.net column

HTTP modules

Eén van de nuttige features van ASP.NET is de mogelijkheid om de HTTP pipeline uit te breiden met zogenaamde *HTTP modules*. Deze zijn op het niveau van de gehele applicatie in staat om het verstuurde request te onderscheppen, zodat bijvoorbeeld een authenticatie controle kan worden uitgevoerd.

Er bestaat vaak de behoefte om binnen een ASP.NET applicatie te controleren of een gebruiker geauthentiseerd is, zonder gebruik te maken van Windows- of Forms-authenticatie. We spreken dan over *custom authenticatie*. Het is natuurlijk niet wenselijk om in alle afzonderlijke pagina's een authenticatie controle uit te voeren. In plaats hiervan kan er gebruik gemaakt worden van een custom *HTTP module*, die op het niveau van de gehele applicatie in staat is om het verstuurde request te onderscheppen, zodat bijvoorbeeld een authenticatie controle kan worden uitgevoerd.

Om te begrijpen wat de rol van HTTP modules zijn binnen ASP.NET, moeten we eerst weten hoe HTTP pipelines werken. Als een request binnen komt over poort 80 (dit is de standaard http poort) wordt het verzoek door een aantal stadia geleid, die tezamen de pipeline vormen, voordat het aangeboden wordt aan de applicatie.

De Microsoft Internet Information Services (IIS) is de eerste deelnemer in de keten. IIS zorgt ervoor dat het verzoek wordt gerouterd naar de ASP.NET runtime op basis van de ASP.NET file extensie en gebruikt hiervoor de *ASPNET_ISAPI.dll*, die meegeleverd wordt met ASP.NET.

Het is de verantwoordelijkheid van de ASPNET_ISAPI.dll om het verzoek door te sturen aan het ASP.NET worker proces (ASPNET_WP.exe). Vanaf dat moment wordt het verzoek 'ingepakt' in een instantie van de *HttpContext* klasse. De HttpContext klasse biedt toegang tot onder meer de Request- en Response-stream en bevat tevens security informatie.

In de volgende stap wordt het verzoek doorgestuurd naar een instantie van de HttpApplication klasse. Dit stadium wordt gebruikt voor het definiëren van methoden, data en events, die op het niveau van de gehele applicatie zichtbaar zijn. Hierna wordt het request verder doorgestuurd aan één of meerdere HttpModule objecten. Er zijn een aantal systeem-level HTTP modules, die standaard functionaliteit bevatten zoals: authenticatie en output caching.

Het laatste onderdeel in de keten is de HttpHandler, die geïmplementeerd wordt door de System.Web.UI.Page klasse. (De ASPX pagina zelf leidt af van de Page klasse)

Een HTTP module is simpelweg een klasse die de IHttpModule interface implementeert.

```
public interface IHttpModule
{
void Dispose();
void Init(HttpApplication
context);
}
```

Als een HttpModule wordt toegevoegd aan de pipeline (via web.config file), dan roept de ASP.NET runtime de *Init* en *Dispose* functies aan. Via deze functies kan de module zich abonneren op diverse events die door het HttpApplication object worden aangeboden, zoals het begin van het request, het einde van het request, een request voor authenticatie, enzovoort.

Eén van de redenen om een HTTP request te onderscheppen via een HTTP module, is om het request vroegtijdig te beëindigen als er wat mis gaat. Als bijvoorbeeld de authenticatie van de gebruiker zelf wordt geregeld en de meegestuurde credentials kloppen niet, dan moet het request worden gestopt. De HttpApplication klasse heeft een functie *CompleteRequest*, waarmee het request wordt beëindigd. Tevens kunnen de *StatusCode* en *StatusDescription* property's van het context object worden gezet om de client te laten weten wat er mis is gegaan.

Het concept van de HTTP pipeline maakt ASP.NET bijzonder uitbreidbaar. Eén van de eenvoudigste manieren om pre- en post processing van het request mogelijk te maken is door het toevoegen van een custom HttpModule. Een HttpModule is eenvoudig te schrijven en implementeert de IHttpModule interface. Via de web.config file worden de HttpModules toegevoegd aan de HTTP pipeline van de ASP.NET applicatie.

Xander Buffart is IT Architect bij Info Support.