Java Packages

Packages, a method of subdividing a Java program and grouping classes

Decomposing Object-Oriented Programs Only By Classes

Works well for small programs e.g.,

- Trek
- GameStatus
- Galaxy
- StarShip
- CommandProcessor

Decomposing Larger Object-Oriented Programs

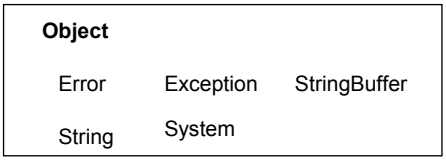
There is another tool to group related classes: packages.

- **Java.lang:** classes that included with the 'core' portion of the Java language:
  - String
  - Math
  - :
- **Java.util.zip:** classes that allow for the reading and writing to zip and gzip compressed files:
  - ZipInputStream
  - ZipOutputStream
  - :
- **Java.awt:** the original collection of classes used for creating graphical user interfaces:
  - Button
  - Menu
  - :

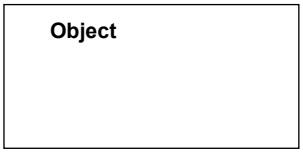
Benefits Of Employing Packages

- Increased ease finding a class
- Can be used to prevent naming conflicts

java.lang

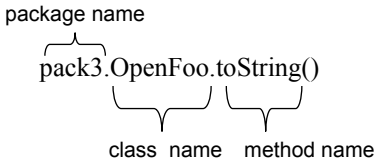


org.omg.CORBA



- An additional permission level (package level) may be set to allow certain classes to be instantiated only within the methods of the classes that belong to the same package

Fully Qualified Names



pack3.ClosedFoo.toString()

Importing Packages

Importing all classes from a package

Format

import <*package name*>.*;

Example

import java.util.*;

Importing a single class from a package

Format

import <package name>.<class name>;

Example

import java.util.Vector;

Importing Packages (2)

When you do not need an import statement:

- When you are using the classes in the java.lang package.
- You do not need an import statement in order to use classes which are part of the same package

Excluding the import requires that the full name be provided:

```
java.util.Random generator = new java.util.Random ();
```

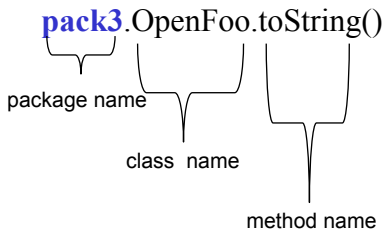
Vs.

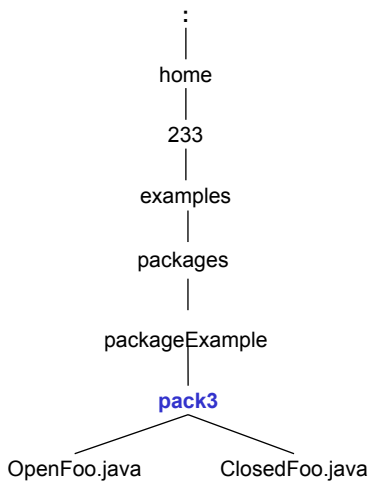
```
import java.util.Random;
Random generator = new Random ();
```


Default Package

- If you do not use a package statement then the class implicitly becomes part of a default package
- All classes which reside in the same directory are part of the default package for that program.

Fully Qualified Names: Matches Directory Structure





Where To Match Classes To Packages

1. In directory structure: The classes that belong to a package must reside in the directory with the same name as the package (previous slide).
2. In the classes' source code: At the top class definition you must indicate the package that the class belongs to.

Format:

package <*package name*>;
<*visibility – public or package*> class <*class name*>
{

}

Matching Classes To Packages (2)

Example

```
package pack3;
public class OpenFoo
{
    :
}

package pack3;
    class ClosedFoo
{
    :
}
```


Matching Classes To Packages (2)

Example

```
package pack3;
public class OpenFoo
{
    :
}

package pack3;
    class ClosedFoo
{
    :
}
```

Public access: Class can be instantiated by classes that aren't a part of package pack3

Package access (default): Class can only be instantiated by classes that are a part of package pack3

