SNS COLLEGE OF TECHNOLOGY

(An Autonomous Institution)
Coimbatore - 641035.

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A++" Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

Department of Computer Applications

Course Code: 23CAT606

Course Name: Java Programming

Unit I: Java Fundamentals

Topic 4: Package

Java Features

01 Class

02 Object

03 Characteristics of Object

04 3 ways to intialize object

05 Anonymous Object

06 Different ways to create object

07 Class and methods

Package

- A **java package** is a group of similar types of classes, interfaces and sub-packages.
- Package in java can be categorized in two ypes,
    1. Java API package (Built-in package)
    2. User-defined package. (Defined by user)

Built in Packages

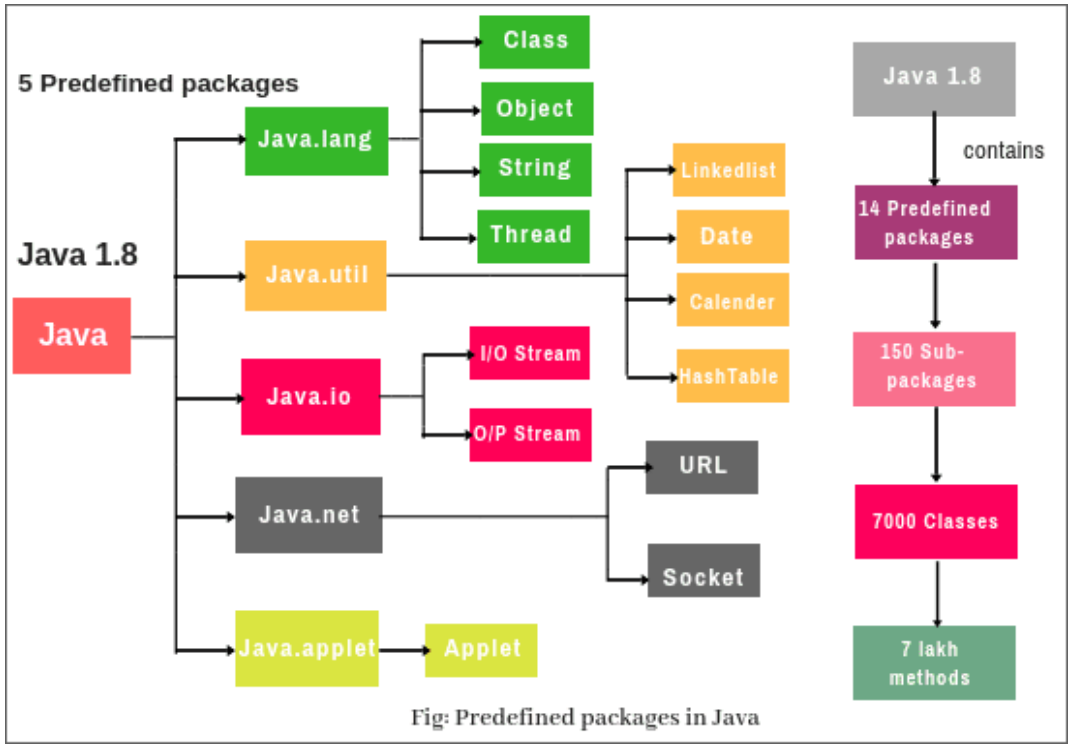
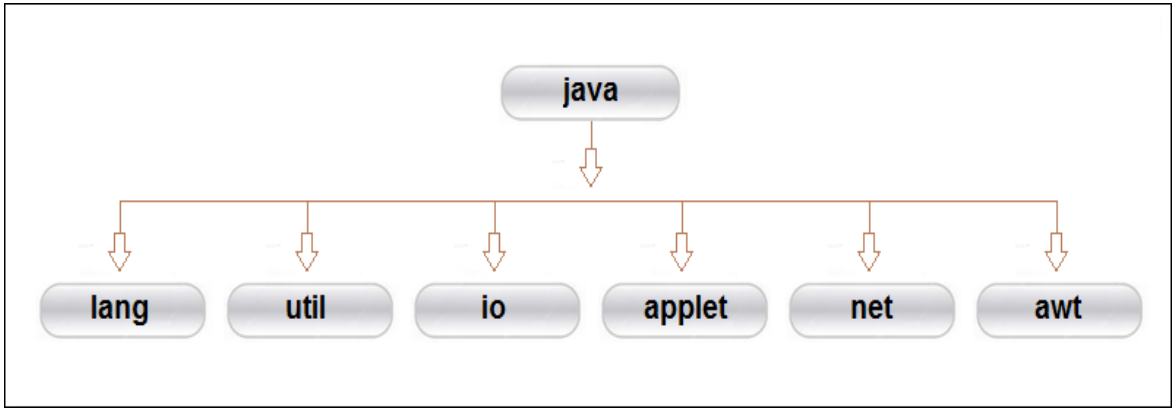


