Good Afternoon

It’s always good to start off with an excuse.

Yesterday, I did a 4 hour workshop in a room that was rather warm.

When I woke up today, my voice was about half of normal.

And while it’s better this afternoon …

I can’t guarantee it will last the entire hour.

So I hope you understand
and that these cute pictures
of my dogs make up for it.

A Lap Around Xamarin

Douglas Starnes
June 2, 2017
Music City Code

Why? What? Who? When? Where?

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Native .NET binding to iOS and Android APIs

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Native .NET binding to iOS and Android APIs

Can consume existing native libraries (Objective-C, Swift, Java)

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Native .NET binding to iOS and Android APIs

Can consume existing native libraries (Objective-C, Swift, Java)

Up to 70% code reuse across platforms

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Native .NET binding to iOS and Android APIs

Can consume existing native libraries (Objective-C, Swift, Java)

Up to 70% code reuse across platforms

Also supports macOS, watchOS, Android Wear, tvOS

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Native .NET binding to iOS and Android APIs

Can consume existing native libraries (Objective-C, Swift, Java)

Up to 70% code reuse across platforms

Also supports macOS, watchOS, Android Wear, tvOS

Share code with existing .NET apps

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Native .NET binding to iOS and Android APIs

Can consume existing native libraries (Objective-C, Swift, Java)

Up to 70% code reuse across platforms

Also supports macOS, watchOS, Android Wear, tvOS

Share code with existing .NET apps

Purchased by Microsoft in March 2016

Why? What? Who? When? Where?

Pronounced ‘Zamarin’ (think ‘xylophone’)

Native .NET binding to iOS and Android APIs

Can consume existing native libraries (Objective-C, Swift, Java)

Up to 70% code reuse across platforms

Also supports macOS, watchOS, Android Wear, tvOS

Share code with existing .NET apps

Purchased by Microsoft in March 2016

//build 2016

Tooling

Tooling

Xamarin Studio for macOS

Tooling

Xamarin Studio for macOS

Supports iOS, Android, macOS

Tooling

Xamarin Studio for macOS

- Supports iOS, Android, macOS
- Free Xamarin Studio Community Edition for all

Tooling

Xamarin Studio for macOS

- Supports iOS, Android, macOS
- Free Xamarin Studio Community Edition for all

Visual Studio for Windows

Tooling

Xamarin Studio for macOS

- Supports iOS, Android, macOS
- Free Xamarin Studio Community Edition for all

Visual Studio for Windows

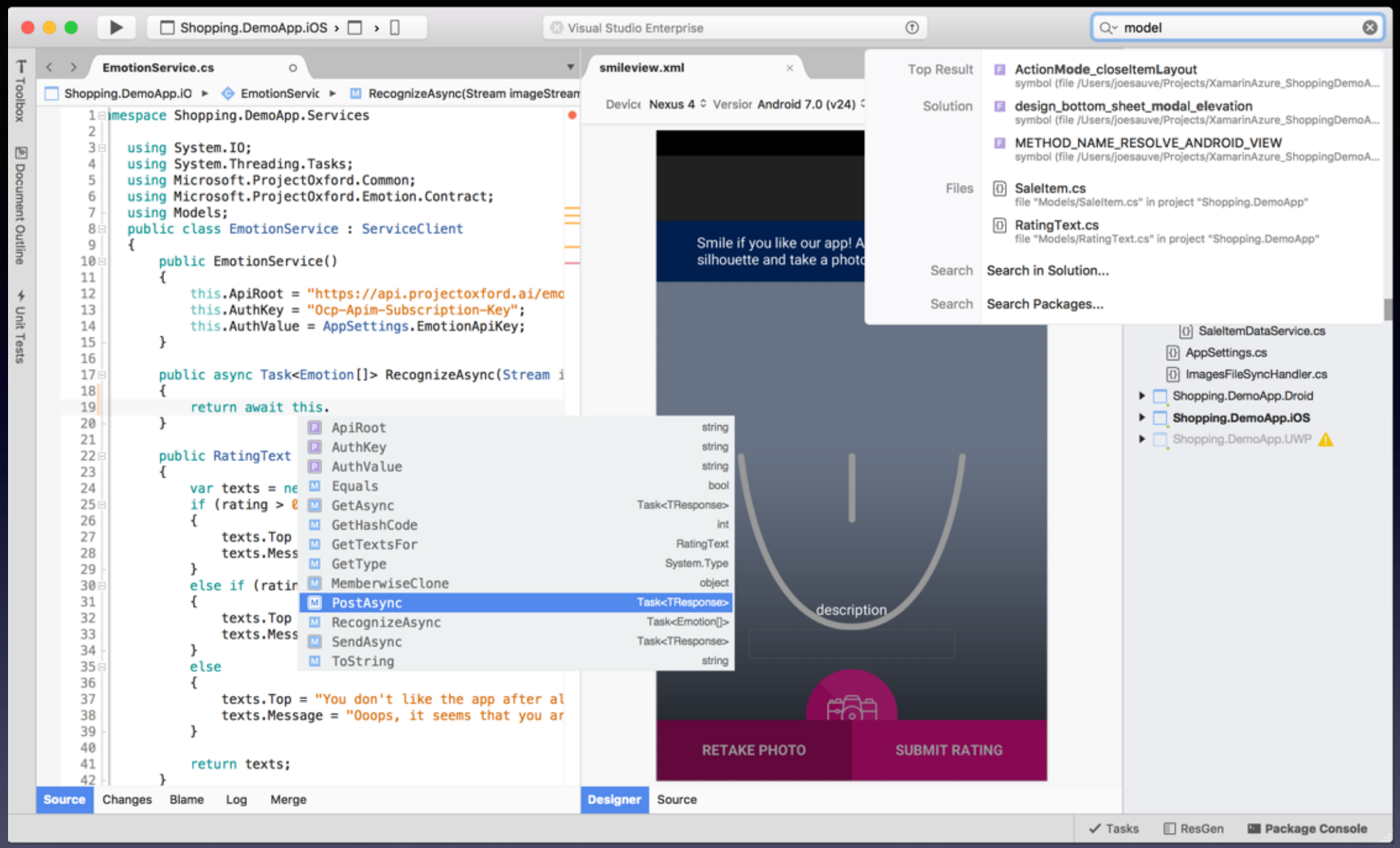
- Supports iOS, Android, Windows Phone

Tooling

- Xamarin Studio for macOS
  - Supports iOS, Android, macOS
  - Free Xamarin Studio Community Edition for all
- Visual Studio for Windows
  - Supports iOS, Android, Windows Phone
  - Must still have a Mac to remotely compile iOS apps

Tooling

- Xamarin Studio for macOS
  - Supports iOS, Android, macOS
  - Free Xamarin Studio Community Edition for all
- Visual Studio for Windows
  - Supports iOS, Android, Windows Phone
  - Must still have a Mac to remotely compile iOS apps
  - All SKUs supported at no extra charge including free Community Edition



Visual Studio for Mac

(yes, for Mac!)

```
class ViewController: UIViewController {
    var data = ["iOS", "Android", "Xamarin", "UWP", "Ionic", "NativeScript", “PhoneGap"]
    // lifecycle events here
}

extension ViewController: UITableViewDataSource {
    func numberOfSections(in tableView: UITableView) -> Int {
        return 1
    }

    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return data.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let cell = tableView.dequeueReusableCell(withIdentifier: "cellId", for: indexPath)
        cell.textLabel?.text = data[indexPath.row]
        return cell
    }
}
```

```
class ViewController: UIViewController {
    var data = ["iOS", "Android", "Xamarin", "UWP", "Ionic", "NativeScript", “PhoneGap"]
    // lifecycle events here
}

extension ViewController: UITableViewDataSource {
    func numberOfSections(in tableView: UITableView) -> Int {
        return 1
    }

    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return data.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let cell = tableView.dequeueReusableCell(withIdentifier: "cellId", for: indexPath)
        cell.textLabel?.text = data[indexPath.row]
        return cell
    }
}
```

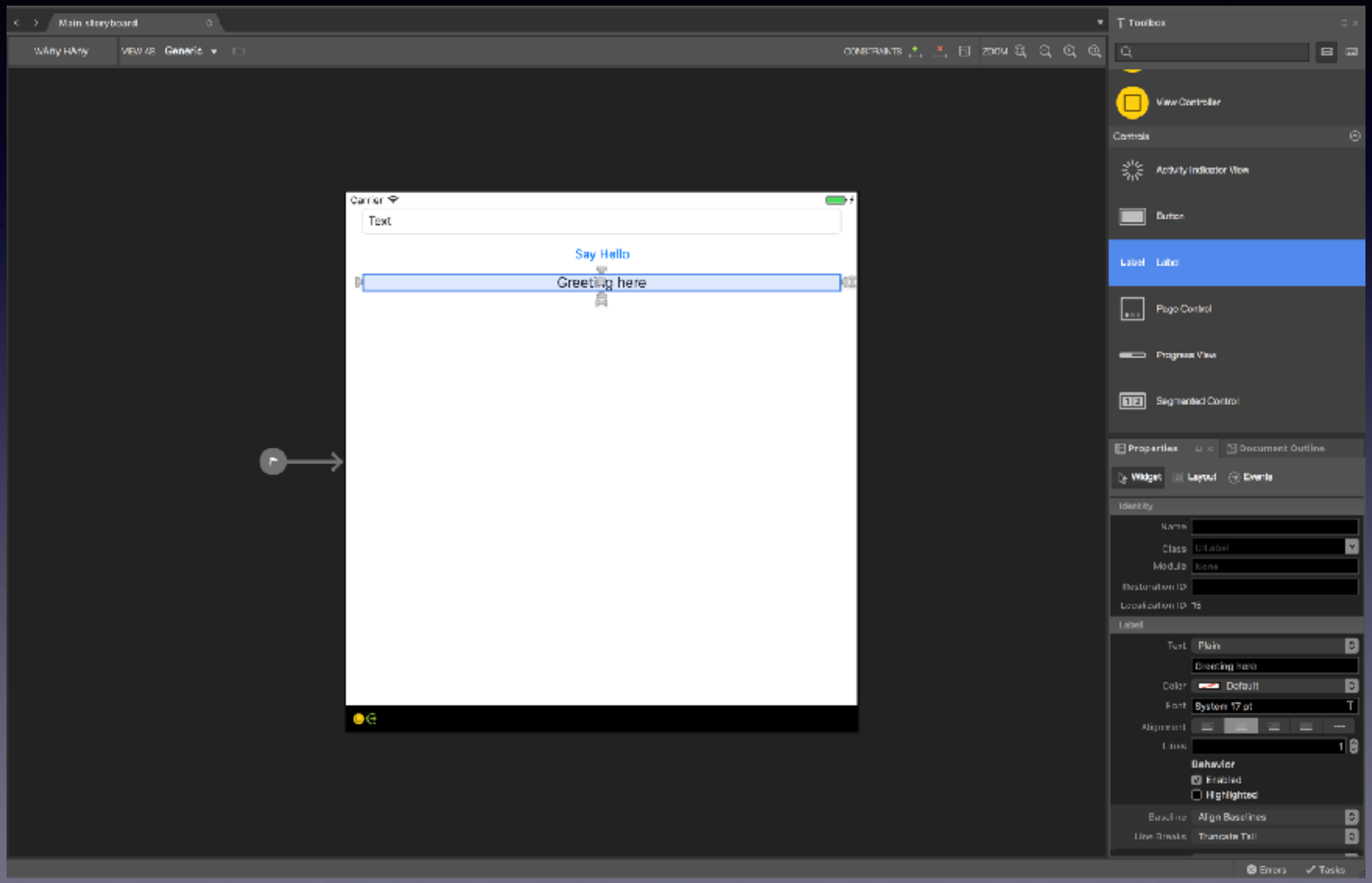
```
public partial class ViewController : UIViewController
{
  public override void ViewDidLoad()
  {
    base.ViewDidLoad();
    // Perform any additional setup after loading the view, typically from a nib.
    tableView.Source = new DemoSource();
  }
}

public class DemoSource : UITableViewSource
{
  private List<String> data = new List<String>()
  {
    "iOS", "Android", "Xamarin", "UWP", "Ionic", "NativeScript", "PhoneGap"
  };

  public DemoSource()
  {
  }

  public override UITableViewCell GetCell(UITableView tableView, NSIndexPath indexPath)
  {
    var cell = tableView.DequeueReusableCell("cellId", indexPath);
    cell.TextLabel.Text = data[indexPath.Row];
    return cell;
  }

  public override nint RowsInSection(UITableView tableview, nint section)
  {
    return data.Count;
  }
}
```

```
Button button = (Button)findViewById(R.id.btnClickMe);

button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // more code here
    }
});
```

