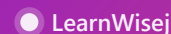


Esercizio di coding con l'IA

Costruisci il progetto AdaptiveOps 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 — confrontandolo anche con la nostra build di riferimento su GitHub.



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 AdaptiveOps; 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

Aggiungi alla Adaptive Operations Console un foglio di stile circoscritto AdaptiveOps.css con classi per le card delle metriche e i badge compatti, applicale tramite CssClass e usa CssStyle solo per un valore calcolato, come la larghezza di una barra di avanzamento. Assegna l'AppearanceKey action-button al comando principale, aggiungi uno stato personalizzato stale alle righe della griglia dei ticket o a un pannello di stato, e implementa nella toolbar un cambio di tema chiaro/scuro che cambia il tema solo per la sessione corrente, senza modificare il tema condiviso.

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

- ✓ AdaptiveOps.css circoscritto con le classi metric-card e compact-badge applicate tramite CssClass
- ✓ CssStyle usato per esattamente un valore calcolato una tantum, con il motivo documentato
- ✓ Comando principale che usa l'AppearanceKey action-button
- ✓ Stato personalizzato stale aggiunto tramite States e con lo stile definito nel tema
- ✓ Cambio di tema chiaro/scuro valido solo per la sessione che lascia intatto il tema condiviso

Perché è fatto così

Il tema AdaptiveOps del Modulo 2 ora possiede l'identità di ogni controllo standard di Adaptive Operations Console, e le richieste che arrivano dopo sono tutte piccole: una card delle metriche ha bisogno di una superficie arrotondata e rialzata, un badge SLA deve essere compatto, una barra di avanzamento ha bisogno di una larghezza calcolata da dati in tempo reale, il comando Assign dovrebbe sembrare l'azione principale, e il turno di notte vuole una variante scura. Ogni richiesta ha una scorciatoia allettante — una stringa CssStyle qui, un BackColor là, il tema condiviso modificato sul posto — e ogni scorciatoia sposta in silenzio una decisione visiva a un livello che non la possiede. Sei mesi dopo nessuno

sa più dire che cosa visualizza il tema, che cosa scavalca il CSS e che cosa rattoppa il codice. Questo modulo mette ciascuna di queste cinque richieste nel meccanismo che la possiede.

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
CssClass	Uno o più nomi di classe aggiunti all'elemento principale del widget nel browser: l'aggancio riutilizzabile a cui un foglio di stile applicativo circoscritto può puntare senza toccare il tema.
CssStyle	Una stringa CSS applicata direttamente come attributo style dell'elemento principale del widget; giusta per un singolo valore calcolato, sbagliata come strumento di stile generale perché gli stili inline sono difficili da ispezionare, da gestire con il tema o da passare a un designer.
foglio di stile applicativo circoscritto	Un file CSS caricato accanto al tema i cui selettori partono da nomi di classe che possiedi tu, come <code>metric-card</code> , e non entrano mai nel DOM interno di un widget Wisej standard.
vecchio extender StyleSheet	Un componente più vecchio che aggiunge classi CSS ai controlli idonei e ospita le regole nella sua proprietà <code>StyleSheetSource</code> ; un aiuto per la migrazione, inutile nel nuovo codice perché ogni controllo ora ha <code>CssClass</code> e <code>CssStyle</code> .
AppearanceKey	Il nome di un'appearance del tema attivo con cui viene visualizzato un controllo, così il C# esprime l'intenzione (<code>action-button</code>) e il tema possiede colori, font e metriche di ogni stato.
stati personalizzati	Nomi aggiunti alla collezione <code>States</code> di un controllo a cui il tema può dare stile come a <code>hovered</code> o <code>focused</code> , per condizioni transitorie della UI come <code>stale</code> , <code>warning</code> o <code>dirty</code> che un designer dovrebbe poter vedere e ridisegnare.
tema di avvio e <code>Application.Theme</code>	Il tema a livello di applicazione si sceglie prima che l'app parta, in <code>Web.config</code> o <code>Default.json</code> ; <code>Application.Theme</code> permette a una sessione di lavorare con un oggetto tema a runtime senza cambiare ciò che vedono le altre sessioni.

