**编码规范**

**（Coding Specification）**

**C# .NET篇**

|  |  |  |  |
| --- | --- | --- | --- |
| 作 者： | 王 健 |  | 2018-4-16 |
|  |  |  | 0.1 |

**变更记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **章节号** | **章节名称** | **变更内容描述** | **变更****日期** | **版本号** | **批准人** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**目 录**

[1. 性能 4](#_Toc296161987)

[2. 异常 4](#_Toc296161988)

[3. 方便性 5](#_Toc296161989)

[4. 维护 5](#_Toc296161990)

[5. 测试 6](#_Toc296161991)

[6. 规范 6](#_Toc296161992)

[7. 技巧 8](#_Toc296161993)

# 性能

* 1. 在定义变量时严格按照位数和业务结合进行定义，比如无符号整形数据就是uint而不是int. 即用多大定义多大。
	2. 在能确定变量类型的地方用变量的当前类型，尽量少用object。
	3. 值类型变量声明时，如果需要给默认空值，则定义为：类型? 并且赋值 null。如整型就是 int?
	4. 不要使用三目运算符，而用 If else。
	5. 对数组的遍历用for。
	6. 对datatable或者对象列表的遍历用foreach。
	7. 用for循环时，一开始把列表的长度取出而不是每次通过变量去判断整个列表的count。如

int rowCount = testList.Length;

for (int i = 0; i < rowCount; i++)

* 1. 在大篇幅的字符串拼接时用StringBuilder.Append而不是string+=的格式。
	2. 如果一个方法的返回值是布尔型，或者一个变量是布尔型，则在对方法的返回值，或者变量进行if判断时，不需要跟true或者false进行判断。比如定义一个变量 bool checkBool = true;

则判断checkBool时只需要 if(checkBool)。

* 1. 对于某些特殊的对象（字节，图片，数据库链接池等）需要释放资源时，使用using语句。

# 异常

* 1. 在高精度的数值计算中，使用decimal，使用decimal时要注意溢出的异常。没有精度要求的计算，使用float或者double，具体用什么参照 (一)1。
	2. 尽量避免强制类型转换，如果不得不进行类型转换用显示转换方式。
	3. 关于对象的强制转换时，如果不确定转换对象是否为null，则使用as语句。如：

Button btnTest = Page.FindControl("btnTest") asButton;

* 1. 对引用类型变量进行取值等操作时，先确定当前变量是否为null。

# 方便性

* 1. 对于VS2010只有参数个数不相同的方法重载，建议使用 “命名参数” 比如定义为：

publicvoid DoSomething(int x = 0, int y = 0){

调用可以为：

DoSomething();

DoSomething(1);

 DoSomething(1, 1);

 DoSomething(x: 1);

 DoSomething(y: 1);

 DoSomething(x: 1, y: 1);

 DoSomething(y: 1, x: 1);

* 1. 对于后台取出的datatable或者对象列表要进行2次处理（分组，排序，筛选）时，使用匿名函数（Lambda表达式）

如：下面给出关于datatable的某一列求和

double sumStopTime = stopTime.AsEnumerable().Sum(row => row.Field<double>("PointValue"));

结果是：dtLogTime 里的PointValue 列 的值的和

* 1. 对于操作数据库时的事务处理，使用分布式事务处理方式

using (TransactionScope tsDataBase = newTransactionScope())

 {

 执行数据库语句组合.

 tsDataBase. Complete();

}

* 1. 需要使用常量时，用static readonly 修饰。

# 维护

* 1. 在写方法时把能共通的部分(方法，模块等)尽量共通。
	2. 如果两个方法里只有细微的差别，用方法重载。
	3. 在创建类时，不要使用静态方法和非静态方法混合定义。
	4. 在接口实现时，使用隐式接口成员实现。
	5. 不要在表示式中使用赋值语句。
	6. 把每个业务模块单独存放，存放方式根据业务大小，大的业务建议单独创建个类，而小的业务建议通过创建方法来实现复杂的业务细化。不要把所有的业务写在一个方法或者一个类里。
	7. 不要让不同层次间的变量重名。

# 测试

* 1. 对于单个方法循环嵌套不超过三层。
	2. 对于单个方法，则单个方法的圈复杂度不超过10。

 具体计算为:

* + 1. If 语句，每个分支为1个复杂度。
		2. 一个循环语句(for,foreach,while,do-while)为2个复杂度。
		3. Swich语句每个case为一个复杂度。
	1. 避免对浮点类型的数据做等于或不等于判断。
	2. 为所有的swich语句提供default分支。

# 规范

* 1. 枚举类型，类，枚举值，事件，只读的静态字段（常量），接口，方法，命名空间，属性 使用“首字母和后面连接的每个单词的首字母都大写” 的方式命名。
	2. 参数，方法内的变量，请使用“首字母小写，而每个后面连接的单词的首字母都大写”的方式命名。
	3. 变量命名要使用类型前缀 比如 string 命名为 str开头。
		1. int => int
		2. unint => uint
		3. long => long
		4. char => char
		5. string => str
		6. Datetime => dtime
		7. Datatable => dt
		8. Database => db
		9. bool => bl
	4. 不要使用关键字来命名。
	5. 不要使用”\_”来命名。
	6. 接口命名加前缀”I”。
	7. 控件命名按照 控件名简写+英文描述，英文描述按照（六）1的方式命名

下面给出控件名和简写对应表

* + 1. Label => lbl
		2. TextBox => txt
		3. Button => btn
		4. LinkButton =>lnkbtn
		5. ImageButton =>imgbtn
		6. DropDownList =>ddl
		7. ListBox =>lst
		8. DataGrid =>dg
		9. DataList =>dl
		10. CheckBox =>chk
		11. CheckBoxList => chkls
		12. RadioButton =>rdo
		13. RadioButtonList =>rdolt
		14. Image =>img
		15. Panel =>pnl
		16. Calender =>cld
		17. AdRotator =>ar
		18. Table =>tbl
		19. RequiredFieldValidator => rfv
		20. CompareValidator =>cv
		21. RangeValidator =>rv
		22. RegularExpressionValidator =>rev
		23. ValidatorSummary =>vs
		24. CrystalReportViewer =>rptvew
		25. GridView =>gv
		26. HiddenField => hid
		27. Combox -> cmb
	1. 每个方法 有方法说明，参数说明，返回值说明。并且方法用#region 和 #endregion 标注
	2. 每行只有一个语句,比如请避免： a++； b++;
	3. 代码列宽控制在110字符左右。
	4. 在以下情况下使用两个空行
		1. 接口和类定义之间。
		2. 枚举和类的定义之间
		3. 类和类的定义之间
	5. 在以下情况下使用一个空行
		1. 方法与方法，属性与属性之间。
		2. 方法中变量声明与语句之间。
		3. 方法中不同逻辑块之间。
		4. 方法中返回语句与其他语句之间。
		5. 属性与方法，属性与字段，方法与字段之间。
		6. 注释与它注释的语句间不空行，但与其他的语句间空一行。
	6. 每个复杂的操作写好相应的注释，并且每次版本修改后添加相应的注释（修改人，修改日期，修改内容）
	7. 尽量保证每两行有一个注释说明（但是与业务相关的操作，数据的操作，以及逻辑类的操作块，一定要有注释说明）

# 技巧

* 1. 在前台aspx页面或者后台cs文件写完后，按下Ctrl+E+D 实现自动对齐。
	2. 写完一个后台文件后，点击右键->组织using->移除和排序。