

Entendendo o Reflection - ASP.NET/C#

Autor: Francke Peixoto

Declaração:

Autorizo a publicação de meu artigo para que todos possam ter acesso a informação.

Descrição:

Este artigo tem como objetivo explicar como funciona e como usar o Reflection nas aplicações.

Tecnologia: ASP.NET / C#

A reflexão é geralmente usada para Visualizar metadados, Associar métodos e propriedades e descobrir tipos.

Com ele você pode examinar os tipos em um assembly e interagir com eles ou instancia-los e também podemos criar tipos no momento de execução.

A principal utilidade da reflexão é criar novos tipos no momento da execução e usa-los para realizar tarefas.

No artigo "Entendendo os Atributos" usamos a reflexão para visualizar os metadados.

usamos o typeof no tipo **_Default**, o que retorna um objeto do tipo type, derivado do **MemberInfo**.

A classe Type é o coração das classes de reflexão. Ela encapsula a representação do tipo de um objeto. A classe type é o primeiro caminho para acesso aos metadados. Ela deriva do MemberInfo e encapsula informações sobre os membros de uma classe (métodos, propriedades, campos, eventos e etc.)

O **GetCustomAttribute** no objeto MemberInfo , passando o tipo do atributo que você deseja encontrar. Você receberá um array do tipo **ComentariosAttribute**

Exemplo:

```
System.Reflection.MemberInfo info = typeof(_Default);  
object[] atts = info.GetCustomAttributes(typeof(ComentariosAttribute), false);
```

Outra coisa interessante que o reflection faz é descobrir tipos

Você pode usar a reflexão para explorar e examinar o conteúdo de uma assembly. Você pode encontrar os tipos associados com um módulo; os métodos, campos , propriedades e eventos associados a um tipo, bem como as assinaturas de cada um dos métodos do tipo; a interface suportada pelo tipo e a sua classe básica.

Bom iremos carregar um assembly dinamicamente com o método estático **Assembly.Load()**. A classe Assembly encapsula o próprio assembly, para fins de reflexão. Uma assinatura para o método é o Load é: **Public static Assembly.Load(AssemblyName)**

Iremos carregar a biblioteca Mscorlib.dll ela possui as classes principais do .NET Framework:

```
Assembly a = Assembly.Load("Mscorlib");
```

Uma vez carregado o assembly, você pode chamar o **GetTypes()** para retornar um array de objetos Type. Ele representa as declarações do tipo (classes, interfaces, arrays, valores e listas):

