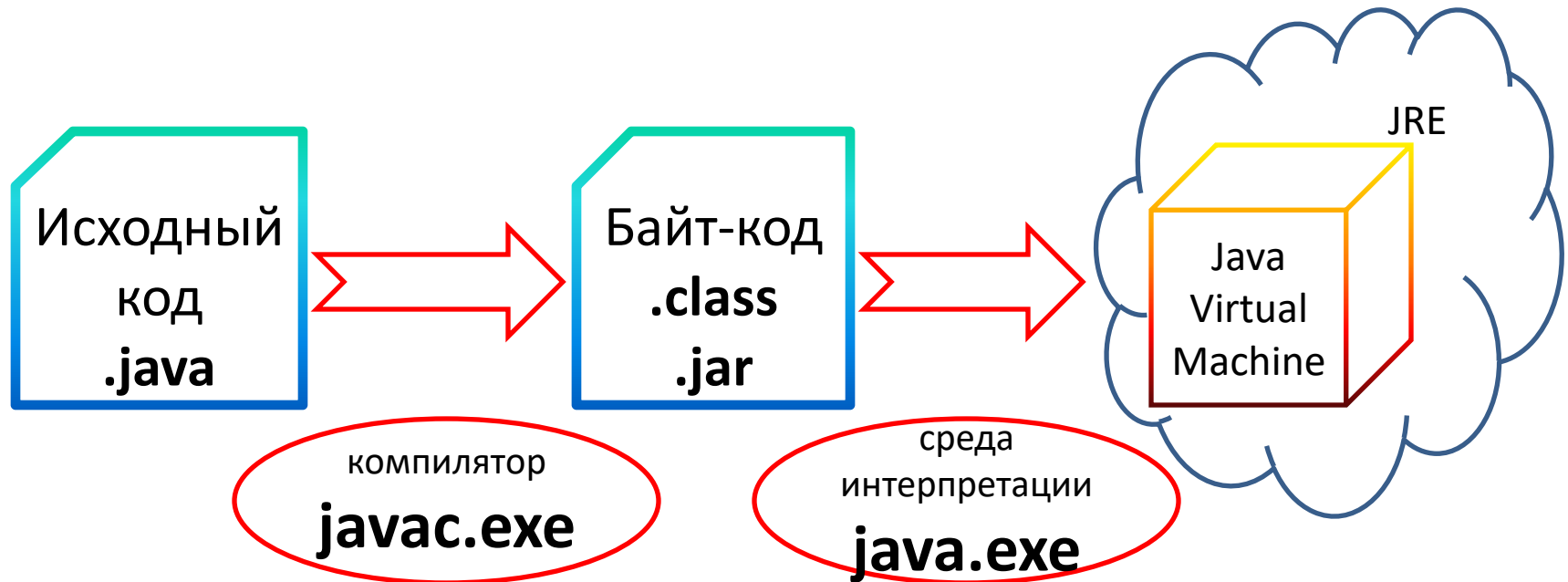


Отличия от C++

- Чистое ООП
- Объекты <- ссылки
- Все объекты – в динамической памяти (куче)
- Нет указателей (*pointer)
- Нет множественного наследования
- Нет освобождения динамической памяти **free()**, **delete**
- Нет заголовочный файлов (.h-файлов)
- Нет структур **struct** и объединений **union**
- Нет перегрузки операций
- Нет беззнаковых целых (**unsigned**)
- Нет деструкторов
- Нет препроцессора



Концепция Java Virtual Machine (JVM)



Написано однажды – работает везде

Запуск приложений

Java Runtime Environment (JRE)

