Xamarin Workshop Setup

Prerequisites

Introduction

Xamarin requires a few components to collaborate for a satisfying development experience. It is important that each participant to the workshop installs and tests the setup prior to the workshop day, so that we don't lose any time and can jump into code as soon as possible😊

Configurations

You can develop for Xamarin using Windows or Mac. However if you use Windows and want to build Xamarin apps for iOS, you need a Mac computer too. This is because of Apple restrictions and cannot be avoided. During the workshop, we will support the following configurations:

From a Windows PC:

- **Preferred:** Develop for Android devices using a physical device connected through USB.
- Develop for Android devices using an emulator running on the PC directly.

From a Mac OS computer:

- **Preferred:** Develop for Android devices using a physical device connected through USB.
- Develop for iOS devices using a physical device connected through USB.
- Develop for Android devices using an emulator running on the Mac directly.
- Develop for iOS devices using a simulator running on the Mac directly.

Keys to success:

- Bring your own Windows PC and/or Mac PC to the workshop. **Make sure to follow the instructions below for the Xamarin setup.**
- If you have one, we recommend that you bring your own Android or iOS devices to the workshop. This is the easiest way to test your application.

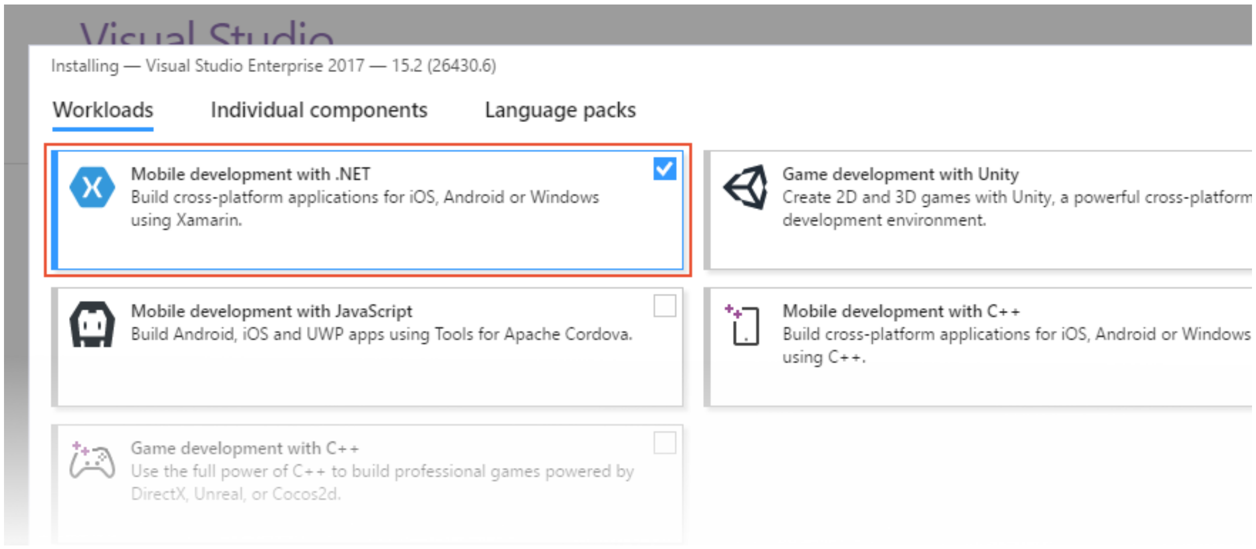
Prerequisites on Windows PC

You can see the requirements and follow the steps for Xamarin environment setup on Windows at: https://docs.microsoft.com/en-us/xamarin/cross-platform/get-started/installation/windows

1. Install Visual Studio 2017 from https://www.visualstudio.com/vs

Note: The free Community Edition is sufficient for this workshop.

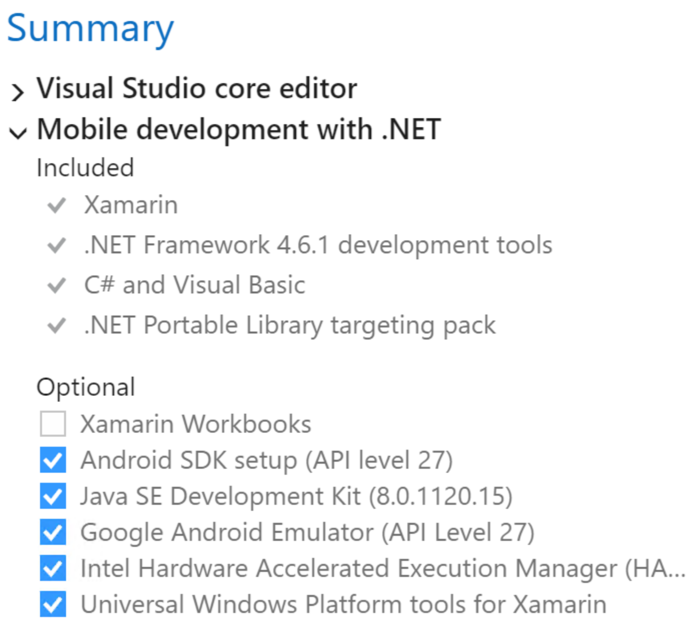
2. Make sure to select the "Mobile development with .NET" workload in the installer screen.



