Xamarin

Me

- App developer since 2011
- iOS, Android, Unity development
- Founded Snapp early 2015
- Xamarin for 3+ years
- 15-20+ apps

What to expect

- Im a developer
- Hands on course
- Ask ask ask
- Lots to talk about
- Ask for more details.

- Xamarin intro
- How to Xamarin
- Cross platform
- (Xamarin.Forms)
- MVVMCross
- iOS + Android (+WP) examples

Xamarin FAQ

- Founded in May 2011
- (now) Owned by Microsoft
- 350+ employees*
- 15,000+ customers*
- Used by 1,400,000+ developers*
- C# Development for iOS and Android (and WP)

*https://www.xamarin.com/about

IDEs

Visual Studio®

Why C#

- You already have:
  - skills
  - tools
  - code
  - Framework
- Multi purpose language (unlike objective-C / swift)

- Generics, Linq, Async and the future…

What xamarin is not

- Xamarin is only part of the solution
  - Learn: Windows Phone
  - Learn: Android
  - Learn: iOS

General advice:

You do have to learn the platform lifecycles and know the frameworks!

Under the hood: Xamarin bindings

```
// @interface MobilePayCancelledPayment : NSObject
[BaseType (typeof(NSObject))]
interface MobilePayCancelledPayment
{
        // @property (readonly, nonatomic, strong) NSString * orderId;
        [Export ("orderId", ArgumentSemantic.Strong)]
        string OrderId { get; }

        // -(instancetype)initWithOrderId:(NSString *)orderId;
        [Export ("initWithOrderId:")]
        IntPtr Constructor (string orderId);
}

// typedef void (^MobilePayPaymentErrorBlock)(NSError * _Nonnull);
delegate void MobilePayPaymentErrorBlock (NSError arg0);

// typedef void (^MobilePayCallbackSuccessBlock)(NSString * _Nullable, NSString * _Nullable, NSString * _Nullable);
delegate void MobilePayCallbackSuccessBlock ([NullAllowed] string arg0, [NullAllowed] string arg1, [NullAllowed] string

// typedef void (^MobilePayCallbackErrorBlock)(NSString * _Nullable, int, NSString * _Nullable);
delegate void MobilePayCallbackErrorBlock ([NullAllowed] string arg0, int arg1, [NullAllowed] string arg2);

// typedef void (^MobilePayCallbackCancelBlock)(NSString * _Nullable);
delegate void MobilePayCallbackCancelBlock ([NullAllowed] string arg0);

// typedef void (^MobilePayPaymentSuccessBlock)(MobilePaySuccessfulPayment * _Nullable);
delegate void MobilePayPaymentSuccessBlock ([NullAllowed] MobilePaySuccessfulPayment arg0);
```

Pricing

Build C# apps on Android, iOS, Windows, and Mac with Xamarin.

Xamarin Studio

Xamarin Studio Community

FREE

A free, full-featured IDE for Mac users to create Android and iOS apps using Xamarin.

- Students
- OSS development
- Small teams

Download

Visual Studio

Visual Studio Community

FREE

A free, full-featured and extensible IDE for Windows users to create Android and iOS apps with Xamarin, as well as Windows apps, web apps, and cloud services.

- Includes Xamarin SDK
- OSS development
- Small teams

Download VS

Visual Studio Professional

Contact us for quote

Professional developer tools and services for individual developers or small teams.

- Includes Xamarin SDK
- No usage restrictions
- Access to additional Xamarin University content
- Additional subscriber benefits

