

Java. Введение

JDK, JRE, JVM, IDE

История создания и эволюция языка Java

Sun Microsystems (www.sun.com) – ныне часть компании Oracle (www.oracle.com)



Patrick Naughton



James Gosling



Scott McNealy

Январь, 1991 г. Начало разработки нового языка программирования.

Главная причина недовольства C++:

Необходимо перекомпилировать, а зачастую и переписывать код под каждую новую архитектуру микропроцессора, операционную систему и т.п.

История создания и эволюция языка Java



Сентябрь, 1992 г.
Оак (дуб)



23 марта 1995 г.
Официальное объявление языка Java

1999 г. - Java 2 SE – Java 2 Standard Edition

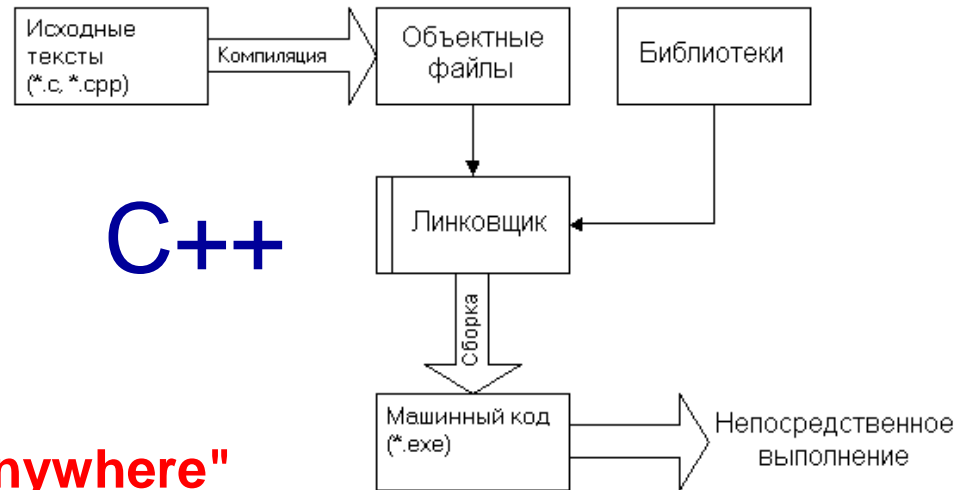
Ключевые особенности языка Java

- архитектурная независимость и переносимость кода
- полная объектная ориентированность
- устойчивость (надежность) кода
- встроенный механизм поддержки многопоточности
- безопасность Java-программ
- встроенная структура коллекций
- удобство разработки GUI

Архитектурная независимость и переносимость кода

"Write Once, Run Anywhere"

