The current state of **.NET Multi-platform App UI**

Almir Vuk

Microsoft MVP

Senior Software Engineer at **AgentLocator** Inc

@almirvuk

Almir Vuk

I am **Software Development Engineer** and **Microsoft MVP** working with **.NET** and **C#,** crafting apps with **ASP.NET Core** for Web and **Xamarin** for mobile. Currently part of **AgentLocator Inc**.

Spending free time, running, playing chess, speaking on conferences, answering questions **StackOverflow** and committing to open-source projects on **GitHub.**

MVP Microsoft® Most Valuable Professional

AGENT LOCATOR

almirvuk.com
@almirvuk

Few questions… before we start ☺

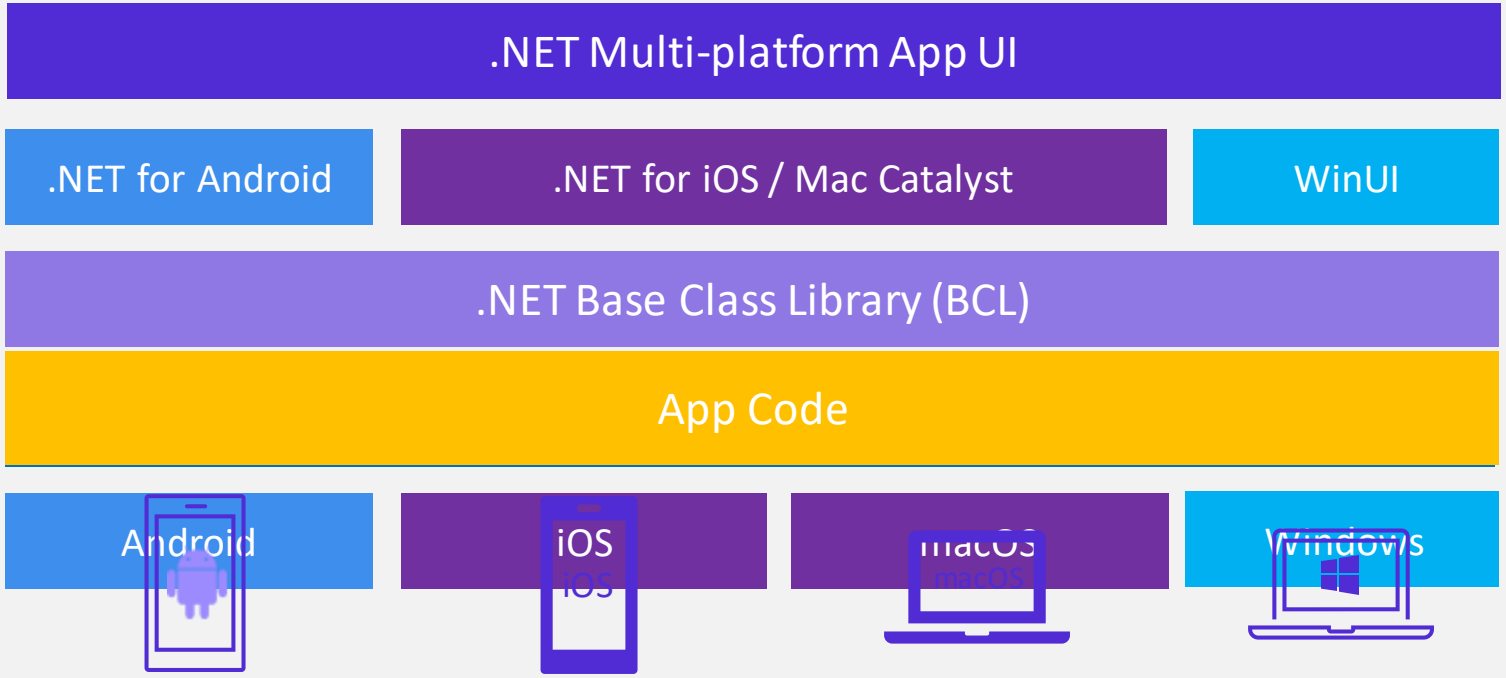
.NET MAUI is the most productive way to develop native apps that perform great on any device that runs Android, iOS, macOS, or Windows from a single codebase.

Who .NET MAUI is for?

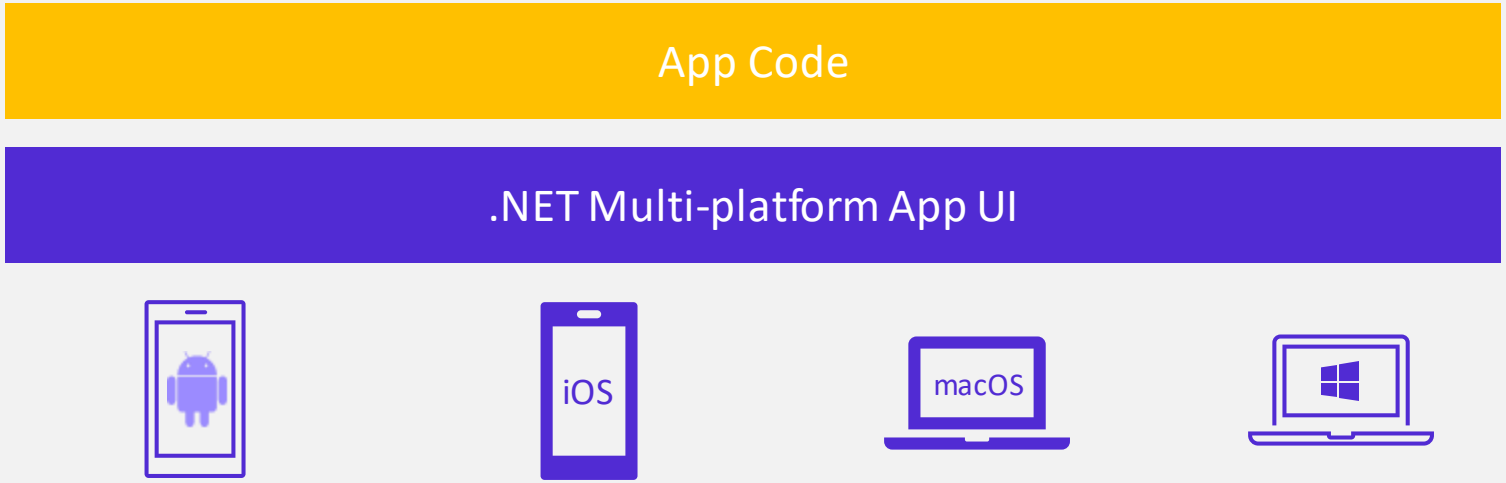
.NET MAUI is for developers who want to:

- Write cross-platform apps in XAML and C#, from a single shared code-base in Visual Studio.
- Share UI layout and design across platforms.
- Share code, tests, and business logic across platforms.

How does .NET MAUI work?

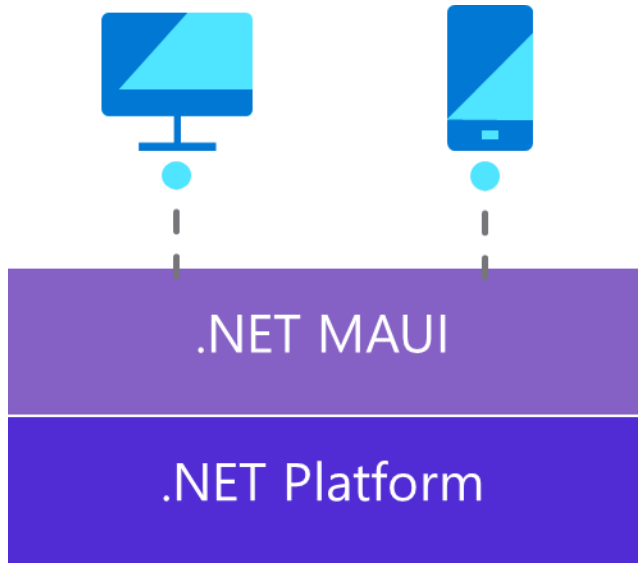


How does .NET MAUI work?



.NET Multi-platform App UI (MAUI)

- Xamarin integrated into .NET
  - github.com/dotnet/maui
  - github.com/dotnet/android
  - github.com/dotnet/macios
- Common .NET experiences
  - CLI
  - Visual Studio Code
- Android, iOS, macOS, and Windows
- Support for modern app models like Blazor and Comet



Building UI in .NET MAUI

Optimized for XAML, C#, and MVVM

```xaml
<StackLayout>
    <Label Text="Welcome to Maui!" />

    <Button Text="{Binding Text}"
            Clicked="{Binding ClickCommand}" />
</StackLayout>
```

```
0 references
public ICommand ClickCommand { get; }
1 reference
public string Text { get; set; } = "Click me";

int count = 0;
0 references
void ExecuteClickCommand()
{
    count++;
    Text = $"You clicked {count} times.";
}
```

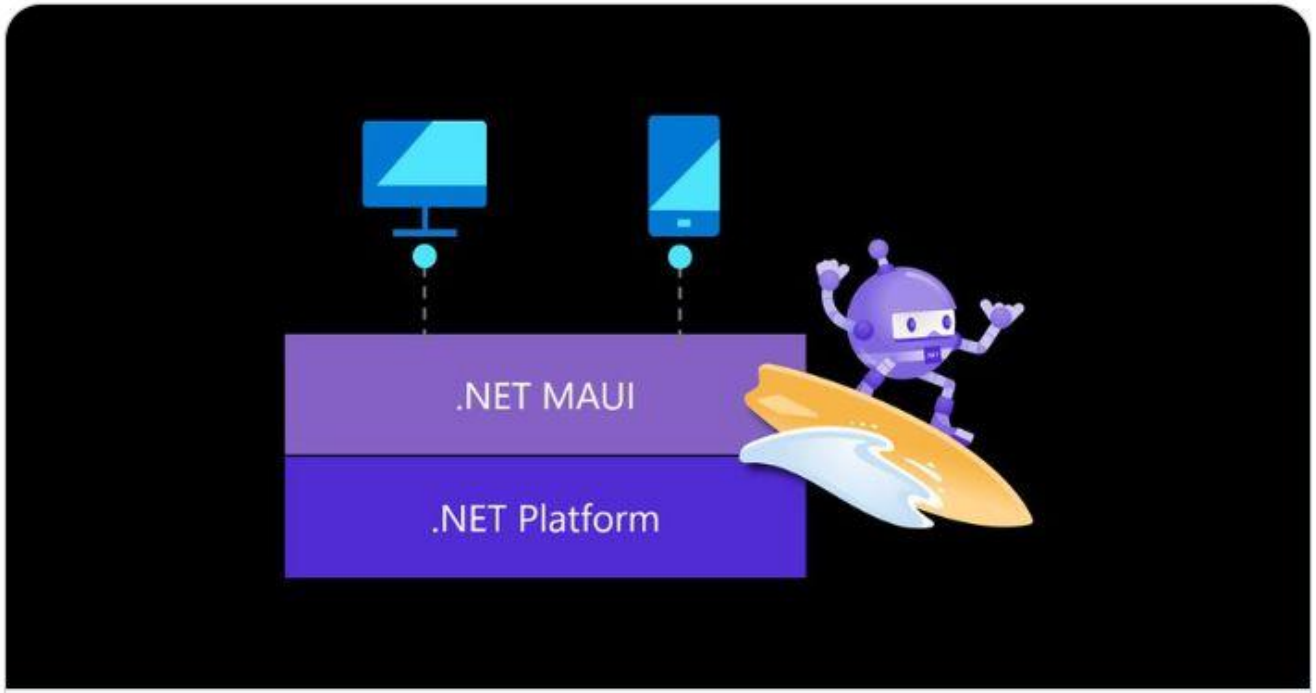
The home of .NET MAUI is:

https://github.com/dotnet/maui

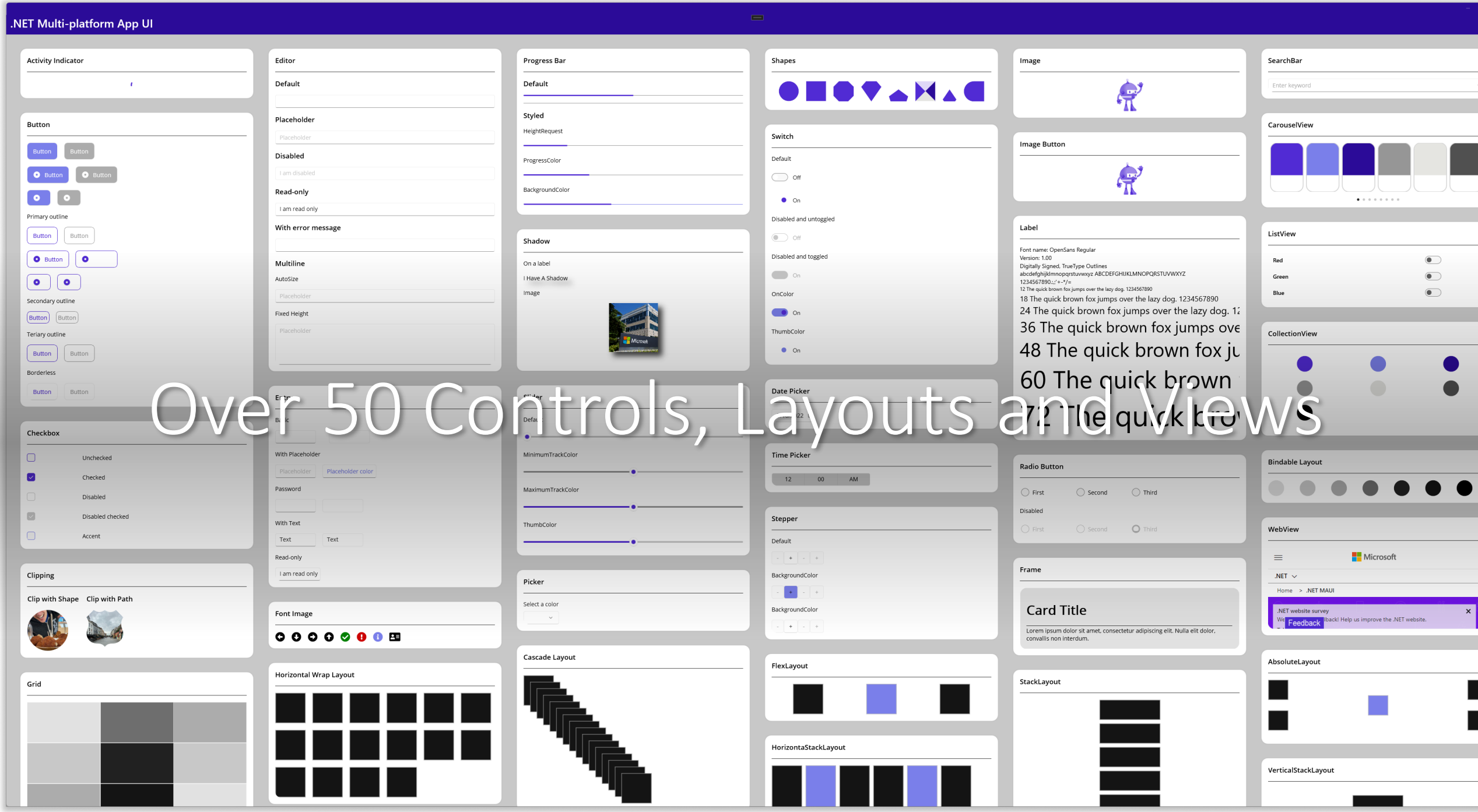
Fun fact 🎉

Redth
@redth

Congrats to @almirvuk who is the first community contributor of a Handler PR for #dotnetmaui github.com/dotnet/maui/pu... ! Thank you! You can help contribute too! github.com/dotnet/maui/wi...



Implement IsTextPredictionEnabled in IEntry Handlers by almirvuk · Pull Request...
Implements #381 Adds bool IsTextPredictionEnabled { get; } to the IEntry interface Adds IsTextPredictionEnabled property map to EntryHandler Adds ...
github.com

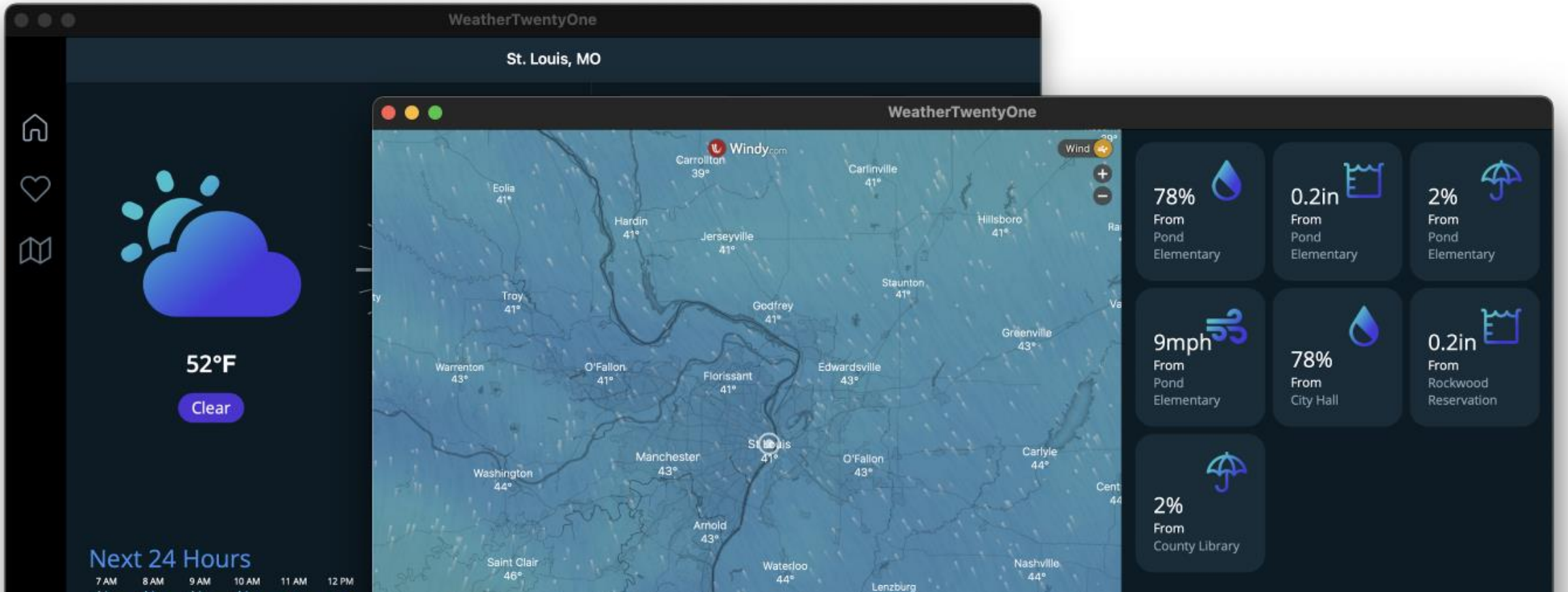

.NET Multi-platform App UI
Activity Indicator
Button
Primary outline
Secondary outline
Teriary outline
Borderless
Checkbox
Unchecked
Checked
Disabled
Disabled checked
Accent
Clipping
Clip with Shape
Clip with Path
Grid
Editor
Default
Placeholder
Disabled
Read-only
I am read only
With error message
Multiline
AutoSize
Fixed Height
With Placeholder
Password
With Text
Font Image
Horizontal Wrap Layout
Progress Bar
Default
Styled
HeightRequest
ProgressColor
BackgroundColor
Shadow
On a label
I Have A Shadow
Image
MinimumTrackColor
MaximumTrackColor
ThumbColor
Picker
Select a color
Cascade Layout
Shapes
Switch
Default
Off
On
Disabled and untoggled
Disabled and toggled
OnColor
ThumbColor
Date Picker
Time Picker
12
00
AM
Stepper
Default
BackgroundColor
FlexLayout
HorizontaStackLayout
Image
Image Button
Label
Font name: OpenSans Regular
Version: 1.00
Digitally Signed, TrueType Outlines
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890.:,'+-*/=
12 The quick brown fox jumps over the lazy dog. 1234567890
18 The quick brown fox jumps over the lazy dog. 1234567890
24 The quick brown fox jumps over the lazy dog. 12
36 The quick brown fox jumps ove
48 The quick brown fox ju
60 The quick brown
72 The quick brow
Radio Button
First
Second
Third
Disabled
Frame
Card Title
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla elit dolor, convallis non interdum.
StackLayout
SearchBar
Enter keyword
CarouselView
ListView
Red
Green
Blue
CollectionView
Bindable Layout
WebView
Microsoft
.NET
Home > .NET MAUI
.NET website survey
Feedback
AbsoluteLayout
VerticalStackLayout
Over 50 Controls, Layouts and Views

Controls

CarouselView	CollectionView	RefreshView		
IndicatorView	Checkbox	RadioButton	ImageButton	SwipeView
ActivityIndicator	BoxView	Button	DatePicker	Editor
Entry	Image	Label	ListView	Map
OpenGLView	Picker	ProgressBar	SearchBar	Slider
Stepper	TableView	TimePicker	WebView	EntryCell
ImageCell	SwitchCell	TextCell	ViewCell	Path
Ellipse	Polygon	Rectangle	Line	Polyline

Multi-window

```
var secondWindow = new Window { Page = new NewPage1 {  } };
Application.Current.OpenWindow(secondWindow);
```



Easily customize controls

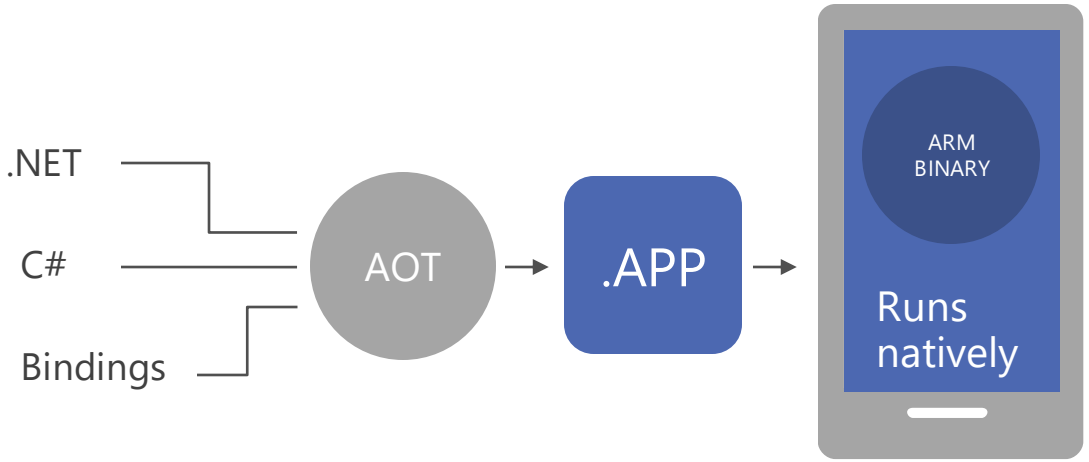
```
#if ANDROID
Handlers.EntryHandler
    .EntryMapper[nameof(IEntry.BackgroundColor)] = (h, v) =>
    {
        (h.NativeView as global::Android.Views.Entry).UnderlineVisible = false
    };
#endif
```