Fig: Predefined packages in Java

Java API Package

- Java **APl(Application Program Interface)** provides a large numbers of classes grouped into different packages according to functionality.
- Most of the time we use the packages available with the Java API.



Java API Package

PACKAGE	CONTENTS
java.lang	Language support classes. They include classes for primitive types, string, math functions, thread and exceptions.
java.util	Language utility classes such as vectors, hash tables, random numbers, data, etc.
java.io	Input/output support classes. They provide facilities for the input and output of data.
java.applet	Classes for creating and implementing applets.
java.net	Classes for networking. They include classes for communicating with local computers as well as with internet servers.
java.awt	Set of classes for implementing graphical user interface. They include classes for windows, buttons, lists, menus and so on.

Using API Package

- The import statements are used at the top of the file, before any class declarations.
- The first statement allows the specified class in the specified package to be imported

Import java.awt.color;

- The above statement imports class color and therefore the class name can now be directly used in the program.
- The below statement imports every class contained in the specified package.

Import java.awt.*;

- The above statement will bring all classes of java.awt package.

User defined Package

Create a Package

- To create a package, a name should be selected for the package.
- Include a **package** statement along with that name at the top of every source file that contains the classes, interfaces.
- The package statement should be the first line in the source file.
- There can be only one package statement in each source file, and it applies to all types in the file.

Creating User defined Package

```
//save as Simple.java
package mypack;          // Package name
public class Simple
{
public static void main(String args[])
  {
    System.out.println("Welcome to package");
  }
  }
```

Compile and run

To compile and run the package,

To Compile: javac -d . Simple.java

To Run: java mypack.Simple

- The -d is a switch that tells the compiler where to put the class file i.e. it represents destination.
- The . (dot) represents the current folder.

OUTPUT: Welcome to package

Accessing the Package

There are three ways to access the package from outside the package.

1. import package.*;
2. import package.classname;
3. fully qualified name.

Using import Package

```
//save by A.java
package pack;
public class A
{
  public void msg()
  {
      System.out.println("Hello");
  }
}
```

```
//save by B.java
package mypack;
import pack.*;
class B
{
 public static void main(args[])
 {
    A obj = new A();
    obj.msg();
  }
  }
```

Output: Hello

Using Package.Classname

```
//save by A.java
package pack;
public class A
  {
    public void msg()
   {
     System.out.println("Hello");
   }
}
```

