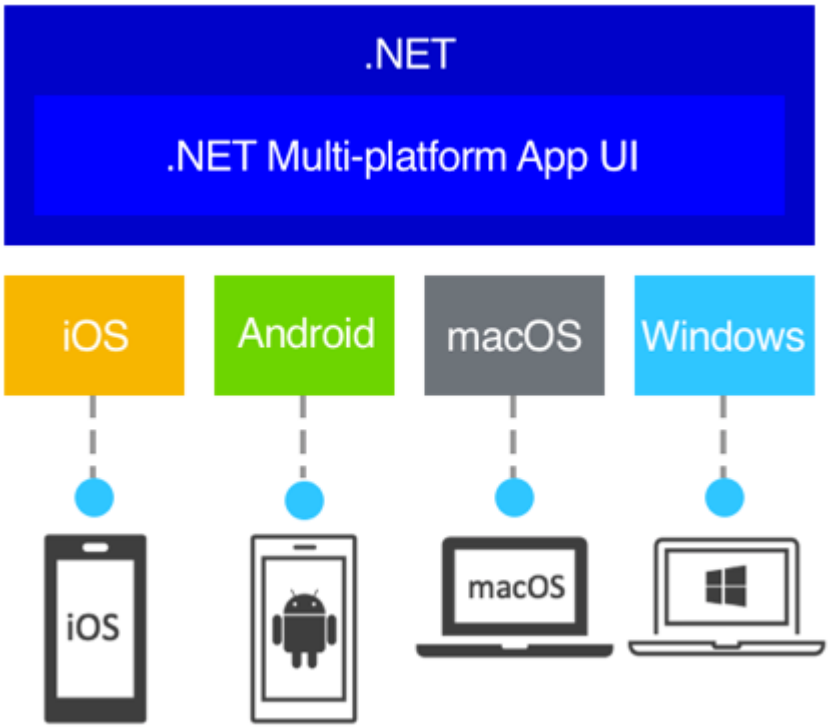
What is .NET MAUI?

Article • 07/25/2023

.NET Multi-platform App UI (.NET MAUI) is a cross-platform framework for creating native mobile and desktop apps with C# and XAML.

Using .NET MAUI, you can develop apps that can run on Android, iOS, macOS, and Windows from a single shared code-base.



.NET MAUI is open-source and is the evolution of Xamarin.Forms, extended from mobile to desktop scenarios, with UI controls rebuilt from the ground up for performance and extensibility. If you've previously used Xamarin.Forms to build cross-platform user interfaces, you'll notice many similarities with .NET MAUI. However, there are also some differences. Using .NET MAUI, you can create multi-platform apps using a single project, but you can add platform-specific source code and resources if necessary. One of the key aims of .NET MAUI is to enable you to implement as much of your app logic and UI layout as possible in a single code-base.

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.

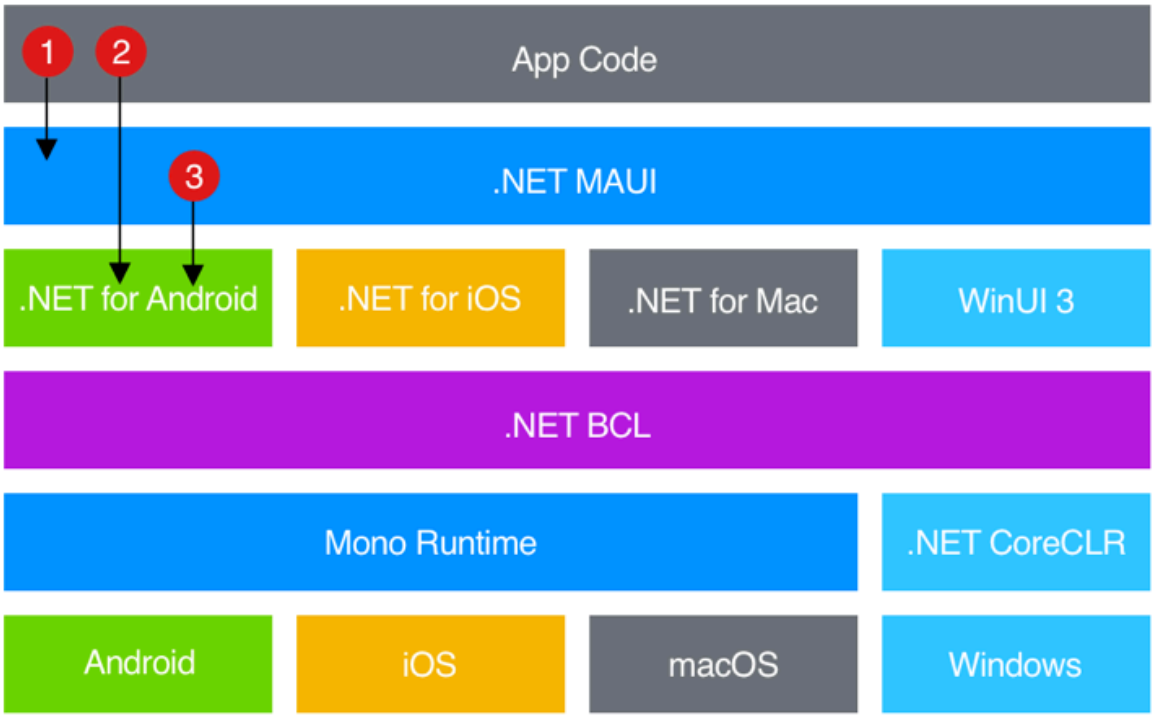
How .NET MAUI works

.NET MAUI unifies Android, iOS, macOS, and Windows APIs into a single API that allows a write-once run-anywhere developer experience, while additionally providing deep access to every aspect of each native platform.

