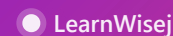


Ejercicio de programación con IA

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.



Instrúyelo, léelo y luego rómpelo.

Este ejercicio tiene tres partes, y solo la primera trata del prompt. Le das al asistente la especificación del módulo y recibes una solución GlobalDesk; lees lo que escribió y decides si de verdad hace lo que pedía el módulo; y después lo llevas más allá de lo que pedía el módulo. En la segunda y la tercera parte es donde se aprende.

Qué estás pidiendo

Amplía los recursos compartidos de GlobalDesk. Añade Strings.de.resx junto a Resources/Strings.resx con exactamente las mismas claves y los valores traducidos, añade una entrada Customer.LastOrder que contenga dos marcadores de posición, y compón la frase con string.Format y Application.CurrentCulture en lugar de concatenar fragmentos. Haz pasar por el helper Texts todos los mensajes que muestra el editor de clientes, incluidas la confirmación de guardado y los dos mensajes de validación, y haz que el servicio de guardado devuelva una clave de resultado estable como CustomerEditor.Saved en lugar de una frase en inglés que la interfaz imprime directamente. Después añade un archivo Resources-de.resx que sobrescriba una etiqueta de sistema de Wisej, demuéstalo en un MessageBox y escribe una nota breve sobre la diferencia entre el texto de tu producto y el texto de sistema de Wisej.

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

- ✓ Strings.de.resx con las mismas claves que el archivo neutro y solo los valores traducidos
- ✓ Recurso Customer.LastOrder compuesto con string.Format y la cultura activa, nunca concatenado
- ✓ Servicio de guardado que devuelve una clave de recurso o un código de resultado en lugar de una frase en inglés
- ✓ Resources-de.resx que sobrescribe al menos una etiqueta de sistema de Wisej, mostrada en un MessageBox
- ✓ Nota de laboratorio que separa el texto del producto, el texto de sistema de Wisej y el texto de diagnóstico

Por qué tiene esta forma

El Módulo 2 dejó el editor de clientes de GlobalDesk con un aspecto convincente en alemán: todas las etiquetas que posee el diseñador cambian con la cultura y el diseño aguanta. Entonces el usuario pulsa Save y lee Customer saved. en inglés, porque esa frase no es una propiedad de ningún control: es un literal

de cadena dentro de un manejador de clic, al lado de This field is required. y de la línea de estado que cuenta los contactos cargados.

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
archivo de recursos compartido de la aplicación	Resources/Strings.resx contiene el texto visible para el usuario que no pertenece a ninguna pantalla diseñada concreta: confirmaciones de guardado, mensajes de validación, líneas de estado, títulos de informes y resultados de servicios.
archivo compañero por cultura	Strings.de.resx y Strings.it.resx se colocan junto al archivo neutro y llevan las mismas claves; un traductor cambia valores, nunca nombres de claves.
clave semántica estable	Un nombre como CustomerEditor.Saved describe el significado del texto, así que reescribir la frase en inglés nunca obliga a cambiar código ni rompe las traducciones.
Wisej.Resources.ResourceManager	El objeto de búsqueda que sigue el modelo de recursos de .NET: dados un nombre base y un ensamblado, encuentra el valor para la cultura activa y se repliega hacia el archivo neutro.
marcador de clave ausente	Un helper que devuelve [Key] cuando una búsqueda falla convierte un defecto invisible en algo que puedes ver y buscar, en lugar de en una etiqueta en blanco.
cadena de formato con marcadores de posición	La frase entera con sus {0} y {1} vive en el recurso y string.Format la rellena, porque cada idioma ordena el nombre, la fecha y el verbo de forma distinta.
Resources-[LANG].resx	Un archivo de recursos de sistema de Wisej.NET aparte, en la raíz del proyecto o en una carpeta /Resources, que sobrescribe o añade etiquetas del framework como \$Ok, \$Cancel y \$Offline.

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 3. Puedes abrir tú mismo [module-03.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 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 3 – «Recursos de la aplicación, ResourceManager y el texto de sistema de Wisej» – 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-03.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: Amplía los recursos compartidos de GlobalDesk. Añade `Strings.de.resx` junto a `Resources/Strings.resx` con exactamente las mismas claves y los valores traducidos, añade una entrada `Customer.LastOrder` que contenga dos marcadores de posición, y compón la frase con `string.Format` y `Application.CurrentCulture` en lugar de concatenar fragmentos. Haz pasar por el helper `Texts` todos los mensajes que muestra el editor de clientes, incluidas la confirmación de guardado y los dos mensajes de validación, y haz que el servicio de guardado devuelva una clave de resultado estable como `CustomerEditor.Saved` en lugar de una frase en inglés que la interfaz imprime directamente. Después añade un archivo `Resources-de.resx` que sobrescriba una etiqueta de sistema de Wisej, demuéstalo en un `MessageBox` y escribe una nota breve sobre la diferencia entre el texto de tu producto y el texto de sistema de Wisej.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - `Strings.de.resx` con las mismas claves que el archivo neutro y solo los valores traducidos
 - Recurso `Customer.LastOrder` compuesto con `string.Format` y la cultura activa, nunca concatenado
 - Servicio de guardado que devuelve una clave de recurso o un código de resultado en lugar de una frase en inglés
 - `Resources-de.resx` que sobrescribe al menos una etiqueta de sistema de Wisej, mostrada en un `MessageBox`
 - Nota de laboratorio que separa el texto del producto, el texto de sistema de Wisej y el texto de diagnóstico
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:

- Borra una clave de `Strings.de.resx` y ejecuta en alemán. Describe qué ve el usuario, por

qué no se rompe nada y cómo detectarías esto en una aplicación real con doscientas claves.

- Tu servicio de guardado devuelve ahora `CustomerEditor.Saved`. ¿Qué se rompería en las pruebas unitarias, y en la pantalla en alemán, si devolviera la frase en sí?
- Un compañero añade la palabra "Cancel" a `Resources-de.resx` bajo una clave inventada por él. Explica por qué el botón del `MessageBox` no cambia y dónde debería haber ido realmente esa cadena.

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:

- ✓ Borra una clave de `Strings.de.resx` y ejecuta en alemán. Describe qué ve el usuario, por qué no se rompe nada y cómo detectarías esto en una aplicación real con doscientas claves.
- ✓ Tu servicio de guardado devuelve ahora `CustomerEditor.Saved`. ¿Qué se rompería en las pruebas unitarias, y en la pantalla en alemán, si devolviera la frase en sí?
- ✓ Un compañero añade la palabra "Cancel" a `Resources-de.resx` bajo una clave inventada por él. Explica por qué el botón del `MessageBox` no cambia y dónde debería haber ido realmente esa cadena.

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.

- ✓ **Revisalo como un pull request.** *"Explícame este código de `GlobalDesk` 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 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."*



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.