Installing — Visual Studio Enterprise 2017 — 15.2 (26430.6)

- [✓] Mobile development with .NET — Build cross-platform applications for iOS, Android or Windows using Xamarin.
- [ ] Game development with Unity — Create 2D and 3D games with Unity, a powerful cross-platform development environment.
- [ ] Mobile development with JavaScript — Build Android, iOS and UWP apps using Tools for Apache Cordova.
- [ ] Mobile development with C++ — Build cross-platform applications for iOS, Android or Windows using C++.
- [ ] Game development with C++ — Use the full power of C++ to build professional games powered by DirectX, Unreal, or Cocos2d.

3. You don't need all options selected but you should choose the followings:



Summary

> Visual Studio core editor

⌄ Mobile development with .NET

Included

- ✓ Xamarin
- ✓ .NET Framework 4.6.1 development tools
- ✓ C# and Visual Basic
- ✓ .NET Portable Library targeting pack

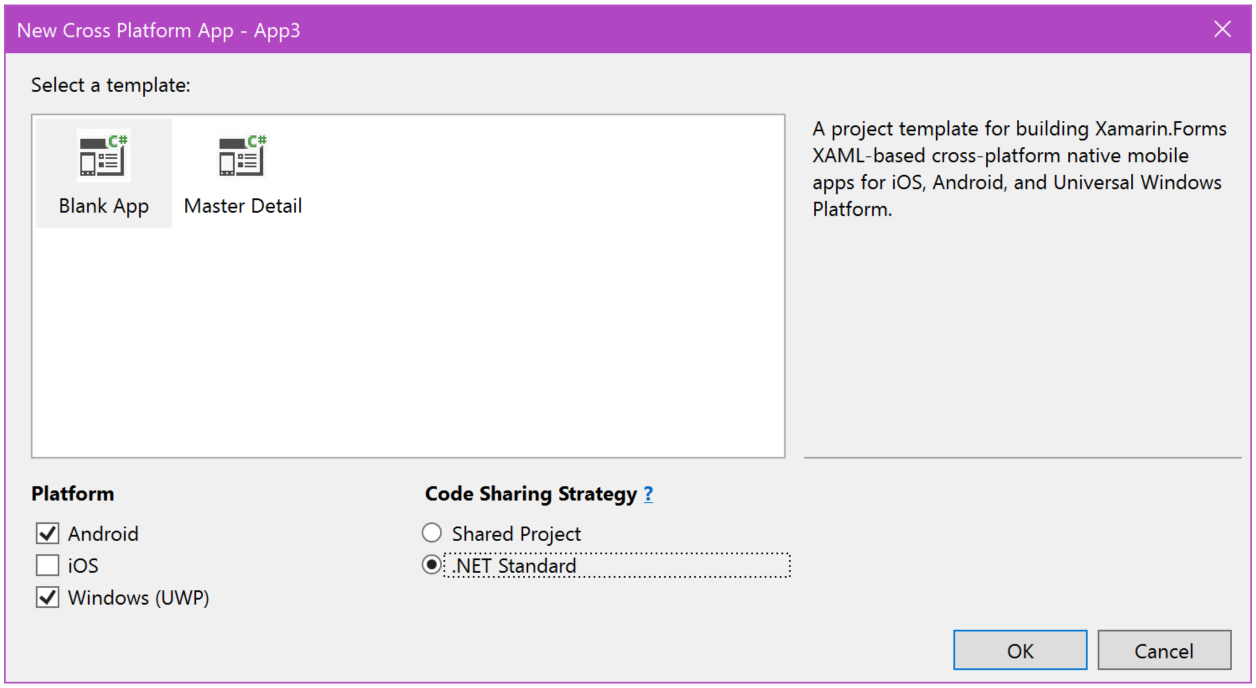
Optional

- [ ] Xamarin Workbooks
- [✓] Android SDK setup (API level 27)
- [✓] Java SE Development Kit (8.0.1120.15)
- [✓] Google Android Emulator (API Level 27)
- [✓] Intel Hardware Accelerated Execution Manager (HA...
- [✓] Universal Windows Platform tools for Xamarin

Testing the setup

After the installation is complete, test the setup with the following steps:

1. In Visual Studio, select File, New, Project.
2. In the “New Project” dialog, select the Cross-Platform category.
3. Select “Mobile App (Xamarin.Forms)”.
4. In the “New Cross Platform App” dialog, select “Blank App”.
5. Under Platform, select “Windows (UWP)” and “Android”. Deselect “iOS”.
6. Under Code Sharing Strategy, select “.NET Standard”.



A project template for building Xamarin.Forms XAML-based cross-platform native mobile apps for iOS, Android, and Universal Windows Platform.

Platform

- [✓] Android
- [ ] iOS
- [✓] Windows (UWP)

Code Sharing Strategy ?

