virtusa®

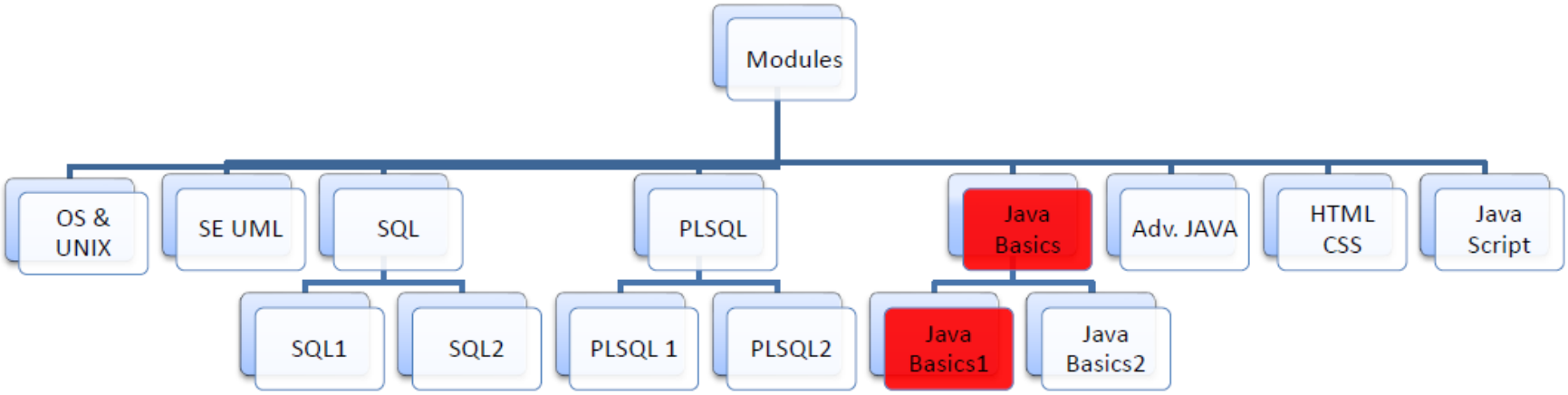
Accelerating Business Outcomes

Java Basics
Part 1

Module Overview

Purpose:

- The following module hierarchy presents the technical modules required to build the basic IT skills and acquaints you with relevant technology basics.
- The current module –**Java Basics 1** (highlight in red) underwrites Basics of Java Programming and will enable you to enhance one's coding skills using java language constructs and OOPS methodology.



*Recommended duration to complete Java Basics1 module: 10 hours

Module Objectives

By the end of this module, you will be able to:

- Define Java and it's features
- Define OOPs concepts and their usage
- Implementation of Inheritance or IS-A relationship
- Implementation of Aggregation or HAS-A relationship
- Implementation of Runtime Polymorphism
- Implementation of Abstract class and Interface
- Define Exceptions and its usage in Java

