C O M P 4 0 1
P A C K A G E S

Instructor: Prasun Dewan
PREREQUISITES

- Objects
GROUPING FACTORIES

Car Factories

Honda factories

Accord Factories

Civic Factories

Corvette Factories

Tablet Factories

Wuhan Factory

Indiana Factory
GROUPING CLASSES
GROUPING CLASSES

Package

- lectures.functions
  - ABMIcalculator.java
  - ABMIMetricConverter.java
  - AMetricConverterWithErrors.java
  - AMonolithicPoundInchBMIcalculator.java
  - AMyAverageBMIcalculator.java
  - AMyBMICalculator.java
  - APoundInchBMIcalculator.java
  - ASquareCalculator.java
  - FunctionsDriver.java
  - SquareCalculatorDriver.java

- lectures.generics
- lectures.graphics
- lectures.inheritance
- lectures.inheritance.deep_shallow_copy
- lectures.inheritance.equals_polymorphism_overloading
- lectures.inheritance.virtual
A SIMPLE INSTANTIATED CLASS

```java
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

public class SquareCalculatorDriver {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
```
**Packages**

```java
package lectures.functions;

public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
package main;
import lectures.functions.ASquareCalculator;

public class SquareCalculatorDriver {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
```

Class in different package must be imported using full name of class (a la full file name).
```java
package lectures.functions;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
package lectures.functions;
public class SquareCalculatorDriver {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
```
No package means package named default, hence import needed
```java
public class ASquareCalculator {
    
    public int square(int x) {
        return x * x;
    }
}

public class SquareCalculatorDriver {

    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}

No package means package named default, hence no import needed here
```
```java
class ASquareCalculator {
    public int square(int x) {
        return x * x;
    }
}

class SquareCalculatorDriver {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
```
package lectures.functions;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package main;
public class SquareCalculatorDriver {
    public static void main (String[] args) {
        lectures.functions.ASquareCalculator squareCalculator = new lectures.functions.ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
LONG NAME WITH NO IMPORT

```java
package lectures.functions;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
package main;
import lectures.functions.*;
public class SquareCalculatorDriver {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
```

Works but does not tell the reader what is being imported from the package name.

Important role of import is documentation.

Programming style violation.
**WHY IMPORTS/FULL NAME?**

```java
class ASquareCalculator {
    public long square(int x) {
        return x*x;
    }
}

class SquareCalculatorDriver {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
```

**Twice the size of ints**

Disambiguates
Ambiguous Import

```java
package lectures.functions;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package lectures.safe_functions;
public class ASquareCalculator {
    public long square(int x) {
        return x*x;
    }
}

package main;
import lectures.functions.ASquareCalculator;
import lectures.safe_functions.ASquareCalculator;
public class SquareCalculatorDriver {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
```
UNSUCCESSFUL IMPORT

package lectures.functions;

class ASquareCalculator {

    public int square(int x) {
        return x*x;
    }

}

package lectures.safe_functions;

public class ASquareCalculator {

    public long square(int x) {
        return x*x;
    }

}

package main;

import lectures.functions.ASquareCalculator;

public class SquareCalculatorDriver {

    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }

}
WHY PACKAGES?

- Can create competing implementations of same class.
  - A la creating files Test.java in different assignment directories/folders
- Groups related classes together
- Can browse/search for related classes
  - A la browsing through all files in an assignment directory/folder.
- Like directories/folders packages can be hierarchical

```java
package recitations.functions;
package lectures.functions;
```
- Provides documentation of what unrelated classes are being used
BROWSING JAVA CLASSES

package lectures.functions;

public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

Package lectures.functions:
- ABMICalculator
- AMetricConverter
- ASquareCalculator
BROWSING JAVA CLASSES

Very useful package
Built-in classes do not have to be explicitly imported.
STARTING OBJECT EDITOR INTERACTIVELY

Java: `classpath .;oeall20.jar Comp110ObjectEditor`

Can we tell short circuit this step?
Starting **ObjectEditor** Programatically

```java
package lectures.functions;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package main;
import lectures.functions.ASquareCalculator;
import bus.uigen.ObjectEditor;
public class SquareCalculatorDriver {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        ObjectEditor.edit(squareCalculator);
    }
}
```

ObjectEditor is predefined packaged class
No Need to Enter Class Name
Remembering Package Names?

```java
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
package main;

public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        ObjectEditor.edit(squareCalculator);
    }
}
```

What if we do not remember the package names?

