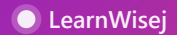


Exercice de code avec l'IA

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.



Écrire le code, c'est désormais la moitié facile.

Obtenir d'un assistant moderne du code qui fonctionne pour « Packs d'icônes Wisej.NET et NuGet » ne demande qu'un bon prompt. Le juger, le corriger et l'étendre demande tout ce que ce module a couvert. C'est sur ce partage que repose l'exercice : vous apportez la spécification et la relecture, l'assistant apporte la frappe.

Ce que vous demandez

Ajoutez à IconDesk deux packages NuGet officiels de packs d'icônes Wisej-4 et construisez une page IconCompare qui montre les mêmes cinq concepts, Save, Delete, Search, User et Settings, tirés des deux familles, pour que la différence visuelle soit impossible à manquer. Choisissez au moins cinq des icônes via l'explorateur d'images de Visual Studio plutôt qu'en saisissant le chemin, affectez les autres depuis le catalogue du pack en C#, recolorez-en deux avec le suffixe ?color=, et observez le panneau réseau pour voir les fichiers d'icônes arriver une requête à la fois, puis provenir du cache du navigateur. Concluez par une courte recommandation désignant la famille d'icônes principale du projet.

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

- ✓ Deux packages NuGet officiels de packs d'icônes Wisej-4 référencés par IconDesk
- ✓ Page IconCompare montrant Save, Delete, Search, User et Settings dans les deux packs
- ✓ Au moins cinq icônes choisies via l'explorateur d'images de Visual Studio plutôt que saisies à la main
- ✓ Deux icônes recolorées avec le suffixe de couleur et une note indiquant quel visuel a accepté le recolorage
- ✓ Recommandation d'un paragraphe désignant la famille d'icônes principale du projet, avec les preuves de mise en cache par le navigateur qui la justifient

Pourquoi il est conçu ainsi

Trois modules plus loin, IconDesk sait déjà afficher une image de cinq façons différentes, mais chaque icône qu'il possède a été dessinée ou téléchargée à la main. Une vraie application a besoin d'une icône Save, Delete, Search, User et Settings dès le premier jour, et de quarante autres d'ici la fin du trimestre, toutes dans le même langage visuel. Copier des fichiers SVG un par un produit un dossier de visuels sans lien, des épaisseurs de trait incohérentes et aucune traçabilité des licences.

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
pack d'icônes	Un assembly.NET ordinaire dont les fichiers d'icônes, le plus souvent des SVG, sont compilés en ressources embarquées; le référencer ajoute un seul fichier à votre déploiement, quel que soit le nombre d'icônes qu'il contient.
ressource embarquée	Un fichier compilé dans l'assembly plutôt que copié à côté, si bien que les icônes ne peuvent être ni perdues, ni renommées, ni oubliées lorsque l'application est déployée sur un autre serveur.
resource.wx	Le gestionnaire Wisej.NET qui sert une ressource embarquée par HTTP; une source d'image de pack d'icônes est une URL resource.wx qui nomme l'assembly du pack et le fichier qu'il contient.
catalogue d'icônes	La classe statique qu'un pack publie pour que les appelants obtiennent l'IntelliSense et un nom vérifié à la compilation au lieu d'un chemin saisi; le namespace et le nom de classe exacts varient d'un pack à l'autre.
requête par icône	Les icônes de packs sont récupérées individuellement, à la première utilisation, puis servies depuis le cache du navigateur, et c'est ce qui rend abordable un catalogue de plusieurs milliers d'icônes.
préchargement des images de thème	Les images de thème voyagent avec le thème et sont déjà dans le cache de thème du client au démarrage de l'application: elles se comportent donc très différemment des icônes de packs au lancement.
famille d'icônes principale	Le pack unique auquel un produit s'engage pour ses commandes quotidiennes, choisi délibérément pour sa couverture, son style et sa licence plutôt qu'accumulé pack après pack.

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 4. Vous pouvez ouvrir vous-même [module-04.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 naviguer sur le web récupérera la spécification.



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.



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 4 - « Packs d'icônes Wisej.NET et NuGet » - 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-04.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 : Ajoutez à IconDesk deux packages NuGet officiels de packs d'icônes Wisej-4 et construisez une page IconCompare qui montre les mêmes cinq concepts, Save, Delete, Search, User et Settings, tirés des deux familles, pour que la différence visuelle soit impossible à manquer. Choisissez au moins cinq des icônes via l'explorateur d'images de Visual Studio plutôt qu'en saisissant le chemin, affectez les autres depuis le catalogue du pack en C#, recolorez-en deux avec le suffixe `?color=`, et observez le panneau réseau pour voir les fichiers d'icônes arriver une requête à la fois, puis provenir du cache du navigateur. Concluez par une courte recommandation désignant la famille d'icônes principale du projet.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - Deux packages NuGet officiels de packs d'icônes Wisej-4 référencés par IconDesk
 - Page IconCompare montrant Save, Delete, Search, User et Settings dans les deux packs
 - Au moins cinq icônes choisies via l'explorateur d'images de Visual Studio plutôt que saisies à la main
 - Deux icônes recolorees avec le suffixe de couleur et une note indiquant quel visuel a accepté le recolorage
 - Recommandation d'un paragraphe désignant la famille d'icônes principale du projet, avec les preuves de mise en cache par le navigateur qui la justifie
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 :

- Référencer un pack de trois mille icônes ajoute un seul assembly au déploiement. Qu'est-ce que cela coûte au navigateur, et à quel moment?
- L'une de vos deux icônes recolorees a ignoré le suffixe de couleur. Qu'a de particulier ce visuel, et que changeriez-vous dans les critères de sélection des packs à cause de cela?

– Un coéquipier veut que chaque icône de commande suive le thème lorsque l'application en change. Lesquelles des cinq icônes d'IconCompare devraient devenir des images de thème, et lesquelles non?

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 :

- ✓ Référencer un pack de trois mille icônes ajoute un seul assembly au déploiement. Qu'est-ce que cela coûte au navigateur, et à quel moment?
- ✓ L'une de vos deux icônes recolorées a ignoré le suffixe de couleur. Qu'a de particulier ce visuel, et que changeriez-vous dans les critères de sélection des packs à cause de cela?
- ✓ Un coéquipier veut que chaque icône de commande suive le thème lorsque l'application en change. Lesquelles des cinq icônes d'IconCompare devraient devenir des images de thème, et lesquelles non?

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.

- ✓ **Faites-le résister aux mauvais jours.** *"Examine ce projet IconDesk pour repérer les modes de défaillance qu'il ne gère pas encore — une saisie erronée ou manquante, une dépendance lente ou indisponible, un traitement qui aboutit après que l'utilisateur a changé de page, et deux actions qui se chevauchent. Pour chacun, dis-moi d'abord s'il s'agit d'un risque réel dans Wisej.NET 4 et pourquoi, et seulement ensuite écris la correction. Là où tu estimes qu'aucune gestion n'est nécessaire, dis-le et explique ton raisonnement au lieu d'ajouter du code défensif."*
- ✓ **Ajoutez le deuxième cas.** *"Étends ce projet IconDesk avec un deuxième cas qui suit le même modèle que le premier — une autre entité, un autre écran ou un autre contrôle de même forme,*

câblé de la même manière. Extrais dans une classe de base ou un service ce qui est réellement partagé, mais seulement là où cela se justifie ; deux choses similaires n'appellent pas toujours une abstraction. Montre-moi le diff et justifie chaque élément que tu as extrait."

- ✓ **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-04.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."



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.