

A close-up photograph of a dark brown plush monkey toy. The monkey has a light brown face with a simple black line for a smile. It is wearing a black long-sleeved hoodie. The word "Xamarin" is printed in white on the front of the hoodie. The background is dark and slightly blurred.

XAM101

Getting Started with Xamarin

Download class materials from
university.xamarin.com



Xamarin University

Information in this document is subject to change without notice. The example companies, organizations, products, people, and events depicted herein are fictitious. No association with any real company, organization, product, person or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user.

Microsoft or Xamarin may have patents, patent applications, trademarked, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any license agreement from Microsoft or Xamarin, the furnishing of this document does not give you any license to these patents, trademarks, or other intellectual property.

© 2014-2017 Xamarin Inc., Microsoft. All rights reserved.

Xamarin, MonoTouch, MonoDroid, Xamarin.iOS, Xamarin.Android, Xamarin Studio, and Visual Studio are either registered trademarks or trademarks of Microsoft in the U.S.A. and/or other countries.

Other product and company names herein may be the trademarks of their respective owners.

Getting Started at Xamarin University

- ❖ To get the most out of your Xamarin University experience, be sure to watch the short welcome video which outlines the features and benefits



university.xamarin.com/getting-started



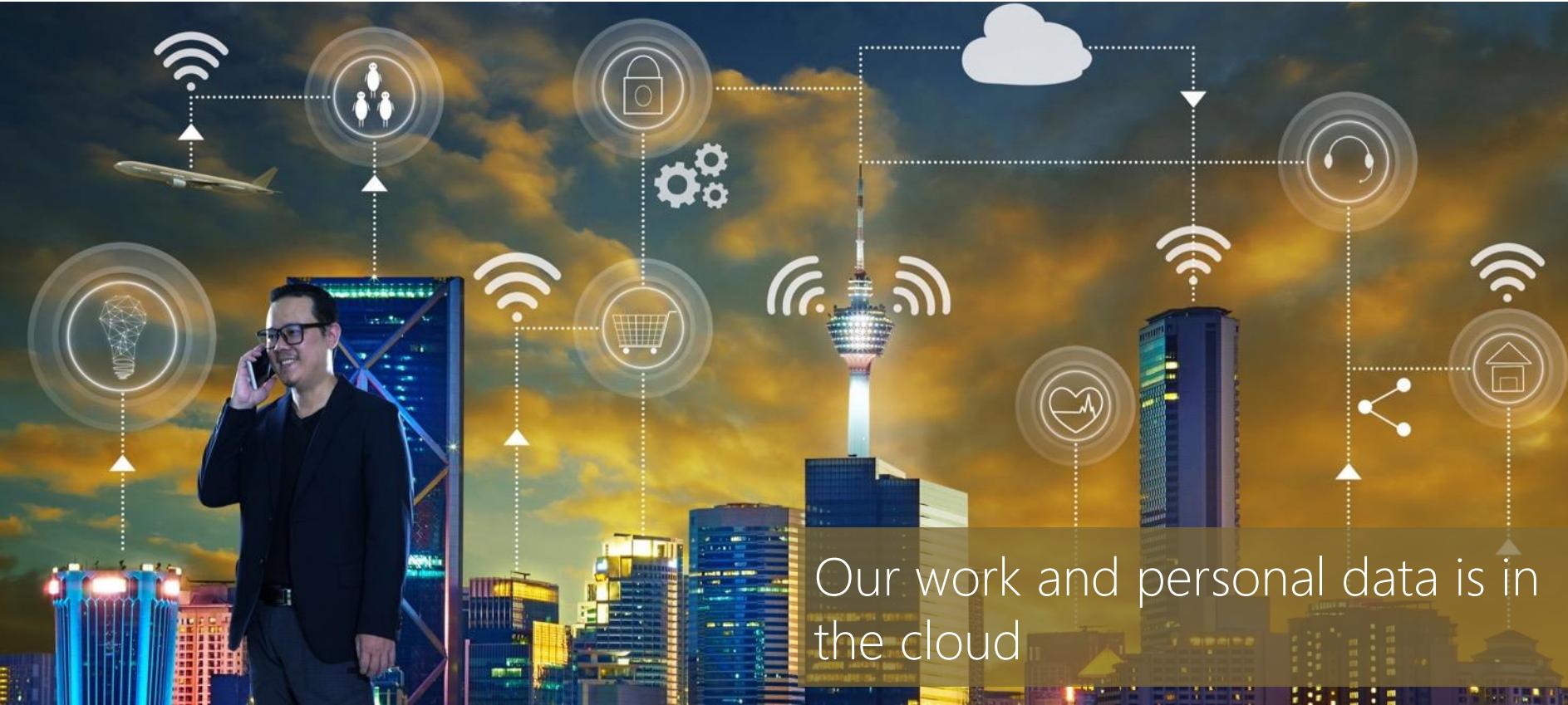
The Xamarin Approach

Tasks

- ❖ The state of mobile development today
- ❖ Discuss mobile app trends
- ❖ Identify approaches to development
- ❖ Discover the Xamarin Approach

