Formation **Rust Perfectionnement**

Maîtrisez les fonctionnalités avancées de Rust avec notre formation 'Rust Perfectionnement'. Conçu pour les développeurs Rust de niveau intermédiaire, ce programme aborde des sujets complexes tels que la concurrence, la programmation asynchrone, pour le web, etc. Améliorez vos compétences, développez des applications performantes et préparez-vous à relever les défis de programmation Rust les plus exigeants.

Durée
4 jours

Objectifs pédagogiques

- Mettre en œuvre les concepts fondamentaux de Rust dans le cadre d'un projet spécifique
- Intégrer des concepts de programmation avancée tels que les constantes calculées, la réflexion, et la gestion de la mémoire non déplaçable
- Appliquer les principes de la programmation asynchrone à travers un cas pratique de développement d'application
- Concevoir des structures de programmation optimisées pour des applications embarquées
- Identifier et comparer les différentes bibliothèques graphiques disponibles
- Développer une interface graphique
- Créer une application web fonctionnelle

Public
Développeurs

Prérequis
Avoir suivi notre formation Rust ou disposer des connaissances équivalentes. Une expérience en développement Rust est également fortement conseillée.

Programme de formation

Introduction à la formation Rust Perfectionnement

Présentation générale de cette formation avancée sur le langage de programmation Rust
Révision des principes de base de Rust

Développement Rust avancé - Partie 1

Création et usage de macros
Interopérabilité avec C/C++ dans Rust
Implémentation de Rust dans des applications C/C++
Gestion du code non sécurisé en Rust (Unsafe Rust)

Développement Rust avancé - Partie 2

Utilisation de constantes calculées via des fonctions const
Manipulation de types avec Any et TypeId
Gestion de la mémoire fixe (techniques pin et unpin)

Techniques de programmation asynchrone

Gestion de la concurrence en programmation Rust
Implémentation des fonctions asynchrones et futures
Gestion de la communication entre futures
Utilisation de Stream et Runtime async

Techniques de programmation pour systèmes embarqués

Développement avec l'option no-std
Assurer la compatibilité des bibliothèques avec no-std
Programmation sans gestionnaire de mémoire
Conception du panic handler et des allocators
Méthodes de cross-compilation pour des systèmes embarqués
Implémentation asynchrone dans un contexte embarqué

Développement pour WebAssembly

Utilisation de Rust pour le backend
Intégration avec des serveurs et frameworks
Gestion des bases de données
Déploiement de wasm côté serveur
Implémentation de Rust dans le navigateur via WebAssembly
Interfaces de communication entre JavaScript et Rust
Programmation asynchrone et utilisation de WebGL

Moyens et méthodes pédagogiques

- La formation alterne entre présentations des concepts théoriques et mises en application à travers d'ateliers et exercices pratiques (hors formation de type séminaire).
- Les participants bénéficient des retours d'expérience terrains du formateur ou de la formatrice
- Un support de cours numérique est fourni aux stagiaires

Modalités d'évaluation

- **En amont de la session de formation**, un questionnaire d'auto-positionnement est remis aux participants, afin qu'ils situent leurs connaissances et compétences déjà acquises par rapport au thème de la formation.

- **En cours de formation**, l'évaluation se fait sous forme d'ateliers, exercices et travaux pratiques de validation, de retour d'observation et/ou de partage d'expérience, en cohérence avec les objectifs pédagogiques visés.

- **En fin de session**, le formateur évalue les compétences et connaissances acquises par les apprenants grâce à un questionnaire reprenant les mêmes éléments que l'auto-positionnement, permettant ainsi une analyse détaillée de leur progression.