Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Build Cross-Platform Apps with .NET MAUI

Class Duration

35 hours of live training delivered over 5 days.

Student Prerequisites

- Professional C# programming experience
- Basic familiarity with .NET project structure and NuGet
- No prior mobile or XAML experience required

Target Audience

C# developers who need to ship native apps for iOS, Android, Windows, and Mac from a single codebase. Ideal for teams with existing .NET investments - backend services, authentication, and tooling - who want to extend them to mobile and desktop without maintaining separate Swift, Kotlin, and WPF codebases. The course includes a dedicated module on maps and geolocation, including integrating third-party mapping SDKs, for teams building location-aware applications.

Description

.NET MAUI is Microsoft’s framework for building native cross-platform apps in C#, and .NET 10 is its most production-ready release yet, with compiled XAML via the new source generator, global XAML namespaces, faster startup, and first-class observability through .NET Aspire service defaults. This five-day course takes professional C# developers from project setup to store-ready apps targeting Android 16 and iOS 26. The first half covers tooling and the single-project model, the handler architecture, XAML and the full layout system, data binding through compiled bindings, and MVVM with the CommunityToolkit.Mvvm source generators. From there the course moves to Shell navigation, data-heavy UI with CollectionView, styling and themes, platform APIs and permissions, camera and media, maps and geolocation, offline-first data with SQLite, and resilient REST consumption with MSAL authentication. The final day completes the production picture: accessibility and localization, testing strategy, performance work with

trimming and NativeAOT on iOS, observability with Aspire and OpenTelemetry, and packaging, signing, and CI/CD for every target platform.

Learning Outcomes

- Set up a productive MAUI environment with Visual Studio 2026, VS Code with the .NET MAUI extension, the dotnet CLI, and emulators, and explain the architecture: single project, app lifecycle, and handlers.
- Write idiomatic XAML with the full layout system and control catalog, using resources, markup extensions, and .NET 10's compiled XAML source generator and global namespaces.
- Apply data binding through compiled bindings and structure apps with MVVM using CommunityToolkit.Mvvm source generators, with clean validation and user input patterns.
- Structure navigation with Shell (routes, parameters, flyouts, tabs), display large data sets efficiently with CollectionView and CarouselView, and style apps with themes, dark mode, and visual states.
- Access device capabilities (sensors, connectivity, camera and media with .NET 10 multi-select and compression) with proper permission handling, and build location-aware features with the Maps control and third-party mapping SDKs.
- Implement offline-first data with SQLite, Preferences, SecureStorage, and sync patterns, and consume REST APIs resiliently with MSAL authentication.
- Make apps accessible and localized, test at the right levels, tune startup and size with trimming and NativeAOT on iOS, and wire up telemetry with Aspire service defaults and OpenTelemetry.
- Package, sign, and distribute apps targeting Android 16 and iOS 26 to the stores and enterprise channels, with CI/CD pipelines that build every platform.

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

Project Setup and Tooling

- Installing .NET 10 and the MAUI workloads on Windows and Mac
- Visual Studio 2026 vs. VS Code with the .NET MAUI extension: choosing and configuring
- The dotnet CLI workflow: creating, building, and running per target
- Android emulators, iOS simulators, and deploying to physical devices
- Hot reload for XAML and C#: a fast inner loop
- Targeting Android 16 and iOS 26: SDK levels and minimum versions

The Single-Project Model and App Lifecycle

- One project, four platforms: how multi-targeting works
- Shared resources: fonts, images, raw assets, splash screens, and app icons
- MauiProgram and the builder pattern: registering services with dependency injection
- App, Window, and Page lifecycle events across platforms
- Per-platform code under the Platforms folders
- Handling suspend, resume, and termination correctly on mobile

The Handler Architecture

- From renderers to handlers: how cross-platform controls become native views
- Property mappers and command mappers
- Customizing a control for one platform without forking your UI
- Conditional compilation and partial classes for platform-specific logic
- When to drop down to native APIs - and how to keep it contained

XAML Fundamentals

- XAML syntax: elements, attributes, and the relationship to C# objects
- Pages, ContentView, and composing a screen
- Attached properties and property element syntax
- Code-behind, event handlers, and x:Name
- Namespaces and bringing your own types into markup

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

XAML in Depth: Resources, Markup Extensions, and Compiled XAML

- ResourceDictionary: scoping, merging, and organizing app resources
- StaticResource vs. DynamicResource and when each matters
- Markup extensions: built-in, OnPlatform/OnIdiom, and writing your own
- The .NET 10 compiled XAML source generator: build-time errors and faster startup
- Global XAML namespaces: cleaner markup without per-file xmlns declarations
- Diagnosing XAML build errors early instead of at runtime

Layouts in Depth

- Grid: rows, columns, spans, and proportional sizing
- VerticalStackLayout and HorizontalStackLayout: when stacks are the right tool
- FlexLayout for wrapping and responsive arrangements
- AbsoluteLayout and when it earns its keep
- ScrollView, SafeArea, and keyboard-aware layout
- Sizing, alignment, spacing, and margins: a mental model that transfers
- Adapting layout to phones, tablets, and desktops with OnIdiom and adaptive triggers