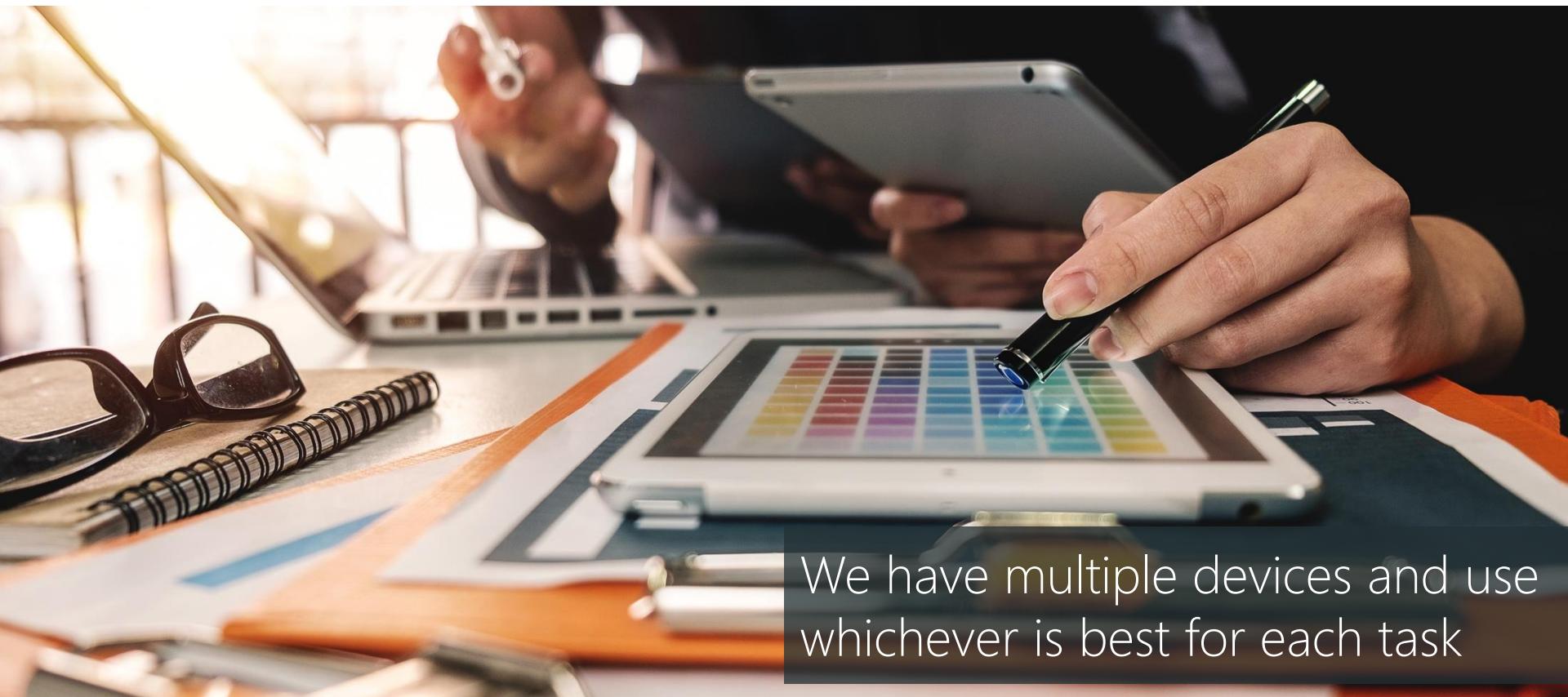


It's a mobile-first, cloud-first world



Our work and personal data is in
the cloud

It's a mobile-first, cloud-first world



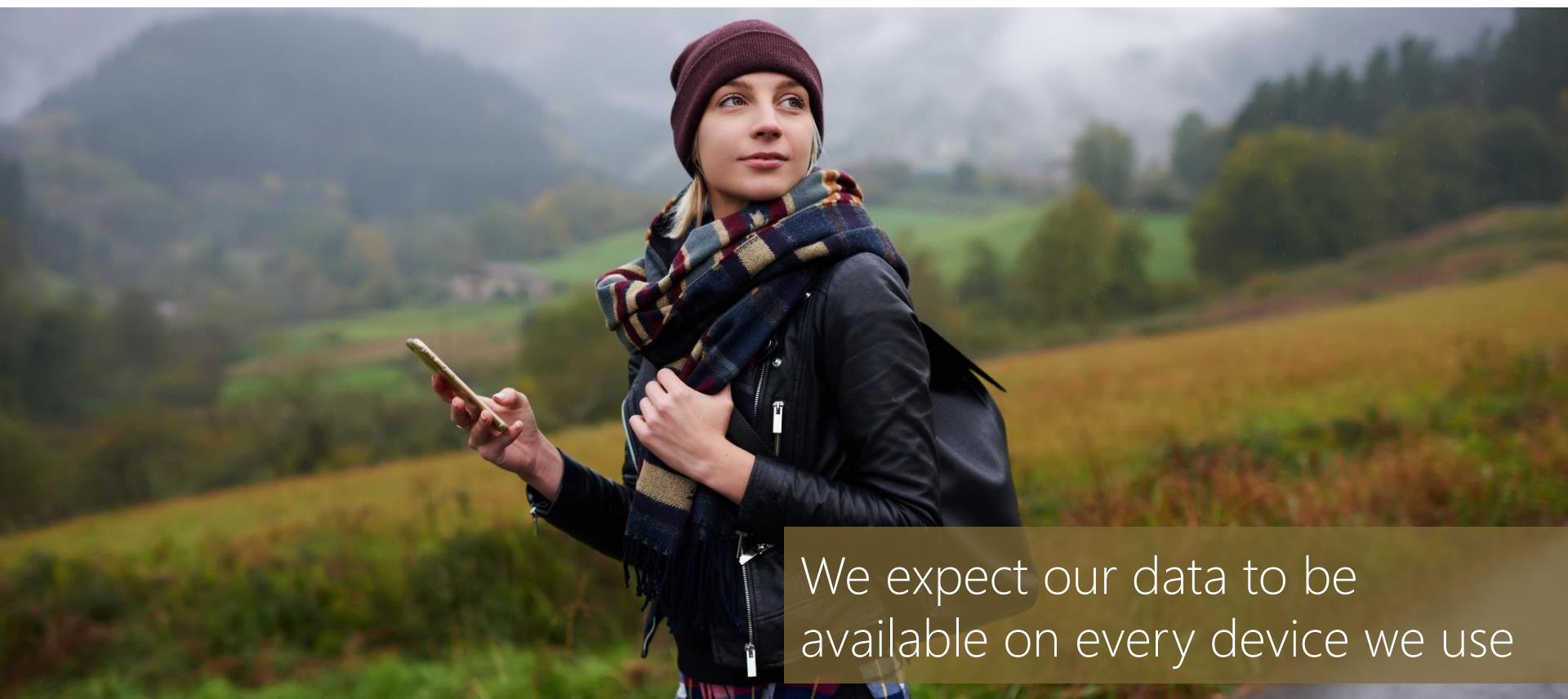
We have multiple devices and use whichever is best for each task

It's a mobile-first, cloud-first world



We use devices in all aspects of our lives

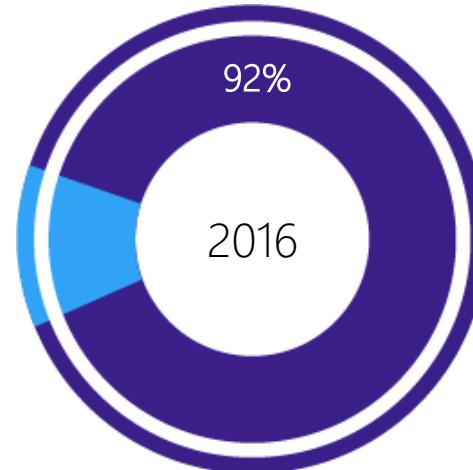
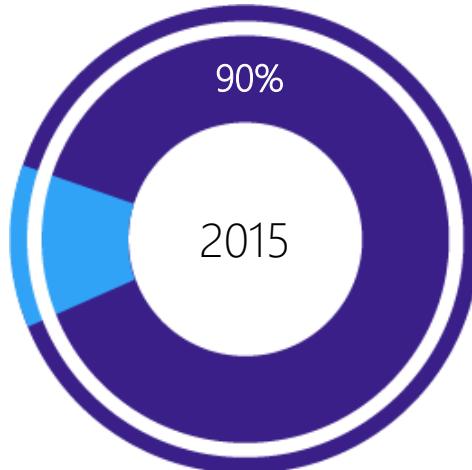
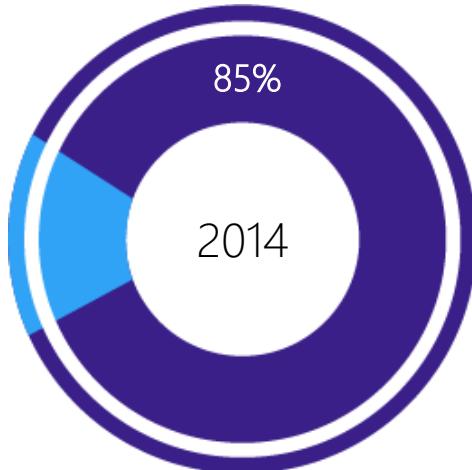
It's a mobile-first, cloud-first world



