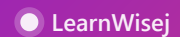


Exercice de code avec l'IA

Construisez l'exemple VisualOperationsStudio 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

Construisez le shell de VisualOperationsStudio et affichez une même mesure de télémétrie sur quatre surfaces à la fois. Créez une VisualOperationsPage qui détient une seule instance du modèle TelemetrySample, puis présentez sa valeur sous la forme (1) d'un simple Label accompagné d'une ProgressBar, (2) d'un Panel doté d'un gestionnaire Paint qui dessine une barre à partir de PaintEventArgs.Graphics et de e.ClipRectangle, (3) d'un Wisej.Web.Canvas dont le gestionnaire Redraw reconstruit la même barre depuis le modèle, et (4) d'un Bitmap hors écran rendu avec Graphics.FromImage et affiché dans une PictureBox. Câblez un bouton qui change la valeur du modèle et rafraîchit les quatre surfaces via Invalidate() et un redessin du Canvas, puis redimensionnez le navigateur et notez quelles surfaces survivent et lesquelles doivent être reconstruites.

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

- ✓ VisualOperationsPage détenant un unique modèle TelemetrySample que toutes les surfaces lisent
- ✓ Gestionnaire Paint dessinant depuis PaintEventArgs.Graphics à l'intérieur de e.ClipRectangle
- ✓ Gestionnaire Redraw du Canvas qui reconstruit toute la scène depuis le modèle après un redimensionnement
- ✓ Bitmap hors écran rendu avec Graphics.FromImage et affiché dans la page
- ✓ SurfaceNotes.md consignant ce qui transite sur le réseau pour chaque surface et quels visuels survivent à un redimensionnement

Pourquoi il est conçu ainsi

VisualOperationsStudio doit afficher la même mesure de machine à quatre endroits: un cadran sur la page d'exploitation, une barre de santé dans une ligne de grille, un nœud sur une carte de topologie, et un PNG

joint au rapport de poste du soir. Chacun se résume à « dessiner un nombre », et pourtant, se saisir de la première API de dessin dont on se souvient sur le bureau produit quatre morceaux de code aux comportements radicalement différents en CPU, en bande passante et au redimensionnement. Ce module place le choix de la surface avant le code de dessin, parce que c'est la surface qui décide où les pixels sont produits et de ce qui voyage jusqu'au navigateur.

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
Paint / PaintEventArgs	L'événement de dessin au niveau du contrôle: votre C# s'exécute sur le serveur sur le Graphics qu'il fournit, à l'intérieur du ClipRectangle qu'il indique, et Wisej.NET transforme le résultat en image pour le navigateur.
DataGridView.CellPaint	Le même mécanisme de dessin, limité à une cellule; son DataGridViewCellPaintEventArgs dérive de PaintEventArgs et ajoute RowIndex et ColumnIndex.
UserPaint	L'interrupteur qu'une cellule ou une colonne entière doit activer avant que la grille ne déclenche CellPaint pour elle, ce qui rend le dessin explicite colonne par colonne au lieu de l'imposer à toute la grille.
Wisej.Web.Canvas	Une surface 2D du navigateur en mode immédiat, qui reflète le modèle CanvasRenderingContext2D de HTML5; les appels de dessin sont exécutés par le client, et non rendus en une image sur le serveur.
Redraw	L'événement du Canvas dans lequel toute la scène est reconstruite depuis votre modèle, parce que le canvas conserve des pixels et les perd dès qu'il est recréé ou redimensionné.
System.Drawing.Managed	Le rendu hors écran, sans contrôle visible: créez un Bitmap, obtenez son Graphics par Graphics.FromImage(), et gardez les octets de l'image pour un rapport, un export ou un téléchargement.
Invalidate()	L'appel qui indique à un contrôle dessiné que son visuel est périmé; il met à jour le widget client ainsi que la surface du concepteur Visual Studio, sa place est donc là où l'état change, et non dans des helpers de dessin.

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 1. Vous pouvez ouvrir vous-même [module-01.md](#) pour voir exactement ce qu'il lira.



Utiliser ChatGPT



Utiliser Claude

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.

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 VisualOperationsStudio 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 1 – « L'architecture de dessin de Wisej.NET » – du cours LearnWisej « Dessin et peinture 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/drawing-and-painting/specs/module-01.md>

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

1. Projet « VisualOperationsStudio », 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 de VisualOperationsStudio et affichez une même mesure de télémétrie sur quatre surfaces à la fois. Créez une VisualOperationsPage qui détient une seule instance du modèle TelemetrySample, puis présentez sa valeur sous la forme (1) d'un simple Label accompagné d'une ProgressBar, (2) d'un Panel doté d'un gestionnaire Paint qui dessine une barre à partir de PaintEventArgs.Graphics et de e.ClipRectangle, (3) d'un Wisej.Web.Canvas dont le gestionnaire Redraw reconstruit la même barre depuis le modèle, et (4) d'un Bitmap hors écran rendu avec Graphics.FromImage et affiché dans une PictureBox. Câblez un bouton qui change la valeur du modèle et rafraîchit les quatre surfaces via Invalidate() et un redessin du Canvas, puis redimensionnez le navigateur et notez quelles surfaces survivent et lesquelles doivent être reconstruites.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
 - VisualOperationsPage détenant un unique modèle TelemetrySample que toutes les surfaces lisent
 - Gestionnaire Paint dessinant depuis PaintEventArgs.Graphics à l'intérieur de e.ClipRectangle
 - Gestionnaire Redraw du Canvas qui reconstruit toute la scène depuis le modèle après un redimensionnement
 - Bitmap hors écran rendu avec Graphics.FromImage et affiché dans la page
 - SurfaceNotes.md consignait ce qui transite sur le réseau pour chaque surface et quels visuels survivent à un redimensionnement
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 :

- Vous définissez la mesure du modèle depuis le bouton mais oubliez l'appel à Invalidate()

sur le panneau dessiné. Le libellé et le canvas se mettent à jour. Expliquez précisément pourquoi le panneau, lui, ne le fait pas, et ce que l'utilisateur regarde jusqu'à ce qu'autre chose force son redessin.

– Votre première version du Canvas dessine la barre une seule fois au chargement de la page au lieu de le faire depuis Redraw. Décrivez ce que voit l'utilisateur après avoir redimensionné le navigateur, et pourquoi le Bitmap hors écran de la quatrième zone se comporte différemment.

– Votre relecteur demande pourquoi la première zone — le simple Label et la ProgressBar — figure dans le lab. Répondez par les coûts que les trois autres surfaces paient et qu'elle ne paie pas, et nommez un besoin qui justifierait de les payer.

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 :

- ✓ Vous définissez la mesure du modèle depuis le bouton mais oubliez l'appel à Invalidate() sur le panneau dessiné. Le libellé et le canvas se mettent à jour. Expliquez précisément pourquoi le panneau, lui, ne le fait pas, et ce que l'utilisateur regarde jusqu'à ce qu'autre chose force son redessin.
- ✓ Votre première version du Canvas dessine la barre une seule fois au chargement de la page au lieu de le faire depuis Redraw. Décrivez ce que voit l'utilisateur après avoir redimensionné le navigateur, et pourquoi le Bitmap hors écran de la quatrième zone se comporte différemment.
- ✓ Votre relecteur demande pourquoi la première zone — le simple Label et la ProgressBar — figure dans le lab. Répondez par les coûts que les trois autres surfaces paient et qu'elle ne paie pas, et nommez un besoin qui justifierait de les payer.

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 VisualOperationsStudio 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/drawing-and-painting/specs/module-01.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 VisualOperationsStudio 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.