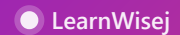


KI-Coding-Übung

Erstellen Sie das Beispielprojekt IconDesk 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

Erstellen Sie das Projekt IconDesk und geben Sie ihm ein Befehls-Panel mit den Buttons Open, Save, Print und Delete, von denen jeder sein Icon über einen anderen Mechanismus erhält. Laden Sie btnOpen aus einem serverseitigen System.Drawing.Image, weisen Sie btnSave.ImageSource ein benanntes Theme-Bild zu und lassen Sie btnPrint und btnDelete über ImageKey auf eine gemeinsame ImageList zeigen. Ergänzen Sie ein Diagnose-Label, das jeden Button über Image untersucht und meldet, welcher Mechanismus tatsächlich im Spiel ist; tauschen Sie dann das unter einem Schlüssel der ImageList gespeicherte Bild aus und prüfen Sie, dass jedes Steuerelement, das diesen Schlüssel verwendet, der Änderung folgt.

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

- ✓ IconDesk-Befehls-Panel mit den Buttons Open, Save, Print und Delete
- ✓ btnOpen mit einem auf dem Server geladenen System.Drawing.Image und btnSave mit einem benannten Theme-Bild auf ImageSource
- ✓ btnPrint und btnDelete, die sich über ImageKey statt ImageIndex eine ImageList teilen
- ✓ Diagnose-Label, das jeden Button über Image liest und den Mechanismus hinter seinem Icon benennt
- ✓ Lab-Notiz, die festhält, was beim Austausch eines ImageList-Schlüssels geschah und was, als zwei Bildeigenschaften auf demselben Button gesetzt wurden

Warum es so aufgebaut ist

Wisej.NET bietet bewusst mehr als einen Weg, eine Grafik auf ein Steuerelement zu bringen, und jeder davon funktioniert. Die Schwierigkeiten beginnen, wenn ein Team versehentlich alle Wege gleichzeitig verwendet: eine Toolbar, bei der ein Button eine Datei von der Serverfestplatte liest, der nächste auf einen Theme-Namen zeigt und ein dritter sein Icon still verloren hat, weil zwei Eigenschaften auf demselben

Steuerelement gesetzt wurden. Sechs Monate später kann niemand mehr erklären, woher ein bestimmtes Bild eigentlich kommt oder warum es nach dem Deployment verschwunden ist.

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
Eigenschaft Image	Ein System.Drawing.Image -Objekt, das im Serverspeicher liegt; Wisej.NET sendet es als PNG-Daten an den Browser, das Bild reist also als Bytes, die Ihr Code bereits dekodiert hat.
String ImageSource	Ein String, der eine Ressource benennt, die der Client auflösen soll: eine relative URL, eine absolute URL, ein benanntes Theme-Bild, ein resource.wx -Pfad oder ein Element in einer Icon-Pack-Assembly.
IconSource	Dieselbe String-Idee auf Steuerelementen, die neben ihrem Hauptbild einen eigenen Icon-Platz besitzen – prüfen Sie also die API des verwendeten Steuerelements, statt anzunehmen, jedes Steuerelement benenne die Eigenschaft gleich.
Vertrag IImage	Die gemeinsame Bildfamilie, die bildfähige Steuerelemente anbieten – Image, ImageAlign, ImageIndex, ImageKey, ImageList und ImageSource –, sodass eine Hilfsmethode einen Button, ein Label oder beliebige andere Steuerelemente einheitlich ausstatten kann.
ImageList	Eine wiederverwendbare Sammlung von Bildobjekten, auf die mehrere Steuerelemente zeigen können, adressiert über Position oder Name, und in der Lage, für die nutzenden Steuerelemente eine gemeinsame Bildgröße festzulegen.
ImageKey gegenüber ImageIndex	Ein Schlüssel wie "delete" trägt Bedeutung und übersteht ein Umsortieren; der Index 3 wird stillschweigend zum falschen Bild, sobald jemand ein Bild darüber einfügt.
Sich gegenseitig ausschließende Zuweisung	All diese Eigenschaften bedienen einen einzigen visuellen Platz, eine Zuweisung kann die anderen also zurücksetzen – deshalb ist ein falsches Icon meist eine zweite Zuweisung und keine defekte Datei.

