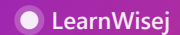


Ejercicio de programación con IA

Construye el ejemplo VisualOperationsStudio 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.



El mismo módulo, construido como de verdad vas a construir.

Todas las demás secciones de este módulo explican cómo funciona «Canvas interactivo: estado, entrada, prueba de impacto y rendimiento». Esta te pide que lo generes — dirigiendo a un asistente con la propia especificación del módulo y comprobando después que el resultado se ejecuta y hace lo que describe el enunciado del laboratorio. Es el flujo de trabajo que usarás en proyectos reales, practicado con algo cuya respuesta correcta ya conoces.

Qué estás pidiendo

Convierte el área de pruebas de Canvas en un editor de topología para VisualOperationsStudio. Crea las clases NodeModel y EdgeModel con identificadores estables y geometría en coordenadas de mundo, guarda la escena en un objeto TopologyScene almacenado por sesión en lugar de en un campo estático, y deja RenderScene como el único método que dibuja. Añade conversiones ToWorld y ToScreen para el desplazamiento y el factor de zoom actuales; después gestiona MouseDown para hacer la prueba de impacto contra los límites de los nodos en coordenadas de mundo y marcar Selected, MouseMove para arrastrar el nodo seleccionado o desplazar el viewport, y MouseUp para terminar el gesto, cambiando cada uno el estado y llamando después a RenderScene. Añade zoom con la rueda o con botones centrado en el puntero, omite al dibujar los nodos que quedan fuera del rectángulo visible, da a quien usa el teclado una forma de seleccionar y desplazar un nodo, y registra el tráfico de peticiones que genera un arrastre junto con el punto a partir del cual llevarías el bucle a JavaScript en el cliente.

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

- ✓ NodeModel, EdgeModel y un TopologyScene por sesión con identificadores estables y sin estado estático mutable
- ✓ Conversiones ToWorld y ToScreen usadas en la prueba de impacto para que la selección siga siendo correcta tras desplazar y hacer zoom
- ✓ Controladores de MouseDown, MouseMove y MouseUp que cambian el estado y después llaman al único método RenderScene
- ✓ Descarte de los nodos que quedan fuera del rectángulo visible, más una vía de teclado para seleccionar y desplazar
- ✓ InteractionNotes.md con el tráfico de peticiones medido durante un arrastre y el límite a partir del cual pasarías al cliente

Por qué tiene esta forma

El Módulo 4 dejó a VisualOperationsStudio con un área de pruebas de Canvas: un método DrawPlayground al que llama el evento Redraw y que reconstruye desde cero todos los trazados, degradados y transformaciones. Es correcto, y es completamente inerte. El personal de operaciones quiere hacer clic en una bomba, arrastrarla a un sitio razonable, desplazarse por la planta y ampliar un grupo de elementos, y un método de dibujo no puede responder a ninguna de esas peticiones, porque el canvas del navegador guarda píxeles y los píxeles no saben qué forma es una bomba. En cuanto una superficie se vuelve interactiva deja de ser un renderizador y pasa a ser una pequeña máquina de estados que, además, dibuja.

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
Modelo, Entrada, Estado, Render	Cuatro capas con una regla entre ellas: la entrada cambia el estado y solo entonces un único método de render convierte el estado actual en llamadas de dibujo.
Objetos de escena	Clases C# como NodeModel y EdgeModel que llevan el identificador estable, los límites en espacio de mundo, el estado y la marca Selected que los píxeles solo pueden insinuar.
Estado de escena por sesión	La escena pertenece a la sesión de un usuario; una lista static mutable daría a todos los visitantes los mismos nodos y la misma selección.
Eventos de entrada de Control	Canvas deriva de Control, así que están disponibles mouse down/move/up, clic, doble clic y los eventos táctiles y de gestos como TouchStart, Tap, Pinch, Rotate, Swipe y Track.
Prueba de impacto basada en el modelo	La API de Canvas de Wisej documentada no expone isPointInPath ni isPointInStroke, así que «¿en qué ha hecho clic el usuario?» se responde comparando un punto con una geometría que te pertenece.
Conversión de pantalla a mundo	Si el renderizado aplica $\text{pantalla} = \text{mundo} \times \text{zoom} + \text{desplazamiento}$, entonces cada coordenada de puntero entrante hay que pasarla por la inversa antes de compararla con nada del modelo.
Descarte por viewport	Omitir los objetos de la escena cuyos límites no intersecan el rectángulo de mundo visible, para que una topología grande cueste solo lo que hay realmente en pantalla.