Java Language

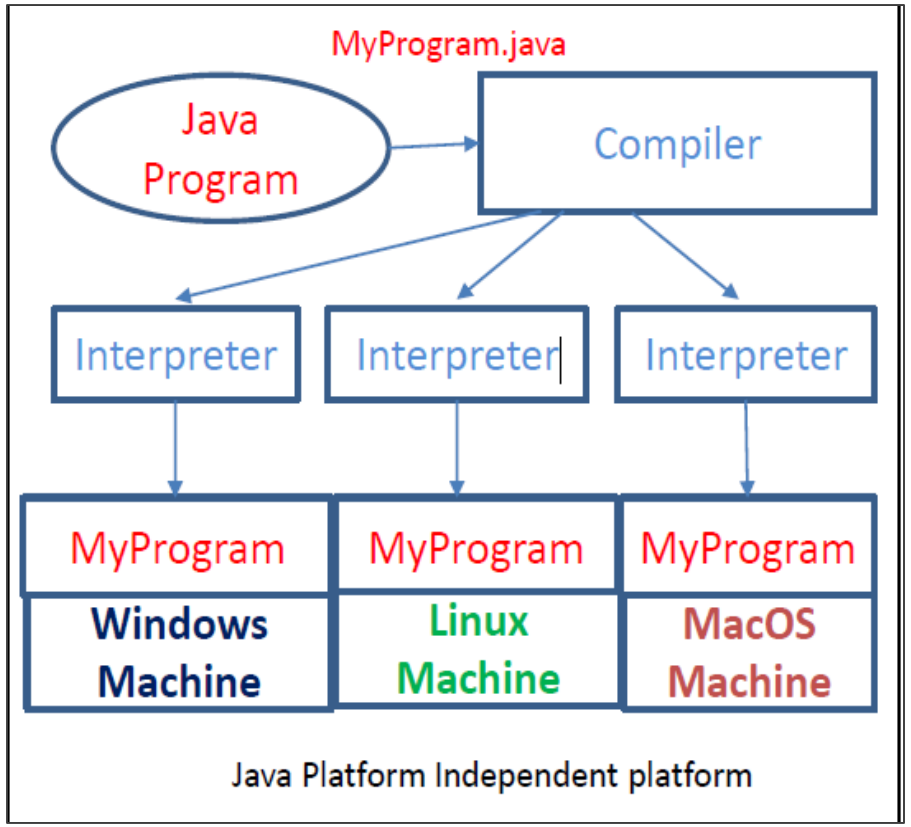
Define the Java language and it's features

What is Java?

- Java is a *programming language* and a *platform.*
- Java is a high level, robust, secured and object-oriented programming language.

Features of Java

- **Platform Independent:** A platform is the hardware or software environment in which a program runs.
- It has two components:
    1. Runtime Environment
    2. API(Application Programming Interface)
- Java code can be run on multiple platforms e.g. Windows, Linux, Sun Solaris, Mac/OS etc.



Reference

- http://www.javatpoint.com/java-tutorial - for Java language and its usage
- http://www.javatpoint.com/features-of-java - for java language features

Object Oriented Programing

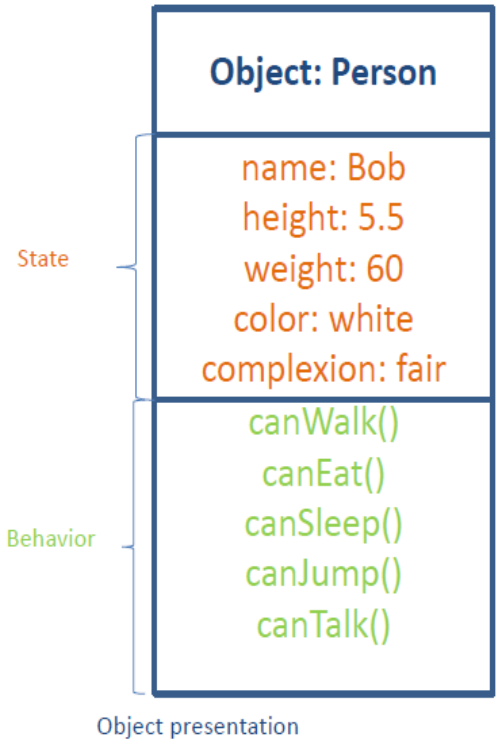
Define OOPs concepts and their usage

What is OOPs?

- Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects
- OOP simplifies the software development and maintenance by providing some concepts:
  - Object
  - Class
  - Inheritance
  - Polymorphism
  - Abstraction
  - Encapsulation

What is an Object?

- An entity that has state and behavior is known as an object e.g. a Person.
- A person object has two characteristics:
  - **State:** person identity, name, height, weight, color, complexion
  - **Behavior:** A person can walk, eat, sleep, jump and talk



Reference

- http://www.javatpoint.com/java-oops-concepts

Object Oriented Programing (continued)

Define OOPs concepts and their usage

What is a Class?

