**Convenções de código para Backend**

Declarações

1. Nomenclatura de projeto

NomeDoCliente.NomeDoProjeto.NomeDaCamada.

ex: Oracle.DataExtractor.DAO

1. Namespace

PascalCase, não utilizar underscore.

ex: Oracle.DataExtractor.DAO

1. Controles visuais (Web Forms, Windows Forms, Windows 8.1, Windows Phone, id para tags HTML)

Notação hungara

ex: Label: lblMyControl

TextBox: txtMyControl

CheckBox: chkMyControl

CheckBoxList: cblMyControl

RadioButton: rbMyControl

RadioButtonList: rblMyControl

DropDownList: ddlMyControl

Button: btnMyButton

Panel: pnlMyControl

Validators: valMyControl

LinkButton/HyperLink: lnkMyLink

GridView: gvMyGridView

Table: tblMyTable

TableRow: trMyRow

TableCell: tdMyCell

Form: frmMyForm

1. Classes (concretas ou abstratas)

Use Pascal Case, substantivo singular.

ex: Noticia.cs

1. Métodos

Use Pascal Case, verbo.

ex: public void Buscar(...)

1. Propriedades (getters, setters)

Pascal Case, substantivo singular.

ex: Nome

Email

Telefone

1. Enumeradores

Pascal Case, substantivo singular para tipos e valores.

ex: Status: Novo, Antigo

1. Interfaces

Pascal Case, substantivo singular, com prefixo "I".

ex: IEqualityComparer

1. Variaveis privadas

Camel case, substantivo singular, prefixo "\_".

ex: \_nome

1. Variaveis públicas

Pascal case.

ex: Nome

1. Variaveis locais

CamelCase.

ex: index, count

1. Constantes

UpperCase.

ex: ERROR

1. Delagates/Eventos

Modificador de acesso preferencialmente publico para eventos.

ex: PascalCase

1. Tipos genéricos

PascalCase, prefixo "T".

ex: TModel, TBusiness

Codificação

1. Nomenclatura de variáveis

Evite usar abreviações, use palavras chaves que façam sentido para a regra específica.

1. Criação/inicialização de variáveis

Crie e inicialize as variáveis o mais proximo possível de sua utilização.

1. Ordenação de componentes de uma classe

1º Constantes (preferencialmente mantenha em um arquivo separado no projeto de Models)

2º Enumeradores (preferencialmente mantenha em um arquivo separado no projeto de Models)

3º Variaveis privadas

4º Variaveis publicas

5º Propriedades (preferencialmente mantenha em um arquivo separado no projeto de Models)

6º Construtores

7º Métodos

8º Eventos

1. Tratamenteo de exceções

Sempre incluir try/catch se possível tratar a exceção no contexto e devolver o tratamento correto (mensagens ou caminho alternativo),

Caso contrário, mantenha a exceção sendo disparada no catch e faça log do erro.

try{ ... } catch(Exception ex) { Log(ex); throw; }

1. Comentários de codigo

Sempre que modificar algum trecho de código do framework,

Documente-o logo em seguida.

Assim os outros entenderão do que se trata e o gerador de documentação utilizará o seu sumario.

1. Dispose

Utilize using sempre que disponível

1. Comparação

Nunca, nunca compare um booleano, por si só ele já é uma resposta.

Não trate um retorno de método dentro de um if, atribua a uma variavel e depois compare

Evite aninhar comparações

1. Parâmetros

Não ultrapasse o limite de no máximo sete parâmetros para um método.

1. Quantidade de linhas de um método

Tente ser o mais coerente possível e fazer apenas uma coisa por método.

Tente manter o método com 30 linhas no máximo.

1. Tipos nativos

Utilize os tipos nativos ao invés dos .net CLS. Use int ao invés de Int32

Glossário

Pascal Case = Inicia em caixa alta, cada palavra em caixa alta

Camel Case = Inicia em caixa baixa, cada palavra em caixa alta

UpperCase = Completamente caixa alta