**HttpModule**

1. **什么是HttpModule？**

-> 【HttpModule】是ASP.NET中的核心对象，我们所俗称的管道则由【HttpModule】所构成

-> 在ASP.NET的管道中对应着一个个【HttpModule】，而不同的【HttpModule】中也有其相应的Event Handler，我们可以通过这一机制在不同的管道流程当中注入我们所需要的Event Handler，简而言之，它是一个基于处理器的事件

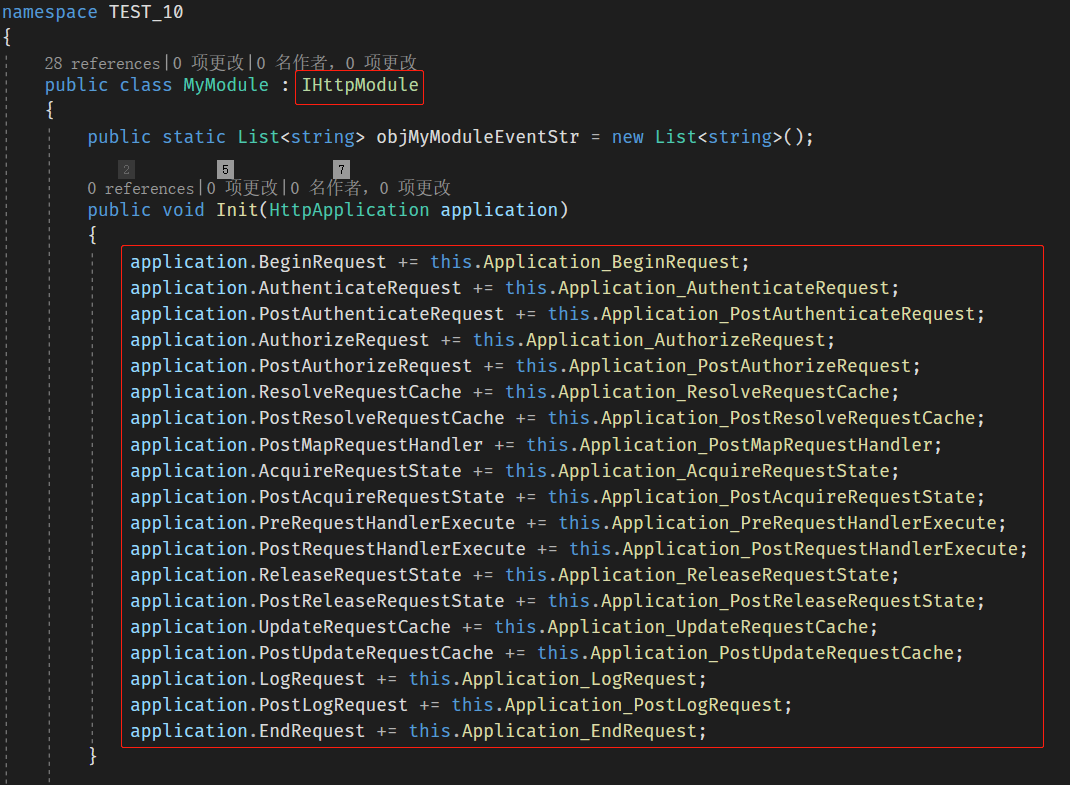
-> 【HttpModule】在ASP.NET初始化的过程当中是通过配置文件上<modules>节点来决定的，像ASP.NET默认的配置文件已经给定了默认的【HttpModule】的Mapping

1. **如何自定义HttpModule**

-> 在初始化管道的过程当中，ASP.NET除了会读取其默认配置下的【HttpModule】的Mapping，还会读取当前资源请求项目工程下的【web.config】中的<modules>节点，下面我们就在WEB项目工程文档下的【web.config】中添加一个<modules>节点



-> 新建一个类并实现【IHttpModule】接口，在ASP.NET初始化【HttpModule】的过程当中，会根据上面说明的配置文件的映射关系，去创建不同的【HttpModule】，并且循环去调用【HttpModule】的【init】函数，如下图，我们通过自定义的【HttpModule】又为不同的【HttpModule】注入我们自定义的Event Handler，至此，自定义【HttpModule】创建完毕