#ifdef if platform specific

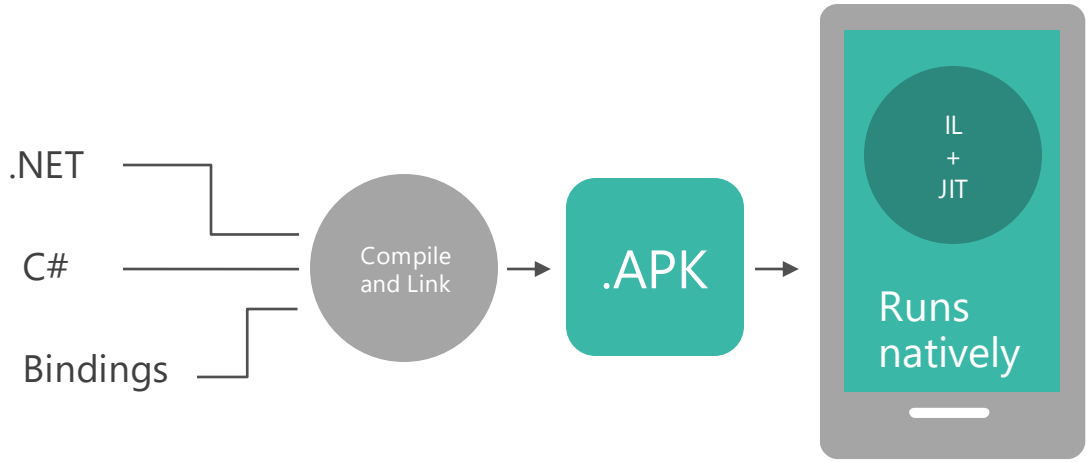
Access the handler from anywhere

Dictionary of all props

Built on trusted technology

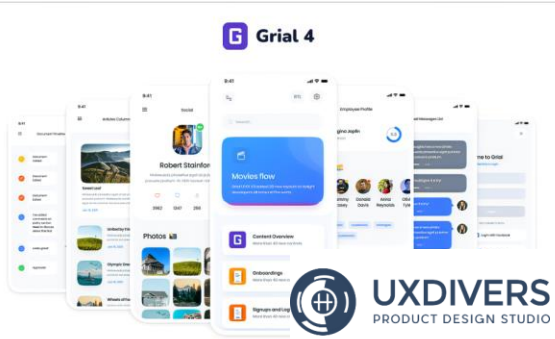


.NET for iOS does full Ahead Of Time (AOT) compilation to produce an ARM binary for Apple's App Store.



.NET for Android takes advantage of Just In Time (JIT) & Ahead of Time (AOT) compilation on the Android device.

Controls, components, plugins, libraries and more...



AndroidX
AlohaKit
CommunityToolkit.MVVM
CommunityToolkit.Maui
CommunityToolkit MauiCompat
CommunityToolkit Markup.MauiCompat
DevExpress
Facebook
FreshMvvm.Maui
Google APIs for iOS
Google Play Services Client Libraries
GrialKit
MauiAnimation
Microsoft.Maui.Graphics
MR.Gestures
MSAL
Prism.Maui
Plugin.Fingerprint
Plugin.InAppBilling
Plugin.StoreReview
Plugin.ValidationRules
Progress Telerik UI for .NET MAUI
ReactiveUI.Maui
Realm by MongoDB
Sentry
Shiny
SkiaSharp & Skottie
Syncfusion .NET MAUI Controls
TemplateUI
User Dialogs

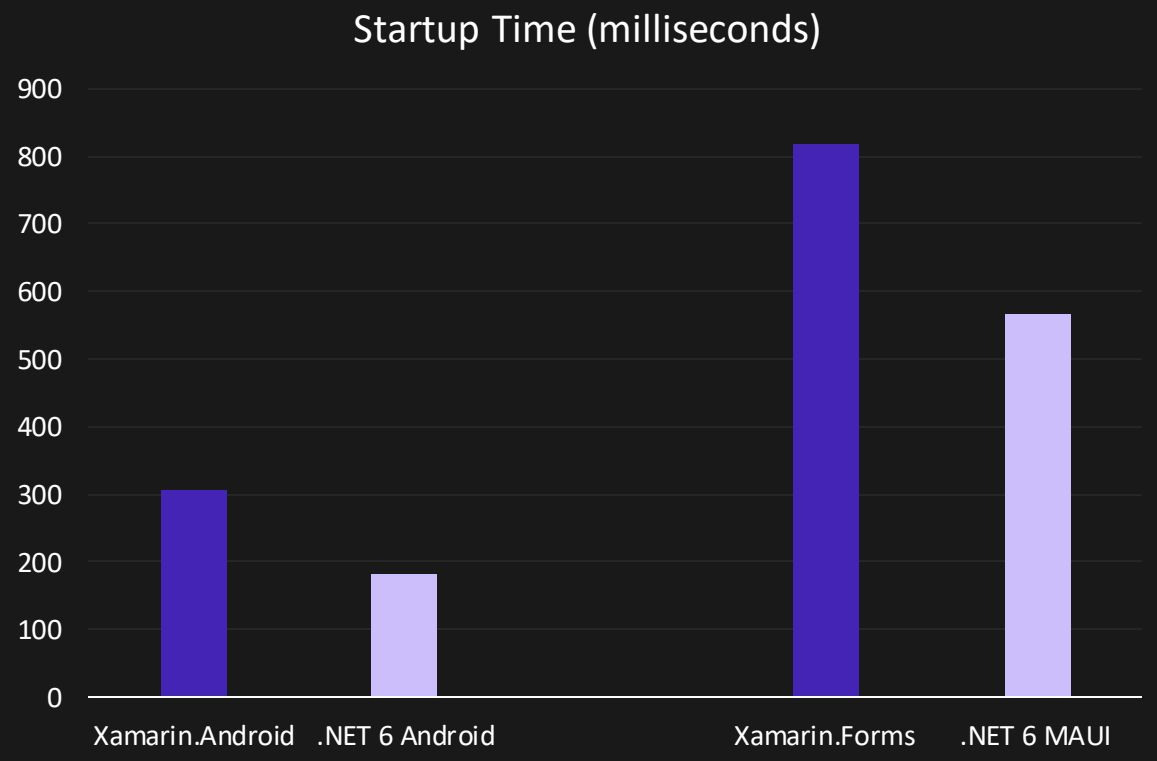
.NET MAUI perf improvements

68%

Faster than Xamarin.Android

44%

Faster than Xamarin.Forms



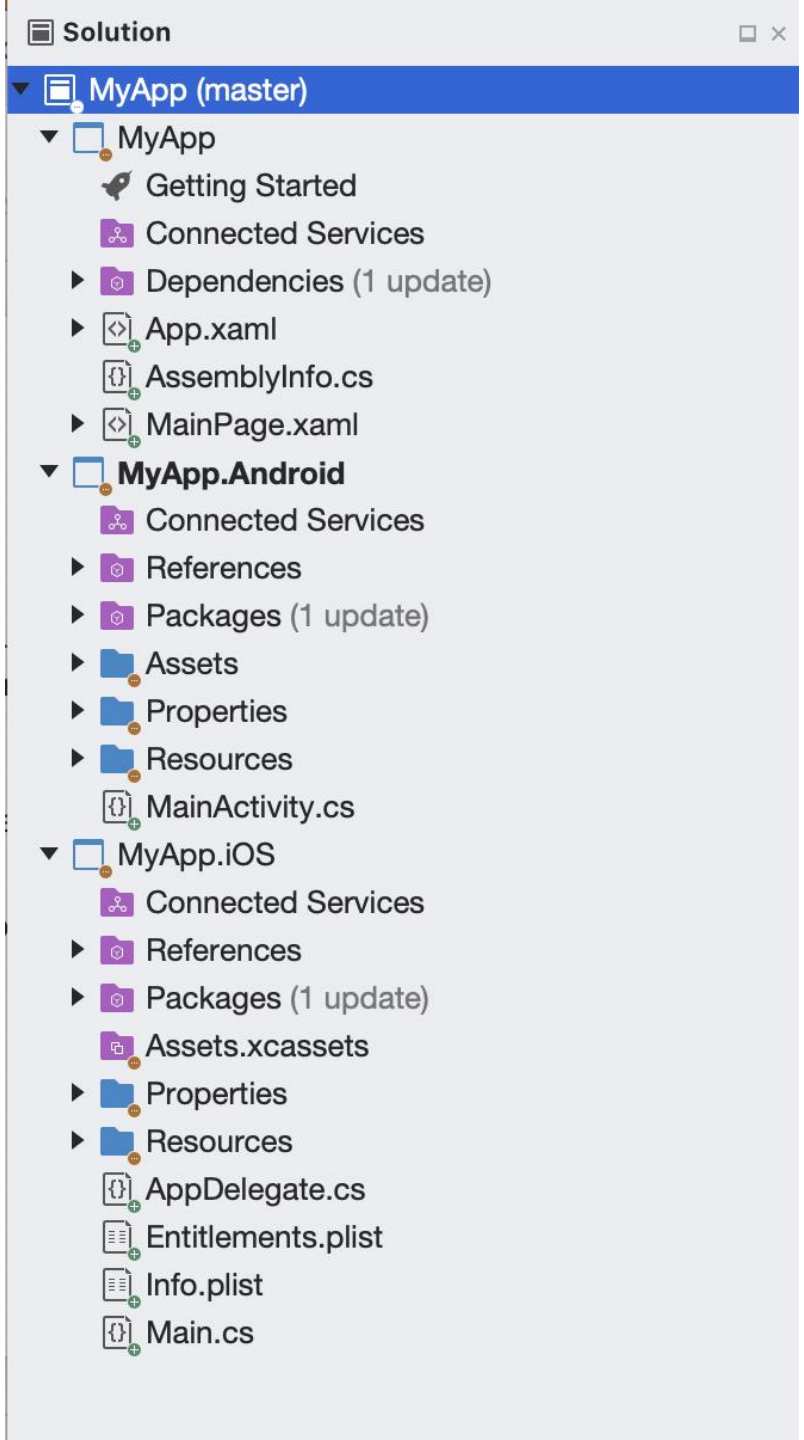
Project System Improvements

Optimized for Cross-Platform Workflows

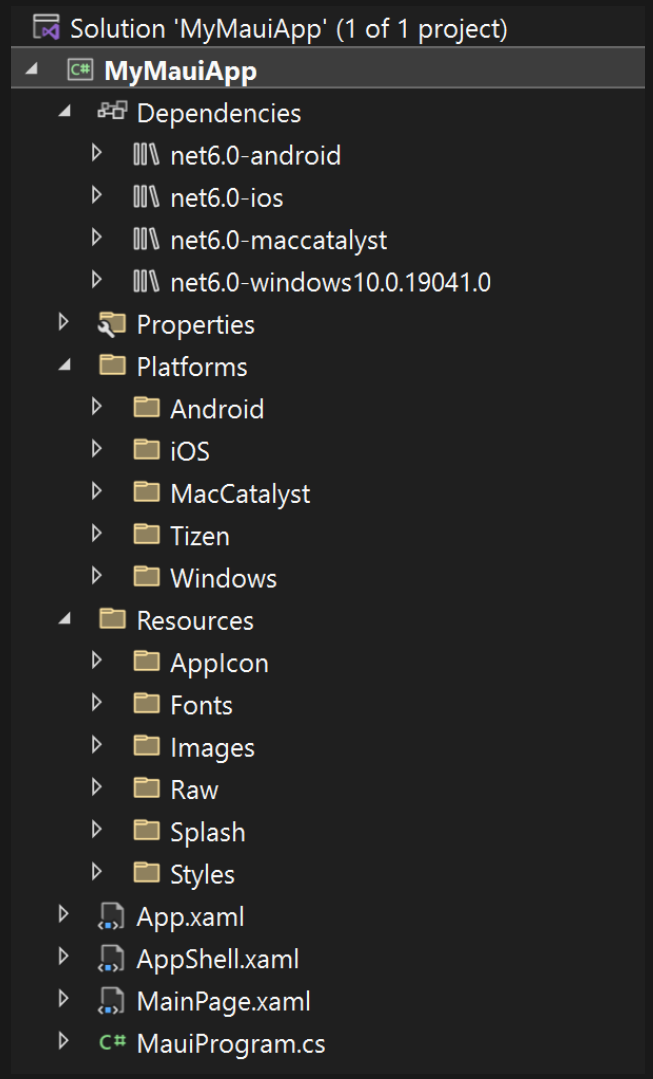
Single Project

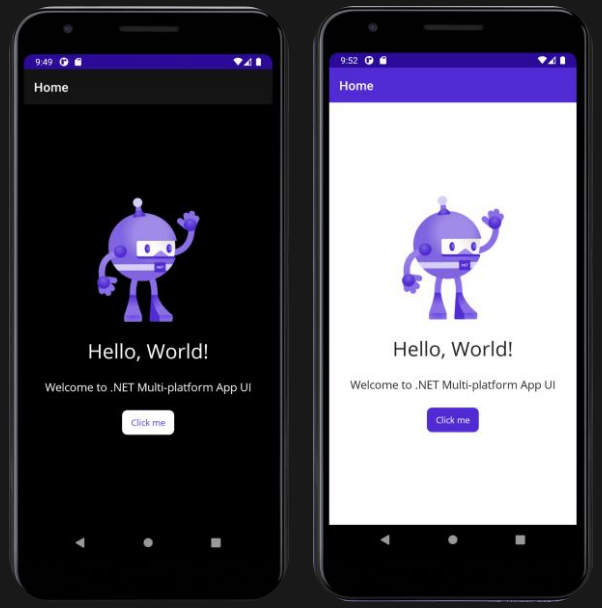
Reducing multiple projects to a single project alleviates several points of friction when developing apps that target multiple platforms.

