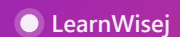


Ejercicio de programación con IA

Construye el ejemplo ValidationClinic de este módulo con ChatGPT o Claude a partir de un prompt listo para usar que dirige al modelo a la especificación del módulo, y después revisa, ejecuta y amplía lo que te devuelva — incluso comparándolo con nuestra propia solución de referencia en GitHub.



No vas a teclear este ejemplo. Vas a dirigirlo.

Ya casi nadie escribe código Wisej.NET partiendo de un archivo vacío — un asistente escribe el primer borrador y el desarrollador decide si es correcto. Eso solo funciona si dominas el tema, y eso es lo que te dio el resto de este módulo. Aquí le pasas ese conocimiento a ChatGPT o a Claude, recibes el ejemplo de este módulo como una solución de Visual Studio que funciona y lo revisas como revisarías el pull request de un compañero.

Qué estás pidiendo

Crea el formulario de alta de clientes de ValidationClinic con los TextBox txtName y txtEmail, un botón Save y un botón Cancel. Asocia controladores Validating que establezcan e.Cancel = true y un mensaje de ErrorProvider cuando el nombre esté vacío o el correo electrónico no contenga @, que borren el mensaje cuando el valor sea válido, y recorta los espacios del nombre en su controlador Validated. Establece el modo AutoValidate del formulario, asegúrate de que Cancel no provoque validación para que el usuario siempre pueda salir, y llama a ValidateChildren() al principio de btnSave_Click para que Save se detenga cuando algún campo no sea válido. Termina con una breve nota del laboratorio que explique el comportamiento del foco que has elegido.

El módulo pide estos entregables, y el prompt de abajo le pide al asistente exactamente esos:

- ✓ Formulario de alta de clientes con txtName y txtEmail, un botón Save y un botón Cancel
- ✓ Controladores Validating que establecen e.Cancel = true y un mensaje de error para un nombre vacío y un correo electrónico no válido
- ✓ Controlador Validated que recorta el nombre aceptado
- ✓ Controlador de Save que llama a ValidateChildren() y se detiene cuando devuelve false
- ✓ Nota del laboratorio que indica el modo AutoValidate elegido y por qué Cancel sigue funcionando en un formulario no válido

Por qué tiene esta forma

El formulario de alta de clientes de ValidationClinic acepta ahora cualquier cosa: un nombre vacío, un correo electrónico sin @ o un clic en Save en un formulario que nadie ha rellenado. La primera idea que tienen casi todos los principiantes es poner una instrucción if dentro del botón Save, pero eso no da

ningún aviso al usuario mientras trabaja y convierte Save en un muro de errores al final. Este módulo coloca la validación donde le corresponde en un formulario de Wisej.NET: en cada editor cuando el usuario sale de él, y una vez más cuando el usuario hace clic en Save.

Estos son los términos que el asistente tiene que dominar. Si lo que te devuelve los usa a la ligera, no ha entendido el módulo:

Término	Qué significa aquí
validación	La decisión de si un valor es aceptable antes de que la aplicación lo use, seguida de un aviso visible y, cuando el valor es incorrecto, de la negativa a continuar.
Validating	El evento que desencadena un control cuando el usuario intenta salir de él (o cuando el código pide una validación); su controlador lee el valor y es el único lugar donde se decide aceptar o rechazar.
CancelEventArgs.Cancel	El indicador que un controlador Validating establece en true para rechazar el valor: el control cuenta como no válido, el foco puede retenerse en él y ValidateChildren() informa del fallo.
Validated	El evento que se desencadena solo después de que Validating haya terminado sin cancelar; el lugar para tareas de limpieza como recortar texto, nunca para guardar.
AutoValidate	Una propiedad de un formulario, una página o un control de usuario que decide qué ocurre con el foco cuando un control hijo no supera la validación.
CausesValidation	Un interruptor por control; establecido en false en un botón como Cancel, hacer clic en él no valida el campo que el usuario está abandonando.
ValidateChildren()	Un método del contenedor que ejecuta la validación de sus controles hijos, incluidos los campos que el usuario nunca visitó, y devuelve false si algún controlador canceló.

Constrúyelo — elige tu asistente

Los dos botones abren una pestaña nueva con el **mismo prompt listo para usar** ya escrito. El prompt dirige al modelo a la especificación de este módulo — un único archivo Markdown con la lección, el enunciado del laboratorio y los entregables — para que trabaje con el material real en lugar de adivinar de qué trata el Módulo 1. Puedes abrir tú mismo [module-01.md](#) para ver exactamente lo que leerá.