In Eclipse CTRL_SHIFT_O
Automatic Imports in Eclipse

```java
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x * x;
    }
}
```

```java
package main;
import math.ASquareCalculator;
import bus.uigen.ObjectEditor;
public class SquareCalculatorTester {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        ObjectEditor.edit(squareCalculator);
    }
}
```

Automatically added
Gives a dialogue box in case multiple classes in same package
Package names will not be given in class examples
PROGRAMMED MAIN AND METHOD CALLS

Editor

creates

ASquareCalculator Source Code

reads

Compiler

writes

ASSquareCalculator Object (byte) Code

Instantiates

ASSquareCalculator Instance

SquareCalculatorDriver

main

calls

calls

Interpreter

square

calls
Interactive Main and Method Invocation

- Editor
  - creates
  - asSquareCalculator Source Code
- Compiler
  - reads
  - writes
  - asSquareCalculator Object (byte) Code
- asSquareCalculator Instance
  - square
  - calls
- ObjectEditor
  - main
  - instantiates
- Interpreter
  - calls
PROGRAMMED MAIN AND INTERACTIVE METHOD INVOCATION

Editor

ASquareCalculator Source Code

Compiler

ASquareCalculator Object (byte) Code

ASquareCalculator Instance

square

calls

ObjectEditor

edit

calls

SquareCalculatorDriver

main

instantiates

Interpreter
Main method starts the computation, and can call other methods.

Can put complete program in main method

Like having one big paragraph in an essay

Method decomposition important modularization technique even in conventional programming

In O-O programming multiple classes involved in computation
TWO CLASSES

```java
package lectures.functions;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
package main;
import lectures.functions.ASquareCalculator;
import bus.uigen.ObjectEditor;
public class SquareCalculatorDriver {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        ObjectEditor.edit(squareCalculator);
    }
}
```
COMBINING THE CLASSES FOR TESTING

```java
package lectures.functions;
import bus.uigen.ObjectEditor;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }

    public static void main (String[] args) {
        ObjectEditor.edit(new ASquareCalculator());
    }
}
```

```
java –classpath .;oeall20.jar ASquareCalculator

Convenient but possibly confusing!
```
**Programmed Main and Interactive Method Invocation**

- Editor creates `ASquareCalculator` Source Code
- Compiler reads `ASquareCalculator` Source Code
- Compiler writes `ASquareCalculator` Object (byte) Code
- `ASquareCalculator` Object (byte) Code instantiates `ASSquareCalculator` Instance
- `ASSquareCalculator` Instance main calls `Interpreter`
- `Interpreter` calls `ObjectEditor`
- `ObjectEditor` edit calls `ASSquareCalculator` Instance square
- `ASSquareCalculator` Instance square calls `ASSquareCalculator` Instance main

32
Extra Slides
```java
public class ASquareCalculator {
    public int square(int x) {
        return x * x;
    }
}

public class SquareCalculatorTester {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
```
```java
package math;

public class ASquareCalculator {
    public int square(int x) {
        return x * x;
    }
}

package main;
import math.ASquareCalculator;

public class SquareCalculatorTester {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
```

Class in different package must be imported using full name of class (a la full file name)
```java
package math;

public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package math;

public class SquareCalculatorTester {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
```

Class in same package need not be imported
**Packages**

```java
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

No package means package named default, hence import needed

```java
import math.ASquareCalculator;
public class SquareCalculatorTester {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
```
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
public class ASquareCalculator {
    public int square(int x) {
        return x * x;
    }
}

package main;
public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package main;

public class SquareCalculatorTester {
    public static void main(String[] args) {
        math.ASquareCalculator squareCalculator = new math.ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
LONG NAME WITH NO IMPORT

package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package main;
import math.*;
public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}

Works but does not tell the reader what is being imported from the package name

Important role of import is documentation

Programming style violation
### Why imports/full name?

