Max Remkes is programmeur bij Avanade

Introductie Silverlight

DE GRAFISCHE KRACHT VAN WPF IN DE BROWSER

Microsoft Silverlight is een compacte .NET-plugin voor browsers die voorheen Windows Presentation Foundation/Everywhere (WPF/e) heette. Deze techniek zorgt dat ontwikkelaars en designers samen interactieve webapplicaties kunnen ontwikkelen. Designers kunnen Expression Blend gebruiken om hun ontwerpen te maken. Hetzelfde project kan in Visual Studio weer geopend worden en daar kunnen ontwikkelaars aan de functionaliteit van de applicatie werken. Silverlight zorgt voor een rijke gebruikerservaring door animatie, video en vectorafbeeldingen te integreren in applicaties.

Bij het maken van interfaces voor webapplicaties zijn de ontwikkelaar en de designer beperkt in hun mogelijkheden. Html is immers gemaakt om gestructureerd tekst en data weer te kunnen geven. Het is lastig om de complexe ontwerpen van huidige sites uit te werken met de bekende taal. Webapplicaties zouden meer gebruik kunnen maken van de mogelijkheden van client-computers om de ervaring van de gebruiker te optimaliseren.

Rich Interactive Applications

Met deze implementatie van het .NET Framework is het mogelijk in verschillende browsers (Internet Explorer, Firefox, Netscape, Safari) op verschillende platformen (Windows, Mac) interactieve applicaties te maken voor het web. Deze applicaties worden ook wel Rich Interactive Applications genoemd. Bij het gebruik van deze applicaties krijgen users reactie op hun acties zonder een nieuwe pagina te openen.
Een Silverlight-applicatie moet er in iedere browser hetzelfde uit zien. Grafische objecten worden op het scherm getoond en gebruikers kunnen die aanklikken, slepen, manipuleren en draaien. Deze mogelijkheden kun je gebruiken om video, animaties en spelletjes te maken. Kortom, voldoende mogelijkheden om applicaties te maken die er mooi uitzien. Om de in Silverlight ontwikkelde applicaties te kunnen gebruiken, moet de gebruiker een klein installatiebestand van enkele MB's downloaden.

Afbeelding 1. De workspace van Expression Blend

Benodigdheden

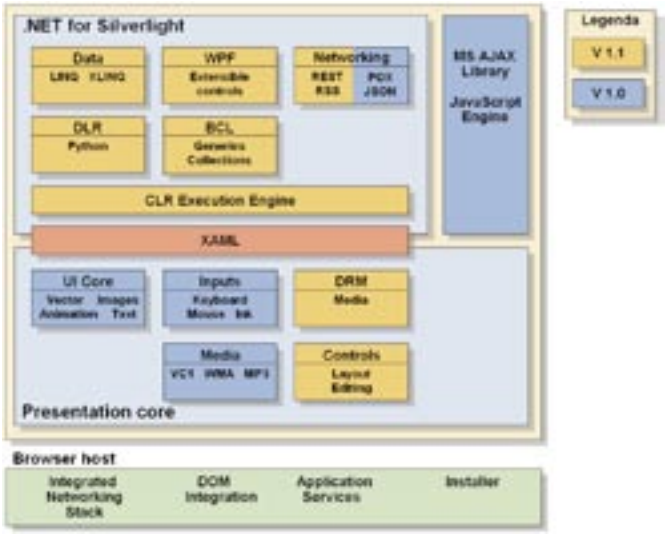
Om een Silverlight-applicatie te maken, heb je de volgende zaken nodig:

- Een runtime-versie van Silverlight
- Microsoft Visual Studio 2008
  Met een work-around ook mogelijk in Visual Studio 2005
- Microsoft Silverlight Tools for Visual Studio

Deze software met verdere instructies is te verkrijgen via de Microsoft-site van Silverlight, zie de link aan het einde van dit artikel. De runtime is dezelfde als die gebruikers downloaden om Silverlight-applicaties te zien. De Microsoft Silverlight Tools zorgen onder andere voor de mogelijkheid om in Visual Studio een Silverlight-project aan te maken en te ontwikkelen.

