Steven Zuelke

Austin Hall

C. William Bafus

TEAM CODING STANDARDS

INTRODUCTION

This document serves as the living definition for the team's coding standards for all source code. It will attempt to cover conventions pertaining to both the C# and Java programming languages.

Like other programming style guides, the issues covered span not only aesthetic issues of formatting, but other types of conventions or coding standards as well. However, this document shall avoid giving *advice* on how to code that isn't clearly enforceable.

CAPITALIZATION

CAMEL CASE

The first letter of each word is capital *except* for the first. This applies to:

- Parameters
- Variables (Non-field)

PASCAL CASE

The first letter of each word is capitalized *including* the first word. This applies to:

- Namespaces (C#)
- Packages (Java)
- Classes
- Interfaces
- Methods
- Properties (C#)
- Events
- Fields
- Enums

SPECIAL CASES

Constants are represented in full uppercase letters. Acronyms in Pascal Case that have three or more letters are all capitalized.

NAMING CONVENTIONS

GENERAL

Names should represent the purpose of the identifier and convey simple and understandable meaning to the programmer.

SPECIAL CASES

These conventions apply to specific circumstances.

Interfaces are prefixed with 'I'. (C#)
Mutators and accessors are prefixed get/set respectively. (C#)
Loop iterators are given letter identifiers and continue alphabetically with each additional inner loop.

COMMENTING CONVENTIONS

Any line break may be preceded by arbitrary whitespace followed by an implementation comment. Such a comment renders the line non-blank.

A blank line (non-commented) separates any file header, constructor, or method comment from the start of the code.

FILE HEADERS

At the top of each source code file (.java, or .cs), the file's name, author, revision number, revision author, and a description of the file is commented.

CONSTRUCTORS

A constructor contains a comment that provides the purpose of the constructor and any necessary or non-self-apparent details.

METHODS

A Method contains a comment the details the inputs, purpose, and return value as the programmer deems necessary for clarity.

SPACING CONVENTIONS

FILE

A blank line (non-commented) separates any file header, constructor, or method comment from the start of the code.

METHODS

Each constructor or general method should be separated from each other (and their header comment) by a blank line.

VARIABLES

Method variables are declared at the top of a method with a blank line between them and the code.

A single blank line always appears between consecutive members or initializers of a class: fields, constructors, methods, nested classes, static initializers, and instance initializers.

Multiple consecutive blank lines are permitted, but not encouraged.

SECURITY

Class variables and fields are set as private with getters/setters as appropriate.

Methods that are accessed only from within the class in which they are written are made private.

BRACES

BRACES ARE USED WHERE OPTIONAL

Braces are used with **if**, **else**, **for**, **do**, and **while** statements, even when the body is empty or contains only a single statement.

NONEMPTY BLOCKS: KERNIGHAN AND RITCHIE STYLE

Braces follow the Kernighan and Ritchie style ("Egyptian Brackets") for nonempty blocks.

No line break before the opening brace
Line break after the opening brace
Line break before the closing brace

BLOCK INDENTION

Each time a new block or block-like construct is opened, the indent level increases by 4 spaces.

Each statement is followed by a line break such that there is only one statement per line.