- ○ Shared Project
- ● .NET Standard

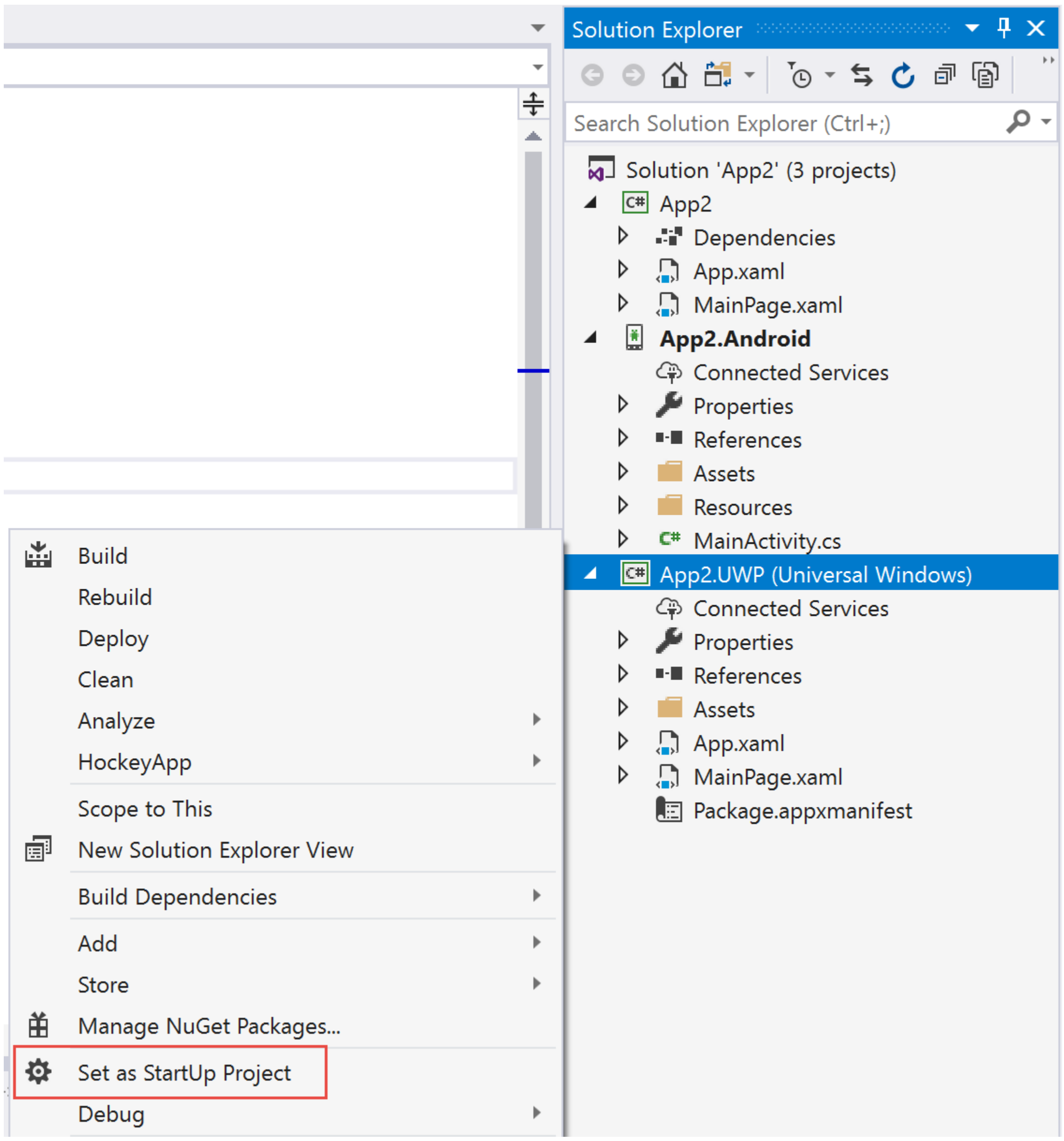
4. Press OK.

With the application created in Visual Studio, we will now test on Windows and on the Android emulator. If this works fine, we will guide you during the workshop to deploy to your devices.

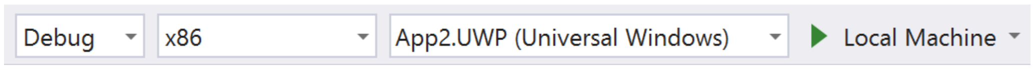
Running on Windows

Now we will try running the application in Windows and Android to see if everything works fine.

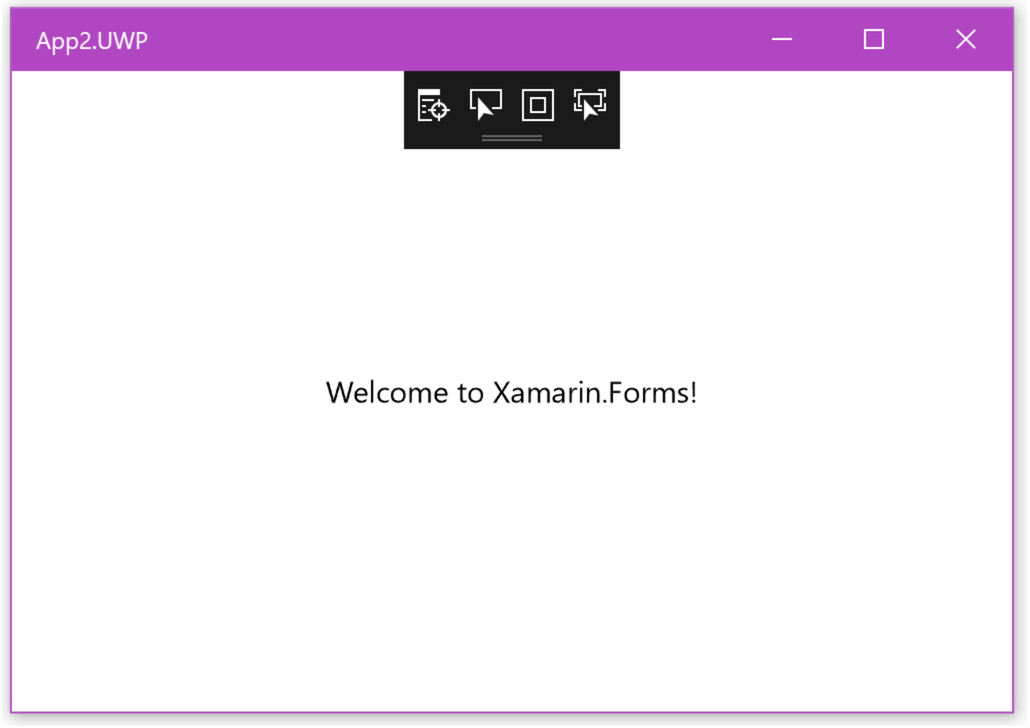
4.1. In the Solution Explorer, right click on the UWP version of the application and select “Set as Startup Project” from the context menu.