Get VS Professional

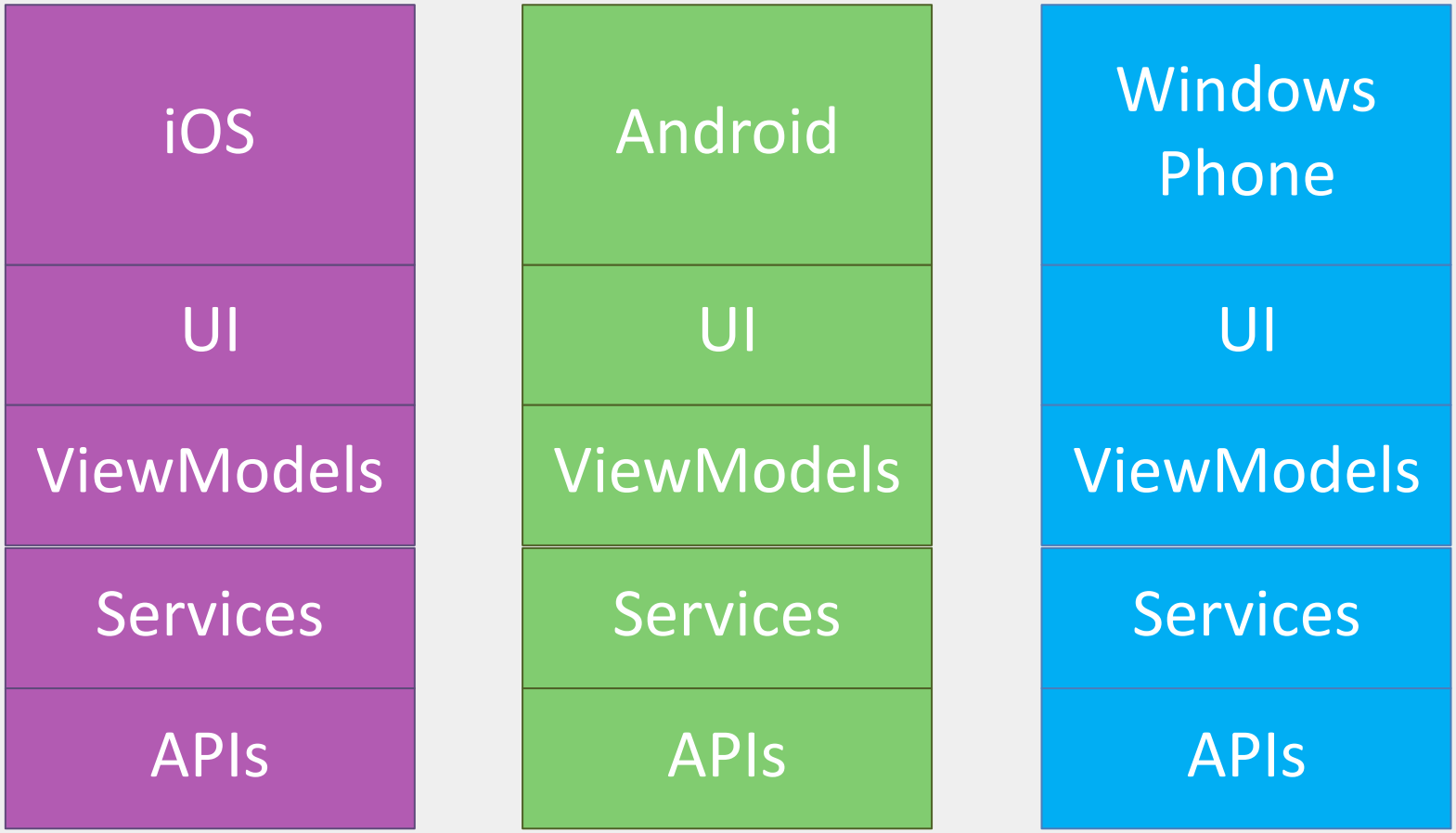
Visual Studio Enterprise

Contact us for quote

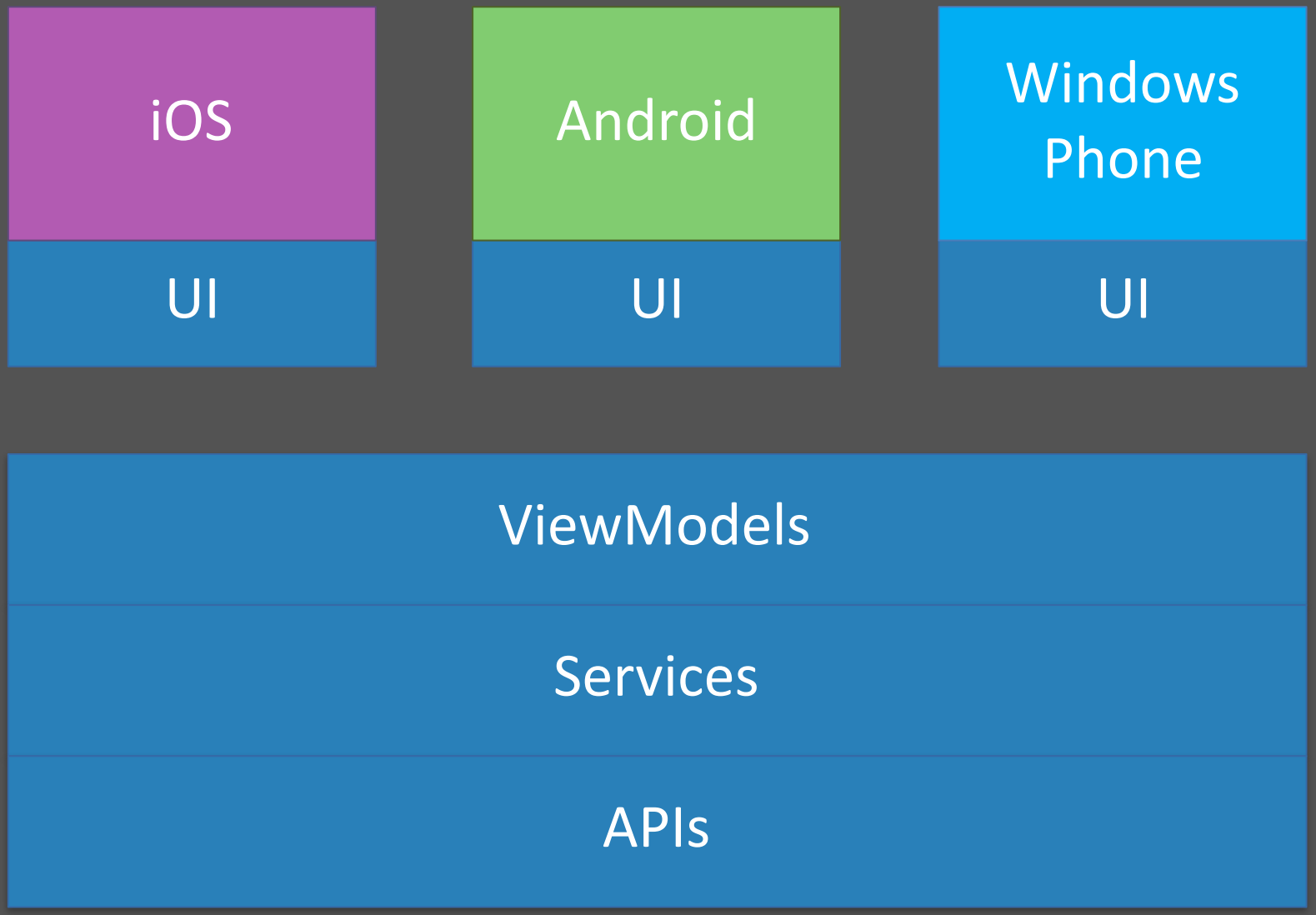
End-to-end solution for teams of any size with demanding quality and scale needs

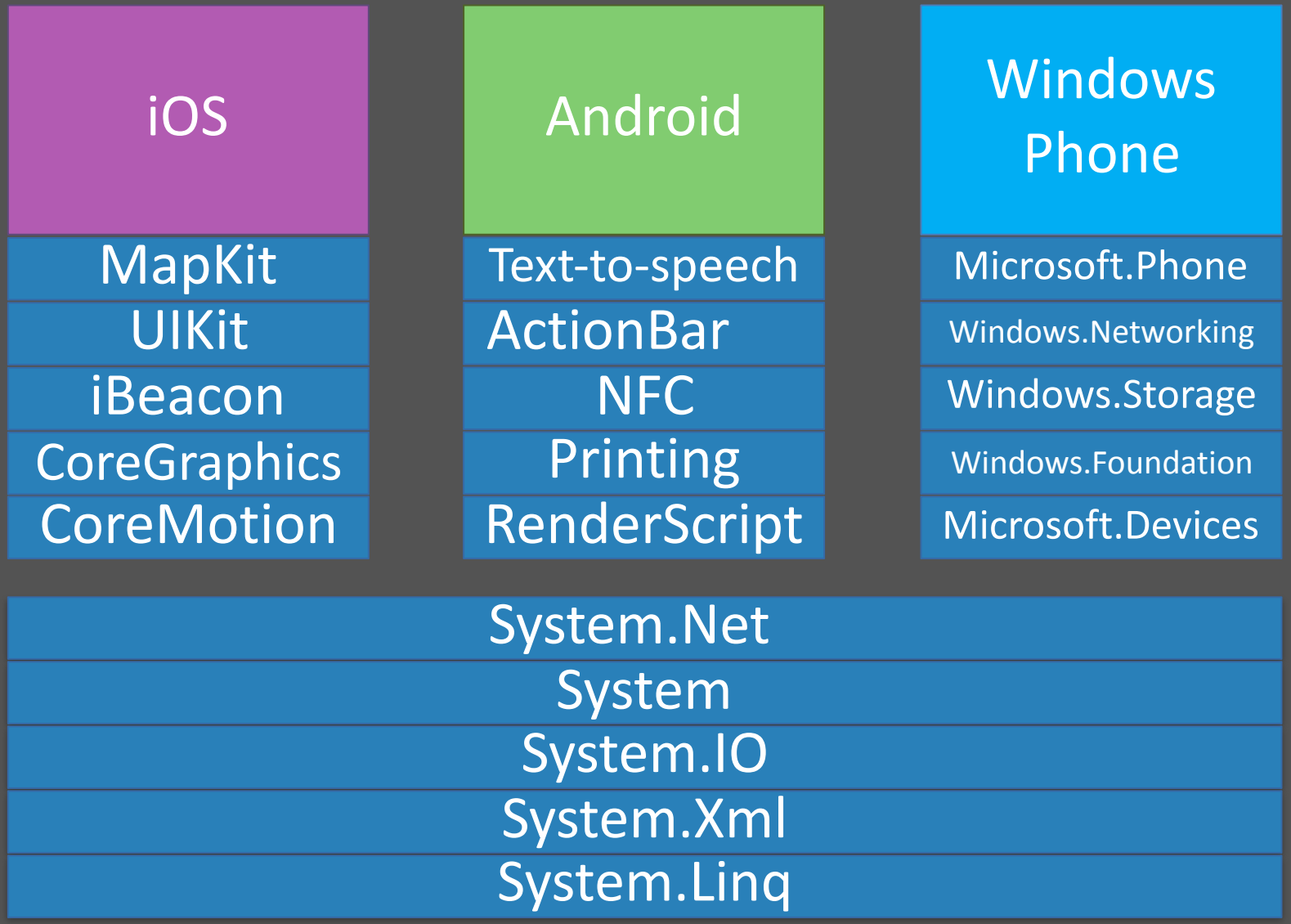
- Includes Xamarin SDK
- Enterprise capabilities
- 25% Xamarin Test Cloud discount
- Additional subscriber benefits

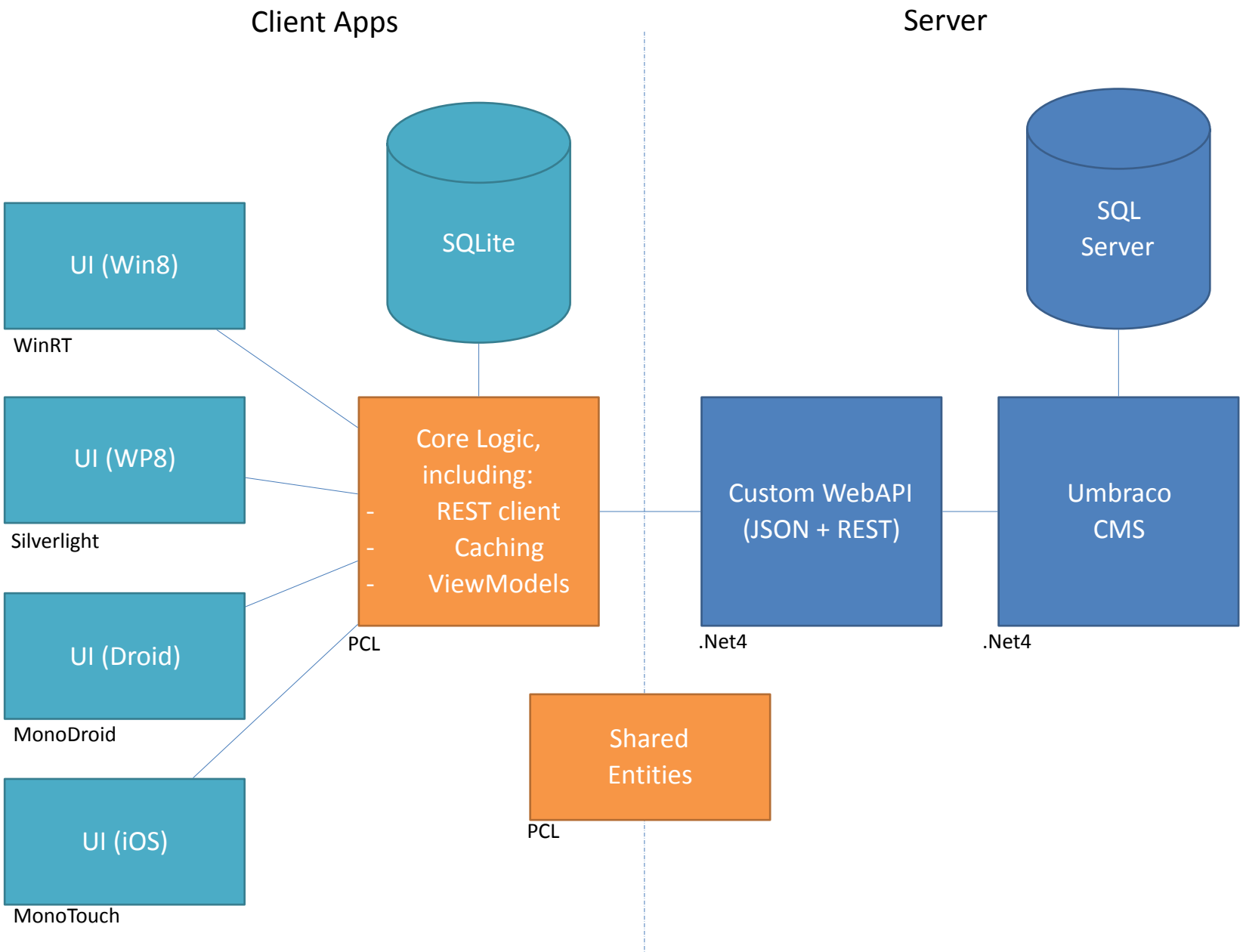
Get VS Enterprise


iOS
UI
ViewModels
Services
APIs
Android
UI
ViewModels
Services
APIs
Windows Phone
UI
ViewModels
Services
APIs