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

Öffnet [chatgpt.com](#) mit dem Prompt dieses Moduls. Funktioniert mit GPT-5 oder



Claude verwenden

Öffnet [claude.ai](#) mit demselben Prompt. Wenn Sie Claude Code nutzen, können Sie

Codex; jeder Tarif mit Websuche ruft die Spezifikation ab.

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 IconDesk 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 – „Das Bildmodell von Wisej.NET: Image, ImageSource und ImageList“ – des LearnWisej-Kurses „Icons und Bilder 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/icons-and-images/specs/module-01.md>

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

1. Projekt „IconDesk“, 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 das Projekt IconDesk und geben Sie ihm ein Befehls-Panel mit den Buttons Open, Save, Print und Delete, von denen jeder sein Icon über einen anderen Mechanismus erhält. Laden Sie `btnOpen` aus einem serverseitigen `System.Drawing.Image`, weisen Sie `btnSave.ImageSource` ein benanntes Theme-Bild zu und lassen Sie `btnPrint` und `btnDelete` über `ImageKey` auf eine gemeinsame `ImageList` zeigen. Ergänzen Sie ein Diagnose-Label, das jeden Button über `IImage` untersucht und meldet, welcher Mechanismus tatsächlich im Spiel ist; tauschen Sie dann das unter einem Schlüssel der `ImageList` gespeicherte Bild aus und prüfen Sie, dass jedes Steuerelement, das diesen Schlüssel verwendet, der Änderung folgt.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - IconDesk-Befehls-Panel mit den Buttons Open, Save, Print und Delete
 - `btnOpen` mit einem auf dem Server geladenen `System.Drawing.Image` und `btnSave` mit einem benannten Theme-Bild auf `ImageSource`
 - `btnPrint` und `btnDelete`, die sich über `ImageKey` statt `ImageIndex` eine `ImageList` teilen
 - Diagnose-Label, das jeden Button über `IImage` liest und den Mechanismus hinter seinem Icon benennt
 - Lab-Notiz, die festhält, was beim Austausch eines `ImageList`-Schlüssels geschah und was, als zwei Bildeigenschaften auf demselben Button gesetzt wurden
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:

- Ihr Diagnose-Label meldet, `btnDelete` verwende eine Bildquelle, obwohl Sie nur `ImageKey` gesetzt haben. Was muss gelaufen sein, und wo würden Sie danach suchen?
- Print und Delete haben sich beide geändert, als Sie einen Eintrag in der `ImageList` ersetzt haben. Wäre das auch so, wenn die Buttons `ImageIndex` verwendeten, und was passierte, wenn ein fünftes Bild oben in die Liste eingefügt würde?

– Der Open-Button liest seine Datei auf dem Server. Welcher Teil dieser Anordnung bricht, wenn die Anwendung auf einem anderen Rechner ausgeliefert wird, und welcher Teil bräche nicht, wenn dasselbe Bild ein Bildquellen-String wäre?

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:

- ✓ Ihr Diagnose-Label meldet, btnDelete verwende eine Bildquelle, obwohl Sie nur ImageKey gesetzt haben. Was muss gelaufen sein, und wo würden Sie danach suchen?
- ✓ Print und Delete haben sich beide geändert, als Sie einen Eintrag in der ImageList ersetzt haben. Wäre das auch so, wenn die Buttons ImageIndex verwendeten, und was passierte, wenn ein fünftes Bild oben in die Liste eingefügt würde?
- ✓ Der Open-Button liest seine Datei auf dem Server. Welcher Teil dieser Anordnung bricht, wenn die Anwendung auf einem anderen Rechner ausgeliefert wird, und welcher Teil bräche nicht, wenn dasselbe Bild ein Bildquellen-String wäre?

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.

- ✓ **Beweisen Sie, dass es wirklich funktioniert.** *"Geh dieses Projekt Ergebnis für Ergebnis anhand der Modulspezifikation unter <https://learnwisej.com/assets/courses/icons-and-images/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 IconDesk-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."

- ✓ **Tauschen Sie ein Teil aus, ohne die Schnittstellen zu verschieben.** *"Wähle den Teil dieses IconDesk-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."*



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.