C++

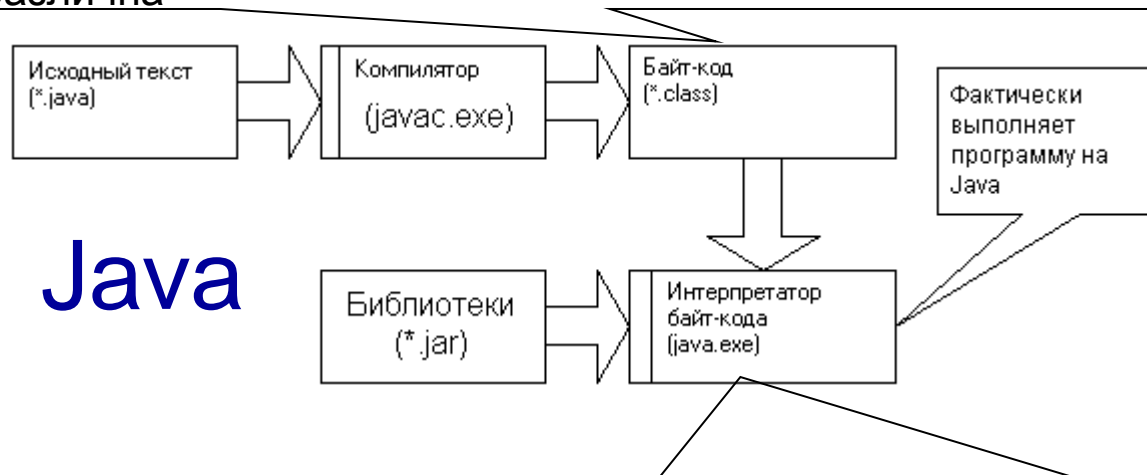


Java



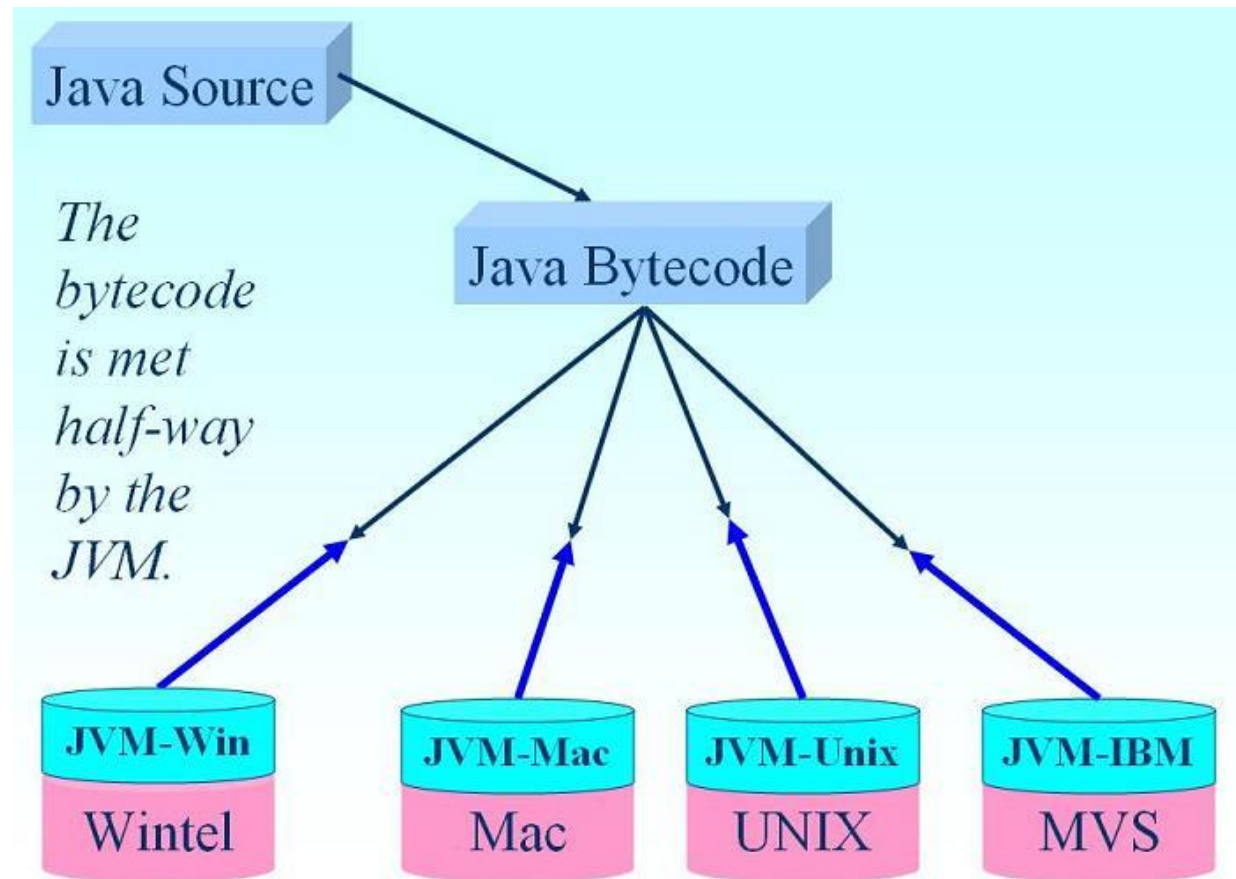
Архитектурная независимость и переносимость кода

Машинно-независимый код низкого уровня, генерируемый транслятором и исполняемый интерпретатором. Большинство инструкций байт-кода эквивалентны одной или нескольким командам ассемблера. Трансляция в байт-код занимает промежуточное положение между компиляцией в машинный код и интерпретацией. **Байт-код** называется так, потому что длина каждого кода операции — один байт, но длина кода команды различна



JVM (Java Virtual Machine, виртуальная Java-машина) – исполняющая система, интерпретирующая байт-код. Абстрактное вычислительное устройство, которое может быть реализовано разными способами: аппаратно или программно. Компиляция в набор команд виртуальной машины происходит почти так же, как и компиляция в набор команд микропроцессора.

