C++代码审查表

版本：0.1

1. 代码风格

|  |  |  |
| --- | --- | --- |
| 代码排版 |  |  |
| 审查项目 | 结论 | 提示 |
| 已经使用clang-format对代码进行了格式化。 |  | 在Visual Studio Code内用Shift+Alt+F即可格式化代码。 |
| 已经使用Code Spell Checker进行了拼写检查。 |  | 在Visual Studio Code内可以安装Code Spell Checker插件。 |
|  |  |  |
| C++版本 |  |  |
| 审查项目 | 结论 | 提示 |
| 没有使用了C++2X特性。 |  |  |
|  |  |  |
|  |  |  |
| 文件结构 |  |  |
| 审查项目 | 结论 | 提示 |
| 头文件和定义文件的名称合理。 |  |  |
| 版权和版本声明完整。 |  | 如果使用了tools/develop.py生成文件，那么版权和版本声明是自动生成的。 |
| 头文件使用了 ifndef/define/endif 预处理块。 |  | 如果使用了tools/develop.py生成文件，那么预处理块是自动生成的。 |
| 头文件中只存放“声明”而不存放“实现”。 |  |  |
| 头文件中没有定义新的namespace。 |  |  |
| 头文件中没有在namespace kiseki外使用using namespace。 |  |  |
|  |  |  |
| 命名 |  |  |
| 审查项目 | 结论 | 提示 |
| 没有错误或者不一致的缩写。 |  | 如number没有写成num或者写成no。 |
| 命名没有冗余。 |  | 如Rectangle类中旋转函数只要命名为Rotate即可，而不是RotateRectangle。 |
| 模块命名是短的小写单词。 |  |  |
| 类/结构/typedef/枚举的命名是首字母大写的有意义的形容词和动词。 |  |  |
| 函数命名是否是首字母大写的有意义的动词和名词。 |  |  |
| 变量命名是用小写单词加下划线“\_”。 |  |  |
| 常量命名是首字母大写的单词连接而成（不用下划线“\_”）。 |  |  |
| 命名的例外情景是绝对必要的。 |  | 例如为了调用Win32 API可以使用类似wParam之类的命名。 |
| 无意义循环变量、纯数学或物理意义的变量合理。 |  | 类似i，j，或者r1，r2等。 |
| 对象的数组、vector、set、map等的命名使用了复数。 |  | 如vector<double> coords。 |
| 对于传递给类成员变量的函数参数，用tXX命名。 |  |  |
|  |  |  |
| 类/结构的声明 |  |  |
| 审查项目 | 结论 | 提示 |
| 类/结构的声明是先public后private。 |  |  |
| 类/结构的声明是先自定义类型（如类、结构、typedef和枚举），后成员函数，再成员变量。 |  |  |
| 变量的声明是先静态再非静态。 |  |  |
| 变量的声明是先常量再变量。 |  |  |
|  |  |  |
| 流程控制 |  |  |
| 审查项目 | 结论 | 提示 |
| 只是用前缀“++x”和“--y”。 |  |  |
| if和while语句的逻辑合理。 |  | 指针必须与nullptr比较；字符必须与’\0’比较；任何变量不可与true，false直接比较。 |
| switch含有default处理语句。 |  |  |
| switch的所有case都含有break（除非故意）。 |  |  |
| goto的使用必要和合理。 |  |  |
|  |  |  |
| const的使用 |  |  |
| 审查项目 | 结论 | 提示 |
| 常量加了const。 |  |  |
| 不会改变的变量加了const。 |  |  |
| 不会被修改的函数的指针或引用参数加了const。 |  |  |
| 不会修改成员的成员函数末尾加了const。 |  |  |
| 只读遍历STL容器时使用const\_iterator。 |  |  |
|  |  |  |
| 函数设计 |  |  |
| 审查项目 | 结论 | 提示 |
| 函数的参数名、类型、顺序合理。 |  | 先输入参数，后输出参数。 |
| 无参数函数用了void。 |  |  |
| 没有用默认参数。 |  |  |
| 该用const的位置用了const。 |  | 和前面const部分类似。 |
| 对类和结构参数不使用了传值传递。 |  |  |
| 对类和结构参数的指针传递是不可以用引用传递代替。 |  |  |
| 参数太多时，使用类或者结构简化参数。 |  |  |
| 参数保持了最大的兼容性，特别是底层函数。 |  |  |
| 没有使用静态局部变量。 |  |  |
| 没有使用内联函数。 |  |  |
| 函数的功能不冗余，不能分解为多个函数。 |  |  |
|  |  |  |
| 数字 |  |  |
| 审查项目 | 结论 | 提示 |
| 没有使用unsigned类型，除非为了兼容性。 |  |  |
| 所有的浮点数要么使用小数点“.” ，要么使用E。 |  | 例如，double x = 2必须写成double x = 2. |
| 精确浮点数最多25位小数。 |  |  |
|  |  |  |
| 禁用特性 |  |  |
| 审查项目 | 结论 | 提示 |
| 没有使用异常。 |  |  |
| 没有使用友元friend。 |  |  |
| auto只用于lambda表达式。 |  |  |
|  |  |  |
| 注释 |  |  |
| 审查项目 | 结论 | 提示 |
| 注释清晰并且必要。 |  |  |
| 注释不会导致误解。 |  |  |
| 在不易理解的地方，注释给出了足够信息了解算法。 |  |  |
| 文件的开头注释了作者信息。 |  |  |
| 头文件中，类、结构、typedef和枚举都给出了@brief注释。必要时给出了@details。 |  |  |
| 头文件中，函数给出了@brief注释。有参数时给出了@param（必要时使用[in],[out]和[in,out]）。有返回值时给出了@return。必要时给出了@details，@warning或者@note。 |  |  |
| 头文件中，类、结构和枚举成员都用///<给出了注释。 |  |  |
| 正确地使用了@ref、@p、@c和@code...@endcode。 |  |  |

2. 构建过程

|  |  |  |
| --- | --- | --- |
| 静态分析 |  |  |
| 审查项目 | 结论 | 提示 |
| 已经使用Cppcheck对代码进行了审查，error都已经解决。 |  |  |
|  |  |  |
| 编译情况 |  |  |
| 审查项目 | 结论 | 提示 |
| 代码在Linux和Windows平台均能顺利编译。 |  |  |
| 编译器报错为0。 |  |  |
| 编译器警告或者为0，或者都确保对程序正常运行无影响。 |  |  |
|  |  |  |
| 文档质量 |  |  |
| 审查项目 | 结论 | 提示 |
| 源代码文档全面准确。 |  |  |
| 用户手册的相关部分全面准确。 |  |  |
| 尽量提供一个案例教程。 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |