Desktop Apps Your Way

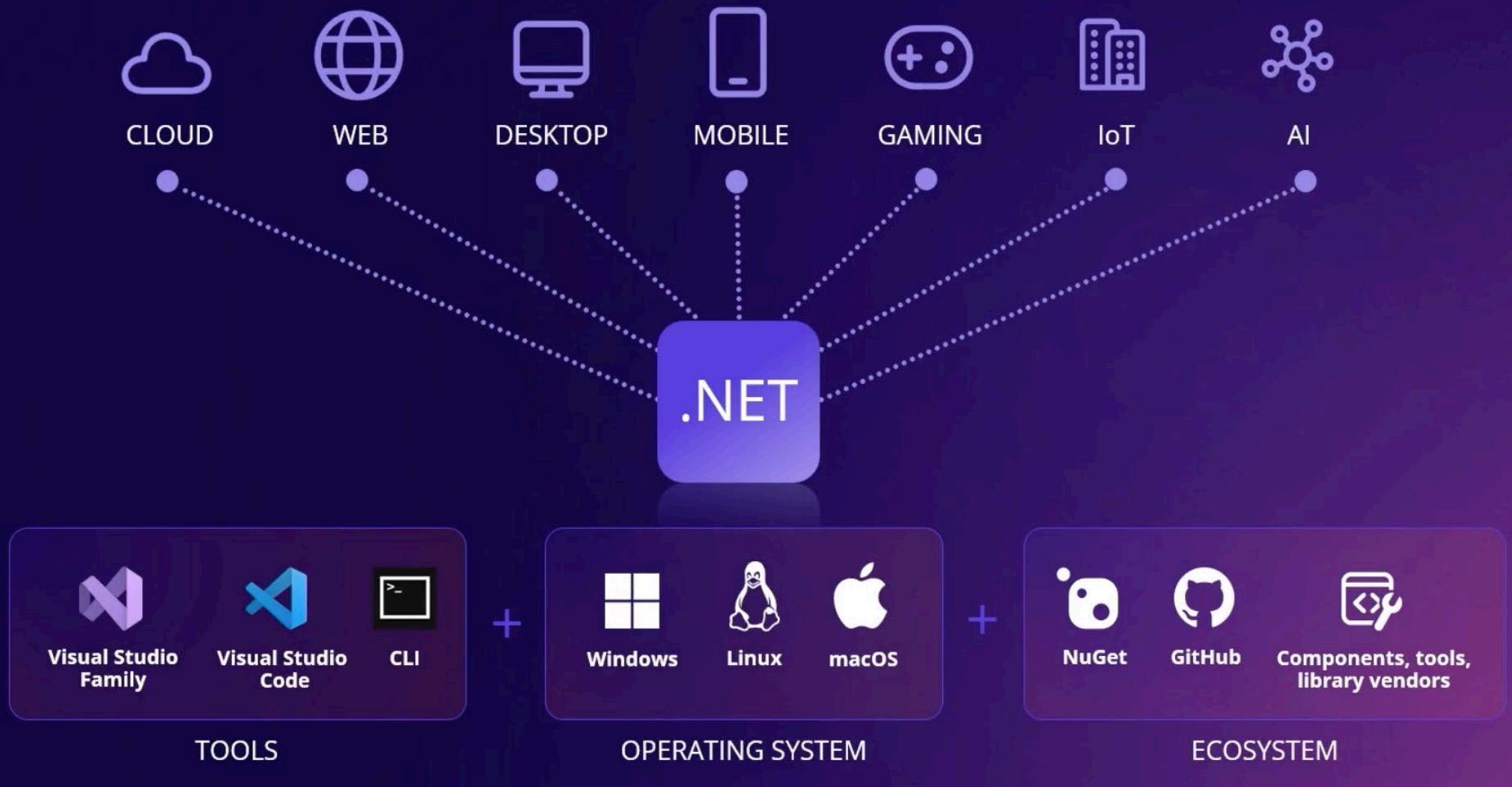
Sam Basu | @samidip
Developer Advocate | Progress