Архитектурная независимость и переносимость кода



Java Virtual Machine (JVM)

- виртуальная машина Java — основная часть исполняющей системы Java, т.н. Java Runtime Environment (JRE).
- Виртуальная машина Java интерпретирует и исполняет байт-код Java, предварительно созданный из исходного текста Java-программы компилятором Java (javac).
- JVM ключевым компонентом платформы Java.
- Использование одного байт-кода для многих платформ позволяет запускать приложения, написанные на java, на разных платформах.

Java Runtime Environment (JRE)

- исполнительная среда Java в которой выполняются программы, написанные на этом языке.
- Среда состоит из JVM и библиотеки Java классов.
- Это минимальная реализация JVM, необходимая для исполнения Java приложений, без компилятора и других средств разработки.
- Именно JRE или его аналог других фирм используется в браузерах, умеющих выполнять программы на Java, операционных системах и системах управления базами данных.
- Установка JRE является необходимым и достаточным условием для выполнения Java программ.
- Для разработки программ JRE недостаточно, необходимо установить Java Development Kit (JDK), который может установить и JRE и дополнительные компоненты.

Полная объектная ориентированность

Основная структурная единица программы – класс, весь код Java-программы должен находиться внутри одного или нескольких классов.

```
// HelloWorld.java Our first Java Application  
  
public class HelloWorld  
{  
    public static void main( String args[])  
    {  
        System.out.println( "Hello World!" );  
    }  
}
```

Устойчивость (надежность) кода

- отсутствие адресной арифметики
- технология «сборки мусора» (garbage collection)
- строгая типизация
- отсутствие множественного наследования классов
- запрет перегрузки операторов
- встроенная обработка исключений

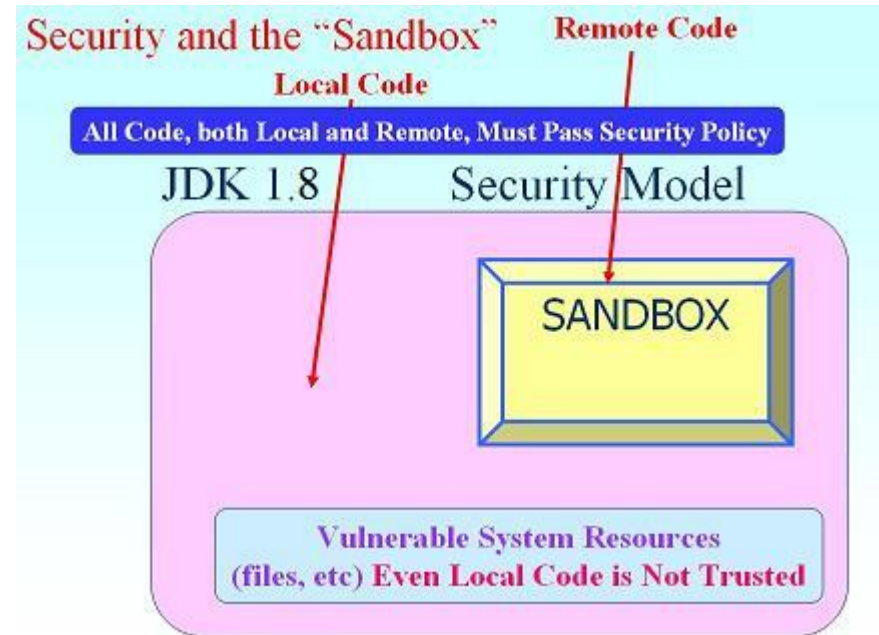
«Java is C++ without the Guns and Knives» (C) J.Gosling

Поддержка многопоточности

Многопоточность Java предоставляет средства создания приложений с множеством одновременно активных потоков. Для эффективной работы с потоками в Java реализован механизм семафоров и средств синхронизации потоков: библиотека языка предоставляет класс Thread, а система выполнения предоставляет средства диспетчеризации и средства, реализующие семафоры.