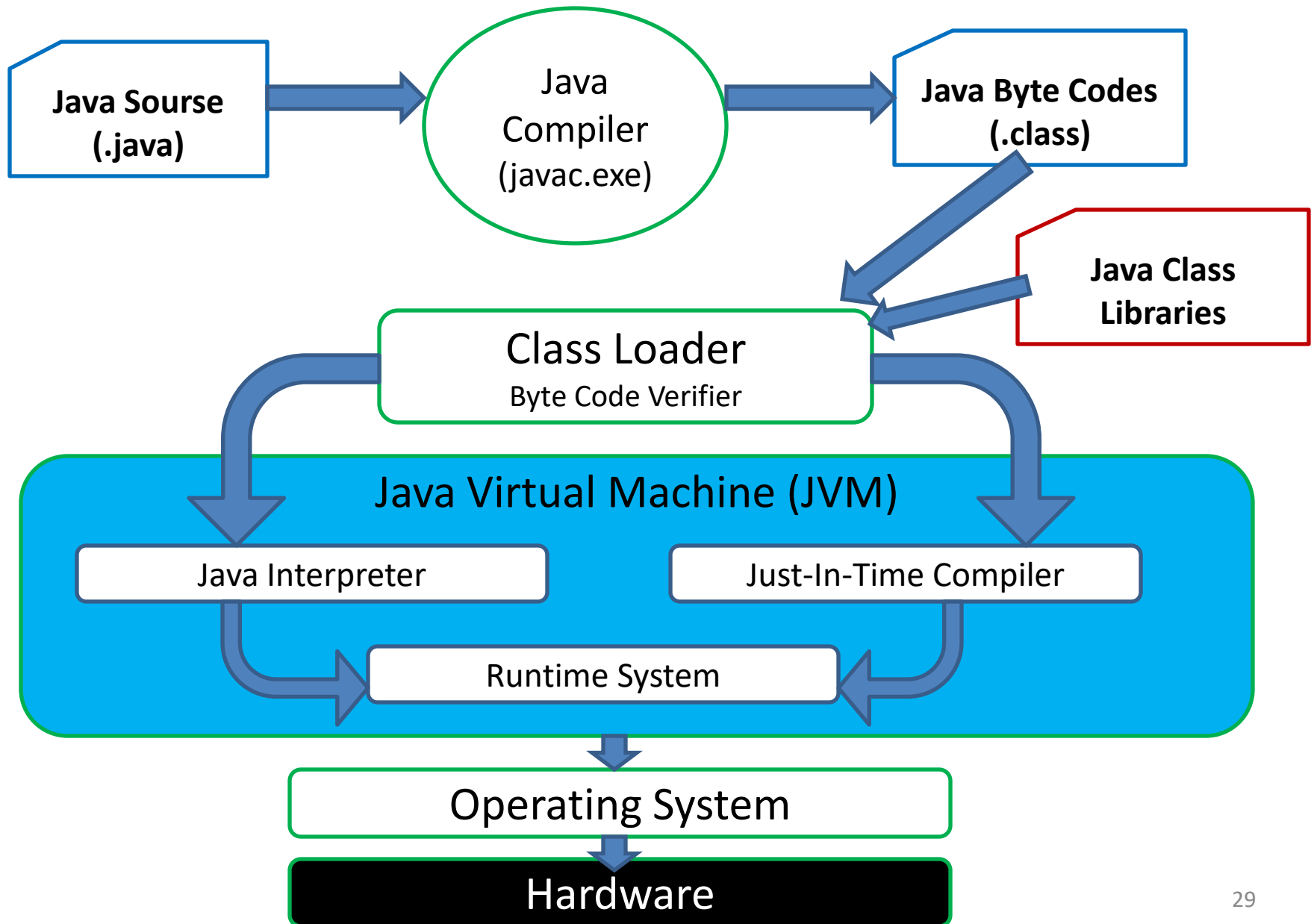
Class Libraries

Создание приложений

Java Development Kit (JDK):

- Компилятор (javac)
- Библиотека классов
- Отладчик (jdb)
- Интерпретатор (java)
- Генератор документации (javadoc)
- Архиватор (jar)
- Другое...
- Integrated Development Environment (не вх.)

Java Тема: Введение в классы и объекты



Приложения

- Клиентские
 - JVM выполняет отдельное приложение из командной строки
 - Классы загружаются с локального диска
- Серверные
 - Обслуживают несколько клиентов
 - Применяются для многозвенных приложений

Апплеты

- Предназначены для встраивания в HTML страницы
- Выполняются внутри браузера и могут взаимодействовать с сервером
- Могут быть преобразованы в обычное приложение

Зарезервированные слова Java

abstract	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
byvalue	extends	instanceof	return	transient
case	false	int	short	true
cast	final	interface	static	try
catch	finally	long	super	var
char	float	native	switch	void
class	for	new		volatile
const	future	null		while
continue	generic			

С. Простое приложение

- Описание класса (упрощенно)

```
class имяКласса {  
    члены_класса  
}
```

- Среди классов должен быть один с методом `main()`

```
public class имяКласса {  
    public static void main(String[] args) {  
        //программный код  
    }  
}
```


- **Пакет (package)** – это контейнер классов; фактически – каталог на диске, в котором размещаются файлы с исходными кодами классов.
- Если пакет у класса не указан, то этот класс размещается в пакете по умолчанию (default package).