We expect our data to be available on every device we use

Mobile app trends

- ❖ Users prefer apps over browsers on their mobile devices



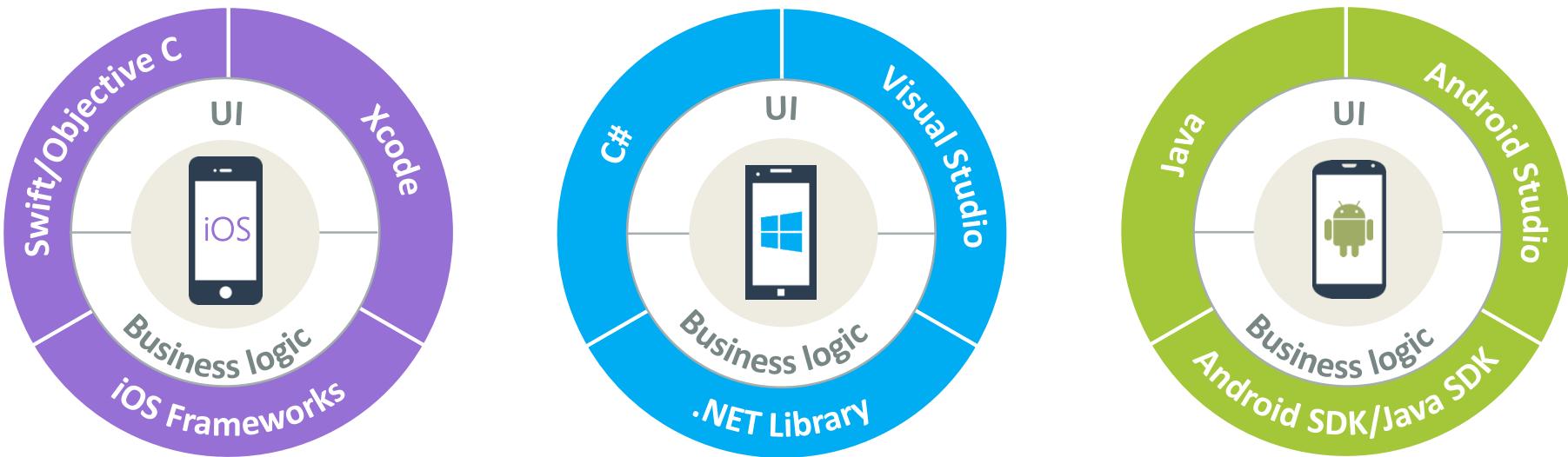
Mobile Web Browser



Apps

Traditional approach [definition]

- ❖ Traditionally, apps have separate code bases written in their native language, are built using native tools, and utilize platform-specific features



Traditional approach [cons]

- ❖ Traditional app development takes longer, requires multiple teams, multiple IDEs, and **cannot share code**

```
double ComputeTax(Item[] items)
{
    ...
    foreach (var item in items)
        ...
}
```

C#

```
func computeTax(items: [Item]) -> Double
{
    ...
    for item in items
        ...
}
```

Swift

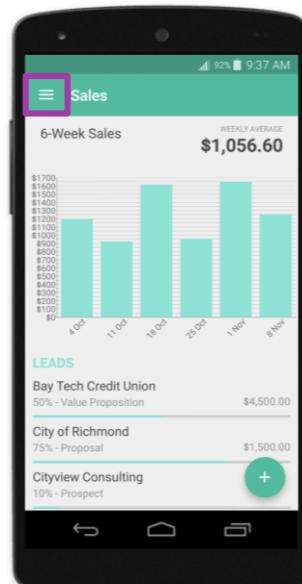
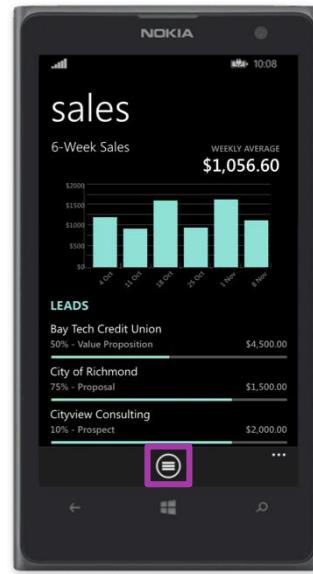
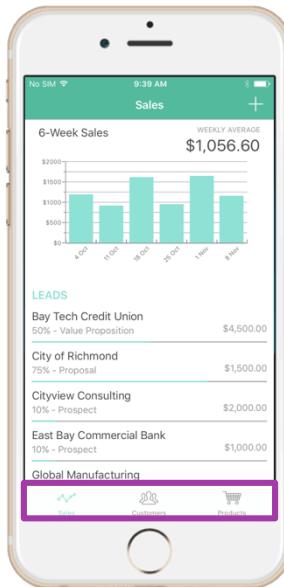
```
double computeTax(Item[] items)
{
    ...
    for (Item item : items)
        ...
}
```

Java

Traditional approach [pros]

- ❖ Traditional apps typically follow each platform's user-experience guidelines for things like navigation style, page layout, settings, etc.

E.g. implement the navigation style that users of each platform expect



What is Xamarin?

- ❖ Xamarin is an app-development platform that lets you build apps for many operating systems from a single, shared code base

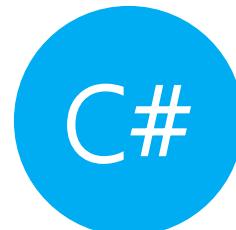


Xamarin tools

- ❖ You use Visual Studio, C#, and the .NET Libraries to build Xamarin apps



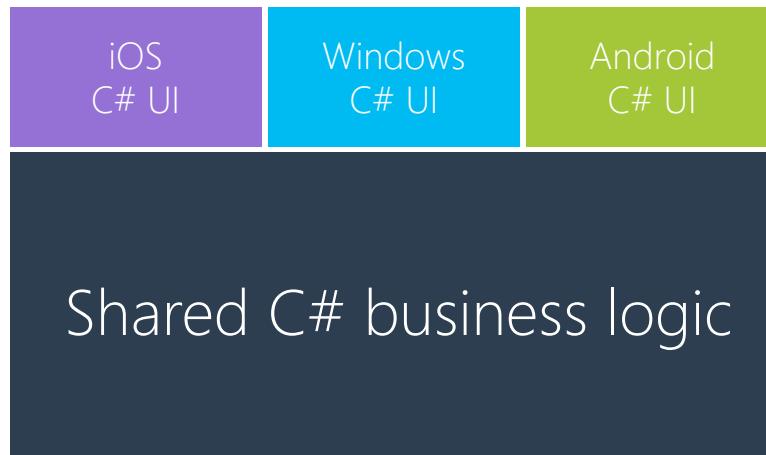
Visual Studio
for Windows or Mac



System.Net	System
System.Data	System.Windows
System.IO	System.Linq
System.Numerics	System.Core
System.Xml	System.ServiceModel

Xamarin development approaches

- ❖ Xamarin offers you two strategies: separate UI or shared UI



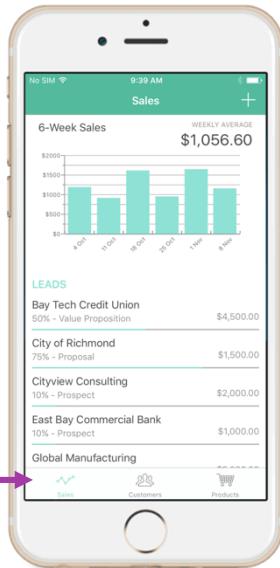
Xamarin.iOS and Xamarin.Android



Xamarin.Forms

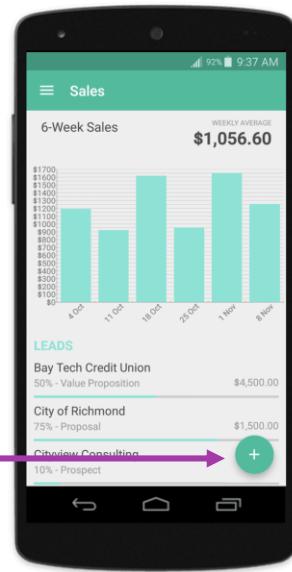
Xamarin.iOS and Xamarin.Android

- ❖ Create your business logic once and share it across platforms, while leveraging all of the native controls/features your users expect



Familiar navigation

Familiar controls



Shared code base

```
double ComputeTax(Item[] items)
{
    ...
    foreach (var item in items)
    {
        ...
    }
}
```

Xamarin.iOS – 100% API Coverage

- ❖ Anything you can do in Swift or Objective C for iOS you can do with Xamarin using C#

MapKit	UIKit	iBeacon	CoreGraphics	CoreMotion
System.Net	System	System.IO	System.Linq	System.Xml
System.Data	System.Windows	System.Numerics	System.Core	System.ServiceModel



