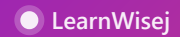


KI-Coding-Übung

Erstellen Sie das Beispielprojekt GlobalDesk 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 „Browser-Kultur, Sitzungskultur und Sprachwechsel zur Laufzeit“ 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

Geben Sie GlobalDesk einen Sprachwechsel, der die Sache zu Ende bringt. Setzen Sie culture in Default.json auf auto, damit die Sitzung mit der Browsersprache beginnt, fügen Sie eine Sprach-ComboBox mit Englisch, Deutsch und einer Sprach-Regions-Kultur hinzu und setzen Sie Application.CurrentCulture in deren SelectedIndexChanged-Handler aus CultureInfo.GetCultureInfo, ohne die Auswahl jemals in einem statischen Feld abzulegen. Behandeln Sie Application.CultureChanged, um eine Methode ApplyTextResources aufzurufen, die jede ressourcengestützte Beschriftung auffrischt, und eine Methode UpdateCulturePreview, die die Datums-, Zahlen- und Währungsbeispiele neu formatiert, und bauen Sie im selben Handler den im Designer lokalisierten Kundeneditor neu auf. Prüfen Sie dieselbe Seite über den URL-Parameter lang und schreiben Sie dann eine Lab-Notiz darüber, was ein statisches Kulturfeld bei einer zweiten, gleichzeitig angemeldeten Person anrichten würde.

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

- ✓ Default.json mit culture auf auto, dazu eine Notiz, wann eine feste Kultur die richtige Wahl ist
- ✓ Handler der Sprach-ComboBox, der Application.CurrentCulture aus CultureInfo.GetCultureInfo setzt
- ✓ Handler für Application.CultureChanged, der ApplyTextResources und UpdateCulturePreview aufruft
- ✓ Im Designer lokalisierter Editor, der nach dem Wechsel neu aufgebaut wird, damit keine Beschriftung in der alten Sprache stehen bleibt
- ✓ Prüfung über den URL-Parameter lang und eine Notiz, warum die Kultur nicht in einem statischen Feld leben darf

Warum es so aufgebaut ist

Nach Modul 3 hat GlobalDesk eine neutrale Ressourcendatei, eine deutsche daneben, einen Texts -Helper, einen im Designer lokalisierten Kundeneditor und sogar eine eigene Systembeschriftung von Wisej. Worauf es noch keine Antwort hat, ist die Frage, die ein echter Benutzer zuerst stellt: Wie bekomme ich das in meiner Sprache, und was passiert mit dem Bildschirm, wenn ich danach frage? Alles bisher wurde während des Bauens der Anwendung entschieden; in diesem Modul geht es um die beiden Momente, in denen die Entscheidung tatsächlich fällt — wenn eine Sitzung startet und wenn ein Benutzer mittendrin seine Meinung ändert.

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
automatische Browsererkennung	Wisej.NET kann die vom Browser gemeldete Sprache lesen und die Sitzung damit starten, sodass jeder Benutzer eine sinnvolle Kultur erhält, ohne dass eine serverweite Einstellung für alle entscheidet.
culture in Default.json	Die Anwendungseinstellung, die entscheidet, woher die Kultur einer neuen Sitzung kommt; sie steht standardmäßig auf auto, und ein fester Name wie de-DE bindet stattdessen jede Sitzung an ein Gebietsschema.
rightToLeft in Default.json	Die Nachbareinstellung, deren Wert auto derjenigen Kultur folgt, bei der die Sitzung schließlich landet; Modul 5 baut darauf auf, aber Sie begegnen ihr hier in derselben Datei.
Application.CurrentCulture	Liest oder setzt die Kultur der aktuellen Wisej.NET-Sitzung — genau der Geltungsbereich, den eine Mehrbenutzer-Webanwendung braucht: Die Wahl eines Benutzers darf niemals den Bildschirm eines anderen neu zeichnen.
URL-Parameter lang	Eine unterstützte Möglichkeit, beim Einstieg eine Sprache anzufordern, zum Beispiel?lang=de, wodurch sich Testlinks und sprachspezifische Einstiegs-URLs mühelos erzeugen lassen.
Application.CultureChanged	Wird ausgelöst, wenn die aktuelle Kultur von der abweicht, mit der die Sitzung gestartet ist; die eine Stelle, an der Sie ressourcengestützten Text erneut anwenden und alles bereits Sichtbare neu formatieren.
im Designer lokalisierte Controls	Ihre lokalisierten Werte werden beim Erstellen der Controls angewendet, deshalb schreibt ein späterer Kulturwechsel sie nicht neu — Sie bauen den Container neu auf oder setzen die Werte im Code erneut.

