Mark Blomsma is softwarearchitect en medeoprichter van Omnext.NET

.NET Framework 2.0: ClickOnce deployment

EENVOUDIG DOWNLOADEN EN INSTALLEREN VAN WINDOWS-APPLICATIES

ClickOnce is de codenaam voor technologie waarmee je Windows-applicaties eenvoudig kunt downloaden en installeren via het netwerk. Deze technologie is onderdeel van het .NET Framework versie 2.0. Visual Studio 2005, waarschijnlijk beter bekend als 'Whidbey', biedt tooling om gebruik te maken van deze nieuwe frameworktechnologie. De doelstelling van de technologie is om het draaien van een Windows-applicatie net zo makkelijk te maken als het klikken van een link op internet.

Een belangrijke reden dat webapplicaties zo populair zijn geworden, is de eenvoud van deployment. Zet een nieuwe versie op de server et voilà: alle gebruikers beschikken over een nieuwe versie. ClickOnce wil het uitrollen van applicaties net zo makkelijk maken als het uitrollen van een webapplicatie.

Alles kan met ClickOnce

Toch heeft ook ClickOnce zijn beperkingen. De assemblies die worden uitgerold met ClickOnce zijn niet voorzien van aparte installer-classes. Applicaties die drivers willen installeren of allerlei COM+-instellingen moeten uitvoeren op de pc waarop ze geïnstalleerd worden, hebben het dus moeilijk. Hiervoor is ClickOnce minder geschikt. De nieuwe generatie smart clients die gebruikmaakt van webservices, maakt naar verwachting echter heel goed gebruik van deze deployment-technologie.

In tabel 1 wordt in tabelvorm weergegeven wanneer ClickOnce wel en wanneer ClickOnce niet geschikt is voor een applicatie.

Kracht in eenvoud

Dus hoe werkt ClickOnce? De basis is eenvoudig: de applicatie wordt op een webserver gezet, samen met een deploymentmanifest. De gebruiker installeert de applicatie vanaf de webserver en kan aan de slag. De volgende keer dat de applicatie gestart wordt, controleert het framework of een nieuwere versie van de applicatie aanwezig is. Als dat het geval is wordt de nieuwere versie gedownload om vervolgens automatisch te worden gestart. Vervolgens zijn er nog legio mogelijkheden over om de deployment af te stemmen op eigen wensen. We lopen aan de hand van een voorbeeld door de werking van ClickOnce.

Stap 1. Maak een Windows-applicatie

ClickOnce werkt alleen voor Windows-applicaties. Assemblies die nodig zijn om de Windows-applicatie te draaien, worden ook via ClickOnce gedeployed, maar het vertrekpunt is een Windows-applicatie. In mijn voorbeeldapplicatie heb ik een scherm gemaakt om het een en ander aan versie-informatie te laten zien.

Stap 2. Publish properties

Klik met de rechtermuisknop op het project en kies voor properties. In het scherm dat nu getoond wordt kiezen we voor het tabblad 'Publish'. Afbeelding 1 geeft aan welke eigenschappen van het project nu kunnen worden gewijzigd.

Op het tabblad zie we de instellingen zoals die actueel zijn voor het huidige project. Linksonder staat het versienummer van de deployment. De deployment krijgt met ClickOnce een versienummer dat losstaat van de versienummers van assemblies die in de applicatie zitten.

	ClickOnce	Windows Installer
Locatie waar de applicatie wordt geïnstalleerd	In de ClickOnce application cache	In de Program Files directory
Installeren van multiple users	Nee	Ja
Installeren van shared files	Nee	Ja
Installeren drivers	Nee	Ja (met eigen uitbreiding)
Installeren in de Global Assembly Cache	Nee	Ja
Applicatie toevoegen aan de Startup-groep	Nee	Ja
Applicatie toevoegen aan het Favorites-menu	Nee	Ja
Registreren van bestandstype	Nee	Ja
Toegang tot de registry	Nee (alleen toegang tot HKLM bij Full Trust Permission)	Ja
Binair patchen van bestanden	Ja	Nee
'On demand' installeren van assemblies	Ja	Nee

Tabel 1. Wanneer is ClickOnce handig?

