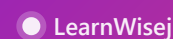


Esercizio di coding con l'IA

Costruisci il progetto IconDesk di questo modulo con ChatGPT o Claude partendo da un prompt già pronto che indirizza il modello alla specifica del modulo, poi rivedi, esegui ed estendi quello che ti restituisce.



Questo progetto non lo scriverai a mano. Lo dirigerai.

Quasi nessuno scrive più codice Wisej.NET da un file vuoto: un assistente scrive la prima bozza e lo sviluppatore decide se è giusta. Funziona però solo se conosci la materia, ed è quello che ti ha dato il resto di questo modulo. Qui affidi quelle conoscenze a ChatGPT o a Claude, ti fai restituire il progetto del modulo come soluzione Visual Studio funzionante, e lo rivedi come rivedresti la pull request di un collega.

Che cosa stai chiedendo

Crea IconDesk.Icons come class library accanto all'applicazione, metti almeno dodici icone SVG in una cartella Resources e marcale Embedded Resource. Aggiungi un catalogo statico Applcons che espone ogni icona come stringa resource.wx dal nome parlante, così che chi la usa non debba mai scrivere un percorso; referencia la libreria da IconDesk, poi assegna alcune icone tramite il designer e altre dal catalogo in C#. Verifica il suffisso di colore su tre icone e annota la pulizia dell'SVG di cui hanno avuto bisogno. Concludi con le regole di denominazione e versionamento che stabiliscono come una nuova icona entra nel pacchetto.

Il modulo chiede questi deliverable, e il prompt qui sotto li chiede all'assistente esattamente così:

- ✓ Class library IconDesk.Icons con almeno dodici file SVG in una cartella Resources, tutti Embedded Resource
- ✓ Catalogo statico Applcons che espone ogni icona come stringa resource.wx dal nome parlante
- ✓ Pagina IconDesk che assegna alcune icone tramite il designer e altre dal catalogo Applcons
- ✓ Tre icone verificate con il suffisso di colore, più una nota sulla pulizia dell'SVG di cui hanno avuto bisogno
- ✓ Regole di denominazione e versionamento che descrivono come una nuova icona entra nel pacchetto e come un'applicazione ne sovrascrive una

Perché è fatto così

Al quinto modulo IconDesk mostra un logo incorporato, un'immagine del tema e le icone di due pacchetti ufficiali, ed è lì che comincia il disordine: le stringhe di risorsa compaiono inline nei gestori di eventi, due schermate scrivono lo stesso file in modo diverso e nessuno sa dire quali icone appartengono al prodotto. Un pacchetto di icone personalizzato è il rimedio a basso costo, e in più ti compra un confine: un assembly contiene la grafica, un catalogo C# le dà un nome e il resto dell'applicazione non scrive mai più un percorso.

Questi sono i termini che l'assistente deve azzeccare. Se quello che ti restituisce li usa con leggerezza, non ha capito il modulo:

Termine	Che cosa significa qui
assembly di risorse	Una semplice class library.NET il cui unico compito è portare i file delle icone come risorse incorporate; non ha bisogno di alcun riferimento all'applicazione né di codice Wisej.NET proprio.
la ricetta in quattro passi	Crea una class library, crea una cartella chiamata Resources, metti dentro i file delle icone, imposta la loro Build Action su Embedded Resource. Tutto quello che viene dopo è comodità.
catalogo delle icone	Una classe statica come Applcons che espone ogni icona come costante stringa dal nome parlante, così che il compilatore e IntelliSense sostituiscano i percorsi copiati e incollati.
nome semantico contro nome del file	Il catalogo espone il vocabolario del tuo prodotto — save, delete, customer, warning — mentre il file mantiene un nome semplice, così rinominarlo cambia una riga e non quaranta schermate.
indirizzo di risorsa qualificato	Un'icona in un assembly separato si indirizza come resource.wx/ più il nome dell'assembly più il percorso della risorsa: è per questo che ogni voce del catalogo condivide un'unica costante radice privata.
contratto di ricolorazione	Il suffisso?color= ricolora solo grafica monocromatica adatta; un SVG con valori di fill fissati nel file è il motivo abituale per cui un'icona si rifiuta di cambiare colore.
set a tema e override	Pacchetti paralleli che condividono un'unica API di catalogo su grafica diversa, più file distribuiti che hanno la precedenza su una risorsa incorporata, sostituiscono le icone senza toccare il codice dell'applicazione.

