Java for Selenium Testers

Refer: Learn automation online YT channel

Packages - collection of similar classes.

Pascal case Ex: myPackage

→ com.lab.javalearning

class - data + methods (function).

single entity comprises of data and methods

```
public static void main (String [] args){ →main method
}
```

→ entry point of execution

jvm searches for main method

- Object Oriented Language

int, double, float, char, long, byte - Primitive data type

Object cannot be created for primitive data type.

Wrapper class

Integer, Float, Double, Character

can create object with wrapper class

used mostly in <Generics>

Example of Simple java program

```
package com.lao.javalearning;

public class BankAccount {

    Long accountNumber = 1234567890l;
    String holderName = "Agni";
    Integer accountBalance = 350;

Public void getBalance() {
    System.out.println("Balance is " + accountBalance);

public static void main (String[] args) {
  // ClassName objname = new classname();
BankAccount account = new BankAccount ();
  account.getBalance();
}
```

---

return type - returns the value.

void - does not return anything

Constructors (Special method)

- Constructor is a block of code that initialize the newly created object

Properties

- Constructor name must be same as its class name
- must not have explicit return type
- Cannot be abstract, Static, final & Synchronized

Types of Constructors

- Default
- No arguments
- Parameterized

- Default Constructor

When we dont provide any constructor, compiler will provide one. We cannot see with naked eye, it will be in class file, used to provide default values to the object like 0, null etc depending on the type

- No Argument Constructor

No arguments will be there
Allow us to write logic when object is created
Not a default Constructor

Constructor Overloading

- Reacting according to situation
- Based on Parameters provided or not

Having only parameterized Constructor and calling no argument constructor will cause compile time error.

Extends class inherits the property of parent class

- Inheritance within child class super(); Keyword will be available if Parent child relationship available

Conditional Statements

- If Condition
- If else
- If - else - if
- Switch Case

∴ Boolean, returns True / false

Default return type - false

equals() - looks for exact match & case sensitive

equalsIgnoreCase() - ignores the case

Looping Statements

When we have to perform certain operations repeatedly

For loop // for(initialization, condition, increment / decrement)

While loop (Entry Controlled Loop)

do while (Exit Controlled Loop)

Static Keyword

- Can be used with class, variable, method & block
- belong to the class instead of a specific instance
  this means if you make a member static, you can access it without object.
- If there are both static and non static methods, we can call static method from non static method but not otherwise.

Static block

Static {

- Initializing the static variable
- gets executed when class is loaded in memory }
- can have multiple static blocks, execution in written order, executed before the main method itself

Static Variables

- common to all instances (Objects) of the class because it is class level Variable
- only a single copy of Static variable is created and shared among all the instances of class

Static class

A class can be made static only if it is a nested class

Inheritance

- Process of acquiring the properties (data + methods)

The one which takes - child class
The one which gives - parent class

Note:
A child can be a parent of another class
A parent can be a child of another class

Using extends keyword we can inherit

Also called IS-A Relationship

Types of Inheritance

1. Single $A \rightarrow B$
2. MultiLevel $A \rightarrow B \rightarrow C$
3. Multiple (not supported in Java) $A + B \rightarrow C$
4. Hierarchical $A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$

Access Modifiers

1. default - when no access modifier is specified
2. Private - only within the class which they are declared
3. Protected - within same Package / Sub classes in different Package
4. public - from everywhere in the program

Polymorphism

Polymorphism is the capability of a method (function) to do different things based on the object that is acting upon

Types:

1. Static / compile / early binding (Overloading)
2. Dynamic / Runtime / late binding (Overriding)

Overloading

- Method name should be same with different parameters
- If there is any change (change in data type, change in order of passing, no of parameters) in the parameter passed to a method then it is Overloading
- happens at compile time
- done in the same class
- achieved by changing the no of arguments and changing datatype of arguments

- We can overload the main method

Overriding

- No change in anything
- happens at run time
- done in parent / child relationship

Dynamic runtime Obj creation syntax

```
ParentClass ref = new childclass();
```

// changing the execution statement of parent class

- Same method name in parent and child class and creating object using runtime polymorphism syntax and printing child class method.

Abstraction

abstract noun - god, devil, ~~ugo~~ (not visible to eyes)

- Hiding the Implementation details

How can we achieve abstraction

1. Abstract class (0 to 100% abstraction)
2. Interface (100% abstraction)

Static Method

- Accessed directly by specifying its name
- Called without creating an object class
- Static method belongs to the class

Abstraction - abstract keyword is used,
- Cannot create objects for abstract class, accessed by inheriting from another class
- abstract method can be used only in abstract class and it does not have a body, body is provided in Subclass

Interface: ~~implements~~ 100% abstraction, Cannot create object,
- To access interface methods, interface must be implemented by another class using 'implements' keyword, The method body is provided in Subclass

Encapsulation :- To make sure that 'sensitive' data is hidden from users, achieved by declaring class variables as private and using get and set methods to access and update the value of variable

Diff b/w Inheritance & Polymorphism

Inheritance lets us to ~~use~~ inherit attributes & methods from another class whereas polymorphism use those methods to perform different tasks