100% API coverage with the added benefit of the .NET APIs

Xamarin.Android – 100% API Coverage

- ❖ Anything you can do in Java for Android you can do with Xamarin using C#

Text-to-speech	Toolbar	Printing Framework	Renderscript	NFC
System.Net	System	System.IO	System.Linq	System.Xml
System.Data	System.Windows	System.Numerics	System.Core	System.ServiceModel



100% API coverage with the added benefit of the .NET APIs

Windows

- ❖ Windows apps are built in C# with all of the Native APIs

Microsoft.Phone	Microsoft.Networking	Windows.Storage	Windows.Foundation	Microsoft.Devices
System.Net	System	System.IO	System.Linq	System.Xml
System.Data	System.Windows	System.Numerics	System.Core	System.ServiceModel



Windows apps support C# natively

Platform libraries

- ❖ Xamarin provides a C# version of every native library type

```
public class TextView : View ...  
{ ...  
    public string Text { get; set; }  
    public event EventHandler<TextChangedEventArgs> TextChanged;  
}
```

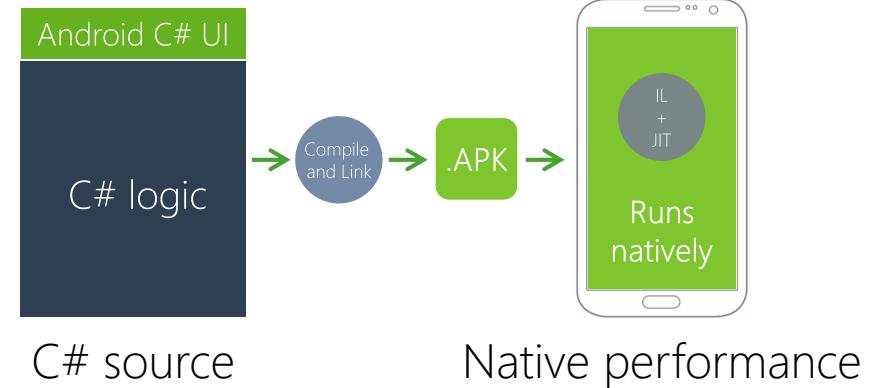
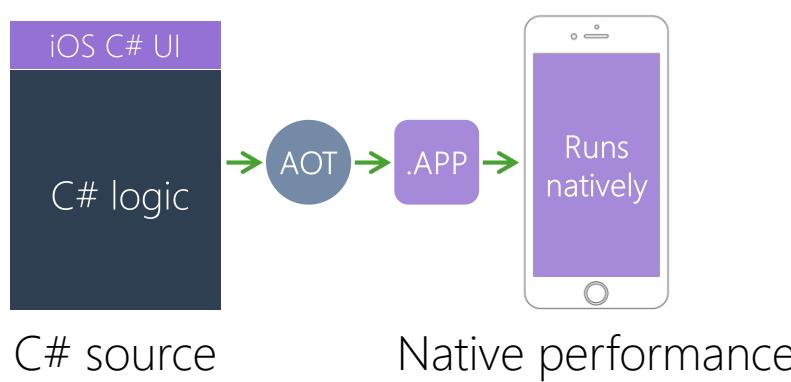


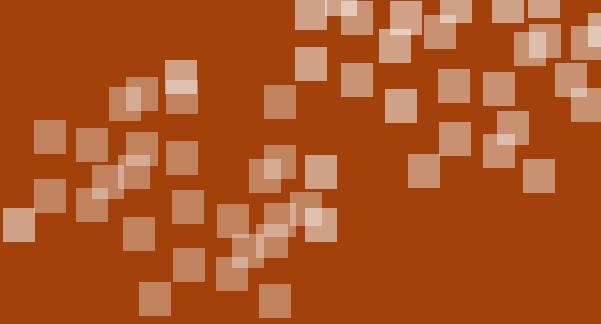
```
public class TextView extends View ...  
{ ...  
    public CharSequence getText() { return null; }  
    public final void setText(CharSequence text) {}  
  
    public void addTextChangedListener(TextWatcher watcher) {}  
    public void removeTextChangedListener(TextWatcher watcher) {}  
}
```



Xamarin performance

- ❖ Xamarin apps are fully native, you get fully native performance with the benefits of shared code





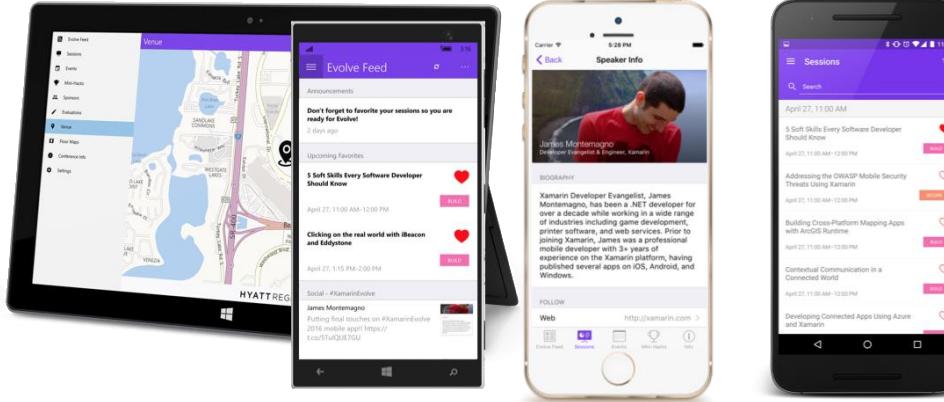
Demonstration

Cross-platform applications with Xamarin



Xamarin.Forms

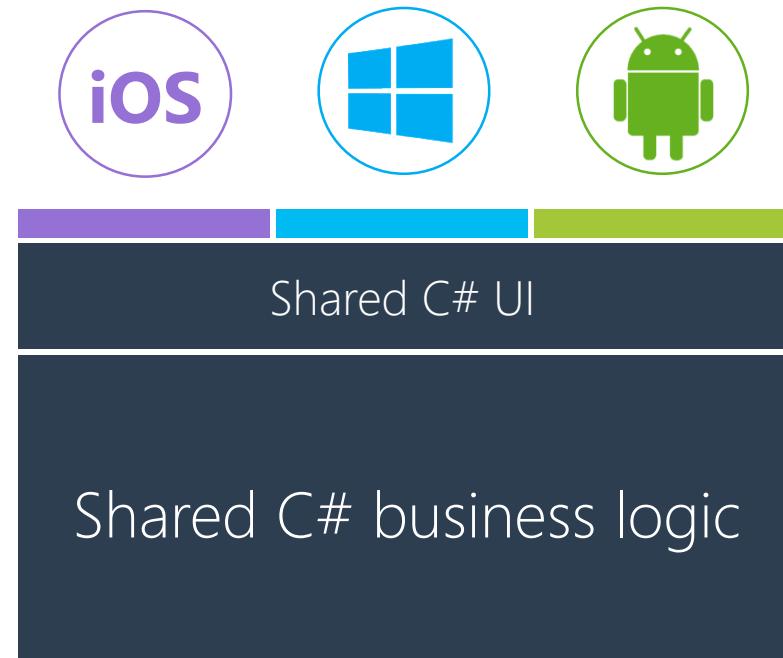
- ❖ Xamarin.Forms enables even more code-sharing through a shared UI definition when deep platform integration is unnecessary



Build native UIs for Android, iOS, and Windows
from a single, shared C# codebase

Included in Xamarin.Forms

- ✓ UI building blocks like pages, layouts, and controls
- ✓ XAML-defined UI
- ✓ Data binding
- ✓ Navigation
- ✓ Animation API
- ✓ Dependency Service
- ✓ Messaging Center



