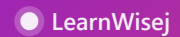


Exercice de code avec l'IA

Construisez l'exemple GlobalDesk 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

Menez un vrai cycle de traduction sur GlobalDesk. Ouvrez la solution dans ResX Resource Manager, ajoutez l'italien comme nouvelle colonne de langue, et servez-vous du filtre des entrées non traduites pour lister toutes les clés encore sans valeur. Ajoutez un commentaire aux trois clés dont le sens est ambigu hors contexte, en commençant par le libellé d'un seul mot Open, puis traduisez ou faites pré-traduire automatiquement les valeurs italiennes et marquez le nom du produit et les deux identifiants techniques comme invariants pour que personne ne les traduise. Exportez la colonne neutre et la colonne italienne, réimportez-les, et relisez le diff .resx obtenu avant de le valider comme une seule modification relisible. Terminez par une note d'une page comparant l'éditeur de Visual Studio, ResX Resource Manager et une plateforme cloud pour une équipe dont les traducteurs ne sont pas développeurs.

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

- ✓ Strings.it.resx produit avec ResX Resource Manager, sans aucune clé non traduite
- ✓ Commentaires de ressource sur au moins trois clés ambiguës pour donner le contexte au traducteur
- ✓ Marqueurs d'invariance sur le nom du produit et sur les identifiants qui ne doivent pas être traduits
- ✓ Export, réimport et diff.resx relu, validés comme une seule modification
- ✓ Note de flux de travail comparant l'éditeur de Visual Studio, ResX Resource Manager et un TMS cloud

Pourquoi il est conçu ainsi

GlobalDesk porte désormais un Resources/Strings.resx neutre, un Strings.de.resx allemand, des ressources de designer sur l'éditeur client et un Resources-de.resx pour les libellés système de Wisej. Ajouter l'italien avec l'éditeur de Visual Studio, c'est ouvrir ces fichiers un onglet à la fois et comparer les listes de clés à

l'œil nu, et la personne qui parle italien n'est en général pas celle qui sait ouvrir la solution. Ce module transforme la traduction en un flux de travail doté d'une liste de manques visible, d'une transmission et d'une étape de relecture.

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
Vue matricielle	Les clés en lignes et les langues en colonnes, si bien qu'une valeur italienne manquante et un terme allemand qui a dérivé se voient d'un seul coup d'œil.
ResX Resource Manager	Un outil de localisation.NET open source, à la fois extension Visual Studio et application autonome, qui présente les ressources RESX d'une solution dans une grille centrale unique.
Colonne de langue	Ajouter une langue cible crée le fichier spécifique à la culture à côté du fichier neutre et vous donne une colonne vide à remplir; les clés, elles, ne changent jamais.
Filtre des entrées non traduites	Ne liste que les entrées sans valeur dans la langue sélectionnée, si bien que « est-ce terminé? » devient un nombre au lieu d'une impression.
Commentaire de ressource	Une note stockée à côté de la valeur qui indique au traducteur si Open est un verbe, un adjectif, un état de ticket ou une commande.
Chaîne invariante	Un marqueur sur les entrées qui ne doivent jamais être traduites, comme le nom du produit et les identifiants techniques.
Aller-retour export / import	Des colonnes envoyées à un traducteur ou à une plateforme capable de lire le RESX et renvoyées sous forme de fichiers.resx spécifiques à une culture, que la compilation traite comme n'importe quelle autre ressource.

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 6. Vous pouvez ouvrir vous-même [module-06.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 GlobalDesk 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 6 - « Flux de traduction : ResX Resource Manager et autres outils » - du cours LearnWisej « Localisation 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/localization-in-wisej-net/specs/module-06.md>

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

1. Projet « GlobalDesk », 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 : Menez un vrai cycle de traduction sur GlobalDesk. Ouvrez la solution dans ResX Resource Manager, ajoutez l'italien comme nouvelle colonne de langue, et servez-vous du filtre des entrées non traduites pour lister toutes les clés encore sans valeur. Ajoutez un commentaire aux trois clés dont le sens est ambigu hors contexte, en commençant par le libellé d'un seul mot Open, puis traduisez ou faites pré-traduire automatiquement les valeurs italiennes et marquez le nom du produit et les deux identifiants techniques comme invariants pour que personne ne les traduise. Exportez la colonne neutre et la colonne italienne, réimportez-les, et relisez le diff .resx obtenu avant de le valider comme une seule modification relisible. Terminez par une note d'une page comparant l'éditeur de Visual Studio, ResX Resource Manager et une plateforme cloud pour une équipe dont les traducteurs ne sont pas développeurs.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - `Strings.it.resx` produit avec ResX Resource Manager, sans aucune clé non traduite
 - Commentaires de ressource sur au moins trois clés ambiguës pour donner le contexte au traducteur
 - Marqueurs d'invariance sur le nom du produit et sur les identifiants qui ne doivent pas être traduits
 - Export, réimport et diff.resx relu, validés comme une seule modification
 - Note de flux de travail comparant l'éditeur de Visual Studio, ResX Resource Manager et un TMS cloud
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 colonne italienne est pleine et l'application semble correcte en it-IT, mais un collègue affirme que deux clés n'ont jamais été traduites. Comment les deux affirmations

peuvent-elles être vraies en même temps, et quelle fonctionnalité de l'outil tranche le débat?

- Un brouillon automatique italien a renvoyé Ultimo ordine: {1} il {0} pour une clé dont la valeur neutre est Last order: {0} on {1}. Qu'est-ce qui casse, à quel moment, et qu'est-ce qui l'aurait détecté avant la validation?
- Votre équipe gère cinq langues, dont deux traduites par une agence qui n'a jamais ouvert Visual Studio. Lequel des trois flux de travail adopteriez-vous, et que faudrait-il changer avant de passer au cran supérieur?

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 colonne italienne est pleine et l'application semble correcte en it-IT, mais un collègue affirme que deux clés n'ont jamais été traduites. Comment les deux affirmations peuvent-elles être vraies en même temps, et quelle fonctionnalité de l'outil tranche le débat?
- ✓ Un brouillon automatique italien a renvoyé Ultimo ordine: {1} il {0} pour une clé dont la valeur neutre est Last order: {0} on {1}. Qu'est-ce qui casse, à quel moment, et qu'est-ce qui l'aurait détecté avant la validation?
- ✓ Votre équipe gère cinq langues, dont deux traduites par une agence qui n'a jamais ouvert Visual Studio. Lequel des trois flux de travail adopteriez-vous, et que faudrait-il changer avant de passer au cran supérieur?

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.

- ✓ **Ajoutez le deuxième cas.** *"Étends ce projet GlobalDesk 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/localization-in-wisej-net/specs/module-06.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 GlobalDesk 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."



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.