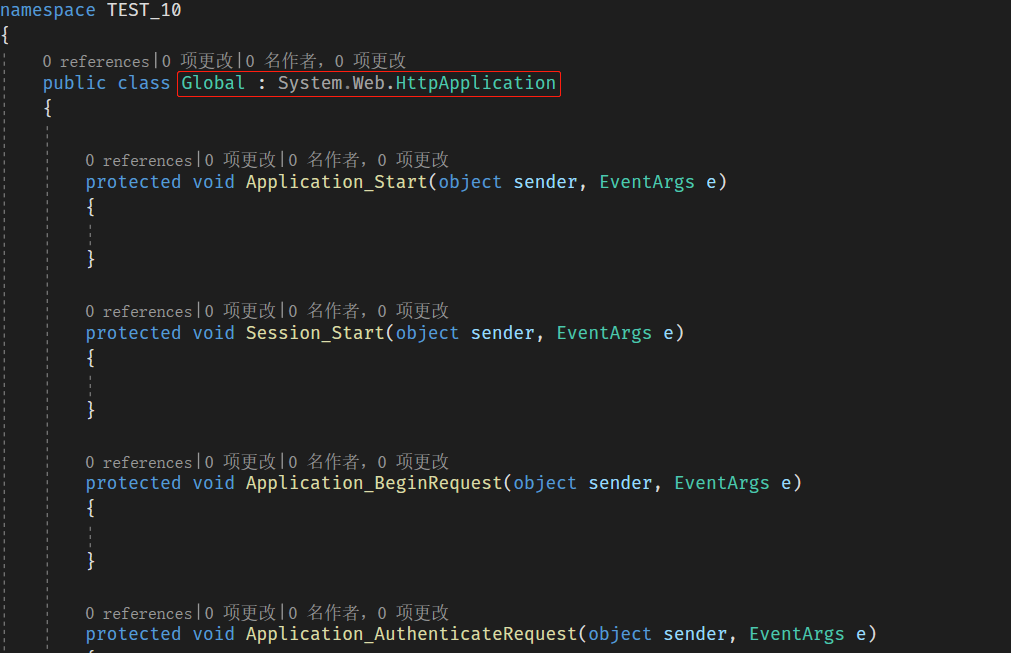


1. **全局处理程序：Global.asax**

-> 在第二篇节点中介绍了【HttpModule】，并且通过自定义的【HttpModule】为ASP.NET管道中的事件注入自己的Event Handler，而更便捷的办法是通过Global.asax来实现，因为管道事件是基于【HttpApplication】的，并且【HttpApplication】的创建是基于Global的，在Global中所声明的、符合管道事件签名的函数，ASP.NET都会在创建【HttpApplication】的过程当中把这些函数自动注入到相应的管道事件中去

-> 在底层事件注册的顺序中，是先注册【HttpModule】中的Event Handler再去注册基于Global.asax中的Event Handler，所以事件的触发顺序也是先触发【HttpModule】中的Event Handler然后再去触发Global.asax中的Event Handler

-> Global.asax文件必须放在网站文件的根目录下



-> 关于Global.asax文件的使用我们需要重点关注这三个Event Handler

1. Application\_Start：当【Application Pool】开始启动的时候会调用，也就是说所发布到服务器上的这个Web Application只有在第一次启动的时候才会被调用



1. Application\_End：当【Application Pool】被关闭的时候才会调用



1. Application\_Error：当【Application Pool】中出现了未能正确捕获到的异常时会调用，可以通过它来进行全局的异常捕获



**HttpHandler**

1. **什么是HttpHandler？**

-> 【HttpHandler】其实我们都不算陌生，像一般处理程序 \*.ashx、WebForm Page和MvcHandler它们都是属于【HttpHandler】的范畴，细心观察发现，一般处理程序、Page和MvcHandler其实都是继承且实现于【IHttpHandler】接口，这也是作为【HttpHandler】的一个前提条件

-> 【HttpHandler】其是对Resource Request处理的Handler，不同的资源后缀都有着他对应的【HttpHandler】对该资源访问的后续处理，其映射关系是通过配置文件中的<handlers>节点来决定的，像ASP.NET默认的配置文件就有其着默认的<handlers>节点，在管道事件执行到第7个管道的时候，HttpRuntime会根据相应的HttpModule的注册去构建一个属于当前HttpModule的HttpHandler，在管道事件执行到第8个事件的时候，除了会根据ASP.NET默认配置文件中的<handlers>的Mapping，还会根据当前Resource Request WEB工程项目文档下的【web.config】中的<handlers>的Mapping去为不同的Resource Request 匹配在第7个事件中所创建的HttpHandler

1. **如何自定义HttpHandler**

-> 上面我们说到，【HttpHandler】对应着对不同Resource Request的处理，比如说一般处理程序 \*.ashx则对应的是一般处理程序的【HttpHandler】，那么下面我们就为一个图片Resource Request构建一个其所对应的【HttpHandler】

-> WEB项目工程文档下的【web.config】中添加一个<handlers>节点



-> 新建一个类并实现【IHttpHandler】接口，在管道事件执行到第十一至十二个事件之中，就会调用刚刚上问所说到的在第7个事件中所创建的【HttpHandler】中的【ProcessRequest】函数，至此我们就能够为当前Resource Request进行一些我们所针对性的处理