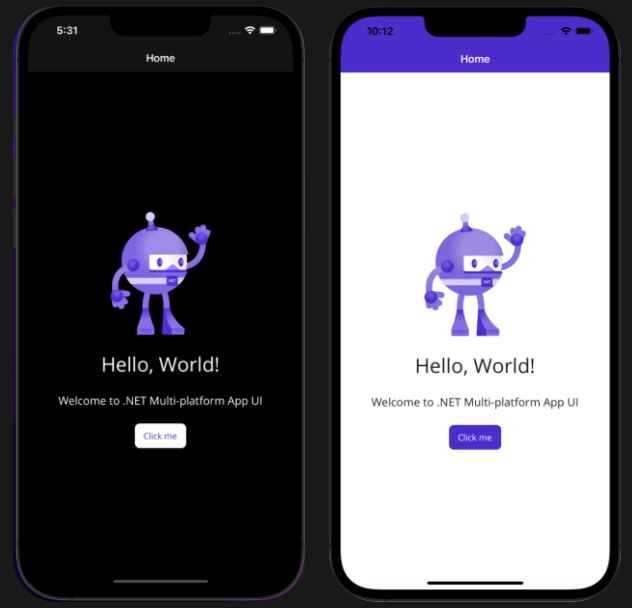
- Images
- Fonts
- Platform code
- Splash screen

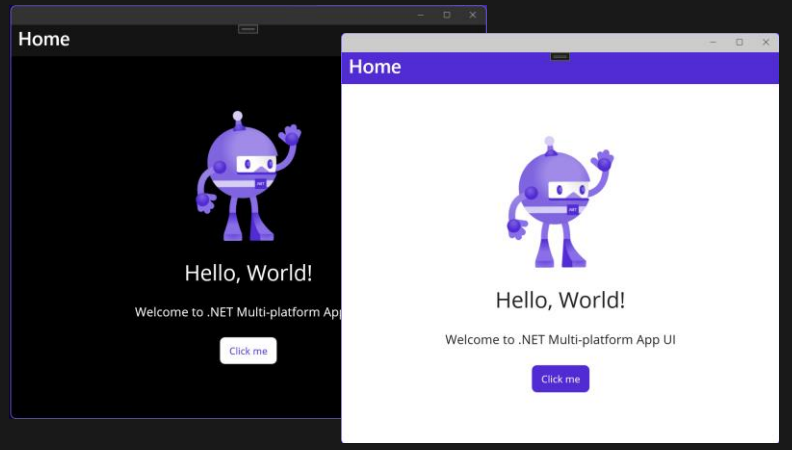


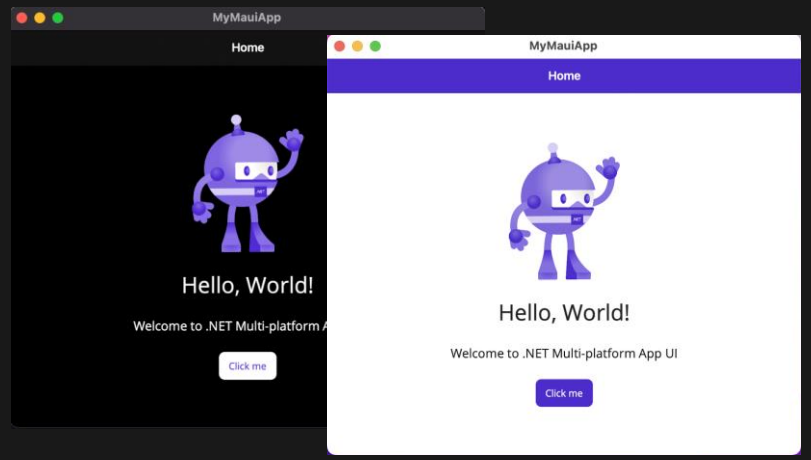
One codebase, single project, multi target

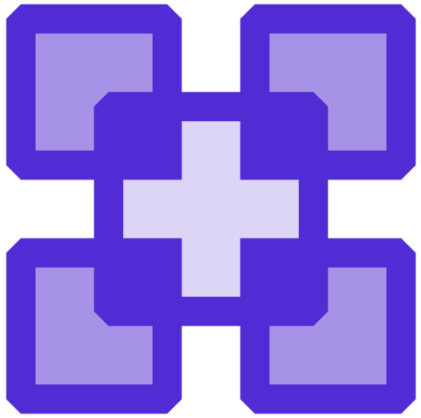












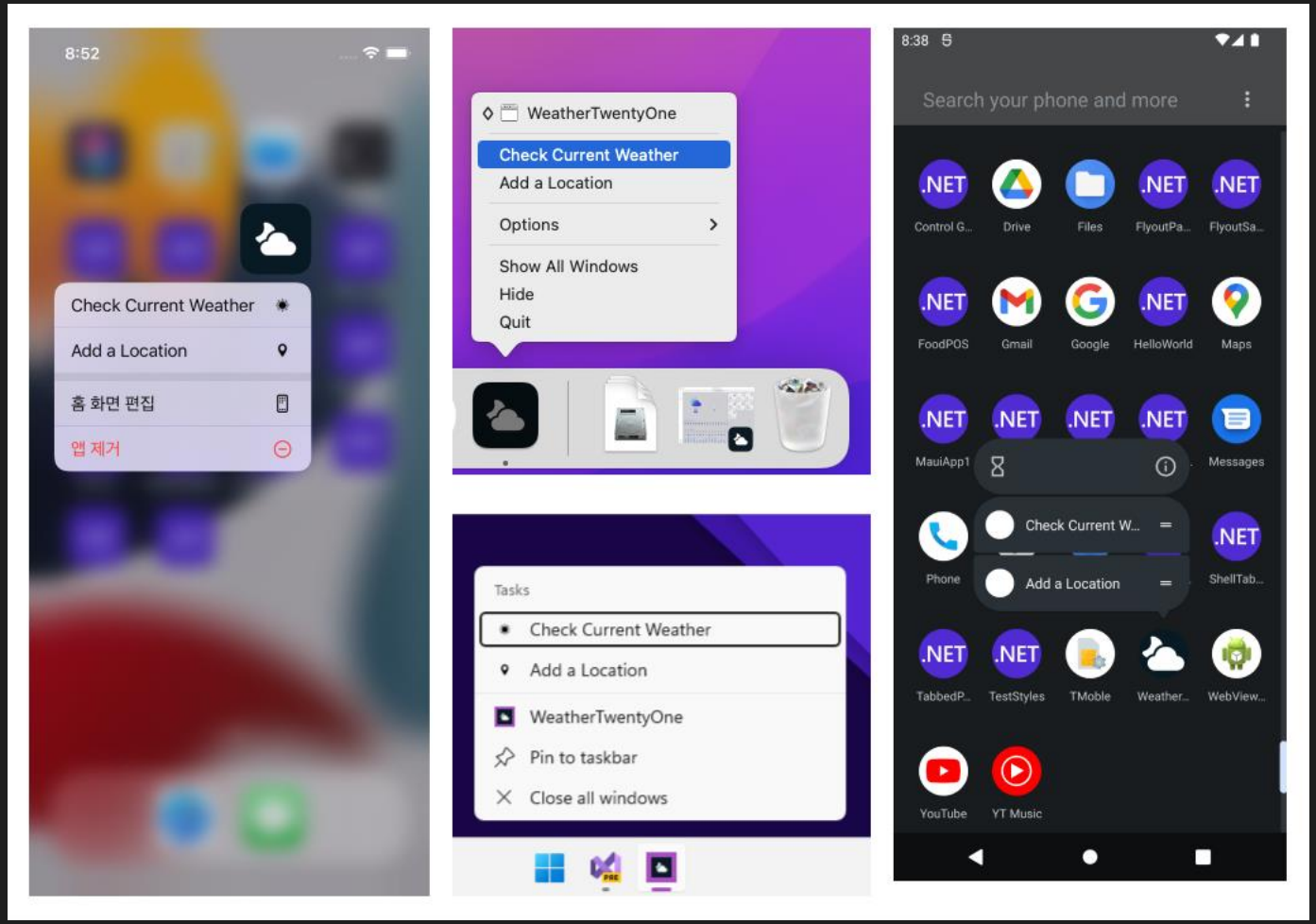
Platform API Access

Beyond UI

Simple access to APIs

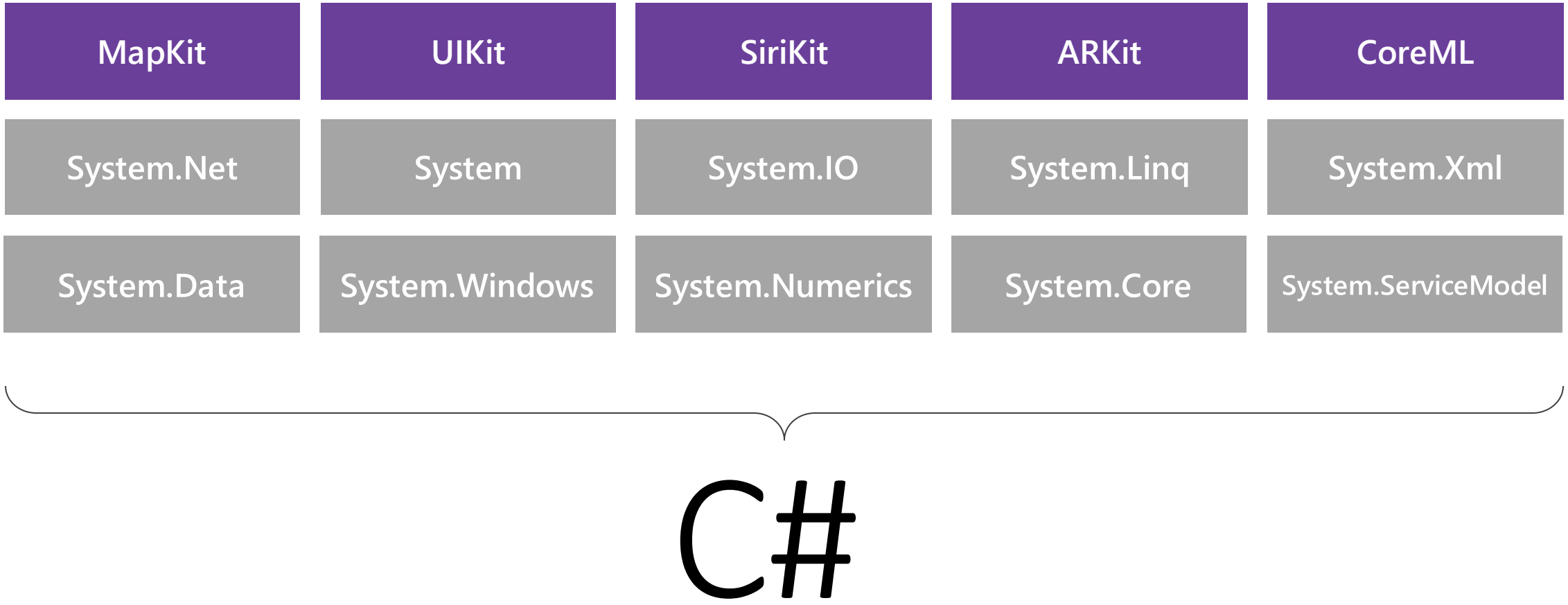
Access services and features of each platform

