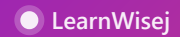


KI-Coding-Übung

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.



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

Führen Sie eine ehrliche Übersetzungsrunde an GlobalDesk durch. Öffnen Sie die Projektmappe im ResX Resource Manager, fügen Sie Italienisch als neue Sprachspalte hinzu und listen Sie mit dem Filter für nicht übersetzte Einträge jeden Schlüssel auf, der noch keinen Wert hat. Versehen Sie die drei Schlüssel, deren Bedeutung ohne Kontext mehrdeutig ist, mit Kommentaren – beginnend mit der einwortigen Beschriftung Open –, übersetzen Sie dann die italienischen Werte oder lassen Sie sie maschinell entwerfen, und markieren Sie den Produktnamen und die beiden technischen Bezeichner als invariant, damit sie niemand übersetzt. Exportieren Sie die neutrale und die italienische Spalte, importieren Sie sie wieder und prüfen Sie den entstandenen .resx-Diff, bevor Sie ihn als eine einzige überprüfbare Änderung committen. Schließen Sie mit einer einseitigen Workflow-Notiz ab, die den Visual Studio-Editor, den ResX Resource Manager und eine Cloud-Plattform für ein Team vergleicht, dessen Übersetzer keine Entwickler sind.

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

- ✓ Strings.it.resx, im ResX Resource Manager erstellt, ohne einen einzigen nicht übersetzten Schlüssel
- ✓ Ressourcenkommentare zu mindestens drei mehrdeutigen Schlüsseln, damit ein Übersetzer den Kontext hat
- ✓ Invariant-Markierungen für den Produktnamen und die Bezeichner, die nicht übersetzt werden dürfen
- ✓ Export, erneuter Import und ein geprüfter.resx-Diff, als eine Änderung committet
- ✓ Workflow-Notiz, die den Visual Studio-Editor, den ResX Resource Manager und ein Cloud-TMS vergleicht

Warum es so aufgebaut ist

GlobalDesk trägt inzwischen eine neutrale Resources/Strings.resx, eine deutsche Strings.de.resx, Designer-Ressourcen auf dem Kundeneditor und eine Resources-de.resx für die Systembeschriftungen von Wisej. Italienisch mit dem Visual Studio-Editor hinzuzufügen heißt, diese Dateien Registerkarte für Registerkarte zu öffnen und Schlüssellisten nach Augenmaß zu vergleichen — und die Person, die Italienisch spricht, ist meist nicht die Person, die die Projektmappe öffnen kann. Dieses Modul macht aus der Übersetzung einen Workflow mit sichtbarer Lückenliste, einer Übergabe und einem Prüfschritt.

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
Matrixsicht	Schlüssel in den Zeilen und Sprachen in den Spalten, sodass ein fehlender italienischer Wert und ein abgedrifteter deutscher Begriff auf einen Blick sichtbar sind.
ResX Resource Manager	Ein quelloffenes.NET-Lokalisierungswerkzeug, als Visual Studio-Erweiterung und als eigenständige Anwendung, das die RESX-Ressourcen einer Projektmappe in einem zentralen Raster zeigt.
Sprachspalte	Eine Zielsprache hinzuzufügen erzeugt die kulturspezifische Datei neben der neutralen und gibt Ihnen eine leere Spalte zum Füllen; die Schlüssel ändern sich nie.
Filter für nicht Übersetztes	Listet nur die Einträge ohne Wert in der gewählten Sprache auf, sodass aus „sind wir fertig?“ eine Zahl wird statt einer Meinung.
Ressourcenkommentar	Eine Notiz neben dem Wert, die einem Übersetzer sagt, ob Open ein Verb, ein Adjektiv, ein Ticketzustand oder ein Befehl ist.
Invariante Zeichenfolge	Eine Markierung auf Einträgen, die niemals übersetzt werden dürfen, etwa dem Produktnamen und technischen Bezeichnungen.
Export-Import-Umlauf	Spalten, die an einen Übersetzer oder eine RESX-fähige Plattform gehen und als kulturspezifische.resx -Dateien zurückkommen, die der Build wie jede andere Ressource behandelt.

