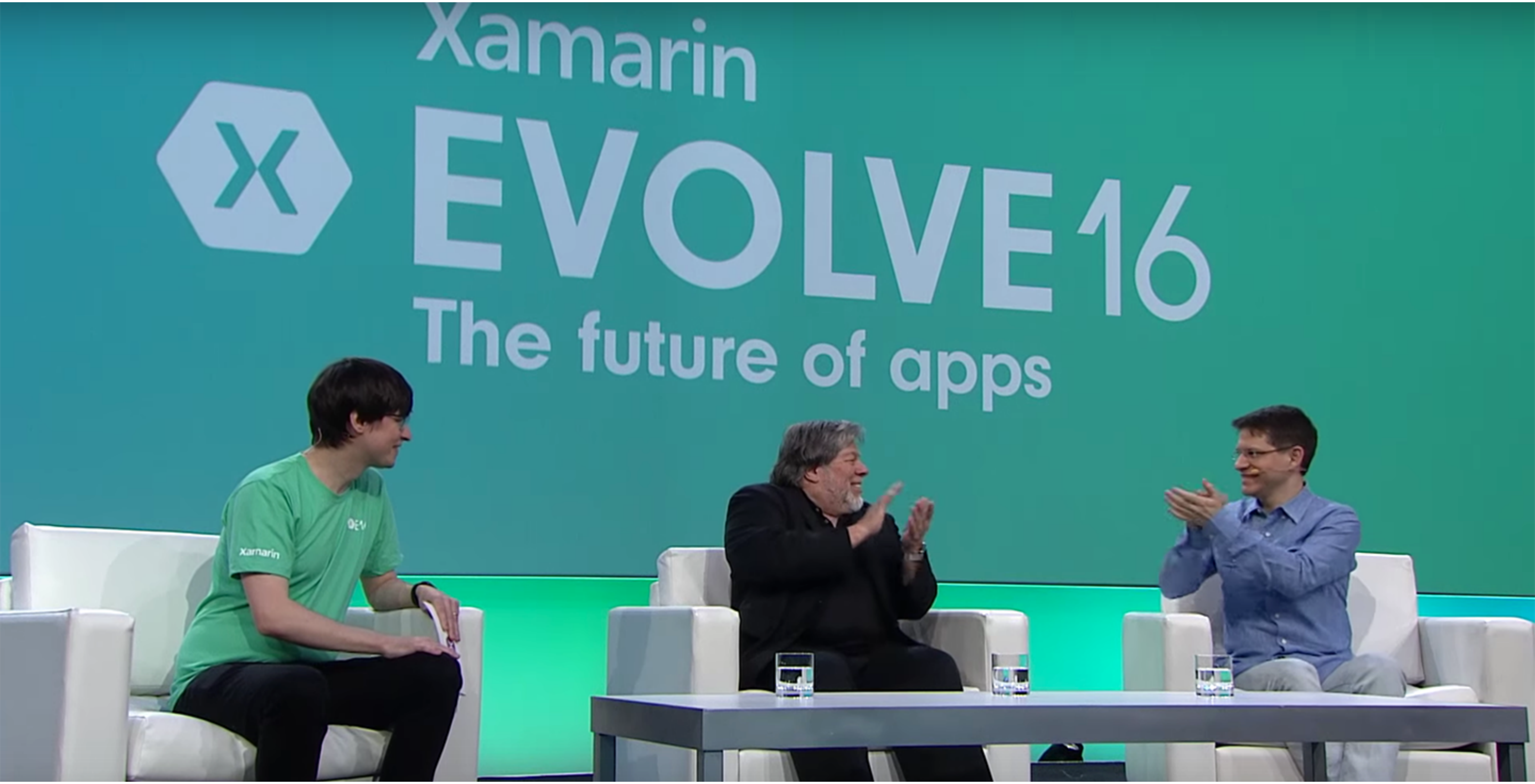
Xamarin Evolve 2016 - A recap



Agenda

- Short Intro
- The long and winding mobile road
- Xamarin
- Evolve
- Buzzwords + Hot Topics
- New Features
- And now?

Short Intro

XITASO GmbH

- Since 2011
- ~ 50 employees (mainly software developers) in Augsburg and Magdeburg
- Technical software, Web applications & Mobile applications
- Focus on development process with Scrum

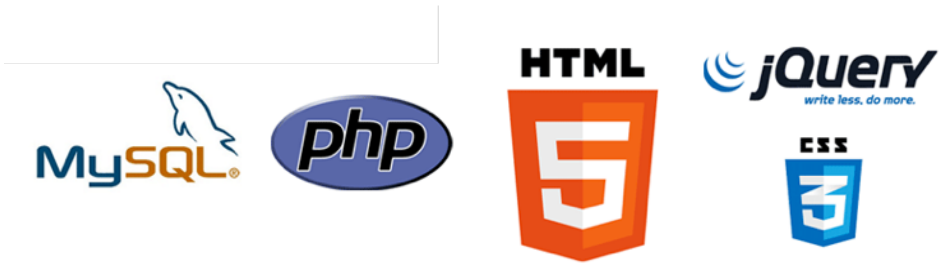
Short Intro

Myself

- Christian Höfle
- Studied Computer Science in Augsburg (Diploma 2010)
- XITASOnian since 2012
- Lead mobile development
- Focus on UI / UX / Usability

The long and winding mobile road

Since 20XX / ??



Since 2011 / 4.X

Since 2012 / 4.X

The long and winding mobile road

PHP =

The long and winding mobile road

Until iOS 6

- Learning Objective-C :(
- 1 Display Resolution (1x & 2x)
- Pixel-perfect Photoshop-Designs
- Skeuomorph Design

The long and winding mobile road

How iOS evolved

- ARC (Automatic Reference Counting)
- Storyboards
- Layout Constraints
- iOS 7 UI
- Size Classes
- Swift

The long and winding mobile road

Knowing iOS, learning Android

- Java >>> Objective-C
- Saving view states before orientation changes???
- Activity contexts
- Damn emulator
- So many devices to test on