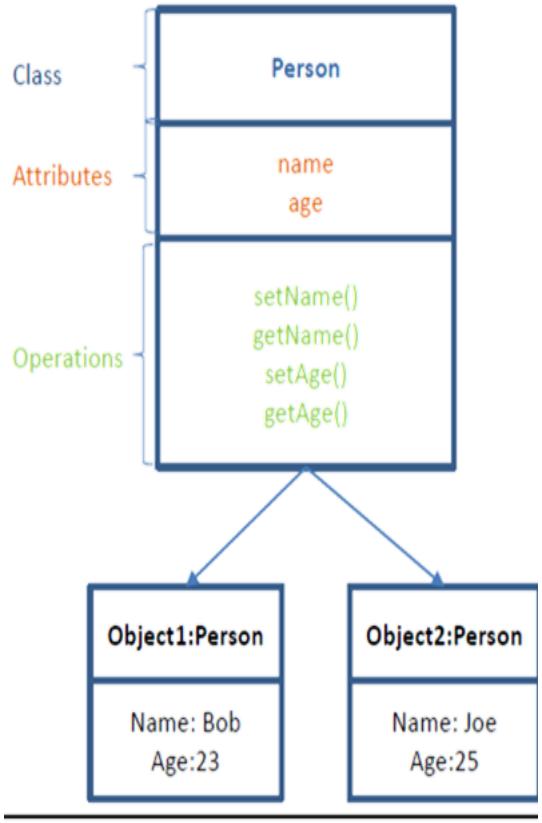
A class is a template/blue print that describes the behaviors/states that object of its type support.

Class vs. Object

Class is a static blueprint (template) of its Objects while objects represent runtime instances of that class.

A class can contain any of the following variable types.

- **Local variables:** Variables defined inside methods, constructors or blocks are called local variables. The variable will be declared and initialized within the method and the variable will be destroyed when the method is completed.
- **Instance variables:** Instance variables are variables within a class but outside any method. These variables are instantiated when the class is loaded. Instance variables can be accessed from inside any method, constructor or blocks of that particular class
- **Class variables:** Class variables are variables declared with in a class, outside any method, with the static keyword



Reference

- http://www.javatpoint.com/java-oops-concepts

Inheritance

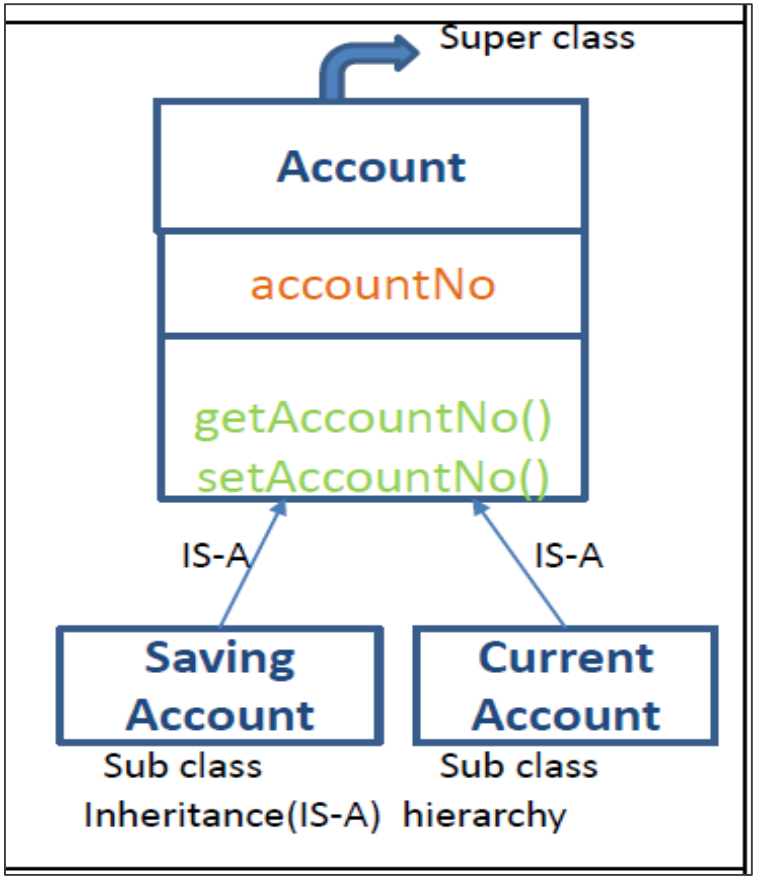
Implement Inheritance or IS-A relationship

