Documentação de uso da DLL NGEnum

* Definição: A DLL NGEnum contém uma estrutura com classes dinâmicas abstratas que servem para ser usadas como enums.
* Vantagens: Ao contrário de Enum tradicionais a NGEnum pode ser herdado, a classe pai tem vários métodos para tratamentos de comparação.
* Implementação:

Para criar seus enums crie uma classe que herda da classe base(NGEnums), que contenha 3 construtores. Cada atributo será uma chave do seu enum elee serão declarados como readonly e o tipo é a própria classe, como mostrado no exemplo abaixo:

public class NomeDoSeuEnum : NGEnums<NomeDoSeuEnum>

{

public static readonly NomeDoSeuEnum ChaveDoEnum = new NomeDoSeuEnum("ChaveDoEnum");

public static readonly NomeDoSeuEnum OutraChave = new NomeDoSeuEnum("OutraChave");

public NomeDoSeuEnum() : base() { }

public NomeDoSeuEnum(object pObject) : base(pObject) { }

public NomeDoSeuEnum(int pId, object pObject) : base(pId, pObject) { }

}

Os atributos chaves podem ser declarados de algumas formas:

public static readonly NomeDoSeuEnum ChaveDoEnum = new NomeDoSeuEnum("ChaveDoEnum");

public static readonly NomeDoSeuEnum ChaveDoEnum = new NomeDoSeuEnum(1,"ChaveDoEnum");

No primeiro exemplo acima é criado um id automático com o hash do objeto.

No segundo exemplo acima o id é declarado explicitamente.

Como o construtor aceita um objeto, poderia ser inserido um objeto qualquer como chave, como outra classe, um type ou até mesmo um enum tradicional.

public class NomeDoSeuEnum : NGEnums<NomeDoSeuEnum>

{

public static readonly NomeDoSeuEnum Classe1 = new NomeDoSeuEnum(typeof(Classe1);

public static readonly NomeDoSeuEnum Classe2 = new NomeDoSeuEnum(typeof(Classe2);

public NomeDoSeuEnum() : base(None) { }

public NomeDoSeuEnum(object pObject) : base(pObject) { }

public NomeDoSeuEnum(int pId, object pObject) : base(pId, pObject) { }

}

Outra forma de criar um enum é herdar de outro enum já criado ao invés da base.

public class HerdadoDeOutroEnum : NomeDoSeuEnum

{

public static readonly NomeDoSeuEnum ChaveDoEnumHerdado = new NomeDoSeuEnum("ChaveDoEnumHerdado");

public static readonly NomeDoSeuEnum OutraChaveHerdado = new NomeDoSeuEnum("OutraChaveHerdado");

}

Note que as propriedades são do tipo do enum pai, e a classe não precisa de construtor.

Como os enums podem ser herdados infinitamente, se necessário pode-se selar a classe para ela não poder ser mais herdada.

public sealed class HerdadoDeOutroEnum : NomeDoSeuEnum

{

public static readonly NomeDoSeuEnum ChaveDoEnumHerdado = new NomeDoSeuEnum("ChaveDoEnumHerdado");

public static readonly NomeDoSeuEnum OutraChaveHerdado = new NomeDoSeuEnum("OutraChaveHerdado");

}

* Métodos de conversão.

Os métodos de conversão são métodos get para acessar os atributos, inteiro, string e objeto

NomeDoSeuEnum.ChaveDoEnum.ToInt();

NomeDoSeuEnum.ChaveDoEnum.ToString();

NomeDoSeuEnum.ChaveDoEnum.ToObject();

* Métodos de comparação.

Para comparação simples do objeto pode-se usar os métodos de comparação padrão ==, !=, Equals(recomendado).

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.ChaveDoEnum;

NomeDoSeuEnum objeto2 = NomeDoSeuEnum.OutraChave;

if (objeto1 == objeto2)

Console.WriteLine("objeto é igual");

if (objeto1 != objeto2)

Console.WriteLine("objeto é diferente");

if (objeto1.Equals(objeto2))

Console.WriteLine("objeto é igual");

Para comparar somente as propriedades use os métodos:

CompareId, para comparar somente o id do enum.

