On JAVA & Android code conventions

October 3, 2012

1 Exceptions

Don't ignore exceptions i.e. always handle the exception, or throw a new one according to the level of abstraction, or handle it gracefully, or throw a new RuntimeException in case there is nothing to do. It is "OK" to ignore it if there is a good reason to do it, but you should comment the reason in the code.

2 Generic Exception

Don't catch a generic exception e.g. "Exception e". Alternatives to do this are:

- Catch each exception with a separate catch block after a single try;
- Refactor the code in order to have muliple try blocks;
- Rethrow the exception and let it be handle in the next level.

3 Finalizers

Don't use finalizers i.e. let the garbage collector do its job.

4 Imports

Full imports. Don't use asterisk (This is catch by checkstyle). Order import statements

- First, android imports;

- Then, third parties libraries (com, junit, net, org);
- Java and javax.

More options are available in checkstyle for ordering things.

5 Comments

Use Javadoc Standard to comment. Like:

```
/**
 * Returns the nothingness
 */
```

or

```
/**
 * Constructs a Bosson Higs and reproduce the Big Bang all over again
 */
```

Use verbs in 3rd person. There is no need for comments in getters and setters that are trivial. Comment every function and do it thoroughly (not sketchy). Comment also errors of the codes where you might not be clear, and lastly use comments sparsely (code should be self commenting).

6 Methods

Write short methods. No more than 40 lines.

7 Variable scope

Keep the scope of variables really small. This means that you should declare a variable in the innermost block of the code that that variable is being used. The only exception is when using try blocks and the variable is used after the catch. Also, it is good to initialize the variable to a proper value. If this is not possible, it means that it is good to postpone the declaration.

8 Indentation

Use spaces NOT tabs. This can be configured in Eclipse by going to Window → Preferences → Editors → Text Editors → Insert spaces for tabs.

- 4 space indents for blocks;
- 8 space indents for line wraps like:

```
Instrument i =
        oneLongExpression(that, wouldNotFit, on, one, line);
```

9 Naming conventions

- Non-public, non-static field names start with m;
- Static field names start with s;
- Other fields start with a lower case letter;
- Public static final fields (constants) are ALL_CAPS_WITH_UNDERSCORES.

```
public class MyClass {
    public static final int SOME_CONSTANT = 42;
    public int publicField;
    private static MyClass sSingleton;
    int mPackagePrivate;
    private int mPrivate;
    protected int mProtected;
}
```

10 Standard Brace Style

```
public void foo() {
    if (...) {
        doSomething();
    }
}
```

11 Line length

Limit line length to 80. Exception: URLs

12 Annotations

Use standard Java Annotations: @Deprecated, @Override, @SuppressWarnings

13 Acronyms

Use acronyms as words e.g. write "XmlHttpRequest" not "XMLHTTPRequest"

14 TODO

Use TODO comments. You can quickly jump to TODOs by opening the TODO tabs in Window → Tasks

15 Logging

When logging use the five levels of logging: ERROR, WARNING, INFORMATIVE, DEBUG, VERBOSE.

16 Checkstyle

- To install checkstyle: Help → Install New Software;
- Add... Location: `http://eclipse-cs.sf.net/update/`;
- Install everything and restart Eclipse;
- Then Windows → Preferences → Checkstyle → New...
- Type: External Configuration File;
- Name: Android Config;
- Download the file at `https://gist.github.com/948425`;
- Remove the line $< modulename = "DoubleCheckedLocking" >< /module >$;
- Browse the downloaded file;
- Now on the project, Right click → Properties → Checkstyle;

- Check "Checkstyle active for this project";
- Choose "Android Config - (Global)";
- Press OK. Restart just to be sure.

17 Header classes

Define a general header for our project and for our classes. A class should contain a description of what it does and the author of the class.

18 Naming

- Define naming for packages;
- Use useful and understandable names for variables and function!

19 Methods

- Order private methods first and public methods last;
- Only configure data that is passed as an argument;
- Javadoc: add at least @return and @param annotations;
- Use space between if (), for () and their braces;
- Use 8 spaces indentation in methods declaration if there are many parameters.

20 Fields

Set to private or static. Comment all the class fields!