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



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 AdaptiveOps 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 - "CSS, CssClass, CssStyle, stati e cambi di tema a runtime" - del corso LearnWisej "Temi e layout responsive", 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/theming-and-responsive-layouts/specs/module-03.md>

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

1. Progetto "AdaptiveOps", 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: Aggiungi alla Adaptive Operations Console un foglio di stile circoscritto `AdaptiveOps.css` con classi per le card delle metriche e i badge compatti, applicale tramite `CssClass` e usa `CssStyle` solo per un valore calcolato, come la larghezza di una barra di avanzamento. Assegna l'`AppearanceKey action-button` al comando principale, aggiungi uno stato personalizzato stale alle righe della griglia dei ticket o a un pannello di stato, e implementa nella toolbar un cambio di tema chiaro/scuro che cambia il tema solo per la sessione corrente, senza modificare il tema condiviso.
3. Produci questi deliverable e dimmi dove si trova ciascuno nella soluzione:
 - `AdaptiveOps.css` circoscritto con le classi `metric-card` e `compact-badge` applicate tramite `CssClass`
 - `CssStyle` usato per esattamente un valore calcolato una tantum, con il motivo documentato
 - Comando principale che usa l'`AppearanceKey action-button`
 - Stato personalizzato stale aggiunto tramite `States` e con lo stile definito nel tema
 - Cambio di tema chiaro/scuro valido solo per la sessione che lascia intatto il tema condiviso
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:

- Un designer chiede un raggio degli angoli più piccolo su ogni card delle metriche. Quale file cambia, e in quanti punti avresti dovuto intervenire se il raggio fosse stato impostato con `CssStyle`?
- Perché il pulsante Assign non ha bisogno di codice per hover o pressed, e che cosa perderesti esattamente sostituendo il suo `AppearanceKey` con valori di `BackColor` e `ForeColor` che corrispondono alla palette?
- Due persone hanno effettuato l'accesso e una delle due passa alla modalità scura. Descrivi che cosa vede ciascuna, e indica la riga del tuo codice che lo garantisce.

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:

- ✓ Un designer chiede un raggio degli angoli più piccolo su ogni card delle metriche. Quale file cambia, e in quanti punti avresti dovuto intervenire se il raggio fosse stato impostato con `CssStyle`?
- ✓ Perché il pulsante Assign non ha bisogno di codice per hover o pressed, e che cosa perderesti esattamente sostituendo il suo `AppearanceKey` con valori di `BackColor` e `ForeColor` che corrispondono alla palette?
- ✓ Due persone hanno effettuato l'accesso e una delle due passa alla modalità scura. Descrivi che cosa vede ciascuna, e indica la riga del tuo codice che lo garantisce.

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.



Questo l'abbiamo già costruito — confronta il tuo con il nostro

L'implementazione di riferimento del Modulo 3 si trova nel repository pubblico degli esempi. Costruisci prima il tuo, poi apri il nostro: la parte interessante sono le differenze.



[Apri l'esempio del Modulo 3 su GitHub](#)

Oppure clona tutto con `git clone https://github.com/iceteagroup/LearnWisej-Samples` e apri `Theming and Responsive Layouts Course/Module 3`.



Il nostro esempio è un punto di partenza, non un prodotto finito

È stato costruito nello stesso modo in cui stai costruendo il tuo — con un prompt, poi corretto — e si ferma deliberatamente dove si ferma il modulo. Aspettati soluzioni provvisorie dove andrebbe codice di produzione, gestione degli errori essenziale e nessun test. Prendilo come un invito: usa quello che hai imparato in questo modulo, più l'assistente, per portarlo più avanti. Migliorare codice che non hai scritto tu è l'abilità che questo corso insegna davvero.

Tre prompt per andare oltre

Punta il tuo assistente sul progetto che hai appena costruito — o sul nostro 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 AdaptiveOps 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 AdaptiveOps 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 AdaptiveOps 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.