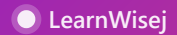


Exercice de code avec l'IA

Construisez l'exemple IconDesk de ce module avec ChatGPT ou Claude à partir d'un prompt prêt à l'emploi qui oriente le modèle vers la spécification du module, puis relisez, exécutez et étendez ce qu'il vous renvoie.



Vous ne taperez pas cet exemple. Vous le piloterez.

Presque plus personne n'écrit du code Wisej.NET à partir d'un fichier vide — un assistant rédige le premier jet et le développeur décide s'il est correct. Cela ne fonctionne que si vous maîtrisez le sujet, et c'est ce que le reste de ce module vous a apporté. Ici, vous confiez ces connaissances à ChatGPT ou à Claude, vous récupérez l'exemple du module sous forme de solution Visual Studio fonctionnelle, et vous le relisez comme vous reliriez la pull request d'un collègue.

Ce que vous demandez

Créez le projet IconDesk et dotez-le d'un panneau de commandes avec les boutons Open, Save, Print et Delete, chacun portant son icône par un mécanisme différent. Chargez btnOpen depuis un System.Drawing.Image côté serveur, affectez une image de thème nommée à btnSave.ImageSource, et faites pointer btnPrint et btnDelete vers une même ImageList partagée via ImageKey. Ajoutez un libellé de diagnostic qui inspecte chaque bouton via Image et indique quel mécanisme est réellement en jeu, puis remplacez l'image stockée sous une clé de l'ImageList et vérifiez que tous les contrôles qui utilisent cette clé suivent le changement.

Le module demande ces livrables, et le prompt ci-dessous demande exactement ceux-là à l'assistant :

- ✓ Panneau de commandes IconDesk avec les boutons Open, Save, Print et Delete
- ✓ btnOpen portant un System.Drawing.Image chargé sur le serveur et btnSave une image de thème nommée sur ImageSource
- ✓ btnPrint et btnDelete partageant une même ImageList via ImageKey plutôt qu'ImageIndex
- ✓ Libellé de diagnostic qui lit chaque bouton via Image et nomme le mécanisme derrière son icône
- ✓ Note de lab consignant ce qui s'est passé lors du remplacement d'une clé de l'ImageList et lorsque deux propriétés d'image ont été définies sur le même bouton

Pourquoi il est conçu ainsi

Wisej.NET propose délibérément plus d'une manière de poser un graphique sur un contrôle, et toutes fonctionnent. L'ennui commence quand une équipe les utilise toutes par accident: une barre d'outils où un bouton lit un fichier sur le disque du serveur, le suivant pointe vers un nom de thème, et un troisième a discrètement perdu son icône parce que deux propriétés ont été définies sur le même contrôle. Six mois

plus tard, personne ne sait expliquer d'où vient réellement une image donnée ni pourquoi elle a disparu après le déploiement.

Voici les termes que l'assistant doit maîtriser. Si ce qu'il vous renvoie les emploie de façon approximative, il n'a pas compris le module :

Terme	Ce qu'il signifie ici
propriété Image	Un objet <code>System.Drawing.Image</code> qui vit dans la mémoire du serveur; Wisej.NET l'envoie au navigateur sous forme de données PNG, si bien que l'image voyage en octets que votre code a déjà décodés.
chaîne ImageSource	Une chaîne qui identifie une ressource à résoudre par le client: une URL relative, une URL absolue, une image de thème nommée, un chemin <code>resource.wx</code> ou un élément d'un assembly de pack d'icônes.
IconSource	La même idée de chaîne, sur les contrôles qui possèdent un emplacement d'icône dédié à côté de leur image principale; vérifiez donc l'API du contrôle que vous utilisez au lieu de supposer que tous nomment la propriété de la même façon.
contrat <code>IImage</code>	La famille d'images commune qu'exposent les contrôles capables d'en porter une — <code>Image</code> , <code>ImageAlign</code> , <code>ImageIndex</code> , <code>ImageKey</code> , <code>ImageList</code> et <code>ImageSource</code> — et qui permet à une seule méthode utilitaire de décorer uniformément un <code>Button</code> , un <code>Label</code> ou n'importe quoi d'autre.
<code>ImageList</code>	Une collection réutilisable d'objets image vers laquelle plusieurs contrôles peuvent pointer, adressée par position ou par nom, et capable d'établir une taille d'image commune pour les contrôles qui l'utilisent.
<code>ImageKey</code> contre <code>ImageIndex</code>	Une clé telle que "delete" porte du sens et survit à un réordonnement; l'index 3 devient silencieusement la mauvaise image le jour où quelqu'un insère une image au-dessus.
affectation mutuellement exclusive	Toutes ces propriétés alimentent un seul emplacement visuel: en affecter une peut donc réinitialiser les autres — et c'est pourquoi une mauvaise icône est généralement une seconde affectation, pas un fichier cassé.

