# Packages

CSC 116 – Section 002
February 2, 2004

---

# Packages

- Collection of classes
  - Perform similar work
  - Represent similar items
  - Ex: I/O, lang
- Objects in a package are in the same directory
- Syntax:

  package *<name of package>;*
  - The package is listed at the top of the class file, before imports and the class declaration
  - Tells the object what package it belongs too

# Packages (2)

- Each level of the package is a subdirectory
  - Ex: java.awt.Rectangle
  - java/awt/Rectangle.class
- The package path is where packages are stored
- When you compile an object with a package name specified, the output is put in the correct directory based on the package path

# Referring to Packages

- If you import a package or object in a package into a class, then you may refer to the object by just the class name
- Ex:

  import java.awt.Rectangle;

  public class myClass {

      private Rectangle myRectangle;

  }

# Referring to Packages (2)

- Import the entire package by using a * instead of the class name
- You can refer to any object in the package by its class name and not the full package name
- Ex:

import java.awt.*;

public class myClass {

    private Rectangle myRectangle;

    private Polygon myPolygon;

}

# Referring to Packages (3)

- If you do not import a package, then you must refer to the object using the package name
- Ex:

public class myClass {

    private java.awt.Rectangle myRectangle;

    private java.awt.Polygon myPolygon;

}

# Referring to Packages (4)

- Objects in the same directory are considered to be in the same package.
- You can refer to objects in the same directory without specifying the package name.

# Has-a Relationship

- Also known as aggregation
- An object is a collection of parts
- These parts are other objects
- Therefore one object has other objects.
- Ex:

  A car has wheels, doors, seats, etc.

  myClass has a Rectangle and Polygon

# References

- Jason Schwarz's Lecture 7 slides:
  http://courses.ncsu.edu/csc116/