Special Topic 5.3

Enumeration Types

In many programs, you use variables that can hold one of a finite number of values. For example, in the tax return class, the `status` instance variable holds one of the values `SINGLE` or `MARRIED`. We arbitrarily declared `SINGLE` as the number 1 and `MARRIED` as 2. If, due to some programming error, the `status` variable is set to another integer value (such as –1, 0, or 3), then the programming logic may produce invalid results.

In a simple program, this is not really a problem. But as programs grow over time, and more cases are added (such as the "married filing separately" and "head of household" categories), errors can slip in. Java version 5.0 introduces a remedy: **enumeration types**. An enumeration type has a finite set of values, for example

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

You can have any number of values, but you must include them all in the `enum` declaration.

You can declare variables of the enumeration type:

```
FilingStatus status = FilingStatus.SINGLE;
```

If you try to assign a value that isn't a `FilingStatus`, such as 2 or `"S"`, then the compiler reports an error.

Use the `==` operator to compare enumeration values, for example:

```
if (status == FilingStatus.SINGLE) . . .
```

It is common to nest an `enum` declaration inside a class, such as

```
public class TaxReturn
{
   public TaxReturn(double anIncome, FilingStatus aStatus) { . . . }
   . . .
   public enum FilingStatus { SINGLE, MARRIED }
   private FilingStatus status;
}
```

To access the enumeration outside the class in which it is declared, use the class name as a prefix:

```
TaxReturn return = new TaxReturn(income, TaxReturn.FilingStatus.SINGLE);
```

Syntax 5.3 Declaring an Enumeration Type

Syntax *accessSpecifier* `enum` *TypeName* `{` $value_1$, $value_2$, . . . `}`

Example

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

Type declaration

Variable declaration

This variable can have values FilingStatus.SINGLE, FilingStatus.MARRIED, or null.

An enumeration type variable can be `null`. For example, the `status` variable in the previous example can actually have three values: `SINGLE`, `MARRIED`, and `null`. This can be useful, for example to identify an uninitialized variable, or a potential pitfall.