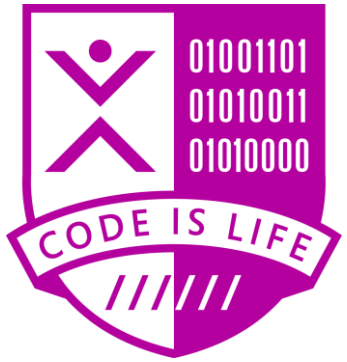


Microsoft Student Partners

Mobile Cross Platform App Development with Xamarin

Daniel Lichtenstern

Justin Lübbers

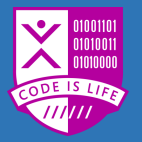
Jakob von der Haar

Microsoft

…or how not to use

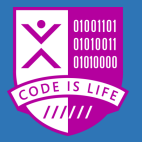
Java, Swift, Objective C, Ionic, Android Studio, XCode …

for developing a single App.



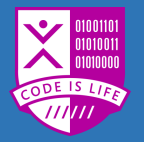
Today, supporting only one platform is not enough.

n platforms = n apps



300h for iOS

iOS + Android = ca. 600h



The Solution:
Cross-Platform Development

300h Developing one App

For iOS + Android


HilariousGifs.com

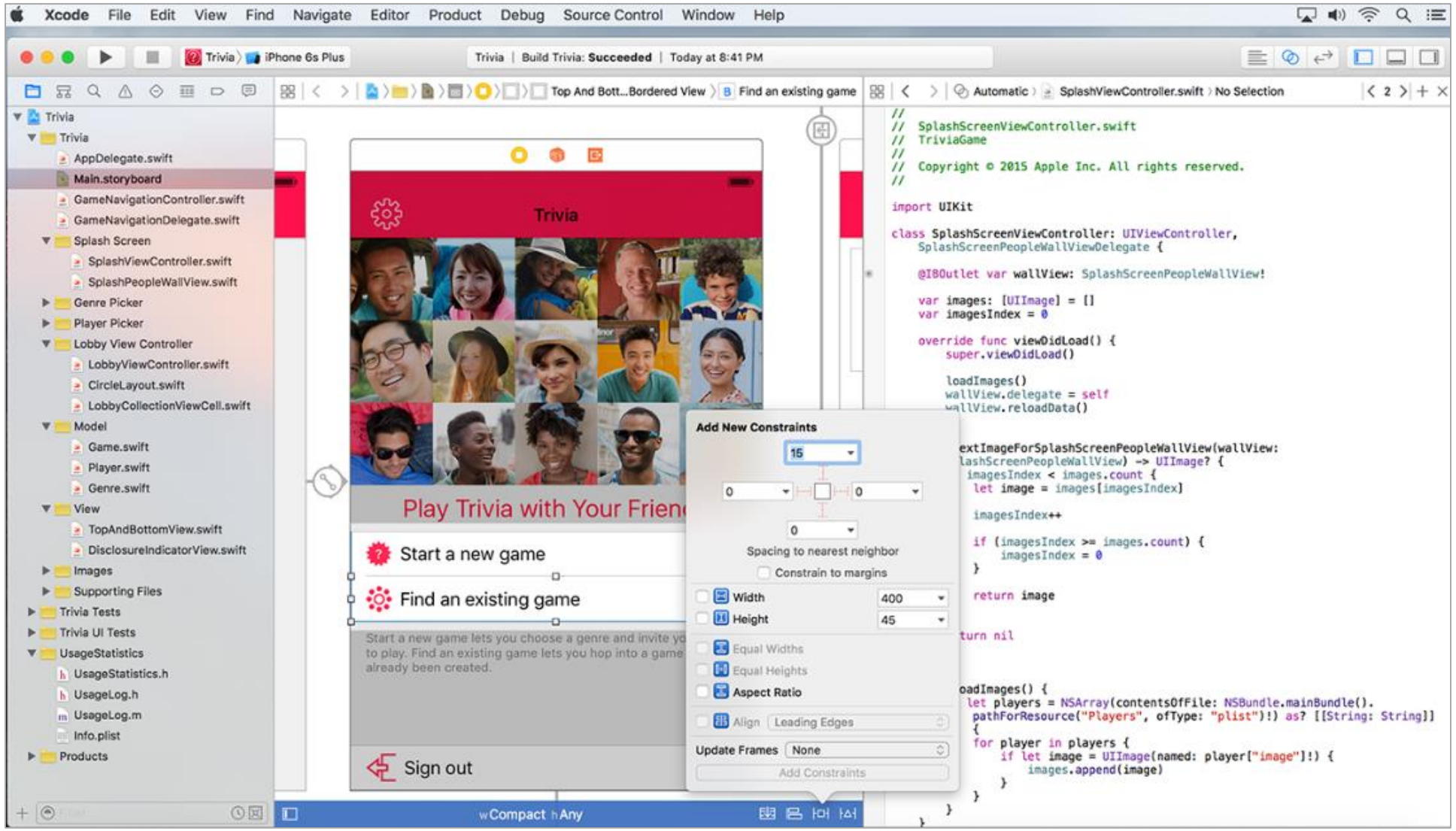

WE GET TO GO HOME!

