Package and Import Statements

The first non-comment line of most Java source files is a package statement.

After that, import statements can follow.

For example:

package java.awt;

import java.awt.peer.CanvasPeer;

Class and Interface Declarations

Class (static) variables: First the public class variables, then the protected, and then the private.

Instance variables First public, then protected, and then private

Class body organization

The body of a class declaration should be organized in the following order:

1. Static variable field declarations
2. Instance variable field declarations
3. Static initializer
4. Static member inner class declarations
5. Static method declarations
6. Instance initializer
7. Instance constructor declarations
8. Instance member inner class declarations
9. Instance method declarations These three elements, fields, constructors, and methods, are collectively referred to as “members”.

Within each numbered group above, sort in lexical order.

Class and Interface Declarations

When coding Java classes and interfaces, the following formatting rules should be followed:

• No space between a method name and the parenthesis "(" starting its parameter list

• Open brace "{" appears at the end of the same line as the declaration statement

• Closing brace "}" starts a line by itself indented to match its corresponding opening statement, except when it is a null statement the "}" should appear immediately after the "{"

```
class Sample extends Object {
    int ivar1;
    int ivar2;

    Sample(int i, int j) {
        ivar1 = i;
        ivar2 = j;
    }

    int emptyMethod() {}

    ...
}
```

Declarations

One declaration per line is recommended since it encourages commenting.

int level; // indentation level

int size; // size of table

is preferred over

int level, size;

In absolutely no case should variables and functions be declared on the same line. Example:

long dbaddr, getDbaddr(); // WRONG!

Do not put different types on the same line.

Example: int foo, fooarray[]; //WRONG!

Placement

Put declarations only at the beginning of blocks.

Don't wait to declare variables until their first use; it can confuse the unwary programmer and hamper code portability within the scope.

Initialization

Try to initialize local variables where they're declared.

The only reason not to initialize a variable where it's declared is if the initial value depends on some computation occurring first.

Naming Conventions

Identifier Type	Rules for Naming	Examples
Classes	Class names should be nouns, in mixed case with the first letter of each internal word capitalized. Try to keep your class names simple and descriptive. Use whole words—avoid acronyms and abbreviations	class Raster; class ImageSprite;
Interfaces	Interface names should be capitalized like class names	interface RasterDelegate; interface Storing;

Methods	Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.	run();
		runFast();
		getBackground();

Variables	Except for variables, all instance, class, and class constants are in mixed case with a lowercase first letter. Internal words start with capital letters Variable names should be short yet meaningful. The choice of a variable name should be mnemonic— that is, designed to indicate to the casual observer the intent of its use.	int i; char name; float myWidth;
Constants	The names of variables declared class constants and of ANSI	int MIN_WIDTH = 4; int MAX_WIDTH = 999;

	constants should be all uppercase with words separated by underscores ("_").	int GET_THE_CPU = 1;
Packages	The prefix of a unique package name is always written in all-lowercase ASCII letters and should be one of the top-level domain names, currently com, edu, gov, mil, net, org, Subsequent components of the package name vary according to an organization's own internal naming conventions. Such conventions might specify that certain directory name components be division, department, project, machine, or login names.	com.sun.eng com.apple.quicktime.v2 edu.cmu.cs.bovik.cheese

Code Styling Convention

Line Length

Avoid lines longer than 80 characters, since they're not handled well by many terminals and tools.

Wrapping Lines

When an expression will not fit on a single line, break it according to these general principles:

- Break after a comma.
- Break before an operator.
- Align the new line with the beginning of the expression at the same level on the previous line.

Here are some examples of breaking method calls:

```
function(longExpression1, longExpression2, longExpression3,
         longExpression4, longExpression5);

var = function1(longExpression1,
                function2(longExpression2,
                          longExpression3));
```

longName1= longName2 * (longName3 + longName4 – longName5)
+ 4 * longName6; //PREFER

longName1= longName2 * (longName3 + longName4
–longName5) + 4 * longName6; //AVOID

References:

1. http://www.oracle.com/technetwork/java/codeconventions-150003.pdf