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



Claude verwenden

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

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 6 – „Übersetzungsworkflow: ResX Resource Manager und weitere Werkzeuge“ – 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-06.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: Führen Sie eine ehrliche Übersetzungsrunde an GlobalDesk durch. Öffnen Sie die Projektmappe im ResX Resource Manager, fügen Sie Italienisch als neue Sprachspalte hinzu und listen Sie mit dem Filter für nicht übersetzte Einträge jeden Schlüssel auf, der noch keinen Wert hat. Versehen Sie die drei Schlüssel, deren Bedeutung ohne Kontext mehrdeutig ist, mit Kommentaren – beginnend mit der einwortigen Beschriftung Open –, übersetzen Sie dann die italienischen Werte oder lassen Sie sie maschinell entwerfen, und markieren Sie den Produktnamen und die beiden technischen Bezeichner als invariant, damit sie niemand übersetzt. Exportieren Sie die neutrale und die italienische Spalte, importieren Sie sie wieder und prüfen Sie den entstandenen .resx-Diff, bevor Sie ihn als eine einzige überprüfbare Änderung committen. Schließen Sie mit einer einseitigen Workflow-Notiz ab, die den Visual Studio-Editor, den ResX Resource Manager und eine Cloud-Plattform für ein Team vergleicht, dessen Übersetzer keine Entwickler sind.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - Strings.it.resx, im ResX Resource Manager erstellt, ohne einen einzigen nicht übersetzten Schlüssel
 - Ressourcenkommentare zu mindestens drei mehrdeutigen Schlüsseln, damit ein Übersetzer den Kontext hat
 - Invariant-Markierungen für den Produktnamen und die Bezeichner, die nicht übersetzt werden dürfen
 - Export, erneuter Import und ein geprüfter.resx-Diff, als eine Änderung committet
 - Workflow-Notiz, die den Visual Studio-Editor, den ResX Resource Manager und ein Cloud-TMS vergleicht
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 italienische Spalte ist voll und die Anwendung sieht unter it-IT richtig aus, aber

ein Kollege behauptet, zwei Schlüssel seien nie übersetzt worden. Wie können beide Aussagen zugleich wahr sein, und welche Funktion des Werkzeugs entscheidet den Streit?

- Ein italienischer maschineller Entwurf hat Ultimo ordine: {1} il {0} für einen Schlüssel geliefert, dessen neutraler Wert Last order: {0} on {1} lautet. Was geht kaputt, wann geht es kaputt, und was hätte es vor dem Commit aufgefangen?
- Ihr Team hat fünf Sprachen, zwei davon übersetzt von einer Agentur, die nie Visual Studio geöffnet hat. Welchen der drei Workflows würden Sie einführen, und was müsste sich ändern, bevor Sie zum nächstgrößeren wechseln?

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 italienische Spalte ist voll und die Anwendung sieht unter it-IT richtig aus, aber ein Kollege behauptet, zwei Schlüssel seien nie übersetzt worden. Wie können beide Aussagen zugleich wahr sein, und welche Funktion des Werkzeugs entscheidet den Streit?
- ✓ Ein italienischer maschineller Entwurf hat Ultimo ordine: {1} il {0} für einen Schlüssel geliefert, dessen neutraler Wert Last order: {0} on {1} lautet. Was geht kaputt, wann geht es kaputt, und was hätte es vor dem Commit aufgefangen?
- ✓ Ihr Team hat fünf Sprachen, zwei davon übersetzt von einer Agentur, die nie Visual Studio geöffnet hat. Welchen der drei Workflows würden Sie einführen, und was müsste sich ändern, bevor Sie zum nächstgrößeren wechseln?

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.

- ✓ **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."

- ✓ **Beweisen Sie, dass es wirklich funktioniert.** "Geh dieses Projekt Ergebnis für Ergebnis anhand der Modulspezifikation unter <https://learnwisej.com/assets/courses/localization-in-wisej-net/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 GlobalDesk-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.