+ Native
+ Cross Platform
= Xamarin

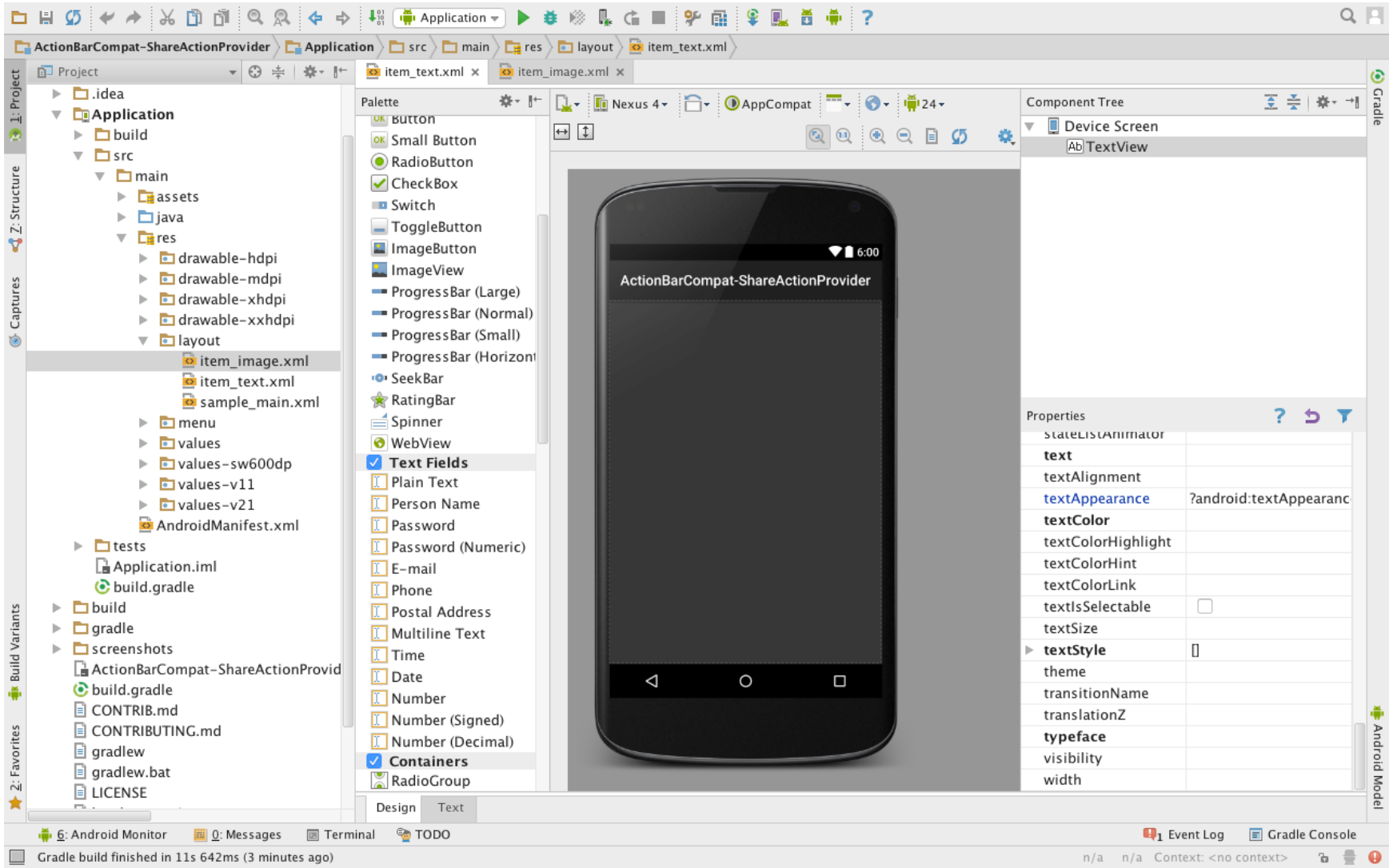
Native Mobile Development

Usage of platform-specific and proprietary tools

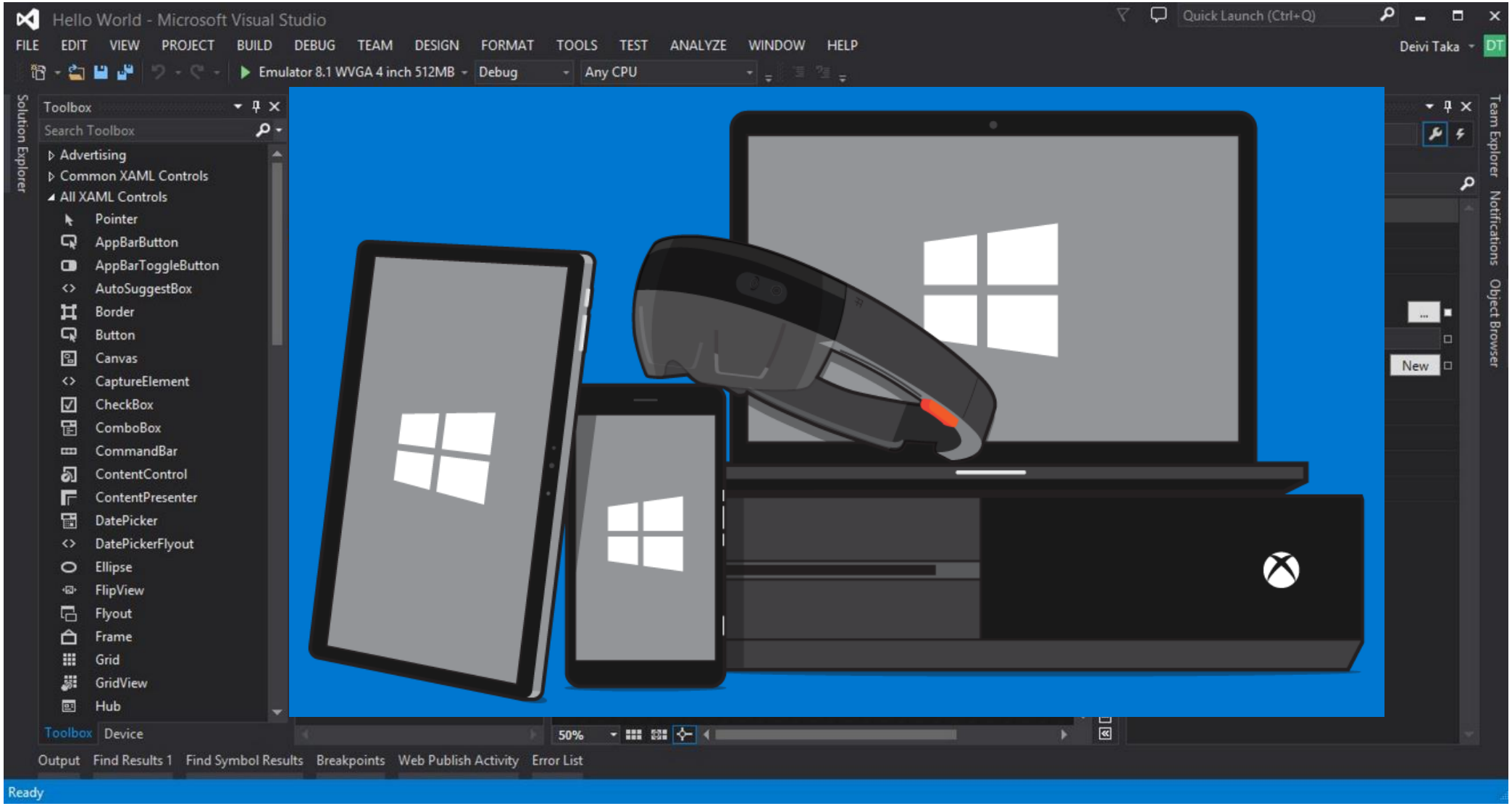
iPhone: Xcode



Android: Android Studio



Windows: Visual Studio



Benefits of Native Development

Supported by Makers of Device

- They want you to make great apps for their device
- Provide support to developer communities

Highest Quality Applications Possible

- All device features accessible
- Optimized for a single platform
- All graphics and interactivity features available

Drawbacks of Native Development

Separate Teams

- Teams have different skills
- Use different programming languages
- Use different tools

Hard to Reuse Code

- Rewrite all code again for each platform
- Code bug fixes and added features separately on all platforms

Native Development Pro/Con

Pros

- Quality
- Performant
- Graphics and Interactivity

Cons

- Development Time
- Costs

Cons

Pros

Hybrid Development

Cross-Platform Mobile Development

Why Cross-Platform Development?