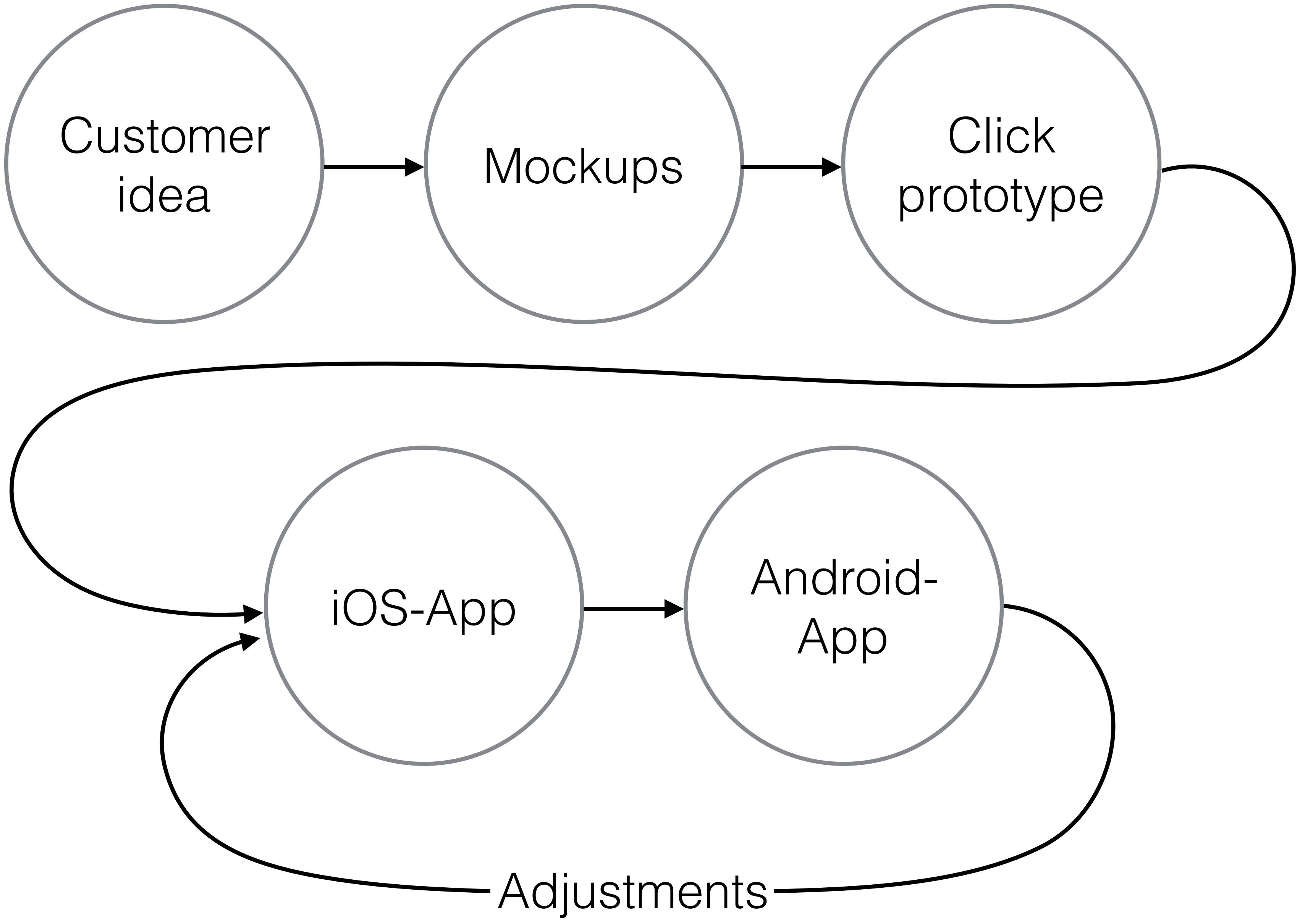
The long and winding mobile road

How Android evolved

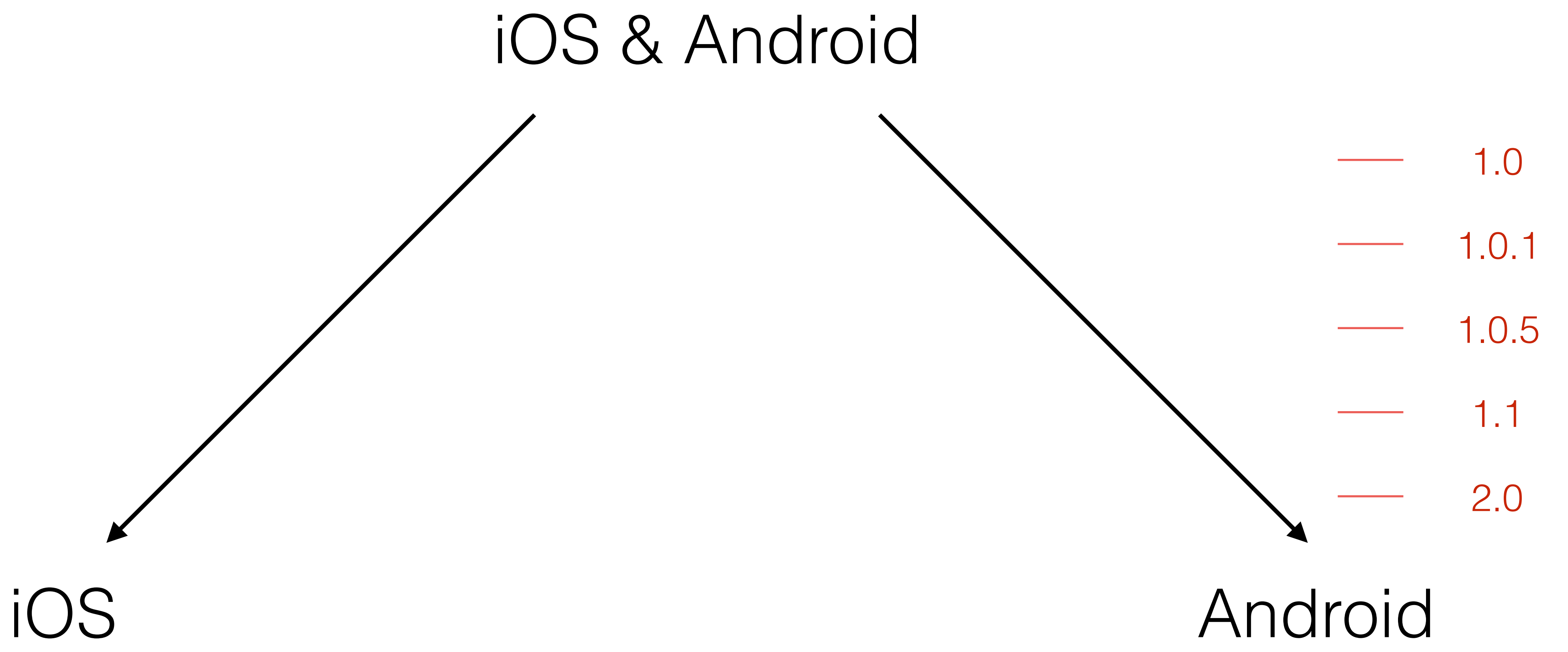
- Fragments
- Actionbar
- Android 4.0
- Ant —> Gradle
- Eclipse —> Android Studio
- Material Design

The long and winding mobile road

Today (Yesterday?)



