

.NET MAUI


Roman Jašek

Riganti s.r.o.

Microsoft Most Valuable Professional (MVP)

roman.jasek@riganti.cz

Who knows?



C#
.NET

MVVM

.NET MAUI

„Standard“ Application Development



iOS

Objective-C
Swift

XCode



Android

Java
Kotlin

Android Studio



Windows

C#

Visual Studio

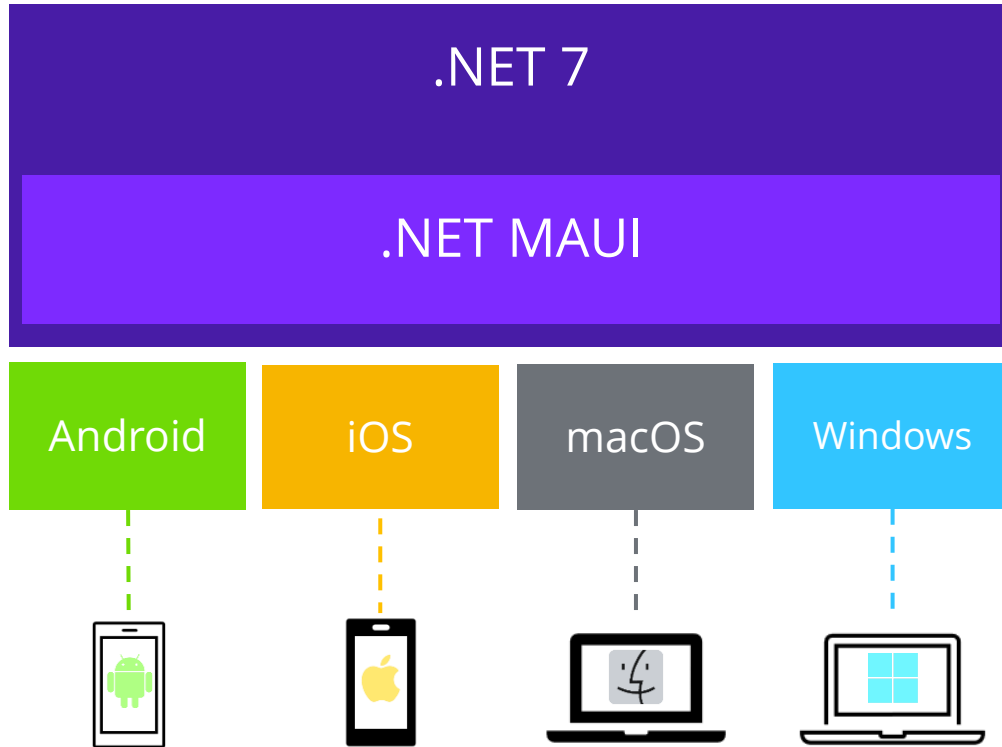


Mac OS

Objective-C
Swift

XCode

.NET Multi-platform App UI



Cross-platform, native UI

Single project system, single codebase

Deploy to multiple devices, mobile & desktop

How It Works - Structure

- Platform specific frameworks
 - .NET for Android
 - .NET for iOS
 - .NET for MacOS
 - Windows UI (WinUI) library
- Common BCL - .NET
- .NET Runtimes
 - Mono – Android, iOS, MacOS
 - WinRT – Windows

How It Works - UI

- Platform specific UI
 - Different platforms - different ways of defining UI
 - Can be defined separately using platform specific APIs
 - .NET for Android, .NET for iOS, .NET for MacOs, WinUI
- Common UI
 - Single framework for defining UI – mobile & desktop
 - XAML