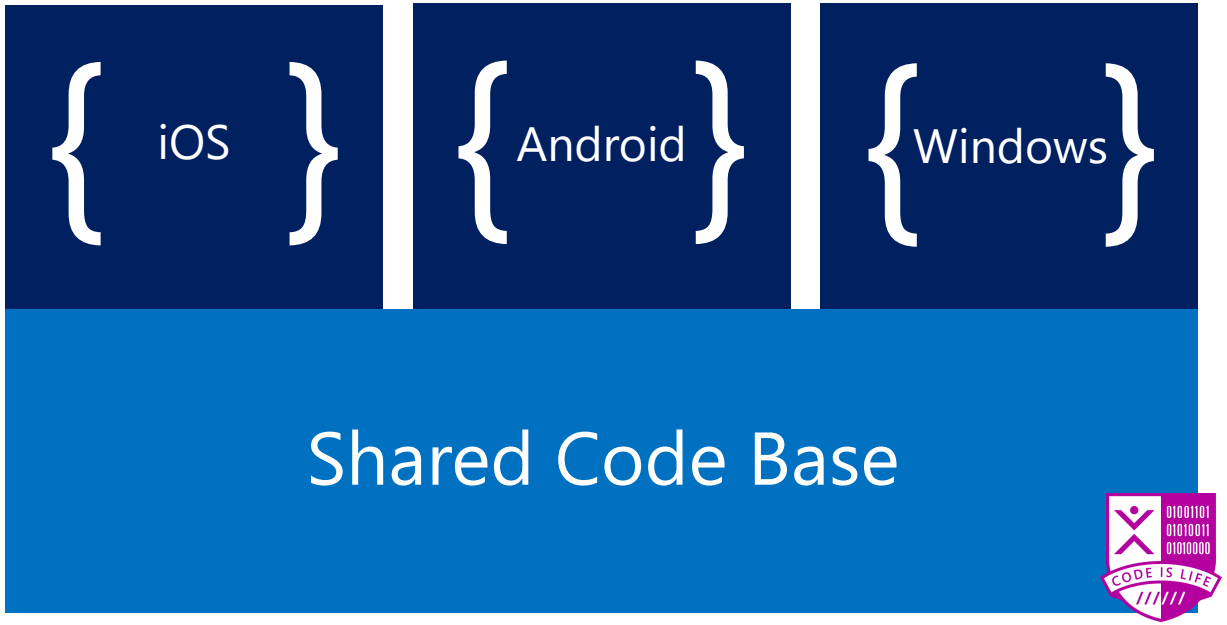
Combine code and development teams

- Shared codebase
- Same development skillset

Shared Codebase

One code base works on multiple platforms

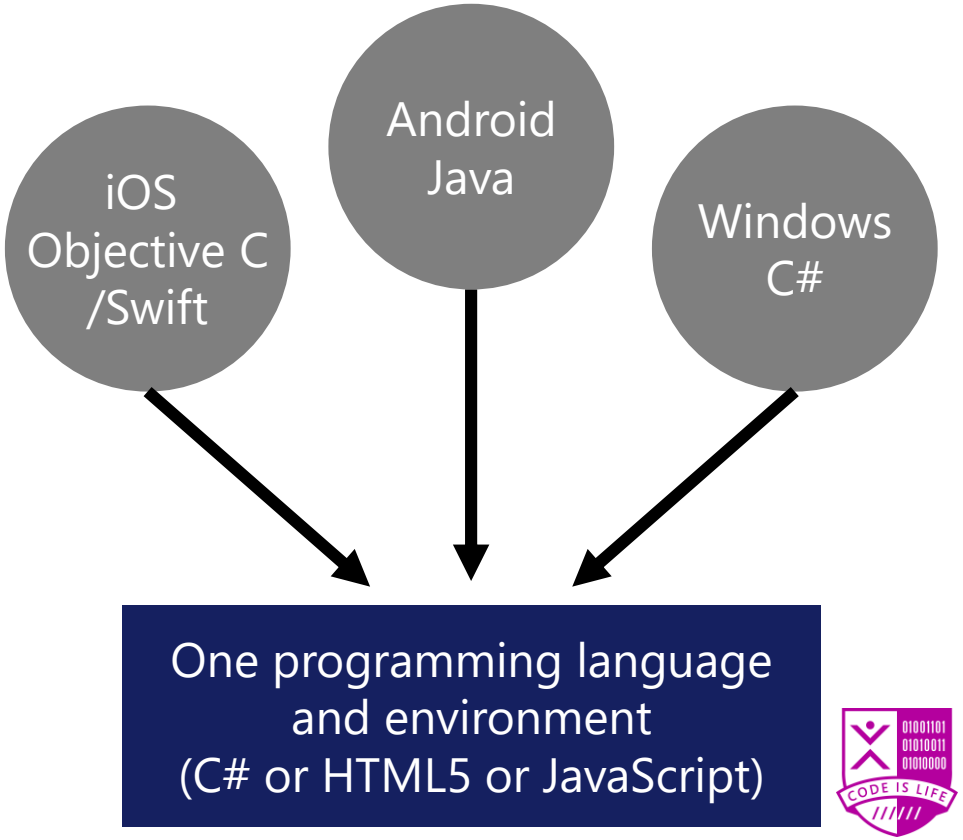
- Shared code base of data access and business logic
- Requires fewer tests
- Increases maintainability



Same Development Skillsets

Developers can use a single technology to build apps

- Diversity of technology is fun for us, the developers
- But it can get expensive for companies
- A cross-platform approach consolidates the skillsets needed



Cross-Platform Solutions

Each framework has its own approach

Development Framework	Defining Characteristics
Xamarin	Code in C# and compile to native
Titanium, NativeScript & React Native	Code in JavaScript and render to native
Cordova	Native shell runs a web app

Titanium, NativeScript, React Native

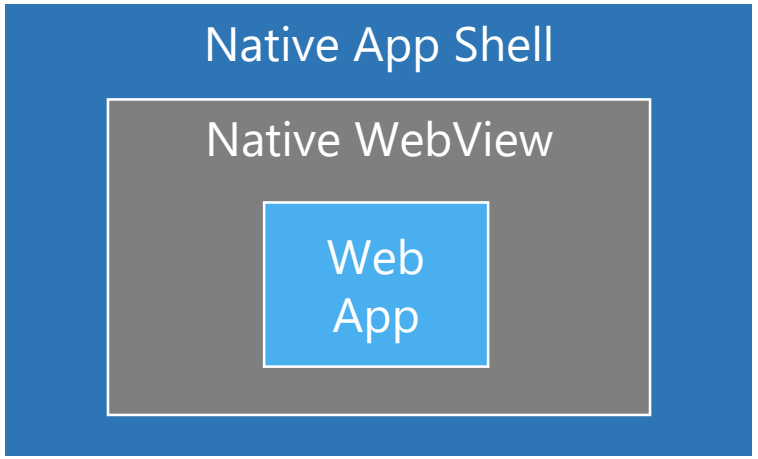
Two sides of development:

- Native app shell
  - XML (or similar) translated to native UIs
- Core logic and UI handled by native JavaScript engine
  - JavaScript used as-is or compiled to native language
  - Controls platform-specific UI using proprietary OS API binding
  - JavaScriptCore (iOS), V8 (Android), Chakra (Windows)

Cordova Hybrid Framework

Web app running within a native shell

- HTML and JavaScript UI
- Native shell allows app store distribution