Xamarin

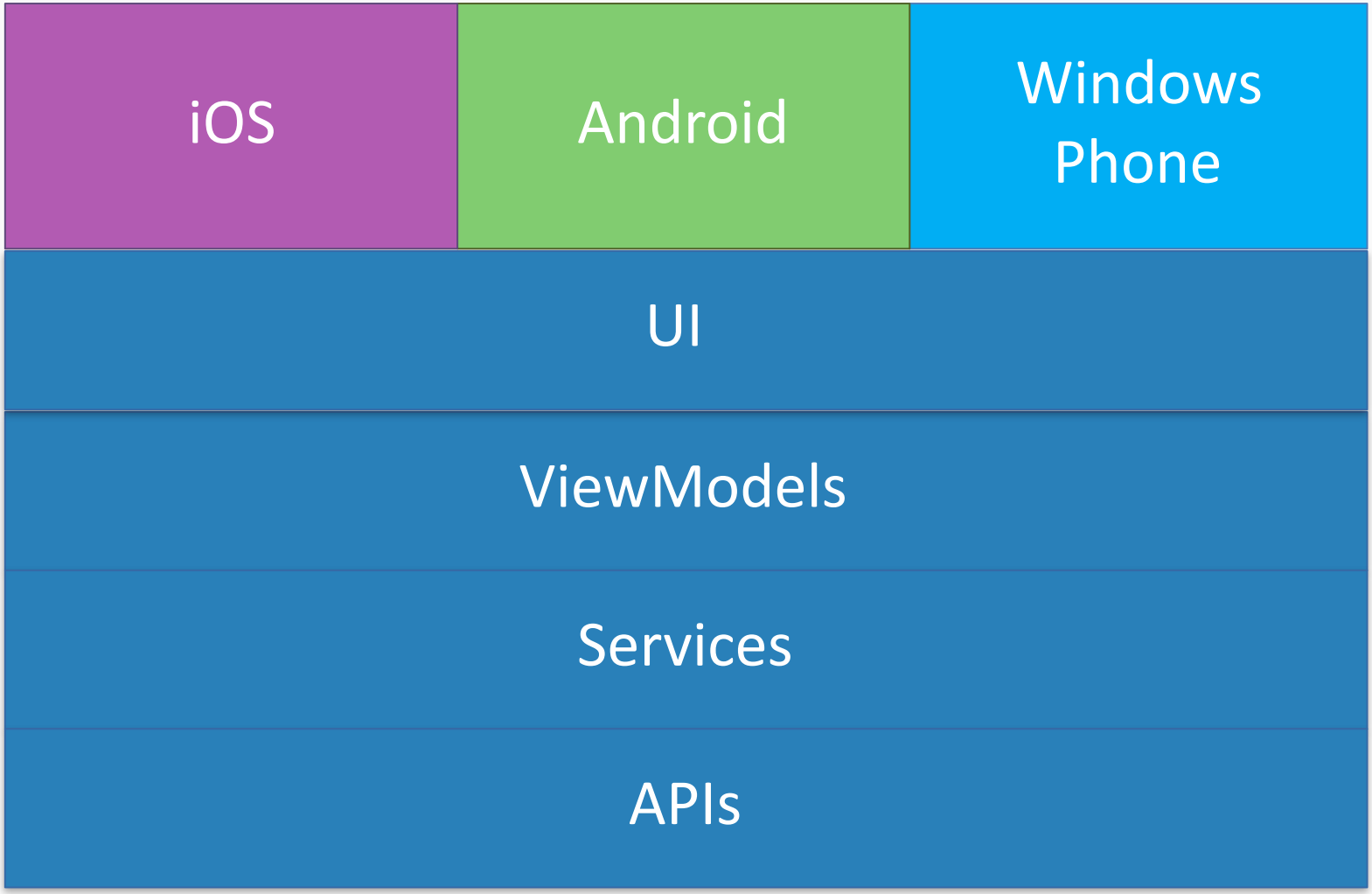

iOS
Android
Windows Phone
UI
UI
UI
ViewModels
Services
APIs


iOS
Android
Windows Phone
MapKit
UIKit
iBeacon
CoreGraphics
CoreMotion
Text-to-speech
ActionBar
NFC
Printing
RenderScript
Microsoft.Phone
Windows.Networking
Windows.Storage
Windows.Foundation
Microsoft.Devices
System.Net
System
System.IO
System.Xml
System.Linq


Client Apps
Server
SQLite
SQL Server
UI (Win8)
WinRT
UI (WP8)
Silverlight
UI (Droid)
MonoDroid
UI (iOS)
MonoTouch
Core Logic, including:
- REST client
- Caching
- ViewModels
PCL
Custom WebAPI (JSON + REST)
.Net4
Umbraco CMS
.Net4
Shared Entities
PCL

DEMO: Hello World (iOS, Android + PCL)

