Object Oriented Development with Java

(CT038-3-2 and Version VC1)

Packages

Topic & Structure of The Lesson

- Introduction
- Using Packages
- Accessing Packages
- Packages Naming Conventions
- Package Declaration
- Adding Class to a Package

Learning Outcomes

- At the end of this topic, You should be able to

  -Describe about package

  -Describe how to Use and access a package

  -Describe how to Add a class to a Package

Key terms you must be able to use

If you have mastered this topic, you should be able to use the following terms correctly in your assessments:

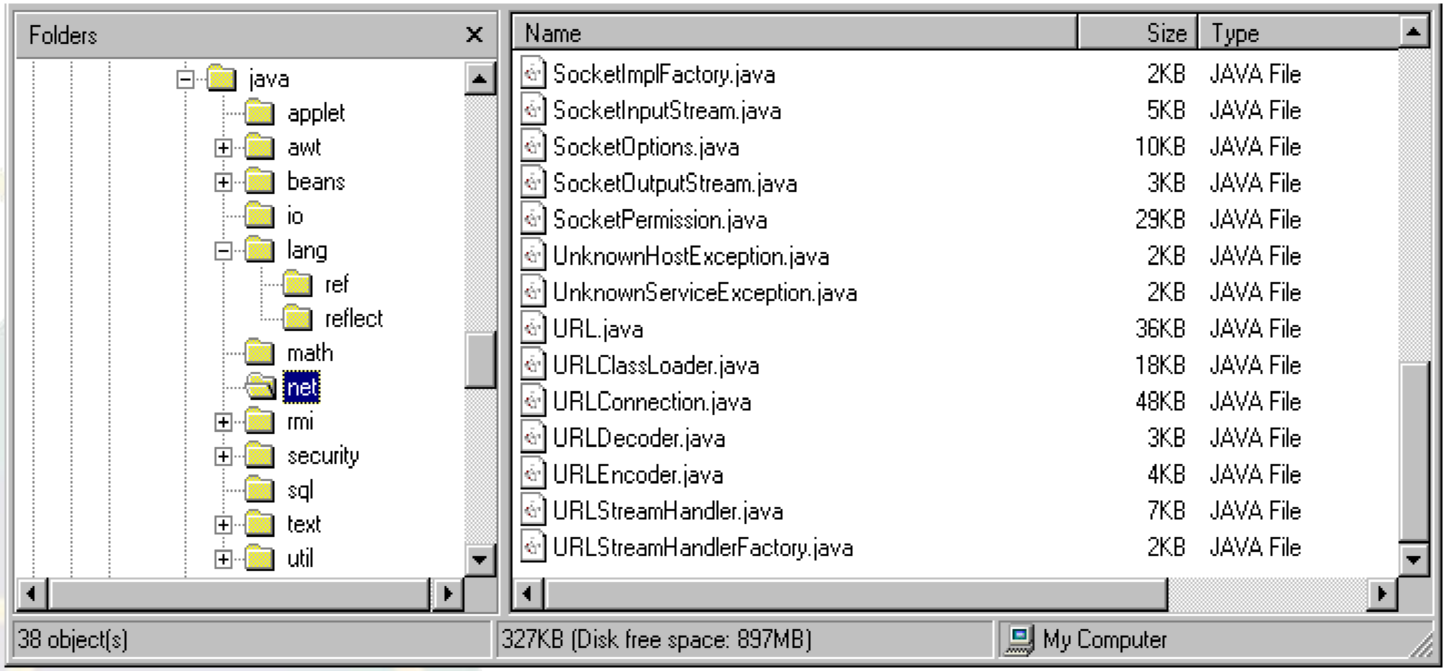
- Package
- Jar files
- Library

Introduction

- Packages are nothing more than the way we organize files into different directories according to their functionality, usability as well as category they should belong to .
- A Java package is a Java programming language mechanism for organizing classes into namespaces.