Costruiscilo — scegli il tuo assistente

Entrambi i pulsanti aprono una nuova scheda con lo **stesso prompt già pronto**. Il prompt indirizza il modello alla specifica di questo modulo — un unico file Markdown con la lezione, la traccia del lab e i deliverable — così lavora sul materiale vero invece di tirare a indovinare che cosa sia il Modulo 6. Puoi aprire tu stesso [module-06.md](#) per vedere esattamente che cosa leggerà.



Usa ChatGPT

Apri [chatgpt.com](#) con il prompt di questo modulo già caricato. Funziona con GPT-5 o Codex; qualsiasi piano che possa navigare il web recupererà la specifica.



Usa Claude

Apri [claude.ai](#) con lo stesso prompt già caricato. Chi usa Claude Code può invece incollarlo in un terminale dentro una cartella vuota.



Mandalo così com'è e ottieni un progetto, non un frammento

Il prompt si chiude chiedendo l'intera soluzione in un unico `.zip` con i passi per compilarla ed eseguirla. Scaricalo, apri la soluzione in Visual Studio, premi F5 e IconDesk si apre su `localhost`. Ti servono l'SDK .NET e il pacchetto NuGet `Wisej-4`; tutto il resto il prompt chiede all'assistente di scriverlo da sé.

Sto seguendo il Modulo 6 - "Costruire il tuo pacchetto di icone Wisej.NET" - del corso LearnWisej "Icane e immagini in Wisej.NET", e voglio costruire con te il progetto del modulo. Rispondimi in italiano.

Comincia leggendo la specifica del modulo. È un unico file Markdown (in inglese) che contiene l'intera lezione, la traccia del lab e i deliverable:

<https://learnwisej.com/assets/courses/icons-and-images/specs/module-06.md>

Poi costruisci un progetto Wisej.NET 4 completo che io possa aprire in Visual Studio:

1. Progetto "IconDesk", C#, net8.0 o successivo, pacchetto NuGet Wisej-4 4.1.x, host Kestrel che chiama `app.UseWisej()`.
2. Costruisci quello che chiede il lab della specifica: Crea `IconDesk.Icons` come class library accanto all'applicazione, metti almeno dodici icone SVG in una cartella `Resources` e marcale `Embedded Resource`. Aggiungi un catalogo statico `AppIcons` che espone ogni icona come stringa `resource.wx` dal nome parlante, così che chi la usa non debba mai scrivere un percorso; referencia la libreria da `IconDesk`, poi assegna alcune icone tramite il designer e altre dal catalogo in C#. Verifica il suffisso di colore su tre icone e annota la pulizia dell'SVG di cui hanno avuto bisogno. Concludi con le regole di denominazione e versionamento che stabiliscono come una nuova icona entra nel pacchetto.
3. Produci questi deliverable e dimmi dove si trova ciascuno nella soluzione:
 - Class library `IconDesk.Icons` con almeno dodici file SVG in una cartella `Resources`, tutti `Embedded Resource`
 - Catalogo statico `AppIcons` che espone ogni icona come stringa `resource.wx` dal nome parlante
 - Pagina `IconDesk` che assegna alcune icone tramite il designer e altre dal catalogo `AppIcons`
 - Tre icone verificate con il suffisso di colore, più una nota sulla pulizia dell'SVG di cui hanno avuto bisogno
 - Regole di denominazione e versionamento che descrivono come una nuova icona entra nel pacchetto e come un'applicazione ne sovrascrive una
4. Segui la specifica ovunque nomini una classe, una proprietà o un evento di Wisej.NET - usa esattamente quello. Dove lascia una scelta aperta, dimmi quale API hai scelto e perché.
5. Tieni l'autorità sul server: stato e validazione vivono in .NET, il browser disegna e riferisce. Non spostare mai una decisione in JavaScript solo perché lì è più comodo.
6. Rendi il risultato dimostrabile: devo poterlo eseguire e vedere il comportamento di cui parla questo modulo, compreso che cosa succede quando qualcosa va storto.

Prima di scrivere codice, riformula con parole tue che cosa costruirai e dove starà ogni pezzo, così posso correggere il progetto finché costa poco.

Quando il codice è pronto, impacchetta l'intera soluzione in un unico file .zip scaricabile e dammi i passi esatti per compilarla ed eseguirla.