Expression Blend

Microsoft heeft voor de designer vier nieuwe tools uitgebracht onder de naam Expression. Deze tools zijn Expression Design, Expression Web, Expression Media en Expression Blend. Met Design is het mogelijk grafische ontwerpen te maken. Deze ont-



Afbeelding 2. Het Silverlight-platform

```
<Canvas x:Name="parentCanvas"
   xmlns="http://schemas.microsoft.com/client/2007"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   x:Class="AvanadeSilverlight.Page;
     assembly=ClientBin/AvanadeSilverlight.dll"
   Loaded="Page_Loaded"
   Width="200"
   Height="200"
   Background="White">
   <TextBlock x:Name="tbSilverlight" Text="Silverlight"
     Canvas.Left="50" Canvas.Top="8"
     FontFamily="Gautami" FontSize="24" />
</Canvas>
```

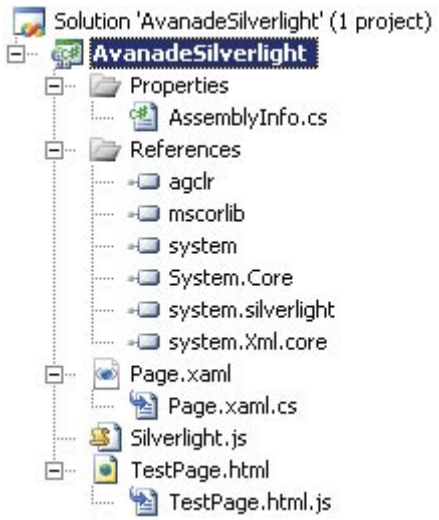
Codevoorbeeld 1.

werpen kunnen als website uitgewerkt worden in Expression Web. Expression Media wordt gebruikt om jouw verzameling video's en afbeeldingen te organiseren. Deze kunnen dan ook bewerkt en geëxporteerd worden in vele videoformaten. De tool Expression Blend, vanaf versie 2, is het interessantst voor designers die met Silverlight aan de slag willen gaan. Met deze tool worden interfaces van Silverlight- applicaties ontworpen. Een trialversie van het programma is beschikbaar op de site van Expression, waarvan de link onderaan het artikel staat.

Als je het programma opent, is de workspace van Expression Blend te zien (afbeelding 1). Aan de linkerkant zijn de tools te vinden om mee te tekenen. Dit gebeurt in het midden van het scherm, het canvas, wat in de XAML-code (eXtensible Application Markup Language) het parentCanvas is. Je kunt daarbij altijd tussen de designview en de XAML-code schakelen. Aan de rechterkant heb je een window waar verschillende properties van de getekende objecten aangepast kunnen worden. Ook is een tree te zien van het project, zoals we dat kennen van de Solution Explorer in Visual Studio. Sterker nog, het project kan direct geopend worden in de IDE, zodat een programmeur gelijk aan de slag kan. Dit is een belangrijke feature van dit programma: designers en ontwikkelaars werken letterlijk in hetzelfde project.

Architectuur

Het Silverlight-platform (afbeelding 2) bestaat onder andere uit presentatiecomponenten en -services. Deze onderdelen zijn vooral gericht op het maken van de gebruiksvriendelijke presentatie van de applicatie. Denk hierbij aan vector-graphics, tekst, animatie en afbeeldingen, audio en video. Een belangrijk onderdeel is het gebruik van XAML. Deze taal wordt gebruikt voor de opmaak van Silverlight-applicaties en voor Windows Presentation Foundation (WPF) applicaties in het .NET Framework 3.0.



Afbeelding 3. De namespaces die in het Silverlight-project worden gebruikt

```
<Rectangle Fill="Red" Stroke="Black" Width="50" Height="50"
   Canvas.Left="25" Canvas.Top="50" x:Name="rectangle"/>
<Path Stretch="Fill" Stroke="Green" StrokeThickness="2"
   Width="100" Height="80" Canvas.Left="50"
   Canvas.Top="80" Data="M41,149 L141,71" x:Name="line"/>

<Canvas Width="50" Height="50" Canvas.Left="120"
   Canvas.Top="125" RenderTransformOrigin="0.5,0.5"
   x:Name="cvEllipse">
   <Canvas.RenderTransform>
     <TransformGroup>
       <RotateTransform Angle="0"/>
     </TransformGroup>
   </Canvas.RenderTransform>

   <Ellipse Stroke="Black" Width="50" Height="50"
     x:Name="ellipse" Fill="Blue" />
   <TextBlock Width="20" Height="20" Text="T"
     Canvas.Left="20" Canvas.Top="15" />
</Canvas>
```

Codevoorbeeld 2.

