The Java™ Tutorials

Trail: Learning the Java Language
Lesson: Classes and Objects
Section: Nested Classes

Inner Class Example

To see an inner class in use, first consider an array. In the following example, you create an array, fill it with integer values, and then output only values of even indices of the array in ascending order.

The DataStructure.java example that follows consists of:

- The DataStructure outer class, which includes a constructor to create an instance of DataStructure containing an array filled with consecutive integer values (0, 1, 2, 3, and so on) and a method that prints elements of the array that have an even index value.
- The EvenIterator inner class, which implements the DataStructureIterator interface, which extends the Iterator< Integer> interface. Iterators are used to step through a data structure and typically have methods to test for the last element, retrieve the current element, and move to the next element.
- A main method that instantiates a DataStructure object (ds), then invokes the printEven method to print elements of the array arrayOfInts that have an even index value.

```
public class DataStructure {

    // Create an array
    private final static int SIZE = 15;
    private int[] arrayOfInts = new int[SIZE];

    public DataStructure() {
        // fill the array with ascending integer values
        for (int i = 0; i < SIZE; i++) {
            arrayOfInts[i] = i;
        }
    }

    public void printEven() {

        // Print out values of even indices of the array
        DataStructureIterator iterator = this.new EvenIterator();
        while (iterator.hasNext()) {
            System.out.print(iterator.next() + " ");
        }
        System.out.println();
    }

    interface DataStructureIterator extends java.util.Iterator<Integer> { }

    // Inner class implements the DataStructureIterator interface,
    // which extends the Iterator<Integer> interface

    private class EvenIterator implements DataStructureIterator {

        // Start stepping through the array from the beginning
        private int nextIndex = 0;

        public boolean hasNext() {

            // Check if the current element is the last in the array
            return (nextIndex <= SIZE - 1);
        }

        public Integer next() {

            // Record a value of an even index of the array
            Integer retValue = Integer.valueOf(arrayOfInts[nextIndex]);

            // Get the next even element
            nextIndex += 2;
            return retValue;
        }
    }
```

```
        public static void main(String s[]) {

            // Fill the array with integer values and print out only
            // values of even indices
            DataStructure ds = new DataStructure();
            ds.printEven();
        }
    }
```

The output is:

```
    0 2 4 6 8 10 12 14
```

Note that the `EvenIterator` class refers directly to the `arrayOfInts` instance variable of the `DataStructure` object.

You can use inner classes to implement helper classes such as the one shown in the this example. To handle user interface events, you must know how to use inner classes, because the event-handling mechanism makes extensive use of them.

Local and Anonymous Classes 本地类、匿名类（在方法体内声明）

本地类：在方法体内声明一个内部类

There are two additional types of inner classes. You can declare an inner class within the body of a method. These classes are known as local classes. You can also declare an inner class within the body of a method without naming the class. These classes are known as anonymous classes.

匿名类：在方法体内声明一个内部类，且无需定义类名

Modifiers

You can use the same modifiers for inner classes that you use for other members of the outer class. For example, you can use the access specifiers `private`, `public`, and `protected` to restrict access to inner classes, just as you use them to restrict access do to other class members.

Previous page: Nested Classes
Next page: Local Classes