Sander Hoogendoorn

Sander Hoogendoorn
www.sanderhoogendoorn.com

Begin november was ik in Barcelona. Behalve dat daar de Microsoft TechEd plaatsvond, is Barcelona natuurlijk op zich al een bezienswaardigheid. Een van mijn favoriete plekken in deze metropool is de Sagrada Familia, de beroemde tempel van Antoni Gaudi. Veelvuldig gebruikt als metafoor voor de kolossale en nooit gereedkomende softwaredevelopmentprojecten waar vooral overheden en financiële instellingen het patent op lijken te hebben. Binnenin de donkere krochten van de Sagrada vinden echter nog andere, veel interessantere processen plaats.

Mozaïek

Nog afgezien van de complete modellen (!) in de kelder, kun je op de begane grond noeste arbeiders aanschouwen die zich twee man sterk, met een peuk in het kanaal, concentreren op minutieuze activiteiten: het maken van een mozaïek uit heel kleine tegeltjes. De eerste bouwvakker tekent het tegeltje af, de tweede snijdt het en geeft het terug aan de eerste die het op de tafel inpast. Prachtig. Ik vermoed dat beide bouwvakkers na een dag werk heerlijk ontspannen met de metro huiswaarts keren, een gelukzalige blik in de ogen. Mooi werk, zonder ook maar enig zicht op de voortgang van het totale project. Een betere analogie voor developers die oplossinkjes vinden voor de kleine uitdaginkjes van alledag is er niet. Herkenbaar toch?

Zo wilde ik eerder deze week dynamisch een aantal linkbuttons aanmaken op een ASP.NET webpagina, en vervolgens het **Click** event van deze buttons laten afhandelen door de pagina waar ze op terecht komen. Dat lijkt voor de hand liggend, maar je moet er toch nog wel wat voor uitzoeken. Een mozaïekje.
Stap een is een makkie. Creëer eerst de button die je wilt plaatsen. Liefst wel vanuit het **OnInit** event!

```
LinkButton button = new LinkButton();
```

Nu is het zaak de EventInfo van het Click event op te sporen. Deze is nodig om later de delegate die het event gaat afvangen aan toe te voegen. Dus

```
Type buttontype = button.GetType();
EventInfo clickeventinfo = buttontype.GetEvent("Click");
```

De lastigste stap in dit mozaïek is het creëeren van de delegate. De makkelijke manier is om dit te doen aan de hand van de naam van de methode op de webpagina die het event moet afhandelen, hier **ButtonClick**. Originele naam. Mijn creativiteit laat me soms in de steek. Bovendien moet je weten welk type delegate moet worden aangemaakt. Dat zoek ik dan ook maar uit.

```
Type delegatetype = clickeventinfo.EventHandlerType;
Delegate delegateinstance = Delegate.
CreateDelegate(delegatetype, ph.Page, handlername);
```

Weer een tegeltje verder. Ik heb een delegate. Nu deze delegate nog meegeven aan het event, zodat, wanneer ik op de button klik, de methode op de pagina ook getriggerd wordt. Dit is een listige. Gelukkig kent de **EventInfo** klasse een methode om te bepalen hoe een delegate aan het **Click** event wordt toegevoegd. De **MethodInfo** voor deze methode haal ik op en ik voer de methode uit via **Invoke**(). Daarbij geef ik de delegate mee als parameter.

```
MethodInfo mi = clickeventinfo.GetAddMethod();
Object[] args = { delegateinstance };

mi.Invoke(button, args);
```

Nu is het werk bijna gedaan. Nog even de button toevoegen aan de controls op de webpagina (of in mijn specifieke geval aan de controls van de placeholder **ph** op het user control op de webpagina) en klaar is Kees.

```
ph.Controls.Add(button);
```

Ik voel me als een bouwvakker in de Sagrada Familia. Moe maar voldaan. Een prachtig mozaiekje als piepklein onderdeeltje van de applicatie waar ik aan werk. Wat is code schrijven toch een mooi vak. Met een gelukzalige glimlach stap ik mijn auto en sluit aan in de file.