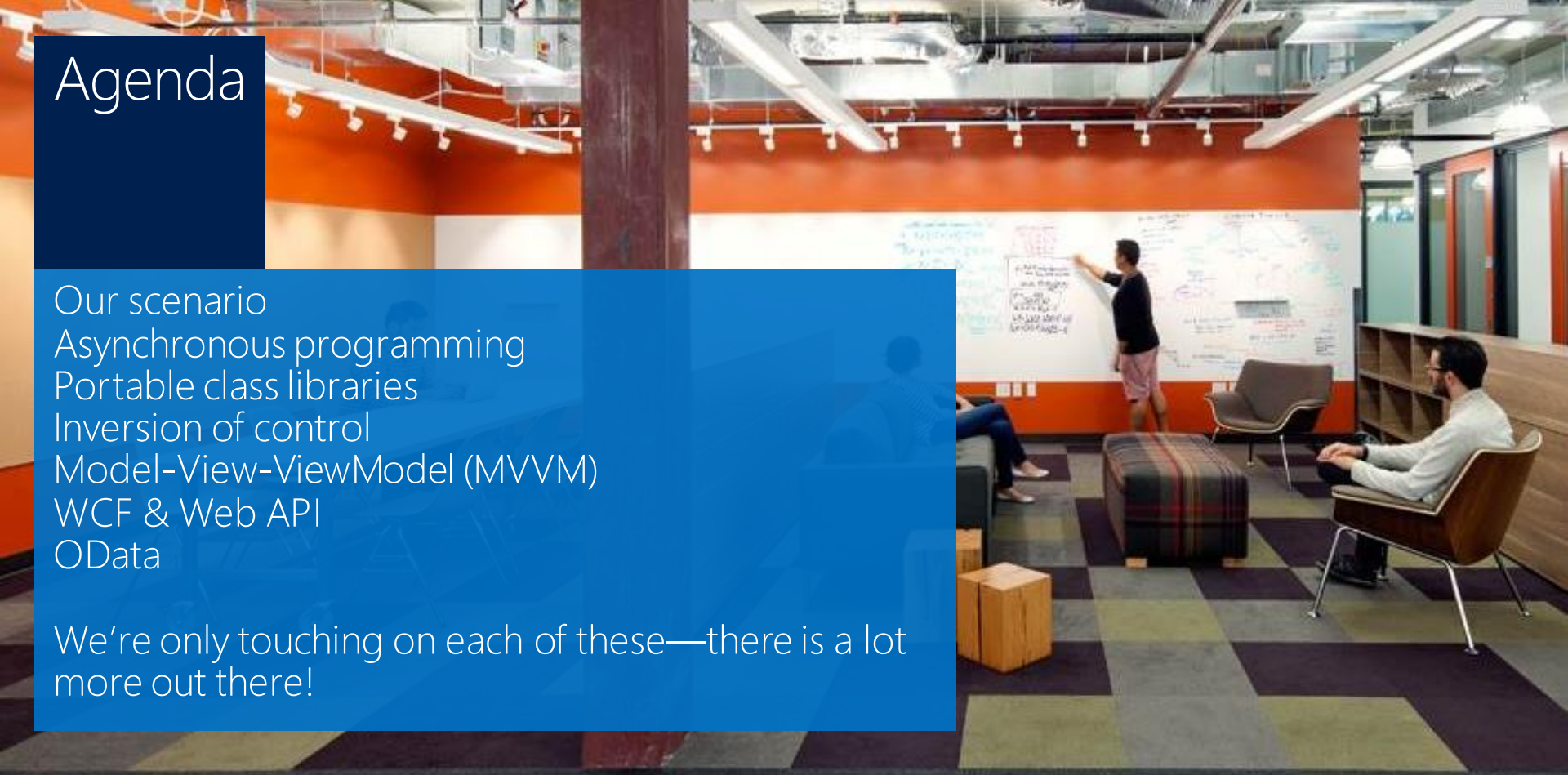
What is a Services Architecture?

Presenter Name

Agenda

Our scenario
Asynchronous programming
Portable class libraries
Inversion of control
Model-View-ViewModel (MVVM)
WCF & Web API
OData

We're only touching on each of these—there is a lot more out there!

