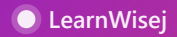


Ejercicio de programación con IA

Construye el ejemplo IconDesk 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.



No vas a teclear este ejemplo. Vas a dirigirlo.

Ya casi nadie escribe código Wisej.NET partiendo de un archivo vacío — un asistente escribe el primer borrador y el desarrollador decide si es correcto. Eso solo funciona si dominas el tema, y eso es lo que te dio el resto de este módulo. Aquí le pasas ese conocimiento a ChatGPT o a Claude, recibes el ejemplo de este módulo como una solución de Visual Studio que funciona y lo revisas como revisarías el pull request de un compañero.

Qué estás pidiendo

Crea IconDesk.Icons como una biblioteca de clases junto a la aplicación, pon al menos doce iconos SVG en un directorio Resources y márcalos como Embedded Resource. Añade un catálogo estático Applcons que exponga cada icono como una cadena resource.wx con nombre para que quien lo use nunca escriba una ruta, referencia la biblioteca desde IconDesk y después asigna algunos iconos mediante el diseñador y otros desde el catálogo en C#. Verifica el sufijo de color en tres iconos y anota la limpieza del SVG que alguno de ellos necesitó. Termina con las reglas de nomenclatura y versionado que deciden cómo entra un icono nuevo en el pack.

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

- ✓ Biblioteca de clases IconDesk.Icons con al menos doce archivos SVG en un directorio Resources, todos como Embedded Resource
- ✓ Catálogo estático Applcons que expone cada icono como una cadena resource.wx con nombre
- ✓ Página de IconDesk que asigna algunos iconos mediante el diseñador y otros desde el catálogo Applcons
- ✓ Tres iconos verificados con el sufijo de color más una nota sobre la limpieza del SVG que necesitaron
- ✓ Reglas de nomenclatura y versionado que describen cómo entra un icono nuevo en el pack y cómo una aplicación sobrescribe uno

Por qué tiene esta forma

Para el quinto módulo IconDesk muestra un logotipo incrustado, una imagen de tema e iconos de dos packs oficiales, y ahí es donde empieza el desorden: aparecen cadenas de recurso en línea dentro de los manejadores de eventos, dos pantallas escriben de forma distinta el mismo archivo y nadie sabe decir qué iconos son propiedad del producto. Un pack propio es la solución barata, y compra una frontera: un

ensamblado guarda la ilustración, un catálogo de C# la nombra y el resto de la aplicación no vuelve a escribir una ruta.

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
Ensamblado de recursos	Una biblioteca de clases.NET corriente cuyo único trabajo es llevar archivos de icono como recursos incrustados; no necesita referenciar la aplicación ni tener código propio de Wisej.NET.
La receta de cuatro pasos	Crea una biblioteca de clases, crea un directorio llamado Resources, pon dentro los archivos de icono y establece su Build Action en Embedded Resource. Todo lo que viene después es comodidad.
Catálogo de iconos	Una clase estática como Applcons que expone cada icono como una constante de cadena con nombre, de modo que el compilador e IntelliSense sustituyen a las rutas copiadas y pegadas.
Nombre semántico frente a nombre de archivo	El catálogo expone el vocabulario de tu producto —guardar, eliminar, cliente, advertencia— mientras el archivo conserva un nombre sencillo, así que renombrarlo cambia una línea y no cuarenta pantallas.
Dirección de recurso calificada	Un icono de otro ensamblado se direcciona como resource.wx/ más el nombre del ensamblado más la ruta del recurso, y por eso todas las entradas del catálogo comparten una constante raíz privada.
Contrato de recoloreado	El sufijo?color= solo recolorea ilustraciones monocromas adecuadas; un SVG con valores de fill fijados en el archivo es la razón habitual de que un icono se niegue a cambiar de color.
Conjuntos temáticos y sobrescrituras	Paquetes paralelos que comparten una misma API de catálogo sobre ilustraciones distintas, más archivos desplegados que tienen prioridad sobre un recurso incrustado, permiten sustituir iconos sin tocar el código de la aplicación.

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 6. Puedes abrir tú mismo [module-06.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 IconDesk 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 6 - «Crea tu propio pack de iconos de Wisej.NET» - del curso de LearnWisej «Iconos e imágenes 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/icons-and-images/specs/module-06.md>

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

1. Proyecto «IconDesk», 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: Crea `IconDesk.Icons` como una biblioteca de clases junto a la aplicación, pon al menos doce iconos SVG en un directorio `Resources` y márcalos como `Embedded Resource`. Añade un catálogo estático `AppIcons` que exponga cada icono como una cadena `resource.wx` con nombre para que quien lo use nunca escriba una ruta, referencia la biblioteca desde `IconDesk` y después asigna algunos iconos mediante el diseñador y otros desde el catálogo en C#. Verifica el sufijo de color en tres iconos y anota la limpieza del SVG que alguno de ellos necesitó. Termina con las reglas de nomenclatura y versionado que deciden cómo entra un icono nuevo en el pack.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Biblioteca de clases `IconDesk.Icons` con al menos doce archivos SVG en un directorio `Resources`, todos como `Embedded Resource`
 - Catálogo estático `AppIcons` que expone cada icono como una cadena `resource.wx` con nombre
 - Página de `IconDesk` que asigna algunos iconos mediante el diseñador y otros desde el catálogo `AppIcons`
 - Tres iconos verificados con el sufijo de color más una nota sobre la limpieza del SVG que necesitaron
 - Reglas de nomenclatura y versionado que describen cómo entra un icono nuevo en el pack y cómo una aplicación sobrescribe uno
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 pack compila y el diseñador no muestra nada en el Image Selector. ¿Qué paso de la receta comprobarías primero y qué prueba te diría que los recursos están de verdad en el ensamblado?
- Dos de tus tres iconos de prueba se recolorean y el tercero se queda gris. ¿Qué hay en ese SVG y cuándo dejarías a propósito un icono sin posibilidad de recoloreado?

– Un cliente quiere sustituir tres iconos solo en su despliegue. ¿Cuál de las vías de sobrescritura elegirías y qué te cuesta esa elección en la siguiente versión del pack?

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 pack compila y el diseñador no muestra nada en el Image Selector. ¿Qué paso de la receta comprobarías primero y qué prueba te diría que los recursos están de verdad en el ensamblado?
- ✓ Dos de tus tres iconos de prueba se recolorean y el tercero se queda gris. ¿Qué hay en ese SVG y cuándo dejarías a propósito un icono sin posibilidad de recolorado?
- ✓ Un cliente quiere sustituir tres iconos solo en su despliegue. ¿Cuál de las vías de sobrescritura elegirías y qué te cuesta esa elección en la siguiente versión del pack?

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/icons-and-images/specs/module-06.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.** *"Explicame este código de IconDesk 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 IconDesk 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.