- JavaScript API communicates with device using OS API
- Cordova is the most commonly used hybrid framework



Pros and Cons

How cross-platform development frameworks stack up

Framework	Advantages	Disadvantages
Xamarin	Performant, enterprise-grade foundation, reliability and comfort of C#, cross-platform UI or 100% access to APIs, truly native apps	Can require knowledge of native platforms, third-party integrations require C# bindings
Titanium, NativeScript, React Native	Well-known language (JavaScript), Shield developer from native API, Native apps	Uneven performance, Lightweight language, Not type-safe
Cordova	Good transition from web development, Rapid prototyping	Uneven performance, Not truly native apps, Look and feel not native

Choosing Xamarin

Many developers and companies are choosing Xamarin

- Because Xamarin provides:
  - A truly native app experience as the *end goal*
  - Fast apps without too much tuning
  - Development using an industrial-strength framework: .NET
  - Programming using a comfortable, heavyweight language: C#

The Xamarin Approach

Xamarin's Approach

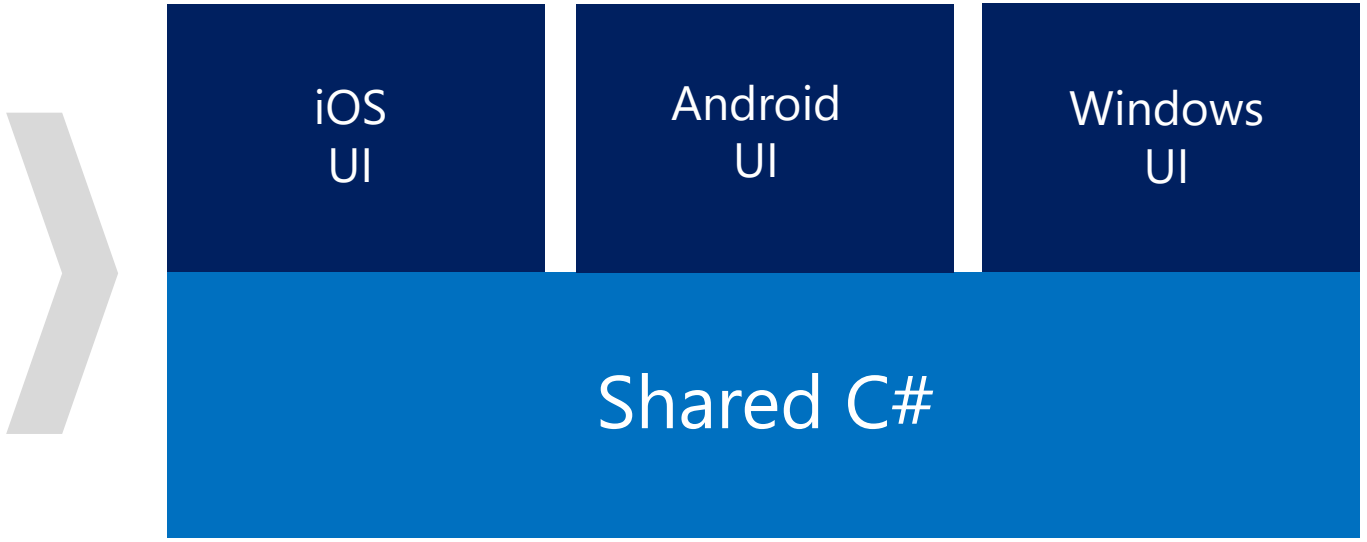
- Fully native apps written entirely in C#
  - Xamarin delivers fully native user interfaces and app functionality
  - Complete access to 100% of the native APIs for iOS, Android and Windows in C#
  - Share app logic and UI code across device platforms