What is Inheritance?

- Inheritance in Java is a mechanism by which one object acquires all the properties and behaviors of parent object.
- Inheritance represents the IS-A relationship, also known as Parent-child relationship.

Usage of inheritance in java

- For method overriding
- For code reusability



Reference

- http://www.javatpoint.com/inheritance-in-java

Aggregation

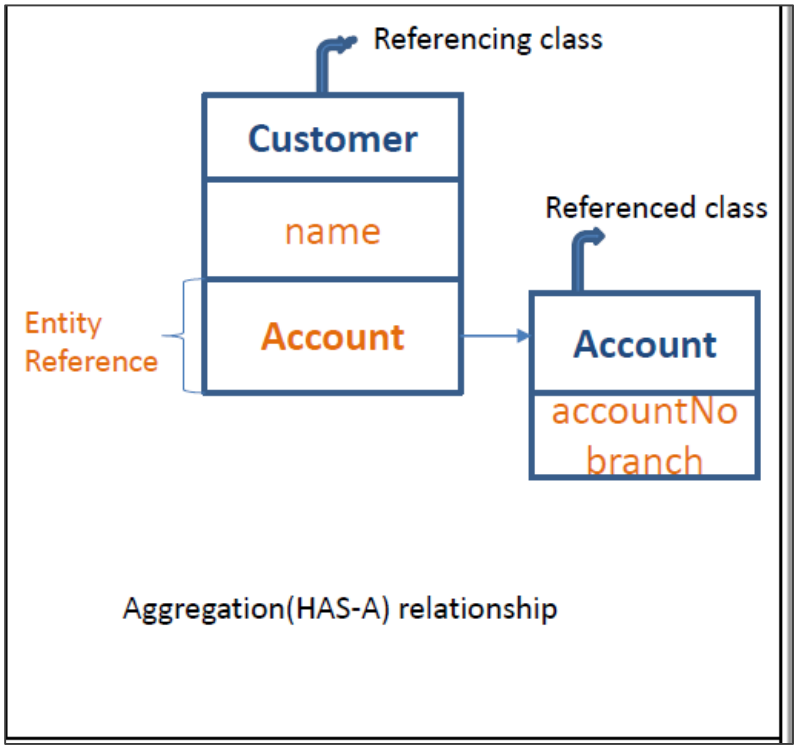
Implement Aggregation or HAS-A relationship

What is Aggregation?

- If a class has an entity reference, it is known as Aggregation.
- Aggregation represents HAS-A relationship.

Usage of Aggregation in java

- Code re-use is also best achieved by aggregation when there is no IS-A relationship.
- Inheritance should be used only if the relationship Is-A is maintained throughout the lifetime of the objects involved; otherwise, aggregation is the best choice.



