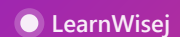


KI-Coding-Übung

Erstellen Sie das Beispielprojekt OperationsConsole 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

Erstellen Sie den Dashboard-Tab der Operations Console: ein ChartJS-Linien- oder Balkendiagramm, dessen Labels und DataSets aus einem Dashboard-ViewModel stammen, eine ProgressBar oder gaugeähnliche Anzeige für einen Statuswert, eine Inhaltsvorschau mit PdfViewer oder HtmlPanel, einen Upload-Ablauf, einen Karten- oder Inhaltsplatzhalter und eine einzige Aktualisierungsmethode, die jedes Dashboard-Control aus einem einzigen Modell aktualisiert.

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

- ✓ Dashboard-Tab mit einem ChartJS-Linien- oder Balkendiagramm
- ✓ Diagramm-Labels und DataSets aus einem Dashboard-ViewModel
- ✓ ProgressBar oder gaugeähnliche Anzeige für einen Statuswert
- ✓ Inhaltsvorschau mit PdfViewer oder HtmlPanel sowie ein Upload-Ablauf
- ✓ Eine Aktualisierungsmethode, die jedes Dashboard-Control aktualisiert

Warum es so aufgebaut ist

Dashboards sind nicht bloß Diagramme. Ein Dashboard ist eine Komposition von Controls, die aus Daten Entscheidungen macht, und der Dashboard-Tab der Operations Console wird der Bildschirm sein, den ein Service-Manager jeden Morgen als Erstes öffnet. Wisej.NET bietet native Inhalts- und Medien-Controls sowie Erweiterungen für Diagramme, Karten, Dokumentanzeige und Gauges; die Fähigkeit auf fortgeschrittenem Niveau besteht darin, zu erkennen, was in das Control-Modell von Wisej.NET gehört, was besser an eine spezialisierte JavaScript-Bibliothek delegiert und als Erweiterung oder Widget gekapselt wird, und wie der ganze Bildschirm als Einheit aktualisiert wird statt als Haufen unabhängig nachladender Visualisierungen.

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
Dashboard	Eine Komposition von Controls, deren einzige Aufgabe es ist, eine Entscheidungsfrage zu beantworten; ein visuelles Element, das niemandem bei einer Entscheidung hilft, ist Ballast.
ChartJS-Erweiterung	Ein NuGet-Paket, das Chart.js als serverseitige Komponente bereitstellt: Der Server besitzt den Diagrammzustand, der Browser rendert die Darstellung, und das Diagramm nimmt am selben Event-Modell teil wie jedes andere Control.
Labels und DataSets	Die beiden Hälften der Diagrammdaten; Labels benennen die Punkte entlang der Achse, und jedes DataSet trägt eine Datenreihe, daher werden sie immer gemeinsam gesetzt.
Dashboard-ViewModel	Ein auf die Anzeige zugeschnittenes Objekt wie DashboardModel, das Monate, Anzahlen und einen Erledigungswert enthält, getrennt von jeder Datenbank-Entität und vom Code, der es zeichnet.
Browsergrenze	Der Client ist der Browser, und der Server ist der Ort, an dem Ihr Wisej.NET-Code läuft; Dateien, PDFs, HTML und Kartenkacheln überqueren diese Grenze in der einen oder anderen Richtung.
Upload vs. lokaler Pfad	Upload überträgt eine Datei vom Browser auf den Server; das ist nicht dasselbe wie das Lesen eines Desktop-Pfads, denn der Server kann niemals die Festplatte des Benutzers durchsuchen.
modellbasierte Aktualisierung	Ein Serviceaufruf liefert ein vollständiges Dashboard-Modell, und eine Methode aktualisiert daraus jedes Control, gebündelt und mit sichtbarem Zeitstempel.

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 OperationsConsole 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 – „Diagramme, Dashboards, Inhalte, Medien und Dokumente“ – des LearnWisej-Kurses „Die Control-Bibliothek meistern“ 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/mastering-the-control-library/specs/module-06.md>

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

1. Projekt „OperationsConsole“, 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: Erstellen Sie den Dashboard-Tab der Operations Console: ein ChartJS-Linien- oder Balkendiagramm, dessen Labels und DataSets aus einem Dashboard-ViewModel stammen, eine ProgressBar oder gaugeähnliche Anzeige für einen Statuswert, eine Inhaltsvorschau mit PdfViewer oder HtmlPanel, einen Upload-Ablauf, einen Karten- oder Inhaltsplatzhalter und eine einzige Aktualisierungsmethode, die jedes Dashboard-Control aus einem einzigen Modell aktualisiert.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - Dashboard-Tab mit einem ChartJS-Linien- oder Balkendiagramm
 - Diagramm-Labels und DataSets aus einem Dashboard-ViewModel
 - ProgressBar oder gaugeähnliche Anzeige für einen Statuswert
 - Inhaltsvorschau mit PdfViewer oder HtmlPanel sowie ein Upload-Ablauf
 - Eine Aktualisierungsmethode, die jedes Dashboard-Control aktualisiert
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:

- Welches Control auf Ihrem Dashboard-Tab leistet die wichtigste Arbeit, und welche Frage bliebe unbeantwortet, wenn Sie es entfernten?
- Was würde zuerst brechen, wenn der Ticket-Datensatz zehnmal größer würde, und warum schützt es das Diagramm, wenn der Service Aggregate liefert?
- Wo befindet sich die hochgeladene Datei in jedem Schritt des Upload-Ablaufs, und warum kann der Server niemals einen Pfad öffnen, den der Benutzer eingetippt hat?

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:

- ✓ Welches Control auf Ihrem Dashboard-Tab leistet die wichtigste Arbeit, und welche Frage bliebe unbeantwortet, wenn Sie es entfernten?
- ✓ Was würde zuerst brechen, wenn der Ticket-Datensatz zehnmal größer würde, und warum schützt es das Diagramm, wenn der Service Aggregate liefert?
- ✓ Wo befindet sich die hochgeladene Datei in jedem Schritt des Upload-Ablaufs, und warum kann der Server niemals einen Pfad öffnen, den der Benutzer eingetippt hat?

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 `Mastering the Control Library 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.

- ✓ **Machen Sie es fit für einen schlechten Tag.** *"Untersuche dieses OperationsConsole-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 OperationsConsole-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/mastering-the-control-library/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."*



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.