Creating a Mobile Experience for an Existing Modern Application

Richard Taylor

@rightincode
rtaylor@rightincode.com
http://www.rightincode.com

TITANIUM SPONSORS

Platinum Sponsors

Gold Sponsors

Who am I?

Huntersville, NC	Web/Mobile Application Development
Co-Organizer of Modern Devs Charlotte - Meetup.com	Organizer of Charlotte Xamarin Developers - Meetup.com
Microsoft MVP	Telerik/Progress Developer Expert
@rightincode	http://www.rightincode.com

Goals of this Talk

- Introduce Xamarin and Xamarin Forms
- Show how to use Xamarin Forms and Visual Studio to:
  - Leverage your existing C# skills
  - Build a cross platform mobile application
  - Review integrating a mobile application with an existing Web API
- Introduce Azure App Service and Mobile Apps

What is Xamarin?

- Allows developers to deliver native Android, iOS, and Windows applications
  - Creates native user interfaces, provides native API access, and delivers native performance on each target platform
- Allows developers to leverage their existing C# skills
- Allows developers to build a common codebase that can be shared between each platform target

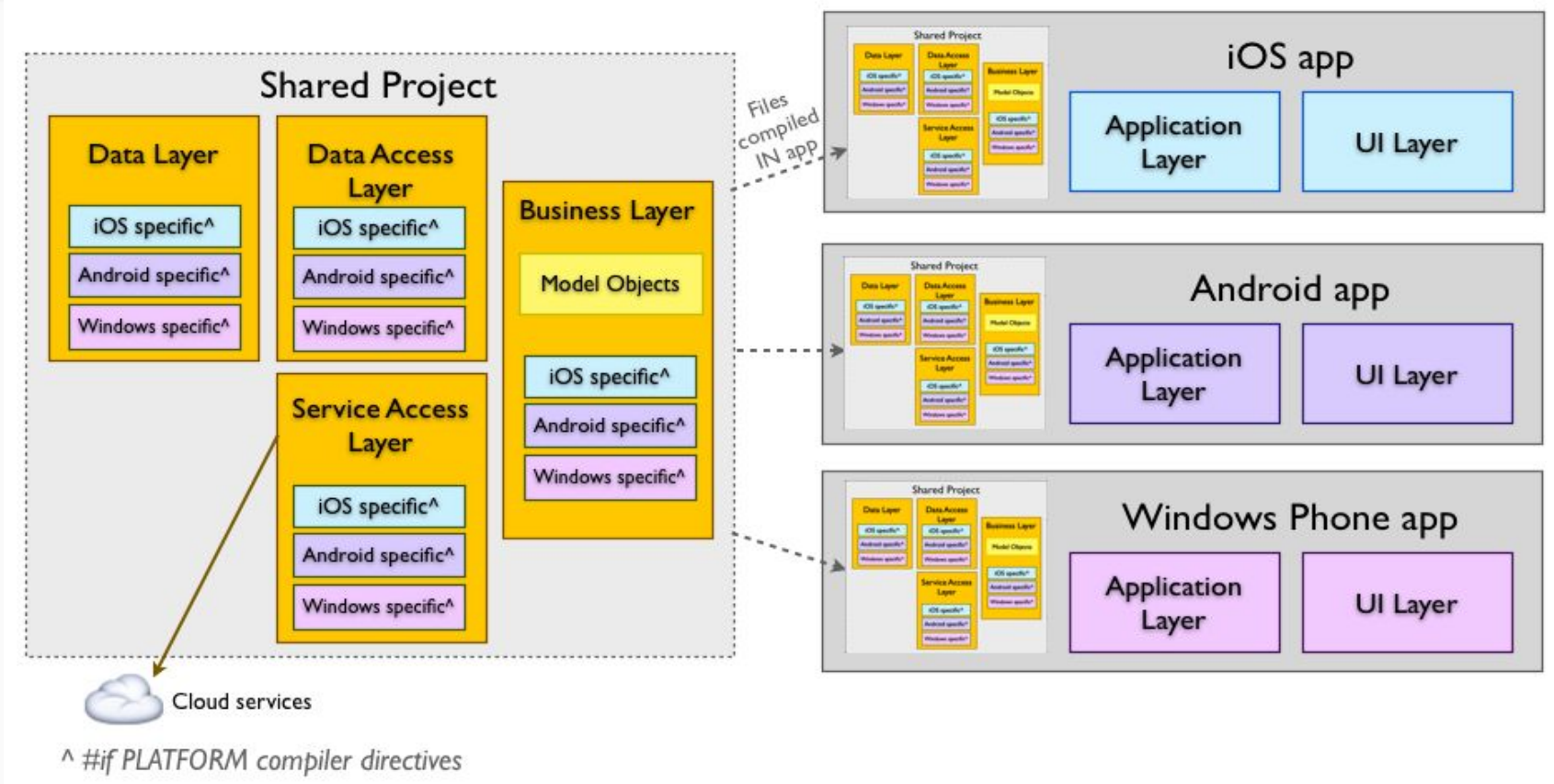
Xamarin - Sharing Code

- Shared Projects
- Portable Class Libraries
- .NET Standard Libraries

Xamarin - Sharing Code (contd.)

- Shared Projects
  - Allows a developer to place code in a common location that can be shared between the platform targets
  - Compiler directives are used to include/exclude platform-specific functionality for each platform target
  - During the build process, the code in the shared project is included in each of the platform target assemblies (there is no output assembly for the shared project)

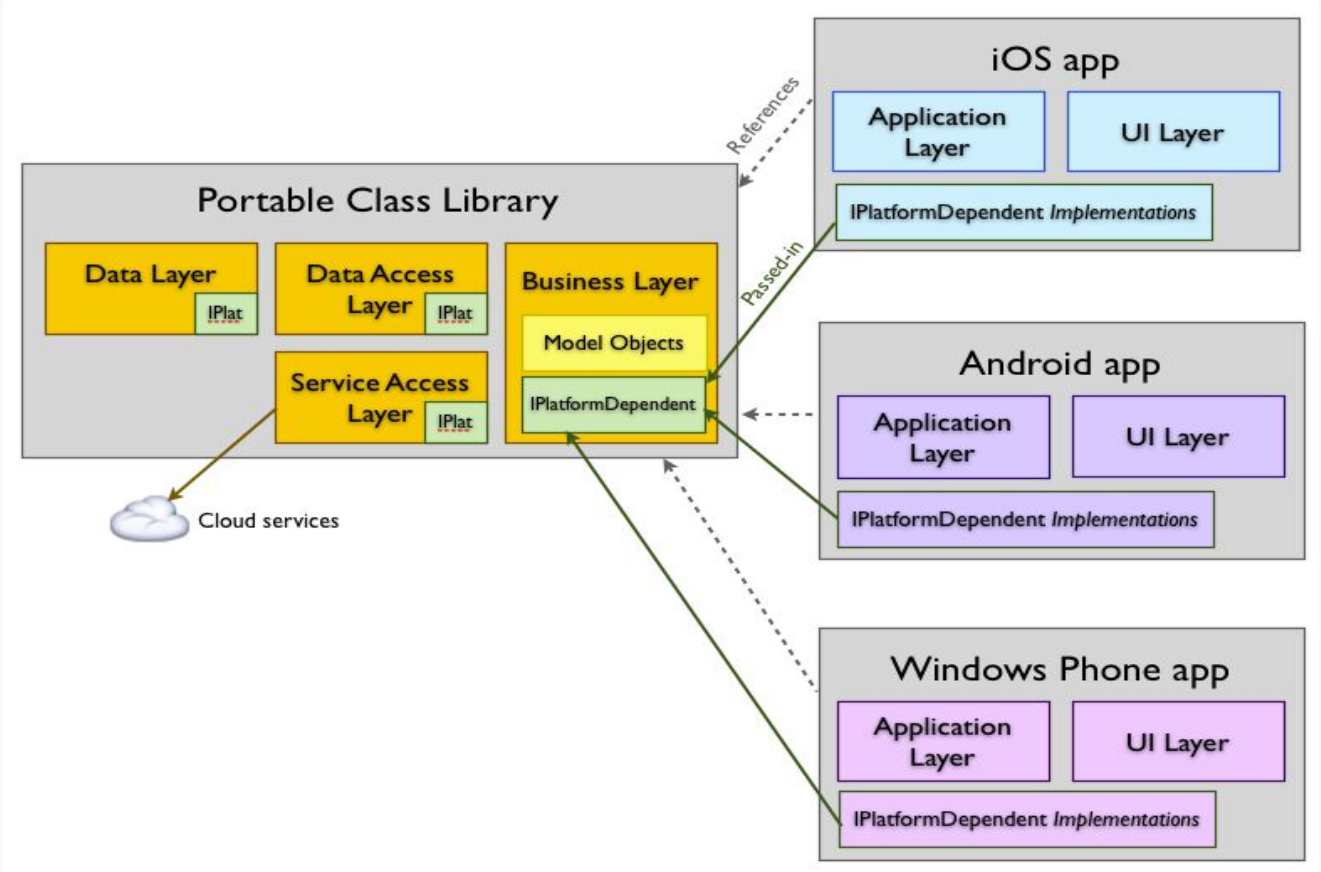
Xamarin - Sharing Code (contd.)