Perhaps not the coolest kid …

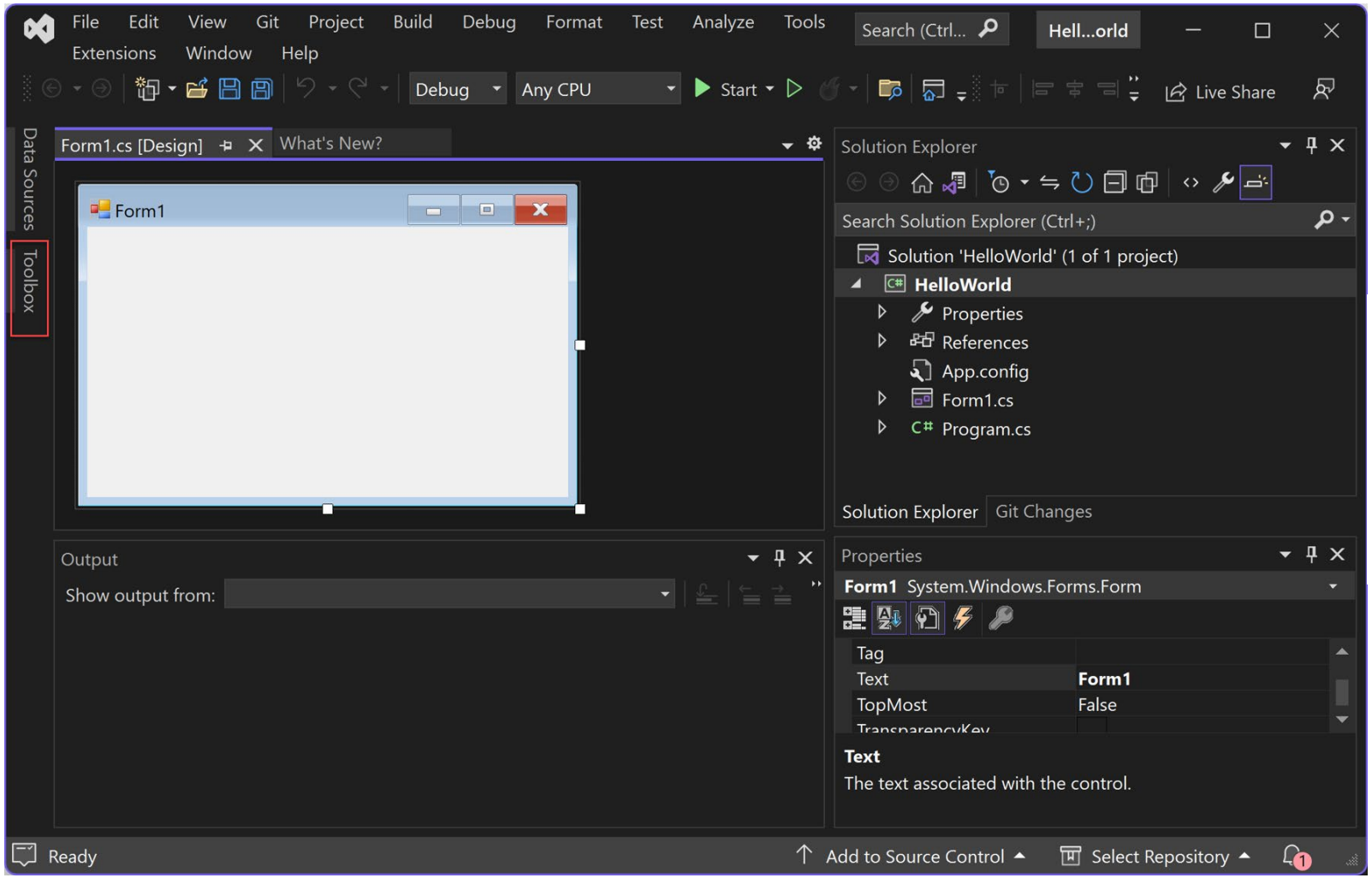
But Desktop runs businesses

Building for Windows

Build anything with a unified platform

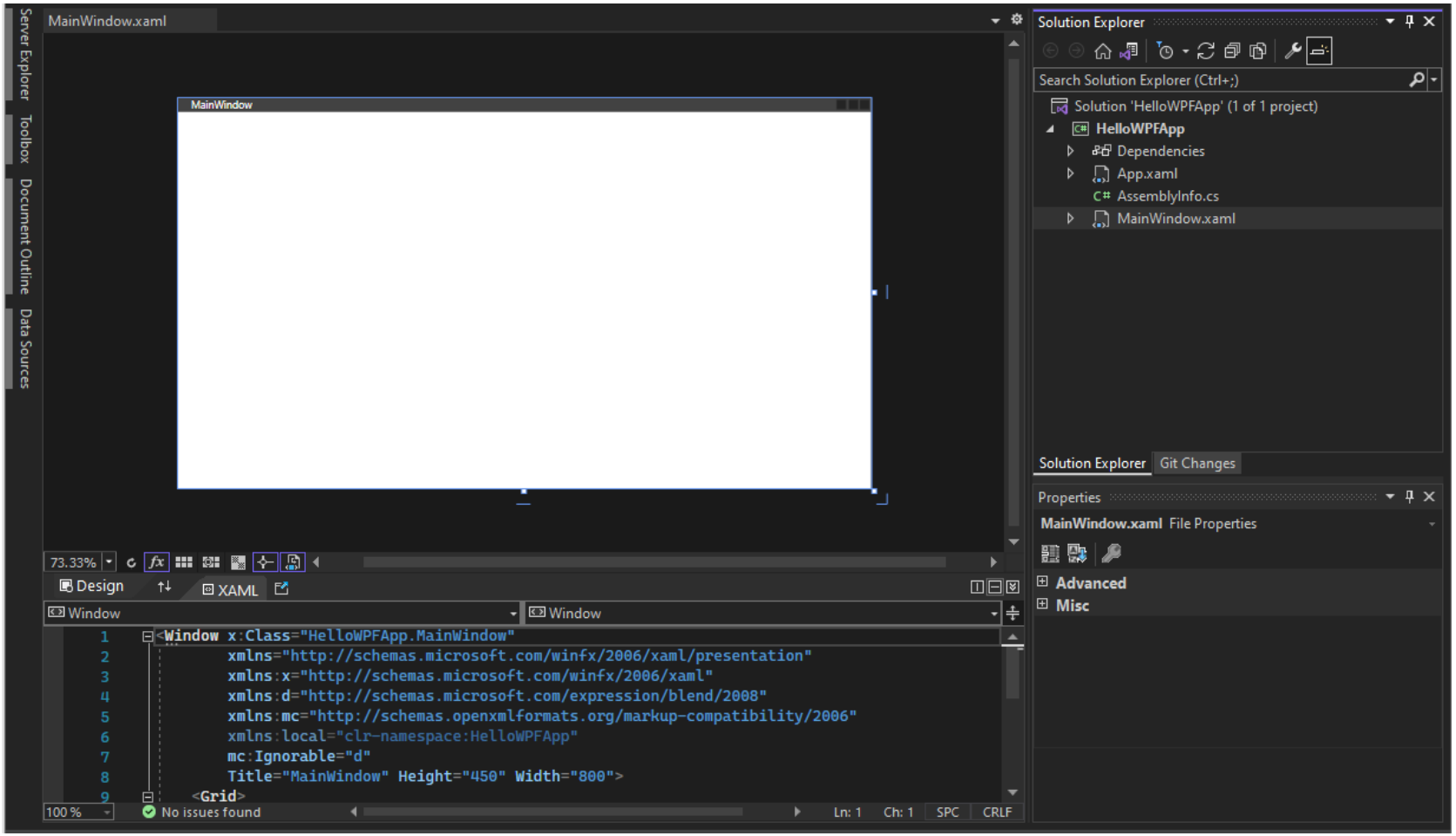


Windows Forms (WinForms)



Microsoft® .NET
Windows Forms

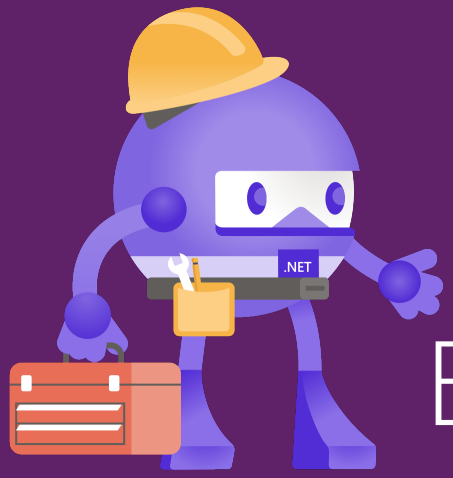
Windows Presentation Foundation (WPF)



