JAVA PROGRAMMING STYLE QUICK REFERENCE GUIDE

1. Indentation

- $\circ\,$ The unit of indentation is 4 spaces. Tabs expand to 8 spaces.
- Always keep line lengths < 80 chars.
- Breaking lines:
 - After a comma or before an operator
 - Try making lines close to 80 chars before breaking
 - The new line must be aligned at the beginning of the previous line expression.
- If the above lead to messy code, just indent to 8 spaces.
 //DON'T USE THIS INDENTATION

```
if ((condition1 && condition2)
    || (condition3 && condition4)
    ||!(condition5 && condition6)) { //BAD WRAPS
    doSomethingAboutIt(); //MAKE THIS LINE EASY TO MISS
}
//USE THIS INDENTATION INSTEAD
```

```
if ((condition1 && condition2)
```

```
|| (condition3 && condition4)
||!(condition5 && condition6)) {
doSomethingAboutIt();
```

3. Declarations

. . .

- One declaration per line, align multiple declared variables right using spaces.
- Put declarations at the outermost code block possible, except from declarations in the condition part of for blocks.
- Class and interface declaration rules:
 - 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 "{".
 - Methods are separated by a blank line.

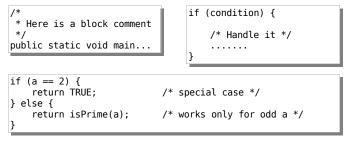
5. Statements

```
• IF - ELSE statements
if (condition) { //BRACE STARTS AT FIRST LINE
    statements;
}
if (condition) {
```

```
statements;
} else { //ELSE IS AT THE SAME LINE
statements; //AS THE CLOSING IF BRACE
```

2. Comments

- Implementation comments
- Block comments: (/* ... */) Provide descriptions of files, methods, data structures and algorithms. Inside a function or method should be indented to the same level as the code they describe. A block comment should be preceded by a blank line to set it apart from the rest of the code.
- Single line comments: Short comments that appear on a single line indented to the level of the code that follows. A single-line comment should be preceded by a blank line.
- Trailing comments: Short comments that appear on the same line as the code they describe. If more than one short comment appears in a chunk of code, they should all be indented to the same tab setting.
- End-of-line comments: (//...) Shouldn't be used on consecutive multiple lines for text comments; however, it can be used in consecutive multiple lines for commenting out sections of code.
- Documentation comments (/** ... */): Describe Java classes, interfaces, constructors, methods, and fields. This comment should appear just before the described element declaration.



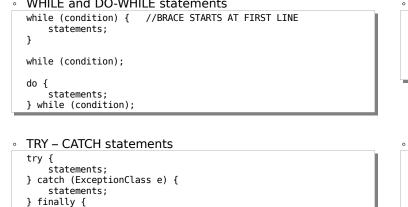
4. Whitespace

Blank lines

- 1 blank line: Between methods, between the local variables in a method and its first statement, before code blocks or single line comments and between logical sections in a method.
- 2 blank lines: Between class and interface definitions and between sections of a source file.
- Blank spaces should be used:
- Between keywords and following parenthesises e.g.
 while (true) {...}
- After commas in argument lists e.g. aMethod(arg1, arg2)
- Between binary operators and their arguments e.g. a += c + d; except from the '.' operator.
- Between expressions in a for statement.
- After the type in a cast expression, e.g. a = (Vector) b;

```
    SWITCH statements
```

WHILE and DO-WHILE statements



6. Naming Conventions

statements;

- **Packages:** Prefix should be a Fully Qualified Domain Suffix and must always written in lowercase ASCII letters (e.g. . com, .gr). Suffix can be anything conforming to external naming conventions.
- Classes & Interfaces: Names can be a series of (preferably) whole-world nouns with first letter of each word capitalised. Acronyms should be avoided, unless widely used (e.g. HTML).
- · Methods: Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.
- 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 not start with underscore _ or dollar sign \$ characters, even though both are allowed. One-character variable names should be avoided except for temporary "throwaway" variables.
- Constants: The names of variables declared class constants should be all uppercase with words separated by underscores ("_").

FOR statements

```
for (initialization; condition; update) {
    statements:
}
```

for (initialization; condition; update);

RETURN statements

return;	//ONLY USE PARENTHESIS //CLARIFY THE RETURNED	
<pre>return myDisk.size();</pre>		
return (size ? size :	defaultSize)	

7. Programming Practices

- Making instance variables public should be avoided, except when using the class as a data structure without behavior.
- Avoid using an object to access a class (static) variable or 0 method. Use a class name instead.
- Numerical constants (literals) should not be coded directly, except for -1, 0, and 1, which can appear in a for loop as counter values.
- Parentheses should be used in expressions involving mixed operators to avoid operator precedence problems.
- · If an expression containing a binary operator appears before the ? in the ternary ?: operator, it should be parenthesized (e.g. $(x \ge 0)$? x : -x;)
- 0 Use XXX in a comment to flag something that is bogus but works. Use FIXME to flag something that is bogus and broken.
- Variable assignments: 0
 - Avoid assigning several variables to the same value in a single statement.
 - Do not use the assignment operator in a place where it can be easily confused with the equality operator (e.g. if (c++ = d++) { ... } //AVOID)
 - Do not use embedded assignments in an attempt to improve run-time performance.

e.g.d = (a = b + c) + r; // AVOID!

8. References

- 1. Sun Java Microsystems. Code conventions for the Programming Revision 20/4/1999. Language. http://java.sun.com/docs/codeconv/html/CodeConvTOC.doc.html
- Oct 2. Geotechnical Software Services. programming guidelines. Ver 4.1, 2004. Java style http://geosoft.no/development/javastyle.html
- 3. Allan Vermeulen, Scott W. Ambler, Greg Bumgardner and Eldon Metz. The elements of Java style. Cambridge University Press. January 2000. http://www.amazon.com/exec/obidos/tg/detail/-/0521777682/102-9436732-1009700