- App actions
- Clipboard
- Device sensors
- File system
- Network
- Notifications
- +More

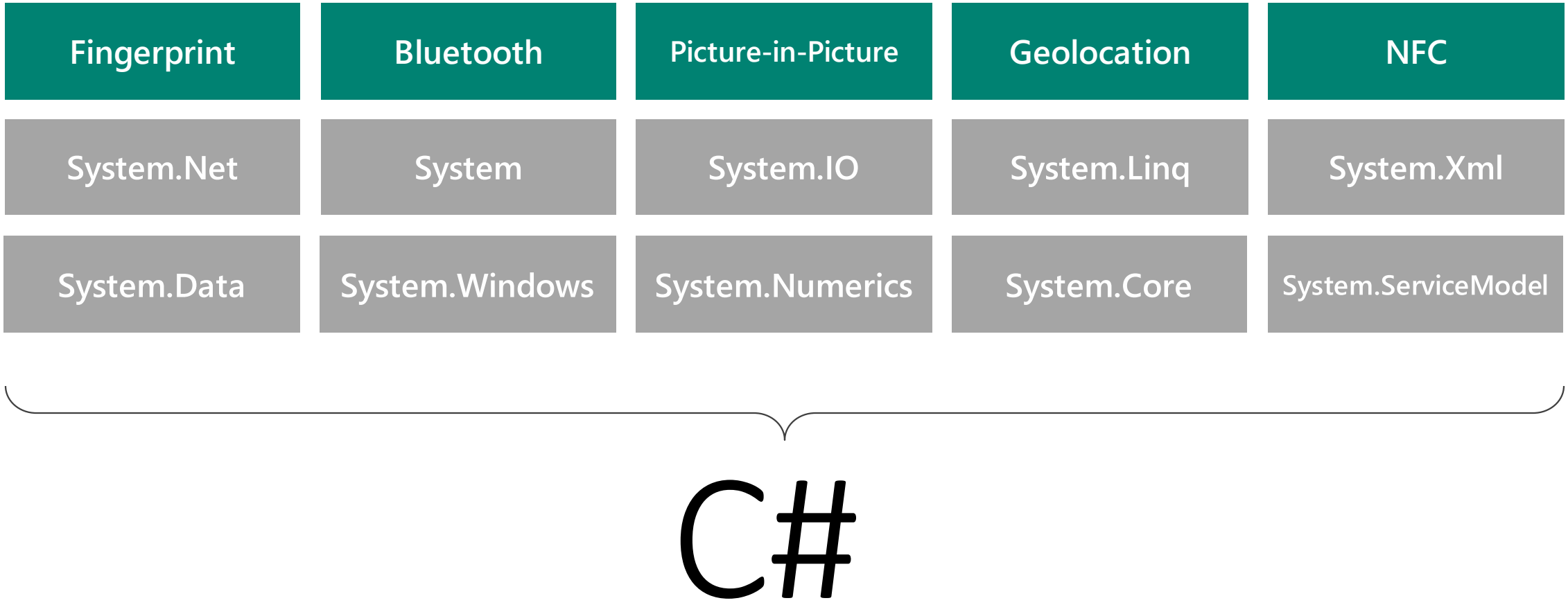



```
AppActions.SetAsync(
    new AppAction("current_info", "Check Current Weather", icon: "current_info"),
    new AppAction("add_location", "Add a Location", icon: "add_location")
    );
```

iOS


MapKit
UIKit
SiriKit
ARKit
CoreML
System.Net
System
System.IO
System.Linq
System.Xml
System.Data
System.Windows
System.Numerics
System.Core
System.ServiceModel
C#

Android


Fingerprint
Bluetooth
Picture-in-Picture
Geolocation
NFC
System.Net
System
System.IO
System.Linq
System.Xml
System.Data
System.Windows
System.Numerics
System.Core
System.ServiceModel
C#

Cross-Platform APIs

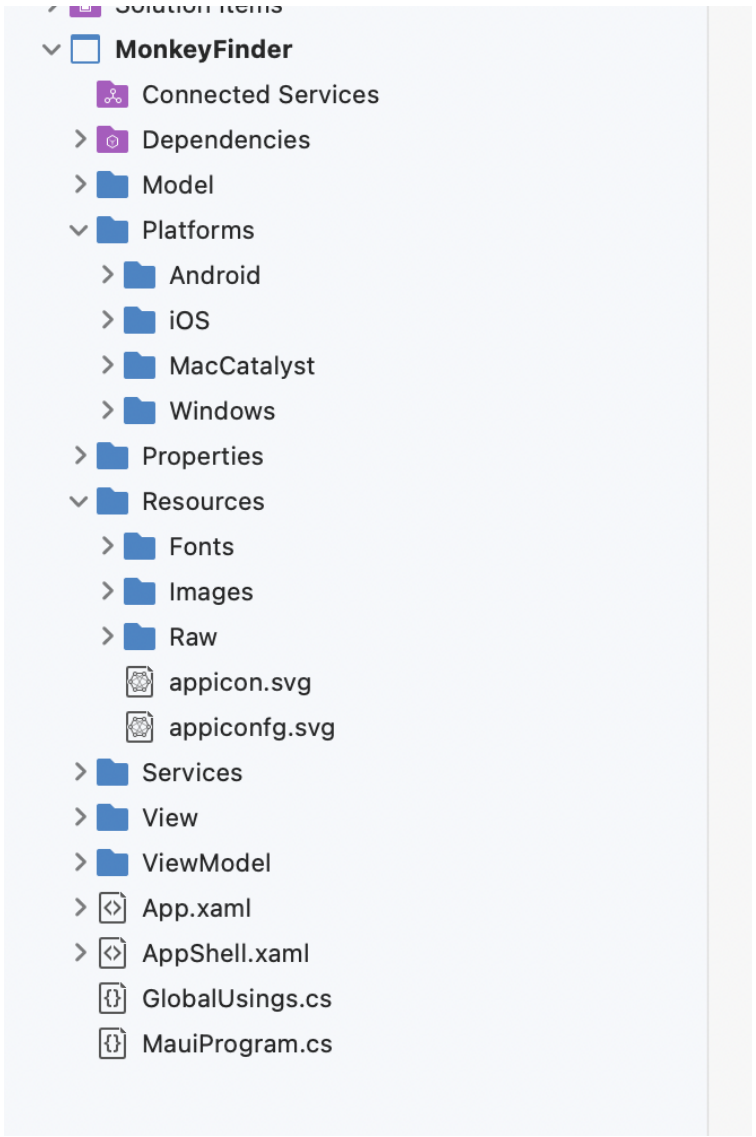
Flashlight
Secure Settings
Clipboard
Geolocation
Phone Dialer
Email
Device Info
Accelerometer
Data Transfer
Geocoding
Vibration
File System
Connectivity
Screen Lock
Battery
Preferences
Sms
Gyroscope
Magnetometer
Device Display Info
Compass
Text to Speech

Building Elements

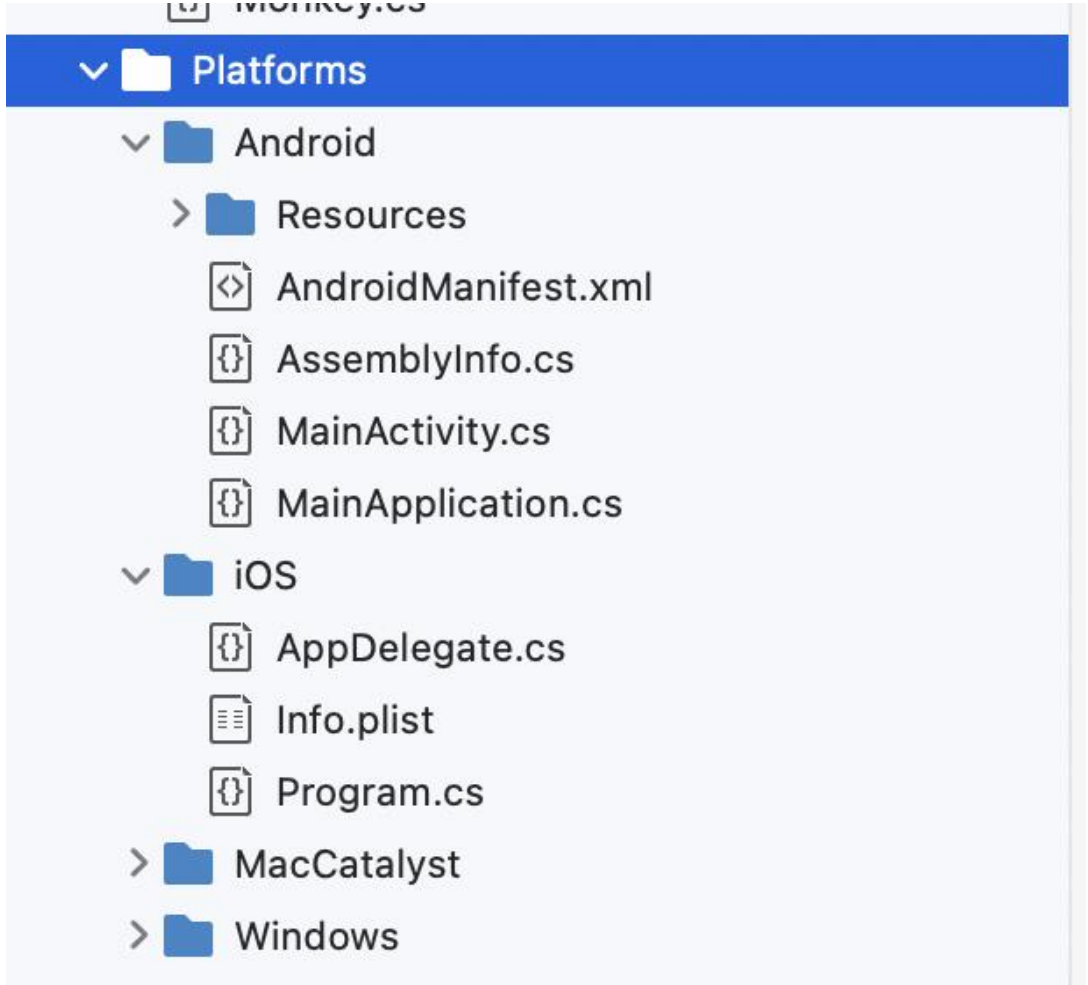
Startup class, registration, fonts ...

```
1 reference
public static class MauiProgram
{
    1 reference
    public static MauiApp CreateMauiApp()
    {
        var builder = MauiApp.CreateBuilder();
        builder
            .UseMauiApp<App>()
            .ConfigureFonts(fonts =>
            {
                fonts.AddFont("OpenSans-Regular.ttf", "OpenSansRegular");
            });

        builder.Services.AddSingleton<IConnectivity>(Connectivity.Current);
        builder.Services.AddSingleton<IGeolocation>(Geolocation.Default);
        builder.Services.AddSingleton<IMap>(Map.Default);

        builder.Services.AddSingleton<MonkeyService>();
        builder.Services.AddSingleton<MonkeysViewModel>();
        builder.Services.AddSingleton<MainPage>();

        builder.Services.AddTransient<MonkeyDetailsViewModel>();
        builder.Services.AddTransient<DetailsPage>();

        return builder.Build();
    }
}
```

Platforms and Resources "folder"



Platforms and Resources "folder"



Resolving dependencies

```
IMap map;
public MonkeyDetailsViewModel(IMap map)
{
    this.map = map;
}
```

Community Toolkit Integration

```
[ObservableProperty]
Monkey monkey;

[RelayC
async T
{
    try
    {

    }
    cat
    {

    }
}
```

class CommunityToolkit.Mvvm.ComponentModel.ObservablePropertyAttribute (+ 1 overload)

An attribute that indicates that a given field should be wrapped by a generated observable property. In order to use this attribute, the containing type has to inherit from ObservableObject, or it must be using ObservableObjectAttribute or INotifyPropertyChangedAttribute. If the containing type also implements the INotifyPropertyChanging (that is, if it either inherits from ObservableObject or is using ObservableObjectAttribute), then the generated code will also invoke ObservableObject.OnPropertyChanging(System.ComponentModel.PropertyChangingEventArgs) to signal that event.