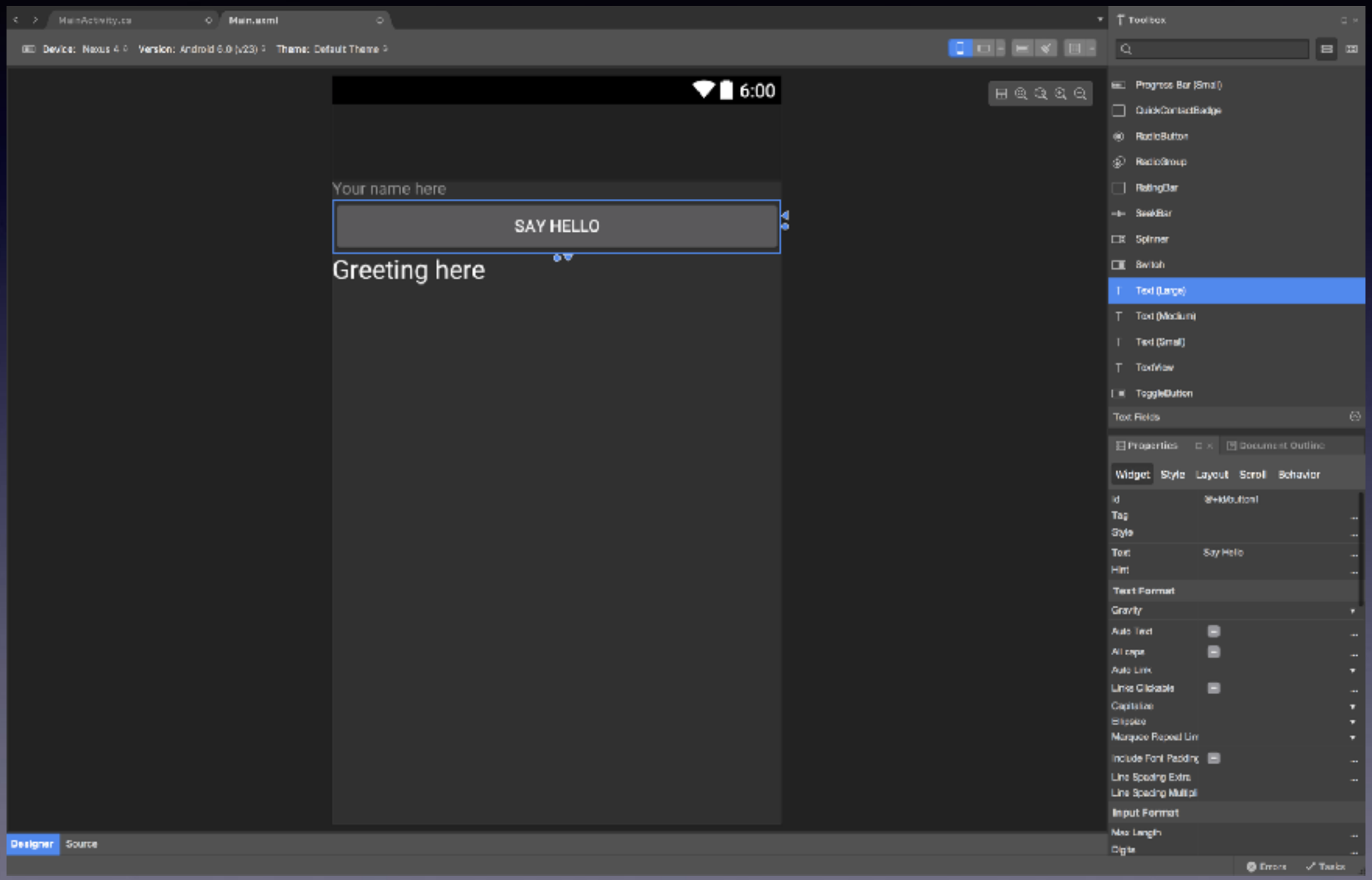
```
Button button = (Button)findViewById(R.id.btnClickMe);

button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // more code here
    }
});
```

```
protected override void OnCreate(Bundle savedInstanceState)
{
      base.OnCreate(savedInstanceState);

      // Set our view from the "main" layout resource
      SetContentView(Resource.Layout.Main);

      // Get our button from the layout resource,
      // and attach an event to it
      Button button = FindViewById<Button>(Resource.Id.btnClickMe);

      button.Click += delegate { /* More code here */ };
}
```

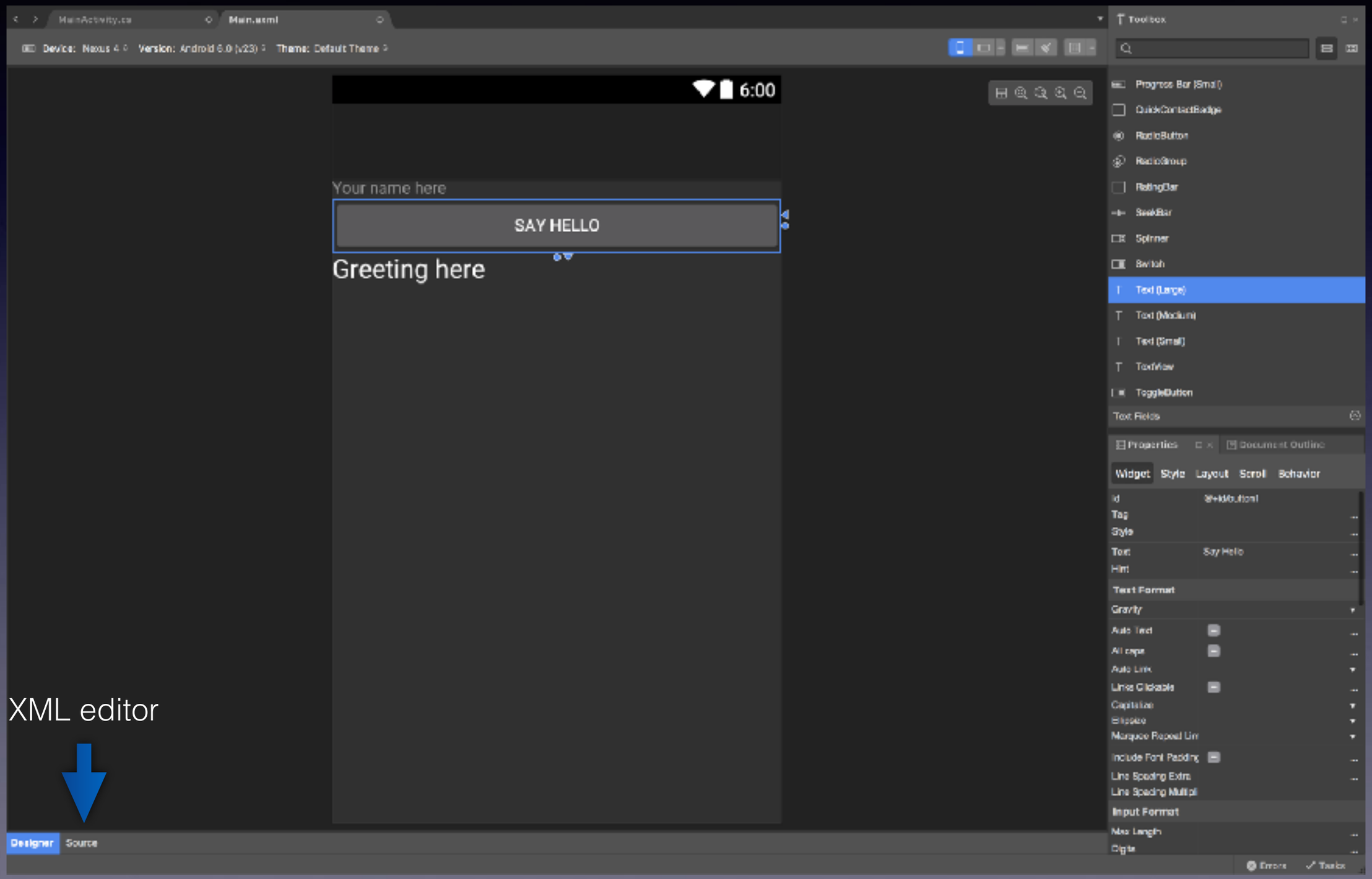
Designing a UI



Designing a UI



Designing a UI



XML editor

```
class MathOps {
  int Add() {
    return a + b;
  }

  int Subtract() {
    return a - b;
  }

  // ...
}
```

```
class MathOps {
  int Add() {
    return a + b;
  }

  int Subtract() {
    return a - b;
  }

  // ...
}
```

```
addButton.TouchUpInside += (sender, e) =>
{
  var mathOps = new MathOps(getOperand1(), getOperand2());
  var sum = mathOps.Add();
  txtResult.Text = $"{sum}";
};
```

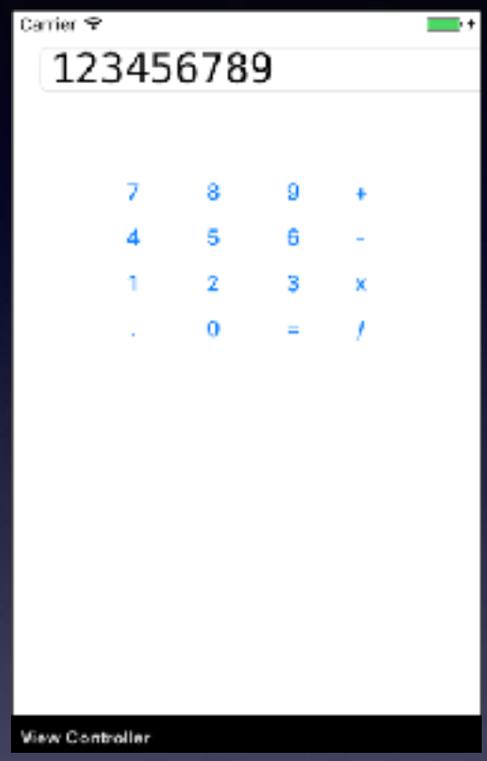
```
class MathOps {
  int Add() {
    return a + b;
  }

  int Subtract() {
    return a - b;
  }

  // ...
}
```

```
addButton.TouchUpInside += (sender, e) =>
{
  var mathOps = new MathOps(getOperand1(), getOperand2());
  var sum = mathOps.Add();
  txtResult.Text = $"{sum}";
};
```


```
Button addButton = FindViewById<Button>(Resource.Id.btnAddButton);

addButton.OnClick += (sender, e) => {
      var mathOps = new MathOps(getOperand1(), getOperand2());
      var sum = mathOps.Add();
      txtResult.Text = $"{sum}";
    }
};
```

```
class MathOps {
  int Add() {
    return a + b;
  }

  int Subtract() {
    return a - b;
  }

  // ...
}
```

```
addButton.TouchUpInside += (sender, e) =>
{
  var mathOps = new MathOps(getOperand1(), getOperand2());
  var sum = mathOps.Add();
  txtResult.Text = $"{sum}";
};
```




```
Button addButton = FindViewById<Button>(Resource.Id.btnAddButton);

addButton.OnClick += (sender, e) => {
      var mathOps = new MathOps(getOperand1(), getOperand2());
      var sum = mathOps.Add();
      txtResult.Text = $"{sum}";
    }
};
```