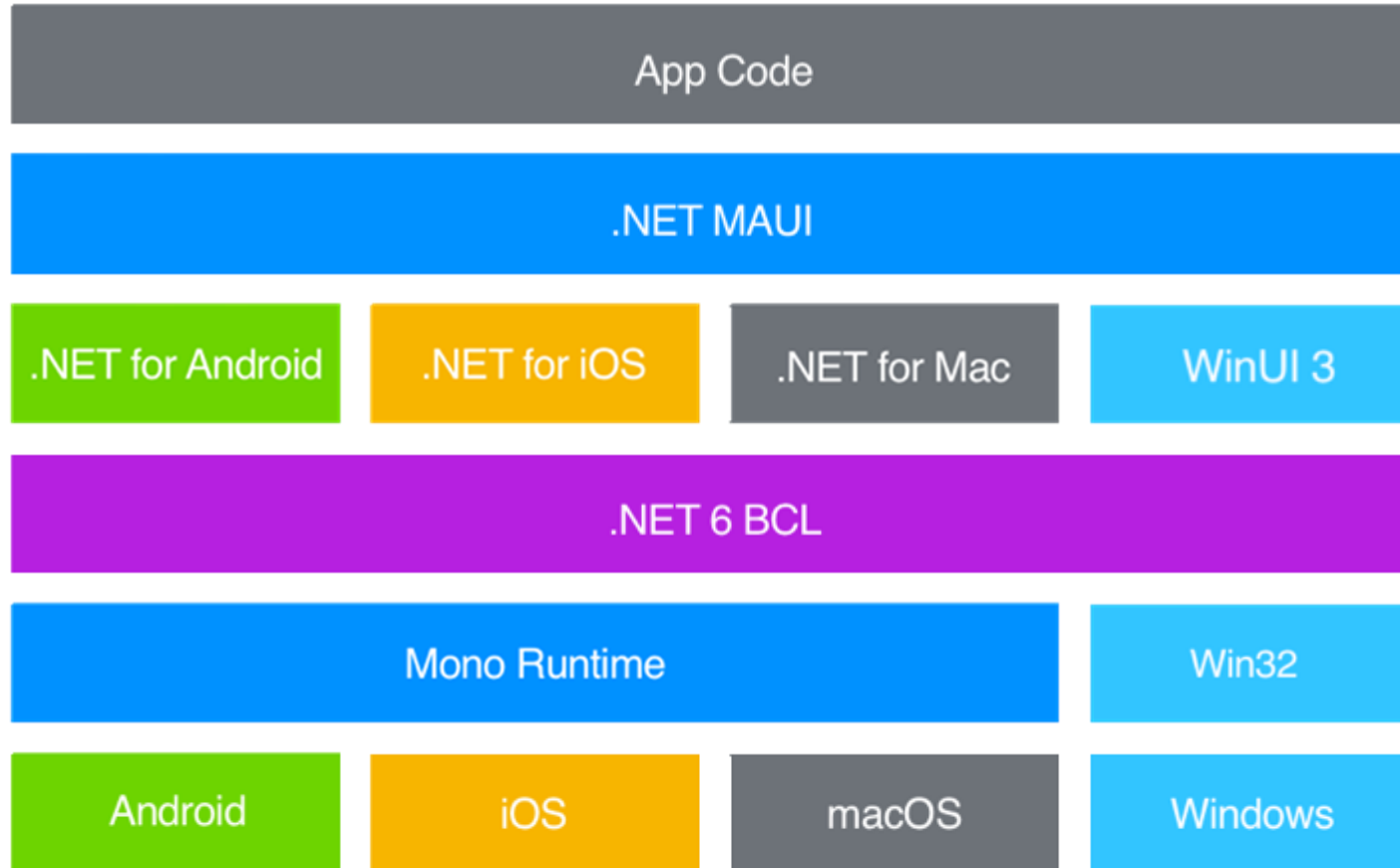
How It Works - compilation

- Android
 - C# compiles to intermediate language (IL)
 - JIT compilation to native assembly on app launch
- iOS
 - Fully ahead-of-time (AOT) compiled to native ARM assembly code
- MacOS
 - Using Mac Catalyst
 - Apple's solution to bring iOS Apps to desktop
 - Provides access to Mac OS APIs
- Windows
 - WinUI 3 library
 - Native apps and UWP

How It Works – Application Startup

- .NET Generic Host – DI, logging, configuration...
- Static **MauiProgram** class
 - Create builder, register dependencies..
 - Create a **MauiApp** instance
- **App** class
 - Derives from Application
 - Initializes application, sets initial page
- Separate startup points for each platform (Platforms folder)
 - **MainApplication.cs** (Android), **Program.cs** (iOS)
 - **App.xaml.cs** (Windows), **Program.cs** (Mac Os)