Xamarin - Sharing Code (contd.)

- Portable Class Library Projects
  - Allows a developer to place code in a common location that can be shared between the platform targets
  - PCL’s are referenced by the platform targets (there is an output assembly)
  - PCL’s cannot contain any platform-specific code
  - PCL’s have a profile that describes which features are supported (typically the broader the profile the smaller the number of available features)

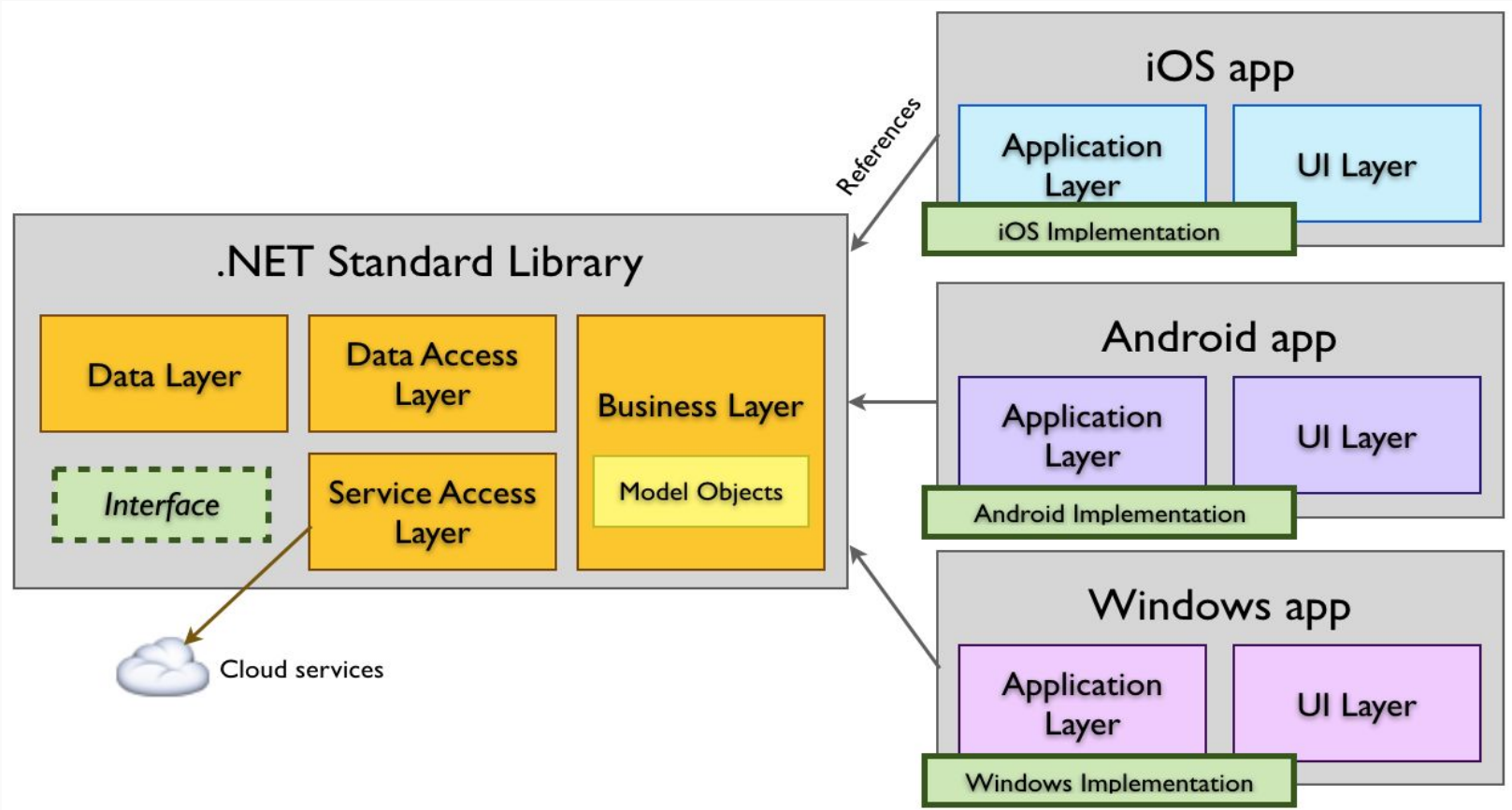
Xamarin - Sharing Code (contd.)



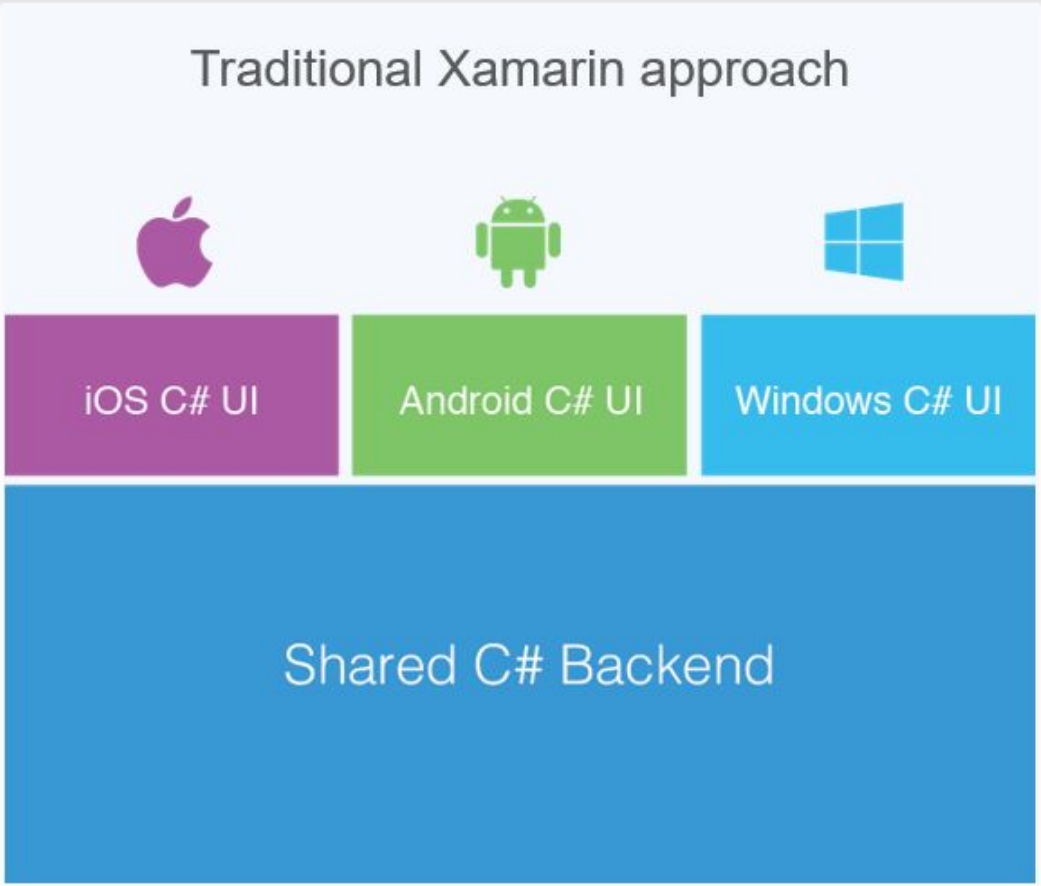
Xamarin - Sharing Code (contd.)

