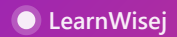


KI-Coding-Übung

Erstellen Sie das Beispielprojekt VisualOperationsStudio 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

Bauen Sie die Shell von VisualOperationsStudio und stellen Sie einen einzigen Telemetriewert gleichzeitig auf vier Oberflächen dar. Erstellen Sie eine VisualOperationsPage, die genau eine TelemetrySample-Modellinstanz hält, und zeigen Sie deren Wert dann (1) als schlichtes Label mit ProgressBar, (2) in einem Panel mit Paint-Handler, der einen Balken aus PaintEventArgs.Graphics und e.ClipRectangle zeichnet, (3) in einer Wisej.Web.Canvas, deren Redraw-Handler denselben Balken aus dem Modell neu aufbaut, und (4) als Off-Screen-Bitmap, die mit Graphics.FromImage gerendert und in einer PictureBox angezeigt wird. Verdrahten Sie einen Button, der den Modellwert ändert und alle vier Oberflächen über Invalidate() und ein Canvas-Redraw aktualisiert, ändern Sie dann die Browsergröße und halten Sie fest, welche Oberflächen das Canvas-Redraw aktualisiert, ändern Sie dann die Browsergröße und halten Sie fest, welche Oberflächen das überstehen und welche neu aufgebaut werden müssen.

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

- ✓ VisualOperationsPage mit einem TelemetrySample-Modell, aus dem jede Oberfläche liest
- ✓ Paint-Handler, der aus PaintEventArgs.Graphics innerhalb von e.ClipRectangle zeichnet
- ✓ Canvas-Redraw-Handler, der die gesamte Szene nach einer Größenänderung aus dem Modell neu aufbaut
- ✓ Off-Screen-Bitmap, die mit Graphics.FromImage gerendert und auf der Seite angezeigt wird
- ✓ SurfaceNotes.md mit der Notiz, was pro Oberfläche über das Netzwerk geht und welche Darstellungen eine Größenänderung überleben

Warum es so aufgebaut ist

VisualOperationsStudio muss denselben Maschinenmesswert an vier Stellen anzeigen: als Rundinstrument auf der Betriebsseite, als Gesundheitsbalken in einer Grid-Zeile, als Knoten auf einer Topologiekarte und als PNG im Anhang des nächtlichen Schichtberichts. Jedes davon heißt „eine Zahl zeichnen“, und doch führt

der Griff zu irgendeiner Zeichen-API, die man noch vom Desktop kennt, zu vier Stücken Code mit völlig unterschiedlichem Verhalten bei CPU, Bandbreite und Größenänderung. Dieses Modul stellt die Wahl der Oberfläche vor den Zeichencode, denn die Oberfläche entscheidet, wo die Pixel entstehen und was zum Browser wandert.

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
Paint / PaintEventArgs	Das Zeichenereignis auf Control-Ebene: Ihr C# läuft auf dem Server gegen das bereitgestellte Graphics, innerhalb des gemeldeten ClipRectangle, und Wisej.NET macht daraus ein Bild für den Browser.
DataGridView.CellPaint	Derselbe Zeichenmechanismus, auf eine Zelle begrenzt; die DataGridViewCellPaintEventArgs leiten sich von PaintEventArgs ab und ergänzen RowIndex und ColumnIndex.
UserPaint	Der Schalter, den eine Zelle oder eine ganze Spalte braucht, bevor das Grid dafür CellPaint auslöst – so bleibt das Zeichnen pro Spalte ausdrücklich gewählt statt gridweit aktiv.
Wisej.Web.Canvas	Eine 2D-Browseroberfläche im Immediate Mode, die das HTML5-Modell CanvasRenderingContext2D spiegelt; die Zeichenaufrufe führt der Client aus, sie werden nicht auf dem Server in ein Bild gerendert.
Redraw	Das Canvas-Event, in dem die gesamte Szene aus Ihrem Modell neu aufgebaut wird, denn die Canvas hält Pixel und verliert sie, sobald sie neu erstellt oder in der Größe geändert wird.
System.Drawing.Managed	Off-Screen-Rendering ganz ohne sichtbares Control: eine Bitmap erzeugen, ihr Graphics über Graphics.FromImage() holen und die Bild-Bytes für einen Bericht, einen Export oder einen Download behalten.
Invalidate()	Der Aufruf, der einem gezeichneten Control sagt, dass seine Darstellung veraltet ist; er aktualisiert das Client-Widget und die Oberfläche im Visual Studio Designer und gehört deshalb dorthin, wo sich der Zustand geändert hat, und nicht in Zeichenhelfer.