Sun's Naming Conventions For Packages

Based on Internet domains (registered web addresses)
e.g., www.tamj.com

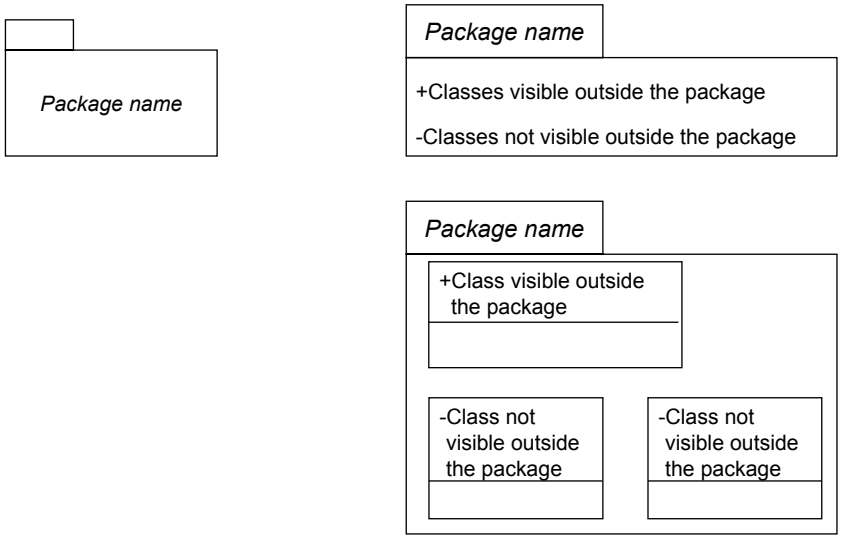
com.tamj .games
.productivity

Sun's Naming Conventions For Packages

Alternatively it could be based on your email address
e.g., tamj@cpsc.ucalgary.ca

ca.ucalgary.cpsc.tamj .games
.productivity

Graphically Representing Packages In UML

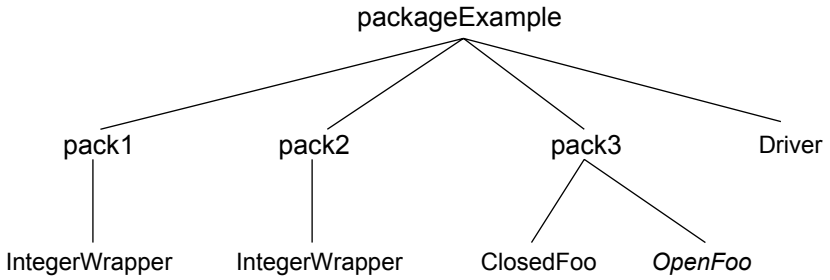


James Tam

Packages An Example

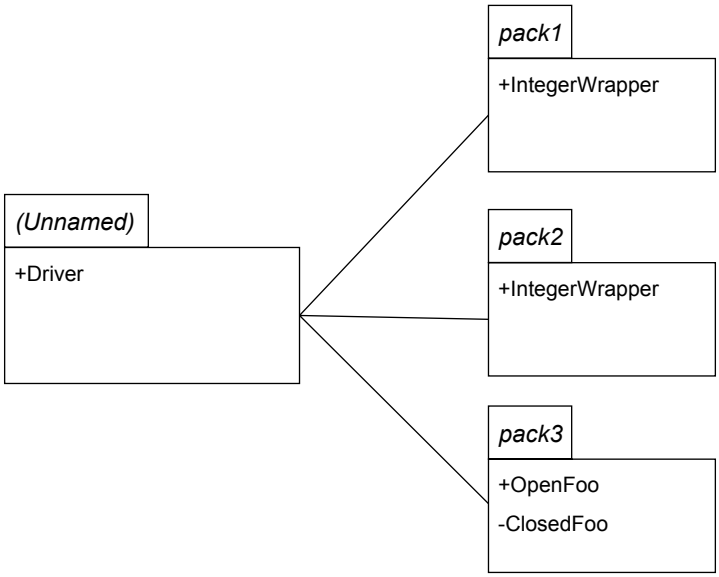
The complete example can be found UNIX in the directory:
/home/coures/219/examples/packages/packageExample

(But you should have guessed the path from the package name)



James Tam

Graphical Representation Of The Example



Package Example: The Driver Class

```
import pack3.*;
public class Driver
{
   public static void main (String [] argv)
   {
      pack1.IntegerWrapper iw1 = new pack1.IntegerWrapper ();
      pack2.IntegerWrapper iw2 = new pack2.IntegerWrapper ();
      System.out.println(iw1);
      System.out.println(iw2);

      OpenFoo of = new OpenFoo ();
      System.out.println(of);
      of.manipulateFoo();
    }
}
```

Package Example: Package Pack1, Class IntegerWrapper

```
package pack1;
public class IntegerWrapper
{
   private int num;

   public IntegerWrapper ()
   {
      num = (int) (Math.random() * 10);
   }
   public IntegerWrapper (int newValue)
   {
      num = newValue;
   }
   public void setNum (int newValue)
   {
      num = newValue;
   }
```


Package Example: Package Pack1, Class IntegerWrapper (2)

```
   public int getNum ()
   {
      return num;
   }

   public String toString ()
   {
      String s = new String ();
      s = s + num;
      return s;
   }
}
```

Package Example: Package Pack2, Class IntegerWrapper

```
package pack2;
public class IntegerWrapper
{
   private int num;

   public IntegerWrapper ()
   {
      num = (int) (Math.random() * 100);
   }
   public IntegerWrapper (int newValue)
   {
      num = newValue;
   }
   public void setNum (int newValue)
   {
      num = newValue;
   }
```