- .NET Standard Libraries
  - Allows a developer to place code in a common location that can be shared between the platform targets
  - .NET Standard Libraries are referenced by the platform targets (there is an output assembly)
  - .NET Standard Libraries cannot contain any platform-specific code
  - .NET Standard Libraries have a larger surface area (available features) than PCL’s
  - .NET Standard Libraries have a uniform API for all .NET Platforms

Xamarin - Sharing Code (contd.)



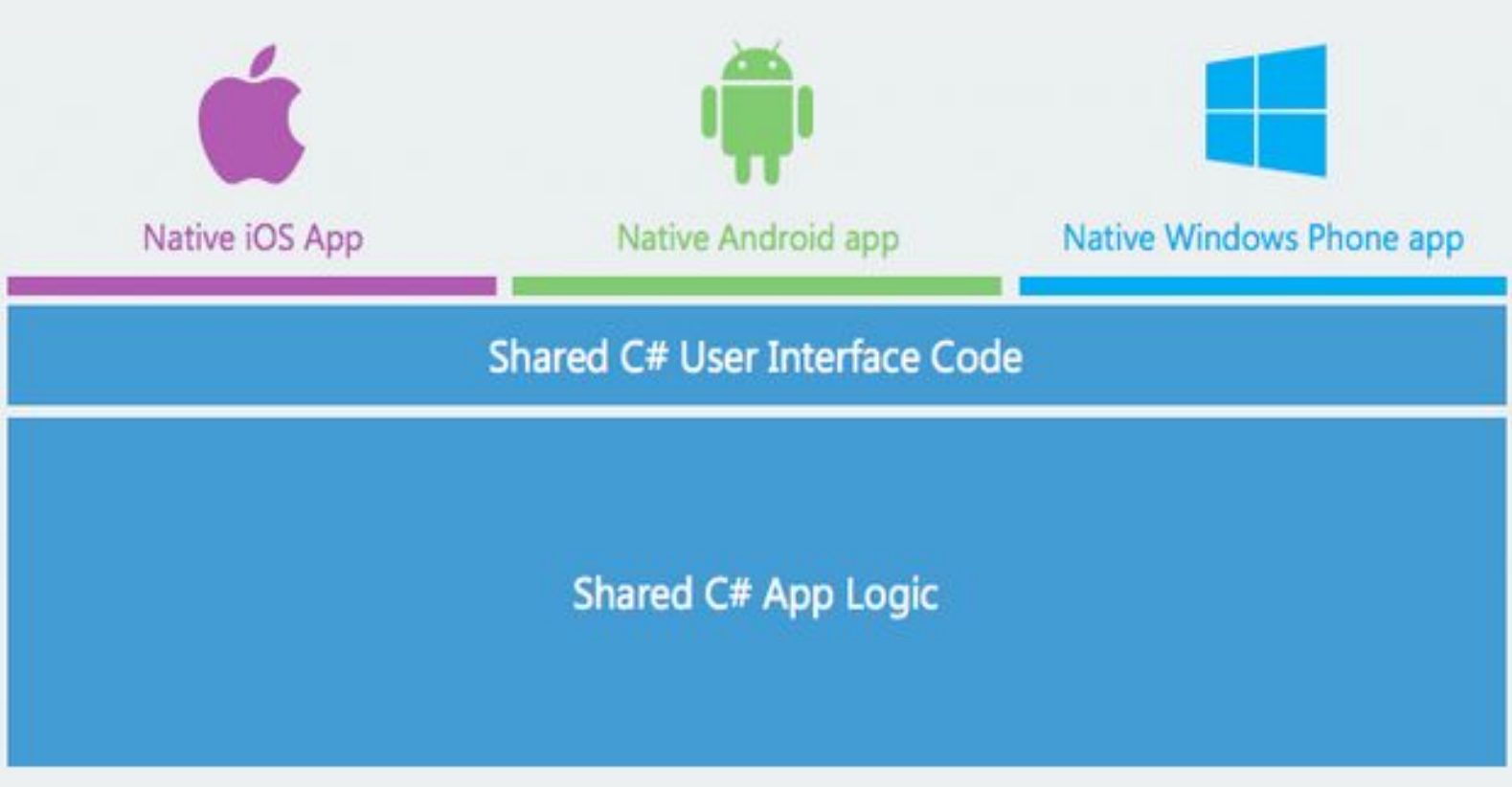
Xamarin - Sharing Code (contd.)



