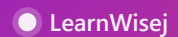


Ejercicio de programación con IA

Construye el ejemplo AdaptiveOps 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 la solución Adaptive Operations Console y su primer shell de escritorio: una Page principal con un panel de barra de herramientas arriba, un panel de navegación a la izquierda, un área de trabajo de relleno, un panel de detalles a la derecha y un panel de estado abajo, contruidos con Dock y Panels anidados. Elige un tema base integrado y documenta la elección, escribe un VisualInventory.md que enumere los controles que necesitan appearances personalizadas, los tokens con nombre de color y fuente, las regiones del diseño y los puntos de corte de escritorio, tableta y teléfono, y añade una etiqueta de estado que muestre el ancho actual del navegador para que los módulos siguientes puedan verificar los perfiles.

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

- ✓ Solución Adaptive Operations Console con un shell de escritorio acoplado formado por los paneles de barra de herramientas, navegación, área de trabajo, detalles y estado
- ✓ Tema base elegido y configurado, con el motivo anotado en las notas del laboratorio
- ✓ VisualInventory.md con los controles que necesitan appearances, los tokens del tema, las regiones del diseño y los puntos de corte
- ✓ Etiqueta de estado que muestra el ancho actual del navegador
- ✓ Captura de pantalla de escritorio y una nota que explique qué capa es responsable de cada decisión visual

Por qué tiene esta forma

Un equipo que trata Wisej.NET como un framework basado ante todo en CSS acaba con BackColor asignado en doscientos controles, selectores que se rompen en la siguiente actualización y manejadores Resize que recalculan Bounds a mano; cuando se pide una versión para teléfono, la única opción que queda es reescribirlo todo. Este módulo aclara de dónde vienen realmente el aspecto y el diseño de un

control, nombra las tres capas de personalización y cuál es responsable de cada decisión, e inicia el shell de Adaptive Operations Console que amplían todos los módulos siguientes.

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
control del servidor y widget del cliente	Escribes controles.NET en el servidor; el navegador muestra widgets JavaScript que Wisej.NET mantiene sincronizados, así que una decisión visual puede estar a cualquiera de los dos lados de esa línea.
tema	Un único archivo JSON de colores, imágenes, fuentes, appearances, estados, estilos, propiedades y ajustes que da a cada control estándar su identidad; el contrato entre ambos mundos.
appearance y AppearanceKey	Cada control expone una clave de appearance; el tema activo contiene una appearance para esa clave, y una clave semántica como action-button selecciona una variante.
estado	Una condición de interacción con nombre dentro de una appearance (default, hovered, pressed, focused, invalid, selected) cuyos estilos y propiedades se aplican mientras está activa.
CssClass y CssStyle	Estilo directo en el navegador: una clase de una hoja de estilos de la aplicación con ámbito, o una cadena de estilo sin procesar en el elemento del widget para un valor puntual poco frecuente.
motor de diseño y Dock	La posición y el tamaño vienen de la jerarquía de contenedores, no del tema; el acoplamiento apila las regiones contra los bordes del padre y rellena lo que queda.
inventario visual	La lista escrita de controles que necesitan appearances, tokens de color y fuente, regiones del diseño y puntos de corte a partir de la cual se planifica el trabajo de tema, diseño y perfiles.

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



Usar Claude

Abre [claude.ai](#) con el mismo prompt ya cargado. Si usas Claude Code, puedes

Codex; cualquier plan que pueda navegar por la web descargará la especificación.

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 AdaptiveOps 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 – «El sistema visual de Wisej.NET» – del curso de LearnWisej «Temas y diseños adaptables», 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/theming-and-responsive-layouts/specs/module-01.md>

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

1. Proyecto «AdaptiveOps», 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 la solución Adaptive Operations Console y su primer shell de escritorio: una Page principal con un panel de barra de herramientas arriba, un panel de navegación a la izquierda, un área de trabajo de relleno, un panel de detalles a la derecha y un panel de estado abajo, contruidos con Dock y Panels anidados. Elige un tema base integrado y documenta la elección, escribe un `VisualInventory.md` que enumere los controles que necesitan appearances personalizadas, los tokens con nombre de color y fuente, las regiones del diseño y los puntos de corte de escritorio, tableta y teléfono, y añade una etiqueta de estado que muestre el ancho actual del navegador para que los módulos siguientes puedan verificar los perfiles.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Solución Adaptive Operations Console con un shell de escritorio acoplado formado por los paneles de barra de herramientas, navegación, área de trabajo, detalles y estado
 - Tema base elegido y configurado, con el motivo anotado en las notas del laboratorio
 - `VisualInventory.md` con los controles que necesitan appearances, los tokens del tema, las regiones del diseño y los puntos de corte
 - Etiqueta de estado que muestra el ancho actual del navegador
 - Captura de pantalla de escritorio y una nota que explique qué capa es responsable de cada decisión visual
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:

- El comando principal de la barra de herramientas necesita el color de marca en todas las pantallas. ¿Cuál de las tres capas es responsable de esa decisión y qué fallaría si asignaras `BackColor` al botón?
- Tu etiqueta de ancho indica 1024 px y el panel de detalles sigue visible. ¿Qué mecanismo de un módulo posterior decidirá si debe estarlo, y por qué una media query CSS es la primera

respuesta equivocada en Wisej.NET?

– Si mañana se cambiara el tema base por otro tema integrado, ¿qué filas de tu VisualInventory.md cambiarían y cuáles no, y qué te dice eso sobre dónde residen las decisiones del inventario?

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:

- ✓ El comando principal de la barra de herramientas necesita el color de marca en todas las pantallas. ¿Cuál de las tres capas es responsable de esa decisión y qué fallaría si asignaras BackColor al botón?
- ✓ Tu etiqueta de ancho indica 1024 px y el panel de detalles sigue visible. ¿Qué mecanismo de un módulo posterior decidirá si debe estarlo, y por qué una media query CSS es la primera respuesta equivocada en Wisej.NET?
- ✓ Si mañana se cambiara el tema base por otro tema integrado, ¿qué filas de tu VisualInventory.md cambiarían y cuáles no, y qué te dice eso sobre dónde residen las decisiones del inventario?

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 `Theming and Responsive Layouts 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.

- ✓ **Añade el segundo caso.** *"Amplía este proyecto AdaptiveOps 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/theming-and-responsive-layouts/specs/module-01.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."*
- ✓ **Revísalo como un pull request.** *"Explícame este código de AdaptiveOps 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 resumes el diseño."*



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.