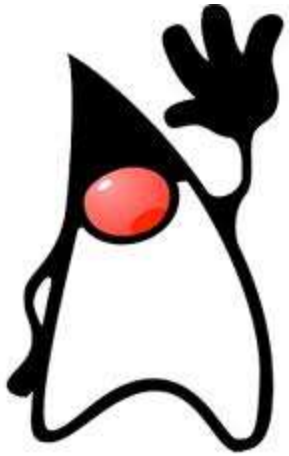
Program Outline

- History - Java
- Object Oriented Programming concepts
- Primitive Data Types
- Wrapper Classes
- Inheritance



History

Toolkits / Frameworks / Object APIs (1990s–Up)				
Java 2 SDK	AWT / J.F.C./Swing	Jini™	JavaBeans™	JDBC™

Object-Oriented Languages (1980s–Up)					
SELF	Smalltalk	Common Lisp Object System	Eiffel	C++	Java

Libraries / Functional APIs (1960s–Early 1980s)				
NASTRAN	TCP/IP	ISAM	X-Windows	OpenLook

High-Level Languages (1950s–Up)				Operating Systems (1960s–Up)			
Fortran	LISP	C	COBOL	OS/360	UNIX	MacOS	Microsoft Windows

Machine Code (Late 1940s–Up)

About Java

- Java programming language was originally developed by Sun Microsystems
  - by James Gosling
  - released in 1995 as core component of Sun Microsystems
  - Java platform (Java 1.0 [J2SE]).
  - Latest release
    - Java Standard Edition 7 Update 25 (1.7.25) (June 18, 2013; 52 days ago)
- Multiple Editions

  - Java Card for smartcards.
  - Java Platform, Micro Edition (Java ME) — targeting environments with limited resources.
  - Java Platform, Standard Edition (Java SE) — targeting workstation environments.
  - Java Platform, Enterprise Edition (Java EE) — targeting large distributed enterprise or Internet environments.

- Sun Microsystems has renamed the new J2 versions as Java SE, Java EE and Java ME respectively.

"Write Once, Run Anywhere"

Features of Java

- **Object Oriented** :
  - In java everything is an Object.
  - Java can be easily extended since it is based on the Object model.

- **Platform independent**:
  - Unlike many other programming languages including C and C++ when Java is compiled, it is compiled into platform independent byte code.

- **Simple** :
  - Java is designed to be easy to learn.

- **Secure**
  - Compiler , Class loader , Byte code verifier , Sandbox

- **Architectural- neutral** :
  - Java compiler generates an architecture-neutral object file format executable on many processors, with the presence Java runtime system.

- **Portable** :
  - being architectural neutral and having no implementation dependent aspects of the specification makes Java portable.

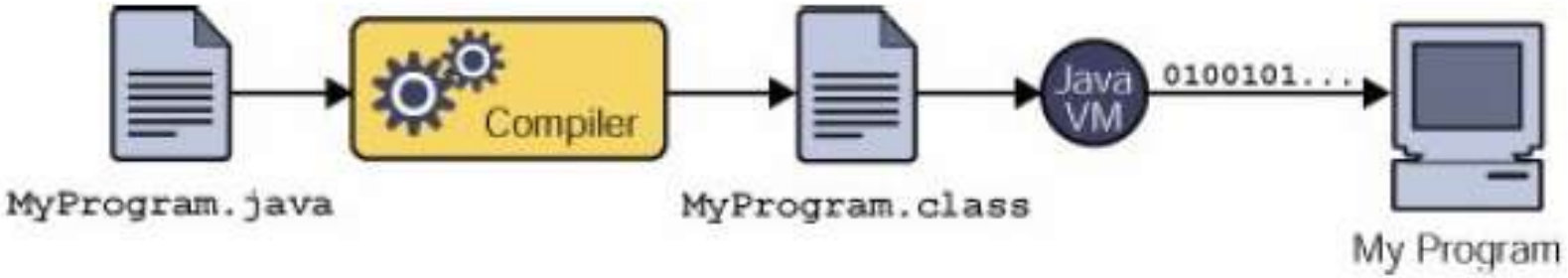
Contd…

- **Robust** :
  - Java makes an effort to eliminate error prone situations by emphasizing mainly on compile time error checking and runtime checking.
- **Multi-threaded** :
  - With Java's multi-threaded feature it is possible to write programs that can do many tasks simultaneously.
- **Interpreted** :
  - Java byte code is translated on the fly to native machine instructions and is not stored anywhere.
- **High Performance**:
  - With the use of Just-In-Time compilers Java enables high performance.
- **Distributed** :
  - Java is designed for the distributed environment of the internet.
- **Dynamic** :
  - Java programs can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.