Windows Presentation Foundation

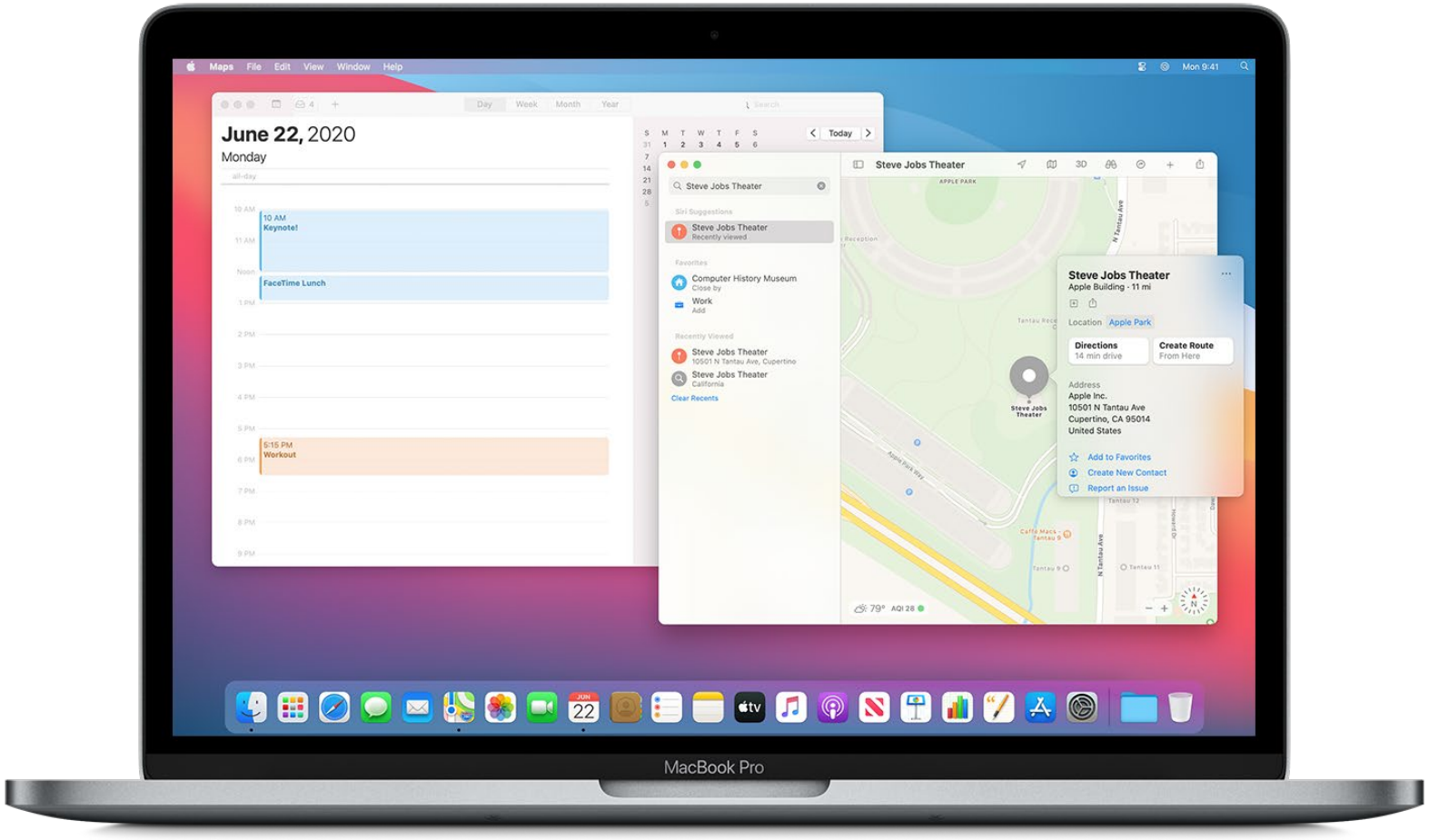
Universal Windows Platform (UWP)

Windows UI 3 (WinUI)



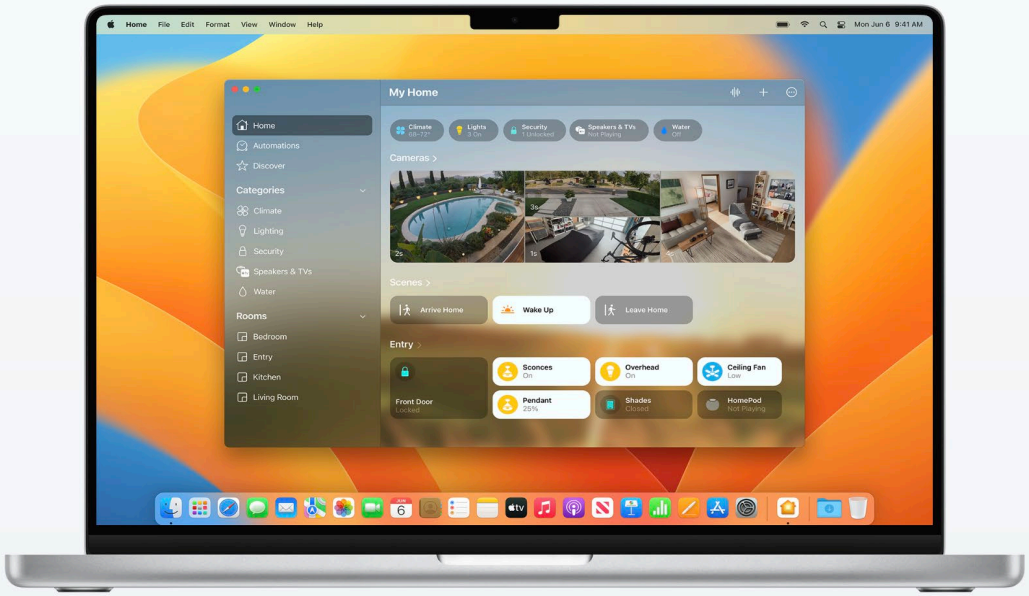
Building for macOS/Linux

AppKit + Xcode + SwiftUI



Mac Catalyst

Mac apps built with Mac Catalyst share code with your iPad apps, and you can add more features just for Mac. The latest desktop-class features in iPadOS 16 translate beautifully onto macOS 13. Your Mac app's toolbars are automatically optimized and document-based apps gain key features, such as menu items and the ability to rename the document in the toolbar. And you can use new Mac Catalyst APIs to enhance multiwindow behaviors, add custom views to your toolbars, and more.



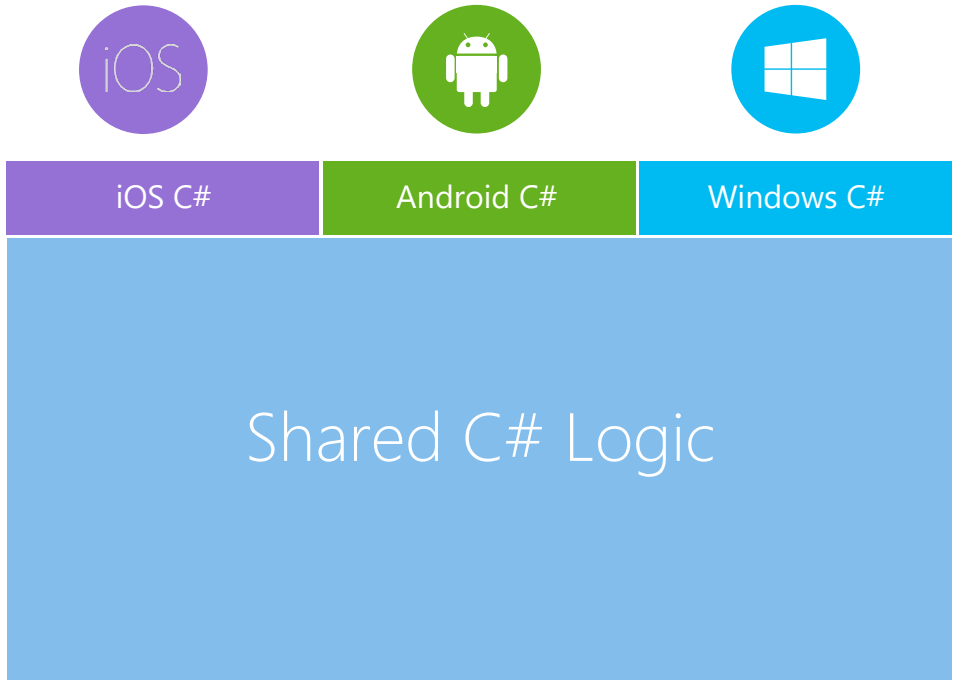
Bring your iPad app to Mac

Get a head start on your Mac app

Your iPad app can transform into an amazing Mac app using the latest version of Xcode. Begin by adding a Mac Catalyst run destination in the project settings of your existing iPad app to create a Mac app that you can enhance further. Your Mac and iPad apps share the same project and source code, so it's easy to make changes in one place.