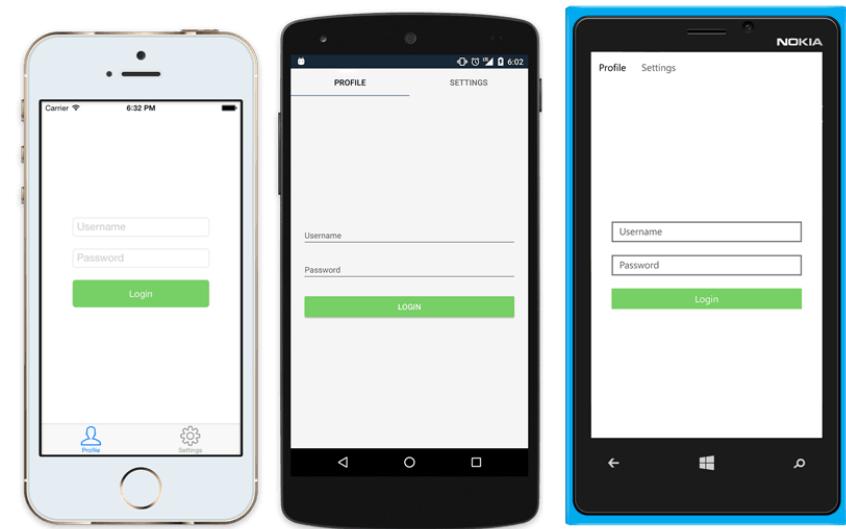
Native controls are used at runtime

```
<?xml version="1.0" encoding="UTF-8"?>
<TabbedPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MyApp.MainPage">

    <ContentPage Title="Profile" Icon="Profile.png">
        <StackLayout Spacing="20" Padding="20"
                     VerticalOptions="Center">
            <Entry Placeholder="Username" Text="{Binding Username}" />
            <Entry Placeholder="Password" Text="{Binding Password}"
                  IsPassword="true"/>
            <Button Text="Login" TextColor="White"
                   BackgroundColor="#77D065"
                   Command="{Binding LoginCommand}"/>
        </StackLayout>
    </ContentPage>

    <ContentPage Title="Settings" Icon="Settings.png">
    </ContentPage>

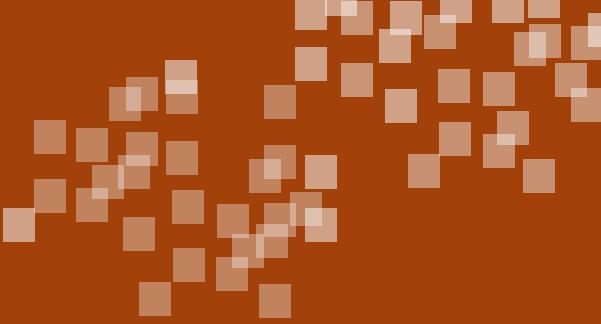
</TabbedPage>
```



UITextField

EditText

TextBox



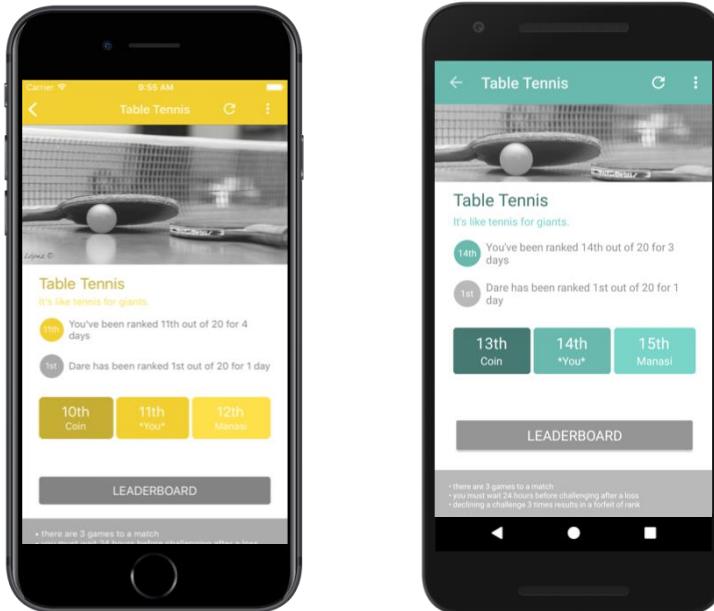
Demonstration

Cross-platform applications with Xamarin



Beautiful apps in less time

- ❖ Create great looking apps that have feature parity with native performance and enjoy the benefit of shared UI and business logic with Xamarin.Forms



Open Source – open.xamarin.com



Build the future of apps
with Xamarin.

Xamarin SDK is now fully available under the MIT license.

One code base, unlimited possibilities

- ❖ With one code base and native performance you can meet your customers where they need to be





Install Xamarin



Before we start...

- ❖ Download and launch the Xamarin Installer *now* on your development machine to begin the automated setup so you are ready when we hit the installation section

Xamarin Installer

www.xamarin.com/download

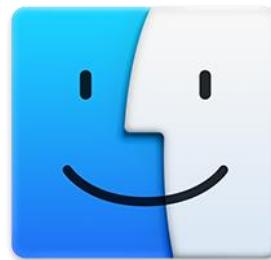
Visual Studio Installer

<https://www.visualstudio.com>

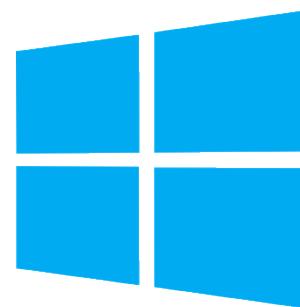
Note: for iOS development with Visual Studio on Windows, you also need to set up a Mac with the Xamarin tools.

Supported operating systems

- ❖ Xamarin tools can be installed on macOS and Windows



iOS, Android, and macOS
Visual Studio for Mac

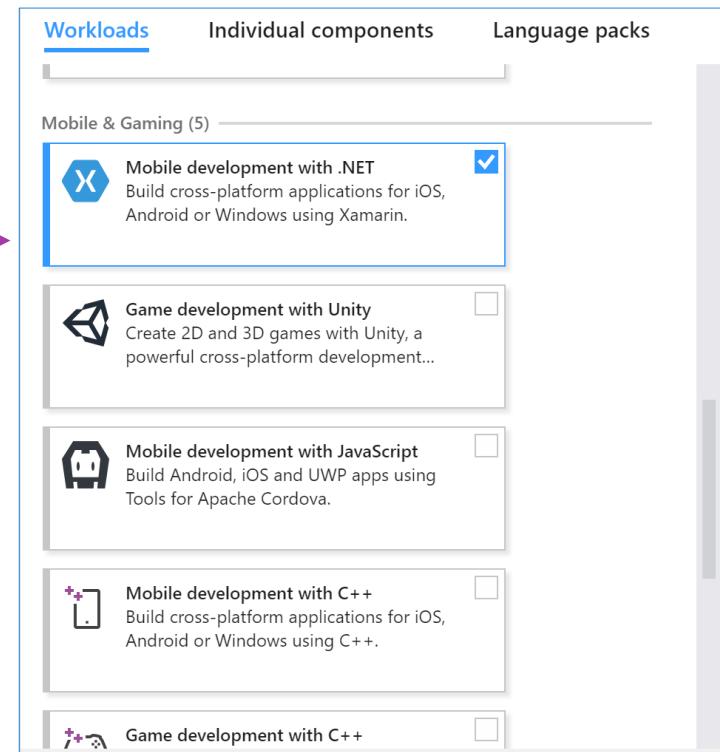


iOS, Android, and Windows
Visual Studio IDE

Install on Windows

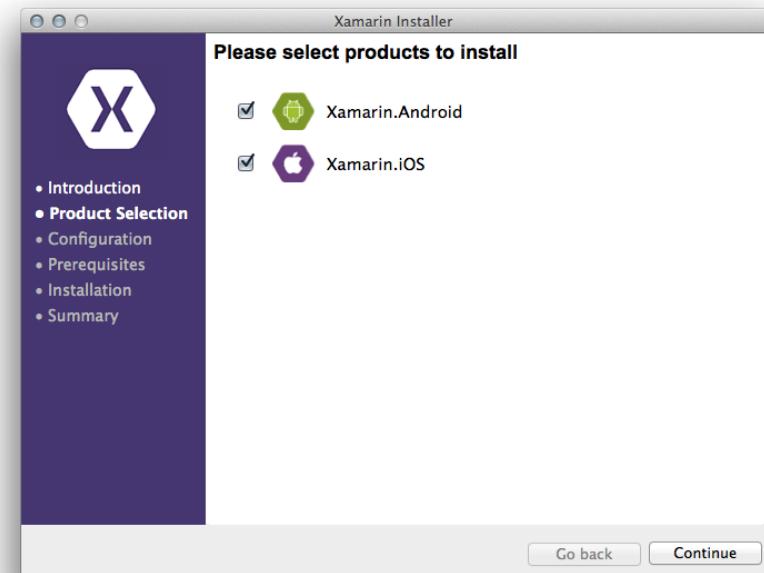
- ❖ On Windows, Xamarin installs directly from the Visual Studio Installer