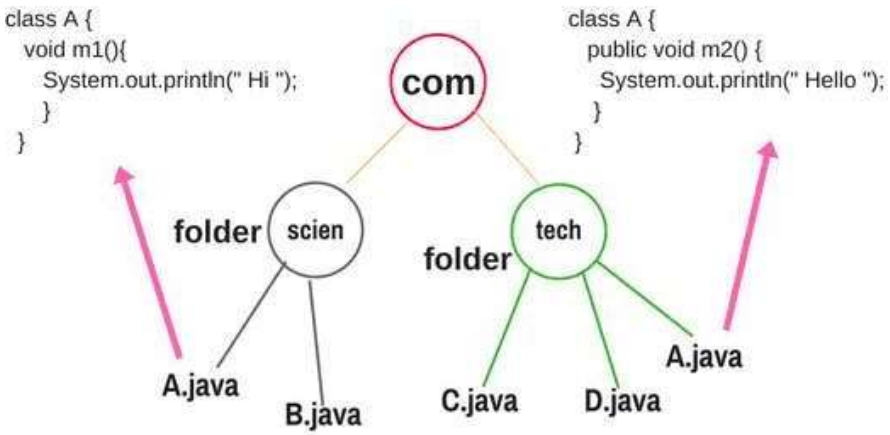
Output: Hello

```
//save by B.java
package mypack;
import pack.A;
class B
  {
     public static void main(args[])
  {
      A obj = new
      A();
      obj.msg();
   }
   }
```

Using fully qualified name

- Using fully qualified name can declared a class of this package will be accessible.
- Now there is no need to import.
- But you need to use fully qualified name every time when you are accessing the class or interface.
- It is generally used when two packages have same class name e.g. java.util and java.sql packages contain Date class.

Using fully qualified name



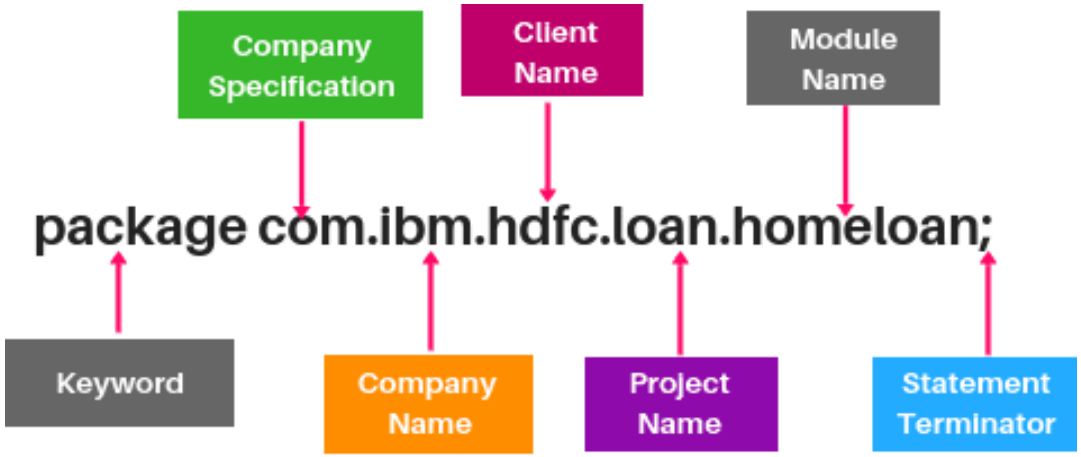
My requirement to call m1 of class A of sub-package scien and m2 of class A of sub-package tech form class B of sub-package scien.

```
package com.scien;
 class B
 {
  void m3()
  {
    System.out.println("Hello Java");
  }
public static void main(String[] args)
 {
 // keep as it is because it is from same package "scien". A a = new
   A(); a.m1();
 // It will direct go to tech package and call the method m2. com.tech.A
   a1 = new com.tech.A();
a1.m2;
B b = new B();
       b.m3();
       }
     }
```

Output:

Hi
Hello
Hello Java

Naming Convention to declare User-defined Package in Real-time Project



While declaring the package name, every character should be lowercase

Fig: Complete Package Structure of Project

Reference

1. Herbert Schildt " The Complete Reference Java 2, 8th edition , Tata McGraw Hill, 2011
2. Ralph Bravaco, Shai Simonson, "Java Programming: From the Ground up Tata McGraw Hill, 2012
3. https://www.scientecheasy.com/2020/06/packages-in-java.html/#2_Predefined_Packages_in_Java_Built-in_Packages
4. https://www.scientecheasy.com/2019/06/java-interface-use.html/

Summary