INTRODUCTION TO JAVA

By,

Hitha Paulson

Assistant Professor, Dept. of Computer Science

LF College, Guruvayoor

- ✓ Invented by **James Gosling**
- ✓ at **SUN Microsystems**
- ✓ Released in **1995**
- ✓ Earlier it was named as **Oak**

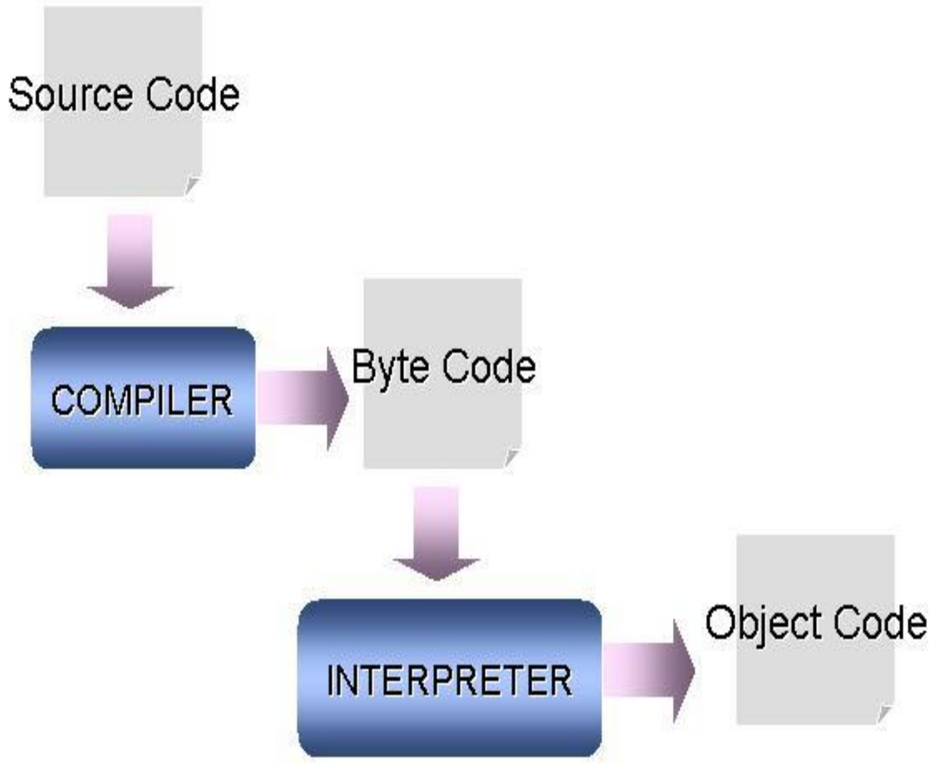
Birth of Java

- Motivation
  - Platform independent language for Embedded systems
  - Language that can use with Internet
- Developed by
  - Green Team led by James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan
- Developed at
  - Sun Microsystems
- Initially called
  - Oak in 1991
- Renamed as
  - JAVA in 1995

JAVA Programming Language

- ✓ Syntax borrowed from C language
- ✓ Fully object oriented language
- ✓ OOP concept derived from C++
- ✓ Compiled and Interpreted
- ✓ Platform Independent

Byte code and class files



- ✓ **Source Code:** .java file
- ✓ **Byte Code:** Intermediate code generated after compilation (.class file)
- ✓ **Object Code:** Ready to execute program

Java Virtual Machine (JVM)

- ✓ A **Java Virtual Machine (JVM)** is a set of computer software programs and data structures that use a virtual machine model for the execution of other computer programs and scripts
- ✓ Java Virtual Machines operate on Java bytecode, which is normally (but not necessarily) generated
  from Java source code

- ✓ **Bytecode** is a highly optimized set of instructions designed to be executed by JVM
- ✓ Different JVM are there for different platforms

Java and C

- ✓ Excluded
  - ➢ Keywords such as **goto, sizeof, typedef**
  - ➢ Data types such as **Struct, union, enum**
  - ➢ Data type modifiers such as **auto, extern, register, signed, unsigned**
  - ➢ **Pointers**
  - ➢ **Preprocessors**
  - ➢ **Variable arguments** for functions
  - ➢ **void** in functions without parameters
- ✓ Included
  - ➢ New operators **instanceof** and **>>>**
  - ➢ Labelled **break** and **continue**

Java and C++

- ✓No operator overloading
- ✓No template classes
- ✓No Multiple Inheritance
- ✓No Global variable
- ✓Destructors are replaced by **finalize**
- ✓Header files are replaced by packages

