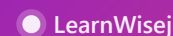


Ejercicio de programación con IA

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 asistente pone las manos rápidas. El arquitecto sigues siendo tú.

Un asistente puede generar el ejemplo de este módulo en minutos. Lo que no puede hacer es decidir si el resultado es el diseño adecuado para tu aplicación — esa decisión sigue siendo tuya. Tú pones el criterio: dale unas buenas instrucciones y luego contrasta lo que te devuelva con lo que te acaba de enseñar el Módulo 7.

Qué estás pidiendo

Completa el proyecto final de VisualOperationsStudio poniendo un único modelo detrás de las cuatro superficies. Reestructura el proyecto en carpetas Models, Geometry, Renderers y de adaptadores de Wisej, de modo que el indicador dibujado, las celdas de cuadrícula dibujadas por el usuario, la topología en Canvas y la exportación a PNG lean todos el mismo OperationsModel y compartan la capa de geometría pura, sin duplicar estado de negocio en cada renderizador. Añade un ayudante RenderMetrics que registre la duración del renderizado, el tamaño del mapa de bits de salida y el número de objetos de la escena dibujados, y úsalo para comparar el indicador dibujado en el servidor con el renderizador fuera de pantalla, antes y después de una optimización que elijas y justifiques. Ejercita las rutas de fallo —datos vacíos, una superficie de tamaño cero, un rango no válido, una fuente ausente— para que cada una degrade a un marcador de posición legible en lugar de provocar una excepción, añade una tabla accesible que enumere todos los valores que muestran los gráficos y redacta las notas del proyecto final cubriendo la arquitectura, la frontera entre navegador y servidor y la matriz de pruebas.

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

- ✓ Capas Models, Geometry, Renderers y de adaptadores con un único OperationsModel que alimenta las cuatro superficies
- ✓ Ayudante RenderMetrics que registra la duración del renderizado, el tamaño del mapa de bits y el número de objetos dibujados por superficie
- ✓ Mediciones antes y después de una optimización, con el razonamiento de por qué la elegiste
- ✓ Rutas de fallo para datos vacíos, tamaño cero, rango no válido y fuente ausente que degradan en lugar de lanzar excepciones
- ✓ CapstoneNotes.md con la arquitectura, la frontera servidor/navegador, la matriz de pruebas y una tabla de datos accesible

Por qué tiene esta forma

VisualOperationsStudio es ya dueño de cuatro superficies de renderizado, y cada una se construyó en su propio módulo con su propia idea privada de lo que significa la salud de una máquina. Así es como se pudre un subsistema de dibujo: alguien mueve un umbral de advertencia, tres superficies coinciden y el PNG exportado contradice en silencio la pantalla de la que salió. Este módulo funde esas cuatro interpretaciones en un único modelo tras una única capa de geometría y, después, demuestra con mediciones, rutas de fallo deliberadas y un equivalente semántico que el resultado sobrevive a datos reales, plataformas reales y usuarios reales.

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 visual	Solo estado semántico: lecturas, rótulos, nodos y aristas, series de tendencia, la severidad como decisión de dominio y la selección compartida, para que ninguna superficie necesite una copia.
Capa de geometría	Cálculo casi puro que convierte el estado del modelo más el tamaño actual de la superficie en rectángulos, ángulos de barrido, posiciones de etiqueta y regiones de impacto, comprobable sin ningún control presente.
Adaptador de superficie	Un renderizador por superficie en lugar de una interfaz común: System.Drawing.Graphics y Wisej.Web.Canvas conservan sus propias primitivas sobre una geometría compartida.
Presupuesto de rendimiento	Una cifra acordada antes de optimizar, por ejemplo un control de estado dibujado por debajo de 5 ms y un redibujado completo de Canvas dentro de unos 16 a 50 ms.
Instrumentación del renderizado	El tiempo transcurrido registrado, las dimensiones del mapa de bits, el número de objetos dibujados y el crecimiento de la carga útil por superficie, para que una optimización se argumente a partir de un antes y un después.
Prueba de imagen de referencia	Un renderizado fuera de pantalla con entrada fija comparado con una referencia guardada usando una tolerancia, porque el suavizado de bordes y los codificadores difieren entre plataformas.
Equivalente semántico	La etiqueta, el AccessibleDescription, el tooltip o la lista acompañante que llevan todos los valores esenciales que muestran los gráficos, para que nada importante exista solo como píxeles.