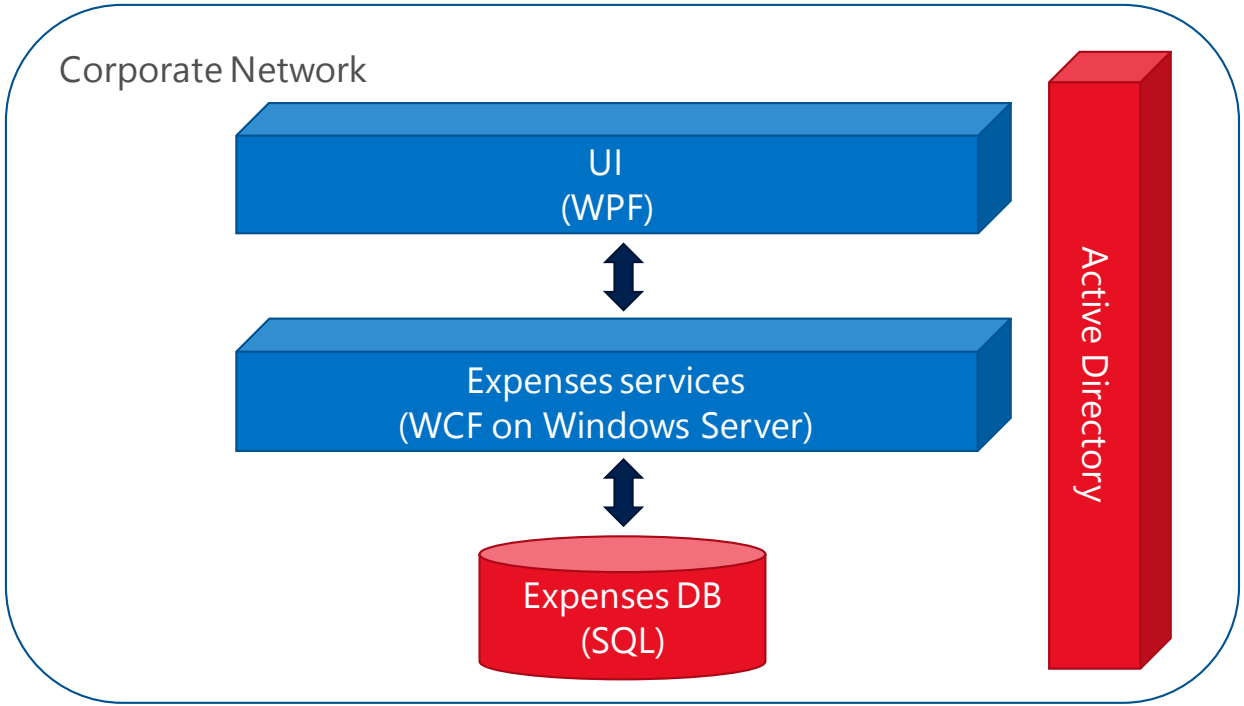


Microsoft

Our scenario: Expense Reporting

- Typical line-of-business application
- Create & submit reports
- View past reports
- Approve reports (if manager)

Application Topography



Feature Backlog

- Improve architecture, maintainability, and quality
  - Adopt a services architecture
- Improve accessibility, scalability, and operations
  - Move to the cloud
- Update the user experience
  - Build a more modern-looking user experience
- Expand device support
  - Companion apps for Windows Store and Windows Phone

Async, PCLs, IoC

Asynchronous Programming

- Critical for responsive UI, scale, & user satisfaction
- Traditionally a challenge due to manual requirements
- APIs often inconsistent
  - Polling
  - Callbacks
  - Events
  - Begin/End
- Progress and cancellation often not supported
- Error reporting unpredictable

Introducing Async & Await

- **`async`** makes your method asynchronous
  - Expects "await" in body
  - Only top-level methods (main, event handlers, etc) should be "async void"
- **`await`** makes the rest of your method a callback
- Paradigm being applied across all Microsoft platforms
  - Microsoft internal guidelines recommend Async for any API that could take longer than 50ms to complete
- Enables consistent progress, cancellation, and error infrastructure
  - Note that API support for progress and/or cancellation itself varies