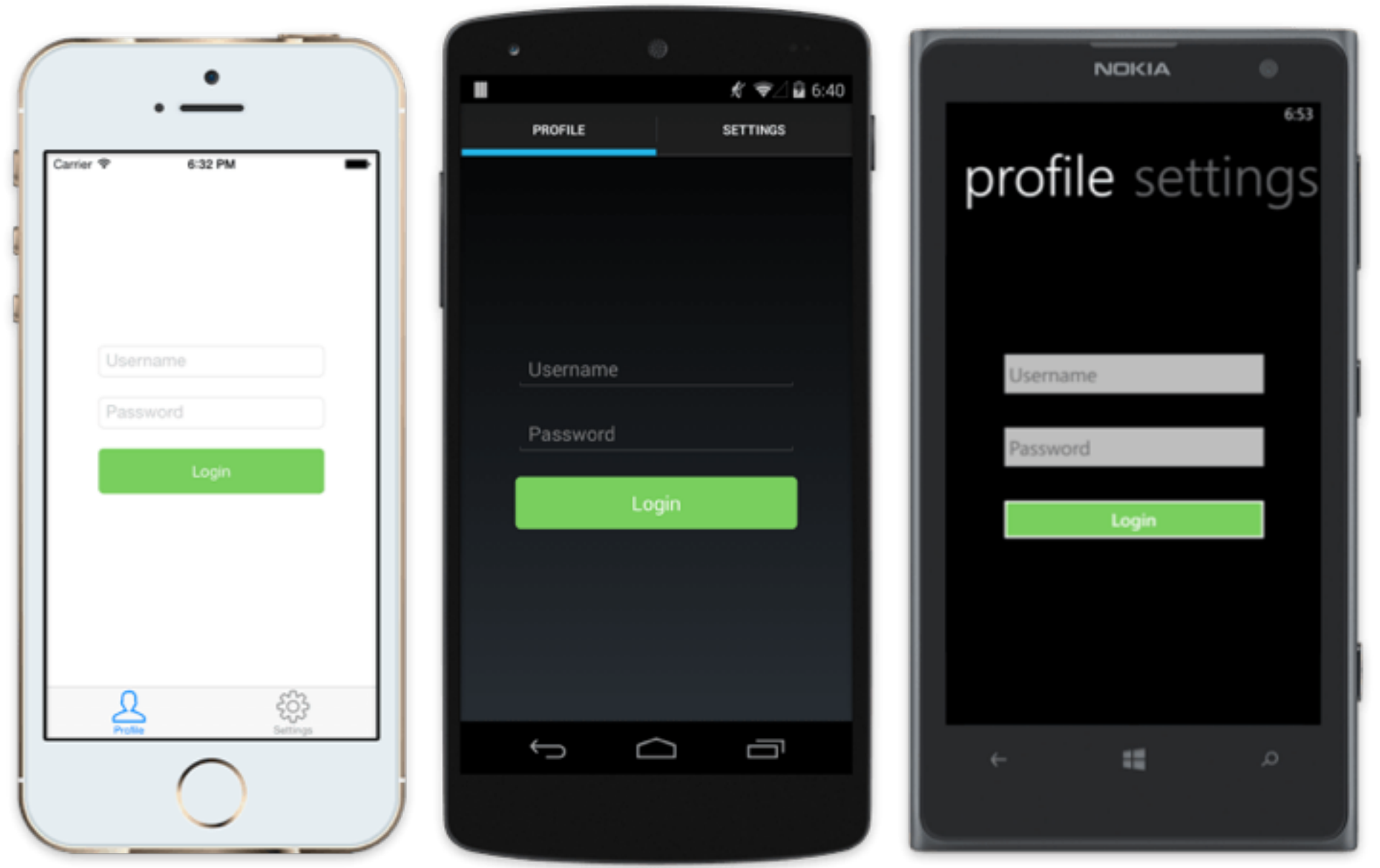
Xamarin.Forms



Xamarin.Forms

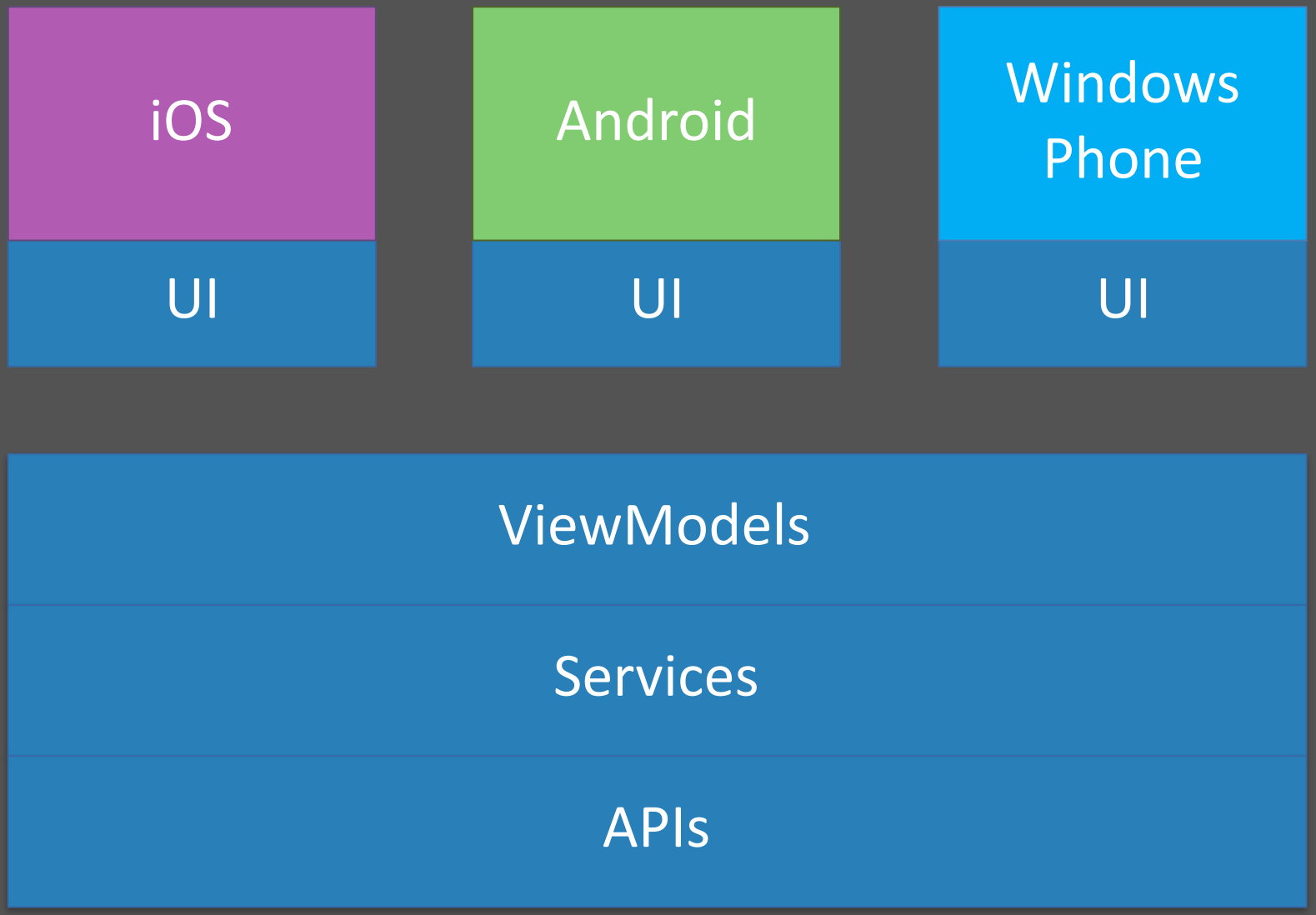
```
using Xamarin.Forms;

var profilePage = new ContentPage {
    Title = "Profile",
    Icon = "Profile.png",
    Content = new StackLayout {
        Spacing = 20, Padding = 50,
        VerticalOptions = LayoutOptions.Center,
        Children = {
            new Entry { Placeholder = "Username" },
            new Entry { Placeholder = "Password", IsPassword = true },
            new Button {
                Text = "Login",
                TextColor = Color.White,
                BackgroundColor = Color.FromHex("77D065") }}}
};

var settingsPage = new ContentPage {
    Title = "Settings",
    Icon = "Settings.png",
    (...)
};

var mainPage = new TabbedPage { Children = { profilePage, settingsPage } };
```

Xamarin.Forms

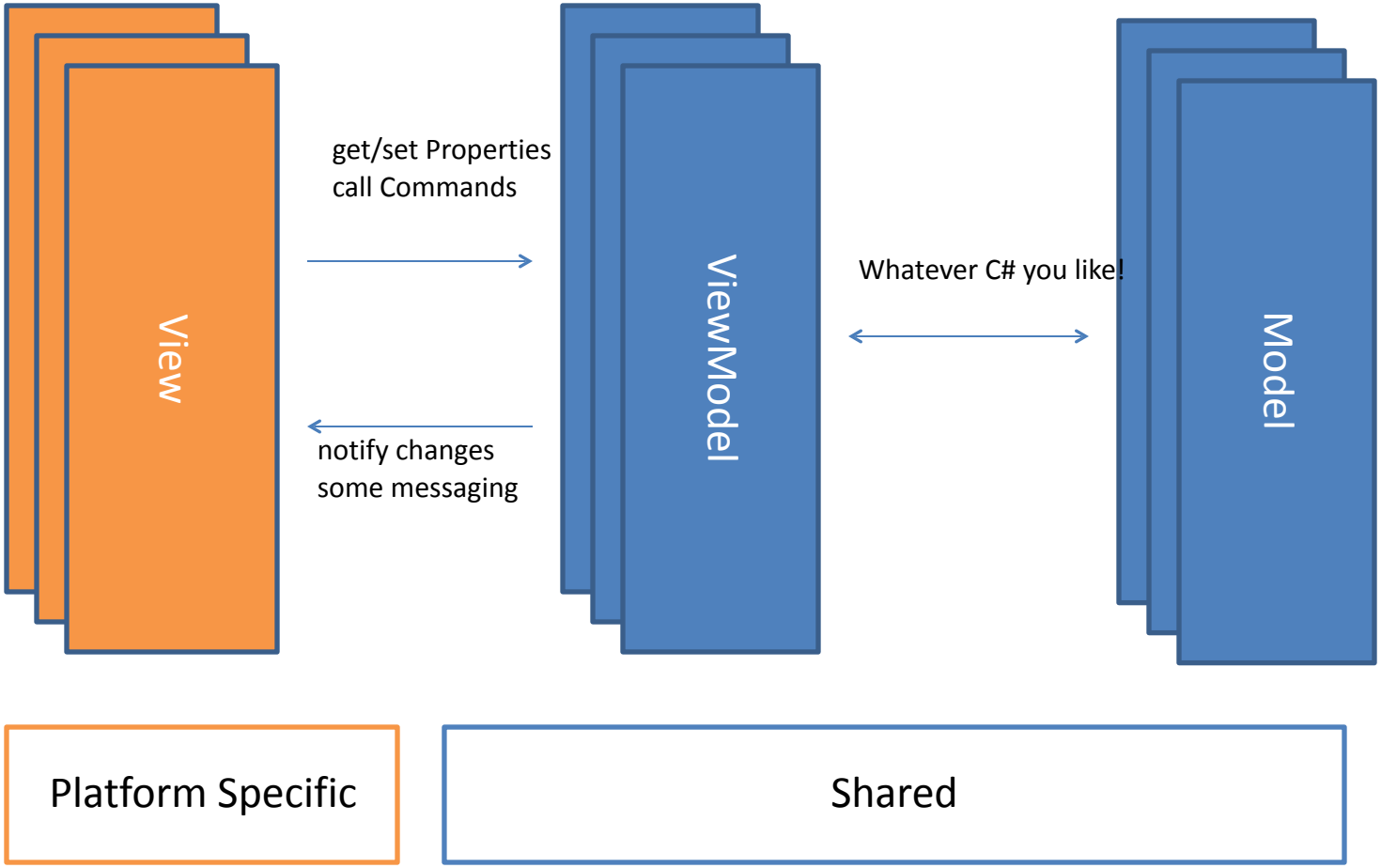


Code Sharing (MVVM)