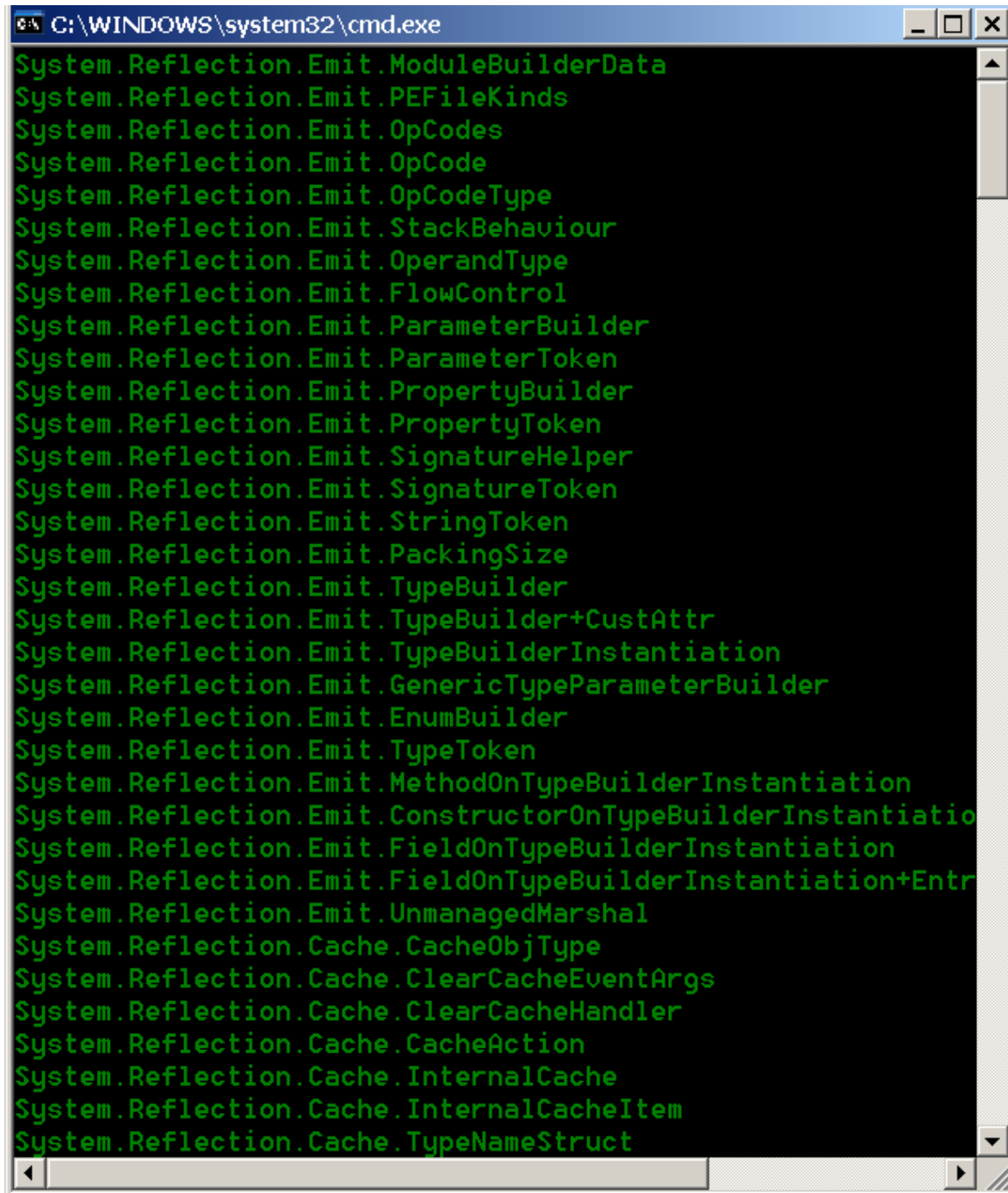
```
Type[] types = a.GetTypes();
```

O Assembly retorna um array de tipos que você pode exibir em um laço foreach, como ilustrado abaixo:

```
foreach (Type t in types) {  
    Console.WriteLine("{0}", t);  
}
```

```
Console.WriteLine("{0}", types.Length);
```

A saída para este código seria algo assim:



```
C:\WINDOWS\system32\cmd.exe
System.Reflection.Emit.ModuleBuilderData
System.Reflection.Emit.PEFileKinds
System.Reflection.Emit.OpCodes
System.Reflection.Emit.Opcode
System.Reflection.Emit.OpcodeType
System.Reflection.Emit.StackBehaviour
System.Reflection.Emit.OperandType
System.Reflection.Emit.FlowControl
System.Reflection.Emit.ParameterBuilder
System.Reflection.Emit.ParameterToken
System.Reflection.Emit.PropertyBuilder
System.Reflection.Emit.PropertyToken
System.Reflection.Emit.SignatureHelper
System.Reflection.Emit.SignatureToken
System.Reflection.Emit.StringToken
System.Reflection.Emit.PackingSize
System.Reflection.Emit.TypeBuilder
System.Reflection.Emit.TypeBuilder+CustAttr
System.Reflection.Emit.TypeBuilderInstantiation
System.Reflection.Emit.GenericTypeParameterBuilder
System.Reflection.Emit.EnumBuilder
System.Reflection.Emit.TypeToken
System.Reflection.Emit.MethodOnTypeBuilderInstantiation
System.Reflection.Emit.ConstructorOnTypeBuilderInstantiation
System.Reflection.Emit.FieldOnTypeBuilderInstantiation
System.Reflection.Emit.FieldOnTypeBuilderInstantiation+Entr
System.Reflection.Emit.UnmanagedMarshal
System.Reflection.Cache.CacheObjType
System.Reflection.Cache.ClearCacheEventArgs
System.Reflection.Cache.ClearCacheHandler
System.Reflection.Cache.CacheAction
System.Reflection.Cache.InternalCache
System.Reflection.Cache.InternalCacheItem
System.Reflection.Cache.TypeNameStruct
```

