Video 3 - Java Programming for Selenium

Java Programming for Selenium

Java OOPS Fundamentals Continuation:

c) Abstraction

- Abstraction is a process of hiding implementation details and showing functionality to the user
- In another way, it shows important things to the user and hides internal details, Ex: Sending an Email
- Abstraction focuses on what the object does instead of how it does

Abstraction can be achieved in two ways,
1) Abstract Class
2) Interface

d) Encapsulation

Encapsulation is a process of wrapping code and data together into a single unit

Ex: Capsule

It provides control over the data

Java Programming
Java Project
Java Package/s
Java Class/s

We write Java code in Java Class…

I) Java Environment Setup

1) Uses of Java
2) Java Syntax
3) Java Environment Setup & Verify

1) Uses of Java

Java is used to develop

- Desktop Applications
- Web Applications
- Enterprise Applications (ex: Banking, Insurance, ERP etc…)
- Mobile Applications
- Embedded Systems
- Smart Cards
- Games Applications
- Scientific Applications Etc…
- Test Automation (with Selenium)

---

2) Java Syntax:

i) Java is a case sensitive language

Note: All Java keywords and reserved words are small letters
(if, for, public, main, true, false, null…)

ii) First letter of class name should be in upper case

sample //Incorrect
Sample //Correct
Firstprogram //Correct
FirstProgram ////Correct

iii) Java Method names should start with lower case

iv) Java program file name should exactly match with class name

v) Java program execution starts from main method, which is mandatory in every Java program

vi) Every statement /step should end with semi colon (;)

vii) Code blocks (conditional, Loop, Method etc…enclosed with {},

viii) Java supports Explicit declaration of Data Types

In Java,
int sno =123;//Correct
int x;//Correct
char a='A'; //Correct
boolean y=true;
abc =100; //Incorrect

In VBScript
Dim city
city =100
.
city ="India"
.
city=1.23
.
city=#10/10/2010#

ix) Java supports Explicit declaration of Variables

int a, b;
a=10;
b=20;
c=30;//Incorrect

In VBScript

Dim a
a=100
b=200 'Correct

---

3) Java Environment Setup

To write and execute Java programs then Java Environment is required

Three important steps in Computer programming,
i) Write a Program (in any Editor)
ii) Compile the Program
iii) Run the program

A) Steps for writing and executing Java programs using command line interface

i) Download Java Software (JDK) and Install
ii) Set Java Environment variable in the OS Environment
iii) Write a Java program in Notepad and save with ".java" extension
iv) Compile the java program file in command prompt and Run…

B) Steps for writing and executing Java programs using Eclipse IDE

i) Download Java Software (JDK) and Install
ii) Set Java Environment variable in the OS Environment
iii) Download Eclipse IDE and Extract
iv) Write Java programs in Eclipse Editor and Run

Download Java Software (JDK) from either java.com or oracle.com, and install
Note: Download Java Software based on your Operating Environment
Ex: Windows 10 32 bit OS

After Java Software installation we get "Java" Folder in C:/Program Files/

Set Java Environment Variable…

Copy jdk bin directory path from your computer,

Step 1: Write a Program in Notepad

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

Save as .java file with class name

Step 2: Compile the Program

- Launch the Command prompt
- Change to Java program file directory
- Type javac File Name/sample.java

Then it will create Java class file (.class)

Step 3: Run the program

- Launch the Command prompt
- Change to Java program file directory
- Type Java Class file name without extension

It will display the output on the Console

Eclipse IDE:

- Eclipse IDE is a platform to write & execute Computer programs like Java, C, C++, Perl, Python, Ruby, PHP etc…
- Eclipse IDE is Open Source
- It provides Editor for writing Programs, Syntax Guidance, Context Help and Auto compilation etc…

Navigation for writing and executing Java programs in Eclipse IDE

- Launch Eclipse IDE
- Create Java project
- Create Java Package under the Java Project
- Create Java Class under the Java Package.

Java program Example:

```
public class Sample {

public static void main(String[] args) {
int a=10, b=20;
System.out.println("Addition of a, b is: "+ (a+b));

if (a>b){
System.out.println("A is a Big Number");
}
else {
System.out.println("B is a Big Number");
}

}

}
```

II) Java Program Structure

i) Java Program Structure
ii) Java Sample Program

i) Java Program Structure

1) Documentation Section

It includes the comments to tell the program's purpose, It improves the readability of the program

2) Package Statement

It includes statement that provides a package declaration

3) Import Statement/s

We import predefined and user defined libraries using "import" keyword

ex:
import java.io.Console;

java - Project
io - Package
Console - Class

import java.io.*; java - Project io - Package io.* - import all classes from io package

Predefined/Built-in,
All libraries in Java are predefined, but a few libraries only automatically loaded in every Java program.