Java Buzzwords

- ✓ Simple
- ✓ Secure
- ✓ Portable
- ✓ Object-Oriented
- ✓ Robust
- ✓ Multithreaded
- ✓ Architecture-neutral (Write once; run anywhere; any time; forever)
- ✓ Interpreted
- ✓ High performance (JIT Compiler)
- ✓ Distributed
- ✓ Dynamic

Java and Internet

- ✓ What is Internet
- ✓ How data is moving in WWW
- ✓ Need of portability
- ✓ Passive data and Dynamic, active program
- ✓ Solution: Java Applet
  - ➢ Java program transmitted over network and executed by a Java enabled Browser
- ✓ Security of data

Types of Java Programs

- ✓ Application Program
- ✓ Applet Program
- ✓ Servlet Program
- ✓ JSP Program
- ✓ J2EE Program
- ✓ J2ME Program

Overview of Java Programs

- All Java Programs are **Fully object oriented**
- Main building blocks of Java programs are **predefined Classes (Class library)**
- Class Libraries are bundled as **Packages**
- All packages are resides under the core package **java**
- Example of packages
  - java.io
  - java.lang
  - java.awt
  - java.net
  - java.swing
- Packages are included by using **import** keyword
  - Eg: import java.io.DataInputStream;

Classes & Methods

- ✓ Java is strongly typed and case sensitive
- ✓ Predefined class names are started with capital letter
- ✓ If class name contains more than one words
  - ➢ Each words are started with Capital Letter
  - ➢ No white space between these words
  - ➢ Eg: DataInputStream
- ✓ Functions of a class are known as methods
- ✓ Predefined method names started with small letter
- ✓ If method name contains more than one words
  - ➢ Each words are started with Capital Letter
  - ➢ No white space between these words
  - ➢ Eg:readLine();

Keywords and Identifiers

- ✓ Most of the keywords are borrowed from C language
- ✓ These keywords have same meaning as in C
- ✓ Eg: static, for, while, void, public
- ✓ Identifiers
  - ➢ Programmer defined tokens
  - ➢ Used to name classes, methods, variables, objects, labels, packages and Interfaces
  - ➢ Rules
    - ❑ Can contain alphabets, digits, underscore and dollar sign
    - ❑ Cannot start with digit
    - ❑ Upper case and lower case letters are distinct
    - ❑ They can be of any length

Java Program Structure

Documentation Section

Package statement

Import statements

Interface statements

Class definitions

Main method class

{

Main method definition

}

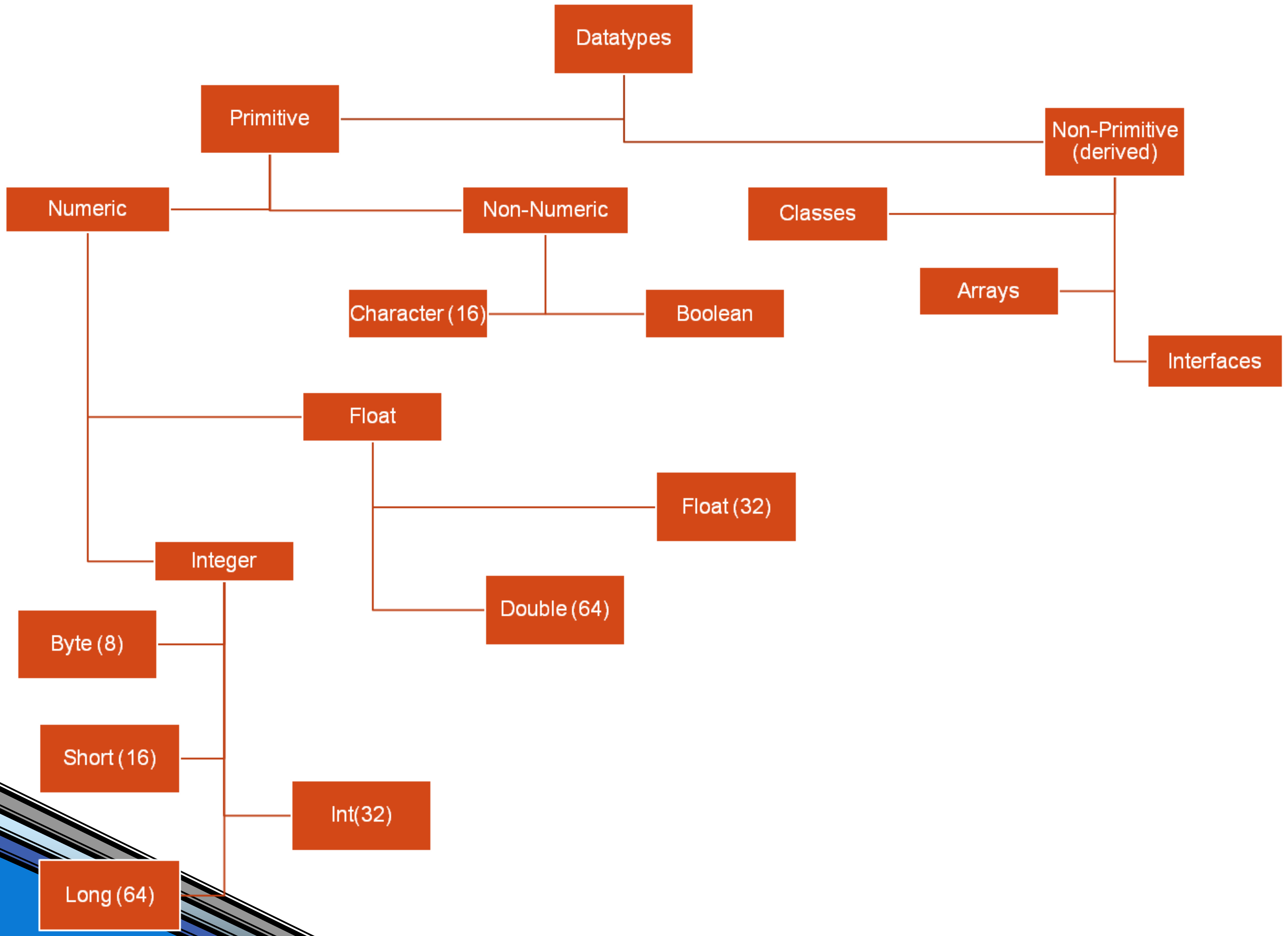
Sample Application Program

```
class testpgm
{
        public static void main(String s[])
        {
                        System.out.println(“Hello”);
        }
}
```

