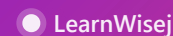


Esercizio di coding con l'IA

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.



Scrivere il codice è ormai la metà facile.

Ottenere codice funzionante per "Cultura del browser, cultura di sessione e cambio lingua a runtime" da un assistente moderno richiede un buon prompt. Giudicarlo, correggerlo ed estenderlo richiede tutto quello che questo modulo ha trattato. È su questa divisione che è costruito l'esercizio: tu metti la specifica e la revisione, l'assistente mette la digitazione.

Che cosa stai chiedendo

Dai a GlobalDesk un cambio lingua che porta a termine il lavoro. Imposta culture su auto in Default.json così che la sessione parta dalla lingua del browser, aggiungi una ComboBox della lingua che offre inglese, tedesco e una cultura lingua-regione, e imposta Application.CurrentCulture da CultureInfo.GetCultureInfo nel suo gestore SelectedIndexChanged senza mai memorizzare la scelta in un campo statico. Gestisci Application.CultureChanged per chiamare un metodo ApplyTextResources che aggiorna ogni etichetta che viene dalle risorse e un metodo UpdateCulturePreview che riformatta gli esempi di data, numero e importo, e ricrea nello stesso gestore l'editor cliente localizzato nel designer. Verifica la stessa pagina attraverso il parametro URL lang, poi scrivi una nota di lab che descrive che cosa farebbe un campo di cultura statico a un secondo utente collegato nello stesso momento.

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

- ✓ Default.json con culture impostato su auto, più una nota su quando una cultura fissa è la scelta giusta
- ✓ Gestore della ComboBox della lingua che imposta Application.CurrentCulture da CultureInfo.GetCultureInfo
- ✓ Gestore di Application.CultureChanged che chiama ApplyTextResources e UpdateCulturePreview
- ✓ Editor localizzato nel designer ricreato dopo il cambio, così nessuna etichetta resta nella lingua precedente
- ✓ Verifica attraverso il parametro URL lang e una nota sul perché la cultura non deve vivere in un campo statico

Perché è fatto così

Dopo il Modulo 3, GlobalDesk ha un file di risorse neutro, uno tedesco accanto, un helper Texts, un editor cliente localizzato nel designer e perfino un'etichetta di sistema di Wisej tutta sua. Quello a cui ancora non sa rispondere è la domanda che un utente vero fa per prima: come lo ottengo nella mia lingua, e che cosa

succede alla schermata quando lo chiedo? Fin qui tutto è stato deciso mentre l'applicazione veniva costruita; questo modulo parla dei due momenti in cui la decisione viene presa davvero — quando una sessione inizia, e quando un utente cambia idea a metà strada.

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
rilevamento automatico del browser	Wisej.NET può leggere la lingua che il browser dichiara e avviare la sessione con quella, così ogni utente ottiene una cultura sensata senza che un'unica impostazione valida per tutto il server decida per tutti.
culture in Default.json	L'impostazione dell'applicazione che decide da dove arriva la cultura di una nuova sessione; il suo valore predefinito è auto, mentre un nome fisso come de-DE fissa invece ogni sessione su un solo locale.
rightToLeft in Default.json	L'impostazione vicina, il cui valore auto segue qualunque cultura finisca per avere la sessione; il Modulo 5 ci costruisce sopra, ma la incontri qui nello stesso file.
Application.CurrentCulture	Legge o imposta la cultura della sessione Wisej.NET corrente, che è esattamente lo scope di cui ha bisogno un'applicazione web multi-utente: la scelta di un utente non deve mai ridisegnare la schermata di un altro.
parametro URL lang	Un modo supportato per chiedere una lingua già in ingresso, per esempio? lang=de, che rende banale produrre link di test e URL di ingresso specifici per lingua.
Application.CultureChanged	Viene sollevato quando la cultura corrente cambia rispetto a quella con cui la sessione è partita; è l'unico posto in cui riapplicare il testo che viene dalle risorse e riformattare tutto ciò che è già sullo schermo.
controlli localizzati nel designer	I loro valori localizzati vengono applicati quando i controlli vengono creati, quindi un cambio di cultura successivo non li riscrive — ricrei il contenitore oppure riapplichi i valori nel codice.