The Outliers

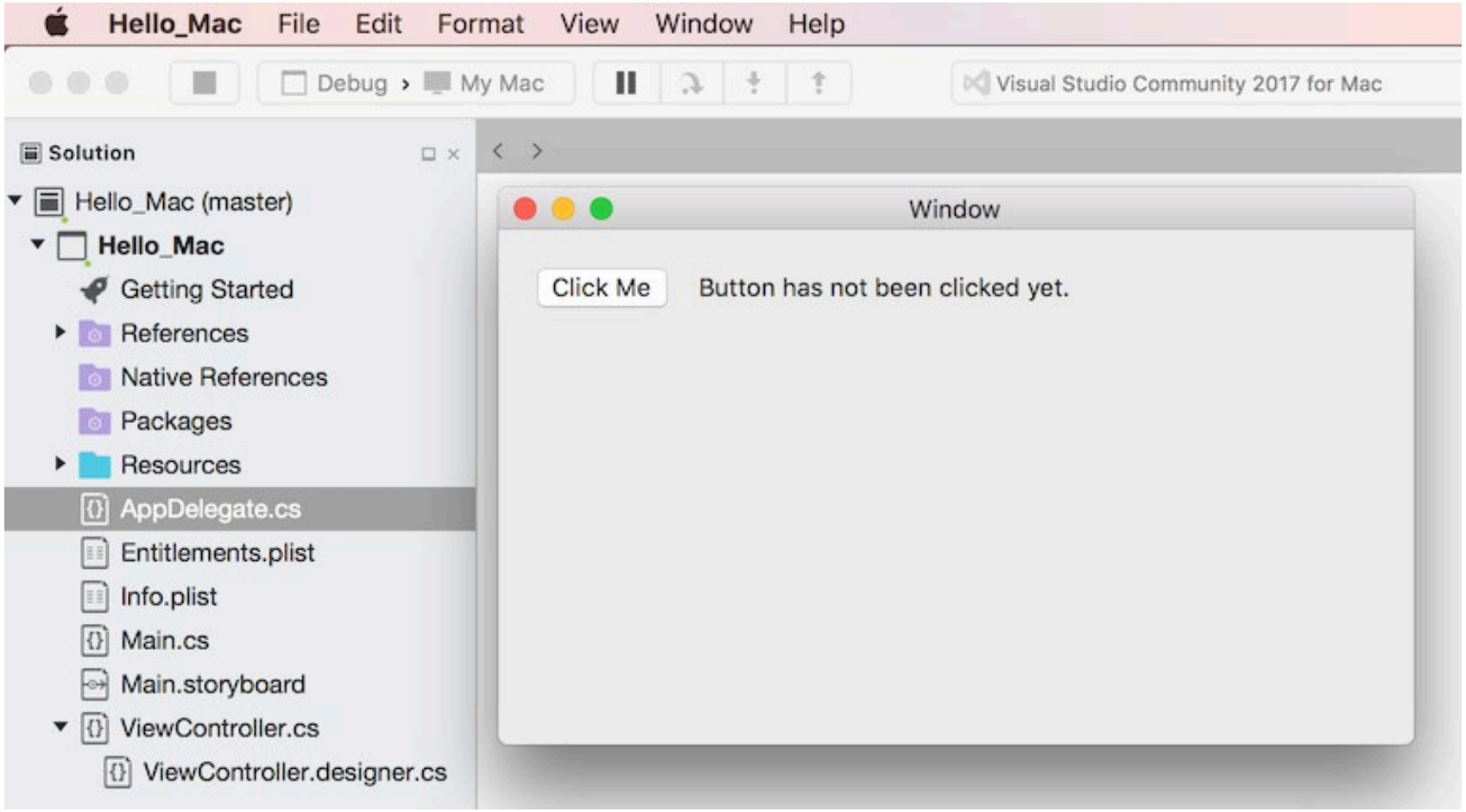
Traditional UI approach



3 Native User Interfaces

Shared App Logic

Xamarin.Mac


Hello_Mac File Edit Format View Window Help
Debug › My Mac
Visual Studio Community 2017 for Mac
Solution
Hello_Mac (master)
Hello_Mac
Getting Started
References
Native References
Packages
Resources
AppDelegate.cs
Entitlements.plist
Info.plist
Main.cs
Main.storyboard
ViewController.cs
ViewController.designer.cs
Window
Click Me
Button has not been clicked yet.