Make sure
Mobile
development with
.NET is checked



Install on a Mac

- ❖ When using a Mac, the first thing you should do is install Xcode and use the Xamarin Unified Installer to download and install required components



Important: To install the latest version of Xcode you may need to update to the latest version of macOS

Visual Studio Enterprise benefits

- ❖ There are additional benefits included with a Visual Studio Enterprise license



Bytecode hiding
for Android APKs



Live app
inspector



Profiler



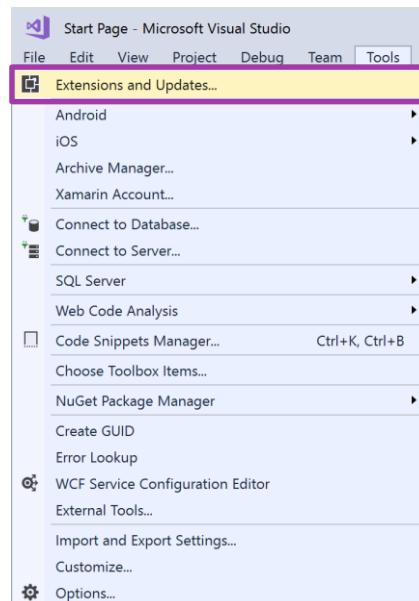
Test Recorder



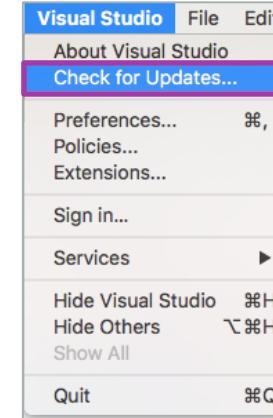
Important: Make sure to use the Enterprise installer from the Visual Studio download page to ensure you get the correct edition of the development environment installed!

Keep Xamarin up to date

- ❖ Xamarin releases updates to add new APIs, match vendor releases, and fix issues



Visual Studio on Windows

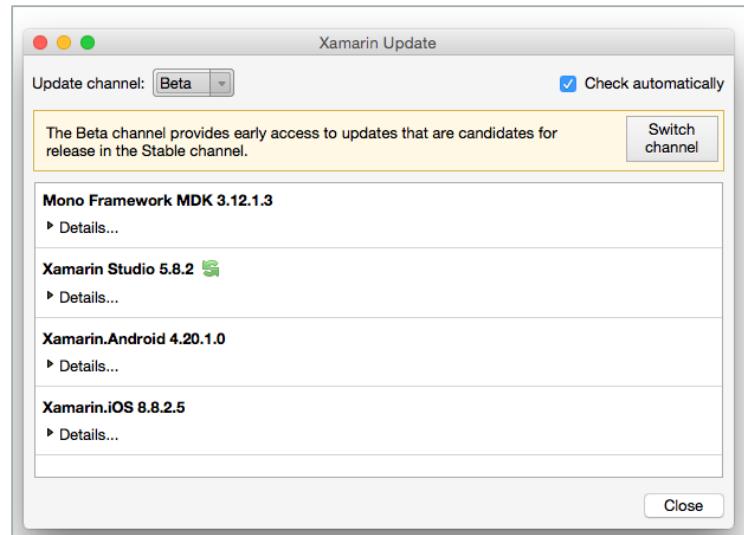


Visual Studio for Mac

Xamarin macOS release channels

- ❖ Xamarin updates are deployed in stages for macOS, and exposed through release channels (Alpha > Beta > Stable)

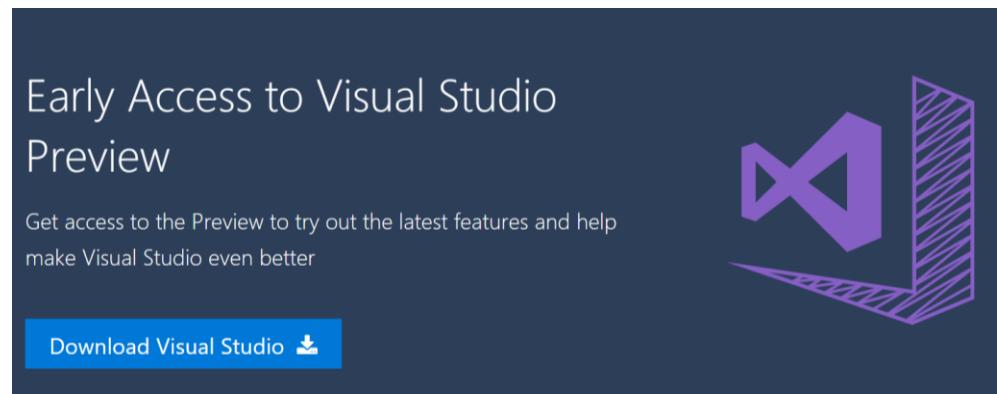
Alpha: most current, least tested
Beta: what's next
Stable: released code, most tested



Important: We recommend the stable channel for most classes

Xamarin pre-release on Windows

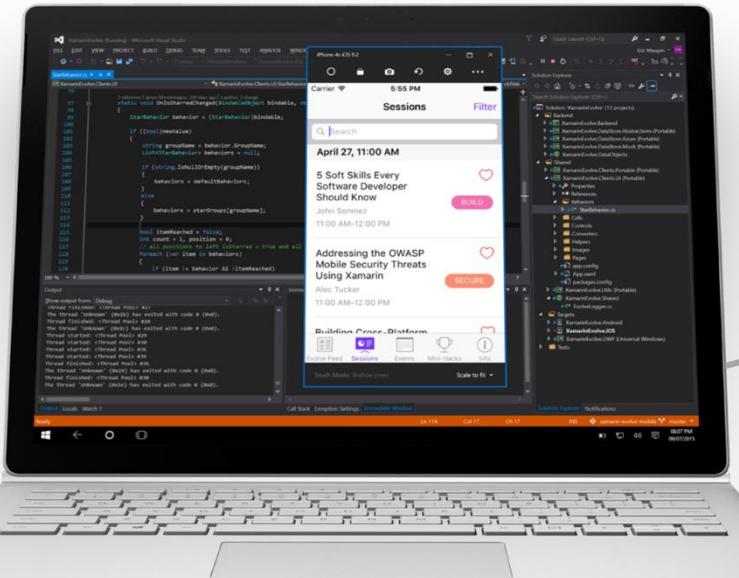
- ❖ Xamarin early access releases are deployed with Visual Studio Preview, available for download