Xamarin.Forms

Xamarin.Forms

Implements the UI via a common C# codebase

Xamarin.Forms

Implements the UI via a common C# codebase

Renders native controls on each individual platform

Xamarin.Forms

Implements the UI via a common C# codebase

Renders native controls on each individual platform

Declarative markup with XAML

Xamarin.Forms

Implements the UI via a common C# codebase

Renders native controls on each individual platform

Declarative markup with XAML

Up to 90% code reuse across platforms

Xamarin.Forms

Implements the UI via a common C# codebase

Renders native controls on each individual platform

Declarative markup with XAML

Up to 90% code reuse across platforms

Supports iOS, Android, Windows 10/UWP

Xamarin.Forms

Implements the UI via a common C# codebase

Renders native controls on each individual platform

Declarative markup with XAML

Up to 90% code reuse across platforms

Supports iOS, Android, Windows 10/UWP

Open source on Github

Xamarin.Forms

Implements the UI via a common C# codebase

Renders native controls on each individual platform

Declarative markup with XAML

Up to 90% code reuse across platforms

Supports iOS, Android, Windows 10/UWP

Open source on Github

No visual designer

Xamarin.Forms

Implements the UI via a common C# codebase

Renders native controls on each individual platform

Declarative markup with XAML

Up to 90% code reuse across platforms

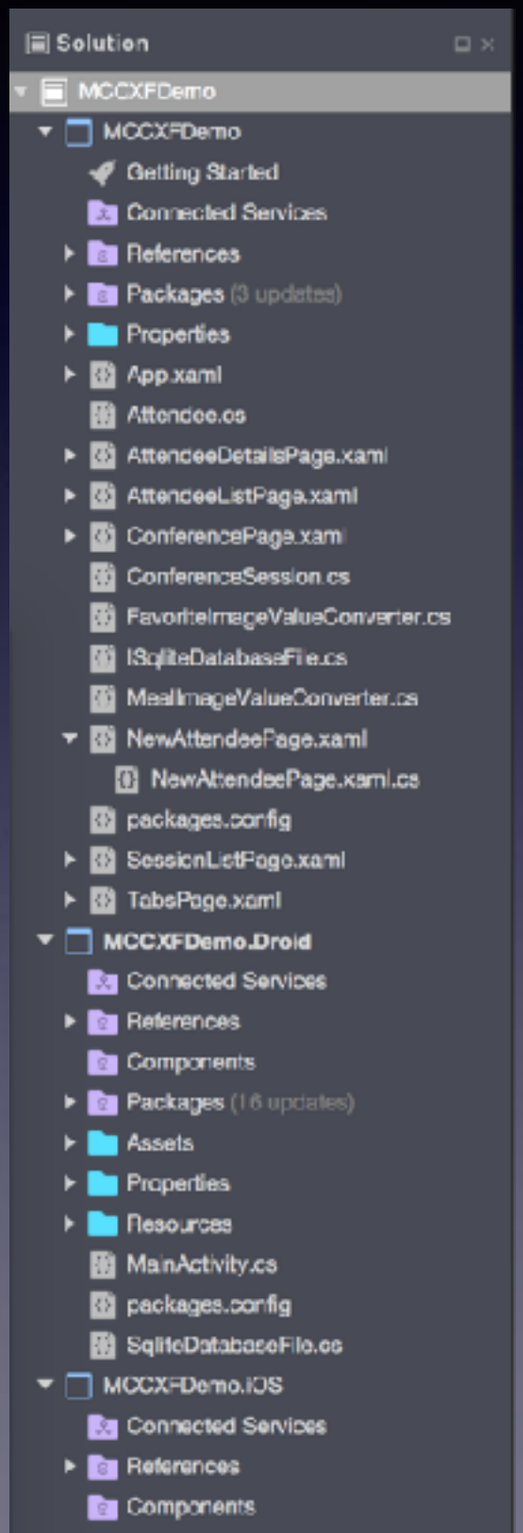
Supports iOS, Android, Windows 10/UWP

Open source on Github

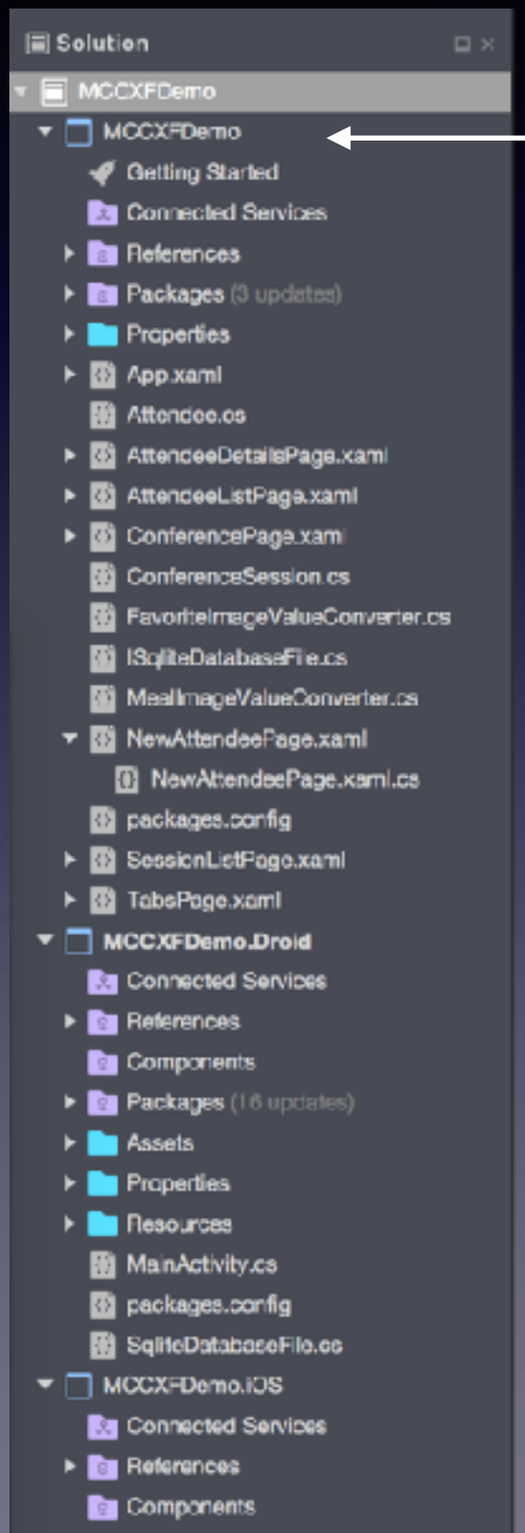
No visual designer

Static previewer (YMMV with Android)

