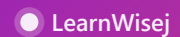


## Exercice de code avec l'IA

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.



### 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 VisualOperationsStudio en plaçant un seul modèle derrière les quatre surfaces. Restructurez le projet en couches Models, Geometry, Renderers et adaptateur Wisej pour que la jauge dessinée, les cellules de grille dessinées par votre code, la topologie Canvas et l'export PNG lisent tous le même OperationsModel et partagent la couche de géométrie pure, sans aucun état métier dupliqué par moteur de rendu. Ajoutez un helper RenderMetrics qui enregistre la durée de rendu, la taille du bitmap produit et le nombre d'objets de scène dessinés, et servez-vous-en pour comparer la jauge dessinée par le serveur au moteur de rendu hors écran, avant et après une optimisation que vous choisissiez et justifiez. Éprouvez les chemins d'échec — données vides, surface de taille nulle, plage invalide, police manquante — afin que chacun se dégrade en un contenu de remplacement lisible plutôt qu'en exception, ajoutez un tableau accessible qui liste toutes les valeurs montrées par les graphiques, et rédigez les notes de synthèse couvrant l'architecture, la frontière navigateur/serveur et la matrice de tests.

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

- ✓ Couches Models, Geometry, Renderers et adaptateur, avec un unique OperationsModel qui pilote les quatre surfaces
- ✓ Helper RenderMetrics enregistrant la durée de rendu, la taille du bitmap et le nombre d'objets dessinés par surface
- ✓ Mesures avant et après pour une optimisation, avec le raisonnement qui l'a fait choisir
- ✓ Chemins d'échec pour des données vides, une taille nulle, une plage invalide et une police manquante, qui se dégradent au lieu de lever une exception
- ✓ CapstoneNotes.md avec l'architecture, la frontière serveur/navigateur, la matrice de tests et un tableau de données accessible

## Pourquoi il est conçu ainsi

VisualOperationsStudio possède désormais quatre surfaces de rendu, et chacune a été construite dans son propre module avec sa propre idée privée de ce que signifie la santé d'une machine. C'est ainsi qu'un sous-système de dessin pourrit: quelqu'un déplace un seuil d'avertissement, trois surfaces s'accordent, et le PNG exporté contredit discrètement l'écran dont il provient. Ce module ramène ces quatre interprétations à un seul modèle derrière une seule couche de géométrie, puis prouve, par des mesures, des chemins d'échec délibérés et un équivalent sémantique, que le résultat survit à de vraies données, à de vraies plateformes et à de vrais utilisateurs.

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                                                                                                                                                                                          |
|---------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Modèle visuel             | De l'état sémantique uniquement: mesures, légendes, nœuds et arêtes, séries de tendance, gravité comme choix métier et sélection partagée, pour qu'aucune surface n'ait besoin d'une copie.                    |
| Couche de géométrie       | Des calculs presque purs qui transforment l'état du modèle et la taille courante de la surface en rectangles, angles de balayage, positions de libellé et régions de détection, testables sans aucun contrôle. |
| Adaptateur de surface     | Un moteur de rendu par surface plutôt qu'une interface commune: System.Drawing.Graphics et Wisej.Web.Canvas gardent leurs primitives propres au-dessus d'une géométrie partagée.                               |
| Budget de performance     | Un chiffre convenu avant tout réglage, par exemple un contrôle d'état dessiné sous 5 ms et un redessin complet du Canvas dans une fourchette d'environ 16 à 50 ms.                                             |
| Instrumentation du rendu  | Temps écoulé relevé, dimensions du bitmap, nombre d'objets dessinés et croissance de la charge utile par surface, pour qu'une optimisation se défende par un avant et un après.                                |
| Test d'image de référence | Un rendu hors écran à entrées fixes comparé à une référence stockée avec une tolérance, parce que l'anticrénelage et les encodeurs diffèrent d'une plateforme à l'autre.                                       |
| Équivalent sémantique     | Le libellé, l' AccessibleDescription, l'info-bulle ou la liste d'accompagnement qui portent chaque valeur essentielle montrée par les graphiques, pour que rien d'important n'existe uniquement en pixels.     |