The long and winding mobile road


iOS & Android

iOS

Android

1.0

1.0.1

1.0.5

1.1

2.0

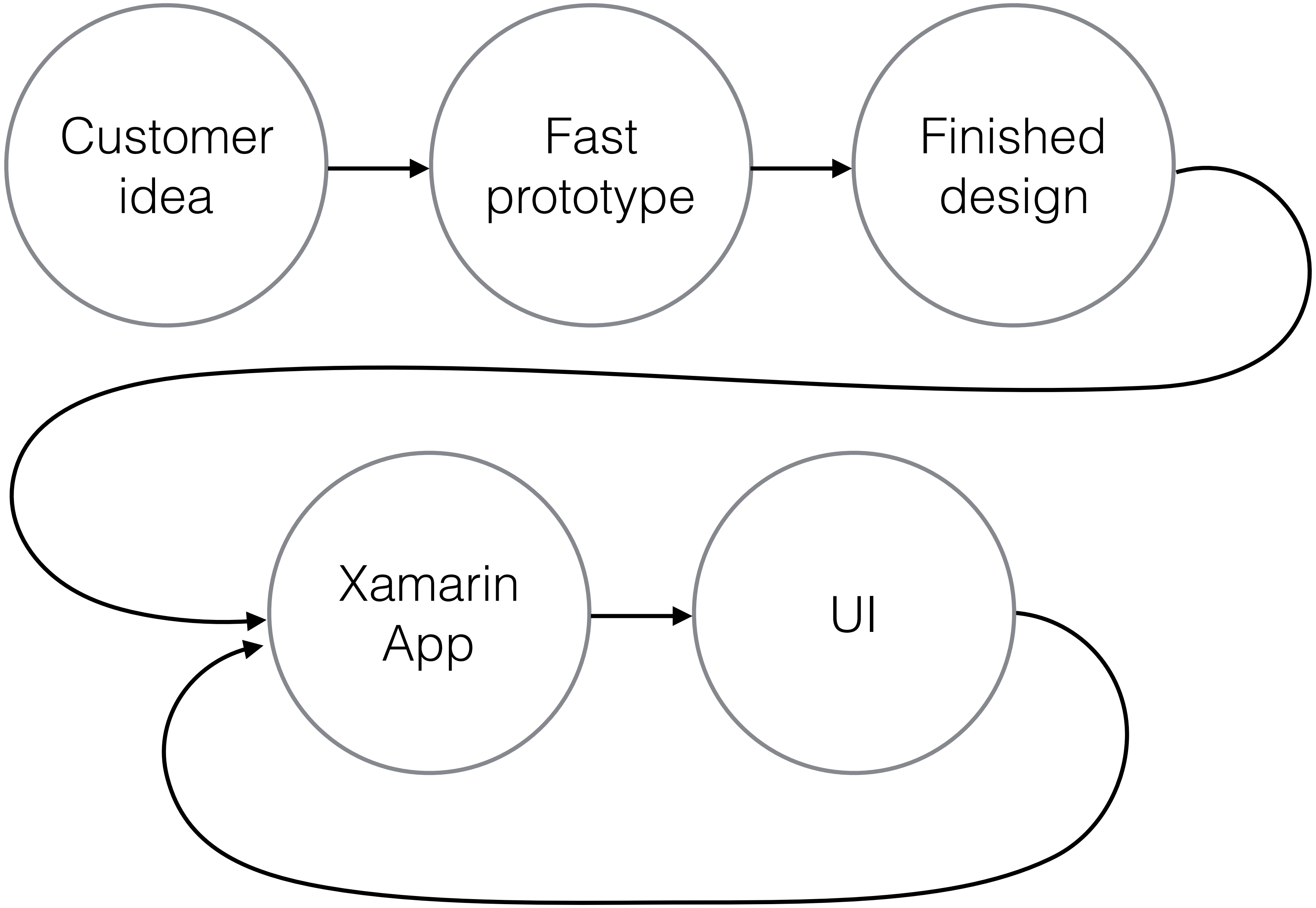
The long and winding mobile road

How to handle 2 platforms?

- 2 Dedicated teams? (expensive)
- Same team, both platforms? (not gonna happen)
- iOS Master, Android Slave? (usually)

The long and winding mobile road

Tomorrow (Today)



