Packages

- **Package:** Set of related classes
- Important packages in the Java library:

Package	Purpose	Sample Class
`java.lang`	Language support	`Math`
`java.util`	Utilities	`Random`
`java.io`	Input and output	`PrintStream`
`java.awt`	Abstract Windowing Toolkit	`Color`
`java.applet`	Applets	`Applet`
`java.net`	Networking	`Socket`
`java.sql`	Database Access	`ResultSet`
`javax.swing`	Swing user interface	`JButton`
`omg.w3c.dom`	Document Object Model for XML documents	`Document`

Organizing Related Classes into Packages

- To put classes in a package, you must place a line

```
package packageName;
```

  as the first instruction in the source file containing the classes
- Package name consists of one or more identifiers separated by periods

Organizing Related Classes into Packages

- For example, to put the `Financial` class introduced into a package named `com.horstmann.bigjava`, the `Financial.java` file must start as follows:

```
package com.horstmann.bigjava;

public class Financial
{
   ...
}
```

- Default package has no name, no `package` statement

Syntax 8.2 Package Specification

Syntax `package` *packageName*;

Example

```
package com.horstmann.bigjava;
```

The classes in this file belong to this package.

A good choice for a package name is a domain name in reverse.

Importing Packages

- Can always use class without importing:

```
java.util.Scanner in = new java.util.Scanner(System.in);
```

- Tedious to use fully qualified name
- Import lets you use shorter class name:

```
import java.util.Scanner;
...
Scanner in = new Scanner(System.in)
```

- Can import all classes in a package:

```
import java.util.*;
```

- Never need to import `java.lang`
- You don't need to import other classes in the same package

Package Names

- Use packages to avoid name clashes

```
java.util.Timer
```

vs.

```
javax.swing.Timer
```

- Package names should be unambiguous
- Recommendation: start with reversed domain name:

```
com.horstmann.bigjava
```

- `edu.sjsu.cs.walters`: for Britney Walters' classes (`walters@cs.sjsu.edu`)
- Path name should match package name:

```
com/horstmann/bigjava/Financial.java
```

Package and Source Files

- **Base directory:** holds your program's Files
- Path name, relative to base directory, must match package name:

```
com/horstmann/bigjava/Financial.java
```

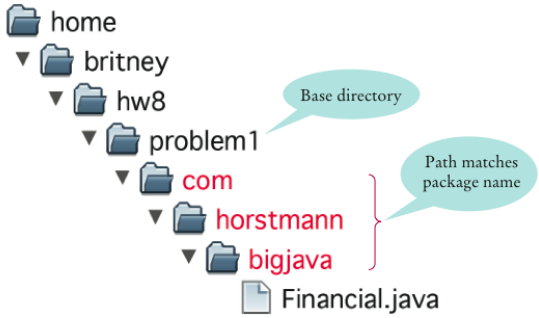

Figure 5
Base Directories and Subdirectories for Packages

Self Check 8.18

Which of the following are packages?

a. `java`

b. `java.lang`

c. `java.util`

d. `java.lang.Math`

Answer:

a. No

b. Yes

c. Yes

d. No

Self Check 8.19

Is a Java program without `import` statements limited to using the default and `java.lang` packages?

Answer: No — you simply use fully qualified names for all other classes, such as `java.util.Random` and `java.awt.Rectangle`.