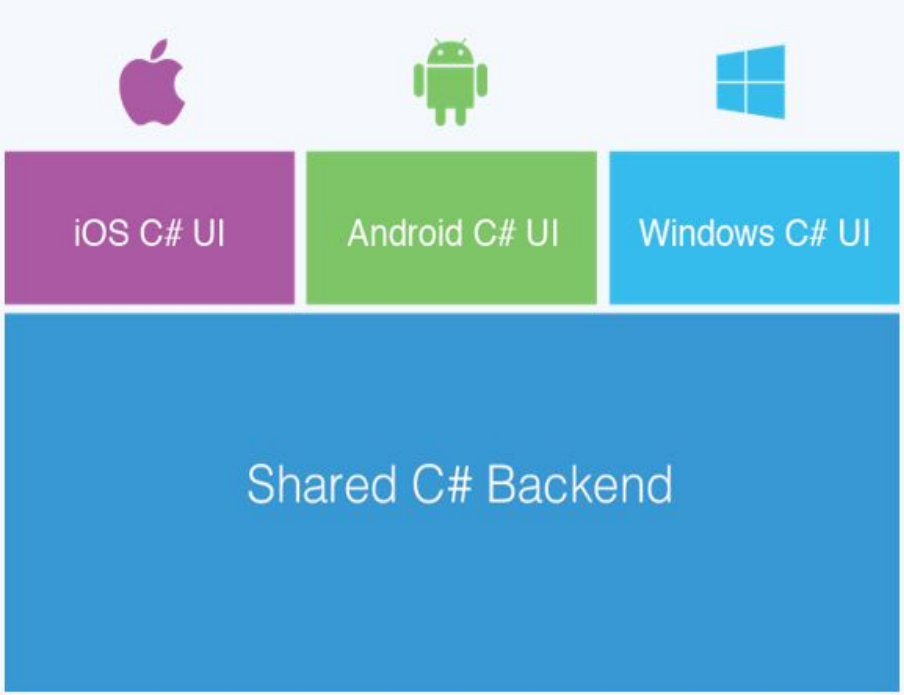
Xamarin Forms

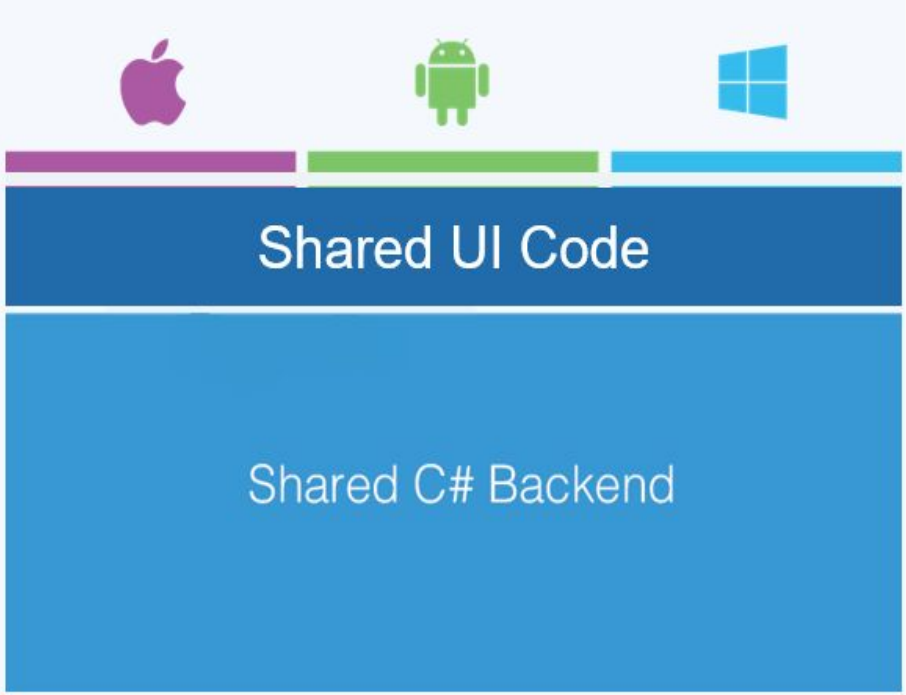
- Allows building of native UI's for iOS, Android, & Windows
- The UI's can be built using C#, XAML, or both
- Screens are represented by pages
- Pages contain various views (controls) that define the UI
- Pages and their views are rendered as native UI elements
- By connecting these views to shared backend code, we have a fully native iOS, Android, and Windows application built with shared C# code.
- Based on the application and technical design, we can achieve over 96% code reuse across platforms

Xamarin Forms (contd.)