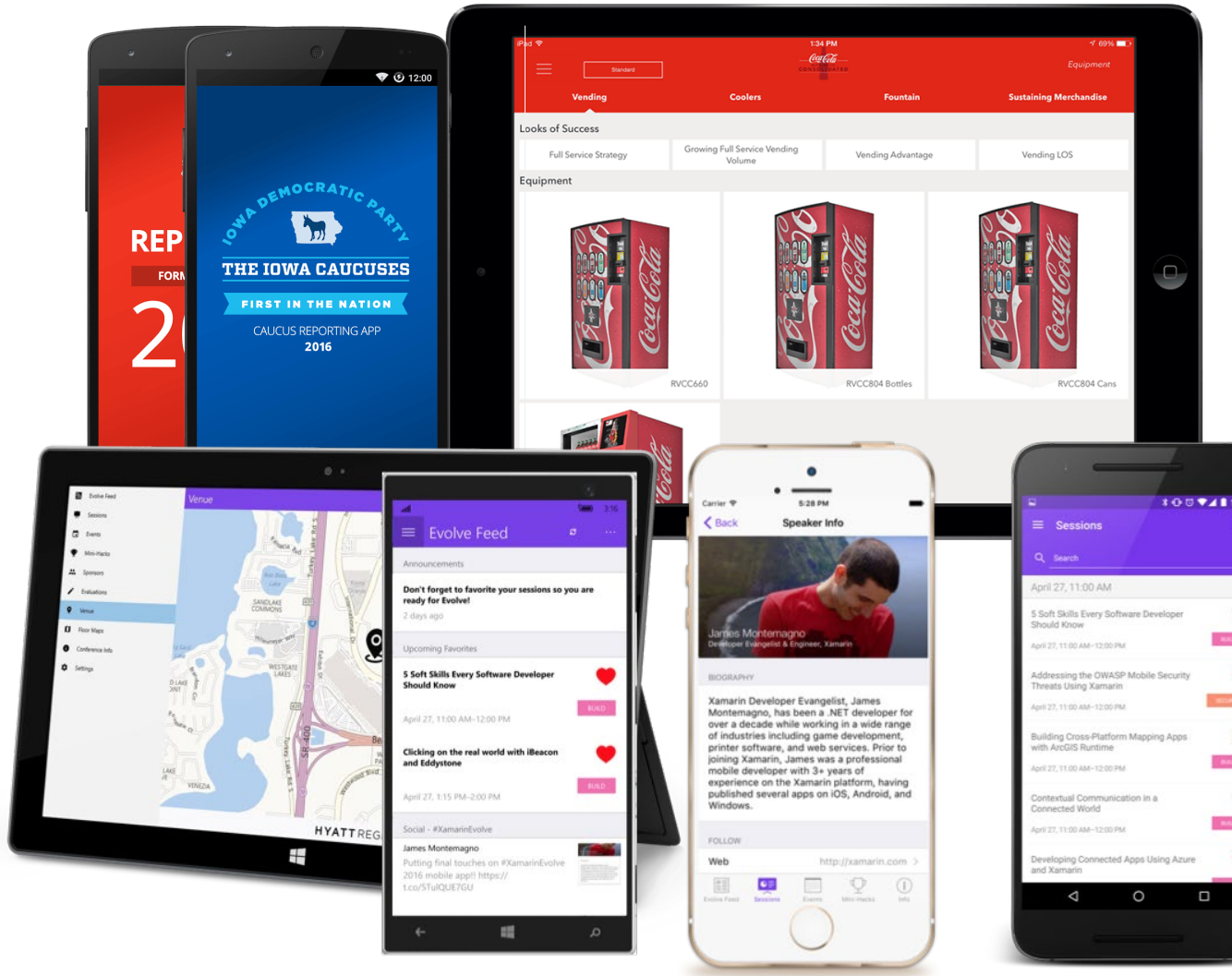
Xamarin.Forms approach

Shared UI Code

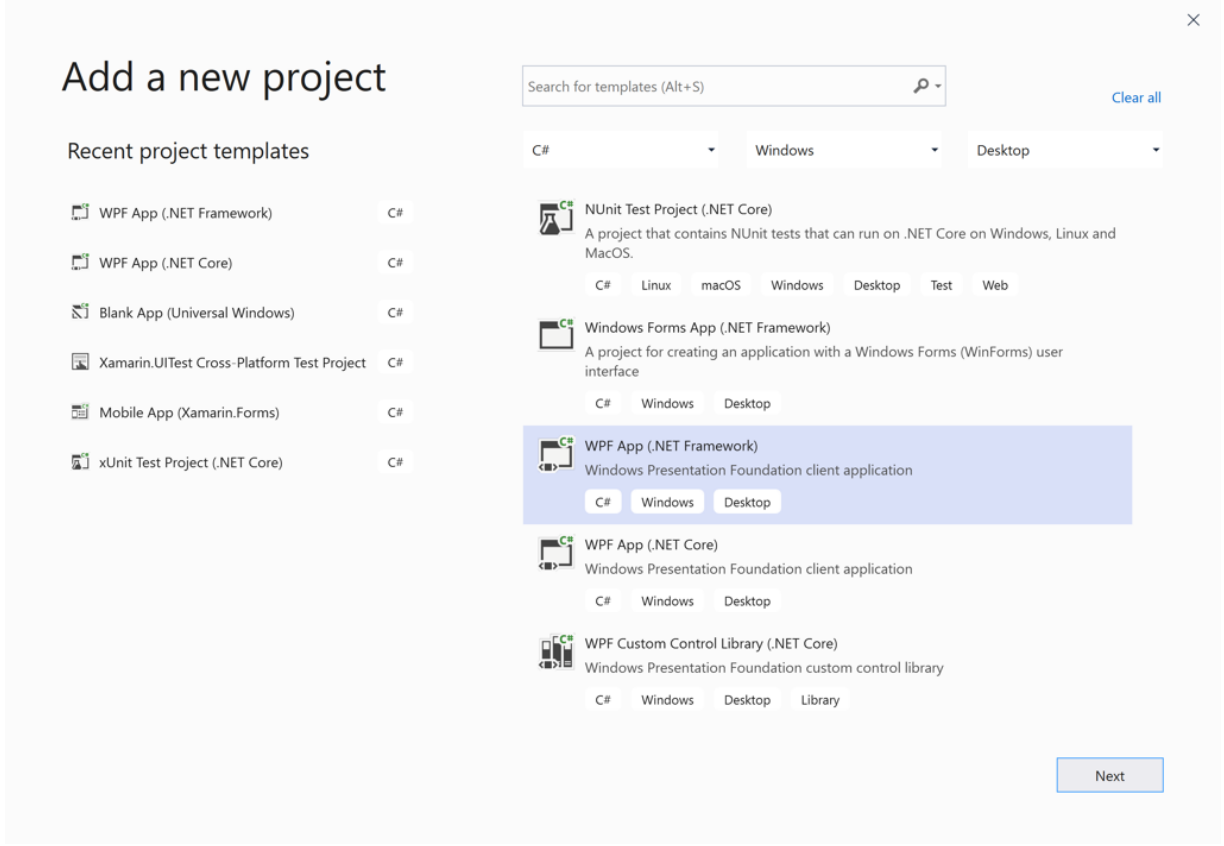
Shared C# Logic

Shared User Interface

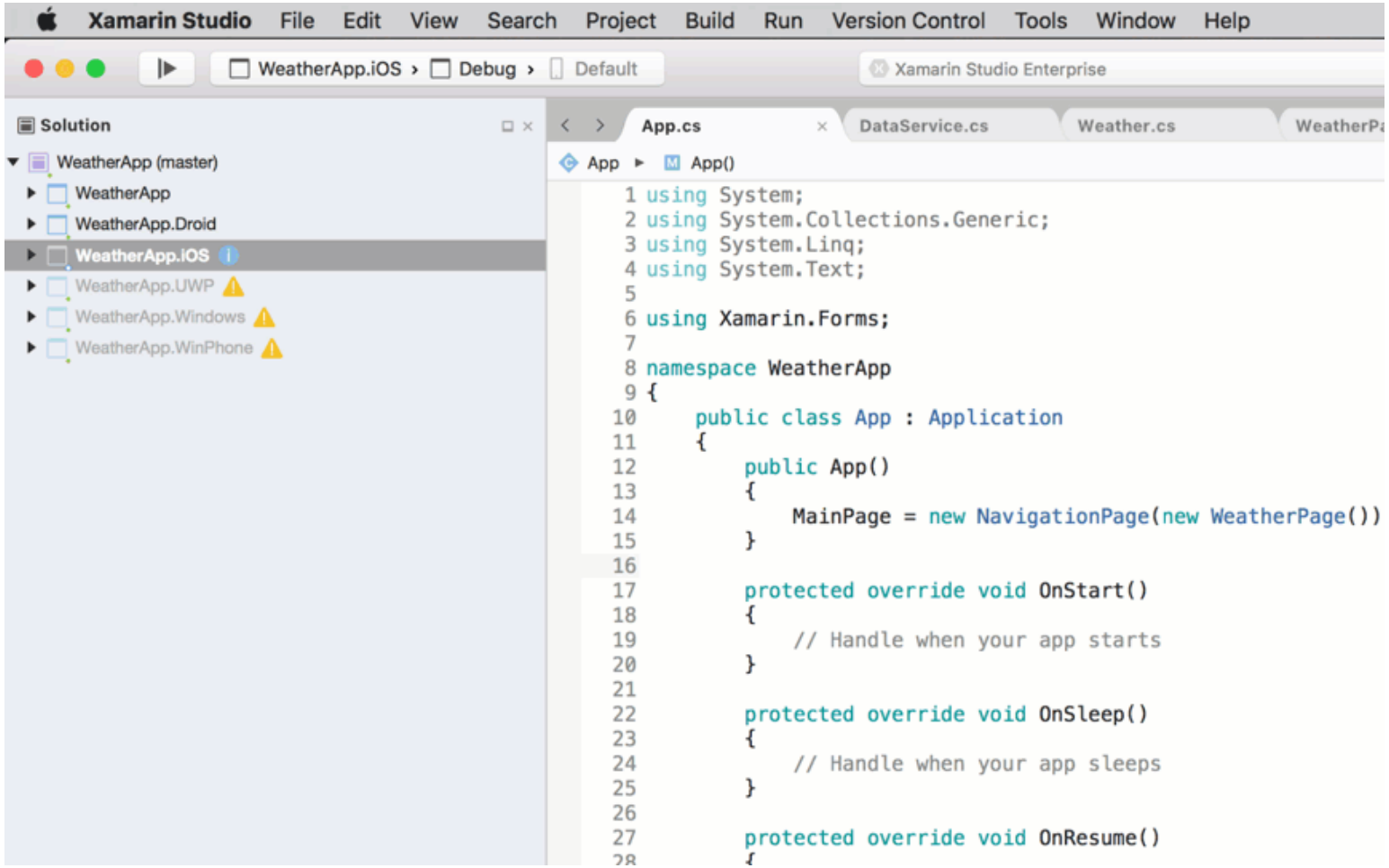
Shared App Logic



Xamarin.Forms Renderers for WPF



Xamarin.Forms Renderers for macOS/GTK



Pixel-Perfect Multi-Platform Applications with C# and WinUI

Open-source UI Platform for building single-codebase applications for Windows, iOS, Android, WebAssembly, macOS, and Linux