```java
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
package safemath;
public class ASquareCalculator {
    public long square(int x) {
        return x*x;
    }
}
```

```java
package main;
import math.ASquareCalculator;
public class SquareCalculatorTester {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println(squareCalculator.square(5));
    }
}
```

Twice the size of ints

Disambiguates
**Ambiguous Import**

```
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package safemath;
public class ASquareCalculator {
    public long square(int x) {
        return x*x;
    }
}
```

```
package main;
import math.ASquareCalculator;
import safemath.ASquareCalculator;
public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
```

Ambiguous
UNSUCCESSFUL IMPORT

package math;
class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package main;
import math.ASquareCalculator;
public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
**Why Packages?**

- Can create competing implementations of same class.
  - A la creating files `Test.java` in different assignment directories/folders
- Groups related classes together
- Can browse/search for related classes
  - A la browsing through all files in an assignment directory/folder.
- Like directories/folders packages can be hierarchical
  ```java
  package math.power;
  public class ACubeCalculator {...}
  ```
- Provides documentation of what unrelated classes are being used
BROWSING JAVA CLASSES

Very useful package
Built-in classes

Do not have to be explicitly imported

Language vs. Library

This document is the API specification for the Java 2 Platform Standard Edition 5.0.

<table>
<thead>
<tr>
<th>Java 2 Platform Packages</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>java.applet</td>
<td>Provides the classes necessary to create an applet uses to communicate with its applet's environment.</td>
</tr>
<tr>
<td>java.awt</td>
<td>Contains all of the classes for creating user interfaces, graphics, and images.</td>
</tr>
<tr>
<td>java.awt.color</td>
<td>Provides classes for color spaces.</td>
</tr>
<tr>
<td>java.awt.datatransfer</td>
<td>Provides interfaces and classes for transfer applications.</td>
</tr>
<tr>
<td>java.awt.dnd</td>
<td>Drag and Drop is a direct manipulation gestural interface system that provides a mechanism for users to interact with objects.</td>
</tr>
<tr>
<td>java.awt.event</td>
<td>Provides interfaces and classes for dealing with AWT events.</td>
</tr>
<tr>
<td>java.awt.font</td>
<td>Provides classes and interfaces relating to font information.</td>
</tr>
<tr>
<td>java.awt.geom</td>
<td>Provides the Java 2D classes for defining and manipulating geometric objects.</td>
</tr>
</tbody>
</table>
public class ASquareCalculator
{
    public int square(int x)
    {
        return x*x;
    }
}

public class SquareCalculatorTester
{
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(5));
    }
}
Changing Parameter

```java
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square(341));
    }
}
```
Rerun Program

How to not re-run program without writing tedious UI code?

```java
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        System.out.println (squareCalculator.square (Integer.parseInt(args[0])));
    }
}
```
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package main;
import math.ASquareCalculator;
import bus.uigen.ObjectEditor;
public class SquareCalculatorTester {
    public static void main(String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        ObjectEditor.edit(squareCalculator);
    }
}
REMEMBERING PACKAGE NAMES?

package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}

package main;
public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        ObjectEditor.edit(squareCalculator);
    }
}
**Automatic Imports in Eclipse**

```java
package math;
public class ASquareCalculator {
    public int square(int x) {
        return x*x;
    }
}
```

```java
package main;
import math.ASquareCalculator;
import bus.uigen.ObjectEditor;
public class SquareCalculatorTester {
    public static void main (String[] args) {
        ASquareCalculator squareCalculator = new ASquareCalculator();
        ObjectEditor.edit(squareCalculator);
    }
}
```

- Automatically added
- Gives a dialogue box in case multiple classes in same package
- Package names will not be given in class examples
## Summary: Java vs. Real-world

<table>
<thead>
<tr>
<th>Java</th>
<th>Real-world</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>Factory</td>
</tr>
<tr>
<td>Computer Object</td>
<td>Manufactured Physical Object</td>
</tr>
<tr>
<td>Method</td>
<td>Operation</td>
</tr>
<tr>
<td>Invoking/Executing a Method</td>
<td>Performing an Operation</td>
</tr>
<tr>
<td>Instance of a Class</td>
<td>Manufactured by a Factory</td>
</tr>
<tr>
<td>Defining/Declaring a Class</td>
<td>Constructing a Factory</td>
</tr>
<tr>
<td>Instantiating a Class</td>
<td>Manufacturing an Object</td>
</tr>
<tr>
<td>Class (Static) Method</td>
<td>Operation on a Factory</td>
</tr>
<tr>
<td>Main Class Method</td>
<td>Initiates computation</td>
</tr>
<tr>
<td>Instance (Non static) method</td>
<td>Operation on an instance of a class</td>
</tr>
<tr>
<td>Instance (Non static) method</td>
<td>Grouping of factories by states, country</td>
</tr>
</tbody>
</table>