4.2. Click on the Start button with “Local Machine”.



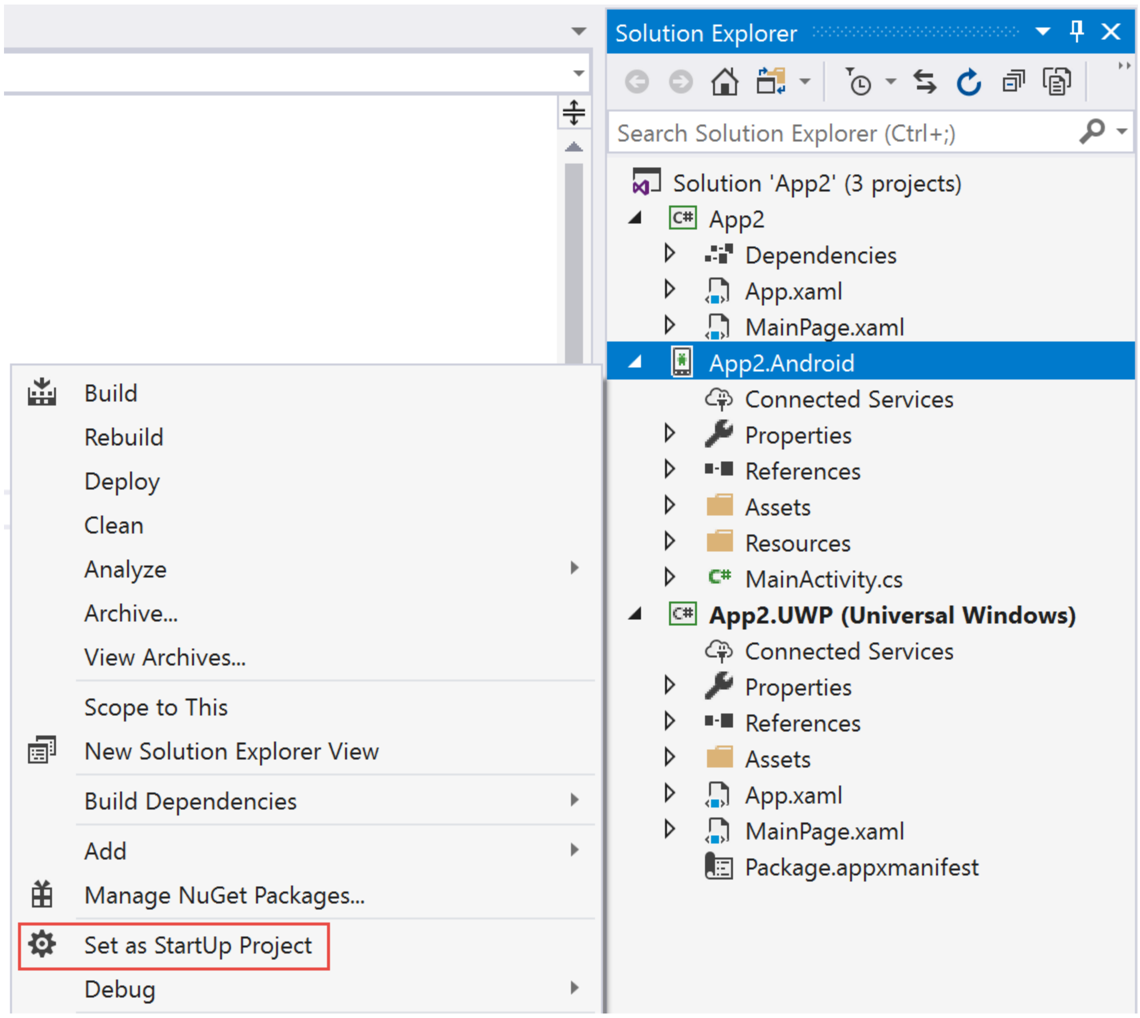
4.3. After a short wait, you should see the UWP version of the application running.



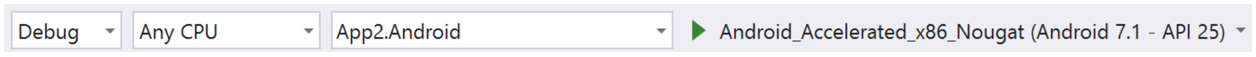
Welcome to Xamarin.Forms!

Running on Android

1. In the Solution Explorer, right click on the Android application and select “Set as Startup Project” from the context menu.