This attribute can be used as follows:

```
partial class MyViewModel : ObservableObject
{
    [ObservableProperty]
    private string name;

    [ObservableProperty]
    private bool isEnabled;
}
```

And with this, code analogous to this will be generated:

```
partial class MyViewModel
{
    public string Name
    {
        get => name;
        set => SetProperty(ref name, value);
    }

    public bool IsEnabled
    {
        get => isEnabled;
        set => SetProperty(ref isEnabled, value);
    }
}
```

Source generators

Navigation and passing the params

```
[RelayCommand]
async Task GoToDetails(Monkey monkey)
{
    if (monkey == null)
    return;

    await Shell.Current.GoToAsync(nameof(DetailsPage), true, new Dictionary<string, object>
    {
        {"Monkey", monkey }
    });
}
```

```
[QueryProperty(nameof(Monkey), "Monkey")]
public partial class MonkeyDetailsViewModel : BaseViewModel
{
    IMap map;
```

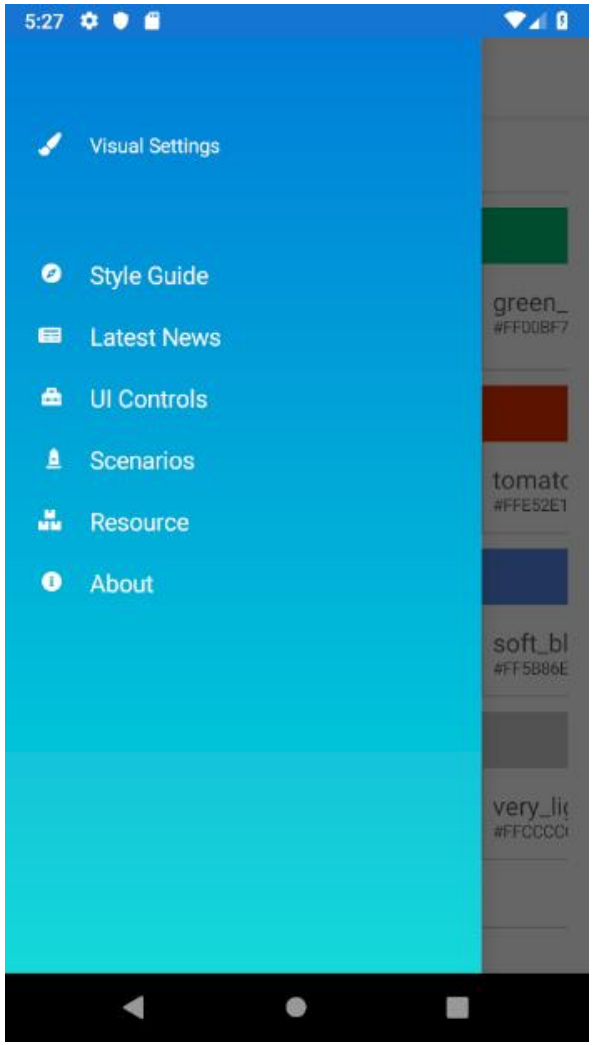
MVVM and XAML are still here

```
<Button
    Grid.Row="1"
    Grid.Column="0"
    Margin="8"
    Command="{Binding GetMonkeysCommand}"
    IsEnabled="{Binding IsNotBusy}"
    Style="{StaticResource ButtonOutline}"
    Text="Get Monkeys" />
```

Writing styles is same

```xml
<?xml version="1.0" encoding="utf-8" ?>
<Application
    x:Class="MonkeyFinder.App"
    xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    xmlns:local="clr-namespace:MonkeyFinder">
    <Application.Resources>
        <ResourceDictionary>
            <Color x:Key="Primary">■ #FFC107</Color>
            <Color x:Key="PrimaryDark">■ #FFA000</Color>
            <Color x:Key="Accent">■ #00BCD4</Color>

            <Color x:Key="LightBackground">□ #FAF9F8</Color>
            <Color x:Key="DarkBackground">■ Black</Color>

            <Color x:Key="CardBackground">□ White</Color>
            <Color x:Key="CardBackgroundDark">■ #1C1C1E</Color>

            <Color x:Key="LabelText">■ #1F1F1F</Color>
            <Color x:Key="LabelTextDark">□ White</Color>

            <Style ApplyToDerivedTypes="True" TargetType="Page">
                <Setter Property="BackgroundColor" Value="{AppThemeBinding Light={StaticResource LightBackgro
            </Style>

            <Style ApplyToDerivedTypes="True" TargetType="NavigationPage">
                <Setter Property="BackgroundColor" Value="{AppThemeBinding Light={StaticResource LightBackgro
                <Setter Property="BarBackgroundColor" Value=■ "{StaticResource Primary}" />
                <Setter Property="BarTextColor" Value=□ "White" />
            </Style>

            <Style TargetType="Label" x:Key="BaseLabel">
                <Setter Property="FontFamily" Value="OpenSansRegular" />
                <Setter Property="TextColor" Value="{AppThemeBinding Light={StaticResource LabelText}, Dark={
            </Style>
```

Shell




```
<Shell>
    <FlyoutItem
        Title="Style Guide"
        Icon="Compass.png">
        <ShellContent
            ContentTemplate="{DataTemplate p:StyleGuidePage}"/>
    </FlyoutItem>
</Shell>
```

.NET MAUI Application

Application lifecycle:

- **OnStart**
- **OnSleep**
- **OnResume**

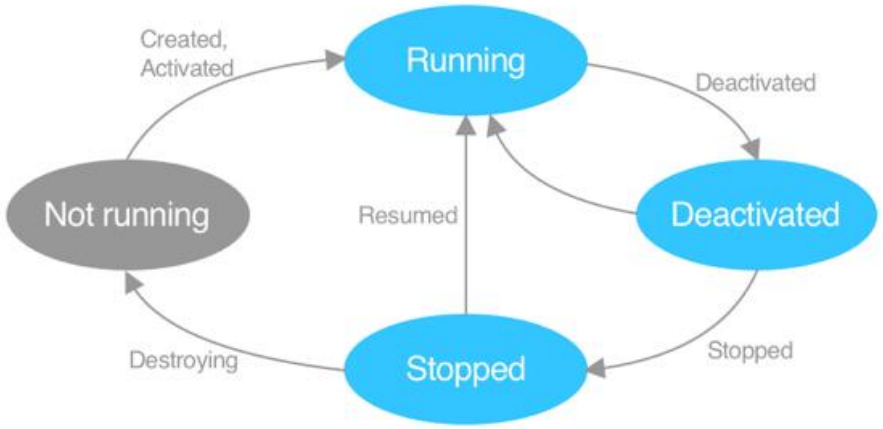
```
public partial class App : Application
{
    0 references
    public App()
    {
        InitializeComponent();

        MainPage = new AppShell();
    }

    0 references
    protected override void OnStart()
    {
        base.OnStart();
    }

    0 references
    protected override void OnSleep()
    {
        base.OnSleep();
    }

    0 references
    protected override void OnResume()
    {
        base.OnResume();
    }
}
```

.NET MAUI Application Lifecycle

Windows lifecycle:

- **Created**
- **Activated**
- **Deactivated**
- **Stopped**
- **Resumed**
- **Destroying**

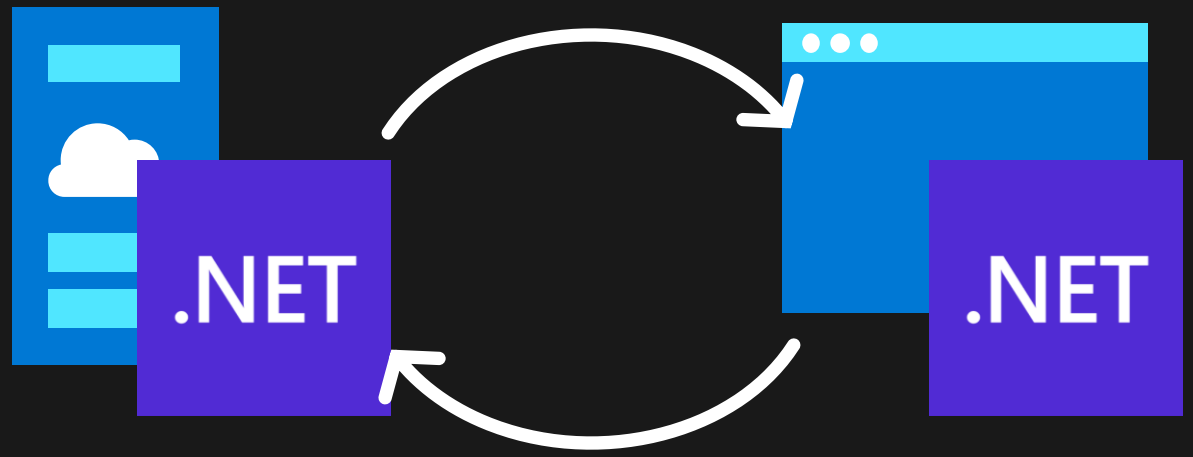
```
protected override Window CreateWindow(IActivationState activationState)
{
    Window window = base.CreateWindow(activationState);

    window.Created += (s, e) =>
    {
        // Custom logic
    };

    window.Activated += (s, e) =>
    {
        // Custom logic
    };

    return window;
}
```



Hybrid

Bring native client capabilities to your web apps.

Blazor – full stack web apps with .NET



Blazor hybrid apps

Reuse UI components across native and web

Native app container & embedded controls

Built on top of .NET Multi-platform App UI

What's new? 

