**C++编程规范之命名规范**

2018年08月06日 20:10:27 [Simple\_Man\_Just](https://me.csdn.net/Simple_Man_Just) 阅读数 925更多

分类专栏： [c++编程语言](https://blog.csdn.net/simple_man_just/article/category/7643015" \t "_blank)

版权声明：本文为博主原创文章，遵循[CC 4.0 BY-SA](http://creativecommons.org/licenses/by-sa/4.0/)版权协议，转载请附上原文出处链接和本声明。

本文链接：<https://blog.csdn.net/Simple_Man_Just/article/details/81460443>

C++编程规范之命名规范

摘要：文档为C++于洋编程时的命名规范。包括通用命名规则、文件命名、类型命名、变量命名、常量命名、函数命名、命名空间、宏命名、命名规则例外。

最重要的一致性规则是命名管理，命名风格直接可以直接确定命名实体是：类型、变量、函数、常量、宏等等，无需查找实体声明，我们大脑中的模式匹配引擎依赖于这些命名规则。

命名具有随意性，在满足编译器规则的情况下，按照个人喜好时，由于程序猿之间的差异性，导致代码的可读性很差，所以按照一定的通用规则进行，很有必要性。

**通用命名规则（General Naming Rules）**

1.命名中只能出现数字、字母和下划线，且不能以数字开头

2.在c/c++中，严格区分大小写

3.尽可能给出描述性名称，不要节约空间，让别人很快理解你的代码的重要性更高

            int num\_errors; // Good

            int num\_completed\_connections; // Good

            丑陋的命名使用模糊的缩写或随意的字符：

                        int n; // Bad – meaningless

                        int nerr; // Bad – ambiguous abbreviation

                        int n\_comp\_conns; // Bad – ambiguous abbreviation

            类型和变量一般为名词：如m\_nOpType

            函数名通常是动词+名词：GetXmlBranch（）

​​​​​​​**文件命名（File Names）**

1.文件命名时，可以包含文件命名中包含的字符（如短线 - ）

2.文件名要全部小写，可以包含下划线(\_)和短线(-)，按照项目的约定来。

            ussdsessionmng.h/ussdsessionmng.cpp

3.内联函数必须放在.h文件中，

如果内联函数比较短，就直接放在.h文件中

 如果内连输比较长，可以放到以.inl结尾的文件中

包含大量内联函数的类有如下：

ussdsessionmng.h // the class declaration

ussdsessionmng.cpp // the class definition

ussdsessionmng.inl // inline functions that include lots of code.

​​​​​​​**类型命名(Type Names)**

类型命名包含类、结构体、枚举、类型定义(typeof)

类型命名每个单词以大写字母开头，不包含下划线：

类型命名的组成：类型说明符+描述单词；

类(C)：CUSSDSessionMng

结构体(stru)：struUSSDProxyCBData

枚举(EN)：

            enum EN\_PROXY\_SUBMIT\_RESP\_CMD\_STATUS\_TAKE\_ERROR

{

              PROXY\_CMD\_STATUS\_NOT\_TAKE\_ERROR = 0,

              PROXY\_CMD\_STATUS\_TAKE\_ERROR = 1,

              EN\_PROXY\_SUBMIT\_RESP\_CMD\_STATUS\_END

};

类型定义(继承原有形式)：typedef CAccess CBccess；

​​​​​​​**变量命名(Variable Names)**

变量命名的组成：作用域(类成员、静态、全局)标示\_类型说明符+描述说明

作用域标示：类成员-m；静态-s；全局-g；静态类成员-sm；静态全局-sg

类型说明符：整形-n；布尔型-b；字符型-c；浮点型-f；指针型变量和迭代子-p；字符串-sz(string terminated by 0 character)；结构体-st；枚举-en；CHString-str；特别指向函数的指针和函数对象指针(point of function)-pfn；数组(grid)-g；类的实例-i或其它

​​​​​​​**常量命名(Constant Names)**

常量：全局常量和非全局常量

全局常量：全部大写，单词之间用下划线连接：const int ERR\_FAIL=0

非常量命名组成：类型标示+常量说明：const int nCount=0

​​​​​​​**函数命名(Function Names)**

DeletUrl

​​​​​​​**命名空间(Namespace Names)**

命名空间全小写，其命名基于项目名称和目录结构

​​​​​​​**枚举命名(Enumerator Names)**

枚举是类、常量、宏的结合体

​​​​​​​**宏命名(Marco Names)**

全部大写单词，用下划线连接

​​​​​​​**命名规则例外(Exceptions to Naming Rules)**

当命名与现有C/C++实体相似的对象时，可参考现有命名约定：  
bigopen()  
             函数名，参考open()  
uint  
             typedef 类型定义  
bigpos  
            struct 或class，参考pos  
sparse\_hash\_map  
            STL 相似实体；参考STL 命名约定  
LONGLONG\_MAX  
            常量，类似INT\_MAX  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
**译者：**命名约定就相对轻松许多，在遵从代码一致性、可读性的前提下，略显随意：  
1. 总体规则：不要随意缩写，如果说ChangeLocalValue写作ChgLocVal还有情可  
原的话，把ModifyPlayerName写作MdfPlyNm就太过分了，除函数名可适当为动  
词外，其他命名尽量使用清晰易懂的名词；  
2. 宏、枚举等使用全部大写+下划线；  
3. 变量（含类、结构体成员变量）、文件、命名空间、存取函数等使用全部小写+下划线 ，  
类成员变量以下划线结尾，全局变量以g\_开头；  
4. 普通函数、类型（含类与结构体、枚举类型）、常量等使用大小写混合，不含下划线；  
5. 参考现有或相近命名约定。