Silverlight

MIT 6.470, IAP 2011

Jeff Wilcox (jwilcox@microsoft.com)

Me

- Senior Software Development Engineer
  - Silverlight Phone & Devices Team
- Contact info
  - http://www.jeff.wilcox.name/
  - jwilcox@microsoft.com
  - Twitter / @jeffwilcox
- University of Michigan
  - CS Class of 2005

My dev role @ Microsoft

- Technical expert in many core areas
- Expected to be a generalist, too
  - able to pick up anything and rock the code & solutions
  - influence other teams, disciplines & devs
- Extremely competitive team ☺
- 50% C++, 25% C#, 25% customers
- Conferences & engage industry engineers

Today

- Why Silverlight?
- Getting Started
- Layout System
- Controls and Templates
- App Model
- Data Visualization & Filtering
- Q & A

Silverlight

- Introduced in ‘07
- Silverlight 4
  - Cross-browser, cross-platform
    - Chrome, Firefox, Safari, IE, “out-of-browser”
    - Mac & PC
  - .NET (C#, VB.NET, IronRuby, IronPython)
  - Rich media, net apps, line-of-business
- Moonlight – Novell’s Linux client
- Phone & Devices

Strong Points

- Ability to built robust apps
- Static typing is great for formal team projects
  - JavaScript developers are using “Script#” and “Coffeescript”
- Thread pool, background work, multi-core
- Decade of .NET open source to pull from
- The layout and animations are super flexible (Grid control rocks compared to CSS)
  - HTML5 graphics by MS’s Aseem Kishore Wed noon

Strong Points

- GPU Acceleration, HD video
- Visual Studio rocks
- Great developer community
  - Silverlight.net tutorials and forums
  - StackOverflow.com
  - Blogs

How's it built?

- Silverlight is mostly native code
  - Cross-platform C++ at Microsoft!
  - Common Language Runtime (CLR) / managed
  - Lightweight PALs and type aliasing
- User interface framework is C#
- We pull together codecs, graphics, parsers, UI frameworks, networking & web services


POPCORN

6.470 Web Stack

- Client
  - Mix & Match (HTML, HTML5, Silverlight, Flash)
- Web Services
  - Any
- Server or Cloud
  - Any
- Source Control
  - Any

Tools

- Visual Studio Express
  - Free version of VS
- Visual Studio 2010
  - Great debugging
  - Intellisense
- Expression Blend
  - “Developer + Designer”
  - Awesome for storyboard & template work
- DreamSpark.com ~ free

Mac?

- Silverlight SDK for Eclipse
- Apple Boot Camp
- VMWare or Parallels

The Silverlight Plugin

- Small and easy one-time install (5MB)
- Instantiate via an object tag
- Can be windowless (above/below HTML)
- No need for full page

Output: A Silverlight Xap

- .Xap extension
  - "ZAP"
  - Just a zip file
- Contains
  - Image and media resources
  - Code compiled into .NET assemblies
  - Referenced libraries
  - XAML resources ("Zammel" / XML object trees)
    - Extensible Markup Language

A Silverlight Solution in Visual Studio

- Application (Silverlight)
- Optional web project (static or ASP.NET)
- Unit test project inc. w/ Silverlight Toolkit
- Reference libraries (.dll)
- Class library projects

XAML Object Graphs

- User Interface Definition
  - Similar to XIBs (iOS) and MXML (Flash)
- Helps enforce separation of logic & UI
  - The best apps use the M-V-VM model (Model View ViewModel)
- Enable rich design surfaces

XAML / Object Graphs

- Every element is a named object instance
  - No aliasing – direct element name to Type
  - Button maps to Button
- Children of an element indicate containment
- Properties are set via attributes or property element syntax

XAML is Code

```
<Button Height="48"
Width="104" Content="Hello"/>
```

=

```
Button b = new Button
{
   Height = 48,
   Width = 104,
   Content = "Hello"
};
```

- Special expanded property syntax

```
<Button Content="Hello"/>
```

=

```
<Button>
   <Button.Content>
      Hello
   </Button.Content>
</Button>
```

TIP:

Spacious XAML is good for your sanity & source control system.