Een ander onderdeel van het platform is een compacte versie van het .NET Framework. Hierin is naast de base class library van .NET ook WPF opgenomen. Je zou dus de objecten die je in een ASP.NET-applicatie hebt gemaakt, opnieuw kunnen gebruiken in Silverlight. De common language runtime (CLR) zorgt voor memory management, garbage collection en exception handling. Bovendien is er ook een Dynamic Language Runtime (DLR) beschikbaar die het mogelijk maakt scripttalen als javascript of IronPython te gebruiken. Deze talen worden dynamisch genoemd, omdat aanpassingen aan de code mogelijk zijn, zonder deze opnieuw te compileren. Deze zaken komen terug in de volgende namespaces die worden gebruikt bij een Silverlight-project; zie afbeelding 3.

AgClr: Bevat voornamelijk de presentatiecontrols.

Mscorlib: Dit is de base class library van .Net met onder andere de Collections-, Diagnostics-, IO-, Security- en Text-classes.

System, System.Core: Aanvulling op de base classes met mogelijkheden voor networking met behulp van de HttpWebRequest en de HttpWebResponse-classes.

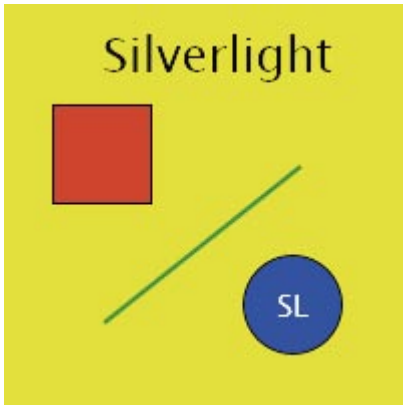
System.Silverlight: Deze namespace zorgt ervoor dat de Silverlight-applicatie in de browser gehost kan worden.

System.Xml en System.Xml.Core: Kan gebruikt worden bij het werken met XML- bestanden.

Opbouw project

Een Silverlight-project bestaat minimaal uit:

- 1 html- of aspx-bestand
- 2 javascriptbestanden
- 1 xaml-bestand
- 1 c#-bestand



Afbeelding 4. De code die in de browser zo wordt weergegeven

```
<Storyboard x:Name="sbButton">
  <DoubleAnimationUsingKeyFrames BeginTime="00:00:00"
  Storyboard.TargetName="cvEllipse" Storyboard.TargetProperty="
  (UIElement.RenderTransform).(TransformGroup.Children)[0].
  (RotateTransform.Angle)">
    <SplineDoubleKeyFrame KeyTime="00:00:00"
      Value="0"/>
    <SplineDoubleKeyFrame KeyTime="00:00:00.5"
      Value="360"/>
  </DoubleAnimationUsingKeyFrames>
</Storyboard>
```

Codevoorbeeld 3.

```
public void Page_Loaded(object o, EventArgs e)
{
  InitializeComponent();
  rectangle.MouseLeftButtonDown += new
    MouseEventHandler(rectangle_MouseLeftButtonDown);
}

void rectangle_MouseLeftButtonDown(object sender, MouseEventArgs e)
{
  sbButton.Begin();
}
```

Codevoorbeeld 4.

De html-pagina wordt opgeroepen door een gebruiker in een browser. In de html worden de javascriptbestanden geïmporteerd. Eén javascriptbestand (Silverlight.js) controleert of Silverlight is geïnstalleerd en zo ja, welke versie. Dit bestand zorgt er ook voor, mocht Silverlight niet op de client geïnstalleerd zijn, dat er een afbeelding van Silverlight wordt getoond met een link naar de downloadpagina. In de andere javascriptfile (Testpage.html.js) staat beschreven welk xaml-bestand als eerste wordt aangeroepen. Het c#-bestand fungeert als een code-behind voor dit xaml-bestand.