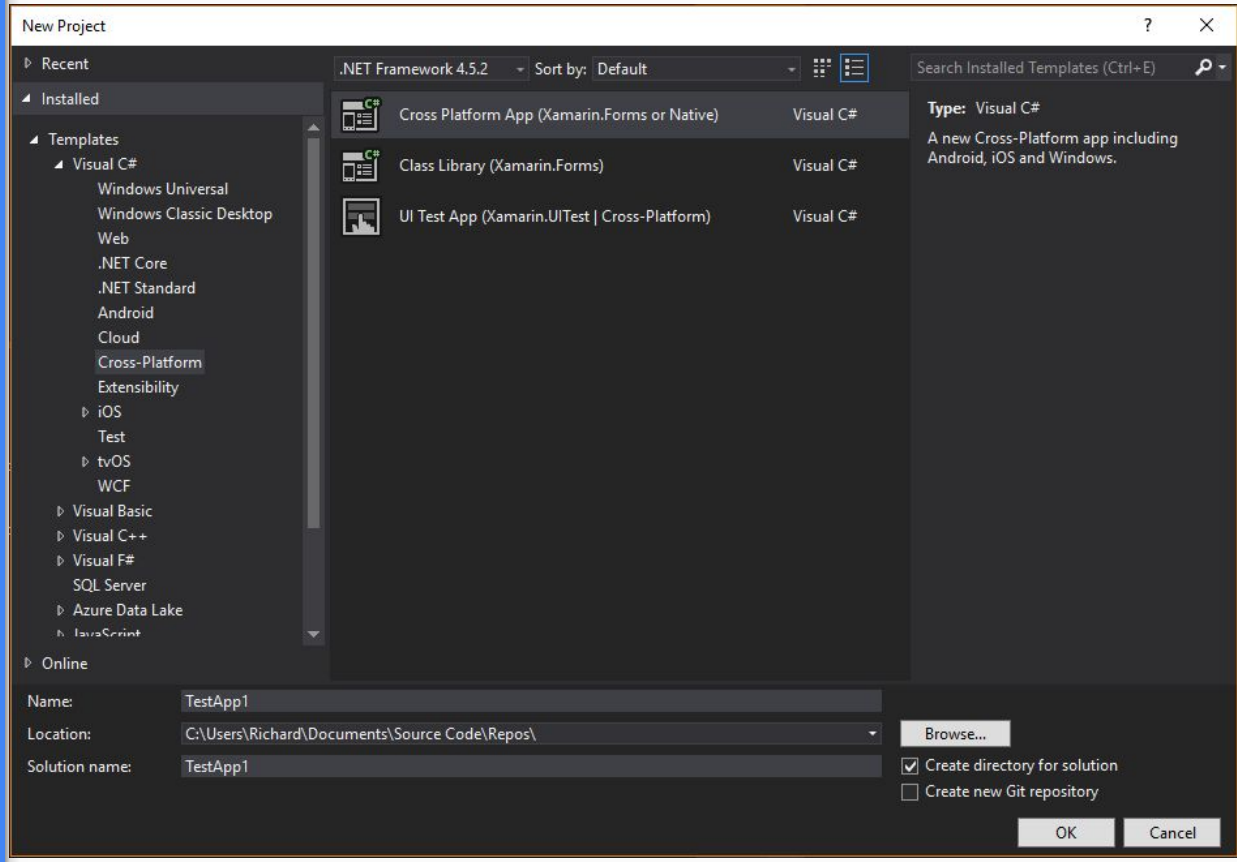
Xamarin <-> Xamarin Forms





Xamarin Forms Project

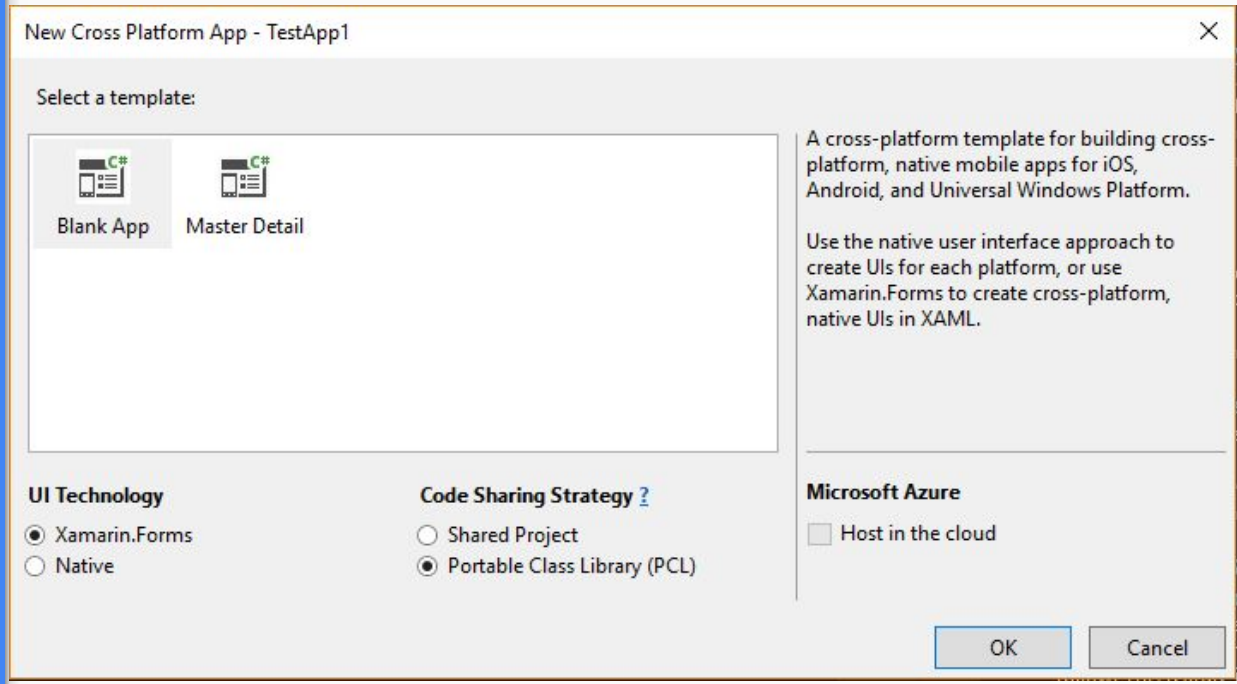
Visual Studio 2017



Richard Taylor - @rightincode

Xamarin Forms Project

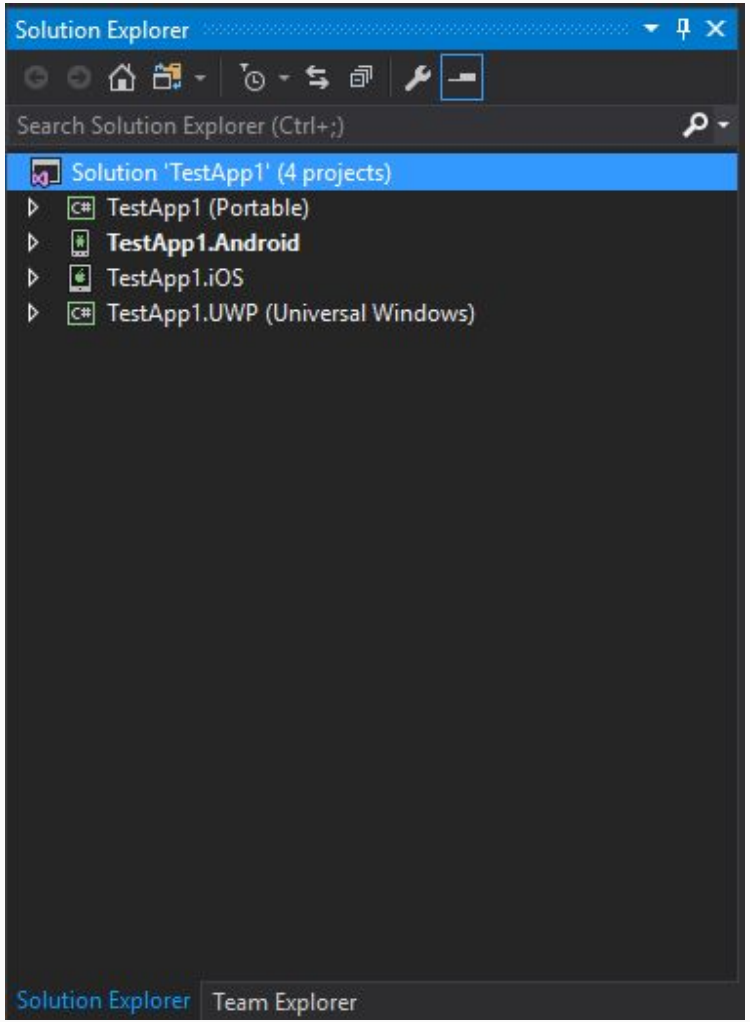
Visual Studio 2017



Richard Taylor - @rightincode

Xamarin Forms Project

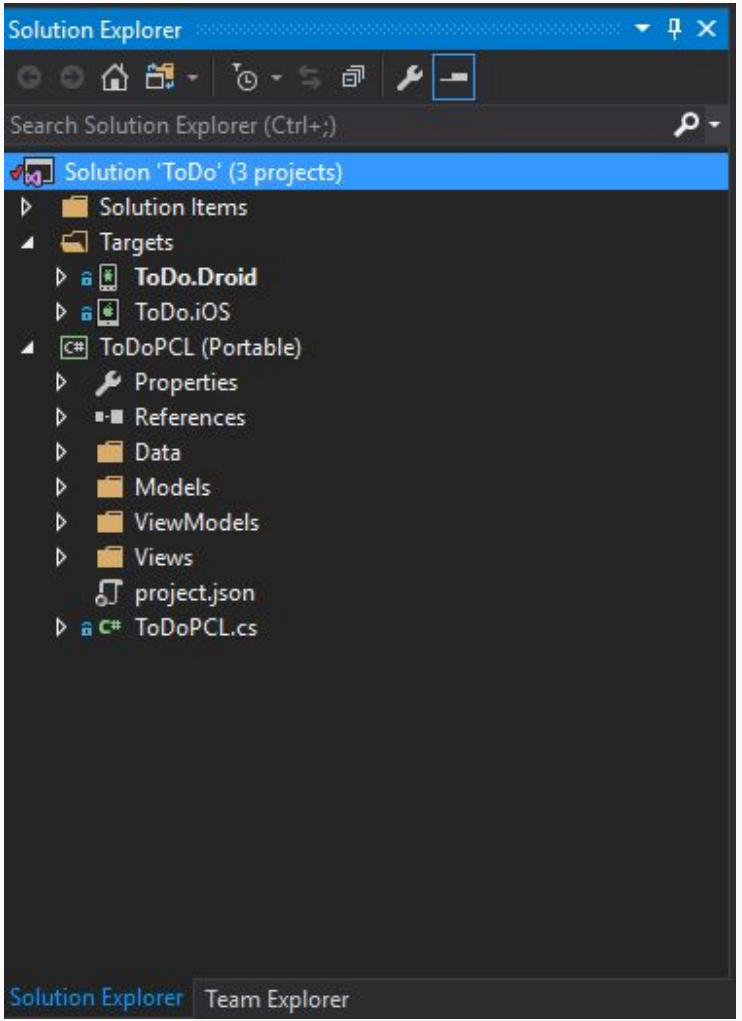
Visual Studio 2017
(Default Project Structure)



Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
(Recommended Project Structure)