- **Комментарии**

// однострочный комментарий

/* многострочный
комментарий */

/** комментарий для генератора
документации javadoc
*/

Пример программы 1.01.

```
/*  
    Пример 1.01.  
    Программа Hello World!  
*/  
  
class HelloWorld {  
    public static void main (String[] args) {  
        System.out.println ("Hello World!");  
    }  
}
```

- Компиляция и исполнение HelloWorld

- Шаг 1. Компиляция программы

> `javac HelloWorld.java`

В результате компиляции файла HelloWorld.java будет сгенерирован файл HelloWorld.class с байт-кодом

- Шаг 2. Интерпретация (исполнение) программы

> `java HelloWorld`

Запускается на исполнение байт-код класса, который находится в файле HelloWorld.class. Для того, чтобы JVM смогла найти этот код, нужно установить в переменной окружения CLASSPATH имена соответствующих каталогов.

- Шаг 3. Можно скопировать байт-код (.class) на другой компьютер и убедиться, что он так же работает.

D. Классы и объекты

[модификаторы_класса]

class ИмяКласса

[extends СуперКласс]

[implements список_интерфейсов]

{

*/*определение класса*/*

}

[модификаторы_класса]

<code>public</code>	класс доступен в данном пакете и вне пакета
<code>final</code>	класс не может иметь подклассов
<code>abstract</code>	класс может содержать абстрактные методы, объект такого класса создать нельзя
<code>static</code>	(Только для вложенных классов). Класс является статическим членом внешнего класса

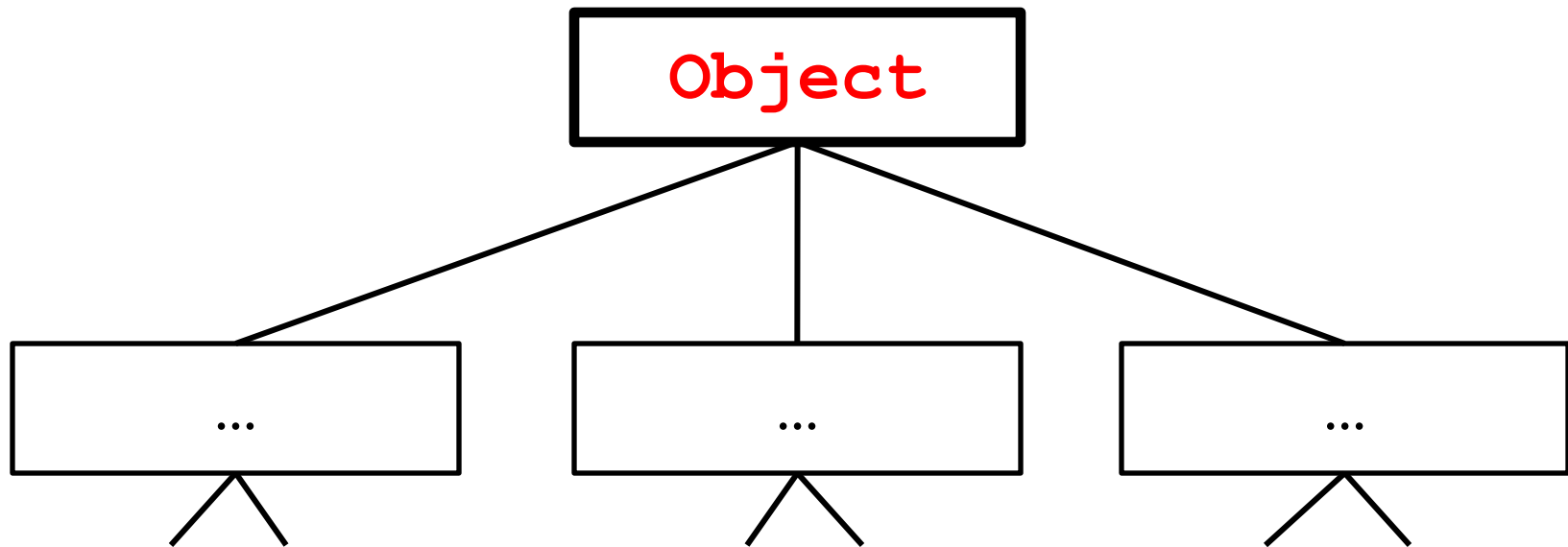
[extends СуперКласс]

