Coding Standard

I. Quy ước đặt tên và kiểu dữ liệu: 2
II. Những chú ý khi lập trình..... 8
III. Tài liệu tham khảo:..... 12
IV. Editors:..... 12

CODING STADARD

I. Quy ước đặt tên và kiểu dữ liệu:

1. Sử dụng kiểu Pascal để đặt tên cho namespace, lớp (namespace, class, struct, enum, interface), phương thức, biến kiểu public và hằng hằng.

```
public class ClassName
{
    const int DefaultSize = 100;
    public void Method()
    {

    }
}
```

2. Sử dụng kiểu Camel để đặt tên cho biến cục bộ và đối số của các hàm/phương thức:

```
public void Method(int myArgument)
{
    int localVariable;
}
```

3. Tiếp đầu ngữ của kiểu interface là I, khai báo tên theo kiểu Pascal:

```
public interface IMyInterface
{
    void Method();
}
```

4. Sử dụng tiếp đầu ngữ m_ cho biến kiểu private của lớp. Sử dụng kiểu Pascal cho phần tên còn lại:

```
public class ClassName
{
    private int m_PrivateMember;
}
```

5. Hậu tố của lớp kế thừa từ Attribute là Attribute:

```
public class MyCustomAttribute: Attribute
{

}
```

6. Hậu tố của lớp kế thừa từ Exception là Exception:

```
class MyCustomException : Exception
{

}
```

7. Đặt tên hàm/phương thức theo kiểu Đông từ + Đối tượng (verbobject):

```
public void ShowDialog()
{
    base.ShowDialog();
}
```

8. Hàm/phương thức có trả về kết quả, tên đặt cho nó phải mô tả được kết quả mà nó trả về:

```
public string GetUserName()
{
    string userName = string.Empty;
    ...
    return userName;
}
```

9. Đặt tên biến miêu tả được nội dung/mục đích sử dụng của nó:

a. Tránh đặt tên biến với chỉ một chữ cái, chẳng hạn như i, t thay vào đó ta nên đặt tên là index, temp.
b. Tránh sử dụng kí pháp Hungary để khai báo cho các biến kiểu public, protected.
c. Không viết tắt, ví dụ không được viết là num thay cho number.

10. Luôn sử dụng kiểu dữ liệu gốc (Predefine type), không sử dụng kiểu được đặt tên lại của kiểu đó trong namespace "System":

Được sử dụng	Không được sử dụng
bool	Boolean
byte	Byte
double	Double
int	Int32
object	Object
short	Int16
string	String

11. Đối với lớp sử dụng chung cho các đối tượng, sử dụng một hoặc nhiều chữ cái đầu tiên và viết hoa để đặt tên cho tên đối tượng đại diện:

- Type(Kiểu dữ liệu) được viết tắt thành T

```
public class Generic<T>
{
    private T[] m_ObjectList;
    public Generic(int numberOfobjects)
    {
        m_ObjectList = new T[numberOfobjects];
    }
}
```

12. Tên của namespace phải mang đầy đủ ý nghĩa, ví dụ: tên công ty hoặc tên sản phẩm.

```
namespace CodingStandardChecker
{

}
```

13. Tránh khai báo kiểu dữ liệu dạng "namespace"[".subnamespace.[...]"]."kiểu dữ liệu" bằng cách sử dụng từ khóa using:

-Tránh sử dụng:

```
System.DateTime myBirthday;
```

- Nên sử dụng:

```
using namespace System;
...
DateTime myBirthday;
```

14. Tránh sử dụng từ khóa using trong một namespace:

Tránh sử dụng:

```
namespace CodingStandardChecker;
{
    using system = System;
}
```

15. Gom tất cả các namespace của cùng nhà cùng cấp thành một nhóm:

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using MyNameSpace;
using MyLibrary;
```

16. Khi sử dụng delegate, phải cung cấp cho delegate hàm mà chúng ta muốn đóng gói, không được thay đổi hàm được đóng gói bởi delegate:

```
public delegate int AddDelegate(int a, int b);
public int Add(int a, int b)
{
    return a + b;
}
AddDelegate = Add;
```

17. Sử dụng tab chèn vào đầu dòng code để định hình, không sử dụng khoảng trắng để thay thế tab. Số lượng tab chèn vào đầu mỗi dòng code không nên quá 4.

18. Cấp của comment(ghi chú) phải cùng cấp với dòng code được comment:

```
//Comment here if necessary;
public int subtract(int a, int b)
{
    //Comment here if necessary;
    int result = a - b;
}
```

19. Comment phải đúng chính tả, comment không đúng chính tả thể hiện sự luộm thuộm, không có hệ thống trong phát triển phần mềm.

20. Các biến thành viên của một lớp phải được khai báo ở trên cùng của lớp, các biến thành viên được phân cách với các phương thức bằng một dòng trống.

