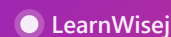


Esercizio di coding con l'IA

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.



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 la soluzione Adaptive Operations Console e il suo primo shell desktop: una Page principale con un pannello toolbar in alto, un pannello di navigazione a sinistra, un'area di lavoro su Fill, un pannello dei dettagli a destra e un pannello di stato in basso, costruiti con Dock e Panel annidati. Scegli un tema di base integrato e annota la scelta, scrivi un VisualInventory.md che elenca i controlli che hanno bisogno di appearance personalizzate, i token con nome per colori e font, le regioni del layout e i breakpoint desktop, tablet e telefono, e aggiungi una label di stato che riporta la larghezza corrente del browser così che i moduli successivi possano verificare i profili.

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

- ✓ Soluzione Adaptive Operations Console con uno shell desktop in dock composto dai pannelli toolbar, navigazione, area di lavoro, dettagli e stato
- ✓ Tema di base scelto e configurato, con il motivo annotato negli appunti del lab
- ✓ VisualInventory.md che elenca i controlli che hanno bisogno di appearance, i token del tema, le regioni del layout e i breakpoint
- ✓ Label di stato che riporta la larghezza corrente del browser
- ✓ Screenshot desktop e una nota che spiega quale livello possiede ogni decisione visiva

Perché è fatto così

Un team che tratta Wisej.NET come un framework in cui il CSS viene prima di tutto finisce con BackColor impostato su duecento controlli, selettori che si rompono al prossimo aggiornamento e gestori Resize che ricalcolano Bounds a mano; quando viene chiesta una versione per il telefono, l'unica opzione rimasta è riscrivere tutto. Questo modulo chiarisce da dove vengono davvero l'aspetto e il layout di un controllo, dà un nome ai tre livelli di personalizzazione e a quale possiede ogni decisione, e avvia lo shell di Adaptive Operations Console che ogni modulo successivo estende.

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
controllo sul server e widget client	Scrivi controlli.NET sul server; il browser visualizza widget JavaScript che Wisej.NET mantiene sincronizzati, quindi una decisione visiva può stare da una parte o dall'altra di quel confine.
tema	Un unico file JSON di colori, immagini, font, appearance, stati, stili, proprietà e impostazioni che dà a ogni controllo standard la sua identità; il contratto tra i due mondi.
appearance e AppearanceKey	Ogni controllo espone una chiave di appearance; il tema attivo contiene un'appearance per quella chiave, e una chiave semantica come action-button seleziona una variante.
stato	Una condizione di interazione con nome dentro un'appearance (default, hovered, pressed, focused, invalid, selected) i cui stili e proprietà si applicano mentre è attiva.
CssClass e CssStyle	Stile applicato direttamente nel browser: una classe da un foglio di stile applicativo circoscritto, oppure una stringa di stile grezza sull'elemento del widget per un raro valore una tantum.
motore di layout e Dock	Posizione e dimensione vengono dalla gerarchia dei contenitori, non dal tema; il docking impila le regioni contro i bordi del genitore e riempie ciò che resta.
inventario visivo	L'elenco scritto di controlli che hanno bisogno di appearance, token di colori e font, regioni del layout e breakpoint, da cui si pianifica il lavoro su tema, layout e profili.

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 1. Puoi aprire tu stesso [module-01.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 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 1 - "Il sistema visivo di Wisej.NET" - 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-01.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: Crea la soluzione Adaptive Operations Console e il suo primo shell desktop: una Page principale con un pannello toolbar in alto, un pannello di navigazione a sinistra, un'area di lavoro su Fill, un pannello dei dettagli a destra e un pannello di stato in basso, costruiti con Dock e Panel annidati. Scegli un tema di base integrato e annota la scelta, scrivi un `VisualInventory.md` che elenca i controlli che hanno bisogno di appearance personalizzate, i token con nome per colori e font, le regioni del layout e i breakpoint desktop, tablet e telefono, e aggiungi una label di stato che riporta la larghezza corrente del browser così che i moduli successivi possano verificare i profili.
3. Produci questi deliverable e dimmi dove si trova ciascuno nella soluzione:
 - Soluzione Adaptive Operations Console con uno shell desktop in dock composto dai pannelli toolbar, navigazione, area di lavoro, dettagli e stato
 - Tema di base scelto e configurato, con il motivo annotato negli appunti del lab
 - `VisualInventory.md` che elenca i controlli che hanno bisogno di appearance, i token del tema, le regioni del layout e i breakpoint
 - Label di stato che riporta la larghezza corrente del browser
 - Screenshot desktop e una nota che spiega quale livello possiede ogni decisione visiva
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 comando principale della toolbar ha bisogno del colore del brand su ogni schermata. Quale dei tre livelli possiede questa decisione, e che cosa andrebbe storto se invece impostassi `BackColor` sul pulsante?
- La tua label della larghezza indica 1024 px e il pannello dei dettagli è ancora visibile. Quale meccanismo di un modulo successivo deciderà se deve esserlo, e perché in Wisej.NET una media query CSS è la prima risposta sbagliata?
- Se domani il tema di base venisse sostituito con un altro tema integrato, quali righe del tuo `VisualInventory.md` cambierebbero e quali no, e che cosa ti dice questo su dove vivono le decisioni dell'inventario?

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 comando principale della toolbar ha bisogno del colore del brand su ogni schermata. Quale dei tre livelli possiede questa decisione, e che cosa andrebbe storto se invece impostassi BackColor sul pulsante?
- ✓ La tua label della larghezza indica 1024 px e il pannello dei dettagli è ancora visibile. Quale meccanismo di un modulo successivo deciderà se deve esserlo, e perché in Wisej.NET una media query CSS è la prima risposta sbagliata?
- ✓ Se domani il tema di base venisse sostituito con un altro tema integrato, quali righe del tuo VisualInventory.md cambierebbero e quali no, e che cosa ti dice questo su dove vivono le decisioni dell'inventario?

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 1 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 1 su GitHub](#)

Oppure clona tutto con `git clone https://github.com/iceteagroup/LearnWisej-Samples` e apri `Theming and Responsive Layouts Course/Module 1`.



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.

- ✓ **Aggiungi il secondo caso.** *"Estendi questo progetto AdaptiveOps 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."*
- ✓ **Dimostra che funziona davvero.** *"Passa in rassegna questo progetto deliverable per deliverable rispetto alla specifica del modulo su <https://learnwisej.com/assets/courses/theming-and-responsive-layouts/specs/module-01.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 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."*



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.