Xamarin


iOS
Android
Windows Phone
UI
UI
UI
ViewModels
Services
APIs

MVVM


View
get/set Properties
call Commands
notify changes
some messaging
ViewModel
Whatever C# you like!
Model
Platform Specific
Shared

MVVM Technical Details

- Properties
- INotifyPropertyChanged
- INotifyCollectionChanged
- Data Binding
- IValueConverter
- ICommand

MVVMCross Technical Details

- IOC Container
- Code Injection
- Centralised navigation logic
- Common view bindings
- Plugins

MVVMCross Demo
(Starter pack)

“Pages” in MVVMCross

- Android: Activity (MvxActivity)
- iOS: UIViewController (MvxViewController)
- Windows: Page (MvxPhonePage/ MvxStorePage)

- MVVMCross: MvxViewModel

IMvxAppStart

```
namespace Cirrious.MvvmCross.ViewModels
{
    public interface IMvxAppStart
    {
        void Start(object hint = null);
    }
}
```

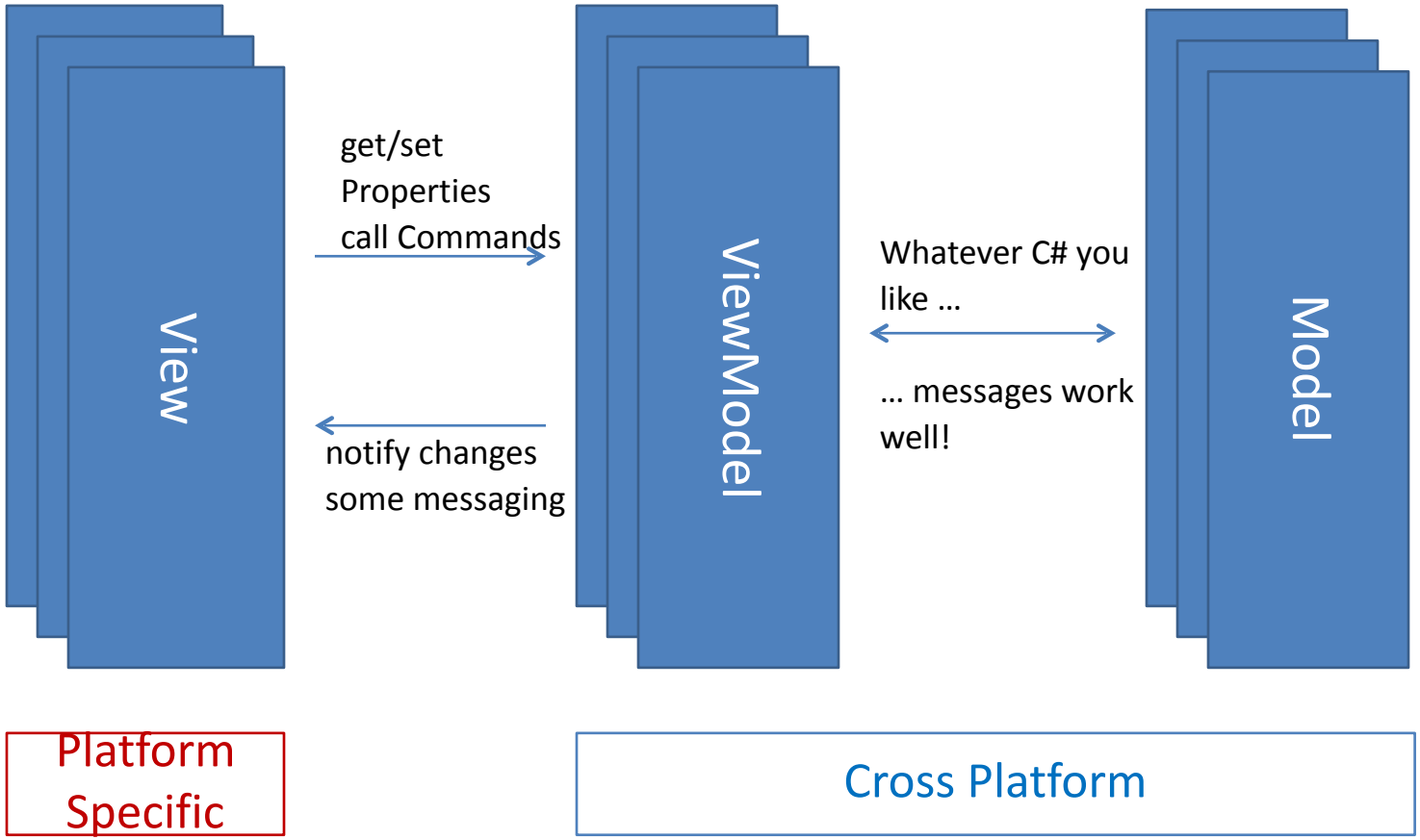
Simple Starts

```
public class App : Cirrious.MvvmCross.ViewModels.MvxApplication
{
    public override void Initialize()
    {
        CreatableTypes()
            .EndingWith("Service")
            .AsInterfaces()
            .RegisterAsLazySingleton();

        RegisterAppStart<ViewModels.FirstViewModel>();
    }
}
```

Conditional Starts

```
public class CustomStart
    : MvxNavigatingObject, IMvxAppStart
{
    public void Start(object hint = null)
    {
        var appSettings = Mvx.Resolve<IAppSettings>();

        if (!appSettings.Authenticated)
            ShowViewModel<LoginViewModel>();
        else
            ShowViewModel<HomeViewModel>();
    }
}
```

Data binding


View
get/set
Properties
call Commands
notify changes
some messaging
ViewModel
Whatever C# you
like …
… messages work
well!
Model
Platform
Specific
Cross Platform

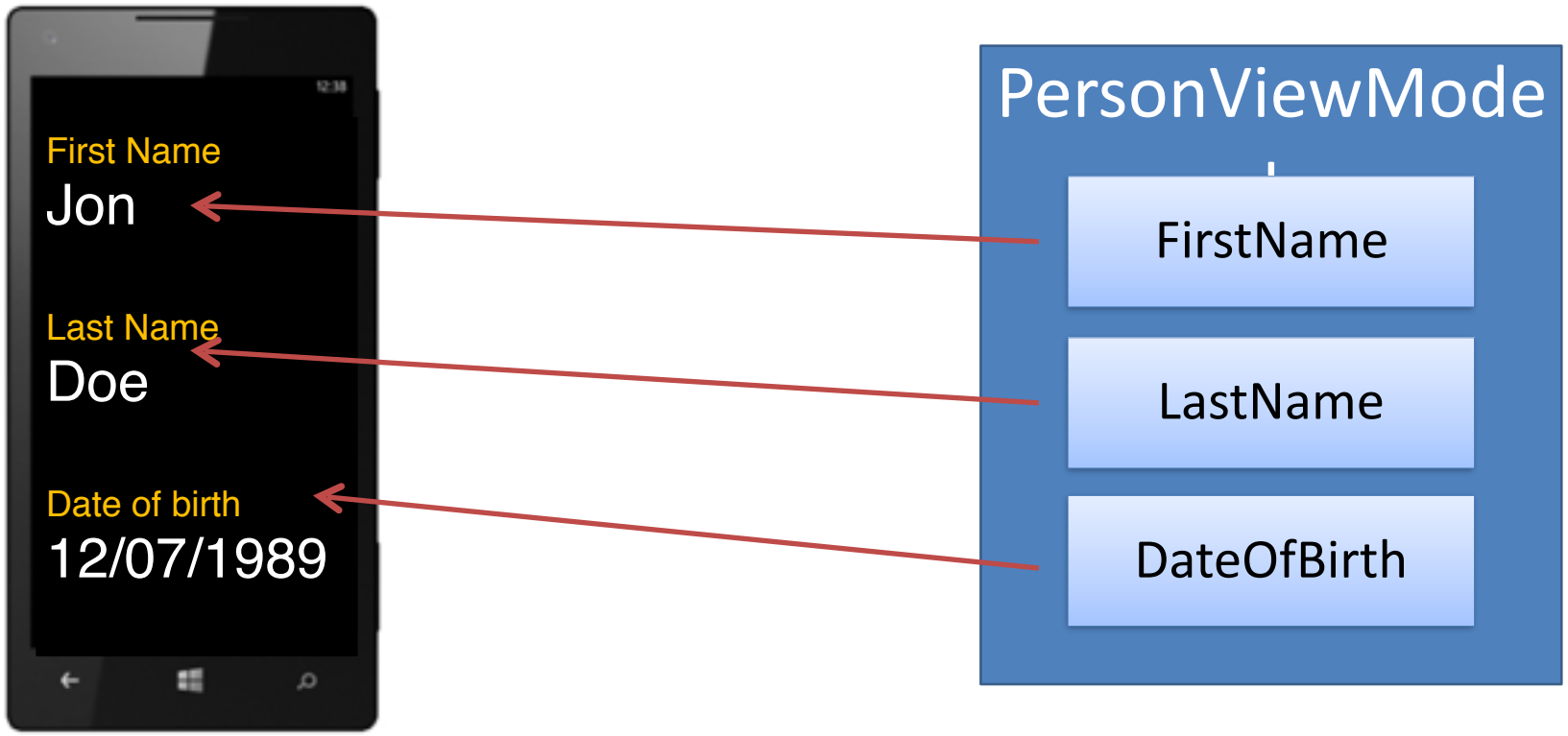
INotifyPropertyChanged

```
namespace System.ComponentModel
{
  public interface INotifyPropertyChanged
  {
    event PropertyChangedEventHandler PropertyChanged;
  }
}
```