How .NET MAUI Works

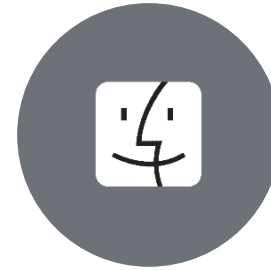
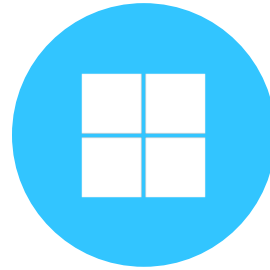


.NET MAUI

- Collection of Controls
- Layout engine for pages
- Navigation – pages, drawers
- Customizable handlers – enable platform specific controls
- APIs for native device features – GPS, accelerometer...
- Graphics library for 2D drawing code
- Single project, multi-targeting system
- .NET hot reload

Platforms

Official support *



Community



* **Tizen .NET** supported by Samsung

XAML

- Format for serialization of hierarchy of objects
- Mostly used for UI
- Possible connection to a code-behind class

Where is XAML used?

- Windows Presentation Foundation (WPF)
- Silverlight
- Windows Phone
- Universal Windows Platform (UWP)
- Windows Workflow Foundation (WF)
 - Not used for UI
- Xamarin.Forms
- WinUI3
- .NET MAUI

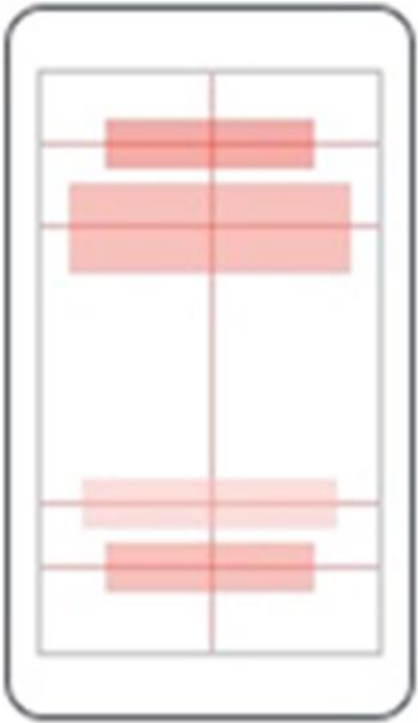
XAML

- **x:Class** ... name of generated class
- **UserControl** ... inheritance
- **xmlns:x** ... special namespace for XAMLu (mandatory)
- **xmlns** ... namespace with built-in framework controls

```
<?xml version="1.0" encoding="utf-8" ?>  
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"  
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"  
             x:Class="CookBook.Mobile.Views.Page1">  
</ContentPage>
```