Construisez-le — choisissez votre assistant

Les deux boutons ouvrent un nouvel onglet avec le **même prompt prêt à l'emploi** déjà rempli. Le prompt oriente le modèle vers la spécification de ce module — un unique fichier Markdown qui contient la leçon, l'énoncé du lab et les livrables — pour qu'il travaille sur le vrai contenu au lieu de deviner de quoi parle le Module 1. Vous pouvez ouvrir vous-même [module-01.md](#) pour voir exactement ce qu'il lira.



Utiliser ChatGPT

Ouvre [chatgpt.com](#) avec le prompt de ce module déjà chargé. Fonctionne avec GPT-5 ou Codex ; toute offre capable de



Utiliser Claude

Ouvre [claude.ai](#) avec le même prompt déjà chargé. Si vous utilisez Claude Code, vous pouvez plutôt le coller dans un terminal ouvert sur un dossier vide.

naviguer sur le web récupérera la spécification.



Envoyez-le tel quel et vous obtenez un projet, pas un extrait

Le prompt se termine en demandant toute la solution sous forme d'un unique `.zip`, avec les étapes pour la compiler et l'exécuter. Téléchargez-le, ouvrez la solution dans Visual Studio, appuyez sur F5 et IconDesk s'ouvre sur `localhost`. Il vous faut le SDK .NET et le package NuGet `Wisej-4`; tout le reste, le prompt demande à l'assistant de l'écrire lui-même.

Je suis le Module 1 – « Le modèle d'image de Wisej.NET : Image, ImageSource et ImageList » – du cours LearnWisej « Icônes et images dans Wisej.NET », et je veux construire avec toi l'exemple de ce module. Réponds-moi en français.

Commence par lire la spécification du module. C'est un unique fichier Markdown (en anglais) qui contient la leçon complète, l'énoncé du lab et les livrables :

<https://learnwisej.com/assets/courses/icons-and-images/specs/module-01.md>

Construis ensuite un projet Wisej.NET 4 complet que je puisse ouvrir dans Visual Studio :

1. Projet « IconDesk », C#, net8.0 ou ultérieur, package NuGet Wisej-4 4.1.x, hôte Kestrel qui appelle `app.UseWisej()`.
2. Construis ce que demande le lab de la spécification : Créez le projet IconDesk et dotez-le d'un panneau de commandes avec les boutons Open, Save, Print et Delete, chacun portant son icône par un mécanisme différent. Chargez `btnOpen` depuis un `System.Drawing.Image` côté serveur, affectez une image de thème nommée à `btnSave.ImageSource`, et faites pointer `btnPrint` et `btnDelete` vers une même `ImageList` partagée via `ImageKey`. Ajoutez un libellé de diagnostic qui inspecte chaque bouton via `IImage` et indique quel mécanisme est réellement en jeu, puis remplacez l'image stockée sous une clé de l'`ImageList` et vérifiez que tous les contrôles qui utilisent cette clé suivent le changement.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - Panneau de commandes IconDesk avec les boutons Open, Save, Print et Delete
 - `btnOpen` portant un `System.Drawing.Image` chargé sur le serveur et `btnSave` une image de thème nommée sur `ImageSource`
 - `btnPrint` et `btnDelete` partageant une même `ImageList` via `ImageKey` plutôt qu'`ImageIndex`
 - Libellé de diagnostic qui lit chaque bouton via `IImage` et nomme le mécanisme derrière son icône
 - Note de lab consignant ce qui s'est passé lors du remplacement d'une clé de l'`ImageList` et lorsque deux propriétés d'image ont été définies sur le même bouton
4. Suis la spécification partout où elle nomme une classe, une propriété ou un événement Wisej.NET – utilise exactement celui-là. Là où elle laisse le choix, dis-moi quelle API tu as choisie et pourquoi.
5. Garde l'autorité côté serveur : l'état et la validation vivent dans .NET, le navigateur affiche et remonte les informations. Ne déplace jamais une décision dans JavaScript parce que c'est plus simple là-bas.
6. Rends le résultat démontrable : je dois pouvoir l'exécuter et voir le comportement dont traite ce module, y compris ce qui se passe quand quelque chose tourne mal.

Avant d'écrire la moindre ligne de code, reformule avec tes propres mots ce que tu vas construire et où vivra chaque élément, pour que je puisse corriger la conception tant que cela ne coûte pas cher.

Quand le code est terminé, rassemble toute la solution dans un unique fichier .zip téléchargeable et donne-moi les étapes exactes pour la compiler et l'exécuter.