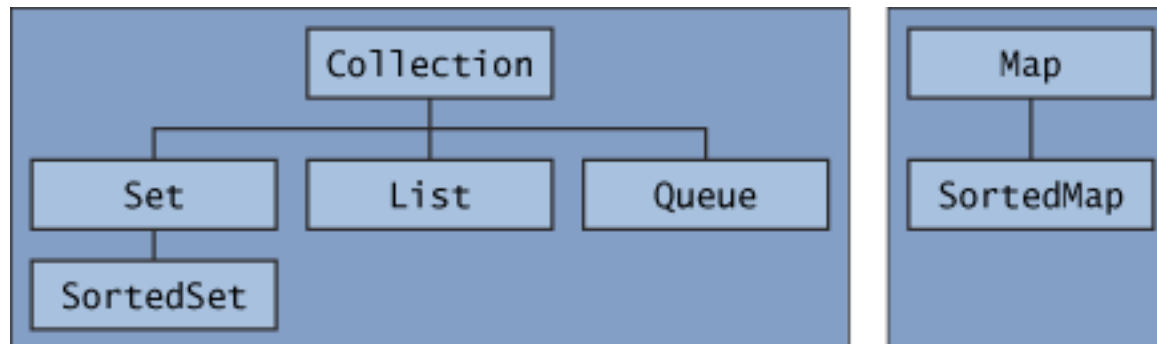
Безопасность

JVM - аналог виртуального компьютера, расположенного в оперативной памяти и интерпретирующего байткод. Все действия Java-программы замкнуты внутри этого виртуального компьютера. JVM может не допускать деструктивных действий Java-программ.



Встроенная структура коллекций

Структура коллекций (collections framework)
Java стандартизирует способ, с помощью
которого программы хранят и
обрабатывают структуры данных.



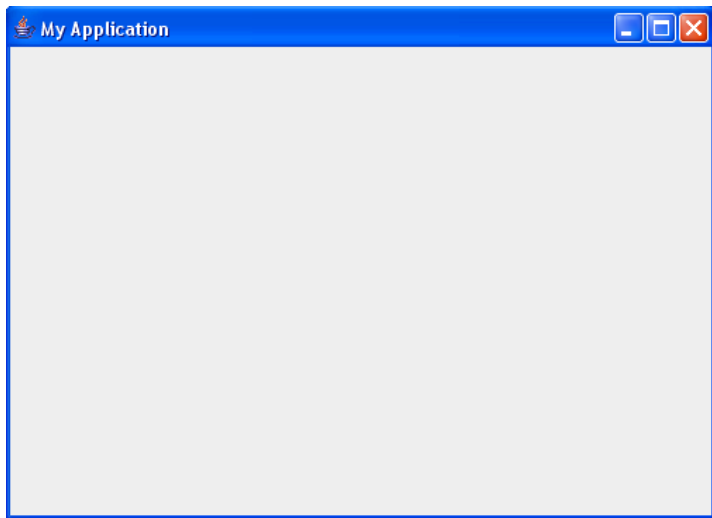
Встроенная структура коллекций

Преимущества использования структуры коллекций:

1. Избавление от рутинных операций по кодированию стандартных структур данных и алгоритмов
2. Высокая эффективность реализации
3. Универсальность и простота изучения (различные типы коллекций работают похожим друг на друга образом и с высокой степенью способности к взаимодействию)
4. Расширяемость
5. Параметризация

Удобство разработки GUI

В состав Java входят 2 библиотеки, предназначенных для разработки GUI:



- AWT (Abstract Window Toolkit) – платформно-зависимая библиотека, вывод осуществляется через вызовы OS API
- Swing - платформно-независимая библиотека, реализованная полностью на Java, через OS API выводится только окно, все остальное рисуется средствами Java

Инструментальные средства

Большая часть инструментария для разработки
Java-программ распространяется бесплатно!

1. **Java Software Development Kit - Oracle**
Текущая версия Java^(TM) SE Development Kit 7
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>
2. **RAD – средства разработки:**
 - **Eclipse Project (open source project)**
 - **NetBeans (open source project)**
 - **IntelliJ IDEA (JetBrains)**
 - **JCreator Pro (Xinox Software)**
 - **Symantec Cafe (Symantec)**
 - **Visual J++ (Microsoft)**
 - **Together (TogetherSoft Corporation)**

Первая программа на Java

```
class FirstProg
{public static void main(String args[ ])
    {
        System.out.println ("Hello, world");
    }
}
```

1. `>javac FirstProg.java` -> **FirstProg.class**

2. `>java FirstProg`

```
C:\Program Files\Java\jdk1.6.0_08\bin>java FirstProg
Hello, World!
```

Java 2 Standart Edition (J2SE)

- стандартная редакция языка Java, используемая для разработки простых Java приложений.
- Используя данную редакцию можно создавать апплеты, консольные приложения, приложения с графическим интерфейсом пользователя.

