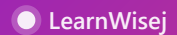


KI-Coding-Übung

Erstellen Sie das Beispielprojekt FieldOps dieses Moduls mit ChatGPT oder Claude aus einem fertigen Prompt, der das Modell auf die Spezifikation des Moduls verweist – und prüfen, starten und erweitern Sie dann, was Sie erhalten.



Den Code zu schreiben ist heute die leichtere Hälfte.

Funktionierenden Code für „Mobile Shells mit .NET MAUI“ bekommen Sie von einem modernen Assistenten mit einem guten Prompt. Ihn zu beurteilen, zu korrigieren und zu erweitern, erfordert alles, was dieses Modul behandelt hat. Genau auf dieser Aufteilung baut die Übung auf: Sie liefern Spezifikation und Review, der Assistent die Tipparbeit.

Worum Sie bitten

Bauen Sie die mobile Shell von FieldOps mit .NET MAUI: eine hybride Seite, die die App unter Android und iOS hostet, eine Session, die den Hintergrundbetrieb übersteht und bei der Rückkehr fortgesetzt wird, die Android-Zurück-Taste, abgebildet auf die Navigation in der App, Safe-Area-Innenabstand für Geräte mit Notch und ein Debug-Build, der auf einem Emulator pro Plattform läuft, mit Screenshots der Liste der Arbeitsaufträge.

Das Modul verlangt diese Ergebnisse, und der Prompt unten fordert genau diese vom Assistenten an:

- ✓ Hybrides.NET-MAUI-Shell-Projekt für Android und iOS
- ✓ Lebenszyklus-Behandlung für Hintergrund und Fortsetzen
- ✓ Zuordnung von Zurück-Taste und Safe Areas
- ✓ Emulator-Builds für beide Plattformen
- ✓ Screenshots aus beiden Emulatoren

Warum es so aufgebaut ist

Die Techniker von FieldOps arbeiten mit einem Smartphone im Technikraum, im Keller oder im Transporter, und ein Browser-Tab ist dafür ein schlechter Container: kein Icon für den Startbildschirm, das man verteilen könnte, kein Store- oder Unternehmenskanal und kein Weg zu Kamera und GPS, die Modul 5 brauchen wird. Das mobile Betriebssystem ist zudem ein weit aggressiverer Host – es suspendiert die App, sobald ein Anruf eingeht, zeichnet sie unter einer Notch neu und gibt dem Techniker eine System-Zurück-Geste in die Hand, um die FieldOps nie gebeten hat.

Diese Begriffe muss der Assistent richtig verwenden. Geht die Antwort nachlässig mit ihnen um, hat er das Modul nicht verstanden:

Begriff	Was er hier bedeutet
Hybride.NET-MAUI-Shell	Eine kleine native App für Android und iOS, deren einzige Seite Ihre Wisej.NET-Anwendung in einer Web-View hostet – der Techniker sieht also die C#-Bildschirme, die Sie bereits ausliefern.
Nativer Web-View-Host	Die in diese Seite eingebettete Browser-Engine der Plattform – Android System WebView, WKWebView unter iOS –, die denselben Client ausführt wie ein Desktop-Browser; deshalb wird kein Bildschirm für Mobilgeräte neu geschrieben.
Hybrid-Bridge	Der bidirektionale Kanal zwischen der gehosteten Seite und nativem Code, damit serverseitiger C#-Code das Gerät um etwas bitten und die Antwort als Event statt als Seitennavigation erhalten kann.
MAUI-Anwendungslebenszyklus	Die Events, die das Betriebssystem rund um die App auslöst – Fenster erstellt, aktiviert, deaktiviert, gestoppt und fortgesetzt, dazu OnSleep und OnResume –, von denen ein Browser-Tab kein einziges auslöst.
Überlebensfenster der Session	Die Zeitspanne im Hintergrund, in der die Wisej.NET-Session auf dem Server noch lebt, sodass sich die App beim Fortsetzen mit dem vorhandenen Zustand verbinden kann, statt neu zu starten.
Safe-Area-Insets	Der gerätespezifische Innenabstand, der Inhalte von Notch, Statusleiste und Home-Indicator-Streifen fernhält – von der Plattform gemeldet, nicht aus der Bildschirmgröße geraten.
Zuordnung der Zurück-Navigation	Die Regel, die festlegt, was die Zurück-Geste von Android bewirkt: innerhalb der eigenen Navigation der App zurückgehen oder das Verlassen bestätigen lassen – niemals stillschweigend einen halb ausgefüllten Arbeitsauftrag zerstören.

Selbst bauen – wählen Sie Ihren Assistenten

Beide Buttons öffnen einen neuen Tab, in dem **derselbe fertige Prompt** bereits eingetragen ist. Der Prompt verweist das Modell auf die Spezifikation dieses Moduls – eine einzelne Markdown-Datei mit der Lektion, der Lab-Aufgabe und den geforderten Ergebnissen –, sodass es mit dem echten Material arbeitet, statt zu raten, worum es in Modul 4 geht. Sie können [module-04.md](#) selbst öffnen, um genau zu sehen, was es lesen wird.