2. In the Start button, the Android emulator should be selected as shown below:

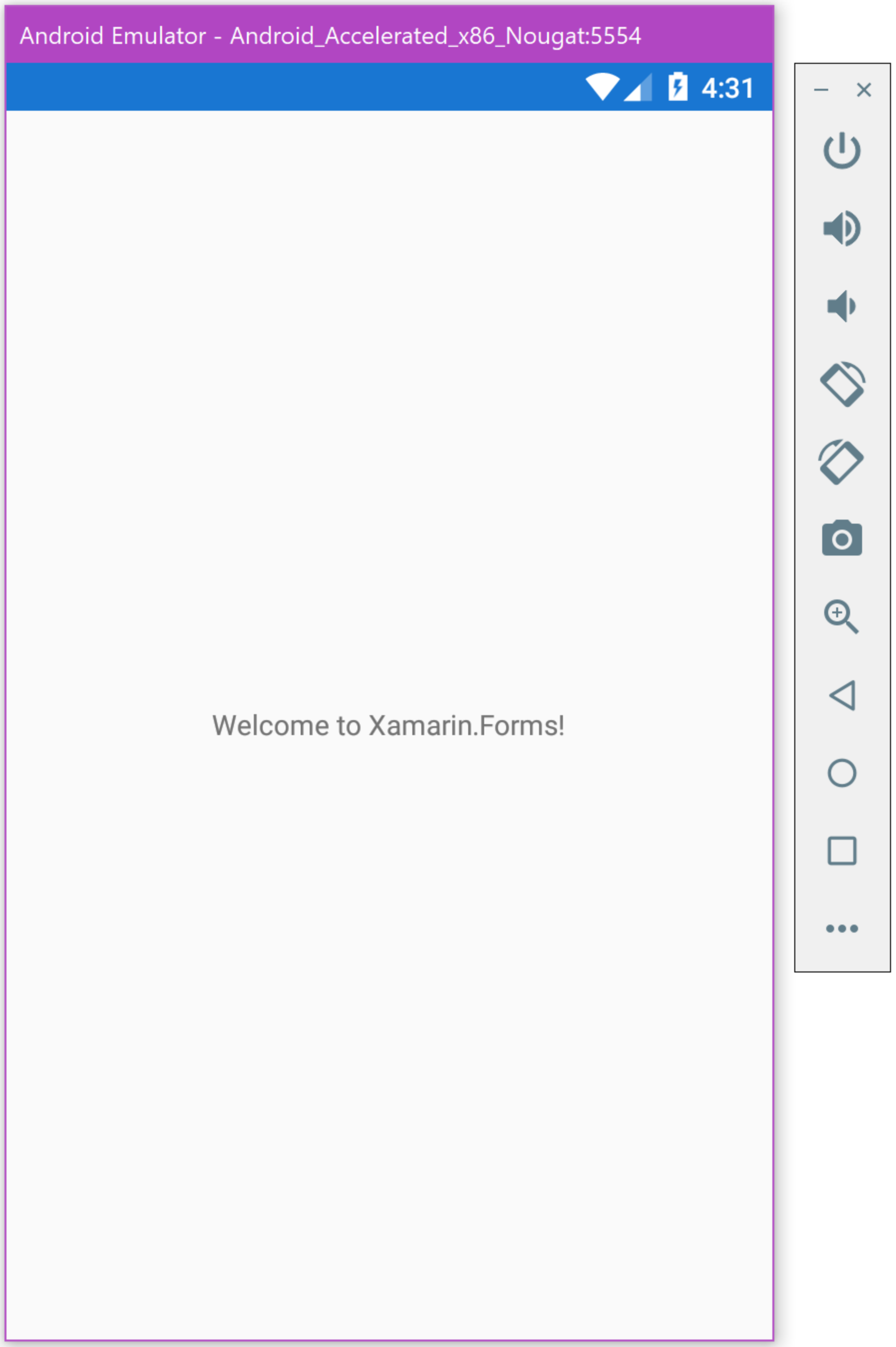


3. Click on the Start button to run the Android emulator.

Note 1: You might have to select a video device when the emulator starts for the first time. You don't have to select anything, you can just press OK.

Note 2: The emulator takes some time to boot, but once it is up and running, you don't need to shut it down. It can just stay up.

After the emulator runs, you should be able to see the application:



Welcome to Xamarin.Forms!

Prerequisites on Mac OS computer:

You can see the requirements and follow the steps for Xamarin environment setup on Windows at: https://docs.microsoft.com/en-us/visualstudio/mac/installation

1. Install Visual Studio 2017 from https://www.visualstudio.com/vs

Note: The free Community Edition is sufficient for this workshop.

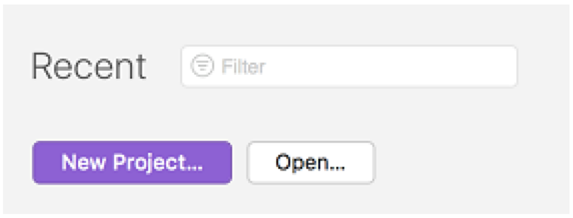
2. Go to the App Store and install XCode. Visual Studio requires XCode to build the iOS applications as well as run them on the Simulator. Make sure to select the last stable version of XCode (at the time of this writing, 9.3.1)

Important: After you finish installing XCode, run it once to make sure that all the necessary component are up to date. Some components are only downloaded the first time that XCode runs.

