Code Conventions for the Java Programming Language

A short excerpt from
http://www.oracle.com/technetwork/java/codeconv-138413.html

Code Conventions: Why?

- 80% of the lifetime cost of a piece of software goes to maintenance
- Hardly any software is maintained for its whole life by the original author
- Code conventions improve the readability of the software, allowing engineers to understand new code more quickly and thoroughly

`.java` File Organization

1. Copyright/licensing header
2. Package/import statements
3. Class/interface comment
4. Class/interface declaration
5. Static variables
6. Instance variables
7. Constructors
8. Methods

Declarations (1/2)

- Prefer one declaration per line:

```
int currentlength;
int maxLenght;
```

is preferred over:

```
int currentlength, maxLenght;
```

- Declarations must be put at the beginning of a block (that is, just after a brace `{`)

Declarations (2/2)

- Avoid hiding previously defined variables:

```
int maxValue = 0;
// some code
if(/* test */) {
      int maxValue = 0; // forbidden!
      // some code
}
```

- Try to initialize variables when declared

Statements

- One line == one statement:

```
argv++;
argc--;
```

is preferred over:

```
argv++; argc--;
```

The Camel-Case Practice (1/2)

Flickr/Camel at Gaochang/tsc_traveller

The Camel-Case Practice (2/2)

- Element forming a camel-case word are joined without any space
- The first letter of each joined element starts with a capitalized letter (other letters are generally in lowercase)
- The first letter of a camel-case word can be uppercase or lowercase
- Eg.: UpperCamelCase, lowerCamelCase

Naming Conventions (1/3)

- A class name
  - Is a noun
  - Starts with a capitalized letter
  - Uses the camel-case practice
  - Eg.: `Resource`, `String`
- An interface name
  - Is a noun or an adverb generally ending with able
  - Starts with a capitalized letter
  - Uses the camel-case practice
  - Eg.: `Comparable`, `Observable`

Naming Conventions (2/3)

- A method name
  - Is a verb
  - Starts with a lowercase letter
  - Uses the camel-case practice
  - Eg.: `save`, `computeBalance`
- Getters and setters
  - A getter starts with `get`…
  - A setter starts with `set`…
  - … followed by the upper camel-case name of the attribute they provide access to
  - Eg.: `getMaxLength`, `setCurrentBalance`

Naming Conventions (3/3)

- Variables and parameters
  - Names should be short yet meaningful
  - Start with a lowercase letter
  - Use the camel-case practice
  - Eg.: `maxLength`, `currentBalance`
- Constants
  - All uppercase
  - Words must be separated using underscores
  - Eg.: `MAX_VALUE`, `LOGGER`, `CLASS_NAME`

Programming Practices (1/2)

- Access static fields using a class and not an object:

```
Integer.MAX_INTEGER;        // ok
anIntInstance.MAX_INTEGER; // ko
```

- Don’t assign a value to several variables in a single statement:

```
i = 0;
j = 0;
```

is preferred over:

```
i = j = 0;
```

Programming Practices (2/2)

- Keep source code lines short (a commonly agreed value is 80 characters)
- Keep documentation lines even shorter
- Keep file length short (commonly agreed value: 2000 lines)

Finally

If you need to maintain a piece of code which uses other conventions, you MUST follow these other conventions.