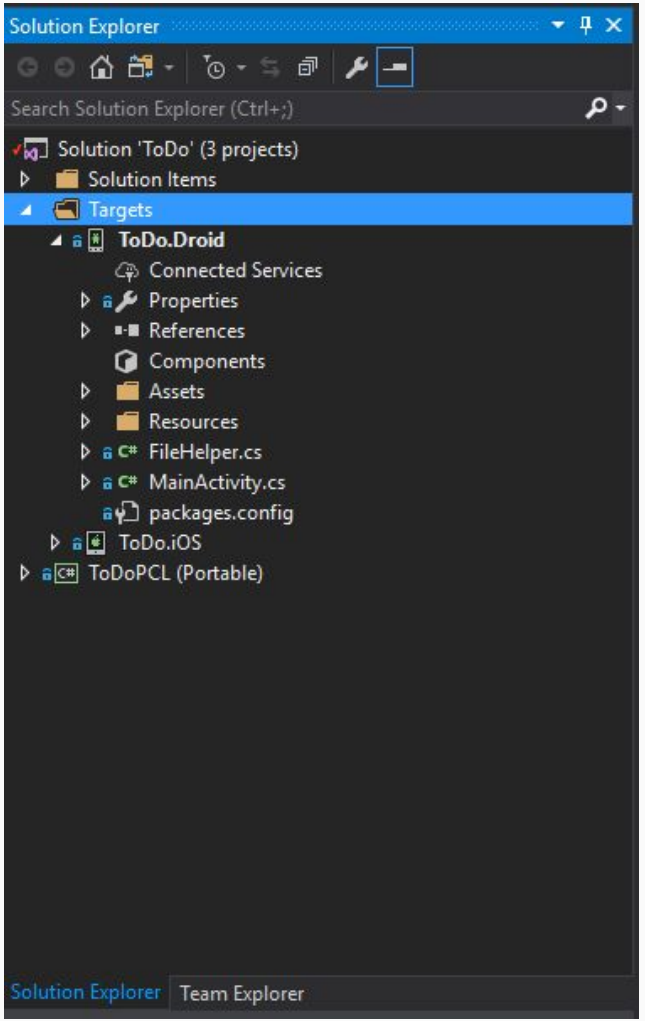


Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017

Android Project

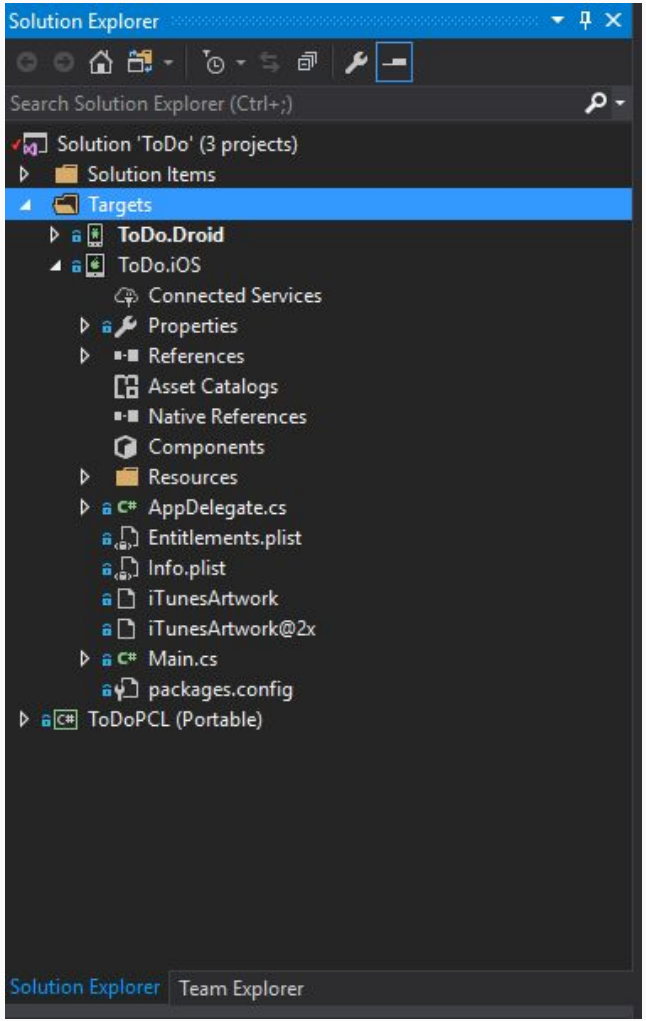


Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017

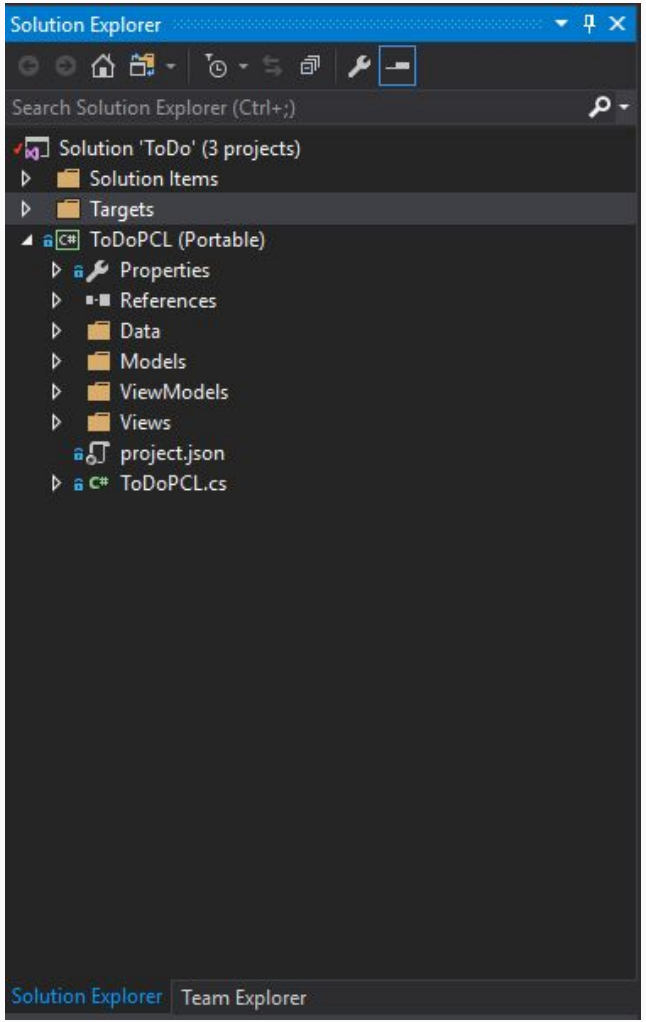
iOS Project



Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
Portable Class Library Project



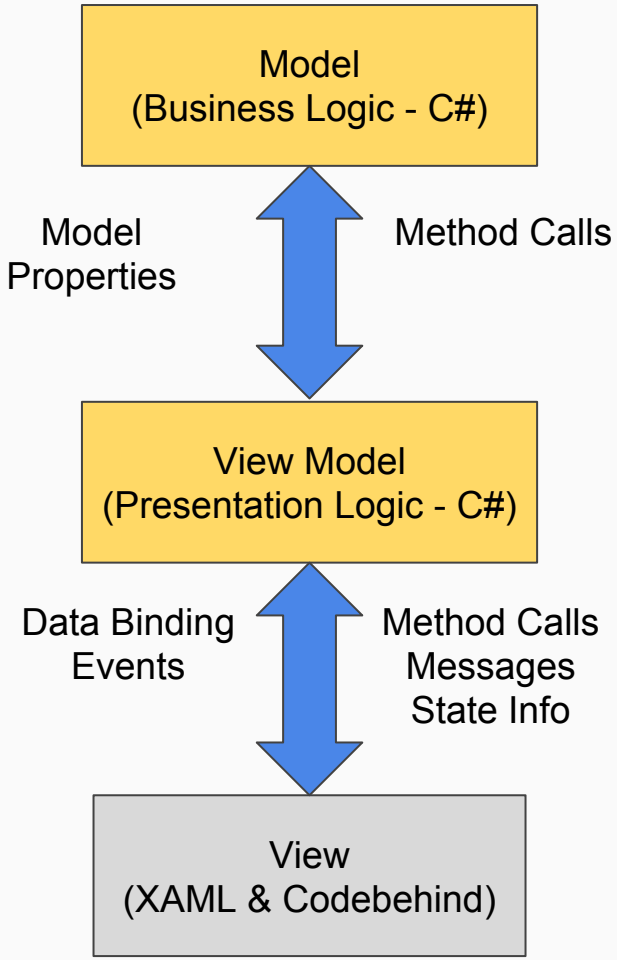
Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017