определяет базовый класс
(один!)

[implements список_интерфейсов]

определяет список интерфейсов,
реализуемых этим классом

**Все классы Java имеют одного
общего предка – суперкласс**



Java Тема: Введение в классы и объекты

Пример программы 1.02...

```
/*
    Пример 1.02.
    Программа определяет классы Bar и RunBar
*/
package java.projects.topics.topic01;
public class Bar {
    private double a, b;

    public void setAB(double a1, double b1){
        a = a1;
        b = b1;
    }

    public double square() {
        return a*b;
    }
}
```

```
public class RunBar {  
  
    public static void main(String[] args) {  
        Bar bar = new Bar();  
        bar.setAB(3.0, 5.0);  
        double s = bar.square();  
        System.out.println("Площадь = " + s);  
    }  
}
```

Е. Сравнение объектов

Операции сравнения ссылок на объекты типа

$$a == b$$

не имеет смысла, так как сравниваются не значения, а *адреса* объектов в памяти.

Для сравнения *значений* объектов необходимо использовать соответствующие методы, например, **`equals()`**.

Java Тема: Введение в классы и объекты

Пример программы 1.03.

```
/* Сравнение строк */  
public class CompareStrings {  
    public static void main(String[] args) {  
  
        String s1, s2;  
  
        s1 = new String("Java");  
        s2 = new String("Java");  
  
        System.out.println(s1 == s2); // результат false  
        System.out.println(s1.equals(s2)); // результат true  
    }  
}
```

Г. Консоль

- **Консольное приложение** – программа с текстовым интерфейсом пользователя.
- Для **вывода** на консоль будет использоваться поток out пакета System, например `System.out.println()` ;
- Для **ввода** с консоли – поток in пакета System через классы
 - InputStreamReader, либо
 - Scanner

Java Тема: Введение в классы и объекты

Пример программы 1.04.

```
/*
    Пример 1.04.
    Программа Hello Имя! Используется Reader.
*/
import java.io.*;
public class HelloName {
    public static void main(String[] args) throws
        IOException {
        String name;
        BufferedReader reader = new BufferedReader(new
            InputStreamReader(System.in));
        System.out.println("What is you name?");
        name = reader.readLine();
        System.out.println("Hello "+name+"!");
        reader.close();
        return;
    }
}
```

Java Тема: Введение в классы и объекты

Пример программы 1.05.

```
/*
    Пример 1.05.
    Программа Привет Имя, Возраст! Используется Scanner
*/
import java.util.*;
public class HelloNameScanner {
    public static void main(String[] args) {
        Scanner inp = new Scanner(System.in);
        String name;
        int age;

        System.out.println("Как вас зовут?");
        name = inp.nextLine();

        System.out.println("Сколько вам лет?");
        age = inp.nextInt();

        System.out.println("Привет, "+name+", "+age+" лет!");
        scanner.close();
    }
}
```

