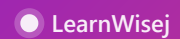


KI-Coding-Übung

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.



Der Assistent liefert die schnellen Hände. Der Architekt bleiben Sie.

Ein Assistent erstellt das Beispielprojekt dieses Moduls in wenigen Minuten. Was er nicht kann, ist zu entscheiden, ob das Ergebnis der richtige Entwurf für Ihre Anwendung ist – diese Entscheidung bleibt bei Ihnen. Das Urteilsvermögen bringen Sie mit: Instruieren Sie ihn sorgfältig und prüfen Sie dann, was zurückkommt, anhand dessen, was Ihnen Modul 7 gerade vermittelt hat.

Worum Sie bitten

Stellen Sie das Abschlussprojekt Adaptive Operations Console fertig: ein eigenes Theme mit benannten Tokens und mindestens vier semantischen Appearances, bereichsbezogene CssClass-Nutzung, eine Desktop-Shell mit Flow-, Table- und Flex-Bereichen, eine ClientProfiles.json mit Verhalten für Smartphone, Tablet, Small Desktop und Desktop, responsive Eigenschaften plus ein idempotenter ResponsiveProfileChanged-Handler und eine im Testmodus sichtbare Profilanzeige. Führen Sie das Barrierefreiheits-Review für Fokusrahmen, ungültige Zustände, Kontrast, Touch-Ziele und Tab-Reihenfolge durch, füllen Sie die responsive QA-Matrix mit Screenshots für jedes Profil aus und schreiben Sie die Architekturnotiz und das Grounding-Paket, die jede Theme-, Layout- und Profilentcheidung erklären.

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

- ✓ Fertige Adaptive Operations Console mit eigenem Theme, semantischen Appearances, bereichsbezogenem CSS sowie Flow-, Table- und Flex-Bereichen
- ✓ ClientProfiles.json, responsive Eigenschaften und ein idempotenter ResponsiveProfileChanged-Handler mit Profilanzeige im Testmodus
- ✓ Barrierefreiheits-Review zu Fokusrahmen, ungültigen Zuständen, Kontrast, Touch-Zielen und Tab-Reihenfolge
- ✓ Responsive QA-Matrix mit Desktop-, Tablet- und Smartphone-Screenshots in beiden Ausrichtungen
- ✓ Architekturnotiz und KI-Grounding-Paket, die jede Theme-, Layout- und Profilentcheidung erklären

Warum es so aufgebaut ist

Die Arbeit aus sechs Modulen steckt jetzt in einer Solution: ein von einem Basis-Theme abgeleitetes Theme, bereichsbezogenes CSS, eine Shell aus verschachtelten Containern, Flow-, Table- und Flex-Bereiche, eine ClientProfiles.json und ein Profil-Handler. Was daraus ein Produkt macht, ist kein weiteres Feature. Es ist der Nachweis, dass die Konsole in jedem Profil benutzbar ist, dass ein Tastatur- oder Touch-Benutzer sie bedienen kann, dass eine Größenänderung keine teuren Neuaufbauten auslöst und dass der nächste Entwickler oder das nächste Wisej.NET-Upgrade das visuelle System nicht stillschweigend brechen kann. Dieses Modul ist dieser Nachweis, dazu das Grounding-Paket, mit dem ein KI-Assistent Fragen zu Ihren Entscheidungen beantworten kann.

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
Ansichten des Abschlussprojekts	Die fertige Adaptive Operations Console: eigenes Theme, semantische Appearances, bereichsbezogenes CSS, eine Desktop-Shell, eine adaptive Tablet-Shell, eine Smartphone-Shell, ein Grid, ein Editor, Validierungs-Feedback, Navigation und eine im Testmodus sichtbare Profilanzeige.
visuelles Barrierefreiheits-Review	Ein Durchgang über Fokusrahmen, deaktivierte und ungültige Zustände, Kontrast, Touch-Abstände, Tooltips an reinen Symbol-Buttons und die Tab-Reihenfolge der Tastatur – durch Bedienen des Bildschirms, nicht durch Lesen des Themes.
idempotenter Profil-Handler	Ein ResponsiveProfileChanged -Handler, der die endgültigen Werte für das aktuelle Profil zuweist, sodass ein zweiter Durchlauf oder ein Durchlauf nach einem Neuladen exakt dieselbe UI ergibt.
Layout-Stabilität	MinimumSize und MaximumSize verhindern, dass Bereiche zusammenfallen, und AutoSize bleibt von großen Container-Hierarchien fern, sodass Größenänderungen vorhersehbar und günstig sind.
Theme-Governance	Theme-JSON, bereichsbezogenes Stylesheet und ClientProfiles.json stehen unter Versionskontrolle, werden als Diffs geprüft, als Tokens dokumentiert und nie innerhalb eines integrierten Themes bearbeitet.
upgradesichere Anpassung	Dokumentierte Appearances, Theme-Builder-Arbeitsabläufe und serverseitige Eigenschaften überstehen Upgrades; CSS-Selektoren, die vom internen DOM eines Widgets abhängen, möglicherweise nicht – daher werden sie eng begrenzt, dokumentiert oder vermieden.
responsive QA-Matrix und Grounding-Paket	Die Matrix hält pro Szenario Breiten, Ausrichtungen, erwartetes Profil und Prüfungen fest; das Grounding-Paket hält den Entscheidungsbaum, die Antwortregeln und die Quellenübersicht fest, aus denen ein Assistent antwortet.