Xamarin

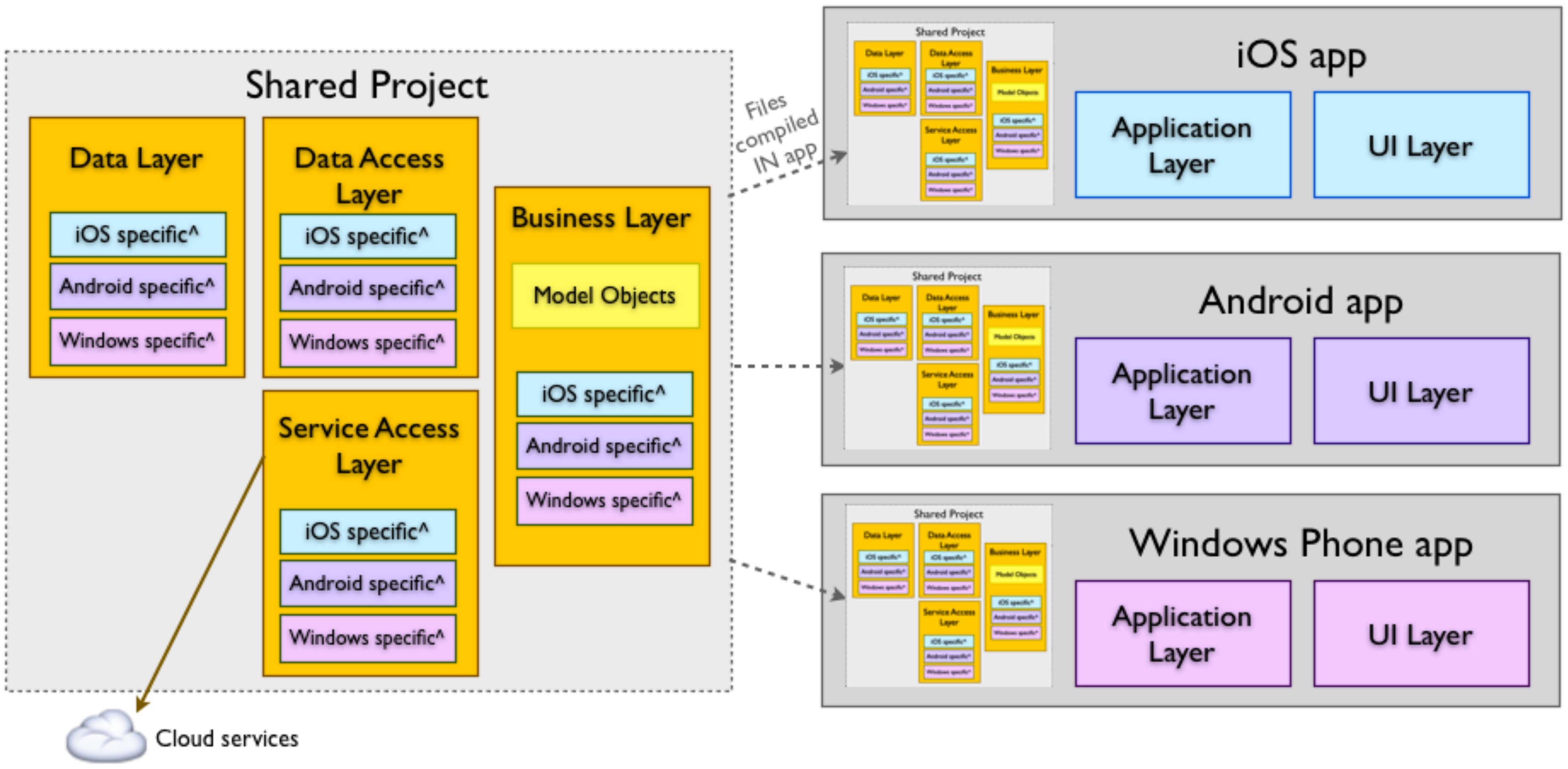
- Founded 2011 by Miguel de Icaza & Nat Friedman
- Based on Mono (2001)
- IDE for Windows and macOS
- .NET / C# / F#

Xamarin

- .NET-Environment —> perfect match
- No more duplicate code, share up to 100%
- C# as language, tools like Resharper
- Full native SKD support for iOS & Android
- Free since MS acquired them in 02 / 2016

Xamarin

Code Sharing: Shared Project



Shared Project

Data Layer
iOS specific^
Android specific^
Windows specific^

Data Access Layer
iOS specific^
Android specific^
Windows specific^

Business Layer
Model Objects
iOS specific^
Android specific^
Windows specific^

Service Access Layer
iOS specific^
Android specific^
Windows specific^

Files compiled IN app

iOS app
Application Layer
UI Layer

Android app
Application Layer
UI Layer

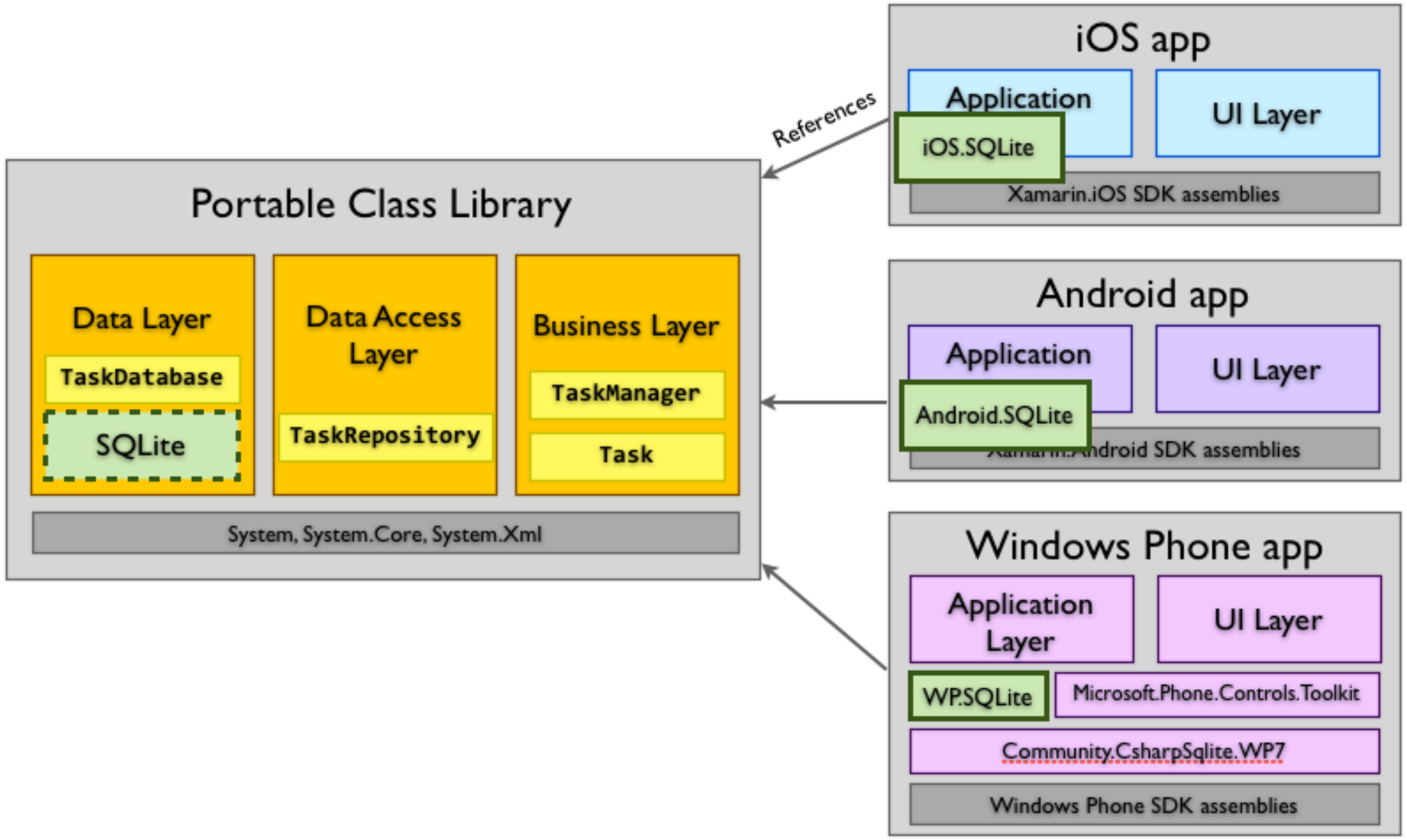
Windows Phone app
Application Layer
UI Layer