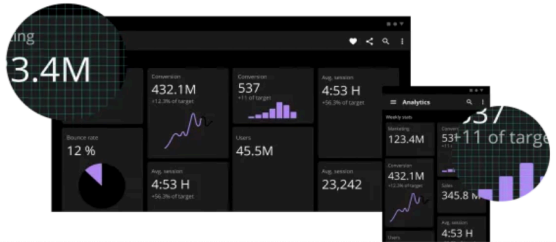
Get Started

Target All Operating Systems & Browsers

The first C# & XAML, **free** and **open-source** platform for creating true single-source, multi-platform applications. Reuse 99% of the business logic and UI layer across native mobile, web, and desktop.

Read More

Pixel Perfect Everywhere



Maintain ***pixel-perfect control*** over where pixels go while having a ***choice*** of developing platform-specific or custom look and feel of your application.

Read More

Introducing Avalonia XPF preview.

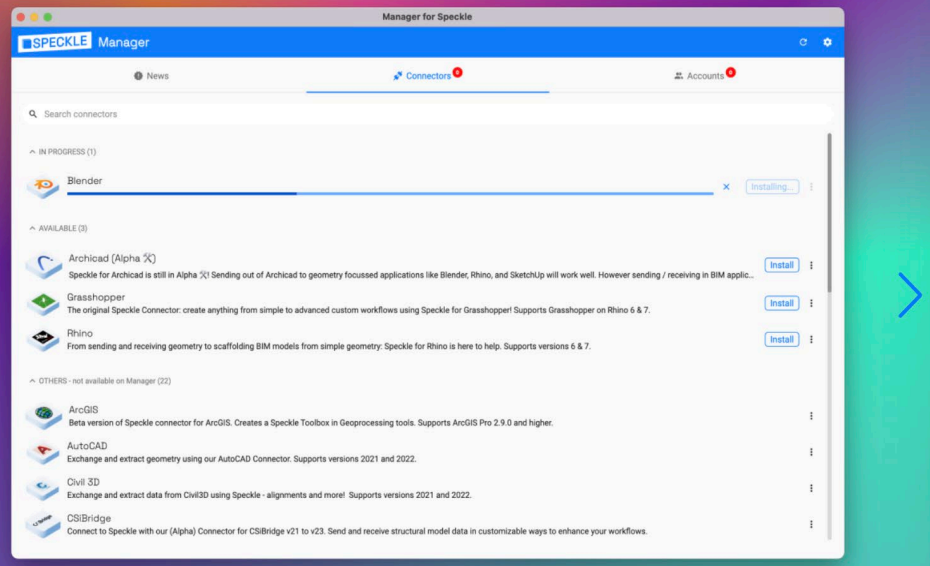
Cross-platform WPF, powered by Avalonia UI.

Learn More →

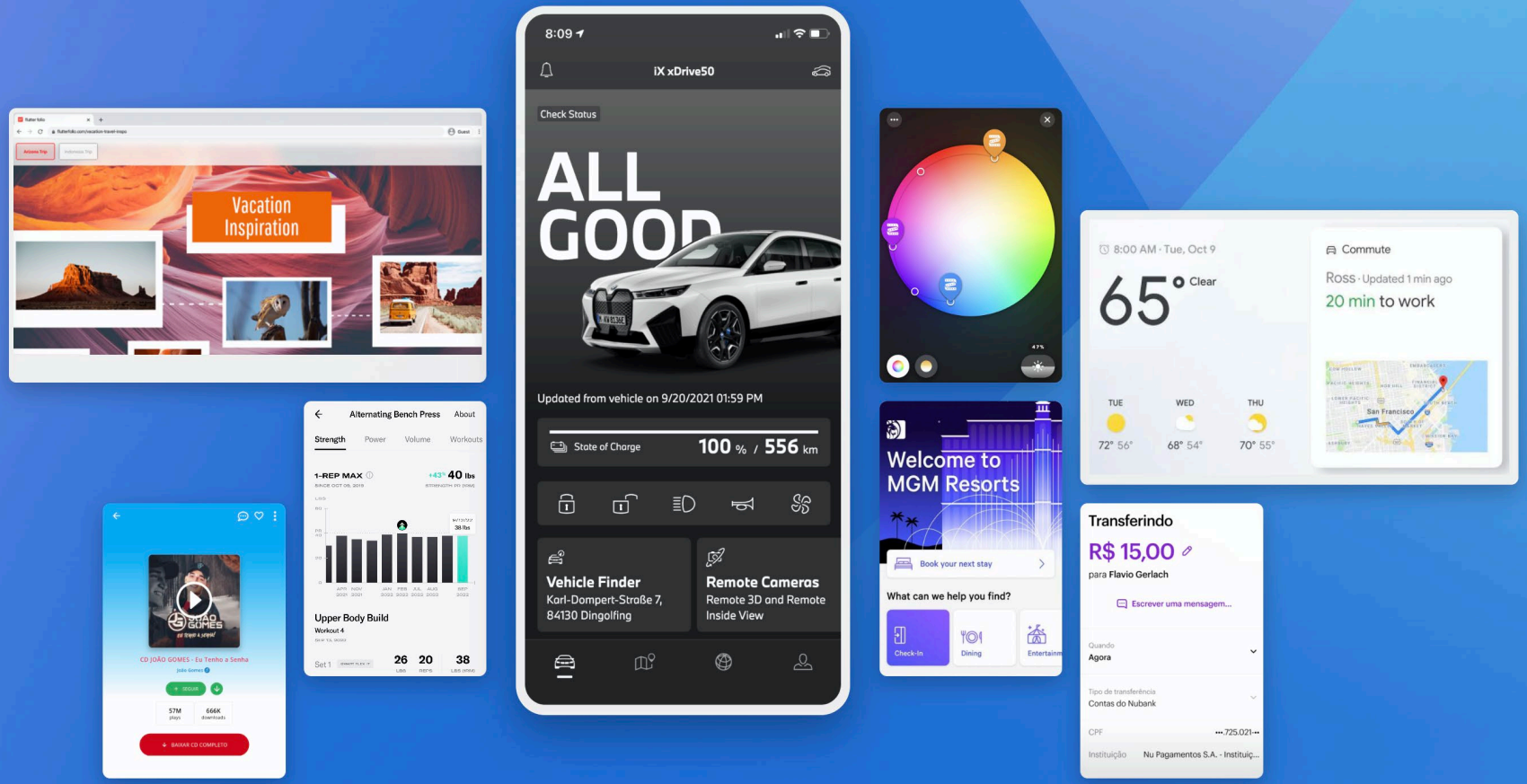
Create Multi-Platform Apps with .NET

Avalonia UI is open-source.