Important: We recommend the standard Visual Studio release for most classes

iOS Development Requires a Mac

You work in Visual Studio on Windows



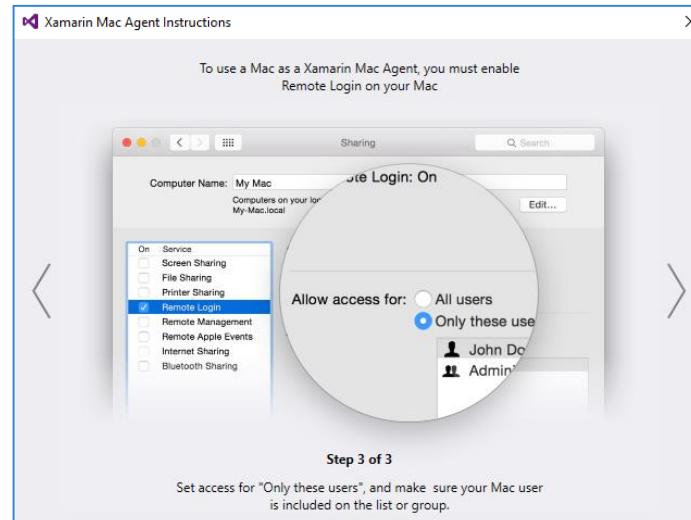
Delegates parts of the build to a Mac using a server process called the *Xamarin Mac Agent*



Run the Xamarin installer on your Mac to setup the Mac Agent

Connecting to Mac Agent

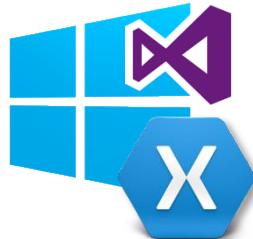
- ❖ Creating or opening an iOS project in VS will login to the associated Mac host, if no host is available, it will launch the connection wizard



 You can also use Tools > iOS > Xamarin Mac Agent to launch connection wizard manually to connect to a different host

Connecting to the Mac

- ❖ Building an iOS application will automatically connect to the build agent



Visual Studio with
Xamarin Tools

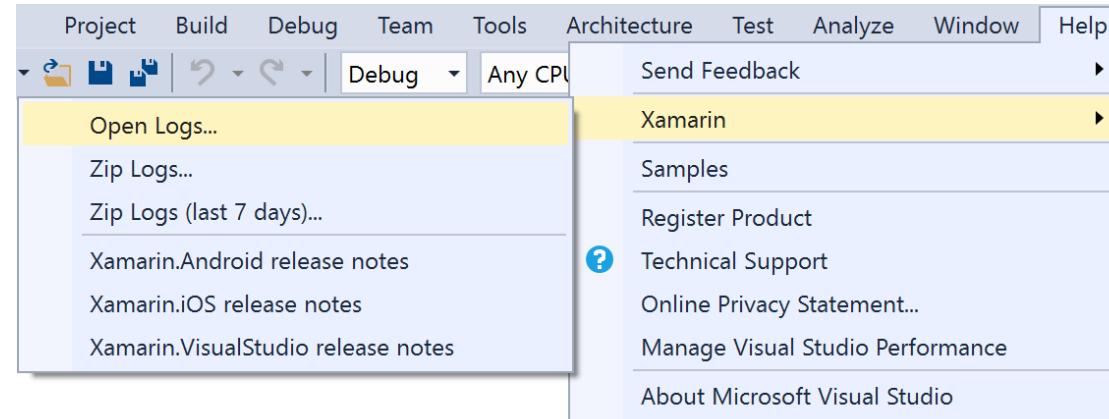


Mac Host with
Xamarin Tools
and Xcode

```
Starting connection to Mac 192.168.0.193...
Starting Broker in port 54837...
Connection successfully established with the Mac 192.168.0.193:54837
Starting agents on Mac 192.168.0.193 (192.168.0.193)
Starting Agent IDB...
Starting Agent Build...
Starting Agent Designer...
Agent Build is running
Agent IDB is running
Agent Designer is running
Connected to the Mac 192.168.0.193 (192.168.0.193) with Full support.
```

Troubleshooting Xamarin Mac Agent

- ❖ The Xamarin Mac Agent will generally diagnose and help correct connection issues; use **Help > Xamarin** for more detailed log information if necessary



Running your applications

- ❖ You need to run applications to test them – can run on devices, or use emulators and simulators which simulate a real device in software

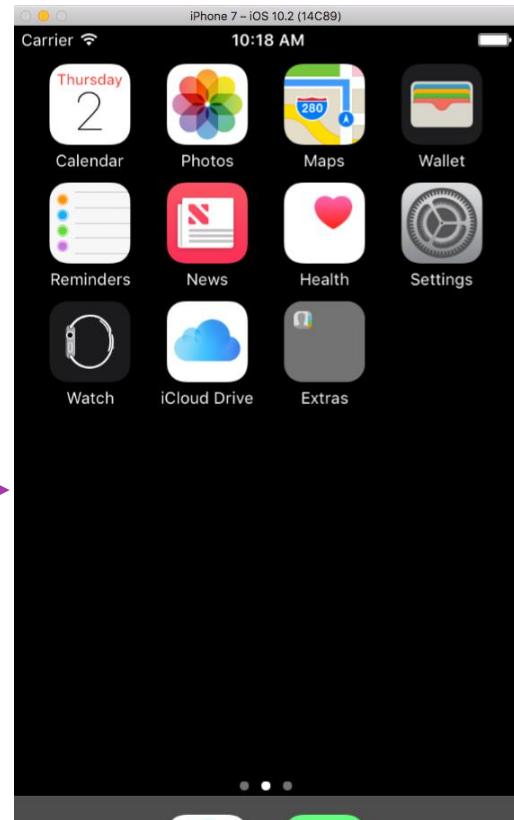


iOS, Android, and Windows all have emulators or simulators

Running iOS apps

- ❖ Apple supplies an iOS simulator with Xcode which can be launched on the Mac host

The simulator →
supports different
devices, resolutions

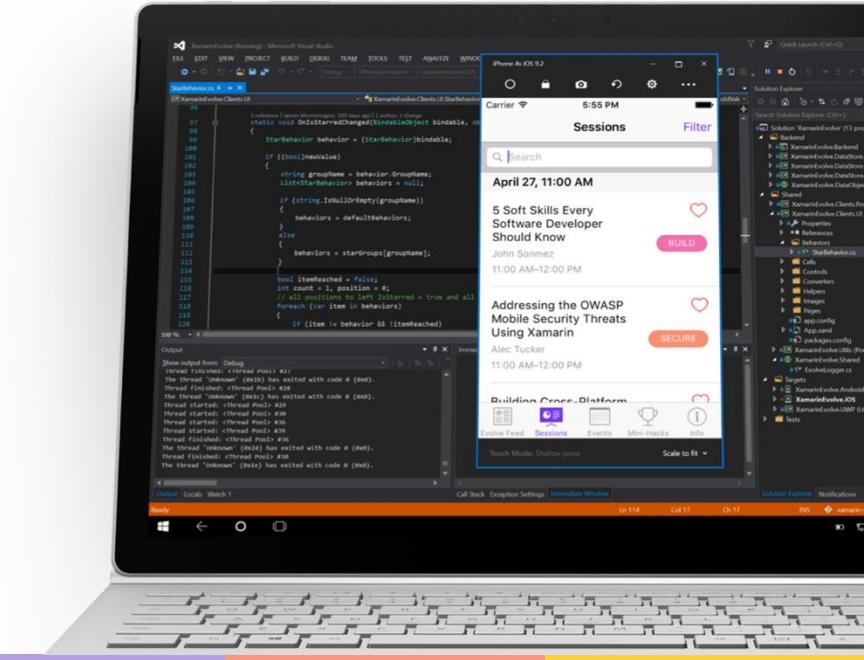


Only the latest iOS release is installed by default, but you can use Xcode to install older iOS versions

Remoted iOS Simulator (for Windows)

- ❖ The Remoted iOS Simulator for Windows makes testing and debugging iOS apps is entirely possible within Visual Studio Enterprise on Windows

