EasyIot代码命名规范

目标

1. 减少代码的阅读难度；
2. 从命名上能体现出所有特性；

文件命名规则

使用**驼峰命名法**，例如：

```
fileUtil.c
```

变量命名规则

变量的命名规则要求用**匈牙利法则**，即开头的字母用变量的类型，其余的部分描述变量的含义，每个单词的首字母大写。

变量名：变量类型+变量英文全拼

普通变量

```
bool 用b开头 b标志寄存器
int 用i开头 iCount
short int 用n开头 nStepCount
long int 用l开头 lSum
char 用c开头 cCount
unsigned char 用by开头
float 用f开头 fAvg
double 用d开头 dDeta
unsigned int(WORD) 用w开头 wCount
unsigned long int(DWORD) 用dw开头 dwBroad
字符串 用s开头 sFileName
用0结尾的字符串 用sz开头 szFileName
```

指针变量

对一重指针变量的基本原则为：“p”+变量类型前缀+命名，如一个float*型应该表示为pfStat。对二重指针变量的基本规则为：“pp”+变量类型前缀+命名。对三重指针变量的基本规则为：“ppp”+变量类型前缀+命名。

全局变量

全局变量用g*开头,如一个全局的长型变量定义为g_lFailCount。即：变量名=g*+变量类型+变量的英文意思(或缩写)。此规则还可避免局部变量和全局变量同名而引起的问题。

静态变量

用s*开头,如一个静态的指针变量定义为s_plPerv_Inst。即：变量名=s*+变量类型+变量的英文意思(或缩写)

枚举类型(enum)

要求用枚举变量或其缩写做前缀。并且要求用大写。如：

```
enum EMDAYS
{
    EMDAYS_MONDAY,
    EMDAYS_MTUESDAY
};
```

结构体

```
struct SPoint
{
    int iX;
    int iY;
};
```

联合体

```
struct UPoint
{
    int iX;
    int iY;
};
```

常量

要求常量名用大写，常量名用英文表达其意思。当需要由多个单词表示时，单词与单词之间必须采用连字符“_”连接，如：

```
#define FILE_NOT_FOUND   (0x20)
```

const 变量

要求在变量的命名规则前加入c。*即：c*+变量命名规则；示例：

```
const char* c_pszFileName;
```

函数命名规范

函数由函数名称、函数参数、函数返回值和函数体组成，各部分要求如下

函数名称 = 返回类型+函数含义（首字母大写）+符合变量命名的参数列表+{}函数体（"{"允许和函数放在同一行，但不允许使用TAB，用4个字节的空格代替），例如：

```
int iGetFileData(const char *c_pszPathname) {
    return 0;
}
```