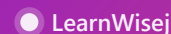


Ejercicio de programación con IA

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.



Escribir el código ya es la mitad fácil.

Conseguir que un asistente moderno genere código que funcione para «Fundamentos del diseño: Dock, Anchor, AutoSize, AutoScroll y composición del shell» solo requiere un buen prompt. Juzgarlo, corregirlo y ampliarlo requiere todo lo que ha cubierto este módulo. Ese es el reparto sobre el que se construye este ejercicio: tú aportas la especificación y la revisión; el asistente, el tecleo.

Qué estás pidiendo

Refactoriza el shell de Adaptive Operations Console en contenedores anidados: una barra de herramientas superior, un carril de navegación a la izquierda, un área de trabajo de relleno, un panel de detalles a la derecha y una región de estado, cada uno un Panel o UserControl con su propio diseño local. Elimina todos los manejadores Resize que asignan Bounds, corrige el orden de acoplamiento en el diseñador, usa Padding para los espacios entre regiones acopladas, asigna un MinimumSize al área de trabajo y a la cuadrícula, y haz que el editor de detalles se desplace con AutoScroll y barras de desplazamiento ocultas en anchos estrechos.

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

- ✓ Shell reconstruido con contenedores anidados y regiones de barra de herramientas, navegación, área de trabajo, detalles y estado
- ✓ Eliminado todo el código Resize manual que asignaba Bounds
- ✓ Orden de acoplamiento, Padding y MinimumSize definidos de forma intencionada en cada región
- ✓ Editor de detalles que se desplaza mediante AutoScroll con barras de desplazamiento ocultas en anchos estrechos
- ✓ Regiones de navegación y detalles extraídas a UserControls

Por qué tiene esta forma

Adaptive Operations Console ya tiene un tema y CSS con ámbito, pero su shell sigue moviendo controles a mano: un manejador Resize recalcula Bounds para la barra de herramientas, el carril de navegación, la cuadrícula de tickets y el editor de detalles cada vez que el navegador cambia de tamaño. Funciona con el ancho que probó el desarrollador, se rompe con cualquier otro y no puede sobrevivir a los perfiles de tableta y teléfono que introduce el Módulo 6. El diseño adaptable en Wisej.NET depende de contenedores

y reglas de diseño, no de coordenadas: regiones que ocupan bordes, un área de trabajo que rellena lo que queda y editores que se desplazan en lugar de superponerse.

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
motor de diseño predeterminado	Cada control de Wisej.NET tiene un motor de diseño; el predeterminado admite posición absoluta, acoplamiento y anclaje, de modo que una pantalla puede combinar la colocación precisa del diseñador con regiones que se redimensionan solas.
prioridad de acoplamiento	Los hijos acoplados ocupan espacio en el orden en que aparecen en la colección de controles del padre, y por eso los mismos cinco paneles producen un shell correcto o uno con superposiciones según el orden de los hijos.
DisplayRectangle y Padding	Los controles acoplados se colocan dentro del DisplayRectangle del padre, que el Padding reduce; el motor predeterminado ignora Margin en los hijos acoplados, así que los espacios vienen del padding o de controles separadores.
anclaje	Un anclaje mantiene la distancia de un control a los lados seleccionados del padre, lo que encaja con regiones de tipo formulario: etiquetas arriba a la izquierda, cuadros de texto de izquierda a derecha, botones de comando abajo a la derecha.
AutoSize y diseño circular	Un contenedor que calcula su tamaño a partir de hijos acoplados o anclados a él no tiene una respuesta estable; ese bucle es la principal advertencia del módulo sobre AutoSize.
AutoScroll con barras ocultas	AutoScroll permite que un contenedor se desplace cuando el contenido no cabe, y ScrollBars.Hidden mantiene el desplazamiento táctil en los teléfonos sin dibujar barras de desplazamiento.
región UserControl	Una región del shell con su propio diseño local y una superficie pública reducida, de modo que el shell muestra, oculta, acopla o mueve la región completa según el perfil en lugar de conocer sus coordenadas internas.

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 4. Puedes abrir tú mismo [module-04.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 4 – «Fundamentos del diseño: Dock, Anchor, AutoSize, AutoScroll y composición del shell» – 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-04.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: Refactoriza el shell de Adaptive Operations Console en contenedores anidados: una barra de herramientas superior, un carril de navegación a la izquierda, un área de trabajo de relleno, un panel de detalles a la derecha y una región de estado, cada uno un Panel o UserControl con su propio diseño local. Elimina todos los manejadores Resize que asignan Bounds, corrige el orden de acoplamiento en el diseñador, usa Padding para los espacios entre regiones acopladas, asigna un MinimumSize al área de trabajo y a la cuadrícula, y haz que el editor de detalles se desplace con AutoScroll y barras de desplazamiento ocultas en anchos estrechos.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Shell reconstruido con contenedores anidados y regiones de barra de herramientas, navegación, área de trabajo, detalles y estado
 - Eliminado todo el código Resize manual que asignaba Bounds
 - Orden de acoplamiento, Padding y MinimumSize definidos de forma intencionada en cada región
 - Editor de detalles que se desplaza mediante AutoScroll con barras de desplazamiento ocultas en anchos estrechos
 - Regiones de navegación y detalles extraídas a UserControls
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:

- Si en el orden de los hijos el panel del área de trabajo se acoplara antes que el panel de estado, ¿qué vería el usuario en la parte inferior de la cuadrícula, y qué comando del diseñador lo corrige sin cambiar ningún tamaño?
- Has asignado un Margin de 12 píxeles al carril de navegación y el espacio no ha aparecido. ¿Por qué no, y qué dos mecanismos sí producen un espacio entre regiones acopladas?
- El editor de detalles es un UserControl con AutoScroll. ¿Qué miembros suyos debe conocer

el shell para moverlo bajo la cuadrícula en una tableta, y cuáles no debe necesitar nunca?

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:

- ✓ Si en el orden de los hijos el panel del área de trabajo se acoplara antes que el panel de estado, ¿qué vería el usuario en la parte inferior de la cuadrícula, y qué comando del diseñador lo corrige sin cambiar ningún tamaño?
- ✓ Has asignado un Margen de 12 píxeles al carril de navegación y el espacio no ha aparecido. ¿Por qué no, y qué dos mecanismos sí producen un espacio entre regiones acopladas?
- ✓ El editor de detalles es un UserControl con AutoScroll. ¿Qué miembros suyos debe conocer el shell para moverlo bajo la cuadrícula en una tableta, y cuáles no debe necesitar nunca?

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 4 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 4 en GitHub](#)

O clona el conjunto completo con `git clone https://github.com/iceteagroup/LearnWisej-Samples` y abre `Theming and Responsive Layouts Course/Module 4`.



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.

- ✓ **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."*
- ✓ **Haz que aguante un mal día.** *"Revisa este proyecto AdaptiveOps 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 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."*



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.