SUPPORTED PLATFORMS



Build apps for any screen



Flutter transforms the app development process. Build, test, and deploy beautiful mobile, web, desktop, and embedded apps from a single codebase.

Welcoming the Web

ASP.NET Core & the Modern Web

Totally Modular

Faster Development Cycle

Seamless transition
from on-premises to cloud

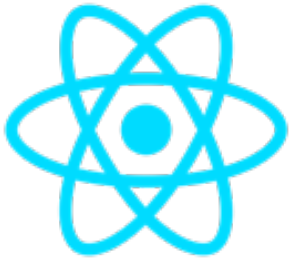
Choose your Editors
and Tools

Open Source
with Contributions

Cross-Platform

Fast

Modern Web Apps



Overview of Progressive Web Apps (PWAs)

Article • 01/24/2023 • 4 minutes to read • 7 contributors

Progressive Web Apps (PWAs) are applications that you build by using web technologies, and that can be installed and can run on all devices, from one codebase.

PWAs provide native-like experiences to your users on supporting devices. They adapt to the capabilities supported by each device and they can also run in web browsers, like websites.

To start building a PWA, see Get started with Progressive Web Apps.

PWA benefits

Native-like experiences

When installed on a device, PWAs function just like other apps. For example:

- PWAs have their own application icons that can be added to a device's home screen or task bar.
- PWAs can be launched automatically when an associated file type is opened.
- PWAs can run when the user signs in.
- PWAs can be submitted to application stores, such as the Microsoft Store.

Advanced capabilities

PWAs also have access to advanced capabilities. For example:

- PWAs can continue working when the device is offline.
- PWAs support push notifications.
- PWAs can perform periodic updates even when the application is not running.
- PWAs can access hardware features.

Web-related advantages

Finally, PWAs can run in web browsers, just like websites. Running like websites gives PWAs with advantages, such as:

- PWAs can be indexed by search engines.
- PWAs can be shared and launched from a standard web link.
- PWAs are safe for users because they use secure HTTPS endpoints and other user safeguards.
- PWAs adapt to the user's screen size or orientation, and input method.
- PWAs can use advanced web APIs such as WebBluetooth, WebUSB, WebPayment, WebAuthn, or WebAssembly.

Lower development cost

PWAs have a **much lower cross-platform development cost** than compiled apps that require a specific, separate codebase for each platform, such as Android, iOS, and each desktop operating system.

With a PWA, you can use a single codebase that's shared between your website, mobile app, and desktop app (across operating systems).

Build cross-platform desktop apps with JavaScript, HTML, and CSS

Docs

Web Technologies

Electron embeds Chromium and Node.js to enable web developers to create desktop applications.

Cross Platform

Compatible with macOS, Windows, and Linux, Electron apps run on three platforms across all supported architectures.

Open Source

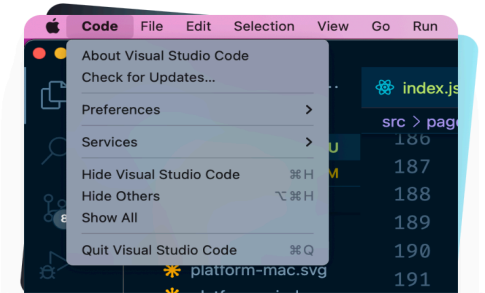
Electron is an open source project maintained by the OpenJS Foundation and an active community of contributors.

Desktop development made easy

Electron takes care of the hard parts so you can focus on the core of your application.

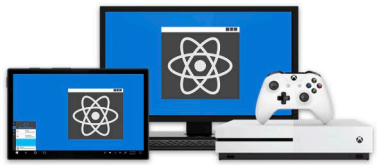
Native graphical user interfaces

Interact with your operating system's interfaces with Electron's main process APIs. Customize your application window appearance, control application menus, or alert users through dialogs or notifications.

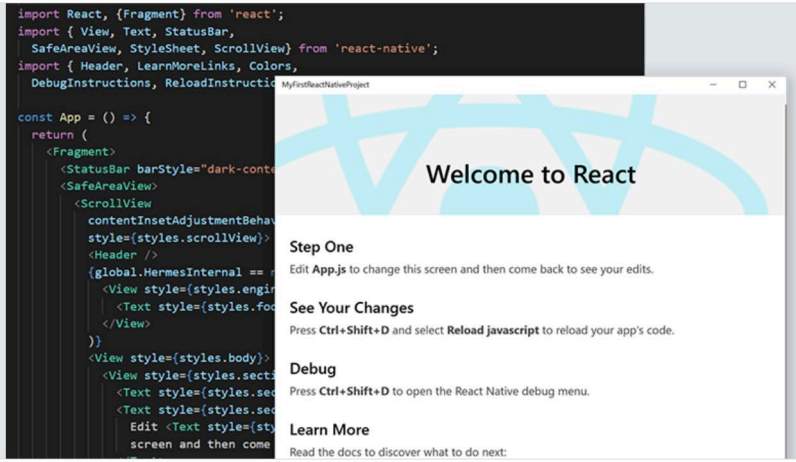


Follow @ReactNativeMSFT

React Native for Windows + macOS



Bring your React Native apps to some of the most powerful devices out there



Build for Windows

GitHub

Take your apps across PC, Xbox, Surface Tablets, and dual-screens with our robust Windows extension to React Native.