Infine rispondi a queste domande della specifica, usando come prova la tua implementazione:

- Il tuo pacchetto compila e il designer non mostra nulla nell'Image Selector. Quale passo della ricetta controlleresti per primo, e quale prova ti direbbe che le risorse sono davvero dentro l'assembly?
- Due delle tue tre icone di prova si ricolorano e la terza resta grigia. Che cosa c'è dentro quell'SVG, e quando lasceresti di proposito un'icona non ricolorabile?
- Un cliente vuole tre icone sostituite solo nella sua distribuzione. Quale delle strade di override sceglieresti, e che cosa ti costa quella scelta alla prossima versione del

pacchetto?

Chiedimi qualsiasi cosa non ti sia chiara prima di cominciare.

Poi rivedi quello che ti è tornato indietro

Codice che compila non è ancora codice giusto. Le domande interessanti sono domande di progetto: è questo il controllo Wisej.NET che il compito richiede davvero, segue il ciclo di vita che il framework ti offre, lo stato autorevole sta sul server o è stato parcheggiato nel browser perché era comodo? Leggi il risultato da revisore, non da cliente.

Questo modulo ti dà la checklist. Rispondi a ciascuna domanda guardando il codice che hai appena ricevuto:

- ✓ Il tuo pacchetto compila e il designer non mostra nulla nell'Image Selector. Quale passo della ricetta controlleresti per primo, e quale prova ti direbbe che le risorse sono davvero dentro l'assembly?
- ✓ Due delle tue tre icone di prova si ricolorano e la terza resta grigia. Che cosa c'è dentro quell'SVG, e quando lasceresti di proposito un'icona non ricolorabile?
- ✓ Un cliente vuole tre icone sostituite solo nella sua distribuzione. Quale delle strade di override sceglieresti, e che cosa ti costa quella scelta alla prossima versione del pacchetto?

Tutto ciò che hai dovuto correggere vale la pena di scriverlo. Una breve nota con le convenzioni e le decisioni a cui sei arrivato, incollata nel primo messaggio della sessione successiva, vale più di qualsiasi prompt ingegnoso.



Per questo corso non c'è ancora un esempio di riferimento — il tuo è il primo

Il [repository degli esempi LearnWisej](#) contiene build già fatte per quasi tutti i corsi, ma non ancora per questo. Significa che non c'è niente con cui confrontarsi e nessuna scorciatoia: la specifica e la tua revisione sono tutta la rete di sicurezza. Tieni quello che costruisci — e se viene bene, ci farebbe piacere vederlo.

Tre prompt per andare oltre

Punta il tuo assistente sul progetto che hai appena costruito e prova questi. Ognuno è un esercizio di revisione ed estensione, non una riscrittura.

- ✓ **Dimostra che funziona davvero.** *"Passa in rassegna questo progetto deliverable per deliverable rispetto alla specifica del modulo su <https://learnwisej.com/assets/courses/icons-and-images/specs/module-06.md> e mostrami la prova che ciascuno è davvero fatto — il percorso nel codice, su che cosa cliccare e che cosa dovrei vedere. Dove un deliverable è soddisfatto solo in parte, dillo chiaramente invece di arrampicarti sugli specchi. Poi aggiungi quello che manca."*
- ✓ **Rivedilo come faresti con una pull request.** *"Accompagnami attraverso questo codice IconDesk come se stessi rivedendo la pull request di un collega e non l'avessi scritto tu. Dove vive lo stato, che cosa succede dall'inizio alla fine su una singola azione dell'utente, che cosa rimanderesti indietro per modifiche, e quali parti non ti sentiresti tranquillo a mettere davanti a un cliente? Sii specifico e cita le righe di cui parli — non riassumermi il progetto."*

- ✓ **Sostituisci un pezzo senza spostare le giunzioni.** *"Scegli la parte di questo progetto IconDesk che più evidentemente sta al posto di qualcosa di reale — una sorgente dati finta, un archivio in memoria, un servizio semplificato — e sostituiscila con un'implementazione di forma produttiva. Il contratto ai due lati non deve cambiare: stessi tipi, stesse chiamate, stesso comportamento in caso di errore. Dimmi che cosa l'implementazione reale ti ha costretto a gestire e che la versione provvisoria non gestiva."*



Portalo dentro il lab

Il lab pratico che chiude questo modulo chiede gli stessi deliverable che hai appena fatto produrre all'assistente. Usa quello che hai qui come forma della tua risposta e come prova a sostegno — non come la risposta in sé. Il lab viene valutato sul tuo ragionamento, e quella parte deve essere tua.