Xamarin Background

- Cross-platform mobile development framework that uses C#
- Compiles to respective native executables

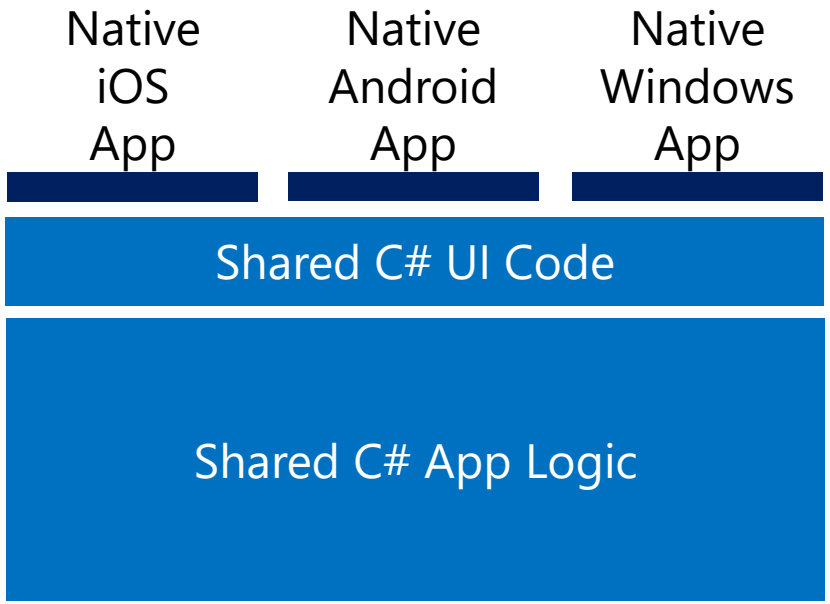
Xamarin Approach

Native with Code Sharing

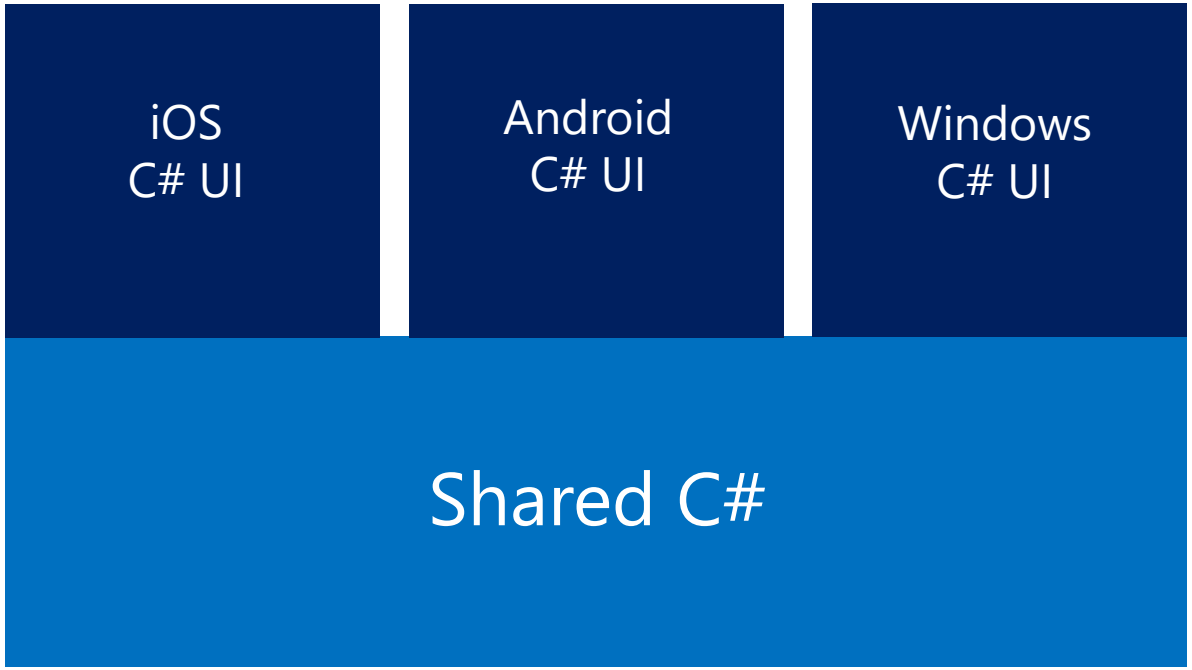


Xamarin.Forms App Architecture

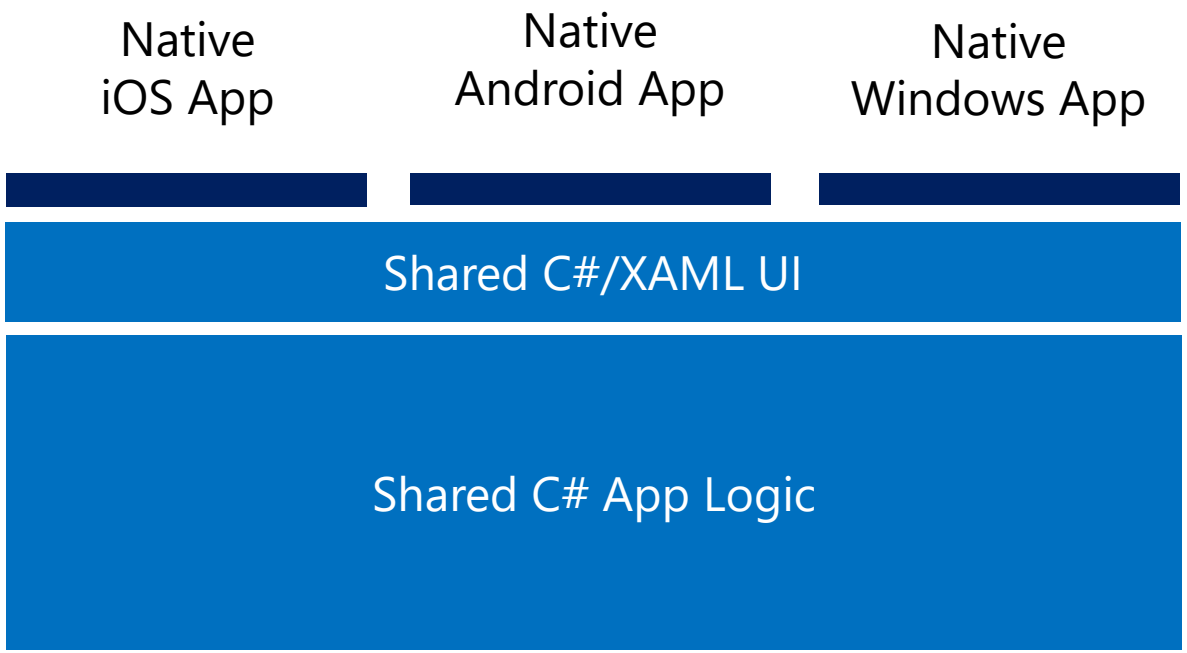