Note: All Classes in **java.lang** package will automatically import

Java Compiler and Interpreter

- ✓ Compiler: javac
  - ➢ Input: .java file
  - ➢ Output: .class file
  - ➢ Eg: javac testpgm.java
- ✓ Interpreter: java
  - ➢ Input: .class file (Mention filename without .class)
  - ➢ Output: get program executed
  - ➢ Eg: java testpgm


Datatypes
Primitive
Non-Primitive (derived)
Numeric
Non-Numeric
Classes
Arrays
Interfaces
Character (16)
Boolean
Float
Float (32)
Double (64)
Integer
Byte (8)
Short (16)
Int(32)
Long (64)

Literals

- Integer Literals
  - Decimal (default). Eg: 1, 67, 987L(Long Integer)
  - Octal. Eg: 01, 067
  - Hexadecimal. Eg: 0x1, 0x67
- Floating-point Literals
  - Float Literal. Eg: 2.0F, 3.14F
  - Double Literal (Default). Eg: 2.0, 3.14
- Boolean Literals
  - True (Not equal to non-zero)
  - False (Not equal to Zero)
- Character Literals
  - Unicode Character Set
  - ASCII Characters. Eg: ‘x’, ‘5’
  - Backslash constants. Eg: ‘\n’, ‘\”’, ‘\f’
- String Literals
  - Eg: “Hello World”, “two \nlines”, “\”this is in quotes\””

Type conversion and Casting

- Automatic Type Conversion
  - Two types must be **compatible**
  - Destination type should be **larger** than source type
  - **Integer** type and **Float** type are compatible
  - **Numeric** type is not compatible with **char** or **boolean**
  - **char** and **boolean** are not compatible each other
- Type casting
  - Narrowing conversion. Eg: int y=100; byte x=(byte) y;
  - Truncation occurs
- Automatic Type Promotion
  - All lower datatype variables will promote to higher data type