XAML

Deze nieuwe taal wordt gebruikt om onder andere WPF- en Silverlight-applicaties te beschrijven en op te maken. In XAML kunnen Silverlight-objecten (UI, data binding, events, et cetera) gestructureerd worden gedeclareerd. Een voorbeeld van de inhoud van een eenvoudig xaml-bestand is te zien in codevoorbeeld 1.
Elk xaml-bestand bij Silverlight heeft als parent een Canvas. Op dit Canvas staat een TextBlock en dit kan vergeleken worden met een Label in ASP.NET of WinForms. Het TextBlock dat als enige te zien is wanneer de applicatie wordt gestart, heeft de naam 'tbSilverlight'. Met deze naam kan het TextBlock aangeroepen worden in de code-behind-file. De namespace voor dit project heet AvanadeSilverlight en alle c#-code wordt gecompileerd en gebundeld in AvanadeSilverlight.dll. De assembly wordt gekopieerd naar de map ClientBin. Deze informatie wordt aan het 'x:Class'-attribuut meegegeven. Als dit xaml-bestand geladen is, wordt als eerste Page_Loaded aangeroepen. Dit event is aangemaakt in de c#-code bij het creëren van de Silverlight-pagina.

Visuele elementen

In de xaml-code kunnen verschillende objecten worden neergezet, zoals een cirkel of een vierkant. De elementen kunnen op allerlei manieren roteren, van vorm veranderen, of geanimeerd worden.
In codevoorbeeld 2 is een aantal van deze objecten in xaml gedefinieerd. Hier worden namelijk een vierkant, cirkel en lijn op het Canvas getekend. Elk element krijgt met behulp van attributen zijn eigen eigenschappen. Deze code ziet er in de browser als volgt uit, zoals te zien is op afbeelding 4.

```
<Canvas
  xmlns="http://schemas.microsoft.com/client/2007"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  x:Name="rootCanvas"
  x:Class="Mix07.DefaultCanvas;
    assembly=ClientBin/SilverlightAirlines.dll"
  xmlns:app="clr-namespace:Mix07;
    assembly=ClientBin/SilverlightAirlines.dll">

  <app:Map x:Name="map" Canvas.Left="5" Canvas.Top="5"
    Width="850" Height="350" />
  <app:Calendar x:Name="calendar" Canvas.Left="860"
    Canvas.Top="5" Width="228" Height="350" />
  <app:ItineraryPicker x:Name="itineraryPicker" Canvas.Left="5"
    Canvas.Top="360" Width="1083" Height="123" />
</Canvas>
```

Codevoorbeeld 5

De cirkel rechtsonder is een Ellipse en een TextBlock gegroepeerd in een Canvas. Als er namelijk op het rode vierkant wordt geklikt, kan deze groep in zijn geheel om zijn as draaien. Hiervoor wordt de RenderTransform-property gebruikt van het Canvas. Een attached property is een nieuw concept in xaml. Je kunt hiermee een property van een parent-element in een child-node aanpassen. In het voorbeeld wordt een TransformGroup gekoppeld aan de RenderTransform van het Canvas waarin de cirkel en het TextBlock staan. In de TransformGroup kan de waarde van Angle veranderd worden. Een TransformGroup wordt ook gebruikt om elementen te schalen, uit te rekken (skew) of te spiegelen. In het voorbeeld staat de waarde van Angle nog op 0. We willen namelijk de cirkel draaien en de waarde van Angle veranderen als op het rode vierkant wordt geklikt. Iedere fractie van een seconde moet de waarde aangepast worden, zodat er een animatie ontstaat. Een animatie kan gedefinieerd worden met behulp van het element Storyboard. Een Storyboard kun je vergelijken met een tijdlijn voor een animatie. Het object Storyboard bevat een aantal methodes om de animatie aan te sturen. Deze zijn Begin(), Pause(), Resume() en Stop(). Codevoorbeeld 3 laat een Storyboard zien in xaml.
Je moet in een Storyboard een target-element en -property opnemen. Iedere stap in de animatie is een keyframe. Met iedere keyframe kun je opgeven hoe de waarde van de property verandert. In het voorbeeld wordt de waarde van Angle in een halve seconde van 0 naar 360 gezet. Zo draait de cirkel om zijn as. Nu enkele elementen op het scherm getekend zijn en een Storyboard is gedefinieerd, kunnen we de animatie laten afspelen als op het vierkant wordt geklikt. Als je de code-behind file opent, staat daar enkel de methode 'Page_Loaded' in. Deze is terug te vinden bij het parentCanvas in de xaml-file. In 'Page_Loaded' wordt 'InitializeComponent' aangeroepen die de xaml-elementen omzet naar .Net-objecten. Dit gaat automatisch, iedere keer als het project gebuild wordt. In codevoorbeeld 4 is de implementatie van het klik-event te zien. De Begin()-methode van het StoryBoard wordt aangeroepen en de animatie speelt af. Tot zover een simpel voorbeeld met de basisfunctionaliteiten van Silverlight. Er is echter nog veel meer mogelijk. Daar volgt hieronder een voorbeeld van.

