Packages

In the previous lecture, we used the class name `javax.swing.JFrame`. In this little lecture, we show how it can be abbreviated as `JFrame`.

In Java, a **package** is a collection of classes that reside in the same directory on your hard drive. In our file-cabinet analogy, think of a package as a room that contains file cabinets with one drawer for each class.

The package is Java’s way of organizing a large collection of classes — Java version 6 comes with over 3775 classes, organized into about 200 packages. Here are four of the packages.

```
package java.io       classes dealing with input/output
package java.net     classes dealing with the internet
package java.awt     classes dealing with GUIs
package javax.swing  newer classes dealing with GUIs
```

These package names act as paths on your hard drive. From the above package names, we can see that there are two main directories, `java` and `javax`, and within directory `java`, there are directories `io`, `net`, and `awt`. Below, we show this directory structure, giving a few classes that are in directory `awt` and `swing`.

```
java
  io
  net
  awt
    Frame
    Button
    TextField
javax
  swing
    JFrame
    JButton
```

You can form your own packages of classes that you write, but we don’t show you how to do that here.

**The import statement**

In the interactions pane of DrJava, let’s type the assignment statement

```
j= new JFrame();
```

and hit the enter key. You see that there is an error: class `JFrame` is undefined.

Now let’s type the import statement

```
import javax.swing.*;
```

This statement gives your program access to all classes in package `javax.swing`, without needing to give the complete path. The asterisk indicates “all classes” in the package.

So, if we type the assignment statement again, there is no error, and we can also show the new JFrame — there it is.

If you want to “import”, or make accessible, only one class in a package, then use its name instead of the asterisk. For example, after importing `java.awt.Frame`, we can access it but no other class in package `java.awt`.

You have seen the use of the import statement in the interactions pane. When we begin writing classes, we’ll show you where to put it.
Packages