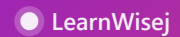


Esercizio di coding con l'IA

Costruisci il progetto GlobalDesk 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.



Istruiscilo, leggilo, poi mettilo alla prova.

Questo esercizio ha tre parti, e solo la prima riguarda il prompt. Istruisci un assistente con la specifica del modulo e ti fai restituire una soluzione GlobalDesk; leggi quello che ha scritto e decidi se fa davvero ciò che il modulo chiedeva; poi lo spingi più in là di quanto il modulo chiedesse. La seconda e la terza parte sono dove si impara.

Che cosa stai chiedendo

Fai crescere le risorse condivise di GlobalDesk. Aggiungi Strings.de.resx accanto a Resources/Strings.resx con esattamente le stesse chiavi e i valori tradotti, aggiungi una voce Customer.LastOrder che contiene due segnaposto, e componi la frase con string.Format e Application.CurrentCulture invece di concatenare frammenti. Fai passare dall'helper Texts ogni messaggio che l'editor cliente mostra, compresi la conferma di salvataggio e i due messaggi di validazione, e fai restituire al servizio di salvataggio una chiave di risultato stabile come CustomerEditor.Saved invece di una frase inglese che l'interfaccia stampa così com'è. Poi aggiungi un file Resources-de.resx che sovrascrive un'etichetta di sistema di Wisej, dimostrarlo in una MessageBox e scrivi una breve nota sulla differenza tra il testo del tuo prodotto e il testo di sistema di Wisej.

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

- ✓ Strings.de.resx con le stesse chiavi del file neutro e solo i valori tradotti
- ✓ Risorsa Customer.LastOrder composta con string.Format e la cultura attiva, mai concatenata
- ✓ Servizio di salvataggio che restituisce una chiave di risorsa o un codice di risultato invece di una frase inglese
- ✓ Resources-de.resx che sovrascrive almeno un'etichetta di sistema di Wisej, mostrata in una MessageBox
- ✓ Nota di lab che distingue testo di prodotto, testo di sistema di Wisej e testo diagnostico

Perché è fatto così

Il Modulo 2 ha lasciato l'editor cliente di GlobalDesk convincente in tedesco: ogni etichetta di cui è proprietario il designer cambia con la cultura e il layout regge. Poi l'utente preme Save e legge Customer.saved. in inglese, perché quella frase non è una proprietà di nessun controllo — è un letterale di stringa dentro un gestore di click, accanto a This field is required. e alla riga di stato che conta i contatti caricati.

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
file di risorse condiviso dell'applicazione	Resources/Strings.resx contiene il testo visibile all'utente che non appartiene a nessuna singola schermata disegnata: conferme di salvataggio, messaggi di validazione, righe di stato, intestazioni di report e risultati dei servizi.
file gemello specifico per cultura	Strings.de.resx e Strings.it.resx stanno accanto al file neutro e portano le stesse chiavi; un traduttore cambia i valori, mai i nomi delle chiavi.
chiave semantica stabile	Un nome come CustomerEditor.Saved descrive il significato del testo, così riformulare la frase inglese non costringe mai a modificare il codice né rompe le traduzioni.
Wisej.Resources.ResourceManager	L'oggetto di ricerca che segue il modello delle risorse di .NET: dato un nome base e un assembly trova il valore per la cultura attiva e ricade verso il file neutro.
marcatore di chiave mancante	Un helper che restituisce [Key] quando una ricerca fallisce trasforma un difetto invisibile in qualcosa che puoi vedere e cercare, invece che in una label vuota.
stringa di formato con segnaposto	L'intera frase con i suoi {0} e {1} vive nella risorsa e string.Format la riempie, perché le lingue ordinano il nome, la data e il verbo in modo diverso.
Resources-[LANG].resx	Un file di risorse di sistema di Wisej.NET separato, nella radice del progetto o in una cartella /Resources, che sovrascrive o aggiunge etichette del framework come \$Ok, \$Cancel e \$Offline.