Java 2 Enterprise Edition (J2EE)

- редакция языка Java для разработки распределенных приложений масштаба предприятия.
- Включает в себя технологию Enterprise Java Beans (EJB), Java Server Pages (JSP) и сервлеты (Servlets).
- Каждая из этих технологии, в свою очередь также имеет свой отдельный номер версии.
- Кроме того, Java EE включает в себя спецификацию на разработку корпоративных приложений, согласно которой следует строить подобные приложения.
- На данный момент J2EE и .Net сейчас два основных соперника на рынке решений для разработки корпоративных приложений.

Java 2 Micro Edition (J2ME)

- редакция языка Java для разработки приложений для микрокомпьютеров (мобильных телефонов, Palm и т.д.).
- В нее входят "облегченные" стандартные классы и классы для написания мидлетов (Midlets).
- Мидлеты – это аналоги апплетов, но только приспособленные специально для небольших устройств. В них также поддерживается графика, звук, реакция на события.
- Java ME наиболее полно соответствует начальному предназначению Java – платформы для написания программ для бытовых устройств.

Java Development Kit (JDK)

Бесплатно распространяемый корпорацией **Sun** комплект разработчика приложений на языке Java, включающий в себя:

- компилятор Java (javac),
- стандартные библиотеки классов Java,
- примеры,
- документацию,
- различные утилиты,
- исполнительную систему Java (JRE).

Набор программ и классов JDK

- компилятор из исходного текста в байт-коды `javac`;
- интерпретатор `java`, содержащий реализацию JVM;
- облегченный интерпретатор `jre` (в последних версиях отсутствует);
- программу просмотра апплетов `appletviewer`, заменяющую браузер;
- отладчик `jdb`;
- дизассемблер `javap`;
- программу архивации и сжатия `jar`;
- программу сбора и генерирования документации `javadoc`;
- программу генерации заголовочных файлов языка C для создания «родных» методов `javah`;
- программу добавления электронной подписи `javakey`;
- программу `native2ascii`, преобразующую бинарные файлы в текстовые;
- программы `rmic` и `rmiregistry` для работы с удаленными объектами;
- программу `serialver`, определяющую номер версии класса;
- библиотеки и заголовочные файлы «родных» методов;
- библиотеку классов Java API (Application Programming Interface).

Схема

Java Development Kit
JDK

Java Runtime Environment
JRE

Java Virtual Machine
JVM

Java-программа

- Состоит из одного или нескольких определений классов, размещенных в одном или нескольких файлах с расширением .java.
- Для компиляции программ используется java-компилятор javac, в результате для каждого класса из исходного файла создается файл класса, содержащий байт-коды класса.
- Имя файла класса совпадает с именем класса, к ней добавляется суффикс .class.
- Один из классов программы должен быть открытым (public) классом и содержать метод main, с которого начинается выполнение программы.

Выполнение Java программы

- Программы, предназначенные для запуска на JVM должны быть скомпилированы в стандартизированном переносимом двоичном формате в виде файлов с расширением .class.
- Программа может состоять из множества классов, размещенных в различных файлах.
- Для облегчения размещения больших программ, часть файлов вида .class могут быть упакованы вместе в .jar файл (сокр. от JavaArchive).
- Виртуальная машина JVM исполняет файлы .class или .jar, эмулируя инструкции, написанные для JVM, путем интерпретирования или использования just-in-time компилятора (JIT).

Integrated Development Environment (IDE)

- В состав JDK не входит интегрированная среда разработки на Java (IDE), поэтому разработчик, использующий только JDK, вынужден использовать внешний текстовый редактор и компилировать свои программы, используя утилиты командной строки.
- Интегрированные среды разработки на Java:
 - Eclipse
 - NetBeans,
 - IntelliJ IDEA,
 - Sun Java Studio Creator,
 - Borland JBuilder.

