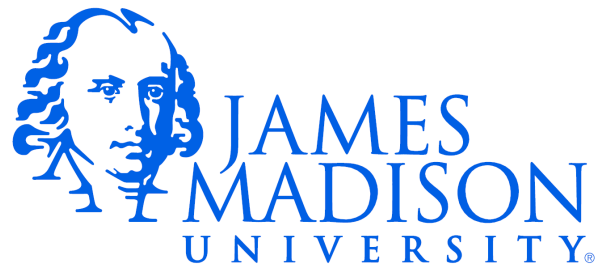


CS159



Why Packages?

- Organizing related classes into packages makes it more convenient to share and re-use those classes for multiple applications.
 - This is related to the concept of cohesion in class design.
- Prevents name conflicts.
 - There are probably 10,000 “Point” classes out there.
 - What happens if I've coded a Point class and I want to use some existing code that already defines a *different* Point class?

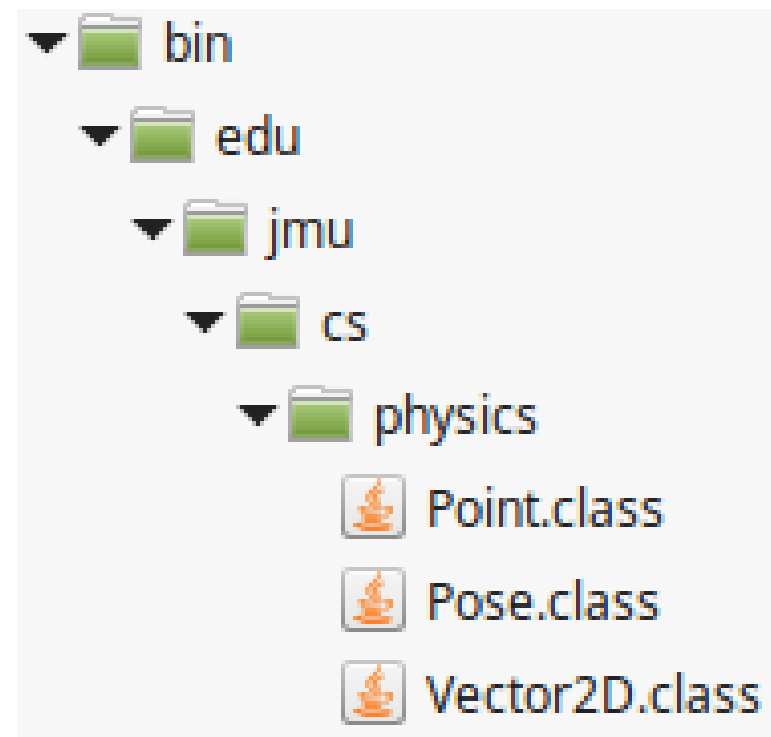
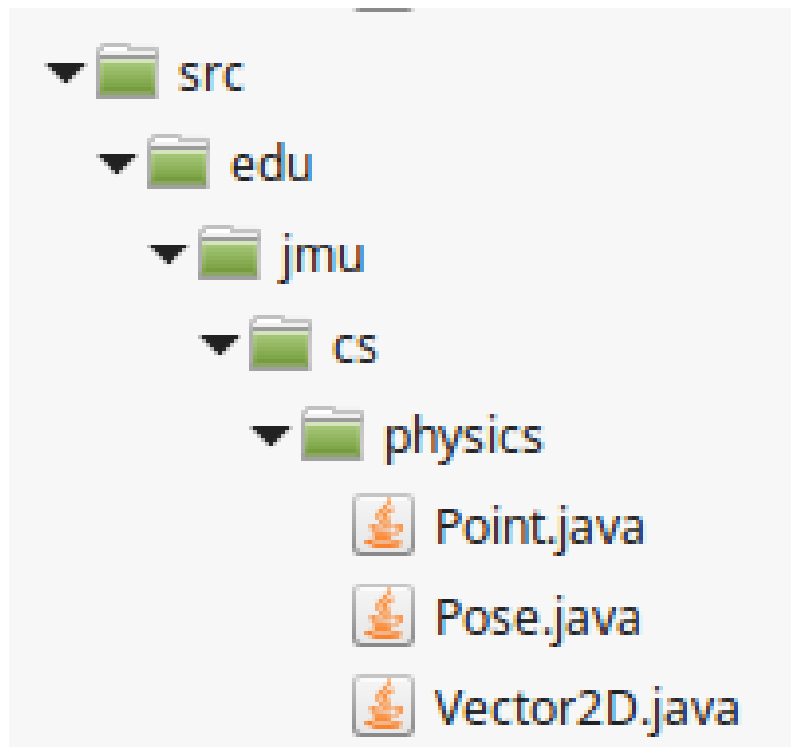
Syntax for Creating Packages

```
package edu.jmu.cs.physics;

public class Point
{
    protected double xPosition;
    protected double yPosition;

    // (Rest of the class not shown)
```