.NET MAUI Extension

- Debug your .NET MAUI app on different devices, emulators, and your local machine
- Optional extension for C# Dev Kit
- Licensed under Visual Studio subscription (same as C# Dev Kit)

+

C# Dev Kit Extension

- Solution Explorer
- Native Test Explorer
- Licensed under Visual Studio subscription

C# Extension

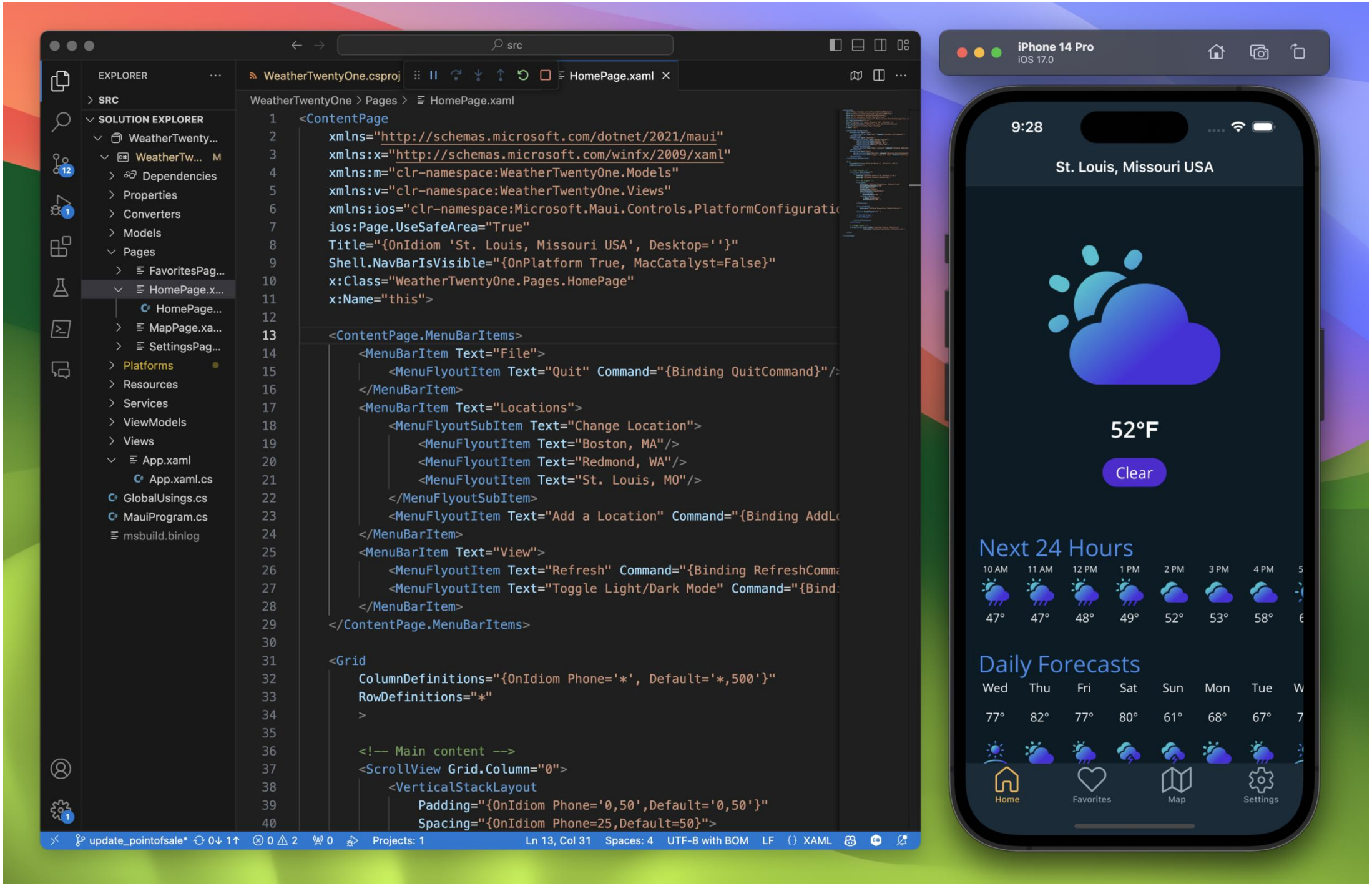
- Fully Open-Source (excluding debugger)
- Evolution from OmniSharp
- Rich C# editing experience
- No change to licensing: MIT Licensing

Visual Studio Code

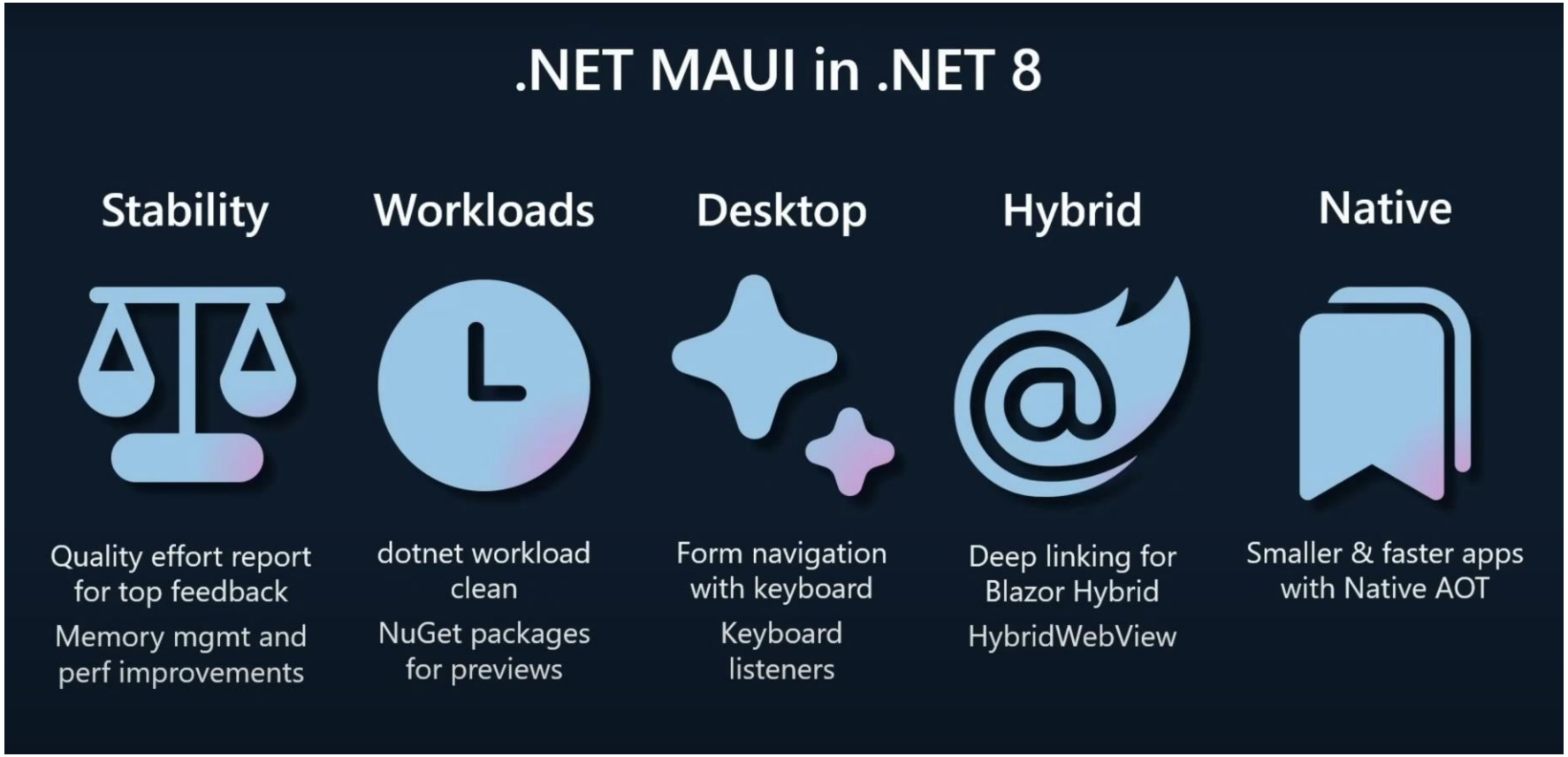
.NET 8.0

.NET Conf 2023

The largest .NET event hosted online
November 14 - 16.


src
EXPLORER
SRC
SOLUTION EXPLORER
WeatherTwenty...
WeatherTw... M
Dependencies
Properties
Converters
Models
Pages
FavoritesPag...
HomePage.x...
HomePage...
MapPage.xa...
SettingsPag...
Platforms
Resources
Services
ViewModels
Views
App.xaml
App.xaml.cs
GlobalUsings.cs
MauiProgram.cs
msbuild.binlog
WeatherTwentyOne.csproj
HomePage.xaml
WeatherTwentyOne > Pages > HomePage.xaml
<ContentPage
xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
xmlns:m="clr-namespace:WeatherTwentyOne.Models"
xmlns:v="clr-namespace:WeatherTwentyOne.Views"
xmlns:ios="clr-namespace:Microsoft.Maui.Controls.PlatformConfigurati
ios:Page.UseSafeArea="True"
Title="{OnIdiom 'St. Louis, Missouri USA', Desktop=''}"
Shell.NavBarIsVisible="{OnPlatform True, MacCatalyst=False}"
x:Class="WeatherTwentyOne.Pages.HomePage"
x:Name="this">
<ContentPage.MenuBarItems>
<MenuBarItem Text="File">
<MenuFlyoutItem Text="Quit" Command="{Binding QuitCommand}"/>
</MenuBarItem>
<MenuBarItem Text="Locations">
<MenuFlyoutSubItem Text="Change Location">
<MenuFlyoutItem Text="Boston, MA"/>
<MenuFlyoutItem Text="Redmond, WA"/>
<MenuFlyoutItem Text="St. Louis, MO"/>
</MenuFlyoutSubItem>
<MenuFlyoutItem Text="Add a Location" Command="{Binding AddL
</MenuBarItem>
<MenuBarItem Text="View">
<MenuFlyoutItem Text="Refresh" Command="{Binding RefreshComma
<MenuFlyoutItem Text="Toggle Light/Dark Mode" Command="{Bind
</MenuBarItem>
</ContentPage.MenuBarItems>
<Grid
ColumnDefinitions="{OnIdiom Phone='*', Default='*,500'}"
RowDefinitions="*"
>
<!-- Main content -->
<ScrollView Grid.Column="0">
<VerticalStackLayout
Padding="{OnIdiom Phone='0,50',Default='0,50'}"
Spacing="{OnIdiom Phone=25,Default=50}">
update_pointofsale* 0↓ 1↑
0 2
0
Projects: 1
Ln 13, Col 31
Spaces: 4
UTF-8 with BOM
LF
XAML
iPhone 14 Pro
iOS 17.0
9:28
St. Louis, Missouri USA
52°F
Clear
Next 24 Hours
10 AM 11 AM 12 PM 1 PM 2 PM 3 PM 4 PM
47° 47° 48° 49° 52° 53° 58°
Daily Forecasts
Wed Thu Fri Sat Sun Mon Tue
77° 82° 77° 80° 61° 68° 67°
Home
Favorites
Map
Settings