MVVM Design Pattern

- Model
- View Model
- View

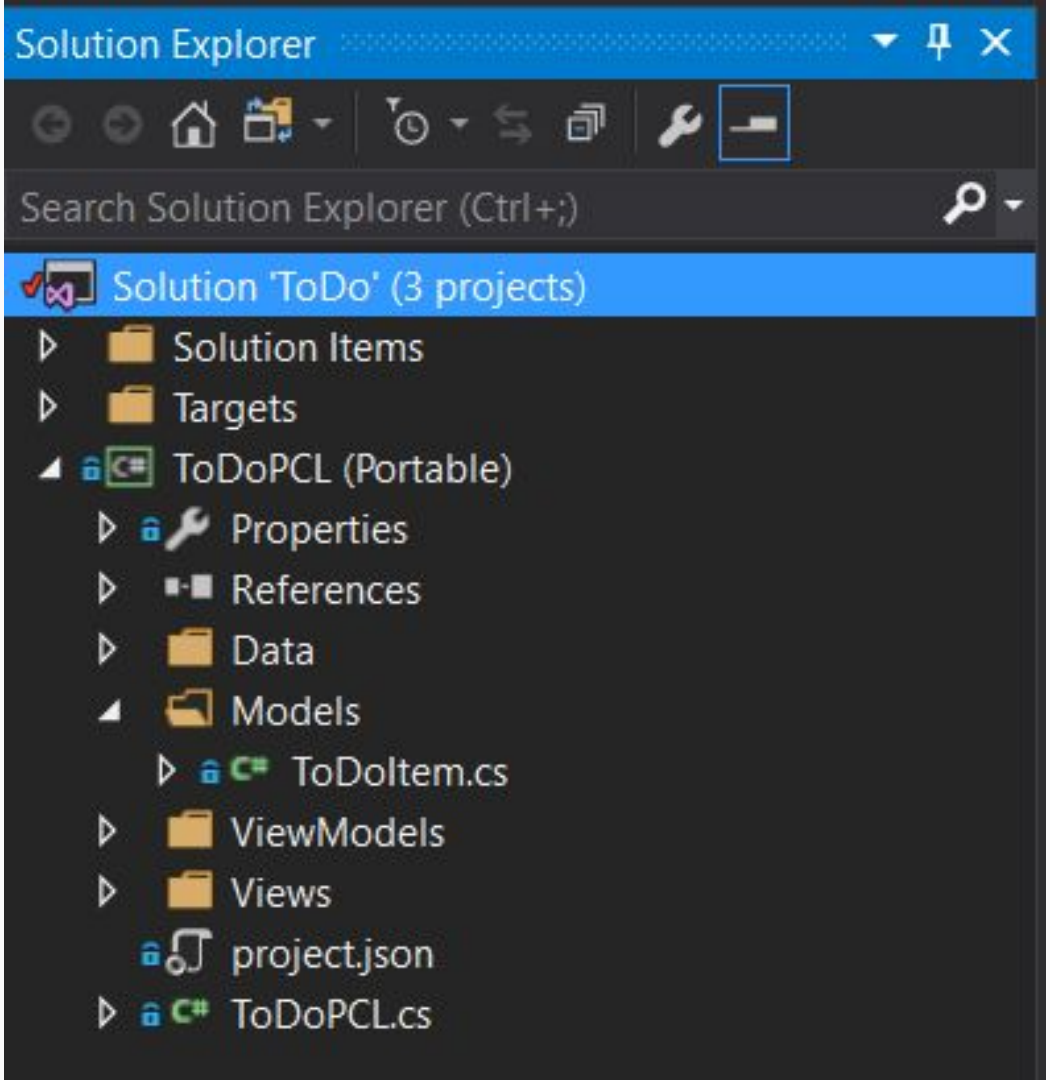


Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
Portable Class Library Project

Models Folder

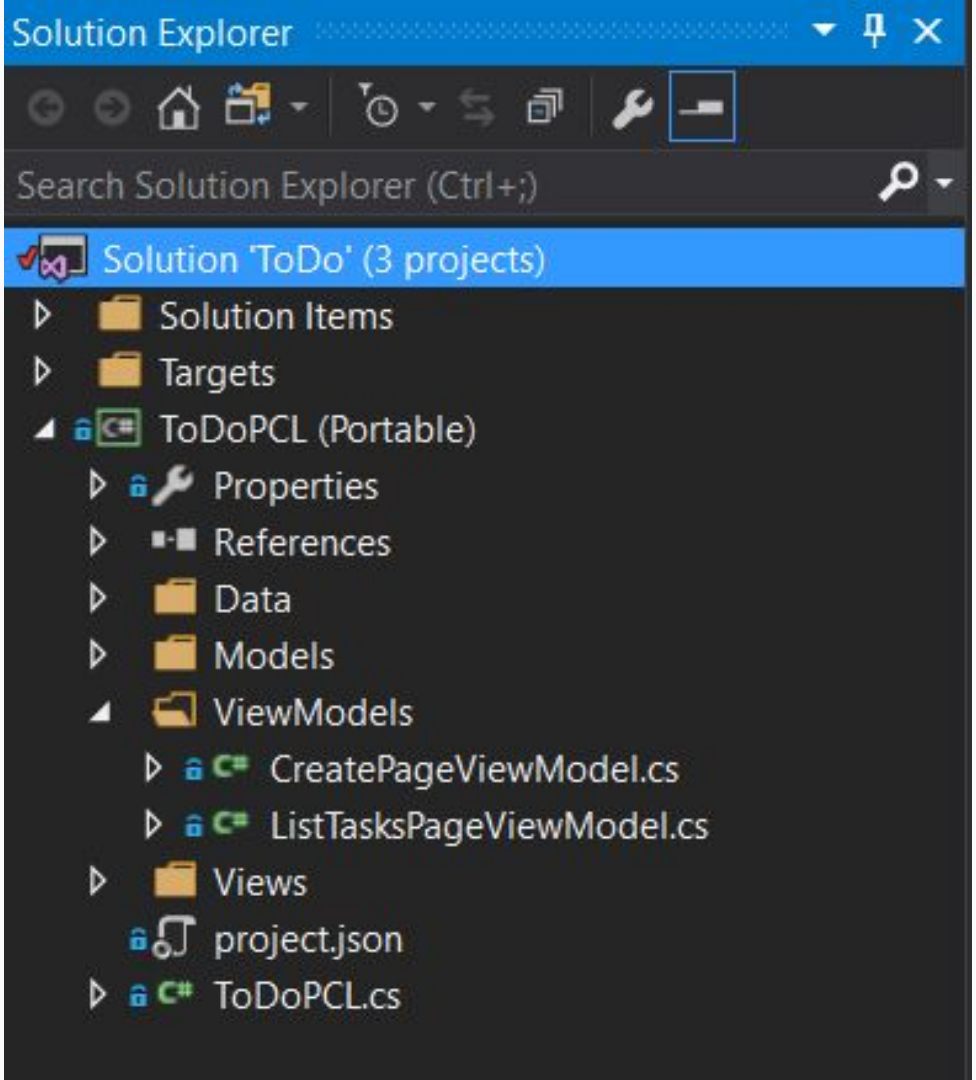


Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
Portable Class Library Project

ViewModels Folder

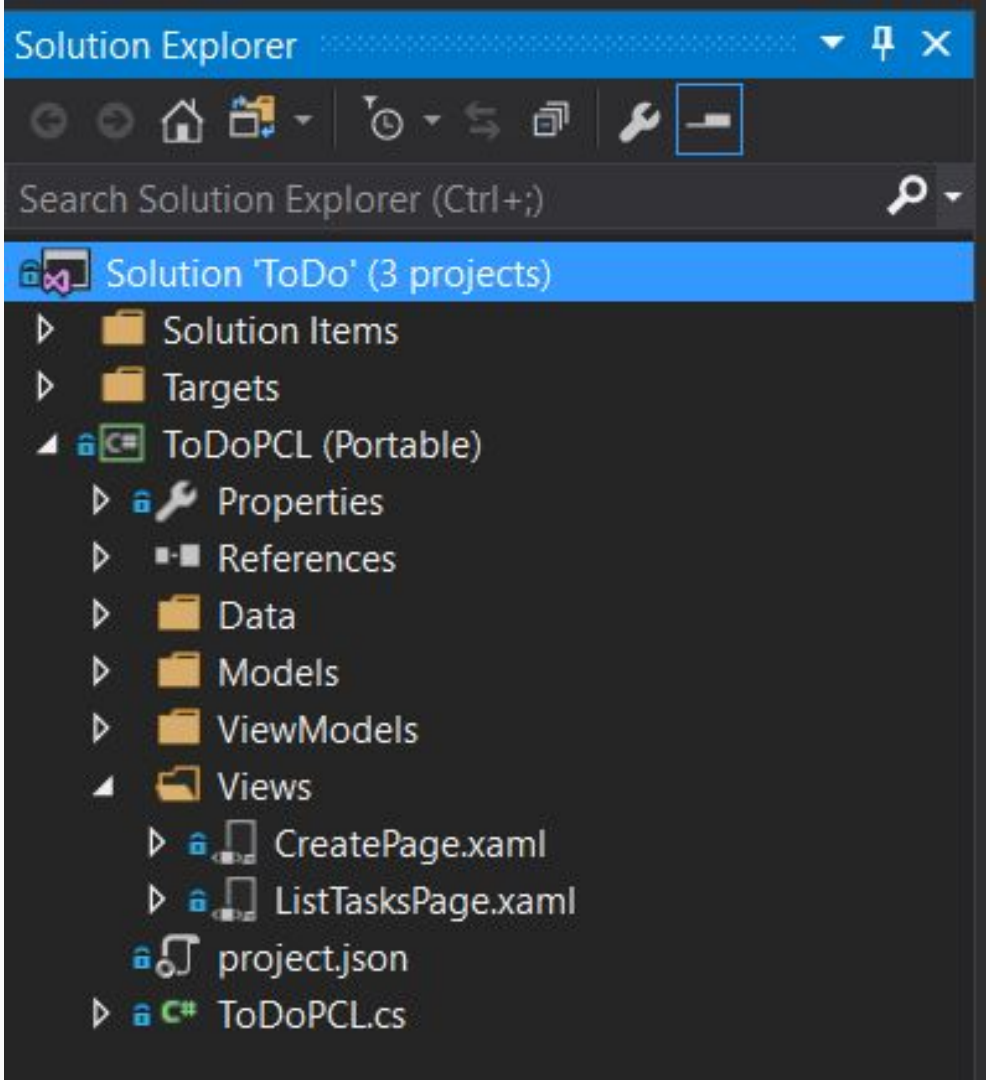


Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
Portable Class Library Project