XAML and Code Behind

- App.xaml
  - Global resources
  - Implicit styles
- MyPage.xaml
  - User interface
  - Resources scoped to the page
- MyPage.xaml.cs
  - Logic
  - Event handlers

XAML interpreted

```
<UserControl x:Class="SilverlightApplication2.Page"
   xmlns="…"   xmlns:x=".. " Width="400" Height="300">
   <Grid x:Name="LayoutRoot" Background="White">
   </Grid>
</UserControl>
```

Design-time

```
public partial class Page :
System.Windows.Controls.UserControl {

      internal System.Windows.Controls.Grid LayoutRoot;
```

Compile-time

```
public void InitializeComponent()
{
   LoadFrom(this, Page.xaml); //Pseudocode
   LayoutRoot = this.FindName("LayoutRoot"); //Pseudocode
}
```

Run-time

demo

- Creating a new Silverlight project and exploring the development environment
- Exploring generated files, x:Name

Silverlight

- Input
  - Keyboard, Mouse
  - Touch
  - Ink
- Modern App Framework
  - Data binding
  - Model – ViewModel (MVVM)
- Data
  - LINQ, LINQ to XML, XML
  - Isolated Storage
- Base Class Library (BCL)
  - Generics
  - Collections
  - Cryptography
  - Threading
- UI Core
  - Vector Shapes
  - Layout
  - Animation
  - Text
  - Images
- XAML
- Media Stack
  - VC1, WMA, MP3
  - H.264, AAC
- Communication
  - REST
  - JSON
  - RSS/ATOM
  - SOAP

Primitives

- Rectangle
- Ellipse
- Path
- Image
- MediaElement
- TextBlock

- No owner draw pixels

- Brush
  - SolidColorBrush
  - GradientBrush
  - ImageBrush
  - VideoBrush

Transforms

- Translation
- Rotation
- Scaling
- 3D Projection

```
<Button Content=“Hi”>
    <Button.RenderTransform>
        <RotateTransform
              Angle=“35”/>
    </Button.RenderTransform>
</Button>
```

demo

- Graphics Primitives

Properties

- Standard CLR Property
- Dependency Property
- Attached Property
- Types with INotifyPropertyChanged

Standard CLR Property

```
public string FirstName { get; set ; }
private string _lastName;
public string LastName {
    get { return _lastName; }
    set { _lastName = value; }    }
public string FullName {
    get {
        return FirstName + (FirstName != null && LastName != null)
        ? “ “ : string.Empty + LastName;
    }
}
```

- No way to indicate a changing value
- Fine for data models, not for UI

Dependency Property

- Special property type
- Defined a specific way on the owning type
- Similar to the Flywheel GoF pattern

- Has a default value
- Framework handles change notifications
- Optional value changed handler
- Don’t do anything in your getters & setters!

Dependency Properties

- Used for
  - Animation
  - Bindings
  - Styling
  - Property inheritance

- Quick demo: A dependency property in code

Attached Property

- Let you extend elements and properties
- Decouples the owner from the object

```
<Button
   Grid.RowSpan=“2”
   Content=“Hello world.”
/>
```

INotifyPropertyChanged

- Used for data and model types
- Enables rich data binding without dep. props.

```
public class FlickrPhoto : INotifyPropertyChanged {
   public event PropertyChanged;  // …
   public string FirstName
   {
       get { return _firstName; }
       set {
             _firstName = value;
             NotifyPropertyChanged(“FirstName”);
       }
   }
}
```

Some properties are inherited

- DataContext
- Foreground
- Font properties
  - FontFamily
  - FontSize
  - etc.

Layout System

- MinHeight
- MaxHeight
- Height

- MinWidth
- MaxWidth
- Width

- Margin
- Padding

Layout System

- Measure Pass
- Arrange Pass

Layout Containers

- StackPanel
- Grid
- Border
- Canvas
- WrapPanel
- Your own Panel

Controls

- Button
- CheckBox
- ComboBox
- DataGrid
- HyperlinkButton
- TextBox
- TreeView

… all the usual suspects

demo

- Layout and controls

Silverlight Toolkit

- Download from http://silverlight.codeplex.com/
- Great way to learn by example control development

demo

- Silverlight Toolkit controls
- Other control examples

The User Interface Thread

- Threading is difficult
- One thread handles
  - Input
  - OS / browser plugin messaging
  - Data binding to UI and control properties
  - Rasterizing*
  - Animations*
- Dispatcher helps with sending cross-thread messages

Class Hierarchy

- DispatcherObject : object
  - Dispatcher
- UIElement : DispatcherObject
  - Visibility
  - RenderTransform
  - Projection
  - Opacity

Class Hierarchy

- FrameworkElement : UIElement
  - DataContext
  - Margin
  - Resources
  - Style
- Control : FrameworkElement
  - Background
  - Padding
  - IsEnabled

3 key types of controls

- ContentControl
- UserControl
- ItemsControl

ContentControl : Control

- Content
- ContentTemplate

UserControl : ContentControl

- Fixed content
- Used in most simple apps / forms / views
- Hook up events in the code behind

ItemsControl

- Contains a bunch of items
  - ListBox
  - TreeView (hierarchical)

- Quick demo: A list box

“Look-less” Controls

- All official controls can have any look
- Controls have
  - A Template
  - A default Style

Resource Dictionaries

- Places to store styles, brushes, converters
- Inside any FrameworkElement's Resources property
- Inside App.xaml
- Inside separate XAML files that can be merged in

Styling & Templating

- CSS-like styles
- Put in Resource Dictionaries
- Styles have a target type and optional name
- “Implicit styles” have no name
  - Like CSS entities without #someId or .someClass
- BasedOn attribute for inheritance

- Quick demo: Resources, Re-templating

Events

- Event handlers are set in XAML just like properties
- Events bubble up the visual tree from the source based on the Handled property

- Quick demo: An event!