Using async APIs with await

```
FileOpenPicker picker = new FileOpenPicker();
picker.FileTypeFilter.Add(".jpg");
StorageFile file =
     await picker.PickSingleFileAsync();
IRandomAccessStream stream =
     await file.OpenAsync(...);
BitmapDecoder decoder =
     await BitmapDecoder.CreateAsync(stream);
decoder. …
```

Exposing an async API

- Return a Task or Task<T>
- “async” is not necessary, unless you are using “await” within the method

Building an async API examples

```
public Task<string> RequestDataAsync(Uri uri)
{
    WebClient webClient = new WebClient();
    return webClient.DownloadStringTaskAsync(uri);
} // Rely on underlying Tasks whenever possible.
```

```
public Task<string> RequestDataAsync(Uri uri)
{
    return Task.Run<string>(
        () =>
        {
            WebClient webClient = new WebClient();
            return webClient.DownloadString(uri);
        });
} // Create Tasks when you have no other choice.
```

```
public Task<string> RequestDataAsync(Uri uri)
{
    var tcs = new
            TaskCompletionSource<string>();

    WebClient webClient = new WebClient();
    webClient.DownloadStringCompleted +=
        (_, args) =>
        {
            tcs.SetResult(args.Result);
        };
    webClient.DownloadStringAsync(uri);

    return tcs.Task;
} // Example of wrapping Async/Completed.
```

```
public async void Method(string[] args)
{
    MyClass instance = new MyClass();
    string result = await instance.RequestDataAsync(new
Uri("..."));
}
```

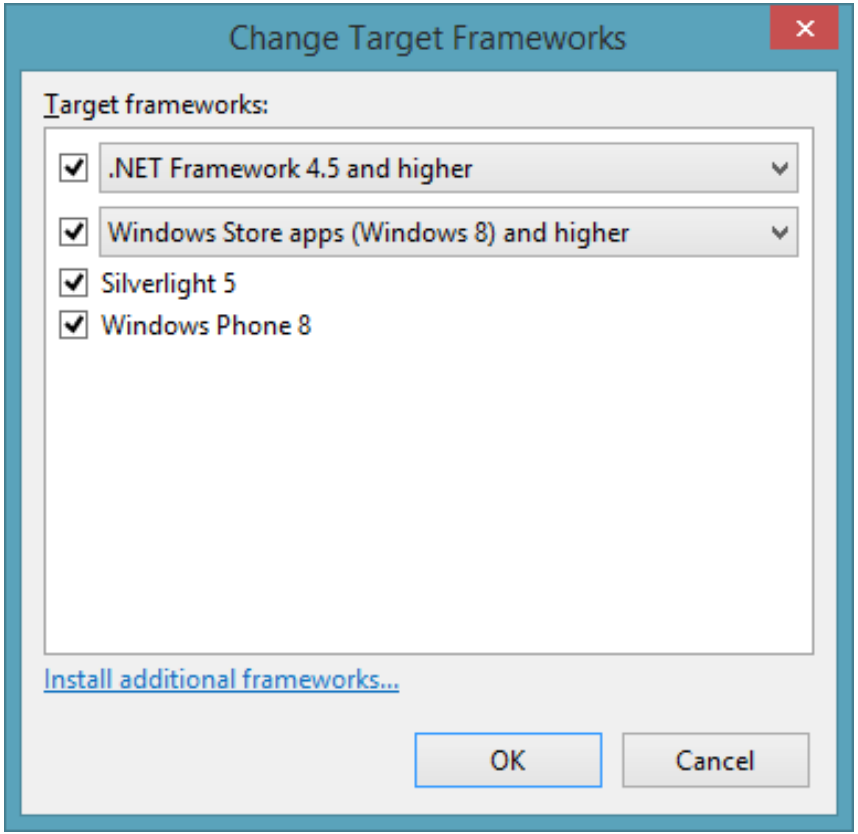
Additional async resources

- Async'ing Your Way to a Successful App with .NET
  - http://channel9.msdn.com/Events/Build/2013/3-301
- Creating Async Libraries That Are Modular, Reusable and Fast, in Microsoft Visual C# and Visual Basic
  - http://channel9.msdn.com/Events/TechEd/Europe/2013/DEV-B318

Portable class libraries (PCLs)

- One source
- One project
- One binary

- Multiple platforms!



