Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

Build Cross-Platform Apps with .NET MAUI and Blazor Hybrid

Class Duration

35 hours of live training delivered over 5 days.

Student Prerequisites

- Professional C# programming experience
- Working knowledge of Blazor and Razor components (see *Build .NET 10 Web Apps with Blazor*)
- Basic HTML and CSS
- No prior mobile or XAML experience required

Target Audience

Blazor and web developers who need to ship native mobile and desktop apps without learning XAML or native UI toolkits. Ideal for teams with existing Blazor web applications who want to reach iOS, Android, Windows, and Mac by reusing their Razor components, design systems, and C# skills — sharing one UI codebase across native apps and the web. This is the web-UI counterpart to our XAML-based *Build Cross-Platform Apps with .NET MAUI* course.

Description

Blazor Hybrid lets you put the Razor components you already have inside a native app: your UI runs in a BlazorWebView with full access to .NET APIs and the device, no WebAssembly limitations and no separate mobile codebase. This five-day course takes Blazor developers from zero mobile experience to store-ready hybrid apps on .NET 10. Day one covers the foundations: how Blazor Hybrid works (and how it differs from Blazor Server, WebAssembly, and plain web views), the MAUI single-project model, app lifecycle, and getting Razor components rendering inside BlazorWebView. Day two is about sharing UI for real: the .NET MAUI Blazor Hybrid and Web App solution template that targets Android, iOS, Windows, Mac, and the web from one set of components, organizing shared component libraries, per-target render logic, and CSS strategies that work in both contexts. Day three goes to the device: calling platform APIs (sensors, geolocation, camera, connectivity)

Copyright 2025 Cloud Contraptions LLC – All Rights Reserved

directly from Razor components through dependency injection, permissions, file system access, and a maps-and-location module including embedding map experiences in hybrid UI. Day four covers data and services: offline-first local data with SQLite, secure storage, resilient REST API consumption, and authentication with MSAL across native and web targets. Day five completes the production picture: state management across the hybrid boundary, JavaScript interop where needed, HybridWebView for embedding existing web apps, testing component and app layers, performance tuning for WebView-hosted UI, and packaging and distributing to the app stores.

Learning Outcomes

- Explain how Blazor Hybrid renders Razor components in a native app and how it compares to Blazor Server, WebAssembly, and XAML MAUI.
- Set up and structure a MAUI Blazor Hybrid project on .NET 10.
- Share one set of Razor components across iOS, Android, Windows, Mac, and the web using the Blazor Hybrid and Web App solution template.
- Organize shared component libraries with per-target rendering and CSS that works across native and web hosts.
- Call platform APIs — geolocation, sensors, camera, connectivity — from Razor components via dependency injection, with correct permission handling.
- Embed map and location experiences in hybrid UI.
- Implement offline-capable local data with SQLite, Preferences, and SecureStorage.
- Consume REST APIs resiliently and authenticate users with MSAL across native and web targets.
- Manage state across the hybrid boundary and apply JavaScript interop where appropriate.
- Use HybridWebView to embed existing web apps and communicate with them from .NET.
- Test Razor components with bUnit and validate app behavior on emulators and devices.
- Tune WebView-hosted UI performance and wire up telemetry with Aspire service defaults.
- Package, sign, and distribute hybrid apps to the Apple App Store, Google Play, and enterprise channels.

Copyright 2025 Cloud Contraptions LLC – All Rights Reserved

Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

Training Materials

Comprehensive courseware is distributed online at the start of class. All students receive a downloadable MP4 recording of the training.

Software Requirements

.NET 10 SDK, Visual Studio 2026 (Windows) or VS Code with the .NET MAUI extension (Windows or Mac), Android SDK with an emulator, and Git. A Mac with Xcode is required only for the iOS build/deployment labs; all other labs run on Windows or Mac.

Training Topics

Blazor Hybrid Foundations

- The hybrid model: Razor components in a BlazorWebView, .NET running natively
- Blazor Hybrid vs. Blazor Server vs. WebAssembly vs. XAML MAUI: choosing per app
- The MAUI single-project model: resources, lifecycle, and dependency injection
- Getting components rendering: root components, host pages, and static assets

Sharing UI Across Native and Web

- The .NET MAUI Blazor Hybrid and Web App solution template
- Razor class libraries for shared components
- Per-target rendering: interactivity, render modes, and device-specific UI
- CSS strategies for native and web hosts; CSS isolation
- Design systems and component libraries in shared UI

Routing, Layouts, and Navigation

- Blazor routing inside a native app
- Layouts, navigation state, and deep linking
- Mixing Blazor navigation with native navigation when needed

Platform Integration from Razor Components

- Injecting platform services into components
- Geolocation, sensors, connectivity, battery, and share

Copyright 2025 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

- Camera and MediaPicker, including .NET 10 multi-select and compression
- Permissions: requesting and degrading gracefully
- File system access across platforms

Maps and Location in Hybrid UI

- Embedding map experiences in hybrid apps
- Geolocation and geocoding from components
- Offline considerations for location-aware apps

Local Data and Remote Services

- SQLite for structured local data; Preferences and SecureStorage
- Offline-first patterns and sync
- HttpClient with resilience: retries, timeouts, connectivity awareness
- Authentication with MSAL across native and web targets
- Token storage and session handling in hybrid apps

State, Interop, and HybridWebView

- App state across the hybrid boundary
- JavaScript interop: when you need it and how to keep it contained
- HybridWebView: embedding existing web apps and bidirectional messaging
- WebView request interception in .NET 10

Quality, Performance, and Observability

- Testing Razor components with bUnit
- Device and emulator validation; automated UI testing
- WebView performance: startup, asset loading, and memory
- Aspire service defaults: OpenTelemetry metrics and tracing

Deployment and Distribution

- Android packaging, signing, and Google Play distribution
- iOS provisioning, signing, and App Store distribution
- Windows and Mac packaging
- Enterprise distribution and CI/CD pipelines

Copyright 2025 Cloud Contraptions LLC – All Rights Reserved