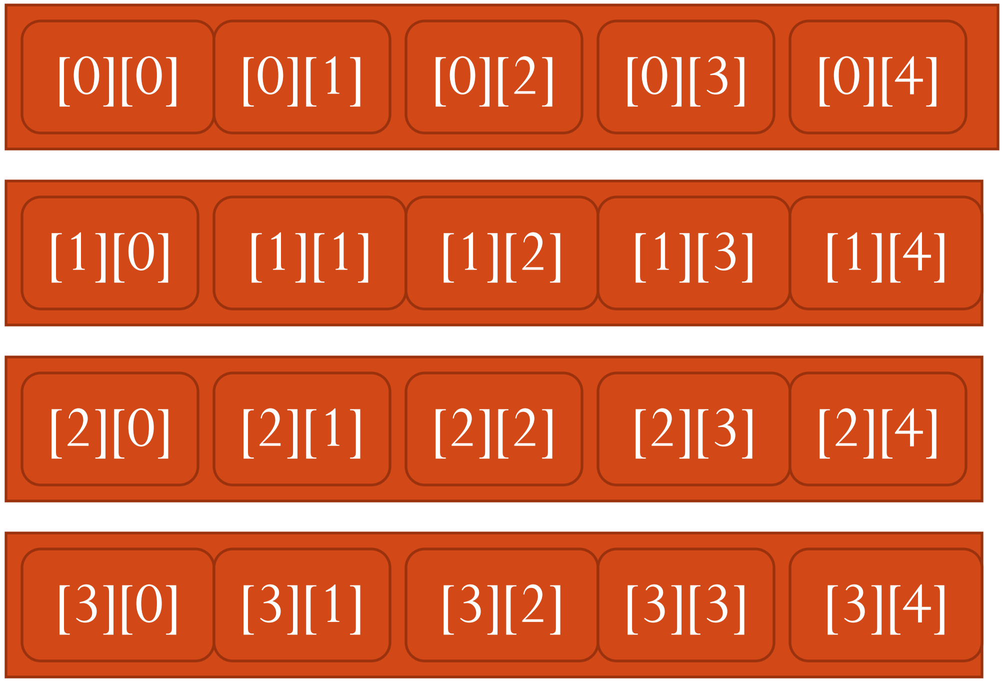
Arrays

- ✓ Array concept is very same
- ✓ Dynamic memory allocation
- ✓ Declaration
  - ➢ Eg: int x[]; x=new int[10];
  - ➢ Eg: int y[]= new int[10];
  - ➢ New is used to allocate memory
  - ➢ Numeric array locations are always initialized to 0
- ✓ Initialization
  - ➢ Eg: float num[] = { 10.1, 11.2, 12.3, 13.4 };
- ✓ When array index goes out of range, java generates Runtime exception called (ArrayIndexOutofBoundsException)

Multidimensional Array

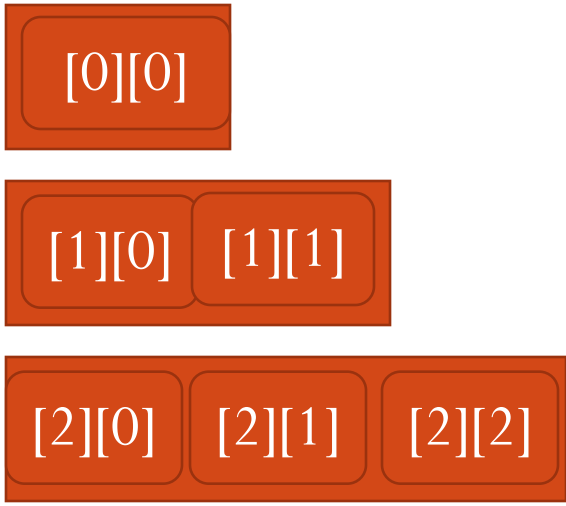
Array of Arrays

- ✓ Eg: int twoD[][] = new int[4][5]



Multidimensional Array – cont..

- ✓ int twoD[][] = new int[4][];
- ✓ twoD[0] = new int[1];
- ✓ twoD[1] = new int[2];
- ✓ twoD[2] = new int[3];
- ✓ Alternate Array Declaration
  - ➢ Int a[] = new int[3];
  - ➢ Int[] b = new int[3];



Java Operators

- ✓ Arithmetic
  - +, -, *, /, %, ++, --, +=, -=, *=, /=, %=
  - Operands must be **numeric** type. **Boolean** is not allowed, but **char** is allowed
  - % can apply to both Integer and Float type operands
- ✓ Relational
  - ==, !=, >, <, >=, <=
- ✓ Logical
  - &, |, ^, ||, &&, !, &=, |=, ^=, = =, !=, ?:
- ✓ Bitwise
  - ~, &, |, ^, >>, >>>, <<, &=, |=, ^=, >>=, >>>=, <<=

Operator Precedence

Highest			
()	[]	.	
++	--	~	!
*	/	%	
+	-		
>>	>>>	<<	
>	>=	<	<=
= =	!=		
&			
^			
|			
&&			
||			
?:			
=	Op=		
Lowest			

Note: Operator precedence can override by using parentheses

Control Statements

- ✓ If stmt
  - ➢ Simple if
  - ➢ Nested if
  - ➢ If-else-if ladder
- ✓ Switch stmt
- ✓ Iteration stmts
  - ➢ While loop
  - ➢ Do..while loop
- ✓ Break & Continue
- ✓ Return stmt