Organizing Packages on Disk



Accessing Classes in Packages

```
public static void main(String[] args)
{
    edu.jmu.cs.physics.Vector2D vec =
        new edu.jmu.cs.physics.Vector2D(0.0, 1.0);

    System.out.println(vec.toString());
}
```

`edu.jmu.cs.physics.Vector2D`
– “fully-qualified” class name

Advantage of Fully Qualified Names

```
public class QualifiedDriver
{
    public static void main(String[] args)
    {
        edu.jmu.cs.physics.Point p1 =
            new edu.jmu.cs.physics.Point(0.0, 1.0);

        java.awt.Point p2 = new java.awt.Point(2, 3);

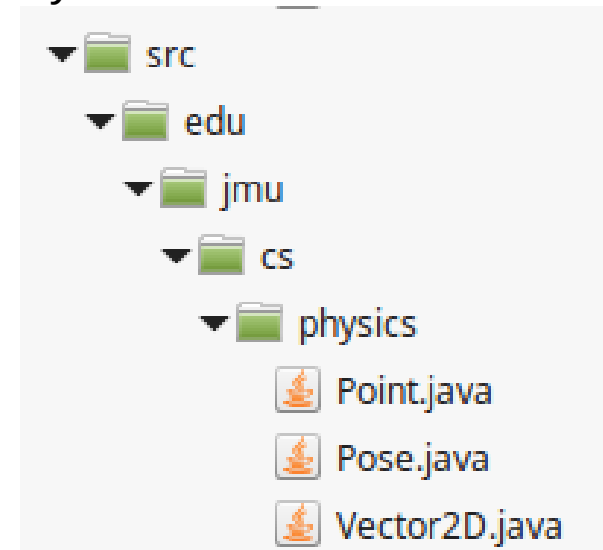
        System.out.println(p1);
        System.out.println(p2);
    }
}
```

Specifying the Class Path

- In the terminal, it is necessary to specify where external code is stored. Something like:

```
javac PhysicsDriver.java -cp /home/spragunr/Physics/
```

▼ Physics



Import Statements

```
import edu.jmu.cs.physics.Vector2D;

public class PhysicsDriver
{
    public static void main(String[] args)
    {
        Vector2D vec = new Vector2D(0.0, 1.0);
        System.out.println(vec.toString());
    }
}
```


Wild-Card Imports

```
import edu.jmu.cs.physics.*;

public class PhysicsDriver
{
    public static void main(String[] args)
    {
        Vector2D vec = new Vector2D(0.0, 1.0);
        System.out.println(vec.toString());
    }
}
```

- Generally frowned upon:
 - Increases the chances for name conflicts.
 - Makes the code harder to read: “Where is this class coming from?”

Static Imports

```
import static java.lang.Math.sqrt;

public class PhysicsDriver
{
    public static void main(String[] args)
    {
        // Normally we would do this:
        System.out.println( Math.sqrt(2) );

        // Import statement allows this:
        System.out.println( sqrt(2) );
    }
}
```

- Again, should not be done routinely.