Layouts

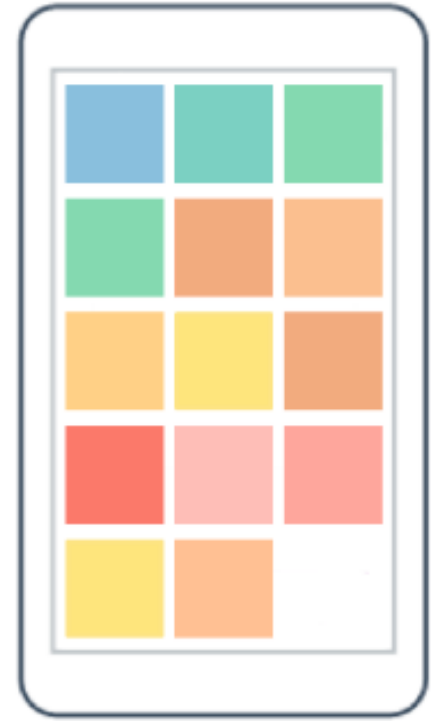
AbsoluteLayout



RelativeLayout

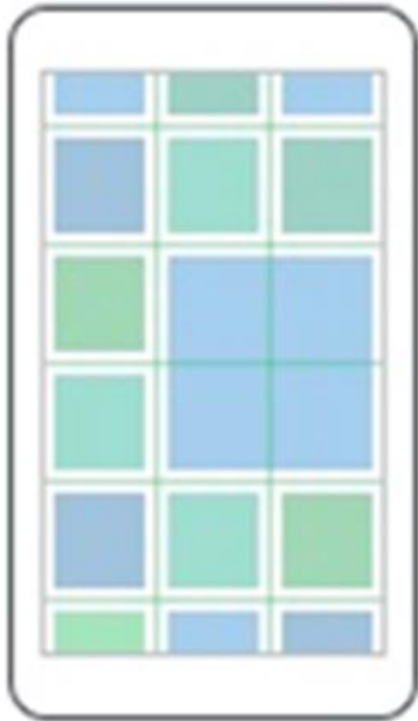


FlexLayout

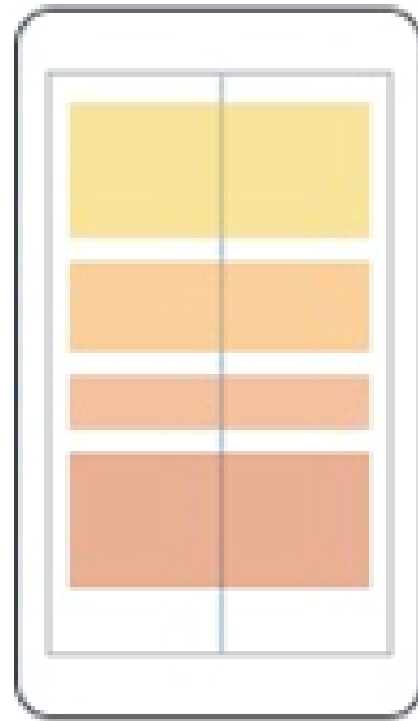


Layouts

Grid

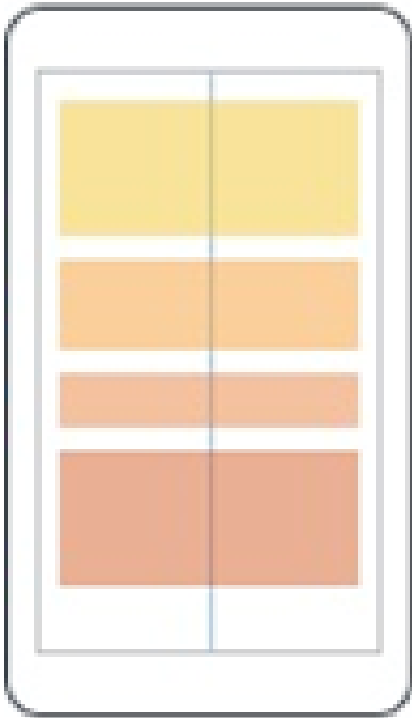


StackLayout



Layouts – StackLayout

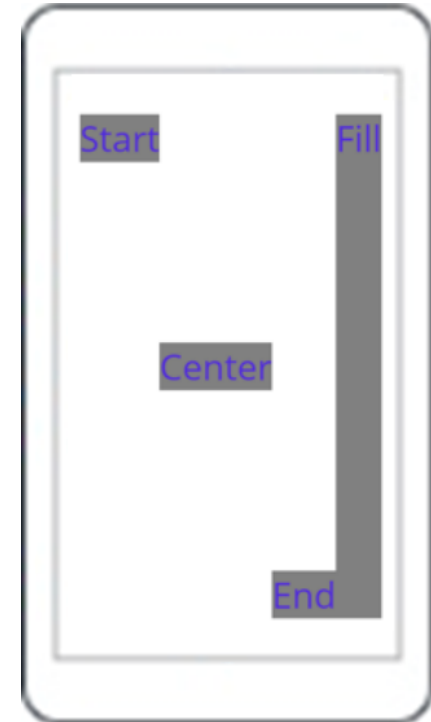
StackLayout



