**[Attribute在.net编程中的应用（二）](http://www.cnblogs.com/dudu/articles/4450.html)**

.NET Framework中对Attribute的支持是一个全新的功能，这种支持来自它的Attribute类。在你的程序中适当地使用这个类，或者是灵活巧妙地利用这个类，将使你的程序获得某种在以往编程中很难做到的能力。我们来看一个例子：  
假如你是一个项目开发小组中的成员，你想要跟踪项目代码检查的信息，通常你可以把代码的检查信息保存在数据库中以便查询；或者把信息写到代码的注释里面，这样可以阅读代码的同时看到代码被检查的信息。我们知道.NET的组件是自描述的，那么是否可以让代码自己来描述它被检查的信息呢？这样我们既可以将信息和代码保存在一起，又可以通过代码的自我描述得到信息。答案就是使用Attribute.  
下面的步骤和代码告诉你怎么做：  
首先，我们创建一个自定义的Attribute,并且事先设定我们的Attribute将施加在class的元素上面以获取一个类代码的检查信息。

using System;

using System.Reflection;

[AttributeUsage(AttributeTargets.Class)] //还记得上一节的内容吗？

public class CodeReviewAttribute : System.Attribute //定义一个CodeReview的Attribute

{

private string reviewer; //代码检查人

private string date; //检查日期

private string comment; //检查结果信息

//参数构造器

public CodeReviewAttribute(string reviewer, string date)

{

this.reviewer=reviewer;

this.date=date;

}

public string Reviewer

{

get

{

return reviewer;

}

}

public string Date

{

get

{

return date;

}

}

public string Comment

{

get

{

return comment;

}

set

{

comment=value;

}

}

}

我们的自定义CodeReviewAttribute同普通的类没有区别，它从Attribute派生，同时通过AttributeUsage表示我们的Attribute仅可以施加到类元素上。

第二步就是使用我们的CodeReviewAttribute, 假如我们有一个Jack写的类MyClass，检查人Niwalker，检查日期2003年7月9日，于是我们施加Attribute如下：

[CodeReview("Niwalker","2003-7-9",Comment="Jack的代码")]

public class MyClass

{

//类的成员定义

}

当这段代码被编译的时候，编译器会调用CodeReviewAttribute的构造器并且把"Niwalker"和"2003-7-9"分别作为构造器的参数。注意到参数表中还有一个Comment属性的赋值，这是Attribute特有的方式，这里你可以设置更多的Attribute的公共属性（如果有的话），需要指出的是.NET Framework1.0允许向private的属性赋值，但在.NET Framework1.1已经不允许这样做，只能向public的属性赋值。

第三步就是取出我们需要的信息，这是通过.NET的反射来实现的，关于反射的知识，限于篇幅我不打算在这里进行说明，也许我会在以后另外写一篇介绍反射的文章。

class test

{

static void Main(string[] args)

{

System.Reflection.MemberInfo info=typeof(MyClass); //通过反射得到MyClass类的信息

//得到施加在MyClass类上的定制Attribute

CodeReviewAttribute att=

(CodeReviewAttribute)Attribute.GetCustomAttribute(info,typeof(CodeReviewAttribute));

if(att!=null)

{

Console.WriteLine("代码检查人:{0}",att.Reviewer);

Console.WriteLine("检查时间:{0}",att.Date);

Console.WriteLine("注释:{0}",att.Comment);

}

}

}

在上面这个例子中，Attribute扮演着向一个类添加额外信息的角色，它并不影响MyClass类的行为。通过这个例子，我们大致可以知道如何写一个自定义的Attribute，以及如何在应用程序使用它。下一节，我将介绍如何使用Attribute来自动生成ADO.NET的数据访问类的代码。  
（待续）  
原文: <http://www.csdn.net/Develop/Read_Article.asp?Id=19564>