Airlines-demo

Om de mogelijkheden van Silverlight te laten zien, heeft Microsoft een demoapplicatie gemaakt met de naam Silverlight Airlines.

Afbeelding 5. De demoapplicatie Silverlight Airlines

```
System.IO.Stream s =
   this.GetType().Assembly.GetManifestResourceStream("Mix07.Map.xaml");
FrameworkElement root =
   this.InitializeFromXaml(new System.IO.StreamReader(s).ReadToEnd());

Canvas cv = root.FindName("cvMap") as Canvas;
```

Codevoorbeeld 6

Deze applicatie is te zien in afbeelding 5. De applicatie is een ASP.NET-website van een fictieve luchtvaartmaatschappij. In het Silverlight-gedeelte is het mogelijk een vlucht te boeken. Dit doe je door een periode te kiezen in een kalender. Alle beschikbare vluchten in deze periode komen in een tabel onder aan de pagina te staan. Bij het aanklikken van iedere vlucht zie je de route in een kaart. De kracht van Silverlight is in deze applicatie te zien. Met een paar clicks is het doel al bereikt en het geheel ziet er ook nog eens aantrekkelijk uit. De drie onderdelen op het scherm, de landkaart, de kalender en de tabel met vluchtinformatie, zijn alle drie Silverlight-usercontrols. In codevoorbeeld 5 is te zien hoe dit er in xaml uitziet. De onderdelen Map, Calendar en ItineryPicker vind je terug in de namespace Mix07.

Het maken van user-controls in Silverlight werkt op de volgende wijze. Je maakt twee bestanden aan: één xaml-bestand en één c#-bestand, bijvoorbeeld Map.xaml en Map.cs. Deze bestanden zijn de bronbestanden voor het control. De xaml-file wordt gebruikt om alle visuele elementen te definiëren. De xaml-file wordt ingeladen in de c#-file met de code uit voorbeeld 6. Met het FrameworkElement is het mogelijk te programmeren tegen de onderdelen die behoren tot de interface. Deze class bevat de methode FindName() en hiermee kunnen alle xaml-objecten opgezocht en gebruikt worden in de c#-code.

Tot slot

Silverlight bezit veel mogelijkheden om webapplicaties interactiever en mooier te maken. Het is vrij eenvoudig om met deze nieuwe techniek te starten. Op de Silverlight-homepage onder de link 'Get Started' staan alle nodige downloads. Daarbij is het nog eens erg leuk om Silverlight-applicaties te maken. Neem een kijkje op de Silverlight-website of bekijk de vele samples die beschikbaar zijn.

Max Remkes is programmeur bij Avanade Nederland (www.avanade.com). Hij werkt op de afdeling Application Management en heeft veel ervaring met webapplicaties. Zijn e-mailadres is maxr@avanade.com.