Cloud services

^ *#if PLATFORM compiler directives*

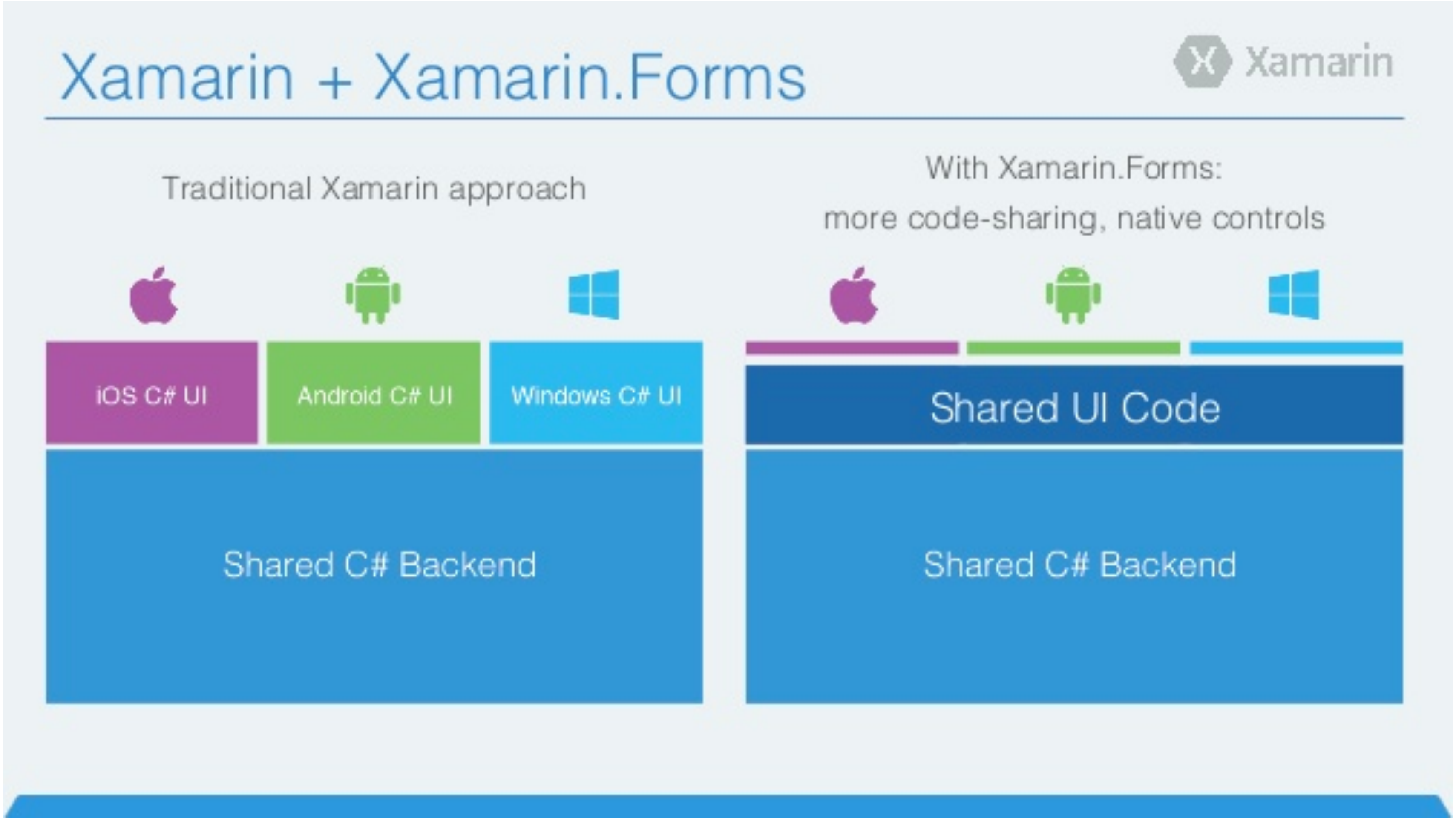
Xamarin

Code Sharing: PCL (Portable Class Library)



Xamarin

Code Sharing: Xamarin.Forms



Xamarin

Code Sharing: Xamarin.Forms

- Use XAML with Code-behind file for UI
- Set of common controls that are mapped to native ones
- Platform specifics can be added via DependencyService and Interfaces

Evolve

- In Orlando, Florida from 24. to 28. April 2016
- Over 1500 developers
- 2 days conference, 2 days training
- Nice events (e. g. Universal Orlando Resort)

