Abstract Classes

Overview

Abstract classes are regular classes, but abstract. This means that they **may** contain abstract methods. These methods do not have implementations, just a signature.

```
abstract class Shape {
    Color color; // assume Color class exists
    Shape() {
        ...
    }
    abstract void draw(); // no implemented
}
public class Square extends Shape {
    Square() {
        ...
    }
    public void draw() {
        *some implementation*
    }
}
```

In this example the Square class extends the Shape class. The Square class provides implementation for the abstract method in the Shape class. If the class that extends an abstract class does not provide implementation for the abstract method, then that class must also be abstract. Too lazy to write the rest of the properties in this paragraph so refer to the **TL;DR** section below.

TL;DR

1. Abstract classes **MAY** contain abstract methods (doesn't need to)
2. Abstract methods in abstract classes should not have method bodies
3. Abstract classes may contain variables like a regular class and final methods
4. Abstract classes **can** be used as references for variables, but **cannot** be created as an instance
5. If a class inherits from an abstract class and doesn't provide a method body for abstract methods, then that class **must** also be abstract
6. Abstract classes may have constructors, which gets called when a child class is initialized (abstract constructor then child constructor)