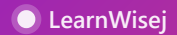


Exercice de code avec l'IA

Construisez l'exemple FieldOps 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 « Shells mobiles avec .NET MAUI » 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

Construisez le shell mobile de FieldOps avec .NET MAUI : une page hybride qui héberge l'application sur Android et iOS, une session qui survit au passage en arrière-plan et reprend au retour, le bouton Retour d'Android associé à la navigation dans l'application, des marges de zone sûre pour les appareils à encoche, et un build de débogage exécuté sur un émulateur par plateforme avec des captures d'écran de la liste des ordres de travail.

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

- ✓ Projet de shell hybride.NET MAUI pour Android et iOS
- ✓ Gestion du cycle de vie pour l'arrière-plan et la reprise
- ✓ Gestion du bouton Retour et des zones sûres
- ✓ Builds sur émulateur pour les deux plateformes
- ✓ Captures d'écran des deux émulateurs

Pourquoi il est conçu ainsi

Les techniciens de FieldOps travaillent depuis un téléphone dans une salle des machines, un sous-sol ou une camionnette, et un onglet de navigateur est un mauvais conteneur pour cela: pas d'icône d'écran d'accueil à distribuer, pas de canal store ou entreprise, et aucun accès à la caméra et au GPS dont le Module 5 aura besoin. Le système d'exploitation mobile est aussi un hôte bien plus agressif — il suspend l'application dès qu'un appel arrive, la redessine sous une encoche et offre au technicien un geste Retour système que FieldOps n'a jamais demandé.

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
shell hybride.NET MAUI	Une petite application native pour Android et iOS dont l'unique page héberge votre application Wisej.NET dans une web view, de sorte que les écrans C# que vous livrez déjà sont ce que voit le technicien.
hôte web view natif	Le moteur de navigateur de la plateforme intégré à cette page — Android System WebView, WKWebView sur iOS — qui exécute le même client qu'un navigateur desktop, ce qui explique qu'aucun écran ne soit réécrit pour le mobile.
bridge hybride	Le canal bidirectionnel entre la page hébergée et le code natif, qui permet au C# côté serveur de demander quelque chose à l'appareil et d'en recevoir la réponse sous forme d'événement plutôt que de navigation de page.
cycle de vie d'une application MAUI	Les événements que le système lève autour de l'application — fenêtre créée, activée, désactivée, arrêtée et reprise, plus OnSleep et OnResume — dont aucun n'est levé par un onglet de navigateur.
fenêtre de survie de la session	La durée passée en arrière-plan pendant laquelle la session Wisej.NET est encore active sur le serveur, de sorte qu'une reprise peut se reconnecter à l'état existant au lieu de redémarrer l'application.
marges de zone sûre	Le padding propre à chaque appareil qui éloigne le contenu de l'encoche, de la barre d'état et de la bande de l'indicateur d'accueil, indiqué par la plateforme plutôt que deviné à partir de la taille de l'écran.
gestion de la navigation Retour	La règle qui décide de l'effet du geste Retour d'Android: revenir en arrière dans la navigation propre à l'application, ou confirmer la sortie — jamais détruire en silence un ordre de travail à moitié rempli.

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 FieldOps 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 - « Shells mobiles avec .NET MAUI » - du cours LearnWisej « Packaging hybride et natif », 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/hybrid-native-packaging/specs/module-04.md>

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

1. Projet « FieldOps », 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 : Construisez le shell mobile de FieldOps avec .NET MAUI : une page hybride qui héberge l'application sur Android et iOS, une session qui survit au passage en arrière-plan et reprend au retour, le bouton Retour d'Android associé à la navigation dans l'application, des marges de zone sûre pour les appareils à encoche, et un build de débogage exécuté sur un émulateur par plateforme avec des captures d'écran de la liste des ordres de travail.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - Projet de shell hybride.NET MAUI pour Android et iOS
 - Gestion du cycle de vie pour l'arrière-plan et la reprise
 - Gestion du bouton Retour et des zones sûres
 - Builds sur émulateur pour les deux plateformes
 - Captures d'écran des deux émulateurs
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 :

- Un technicien est à mi-chemin d'une mise à jour de statut lorsqu'un appel de trois minutes arrive. Qu'est-ce qui doit être vrai sur le serveur et dans le shell pour que la mise à jour inachevée soit toujours là ensuite, et qu'est-ce qui change si l'appel dure une heure?
- Le shell ne sait rien de l'écran FieldOps ouvert. Comment un geste Retour décide-t-il entre revenir de la page de détail, fermer une boîte de dialogue et quitter l'application, et où cet état se trouve-t-il réellement?
- Lesquelles des modifications de ce module un technicien remarquerait-il encore s'il ouvrait FieldOps dans le navigateur du téléphone plutôt que dans le shell, et lesquelles disparaîtraient? Qu'est-ce que cela vous apprend sur l'endroit où placer le padding de zone sûre?

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 :

- ✓ Un technicien est à mi-chemin d'une mise à jour de statut lorsqu'un appel de trois minutes arrive. Qu'est-ce qui doit être vrai sur le serveur et dans le shell pour que la mise à jour inachevée soit toujours là ensuite, et qu'est-ce qui change si l'appel dure une heure?
- ✓ Le shell ne sait rien de l'écran FieldOps ouvert. Comment un geste Retour décide-t-il entre revenir de la page de détail, fermer une boîte de dialogue et quitter l'application, et où cet état se trouve-t-il réellement?
- ✓ Lesquelles des modifications de ce module un technicien remarquerait-il encore s'il ouvrait FieldOps dans le navigateur du téléphone plutôt que dans le shell, et lesquelles disparaîtraient? Qu'est-ce que cela vous apprend sur l'endroit où placer le padding de zone sûre?

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/hybrid-native-packaging/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."*
- ✓ **Relisez-le comme une pull request.** *"Fais-moi parcourir ce code FieldOps 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 FieldOps 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."*



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.