Para encontrar um membro específico podemos usar o **FindMembers**.

O FindMembers recebe 4 parâmetros.

MembersType: Um objeto MemberTypes indica o tipo do membro a ser localizado, pode ser All, Constructor, Vustom, Event, Field, Method, Nestedtype, Property e TypeInfo.

BindingFlags: Uma lista que controla a maneira como as busca serão conduzidas pela reflexão. Alguns dos valores BindingFlags são: IgnoreCase, Instance, Public, Static e outros.

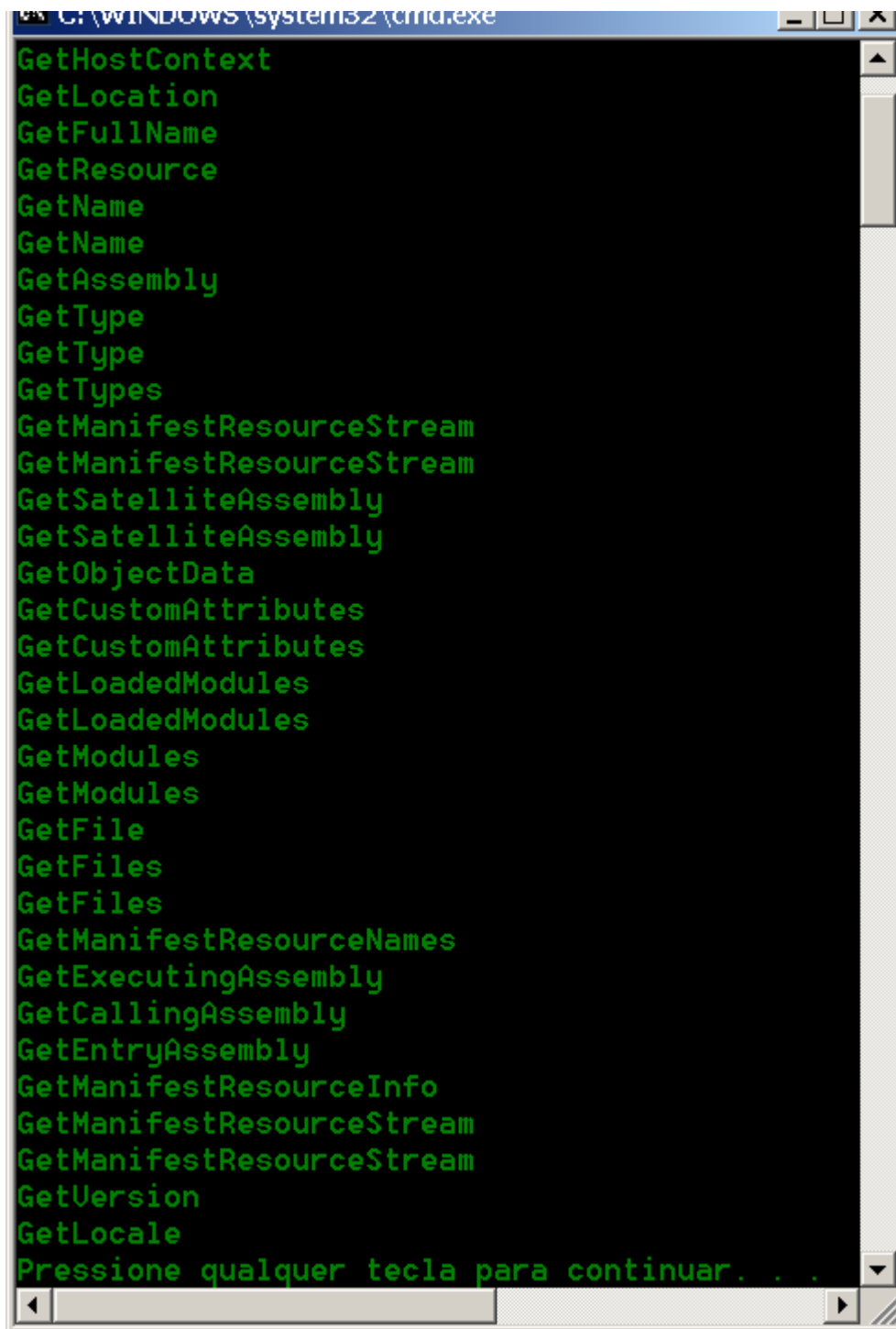
MemberFilter: Um delegado que filtra a lista dos membros no array de objetos MemberInfo.

