JagTrack
Programming Guidelines

Version 1.0

Revision History

Date	Version	Description	Author
01/Apr/12	0.1	Creation	Hayden Chudy

Table of Contents

Programming Guidelines

1. Introduction

1.1 Purpose

The purpose of the programming guidelines is to provide a reference guide for all programmers on the JagTrack project. This will result in a unified style of coding that will be easier to review and write in general.

1.2 Scope

The programming guidelines for the JagTrack project cover both the Android code in Java, the server code in PHP or Java, and the SQL tables and queries.

1.3 Definitions, Acronyms, and Abbreviations

- Like Statements – multiple lines of code that, when run together, perform a more abstract goal. Example: moving a bus (changing the bus' x and y coordinates, putting them in the database, etc.)

1.4 References

1.5 Overview

2. Code Organization and Style

Code should be spacious and blocks and functions should be formatted like:

```
function(int arg1, char arg2) {
    code;
    while(1) {
        int a = 4;
    }
}
```

That is:

- There should be no spaces between a function name and the opening parenthesis.
- There should be no spaces between the opening parenthesis and the first argument (or its type).
- Arguments should be separated by a comma followed by a space.
- No space should follow the final argument and the closing parenthesis.
- After the closing parenthesis, there should be a space, and the opening bracket.
- The closing bracket should be on a line of its own.
- Tabs should be used instead of spaces.

Code should be spacious and empty line breaks are encouraged. Lines of code that extend beyond 80 characters are discouraged and should, preferably, be split onto multiple lines if possible. Blocks of code should be indented one level in.

3. Comments

[A description of the use of comments.]

Comments explaining the purpose of code or what a variable represents conceptually are required. Like statements can be grouped together, with no empty line breaks between them, and their purpose can be stated with one comment

Comments should be on the line above the statement they describe, not beside it. Like so:

```
//comment for statement1
statement1;
```

Font holder

4. Naming

Types and variables can be named based on the concepts they represent.

Subprograms should be prefixed with a jt, like so:

```
jt_this_function()
```

in order to make the functions easily recognizable from other API or library functions.

5. Declaration

Any declarations should be placed at the top of their respective parent blocks, i.e. loop variables should go right after the beginning of the loop and function variables right after its declaration. Variables can be given values in their declarations. Example

```
function example() {
    int cntr = 6;
    while(cntr--) {
        float flt;
        flt *= 1.2;
    }
}
```

Font holder

6. Expressions and Statements

To be determined.

7. Memory Management

Since the application is written in Java, memory management is handled by the Garbage Collector. PHP may be used for the server code, which will still handle memory management via a Garbage Collector.

The SQL tables should be as small and compact as possible.

8. Error Handling and Exceptions

Any risky code should be put in a try block and the catch block should at least print out an error code, and ideally should fully handle the risk the code creates.

Error codes should be documented in the comments at the top of each file that describe the logical flow required to get to the error situation, what caused this, and if any outside, manual action can be taken to fix it. All error codes should be documented in final documents.

9. Portability

None currently required.

10. Reuse

11. Compilation Issues

12. Annex: Summary of Guidelines

- There should be no spaces between a function name and the opening parenthesis.
- There should be no spaces between the opening parenthesis and the first argument (or its type).
- Arguments should be separated by a comma followed by a space.
- No space should follow the final argument and the closing parenthesis.
- After the closing parenthesis, there should be a space, and the opening bracket.
- The closing bracket should be on a line of its own.
- Try blocks should be used and the catch block should at least print out an error code, which will be documented in the comments at the top of each file.
- Any declarations should be placed at the top of their respective parent blocks.
- Types and variables can be named based on the concepts they represent.
- Subprograms should be prefixed with a jt.
- Comments explaining the purpose of code or what a variable represents conceptually are required.
- Comments should be on the line above the statement they describe, not beside it.
- Like statements can be grouped together, with no empty line breaks between them, and their purpose can be stated with one comment.
- Code should be spacious and empty line breaks are encouraged.
- Lines of code that extend beyond 80 characters are discouraged and should, preferably, be split onto multiple lines if possible.
- Blocks of code should be indented one level in.
- Tabs should be used instead of spaces.