Increase Code Sharing Up to 100% and Deliver a Fully Native App



Xamarin App Architectures



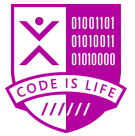
Platform-specific



Xamarin.Forms

Xamarin Approach

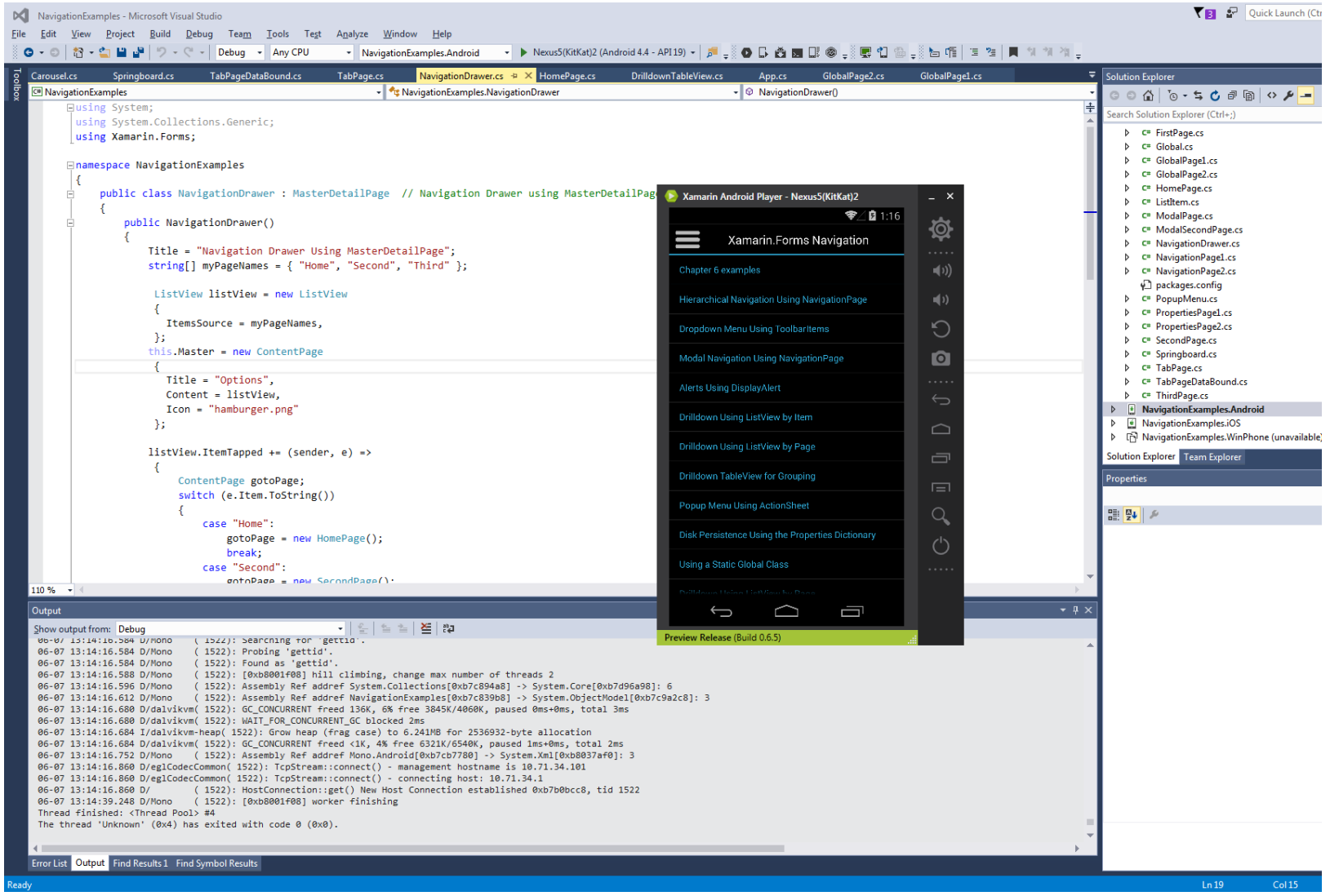
- As much shared code as possible
- As little platform-specific code as needed