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 7. Puedes abrir tú mismo [module-07.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 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 7 – «Arquitectura de dibujo en producción, rendimiento y proyecto final» – 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-07.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: Completa el proyecto final de VisualOperationsStudio poniendo un único modelo detrás de las cuatro superficies.
Reestructura el proyecto en carpetas Models, Geometry, Renderers y de adaptadores de Wisej, de modo que el indicador dibujado, las celdas de cuadrícula dibujadas por el usuario, la topología en Canvas y la exportación a PNG lean todos el mismo OperationsModel y compartan la capa de geometría pura, sin duplicar estado de negocio en cada renderizador. Añade un ayudante RenderMetrics que registre la duración del renderizado, el tamaño del mapa de bits de salida y el número de objetos de la escena dibujados, y úsalo para comparar el indicador dibujado en el servidor con el renderizador fuera de pantalla, antes y después de una optimización que elijas y justifiques. Ejercita las rutas de fallo –datos vacíos, una superficie de tamaño cero, un rango no válido, una fuente ausente– para que cada una degrade a un marcador de posición legible en lugar de provocar una excepción, añade una tabla accesible que enumere todos los valores que muestran los gráficos y redacta las notas del proyecto final cubriendo la arquitectura, la frontera entre navegador y servidor y la matriz de pruebas.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Capas Models, Geometry, Renderers y de adaptadores con un único OperationsModel que alimenta las cuatro superficies
 - Ayudante RenderMetrics que registra la duración del renderizado, el tamaño del mapa de bits y el número de objetos dibujados por superficie
 - Mediciones antes y después de una optimización, con el razonamiento de por qué la elegiste
 - Rutas de fallo para datos vacíos, tamaño cero, rango no válido y fuente ausente que degradan en lugar de lanzar excepciones
 - CapstoneNotes.md con la arquitectura, la frontera servidor/navegador, la matriz de pruebas y una tabla de datos accesible
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:

- Tu capa de geometría recibe el modelo y un tamaño y devuelve formas, pero un compañero de equipo quiere que devuelva un `GraphicsPath` para que el renderizador del indicador quede más corto. ¿Qué les cuesta eso al renderizador de Canvas y a las pruebas unitarias, y dónde pondrías tú el límite?
- El PNG exportado y el Canvas en pantalla difieren en unos pocos píxeles en el agente de compilación Linux, pero coinciden en Windows. ¿Qué mediciones y qué tipo de prueba te dirían si eso es un defecto o una diferencia de plataforma esperada?
- Quien revisa tu trabajo dice que la tabla accesible es redundante porque todos los indicadores ya tienen un tooltip. Explica qué le da la tabla a quien usa solo el teclado o un lector de pantalla y el tooltip no, y qué implica eso para los chips de severidad que en la cuadrícula solo usan color.

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:

- ✓ Tu capa de geometría recibe el modelo y un tamaño y devuelve formas, pero un compañero de equipo quiere que devuelva un `GraphicsPath` para que el renderizador del indicador quede más corto. ¿Qué les cuesta eso al renderizador de Canvas y a las pruebas unitarias, y dónde pondrías tú el límite?
- ✓ El PNG exportado y el Canvas en pantalla difieren en unos pocos píxeles en el agente de compilación Linux, pero coinciden en Windows. ¿Qué mediciones y qué tipo de prueba te dirían si eso es un defecto o una diferencia de plataforma esperada?
- ✓ Quien revisa tu trabajo dice que la tabla accesible es redundante porque todos los indicadores ya tienen un tooltip. Explica qué le da la tabla a quien usa solo el teclado o un lector de pantalla y el tooltip no, y qué implica eso para los chips de severidad que en la cuadrícula solo usan color.

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.

- ✓ **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-07.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."*
- ✓ **Revisalo como un pull request.** *"Explícame este código de VisualOperationsStudio como si estuvieras revisando el pull request de un compañero y no lo hubieras escrito tú. ¿Dónde vive el estado?, ¿qué ocurre de principio a fin con una acción del usuario?, ¿qué devolverías para que lo cambien? y ¿qué partes no te sentirías cómodo poniendo delante de un cliente? Sé concreto y cita las líneas a las que te refieres — no me resumas el diseño."*
- ✓ **Cambia una pieza sin mover las juntas.** *"Elige la parte de este proyecto VisualOperationsStudio 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."*



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.