Data Binding

- Easily connect information and UI
- A binding has a source and a target
  - Source: a dep. prop. or INotifyPropertyChanged
  - Target: a dependency property
- If the source isn't specified, the DataContext is used.
- Value converters can interpret inputs and offer a specialized output
- Curly-brace syntax

```
<TextBox Text="{Binding FirstName, Mode=TwoWay}" />
```

demo

- Data binding
- Value converters

Navigation Framework

- Building out applications (vs. Silverlight islands)
- Pages (UserControl) are on the client side

- Address Bar: MyPage.html#/Home

App Model

- App Lifetime
- Threading
- Networking & Web Services
- Security
- HTML Interoperability
- Error handling

Asyncronous

- We live in an async world. Code for it!

- Event callback pattern:
  - Obj.SomethingCompleted += …;
  - Obj.SomethingAsync();
- Callback pattern:
  - Void SomeMethod(Action<string, Exception> a);

Threading

- .NET Threads
  - ThreadPool
  - Your own Thread
  - BackgroundWorker
- Dispatch or use a sync context for the UI thread

```
System.Deployment.Current.Dispatcher.BeginInvoke( () => /* some
code here */ { } );
```

- Quick demo: Background worker, dispatch errors

Networking & Web Services

- “Add Service Reference” creates proxies
- Hand-coding REST is fun ☺
- XML & JSON are easy to work with

Web Requests

- HttpWebRequest and WebClient
- WebClient is a simplified wrapper
- GET with DownloadString, POST UploadString

```
WebClient c = new WebClient();
c.DownloadStringCompleted += (x, xe) =>
    MessageBox.Show(xe.Result);
c.DownloadStringAsync();
```

Security

- Silverlight plays in its own sandbox
  - There are cross-domain web request restrictions
  - Isolated storage (like HTML5 local storage)
  - Open file dialog, save file dialog
- Only one level of trust for the plugin
  - Nothing bad can happen
  - There is an “out of browser” elevation story for business apps

HTML Interoperability

- Can communicate with the page
  - Get, build and modify the HTML DOM
  - System.Windows.Browser.HtmlPage.Window obj
  - Invoke script objects
- JavaScript can communicate with your types
  - Can add a [ScriptObject] attribute to your objects
  - JS can then call methods, set properties, etc.

6.470 Requirements

- Your site must present its content in an efficient matter. You must achieve this goal by implementing *one* of the following.
- **Data visualization.** Your application presents data in a visual manner that enhances the users' understanding of the data. The visualization should highlight some feature of the data that isn't obvious or trivial. The user must be able to influence which data gets displayed in the visualization's. You are allowed to use a preexisting implementation for your visualization, like a jQueryUI widget, provided that it fits with your application. Original visualizations will be recognized with special awards.
- **Dynamic filtering.** Your application shows a small (< 10%) subset of data which is most relevant to the user at the moment, or ranks the data such that the most relevant items are shown first. The subset and/or the ordering of the data must change in response to the user's actions. Three examples that would fulfill this requirement are full-text search, criteria-based search, and a "recommended for you" section. For the latter, you will have to convince us that the contents changes based on user actions.

Data Visualization

- Silverlight Toolkit’s Charting Controls
- Deep Zoom
- PivotViewer

Dynamic Filtering

- Local filtering
  - Easy with C#
    - LINQ
    - LINQ to XML
  - With a local dataset, very quick results
  - Can use threading to keep the UI responsive
- Keep it the service tier