Package Example: Package Pack2, Class IntegerWrapper (2)

```
    public int getNum ()
   {
      return num;
   }

   public String toString ()
   {
      String s = new String ();
      s = s + num;
      return s;
   }
}
```

Package Example: Package Pack3, Class OpenFoo

```
package pack3;
public class OpenFoo
{
   private boolean bool;
   public OpenFoo () { bool = true; }
   public void manipulateFoo ()
   {
      ClosedFoo cf = new ClosedFoo ();
      System.out.println(cf);
   }
   public boolean getBool () { return bool; }
   public void setBool (boolean newValue) { bool = newValue; }
   public String toString ()
   {
      String s = new String ();
      s = s + bool;
      return s;
   }
}
```


Package Example: Package Pack3, Class ClosedFoo

```
package pack3;
class ClosedFoo
{
   private boolean bool;

   public ClosedFoo () { bool = false; }
   public boolean getBool () { return bool; }

   public void setBool (boolean newValue) { bool = newValue; }

   public String toString ()
   {
      String s = new String ();
      s = s + bool;
      return s;
   }
}
```

Updated Levels Of Access Permissions: Attributes And Methods

Private "-"

- Can only access the attribute/method in the methods of the class where it's originally defined.

Protected "#"

- Can access the attribute/method in the methods of the class where it's originally defined or the subclasses of that class or in classes of the same package.

Package - no UML symbol for this permission level

- Can access the attribute/method from the methods of the classes within the same package
- *For Java: If the level of access is unspecified in a class definition this is the default level of access*

Public "+"

- Can access attribute/method anywhere in the program

James Tam

Updated Levels Of Access Permissions

Access level	Accessible to			
	Same class	Class in same package	Subclass in a different package	Not a subclass, different package
Public	**Yes**	**Yes**	**Yes**	**Yes**
Protected	**Yes**	**Yes**	**Yes**	No
Package	Yes	Yes	No	No
Private	Yes	No	No	No

James Tam

Updated Levels Of Access Permissions

Access level	Accessible to			
	Same class	Class in same package	Subclass in a different package	Not a subclass, different package
Public	**Yes: e.g., #1**	**Yes: e.g., #5**	**Yes: e.g., #9**	**Yes: e.g., #13**
Protected	**Yes: e.g., #2**	**Yes: e.g., #6**	**Yes: e.g., #10**	No: e.g., #14
Package	**Yes: e.g., #3**	**Yes: e.g., #7**	No: e.g., #11	No: e.g., #15
Private	**Yes: e.g., #4**	No: e.g., #8	No: e.g., #12	No, e.g., #16

Levels Of Permission, Same Class

Within the methods of the class, all attributes and methods may be accessed.

```
public class ClassOne
{
   public int num1;
   protected int num2;
   int num3;
   private int num4;

   public ClassOne ()
   {
      num1 = 1;   // Example #1
      num2 = 2;   // Example #2
      num3 = 3;   // Example #3
      num4 = 4;   // Example #4
   }
}
```

Levels Of Permission, Class In Same Package

```
package pack1;
public class ClassOne
{
   public int num1;
   protected int num2;
   int num3;
   private int num4;
}
```

```
package pack1;
public class ClassTwo
{
   private ClassOne c1;
   public ClassTwo ()
   {
      c1 = new pack1.ClassOne ();
      c1.num1 = 1;              // Example #5
      c1.num2 = 2;              // Example #6
      c1.num3 = 3;              // Example #7
      // c1.num4 = 4;           // Example #8
   }
}
```


Levels Of Permission, Subclass In Different Package

```
package pack1;
public class ClassOne
{
   public int num1;
   protected int num2;
   int num3;
   private int num4;
}
```

```
package pack2;
import pack1.ClassOne;
public class ClassThree extends ClassOne
{
   public ClassThree ()
   {
      super.num1 = 1;     // Example #9
      super.num2 = 2;     // Example #10
      // super.num3 = 3;  // Example #11
      // super.num4 = 4;  // Example #12
   }
}
```

Levels Of Permission, Not A Subclass, Not In Same Package

```
package pack1;
public class ClassOne
{
   public int num1;
   protected int num2;
   int num3;
   private int num4;
}
```

```
public class Driver
{
   public static void main (String [] args)
   {
      pack1.ClassOne c1 = new pack1.ClassOne ();
      c1.num1 = 1;        // Example #13
      // c1.num2 = 2;     // Example #14
      // c1.num3 = 3;     // Example #15
      // c1.num4 = 4;     // Example #16
   }
}
```


You Should Now Know

How packages work in Java

- How to utilize the code in pre-defined packages
- How to create your own packages

- How the 4 levels of access permission work in conjunction with classes in the same package, sub classes and classes that are neither in the same subclass nor in the same package.