What can I use and where?

Feature	.NET Framework	Windows Store	Silverlight	Windows Phone
Core	√	√	√	√
LINQ	√	√	√	√
IQueryable	√	√	√	7.5 and higher
Dynamic keyword	4.5 and higher	√	√	
Managed Extensibility Framework (MEF)	√	√	√	
Network Class Library (NCL)	√	√	√	√
Serialization	√	√	√	√
Windows Communication Foundation (WCF)	√	√	√	√
Model-View-View Model (MVVM)	4.5 and higher	√	√	√
Data annotations	4.0.3 and 4.5+	√	√	
XLINQ	4.0.3 and 4.5+	√	√	√
System.Numerics	√	√	√	

PCL tips

- Consider the platform tradeoffs from pulling in new libraries
  - Also check out NuGet for similar/updated packages supporting more platforms
- The more you do in PCLs, the broader that functionality can be leveraged

PCL tips

- Be careful about building PCLs too directly for a single consumer if others are planned, such as:
  - Assuming one auth model (like Basic) when other scenarios (AD, Oauth, etc) could be well supported with a little more planning
  - User selection of a file on desktops is significantly different from devices
- Abstract platform services as interfaces, and require those dependencies to be provided by the consumer
  - File access, UI, system services, etc.

Inversion of control (IoC)

- The Hollywood Principle
  - "Don't call us, we'll call you"

Inversion of Control

- Objects rely on their dependencies from the outside
  - Data connections
  - Algorithms
  - Platform-specific services
- Dependencies are usually provided as abstractions
  - IExpenseRepository
  - INavigationService
  - IPlatformService
- Enables thorough automated testing

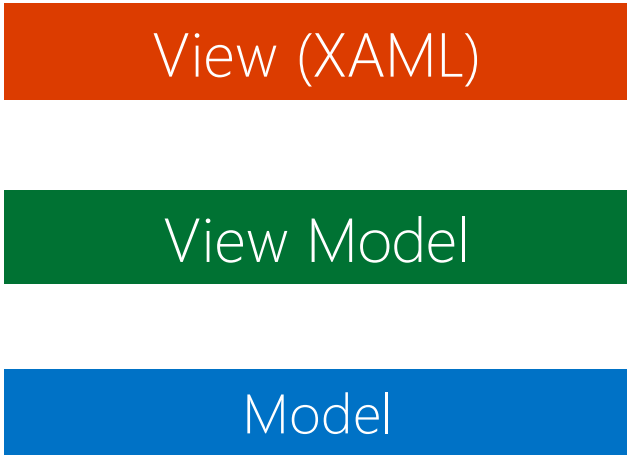
IoC patterns

- Factory
- Dependency injection
  - Constructor
  - Parameter
  - Setter
- Service locator is considered an anti-pattern in many circles, but is sometimes the most cost-effective options
  - Framework requires parameterless constructors
  - No support from MVVM libraries

MVVM

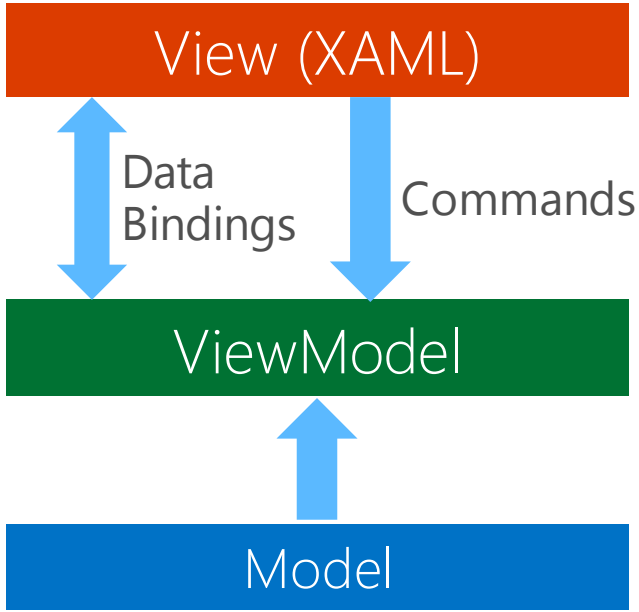
MVVM refresher

- Design/development separation
- Code reuse
- Multiple views on the same logic
- View-logic testability



