# 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：

// 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