JAVA

Naveen Kumar K S

Course Objective

- To introduce Java Architecture & appreciation of the basic syntax in Java Language
- To illustrate how to make use of standard Java Class Library and create reusable classes.
- To introduce Exception Handling in Java
- To introduce User Interface Concepts & Event Handling

AGENDA

- The Genesis of Java
- An Overview of Java
- Data Types, Variables and Arrays
- Operators
- Control Statements

- **The Genesis of Java**
- An Overview of Java
- Data Types, Variables and Arrays
- Operators
- Control Statements

The Genesis of Java

- ‘C’ in 1970’s By Dennis Ritchie at Bell Lab
- Evolution of “C++”
- The Creation of JAVA
- Java Applets & Application
- Security
- Portability

About JAVA

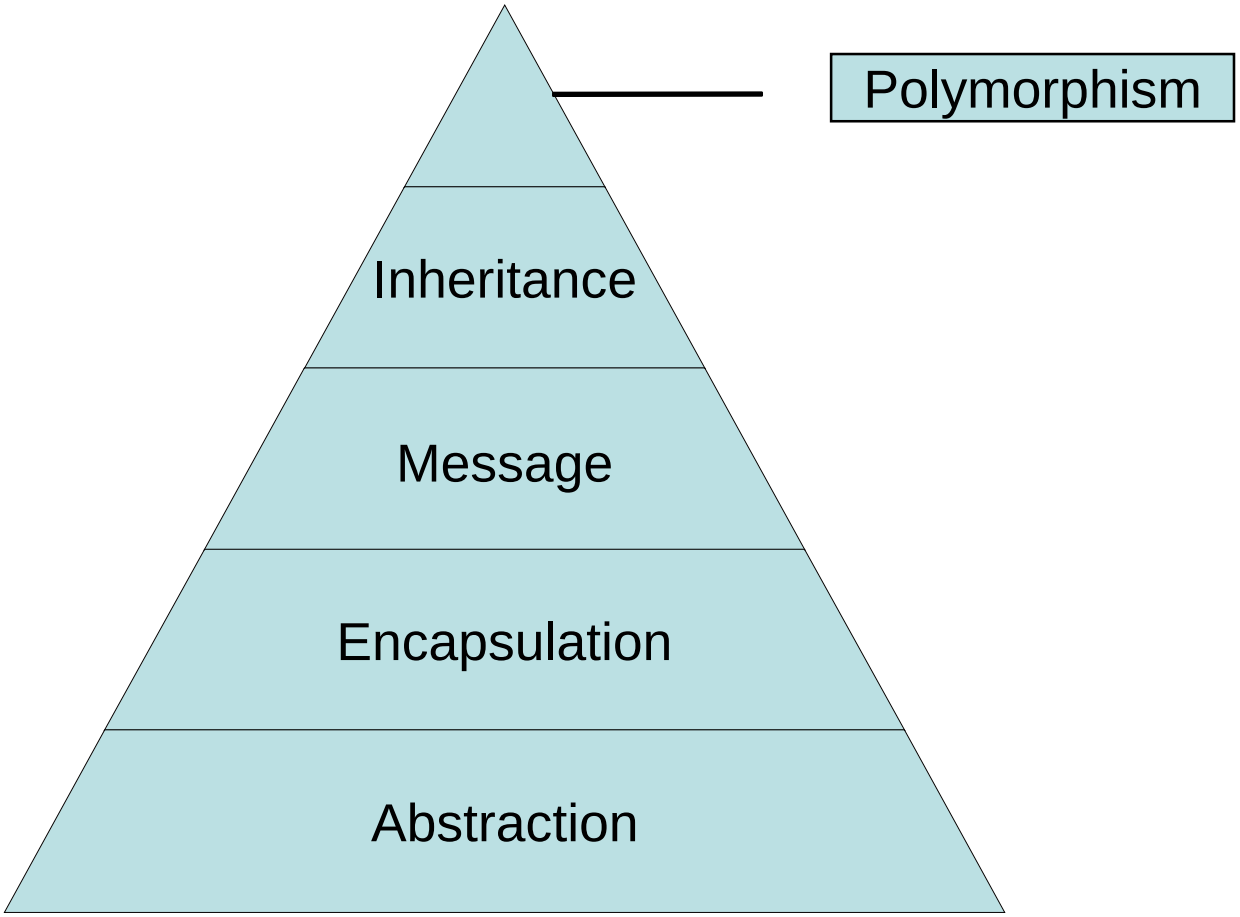
- ✓ Java is a language that is platform independent.
- ✓ A platform is the hardware or software environment in which a program runs
- ✓ Once compiled, code will run on any platform without recompiling or any kind of modification.
- ✓ The .class file that is generated is the machine code of this processor.
- ✓ This is made possible by making use of a **Java Virtual Machine** (JVM)

About JAVA (cont'd)

- Java's Magic: The Byte code
- Simple
- Object-Oriented
- Multithreaded
- Architecture-Neutral
  - Write once; run anywhere, any time, forever
- Java Is Not an Enhanced HTML

- The Genesis of Java
- **An Overview of Java**
- Data Types, Variables and Arrays
- Operators
- Control Statements