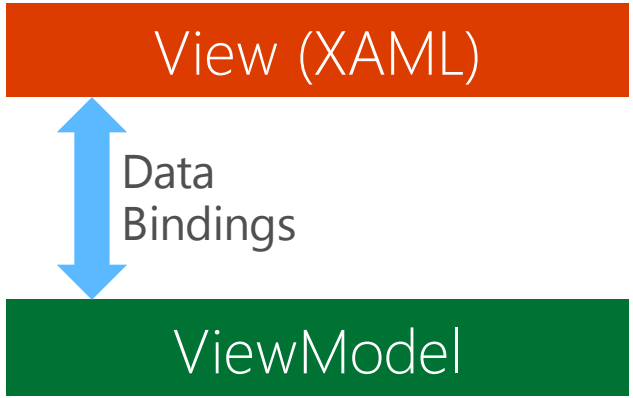
MVVM overview

- View
  - User interface
  - Navigate to views
  - Interaction layer
- ViewModel
  - Application logic
  - Service calls
  - Data management
- Model
  - Simple representation of data
  - No logic or functionality



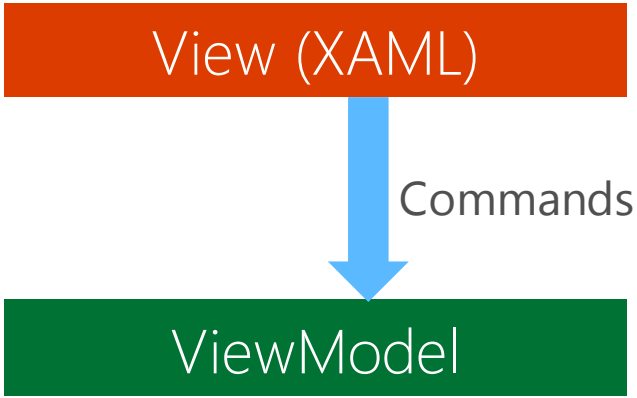
Data binding

- View - XAML
  - Text="{Binding MyProperty}"
- ViewModel - C#
  - INotifyPropertyChanged

