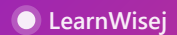


KI-Coding-Übung

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.



Den Code zu schreiben ist heute die leichtere Hälfte.

Funktionierenden Code für „Wisej.NET-Icon-Packs und NuGet“ bekommen Sie von einem modernen Assistenten mit einem guten Prompt. Ihn zu beurteilen, zu korrigieren und zu erweitern, erfordert alles, was dieses Modul behandelt hat. Genau auf dieser Aufteilung baut die Übung auf: Sie liefern Spezifikation und Review, der Assistent die Tipparbeit.

Worum Sie bitten

Fügen Sie IconDesk zwei offizielle Wisej-4-Icon-Pack-NuGet-Pakete hinzu und bauen Sie eine Seite IconCompare, die dieselben fünf Konzepte – Save, Delete, Search, User und Settings – aus beiden Familien zeigt, sodass der optische Unterschied nicht zu übersehen ist. Wählen Sie mindestens fünf der Icons über den Bild-Explorer von Visual Studio statt den Pfad zu tippen, weisen Sie die übrigen in C# aus dem Katalog des Packs zu, färben Sie zwei davon mit dem Suffix ?color= um und beobachten Sie im Netzwerk-Panel, wie die Icon-Dateien Anfrage für Anfrage eintreffen und danach aus dem Browser-Cache kommen. Schließen Sie mit einer kurzen Empfehlung ab, die die primäre Icon-Familie für das Projekt benennt.

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

- ✓ Zwei offizielle Wisej-4-Icon-Pack-NuGet-Pakete, die IconDesk referenziert
- ✓ Seite IconCompare, die Save, Delete, Search, User und Settings aus beiden Packs zeigt
- ✓ Mindestens fünf Icons, die über den Bild-Explorer von Visual Studio ausgewählt und nicht von Hand getippt wurden
- ✓ Zwei mit dem Farbsuffix umgefärbte Icons und eine Notiz dazu, welche Grafik das Umfärben angenommen hat
- ✓ Ein Absatz Empfehlung, der die primäre Icon-Familie des Projekts benennt, mit dem Beleg aus dem Browser-Cache

Warum es so aufgebaut ist

Nach drei Modulen kann IconDesk ein Bild bereits auf fünf verschiedene Arten zeigen, aber jedes Icon, das es besitzt, wurde von Hand gezeichnet oder heruntergeladen. Eine echte Anwendung braucht am ersten Tag ein Icon für Save, Delete, Search, User und Settings und bis zum Quartalsende weitere vierzig, alle in derselben visuellen Sprache. SVG-Dateien einzeln hineinzukopieren erzeugt einen Ordner voll zusammenhangloser Grafiken, uneinheitlicher Strichstärken und ohne Lizenznachweis.

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
Icon-Pack	Eine gewöhnliche.NET-Assembly, deren Icon-Dateien, meist SVG, als eingebettete Ressourcen einkompiliert sind; sie zu referenzieren fügt Ihrem Deployment eine Datei hinzu, egal wie viele Icons darin stecken.
eingebettete Ressource	Eine Datei, die in die Assembly kompiliert statt daneben kopiert wird – die Icons können also beim Deployment auf einen anderen Server nicht verloren gehen, umbenannt oder vergessen werden.
resource.wx	Der Wisej.NET-Handler, der eine eingebettete Ressource über HTTP ausliefert; eine Icon-Pack-Bildquelle ist eine resource.wx -URL, die die Pack-Assembly und die Datei darin benennt.
Icon-Katalog	Die statische Klasse, die ein Pack veröffentlicht, damit Aufrufer IntelliSense und einen zur Compile-Zeit geprüften Namen statt eines getippten Pfads bekommen; Namespace und Klassenname unterscheiden sich von Pack zu Pack.
Anfrage pro Icon	Pack-Icons werden einzeln bei der ersten Verwendung geholt und danach aus dem Browser-Cache geliefert – das macht einen Katalog mit Tausenden Icons verkraftbar.
Vorladen der Theme-Bilder	Theme-Bilder reisen mit dem Theme und liegen beim Start der Anwendung bereits im Theme-Cache des Clients – sie verhalten sich beim Start also ganz anders als Pack-Icons.
primäre Icon-Familie	Das eine Pack, auf das sich ein Produkt für seine alltäglichen Befehle festlegt, bewusst gewählt nach Abdeckung, Stil und Lizenz statt Pack für Pack angesammelt.

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 4 geht. Sie können [module-04.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 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 4 – „Wisej.NET-Icon-Packs und NuGet“ – 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-04.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: Fügen Sie IconDesk zwei offizielle Wisej-4-Icon-Pack-NuGet-Pakete hinzu und bauen Sie eine Seite IconCompare, die dieselben fünf Konzepte – Save, Delete, Search, User und Settings – aus beiden Familien zeigt, sodass der optische Unterschied nicht zu übersehen ist. Wählen Sie mindestens fünf der Icons über den Bild-Explorer von Visual Studio statt den Pfad zu tippen, weisen Sie die übrigen in C# aus dem Katalog des Packs zu, färben Sie zwei davon mit dem Suffix `?color=` um und beobachten Sie im Netzwerk-Panel, wie die Icon-Dateien Anfrage für Anfrage eintreffen und danach aus dem Browser-Cache kommen. Schließen Sie mit einer kurzen Empfehlung ab, die die primäre Icon-Familie für das Projekt benennt.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - Zwei offizielle Wisej-4-Icon-Pack-NuGet-Pakete, die IconDesk referenziert
 - Seite IconCompare, die Save, Delete, Search, User und Settings aus beiden Packs zeigt
 - Mindestens fünf Icons, die über den Bild-Explorer von Visual Studio ausgewählt und nicht von Hand getippt wurden
 - Zwei mit dem Farbsuffix umgefärbte Icons und eine Notiz dazu, welche Grafik das Umfärben angenommen hat
 - Ein Absatz Empfehlung, der die primäre Icon-Familie des Projekts benennt, mit dem Beleg aus dem Browser-Cache
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:

- Ein Pack mit dreitausend Icons zu referenzieren fügt dem Deployment eine Assembly hinzu. Was kostet das den Browser, und wann?
- Eines Ihrer beiden umgefärbten Icons hat das Farbsuffix ignoriert. Was ist das Besondere an dieser Grafik, und was würden Sie deshalb an den Auswahlkriterien für Packs ändern?
- Ein Teammitglied möchte, dass jedes Befehls-Icon dem Theme folgt, wenn die Anwendung das

Theme wechselt. Welche der fünf IconCompare-Icons sollten stattdessen Theme-Bilder werden und welche nicht?

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:

- ✓ Ein Pack mit dreitausend Icons zu referenzieren fügt dem Deployment eine Assembly hinzu. Was kostet das den Browser, und wann?
- ✓ Eines Ihrer beiden umgefärbten Icons hat das Farbsuffix ignoriert. Was ist das Besondere an dieser Grafik, und was würden Sie deshalb an den Auswahlkriterien für Packs ändern?
- ✓ Ein Teammitglied möchte, dass jedes Befehls-Icon dem Theme folgt, wenn die Anwendung das Theme wechselt. Welche der fünf IconCompare-Icons sollten stattdessen Theme-Bilder werden und welche nicht?

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.

- ✓ **Machen Sie es fit für einen schlechten Tag.** *"Untersuche dieses IconDesk-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 IconDesk-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/icons-and-images/specs/module-04.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.