ClassLoader

- Абстрактный класс
- Каждый раз, когда загружается какой-либо .class-файл, например, вследствие обращения к конструктору или статическому методу соответствующего класса – на самом деле это действие выполняет один из наследников класса `ClassLoader`.
- Существует стандартный вариант реализации `ClassLoader` – так называемый системный загрузчик классов. Этот загрузчик используется по умолчанию при запуске приложений Java командой:

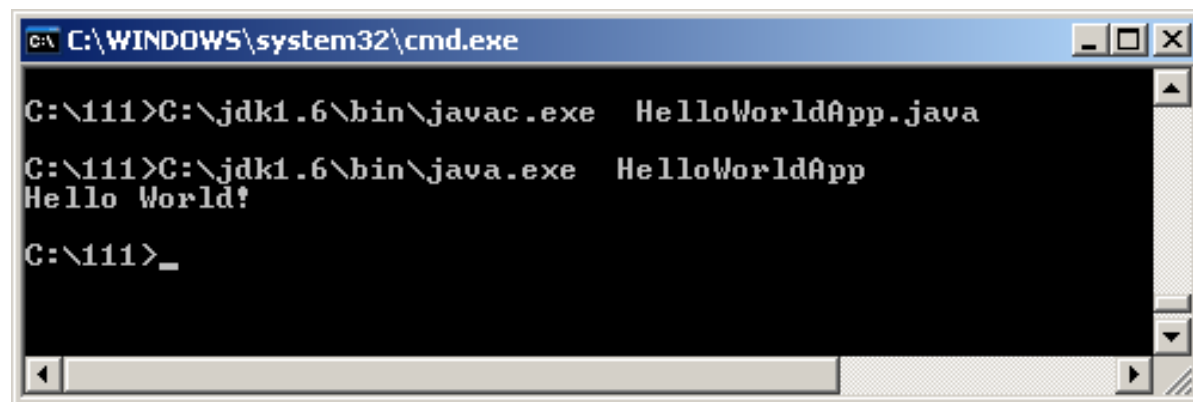
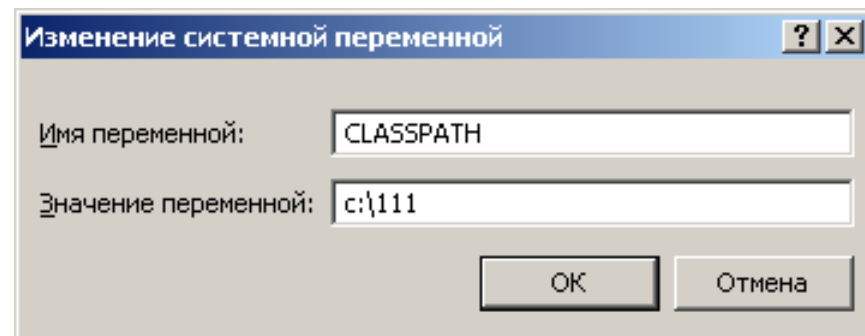
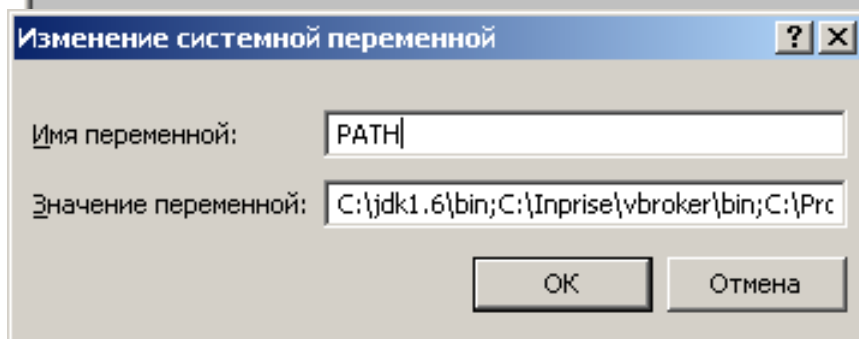
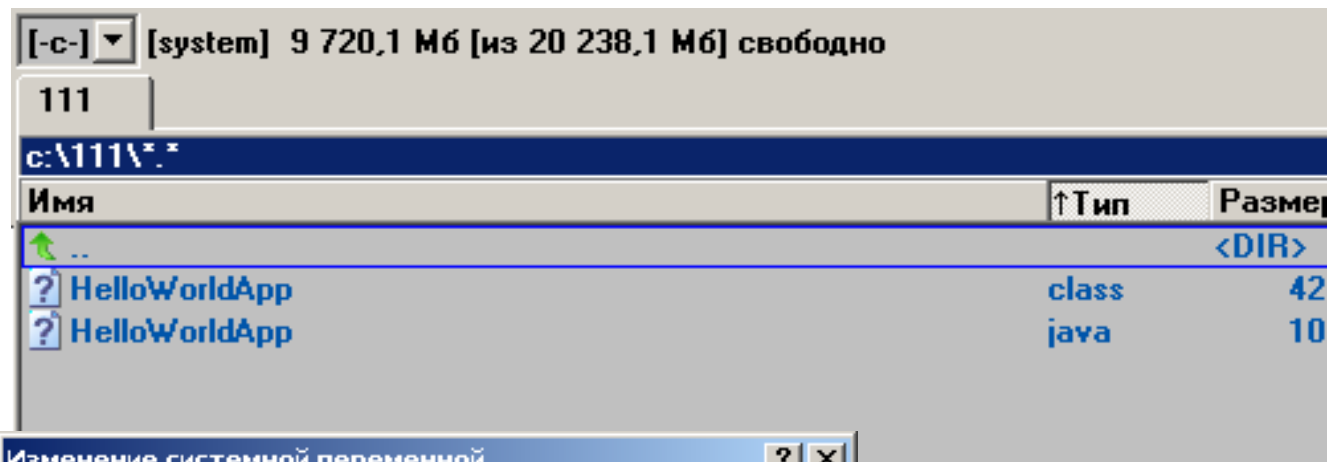
```
java Имя_главного_класса
```

