Core Java

UNIT III – DEFAULT AND STATIC METHODS IN INTERFACE, FUNCTIONAL INTERFACES AND PACKAGES

01

interface interface1
---------- interface2

02

What is the concept of nesting interface?

03

What is the use of Lambda Expression?

Review

Default Methods in Interface

The default methods were introduced to provide backward compatibility so that existing interfaces can use the lambda expressions without implementing the methods in the implementation class.

They are also known as **defender methods** or **virtual extension methods**.

Example

```
interface TestInterface
{   // abstract method
     public void square(int a);

     // default method
      default void show() {
      System.out.println("Default Method Executed");
    }
 }
```

Static Methods in Interface

The interfaces can have static methods as well which is similar to static method of classes.

Example

```
interface TestInterface
{
    // abstract method
    public void square (int a);
    // static method
    static void show()
    {
        System.out.println("Static Method Executed");
    }
}
```

Functional Interfaces

A functional interface is an interface that contains only one abstract method. They can have only one functionality to exhibit.

lambda expressions can be used to represent the instance of a functional interface.

A functional interface can have any number of default methods.

Runnable, ActionListener are some of the examples of functional interfaces.

Annotate your functional interfaces with **@FunctionalInterface**.

Functional Interfaces

Example

```
@FunctionalInterface

public interface Flower {

    String setName(String name);

    default void defaultMethod() {}
}
```

Packages

Packages are Java’s way of grouping a variety of classes and /or interfaces together.

Benefits

- Reuse of Classes & Interfaces
- Provides a way to “hide” classes thus preventing other programs or packages from accessing classes that are meant for internal use only.
- Provides a way for separating “design ” and “coding ”.
- Classes in two different packages can have the same name.

Package	Contents
java.lang	Languages support classes. These are classes that Java compiler itself uses and therefore automatically imported. String, Math function, exception.
java.util	Languages utility classes such as vectors, hash tables, random numbers, date etc.
java.io	ip/op support classes.
java.awt	Set of classes for implementing user interface.
java.net	Include classes for communicating with local computer as well as with internet servers.
java.applet	Classes for creating and implementing applets.

Java API Packages

Creating Packages

Steps to create your own package

1. Declare the package at the beginning of a file using the form
2. package packagename;
3. Define the class that is to be put in the package and declare it public.
4. Create a subdirectory (with same name as package)under the directory where the main source files are stored.
5. Store the listing as the classname.java file in the subdirectory created.
6. Compile the file. This creates .class file in the subdirectory.

Example

```
package p1;
        public class B
        {
        //body of B
         }
```

Thank You