Style Guide

Contents

[Header Files 2](#_Toc119311369)

[Classes 2](#_Toc119311370)

[Classes vs. Structs 2](#_Toc119311371)

[Functions 2](#_Toc119311372)

[If, For, While 3](#_Toc119311373)

[Spacing 3](#_Toc119311374)

[Naming 3](#_Toc119311375)

[Comments 3](#_Toc119311376)

[Single Line Comments 3](#_Toc119311377)

[Multi-line Comments 4](#_Toc119311378)

[Commented Code 4](#_Toc119311379)

[Single Line 4](#_Toc119311380)

[Multi Line 4](#_Toc119311381)

[Miscellaneous 4](#_Toc119311382)

[Spaces vs. Tabs 4](#_Toc119311383)

[Includes 4](#_Toc119311384)

# Header Files

All header files should have an accompanying code file with the same name. They must have a header guard at the top of the header file. All includes must be in the header file just after the header guard.

|  |
| --- |
| #ifndef filename\_hpp  #define filename\_hpp  ...  #endif |

# Classes

No class should implicitly define members as public or private, all classes need to define every member as public, private, or protected. Function definitions should have both the type and parameter name in the declaration. Inline functions must only be a single line, multiple lines must be moved outside the class declaration.

|  |
| --- |
| class SomeClass {  …  } |

|  |
| --- |
| ReturnType callSomething(Type param1, Type param2); |

|  |
| --- |
| void callSomething() { doSomething(); doSomethingElse(); } |

## Classes vs. Structs

Use a struct only for passive objects that carry only data, everything else is a class.

# Functions

Functions and Function Declarations must have a space between the return type and class name/function name, with no space between the function name and parameters. The bracket will start on the same line.

|  |
| --- |
| ReturnType ClassName::FunctionName(Type param1, Type param2) {  DoSomething();  ...  } |

|  |
| --- |
| ReturnType ClassName::FunctionName(Type param1, Type param2); |

# If, For, While

If, For, and While’s will have a space between the last character and the condition. The bracket will start on the same line.

|  |
| --- |
| while (condition) {  …  } |

# Spacing

Vertical spacing should be limited, there should be no blank line between the last line of a function and the line for the closing bracket. However, inside a function lines should be grouped together in a consistent manner within the function. Functions should be grouped together by classes with two blank lines between class function groups.

|  |
| --- |
| int main() {  …  doLastThing();  } |

|  |
| --- |
| ClassA::Function1() {…}  ClassA::Function2() {…}  ClassB::Function1() {…} |

# Naming

All names should follow the CamelCase convention. With the exception of very similar objects.

|  |
| --- |
| doSomethingNow  callSomething\_static  callSomething\_dynamic |

# Comments

## Single Line Comments

Single line comments must have a space between the comment character and the comment text. If commenting after code, place at least two spaces between the last character and the comment character.

|  |
| --- |
| // Comment Something  DoSomething();  DoSomethingElse(); // Comment Something Else |

## Multi-line Comments

Comments spanning over a single line must use a multi-line comment. Their must be a space between the comment character and first character, and between the last character and the comment character. A multi-line comment must not start on the same line after code.

|  |
| --- |
| /\* Hello There,  this is some  nice code here \*/ |

## Commented Code

### Single Line

Single line commented out code will use the single line comment with no space at the beginning.

|  |
| --- |
| //SomeCode(); |

### Multi Line

Commented out code spanning multiple lines will use the multi-line comment with no space at the beginning or end.

|  |
| --- |
| /\*if (testSomething) {  doSomething();  …  }\*/ |

# Miscellaneous

## Spaces vs. Tabs

Use only tabs, and indent with one tab per indent. The number of spaces equal to a tab can change between developer preference.

## Includes

Includes should be laid out with standard C++ files first, added libraries grouped together second, and user files last.

|  |
| --- |
| #include <Standard C++ File>  #include "Added Library"  #include “Added Library”  #include “Added Library 2”  #include “User File” |