Xamarin.Forms

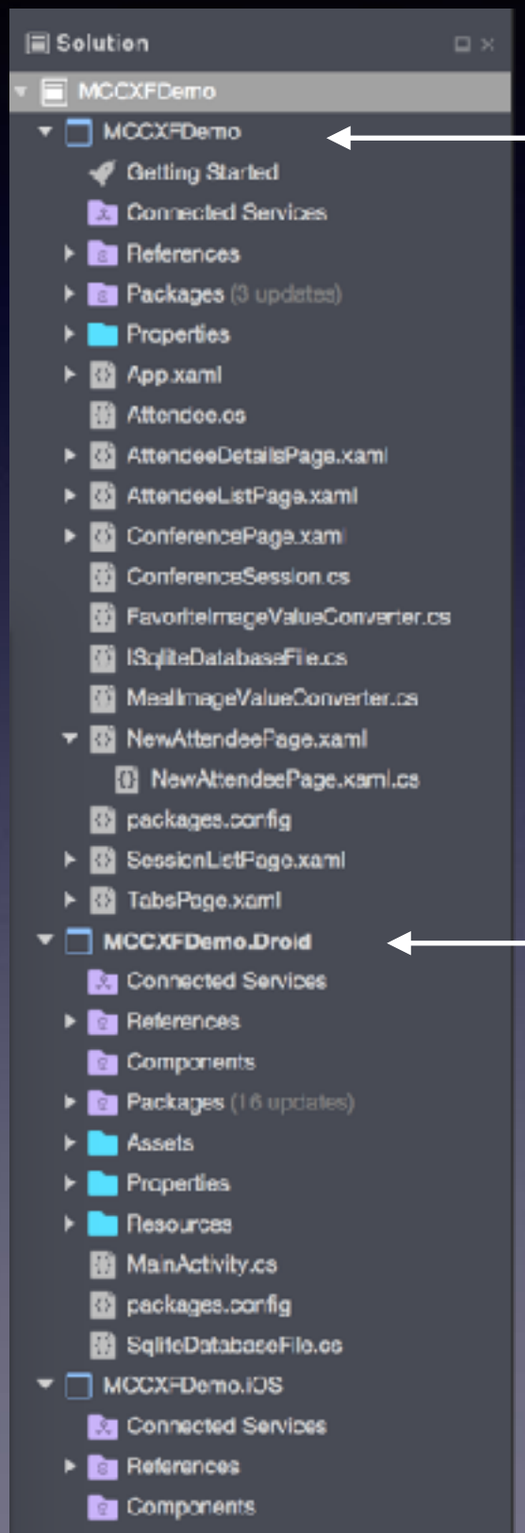


Xamarin.Forms



Xamarin.Forms Project

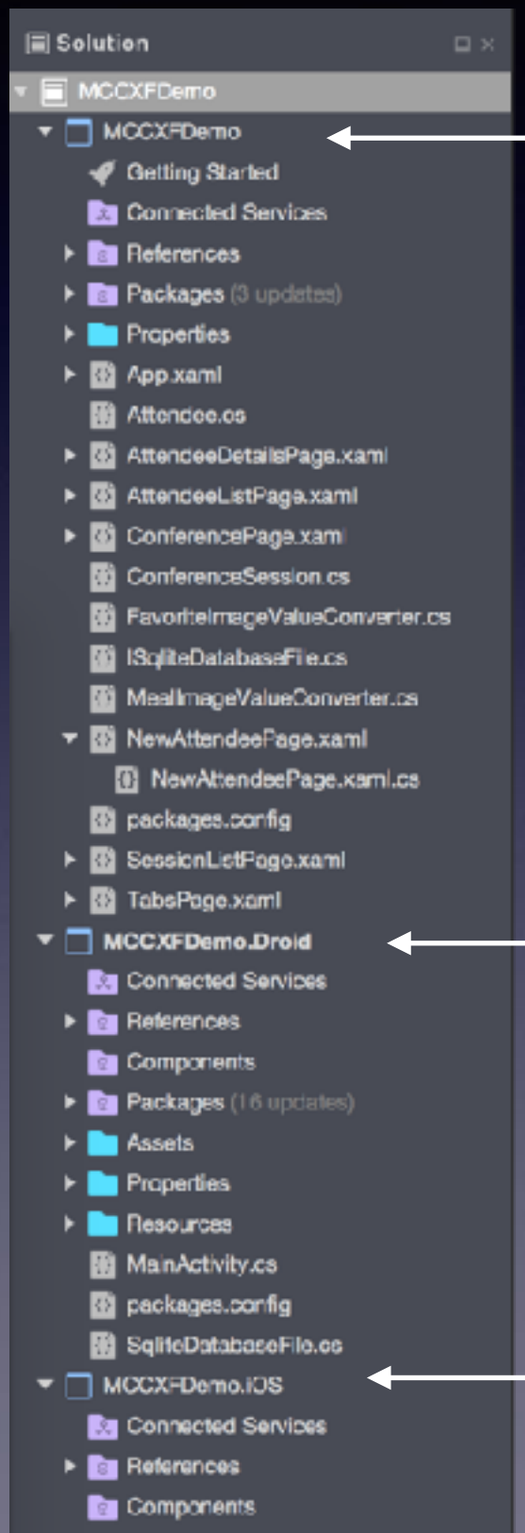
Xamarin.Forms



Xamarin.Forms Project

Xamarin.Android Project

Xamarin.Forms



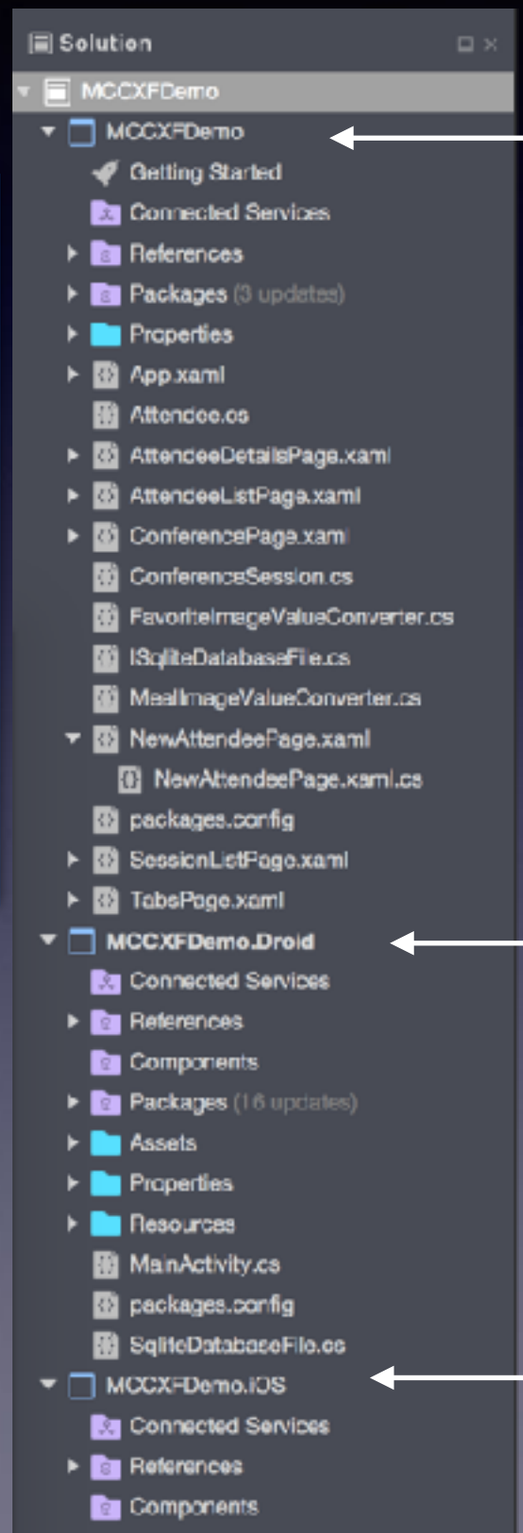
Xamarin.Forms Project

Xamarin.Android Project

Xamarin.iOS Project

Xamarin.Forms

90%



Xamarin.Forms Project

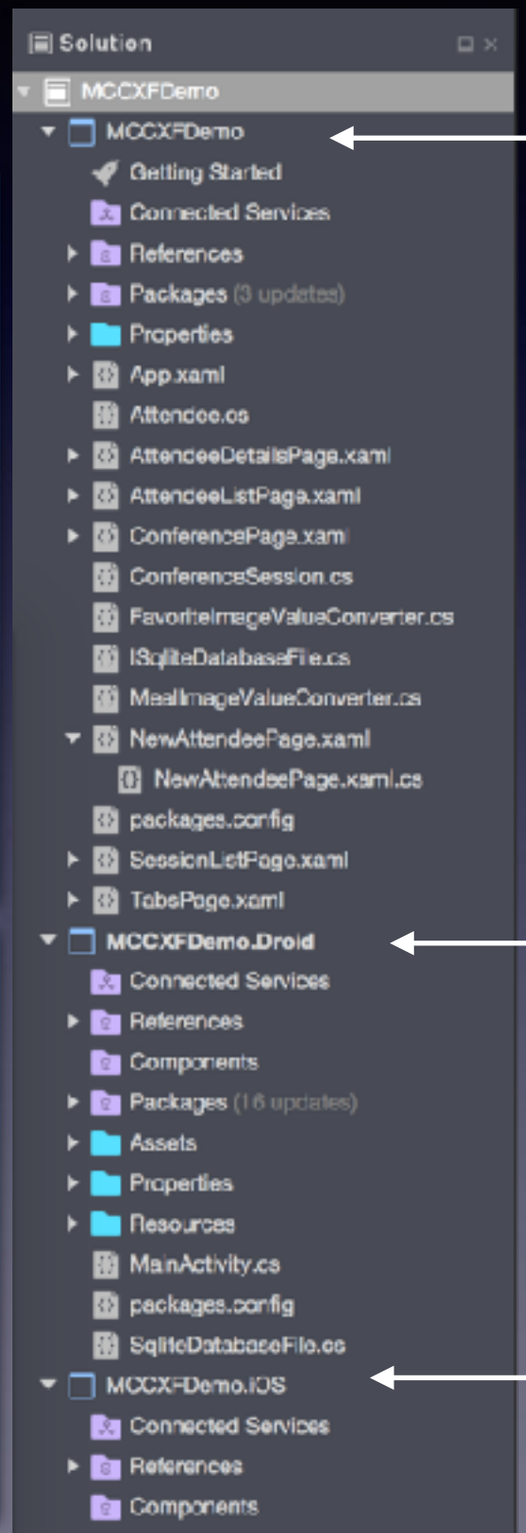
Xamarin.Android Project

Xamarin.iOS Project

Xamarin.Forms

90%

10%



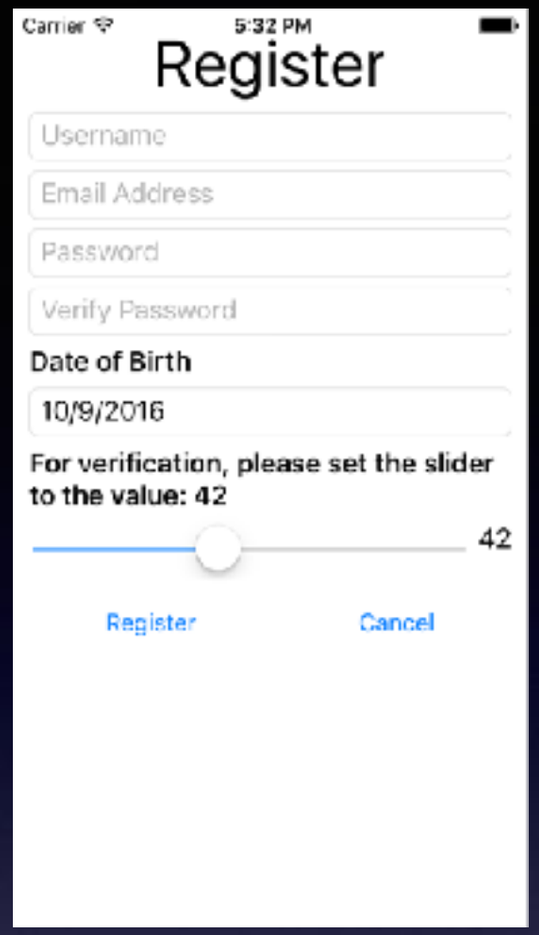
Xamarin.Forms Project

Xamarin.Android Project

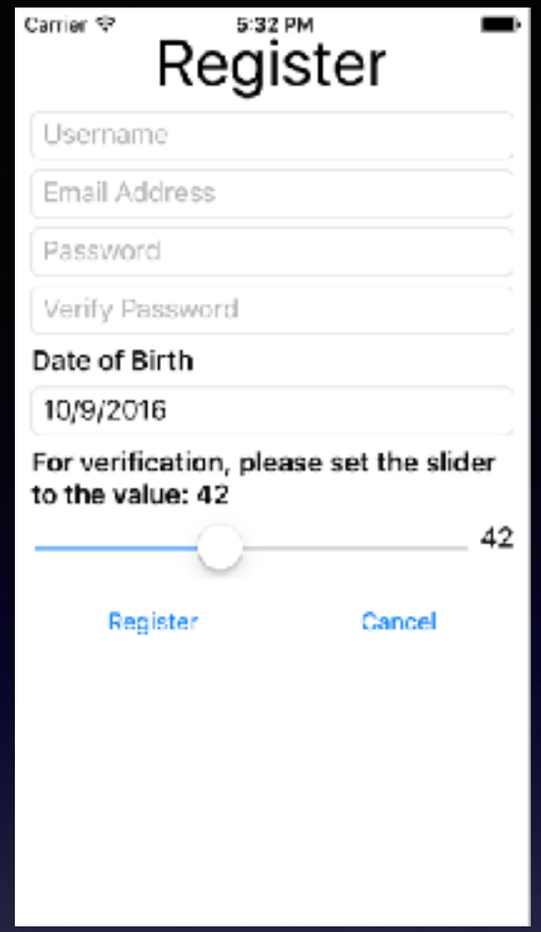
Xamarin.iOS Project

```
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:MCCXFDemo"
             x:Class="MCCXFDemo.MCCXFDemoPage">
       <ContentPage.Content>
             <StackLayout Orientation="Vertical" Padding="10">
                    <Label FontSize="40" HorizontalTextAlignment="Center" Text="Register"/>
                    <Entry Placeholder="Username" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Email Address" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Entry Placeholder="Verify Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Label Text="Date of Birth" FontAttributes="Bold"/>
                    <DatePicker/>
                    <Label HorizontalTextAlignment="Start" FontAttributes="Bold">
                           For verification, please set the slider to the value: 42
                    </Label>
                    <StackLayout Orientation="Horizontal">
                           <Slider HorizontalOptions="FillAndExpand" Minimum="0" Maximum="100" Value="42" />
                           <Label Text="42" />
                    </StackLayout>
                    <StackLayout Orientation="Horizontal">
                           <Button Text="Register" HorizontalOptions="FillAndExpand"/>
                           <Button Text="Cancel" HorizontalOptions="FillAndExpand"/>
                    </StackLayout>
             </StackLayout>
       </ContentPage.Content>
</ContentPage>
```

```
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:MCCXFDemo"
             x:Class="MCCXFDemo.MCCXFDemoPage">
      <ContentPage.Content>
             <StackLayout Orientation="Vertical" Padding="10">
                    <Label FontSize="40" HorizontalTextAlignment="Center" Text="Register"/>
                    <Entry Placeholder="Username" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Email Address" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Entry Placeholder="Verify Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Label Text="Date of Birth" FontAttributes="Bold"/>
                    <DatePicker/>
                    <Label HorizontalTextAlignment="Start" FontAttributes="Bold">
                           For verification, please set the slider to the value: 42
                    </Label>
                    <StackLayout Orientation="Horizontal">
                           <Slider HorizontalOptions="FillAndExpand" Minimum="0" Maximum="100" Value="42" />
                           <Label Text="42" />
                    </StackLayout>
                    <StackLayout Orientation="Horizontal">
                           <Button Text="Register" HorizontalOptions="FillAndExpand"/>
                           <Button Text="Cancel" HorizontalOptions="FillAndExpand"/>
                    </StackLayout>
             </StackLayout>
      </ContentPage.Content>
</ContentPage>
```

