#### C++技术管理规范

**一、代码风格**

**a. 强制**

1. 所有代码必须使用UTF-8编码

2. 使用4个空格缩进，禁止使用Tab

3. 每行代码长度不得超过120个字符

4. 头文件必须使用.h或.hpp后缀，源文件必须使用.cpp后缀

5. 类名使用大驼峰命名法(如MyClassName)，变量和函数使用小驼峰命名法(如myVariableName)

**b. 推荐**

6. 左大括号{不换行，与语句同行

7. 运算符前后、逗号后添加空格

8. 指针和引用符号(\*和&)与类型名相连，与变量名分开(如Type\* ptr)

9. 每个文件应包含不超过一个顶级类或结构体定义

**c. 允许**

10. 简单的getter/setter方法可以写在一行

11. 短小的lambda表达式可以写在一行

**二、头文件管理**

**a. 强制**

12. 头文件必须包含#pragma once防止重复包含

13. 头文件应包含其所有依赖的必要头文件

14. 禁止在头文件中使用using namespace

15. 头文件中的函数声明必须包含参数名(不只是类型)

**b. 推荐**

16. 头文件应按以下顺序包含：相关头文件、C系统头文件、C++标准库头文件、其他库头文件、项目内头文件

1. 尽量减少头文件间的依赖关系
2. 模板定义应放在头文件中

**c. 允许**

19. 在源文件中可以使用using namespace，但仅限于当前源文件作用域

20. 在实现文件中使用前向声明减少头文件包含

**三、类设计**

**a. 强制**

21. 单参数构造函数必须使用explicit关键字

22. 禁止在构造函数和析构函数中调用虚函数

23. 含有虚函数的类必须将析构函数声明为虚函数

24. 遵循RAII原则管理资源

**b. 推荐**

25. 类成员变量应声明为private，通过方法提供访问

26. 保持类短小，单一职责

27. 优先使用组合而非继承

28. 接口类(纯虚类)命名以I开头(如IInterface)

**c. 允许**

29. 简单的数据聚合类可以使用struct并公开成员变量

30. 在性能关键路径可以使用PImpl惯用法

**四、内存管理**

**a. 强制**

31. 禁止使用裸new和delete，使用智能指针(unique\_ptr, shared\_ptr)

32. 禁止使用malloc和free

33. 禁止内存泄漏，使用工具定期检查

**b. 推荐**

34. 优先使用unique\_ptr而非shared\_ptr

35.使用make\_shared和make\_unique而非直接构造智能指针

36. 大对象或容器考虑使用std::vector而非数组

**c. 允许**

37. 在性能极其敏感的场景可以谨慎使用裸指针，但必须有明确的资源管理方案

38. 与C接口交互时可以使用裸指针，但应立即封装

**五、现代C++特性**

**a. 强制**

39. 使用nullptr而非NULL

40. 使用auto减少冗余类型声明，但必须保证代码可读性

41. 使用override关键字明确表示重写虚函数

42. 使用constexpr表示编译期常量

**b. 推荐**

43. 优先使用enum class而非传统enum

44. 使用范围for循环(for(auto& item : collection))

45. 使用=default和=delete明确特殊成员函数意图

46. 适当使用移动语义和完美转发

**c. 允许**

47. 在简单场景可以使用结构化绑定

48. 在模板元编程中可以使用SFINAE或概念(concepts)

**六、错误处理**

**a. 强制**

49. 禁止使用异常作为流程控制手段

50. 错误处理必须明确，不能忽略返回值

**b. 推荐**

51. 优先使用错误码而非异常

52. 使用noexcept标记不会抛出异常的函数

1. 在构造函数中失败应使用错误码或工厂模式

**c. 允许**

54. 在底层库中可以谨慎使用异常，但必须明确文档说明