Typical ViewModel Property

```
private double _lng;
public double Lng
{
    get { return _lng; }
    set { _lng = value; RaisePropertyChanged(() => Lng); }
}
```

One Way Binding



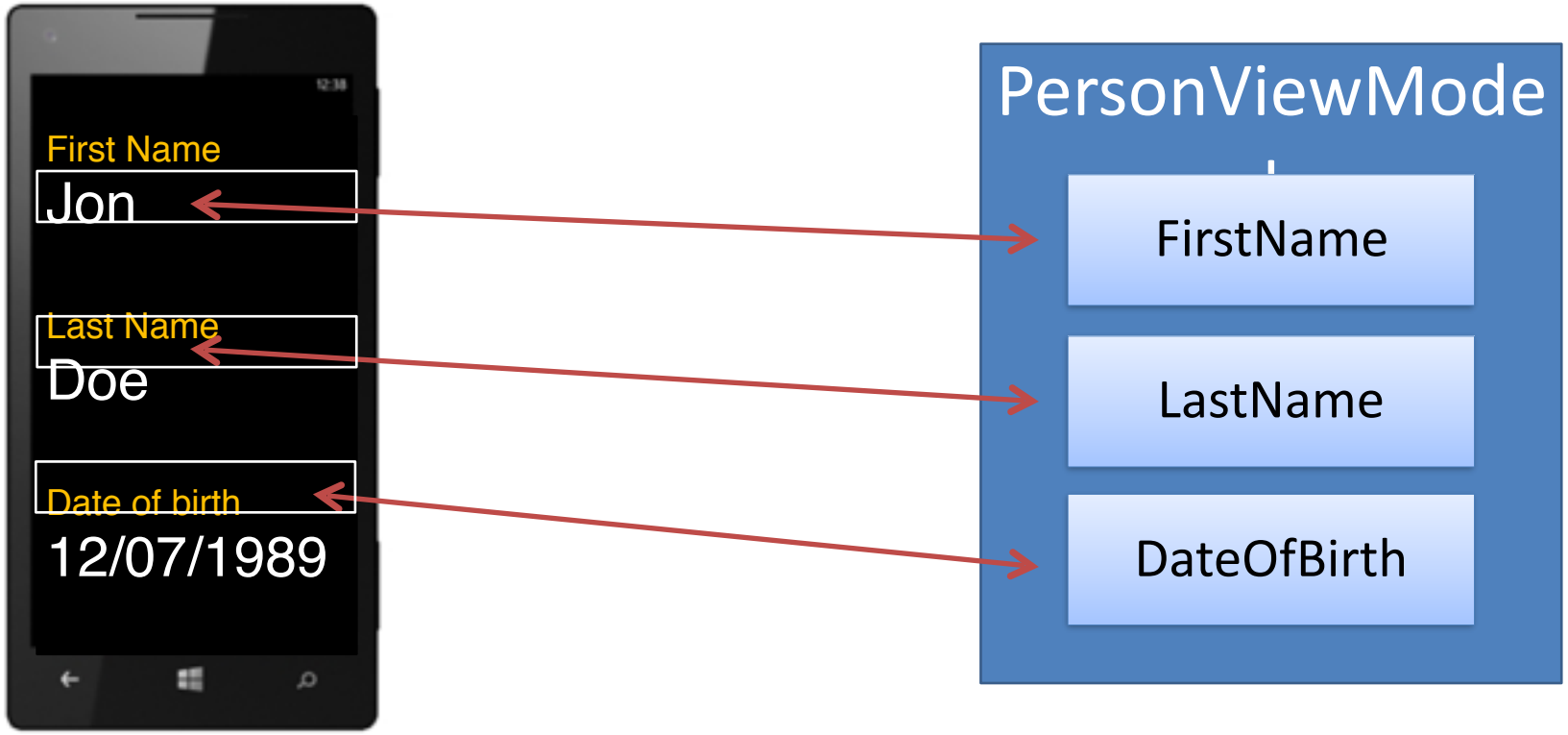
UI Syntax

```
<TextBlock
    Text="{Binding FirstName}"  />
```

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    local:MvxBind="Text FirstName" />
```

```
var label = new UILabel();
Add(label);
this.CreateBinding(label)
    .To<FirstViewModel>(vm => vm.FirstName)
    .Apply();
```

Two Way Binding



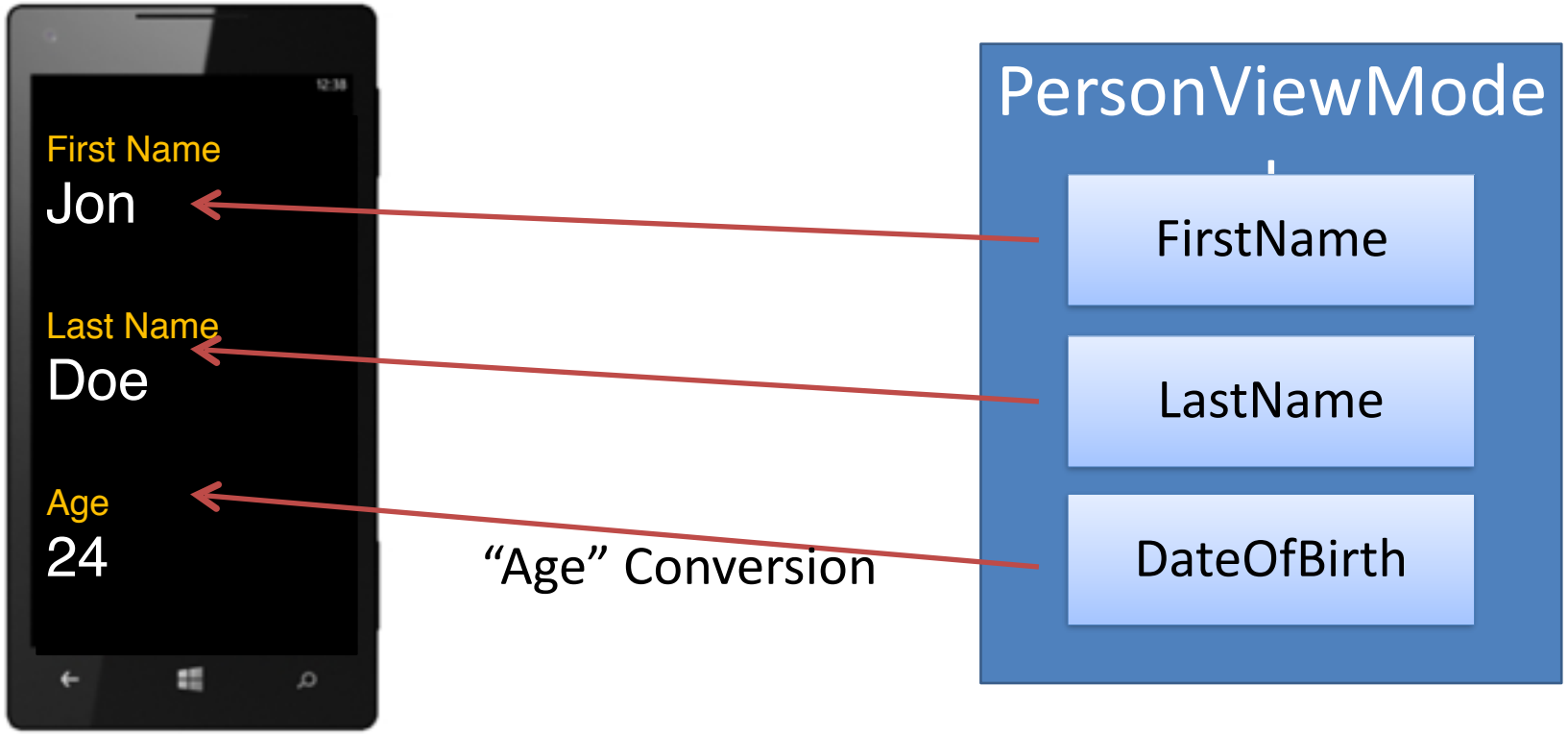
UI Syntax

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

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    local:MvxBind="Text FirstName" />
```

```
var label = new UILabel();
Add(label);
this.CreateBinding(label)
    .To<FirstViewModel>(vm => vm.FirstName)
    .Apply();
```

Value Conversion



IMvxValueConverter

```
namespace Cirrious.CrossCore.Converters
{
    public interface IMvxValueConverter
    {
        object Convert(
            object value,
            Type targetType,
            object parameter,
            CultureInfo culture);
        object ConvertBack(
            object value,
            Type targetType,
            object parameter,
            CultureInfo culture);
    }
}
```