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 1 geht. Sie können [module-01.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 VisualOperationsStudio 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 1 – „Die Zeichenarchitektur von Wisej.NET“ – des LearnWisej-Kurses „Zeichnen und Grafik 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/drawing-and-painting/specs/module-01.md>

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

1. Projekt „VisualOperationsStudio“, 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: Bauen Sie die Shell von `VisualOperationsStudio` und stellen Sie einen einzigen Telemetriewert gleichzeitig auf vier Oberflächen dar. Erstellen Sie eine `VisualOperationsPage`, die genau eine `TelemetrySample`-Modellinstanz hält, und zeigen Sie deren Wert dann (1) als schlichtes Label mit `ProgressBar`, (2) in einem Panel mit `Paint`-Handler, der einen Balken aus `PaintEventArgs.Graphics` und `e.ClipRectangle` zeichnet, (3) in einer `Wisej.Web.Canvas`, deren `Redraw`-Handler denselben Balken aus dem Modell neu aufbaut, und (4) als `Off-Screen-Bitmap`, die mit `Graphics.FromImage` gerendert und in einer `PictureBox` angezeigt wird. Verdrahten Sie einen Button, der den Modellwert ändert und alle vier Oberflächen über `Invalidate()` und ein `Canvas-Redraw` aktualisiert, ändern Sie dann die Browsergröße und halten Sie fest, welche Oberflächen das überstehen und welche neu aufgebaut werden müssen.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - `VisualOperationsPage` mit einem `TelemetrySample`-Modell, aus dem jede Oberfläche liest
 - `Paint`-Handler, der aus `PaintEventArgs.Graphics` innerhalb von `e.ClipRectangle` zeichnet
 - `Canvas-Redraw`-Handler, der die gesamte Szene nach einer Größenänderung aus dem Modell neu aufbaut
 - `Off-Screen-Bitmap`, die mit `Graphics.FromImage` gerendert und auf der Seite angezeigt wird
 - `SurfaceNotes.md` mit der Notiz, was pro Oberfläche über das Netzwerk geht und welche Darstellungen eine Größenänderung überleben
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:

- Sie setzen den Messwert des Modells über den Button, vergessen aber den `Invalidate()` – Aufruf auf dem gezeichneten Panel. Label und Canvas aktualisieren sich. Erklären Sie genau, warum das Panel es nicht tut und was die Benutzerin so lange sieht, bis etwas anderes ein

Neuzeichnen erzwingt.

- Ihre erste Canvas-Fassung zeichnet den Balken einmal beim Laden der Seite statt aus Redraw. Beschreiben Sie, was die Benutzerin nach einer Größenänderung des Browsers sieht, und warum sich die Off-Screen- Bitmap im vierten Bereich anders verhält.
- Ihr Reviewer fragt, warum der erste Bereich – das schlichte Label und die ProgressBar – überhaupt im Lab vorkommt. Antworten Sie mit den Kosten, die die anderen drei Oberflächen zahlen und er nicht, und nennen Sie eine Anforderung, die diese Kosten rechtfertigen 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:

- ✓ Sie setzen den Messwert des Modells über den Button, vergessen aber den Invalidate() -Aufruf auf dem gezeichneten Panel. Label und Canvas aktualisieren sich. Erklären Sie genau, warum das Panel es nicht tut und was die Benutzerin so lange sieht, bis etwas anderes ein Neuzeichnen erzwingt.
- ✓ Ihre erste Canvas-Fassung zeichnet den Balken einmal beim Laden der Seite statt aus Redraw. Beschreiben Sie, was die Benutzerin nach einer Größenänderung des Browsers sieht, und warum sich die Off-Screen- Bitmap im vierten Bereich anders verhält.
- ✓ Ihr Reviewer fragt, warum der erste Bereich — das schlichte Label und die ProgressBar — überhaupt im Lab vorkommt. Antworten Sie mit den Kosten, die die anderen drei Oberflächen zahlen und er nicht, und nennen Sie eine Anforderung, die diese Kosten rechtfertigen 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.



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 VisualOperationsStudio-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/drawing-and-painting/specs/module-01.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 VisualOperationsStudio-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.