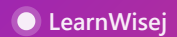


KI-Coding-Übung

Erstellen Sie das Beispielprojekt AdaptiveOps 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, auch im Vergleich mit unserem eigenen Referenz-Build auf GitHub.



Dieses Beispiel tippen Sie nicht ab. Sie steuern es.

Kaum jemand schreibt Wisej.NET-Code heute noch in einer leeren Datei: Ein Assistent liefert den ersten Entwurf, und der Entwickler entscheidet, ob er stimmt. Das funktioniert aber nur, wenn Sie den Stoff beherrschen – und genau den hat Ihnen der Rest dieses Moduls vermittelt. Hier geben Sie dieses Wissen an ChatGPT oder Claude weiter, erhalten das Beispielprojekt des Moduls als lauffähige Visual-Studio-Projektmappe zurück und prüfen es so, wie Sie den Pull-Request eines Kollegen prüfen würden.

Worum Sie bitten

Fügen Sie der Adaptive Operations Console eine ClientProfiles.json mit den Profilen Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop und Desktop hinzu, von schmal nach breit geordnet. Verwenden Sie im Designer responsive Eigenschaften, um Navigationsleiste und Details-Panel auf dem Smartphone auszublenden, die Toolbar-Buttons auf eine reine Symbol-Display umzustellen und den Detailbereich auf dem Tablet unter dem Grid neu zu docken. Behandeln Sie ResponsiveProfileChanged, um den Namen des aktiven Profils im Status-Label anzuzeigen, jeden Wechsel zu protokollieren und den Details-Editor auf dem Smartphone als modales Formular statt als Seitenpanel zu öffnen.

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

- ✓ ClientProfiles.json mit den Profilen Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop und Desktop in First-Match-Reihenfolge
- ✓ Im Designer zugewiesene Werte responsiver Eigenschaften, die Navigationsleiste und Details-Panel auf dem Smartphone ausblenden
- ✓ Reine Symbol-Display der Toolbar und neu gedockter Detailbereich für Tablet, gesetzt über responsive Eigenschaften
- ✓ ResponsiveProfileChanged-Handler, der den Namen des aktiven Profils anzeigt und protokolliert
- ✓ Details-Editor, der auf dem Smartphone über den Profil-Handler als modales Formular geöffnet wird

Warum es so aufgebaut ist

Die Adaptive Operations Console hat jetzt eine gethemte Shell aus Flow-, Table- und Flex-Containern, die eine Größenänderung ohne Resize-Handler übersteht, zeigt aber allen denselben Bildschirm: Auf einem

390 Pixel breiten Smartphone konkurrieren Navigationsleiste, Ticket-Grid und Details-Editor um dieselbe Breite. CSS-Media-Queries können das nicht lösen, weil die Controls auf dem Server leben: Ein Panel auszublenden, einen Button auf reine Symbolanzeige umzustellen oder einen Bereich neu zu docken, ist eine serverseitige Eigenschaftsänderung, und ein Smartphone-Benutzer, der das Details-Panel verliert, braucht einen anderen Weg, ein Ticket zu bearbeiten, keine verkleinerte Kopie des Desktops. Dieses Modul fügt den Mechanismus hinzu, den Wisej.NET dafür vorsieht: Client-Profile, responsive Eigenschaften und das Event ResponsiveProfileChanged.

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
ClientProfile	Ein benannter Satz gerätebezogener Abgleichregeln (Browserbreite, Bildschirmbreite, Gerätetyp, User-Agent, Querformat), der beschreibt, wie eine Klasse von Clients genutzt wird – kein Marketingname eines Geräts.
ClientProfiles.json	Die Datei, die Wisej.NET liest; Profile werden von oben nach unten ausgewertet, und das erste passende gewinnt. Die Reihenfolge der Einträge ist also Teil der Konfiguration.
Application.ActiveProfile	Das aktuell für diese Session ausgewählte Profil; sein Name ist das, was das Status-Label anzeigt und worauf der Profil-Handler verzweigt.
responsive Eigenschaft	Eine normale.NET-Eigenschaft, markiert mit ResponsivePropertyAttribute, die einen Wert pro Client-Profil hält und den passenden Wert anwendet, sobald sich das aktive Profil ändert.
Designer-first-Anpassung	Responsive Werte werden ausschließlich im Visual-Studio-Designer zugewiesen, was sie serialisiert, sichtbar und deterministisch hält; Code ist für Anpassungen da, die von Daten oder vom Laufzeitzustand abhängen.
ResponsiveProfileChanged	Das Event, das bei einer Änderung des aktiven Profils auf Application, auf Containern wie Form, Page, Desktop und UserControl sowie auf DataGridView und ListView ausgelöst wird; ResponsiveProfileChangedEventArgs.CurrentProfile enthält das neue Profil.
responsive QA-Matrix	Der Testplan mit Breite, Höhe, Ausrichtung, Gerätetyp, Touch, Scrollbars, Schriftskalierung und dem Profilnamen, den Sie bei jeder Größe erwarten.

