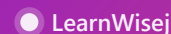


Ejercicio de programación con IA

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.



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 Adaptive Operations Console: un tema personalizado con tokens con nombre y al menos cuatro appearances semánticas, uso de CssClass con ámbito, un shell de escritorio con regiones Flow, Table y Flex, un ClientProfiles.json con comportamiento para teléfono, tableta, escritorio pequeño y escritorio, propiedades responsive junto con un manejador ResponsiveProfileChanged idempotente, y un indicador de perfil visible en modo de prueba. Realiza la revisión de accesibilidad de marcos de foco, estados no válidos, contraste, objetivos táctiles y orden de tabulación, completa la matriz de QA responsive con capturas para cada perfil, y redacta la nota de arquitectura y el paquete de contexto que expliquen cada decisión de tema, diseño y perfil.

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

- ✓ Adaptive Operations Console terminada con tema personalizado, appearances semánticas, CSS con ámbito y regiones Flow, Table y Flex
- ✓ ClientProfiles.json, propiedades responsive y un manejador ResponsiveProfileChanged idempotente con un indicador de perfil en modo de prueba
- ✓ Revisión de accesibilidad que cubre marcos de foco, estados no válidos, contraste, objetivos táctiles y orden de tabulación
- ✓ Matriz de QA responsive con capturas de escritorio, tableta y teléfono en ambas orientaciones
- ✓ Nota de arquitectura y paquete de contexto para IA que expliquen cada decisión de tema, diseño y perfil

Por qué tiene esta forma

Seis módulos de trabajo conviven ahora en una sola solución: un tema derivado de un tema base, CSS con ámbito, un shell de contenedores anidados, regiones Flow, Table y Flex, un ClientProfiles.json y un manejador de perfil. Lo que convierte eso en un producto no es otra funcionalidad. Es la prueba de que la

consola se puede usar en todos los perfiles, de que un usuario de teclado o táctil puede manejarla, de que un redimensionado no desencadena reconstrucciones costosas, y de que el siguiente desarrollador, o la siguiente actualización de Wisej.NET, no puede romper sin avisar el sistema visual. Este módulo es esa prueba, junto con el paquete de contexto que permite a un asistente de IA responder preguntas sobre las decisiones que tomaste.

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
conjunto de pantallas del proyecto final	Adaptive Operations Console terminada: tema personalizado, appearances semánticas, CSS con ámbito, un shell de escritorio, un shell adaptable para tableta, un shell para teléfono, una cuadrícula, un editor, mensajes de validación, navegación y un indicador de perfil visible en modo de prueba.
revisión de accesibilidad visual	Una revisión de los marcos de foco, los estados deshabilitados y no válidos, el contraste, el espaciado táctil, los tooltips de los botones solo icono y el orden de tabulación con teclado, hecha manejando la pantalla y no leyendo el tema.
manejador de perfil idempotente	Un manejador <code>ResponsiveProfileChanged</code> que asigna valores finales para el perfil actual, de modo que ejecutarlo dos veces, o después de una recarga, produce exactamente la misma interfaz.
estabilidad del diseño	<code>MinimumSize</code> y <code>MaximumSize</code> evitan que las regiones se colapsen, y <code>AutoSize</code> se mantiene fuera de las jerarquías grandes de contenedores, de modo que los redimensionados son predecibles y ligeros.
gobernanza del tema	El JSON del tema, la hoja de estilos con ámbito y <code>ClientProfiles.json</code> están bajo control de código fuente, se revisan como diffs, se documentan como tokens y nunca se editan dentro de un tema integrado.
personalización segura ante actualizaciones	Las appearances documentadas, los flujos del Theme Builder y las propiedades del servidor sobreviven a las actualizaciones; los selectores CSS que dependen del DOM interno de un widget pueden no hacerlo, así que se limitan en ámbito, se documentan o se evitan.
matriz de QA responsive y paquete de contexto	La matriz registra anchos, orientaciones, perfil esperado y comprobaciones por escenario; el paquete de contexto registra el árbol de decisión, las reglas de respuesta y el mapa de fuentes a partir de los que responde un asistente.

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 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 7 – «Proyecto final listo para móvil: accesibilidad, rendimiento y gobernanza en producción» – 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-07.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: Completa el proyecto final de Adaptive Operations Console: un tema personalizado con tokens con nombre y al menos cuatro appearances semánticas, uso de `CssClass` con ámbito, un shell de escritorio con regiones Flow, Table y Flex, un `ClientProfiles.json` con comportamiento para teléfono, tableta, escritorio pequeño y escritorio, propiedades responsive junto con un manejador `ResponsiveProfileChanged` idempotente, y un indicador de perfil visible en modo de prueba. Realiza la revisión de accesibilidad de marcos de foco, estados no válidos, contraste, objetivos táctiles y orden de tabulación, completa la matriz de QA responsive con capturas para cada perfil, y redacta la nota de arquitectura y el paquete de contexto que expliquen cada decisión de tema, diseño y perfil.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Adaptive Operations Console terminada con tema personalizado, appearances semánticas, CSS con ámbito y regiones Flow, Table y Flex
 - `ClientProfiles.json`, propiedades responsive y un manejador `ResponsiveProfileChanged` idempotente con un indicador de perfil en modo de prueba
 - Revisión de accesibilidad que cubre marcos de foco, estados no válidos, contraste, objetivos táctiles y orden de tabulación
 - Matriz de QA responsive con capturas de escritorio, tableta y teléfono en ambas orientaciones
 - Nota de arquitectura y paquete de contexto para IA que explican cada decisión de tema, diseño y perfil
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 manejador `ResponsiveProfileChanged` se ejecuta dos veces para el perfil Tablet después de recargar el navegador. ¿Qué línea tuya garantiza que la segunda ejecución no cambie nada,

y qué vería el usuario si faltara esa línea?

– Un revisor te pide quitar el marco de foco de `action-button` porque «parece anticuado».

¿Qué cambias en su lugar, qué token lo controla y qué otros dos estados vuelves a verificar después en el Theme Builder?

– El encabezado de la cuadrícula de tickets necesita un ajuste al que solo llega un selector sobre el DOM interno del widget. ¿Qué pruebas primero, qué debe cumplirse antes de permitir el selector, y qué dejas por escrito para que la siguiente actualización no sorprenda al equipo?

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 manejador `ResponsiveProfileChanged` se ejecuta dos veces para el perfil Tablet después de recargar el navegador. ¿Qué línea tuya garantiza que la segunda ejecución no cambie nada, y qué vería el usuario si faltara esa línea?
- ✓ Un revisor te pide quitar el marco de foco de `action-button` porque «parece anticuado». ¿Qué cambias en su lugar, qué token lo controla y qué otros dos estados vuelves a verificar después en el Theme Builder?
- ✓ El encabezado de la cuadrícula de tickets necesita un ajuste al que solo llega un selector sobre el DOM interno del widget. ¿Qué pruebas primero, qué debe cumplirse antes de permitir el selector, y qué dejas por escrito para que la siguiente actualización no sorprenda al equipo?

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 7 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 7 en GitHub](#)

O clona el conjunto completo con `git clone https://github.com/iceteagroup/LearnWisej-Samples` y abre `Theming and Responsive Layouts Course/Module 7`.



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.

- ✓ **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-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 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 resumas el diseño."*
- ✓ **Cambia una pieza sin mover las juntas.** *"Elige la parte de este proyecto AdaptiveOps 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.