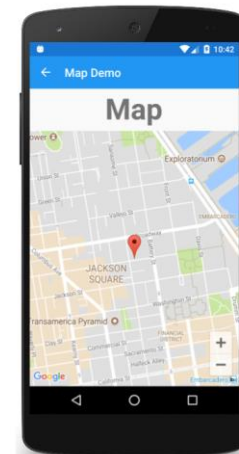
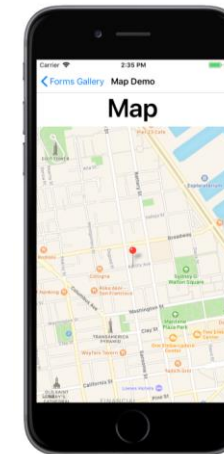
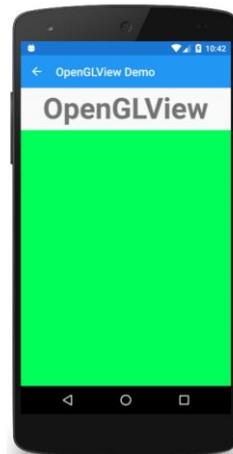
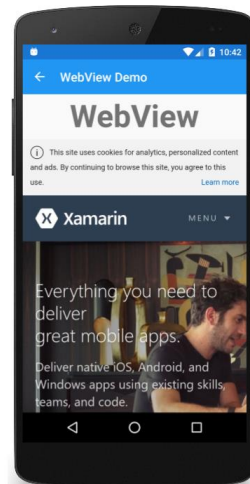
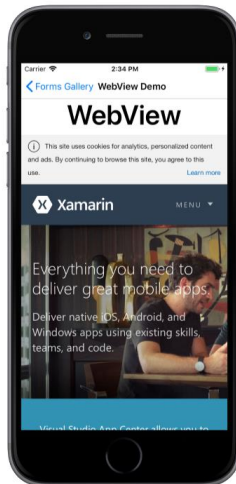
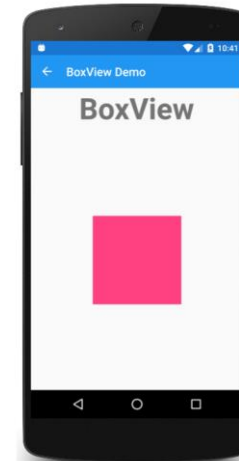
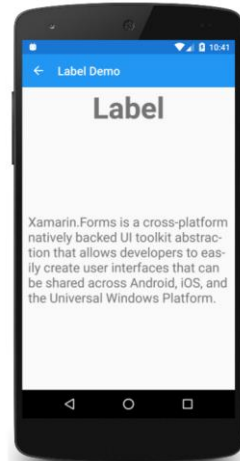
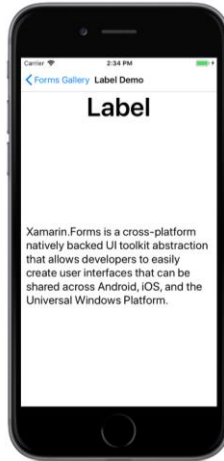
VerticalStackLayout