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 7 geht. Sie können [module-07.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 7 – „Mobilitäugliches Abschlussprojekt, Barrierefreiheit, Performance und Governance im Produktivbetrieb“ – 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-07.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: Stellen Sie das Abschlussprojekt Adaptive Operations Console fertig: ein eigenes Theme mit benannten Tokens und mindestens vier semantischen Appearances, bereichsbezogene `CssClass`-Nutzung, eine Desktop-Shell mit Flow-, Table- und Flex-Bereichen, eine `ClientProfiles.json` mit Verhalten für Smartphone, Tablet, Small Desktop und Desktop, responsive Eigenschaften plus ein idempotenter `ResponsiveProfileChanged`-Handler und eine im Testmodus sichtbare Profilanzeige. Führen Sie das Barrierefreiheits-Review für Fokusrahmen, ungültige Zustände, Kontrast, Touch-Ziele und Tab-Reihenfolge durch, füllen Sie die responsive QA-Matrix mit Screenshots für jedes Profil aus und schreiben Sie die Architekturnotiz und das Grounding-Paket, die jede Theme-, Layout- und Profilentcheidung erklären.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - Fertige Adaptive Operations Console mit eigenem Theme, semantischen Appearances, bereichsbezogenem CSS sowie Flow-, Table- und Flex-Bereichen
 - `ClientProfiles.json`, responsive Eigenschaften und ein idempotenter `ResponsiveProfileChanged`-Handler mit Profilanzeige im Testmodus
 - Barrierefreiheits-Review zu Fokusrahmen, ungültigen Zuständen, Kontrast, Touch-Zielen und Tab-Reihenfolge
 - Responsive QA-Matrix mit Desktop-, Tablet- und Smartphone-Screenshots in beiden Ausrichtungen
 - Architekturnotiz und KI-Grounding-Paket, die jede Theme-, Layout- und Profilentcheidung erklären
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:

- Ihr `ResponsiveProfileChanged` -Handler läuft nach einem Neuladen des Browsers zweimal für

das Tablet-Profil. Welche Ihrer Zeilen garantiert, dass der zweite Durchlauf nichts ändert, und was sähe der Benutzer, wenn diese Zeile fehlte?

- Ein Reviewer bittet Sie, den Fokusrahmen von action-button zu entfernen, weil er „altmodisch aussieht“. Was ändern Sie stattdessen, welches Token steuert es, und welche zwei anderen Zustände prüfen Sie danach erneut im Theme Builder?
- Der Header des Ticket-Grids braucht eine Anpassung, die nur ein Selektor auf das interne DOM des Widgets erreicht. Was versuchen Sie zuerst, was muss zutreffen, bevor der Selektor erlaubt ist, und was schreiben Sie auf, damit das nächste Upgrade das Team nicht überrascht?

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:

- ✓ Ihr ResponsiveProfileChanged -Handler läuft nach einem Neuladen des Browsers zweimal für das Tablet-Profil. Welche Ihrer Zeilen garantiert, dass der zweite Durchlauf nichts ändert, und was sähe der Benutzer, wenn diese Zeile fehlte?
- ✓ Ein Reviewer bittet Sie, den Fokusrahmen von action-button zu entfernen, weil er „altmodisch aussieht“. Was ändern Sie stattdessen, welches Token steuert es, und welche zwei anderen Zustände prüfen Sie danach erneut im Theme Builder?
- ✓ Der Header des Ticket-Grids braucht eine Anpassung, die nur ein Selektor auf das interne DOM des Widgets erreicht. Was versuchen Sie zuerst, was muss zutreffen, bevor der Selektor erlaubt ist, und was schreiben Sie auf, damit das nächste Upgrade das Team nicht überrascht?

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 7 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 7 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 7](#).



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.

- ✓ **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-07.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."*
- ✓ **Tauschen Sie ein Teil aus, ohne die Schnittstellen zu verschieben.** *"Wähle den Teil dieses AdaptiveOps-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.