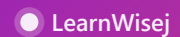


Ejercicio de programación con IA

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 2.

Qué estás pidiendo

Añade a VisualOperationsStudio un control TelemetryGauge reutilizable que dibuje un anillo, zonas de umbral coloreadas, una aguja, un valor numérico y un rótulo. Dale propiedades Value, Minimum, Maximum, WarningThreshold y CriticalThreshold cuyos setters acoten la entrada, salgan de inmediato cuando nada haya cambiado y llamen a Invalidate() una sola vez. Pon los cálculos de coordenadas en un ayudante puro GaugeGeometry que reciba un Rectangle y devuelva los límites del anillo, los ángulos de barrido, los puntos de la aguja y el rectángulo de la etiqueta; después dibuja a partir de esos valores con pens, brushes y un GraphicsPath declarados con using, sin tocar el Graphics que se te entrega. Gestiona un control de tamaño cero, un rango invertido y un valor fuera de rango sin lanzar excepciones, y expón la lectura mediante AccessibleDescription para que no exista solo como píxeles.

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

- ✓ Control TelemetryGauge con propiedades Value, Minimum, Maximum y de umbral que acotan el valor y llaman a Invalidate() una vez por cambio real
- ✓ Ayudante puro GaugeGeometry que devuelve los límites del anillo, los ángulos de barrido, los puntos de la aguja y el rectángulo de la etiqueta
- ✓ Controlador de Paint que libera todos los pens, brushes, fuentes y paths que crea y nunca libera e.Graphics
- ✓ Comprobaciones para tamaño cero, rango invertido y valores fuera de rango, con un AccessibleDescription que lleve la lectura
- ✓ PaintCost.md que anote el tamaño del redibujado, con qué frecuencia se redibuja y cuándo saldría más barato un control con tema

Por qué tiene esta forma

El Módulo 1 puso una lectura de telemetría en cuatro superficies para que vieras dónde se produce cada píxel. VisualOperationsStudio quiere ahora esa lectura en seis sitios, y copiar un controlador de Paint en seis paneles es la forma de acabar con seis indicadores ligeramente distintos, seis pens que nunca se liberan y un redibujado cada vez que cambia un campo que no tiene nada que ver. La solución es dejar de escribir controladores y escribir un control: un único TelemetryGauge cuyas propiedades públicas sean lo único capaz de cambiar su imagen, y cuyo coste puedas expresar en kilobytes.

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
Wisej.Web.PaintEventArgs	El argumento del evento Paint; te entrega un System.Drawing.Graphics que no has creado y un rectángulo dentro del cual dibujar, y tu código se ejecuta en el servidor.
ClipRectangle	La región que Wisej.NET ha reservado para esta operación de dibujo; el diseño puede consultar el tamaño de cliente del control, pero el dibujo sigue acotado por el rectángulo que se te ha dado.
Ayudante GaugeGeometry	Una función pura que va de un Rectangle más un valor normalizado a los límites del anillo, los ángulos de barrido, los puntos de la aguja y un rectángulo de etiqueta, de modo que los cálculos se pueden comprobar sin un control vivo.
Invalidate()	La única llamada que pide a Wisej.NET volver a renderizar la superficie del control; el setter de una propiedad la hace exactamente una vez, y solo cuando el cambio puede alterar un píxel.
Propiedad de los recursos	El controlador libera todos los Pen, Brush, Font, GraphicsPath y Bitmap que crea, y nunca libera el Graphics que ha proporcionado el framework.
Presupuesto de coste de dibujado	Tres números por control dibujado: la CPU de servidor que se gasta dibujando, la memoria de servidor para la imagen temporal y los bytes que el navegador tiene que descargar y decodificar.
AccessibleDescription	Lleva la misma lectura como texto, de modo que un valor que existe como un arco dibujado siga estando disponible para quien nunca ve ese arco.

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 2. Puedes abrir tú mismo [module-02.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 2 – «Dominar el evento Paint de los controles» – 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-02.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: Añade a `VisualOperationsStudio` un control `TelemetryGauge` reutilizable que dibuje un anillo, zonas de umbral coloreadas, una aguja, un valor numérico y un rótulo. Dale propiedades `Value`, `Minimum`, `Maximum`, `WarningThreshold` y `CriticalThreshold` cuyos setters acoten la entrada, salgan de inmediato cuando nada haya cambiado y llamen a `Invalidate()` una sola vez. Pon los cálculos de coordenadas en un ayudante puro `GaugeGeometry` que reciba un `Rectangle` y devuelva los límites del anillo, los ángulos de barrido, los puntos de la aguja y el rectángulo de la etiqueta; después dibuja a partir de esos valores con pens, brushes y un `GraphicsPath` declarados con `using`, sin tocar el `Graphics` que se te entrega. Gestiona un control de tamaño cero, un rango invertido y un valor fuera de rango sin lanzar excepciones, y expón la lectura mediante `AccessibleDescription` para que no exista solo como píxeles.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Control `TelemetryGauge` con propiedades `Value`, `Minimum`, `Maximum` y de umbral que acotan el valor y llaman a `Invalidate()` una vez por cambio real
 - Ayudante puro `GaugeGeometry` que devuelve los límites del anillo, los ángulos de barrido, los puntos de la aguja y el rectángulo de la etiqueta
 - Controlador de Paint que libera todos los pens, brushes, fuentes y paths que crea y nunca libera e `Graphics`
 - Comprobaciones para tamaño cero, rango invertido y valores fuera de rango, con un `AccessibleDescription` que lleve la lectura
 - `PaintCost.md` que anote el tamaño del redibujado, con qué frecuencia se redibuja y cuándo saldría más barato un control con tema
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 traslada los cálculos del anillo y la aguja directamente al controlador de

Paint porque «son solo unas líneas». ¿Qué dos cosas dejan de poder hacerse de forma barata a partir de entonces, y de cuál de ellas depende el Módulo 6?

– Tu indicador está enlazado a un registro de máquina que lleva además el nombre del operario y una fecha de mantenimiento. Si esos campos cambian, ¿debería invalidarse el control? Justifica tu respuesta en términos de dependencias visuales y no en términos del modelo.

– Ese mismo indicador va a ir a un panel que muestra sesenta de ellos y se actualiza cada dos segundos. ¿Qué medirías primero y qué cambiarías si las mediciones fueran malas?

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 traslada los cálculos del anillo y la aguja directamente al controlador de Paint porque «son solo unas líneas». ¿Qué dos cosas dejan de poder hacerse de forma barata a partir de entonces, y de cuál de ellas depende el Módulo 6?
- ✓ Tu indicador está enlazado a un registro de máquina que lleva además el nombre del operario y una fecha de mantenimiento. Si esos campos cambian, ¿debería invalidarse el control? Justifica tu respuesta en términos de dependencias visuales y no en términos del modelo.
- ✓ Ese mismo indicador va a ir a un panel que muestra sesenta de ellos y se actualiza cada dos segundos. ¿Qué medirías primero y qué cambiarías si las mediciones fueran malas?

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-02.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.