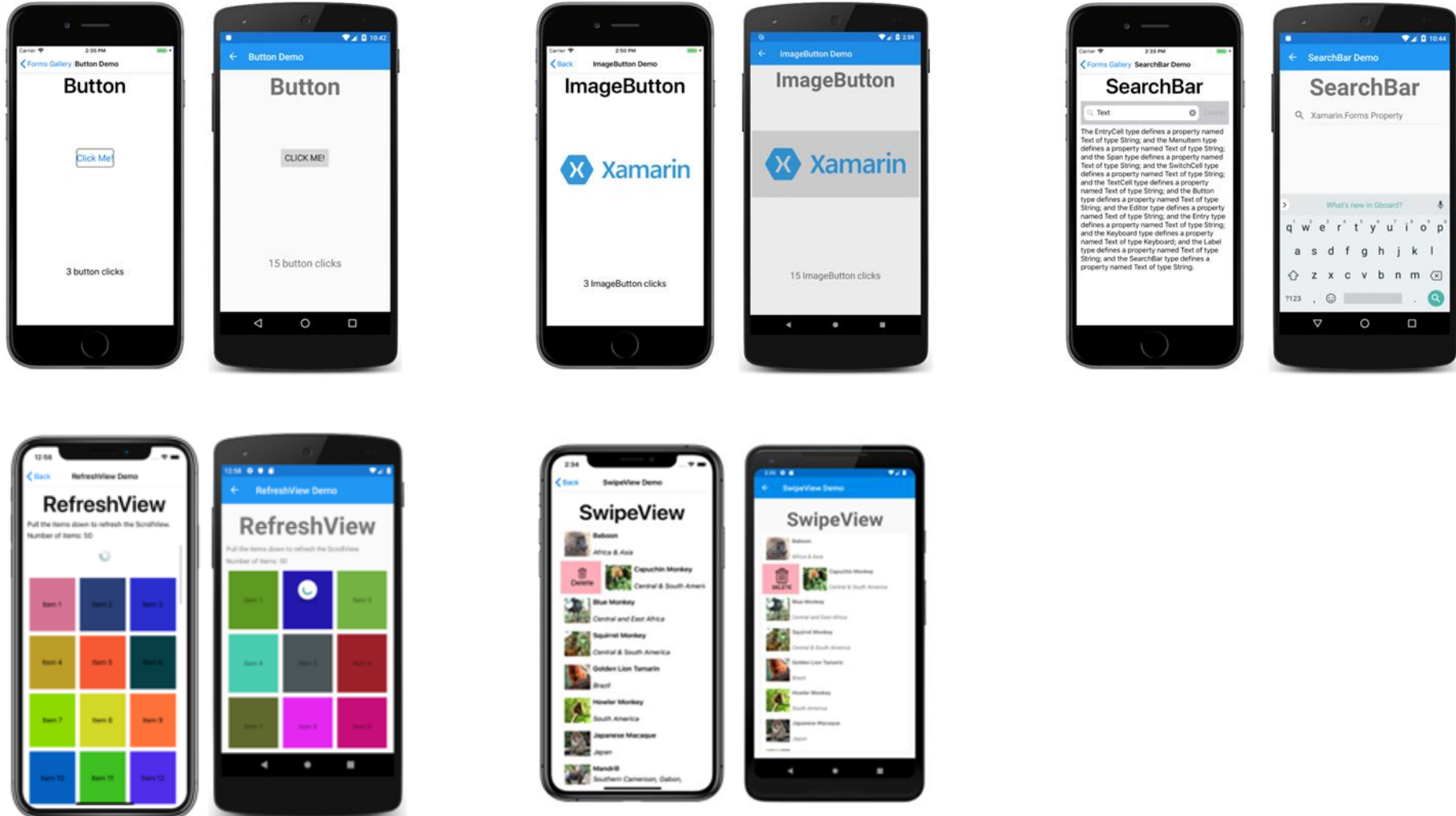
HorizontalStackLayout



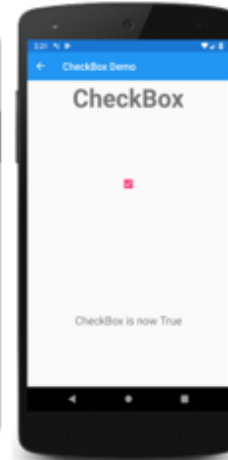
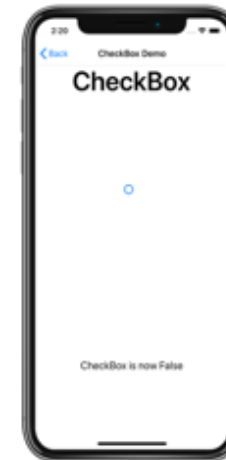
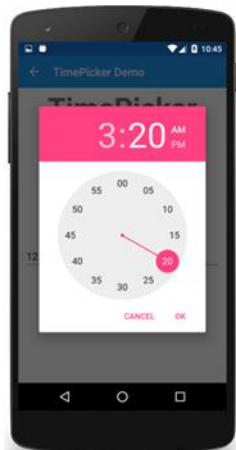
Content Presentation



