Nested classes

Introduction

To the left below are two classes, C and IC. Suppose that class IC is used only by class C, and there is no reason to use it in other classes. But since IC is public, other classes can see it and use it. Even if we make it protected, or give it no access modifier so it has access “package”, some other classes would still be able to see it and use it.

Further, there is no easy way for objects of class IC to reference fields and methods of class C, and for some applications this would be useful.

So, instead, we make IC a *nested* class, declaring it in class C as shown to the right below. The syntax rules of Java give class C access to all components of nested class IC, even if they are private. Also, IC is now not seen outside class C, since it is private. Declaring IC as a nested class of C creates a closer mutual relationship between them, making it easier to keep the program simple and efficient. And, IC is completely hidden from the outside.

Two classes C and IC

```
public class C {
    private int b;

    public void m(int x) {}
}

public class IC {
    private int d;

    public void p() {}
}
```

IC as an nested class in C

```
public class C {
    private int b;

    public void m(int x) {}

    private class IC {
        private int d;

        public void p() {}
    }
}
```

Thus, the nested class is a useful tool for structuring programs. We summarize its advantages.

1. It hides nested class IC from the rest of the program. Class C may need class IC to do its job, but there may be no reason for the rest of the program to know about it.
2. By the inside-out rule, methods in nested class C have access to fields and methods of outer class C. With C and IC on the same level, as in the box to the left above, methods in IC cannot easily access methods and fields of C. And if we find a way to give them that access, then other parts of the program may gain that access too; that may be unfortunate.
3. Even though nested class IC (in the box to the right) and its fields are private, the Java syntax rules for nested classes allow methods in outer class C to reference objects of class IC and their components. The access modifier private (and others) can be used to hinder other parts of the program from accessing class IC and its components, but not class C itself.

Here is some more terminology. There are (at least) two kinds of nested class:

1. If a nested class is static (yes, they can be static), it is called a *static nested class*. A static nested class can reference only static components of the outer class.
2. If a nested class is not static (like class IC to the right above), it is called an *inner class*.

In other notes, we give examples to illustrate how inner classes can be used.