# Rust, maitriser les fondamentaux du langage

Pour une programmation sûre et performante

Présentiel ou en classe à distance

3 jours (21 h)

Prix inter : 2.150,00 € HT
Forfait intra : 6.350,00 € HT

Réf.: LA400

Cette formation **Rust** est idéale pour les développeurs cherchant à maîtriser ce langage puissant, conçu pour la **sécurité** et la **performance** . Elle offre une couverture complète des fondamentaux, de la gestion de la mémoire à la modularité, en passant par des concepts avancés comme la **concurrence** et la **programmation non sécurisée** .
Les participants apprendront à structurer un code lisible et maintenable, tout en explorant des notions complexes comme la propriété, l'emprunt et les durées de vie. Avec une approche équilibrée entre théorie et pratique, cette formation permet aux développeurs d'acquérir des compétences solides pour **garantir la robustesse** et **l'efficacité de leur application** .

## A qui s'adresse cette formation ?

### Pour qui

- Développeurs applicatifs ou embarqués

### Prérequis

- Maîtrise de la programmation (C, C#, Java)
- Disposez-vous des connaissances nécessaires pour suivre cette formation ? Testez-vous !

## Programme

### 1 - Introduction à Rust

- Pourquoi Rust ? Sécurité, performance, et concurrence sans crainte
- Installation de Rust et outils associés (rustup, Cargo)
- Aperçu de la communauté et des ressources (crates.io, docs.rs, Rust Book)
- Structure d'un projet
  Atelier

  Installer Rust

  Configurer l'environnement de développement

  Créer un nouveau projet avec Cargo

### 2 - Fondamentaux de Rust

- Syntaxe de base : variables, types primitifs, mutabilité et inférence de types
- Expressions, fonctions et macros,
- Contrôle de flux (if, match, boucles)
- Ranges : utilisation de plages de valeurs inclusives et exclusives
- Déstructuration conditionnelle et itérative avec if let et while let
- Shadowing : redéfinition de variables avec le même nom
- Données constantes et statiques
  Atelier

Écrire un programme déclarant des variables immuables et mutables, en résolvant les erreurs

Utiliser des boucles for et while pour itérer sur des plages de valeurs

Manipuler des options avec if let et while let, et utiliser des données statiques

## 3 - Gestion de la mémoire et propriété

- Concepts de propriété (ownership) et de transfert (move)
- Emprunt (borrowing) et durée de vie (lifetime)
- Références immuables vs références mutables
- Analyse et compréhension des erreurs courantes
  Atelier

  Créer plusieurs fonctions qui transfèrent la propriété de variables

  Écrire des fonctions qui empruntent des références immuables et mutables, en résolvant les conflits de durée de vie

  Identifier et corriger des erreurs de gestion de mémoire

## 4 - Structures de données et collections

- Tuples, tableaux et la classe String
- Les slices
- Introduction aux Vec et leur manipulation
- Utilisation de HashMap pour les associations clé-valeur
- Itération et manipulation des éléments dans les collections
- Les énumérations et leurs variants
  Atelier

  Créer un programme manipulant des tuples, tableaux et chaînes de caractères

  Manipuler une Vec, ajouter, retirer et modifier des éléments

  Utiliser une HashMap pour stocker et récupérer des associations clé-valeur

## 5 - Gestion des Erreurs en Rust

- Différences entre panics et erreurs récupérables
- Propagation des erreurs avec l'opérateur ?
- Techniques de robustesse : le type Result, création de types d'erreurs personnalisés
  Atelier

  Créer une fonction échouant avec Result pour gérer les erreurs

  Utiliser l'opérateur ? pour propager des erreurs

  Implémenter un type d'erreur personnalisé pour une application simple

## 6 - Modularité avec les modules et les crates

- Organisation du code, séparation des responsabilités, visibilité (pub) et encapsulation
- Structure d'un crate : src/lib.rs, src/main.rs, modules internes, tests
- Utilisation des crates externes : gestion des dépendances avec Cargo