4) Interface Section
It includes method declaration

5) Class Definition
ex:
public class Sample{
.
}

6) main Method (java program execution starts from main method

```
public static void main (String [] args){
….
}
```

public - Access Modifier
static - Non Access Modifier (use main Method without invoking any Object)
void - Returns nothing
main - Method name

(String [] args) -?

7) Declaration Statement/s

We declare Variables and Constants

```
int a;
a=100;

int b=200;
b=300;
c=400;

final int y=1000; (Constant)
y=2000; //Incorrect
```

Variables vs. Constants

```
int a;//Correct
a=10;
a=30;
int b=200;
b=400;

final int x; //Incorrect
final int y=300; //Correct
y=300; //Incorrect
```

8) Normal Statements

```
c=a+b;
System.out.println("Hello");
```

System - Predefined Class
out - Object
println - Method
"Hello" - Message

9) Code Blocks

Conditions,
Loops,
Methods, etc…

10) Object Creation Statement

Note 1: We can create Object at beginning of the program or middle of the program
or end of the program

Note 2: Usually we create Object/Instance of the Class within main method, but we can also create
Objects outside of the main method

Syntax:
ClassName objectName= new ClassClassName();

---

ii) Java Sample Program

```
//It is Sample Program to Understand the Java program Structure.
package abcd;

public class Sample {
//Create a Method with Arguments and return a value (Non Static method)
public int add(int a, int b){
int result;
result=a+b;
return result;
}
//Create a method without Arguments and returns nothing (Non Static method)
public void comparison(){
int x=100, y=20;

if (x>y){
System.out.println("X is a Big Number");
}

else{
System.out.println("Y is a Big Number");
```

```
}
}
//Create a Method with Arguments and return a value (Static method)
public static int sub(int a, int b){
int result=a-b;
return result;
}
//Create a Method without and returns nothing (Static method)
public static void comparision2(){
int a=100, b=200;

if (a>b){
System.out.println("A is a Big Number");
}
else{
System.out.println("B is a Big Number");
}
}
public static void main (String [] args){
//Create Object to call Non Static methods
Sample obj = new Sample();
int res = obj.add(100, 200);
System.out.println(res);//300
//Or
System.out.println(obj.add(100, 200));//300

obj.comparison();//X is a Big Number
//-----------------------------------------
//Call Static Methods using Class name
res = Sample.sub(100, 50);
System.out.println(res);//50
//Or
System.out.println(Sample.sub(200, 100));//100

Sample.comparision2();//B is a Big Number
//-----------------------------------------
//Call Static Methods without using Class name
int x= sub(10, 5);
System.out.println(x);//5

System.out.println(sub(20,10));//10

comparision2();//B is a Big Number
```

```
int a;//Variable Declaration
a=100; //Initialization
int b=200; //Variable Declaration with Initialization
int c, d, e; //Declare multiple variables
int f=40, g=50, h=60; //Declare multiple variables with initialization

double l=123.45678;
char m='*';
boolean p=true;
String q="Selenium Testing";

System.out.println(q);//Selenium Testing
System.out.println(l);//123.45678
System.out.println("Hello Java");

final int price =100;
System.out.println(price);

if (a>b){
System.out.println("A is a Big Number");
}
else
{
System.out.println("B is a Big Number");
}

char grade ='U';
switch (grade){
case 'A':
System.out.println("Good");
break;
case 'B':
System.out.println("Excellent");
break;
case 'C':
System.out.println("Better");
break;

default:
System.out.println("Invalid Grade");
}

//Print 1 to 5 Numbers except 4 using for loop
for (int i=1; i<=5; i++){
```

```
if (i != 4) {
System.out.println(i);
}
}
//Print 1 to 5 numbers using while loop
int j=10;
while (j<=15){
System.out.println(j);
j++;
}

//do while loop
int k=100;
do
{
System.out.println(k);
k++;
} while (k<=8);

//enhanced for loop
String [] tools ={"Selenium", "UFT", "RFT", "SilkTest"};

for (String mytool: tools){
System.out.println(mytool);

}
}
}
```
