D6K2编码规范

1. 变量

1.1 命名

类成员变量：int32 m\_nCount; string m\_strName; CTest\* m\_pTest;

结构成员变量：int32 Count; string Name; Point \*pTest;

局部变量：int32 nCount; string strName; CTest\* pTest;

全局变量：int32 g\_nCount; string g\_strName; CTest\* g\_pTest;

1.2 初始化

任何变量在使用前必须初始化

1.3 使用

指针在使用前，必须判断是否为NULL

1. 函数

2.1 命名

int32 ThisIsAFunction(int32 nCount,string strName,CTest\* pTest);

函数命名采用大小写字母结合的形式，但专有名词不受限制。

2.2 代码格式

函数体明确分三部分，初始化区，核心逻辑区，返回值区

2.3 复杂度

函数主体不允许前后出现多个循环

函数代码长度不得超过200行

函数过于复杂则拆分为多个函数

1. 文件

3.1 头文件

定义：一个头文件只允许定义一个外部使用的类，该类内部使用的结构或类可在同一头文件定义

包含：在.h文件中，尽可能少的包含其他头文件，尽可能使用类的前置声明。顺序为：系统，库，自定义

3.3 实现文件

包含：尽可能在实现文件中包含所需的头文件。顺序为：系统，库，自定义

格式：实现文件中的函数顺序，与头文件保持一致

3.4 文件名

文件名一般用单词命名，全小写，具有自描述性；多个词组中间用“\_”隔开。

1. 类与结构

4.1 命名

类：class CTest，结构 struct TEST

4.2 代码格式

类头文件定义分类区，辅助区和多个功能区；

类区：存放构造和析构函数，以及类级成员变量声明(如：类的父指针)和常量定义，类的初始化和清理接口和类的核心功能函数

辅助区：存放类的辅助功能函数

功能区：类的子功能按区存放（子功能的成员函数，成员变量和常量定义）

4.3 复杂度

类的子功能过多或过于复杂，则需拆分子类实现

1. 命名空间

最好不要在全局使用using指示符，以保证命名空间下的所有名称都可以正常使用。

// 禁止 —— 污染名字空间

using namespace std;

在 .cpp 文件, .h 文件的**函数**, **方法**或**类**中, 可以使用 *using* 关键字.

// 允许: .cpp 文件中

// .h 文件的话, 必须在函数, 方法或类的内部使用

using ::foo::bar;

在 .cpp 文件, .h 文件的函数, 方法或类中, 允许使用名字空间别名.

// 允许: .cpp 文件中

// .h 文件的话, 必须在函数, 方法或类的内部使用

namespace fbz = ::foo::bar::baz;

1. 注释

标准注释

1. 工程

SOLUTION视图中，根据各代码文件的用途，建立相应的目录存放

1. 系统

8．1 通用系统变量命名

模拟量:AIN

开关量:DIN

遥调:AOUT

遥控:DOUT

8．2 通用数据变量类型

UINT32 uint32

INT32 int32

参见datatypes.h