# 过滤器

**过滤器**（Filter）把附加逻辑注入到MVC框的请求处理，实现了交叉关注。所谓**交叉关注**（Cross-Cutting Concerns），是指可以用于整个应用程序，而又不适合放置在某个局部位置的功能，否则会打破关注分离模式。典型的例子有：登录、授权、缓存等等。

## 使用过滤器

如果希望动作方法只能被认证用户所使用，可以在每个动作方法中检查请求的授权状态。如在动作方法中使用Request.IsAuthenticated方法明确地检查授权：if(!Request.IsAuthenticated){如果未被授权，则…}。

但是，如果在项目中这么做，那会非常繁琐，他需要在每一个需要有授权认证的动作方法中进行判断。所以，采用过滤器才是最好的办法，如清单1：

// 说明：

// 过滤器是 .NET 的注解属性，可以把它们运用于动作方法或控制器类。

// 当被用于控制器类时，其作用效果将覆盖当前控制器中的每一个方法。

[Authorize]

public class AdminController : Controller

{

public ViewResult Index()

{

…

}

public ViewResult Edit(int productId)

{

…

}

}

.NET的注解属性，一个新鲜的事物

**注解属性**（Attribute）是派生于System.Attribute的特殊的.NET类。它们可以被附加到其他代码元素上，包括类、方法、属性以及字段等。目的是把附加信息嵌入到已编译的代码中，以便在运行时读回这些信息。

在C#中，注解属性用方括号进行附加，而且可以用已命名参数语法给它们的public属性赋值（如：[MyAttribute(SomeProperty=value)]）。在C#的编译器命名约定中，如果这些注解属性类的名称以单词Attribute结尾，则可以忽略这一部分（如，对于AuthorizeAttribute注解属性，可以写成[Authorize]）。

### 过滤器的四种基本类型

MVC 框架支持四种不同类型的过滤器。分别是：认证过滤器、动作过滤器、结果过滤器、异常过滤器，如下表：

|  |  |  |  |
| --- | --- | --- | --- |
| 过滤器类型 | 接口 | 默认实现 | 描述 |
| Authorization  (认证过滤器) | IAuthorizationFilter | AuthorizeAtrribute | 最先运行，在任何其他过滤器或动作方法之前 |
| Action  (动作过滤器) | IActionFilter | ActionFilterAtrribute | 在动作方法之前及之后运行 |
| Result  (结果过滤器) | IResultFilter | ActionFilterAtrribute | 在动作结果被执行之前和之后运行 |
| Exception  (异常过滤器) | IExceptionFilter | HandleErrorAtrribute | 仅在另一个过滤器、动作方法、或动作结果抛出异常时运行 |

在框架调用一个动作之前，会首先检测该方法的定义，以查看它是否具有这些过滤器设置。如果有，那么便会在请求管道的相应点上调用这些接口所定义的方法。当然，框架默认实现了这些接口。

注：ActionFilterAtrribute类即实现了IActionFilter，也实现了IResultFilter接口，但这是一个抽象类，要求我们必须提供一个实现。

### 将过滤器运用于控制器和动作方法

过滤器可以应用于动作方法，也可以运用于整个控制器。清单1,将Authorize过滤器运用于AdminController类，其效果与将其运用于控制器中的每一个方法相同，如清单2：

public class AdminController : Controller

{

**[Authorize]**

public ViewResult Index()

{

…

}

**[Authorize]**

public ViewResult Edit(int productId)

{

…

}

}

可以运用多个过滤器，也可以混搭它们运用的层级——即，将它们运用于整个控制器或某个动作方法。如清单3演示了三个不同过滤器的使用方式：

**[Authorize(Roles="trader")]** // applies to all actions(运用于所有动作)

public class ExampleController : Controller

{

**[ShowMessage]** // applies to just this action(仅用于本动作)

**[OutputCache(Duration=60)]** // applies to just this action(仅用于本动作)

public ViewResult Index()

{

…

}

}

注：如果为控制器定义了一个自定义基类，那么运用于基类上的任何过滤器都会影响其派生类。