CompareKey, para comparar a string.

CompareObject, para comparar somente o objeto passado (funciona para string também se o objeto passa do for uma chave string, mas recomendamos usar o CompareKey para tal comparação).

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.ChaveDoEnum;

NomeDoSeuEnum objeto2 = NomeDoSeuEnum.ChaveDoEnum;

if (objeto1.CompareId(objeto2.ToInt())

Console.WriteLine("o id do objeto é o igual");

if (objeto1.CompareKey("ChaveDoEnum"))

Console.WriteLine("a chave do objeto é igual");

if (objeto1.CompareObject("ChaveDoEnum"))

Console.WriteLine("o objeto do objeto é igual");

* Enum composto.

NGEnum também possibilita a criação de enums compostos, pode-se adicionar elementos em um objeto já criado usando o método Add

Obs. O método Add retorna um novo enum com os elementos adicionados, ele deve ser atribuído

Obs. O id do enum composto será criado automaticamente

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.ChaveDoEnum;

objeto1 = objeto1.Add(NomeDoSeuEnum.OutraChave);

Console.WriteLine(objeto1.ToString());

Saída: ChaveDoEnum|OutraChave

Também podemos criar um enum composto usando o método New.

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.New(NomeDoSeuEnum.ChaveDoEnum, NomeDoSeuEnum.OutraChave); Console.WriteLine(objeto1.ToString());

Saída: ChaveDoEnum|OutraChave

* Métodos de comparação enum composto

Para comparar enum composto use o método CompareExact, ele irá comoarar se o enum é exatamente o mesmo passado no parâmetro, os mesmos elementos na mesma ordem.

Obs. O método irá funcionar com enum simples, mas recomendamos usar apenas para enum composto.

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.New(NomeDoSeuEnum.ChaveDoEnum, NomeDoSeuEnum.OutraChave);

if (objeto1.CompareExact(NomeDoSeuEnum.ChaveDoEnum, NomeDoSeuEnum.OutraChave))

Console.WriteLine("o enum composto é igual");

Também podemos verificar se existem o elemento no enum composto usando o método CompareSome.

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.New(NomeDoSeuEnum.ChaveDoEnum, NomeDoSeuEnum.OutraChave); Console.WriteLine(objeto1.ToString());

Saída: ChaveDoEnum|OutraChave

Ele irá verificar se existe os elementos não importando a ordem.

Obs. O método irá funcionar com enum simples, mas recomendamos usar apenas para enum composto.

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.New(NomeDoSeuEnum.ChaveDoEnum, NomeDoSeuEnum.OutraChave);

if (objeto1.CompareSome(NomeDoSeuEnum.ChaveDoEnum))

Console.WriteLine("O elemento existe no enum");

if (objeto1.CompareSome(NomeDoSeuEnum.OutraChave, NomeDoSeuEnum.ChaveDoEnum))

Console.WriteLine("Os elementos existem no enum");

Também podemos verificar se existem algum dos enums usando o método CompareAny.

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.New(NomeDoSeuEnum.ChaveDoEnum, NomeDoSeuEnum.OutraChave); Console.WriteLine(objeto1.ToString());

Saída: ChaveDoEnum|OutraChave

Ele irá verificar se existe qualquer dos elementos passados no parâmetro

Obs. O método irá funcionar com enum simples, mas recomendamos usar apenas para enum composto.

NomeDoSeuEnum objeto1 = NomeDoSeuEnum.New(NomeDoSeuEnum.ChaveDoEnum);

if (objeto1.CompareAny(NomeDoSeuEnum.ChaveDoEnum))

Console.WriteLine("Existe o enums");

if (objeto1.CompareAny(NomeDoSeuEnum.OutraChave, NomeDoSeuEnum.ChaveDoEnum))

Console.WriteLine("Existe algum dos enums");