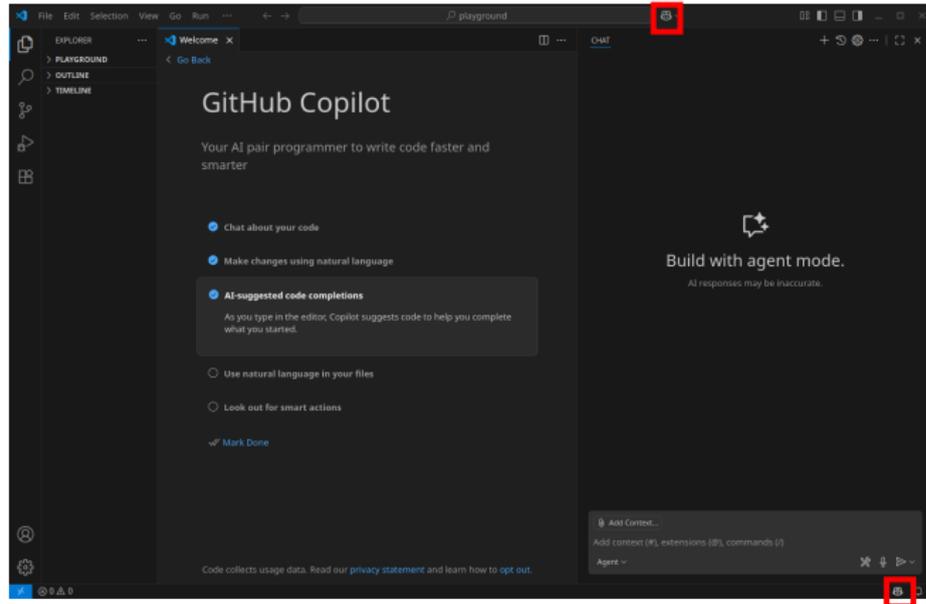


Expérimentations avec Copilot : une nouvelle forme de programmation ?

GT Innovation Pédagogique

D. Renault
ENSEIRB-Matmeca
23 septembre 2025

- LLMs disponibles pour échanger sur des thèmes variés, dont l'informatique (ChatGPT, Claude, Copilot, Gemini, Grok, Mistral ...)
- VSCode (février 2025) intègre directement Copilot dans son interface.



Copilot dans VsCode

- Copilot est un environnement facile d'accès (que ce soit pour universitaires et étudiants, ne demande qu'un compte Github)
- Il se présente comme un **assistant de code**, techniquement un LLM avec un accès à l'ensemble d'un dépôt de code, capable de :
 - ▶ proposer des complétions de code ;
 - ▶ répondre à des questions sur le code ;
 - ▶ réaliser des tâches de programmation complexes ;

Tout cela en plus des capacités classiques d'un LLM adapté à la programmation (vaste base de connaissances, pages de manuel ...)

- C'est une IA **agentique**, que les gens d'IBM qualifient comme :
"Agentic AI exhibits autonomy, goal-driven behavior and adaptability."
(cf. aussi la vidéo de MonsieurPhi sur les IAs "manipulatrices")

〈 Démonstration 〉

Différentes expérimentations dans trois contextes proches des enseignements de l'ENSEIRB-Matmeca :

- projets du S5 en C : implémentation d'une bibliothèque de Sudoku (avec algorithmes de résolution avancés)
- projets du S6 en Javascript : implémentation d'un type de données fonctionnels *finger tree*
- projets du S7 en Java : implémentation d'une bibliothèque en programmation orientée objet

- Ce qui a été expérimenté (avec le concours d'E. Fleury) :
 - ❶ la capacité à réaliser un projet en C (Makefile, compilation séparée, tests) ;
 - ❷ la capacité à faire cela purement avec des opérations de prompts ;
 - ❸ la capacité à implémenter des algorithmes de résolution non triviaux (preemptive sets, cross-hatching, iterative deepening . . .) ;
 - ❹ la capacité à généraliser le code (taille des sudokus).
- Conclusions :
 - ▶ le point ❶ est réalisé avec un degré de correction **dépassant** largement ceux des élèves de S5, tout en restant dans les limites du point ❷.
 - ▶ le mélange des points ❸ et ❹ fait que le LLM **s'emmêle les pinceaux**. Il faut le guider pour produire un code satisfaisant.
 - ▶ Des bugs non triviaux peuvent se retrouver au sein du code.
 - ▶ L'implémentation finale reste **inférieure en terme de performances** à l'implémentation de référence de E. Fleury.

- Ce qui a été expérimenté :
 - ❶ la capacité à s'intégrer à un projet "**fonctionnel**" en Javascript avec **existant** (package.json, Typescript, eslint, tests);
 - ❷ la capacité à faire cela purement avec des opérations de prompts;
 - ❸ la capacité à implémenter une structure de données non triviale;
- Conclusions :
 - ▶ le point ❶ est réalisé avec un degré de correction **très raisonnable**, tout en restant dans les limites du point ❷.
 - ▶ le point ❸ a aussi amené le LLM à **s'emmêler les pinceaux**. Il faut le corriger pour produire un code satisfaisant les fonctionnalités.

