CS/IT115 Lecture 7

2014-02-24
Alex Dusenbery
Creating our own Classes
Project 1 Due Wednesday

• Main points of project 1:
  – Using arrays
  – Managing array data via loops
  – Using objects and their methods (e.g. Scanner, Point)

• Any more questions on this project?
Object State

- We've seen that objects have a state and that the state of an object can change as different actions are taken (Scanner example)
- How do we handle internal state ourselves?
- We can use fields defined in the class from which the object was instantiated.
public class Point {
    int x;
    int y;
}

• This says that a Point holds an int called x and an int called y.
Our own Point Class

- This is a pretty simplified example, but it works. Allows Points that can hold x and y values inside them.
- However, it doesn't allow us to construct new points from coordinates, as we can with the JDK Point class.
Points

• With our current, simple class, we can do this:

```java
Point p1 = new Point();
// now p1 exists with x = y = 0
System.out.println("p1 is (" + p1.x + "," + p1.y + ")");
// Print out initial values of
// x and y
```
Points

// now set p1's coordinates
p1.x = 7;
p1.y = 2;
// now translate by (11, 6)
p1.x += 11;
p1.y += 6;
Array of Points

- We can set up an array of these points:

```java
Point[] pair = new Point[2];
pair[0] = p1;
pair[1] = p2; // assume we have p2
```

![Diagram showing array of points with coordinates (11, 6) and (5, 10)]
What an object is

- An object has state, which is defined by its fields.
- An object has behavior, which is defined by its methods.
- A class serves as a “blueprint” for creating new objects, which are “instances” of the class.
- We call creating a new object from a class instantiation.
Points

• Now we have really simple Point objects, but with no behavior – no methods.

• How can we get a Point that supports a translate method?

• We want to be able to do this:

  \texttt{p1.translate(1, 1);}

• This should change \texttt{p1} from (18, 8) to (19, 9).
Point.translate(…)  

- translate should take two int params – the x and y coordinates to shift the point by.

```java
public class Point {
    int x;
    int y;
    public void translate(int dx, int dy){
        // code goes here
    }
}
```
Point.translate(int dx, int dy)

- Note the lack of the keyword static here.
- We rarely use static in an object class.
- To start with, just avoid it completely when creating a class that defines an object.
- There should be no main method in an object class – we'll use a separate class to have the main method.
Implementing translate method

• We just need ordinary java code for the body of our translate method, but in an object class setting.

• Inside the object, method parameters and object fields are easy to access.

• The outside world is harder to access.

• Try to do everything in terms of parameters and fields.
Implementing translate method

- To translate x by dx: x += dx
- To translate y by dy: y += dy

... public void translate(int dx, int dy){
  x += dx; // use fields and parameters
  y += dy;
}
Translate method

- Note we didn't use “x” and “y” as parameter names to avoid ambiguity.
- Now if we have Points p1 → (1, 2) and p2 → (10, 20), we can do this:
  ```java
  p1.translate(1, 1); // change to (2, 3)
  p2.translate(1, 1); // p2 now (11, 21)
  ```
Fields and methods

- So, we created method code that worked on the object's own field values, thus changing its own internal state.

- Analogy from real world: when driving a car, our universe of control of the car is all inside the car. The horn we are “interested in” is the horn inside our car.

- When we write class methods, we use the variables storing a particular object's state.
Running this code

- Can download Point.java and PointMain.java from Point_2 directory of chapter 8 code.
- Java will find main in PointMain and execute it after compiling and running.
- It refers to code in Point.java to make Point objects and use methods.
Printing Points

• If we did System.out.println("p1 is " + p1); we would get something like:
  - "p1 is Point@1aed45f"

• How can we get something more user-friendly?

• Need to implement the toString() method.
Printing Points

• Unlike translate, toString has built-in significance in Java. It returns the String representation of an object's state.

• We used Scanner's toString earlier.

```java
public String toString() {
    return "(" + x + "," + y + ")";
}
```
Point Constructor

● We want to be able to do this:
  - Point p = new Point(2, 3);

● This requires that we write a constructor:

```java
public class Point {

    ...

    public Point (int x0, int y0) {
        x = x0;
        y = y0;
    }

}
```
Now we have a nice object

- We have a constructor, methods.
- Could easily add `getX()`, etc., but we don't yet have **encapsulation**
- Idea: imaging locking your car. How does this relate to programming?
  - Data that's “lying around” can be damaged, or misused by mistake.
  - When this happens, it's hard to track down the cause.
Encapsulation

• It's better to “seal up” our data in larger units, then allow access at the right time, in the correct way.

• For our point example, encapsulation would mean using the “private” keyword for x and y fields
  – then get rid of calls to p.x, p.y
  – replace with p.getX(), p.getY()