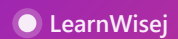


KI-Coding-Übung

Erstellen Sie das Beispielprojekt OpsMonitor 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.



Dasselbe Modul – so gebaut, wie Sie in der Praxis wirklich bauen.

Alle anderen Abschnitte dieses Moduls erklären, wie „Daten-Controls, Browser-Payloads und große UI-Flächen“ funktioniert. Dieser fordert Sie auf, es herzustellen: indem Sie einen Assistenten mit der Spezifikation des Moduls steuern und dann prüfen, ob das Ergebnis läuft und tut, was die Lab-Aufgabe beschreibt. Es ist der Arbeitsablauf, den Sie in echten Projekten nutzen werden – geübt an etwas, dessen richtige Lösung Sie bereits kennen.

Worum Sie bitten

Verkleinern Sie die beiden größten UI-Flächen in WisejPerfLab. Das Ticket-Grid bindet 50.000 vollständige Entitäten mit Navigationseigenschaften: Ersetzen Sie das Binding durch eine TicketGridRow-Projektion nur der sichtbaren Spalten, setzen Sie VirtualMode und RowCount und liefern Sie die Zellen aus einem CellValueNeeded-Handler, hinter dem ein Service mit seitenweisen Abfragen steht. Der Kundenbaum baut beim Start jeden Knoten auf: Laden Sie die Kindknoten stattdessen beim Aufklappen und verwenden Sie für zugeklappte Zweige einen Platzhalter mit der Anzahl. Messen Sie jede Maske vorher und nachher mit CPU Usage und .NET Object Allocation auf dem Server sowie mit dem Netzwerk-Panel des Browsers für die Größe des Update-Payloads.

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

- ✓ TicketGridRow-Projektion statt vollständiger Entitäten gebunden, nur mit den angezeigten Spalten
- ✓ Grid im Virtual Mode mit RowCount und einem CellValueNeeded-Handler, der aus einem Service mit seitenweisen Abfragen bedient wird
- ✓ Kundenbaum, der Kindknoten beim Aufklappen lädt, mit einem Platzhalter mit der Anzahl für zugeklappte Zweige
- ✓ CPU- und Allokationswerte vorher und nachher für die Szenarien Grid laden und Baum aufklappen
- ✓ Netzwerkbelege aus dem Browser, die die Größe des Update-Payloads vorher und nachher vergleichen, mit dem Hinweis zur Komprimierung

Warum es so aufgebaut ist

Große UI-Flächen sind die häufigste Ursache für Performance-Probleme in Wisej.NET – und die Kategorie, die der Profiler nicht vollständig sehen kann. Das Ticket-Grid in WisejPerfLab bindet fünfzigtausend Entitäten mit ihren Navigationseigenschaften; der Kundenbaum baut beim Start jeden Knoten auf. Auf dem Desktop war dieses Muster vertretbar, weil Control und Daten im selben Prozess lebten wie die Maske. Hier zahlt derselbe Code vierfach: Serverspeicher, um die Objekte zu halten, Server-CPU, um den UI-Zustand aufzubauen und zu serialisieren, Netzwerk, um das Update zu transportieren, und Browserzeit, um es zu rendern. In diesem Modul ist die beste Optimierung meist kein schnellerer Algorithmus, sondern weniger gleichzeitig existierende UI.

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
Projektionszeile	Ein kleiner Typ, der nur die Spalten enthält, die eine Maske anzeigt, und statt des Entitätsgraphen gebunden wird – so tragen weder Server noch Browser Daten mit sich, die niemand ansieht.
Virtual Mode	Das Grid hält eine Zeilenanzahl statt Zeilenobjekten und fragt Zellwerte erst ab, wenn sie gebraucht werden. Damit entfallen die Objekte pro Zeile und das Rendern von Zeilen, zu denen niemand gescrollt hat.
CellValueNeeded	Das Event, das im Virtual Mode einen Wert bei Bedarf liefert; es läuft sehr häufig, ist also konstruktionsbedingt ein Hot Path und muss schnell zurückkehren.
serverseitiges Paging und Filtern	Die Datenbank nach genau der Seite fragen, die der Benutzer gerade ansieht, statt alles zu laden und im UI-Code zu filtern.
verzögertes Laden von Knoten	Kindknoten erst aufbauen, wenn ein Zweig aufgeklappt wird, mit einem Platzhalter mit der Anzahl, solange er zugeklappt ist – statt beim Start die gesamte Hierarchie zu materialisieren.
Update-Payload	Die serialisierte Differenz, die der Server nach einem Handler sendet; große oder häufige Payloads kosten Netzwerk- und Browserzeit, die kein Visual Studio-Tool ausweist.
Komprimierung macht Arbeit nicht kostenlos	Wisej.NET komprimiert HTTP- und WebSocket-Datenverkehr. Das verringert die Bytes, macht ein unnötiges Update aber nicht günstig in Erzeugung, Übertragung oder Rendering.

