**Coders For Christ CodeStyle**

**General Rules:**

Curly braces will be on their own lines

camelCase for variable names

Constants will be in UPPERCASE\_SNAKE\_CASE

Methods names are also in camelCase

Whitespace: Newlines will be added between blocks of code that are dedicated for a certain

Logic (unless there are curly braces separating complicated logics). There is also one newline between method definitions

Indents will be made with tabs instead of spaces. Every line contained within a set of {} will be indented relative to the top line of the curly brace.

Methods that return a boolean will start with is such as isWord().

If logic is determined by something other than a bool, such as an int (if 1 do x, if 2 do y, if 3 do z) then we will use an enum instead.

Member Variables should be *protected* or *private*. Accessors should be used to get or set their values.

class initializer list for constructor initialization

Avoid one or two letter names (unless it’s common use). Prefer long, descriptive names. Be precise.

Use size\_t to loop through elements in arrays.

**Comments:**

C++ header comments for method headers that use @brief, @return, and @param. Used in .h files (methods in .cpp should not have repeated header comments).

/// @brief

/// @param

/// @return

returnType methodName (inputType);

// In-line comments for complicated logic in sections of code. (use sparingly)

**Files:**

There will be a header file where all the comments and documentation go. Implementation will be done in the .cpp file. The .h and .cpp files will have the same name with different extensions. The name should be the class name. The names will use camelCase (ex: pixelEditorModel.cpp and pixelEditorModel.h with the caveat that mainwindow remains the same as it is default).

Resource folder where images for icons go