

Java Style Guidelines

In programming, “style” refers to how we write, organize, and lay out code. “Good style” involves coding practices that make code easier to understand and maintain. (To “maintain” code is to enhance it or fix it after it’s basically finished; for a major software product maintenance will be roughly 90% of the software’s lifespan, so maintaining software is a very important consideration.)

Most companies or organizations that develop code that must be shared among programmers will adopt style guidelines that the programmers are expected to follow. In this class, there are just a few style guidelines. Part of your grade on each project will be based on following the style guidelines.

Use of White Space

The Java compiler doesn’t care about white space but the human eye does. Use white space to help keep related code together and unrelated code apart. For example, if you have the related properties **height** and **width** you might want to keep those on adjacent lines:

```
double height;  
double width;
```

However, if you had another String property called `ownersName` you might be a blank line between that and the numeric properties. See the example code at the end of this document for further examples.

Indentation is another important part of white space usage. Curly brackets (`{ }`) are use to mark the beginning and end of a block of code. These curly brackets should be at the same indentation, and everything inside them should be indented one level further. Again, see the code sample for an example of this.

Capitalization

Java is case-sensitive (RealBASIC is not). That means that **SmallPotatoes**, **smallpotatoes**, **smallPotatoes**, and **sMALLpOTATOES** are all different names to Java. Consistent use of capitalization not only make your code easier to read, it makes it less likely you will make a careless typographical error.

“Mixed-case” is very common in Java code. **SmallPotatoes** and **smallPotatoes** are examples of mixed-case with and without a leading capital letter, respectively.

Here are the capitalization guidelines for this course:

Constants should be all uppercase with words separated by an underscore (example: **SMALL_POTATO_MAXIMUM_SIZE**).

Class names should be mixed case with a leading capital (example: **PotatoFarm**).

Properties, variable names, and method names should be mixed case with a leading lowercase letter (e.g., **potatoCount**). (Note that for method names this is not a hard-and-fast guideline; if you write method names with leading capitals that's o.k. as long as you are consistent.)

Identifiers

An “identifier” is the name we give to a class, variable, property, or method (all the bold-faced words above are examples of identifiers).

Identifiers should be long enough to give a clear idea about what is being identified, but not so long as to be cumbersome. Obviously this is a bit subjective. Also, here are some specific guidelines for different types of identifiers.

Class names should be nouns and are almost always *singular*. (Example: **Potato** not **Potatoes**, although **BagOfPotatoes** would be fine since the class is really about the “bag” which is singular.)

Properties should be nouns/characteristics. (Example: **color**, **quality**, **farmWhereGrown**.)

Methods should be verbs or operations that you would perform on an object of that class (Example: **cook()**, **peel()**).

Loop variables are usually single letters like *i* or *j* (*i* is the traditional loop variable). However, sometimes longer names may be clearer.

It's o.k. to use single-letter variables for well-known math concepts (e.g., **x** and **y** for coordinates).

Comments

Commenting your code is an important part of good style. Every class, property, or method should have a comment explaining what it does or what it's for.

Local variables and arguments to methods may or may not need a comment. Usually method arguments for public methods will need comments to explain to someone who wants to use the method what they need to pass in to the method. However, no one expects a comment for a loop variable or an obvious variable like **usersLastName**.

Complicated code may require comments interspersed in the code to explain what is going on.

```

// class WhiteSpaceExample – bogus class of sample code that shows
// examples of all the main style guidelines

public class WhiteSpaceExample {
    // constants
    public static final int EXAMPLE_CODE_NUMBER = 1;

    // properties
    double height;           // height of this example in inches
    double weight;          // weight of this example in nanograms

    String ownersName;      // owner of this example

    //constructor
    // sillyArg – an example argument that can be any number
    public WhiteSpaceExample(int sillyArg) {
        height = weight = (double) sillyArg;
        ownersName = "Doormat";
    }

    // other methods

    // doWhitespaceStuff – does random operations for example purposes only
    public doWhitespaceStuff() {
        // amount of random stuff to do
        int randomVariable = (int) (Math.random() * 10);

        // this loop wishes it was more complicated than it is; if itg
        // were that complicated you might put a comment here
        for (int i = 0; i < randomVariable; i++) {
            if (weight > 5.0) {
                height = height / 2.0;
            } else if (weight < 0.01) {
                ownersName = ownersName + " the small potato dude";
            }
        }
    }
}

```