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 4. Puoi aprire tu stesso [module-04.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



Usa Claude

Apri [claude.ai](#) con lo stesso prompt già caricato. Chi usa Claude Code può invece

Codex; qualsiasi piano che possa navigare il web recupererà la specifica.

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 4 - "Cultura del browser, cultura di sessione e cambio lingua a runtime" - 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-04.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: Dai a GlobalDesk un cambio lingua che porta a termine il lavoro. Imposta culture su auto in Default.json così che la sessione parta dalla lingua del browser, aggiungi una ComboBox della lingua che offre inglese, tedesco e una cultura lingua-regione, e imposta `Application.CurrentCulture` da `CultureInfo.GetCulturesInfo` nel suo gestore `SelectedIndexChanged` senza mai memorizzare la scelta in un campo statico. Gestisci `Application.CultureChanged` per chiamare un metodo `ApplyTextResources` che aggiorna ogni etichetta che viene dalle risorse e un metodo `UpdateCulturePreview` che riformatta gli esempi di data, numero e importo, e ricrea nello stesso gestore l'editor cliente localizzato nel designer. Verifica la stessa pagina attraverso il parametro URL `lang`, poi scrivi una nota di lab che descrive che cosa farebbe un campo di cultura statico a un secondo utente collegato nello stesso momento.
3. Produci questi deliverable e dimmi dove si trova ciascuno nella soluzione:
 - Default.json con culture impostato su auto, più una nota su quando una cultura fissa è la scelta giusta
 - Gestore della ComboBox della lingua che imposta `Application.CurrentCulture` da `CultureInfo.GetCulturesInfo`
 - Gestore di `Application.CultureChanged` che chiama `ApplyTextResources` e `UpdateCulturePreview`
 - Editor localizzato nel designer ricreato dopo il cambio, così nessuna etichetta resta nella lingua precedente
 - Verifica attraverso il parametro URL `lang` e una nota sul perché la cultura non deve vivere in un campo statico
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 metodo `ApplyTextResources` aggiorna perfettamente la dashboard, eppure l'editor cliente conserva le sue etichette inglesi dopo il passaggio al tedesco. Spiega che cosa hanno di diverso quelle etichette, e indica due strategie che risolvono il problema.
- Un collega propone di mettere in cache in un campo statico il `CultureInfo` scelto "per non

doverlo cercare due volte". Descrivi che cosa vedrebbe davvero un secondo utente, collegato da un altro paese nello stesso momento.

– L'anteprima della cultura mostra un importo in valuta. Dopo un passaggio da en-US a de-DE, che cosa è cambiato in memoria e che cosa è cambiato solo sullo schermo – e quale delle due cose scriveresti mai nel database?

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 metodo `ApplyTextResources` aggiorna perfettamente la dashboard, eppure l'editor cliente conserva le sue etichette inglesi dopo il passaggio al tedesco. Spiega che cosa hanno di diverso quelle etichette, e indica due strategie che risolvono il problema.
- ✓ Un collega propone di mettere in cache in un campo statico il `CultureInfo` scelto "per non doverlo cercare due volte". Descrivi che cosa vedrebbe davvero un secondo utente, collegato da un altro paese nello stesso momento.
- ✓ L'anteprima della cultura mostra un importo in valuta. Dopo un passaggio da en-US a de-DE, che cosa è cambiato in memoria e che cosa è cambiato solo sullo schermo — e quale delle due cose scriveresti mai nel database?

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.

- ✓ **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."
- ✓ **Aggiungi il secondo caso.** "Estendi questo progetto GlobalDesk con un secondo caso che segue lo stesso schema del primo — un'altra entità, schermata o controllo della stessa forma, collegato allo stesso modo. Estrai in una classe base o in un servizio ciò che è davvero condiviso, ma solo dove se lo merita; due cose simili non chiedono sempre un'astrazione. Mostrami il diff e giustifica ogni cosa che hai estratto."



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.