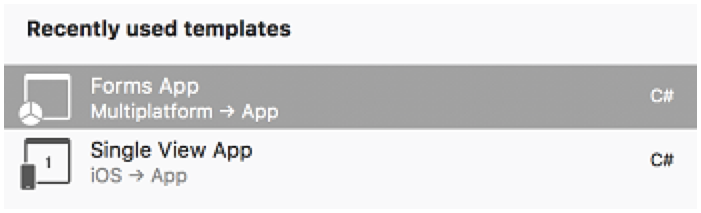
Testing the setup

After the installation is complete, test the setup with the following steps:

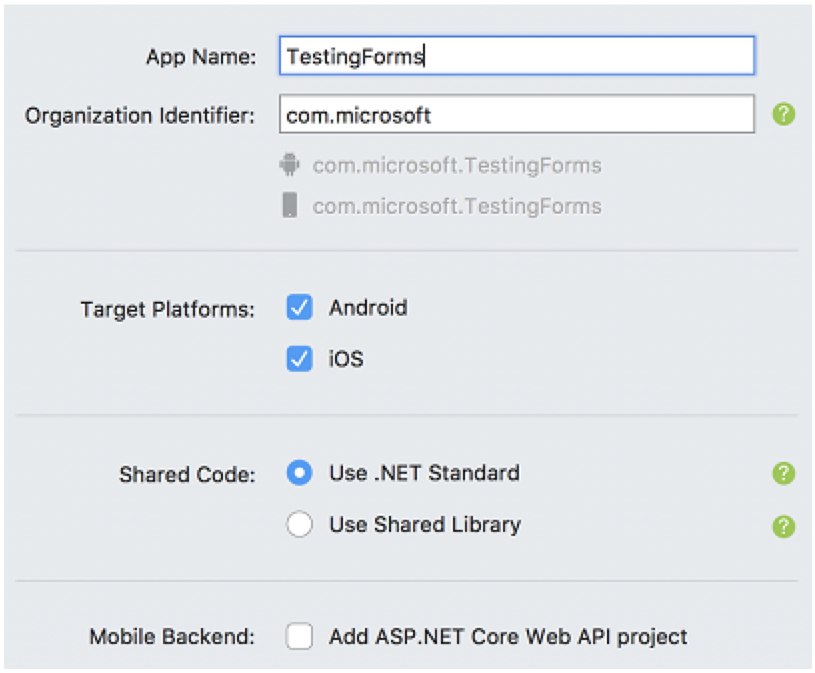
1. In Visual Studio for Mac, select New Project.



2. In the “Choose a template…” dialog, select Multiplatform, App, Forms App, then press Next.



3. Enter a name for the application and your company name, then press Next.



App Name: TestingForms

Organization Identifier: com.microsoft

com.microsoft.TestingForms

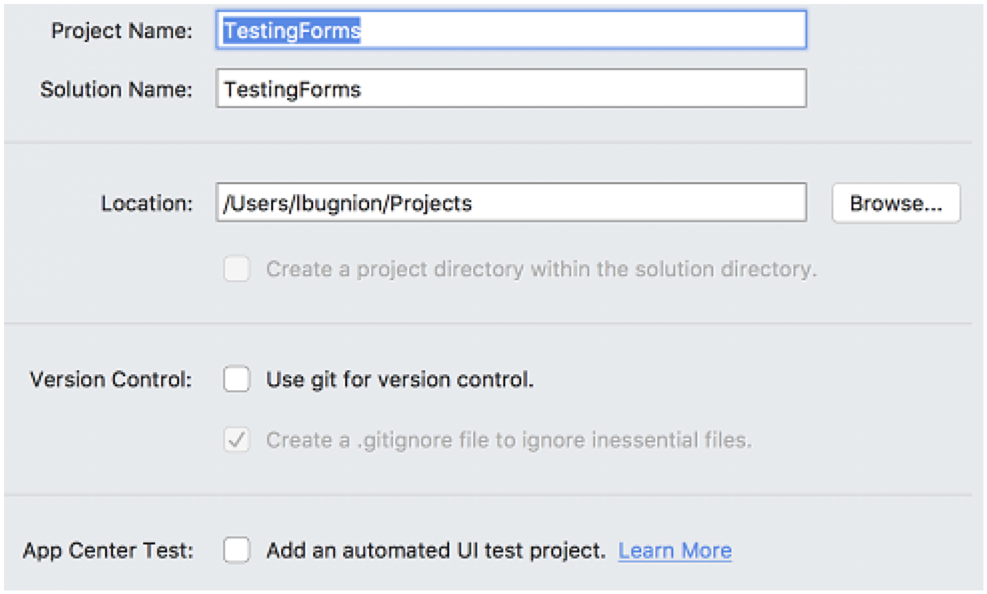
com.microsoft.TestingForms

Target Platforms: [✓ Android] [✓ iOS]

Shared Code: [✓ Use .NET Standard] Use Shared Library

Mobile Backend: Add ASP.NET Core Web API project

4. Check the project location and details, and press Create.



Project Name: TestingForms

Solution Name: TestingForms

Location: /Users/lbugnion/Projects

Create a project directory within the solution directory.

Version Control: Use git for version control.

[✓ Create a .gitignore file to ignore inessential files.]