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 5. Puedes abrir tú mismo [module-05.md](#) para ver exactamente lo que leerá.



Usar ChatGPT



Usar Claude

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.

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 VisualOperationsStudio 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 5 – «Canvas interactivo: estado, entrada, prueba de impacto y rendimiento» – del curso de LearnWisej «Dibujo y pintura 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/drawing-and-painting/specs/module-05.md>

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

1. Proyecto «VisualOperationsStudio», 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: Convierte el área de pruebas de Canvas en un editor de topología para VisualOperationsStudio. Crea las clases `NodeModel` y `EdgeModel` con identificadores estables y geometría en coordenadas de mundo, guarda la escena en un objeto `TopologyScene` almacenado por sesión en lugar de en un campo estático, y deja `RenderScene` como el único método que dibuja. Añade conversiones `ToWorld` y `ToScreen` para el desplazamiento y el factor de zoom actuales; después gestiona `MouseDown` para hacer la prueba de impacto contra los límites de los nodos en coordenadas de mundo y marcar `Selected`, `MouseMove` para arrastrar el nodo seleccionado o desplazar el viewport, y `MouseUp` para terminar el gesto, cambiando cada uno el estado y llamando después a `RenderScene`. Añade zoom con la rueda o con botones centrado en el puntero, omite al dibujar los nodos que quedan fuera del rectángulo visible, da a quien usa el teclado una forma de seleccionar y desplazar un nodo, y registra el tráfico de peticiones que genera un arrastre junto con el punto a partir del cual llevarías el bucle a JavaScript en el cliente.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - `NodeModel`, `EdgeModel` y un `TopologyScene` por sesión con identificadores estables y sin estado estático mutable
 - Conversiones `ToWorld` y `ToScreen` usadas en la prueba de impacto para que la selección siga siendo correcta tras desplazar y hacer zoom
 - Controladores de `MouseDown`, `MouseMove` y `MouseUp` que cambian el estado y después llaman al único método `RenderScene`
 - Descarte de los nodos que quedan fuera del rectángulo visible, más una vía de teclado para seleccionar y desplazar
 - `InteractionNotes.md` con el tráfico de peticiones medido durante un arrastre y el límite a partir del cual pasarías al cliente
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:

- Un compañero informa de que la selección va bien hasta que amplía, y a partir de ahí los clics caen sobre la bomba equivocada o sobre nada. El código de dibujo no ha cambiado. ¿Qué única conversión falta y por qué el defecto se mantiene invisible con zoom 1,0 y sin desplazamiento?
- Dos operarios abren la topología a la vez y cada uno ve moverse bajo su cursor el nodo del otro. ¿Dónde se guardó la escena y qué te dice eso sobre dónde corresponde el estado interactivo en un framework web del lado del servidor?
- Tu arrastre produce una petición por cada movimiento del puntero. ¿En qué punto medido dejarías de optimizar el bucle del servidor y llevarías la interacción a JavaScript en el cliente, y qué tendría que seguir siendo autoritativo en el servidor?

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:

- ✓ Un compañero informa de que la selección va bien hasta que amplía, y a partir de ahí los clics caen sobre la bomba equivocada o sobre nada. El código de dibujo no ha cambiado. ¿Qué única conversión falta y por qué el defecto se mantiene invisible con zoom 1,0 y sin desplazamiento?
- ✓ Dos operarios abren la topología a la vez y cada uno ve moverse bajo su cursor el nodo del otro. ¿Dónde se guardó la escena y qué te dice eso sobre dónde corresponde el estado interactivo en un framework web del lado del servidor?
- ✓ Tu arrastre produce una petición por cada movimiento del puntero. ¿En qué punto medido dejarías de optimizar el bucle del servidor y llevarías la interacción a JavaScript en el cliente, y qué tendría que seguir siendo autoritativo en el servidor?

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.



Todavía no hay un ejemplo de referencia para este curso — el tuyo es el primero

El [repositorio de ejemplos de LearnWisej](#) tiene soluciones completas para la mayoría de los cursos, pero todavía no para este. Eso significa que no hay nada con qué compararte ni ningún atajo: la especificación y tu propia revisión son toda tu red de seguridad. Guarda lo que construyas — y si sale bien, nos encantaría verlo.

Tres prompts para ir más allá

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

- ✓ **Haz que aguante un mal día.** *"Revisa este proyecto VisualOperationsStudio 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 VisualOperationsStudio 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/drawing-and-painting/specs/module-05.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.