Usar ChatGPT

Abre [chatgpt.com](#) con el prompt de este módulo ya cargado. Funciona con GPT-5 o Codex; cualquier plan que pueda navegar por la web descargará la especificación.



Usar Claude

Abre [claude.ai](#) con el mismo prompt ya cargado. Si usas Claude Code, puedes pegarlo en una terminal abierta en una carpeta vacía.



Envíalo tal cual y obtendrás un proyecto, no un fragmento

El prompt termina pidiendo toda la solución como un único `.zip` con los pasos para compilarla y ejecutarla. Descárgalo, abre la solución en Visual Studio, pulsa F5 y ValidationClinic se abrirá en `localhost`. Necesitas el SDK de .NET y el paquete NuGet `Wisej-4`; todo lo demás, el prompt le pide al asistente que lo escriba él mismo.

Estoy siguiendo el Módulo 1 – «Fundamentos de la validación en Wisej.NET» – del curso de LearnWisej «Validación en Wisej.NET», y quiero construir contigo el ejemplo de este módulo. Respóndeme en español.

Empieza leyendo la especificación del módulo. Es un único archivo Markdown (en inglés) que contiene la lección completa, el enunciado del laboratorio y los entregables:

<https://learnwisej.com/assets/courses/validation-in-wisej-net/specs/module-01.md>

Después construye un proyecto Wisej.NET 4 completo que pueda abrir en Visual Studio:

1. Proyecto «ValidationClinic», C#, net8.0 o posterior, paquete NuGet Wisej-4 4.1.x, host Kestrel que llame a `app.UseWisej()`.
2. Construye lo que pide el laboratorio de la especificación: Crea el formulario de alta de clientes de ValidationClinic con los TextBox `txtName` y `txtEmail`, un botón Save y un botón Cancel. Asocia controladores Validating que establezcan `e.Cancel = true` y un mensaje de ErrorProvider cuando el nombre esté vacío o el correo electrónico no contenga @, que borren el mensaje cuando el valor sea válido, y recorta los espacios del nombre en su controlador Validated. Establece el modo AutoValidate del formulario, asegúrate de que Cancel no provoque validación para que el usuario siempre pueda salir, y llama a `ValidateChildren()` al principio de `btnSave_Click` para que Save se detenga cuando algún campo no sea válido. Termina con una breve nota del laboratorio que explique el comportamiento del foco que has elegido.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Formulario de alta de clientes con `txtName` y `txtEmail`, un botón Save y un botón Cancel
 - Controladores Validating que establecen `e.Cancel = true` y un mensaje de error para un nombre vacío y un correo electrónico no válido
 - Controlador Validated que recorta el nombre aceptado
 - Controlador de Save que llama a `ValidateChildren()` y se detiene cuando devuelve false
 - Nota del laboratorio que indica el modo AutoValidate elegido y por qué Cancel sigue funcionando en un formulario no válido
4. Sigue la especificación siempre que nombre una clase, propiedad o evento de Wisej.NET – usa exactamente ese. Donde deje la elección abierta, dime qué API elegiste y por qué.
5. Mantén la autoridad en el servidor: el estado y la validación viven en .NET, y el navegador muestra e informa. Nunca traslades una decisión a JavaScript porque allí resulte más fácil.
6. Haz que el resultado se pueda demostrar: debo poder ejecutarlo y ver el comportamiento del que trata este módulo, incluido lo que ocurre cuando algo sale mal.

Antes de escribir una sola línea de código, explica con tus propias palabras qué vas a construir y dónde vivirá cada pieza, para que pueda corregir el diseño mientras todavía es barato.

Cuando el código esté terminado, empaqueta toda la solución en un único archivo .zip descargable y dame los pasos exactos para compilarla y ejecutarla.

Por último, responde a estas preguntas de la especificación usando tu propia implementación como prueba:

- Quitas `e.Cancel = true` de `txtEmail_Validating` pero mantienes la llamada a `SetError`. Describe exactamente qué ve el usuario cuando sale con Tab de «maria.example.com» y después hace clic en Save.
- ¿Cuáles de tus comprobaciones se ejecutan mientras el usuario edita y cuáles solo porque Save ha llamado a `ValidateChildren()`? Indica una acción del usuario que llegue a Save con un

campo cuyo controlador nunca se ha ejecutado.

– Cambias el formulario de `EnablePreventFocusChange` a `EnableAllowFocusChange`. ¿Qué cambia para el usuario, qué sigue igual y sigue importando `btnCancel.CausesValidation = false`?