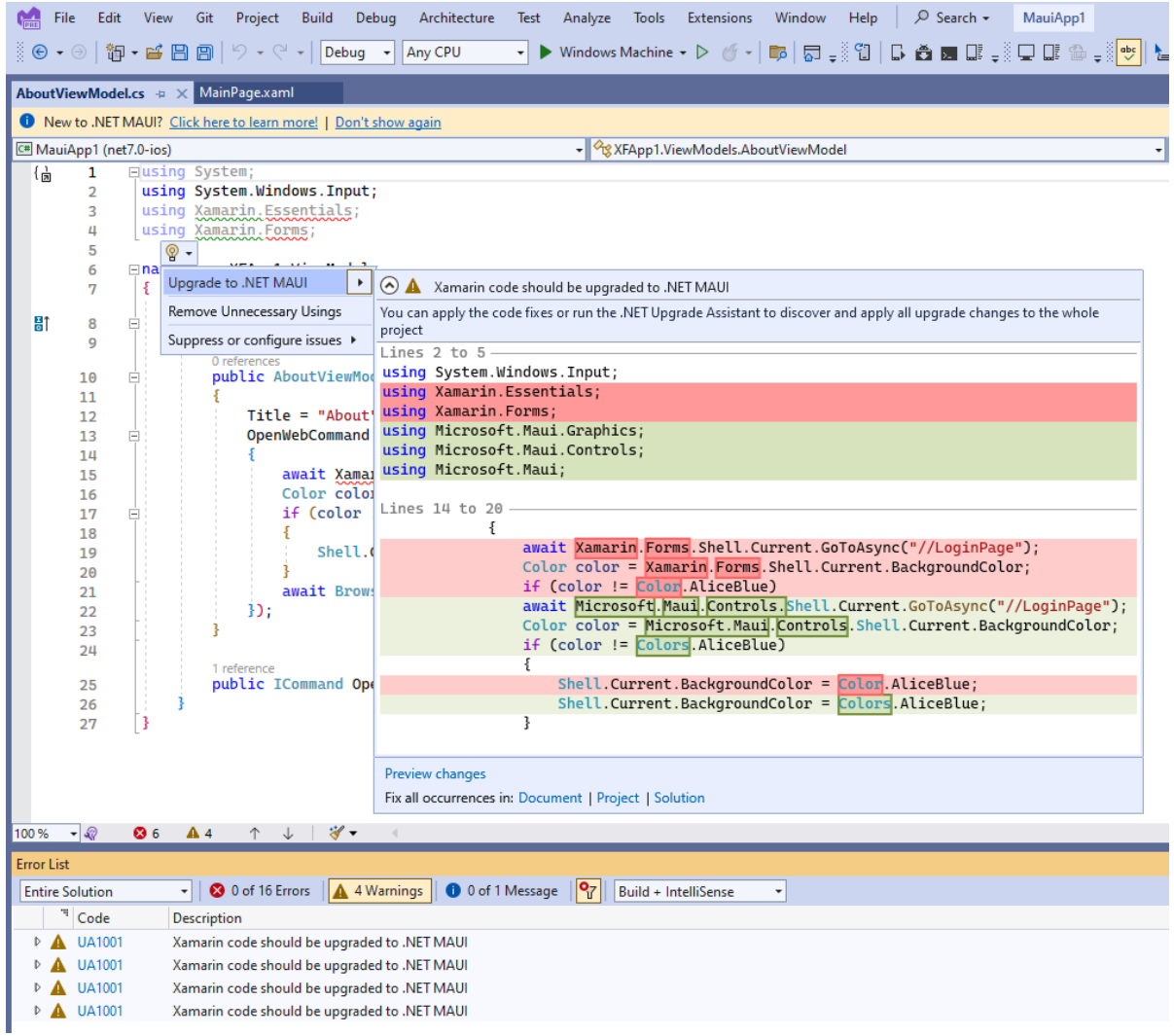
What's new?


.NET MAUI in .NET 8
Stability
Workloads
Desktop
Hybrid
Native
Quality effort report for top feedback
Memory mgmt and perf improvements
dotnet workload clean
NuGet packages for previews
Form navigation with keyboard
Keyboard listeners
Deep linking for Blazor Hybrid
HybridWebView
Smaller & faster apps with Native AOT

Migration?

https://learn.microsoft.com/en-us/dotnet/maui/migration/upgrade-assistant

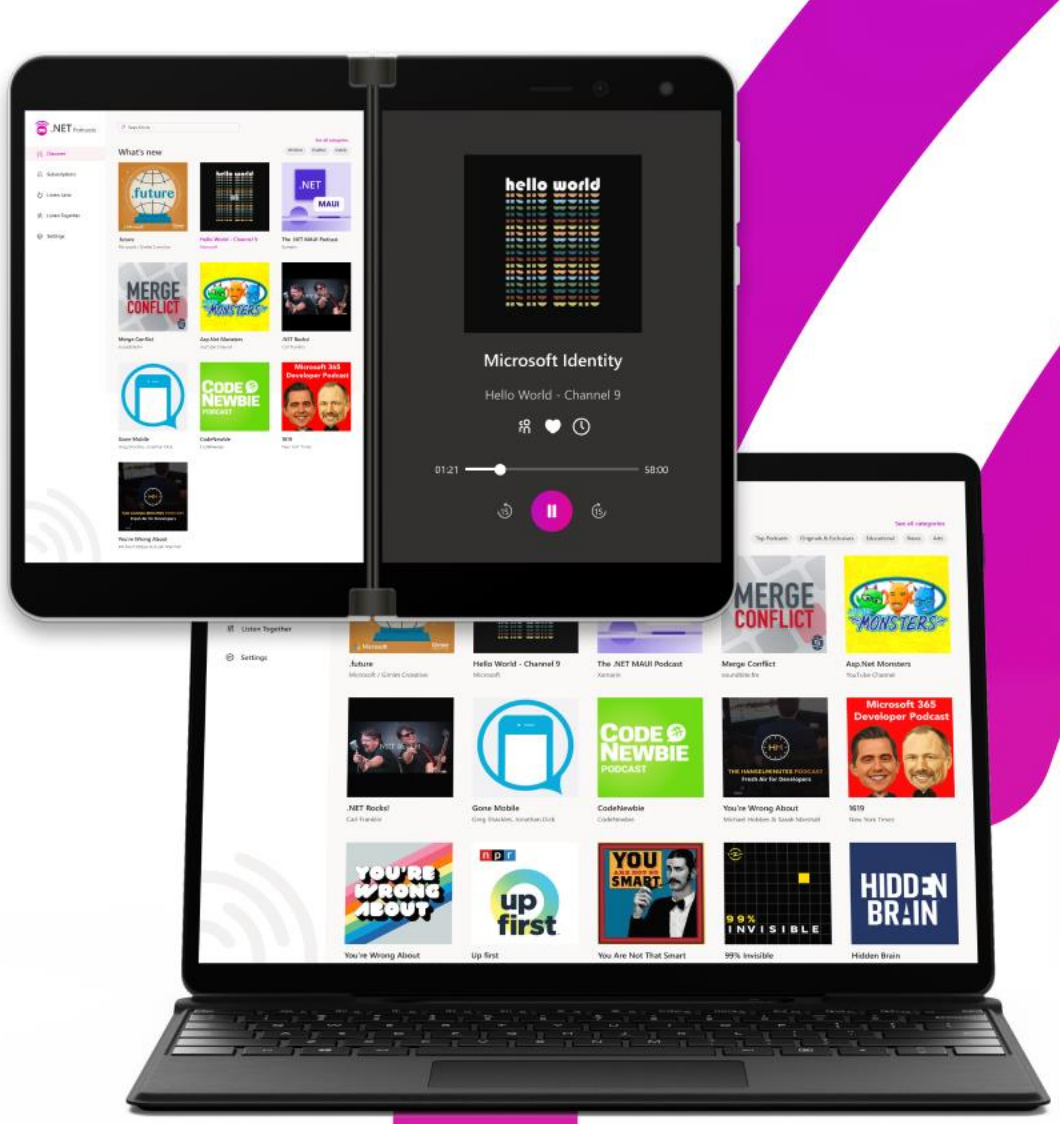
.NET Upgrade Assistant

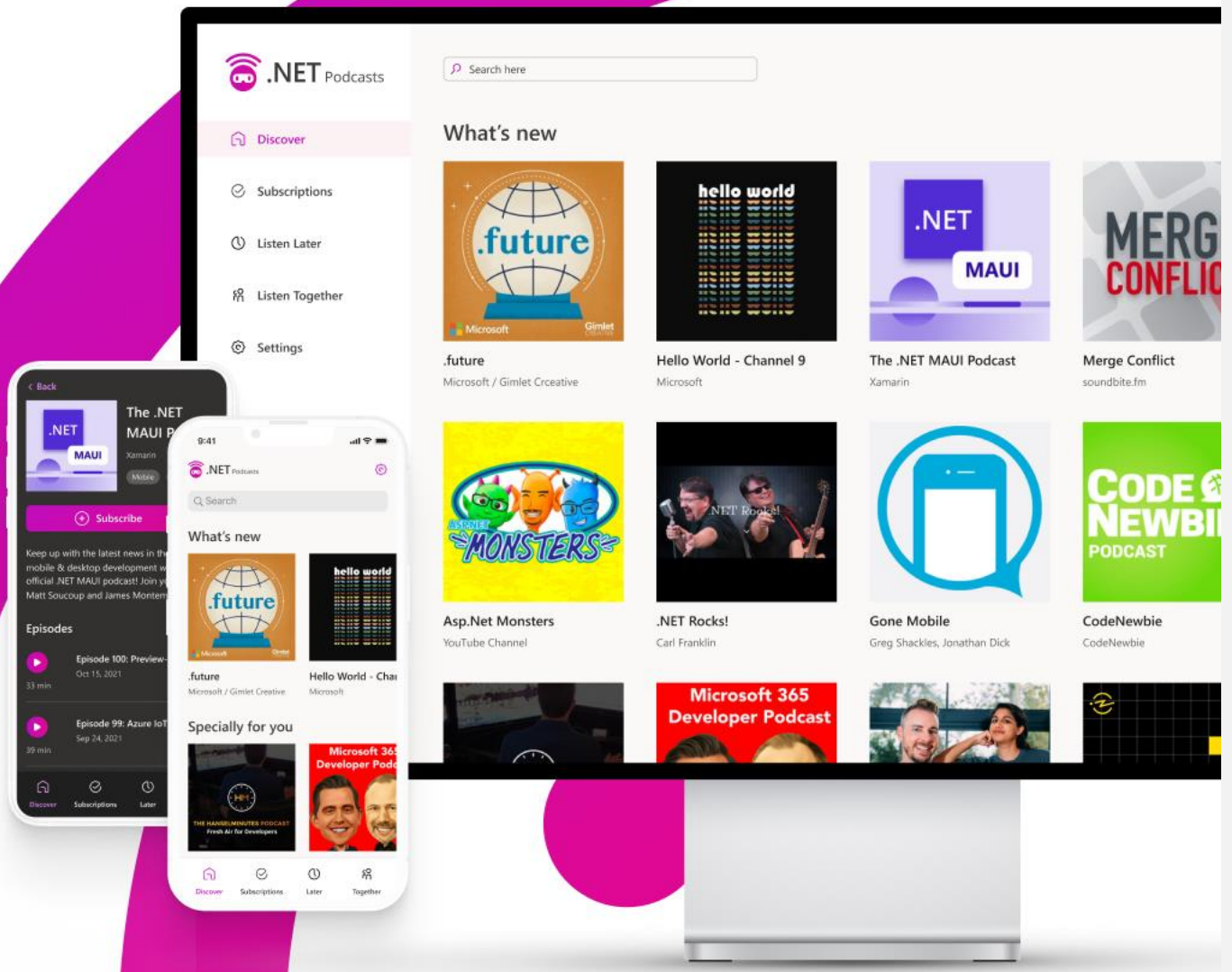


Homework 

https://dotnet.microsoft.com/en-us/learn/maui/first-app-tutorial/intro

.NET Podcasts





https://github.com/microsoft/dotnet-podcasts/tree/main/src/Mobile

Summary

- .NET MAUI is a cross-platform, write once, run anywhere (WORA) UI application platform. You can build just one .NET MAUI app and it will run on multiple platforms without further modification.
- You can write native apps with .NET MAUI.

- Same as with XF at the end of the day .NET MAUI apps are native apps.
- You can build apps in .NET MAUI that have functional, performance and security advantages over web apps.
- You can use the entire .NET ecosystem to build .NET MAUI apps. This includes all your favourite NuGet packages, as well as your existing skills as a .NET developer.
- You can write .NET MAUI app UIs in XAML, C#, F# or Blazor

Thank you!

ευχαριστώ Salamat Po متشكرم شكراً Grazie

благодаря ありがとうございます Kiitos Teşekkürler 谢谢

ขอบคุณครับ Obrigado شكريه Terima Kasih Dziękuję

Hvala Köszönöm Tak Dank u wel дякую Tack

Mulţumesc спасибо Danke Cám ơn Gracias

多謝晒 Ďakujem תודה நன்றி Děkuji 감사합니다

Almir Vuk

almirvuk@outlook.com
almirvuk.com | @almirvuk

Thank you!

ευχαριστώ Salamat Po متشكرم شكراً Grazie

благодаря ありがとうございます Kiitos Teşekkürler 谢谢

ขอบคุณครับ Obrigado شكريه Terima Kasih Dziękuję