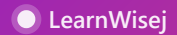


Exercice de code avec l'IA

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.



L'assistant, ce sont les mains rapides. L'architecte, c'est toujours vous.

Un assistant peut produire l'exemple de ce module en quelques minutes. Ce qu'il ne peut pas faire, c'est décider si le résultat est la bonne conception pour votre application — cette décision vous revient. C'est vous qui apportez le jugement : briefez-le correctement, puis vérifiez ce qu'il vous renvoie à la lumière de ce que le Module 7 vient de vous apprendre.

Ce que vous demandez

Terminez le projet de synthèse IconDesk. Référez une feuille de style de police d'icônes depuis Default.html, activez AllowHtml sur une petite barre d'outils ou une carte et affichez ses icônes sous forme de glyphes de police, puis construisez une colonne DataGridView dont les cellules contiennent des actions en police d'icônes et lisez l'élément sur lequel l'utilisateur a cliqué pour qu'Edit et Delete fassent des choses différentes. Construisez une page de synthèse qui démontre d'un seul coup tous les mécanismes vus dans le cours : une image de thème, une icône d'un pack officiel, une icône d'un pack personnalisé, un logo embarqué et un SVG recoloré. Rédigez ensuite la politique d'icônes de production du projet et effectuez une passe de QA dans un thème clair et un thème sombre.

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

- ✓ Feuille de style de police d'icônes référencée depuis Default.html et une barre d'outils ou une carte en AllowHtml affichant des glyphes de police
- ✓ Colonne DataGridView dont les cellules affichent des actions en police d'icônes et réagissent à l'élément cliqué
- ✓ Page de synthèse IconDesk démontrant une image de thème, une icône d'un pack officiel, une icône d'un pack personnalisé, une ressource embarquée et un SVG recoloré
- ✓ Politique d'icônes de production écrite couvrant l'accessibilité, les licences, la mise en cache et les dépendances externes
- ✓ Passe de QA effectuée dans un thème clair et un thème sombre avec les constats consignés

Pourquoi il est conçu ainsi

Après six modules, IconDesk sait dessiner une icône de cinq façons différentes, et toutes fonctionnent. Ce qui manque encore au projet, c'est une règle de décision: quand un nouvel écran a besoin d'un symbole Save, personne ne devrait avoir à deviner s'il vient du thème, d'un pack officiel Wisej-4 ou

d'IconDesk.Icons. Ce module ajoute la dernière option, les polices d'icônes rendues en HTML, puis écrit la politique qui dit vers laquelle se tourner et ce qu'il faut vérifier avant qu'un écran ne parte.

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
police d'icônes	Une police web dont les glyphes sont des images au lieu de lettres, choisies par des noms de classes CSS et dessinées par le navigateur dans du contenu HTML plutôt que via une propriété d'image.
AllowHtml	L'interrupteur par contrôle qui indique à Wisej.NET d'interpréter le texte d'un contrôle comme du markup; sans lui, les balises apparaissent en caractères littéraux, car le texte est du texte simple par défaut.
en-tête de Default.html	La page hôte de l'application, où un seul lien de feuille de style rend le CSS et les fichiers de police de la police d'icônes disponibles pour chaque écran ouvert par la session.
clic d'élément fondé sur le rôle	Un clic dans du contenu HTML peut indiquer quel élément en ligne a été atteint, si bien qu'une seule cellule de grille peut offrir une cible Edit et une cible Delete sans deux contrôles distincts.
CDN contre auto-hébergement	Pointer le lien vers un CDN public est une décision de production touchant à la disponibilité, à la confidentialité, à la Content Security Policy et aux licences; les applications internes hébergent souvent les fichiers elles-mêmes.
politique d'icônes par couches	Une règle écrite qui donne un foyer unique à chaque type de symbole: images de thème pour les sémantiques, un pack principal pour les commandes, le pack personnalisé pour les visuels du produit, fichiers raster pour les photos.
passe de QA sur les icônes	Un parcours des écrans finis dans un thème clair et un thème sombre, à plus d'une échelle d'affichage, pour vérifier que chaque icône est toujours visible et dispose toujours d'un équivalent textuel.

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 7. Vous pouvez ouvrir vous-même [module-07.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 7 – « Polices d'icônes, contenu HTML et stratégie d'icônes en production » – 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-07.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 : Terminez le projet de synthèse IconDesk. Référenciez une feuille de style de police d'icônes depuis `Default.html`, activez `AllowHtml` sur une petite barre d'outils ou une carte et affichez ses icônes sous forme de glyphes de police, puis construisez une colonne `DataGridView` dont les cellules contiennent des actions en police d'icônes et lisez l'élément sur lequel l'utilisateur a cliqué pour qu'`Edit` et `Delete` fassent des choses différentes. Construisez une page de synthèse qui démontre d'un seul coup tous les mécanismes vus dans le cours : une image de thème, une icône d'un pack officiel, une icône d'un pack personnalisé, un logo embarqué et un SVG recoloré. Rédigez ensuite la politique d'icônes de production du projet et effectuez une passe de QA dans un thème clair et un thème sombre.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - Feuille de style de police d'icônes référencée depuis `Default.html` et une barre d'outils ou une carte en `AllowHtml` affichant des glyphes de police
 - Colonne `DataGridView` dont les cellules affichent des actions en police d'icônes et réagissent à l'élément cliqué
 - Page de synthèse IconDesk démontrant une image de thème, une icône d'un pack officiel, une icône d'un pack personnalisé, une ressource embarquée et un SVG recoloré
 - Politique d'icônes de production écrite couvrant l'accessibilité, les licences, la mise en cache et les dépendances externes
 - Passe de QA effectuée dans un thème clair et un thème sombre avec les constats consignés
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 :

- Le lien de la feuille de style est dans `Default.html` et les noms de classes sont corrects,

mais un Label affiche les balises brutes alors que le Button voisin affiche le glyphe. Qu'est-ce qui diffère entre les deux contrôles, et pourquoi le réglage est-il par contrôle plutôt que par application?

– Votre colonne d'actions fonctionne, et un collègue demande à placer le texte de la note du client dans la même cellule pour pouvoir le styler. Que lui répondez-vous, et que construiriez-vous à la place?

– Sur la page de synthèse, lesquelles des cinq images changent quand l'application change de thème et lesquelles restent exactement telles qu'elles ont été dessinées? Que cela vous dit-il sur l'endroit où chacune est stockée?

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 :

- ✓ Le lien de la feuille de style est dans Default.html et les noms de classes sont corrects, mais un Label affiche les balises brutes alors que le Button voisin affiche le glyphe. Qu'est-ce qui diffère entre les deux contrôles, et pourquoi le réglage est-il par contrôle plutôt que par application?
- ✓ Votre colonne d'actions fonctionne, et un collègue demande à placer le texte de la note du client dans la même cellule pour pouvoir le styler. Que lui répondez-vous, et que construiriez-vous à la place?
- ✓ Sur la page de synthèse, lesquelles des cinq images changent quand l'application change de thème et lesquelles restent exactement telles qu'elles ont été dessinées? Que cela vous dit-il sur l'endroit où chacune est stockée?

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.

- ✓ **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."*
- ✓ **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."*



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.