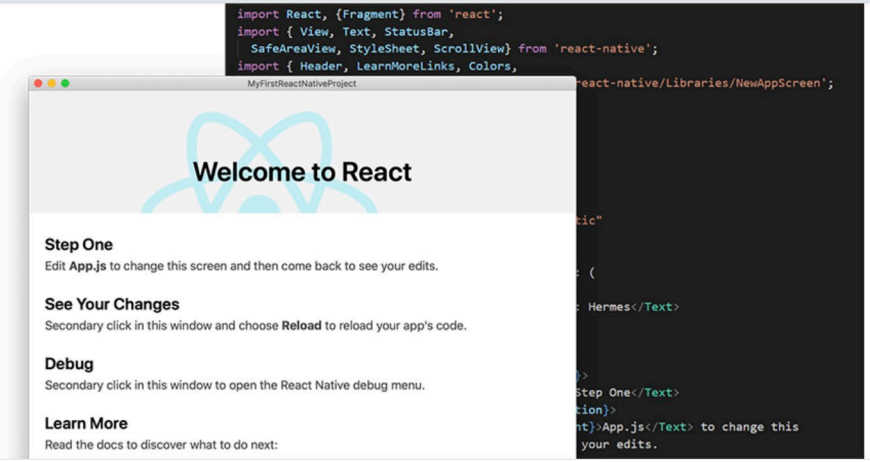
Get started with Windows

Build for macOS

GitHub

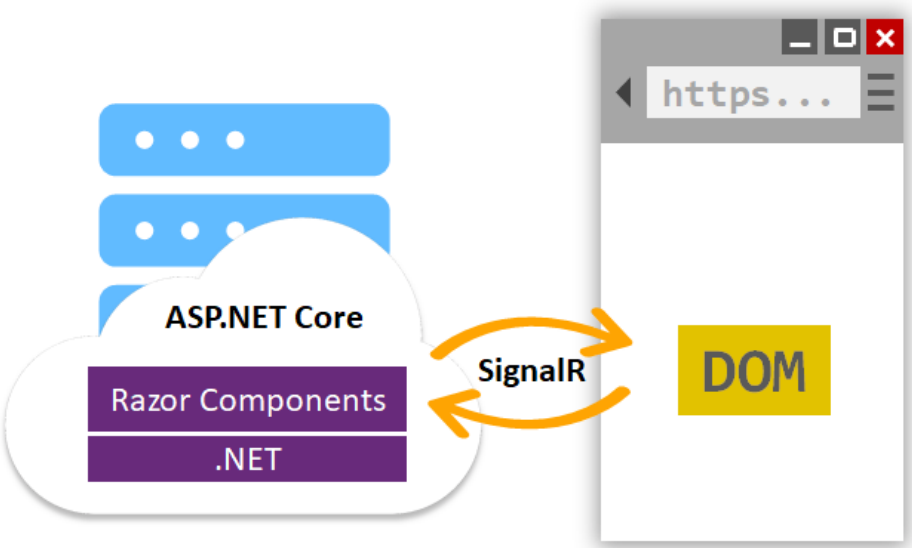
Extend your desktop experience to more than just Windows!
Try out our fully supported macOS extension to React Native.

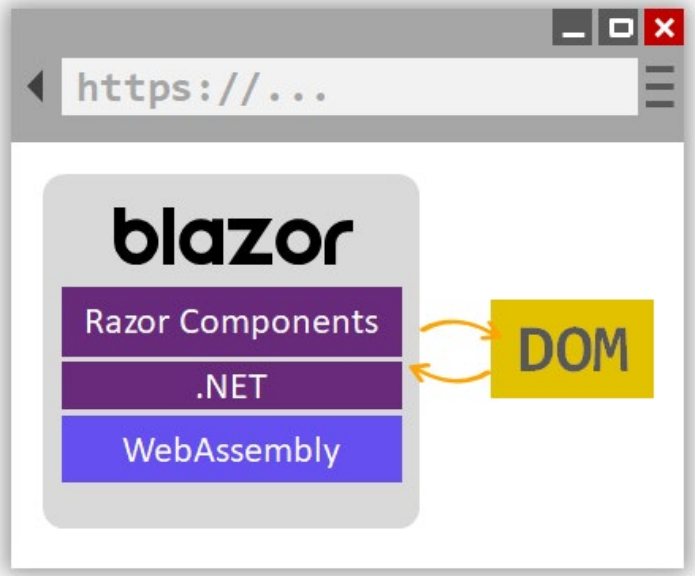
Get started with macOS

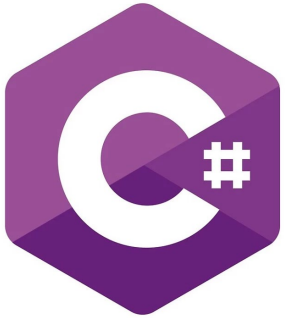


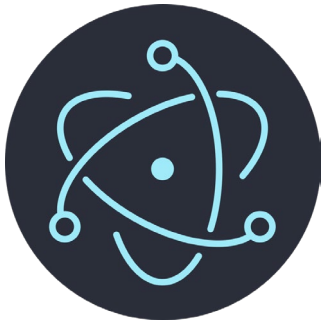
Blazing a Trail ..

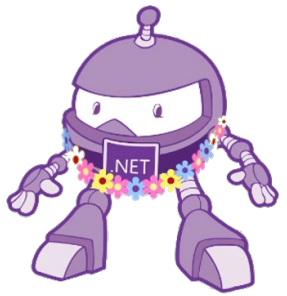
Blazing a trail ..







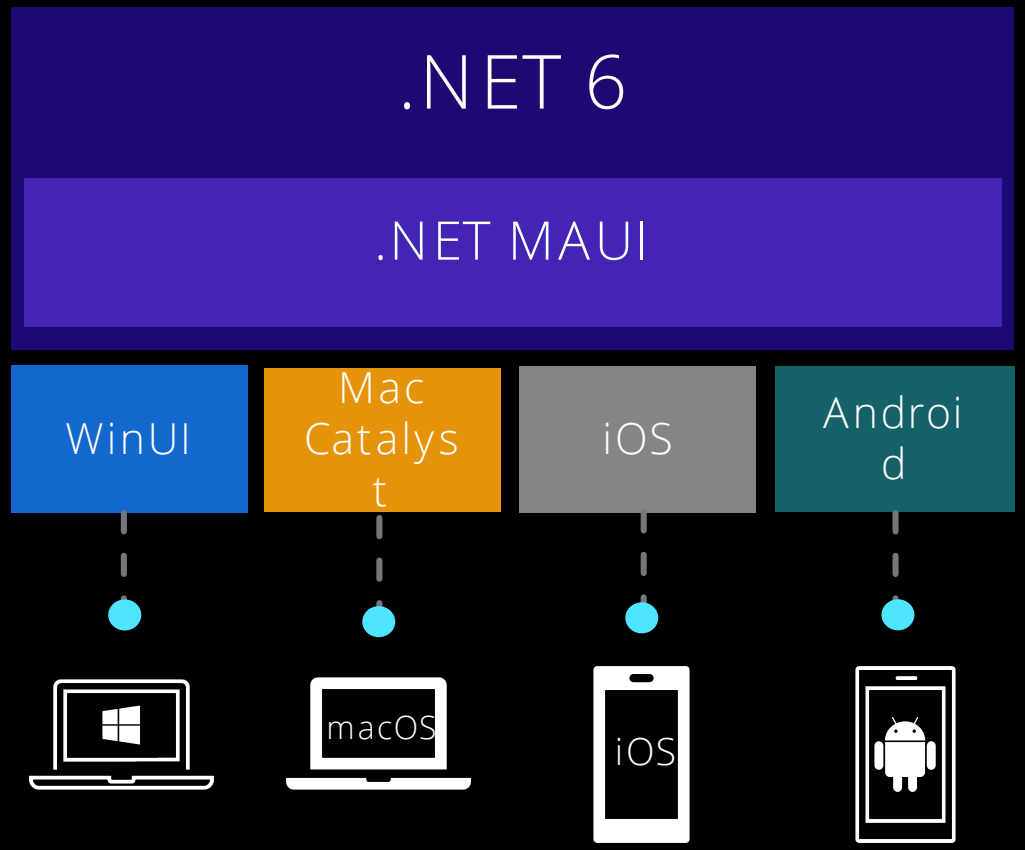




Embracing the New


.NET
MAUI

.NET MAUI Benefits



True Cross-platform | For Mobile & Desktop

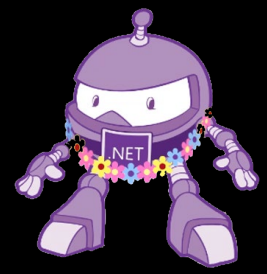
Native UI & Performance | .NET Builder

Single project system | Shared Resources

Easy Platform API access

Handler Architecture | Easy Customization

Blazor Integration | Expansive ecosystem



github.com/dotnet/maui

Blazor Hybrid: + .NET MAUI

Reusability is key ..

Upgrade Assistants can Help

Migration & Modernization

Microsoft® .NET Windows Forms

Windows Presentation Foundation

Universal Windows Platform

Web API

jQuery

Seeing is Believing ..

You have choices for Desktop

The world runs on software

Thank You!

Desktop Apps Your Way

Let's keep chatting | @samidip