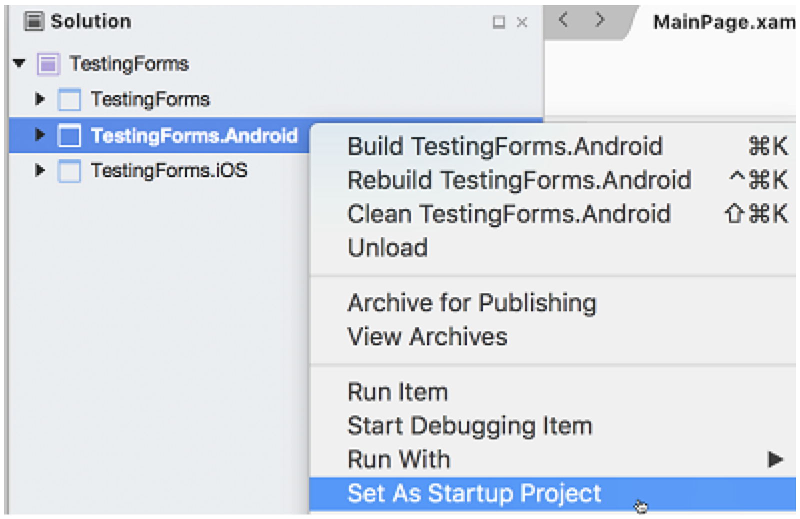
App Center Test: Add an automated UI test project. Learn More

With the application created in Visual Studio for Mac, we will now test on the Android emulator and the iOS simulator. If this works fine, we will guide you during the workshop to deploy to your devices.

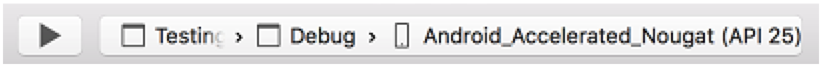
Note: It is possible that Visual Studio requires you to install the Android SDK (or updates to it) when you create the new application. You should accept and wait until the SDK is installed.

Running the test app in the Android emulator

1. In the Solution Explorer, right click on the Droid project and select Set as Startup Project from the context panel.

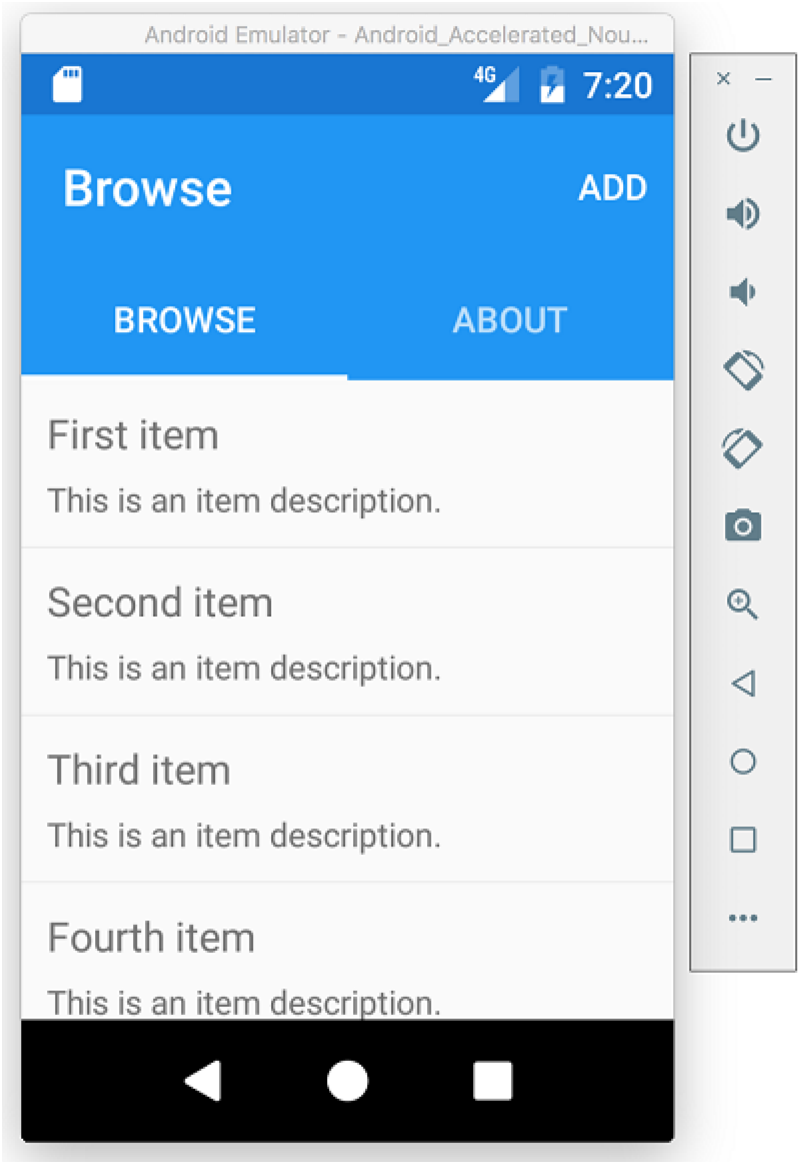


2. In the top bar, make sure that the Android emulator is selected (for example "Android_Accelerated_Nougat (API 25)").



3. Press the Run button (on the left with the triangle).

This will build the application and start the Android emulator. After a short wait, you should see the application running on the emulator.



First item

This is an item description.

Second item

This is an item description.

Third item

This is an item description.