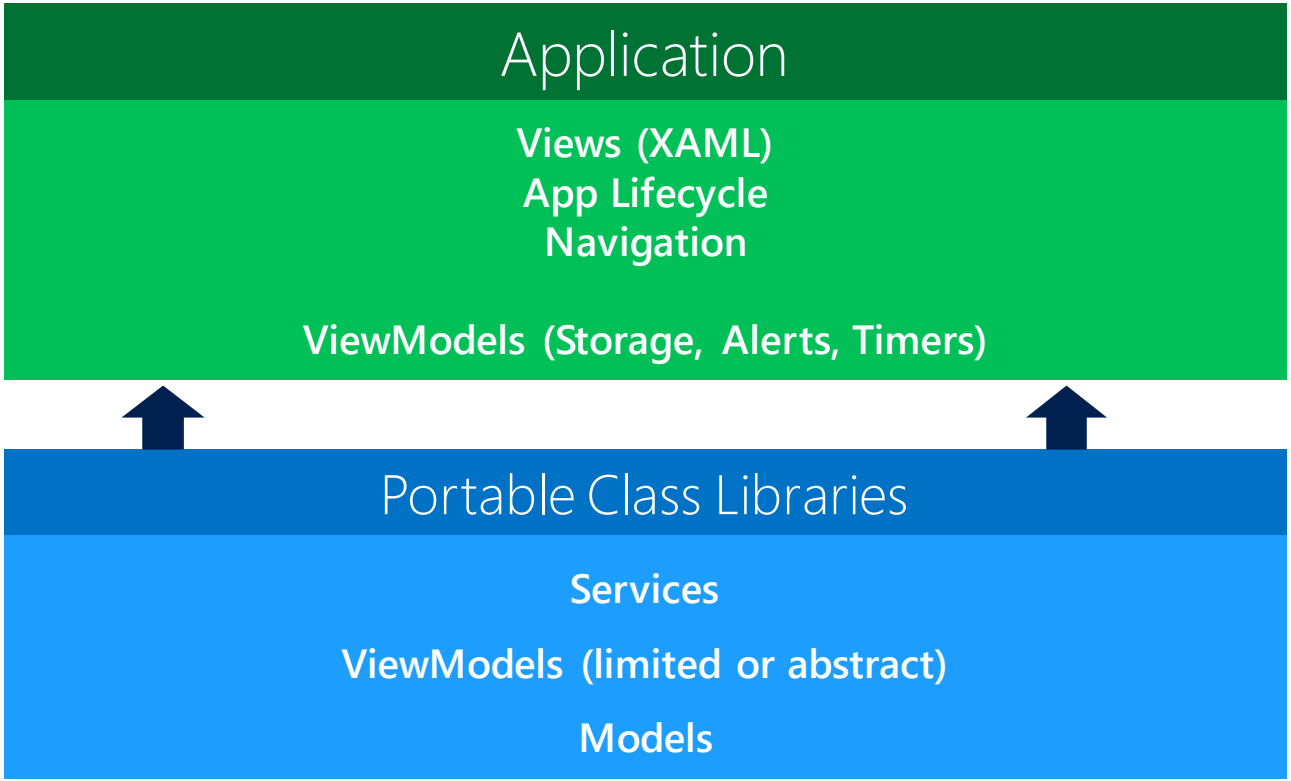


Commands

- View XAML
  - Command="{Binding MyCommand}"
- ViewModel - C#
  - ICommand
  - DelegateCommand
  - RelayCommand



Portable MVVM structure



MVVM tips

- No philosophy is perfect, especially not MVVM
- Understand the benefits of MVVM frameworks
  - PRISM
  - MVVM Light
  - Many others…
- Rely on tested patterns
  - Commanding
  - Dependency Injection
  - Inversion of Control
  - Observer
  - Repository
  - Service Locator
  - Many others…

Data Services

External services

- Always a good idea to abstract
- Protect DB calls
  - Even a simple service layer is worth it
- When in doubt, use a standard
  - SOAP, REST over HTTP, etc
- Use a service bus to traverse network boundaries
  - Internet client relying on an intranet service, for example

Windows Communication Foundation vs. Web API

- WCF
  - Multiple transport protocols
  - Multiple encodings
    WS-* standards
    Supports Request-Reply, One Way, and Duplex message exchange patterns
- Web API (preferred for new projects)
  - HTTP only
  - Supports a wide variety of media types (XML, JSON, etc)
  - Uses basic protocol and formats including HTTP, WebSockets, SSL
  - Is request/response due to nature of HTTP, but more patterns available via SignalR and WebSockets integration

OData, the Open Data Protocol

- A standardized protocol built on HTTP
- Uses the REST methodology
- Designed for formats like XML, ATOM, and JSON
- Provides a uniform way to represent metadata
- Client data context-friendly
- Great ecosystem of tools and developers
- Available via WCF Data Services and Web API

Summary

- Asynchronous programming
- Portable class libraries
- Inversion of control
- Model-View-ViewModel (MVVM)
- WCF & Web API
- OData

Resources

- Asynchronous Programming in the Microsoft .NET Framework 4.5
  - http://channel9.msdn.com/Events/TechEd/NorthAmerica/2013/DEV-H302#fbid=kt1W5OJuY58
- Modernizing WPF Line-of-Business Applications
  - http://channel9.msdn.com/Events/TechEd/NorthAmerica/2013/DEV-B325#fbid=kt1W5OJuY58
- Understanding Dependency Injection and Those Pesky Containers
  - http://channel9.msdn.com/Events/TechEd/NorthAmerica/2013/DEV-B207#fbid=kt1W5OJuY58

Resources

- Using Portable Class Libraries
  - http://channel9.msdn.com/Events/TechEd/NorthAmerica/2013/DEV-H323#fbid=kt1W5OJuY58
- Getting Started with MVVM
  - http://channel9.msdn.com/Shows/Visual-Studio-Toolbox/Getting-Started-with-MVVM


Q&A
Microsoft

Hands On Lab

Lab 1 - Adopt a services architecture
45 minutes

Microsoft