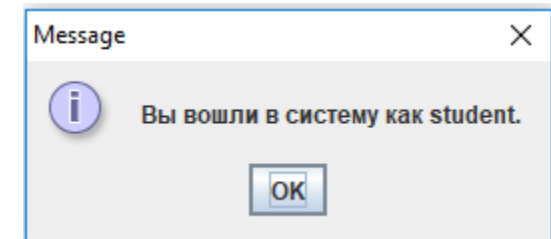
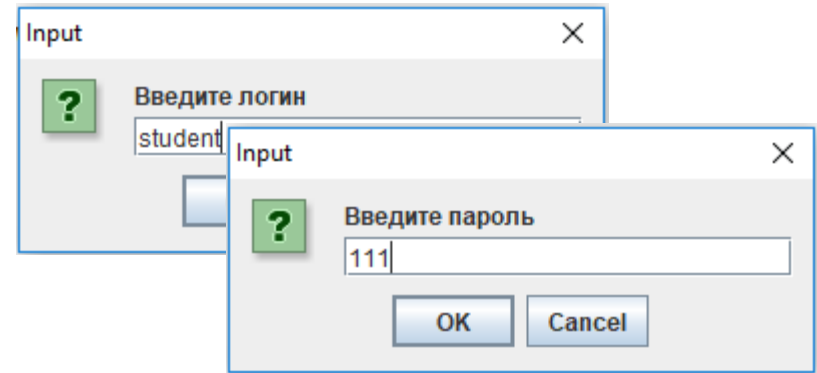
Java Тема: Введение в классы и объекты

Пример программы 1.06.

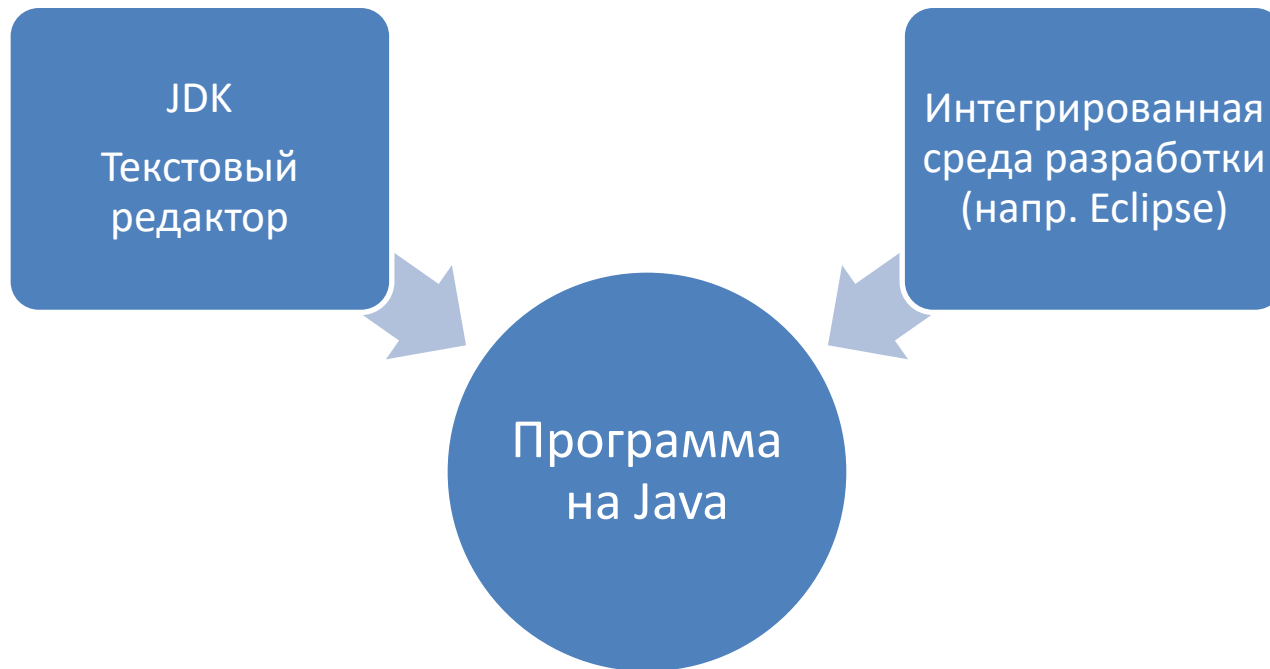
```
/* Пример 1.06. Ввод логина и пароля, сравнение с заданными
   Использование элементов графического интерфейса библиотеки Swing
*/
import javax.swing.*;

public class MyClass {
    public static void main(String[] args)
    {
        String login, password;
        login=JOptionPane.showInputDialog("Введите логин");
        password = JOptionPane.showInputDialog("Введите пароль");
        String userLogin = "student";
        String userPassword = "111";

        if ((login.equals(userLogin)) && (password.equals(userPassword)))
            JOptionPane.showMessageDialog(null, "Вы вошли в систему как "
                +login +".");
        else
            JOptionPane.showMessageDialog(null,
                "Логин или пароль не корректны.");
    }
}
```



