**编码规范：**

统一使用UTF-8编码

**命名规范：**

1.用Pascal规则来命名方法和类型，即每个单词首字母大写。   
public   class   DataGrid   
{   
          public   void   DataBind()   
         {   
  
          }   
}   
  
2.用Camel规则来命名局部变量和方法的参数，即第一个单词首字母小写，其他的大写.     
public   class   Product   
{   
          private   string   m\_productId;   
          private   string   m\_productName;   
  
         public   void   AddProduct(string   productId,string   productName)   
         {   
  
         }   
} 

3.成员变量的命名。要加m\_前缀，构建函数的参数尽量采用与成员变量去掉前缀后相同的名字。

4.方法的命名。一般将其命名为动宾短语。   
public   class   File   
{   
            public   void   CreateFile(string   filePath)   
           {   
  
             }   
             public   void   GetPath(string   path)   
            {   
  
            }   
}     
  
5．局部变量的名称要有意义。   
不要用x，y，z等等，用For循环变量中可使用i,   j,   k,   l,   m,   n。   
public   class   User   
{   
          public   void   GetUser()   
         {   
                   string[]   userIds={ "ziv ", "zorywa ", "zlh "};   
  
                   for(int   i=0,k=userIds.Length;i <k;i++)   
                  {   
  
                  }   
         }   
}

6. 对象初始化，能简写最好简写：

ValueText vt=new ValueText{Value=10, Text="Age"};

7.文件名要能反应类的内容，最好是和类同名并且单词首字母大写，一个文件中一个类或一组关连类。

**书写规范：**

1.类的声明。成员变量和方法都是按照私有、保护、公开的顺序声明；先变量、后信号/槽、最后是方法。

2.类的实现。静态变量的初始化要放到首部；成员方法的实现顺序与声明顺序一致。

3．建议局部变量在最接近使用它时再声明。   
  
  
4.把引用的系统的命名空间或头文件和自定义或第三方的用一个换行把它们分开。     
using   System;   
using   System.Web.UI;   
  
using   CSharpCode;   
using   CSharpCode.Style; 

5.一个类要分头文件和实现文件，并且头文件要有防止重复包含的措施：

#ifndef CONFIG\_H

#define CONFIG\_H

…..

#endif   
  
6.大括号 "{“要新起一行。   
public   Sample()   
{   
           //   
           //   TODO:   在此处添加构造函数逻辑   
           //   
}

**注释规范：**

1 源文件头部注释

Ø 列出：版权、作者、编写日期和描述。

Ø 示例：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Copyright:bupt

Author:

Date:2010-08-25

Description:描述主要实现的功能

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

每行不要超过80个字符的宽度。

2 函数头部注释

Ø 列出：函数的目的/功能、输入参数、输出参数、返回值、调用关系（函数、表）等。

Ø 示例：下面这段函数的注释比较标准，当然，并不局限于此格式，但上述信息建议

要包含在内。

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function:       // 函数名称

Description:    // 函数功能、性能等的描述

Calls:          // 被本函数调用的函数清单

Table Accessed: // 被访问的表（此项仅对于牵扯到数据库操作的程序）

Table Updated: // 被修改的表（此项仅对于牵扯到数据库操作的程序）

Input:          // 输入参数说明，包括每个参数的作

                  // 用、取值说明及参数间关系。

Output:         // 对输出参数的说明。

Return:         // 函数返回值的说明

Others:         // 其它说明

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

5 对代码的注释

注释总是加在程序的需要一个概括性说明或不易理解或易理解错的地方。注释语言应该简练、易懂而又含义准确，避免二义性；所采用的语种首选是中文，如有输入困难、编译环境限制或特殊需求也可采用英文。注释应与其描述的代码相近，对代码的注释统一放在其上方，避免在一行代码或表达式中间使用注释。采用//注释，注释应与所描述内容进行同样的缩进。