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 3. Puoi aprire tu stesso [module-03.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 GlobalDesk 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 3 - "Risorse dell'applicazione, ResourceManager e i test di sistema di Wisej" - del corso LearnWisej "Localizzazione 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/localization-in-wisej-net/specs/module-03.md>

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

1. Progetto "GlobalDesk", 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: Fai crescere le risorse condivise di GlobalDesk. Aggiungi `Strings.de.resx` accanto a `Resources/Strings.resx` con esattamente le stesse chiavi e i valori tradotti, aggiungi una voce `Customer.LastOrder` che contiene due segnaposto, e componi la frase con `string.Format` e `Application.CurrentCulture` invece di concatenare frammenti. Fai passare dall'helper `Texts` ogni messaggio che l'editor cliente mostra, compresi la conferma di salvataggio e i due messaggi di validazione, e fai restituire al servizio di salvataggio una chiave di risultato stabile come `CustomerEditor.Saved` invece di una frase inglese che l'interfaccia stampa così com'è. Poi aggiungi un file `Resources-de.resx` che sovrascrive un'etichetta di sistema di Wisej, dimostralò in una `MessageBox` e scrivi una breve nota sulla differenza tra il testo del tuo prodotto e il testo di sistema di Wisej.
3. Produci questi deliverable e dimmi dove si trova ciascuno nella soluzione:
 - `Strings.de.resx` con le stesse chiavi del file neutro e solo i valori tradotti
 - Risorsa `Customer.LastOrder` composta con `string.Format` e la cultura attiva, mai concatenata
 - Servizio di salvataggio che restituisce una chiave di risorsa o un codice di risultato invece di una frase inglese
 - `Resources-de.resx` che sovrascrive almeno un'etichetta di sistema di Wisej, mostrata in una `MessageBox`
 - Nota di lab che distingue testo di prodotto, testo di sistema di Wisej e testo diagnostico
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:

- Elimina una chiave da `Strings.de.resx` ed esegui in tedesco. Descrivi che cosa vede l'utente, perché non va in crash niente, e come te ne accorgeresti in un'applicazione reale con duecento chiavi.
- Il tuo servizio di salvataggio oggi restituisce `CustomerEditor.Saved`. Che cosa si romperebbe negli unit test, e nella schermata in tedesco, se restituisse invece la frase

vera e propria?

– Un collega aggiunge la parola "Cancel" a Resources-de.resx sotto una chiave di sua invenzione. Spiega perché il pulsante della MessageBox non cambia, e dove apparteneva davvero quella stringa.

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:

- ✓ Elimina una chiave da Strings.de.resx ed esegui in tedesco. Descrivi che cosa vede l'utente, perché non va in crash niente, e come te ne accorgeresti in un'applicazione reale con duecento chiavi.
- ✓ Il tuo servizio di salvataggio oggi restituisce CustomerEditor.Saved. Che cosa si romperebbe negli unit test, e nella schermata in tedesco, se restituisse invece la frase vera e propria?
- ✓ Un collega aggiunge la parola "Cancel" a Resources-de.resx sotto una chiave di sua invenzione. Spiega perché il pulsante della MessageBox non cambia, e dove apparteneva davvero quella stringa.

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.

- ✓ **Rivedilo come faresti con una pull request.** *"Accompagnami attraverso questo codice GlobalDesk 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 GlobalDesk 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."*
- ✓ **Fallo sopravvivere a una brutta giornata.** *"Esamina questo progetto GlobalDesk cercando le modalità di guasto che non gestisce ancora — input sbagliato o mancante, una dipendenza lenta o non disponibile, lavoro che arriva dopo che l'utente ha cambiato pagina, e due azioni che si accavallano. Per ciascuna dimmi prima se è un rischio reale in Wisej.NET 4 e perché, e solo dopo scrivi la correzione. Dove decidi che non serve gestire nulla, dillo e spiega il ragionamento invece di aggiungere codice difensivo."*



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.