- Supports rotation, screenshots, and location changes
- Multi-touch and pressure-sensitive interaction
- Performant



Running Android apps

- ❖ Google provides the standard Android emulator and includes it with the Android SDK and often include Google apps support automatically

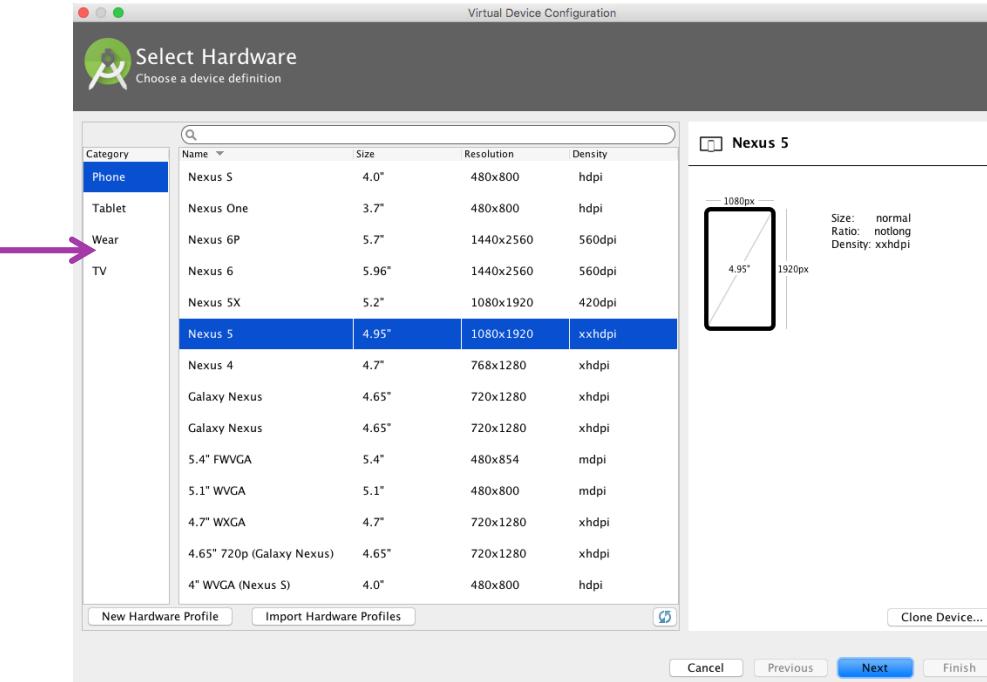


Improved emulator engine and configuration support is available if you install the Android Studio IDE from Google

Installing the Google emulators

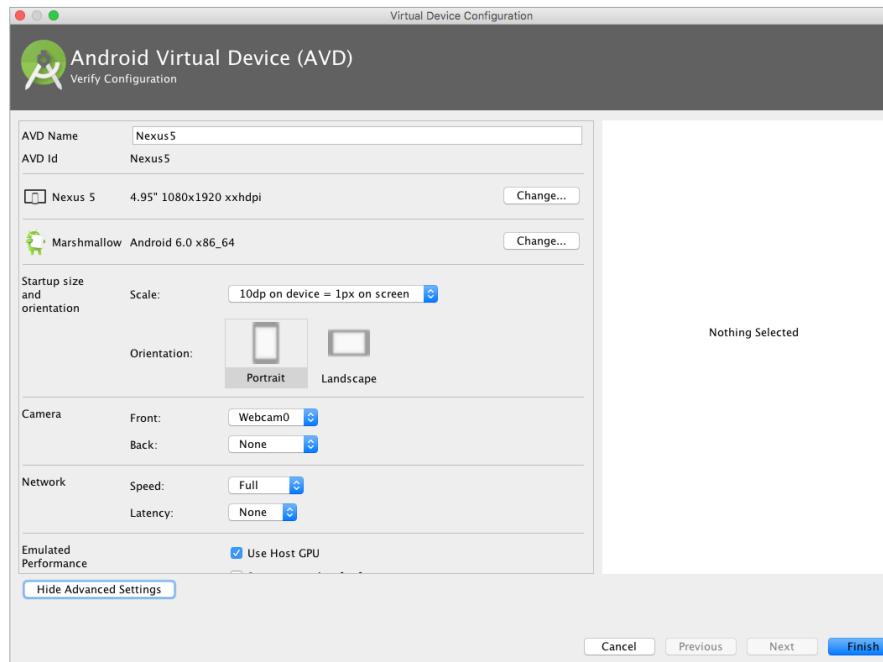
- ❖ Google supports the widest variety of Android devices and versions

Includes TV
and wearable
definitions



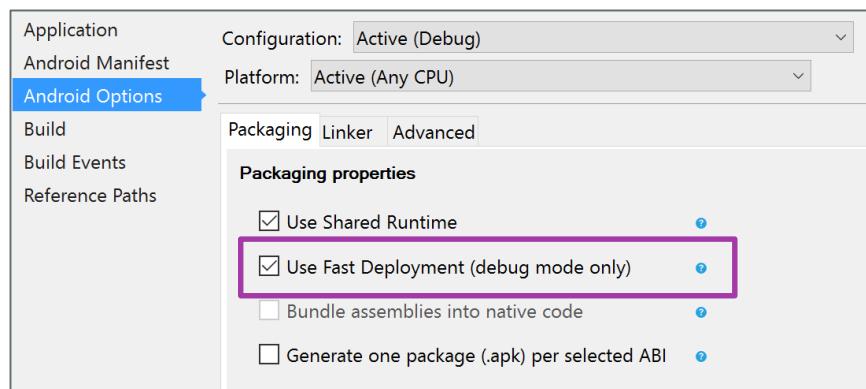
Creating a new Google emulator

- ❖ Android Studio provides access to a much nicer configuration dialog



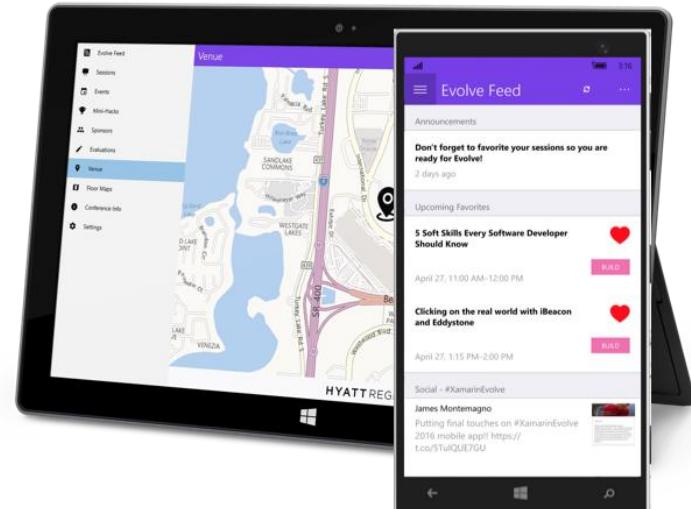
Running apps on Android emulators

- ❖ Some Android emulators do not support the "Fast Deployment" optimization which updates the app in-place on the device, if your application will not install, try turning this feature off in the project settings



Running UWP apps (Windows)

- ❖ Visual Studio can deploy to local or remote Windows 10 devices as well as a optional Windows simulators



Be aware that simulators require Hyper-V and can interfere with virtualization software like VMware and Virtual Box

Using a real device

- ❖ Can use a physical device to run and debug your applications – requires some one-time platform-specific setup

- iOS: <http://bit.ly/1R7YmH8>
- Android: <http://bit.ly/1PjDlFz>
- Windows: <http://bit.ly/2nsGf7i>



Selecting a device or emulator

- ❖ Select the device (or emulator) to run your project using the drop-down on the Standard Toolbar





Setup Validation

Let's make sure your environment is ready to go!

Thank You!

Please complete the class survey in your profile:
university.xamarin.com/profile