Stap 3. Publish de applicatie
We gaan onze applicatie nu 'publishen'. Hiervoor gebruiken we de Publish Wizard. Deze kan via het eerder getoonde Publish-propertyscherm gestart worden, of via de menukeuze 'Publish'. Afbeelding 2 toont het eerste scherm van de wizard. Hier kunnen we aangeven of we naar een webserver of fileshare willen 'publishen'.

Ik heb de applicatie getest met zowel webserver- als fileshare-deployment en beide werken prima. Het enige verschil is dat de deployment wizard bij het 'publishen' naar een webserver ook een installatiepagina genereert, terwijl bij fileshare-deployment alleen een snelkoppeling wordt aangemaakt.

Stap 4. Klik op Next >
Afbeelding 3 laat zien dat we in het volgende scherm van de wizard kunnen aangeven of de applicatie alleen online of ook offline beschikbaar moet zijn. Let op! Het bepalen of de applicatie online of offline is, wordt gedaan door te kijken of de deployment-server beschikbaar is. Dus als iemand een fat client uitrolt die zelf connect naar een database zal men waarschijnlijk niet afhankelijk willen zijn van uw deployment-server. Als een developer een smart client ontwikkelt die alleen webservices gebruikt, geldt eigenlijk hetzelfde. Denk hier dus goed over na! We kiezen ervoor om de applicatie zowel online als offline beschikbaar te stellen.

Stap 5. Klik opnieuw op Next >
In de laatste stap van de wizard (afbeelding 4) moeten we aangeven welke strong name key toegepast moet worden. Let op! Dit moet dezelfde strong name key zijn als waarmee de assembly is gesigned. Anders treedt bij het opstarten van de applicatie een fout op, speciaal in een *System.Deployment.SubscriptionStateException* met melding 'The deployment identity does not match the subscription.'.

Stap 6. Finish
We drukken nu op Finish (of op 'Next' voor een overzicht van de gedane instellingen) en de applicatie wordt nu gebuild, gepackaged en naar de website gepusht. Visual Studio 2005 opent vervolgens de browser en surft netjes naar de gegenereerde installatiepagina; zie afbeelding 5.

```
// using System.Deployment

ApplicationDeployment updater = ApplicationDeployment.CurrentDeployment;

Version runningVersion = updater.RunningVersion;

this.lblRunningVersion.Text = runningVersion.ToString();

UpdateCheckInfo info = updater.GetUpdateCheckInfo();
if ( info != null )
{
   Version availableVersion = info.AvailableVersion;
   if ( availableVersion != null )
   {
   this.lblAvailableVersion.Text = availableVersion.ToString();
   }
   else
   {
   this.lblAvailableVersion.Text = "Versie onbekend.";
   }
}
else
{
   this.lblAvailableVersion.Text = "Onbekend.";
}
```

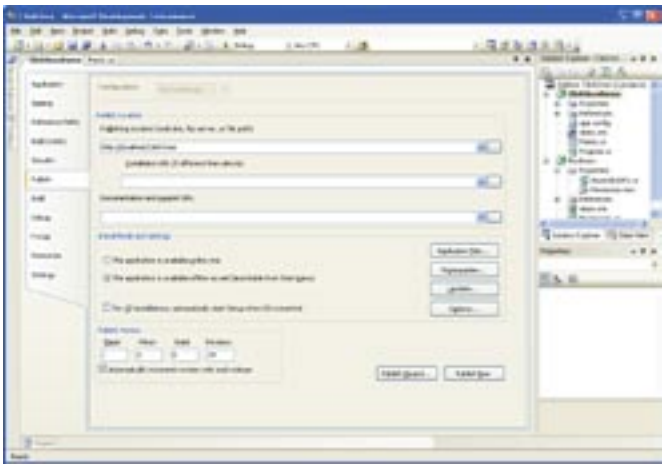
Codevoorbeeld 1.
Uitlezen van RunningVersion en AvailableVersion

Even voor de duidelijkheid: de naam van mijn applicatie was 'ClickOnceDemo'. Daarom keert deze naam dus terug op deze pagina's. Door op 'Install ClickOnceDemo' te klikken wordt de applicatie geïnstalleerd op mijn lokale machine. De applicatie wordt in de ClickOnce-applicatiecache geplaatst. Dit is een cache die, vanwege onder andere security-redenen, niet gedeeld wordt met andere gebruikers. Gezien de omvang van de huidige gemiddelde harddisk zal ook de ruimte die de cache in beslag neemt nauwelijks meer een probleem vormen. Na de installatie is er een menu en snelkoppeling aangemaakt in het Start-menu en kunnen we de applicatie opstarten.