AgeValueConverter

```
public class AgeValueConverter
    : MvxValueConverter<DateTime, int>
{
    protected override int Convert(DateTime value, Type targetType,
        object parameter, System.Globalization.CultureInfo culture)
    {
        var now = DateTime.Now;
        var years = now.Year - value.Year;
        if (now.Month > value.Month)
            years++;
        else if (now.Month == value.Month && now.Day >= value.Day)
            years++;
        return years;
    }
}
```

UI Syntax

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    local:MvxBind="Text DateOfBirth, Converter=Age" />
```

```
var label = new UILabel();
Add(label);
this.CreateBinding(label)
    .To<FirstViewModel>(vm => vm.DateOfBirth)
    .WithConversion("Age")
    .Apply();
```

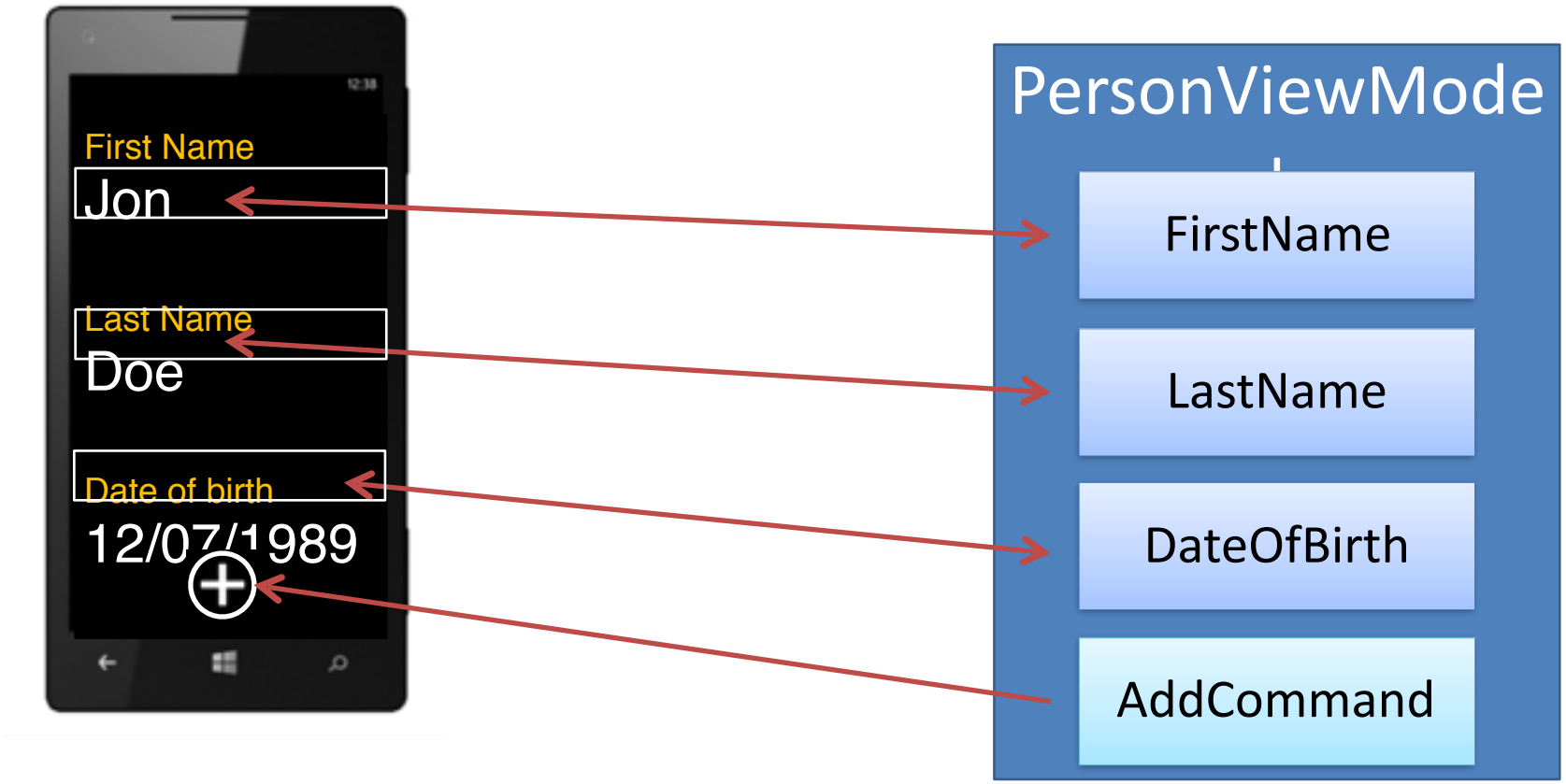
UI Syntax

```
public class NativeAgeValueConverter
    : MvxNativeValueConverter<AgeValueConverter>
{
}
```

```
<views:MvxPhonePage.Resources>
    <converters:NativeAgeValueConverter x:Key="Age" />
</views:MvxPhonePage.Resources>
```

```
<TextBlock
    Text="{Binding DateOfBirth, Converter={StaticResource Age}}" />
```

Binding Actions



ICommand

```
namespace System.Windows.Input
{
  public interface ICommand
  {
    bool CanExecute(object parameter);
    void Execute(object parameter);
    event EventHandler CanExecuteChanged;
  }
}
```

Typical ViewModel Command

```
private MvxCommand _addCommand;
public System.Windows.Input.ICommand AddCommand
{
    get
    {
        _addCommand = _addCommand ?? new MvxCommand(DoAdd);
        return _addCommand;
    }
}

private void DoAdd()
{
    // do the add
}
```

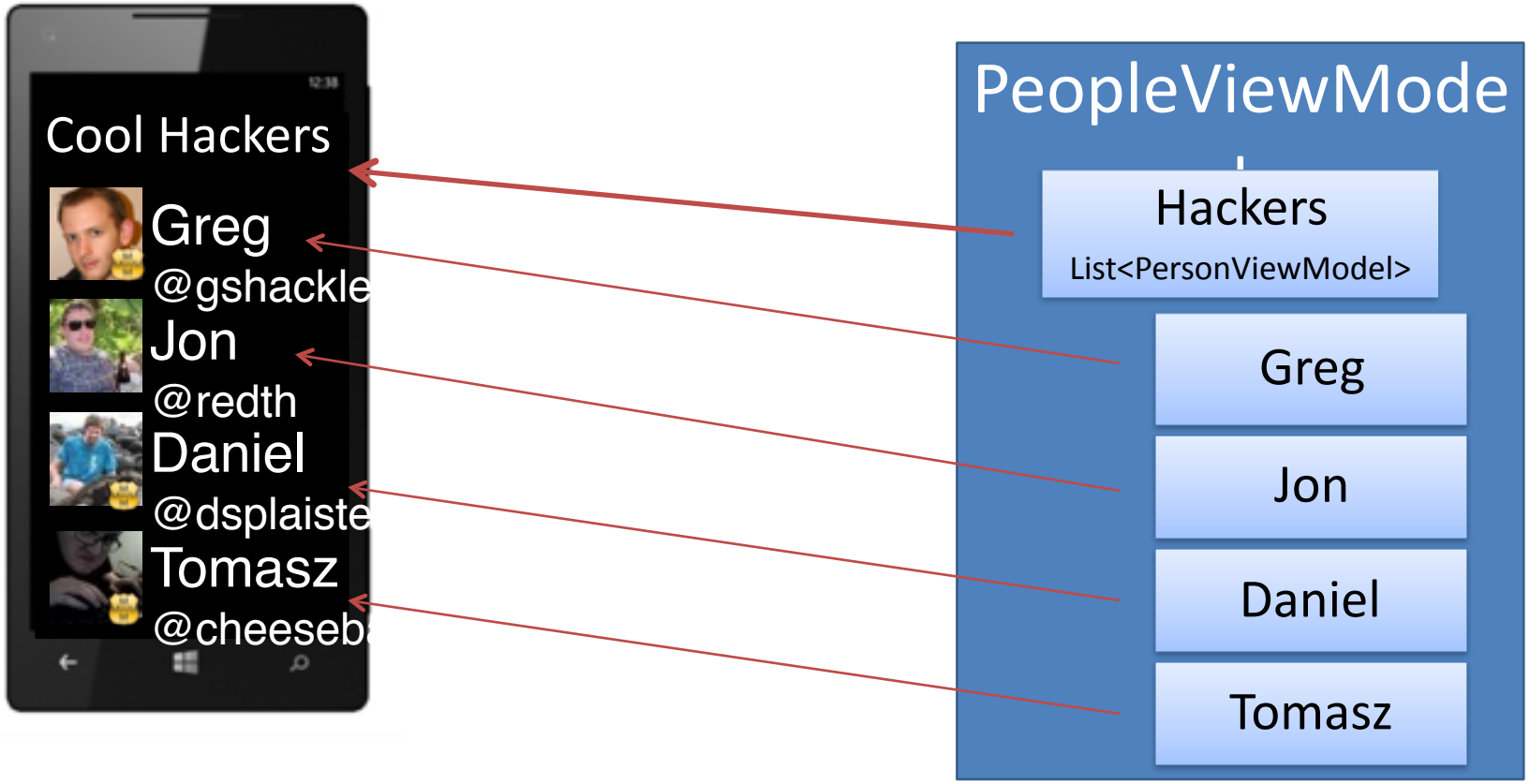
UI Syntax