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 5 geht. Sie können [module-05.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 OpsMonitor 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 5 – „Daten-Controls, Browser-Payloads und große UI-Flächen“ – des LearnWisej-Kurses „Performance und Profiling“ 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/performance-and-profiling/specs/module-05.md>

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

1. Projekt „OpsMonitor“, 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: Verkleinern Sie die beiden größten UI-Flächen in `WisejPerfLab`. Das Ticket-Grid bindet 50.000 vollständige Entitäten mit Navigationseigenschaften: Ersetzen Sie das Binding durch eine `TicketGridRow`-Projektion nur der sichtbaren Spalten, setzen Sie `VirtualMode` und `RowCount` und liefern Sie die Zellen aus einem `CellValueNeeded`-Handler, hinter dem ein Service mit seitenweisen Abfragen steht. Der Kundenbaum baut beim Start jeden Knoten auf: Laden Sie die Kindknoten stattdessen beim Aufklappen und verwenden Sie für zugeklappte Zweige einen Platzhalter mit der Anzahl. Messen Sie jede Maske vorher und nachher mit CPU Usage und .NET Object Allocation auf dem Server sowie mit dem Netzwerk-Panel des Browsers für die Größe des Update-Payloads.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - `TicketGridRow`-Projektion statt vollständiger Entitäten gebunden, nur mit den angezeigten Spalten
 - Grid im Virtual Mode mit `RowCount` und einem `CellValueNeeded`-Handler, der aus einem Service mit seitenweisen Abfragen bedient wird
 - Kundenbaum, der Kindknoten beim Aufklappen lädt, mit einem Platzhalter mit der Anzahl für zugeklappte Zweige
 - CPU- und Allokationswerte vorher und nachher für die Szenarien Grid laden und Baum aufklappen
 - Netzwerkbelege aus dem Browser, die die Größe des Update-Payloads vorher und nachher vergleichen, mit dem Hinweis zur Komprimierung
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:

- Der Virtual Mode hat die Zeilenobjekte beseitigt, aber Ihr CPU-Trace hat sich kaum verbessert. Was sagt Ihnen das darüber, wo die Kosten tatsächlich lagen, und welche Messung würden Sie sich als Nächstes ansehen?

- Ihr CellValueNeeded -Handler ist korrekt, aber bei schnellem Scrollen langsam. Nennen Sie drei Dinge, die er möglicherweise tut und die ein Handler pro Zelle niemals tun darf.
- Der komprimierte Payload beim Aufklappen des Baums hat sich nach Einführung des verzögerten Ladens kaum verändert, aber die Benutzer sagen, die Maske fühle sich schneller an. Geben Sie eine plausible Erklärung, die die Byte-Zahl allein verbergen würde.

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:

- ✓ Der Virtual Mode hat die Zeilenobjekte beseitigt, aber Ihr CPU-Trace hat sich kaum verbessert. Was sagt Ihnen das darüber, wo die Kosten tatsächlich lagen, und welche Messung würden Sie sich als Nächstes ansehen?
- ✓ Ihr CellValueNeeded -Handler ist korrekt, aber bei schnellem Scrollen langsam. Nennen Sie drei Dinge, die er möglicherweise tut und die ein Handler pro Zelle niemals tun darf.
- ✓ Der komprimierte Payload beim Aufklappen des Baums hat sich nach Einführung des verzögerten Ladens kaum verändert, aber die Benutzer sagen, die Maske fühle sich schneller an. Geben Sie eine plausible Erklärung, die die Byte-Zahl allein verbergen würde.

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 5 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 5 auf GitHub öffnen](#)

Oder klonen Sie alle Beispiele mit `git clone https://github.com/iceteagroup/LearnWisej-Samples` und öffnen Sie `Performance and Profiling Course/Module 5`.



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 OpsMonitor-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 OpsMonitor-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/performance-and-profiling/specs/module-05.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.