OO Basics


Polymorphism
Inheritance
Message
Encapsulation
Abstraction

An Overview of JAVA

- The Three OOP Principles
  - Encapsulation
    - ✓ The Mechanism that binds together code and the date it manipulates, and keeps both safe from outside interference and misuse.
  - Inheritance
    - ✓ The process by which one object acquires the properties of another object (Hierarchical classification)
  - Polymorphism
    - ✓ Meaning “Many forms” . All share the same names

Introduction to Java – Setting up Java

- Before we begin, something on installation
  - Java Installation
  - JDK(v1.5 or higher) (http://java.sun.com)
  - Make sure that the path environment variable is set after installation (eg : Path=%Path%;C:\jdk1.5\bin)
  - to test open a command window and type **javac**

Basic Rules

- The name of the file must always be the name of the “public class”
- It is 100% case sensitive
- You can have only one public class in a file (i.e. in one .java file)
- Every “stand alone” Java program must have a public static void main defined
  - it is the starting point of the program.

A First Simple Program in JAVA

```
/*
        This is a simple Java Program.
        Call this file “First.java”
*/
class First{
    //Your program begins with a call to main()
    public static void main (String args[]) {
    System.out.Println(“ First program in JAVA by Naveen”);
}
}
```

Lexical Issues

- White space
  - ✓ Space, tab or New line
- Identifiers
  - ✓ Used for class names, method names and variable names
- Literals
  - ✓ The constant value in java is representation of Literals
- Comments
- Separators
  - ✓ ( ) { } [ ] ; , .

Java Keywords

- **There are 52 reserved keywords defined in the Java language**

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

- The Genesis of Java
- An Overview of Java
- **Data Types, Variables and Arrays**
- Operators
- Control Statements