## 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



Utiliser Claude

Ouvre [chatgpt.com](https://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](https://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 7 – « Architecture de dessin en production, performances et projet de synthèse » – 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-07.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 : Terminez le projet de synthèse VisualOperationsStudio en plaçant un seul modèle derrière les quatre surfaces. Restructurez le projet en couches Models, Geometry, Renderers et adaptateur Wisej pour que la jauge dessinée, les cellules de grille dessinées par votre code, la topologie Canvas et l'export PNG lisent tous le même OperationsModel et partagent la couche de géométrie pure, sans aucun état métier dupliqué par moteur de rendu. Ajoutez un helper RenderMetrics qui enregistre la durée de rendu, la taille du bitmap produit et le nombre d'objets de scène dessinés, et servez-vous-en pour comparer la jauge dessinée par le serveur au moteur de rendu hors écran, avant et après une optimisation que vous choisirez et justifiez. Éprouvez les chemins d'échec – données vides, surface de taille nulle, plage invalide, police manquante – afin que chacun se dégrade en un contenu de remplacement lisible plutôt qu'en exception, ajoutez un tableau accessible qui liste toutes les valeurs montrées par les graphiques, et rédigez les notes de synthèse couvrant l'architecture, la frontière navigateur/serveur et la matrice de tests.
3. Produis ces livrables et indique-moi où se trouve chacun d'eux dans la solution :
  - Couches Models, Geometry, Renderers et adaptateur, avec un unique OperationsModel qui pilote les quatre surfaces
  - Helper RenderMetrics enregistrant la durée de rendu, la taille du bitmap et le nombre d'objets dessinés par surface
  - Mesures avant et après pour une optimisation, avec le raisonnement qui l'a fait choisir
  - Chemins d'échec pour des données vides, une taille nulle, une plage invalide et une police manquante, qui se dégradent au lieu de lever une exception
  - CapstoneNotes.md avec l'architecture, la frontière serveur/navigateur, la matrice de tests et un tableau de données accessible
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 couche de géométrie prend le modèle et une taille et renvoie des formes, mais un coéquipier veut qu'elle renvoie un GraphicsPath pour raccourcir le moteur de rendu de la jauge. Que cela coûte-t-il au moteur de rendu Canvas et aux tests unitaires, et où placeriez-vous la limite à la place?
- Le PNG exporté et le Canvas à l'écran divergent de quelques pixels sur l'agent de compilation Linux, mais concordent sous Windows. Quelles mesures et quel type de test vous diraient s'il s'agit d'un défaut ou d'une différence de plateforme attendue?
- Un relecteur affirme que le tableau accessible est redondant puisque chaque jauge a déjà une info-bulle. Expliquez ce que le tableau apporte à un utilisateur au clavier seul ou avec lecteur d'écran que l'info-bulle n'apporte pas, et ce que cela implique pour les pastilles de gravité codées par la seule couleur dans la grille.

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 couche de géométrie prend le modèle et une taille et renvoie des formes, mais un coéquipier veut qu'elle renvoie un GraphicsPath pour raccourcir le moteur de rendu de la jauge. Que cela coûte-t-il au moteur de rendu Canvas et aux tests unitaires, et où placeriez-vous la limite à la place?
- ✓ Le PNG exporté et le Canvas à l'écran divergent de quelques pixels sur l'agent de compilation Linux, mais concordent sous Windows. Quelles mesures et quel type de test vous diraient s'il s'agit d'un défaut ou d'une différence de plateforme attendue?
- ✓ Un relecteur affirme que le tableau accessible est redondant puisque chaque jauge a déjà une info-bulle. Expliquez ce que le tableau apporte à un utilisateur au clavier seul ou avec lecteur d'écran que l'info-bulle n'apporte pas, et ce que cela implique pour les pastilles de gravité codées par la seule couleur dans la grille.

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/drawing-and-painting/specs/module-07.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."*
- ✓ **Remplacez une pièce sans déplacer les jointures.** *"Choisis la partie de ce projet VisualOperationsStudio 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.