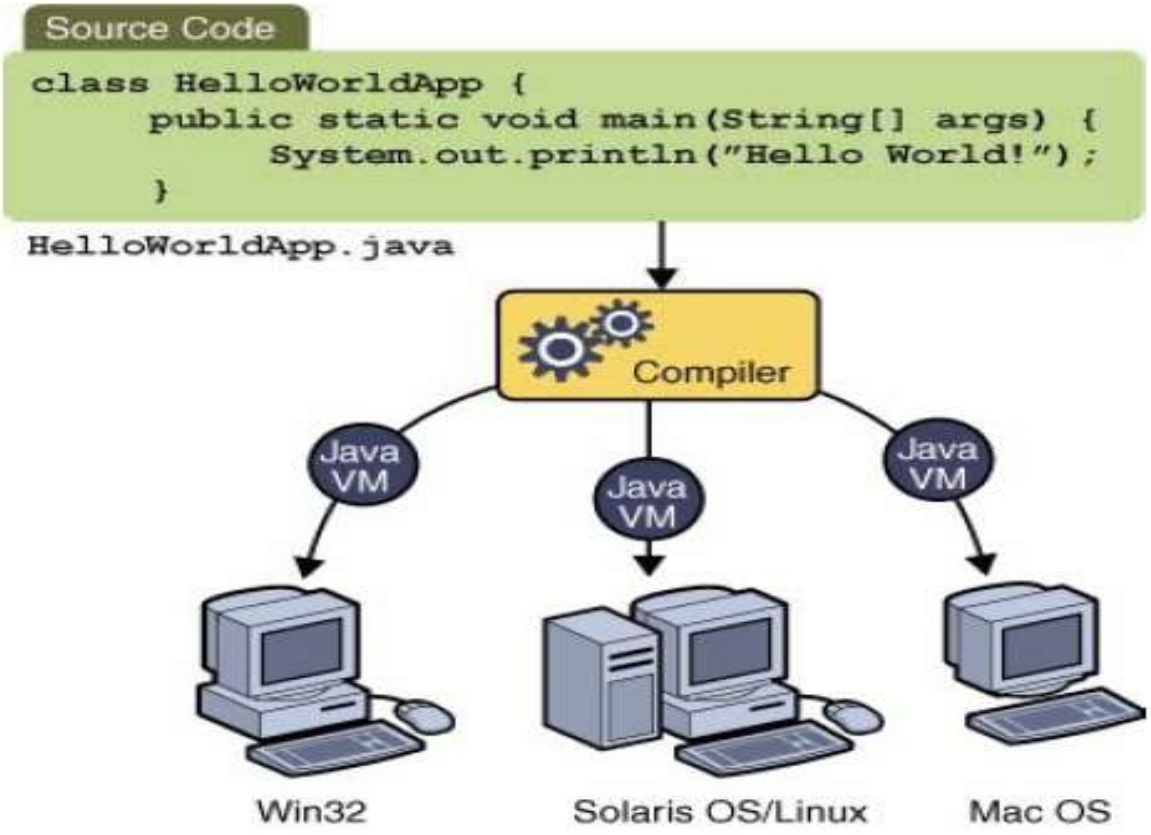
Java Program Lifecycle

- Java programs normally undergo four phases
  - Edit
    - Programmer writes program (and stores program on disk)
  - Compile
    - Compiler creates *byte codes* from program (.class)
  - Load
    - Class loader stores byte codes in memory
  - Execute
    - Interpreter: translates *byte codes* into machine language

