



SNS COLLEGE OF ENGINEERING



Kurumbapalayam(Po), Coimbatore – 641 107

Accredited by NAAC-UGC with 'A' Grade

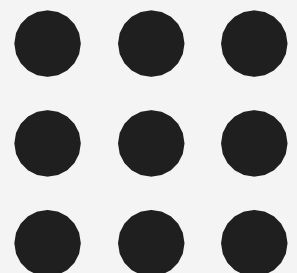
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

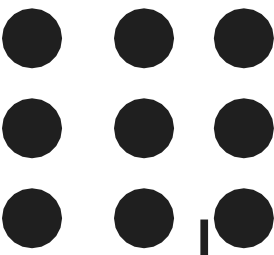
Department of Information Technology

19CS204 OBJECT ORIENTED PROGRAMMING

I YEAR /II SEMESTER

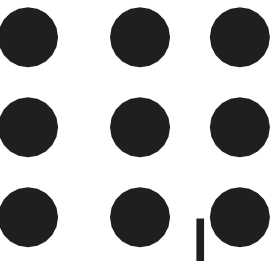
Topic – Packages





Packages

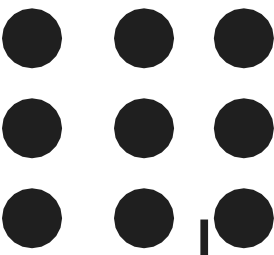
- Packages are containers for classes. They are used to keep the class name space compartmentalized.
- Packages are stored in a hierarchical manner and are explicitly imported into new class definitions.
- Package in Java is a mechanism to encapsulate a group of classes, sub packages and interfaces
- A java package is a group of similar types of classes, interfaces and sub-packages.
- Packages are divided into two categories:
 - Built-in Packages (packages from the Java API)
 - User-defined Packages (create your own packages)



User Defined Packages

Packages are used for:

- Preventing naming conflicts. For example there can be two classes with name Employee in two packages, college.staff.cse.Employee and college.staff.ee.Employee
- Java package is used to categorize the classes and interfaces so that they can be easily maintained.
- Java package provides access protection.
- Packages can be considered as data encapsulation



User Defined Packages

- To create a package is quite easy: simply include a package command as the first statement in a Java source file.
- Any classes declared within that file will belong to the specified package.
- The package statement defines a name space in which classes are stored.
- This is the general form of the package statement:
`package pkg;`
- Here, pkg is the name of the package. For example, the following statement creates a package called MyPackage:
`package MyPackage;`



User Defined Packages



```
package MyPack;
class Balance {
String name;
double bal;
Balance(String n, double b) {
name = n;
bal = b;
}
void show() {
if(bal<0)
System.out.print("--> ");
System.out.println(name + ": $" + bal);
}
}
```

```
class AccountBalance {
public static void main(String args[]) {
Balance current[] = new Balance[3];
current[0] = new Balance("K. J. Fielding", 123.23);
current[1] = new Balance("Will Tell", 157.02);
current[2] = new Balance("Tom Jackson", -12.33);
for(int i=0; i<3; i++) current[i].show();
}
}
```

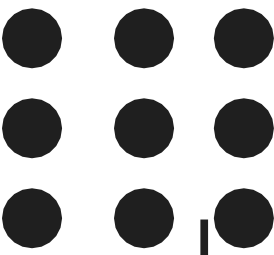


Import in Packages

How to Access Packages

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

- `import package.*;`
- `import package.classname;`
- fully qualified name.





Import in Packages



How to Access Packages

Using `import package.*;` to access all the classes of package

```
// Save it as sample.java
// It belongs to package Pack1
package Pack1;
public class sample
{
    public void msg()
    {
        System.out.println("This is Pack1 sample ");
    }
}
```

```
// Save it as sample2.java
//It also belong to package Pack1
package Pack1;
public class sample2
{
    public void display()
    {
        System.out.println("This is Pack1 sample2 ");
    }
}
```




Import in Packages



How to Access Packages

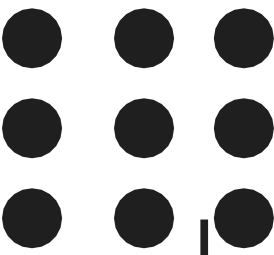
Using `import package.*;` to access all the classes of package

```
// Save as sample1.java
// This belongs to Pack2
package Pack2;
import Pack1.*; // Importing all the classes of Pack1 i.e Both classes sample and sample2

class sample1 {
    public static void main(String args[]){
        sample obj = new sample();
        sample2 obj1 = new sample2();
        obj.msg();
        obj1.display();
    }
}
```




Import in Packages



How to Access Packages

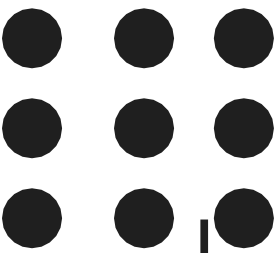
Using `import package.class name;` to access only a particular class of package

```
// Save as sample1.java
// This belongs to Pack2
package Pack2;
import Pack1.sample; // Importing only sample class in Pack1

class sample1 {
    public static void main(String args[]){
        sample obj = new sample();
        obj.msg();
    }
}
```



Import in Packages



How to Access Packages

Using fully qualified name to access a class in package Pack1

Here no need to use import package statement

```
// Save as sample1.java
```

```
// This belongs to Pack2
```

```
package Pack2;
```

```
class sample1 {
```

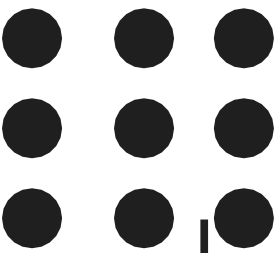
```
    public static void main(String args[]){
```

```
        Pack1.sample obj = new Pack1.sample();
```

```
        obj.msg();
```

```
    }
```

```
}
```



Import in Packages

Built-In Packages

- The Java API is a library of prewritten classes, that are free to use, included in the Java Development Environment.
- The library is divided into packages and classes. Meaning you can either import a single class (along with its methods and attributes), or a whole package that contain all the classes that belong to the specified package.
- To use a class or a package from the library, you need to use the import keyword:

Syntax

```
import package.name.Class; // Import a single class
```

```
import package.name.*; // Import the whole package
```



Import in Packages

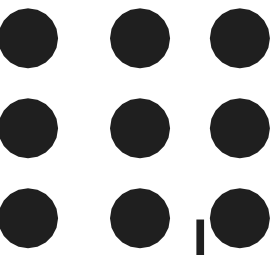


In java we have several built-in packages, for example when we need user input, we import a package like this:

```
import java.util.Scanner
```

Here:

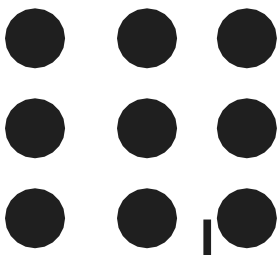
- java is a top level package
- util is a sub package
- and Scanner is a class which is present in the sub package util.



Import in Packages

Some of the commonly used built-in packages are:

- 1) **java.lang:** Contains language support classes(e.g classed which defines primitive data types, math operations). This package is automatically imported.
- 2) **java.io:** Contains classed for supporting input / output operations.
- 3) **java.util:** Contains utility classes which implement data structures like Linked List, Dictionary and support ; for Date / Time operations.
- 4) **java.applet:** Contains classes for creating Applets.
- 5) **java.awt:** Contain classes for implementing the components for graphical user interfaces (like button , ;menus etc).
- 6) **java.net:** Contain classes for supporting networking operations.



THANK YOU