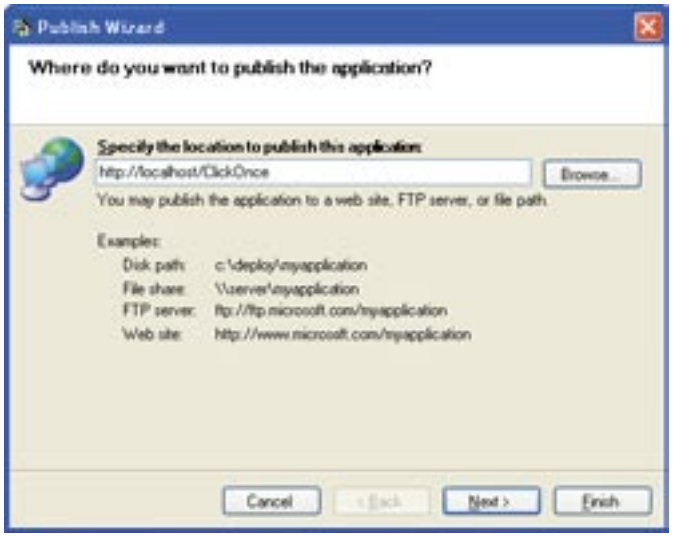
Als we stap 3 opnieuw uitvoeren, ontstaat een nieuwe versie van de deployment. Op het moment dat we opnieuw de applicatie opstarten, detecteert het framework dat er een nieuwe versie is. De vraag verschijnt of we willen upgraden. Is het antwoord 'ja', dan wordt de nieuwe versie automatisch gedownload. Overigens is het mogelijk om in de publish-properties aan te geven dat we willen forceren dat de nieuwe versie altijd gedownload wordt. Dit kan handig zijn als we een critical update willen doorvoeren en absoluut niet willen dat er nog oude versies van de applicatie gebruikt worden. Deze mogelijkheid is op dit moment nog niet voor 100% zeker. De kans bestaat dat mensen de applicatie wel gestart, maar nog niet hebben afgesloten. Diverse blogs op Internet geven aan dat er nog gewerkt wordt aan het dichten van dit gat, maar met de huidige deployment-API is het ook mogelijk om dit zelf te programmeren.

Programmeren met de ClickOnce API

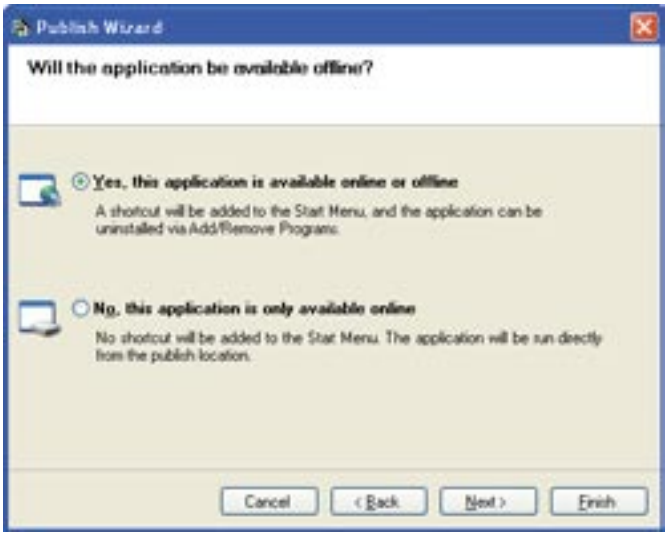
Bij het installeren van .NET Framework 2.0 is er een nieuwe assembly in de Global Assembly Cache (GAC) geplaatst, en wel