Enfin, réponds à ces questions de la spécification en t'appuyant sur ta propre implémentation :

- Votre libellé de diagnostic indique que `btnDelete` utilise une source d'image, alors que vous n'avez jamais défini que son `ImageKey`. Qu'est-ce qui a bien pu s'exécuter, et où le chercheriez-vous?
- L'impression et la suppression ont toutes deux changé lorsque vous avez remplacé une entrée de l'`ImageList`. Serait-ce encore vrai si les boutons utilisaient `ImageIndex`, et que

se passerait-il si une cinquième image était insérée en tête de liste?

– Le bouton Open lit son fichier sur le serveur. Quelle partie de ce montage casse lorsque l'application est déployée sur une autre machine, et quelle partie ne casserait pas si la même image était une chaîne de source d'image?

Pose-moi des questions sur tout ce qui est ambigu avant de commencer.

Puis relisez ce qui vous revient

Un code qui compile n'est pas encore un code juste. Les questions intéressantes sont des questions de conception : est-ce bien le contrôle Wisej.NET que la tâche exige, suit-il le cycle de vie que le framework vous offre, l'état de référence est-il sur le serveur ou a-t-il été laissé dans le navigateur parce que c'était pratique ? Lisez le résultat en relecteur, pas en client.

Ce module vous donne la checklist. Répondez à chaque point en vous appuyant sur le code que vous venez de recevoir :

- ✓ Votre libellé de diagnostic indique que btnDelete utilise une source d'image, alors que vous n'avez jamais défini que son ImageKey. Qu'est-ce qui a bien pu s'exécuter, et où le chercheriez-vous?
- ✓ L'impression et la suppression ont toutes deux changé lorsque vous avez remplacé une entrée de l' ImageList. Serait-ce encore vrai si les boutons utilisaient ImageIndex, et que se passerait-il si une cinquième image était insérée en tête de liste?
- ✓ Le bouton Open lit son fichier sur le serveur. Quelle partie de ce montage casse lorsque l'application est déployée sur une autre machine, et quelle partie ne casserait pas si la même image était une chaîne de source d'image?

Tout ce que vous avez dû corriger mérite d'être noté. Une courte note des conventions et des décisions retenues, collée dans le premier message de votre prochaine session, vaut plus que n'importe quel prompt astucieux.



Pas encore d'exemple de référence pour ce cours — le vôtre est le premier

Le [dépôt des exemples LearnWisej](#) contient des solutions complètes pour la plupart des cours, mais pas encore pour celui-ci. Il n'y a donc rien à quoi vous comparer et aucun raccourci : la spécification et votre propre relecture sont votre seul filet de sécurité. Gardez ce que vous construisez — et si le résultat est réussi, nous aimerions le voir.

Trois prompts pour aller plus loin

Orientez votre assistant vers le projet que vous venez de construire et essayez ceux-ci. Chacun est un exercice de relecture et d'extension, pas une réécriture.

- ✓ **Prouvez que ça fonctionne vraiment.** *"Passe en revue ce projet livrable par livrable par rapport à la spécification du module sur <https://learnwisej.com/assets/courses/icons-and-images/specs/module-01.md> et montre-moi la preuve que chacun est réellement terminé — le chemin dans le code, où cliquer et ce que je dois voir. Là où un livrable n'est que partiellement satisfait, dis-le clairement au lieu de le défendre. Ensuite, ajoute ce qui manque."*

- ✓ **Relisez-le comme une pull request.** *"Fais-moi parcourir ce code IconDesk comme si tu relisais la pull request d'un collègue et que tu ne l'avais pas écrit. Où vit l'état, que se passe-t-il de bout en bout lors d'une action de l'utilisateur, que renverrais-tu pour correction, et quelles parties ne serais-tu pas à l'aise de montrer à un client ? Sois précis et cite les lignes dont tu parles — ne me résume pas la conception."*
- ✓ **Remplacez une pièce sans déplacer les jointures.** *"Choisis la partie de ce projet IconDesk qui remplace le plus manifestement quelque chose de réel — une source de données factice, un stockage en mémoire, un service simplifié — et remplace-la par une implémentation digne de la production. Le contrat de part et d'autre ne doit pas changer : mêmes types, mêmes appels, même comportement en cas d'échec. Dis-moi ce que l'implémentation réelle t'a obligé à gérer et que la version provisoire ne gèrait pas."*



Emportez-le dans le lab

Le lab pratique qui clôt ce module demande les mêmes livrables que ceux que vous venez de faire produire à l'assistant. Servez-vous de ce que vous avez ici comme forme de votre réponse et comme preuve à l'appui — pas comme la réponse elle-même. Le lab est noté sur votre raisonnement, et cette partie doit venir de vous.