Java Program Lifecycle



JVM - Portability



Object Oriented Programming

- Encapsulation
- Abstraction
- Polymorphism
- Inheritance

Encapsulation

- Hiding the irrelevant information
  - Class
    - State & Behavior
  - Access Specifiers
    - Private
  - Hides the implementation details of a class
  - Forces the user to use an interface to access data
  - Makes the code more maintainable

Abstraction

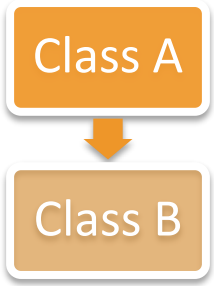
- Abstraction means to show only the necessary details to the client of the object.
  - Public methods
  - Perspective based

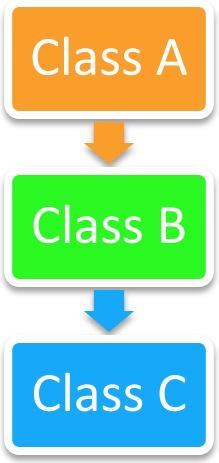
Access Specifiers

Modifier	Same Class	Same Package	Subclass	Universe
`private`	Yes			
default	Yes	Yes		
`protected`	Yes	Yes	Yes	
`public`	Yes	Yes	Yes	Yes

Inheritance

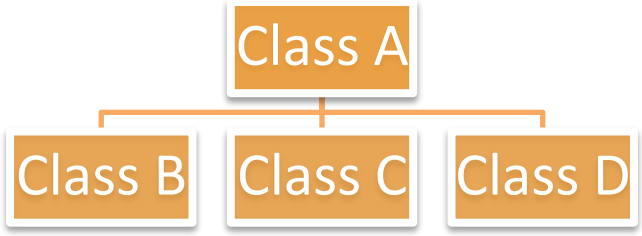
Single Inheritance





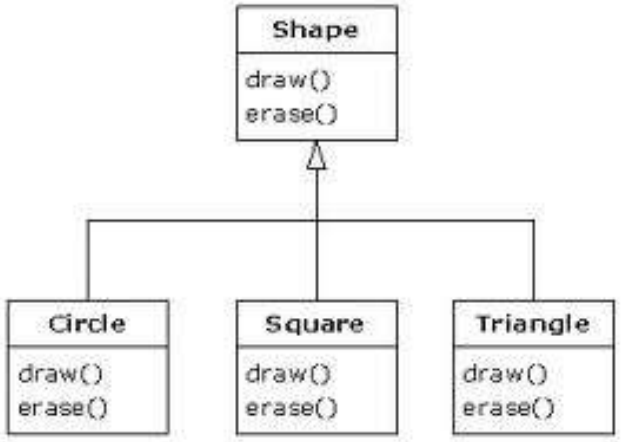
Multilevel Inheritance

Hierarchical Inheritance

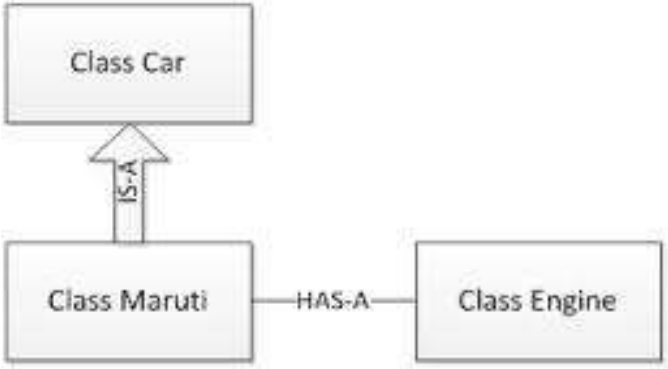


Relationship

- Is-a
  - Mammal IS-A Animal
  - Reptile IS-A Animal
  - Dog IS-A Mammal



- Has-a



Polymorphism

- Taking multiple forms
  - Compile time Polymorphism
    - Over Loading
  - Runtime Polymorphism
    - Overriding

Primitive Data Types

The Java programming language defines eight primitive types:

- Logical – boolean
- Textual – char
- Integral – byte, short, int, and long
- Floating – double and float

Logical –boolean

The boolean primitive has the following characteristics:

- The boolean data type has two literals, true andfalse.
- For example, the statement:

boolean truth = true;

declares the variable truth as boolean type and assigns it a value of true.