Pregúntame por cualquier cosa ambigua antes de empezar.

Después revisa lo que te devuelve

Que el código compile no significa que sea correcto. Las preguntas interesantes son de diseño: ¿es este el control de Wisej.NET que la tarea realmente necesita?, ¿sigue el ciclo de vida que te ofrece el framework?, ¿el estado de referencia está en el servidor o se quedó en el navegador porque era más cómodo? Lee el resultado como revisor, no como cliente.

Este módulo te da la lista de comprobación. Responde a cada punto basándote en el código que acabas de recibir:

- ✓ Quitas `e.Cancel = true` de `txtEmail_Validating` pero mantienes la llamada a `SetError`. Describe exactamente qué ve el usuario cuando sale con Tab de «`maria.example.com`» y después hace clic en Save.
- ✓ ¿Cuáles de tus comprobaciones se ejecutan mientras el usuario edita y cuáles solo porque Save ha llamado a `ValidateChildren()`? Indica una acción del usuario que llegue a Save con un campo cuyo controlador nunca se ha ejecutado.
- ✓ Cambias el formulario de `EnablePreventFocusChange` a `EnableAllowFocusChange`. ¿Qué cambia para el usuario, qué sigue igual y sigue importando `btnCancel.CausesValidation = false`?

Todo lo que hayas tenido que corregir merece quedar por escrito. Una nota breve con las convenciones y decisiones que acordaste, pegada en el primer mensaje de tu próxima sesión, vale más que cualquier prompt ingenioso.



Este ya lo construimos — compara tu versión con la nuestra

La implementación de referencia del Módulo 1 está en el repositorio público de ejemplos. Construye primero la tuya y después abre la nuestra: lo interesante son las diferencias.



[Abrir el ejemplo del Módulo 1 en GitHub](#)

O clona el conjunto completo con `git clone https://github.com/iceteagroup/LearnWisej-Samples` y abre `Validation in Wisej.NET Course/Module 1`.



Nuestro ejemplo es un punto de partida, no un producto terminado

Se construyó igual que estás construyendo el tuyo — con un prompt y luego corregido — y se detiene a propósito donde se detiene el módulo. Espera soluciones provisionales donde iría código de producción, un manejo de errores mínimo y ninguna prueba. Tómallo como una invitación: usa lo que aprendiste en este módulo, junto con el asistente, para llevarlo más lejos. Mejorar código que no escribiste tú es precisamente la habilidad que enseña este curso.

Tres prompts para ir más allá

Dirige tu asistente al proyecto que acabas de construir — o al nuestro y prueba estos. Cada uno es un ejercicio de revisión y ampliación, no una reescritura.

- ✓ **Cambia una pieza sin mover las juntas.** *"Elige la parte de este proyecto ValidationClinic que sea más claramente un sustituto de algo real — una fuente de datos falsa, un almacén en memoria, un servicio simplificado — y reemplázala por una implementación con forma de producción. El contrato a ambos lados no debe cambiar: los mismos tipos, las mismas llamadas, el mismo comportamiento ante fallos. Dime qué te obligó a gestionar la implementación real que el sustituto no gestionaba."*
- ✓ **Haz que aguante un mal día.** *"Revisa este proyecto ValidationClinic en busca de los modos de fallo que todavía no gestiona — una entrada incorrecta o ausente, una dependencia lenta o no disponible, trabajo que llega después de que el usuario haya salido de la pantalla y dos acciones que compiten entre sí. Para cada uno, dime primero si es un riesgo real en Wisej.NET 4 y por qué, y solo después escribe la corrección. Si decides que algo no necesita gestión, dilo y explica el razonamiento en lugar de añadir código defensivo."*
- ✓ **Añade el segundo caso.** *"Amplía este proyecto ValidationClinic con un segundo caso que siga el mismo patrón que el primero — otra entidad, pantalla o control con la misma forma, conectado de la misma manera. Extrae a una clase base o a un servicio lo que de verdad sea común, pero solo donde se justifique; dos cosas parecidas no siempre piden una abstracción. Muéstrame el diff y justifica cada cosa que hayas extraído."*



Llévalo al laboratorio

El laboratorio práctico que cierra este módulo pide los mismos entregables que acabas de hacer que generara el asistente. Usa lo que tienes aquí como forma de tu respuesta y como prueba que la respalda — no como la respuesta en sí. El laboratorio se califica por tu razonamiento, y esa parte tiene que ser tuya.