Extrait de conversation avec Copilot

```
"Perfect! Now I understand the code structure. Let me implement a functional finger tree.
A finger tree is a functional data structure that provides efficient access to both ends
and supports efficient concatenation and splitting."
"Now let me create the main finger tree implementation file (...) "
"Now let me create a comprehensive test file for the finger tree (...) "
"There's an issue with my concatenation implementation. Let me fix it (...) "
"Now let's try running the tests again (...) "
"Great! Almost all tests pass. There's just one failing test for concatenation.
Let me check what the issue is: "
"I see the issue! The concatenation is not working correctly.
Let me fix the concat function (...) "
"The issue is still there. Let me debug the pushBack function itself. (...) "
"I think I found the issue! When the suffix overflows in the 'pushBack' function (...)
But I think there's another issue."
"The issue is still there. Let me step back and make sure I understand the problem.
Let me add more debugging to see what happens during the fromArray construction (...)"
"Perfect! Now I can see the exact issue."
"Actually, let me take a step back and implement a much simpler approach.
The complex finger tree with nodes is quite intricate to get right.
Let me simplify this by avoiding the middle tree complexity for now and use
a simpler approach (...) "
```

Extrait de code produit par Copilot

```
function concat(left, right) {
  if (!left || isEmpty(left)) return right;
  if (!right || isEmpty(right)) return left;

  if (left.type === 'Single')
    return pushFront(right, left.value.value);
  if (right.type === 'Single')
    return pushBack(left, right.value.value);

  // Both are deep trees
  if (left.type === 'Deep' && right.type === 'Deep') {
    const middleElements = [...left.suffix, ...right.prefix];
    const newMiddle = addToMiddle(left.middle, middleElements,
                                   right.middle);
    return deep(left.prefix, newMiddle, right.suffix);
  }

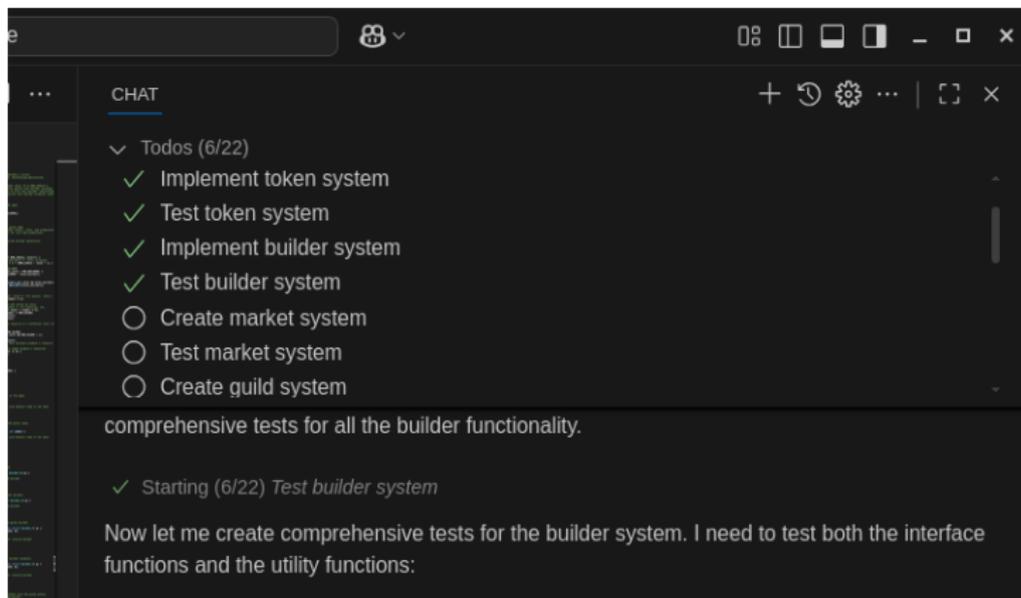
  // Fallback to array conversion for edge cases
  const leftArray = toArray(left);           // Transformation into array ???
  const rightArray = toArray(right);        // Again ???
  return fromArray(leftArray.concat(rightArray)); // Usage of array concat ???
}
```

- Ce qui a été expérimenté :
 - ❶ la capacité à réaliser un projet “**orienté-objet**” en Java (gradle, bibliothèques externes, tests) ;
 - ❷ la capacité à faire cela purement avec des opérations de prompts ;
 - ❸ la capacité à produire une architecture objet convenable ;
 - ❹ la capacité à faire des refactorings en Java (hors cours POO : sealed interface, algebraic data types, generics).
- Conclusions :
 - ▶ le point ❶ est réalisé avec un degré de correction **dépassant la moyenne** des élèves du S7, tout en restant dans les limites du point ❷.
 - ▶ le point ❸ a permis de construire une architecture objet équivalente à au corrigé (produit par moi-même) ;
 - ▶ le point ❹ est lui aussi largement **dans les capacités** du LLM.