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 6 geht. Sie können [module-06.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 AdaptiveOps 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 6 – „ClientProfile und responsive Eigenschaften“ – des LearnWisej-Kurses „Theming und responsive Layouts“ 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/theming-and-responsive-layouts/specs/module-06.md>

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

1. Projekt „AdaptiveOps“, 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: Fügen Sie der Adaptive Operations Console eine `ClientProfiles.json` mit den Profilen Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop und Desktop hinzu, von schmal nach breit geordnet. Verwenden Sie im Designer responsive Eigenschaften, um Navigationsleiste und Details-Panel auf dem Smartphone auszublenden, die Toolbar-Buttons auf eine reine Symbol-Display umzustellen und den Detailbereich auf dem Tablet unter dem Grid neu zu docken. Behandeln Sie `ResponsiveProfileChanged`, um den Namen des aktiven Profils im Status-Label anzuzeigen, jeden Wechsel zu protokollieren und den Details-Editor auf dem Smartphone als modales Formular statt als Seitenpanel zu öffnen.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - `ClientProfiles.json` mit den Profilen Phone, Phone Landscape, Tablet, Tablet Landscape, Small Desktop und Desktop in First-Match-Reihenfolge
 - Im Designer zugewiesene Werte responsiver Eigenschaften, die Navigationsleiste und Details-Panel auf dem Smartphone ausblenden
 - Reine Symbol-Display der Toolbar und neu gedockter Detailbereich für Tablet, gesetzt über responsive Eigenschaften
 - `ResponsiveProfileChanged`-Handler, der den Namen des aktiven Profils anzeigt und protokolliert
 - Details-Editor, der auf dem Smartphone über den Profil-Handler als modales Formular geöffnet wird
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:

- Würde Small Desktop an den Anfang von `ClientProfiles.json` verschoben: Welche der anderen fünf Profile könnten dann noch getroffen werden, und was sähe ein Tablet im Querformat?
- Das Neu-Docken auf dem Tablet liegt im Designer, der modale Editor auf dem Smartphone im

Code. Was entscheidet, welcher Mechanismus für welche Änderung zuständig ist, und was ginge schief, wenn Sie sie vertauschen?

– Ein Tester hat ein Smartphone viermal gedreht, und Ihr Handler ist viermal gelaufen.

Welcher Zustand muss nach dem vierten Durchlauf identisch mit dem nach dem ersten sein, und wie garantiert Ihr Code das?

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:

- ✓ Würde Small Desktop an den Anfang von ClientProfiles.json verschoben: Welche der anderen fünf Profile könnten dann noch getroffen werden, und was sähe ein Tablet im Querformat?
- ✓ Das Neu-Docken auf dem Tablet liegt im Designer, der modale Editor auf dem Smartphone im Code. Was entscheidet, welcher Mechanismus für welche Änderung zuständig ist, und was ginge schief, wenn Sie sie vertauschen?
- ✓ Ein Tester hat ein Smartphone viermal gedreht, und Ihr Handler ist viermal gelaufen. Welcher Zustand muss nach dem vierten Durchlauf identisch mit dem nach dem ersten sein, und wie garantiert Ihr Code das?

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.



Dieses Beispiel haben wir schon gebaut – vergleichen Sie Ihres mit unserem

Die Referenzimplementierung zu Modul 6 liegt im öffentlichen Beispiel-Repository. Bauen Sie zuerst Ihre eigene Version und öffnen Sie dann unsere: Die Unterschiede sind der interessante Teil.



[Beispiel zu Modul 6 auf GitHub öffnen](#)

Oder klonen Sie alle Beispiele mit `git clone https://github.com/iceteagroup/LearnWisej-Samples` und öffnen Sie [Theming and Responsive Layouts Course/Module 6](#).



Unser Beispiel ist ein Ausgangspunkt, kein fertiges Produkt

Es ist genauso entstanden wie Ihres – per Prompt und anschließend korrigiert – und endet bewusst dort, wo das Modul endet. Rechnen Sie mit Platzhaltern, wo Produktivcode hingehört, mit knapper Fehlerbehandlung und ohne Tests. Verstehen Sie das als Einladung: Nutzen Sie, was Sie in diesem Modul gelernt haben, und dazu den Assistenten, um es weiterzuentwickeln. Code zu verbessern, den Sie nicht selbst geschrieben haben, ist die Fähigkeit, die dieser Kurs eigentlich vermittelt.

Drei Prompts für den nächsten Schritt

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

- ✓ **Ergänzen Sie den zweiten Fall.** *"Erweitere dieses AdaptiveOps-Projekt um einen zweiten Fall, der demselben Muster folgt wie der erste – eine weitere Entität, Maske oder ein Control derselben Form, auf dieselbe Weise verdrahtet. Lagere wirklich Gemeinsames in eine Basisklasse oder einen Service aus, aber nur dort, wo es sich lohnt; zwei ähnliche Dinge brauchen nicht immer eine Abstraktion. Zeig mir den Diff und begründe jede Auslagerung."*
- ✓ **Beweisen Sie, dass es wirklich funktioniert.** *"Geh dieses Projekt Ergebnis für Ergebnis anhand der Modulspezifikation unter <https://learnwisej.com/assets/courses/theming-and-responsive-layouts/specs/module-06.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 AdaptiveOps-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."*



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.