Working with objects

Mar. 28, 2017

Modifiers

Modifiers

- Public/private are visibility modifiers
  - Used to indicate visibility of methods and attributes

Modifiers

- Java has a range of other modifiers
  - Control “ownership” of a method or attribute
  - Control when and how variable can be initialised
  - Control inheritance of methods (and whether they can be overridden by a sub-class)

LAB# 4-1

- When we use an array, if we set up the index out of the size of the array, we will encounter a run-time error.
- Create a class for a safe array that checks if the given index is within the scope of its index

```
public class SafeArray {
	private int a[];
	public int length;
	public SafeArray(int size) {
		a = new int[size];
		length = size;
	}
	public int get(int index) {
		??
	}
	public void put(int index, int value) {
		??
	}
}
```

Static

- `static` – indicates a *class variable* or *class method*.
- It's not owned by an individual object
  - This means we don't have to create an object to use it
  - `Arrays.sort` and `System.arrayCopy` are static methods

Static -- Example

```
public class MyClass
{
  public static void utilityMethod() { … }
  public void otherMethod() { … }
}

//using the above:
MyClass.utilityMethod();
MyClass objectOfMyClass = new MyClass();
objectOfMyClass.otherMethod();
objectOfMyClass.utilityMethod();

//this is illegal:
MyClass.otherMethod();
```

Final

- `final` – to make a variable that can have a single value
  - Can be assigned to *once and once only*
  - Useful to ensure a variable isn't changed once its assigned.

```
final int count;
count = 10;
//the following will cause an error
count = 20;
```

Defining Constants

- Unlike other languages, Java has no `const` keyword
- Must use a combination of modifiers to make a constant
  - `static` – to indicate its owned by the class
  - `final` – to make sure it can't be changed (and initialise it when its declared)
- Naming convention for constants is to use all capitals
- Example…

Constants – An Example

```
public class MyClass
{
  public static final int COUNT = 0;
  public static final boolean SWITCHED_ON = false;
}

//example usage:
if (MyClass.COUNT > 0) { … }

if (MyClass.SWITCHED_ON) {…}
```