Textual –char

The textual char primitive has the following characteristics:

- Represents a 16-bit Unicode character
- Must have its literal enclosed in single quotes (’ ’)
- Uses the following notations:

'a'	The letter a
'\t'	The tab character
'\u????‘	A specific Unicode character, ????, is replaced with exactly four hexadecimal digits .

For example, ’\u03A6’ is the Greek letter phi [Φ].

Integral –byte,short,int, andlong

The integral primitives have the following characteristics:

- Integral primates use three forms: Decimal, octal, or hexadecimal
- Literals have a default type of int.
- Literals with the suffix L or l are of type long.

Integer Length	Name or Type	Range
8 bits	byte	-2^7 to 2^7-1
16 bits	short	-2^{15} to $2^{15}-1$
32 bits	int	-2^{31} to $2^{31}-1$
64 bits	long	-2^{63} to $2^{63}-1$

Floating Point –floatanddouble

The floating point primitives have the following characteristics:

- Floating-point literal includes either a decimal point or one of the following:
- E or e (add exponential value)
- F or f (float)
- D or d (double)

Examples:

3.14	A simple floating-point value (a double)
6.02E23	A large floating-point value
2.718F	A simple float size value
123.4E+306D	A large double value with redundant D

Float Length	Name or Type
32 bits	`float`
64 bits	`double`

Primitive Data Types contd..

- byte:
  - Byte data type is a 8-bit signed two's complement integer.
  - Minimum value is -128 (-2^7)
  - Maximum value is 127 (inclusive)(2^7 -1)
  - Default value is 0
  - Byte data type is used to save space in large arrays, mainly in place of integers, since a byte is four times smaller than an int.
  - Example : byte a = 100 , byte b = -50

- short:
  - Short data type is a 16-bit signed two's complement integer.
  - Minimum value is -32,768 (-2^15)
  - Maximum value is 32,767(inclusive) (2^15 -1)
  - Short data type can also be used to save memory as byte data type. A short is 2 times smaller than an int
  - Default value is 0.
  - Example : short s= 10000 , short r = -20000

- int:
  - Int data type is a 32-bit signed two's complement integer.
  - Minimum value is - 2,147,483,648.(-2^31)
  - Maximum value is 2,147,483,647(inclusive).(2^31 -1)
  - Int is generally used as the default data type for integral values unless there is a concern about memory.
  - The default value is 0.
  - Example : int a = 100000, int b = -200000

- long:
  - Long data type is a 64-bit signed two's complement integer.
  - Minimum value is -9,223,372,036,854,775,808.(-2^63)
  - Maximum value is 9,223,372,036,854,775,807 (inclusive). (2^63 -1)
  - This type is used when a wider range than int is needed.
  - Default value is 0L.
  - Example : long a = 100000L, int b = -200000L

- float:
  - Float data type is a single-precision 32-bit IEEE 754 floating point.
  - Float is mainly used to save memory in large arrays of floating point numbers.
  - Default value is 0.0f.
  - Float data type is never used for precise values such as currency.
  - Example : float f1 = 234.5f

- double:
  - double data type is a double-precision 64-bit IEEE 754 floating point.
  - This data type is generally used as the default data type for decimal values. generally the default choice.
  - Default value is 0.0d.
  - Example : double d1 = 123.4

- boolean:
  - boolean data type represents one bit of information.
  - There are only two possible values : true and false.
  - This data type is used for simple flags that track true/false conditions.
  - Default value is false.
  - Example : boolean one = true

- char:
  - char data type is a single 16-bit Unicode character.
  - Minimum value is '\u0000' (or 0).
  - Maximum value is '\uffff' (or 65,535 inclusive).
  - Char data type is used to store any character.
  - Example . char letterA ='A'

Reference Data Types

- Reference variables are created using defined constructors of the classes.
- Class objects, and various type of array variables come under reference data type.
- Default value of any reference variable is null.
- A reference variable can be used to refer to any object of the declared type or any compatible type.
  - Example : Animal animal = new Animal("giraffe");

Inheritance

- extends - keyword
- Single , Multilevel, Hierarchical – supported

Thank You