9.Атрибути

Работа с метаданни в .NET

Емил Стойчев

Съдържание

- Атрибути – синтаксис и семантика
- Прилагане на атрибут
- Наименуване
- Създаване на атрибути
- Атрибута AttributeUsage
- Достъп до атрибути
- Особености

Атрибути

- Механизъм за добавяне на декларативна информация към .NET код
- Типове атрибути
  - Вградени в CLR атрибути
  - Добавени от разработчика (Custom attributes)
- По време на компилация се сереализират и се прибавят към асемблито
- Могат да бъдат манипулирани програмно

Създаване на атрибути

- .NET позволява разработка на нови атрибути
- Те трябва да са класове, наследяващи **`System.Attribute с поне един публичен конструктор`**
- Имената им трябва да завършват на Attribute
- Могат да се прилагат рестрикции върху какво могат да бъдат приложени чрез атрибута **`AttributeUsage`**

Атрибута AttributeUsage

- Определя върху какво може да се прилага даден атрибут
  - All
  - Assembly
  - Class
  - Constructor
  - Delegate
  - Enum
  - Event
  - Field
  - GenericParameter
  - Interface
  - Method
  - Module
  - Parameter
  - Property
  - ReturnValue
  - Struct

Атрибута AttributeUsage

- Определя дали е разрешено неколкократно прилагане на един и същ атрибут върху една и съща цел
- Дефинира дали атрибутите приложени върху даден клас автоматично се прилагат и върху наследниците му

```
[AttributeUsage(AttributeTargets.Method,
AllowMultiple=true, Inherited=false)]
public class MyCustomAttribute : System.Attribute
{ ... }
```

Създаване на атрибути

- Могат да има два типа параметри
  - **Позиционни** (positional) – задължителни, подават се през конструктора
  - **Именувани** (named) – опционални, дефинират се като нестатичен field или property

Създаване на атрибути

```
[AttributeUsage(AttributeTargets.Class |
AttributeTargets.Method, AllowMultiple=false,
Inherited=false)]
public class HelpAttribute : Attribute
{
    public readonly string HelpText;
    public string Author { get; set; }

    public HelpAttribute(string helpText)
    {
        this.HelpText = helpText;
    }
}
```

Прилагане на атрибут

- Имат специфичен “олекотен” синтаксис и могат да се добавят под формата на декларативни тагове в програмния код

```
[HelpAttribute("http://localhost/MyClassInfo")]
class MyClass { }
```

- Изрично задаване на цел на прилагане на атрибута

```
[assembly: AssemblyVersion("2.0.1.37")]
```

Наименуване

- Името на класовете от тип атрибут завършва на Attribute – например **`System.HelpAttribute`**
- При прилагане на атрибут може да се използва както пълното име (**`HelpAttribute`**), така и съкратена форма (**`Help`**

Достъп до атрибути

- Атрибутите могат да се достъпват и манипулират програмно
- Достъпът на атрибутите става с помощта на отразени типове – Reflection
- Reflection е техника за програмен достъп до характеристиките на .NET тип
- Класът **`System.Type`**, операторът **`typeof()`** и метода **`GetType()`** дава достъп до характеристиките на класове

Достъп до атрибути

```
static void Main()
{
    Type type = typeof(Product);
    object[] helpAttributes =
      type.GetCustomAttributes(typeof(HelpAttribute)
      ,false);
    if (helpAttributes != null &&
      helpAttributes.Length > 0)
    {
        HelpAttribute help = helpAttributes[0] as
        HelpAttribute;
        Console.WriteLine(help.HelpText);
        Console.WriteLine(help.Author);
    }
}
```

Достъп до атрибути - особености

- Вградените атрибути се достъпват чрез свойството **`Attributes`** на класа **`Type`**

```
Type type = typeof(Product);
TypeAttributes attributes  = type.Attributes;
```

- Атрибутите се създават при поискване, а не при създаване на класа

Как се съхраняват атрибутите

- Програмно прилагаме атрибут
- При компилация
  - компилаторът инстанциира атрибута
  - Инстанцията на атрибута се сериализира
  - Записва се в таблицата с мета данни на съответния тип върху който е приложен
- По време на изпълнение при поискване атрибута се десериализира

CLR атрибути

- **DebuggerDisplay** – бърз преглед на характеристики на тип при дебъг
- **DebuggerStepThrough** – инструктира дебъгера да НЕ влиза в даден метод
- **Browsable** – дефинира дали property да се показва в properties прозореца във VS
- **DefaultValue**
- **DisplayName** – под какво име да се показва в Properties

Въпроси?

Полезни линкове

- Кратко и ясно описание на атрибутите в .NET на C#
  - http://www.csharphelp.com/archives3/archive558.html
  - http://www.codeguru.com/csharp/csharp/cs_syntax/attributes/article.php/c5831/
- Тема с по-интересните и използвани атрибути в C#
  - http://stackoverflow.com/questions/144833/most-useful-attributes-in-c

Задачи

- Напишете атрибут **`NonVisualProperty`**, който да се прилага само върху свойства (properties) на даден тип. Създайте тестов клас **`Person`** с няколко свойства и приложете върху някой от тях атрибута. Предефинирайте метода **`ToString()`** на тестовия клас, така че да отпечатва само свойства които не са декорирани с **`NonVisualProperty`** атрибута. Добавете клас **`Student`**, който наследява **`Person`** без да добавя функционалност. Инстанциирайте **`Person`** и извикайте **`ToString()`**. Какъв е резултата и защо?

Задачи

- Напишете generic метод който връща типизирани атрибутите на даден клас. Реализирайте метода като extension метод на класа Type.