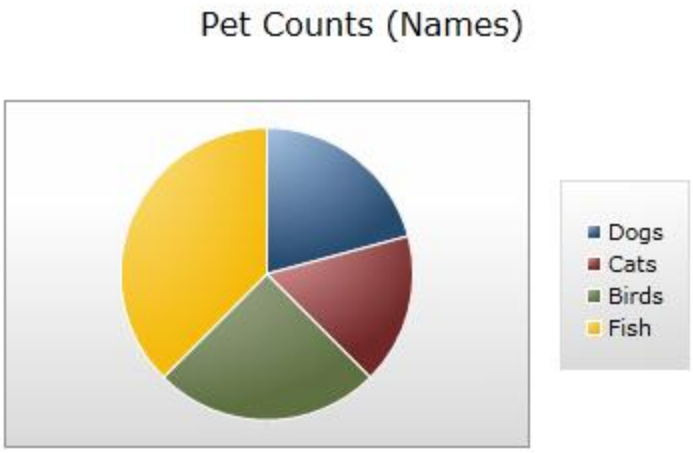
LINQ to XML

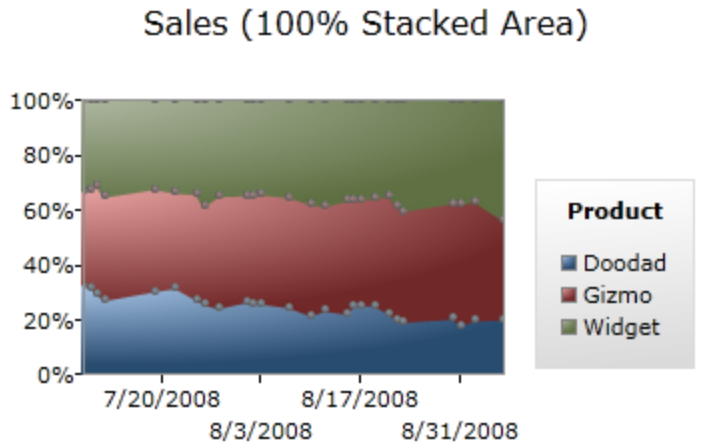
- Local & fast way to query and filter XML
- Two syntaxes
  - SQL-style
  - C#-style

```
var doc = Xdocument.Parse(someString);
ListBox.ItemsSource = from person in doc.Descendants(“Person”)
where person.Element(“City”).Value == textBox1.Text select
person.FullName;
```

Silverlight Chart Controls

- Included in the Silverlight Toolkit
- Many styles, can customize with Blend
- Support observable (dynamic) data points





Deep Zoom

- Based on image pyramids (DZC/DZI)
- Can be interactive and dynamic

- http://memorabilia.hardrock.com/

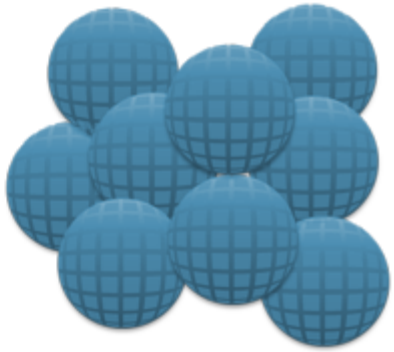
PivotViewer

- Based on item metadata and Deep Zoom image sets

Sample link (cars)
Sample link (world leaders)

- http://www.silverlight.net/learn/pivotviewer/

PivotViewer

Simple Collections	Linked Collections	Just in Time Collections
		
Difficulty: Easy **Size:** Up to 3,000 items	**Difficulty:** Medium **Size:** Limited by storage complexity	**Difficulty:** Hard **Size:** Unbounded
Details: ▪ The most common type of collection. ▪ Data is static and loaded all at once. ▪ Visuals are static and contained in one DZC (see: Collection Image Content).	**Details:** ▪ Generally used for collections in the several thousands of items. ▪ Collection is stored as a composition of inter-linked *simple* collections. ▪ Data is static and loaded one *simple* collection at a time. ▪ Visuals are static and contained in multiple DZCs (see: Collection Image Content).	**Details:** ▪ Used for very large data sets (hundreds of thousands of items or more). ▪ Data is dynamic (see Collection Hosting) and loaded in response to a query. ▪ Visuals are partially dynamic and contained in dynamic DZCs (see: Collection Image Content).

Open Source at Microsoft

- Ms-PL
  - “Microsoft Permission License”
  - Do anything with it

- OSI Approved
- IANAL: Similar to MIT license

Silverlight & Open Source

- SDK Controls
- Silverlight Toolkit
- Silverlight for Windows Phone Toolkit
- Team Member Blogs
- Community Projects

Get the tools

- http://www.dreamspark.com/
  - Visual Studio 2010 Professional
  - Expression Blend 4
- http://www.silverlight.net/
  - Silverlight Tools for SL4
- http://silverlight.codeplex.com/
  - Silverlight Toolkit
- http://create.msdn.com/
  - If interested in phone development

Get help

- http://www.silverlight.net/
  - Silverlight forums, tutorials, blogs
- http://www.stackoverflow.com/
- Twitter
  - Vibrant community including professionals, students, “MVPs”, Microsoft developers
  - #Silverlight, #sldev

Questions?

- Jeff Wilcox
  - jwilcox@microsoft.com
  - @jeffwilcox on Twitter
  - http://www.jeff.wilcox.name/