Special Topic 10.7

Enumeration Types Revisited

In Special Topic 5.3, we introduced the concept of an enumeration type: a type with a finite number of values. An example is

```
public enum FilingStatus { SINGLE, MARRIED }
```

In Java, enumeration types are classes with special properties. They have a finite number of instances, namely the objects declared inside the braces. For example, there are exactly two objects of the FilingStatus class: FilingStatus.SINGLE and FilingStatus.MARRIED. Since FilingStatus has no public constructor, it is impossible to construct additional objects.

Enumeration classes extend the Enum class, from which they inherit toString and clone methods. The toString method returns a string that equals the object's name. For example, FilingStatus.SINGLE.toString() returns "SINGLE". The clone method returns the given object *without making a copy.* After all, it should not be possible to generate new objects of an enumeration class.

The Enum class inherits the equals method from its superclass, Object. Thus, two enumeration constants are only considered equal when they are identical.

You can add your own methods and constructors to an enumeration class, for example

```
public enum CoinType
{
    private double value;
    PENNY(0.01), NICKEL(0.05), DIME(0.1), QUARTER(0.25);
    CoinType(double aValue) { value = aValue; }
    public double getValue() { return value; }
}
```

This CoinType class has exactly four instances: CoinType.PENNY, CoinType.NICKEL, CoinType.DIME, and CoinType.QUARTER. If you have one of these four CoinType objects, you can apply the getValue method to obtain the coin's value.

Note that there is a major philosophical difference between this CoinType class and the Coin class that we have discussed elsewhere in this chapter. A Coin object represents a particular coin. You can construct as many Coin objects as you like. Different Coin objects can be equal to another. We consider two Coin objects equal when their names and values match. However, CoinType describes a type of coins, not an individual coin. The four CoinType objects are distinct from each other.