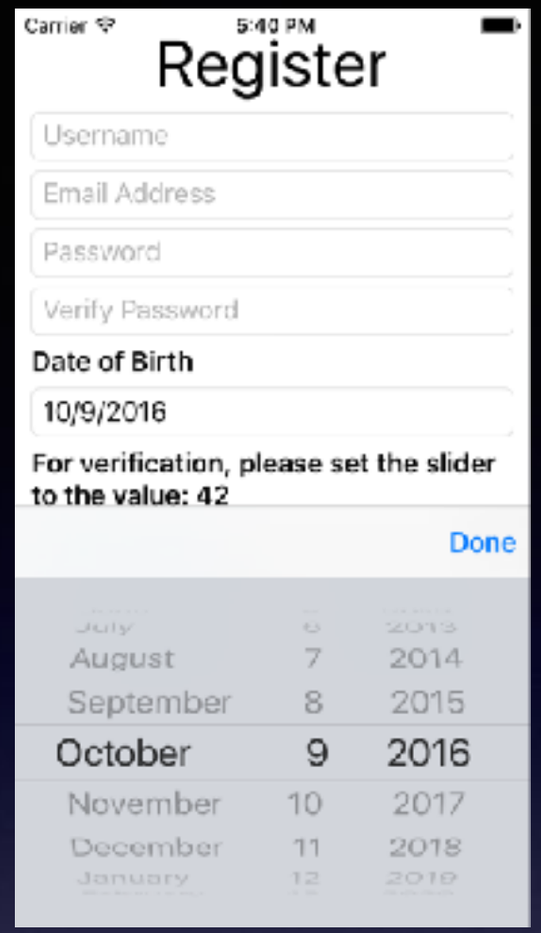


```
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:MCCXFDemo"
             x:Class="MCCXFDemo.MCCXFDemoPage">
      <ContentPage.Content>
             <StackLayout Orientation="Vertical" Padding="10">
                    <Label FontSize="40" HorizontalTextAlignment="Center" Text="Register"/>
                    <Entry Placeholder="Username" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Email Address" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Entry Placeholder="Verify Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Label Text="Date of Birth" FontAttributes="Bold"/>
                    <DatePicker/>
                    <Label HorizontalTextAlignment="Start" FontAttributes="Bold">
                           For verification, please set the slider to the value: 42
                    </Label>
                    <StackLayout Orientation="Horizontal">
                           <Slider HorizontalOptions="FillAndExpand" Minimum="0" Maximum="100" Value="42" />
                           <Label Text="42" />
                    </StackLayout>
                    <StackLayout Orientation="Horizontal">
                           <Button Text="Register" HorizontalOptions="FillAndExpand"/>
                           <Button Text="Cancel" HorizontalOptions="FillAndExpand"/>
                    </StackLayout>
             </StackLayout>
      </ContentPage.Content>
</ContentPage>
```

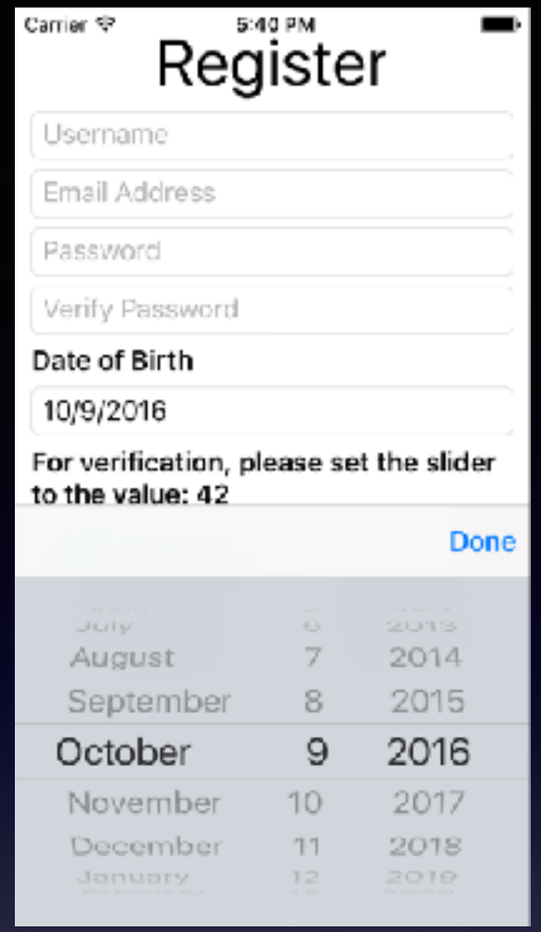


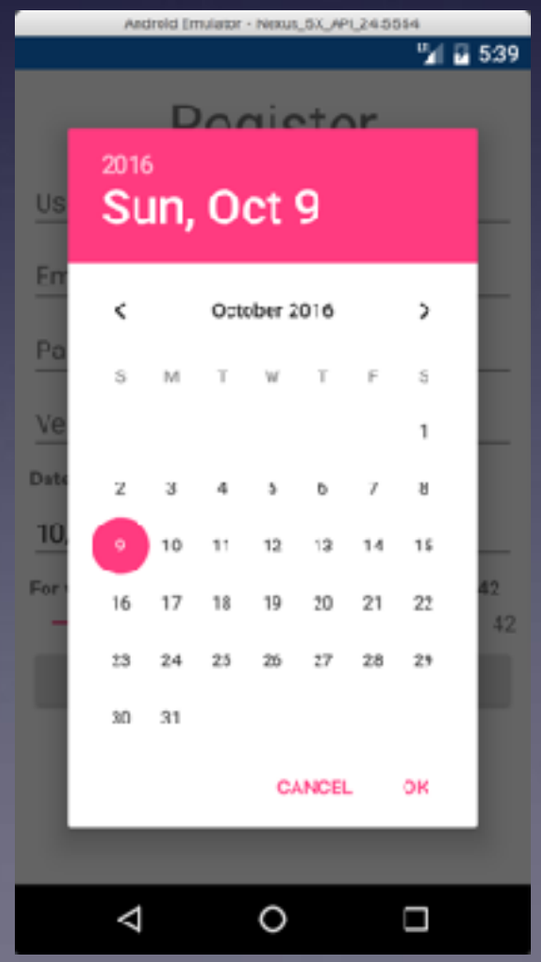
```
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:MCCXFDemo"
             x:Class="MCCXFDemo.MCCXFDemoPage">
       <ContentPage.Content>
             <StackLayout Orientation="Vertical" Padding="10">
                    <Label FontSize="40" HorizontalTextAlignment="Center" Text="Register"/>
                    <Entry Placeholder="Username" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Email Address" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Entry Placeholder="Verify Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Label Text="Date of Birth" FontAttributes="Bold"/>
                    <DatePicker/>
                    <Label HorizontalTextAlignment="Start" FontAttributes="Bold">
                           For verification, please set the slider to the value: 42
                    </Label>
                    <StackLayout Orientation="Horizontal">
                           <Slider HorizontalOptions="FillAndExpand" Minimum="0" Maximum="100" Value="42" />
                           <Label Text="42" />
                    </StackLayout>
                    <StackLayout Orientation="Horizontal">
                           <Button Text="Register" HorizontalOptions="FillAndExpand"/>
                           <Button Text="Cancel" HorizontalOptions="FillAndExpand"/>
                    </StackLayout>
             </StackLayout>
       </ContentPage.Content>
</ContentPage>
```

```
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:MCCXFDemo"
             x:Class="MCCXFDemo.MCCXFDemoPage">
      <ContentPage.Content>
             <StackLayout Orientation="Vertical" Padding="10">
                    <Label FontSize="40" HorizontalTextAlignment="Center" Text="Register"/>
                    <Entry Placeholder="Username" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Email Address" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Entry Placeholder="Verify Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Label Text="Date of Birth" FontAttributes="Bold"/>
                    <DatePicker/>
                    <Label HorizontalTextAlignment="Start" FontAttributes="Bold">
                           For verification, please set the slider to the value: 42
                    </Label>
                    <StackLayout Orientation="Horizontal">
                           <Slider HorizontalOptions="FillAndExpand" Minimum="0" Maximum="100" Value="42" />
                           <Label Text="42" />
                    </StackLayout>
                    <StackLayout Orientation="Horizontal">
                           <Button Text="Register" HorizontalOptions="FillAndExpand"/>
                           <Button Text="Cancel" HorizontalOptions="FillAndExpand"/>
                    </StackLayout>
             </StackLayout>
      </ContentPage.Content>
</ContentPage>
```



```
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:MCCXFDemo"
             x:Class="MCCXFDemo.MCCXFDemoPage">
      <ContentPage.Content>
             <StackLayout Orientation="Vertical" Padding="10">
                    <Label FontSize="40" HorizontalTextAlignment="Center" Text="Register"/>
                    <Entry Placeholder="Username" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Email Address" HorizontalOptions="FillAndExpand"/>
                    <Entry Placeholder="Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Entry Placeholder="Verify Password" HorizontalOptions="FillAndExpand" IsPassword="true"/>
                    <Label Text="Date of Birth" FontAttributes="Bold"/>
                    <DatePicker/>
                    <Label HorizontalTextAlignment="Start" FontAttributes="Bold">
                           For verification, please set the slider to the value: 42
                    </Label>
                    <StackLayout Orientation="Horizontal">
                           <Slider HorizontalOptions="FillAndExpand" Minimum="0" Maximum="100" Value="42" />
                           <Label Text="42" />
                    </StackLayout>
                    <StackLayout Orientation="Horizontal">
                           <Button Text="Register" HorizontalOptions="FillAndExpand"/>
                           <Button Text="Cancel" HorizontalOptions="FillAndExpand"/>
                    </StackLayout>
             </StackLayout>
      </ContentPage.Content>
</ContentPage>
```





That Last 10%

That Last 10%

A platform specific feature (NFC, Touch 3D)

That Last 10%

A platform specific feature (NFC, Touch 3D)

A platform specific implementation (SQLite, camera)

That Last 10%

A platform specific feature (NFC, Touch 3D)

A platform specific implementation (SQLite, camera)

Custom user interface

That Last 10%

A platform specific feature (NFC, Touch 3D)

A platform specific implementation (SQLite, camera)

Custom user interface

DependencyService

That Last 10%

A platform specific feature (NFC, Touch 3D)

A platform specific implementation (SQLite, camera)