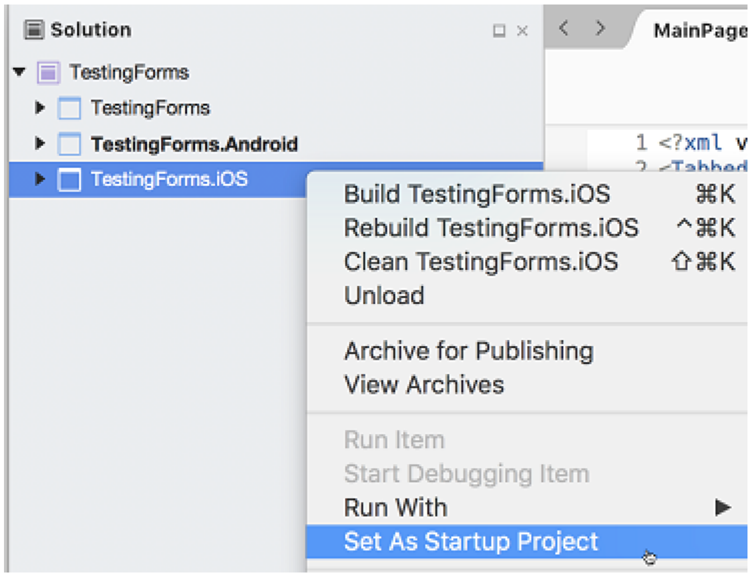
Fourth item

This is an item description.

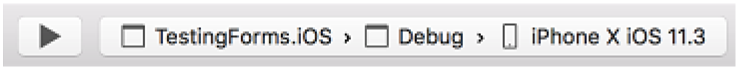
Running the test app in the iOS Simulator

In Visual Studio, stop the application (if it was running). Then follow the steps to run in the iOS simulator.

1. Right click on the iOS project in the Solution Explorer and select Set As Startup Project from the context menu.

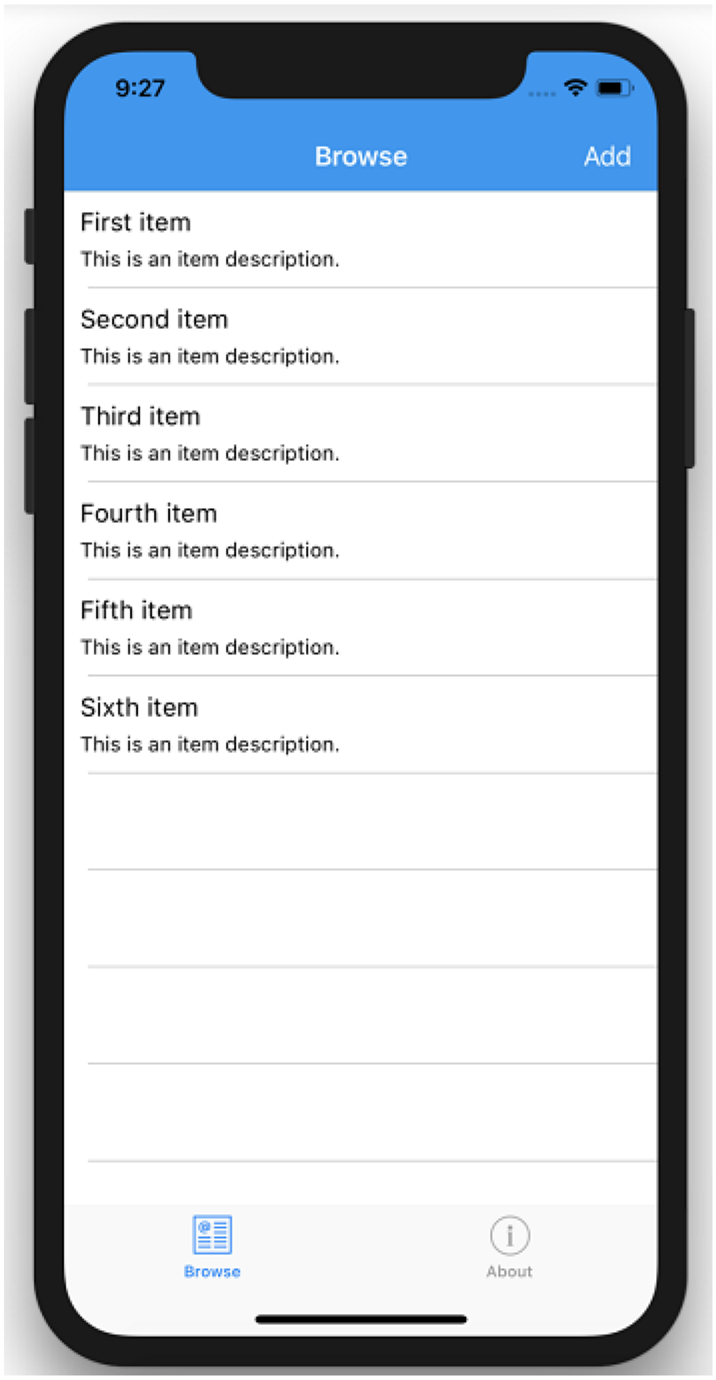


2. In the top bar, select the simulator that you want to test on (for example iPhone X iOS 11.3).



3. Press the Run button (on the left with the triangle).

This will build the iOS application and start the simulator. After a short wait you should be able to see the application in the simulator.



First item
This is an item description.
Second item
This is an item description.
Third item
This is an item description.
Fourth item
This is an item description.
Fifth item
This is an item description.
Sixth item
This is an item description.

Making an Android device a developer device

The last step of preparation you can do is to make your Android device (if available) a developer device. This will ensure that you can load the applications to the device, debug on the device, etc. Please follow the steps:
More information at https://developer.android.com/studio/run/device

1. Open the Settings app.
2. (Only on Android 8.0 or higher) Select System.

3. Scroll to the bottom and select About phone.
4. Scroll to the bottom and tap Build number 7 times.
5. Return to the previous screen to find Developer options near the bottom.
6. Select Developer options, and then enable USB debugging.