G. Обзор среды разработки Eclipse



Рекомендуется для самоподготовки использовать видеоуроки,
например на <http://itlift.ru>



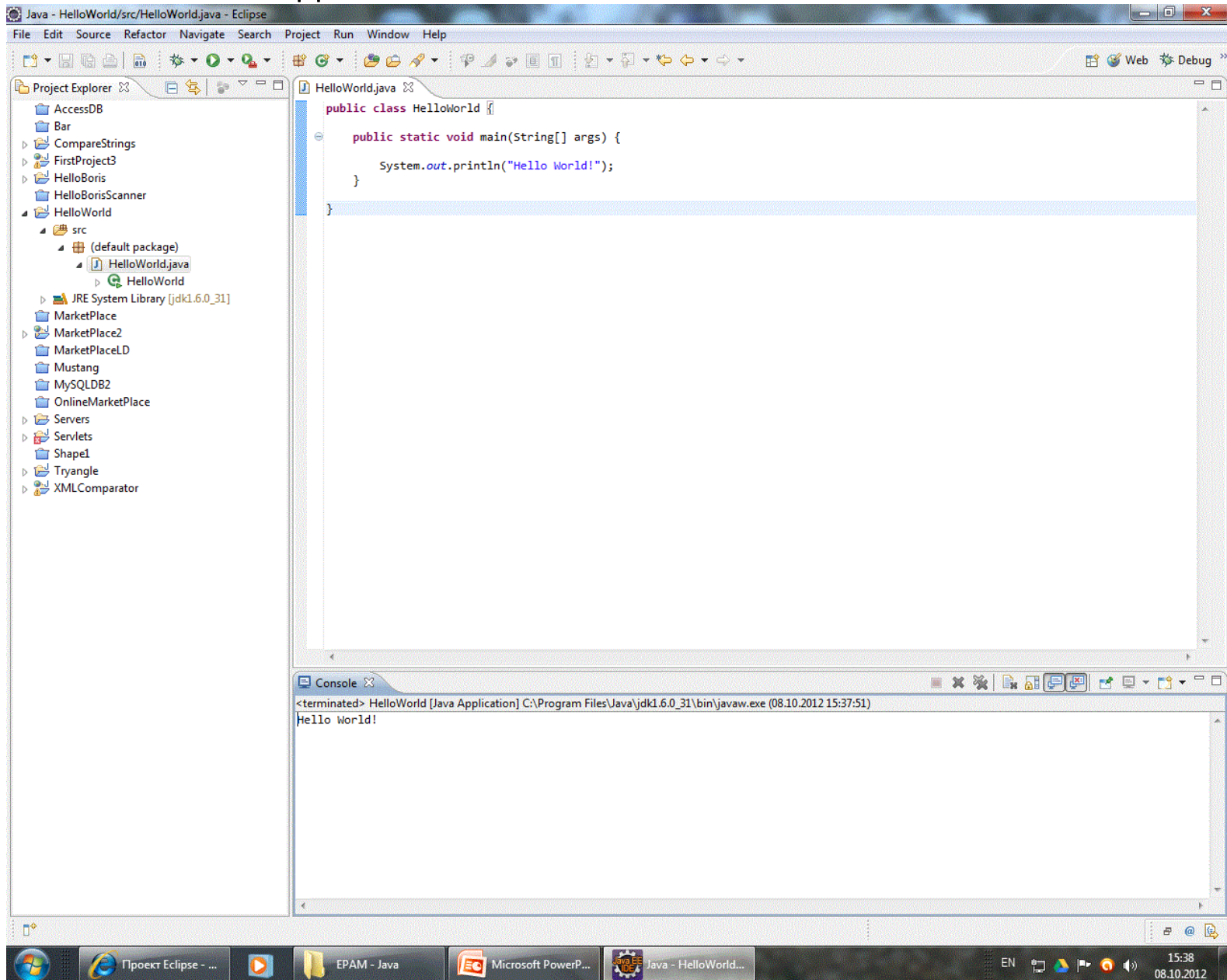
полнофункциональная индустриальная
платформа коммерческого качества для
разработки сильно-интегрированных
приложений.