Views Folder



Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
Portable Class Library Project

Views - XAML Pages

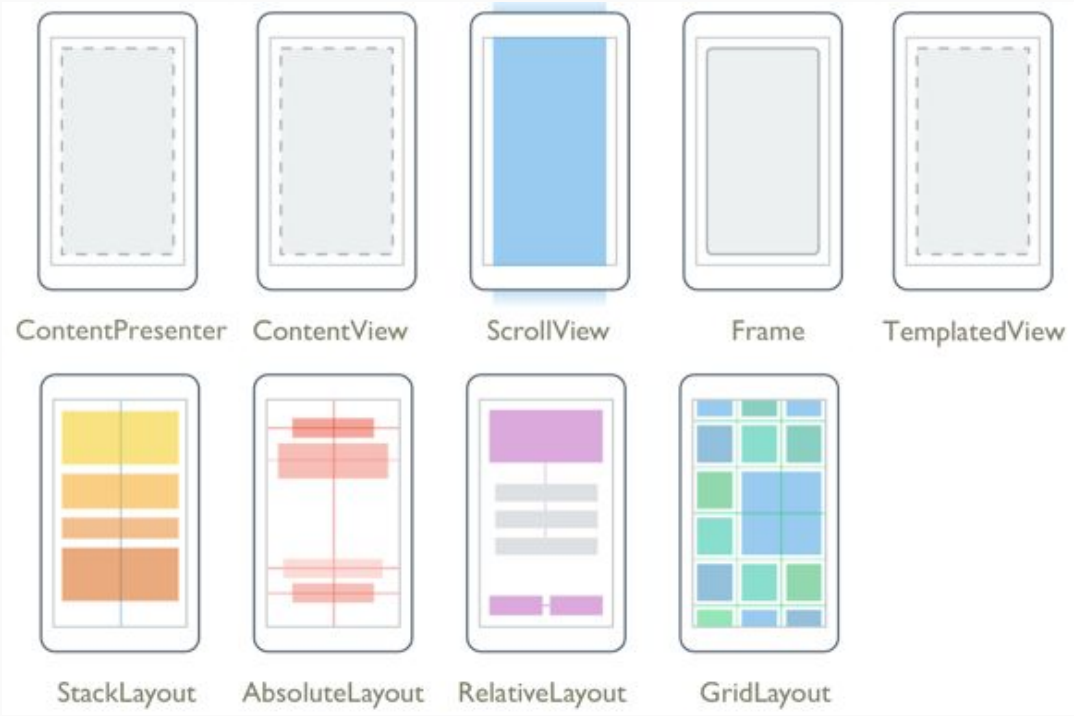
https://developer.xamarin.com/guides/xamarin-forms/user-interface/controls/pages/

Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
Portable Class Library Project

Views - XAML Layouts



https://developer.xamarin.com/guides/xamarin-forms/user-interface/controls/layouts/

Richard Taylor - @rightincode

Xamarin Forms Project

Visual Studio 2017
Portable Class Library Project

Views - XAML Views

ActivityIndicator	BoxView	Button	DatePicker
Editor	Entry	Image	Label
ListView	OpenGLView	Picker	ProgressBar
SearchBar	Slider	Stepper	Switch
TableView	TimePicker	WebView	

https://developer.xamarin.com/guides/xamarin-forms/user-interface/controls/views/

Richard Taylor - @rightincode

Xamarin Forms

Consuming a RESTful Web Service

- HttpClient
- Exchanging Data
  - HTTP Verbs
    - Get
    - Post
    - Put
    - Delete
  - JSON

Richard Taylor - @rightincode

Xamarin Forms

Consuming a RESTful Web Service

HttpClient

```
public class RestService : IRestService
{
  HttpClient client;
  ...

  public RestService()
  {
    client = new HttpClient();
    client.MaxResponseContentBufferSize = 256000;
    ...
  }
  ...
}
```

Xamarin Forms

Consuming a RESTful Web Service

Requesting Data

```
public async Task<List<TodoItem>> GetTodoItemList()
{
  ...
  // RestUrl = http://app.todo.com/api/todoitems/{0}
  var uri = new Uri (string.Format (Constants.RestUrl,
string.Empty));
  ...
  var response = await client.GetAsync(uri);
  if (response.IsSuccessStatusCode) {
      var content = await response.Content.ReadAsStringAsync();
      Items = JsonConvert.DeserializeObject <List<TodoItem>>
(content);
  }
  ...
}
```

Xamarin Forms

Consuming a RESTful Web Service

Creating Data

```
public async Task SaveTodoItemAsync (TodoItem item, bool isNewItem = false)
{
  // RestUrl = http://app.todo.com/api/todoitems/{0}
  var uri = new Uri (string.Format (Constants.RestUrl, item.ID));

  ...
  var json = JsonConvert.SerializeObject (item);
  var content = new StringContent (json, Encoding.UTF8, "application/json");

  HttpResponseMessage response = null;
  if (isNewItem) {
    response = await client.PostAsync (uri, content);
  }
  ...
  if (response.IsSuccessStatusCode) {
    Debug.WriteLine (@"              TodoItem successfully saved.");

  }
  ...
}
```

Richard Taylor - @rightincode

Xamarin Forms

Consuming a RESTful Web Service

Updating Data

```
public async Task SaveTodoItemAsync (TodoItem item, bool isNewItem = false)
{
  // RestUrl = http://app.todo.com/api/todoitems/{0}
  var uri = new Uri (string.Format (Constants.RestUrl, item.ID));
  ...
  var json = JsonConvert.SerializeObject (item);
  var content = new StringContent (json, Encoding.UTF8, "application/json");

  HttpResponseMessage response = null;
  if (isNewItem) {
    response = await client.PostAsync (uri, content);
  } else {

     response = await client.PutAsync (uri, content);

  }
  ...
  if (response.IsSuccessStatusCode) {
    Debug.WriteLine (@"             TodoItem successfully saved.");
  }
  ...
}
```

Xamarin Forms

Consuming a RESTful Web Service

Deleting Data

```
public async Task DeleteTodoItemAsync (int id)
{
  // RestUrl = http://app.todo.com/api/todoitems/{0}
  var uri = new Uri (string.Format (Constants.RestUrl,
id));
  ...
  var response = await client.DeleteAsync (uri);
  if (response.IsSuccessStatusCode) {
    Debug.WriteLine (@"              TodoItem successfully
deleted.");
  }
  ...
}
```

Demo - Xamarin Forms Application

Azure App Service

- Platform-as-a-service offering of Microsoft Azure
- It can be used to:
  - Create web and mobile apps for any platform or device
  - Integrate with SaaS solutions
  - Connect to on-premises applications
  - Automate business processes
- Apps created run on fully managed virtual machines in Azure

Azure App Service - Why Use?

- Support multiple languages and frameworks (ASP.NET, Node, Java, PHP, and Python)
- Supports DevOps optimization (continuous integration and deployment)
- Scale globally with high availability
- Selection of connectors that can be used with popular SaaS platforms and on-premises data
- Security and compliance
- Selection of application templates in the Azure Marketplace
- Visual Studio integration

Azure App Service - App Types

- **Web Apps** - Used to host websites and web applications
- **Mobile Apps** - Used to host mobile app back ends
- **Api Apps** - Used to host RESTful API’s
- **Logic Apps** - Used to automate business processes and integrating systems and data without writing code

Azure App Service - Mobile Apps

Demo - Azure App Service Mobile App

Resources

- Repo: (code and slides - branch: cltazurebootcamp)
  - https://github.com/rightincode/Xamarin-Forms-ToDo
- Xamarin:
  - https://developer.xamarin.com/guides/
- Xamarin Forms:
  - https://developer.xamarin.com/guides/xamarin-forms/
- Azure App Service:
  - https://docs.microsoft.com/en-us/azure/app-service/
- Mobile Apps:
  - https://docs.microsoft.com/en-us/azure/app-service-mobile/

Thanks!

Questions?