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 GlobalDesk 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 – „Browser-Kultur, Sitzungskultur und Sprachwechsel zur Laufzeit“ – des LearnWisej-Kurses „Lokalisierung in Wisej.NET“ 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/localization-in-wisej-net/specs/module-04.md>

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

1. Projekt „GlobalDesk“, 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: Geben Sie GlobalDesk einen Sprachwechsel, der die Sache zu Ende bringt. Setzen Sie `culture` in `Default.json` auf `auto`, damit die Sitzung mit der Browsersprache beginnt, fügen Sie eine `Sprach-ComboBox` mit Englisch, Deutsch und einer `Sprach-Regions-Kultur` hinzu und setzen Sie `Application.CurrentCulture` in deren `SelectedIndexChanged`-Handler aus `CultureInfo.GetCultureInfo`, ohne die Auswahl jemals in einem statischen Feld abzulegen. Behandeln Sie `Application.CultureChanged`, um eine Methode `ApplyTextResources` aufzurufen, die jede ressourcengestützte Beschriftung auffrischt, und eine Methode `UpdateCulturePreview`, die die Datums-, Zahlen- und Währungsbeispiele neu formatiert, und bauen Sie im selben Handler den im Designer lokalisierten `Kundeneditor` neu auf. Prüfen Sie dieselbe Seite über den URL-Parameter `lang` und schreiben Sie dann eine Lab-Notiz darüber, was ein statisches Kulturfeld bei einer zweiten, gleichzeitig angemeldeten Person anrichten würde.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - `Default.json` mit `culture` auf `auto`, dazu eine Notiz, wann eine feste Kultur die richtige Wahl ist
 - Handler der `Sprach-ComboBox`, der `Application.CurrentCulture` aus `CultureInfo.GetCultureInfo` setzt
 - Handler für `Application.CultureChanged`, der `ApplyTextResources` und `UpdateCulturePreview` aufruft
 - Im Designer lokalisierter Editor, der nach dem Wechsel neu aufgebaut wird, damit keine Beschriftung in der alten Sprache stehen bleibt
 - Prüfung über den URL-Parameter `lang` und eine Notiz, warum die Kultur nicht in einem statischen Feld leben darf
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:

- Ihre Methode `ApplyTextResources` frischt das Dashboard einwandfrei auf, der Kundeneditor behält nach einem Wechsel zu Deutsch aber seine englischen Beschriftungen. Erklären Sie, was an diesen Beschriftungen anders ist, und nennen Sie zwei Strategien, die das beheben.
- Ein Kollege schlägt vor, die gewählte `CultureInfo` in einem statischen Feld zwischenzuspeichern, „um sie nicht zweimal nachschlagen zu müssen“. Beschreiben Sie, was ein zweiter Benutzer, im selben Moment aus einem anderen Land angemeldet, tatsächlich sehen würde.
- Die Kulturvorschau zeigt einen Währungsbetrag. Was hat sich nach einem Wechsel von en-US zu de-DE im Speicher geändert und was nur auf dem Bildschirm – und welches von beidem würden Sie je in die Datenbank schreiben?

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:

- ✓ Ihre Methode `ApplyTextResources` frischt das Dashboard einwandfrei auf, der Kundeneditor behält nach einem Wechsel zu Deutsch aber seine englischen Beschriftungen. Erklären Sie, was an diesen Beschriftungen anders ist, und nennen Sie zwei Strategien, die das beheben.
- ✓ Ein Kollege schlägt vor, die gewählte `CultureInfo` in einem statischen Feld zwischenzuspeichern, „um sie nicht zweimal nachschlagen zu müssen“. Beschreiben Sie, was ein zweiter Benutzer, im selben Moment aus einem anderen Land angemeldet, tatsächlich sehen würde.
- ✓ Die Kulturvorschau zeigt einen Währungsbetrag. Was hat sich nach einem Wechsel von en-US zu de-DE im Speicher geändert und was nur auf dem Bildschirm — und welches von beidem würden Sie je in die Datenbank schreiben?

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.

- ✓ **Tauschen Sie ein Teil aus, ohne die Schnittstellen zu verschieben.** *"Wähle den Teil dieses GlobalDesk-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."*
- ✓ **Machen Sie es fit für einen schlechten Tag.** *"Untersuche dieses GlobalDesk-Projekt auf Fehlerszenarien, die es noch nicht behandelt – fehlerhafte oder fehlende Eingaben, eine Abhängigkeit, die langsam oder nicht erreichbar ist, Arbeit, die eintrifft, nachdem der Benutzer die Seite verlassen hat, und zwei Aktionen, die sich gegenseitig in die Quere kommen. Sag mir für jedes zuerst, ob es in Wisej.NET 4 ein echtes Risiko ist und warum, und schreib erst dann die Korrektur. Wo du entscheidest, dass nichts behandelt werden muss, sag das und begründe es, statt defensiven Code hinzuzufügen."*
- ✓ **Ergänzen Sie den zweiten Fall.** *"Erweitere dieses GlobalDesk-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."*



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.