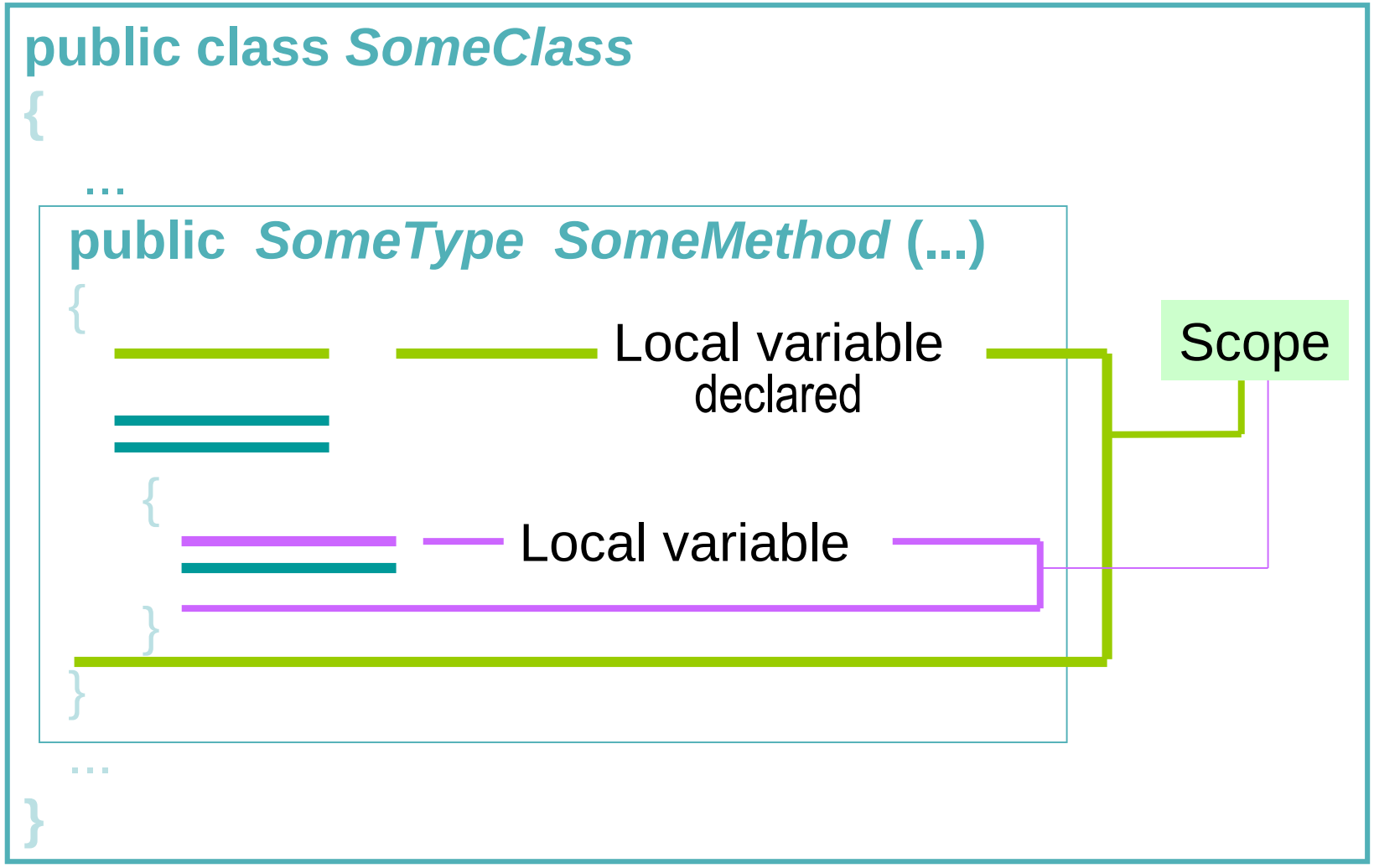
Data Types, Variables & Arrays

Type	In bits	Range
Byte	**8**	-2^{7} to $2^{7}-1$
Short	**16**	-2^{15} to $2^{15}-1$
Int	**32**	-2^{31} to $2^{31}-1$
Long	**64**	-2^{63} to $2^{63}-1$
Float	**32**	3.4e-038 to 3.4e+038
Double	**64**	1.7e-308 to 1.7e+308
Boolean	**8**	True/False
Char	**16**	0 to 65,636 ($2^{16}-1$)

Variables

- Variables can be of different data types: int, char, double, boolean, etc.
- The programmer gives names to variables.
- Names usually start with a lowercase letter.
- A variable must be declared before it can be used
- Local variables lose their values and are destroyed once the constructor or the method is exited.

Local Variables (cont'd)



Variables (cont'd)

- The assignment operator = sets the variable's value:

```
count = 5;
x = 0;
go = new JButton("Go");
firstName = args[0];
```

Assignments

- A variable can be initialized in its declaration:

```
int count = 5;
JButton go = new JButton("Go");
String firstName = args[0];
```

Declarations with initialization

Constants

- Symbolic constants are initialized **final** variables:

```
private final int delay = 30;
private final double aspectRatio = 0.7;
```

Friends Are Friends - **Arrays**

- An *array* is a group of like-typed variables that are referred to by a common name.
- The size of an array is fixed and cannot increase to accommodate more elements
- All elements in the array must be of the same data type

Syntax

```
type var-name[];
array-var=new type[size];
```

Example

```
int month_days[];
month_days=new int[12];
```

Type Conversion & Casting

Automatic Conversions

- ✓ The two types are compatible
- ✓ The destination type s larger than the source type

Casting Incompatible Types

(target-type) value

Ex:

```
i=(int) d;
```

- The Genesis of Java
- An Overview of Java
- Data Types, Variables and Arrays
- **Operators**
- Control Statements

Operators

- Most operators in Java work just like they do in C/C++.
- Arithmetic Operators: **+, -, /, * , %**
- Increment & Decrement Operators : ++, --
- Bitwise Operators:
    - ~ Unary NOT
    - & AND
    - | OR
    - ^ Exclusive OR
    - >> Shift right
    - << Shift Left

Operators (cont'd)

- Relational Operator: == != > < >= <=
- Logical Operators: & | ^ || && ! &= |=
- The ? Operator

Syntax: :

expression1? expression2 : expression3

Example:

ratio=denom==0 ? 0: num/denom;

Relational Operators

- Be careful using == and != with objects (e.g., Strings): they compare references (addresses) rather than values (the contents)
  - String cmd = console.readLine();
  - if (cmd == "Help") ...

Wrong!
(always false)

Relational Operators (cont'd)

- Use the **equals** method to compare Strings:

```
String cmd = console.readLine();
if ( cmd.equals ("Help")  )   ...
```

or

```
if (  "Help".equals (cmd)  )   ...
```

Short-Circuit Evaluation

if (*condition1* **&&** *condition2*) ...

If *condition1* is false, *condition2* is not evaluated (the result is false anyway)

---

if (*condition1* **||** *condition2*) ...

If *condition1* is true, *condition2* is not evaluated (the result is true anyway)

```
if ( x >= 0  &&  Math.sqrt (x) < 15.0) ...
```

- The Genesis of Java
- An Overview of Java
- Data Types, Variables and Arrays
- Operators
- **Control Statements**

Control Statements

- if statement, Nested ifs, if-else-if Ladder

```
if ( <condition> )
{
    < statements >
}
else
{
    < other statements >
}
```

```
if ( <condition> )
{
    < statements >
}
```

else clause is optional

if-else-if

```
if (drinkSize.equals(”Large"))
{
    price += 1.39;
}
else  if (drinkSize.equals("Medium"))
{
    price += 1.19;
}
else   // if "Small" drink size
{
    price += 0.99;
}
```

The switch Statement

```
switch  (expression)
{
   case value1:
      ...
      break;

   case value2:
      ...
      break;
   ...
   ...
   default:
      ...
      break;
}
```

Reserved words

switch
case
default
break

Don't forget breaks!

Iteration Statements

Java's iteration Statements are for, while & do-while

while

```
while(condition){
        // body of loop
}
```

do- while

```
do{
        // body of loop
}while(condition);
```

Iteration Statements (Cont'd)

for loop

```
for (initialization; condition; iteration)
{
        statements;
}
```

```
for(x=0; x<10; x=x+1)
{
        System.out.println("X is "+x);
}
```

Questions….?

Thank You!