```
public class MyClass
{
    private int m_FirstVariable;
    private int m_SecondVariable;



    public void FirstMethod()
    {
        ...
    }
    ...
```

```
}
```

21. Khai báo biến cục bộ gần nhất so với vị trí lần đầu tiên nó được sử dụng:

```
public List<int> GetRandomIntegersList(int numberOfintegers)
{
    List<int> randomIntegerslist = new List<int>();
    Random randomCreator = new Random(DateTime.Now.Millisecond);
    for (int i = 0; i < numberOfintegers; ++i)
    {
        randomIntegerslist.Add(randomCreator.Next());
    }
    return randomIntegerslist;
}
```

22. Tên của file chứa source code của một lớp(class) phải cùng tên với lớp đó.

23. Khi sử dụng kiểu partial để chia một lớp ra thành nhiều file thì phải đặt tên file đúng với tên lớp và nhiệm vụ mà phần lớp chứa trong file đó đảm nhiệm một cách logic:

Trong file MyClass.cs: chứa các thuộc tính và phương thức thực thi của lớp

```
public partial MyClass
{
    ...
}
```

Trong file MyClass.Designer.Cs: Chứa các phương thức khởi tạo và vẽ giao diện của lớp.

```
public partial MyClass
{
    ...
}
```

24. Đặt dấu '{' và dấu '}' riêng biệt ở một dòng.

25. Đối với những hàm không tên, canh chỉnh khai báo hàm theo khai báo delegate của hàm đó:

```
delegate void AnonymousMethodDelegate(String);
public void GiveMessage()
{
    AnonymousMethodDelegate = delegate(String message)
                                {
                                    MessageBox.Show(message);
```

```
                                    };
        AnoymousMethodDelegate("I'm noname");
    }
```

26. Bắt buộc sử dụng '()' sau delegate:

```
    delegate void DoSomethingDelegate();

    - Đúng qui tắc:
    DoSomethingDelegate = delegate()
                          {
                             //DoSomething;
                          };

    - Sai qui tắc:
    DoSomethingDelegate = delegate
                          {
                             //DoSomething;
                          };
```

27. Khi sử dụng biểu thức Lambda("(bien)=>{khối mã lệnh}") để không khai báo kiểu của biến cho hàm không tên, canh chỉnh biểu thức theo khai báo của biến Lambda:

```
    delegate void LambdaDelegate(String Message);
    public void GiveMessage()
    {

        AnonymousMethodDelegate = (message) =>
                                  {
                                     string finalMessage = "I Said that " +
message;
                                     MessageBox.Show(message);
                                  };
        AnoymousMethodDelegate("I'm noname");
    }
```

28. Sử dụng biểu thức Lambda in-line(biểu thức Lambda được đặt trên một dòng và được thực thi duy nhất môt lần) khi và chỉ khi biểu thức đó có một dòng lệnh trong khối lệnh:

```
    delegate void LambdaDelegate(String Message);
    public void FunctionUsingLambdaDelegate(LambdaDelegate
myLambdadelegate)
    {
        //Do something;
    }
```

```
FunctionUsingLambdaDelegate(message=>MessageBox.Show(message))
;
```

II. Những chú ý khi lập trình

1. Tránh đặt nhiều class trong 1 file.

2. Tránh có nhiều namespace trong 1 file. 1 file nên kết hợp các type vào 1 namespace.

3. Tránh 1 file có nhiều hơn 500 dòng (không bao gồm mã máy)

4. Tránh những phương thức có nhiều hơn 200 dòng.

5. Tránh những phương thức có hơn 5 đối số.

6. Một dòng không được quá 120 ký tự.

7. Không nên tự sửa bất kỳ dòng nào của mã máy..
   a. Nếu thay đổi mã máy, hãy theo format và style của chuẩn code này.
   b. Sử dụng từ khóa partial để chi lớp thành nhiều file chính giữ nhiệm vụ khác nhau.

8. Tránh ghi chú thích những điểu quá hiển nhiên trog code. Tự code nên giải thích nó. code được gọi là tốt khi tên biến và tên hàm rõ ràng, không cần giải thích.

9. Văn bản chỉ dùng để giải thích giải thuật

10. Tránh giải thích, chứng minh phương pháp.
    a. Dùng extensive external documentation cho API documentation.
    b. Dùng comment method-level …

11. Nếu ngoại lệ là 0 và 1, không bao giờ được code cứng bằng số. Thay vào đó, luôn luôn định nghĩa 1 constant.

12. Chỉ sử dụng const cho những hằng số hiển nhiên như số ngày một tuần.

13. Tránh dùng const trên biến chỉ đọc. thay vào đó, dùng readonly.