- Ce qui a été expérimenté :
 - ❶ la capacité à travailler directement à partir d'une spécification :
<https://www.labri.fr/perso/reynald/working/teaching/projets/2023-24-S5-Munificence.php>
 - ❷ la capacité à organiser le développement du projet étape par étape ;
 - ❸ la capacité à committer le développement sous git.
 - ❹ la capacité à écrire un rapport (en suivant les consignes).

- Conclusions :
 - ▶ le développement se passe **sans problèmes** majeurs ;
 - ▶ en 39 prompts, dont 3 résolutions de bugs et 3 pour l'écriture du rapport, Copilot a produit 4600 lignes de code (dont la moitié de tests).
 - ▶ cela en 3h30 de travail, pour un projet planifié sur 8 semaines.

Planification du projet



Dépôt git final

● master	Add comprehensive French academic project report	David Renault	David Renault	#6f0b39ec836db5451...	3 minutes ago
●	Fix builder affordability consistency bug	David Renault	David Renault	6cf71a75388cf7d4ac...	An hour ago
●	Fix random seed override issue causing deterministic games	David Renault	David Renault	2edd6422be9d25c9f3...	An hour ago
●	Fix game forfeit bug in builder affordability check	David Renault	David Renault	024bb8d624edadadd...	2 hours ago
●	Fix compilation warnings and corrupted comment in guild system	David Renault	David Renault	03b437e10c73086d2...	2 hours ago
●	Complete Munificence game project delivery (Task 20)	David Renault	David Renault	bc50ba593151836df4...	2 hours ago
●	Complete final testing and validation (Task 19)	David Renault	David Renault	1801ce2f6fb9ed0057...	2 hours ago
●	Create main project executable (Task 18)	David Renault	David Renault	fd7ce768b6b521522c...	2 hours ago
●	Add comprehensive game system tests (Task 17)	David Renault	David Renault	d8f5a4c5dc8e381f6c...	2 hours ago
●	Implement game logic system (Task 16)	David Renault	David Renault	a4670cd8380a7fce95...	2 hours ago
●	Create game interface (Task 15)	David Renault	David Renault	4feada1171f88aae7a...	2 hours ago
●	Add player system with token/builder management	David Renault	David Renault	9b340f0244cb11203a...	2 hours ago
●	Implement comprehensive guild system tests (Task 10)	David Renault	David Renault	bf9df8a6f312d62f3f66...	2 hours ago
●	Implement guild system (Task 9)	David Renault	David Renault	7aafc386059909f50b...	2 hours ago
●	Update .gitignore for market test executable	David Renault	David Renault	be16a62833aacdd80...	3 hours ago
●	Implement comprehensive market system tests (Task 8)	David Renault	David Renault	f09d40ee69bf7828ff5f...	3 hours ago
●	Implement market system (Task 7)	David Renault	David Renault	9698301d3da220ed6...	3 hours ago
●	Add test executables to .gitignore	David Renault	David Renault	644c361913c9f26036...	3 hours ago
●	Implement comprehensive builder system tests (Task 6)	David Renault	David Renault	92878d8dce946e238...	3 hours ago
●	Implement builder system with static allocation (Task 5)	David Renault	David Renault	37451433b233476a4...	3 hours ago
●	Implement comprehensive token system tests (Task 4)	David Renault	David Renault	310ea5085d9a66249...	3 hours ago
●	Implement token system (Task 3)	David Renault	David Renault	6f93b39e02d35b380...	3 hours ago
●	Implement comprehensive color function tests (Task 2)	David Renault	David Renault	06581f110f1d230cc7...	3 hours ago
●	Implement color functions (Task 1)	David Renault	David Renault	a3c9bbf3897f27eaf74...	3 hours ago
●	Add comprehensive TODO.md with test-driven development approach	David Renault	David Renault	342c5ce5f960faa34e...	3 hours ago
●	Initial commit	David Renault	David Renault	0638903ec6e4a579c...	3 hours ago

Et après ? ...

- Une technique nouvelle de programmation modifiant radicalement le développement logiciel en général ...
- ... et l'apprentissage de la programmation en particulier.
- L'usage de Copilot transforme un développement "à petites marches" en un développement "à grandes marches".
 - ▶ le code trivial / boilerplate est écrit instantanément ;
 - ▶ la taille / complexité du code augmente plus rapidement ;
 - ▶ les bugs de développement deviennent plus subtils.
- La notion de **spécification** devient largement plus importante.
- L'**écriture** de code devient une activité secondaire.
La **lecture** de code devient une activité essentielle.