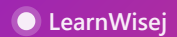


Exercice de code avec l'IA

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.



Briefez-le, lisez-le, puis cassez-le.

Cet exercice comporte trois parties, et seule la première concerne le prompt. Vous briefez un assistant avec la spécification du module et récupérez une solution IconDesk ; vous lisez ce qu'il a écrit et décidez s'il fait vraiment ce que le module demandait ; puis vous le poussez plus loin que ce que le module demandait. C'est dans la deuxième et la troisième partie que l'on apprend.

Ce que vous demandez

Remplissez une page IconGallery dans IconDesk à l'aide de l'Image Selector du designer plutôt qu'en saisissant les chaînes de source à la main. Affectez des images à cinq contrôles : une image de thème nommée, un SVG du projet référencé par URL relative, une URL absolue et au moins deux ressources issues d'un pack d'icônes si un pack est installé. Recolez un SVG monochrome avec le suffixe ?color= et un nom de couleur du thème, ouvrez le Designer.cs généré et lisez les chaînes de source d'image produites par le sélecteur, puis basculez l'application entre deux thèmes Wisej et notez quelles ressources ont changé et lesquelles sont restées strictement identiques.

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

- ✓ Page IconGallery avec cinq contrôles dont les images ont été affectées via l'Image Selector du designer
- ✓ Parmi elles, une image de thème nommée, un SVG relatif du projet, une URL absolue et au moins deux ressources de pack d'icônes
- ✓ SVG monochrome recoloré via le suffixe?color= avec un nom de couleur du thème
- ✓ Extrait de Designer.cs montrant les chaînes de source d'image générées par le sélecteur
- ✓ Note de changement de thème consignant les ressources qui ont suivi le thème et celles qui sont restées fixes

Pourquoi il est conçu ainsi

IconDesk possède désormais un panneau de commandes et un atelier d'images, et chaque image qu'ils contiennent est un fichier livré assorti d'un chemin saisi par quelqu'un. Cela tient jusqu'à ce que l'application change de thème, soit déployée sous un autre répertoire virtuel, ou qu'on lui demande d'afficher le même symbole dans une couleur d'avertissement sur un écran et dans une couleur neutre sur un autre. Ce module fait sortir l'origine d'une image du dossier du projet pour la placer dans le thème,

dans une URL ou dans le designer, afin que la chaîne portée par un contrôle signifie encore quelque chose après un changement de visuel, de thème ou de serveur.

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
image de thème nommée	Un nom sémantique tel qu' icon-print que le thème actif associe à un visuel, si bien qu'un autre thème peut fournir une autre image sans le moindre changement en C#.
cache de thème du client	Les images de thème voyagent avec le thème et sont déjà dans le navigateur quand un contrôle en demande une, ce qui fait d'un thème le bon foyer pour un petit jeu de symboles de commande et le mauvais foyer pour des milliers.
source en URL relative	Une chaîne de source d'image résolue par rapport à l'application: la forme la plus simple pour un logo déployé avec l'application.
source en URL absolue	Une adresse externe complète qui transforme l'image en dépendance envers un autre système, avec des conséquences de disponibilité, de confidentialité, de CORS et de Content Security Policy.
image base64 embarquée	Les octets de l'image écrits dans la définition de thème elle-même, si bien que le visuel n'a besoin d'aucune seconde requête, au prix d'un thème plus lourd.
Image Selector	Le sélecteur au design de Visual Studio qui parcourt les images du projet, les images de thème et les ressources des packs d'icônes référencés, puis sérialise la chaîne de source choisie dans Designer.cs.
suffixe?color=	Un suffixe sur une source d'image qui demande à Wisej.NET de recolorer un SVG monochrome compatible avec une couleur CSS ou un nom de couleur du thème.

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 3. Vous pouvez ouvrir vous-même [module-03.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 3 – « Images de thème, URL, couleur des SVG et sélecteur d'images du designer » – 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-03.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 : Remplissez une page `IconGallery` dans `IconDesk` à l'aide de l'`Image Selector` du designer plutôt qu'en saisissant les chaînes de source à la main. Affectez des images à cinq contrôles : une image de thème nommée, un SVG du projet référencé par URL relative, une URL absolue et au moins deux ressources issues d'un pack d'icônes si un pack est installé. Recolorez un SVG monochrome avec le suffixe `?color=` et un nom de couleur du thème, ouvrez le `Designer.cs` généré et lisez les chaînes de source d'image produites par le sélecteur, puis basculez l'application entre deux thèmes `Wisej` et notez quelles ressources ont changé et lesquelles sont restées strictement identiques.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - Page `IconGallery` avec cinq contrôles dont les images ont été affectées via l'`Image Selector` du designer
 - Parmi elles, une image de thème nommée, un SVG relatif du projet, une URL absolue et au moins deux ressources de pack d'icônes
 - SVG monochrome recoloré via le suffixe `?color=` avec un nom de couleur du thème
 - Extrait de `Designer.cs` montrant les chaînes de source d'image générées par le sélecteur
 - Note de changement de thème consignait les ressources qui ont suivi le thème et celles qui sont restées fixes
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 :

- La galerie s'affiche correctement sur votre machine et ne montre rien pour l'URL absolue sur celle d'un collègue. Lequel des cinq mécanismes pourrait produire cette différence, et que vérifieriez-vous en premier?
- Deux de vos icônes refusent de changer de couleur quand le suffixe est ajouté. Qu'y a-t-il

à l'intérieur de ces fichiers, et quand les modifier est-il la mauvaise réponse?

- Un designer demande le même symbole dans la galerie, dans une barre d'outils et dans une cellule de grille. Entre la clé de thème, l'URL relative et la ressource de pack, laquelle coûtera le moins cher à changer plus tard, et pourquoi?

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 :

- ✓ La galerie s'affiche correctement sur votre machine et ne montre rien pour l'URL absolue sur celle d'un collègue. Lequel des cinq mécanismes pourrait produire cette différence, et que vérifieriez-vous en premier?
- ✓ Deux de vos icônes refusent de changer de couleur quand le suffixe est ajouté. Qu'y a-t-il à l'intérieur de ces fichiers, et quand les modifier est-il la mauvaise réponse?
- ✓ Un designer demande le même symbole dans la galerie, dans une barre d'outils et dans une cellule de grille. Entre la clé de thème, l'URL relative et la ressource de pack, laquelle coûtera le moins cher à changer plus tard, et pourquoi?

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.

- ✓ **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 gérait pas."*

- ✓ **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."*



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.