Object : Um valor string usado pelo filtro. Nesse caso, você passa o Get* para encontrar apenas aqueles métodos que comecem por Get.

Aqui vamos encontrar um membro específico:

```
Type _type = Type.GetType("System.Reflection.Assembly");
//APENAS MEMBROS QUE SEJAM METODOS COMECANDO POR GET
MemberInfo[] _minfo = _type.FindMembers(MemberTypes.Method,
    BindingFlags.NonPublic |
    BindingFlags.Public |
    BindingFlags.Static |
    BindingFlags.Instance |
    BindingFlags.DeclaredOnly, Type.FilterName, "Get*");
foreach (MemberInfo _info in _minfo)
{
    Console.WriteLine("{0}", _info.Name);
}
```

Resultado:

A screenshot of a Windows command prompt window. The title bar at the top reads "C:\WINDOWS\system32\cmd.exe". The window has a black background with green text. A list of reflection methods is displayed, with some methods appearing twice. At the bottom, a green prompt asks the user to press a key to continue. The methods listed are: GetHostContext, GetLocation, GetFullName, GetResource, GetName (twice), GetAssembly, GetType (twice), GetTypes, GetManifestResourceStream (twice), GetSatelliteAssembly (twice), GetObjectData, GetCustomAttributes (twice), GetLoadedModules (twice), GetModules (twice), GetFile, GetFiles (twice), GetManifestResourceNames, GetExecutingAssembly, GetCallingAssembly, GetEntryAssembly, GetManifestResourceInfo, GetManifestResourceStream (twice), GetVersion, GetLocale, and a final line in Portuguese: "Pressione qualquer tecla para continuar. . . .".

```
C:\WINDOWS\system32\cmd.exe
GetHostContext
GetLocation
GetFullName
GetResource
GetName
GetName
GetAssembly
GetType
GetType
GetTypes
GetManifestResourceStream
GetManifestResourceStream
GetSatelliteAssembly
GetSatelliteAssembly
GetObjectData
GetCustomAttributes
GetCustomAttributes
GetLoadedModules
GetLoadedModules
GetModules
GetModules
GetFile
GetFiles
GetFiles
GetManifestResourceNames
GetExecutingAssembly
GetCallingAssembly
GetEntryAssembly
GetManifestResourceInfo
GetManifestResourceStream
GetManifestResourceStream
GetVersion
GetLocale
Pressione qualquer tecla para continuar. . . .
```

Espero ter deixado claro com funciona o Reflection.

Até o próximo artigo!

[]'s