Evolve

Highlights

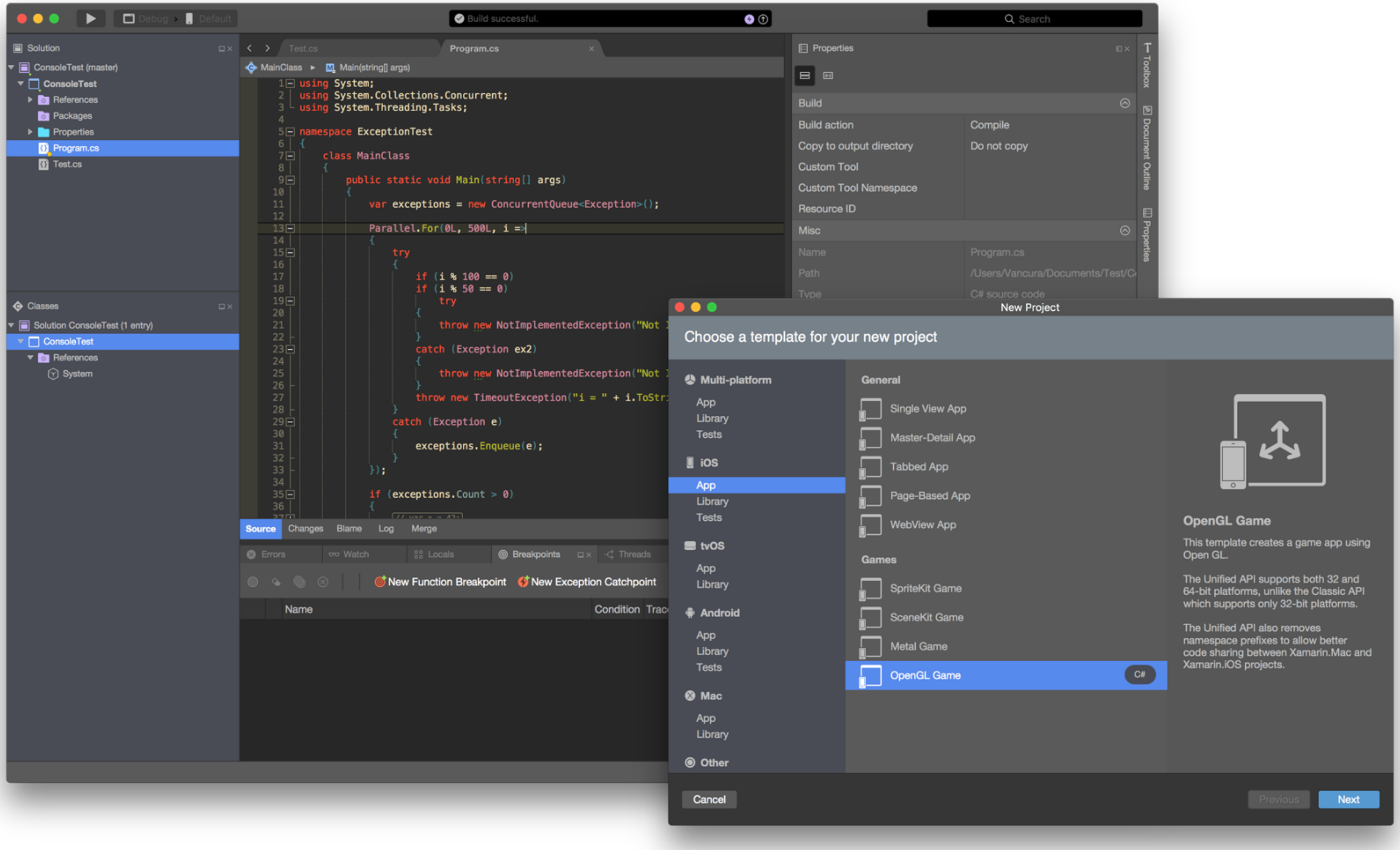
- Steve Wozniak
- Open Source Announcement
- Xamarin.Forms Previewer
- Test Recorder
- Xamarin.Studio Redesign
- Workbooks
- Hacking Events

Buzzwords & Hot Topics

- MVVM
- UI Design with XAML —> Xamarin.Forms
- Azure
- iOT
- Industry 4.0
- Testing / CD / CI
- Security

New Features

Xamarin Studio: New Layout & Themes, Roslyn



Build successful.

```
using System;
using System.Collections.Concurrent;
using System.Threading.Tasks;

namespace ExceptionTest
{
    class MainClass
    {
        public static void Main(string[] args)
        {
            var exceptions = new ConcurrentQueue<Exception>();

            Parallel.For(0L, 500L, i =>
            {
                try
                {
                    if (i % 100 == 0)
                    if (i % 50 == 0)
                        try
                    {
                        throw new NotImplementedException("Not 
                    }
                    catch (Exception ex2)
                    {
                        throw new NotImplementedException("Not 
                    }
                    throw new TimeoutException("i = " + i.ToStr
                }
                catch (Exception e)
                {
                    exceptions.Enqueue(e);
                }
            });

            if (exceptions.Count > 0)
            {
```

Properties	
Build	
Build action	Compile
Copy to output directory	Do not copy
Custom Tool	
Custom Tool Namespace	
Resource ID	
Misc	
Name	Program.cs
Path	/Users/Vancura/Documents/Test/C
Type	C# source code

New Project

Choose a template for your new project

OpenGL Game

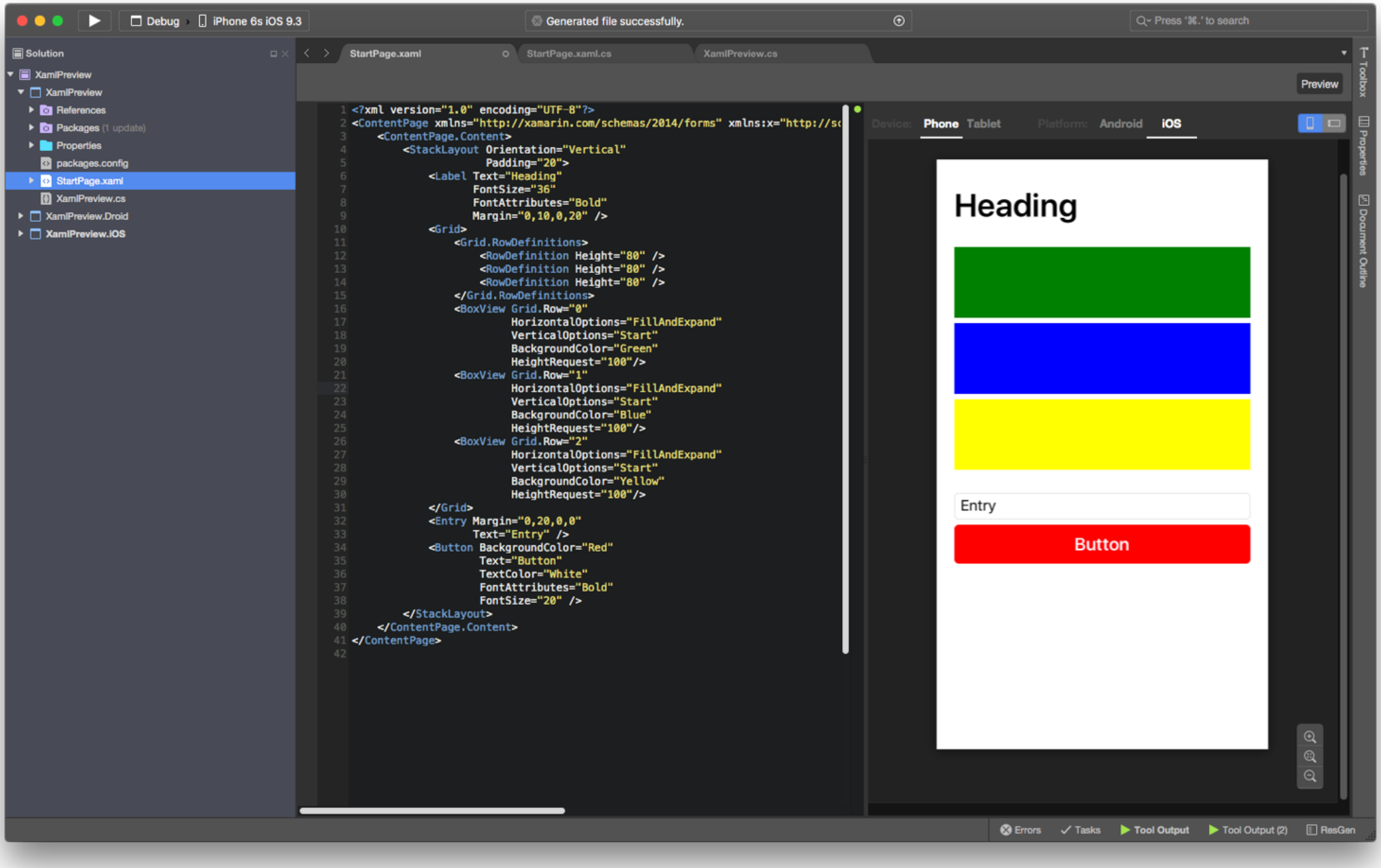
This template creates a game app using Open GL.