- | | |
|--|---|
| <ul style="list-style-type: none">• создание кода на ряде языков• независимость от языка• редактор с подсветкой синтаксиса• инкрементальная компиляция кода,• потокобезопасный отладчик• навигатор по классам• менеджеры файлов и проектов | <ul style="list-style-type: none">• рефакторинг кода• автоматическое обновление/установка кода• список текущих задач• intellisense - автоподсказка• и др. |
|--|---|

- Понятия:

- **рабочая область** (*workspace*) - это прежде всего контейнер для проектов. Кроме этого, в ней содержатся настройки, сохраняемые компонентами Eclipse, и вспомогательные данные
- **проект** (*project*) - набор ресурсов (файлов и папок), правил для их обработки и других свойств проекта
- **билдеры** (*builders*) –компоненты, обрабатывающие ресурсы проекта
- **тип проекта** (*nature*) определяет множество билдеров, которые могут работать с ресурсами, входящими в проект
- **перспектива** (*perspective*) - конфигурация платформы, соответствующая определенной роли. Функциональность, необходимая выступающему в этой роли пользователю, определяет набор и расположение окон, меню, горячих клавиш, и прочих элементов интерфейса.

Java Тема: Введение в классы и объекты



Типичный интерфейс среды:

1. Project Explorer

- иерархическая структура проекта, включая папки с файлами с исходным java-кодом, HTML, JSP, JavaScript, библиотеки, настройки и др.

