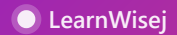


Ejercicio de programación con IA

Construye el ejemplo GlobalDesk 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 «Cultura del navegador, cultura de sesión y cambio de idioma en tiempo de ejecución» 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

Dale a GlobalDesk un cambio de idioma que termine el trabajo. Pon culture en auto en Default.json para que la sesión arranque con el idioma del navegador, añade un ComboBox de idioma que ofrezca inglés, alemán y una cultura de idioma y región, y establece Application.CurrentCulture con CultureInfo.GetCultInfo en su manejador SelectedIndexChanged sin guardar nunca la elección en un campo estático. Gestiona Application.CultureChanged para llamar a un método ApplyTextResources que actualice todas las etiquetas respaldadas por recursos y a un método UpdateCulturePreview que vuelva a formatear los ejemplos de fecha, número y moneda, y recrea en ese mismo manejador el editor de clientes localizado en el diseñador. Verifica la misma página con el parámetro de URL lang y después escribe una nota de laboratorio que describa qué le haría un campo de cultura estático a un segundo usuario conectado al mismo tiempo.

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

- ✓ Default.json con culture en auto, más una nota sobre cuándo una cultura fija es la opción correcta
- ✓ Manejador del ComboBox de idioma que establece Application.CurrentCulture con CultureInfo.GetCultInfo
- ✓ Manejador de Application.CultureChanged que llama a ApplyTextResources y UpdateCulturePreview
- ✓ Editor localizado en el diseñador recreado tras el cambio para que ninguna etiqueta se quede en el idioma anterior
- ✓ Verificación con el parámetro de URL lang y una nota sobre por qué la cultura no debe vivir en un campo estático

Por qué tiene esta forma

Tras el Módulo 3, GlobalDesk tiene un archivo de recursos neutro, uno alemán a su lado, un helper Texts, un editor de clientes localizado en el diseñador e incluso una etiqueta de sistema de Wisej propia. Lo que todavía no tiene es respuesta para la pregunta que un usuario real hace primero: ¿cómo consigo esto en mi idioma y qué le pasa a la pantalla cuando lo pido? Todo lo anterior se decidió mientras se construía la aplicación; este módulo trata de los dos momentos en que la decisión se toma de verdad: cuando arranca una sesión y cuando un usuario cambia de opinión a mitad de camino.

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
detección automática del navegador	Wisej.NET puede leer el idioma que informa el navegador y arrancar la sesión con él, así que cada usuario obtiene una cultura sensata sin que un único ajuste del servidor decida por todos.
culture en Default.json	El ajuste de la aplicación que decide de dónde procede la cultura de una sesión nueva; su valor predeterminado es auto, y un nombre fijo como de-DE fija en cambio todas las sesiones a una única configuración regional.
rightToLeft en Default.json	Su ajuste vecino, cuyo valor auto sigue a la cultura con la que acabe la sesión; el Módulo 5 se apoya en él, pero lo conoces aquí en el mismo archivo.
Application.CurrentCulture	Obtiene o establece la cultura de la sesión actual de Wisej.NET, que es justo el ámbito que necesita una aplicación web multiusuario: la elección de un usuario nunca debe repintar la pantalla de otro.
parámetro de URL lang	Una forma admitida de pedir un idioma al entrar, por ejemplo?lang=de, que hace triviales los enlaces de prueba y las URL de entrada específicas de un idioma.
Application.CultureChanged	Se genera cuando la cultura actual cambia respecto a aquella con la que arrancó la sesión; es el único sitio donde volver a aplicar el texto respaldado por recursos y reformatear lo que ya está en pantalla.
controles localizados en el diseñador	Sus valores localizados se aplican cuando se crean los controles, así que un cambio de cultura posterior no los reescribe: recreas el contenedor o vuelves a aplicar los valores desde el código.

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 GlobalDesk 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 – «Cultura del navegador, cultura de sesión y cambio de idioma en tiempo de ejecución» – del curso de LearnWisej «Localización 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/localization-in-wisej-net/specs/module-04.md>

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

1. Proyecto «GlobalDesk», 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: Dale a GlobalDesk un cambio de idioma que termine el trabajo. Pon culture en auto en Default.json para que la sesión arranque con el idioma del navegador, añade un ComboBox de idioma que ofrezca inglés, alemán y una cultura de idioma y región, y establece `Application.CurrentCulture` con `CultureInfo.GetCultureInfo` en su manejador `SelectedIndexChanged` sin guardar nunca la elección en un campo estático. Gestiona `Application.CultureChanged` para llamar a un método `ApplyTextResources` que actualice todas las etiquetas respaldadas por recursos y a un método `UpdateCulturePreview` que vuelva a formatear los ejemplos de fecha, número y moneda, y recrea en ese mismo manejador el editor de clientes localizado en el diseñador. Verifica la misma página con el parámetro de URL `lang` y después escribe una nota de laboratorio que describa qué le haría un campo de cultura estático a un segundo usuario conectado al mismo tiempo.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Default.json con culture en auto, más una nota sobre cuándo una cultura fija es la opción correcta
 - Manejador del ComboBox de idioma que establece `Application.CurrentCulture` con `CultureInfo.GetCultureInfo`
 - Manejador de `Application.CultureChanged` que llama a `ApplyTextResources` y `UpdateCulturePreview`
 - Editor localizado en el diseñador recreado tras el cambio para que ninguna etiqueta se quede en el idioma anterior
 - Verificación con el parámetro de URL `lang` y una nota sobre por qué la cultura no debe vivir en un campo estático
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 método `ApplyTextResources` actualiza el dashboard a la perfección, pero el editor de

clientes conserva sus etiquetas en inglés tras cambiar al alemán. Explica qué tienen de distinto esas etiquetas y nombra dos estrategias que lo resuelvan.

– Un compañero propone guardar en caché el CultureInfo elegido en un campo estático "para no buscarlo dos veces". Describe qué vería en realidad un segundo usuario conectado desde otro país en ese mismo momento.

– La vista previa de la cultura muestra un importe monetario. Tras cambiar de en-US a de-DE, ¿qué ha cambiado en memoria y qué ha cambiado solo en pantalla, y cuál de las dos cosas escribirías alguna vez en la base de datos?

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 método ApplyTextResources actualiza el dashboard a la perfección, pero el editor de clientes conserva sus etiquetas en inglés tras cambiar al alemán. Explica qué tienen de distinto esas etiquetas y nombra dos estrategias que lo resuelvan.
- ✓ Un compañero propone guardar en caché el CultureInfo elegido en un campo estático "para no buscarlo dos veces". Describe qué vería en realidad un segundo usuario conectado desde otro país en ese mismo momento.
- ✓ La vista previa de la cultura muestra un importe monetario. Tras cambiar de en-US a de-DE, ¿qué ha cambiado en memoria y qué ha cambiado solo en pantalla, y cuál de las dos cosas escribirías alguna vez en la base de datos?

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.

- ✓ **Cambia una pieza sin mover las juntas.** *"Elige la parte de este proyecto GlobalDesk 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 GlobalDesk 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 GlobalDesk 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.