Intodunction

- Java source files belonging to the same category or providing similar functionality can include a **package** statement at the top of the file to designate the package for the classes the source file defines.
- Java packages can be stored in compressed files called JAR files.
- An obvious example of packaging is the JDK package from SUN (java.xxx.yyy) as shown below:



Name	Size	Type
SocketImplFactory.java	2KB	JAVA File
SocketInputStream.java	5KB	JAVA File
SocketOptions.java	10KB	JAVA File
SocketOutputStream.java	3KB	JAVA File
SocketPermission.java	29KB	JAVA File
UnknownHostException.java	2KB	JAVA File
UnknownServiceException.java	2KB	JAVA File
URL.java	36KB	JAVA File
URLClassLoader.java	18KB	JAVA File
URLConnection.java	48KB	JAVA File
URLDecoder.java	3KB	JAVA File
URLEncoder.java	4KB	JAVA File
URLStreamHandler.java	7KB	JAVA File
URLStreamHandlerFactory.java	2KB	JAVA File

Introduction

- Packaging also help us to avoid class name collision when we use the same class name as that of others.

- For example, if we have a class name called "Vector", its name would crash with the Vector class from JDK. However, this never happens because JDK uses java.util as a package name for the Vector class (java.util.Vector).

- Understanding the concept of a package will also help us manage and use files stored in jar files in more efficient ways.

Using Packages

- To use a package inside a Java source file, it is convenient to import the classes from the package with an import statement.
- import java.awt.event.*;
- The above statement imports all classes from the java.awt.event package.

Package access protection

- Classes within a package can access classes and members declared with *default access* and class members declared with the *protected* access modifier.
- Default access is enforced when neither the public, protected nor private access modifier is specified in the declaration.

Creation Of Jar Files

- In Java source files the package the file belongs to is specified with the package keyword .
- package java.awt.event;
- JAR Files are created with the jar command-line utility.
- The command “jar cf myPackage.jar *.class” compresses all *.class files into the JAR file *myPackage.jar*.

Package Naming Conventions

- Packages are usually defined using a hierarchical naming pattern, with levels in the hierarchy separated by periods (.) .
- Although packages lower in the naming hierarchy are often referred to a "subpackages" of the corresponding packages higher in the hierarchy, there is no semantic relationship between packages.

Package Declaration

- Package declaration is file based;
  - All classes in the same source file belong to the same package.
  - Each source file may contain an optional package declaration in the following form.

    Package packagename;
  - Let us consider the source file ElevatorFrame.java, for example.

    Package elevator;

    Public class ElevatorFrame

    { public double x; //……..}

Package Declaration

- The package declaration at the top of the source file declares that the ElevatorFrame class belongs to the package named elevator.
- When the package declaration is absent from a file, all the classes contained in the file belong to unnamed package.
- A class in a named package can be referred in two ways.

Using Packages

– Class in a named package can be referred to in two different ways
  - Using the fully qualified name packagename.ClassName
  - We can refer to the ElevatorPanel class in package elevator as

    elevator.ElevatorPlanel

Importing a class in the package

- Importing the class using the simple class name
  - We can import a class or all the classes in the designated package using

    Import packagename.ClassName;

    Import packagename.*;
  - The ElevatorPanel class in package elevator can simply be referred to as elevator when either of the following import clauses occurs at the top of source file

    Import elevator.ElevatorPanel;

    Import elevator.*;

Quick Review Question

- What is Package
- How to access a Package
- How to declare a Package

Summary of Main Teaching Points

-Introduction
-Using Packages
-Accessing Packages
-Packages Naming Conventions
-Package Declaration
-Adding Class to a Package

Question and Answer Session

Q & A

Next Session

- Exception Handler
- Exception Class
- Handling Exception
  - Handling multiple exceptions
  - `finally` clause
  - Checked and unchecked exceptions
  - `throw` exception
- Creating (your own) exception class