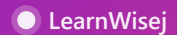


Esercizio di coding con l'IA

Costruisci il progetto ValidationClinic 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 il form di registrazione clienti di ValidationClinic con i TextBox txtName e txtEmail, un pulsante Save e un pulsante Cancel. Collega gestori Validating che impostano e.Cancel = true e un messaggio dell'ErrorProvider quando il nome è vuoto o l'email non contiene @, cancellano il messaggio quando il valore è valido, poi toglie gli spazi dal nome nel suo gestore Validated. Imposta la modalità AutoValidate del form, assicurati che Cancel non avvii la validazione così che l'utente possa sempre uscire, e chiama ValidateChildren() all'inizio di btnSave_Click così che Save si fermi quando un campo non è valido. Concludi con una breve nota del lab che spiega il comportamento del focus che hai scelto.

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

- ✓ Form di registrazione clienti con txtName e txtEmail, un pulsante Save e un pulsante Cancel
- ✓ Gestori Validating che impostano e.Cancel = true e un messaggio di errore per un nome vuoto e un'email non valida
- ✓ Gestore Validated che toglie gli spazi dal nome accettato
- ✓ Gestore di Save che chiama ValidateChildren() e si ferma quando restituisce false
- ✓ Nota del lab che indica la modalità AutoValidate scelta e perché Cancel funziona anche su un form non valido

Perché è fatto così

Il form di registrazione clienti di ValidationClinic oggi accetta qualsiasi cosa: un nome vuoto, un'email senza @ o un clic su Save in un form che nessuno ha compilato. La prima idea della maggior parte dei principianti è un'istruzione if dentro il pulsante Save, ma così l'utente non riceve alcun feedback mentre lavora e Save diventa un muro di errori alla fine di tutto. Questo modulo mette la validazione dove deve stare in un form Wisej.NET: su ogni editor quando l'utente lo lascia, e un'altra volta quando l'utente fa clic su Save.

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
validazione	La decisione se un valore è accettabile prima che l'applicazione lo usi, seguita da un feedback visibile e, quando il valore è sbagliato, dal rifiuto di proseguire.
Validating	L'evento che un controllo genera quando l'utente prova a lasciarlo (o quando il codice chiede la validazione); il suo gestore legge il valore ed è l'unico punto in cui si decide se accettarlo o rifiutarlo.
CancelEventArgs.Cancel	Il flag che un gestore Validating imposta a true per rifiutare il valore: il controllo risulta non valido, il focus può essere trattenuto al suo interno e ValidateChildren() segnala il fallimento.
Validated	L'evento generato solo dopo che Validating è terminato senza annullare; il posto per le rifiniture come togliere gli spazi dal testo, mai per salvare.
AutoValidate	Una proprietà di form, pagina o user control che decide che cosa succede al focus quando un controllo figlio non supera la validazione.
CausesValidation	Un interruttore per singolo controllo; se è impostato a false su un pulsante come Cancel, il clic non valida il campo che l'utente sta lasciando.
ValidateChildren()	Un metodo del contenitore che esegue la validazione dei suoi controlli figli, compresi i campi che l'utente non ha mai visitato, e restituisce false se un gestore ha annullato.

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 ValidationClinic 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 - "Fondamenti di validazione in Wisej.NET" - del corso LearnWisej "Validazione 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/validation-in-wisej-net/specs/module-01.md>

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

1. Progetto "ValidationClinic", 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 il form di registrazione clienti di ValidationClinic con i TextBox txtName e txtEmail, un pulsante Save e un pulsante Cancel. Collega gestori Validating che impostano e.Cancel = true e un messaggio dell'ErrorProvider quando il nome è vuoto o l'email non contiene @, cancellano il messaggio quando il valore è valido, poi toglie gli spazi dal nome nel suo gestore Validated. Imposta la modalità AutoValidate del form, assicurati che Cancel non avvii la validazione così che l'utente possa sempre uscire, e chiama ValidateChildren() all'inizio di btnSave_Click così che Save si fermi quando un campo non è valido. Concludi con una breve nota del lab che spiega il comportamento del focus che hai scelto.
3. Produci questi deliverable e dimmi dove si trova ciascuno nella soluzione:
 - Form di registrazione clienti con txtName e txtEmail, un pulsante Save e un pulsante Cancel
 - Gestori Validating che impostano e.Cancel = true e un messaggio di errore per un nome vuoto e un'email non valida
 - Gestore Validated che toglie gli spazi dal nome accettato
 - Gestore di Save che chiama ValidateChildren() e si ferma quando restituisce false
 - Nota del lab che indica la modalità AutoValidate scelta e perché Cancel funziona anche su un form non valido
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:

- Togli e.Cancel = true da txtEmail_Validating ma lasci la chiamata a SetError. Descrivi con precisione che cosa vede l'utente quando esce con Tab da "maria.example.com" e poi fa clic su Save.
- Quali delle tue verifiche vengono eseguite mentre l'utente sta modificando, e quali solo perché Save ha chiamato ValidateChildren()? Indica un'azione dell'utente che arriva a Save con un campo il cui gestore non è mai stato eseguito.
- Passi il form da EnablePreventFocusChange a EnableAllowFocusChange. Che cosa cambia per l'utente, che cosa resta uguale, e btnCancel.CausesValidation = false serve ancora?

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:

- ✓ Togli `e.Cancel = true` da `txtEmail_Validating` ma lasci la chiamata a `SetError`. Descrivi con precisione che cosa vede l'utente quando esce con Tab da "maria.example.com" e poi fa clic su Save.
- ✓ Quali delle tue verifiche vengono eseguite mentre l'utente sta modificando, e quali solo perché Save ha chiamato `ValidateChildren()`? Indica un'azione dell'utente che arriva a Save con un campo il cui gestore non è mai stato eseguito.
- ✓ Passi il form da `EnablePreventFocusChange` a `EnableAllowFocusChange`. Che cosa cambia per l'utente, che cosa resta uguale, e `btnCancel.CausesValidation = false` serve ancora?

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 `Validation in Wisej.NET 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.

- ✓ **Sostituisci un pezzo senza spostare le giunzioni.** *"Scegli la parte di questo progetto ValidationClinic 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 ValidationClinic 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 ValidationClinic 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.