```
<Button
    Content="Add"
    Command="{Binding AddCommand}" />
```

```
<Button
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:text="Add"
      local:MvxBind="Click AddCommand" />
```

```
var button = new UIButton();
Add(button);
this.CreateBinding(button)
    .To<FirstViewModel>(vm => vm.AddCommand)
    .Apply();
```

Collections



INotifyCollectionChanged

```
namespace System.Collections.Specialized
{
  public interface INotifyCollectionChanged
  {
    event NotifyCollectionChangedEventHandler CollectionChanged;
  }
}
```

ObservableCollection

```
namespace System.Collections.ObjectModel
{
  public class ObservableCollection<T> : Collection<T>, INotifyCollectionChanged, INotifyPropertyChanged
  {
    public ObservableCollection();
    public ObservableCollection(IEnumerable<T> collection);
    protected override void ClearItems();
    protected override void InsertItem(int index, T item);
    protected virtual void OnCollectionChanged(NotifyCollectionChangedEventArgs e);
    protected virtual void OnPropertyChanged(PropertyChangedEventArgs e);
    protected override void RemoveItem(int index);
    protected override void SetItem(int index, T item);
    public event NotifyCollectionChangedEventHandler CollectionChanged;
    protected event PropertyChangedEventHandler PropertyChanged;
    event PropertyChangedEventHandler INotifyPropertyChanged.PropertyChanged;
  }
}
```

ViewModel Collection Property

```
private ObservableCollection<Person> _people;
public ObservableCollection<Person> People
{
    get { return _people; }
    set { _people = value; RaisePropertyChanged(() => People); }
}
```

UI Syntax

```
<ListBox ItemsSource="{Binding People}" >
    <ListBox.ItemTemplate>
        <DataTemplate>
            <StackPanel Orientation="Horizontal">
                <Image Height="75" Width="75" Source="{Binding ImageUrl}" />
                <StackPanel>
                    <TextBlock Text="{Binding Name}" />
                    <TextBlock Text="{Binding Twitter}" />
                </StackPanel>
            </StackPanel>
        </DataTemplate>
    </ListBox.ItemTemplate>
</ListBox>
```

UI Syntax

```
<Mvx.MvxListView
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  local:MvxBind="ItemsSource People"
  local:MvxItemTemplate="@layout/item_person"
  />
```

UI Syntax

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:local="http://schemas.android.com/apk/res-auto"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <Mvx.MvxImageView
        android:layout_width="75dp"
        android:layout_height="75dp"
        local:MvxBind="ImageUrl ImageUrl" />
    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            local:MvxBind="Text Name" />
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            local:MvxBind="Text Twitter" />
    </LinearLayout>
</LinearLayout>
```

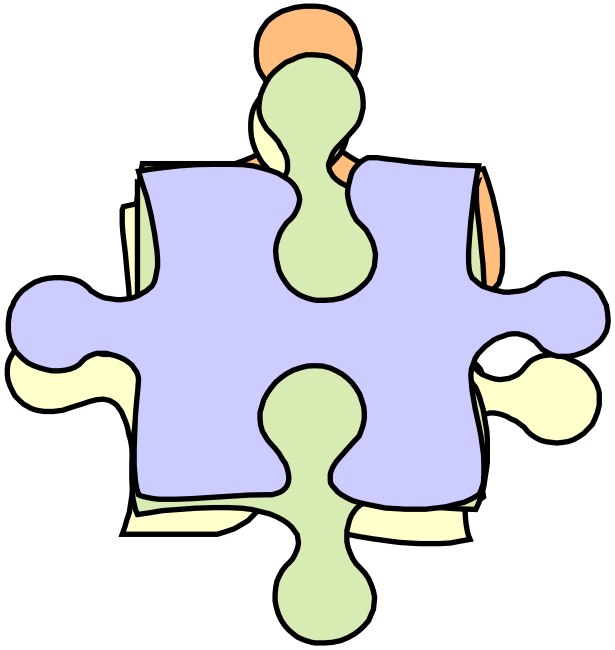
UI Syntax

```
var source = new MvxStandardTableViewSource(
                            TableView,
                            @"TitleText Name;
                              SubTitleText Twitter;
                              ImageUrl ImageUrl");
TableView.Source = source;

var set = this.CreateBindingSet<FirstView, FirstViewModel>();
set.Bind(source).To(vm => vm.People);
set.Apply();

TableView.ReloadData();
```

DEMO
(Pictures)

http://jsonplaceholder.typicode.com/photos?_start=0&_end=30

Mvx.Register<T>



Mvx.Register<T>

- RegisterSingleton

```
// immediate singleton
Mvx.RegisterSingleton<IAppSettings>(new AppSettings());
```

- Lazy - RegisterSingleton

```
// lazy singleton
Mvx.RegisterSingleton<IAppSettings>(() => new AppSettings());
```

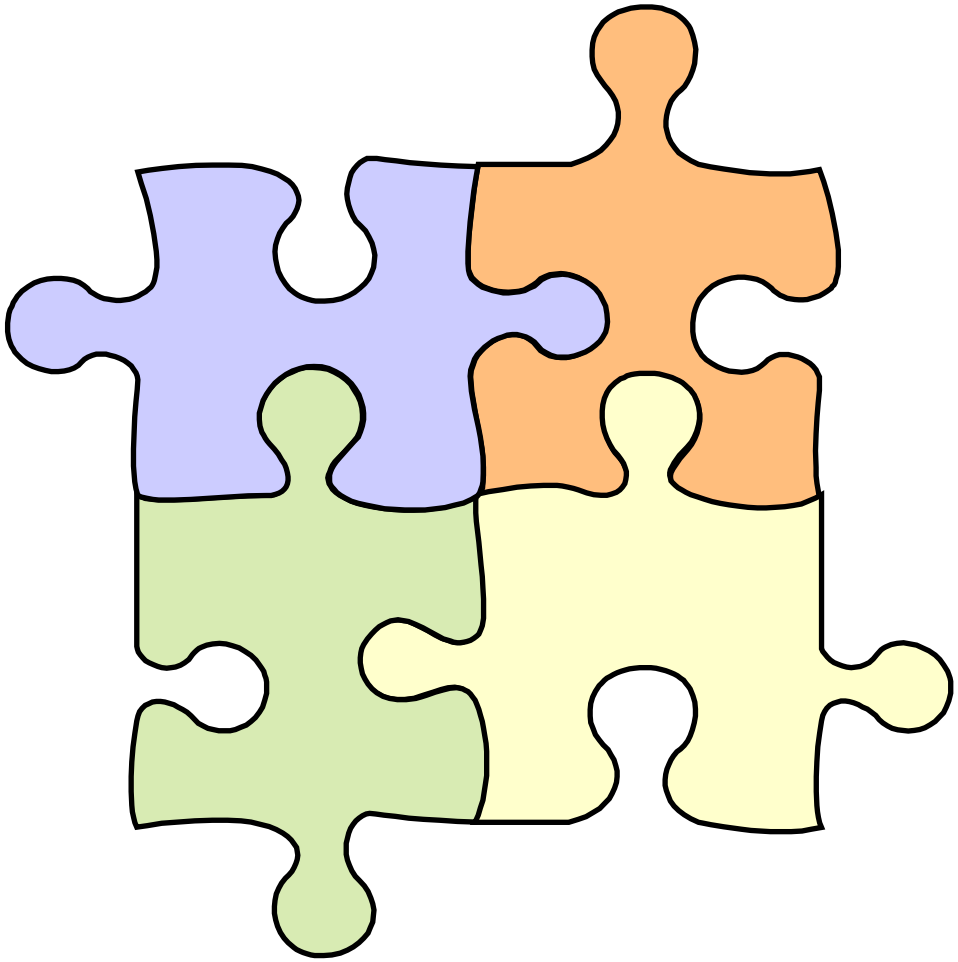
- RegisterType

```
//  instance per Resolve
Mvx.RegisterType<IAppSettings, AppSettings>();
```

Automatic Registration

```
public class App : Cirrious.MvvmCross.ViewModels.MvxApplication
{
    public override void Initialize()
    {
        CreatableTypes()
            .EndingWith("Service")
            .AsInterfaces()
            .RegisterAsLazySingleton();

        RegisterAppStart<ViewModels.FirstViewModel>();
    }
}
```

Mvx.Resolve<T>



Mvx.Resolve<T>

- Resolve

```
var settings = Mvx.Resolve<IAppSettings>();
```

- CanResolve

```
var exists = Mvx.CanResolve<IAppSettings>();
```

- TryResolve

```
var success = Mvx.TryResolve<IAppSettings>(out settings);
```

Mvx Construction

- Constructor resolution

```
public class Foo
{
    private readonly IAppSettings _appSettings;

    public Foo(IAppSettings appSettings)
    {
        _appSettings = appSettings;
    }
```

- IoCConstruct

```
// create a Foo, resolving all constructor parameters
var foo = Mvx.IocConstruct<Foo>();
```

DEMO
(Alert)