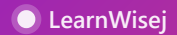


Ejercicio de programación con IA

Construye el ejemplo FieldOps 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.



Escribir el código ya es la mitad fácil.

Conseguir que un asistente moderno genere código que funcione para «Shells móviles con .NET MAUI» 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

Construye el shell móvil de FieldOps con .NET MAUI: una página híbrida que aloje la aplicación en Android e iOS, una sesión que sobreviva al paso a segundo plano y se reanude al volver, el botón Atrás de Android asignado a la navegación dentro de la aplicación, márgenes de área segura para dispositivos con notch, y una compilación de depuración ejecutándose en un emulador por plataforma con capturas de pantalla de la lista de órdenes de trabajo.

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

- ✓ Proyecto de shell híbrido de .NET MAUI para Android e iOS
- ✓ Gestión del ciclo de vida para el segundo plano y la reanudación
- ✓ Asignación del botón Atrás y del área segura
- ✓ Compilaciones en emulador para ambas plataformas
- ✓ Capturas de pantalla de ambos emuladores

Por qué tiene esta forma

Los técnicos de FieldOps trabajan con un teléfono en una sala de máquinas, un sótano o una furgoneta, y una pestaña del navegador es un mal contenedor para eso: no hay icono en la pantalla de inicio que repartir, ni canal de tienda o empresarial, ni acceso a la cámara y al GPS que necesitará el Módulo 5. Además, el sistema operativo móvil es un anfitrión mucho más agresivo: suspende la aplicación en cuanto entra una llamada, la vuelve a dibujar bajo un notch y le da al técnico un gesto Atrás del sistema que FieldOps nunca pidió.

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
shell híbrido de.NET MAUI	Una pequeña aplicación nativa para Android e iOS cuya única página aloja tu aplicación Wisej.NET en una vista web, de modo que lo que ve el técnico son las pantallas C# que ya publicas.
vista web nativa anfitriona	El motor de navegador de la plataforma integrado en esa página (Android System WebView, WKWebView en iOS), que ejecuta el mismo cliente que recibe un navegador de escritorio; por eso ninguna pantalla se reescribe para móvil.
puente híbrido	El canal bidireccional entre la página alojada y el código nativo, para que el código C# del servidor pueda pedir algo al dispositivo y recibir la respuesta como un evento en lugar de como una navegación de página.
ciclo de vida de la aplicación MAUI	Los eventos que el sistema operativo genera alrededor de la aplicación (ventana creada, activada, desactivada, detenida y reanudada, más OnSleep y OnResume), ninguno de los cuales genera una pestaña del navegador.
margen de supervivencia de la sesión	El tiempo en segundo plano durante el cual la sesión de Wisej.NET sigue viva en el servidor, de modo que una reanudación puede reconectar con el estado existente en lugar de volver a iniciar la aplicación.
márgenes del área segura	El relleno específico de cada dispositivo que mantiene el contenido fuera del notch, de la barra de estado y de la franja del indicador de inicio, indicado por la plataforma y no adivinado a partir del tamaño de la pantalla.
asignación de la navegación Atrás	La regla que decide qué hace el gesto Atrás de Android: retroceder dentro de la navegación propia de la aplicación o confirmar la salida, pero nunca destruir en silencio una orden de trabajo a medio rellenar.

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 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 FieldOps 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 - «Shells móviles con .NET MAUI» - del curso de LearnWisej «Empaquetado híbrido y nativo», 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/hybrid-native-packaging/specs/module-04.md>

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

1. Proyecto «FieldOps», 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: Construye el shell móvil de FieldOps con .NET MAUI: una página híbrida que aloje la aplicación en Android e iOS, una sesión que sobreviva al paso a segundo plano y se reanude al volver, el botón Atrás de Android asignado a la navegación dentro de la aplicación, márgenes de área segura para dispositivos con notch, y una compilación de depuración ejecutándose en un emulador por plataforma con capturas de pantalla de la lista de órdenes de trabajo.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Proyecto de shell híbrido de .NET MAUI para Android e iOS
 - Gestión del ciclo de vida para el segundo plano y la reanudación
 - Asignación del botón Atrás y del área segura
 - Compilaciones en emulador para ambas plataformas
 - Capturas de pantalla de ambos emuladores
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 técnico está a mitad de una actualización de estado cuando recibe una llamada de tres minutos. ¿Qué debe cumplirse en el servidor y en el shell para que la actualización a medias siga ahí después, y qué cambia si la llamada dura una hora?
- El shell no sabe qué pantalla de FieldOps está abierta. ¿Cómo decide un gesto Atrás entre volver desde la página de detalle, cerrar un cuadro de diálogo y salir de la aplicación, y dónde está realmente ese estado?
- ¿Cuáles de los cambios de este módulo seguiría notando un técnico si abriera FieldOps en el navegador del teléfono en lugar de en el shell, y cuáles desaparecerían? ¿Qué te dice eso sobre dónde poner el relleno del área segura?

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 técnico está a mitad de una actualización de estado cuando recibe una llamada de tres minutos. ¿Qué debe cumplirse en el servidor y en el shell para que la actualización a medias siga ahí después, y qué cambia si la llamada dura una hora?
- ✓ El shell no sabe qué pantalla de FieldOps está abierta. ¿Cómo decide un gesto Atrás entre volver desde la página de detalle, cerrar un cuadro de diálogo y salir de la aplicación, y dónde está realmente ese estado?
- ✓ ¿Cuáles de los cambios de este módulo seguiría notando un técnico si abriera FieldOps en el navegador del teléfono en lugar de en el shell, y cuáles desaparecerían? ¿Qué te dice eso sobre dónde poner el relleno del área segura?

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/hybrid-native-packaging/specs/module-04.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 FieldOps 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 FieldOps 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.