.NET 6 or greater provides a series of platform-specific frameworks for creating apps: .NET Android, .NET iOS, .NET macOS, and Windows UI 3 (WinUI 3) library. These frameworks all have access to the same .NET Base Class Library (BCL). This library abstracts the details of the underlying platform away from your code. The BCL depends on the .NET runtime to provide the execution environment for your code. For Android, iOS, and macOS, the environment is implemented by Mono, an implementation of the .NET runtime. On Windows, .NET CoreCLR provides the execution environment.

While the BCL enables apps running on different platforms to share common business logic, the various platforms have different ways of defining the user interface for an app, and they provide varying models for specifying how the elements of a user interface communicate and interoperate. You can craft the UI for each platform separately using the appropriate platform-specific framework (.NET Android, .NET iOS, .NET macOS, or WinUI 3), but this approach then requires you to maintain a code-base for each individual family of devices.

.NET MAUI provides a single framework for building the UIs for mobile and desktop apps. The following diagram shows a high-level view of the architecture of a .NET MAUI app:



In a .NET MAUI app, you write code that primarily interacts with the .NET MAUI API (1). .NET MAUI then directly consumes the native platform APIs (3). In addition, app code may directly exercise platform APIs (2), if required.

.NET MAUI apps can be written on PC or Mac, and compile into native app packages:

- Android apps built using .NET MAUI compile from C# into an intermediate language (IL) which is then just-in-time (JIT) compiled to a native assembly when the app launches.
- iOS apps built using .NET MAUI are fully ahead-of-time (AOT) compiled from C# into native ARM assembly code.
- macOS apps built using .NET MAUI use Mac Catalyst, a solution from Apple that brings your iOS app built with UIKit to the desktop, and augments it with additional AppKit and platform APIs as required.
- Windows apps built using .NET MAUI use Windows UI 3 (WinUI 3) library to create native apps that target the Windows desktop. For more information about WinUI 3, see Windows UI Library.

> **Note**
>
> Building apps for iOS and macOS requires a Mac.

What .NET MAUI provides

.NET MAUI provides a collection of controls that can be used to display data, initiate actions, indicate activity, display collections, pick data, and more. In addition to a collection of controls, .NET MAUI also provides:

- An elaborate layout engine for designing pages.
- Multiple page types for creating rich navigation types, like drawers.
- Support for data-binding, for more elegant and maintainable development patterns.
- The ability to customize handlers to enhance the way in which UI elements are presented.
- Cross-platform APIs for accessing native device features. These APIs enable apps to access device features such as the GPS, the accelerometer, and battery and network states. For more information, see Cross-platform APIs for device features.
- Cross-platform graphics functionality, that provides a drawing canvas that supports drawing and painting shapes and images, compositing operations, and graphical object transforms.
- A single project system that uses multi-targeting to target Android, iOS, macOS, and Windows. For more information, see .NET MAUI Single project.
- .NET hot reload, so that you can modify both your XAML and your managed source code while the app is running, then observe the result of your modifications without rebuilding the app. For more information, see .NET hot reload.

Cross-platform APIs for device features

.NET MAUI provides cross-platform APIs for native device features. Examples of functionality provided by .NET MAUI for accessing device features includes:

- Access to sensors, such as the accelerometer, compass, and gyroscope on devices.
- Ability to check the device's network connectivity state, and detect changes.
- Provide information about the device the app is running on.
- Copy and paste text to the system clipboard, between apps.
- Pick single or multiple files from the device.
- Store data securely as key/value pairs.
- Utilize built-in text-to-speech engines to read text from the device.
- Initiate browser-based authentication flows that listen for a callback to a specific app registered URL.

Single project

.NET MAUI single project takes the platform-specific development experiences you typically encounter while developing apps and abstracts them into a single shared

project that can target Android, iOS, macOS, and Windows.

.NET MAUI single project provides a simplified and consistent cross-platform development experience, regardless of the platforms being targeted. .NET MAUI single project provides the following features:

- A single shared project that can target Android, iOS, macOS, and Windows.
- A simplified debug target selection for running your .NET MAUI apps.
- Shared resource files within the single project.
- A single app manifest that specifies the app title, id, and version.
- Access to platform-specific APIs and tools when required.
- A single cross-platform app entry point.

.NET MAUI single project is enabled using multi-targeting and the use of SDK-style projects. For more information about .NET MAUI single project, see .NET MAUI single project.

Hot reload

.NET MAUI includes support for .NET hot reload, which enables you to modify your managed source code while the app is running, without the need to manually pause or hit a breakpoint. Then, your code edits can be applied to your running app without recompilation.

.NET MAUI also includes support for XAML hot reload, which enables you to save your XAML files and see the changes reflected in your running app without recompilation. In addition, your navigation state and data will be maintained, enabling you to quickly iterate on your UI without losing your place in the app.

Collaborate with us on GitHub

The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see our contributor guide.

.NET MAUI feedback

.NET MAUI is an open source project. Select a link to provide feedback:

Open a documentation issue

Provide product feedback