# C# 反射（Reflection）

反射指程序可以访问、检测和修改它本身状态或行为的一种能力。

## 反射（Reflection）的用途

反射（Reflection）有下列用途：

* 它允许在运行时查看特性（attribute）信息。
* 它允许审查集合中的各种类型，以及实例化这些类型。
* 它允许延迟绑定的方法和属性（property）。
* 它允许在运行时创建新类型，然后使用这些类型执行一些任务。

## 查看元数据

我们已经在上面的章节中提到过，使用反射（Reflection）可以查看特性（attribute）信息。

**System.Reflection** 类的 **MemberInfo** 对象需要被初始化，用于发现与类相关的特性（attribute）。为了做到这点，您可以定义目标类的一个对象，如下：

System.Reflection.MemberInfo info = typeof(MyClass);

例：

System.Reflection.MemberInfo info = typeof(MyClass); //取得类的成员信息

object[] attributes = info.GetCustomAttributes(true); //取得特性

for (int i = 0; i < attributes.Length; i++)

{

System.Console.WriteLine(attributes[i]);

}

输出：

HelpAttribute //特性名称

例二：

Type type = typeof(Rectangle);

// 遍历 Rectangle 类的特性

foreach (Object attributes in type.GetCustomAttributes(false))

{

DeBugInfo dbi = (DeBugInfo)attributes;

if (null != dbi)

{

Console.WriteLine("Bug no: {0}", dbi.BugNo);

Console.WriteLine("Developer: {0}", dbi.Developer);

Console.WriteLine("Last Reviewed: {0}",

dbi.LastReview);

Console.WriteLine("Remarks: {0}", dbi.Message);

}

}

// 遍历方法特性

foreach (MethodInfo m in type.GetMethods())

{

foreach (Attribute a in m.GetCustomAttributes(true))

{

DeBugInfo dbi = (DeBugInfo)a;

if (null != dbi)

{

Console.WriteLine("Bug no: {0}, for Method: {1}",

dbi.BugNo, m.Name);

Console.WriteLine("Developer: {0}", dbi.Developer);

Console.WriteLine("Last Reviewed: {0}",

dbi.LastReview);

Console.WriteLine("Remarks: {0}", dbi.Message);

}

}

}