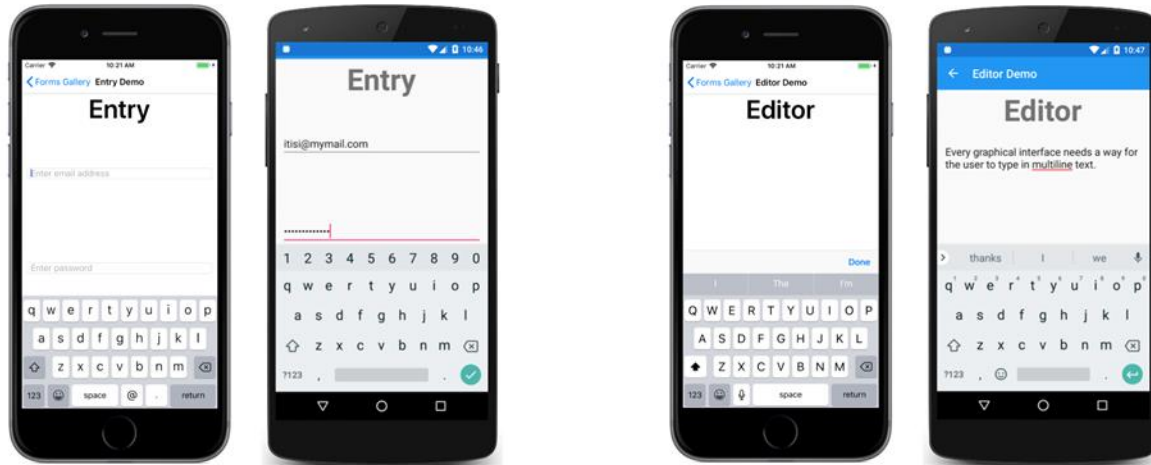
Actionable Controls



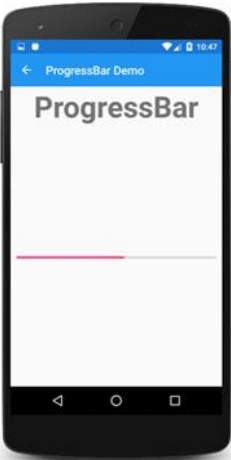
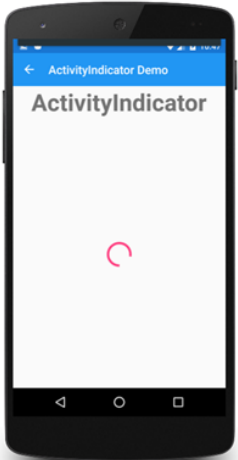
Setting Values



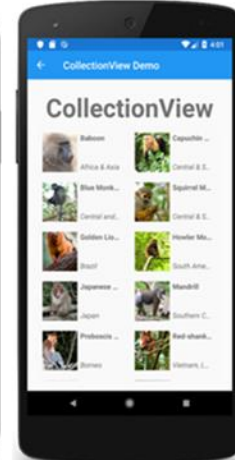
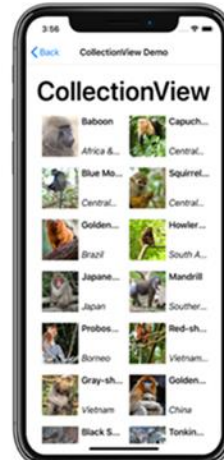
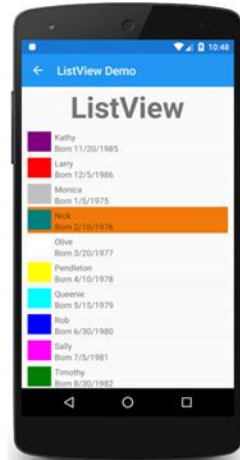
Editing Text