The Control Catalog

- Text and input: Label, Entry, Editor, formatted text, and keyboards
- Buttons, ImageButton, and gesture recognizers
- Selection controls: Picker, DatePicker, TimePicker, Switch, Slider, Stepper
- Images, Shapes, Border, Shadow, and brushes
- ActivityIndicator, ProgressBar, and communicating busy state
- WebView and content presentation controls

Data Binding Fundamentals

- Binding sources, paths, and the BindingContext
- Binding modes: OneWay, TwoWay, OneTime, OneWayToSource
- Value converters and string formatting
- Relative bindings, binding to ancestors, and fallback values
- BindableLayout for small repeated collections
- INotifyPropertyChanged: what the UI actually listens to

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Compiled Bindings and MVVM with Source Generators

- Compiled bindings with x:DataType: compile-time safety and runtime speed
- Why reflection-based bindings cost you on mobile
- MVVM: separating view, view model, and model for testability
- CommunityToolkit.Mvvm: ObservableProperty and RelayCommand via source generators
- Async commands, CanExecute, and command parameters
- Messaging between view models with WeakReferenceMessenger

Validation and User Input

- Validation where it belongs: view model rules, not code-behind
- ObservableValidator and data annotations
- Surfacing errors: styles, visual states, and inline messages
- Input behaviors: masking, max length, and keyboard return handling
- Debouncing input and validating asynchronously

Navigation with Shell

- Shell structure: flyouts, tabs, and page hierarchy in one place
- Route registration and URI-based navigation
- Passing parameters with query properties and full objects
- Returning results to the caller
- Modal navigation and presentation styles
- Navigation guards, redirects, and login flows
- Deep links from outside the app

Lists and Data-Heavy UI

- CollectionView: data templates, template selectors, and item sizing
- Selection, grouping, headers, footers, and empty views
- Virtualization and scroll performance with large data sets
- CarouselView and IndicatorView
- RefreshView and incremental loading patterns
- The optimized CollectionView and CarouselView handlers, default in .NET 10

Styling, Themes, and Visual States

- Implicit and explicit styles; style inheritance and BasedOn
- App-wide theming and resource organization

- Dark mode with AppThemeBinding and responding to theme changes
- Visual State Manager: states, state groups, and adaptive UI
- Triggers: property, data, and event triggers
- Custom fonts and font icons

Platform APIs, Permissions, and Media

- The essentials APIs: connectivity, battery, device info, share, clipboard, launcher
- Sensors: accelerometer, gyroscope, compass, and shake detection
- The permission model on Android 16 and iOS 26: declaring, requesting, checking, and degrading gracefully
- File system access and file/folder pickers across platforms
- Camera and MediaPicker, including .NET 10 multi-select and built-in image compression
- Handling large media: resizing, caching, and memory pressure
- Media playback with MediaElement; privacy manifest entries for camera and library access

Maps and Geolocation

- The Maps control: pins, polygons, polylines, and user location
- Geolocation: one-shot and continuous location, accuracy, and battery trade-offs
- Geocoding and reverse geocoding
- Foreground location patterns and permission flows
- Integrating third-party mapping SDKs into MAUI apps
- Offline considerations for location-aware apps

Local Data and Offline-First Sync

- SQLite with sqlite-net: schema, queries, and async access
- Preferences for settings; SecureStorage for secrets and tokens
- Choosing storage: files vs. preferences vs. database
- Offline-first design: local writes, background sync, and queued operations
- Conflict detection and resolution strategies
- Connectivity-aware behavior: detecting, reacting, and retrying

Remote Services and Authentication

- HttpClient done right: IHttpClientFactory and typed clients

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- Resilience: retries, timeouts, circuit breakers, and connectivity awareness
- System.Text.Json with source-generated serializers
- Authentication with MSAL: sign-in flows, token caching, and refresh
- Storing tokens with SecureStorage and handling expiry

Accessibility and Localization

- Semantic properties: descriptions, hints, and headings for screen readers
- TalkBack and VoiceOver behavior and testing
- Focus order, touch target sizes, and contrast
- Localization with resource files and culture-aware formatting
- Right-to-left layouts and pseudo-localization for early feedback

Testing, Performance, and Observability

- A testing strategy for MAUI: what to test at which level
- Unit testing view models and services; device-based test execution
- Startup performance: measuring, profiling, and the .NET 10 improvements
- Trimming, R8/ProGuard, and NativeAOT on iOS for size and speed
- Aspire service defaults: OpenTelemetry metrics, tracing, and service discovery
- Connecting client telemetry to your backend’s observability story

Packaging, Distribution, and CI/CD

- Android: signing, AAB packaging, and Google Play distribution targeting Android 16
- iOS: provisioning profiles, certificates, and App Store distribution targeting iOS 26
- Windows packaging and Mac Catalyst distribution
- Enterprise and ad-hoc distribution channels
- CI/CD pipelines that build, sign, and publish all targets
- Versioning, release channels, and staged rollouts