ChatGPT verwenden

Öffnet [chatgpt.com](#) mit dem Prompt dieses Moduls. Funktioniert mit GPT-5 oder Codex; jeder Tarif mit Websuche ruft die Spezifikation ab.



Claude verwenden

Öffnet [claude.ai](#) mit demselben Prompt. Wenn Sie Claude Code nutzen, können Sie ihn stattdessen in einem leeren Ordner in ein Terminal einfügen.



Unverändert abgeschickt, erhalten Sie ein Projekt, kein Code-Fragment

Der Prompt verlangt am Ende die gesamte Projektmappe als einzelne `.zip`-Datei mit Anleitung zum Kompilieren und Starten. Laden Sie sie herunter, öffnen Sie die Projektmappe in Visual Studio, drücken Sie F5, und FieldOps startet auf `localhost`. Sie benötigen das .NET SDK und das NuGet-Paket `Wisej-4`; alles andere soll der Assistent laut Prompt selbst schreiben.

Ich arbeite gerade Modul 4 – „Mobile Shells mit .NET MAUI“ – des LearnWisej-Kurses „Hybride und native Paketierung“ durch und möchte das Beispielprojekt des Moduls gemeinsam mit dir erstellen. Bitte antworte auf Deutsch.

Lies zuerst die Modulspezifikation. Sie ist eine einzelne Markdown-Datei (auf Englisch) mit der vollständigen Lektion, der Lab-Aufgabe und den geforderten Ergebnissen:

<https://learnwisej.com/assets/courses/hybrid-native-packaging/specs/module-04.md>

Erstelle dann ein vollständiges Wisej.NET-4-Projekt, das ich in Visual Studio öffnen kann:

1. Projekt „FieldOps“, C#, net8.0 oder neuer, NuGet-Paket Wisej-4 4.1.x, Kestrel-Host, der `app.UseWisej()` aufruft.
2. Setze um, was das Lab der Spezifikation verlangt: Bauen Sie die mobile Shell von FieldOps mit .NET MAUI: eine hybride Seite, die die App unter Android und iOS hostet, eine Session, die den Hintergrundbetrieb übersteht und bei der Rückkehr fortgesetzt wird, die Android-Zurück-Taste, abgebildet auf die Navigation in der App, Safe-Area-Innenabstand für Geräte mit Notch und ein Debug-Build, der auf einem Emulator pro Plattform läuft, mit Screenshots der Liste der Arbeitsaufträge.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - Hybrides.NET-MAUI-Shell-Projekt für Android und iOS
 - Lebenszyklus-Behandlung für Hintergrund und Fortsetzen
 - Zuordnung von Zurück-Taste und Safe Areas
 - Emulator-Builds für beide Plattformen
 - Screenshots aus beiden Emulatoren
4. Halte dich an die Spezifikation, wo immer sie eine Wisej.NET-Klasse, -Eigenschaft oder ein -Event nennt – verwende genau diese. Wo sie eine Wahl offenlässt, sag mir, welche API du gewählt hast und warum.
5. Der Server bleibt maßgeblich: Zustand und Validierung leben in .NET, der Browser stellt dar und meldet zurück. Verlagere nie eine Entscheidung nach JavaScript, nur weil es dort einfacher ist.
6. Mach das Ergebnis vorführbar: Ich muss es starten und das Verhalten sehen können, um das es in diesem Modul geht – auch, was passiert, wenn etwas schiefgeht.

Bevor du Code schreibst, fasse in eigenen Worten zusammen, was du bauen wirst und wo jedes Teil liegen wird, damit ich den Entwurf korrigieren kann, solange das noch wenig kostet.

Wenn der Code fertig ist, packe die gesamte Projektmappe in eine einzelne .zip-Datei zum Herunterladen und nenne mir die genauen Schritte, um sie zu kompilieren und auszuführen.

Beantworte zum Schluss diese Fragen aus der Spezifikation und nutze dabei deine eigene Implementierung als Beleg:

- Ein Techniker ist mitten in einer Statusaktualisierung, als ein dreiminütiger Anruf eingeht. Was muss auf dem Server und in der Shell gelten, damit die halb fertige Aktualisierung danach noch da ist, und was ändert sich, wenn der Anruf eine Stunde dauert?
- Die Shell weiß nicht, welcher FieldOps-Bildschirm geöffnet ist. Wie entscheidet eine Zurück-Geste zwischen dem Zurückkehren von der Detailseite, dem Schließen eines Dialogs und dem Verlassen der App, und wo lebt dieser Zustand tatsächlich?
- Welche Änderungen dieses Moduls würde ein Techniker noch bemerken, wenn er FieldOps im Browser des Smartphones statt in der Shell öffnet, und welche würden verschwinden? Was sagt Ihnen das darüber, wo der Safe-Area-Innenabstand hingehört?