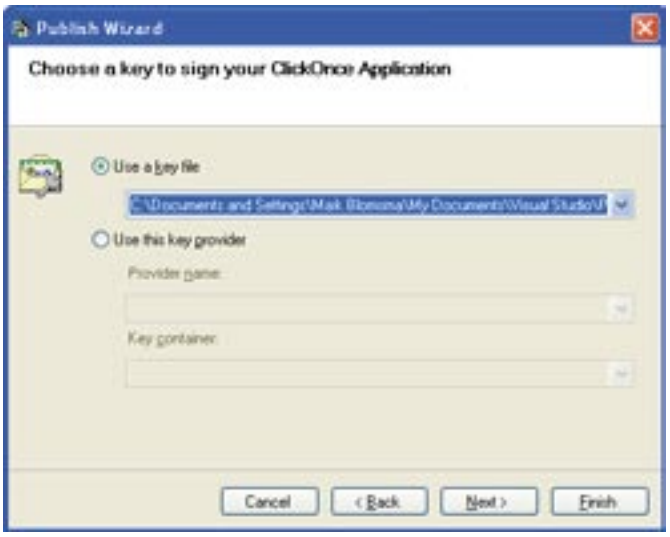
Afbeelding 1. Publish properties



Afbeelding 2. Kies hoe u wilt 'publishen'



Afbeelding 3. Online of offline?



Afbeelding 4. Een ClickOnce-applicatie moet verplicht een strong name hebben

System.Deployment.dll. Door vanuit het project een referentie te leggen naar deze assembly kunnen we gebruikmaken van de ClickOnce API. Codevoorbeeld 1 laat zien hoe we met het CurrentDeployment de objectinformatie kunnen opvragen van de huidige versie van de applicatie, de runningVersion en de op de deploymentserver aanwezige 'availableVersion'.

Met een background thread en een timer is het niet ingewikkeld om zelf periodiek te controleren of er een nieuwe versie van de applicatie op de deploymentserver staat. Zodra deze aanwezig is, is het mogelijk de gebruiker een boodschap te laten zien waarin staat

```
private void button2_Click(object sender, EventArgs e)
{
   ApplicationDeployment updater = ApplicationDeployment.CurrentDeployment;
   updater.UpdateCompleted +=
      new AsyncCompletedEventHandler( OnUpdateComplete );
   updater.Update();
}

public void OnUpdateComplete( object sender, EventArgs e )
{
   MessageBox.Show( "Nieuwe versie downloaded. De applicatie
      wordt opnieuw gestart." );
   System.Windows.Forms.Application.Restart();
}
```

Codevoorbeeld 2.
Een update downloaden en de applicatie herstarten

Afbeelding 5. De standaard installatiepagina

dat hij de applicatie moet upgraden. Nog mooier is het natuurlijk om dit in de achtergrond al voor te bereiden. In codevoorbeeld 2 zien we hoe dit met de framework-classes in de System.Deployment-namespace eenvoudig te realiseren is.

Het *updater.Update()*-statement zorgt ervoor dat er asynchroon een download van de laatste versie van de applicatie plaatsvindt. In de *OnUpdateComplete*-methode wordt na het ophalen van de nieuwe versie een messagebox getoond, waarna met de *Application.Restart()*-methode de applicatie herstart wordt. Deze laatste methode is overigens ook nieuw in .NET 2.0. Uiteraard is de code zoals die hier getoond wordt nog niet erg gebruikersvriendelijk, maar het laat wel zien dat met slechts een paar regels code een enorm complex probleem opgelost kan worden.

Tot slot

Dit artikel is gebaseerd op Visual Studio 2005 Bèta 1, Community Refresh. In deze versie zitten op het gebied van ClickOnce nog wel een paar bugs, en is niet alles af. Verder is er een aantal weblogs op internet dat doet vermoeden dat er nog hard gesleuteld wordt aan het nog vollediger maken van de ClickOnce-functionaliteit; functionaliteit die op basis van bèta 1 zeker niet misselijk is. Helaas is ClickOnce nog niet beschikbaar voor het .NET Compact Framework, maar wat niet is kan nog komen!

Mark Blomsma

is medeoprichter van Omnext.NET bv in Veenendaal (www.omnext.net). Hij is gespecialiseerd softwarearchitect voor .NET en heeft de Most Valuable Professional Award ontvangen voor het delen van zijn kennis en zijn inzet voor het C# Developer Network (www.sdgn.nl)

Nuttige internetadressen

http://lab.msdn.microsoft.com/vs2005/
http://www.windowsforms.net
MSDN TV: http://msdn.microsoft.com/msdntv/episode.aspx?xml=episodes/en/20040108ClickOnceJC/manifest.xml
http://msdn.microsoft.com/msdnmag/issues/04/05/ClickOnce/default.aspx
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwinforms/html/ClickOnce.asp