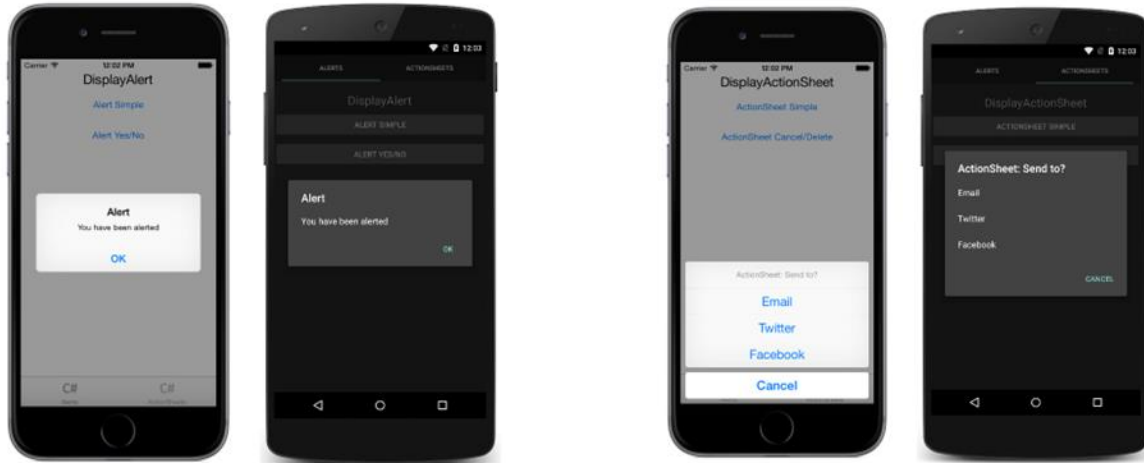
Activity Indication



Collections



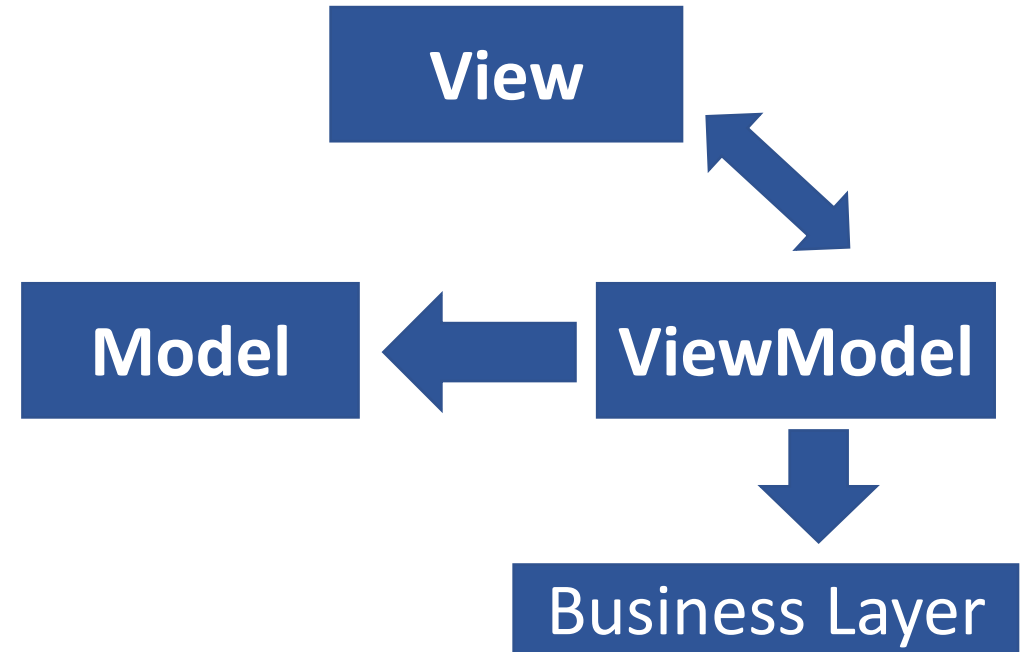
Pop-ups



Demo

Model View ViewModel

- View
 - Displays data and deals with user interaction
- Model
 - Represents data structure
- ViewModel
 - Keeps context of current view
 - Notifies View when changes happen



Data binding

- Connection of code and XAML
- **Between ViewModel and View**
- `Text="{Binding Path=Operand1}"`
- **Between controls in View**
- `Text="{Binding Source={x:Reference Name=DisplayAlertButton}, Path=Text}"`

MVVM Frameworks

- MVVM Cross
- Simple MVVM
- MVVM Light
- Catel
- ReactiveUI

.NET MAUI Blazor

- Use Blazor in applications
- Razor, standard Blazor code
- Hosted in WebView
- Runs natively
 - No SignalR, no WebAssembly
 - Access to system APIs (file access, network access...)
- But why?
 - Blazor/web developer using razor – develop mobile applications
 - Use Blazor components in any .NET applications (MAUI, WPF, UWP...)
 - The cross-platform UI stuff is already solved – it's in a browser!

MAUI vs. Xamarin

- .NET 7, C# 11
- Single project
 - Unified working with images, fonts, splash screen and other resources
 - Custom Renderers -> Handlers
- IoC/DI
- Unified working with WinRT
- Official support for Mac OS

Advantages of .NET MAUI

- XAML
- .NET
- Good documentation
- Sharing code – 90%+

Materials

- Lecture materials: <https://github.com/jasho/pv239-maui>
- Sample project: <https://github.com/jasho/cookbook-maui>