```
Public class Myclass
{
    Public const int DayInWeek = 7 ;
    Public readonly int Number;
```

```
    Public Myclass (int someValue)
    {
       Number = someValue;
    }
}
```

14. Trong quá trình lập trình hãy kiểm tra sự hợp lệ của dữ liệu bằng Assert.

```
Using System.Diagnostics;
Object GetObject()
{…}
Object someObject = GetObject();
Debug.Assert(someObject != null);
```

15. Mỗi dòng code nên được kiểm tra kĩ lưỡng bởi lập trình viên (có thể sử dụng unit test).

16. Chỉ bắt những exception(ngoại lệ) mà bạn muốn.

17. Trong câu lệnh catch mà throw ngoại lệ, luôn luôn throw những exception nguyên bản (hay những ngoại lệ dựa trên nguyên bản) để duy trì vị trí trong stack của những ngoại lệ này.

```
Catch ( Exception ex)
{
   Messagebox.Show(ex.Message);
   Throw;
}
```

18. Tránh những lỗi như hàm trả về giá trị không đúng kiểu của dữ liệu.

19. Tránh tự định nghĩa các lớp Exception(các lớp này kế thừ từ lớp Exception).

20. Khi định nghĩa 1 lớp ngoại lệ:
    a. Lấy những ngoại lệ thông thường từ lớp Exception(kế thừa từ Exception).
    b. Cung cấp thông điệp của ngoại lệ mới và đánh số thứ tự cho ngoại lệ đó.

21. Tránh viết nhiều hàm Main() trong 1 chương trình.

22. Chỉ để public khi thật sự cần, hãy đặt là internal cho tất cả.

23. Tránh cho lớp khác truy xuất trực tiếp vào biến thành viên của lớp.

24. Tránh viết chương trình thực thi trên một vị trí đặc biệt.

25. Giảm thiểu kích thước của file biên dịch(.exe) bằng cách sử dụng class library(tạo ra file *.dll).

26. Tránh gán những giá trị cố định cho enum trừ khi giá trị của nó không theo thứ tự tăng dần.

```
//Đúng
Public enum color
{
   Red,Green,Blue
}

//Tránh
Public enum color
{
   Red = 1;
   Green = 2;
   Blue = 3;
}
```

27. Tránh đặt kiểu cho enum.

```
//Tránh
Public enum color: long
{

  Red,Green,Blue
}
```

28. Luôn luôn sử dụng dấu {} trong hàm if kể cả khi nó chỉ có 1 dòng lệnh.

29. Tránh sử dụng toán tử điều kiện ternary ((biểu thức điều kiện) ? kết quả 1 : Kết quả 2.

30. Tránh sử dụng tiền xử lý (#if … #endif). Hãy sử dụng hàm điều kiện :

```
[conditional ("MySpecialCondition")]
Public void MyMethod()
{
}
```

31. Tránh để hàm trả về giá trị boolean trong mệnh đề điều kiện, hãy đặt một biến cục bộ và kiểm tra nó. vd :

```
bool IsEverythingOk()
{
```

```
}

//Tránh
if(IsEverythingOk()
{
}

//Đúng
bool ok = IsEverythingOk();
if(ok)
{}
```

32. Luôn sử dụng mảng bắt đầu từ 0;

33. Trong tập hợp có chỉ mục(index), chỉ mục bắt đầu từ không.

34. Luôn luôn khởi tạo mảng reference type bằng vòng lặp for

```
Public class MyClass()
{}
Const int ArrarSize = 100;
MyClass[] array = new MyClass[ArraySize];
For(int index = 0; index < array.Length; index++)
{
   Array[index] = new MyClass[]
}
```

35. Tránh cung cấp thuộc tính public hay protected, sử dụng propeties.

36. Tránh sử dụng phương thức mà nó không làm gì khác ngoài việc truy xuất biến thành viên của lớp, thay vào đó hãy sử dụng phương thức tự động:

```
- Sử dụng:
public class ClassName
{
   public int Number
   {
      get;
      set;
   }
}

- Tránh sử dụng:
public class ClassName
{
   private m_Number;

   public int Number
   {
```

```
        get
        {
            return m_Number;
        }
        set
        {
            m_Number = value;
        }
    }
}
```

37. Không viết lại phương thức mới mà đã có khi lớp này kế thừa từ lớp khác, sử dụng từ khóa override để làm việc đó.

III. Tài liệu tham khảo:

- *IDesign CSharp Coding Standard của Juval Nowy.*

IV. Editors:

Võ Thị Mỹ Hạnh

Nguyễn Phan Thanh Trúc

Tăng Phương Quý