Reference

- http://www.javatpoint.com/aggregation-in-java

Runtime Polymorphism

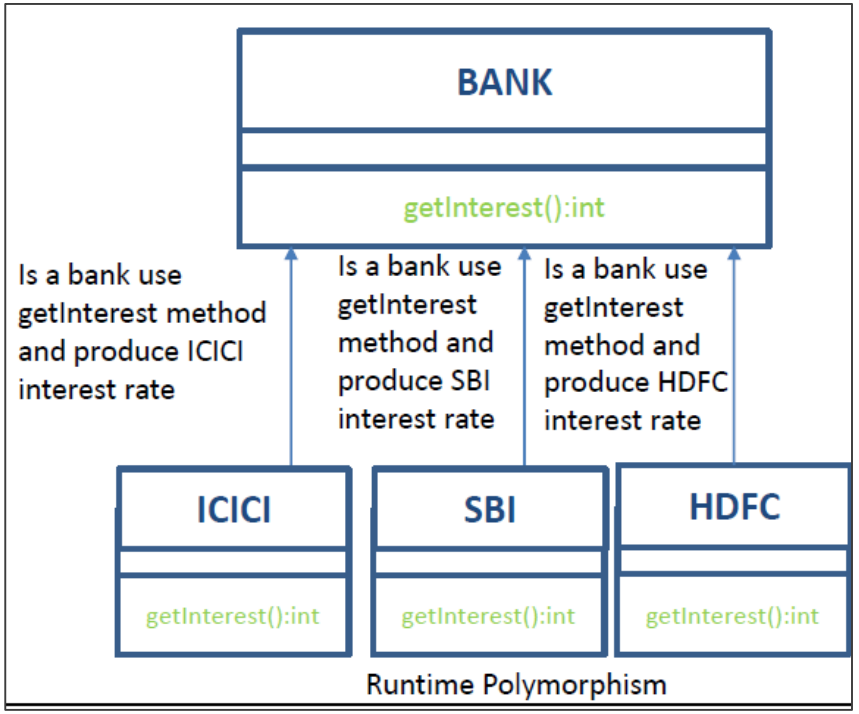
Implement Runtime Polymorphism

What is Polymorphism?

- Polymorphism in java is a concept by which we can perform a single action in different ways.
- There are two types of polymorphism in java - compile time polymorphism and runtime polymorphism.
- We can perform polymorphism in java by method overloading and method overriding.

Usage of Polymorphism in java

- Runtime polymorphism is a process in which a call to an overridden method is resolved at runtime rather than compile-time.
- In this process, an overridden method is called through the reference variable of a superclass.



Reference

- http://www.javatpoint.com/runtime-polymorphism-in-java

Abstract Class and Interface

Implement Abstract class and Interface

What is Abstract class?

- If a class contains any abstract method then the class is declared as **abstract class**. An abstract class is never *instantiated.*
- It is used to provide abstraction.

Usage of Abstract class:

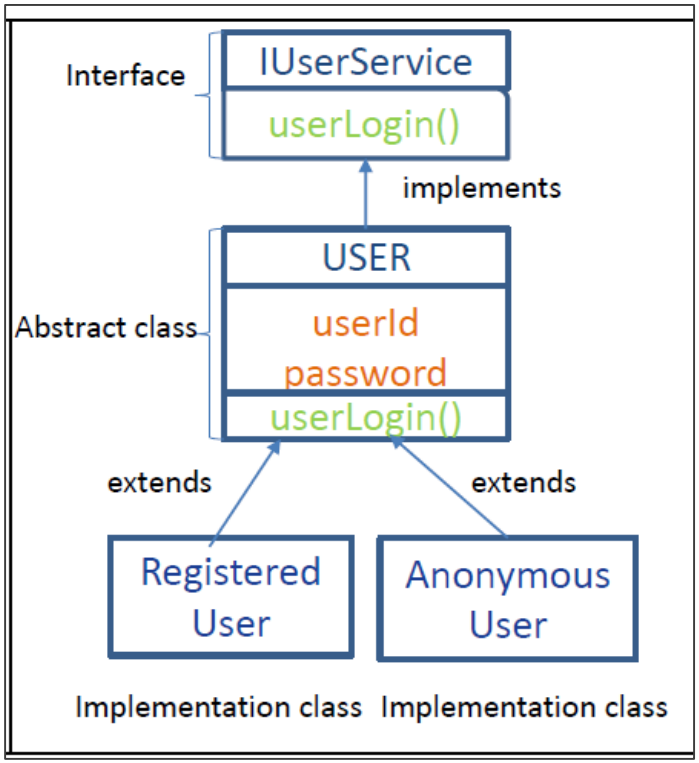
- Abstract class is used in case of IS-A relationship.