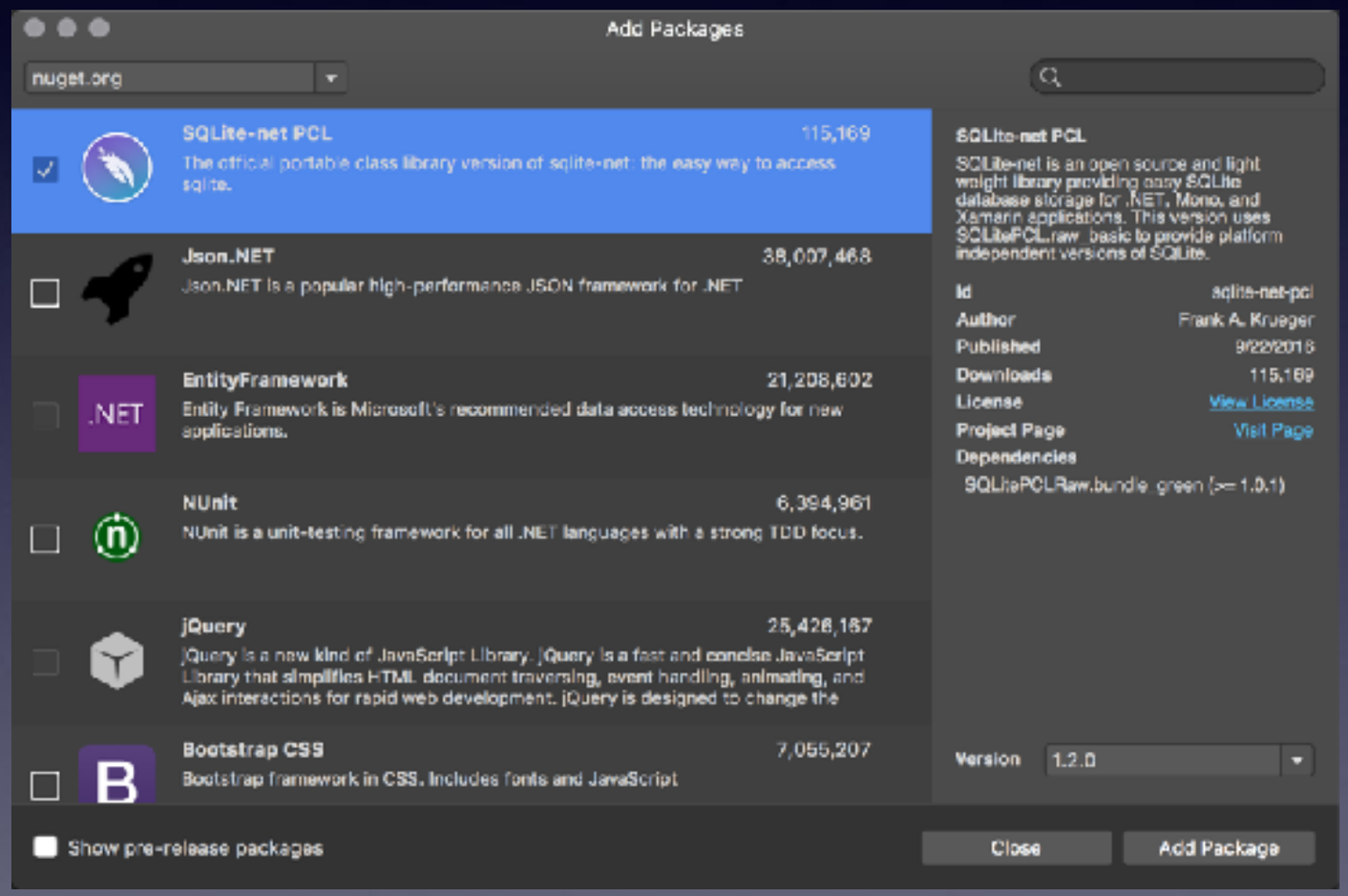
Custom user interface

DependencyService

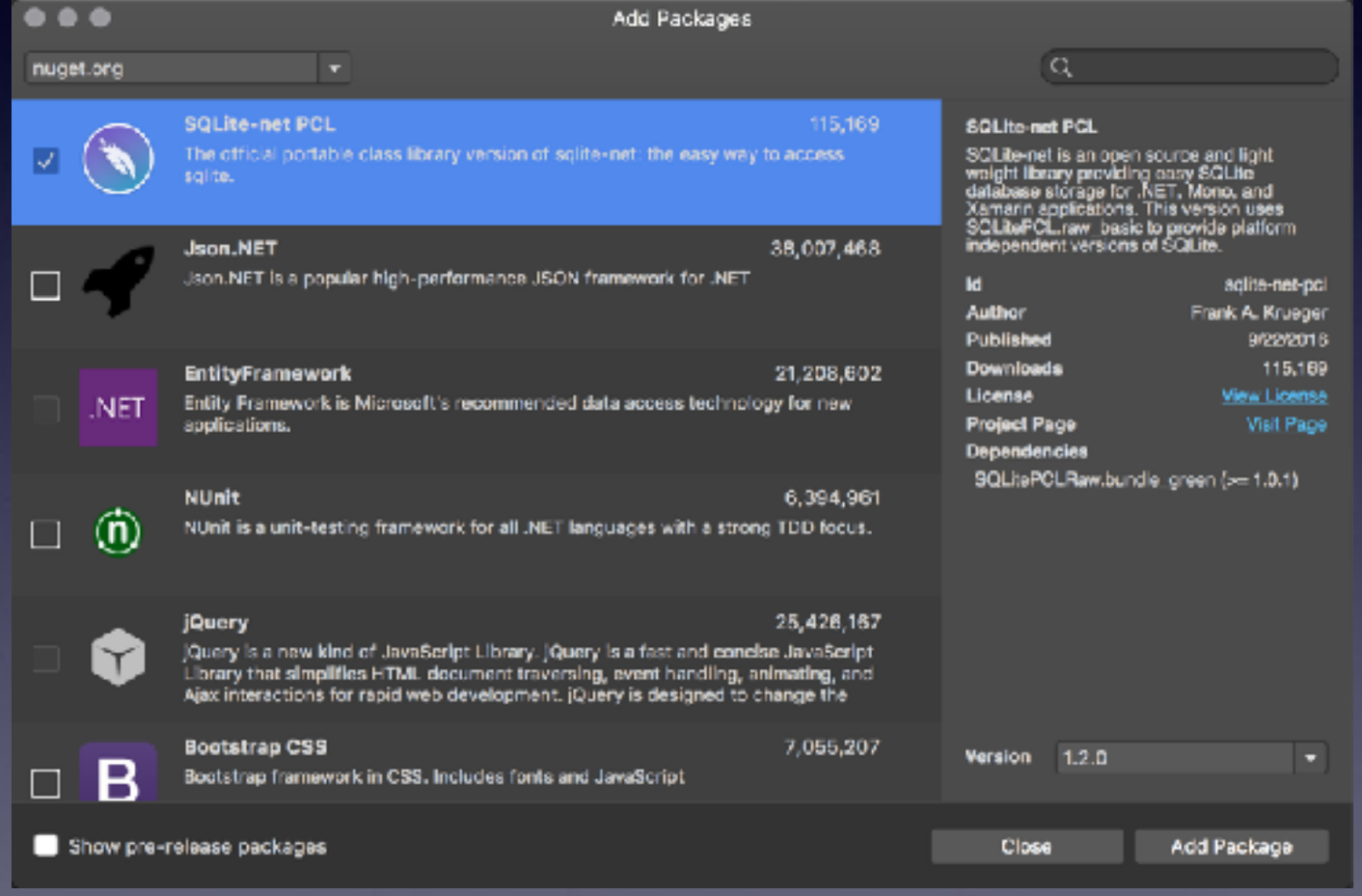
Custom Renderers

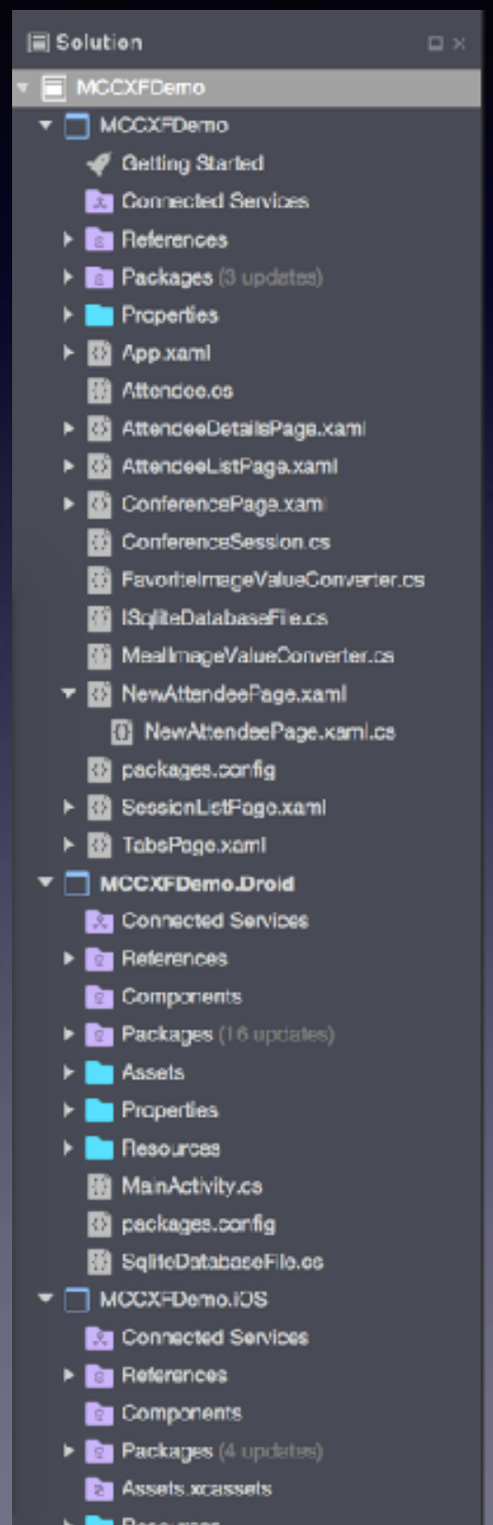
Dependency Service

Dependency Service

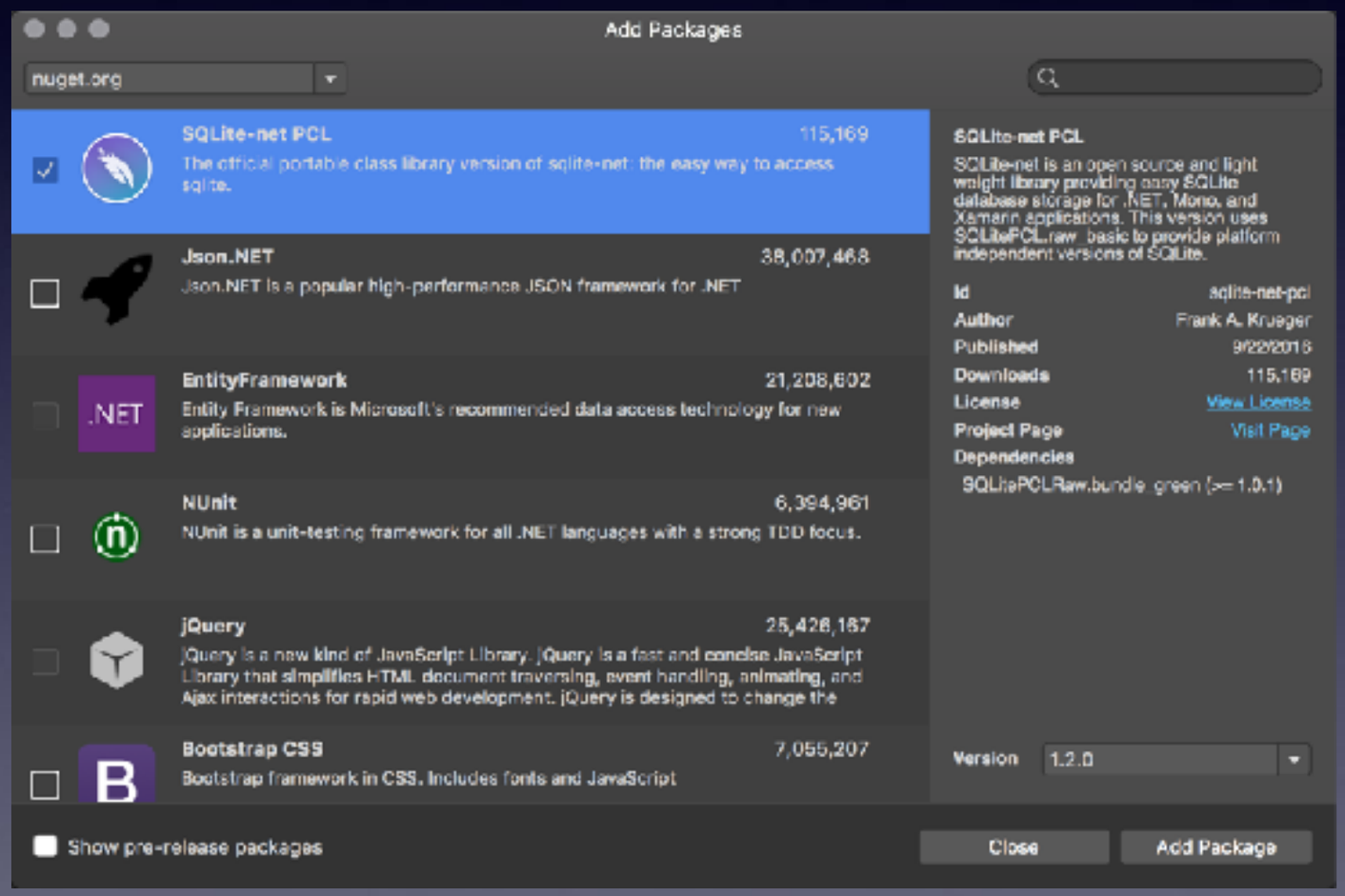


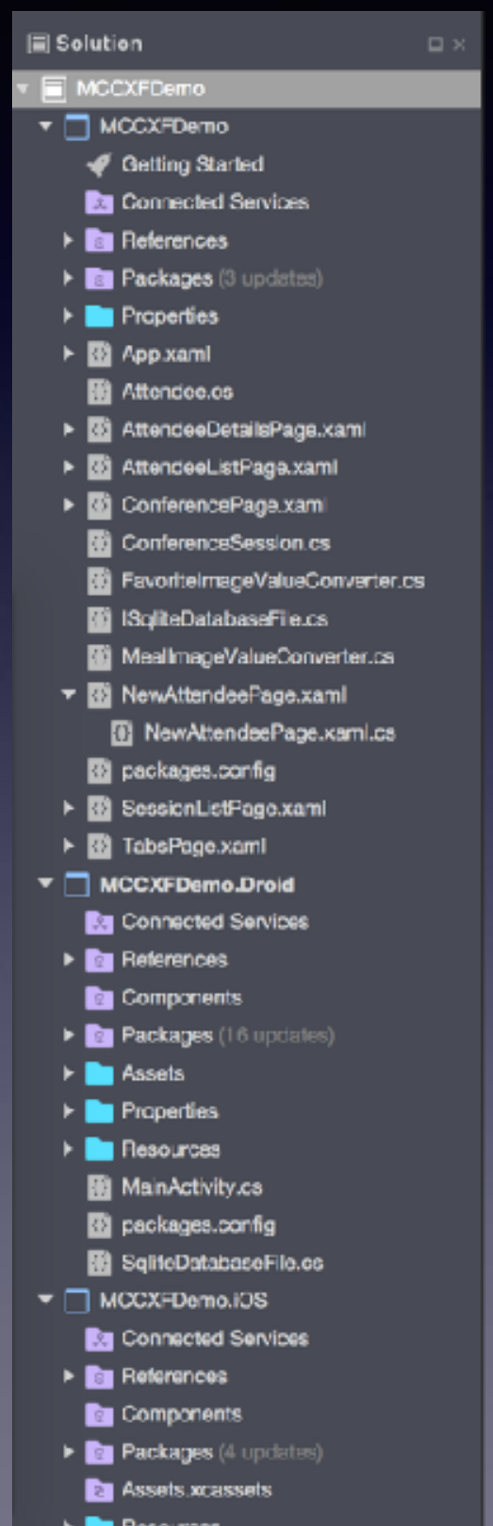
Dependency Service



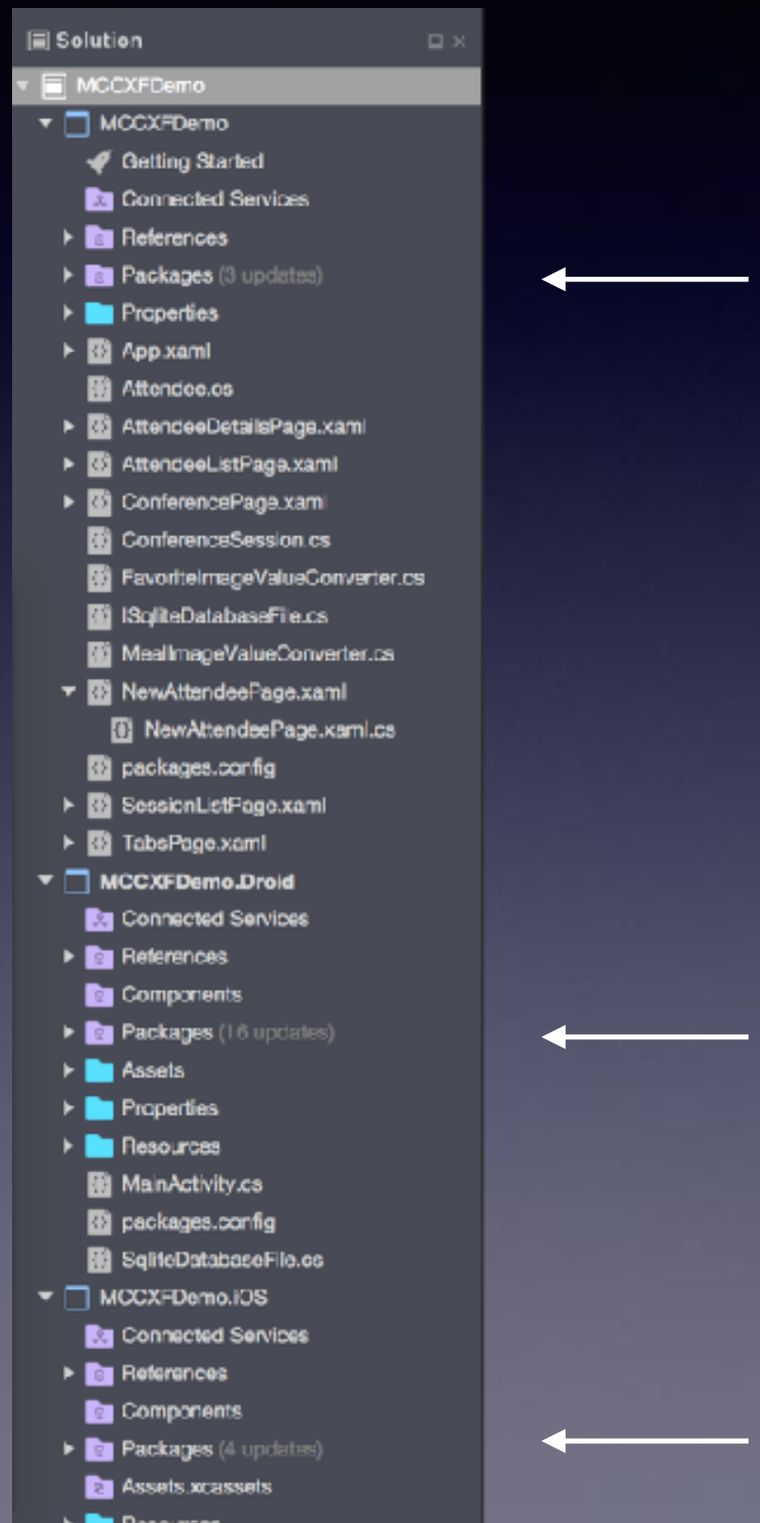


Dependency Service



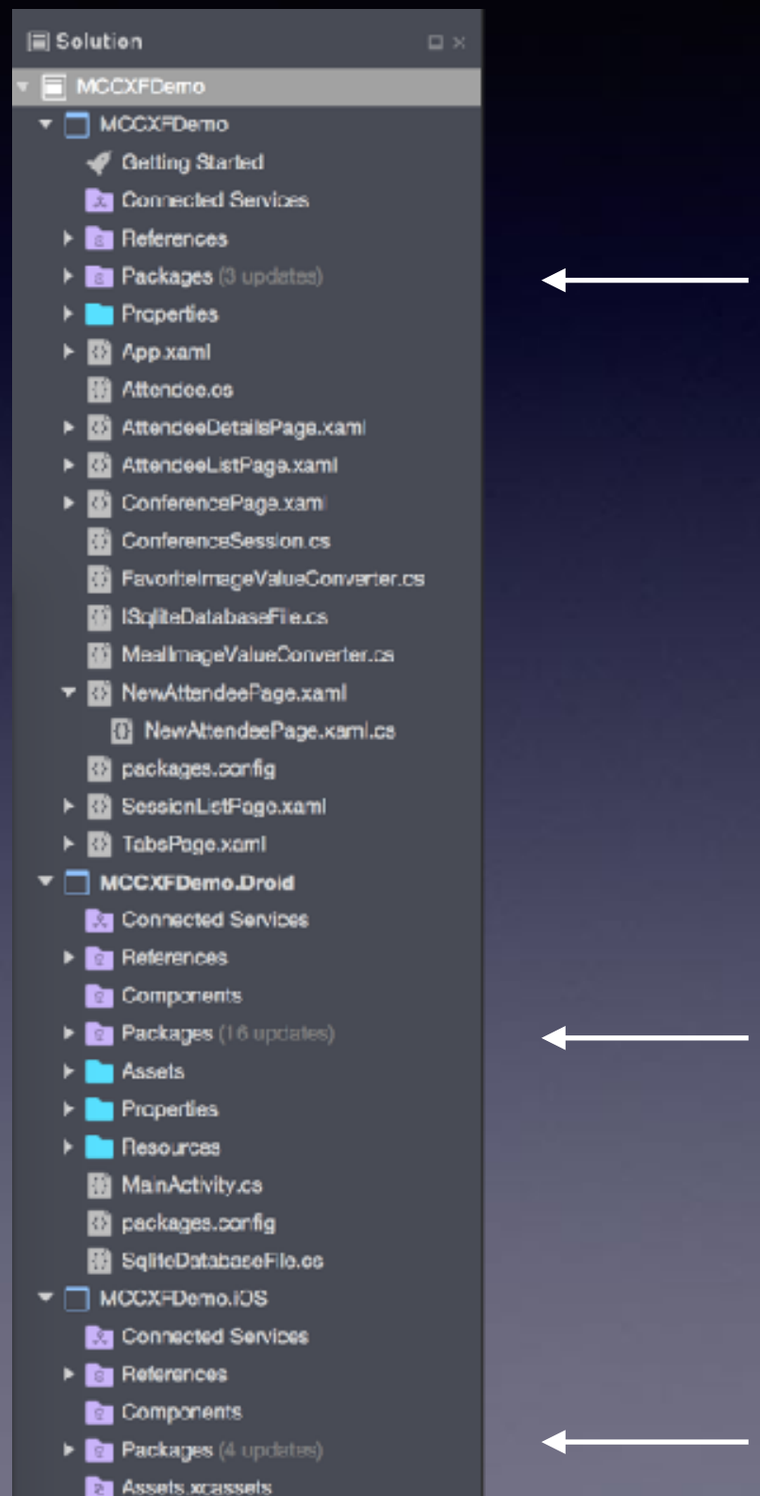


Dependency Service



Dependency Service

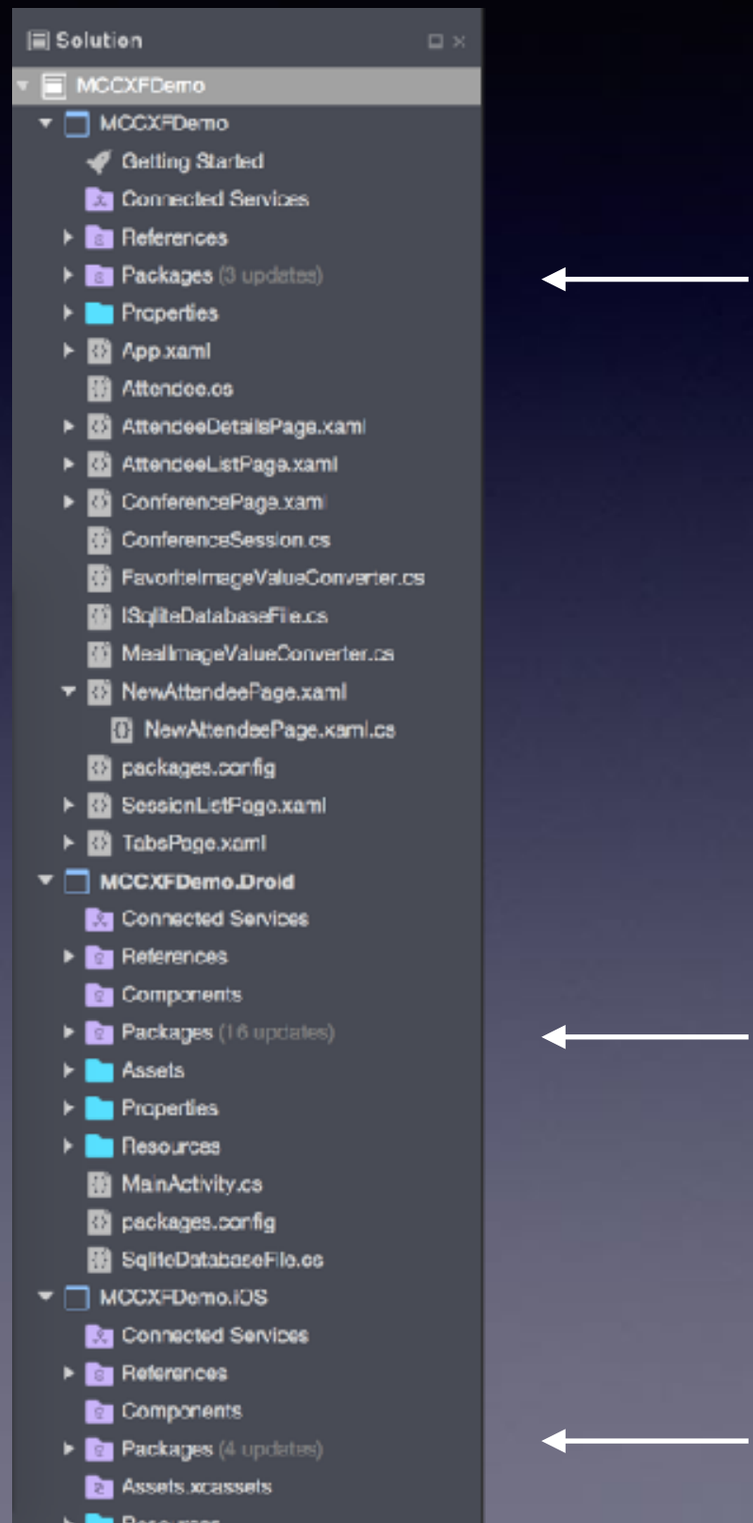
```
public interface ISQLite
{
	SQLiteConnection Connect();
}
```



Dependency Service

```
public interface ISQLite
{
	SQLiteConnection Connect();
}
```

```
[assembly: Dependency(typeof(SQLite_Droid))]
namespace DroidNamespace {
  public class SQLite_Droid : ISQLite {
    public SQLiteConnection Connect() {
      return new SQLiteConnection(databasePath);
    }
  }
}
```