Frag mich nach allem, was unklar ist, bevor du anfängst.

Prüfen Sie dann, was zurückkommt

Code, der kompiliert, ist noch lange kein richtiger Code. Die spannenden Fragen sind Designfragen: Ist das wirklich das Wisej.NET-Control, das die Aufgabe verlangt? Folgt der Code dem Lebenszyklus, den das Framework vorgibt? Liegt der maßgebliche Zustand auf dem Server – oder wurde er im Browser abgelegt, weil das bequemer war? Lesen Sie das Ergebnis als Reviewer, nicht als Kunde.

Die Checkliste liefert Ihnen dieses Modul. Beantworten Sie jeden Punkt anhand des Codes, den Sie gerade erhalten haben:

- ✓ Ein Techniker ist mitten in einer Statusaktualisierung, als ein dreiminütiger Anruf eingeht. Was muss auf dem Server und in der Shell gelten, damit die halb fertige Aktualisierung danach noch da ist, und was ändert sich, wenn der Anruf eine Stunde dauert?
- ✓ Die Shell weiß nicht, welcher FieldOps-Bildschirm geöffnet ist. Wie entscheidet eine Zurück-Geste zwischen dem Zurückkehren von der Detailseite, dem Schließen eines Dialogs und dem Verlassen der App, und wo lebt dieser Zustand tatsächlich?
- ✓ Welche Änderungen dieses Moduls würde ein Techniker noch bemerken, wenn er FieldOps im Browser des Smartphones statt in der Shell öffnet, und welche würden verschwinden? Was sagt Ihnen das darüber, wo der Safe-Area-Innenabstand hingehört?

Alles, was Sie korrigieren mussten, sollten Sie festhalten. Eine kurze Notiz zu den Konventionen und Entscheidungen, auf die Sie sich festgelegt haben – eingefügt in die erste Nachricht Ihrer nächsten Sitzung –, ist mehr wert als jeder noch so clevere Prompt.



Für diesen Kurs gibt es noch kein Referenzbeispiel – Ihres ist das erste

Das [LearnWisej-Beispiel-Repository](#) enthält ausgearbeitete Builds für die meisten Kurse, für diesen aber noch nicht. Es gibt also nichts zum Vergleichen und keine Abkürzung: Die Spezifikation und Ihr eigenes Review sind das ganze Sicherheitsnetz. Behalten Sie, was Sie bauen – und wenn es gut wird, würden wir es gern sehen.

Drei Prompts für den nächsten Schritt

Richten Sie Ihren Assistenten auf das Projekt, das Sie gerade gebaut haben, und probieren Sie diese Prompts aus. Jeder ist eine Übung zum Prüfen und Erweitern, kein Neuschreiben.

- ✓ **Beweisen Sie, dass es wirklich funktioniert.** *"Geh dieses Projekt Ergebnis für Ergebnis anhand der Modulspezifikation unter <https://learnwisej.com/assets/courses/hybrid-native-packaging/specs/module-04.md> durch und zeig mir den Beleg, dass jedes wirklich erledigt ist – den Codepfad, worauf ich klicken muss und was ich sehen sollte. Wo ein Ergebnis nur teilweise erfüllt ist, sag das klar, statt es schönzureden. Ergänze dann, was fehlt."*
- ✓ **Prüfen Sie es wie einen Pull-Request.** *"Führ mich durch diesen FieldOps-Code, als würdest du den Pull-Request eines Kollegen prüfen und hättest ihn nicht selbst geschrieben. Wo lebt der Zustand, was passiert bei einer einzelnen Benutzeraktion von Anfang bis Ende, was würdest du zur*

Überarbeitung zurückgeben, und welche Teile würdest du nicht guten Gewissens einem Kunden vorlegen? Sei konkret und zitiere die Zeilen, die du meinst – fasse mir nicht den Entwurf zusammen."

- ✓ **Tauschen Sie ein Teil aus, ohne die Schnittstellen zu verschieben.** "Wähle den Teil dieses FieldOps-Projekts, der am offensichtlichsten nur ein Platzhalter für etwas Echtes ist – eine Fake-Datenquelle, ein In-Memory-Speicher, ein vereinfachter Service – und ersetze ihn durch eine praxisnahe Implementierung. Der Vertrag auf beiden Seiten darf sich nicht ändern: dieselben Typen, dieselben Aufrufe, dasselbe Verhalten im Fehlerfall. Sag mir, was du durch die echte Implementierung behandeln musstest, was der Platzhalter nicht behandelt hat."



Nehmen Sie es mit ins Lab

Das praktische Lab am Ende dieses Moduls verlangt dieselben Ergebnisse, die Sie gerade vom Assistenten haben erstellen lassen. Nutzen Sie, was Sie hier haben, als Gerüst für Ihre Antwort und als Beleg dafür – nicht als die Antwort selbst. Bewertet wird im Lab Ihre Argumentation, und dieser Teil muss von Ihnen stammen.