Системный загрузчик классов

- Системный загрузчик классов реализует стандартный алгоритм загрузки из каталогов и JAR-файлов, перечисленных в переменной окружения CLASSPATH, а также из JAR-файлов, содержащих стандартные системные классы
- В Java можно реализовать свой собственный загрузчик классов – наследник `ClassLoader` – и использовать его вместо системного.
- При запуске приложения Java с помощью стандартной команды виртуальная машина Java первым делом создает системный загрузчик, загружает с его помощью `.class`-файл главного класса и вызывает статический метод класса, соответствующий объявлению
`public static void main(String[] argv)`
(или же сообщает об ошибке, не обнаружив такого метода).

Отложенная загрузка кода

- Java – язык с отложенной загрузкой кода. Первоначально загружается только один класс – тот, который передан в качестве параметра утилите «java» (на самом деле, вначале загружается целый ряд системных классов, в частности, системный загрузчик и все используемые ими классы).
- Как только код этого класса обращается к какому-то другому классу (любым способом: вызовом конструктора, обращением к статическому методу или полю), загружается другой класс.
- По мере выполнения кода, загружаются всё новые и новые классы. Ни один класс не загружается до тех пор, пока в нем не возникнет реальная потребность. (Такое поведение заложено в стандартный системный загрузчик.)



PATH & CLASSPATH

Переменные окружения

`PATH=%PATH%;c:\jdk1.8\bin`

путь указывает на месторасположение файлов
javac.exe и **java.exe**.

`CLASSPATH=.;d:\users\user\workspace\`

каталог, который компилятор Java будет
рассматривать как корневой для иерархии
пакетов

Литература и источники информации в сети Интернет

1. Ноутон П., Шилдт Г. Java 2 в подлиннике. – СПб, «БНВ», 2001.
 2. Г.Шилдт. Полный справочник по Java. – М.: Вильямс, 2007.
 3. Б. Эккель. Thinking in Java, 4th ed. – Спб.:Питер, 2009.
 4. Joshua Bloch. Effective Java: Second Edition. – Prentice Hall, 2008.
 5. Майкл Морган. Java 2. Руководство разработчика. Издательский дом "Вильямс". Москва. 2000 г.
 6. Horstmann C., Cornell G. - Core Java. Vol. I-II, Fundamentals, 8 ed .
 7. R. Lafore. Data Structures and Algorithms in Java, 2nd ed.
 8. М.Гранд. Шаблоны проектирования в JAVA. Каталог популярных шаблонов проектирования, проиллюстрированных при помощи UML. – М.: Новое знание, 2004.
- <http://docs.oracle.com/javase/tutorial/>
 - <http://www.intuit.ru/departement/pl/javapl/>