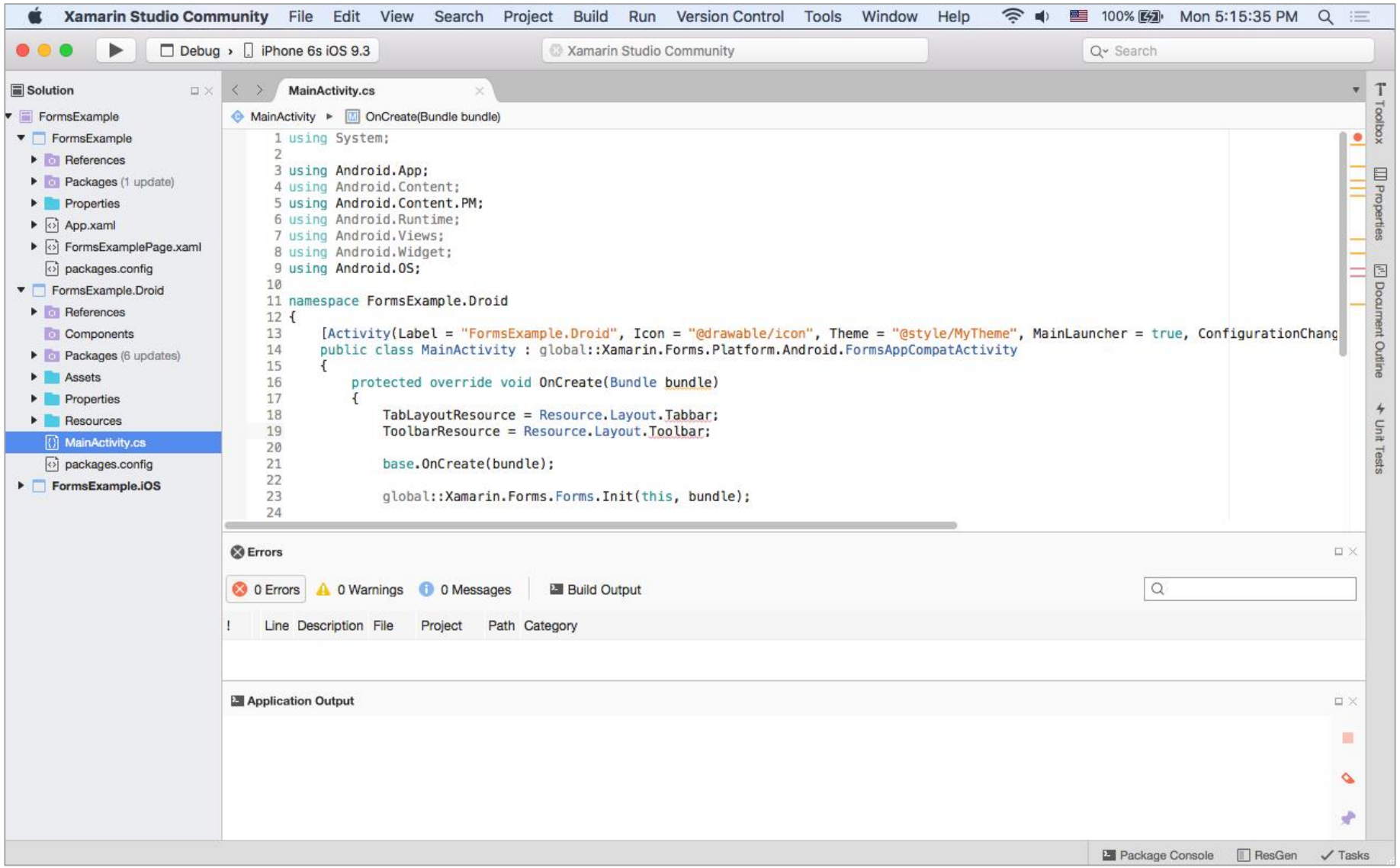
Xamarin's History

- Started as Mono and acquired by several companies since 2003 (Novell, …)
- de Icaza founded Xamarin to continue/expand Mono project (2011)
- Xamarin got acquired by Microsoft (2016)