Referenties

Silverlight-homepage: http://silverlight.net/

Silverlight Wikipedia: http://en.wikipedia.org/wiki/Microsoft_Silverlight

Silverlight op MSDN: http://msdn2.microsoft.com/en-us/silverlight/default.aspx

Silverlight Airlines: http://delay.members.winisp.net/SilverlightAirlinesDemo/

50 Silverlight-samples: http://blogs.msdn.com/tims/archive/2007/07/07/from-a-to-z-50-silverlight-applications.aspx

Microsoft Expression Blend: http://www.microsoft.com/netherlands/expression

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Net;
using System.Text;
using System.Reflection;

namespace Ordina.MicrosoftSolutions
{
    /// <summary>
    /// Abstract class voor een .NET specialist.
    /// </summary>
    /// <remarks>
    /// .NET specialisten moeten een class hiervan afleiden en de abstracte properties en methodes implementeren
    /// in de .exe of in een aparte class library die in dezelfde map geplaatst moet worden.
    /// </remarks>
    public abstract class DotNetSpecialist
    {
        public static void Main(string[] args)
        {
            DotNetSpecialist specialist = FindSpecialistWannaBe();

            if (specialist == null)
            {
                Console.WriteLine("Geen DotNetSpecialist implementatie gevonden!");
            }
            else
            {
                specialist.Start();
            }
        }

        /// <summary>
        /// Naam van de specialist.
        /// </summary>
        public abstract string Name { get; }

        // Antwoorden van de specialist.
        public abstract string WaarStaatUMLVoor();
        public abstract string WaarStaatRUPVoor();
        public abstract string WaarStaatXAMLVoor();
        public abstract string WaarStaatMVCVoor();
        public abstract string WaarStaatWCFVoor();
        public abstract string WaarStaatWFVoor();
        public abstract string WaarStaatDSLVoor();

        private void Start()
        {
            String report = "Volgens de specialist " + Name + "\nHebben de volgende termen de betekenis"
                + "\n\u0055\u004d\u004c " + WaarStaat\u0055\u004d\u004cVoor() + "\n\u0052\u0055\u0050 "
                + WaarStaat\u0052\u0055\u0050Voor() + "\n\u0058\u0041\u004d\u004c " + WaarStaat\u0058\u0041\
u004d\u004cVoor()
                + "\n\u004d\u0056\u0043 " + WaarStaat\u004d\u0056\u0043Voor() + "\n\u0057\u0043\u0046 "
                + WaarStaat\u0057\u0043\u0046Voor() + "\n\u0057\u0046 " + WaarStaat\u0057\u0046Voor()
                + "\n\u0044\u0053\u004c " + WaarStaat\u0044\u0053\u004cVoor();
            Console.WriteLine(report);
            Console.WriteLine("Druk op een toets en kijk voor een leuke baan op de jobportal van ordina http://jobportal.ordina.
nl/.");
            Console.ReadKey();
            KijkBijOrdina();
            Console.ReadKey();
        }

        private static DotNetSpecialist FindSpecialistWannaBe()
        {
            Assembly thisAssembly = Assembly.GetExecutingAssembly();
            DotNetSpecialist specialist = GetSpecialistFromAssembly(thisAssembly);
            if (specialist == null)
            {
                string assemblyLocation = new FileInfo(thisAssembly.Location).DirectoryName;
                string[] assembliesToLoad = System.IO.Directory.GetFiles(assemblyLocation, "*.dll");

                foreach (string assemblyToLoad in assembliesToLoad)
                {
                    Assembly loadedAssembly = Assembly.LoadFile(assemblyToLoad);
                    specialist = GetSpecialistFromAssembly(loadedAssembly);
                    if (specialist != null) break;
                }
            }
            return specialist;
        }

        private static DotNetSpecialist GetSpecialistFromAssembly(Assembly assembly)
        {
            Type[] typesInAssembly = assembly.GetTypes();
            foreach (Type t in typesInAssembly)
            {
                if (t.IsSubclassOf(typeof(DotNetSpecialist)))
                {
                    return (DotNetSpecialist)t.GetConstructor(Type.EmptyTypes).Invoke(null);
                }
            }
            return null;
        }

        private void KijkBijOrdina()
        {
            const string url = "http://jobportal.ordina.nl/";
            const string internetExplorer = "iexplore.exe";

            System.Diagnostics.Process.Start(internetExplorer, url);
        }
    }
}
```