The Unified API supports both 32 and 64-bit platforms, unlike the Classic API which supports only 32-bit platforms.

The Unified API also removes namespace prefixes to allow better code sharing between Xamarin.Mac and Xamarin.iOS projects.

New Features

XAML Preview

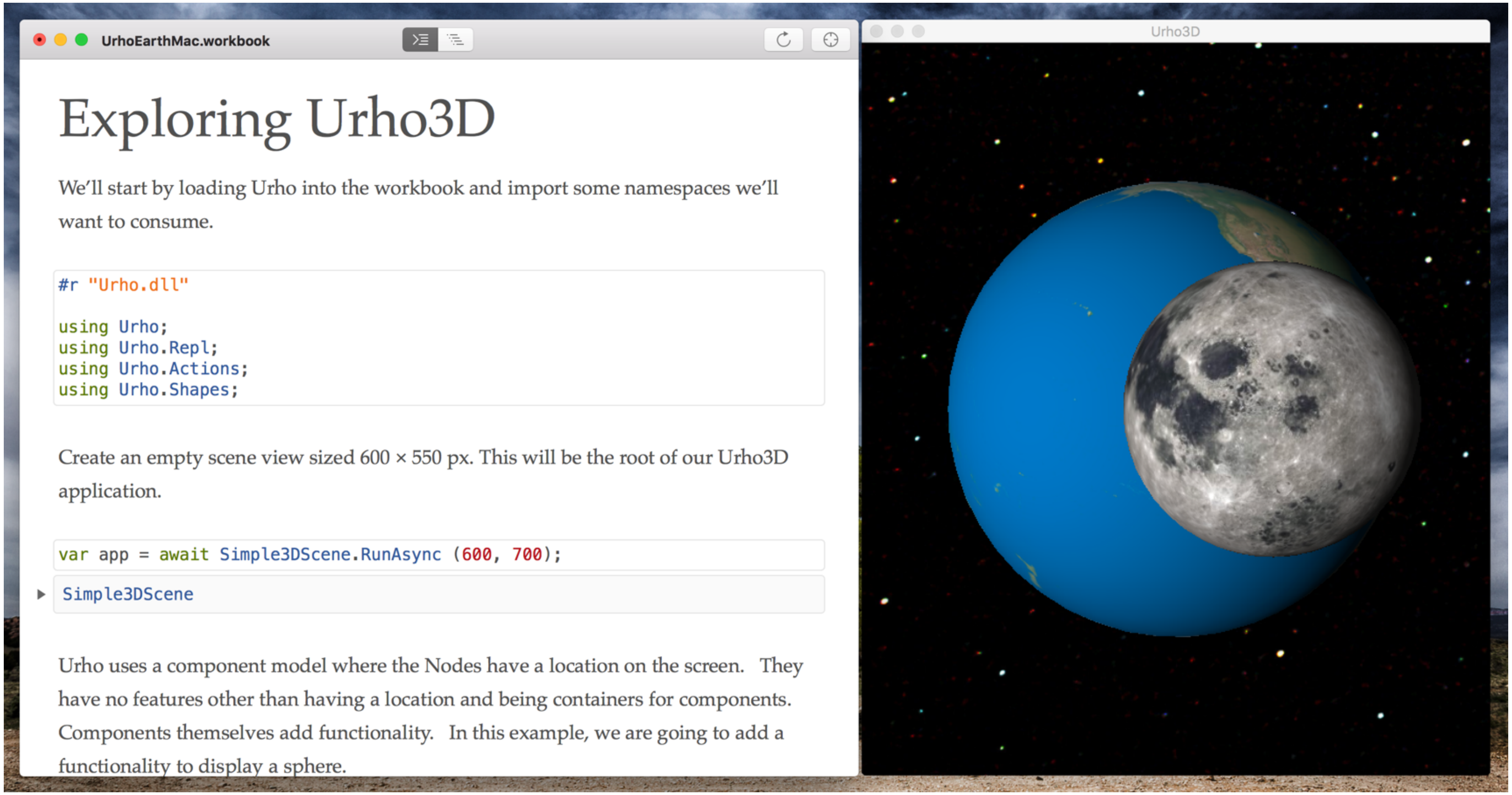


Generated file successfully.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms" xmlns:x="http://sc
    <ContentPage.Content>
        <StackLayout Orientation="Vertical"
                     Padding="20">
            <Label Text="Heading"
                   FontSize="36"
                   FontAttributes="Bold"
                   Margin="0,10,0,20" />
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="80" />
                    <RowDefinition Height="80" />
                    <RowDefinition Height="80" />
                </Grid.RowDefinitions>
                <BoxView Grid.Row="0"
                         HorizontalOptions="FillAndExpand"
                         VerticalOptions="Start"
                         BackgroundColor="Green"
                         HeightRequest="100"/>
                <BoxView Grid.Row="1"
                         HorizontalOptions="FillAndExpand"
                         VerticalOptions="Start"
                         BackgroundColor="Blue"
                         HeightRequest="100"/>
                <BoxView Grid.Row="2"
                         HorizontalOptions="FillAndExpand"
                         VerticalOptions="Start"
                         BackgroundColor="Yellow"
                         HeightRequest="100"/>
            </Grid>
            <Entry Margin="0,20,0,0"
                   Text="Entry" />
            <Button BackgroundColor="Red"
                    Text="Button"
                    TextColor="White"
                    FontAttributes="Bold"
                    FontSize="20" />
        </StackLayout>
    </ContentPage.Content>
</ContentPage>
```

New Features

Workbooks



Exploring Urho3D

We'll start by loading Urho into the workbook and import some namespaces we'll want to consume.

