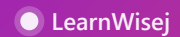


Ejercicio de programación con IA

Construye el ejemplo EnterpriseOps 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.



Instrúyelo, léelo y luego rómpelo.

Este ejercicio tiene tres partes, y solo la primera trata del prompt. Le das al asistente la especificación del módulo y recibes una solución EnterpriseOps; lees lo que escribió y decides si de verdad hace lo que pedía el módulo; y después lo llevas más allá de lo que pedía el módulo. En la segunda y la tercera parte es donde se aprende.

Qué estás pidiendo

Construye un control StatusTimeline reutilizable y un Widget de gráfico o visualización encapsulado para el historial de las órdenes de trabajo. Expón propiedades C#, eventos de servidor, recursos incrustados y un modo de ejemplo en tiempo de diseño.

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

- ✓ UserControl StatusTimeline
- ✓ Envoltorio de Widget o control personalizado
- ✓ Paquete de recursos incrustados
- ✓ Documento del contrato de eventos
- ✓ Pantalla de ejemplo de uso

Por qué tiene esta forma

Los equipos avanzados suelen necesitar componentes empresariales reutilizables o widgets JavaScript de terceros. El reto es exponerlos como controles de Wisej.NET mantenibles sin perder la productividad del diseñador ni la seguridad del lado del servidor.

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
UserControl personalizado	Un control compuesto construido a partir de controles existentes de Wisej.NET que empaqueta el diseño y el comportamiento para reutilizarlos en varias pantallas.
control heredado	Una subclase de un control integrado que añade valores predeterminados, validación o comportamiento sin perder el soporte completo del diseñador y de los temas.
envoltorio de Widget	Un control que aloja una biblioteca JavaScript de terceros dentro de Wisej.NET. Solo debería exponer valores del lado del navegador, mientras las reglas de negocio permanecen en el servidor.
recursos incrustados	Scripts, estilos e imágenes compilados en el ensamblado del control para que el componente se distribuya como un único paquete sin archivos sueltos que desplegar.
contrato de eventos cliente/servidor	El conjunto acordado de eventos y formas de carga útil que se intercambian el widget del navegador y su control de servidor, documentado para que ambos lados puedan evolucionar con seguridad.
valores predeterminados del diseñador	Valores de propiedades y atributos predeterminados razonables para que un control colocado en un formulario funcione de inmediato y tenga un aspecto coherente con el resto de la aplicación.
API del componente	Las propiedades, los métodos y los eventos públicos que expone un control reutilizable. Una API pequeña e intencionada es más fácil de mantener que una que filtra detalles de implementación.

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 8. Puedes abrir tú mismo [module-08.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 EnterpriseOps 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 8 – «Custom Controls, Extensions, Widget Wrappers, Reusable Components» – del curso de LearnWisej «Enterprise Wisej.NET: Architecture to Cloud», 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/azure-docker/specs/module-08.md>

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

1. Proyecto «EnterpriseOps», 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: Construye un control `StatusTimeline` reutilizable y un Widget de gráfico o visualización encapsulado para el historial de las órdenes de trabajo. Expón propiedades C#, eventos de servidor, recursos incrustados y un modo de ejemplo en tiempo de diseño.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - `UserControl StatusTimeline`
 - Envoltorio de Widget o control personalizado
 - Paquete de recursos incrustados
 - Documento del contrato de eventos
 - Pantalla de ejemplo de uso
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:

- ¿Puede otro desarrollador usar el componente sin leer su funcionamiento interno?
- ¿Qué datos pasan del servidor al cliente?
- ¿Se renderiza el componente de forma segura en el diseñador?

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:

- ✓ ¿Puede otro desarrollador usar el componente sin leer su funcionamiento interno?
- ✓ ¿Qué datos pasan del servidor al cliente?
- ✓ ¿Se renderiza el componente de forma segura en el diseñador?

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 8 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 8 en GitHub](#)

O clona el conjunto completo con `git clone https://github.com/iceteagroup/LearnWisej-Samples` y abre `Enterprise Wisej.NET Course/Module 8`.



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.

- ✓ **Haz que aguante un mal día.** *"Revisa este proyecto EnterpriseOps 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 EnterpriseOps 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."*
- ✓ **Demuestra que de verdad funciona.** *"Recorre este proyecto entregable por entregable frente a la especificación del módulo en <https://learnwisej.com/assets/courses/azure-docker/specs/module-08.md> y muéstrame la prueba de que cada uno está realmente terminado — la ruta del código, dónde hacer clic y qué debería ver. Si un entregable solo se cumple en parte, dilo claramente en lugar de justificarlo. Después añade lo que falte."*



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.