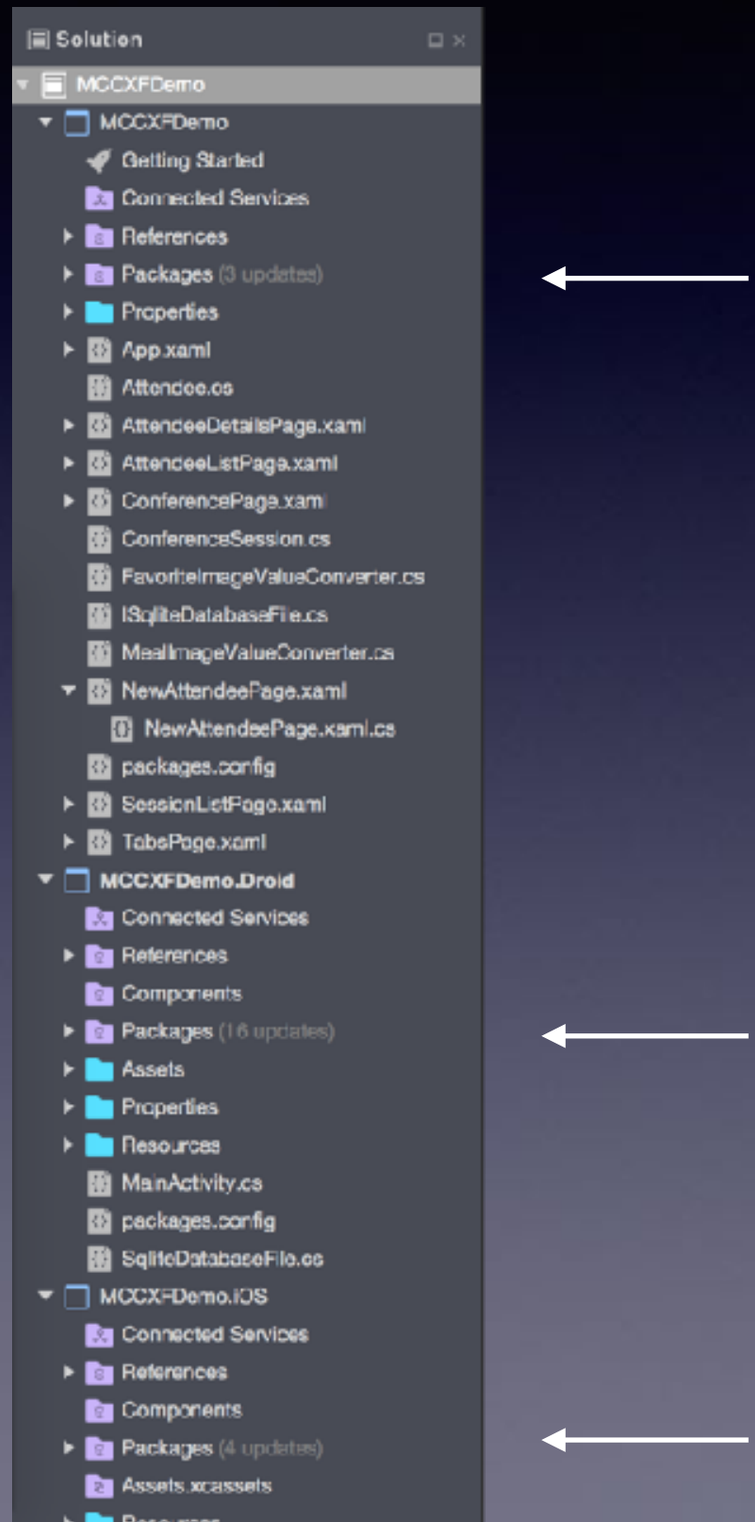


Dependency Service

```
public interface ISQLite
{
	SQLiteConnection Connect();
}
```

```
[assembly: Dependency(typeof(SQLite_Droid))]
namespace DroidNamespace {
  public class SQLite_Droid : ISQLite {
    public SQLiteConnection Connect() {
      return new SQLiteConnection(databasePath);
    }
  }
}
```

```
[assembly: Dependency(typeof(SQLite_IOS))]
namespace IOSNamespace {
  public class SQLite_IOS : ISQLite {
    public SQLiteConnection Connect() {
      return new SQLiteConnection(databasePath);
    }
  }
}
```

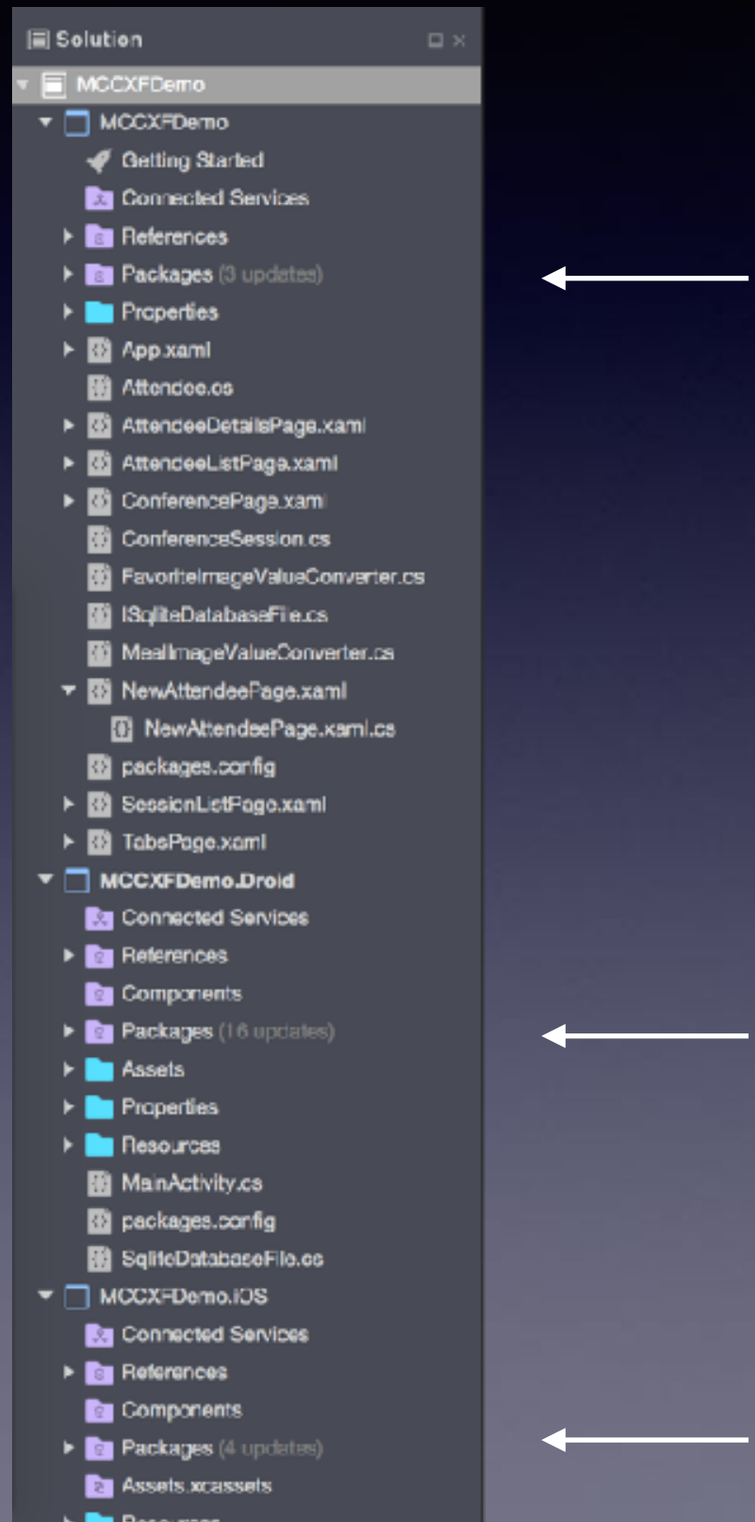


Dependency Service

```
public interface ISQLite
{
        SQLiteConnection Connect();
}

var conn = DependencyService.Get<ISQLite>().Connect();
```

```
[assembly: Dependency(typeof(SQLite_Droid))]
namespace DroidNamespace {
  public class SQLite_Droid : ISQLite {
    public SQLiteConnection Connect() {
      return new SQLiteConnection(databasePath);
    }
  }
}
```

```
[assembly: Dependency(typeof(SQLite_IOS))]
namespace IOSNamespace {
  public class SQLite_IOS : ISQLite {
    public SQLiteConnection Connect() {
      return new SQLiteConnection(databasePath);
    }
  }
}
```



DEMOS

What’s Next?

What’s Next?

iOS Simulator Remoting

What’s Next?

iOS Simulator Remoting

iOS USB Remoting

What’s Next?

iOS Simulator Remoting

iOS USB Remoting

Xamarin.Forms 3.0

What's Next?

iOS Simulator Remoting

iOS USB Remoting

Xamarin.Forms 3.0 // adds WPF, macOS and Linux!

What's Next?

iOS Simulator Remoting

iOS USB Remoting

Xamarin.Forms 3.0 // adds WPF, macOS and Linux!

Xamarin Workbooks

What's Next?

iOS Simulator Remoting

iOS USB Remoting

Xamarin.Forms 3.0 // adds WPF, macOS and Linux!

Xamarin Workbooks

Xamarin Live Player

Thank You!

douglas@douglasstarnes.com
@poweredbyaltnet
douglasstarnes.com