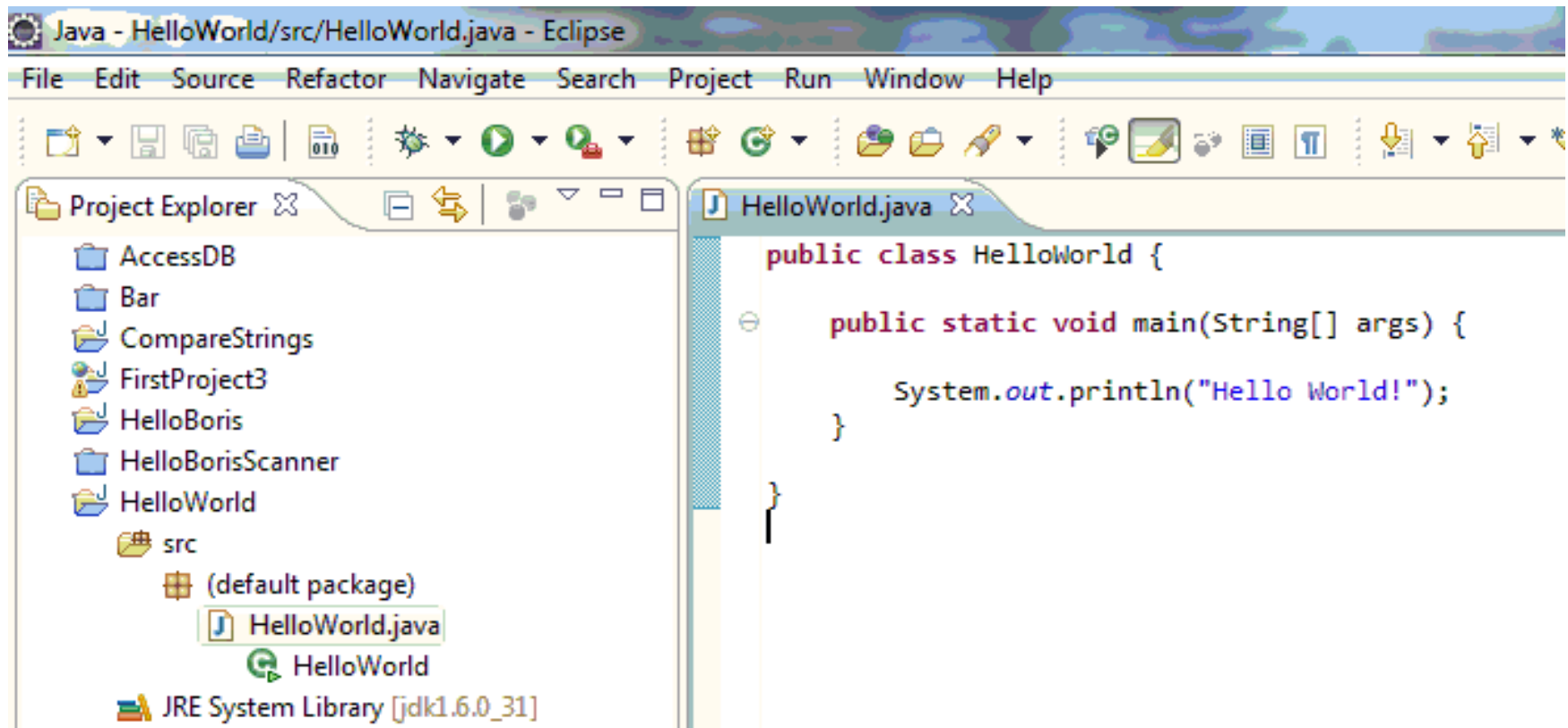
2. Окна редактора

- Редактор обладает функциональностью: подсветка ключевых слов, ассистент кода (поиск ошибок в коде и предложение вариантов их разрешения), автозавершение кода, сворачивание/разворачивание фрагментов кода и др.

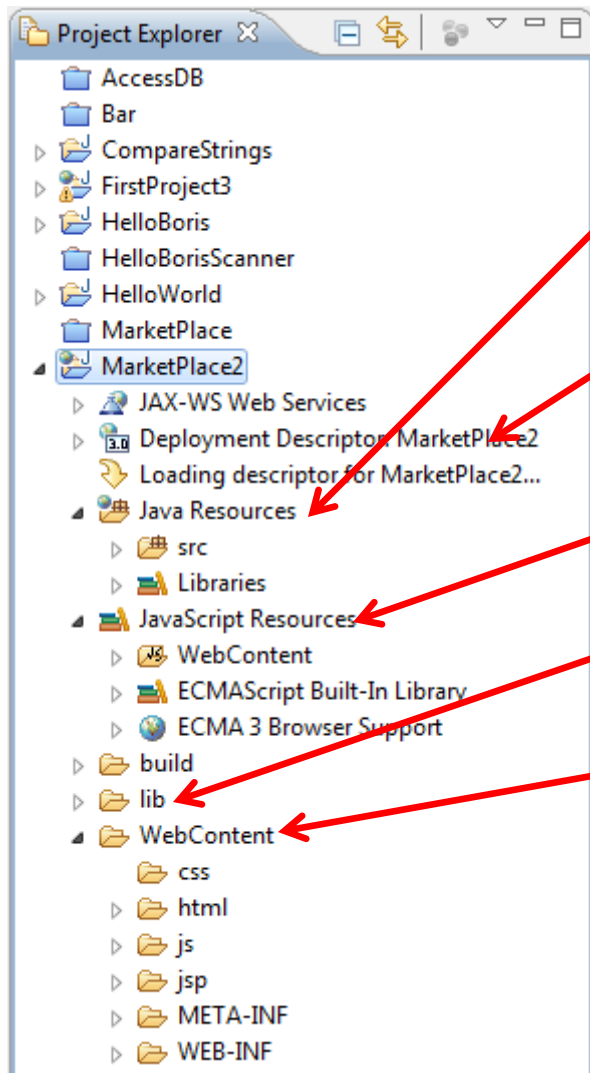
3. Окна с отладочной информацией

- содержит вкладки Problems, Console и др.

Java Тема: Введение в классы и объекты



Java Тема: Введение в классы и объекты



Типичный проект содержит:

JavaResources

- src – исходный код
- Libraries – подкл.библиотеки

Deployment Descriptor – дескриптор развертывания

JavaScript Resources

lib

WebContent – страницы и Web-ресурсы

- css
- html
- js
- jsp
- META-INF
- WEB-INF

Задания к Теме 2:

1. Создать программу для ввода пароля и сравнения его со строкой-образцом.
2. Создать программу ввода двух целых чисел, подсчета их суммы и произведения и вывода результата на консоль.
3. Создать программу, вводящую целое положительное число n и выводящее все целые числа, которые являются делителями числа n .
4. Создать программу, вводящую целое положительное число n и выводящую разложение этого числа на простые множители.
5. Создать программу, вводящую целое положительное число n и выводящее все целые числа, не превосходящие n , в десятичной записи которых все цифры разные.