GitHub Copilot : comment utiliser l’IA pour mieux programmer?



L’intelligence artificielle transforme progressivement le travail des développeurs. Avec GitHub Copilot, il devient possible d’obtenir des suggestions de code, de mieux comprendre un programme existant et d’accélérer certaines tâches répétitives directement dans son environnement de développement.

GitHub Copilot ne remplace toutefois ni les compétences techniques ni la réflexion du développeur. Il agit plutôt comme un assistant qui propose des solutions à analyser, à tester et à améliorer.

Qu’est-ce que GitHub Copilot?

GitHub Copilot est un assistant de programmation alimenté par l’intelligence artificielle. Il s’intègre notamment à Visual Studio Code et à d’autres environnements de développement compatibles.

Selon le contexte du projet et les instructions fournies, il peut notamment :

- compléter une ligne ou un bloc de code;
- générer une fonction à partir d’un commentaire;
- expliquer le fonctionnement d’un programme;
- proposer une correction pour une erreur;
- générer des tests unitaires;

- améliorer ou simplifier du code existant;
- produire de la documentation technique.

Le développeur conserve la responsabilité de vérifier les résultats avant de les intégrer au projet.

Gagner du temps sans perdre le contrôle

GitHub Copilot peut accélérer la rédaction du code, particulièrement pour les structures courantes et les tâches répétitives.

Par exemple, un développeur peut décrire la fonction souhaitée dans un commentaire, puis examiner la proposition générée. Il peut l'accepter, la modifier ou la refuser.

Cette approche permet de consacrer davantage de temps à la logique métier, à l'architecture, à la qualité du code et à la résolution des problèmes plus complexes.

Utiliser GitHub Copilot Chat

Copilot Chat permet de dialoguer avec l'assistant en langage naturel. Il peut être utilisé pour :

- faire expliquer un fichier ou une fonction;
- comprendre une erreur;
- demander une amélioration;
- générer des commentaires;
- préparer un fichier README;
- proposer des tests;
- comparer différentes solutions techniques.

La qualité de la réponse dépend fortement du contexte et de la précision de la demande.

Comment rédiger de meilleurs prompts?

Un bon prompt doit expliquer clairement le résultat attendu. Il peut préciser :

- le langage de programmation;
- le contexte du projet;
- les données reçues par la fonction;
- le résultat attendu;
- les contraintes techniques;
- le style ou la structure souhaitée;
- les cas particuliers à traiter.

Au lieu de demander simplement « Crée une fonction », il est préférable d'indiquer son objectif, ses paramètres, son format de sortie et les erreurs qu'elle doit gérer.

Les bonnes pratiques à respecter

Le code généré par une intelligence artificielle peut contenir des erreurs, des failles de sécurité ou des choix inadaptés au projet.

Il est donc important de :

- relire systématiquement les suggestions;
- exécuter les tests nécessaires;
- vérifier la sécurité et les performances;
- respecter les normes du projet;
- ne pas inclure d’informations confidentielles dans les prompts;
- utiliser son jugement professionnel avant d’accepter une proposition.

GitHub Copilot doit être considéré comme un assistant, et non comme une source automatiquement fiable.

Apprendre GitHub Copilot avec Doussou Formation



La Formation GitHub Copilot : l'IA au service du code de Doussou Formation propose une approche pratique pour apprendre à intégrer cet assistant dans son travail quotidien.

La formation permet notamment de :

- installer et configurer GitHub Copilot;
- utiliser les suggestions dans Visual Studio Code;
- générer et améliorer du code;
- utiliser Copilot Chat;
- rédiger des prompts plus efficaces;
- analyser et corriger les propositions générées;
- adopter des pratiques responsables.

Consultez le programme de la formation GitHub Copilot et découvrez comment utiliser l'intelligence artificielle pour programmer plus efficacement tout en conservant la maîtrise de votre code.