Return to Module Overview Page (https://onlinelearning.berkeley.edu/courses/665040/pages/module-three)

3.6: Enumerations

Using constants for single values is fine, but when defining a series of constants, some problems come up.

Consider the following series of movie genre constants:

```
final int ACTION = 1;
final int COMEDY = 2;
final int DOCUMENTARY = 3;
final int SCI_FI = 4;
```

To use these constants, we would need to follow the same type as its assigned value. In our example above, ints are used. We would then need to declare/define an int variable:

```
int movieGenre = ACTION;
```

Using a series of constants works well if you do not have to add new constants or change the values. However, as the size and complexity of your application grows, this approach has its limitations.

One problem with this approach is that it is not type-safe. Our movieGenre variable can be set to any int value outside our defined range, and it would not be caught during compile time. The following would be valid, but is not something we want:

```
movieGenre = -5;
```

Another problem is that if we had several different series of constants (another for music genre, for example), we could easily mix them up. For example:

```
final int ROCK = 1;
final int CLASSICAL = 2;

movieGenre = ROCK;
```

With the release of Java 5, we can use enumerations. Enumerations are user-defined types for which there are limited specific values. Enumerations fix the problems I've mentioned and also add a few new features.

To define an enumeration, we use the enum keyword, followed by the enumeration type name and then a series of enumeration constants surrounded by curly braces. Rewriting our previous example to use enumerations, we get:

```
enum MovieGenre {
   ACTION, COMEDY, DOCUMENTARY, SCI_FI
}
```

Instead of having to use awkward ints, we have an actual class type in MovieGenre. Now assignments of the wrong type are caught during compile time:

```
MovieGenre movieGenre = MovieGenre.ACTION;
```

Enumerations can be conveniently used in **switch**/**case** statements:

```
switch(movieGenre) {
      case ACTION:
            System.out.println("The Terminator");
            break;
      case COMEDY:
            System.out.println("Twins");
            break;
      case DOCUMENTARY:
            System.out.println("Pumping Iron");
            break;
      case SCI_FI:
            System.out.println("Total Recall");
      default:
}
```

. . . as well as **for** loops values( ) method):

```
for (MovieGenre mg:MovieGenre.values()) {
      System.out.println("Type: " + mg);
}
```