What is an Interface?

- Interface is a *pure abstract class*. One cannot create instance of an Interface. Their methods are declared without any body.
- Interface is used to achieve complete abstraction in Java.

Usage of Interface:

- Interface is used in case of Multiple Inheritance in Java.



Reference

- http://www.javatpoint.com/abstract-class-in-java
- http://www.javatpoint.com/interface-in-java

Exception Handling

Define Exceptions and its usage

What is Exception Handling?

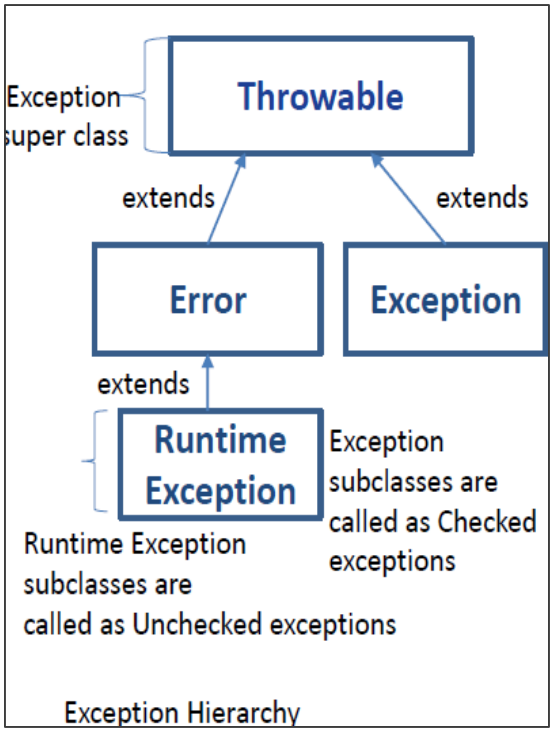
- The exception handling in java is one of the powerful mechanism to handle the runtime errors so that normal flow of the application can be maintained.
- Exception Handling is a mechanism to handle runtime errors such as Class Not Found, IO, SQL, Remote etc.

Advantage of Exception Handling

- The core advantage of exception handling is to maintain the normal flow of the application. Exception normally disrupts the normal flow of the application that is why we use exception handling.

Exception in Java

- A Java Exception is an object that describes the exception that occurs in a program. When an exceptional events occur in java, an exception is said to be thrown. The code that's responsible for doing something about the exception is called an exception handler.



Exception Hierarchy

Reference

- http://www.javatpoint.com/exception-handling-in-java

Additional References

To explore more on the subject, refer the below links and books:

Links:

- http://www.javatpoint.com/java-tutorial
- http://docs.oracle.com/javase/tutorial/java/javaOO
- http://javabeginnerstutorial.com/core-java-tutorial/java-basicsgetting-started-with-java/

Books:

- Java Complete Reference
- Head First Java

Self Check

Instructions to write Self Evaluation Sheet:

Open the excel sheet, refer Java Basics Part 1 sheet, write down the solutions for all questions, save a local copy in your machine.

Lab Assignment

- Refer ***Assignment Document*** for this module to proceed with **Lab Assignment.**
- Do **submit the Solutions** for the given assignment and refer the ***Participant guide*** for submission procedure.

Module Summary

Now that you have completed this module, you will be able to:

- Explain the Java program structure and it’s execution steps
- Write java program using OOPs concepts
- Use the Abstract classes and Interfaces in a Java application
- Handle built-in exceptions in a java program
- Write and manage user-defined exceptions in a java program

Thank you