Visual Studio with Xamarin Plug-in



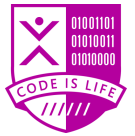
Xamarin Studio



Code Reuse

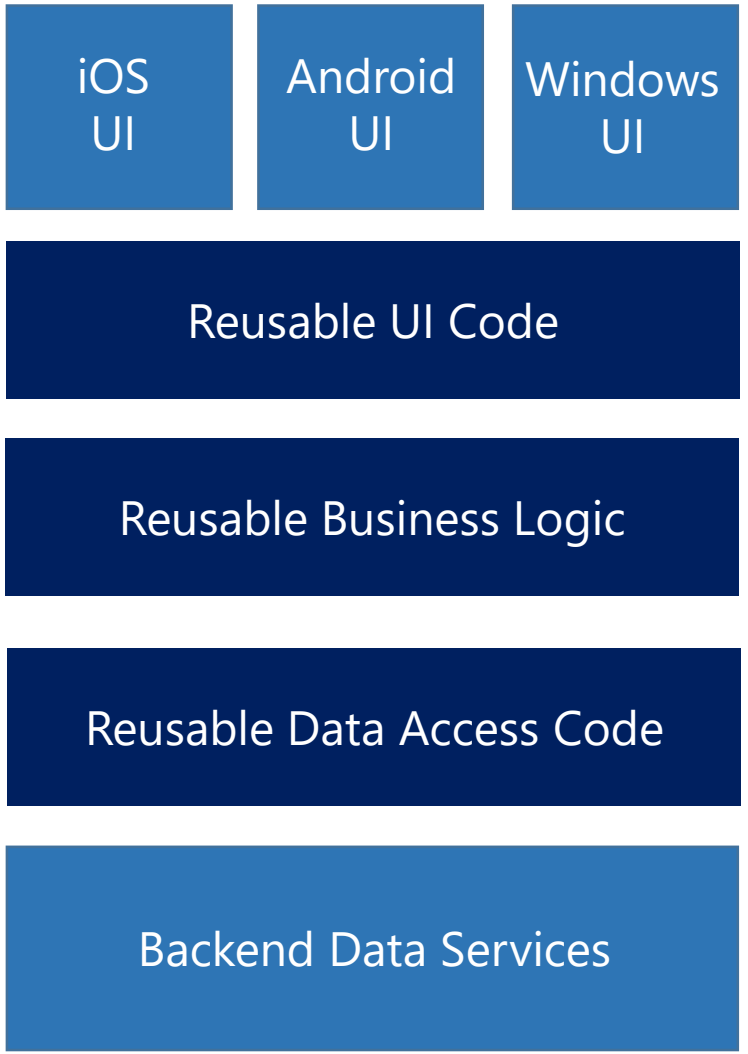
One main benefit of cross-platform mobile development

- Maximize Code Reuse whenever possible:
  - Saves time and money
  - Promotes good programming practices:
  - Encapsulation
  - Separation of concerns (SoC)



Application Architecture

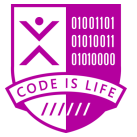
- Isolate and minimize platform-specific code
- Abstract platform-specific logic
  - Separate, platform-specific implementations
  - API/UI layers
- Reuse data access, business logic, and back end web and data services
  - Most or all of this code should be reusable



Sharing Code Options

Use of platform specific functionality

- **Shared Projects** – Use the Shared Asset Project type to organize your source code, and use #if compiler directives as required to manage platform-specific requirements.
- **Portable Class Libraries (obsolete)** – Create a Portable Class Library (PCL) targeting the platforms you wish to support, and use Interfaces to provide platform-specific functionality.
- **.NET Standard Class Libraries** – .NET Standard projects (SCL) work similarly to PCLs, requiring the use of Interfaces to inject platform-specific functionality.




BOSCH


Cognizant


3M


OUTBACK
STEAKHOUSE®


DUPONT
®


Alaska
AIRLINES


CINEMARK®


McKESSON


avanade®


Honeywell


slack


NATIONAL
INSTRUMENTS™

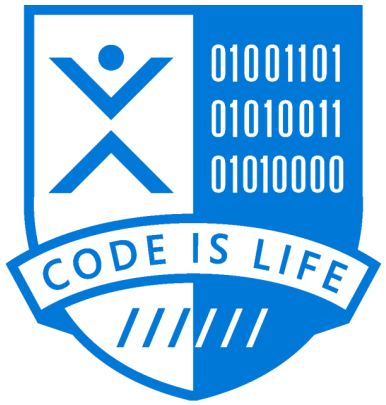

jetBlue


Kellogg's


Kimberly-Clark

Do you want to attend more events like this?

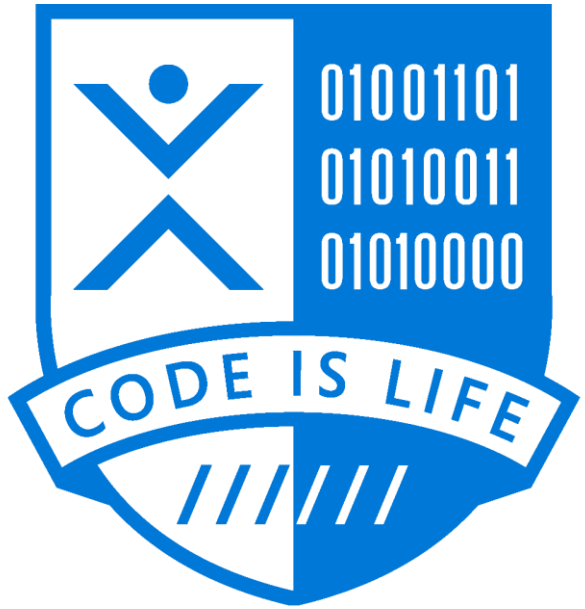
Then join our community:
facebook.com/groups/2015347855345036/



Microsoft **Student Community**

Why should you join the MSC?

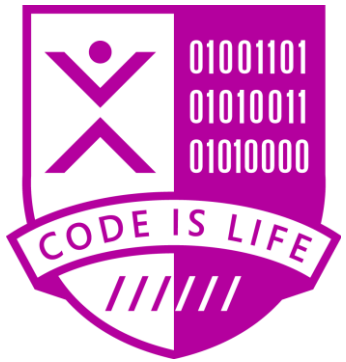
- Share your tech passion with further students
- Get in touch and learn about new and fancy technologies
- Have privileged access to all our events and join us on MSC exclusive events
- Get in contact with the MSPs and Microsoft



Further Ressources

https://docs.microsoft.com/de-de/xamarin/

https://university.xamarin.com/



Microsoft Student Partners

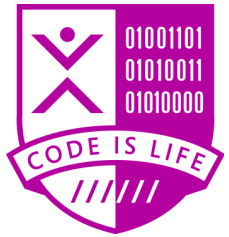
Thank you for your attention!

facebook.com/MicrosoftStudentPartnersTUM/

Daniel Lichtenstern

Justin Lübbers

Jakob von der Haar (Twitter: @Jakob_vdH)



Microsoft Student Partners