```
#r "Urho.dll"

using Urho;
using Urho.Repl;
using Urho.Actions;
using Urho.Shapes;
```

Create an empty scene view sized 600 × 550 px. This will be the root of our Urho3D application.

```
var app = await Simple3DScene.RunAsync (600, 700);
```

```
Simple3DScene
```

Urho uses a component model where the Nodes have a location on the screen. They have no features other than having a location and being containers for components. Components themselves add functionality. In this example, we are going to add a functionality to display a sphere.

New Features

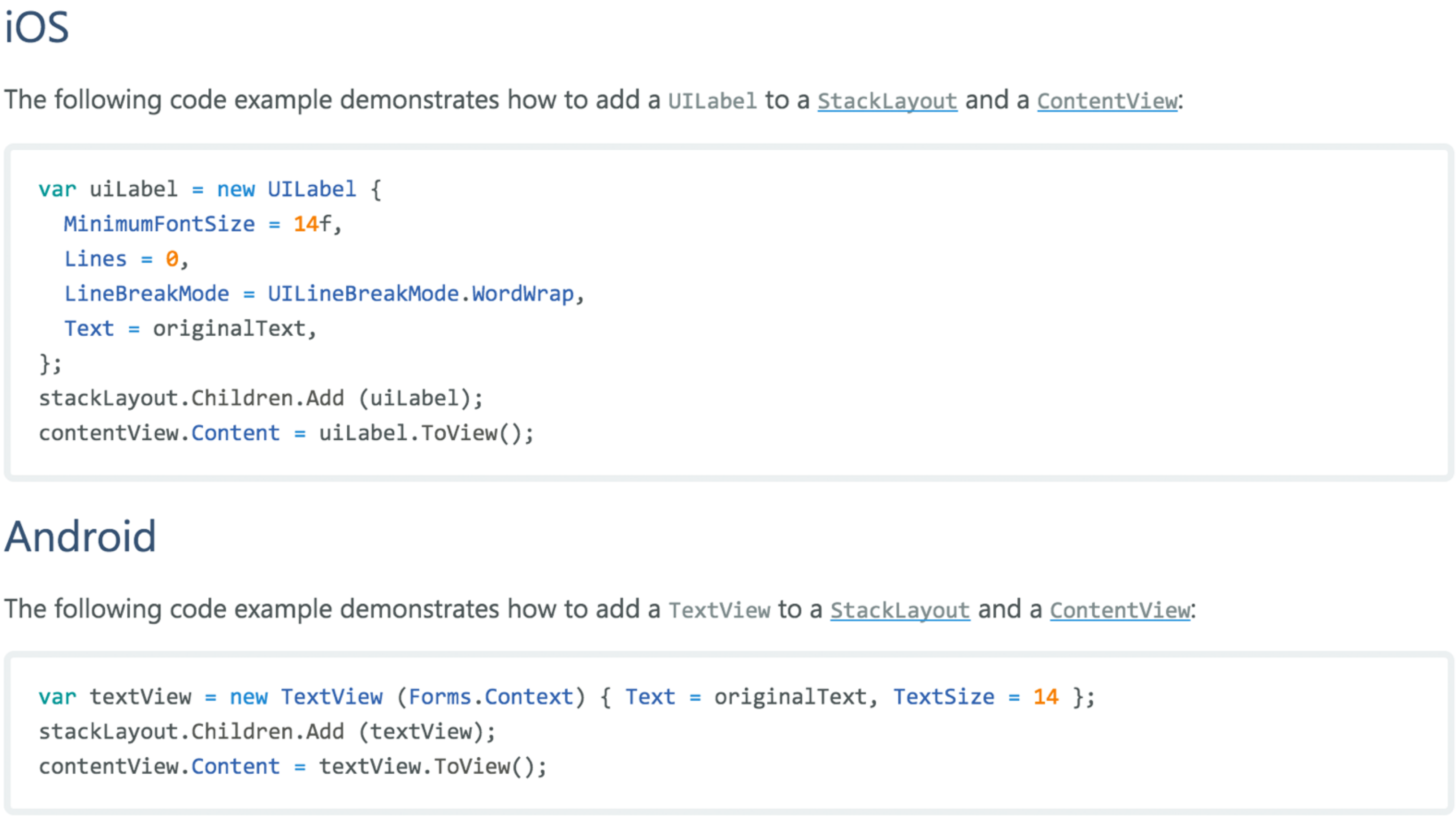
Workbooks

https://youtu.be/jgXCB51e4ak?t=36m51s

https://youtu.be/jgXCB51e4ak?t=45m51s

New Features

Embedded platform-specific controls (Shared projects only)



iOS

The following code example demonstrates how to add a `UILabel` to a `StackLayout` and a `ContentView`:

```
var uiLabel = new UILabel {
  MinimumFontSize = 14f,
  Lines = 0,
  LineBreakMode = UILineBreakMode.WordWrap,
  Text = originalText,
};
stackLayout.Children.Add (uiLabel);
contentView.Content = uiLabel.ToView();
```

Android

The following code example demonstrates how to add a `TextView` to a `StackLayout` and a `ContentView`:

```
var textView = new TextView (Forms.Context) { Text = originalText, TextSize = 14 };
stackLayout.Children.Add (textView);
contentView.Content = textView.ToView();
```

New Features

Test Cloud Live, Xamarin Test Recorder, HockeyApp Redesign

- remote-debugging on a real device in the Xamarin Test Cloud (2000 real devices)
- Record UI Test scenario that runs on Test Cloud
- Full CD chain with Xamarin Insights / HockeyApp —> Crash Reporting, Ad-Hoc distribution

And now?

- Xamarin.Forms is already usable for production
- Xamarin can fully replace native development with Xcode / Android Studio
- Xamarin University is a good platform for learning (live classes)

Thank you!

Questions?