## 7 - Conception objet en Rust

- Eléments de programmation orientée objet en Rust : struct, impl, et traits
- Les constructeurs en Rust (new et autres méthodes associées)
- Ajout d'un destructeur avec le trait Drop
- Implémentation manuelle et dérivée des Traits
- Méthodes Mutables et Non Mutables
- Encapsulation et visibilité
- Traits génériques et implémentations
- Polymorphisme statique vs polymorphisme dynamique (dyn Trait)
- Composition vs héritage : pourquoi Rust favorise la composition
  Atelier

  Définir une struct et implémenter des méthodes avec impl

  Utiliser des traits pour partager des comportements entre structures

  Comparer l'utilisation de traits génériques et dynamiques, en illustrant la composition par des structs

### 8 - Concurrence

- Modèle de concurrence de Rust : threads, Send, Sync
- Partage de données entre threads : Mutex, RwLock
- Channels pour la communication entre threads
  Atelier

  Écrire un programme utilisant plusieurs threads avec partage de données via un Mutex

  Utiliser un RwLock pour gérer des lectures concurrentes

  Utiliser des channels pour permettre la communication entre threads

### 9 - Programmation Unsafe

- Introduction au bloc unsafe : pourquoi et quand l'utiliser
- Accéder directement à la mémoire avec les pointeurs bruts (*const T, *mut T)
- Erreurs communes dans les blocs unsafe et comment les éviter
  Atelier

  Écrire un programme manipulant directement des pointeurs bruts dans un bloc unsafe

  Identifier et corriger des erreurs communes liées à l'utilisation de blocs unsafe

### 10 - Tests et qualité du code

- Tests unitaires et d'intégration
- Macro #[test], cargo test

## Les objectifs de la formation

- Comprendre les fondamentaux de Microsoft 365 Copilot et Copilot Studio
- Créer et gérer des rubriques personnalisées
- Intégrer des données externes et des connecteurs
- Configurer des conversations personnalisées pour des scénarios prévisibles
- Gérer les entrées utilisateurs non reconnus et optimiser les rubriques

## Evaluation

- Pendant la formation, le formateur évalue la progression pédagogique des participants via des QCM, des mises en situation et des travaux pratiques. Les participants passent un test de positionnement avant et après la formation pour valider leurs compétences acquises.

## Les points forts de la formation

- Une formation qui présente les bases essentielles de Rust, avec un accent particulier sur la rédaction de code structuré, clair et conforme aux meilleures pratiques de développement
- Gestion avancée de la mémoire et de la sécurité : les participants apprendront à maîtriser les concepts de propriété, d'emprunt, et de durée de vie, ainsi qu'à gérer efficacement les erreurs
- Un équilibre entre théorie et pratique, alternant cours et ateliers, pour développer des applications concrètes en utilisant les bibliothèques et Framework populaires de Rust, avec des cas d'usage réels
- 77% des participants à cette formation se sont déclarés satisfaits ou très satisfaits au cours des 12 derniers mois.

## Dates et villes 2026 - Référence LA400

Dernières places disponibles Session garantie

### Strasbourg

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |

### Rouen

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |

### Marseille

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |

### Lille

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |

### Sophia Antipolis

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |

### Aix-en-Provence

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |

### A distance

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |
| du 13 avr. au 15 avr. | du 31 août au 2 sept. | du 30 nov. au 2 déc. |

## Toulouse

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |

## Paris

| | | |
|---|---|---|
| du 9 févr. au 11 févr. | du 22 juin au 24 juin | du 12 oct. au 14 oct. |
| du 13 avr. au 15 avr. | du 31 août au 2 sept. | du 30 nov. au 2 déc. |

## Nantes

| | | |
|---|---|---|
| du 13 avr. au 15 avr. | du 31 août au 2 sept. | du 30 nov. au 2 déc. |

## Bordeaux

| | | |
|---|---|---|
| du 13 avr. au 15 avr. | du 31 août au 2 sept. | du 30 nov. au 2 déc. |

## Rennes

| | | |
|---|---|---|
| du 13 avr. au 15 avr. | du 31 août au 2 sept. | du 30 nov. au 2 déc. |

## Lyon

| | | |
|---|---|---|
| du 13 avr. au 15 avr. | du 31 août au 2 sept. | du 30 nov. au 2 déc. |