Rust, programmation

Cours Pratique de 3 jours - 21h
Réf : RUX - Prix 2024 : 1 870CHF HT

Rust est un langage de programmation de la fondation Mozilla, multiparadigmes : objet, fonctionnel, générique, doté de bibliothèques modernes. Cette formation vous apprendra tout ce qu'il y a à savoir pour prendre le langage en main et le comprendre. Vous verrez également comment gérer ses fonctionnalités.

OBJECTIFS PÉDAGOGIQUES

À l'issue de la formation l'apprenant sera en mesure de :

Concevoir et lire des programmes en Rust

Programmer objet avec les structs et les traits

Utiliser les lambdas, l'inférence de type, le pattern matching

Maîtriser les cycles de vie des objets

Utiliser les API les plus courantes de Rust

MÉTHODES PÉDAGOGIQUES

Méthode pédagogique : stage pratique alternant 50 % de théorie et 50 % d'exercices.

LE PROGRAMME

dernière mise à jour : 01/2024

1) Introduction et outillage Rust

- Un langage sécurisé et performant : détection des erreurs à la compilation et panique contrôlée.
- Compilateur Rust.
- Cargo pour la gestion des projets et paquets.
- Documentation avec Rustdoc.
- Les environnements de développement intégré (IDE) pour Rust.

Travaux pratiques : Prise en main du langage et de son outillage.

2) Bases de Rust

- Les types de base de Rust. Variables mutables et immuables.
- Différents types de chaînes de caractères.
- Programmation procédurale et retour des fonctions implicites.
- Contrôle de flux et différents types de boucles, pattern matching.
- Programmation générique avec Rust.
- Expressions lambdas et captures.
- Les macros de Rust pour la métaprogrammation.

Travaux pratiques : Implémentation d'algorithmes procéduraux et génériques avec Rust.

3) Vers une programmation orientée objet

- Les "traits" standards les plus communs : Copy, Clone, Into, Drop…
- Visibilité, publicité, privauté à travers les modules et les crates.
- Agréger les données avec les structs et les tuples.
- Implémentation des structs.
- Enum pour codifier et structurer.
- Contrats fortement typés avec les traits et leur implémentation.

Travaux pratiques : Conception à base d'interface.

PARTICIPANTS

Développeurs, chefs de projets, architectes.

PRÉREQUIS

Bonne connaissance d'un langage de programmation, et connaissance des pointeurs/références.

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…
Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les stages pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque stage ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le stagiaire a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Vous avez un besoin spécifique d'accessibilité ? Contactez Mme FOSSE, référente handicap, à l'adresse suivante psh-accueil@orsys.fr pour étudier au mieux votre demande et sa faisabilité.

4) Cycle de vie des objets, gestion mémoire

- Cycle de vie des objets, portées, possession et emprunt des objets.
- Mettre un label sur les portées des objets pour les cycles de vie complexes.
- Compteurs de référence Rc<T> pour le partage des objets.
- Allocation dynamique avec Box<T> pour la liberté du cycle de vie.
- Pointeurs, blocs unsafes et pointeurs intelligents avec Unique<T>, Shared<T> pour le bas niveau.

Travaux pratiques : Implémentation d'arborescence et gestion de cycle de vie.

5) Bibliothèques de Rust

- Appeler Rust avec d'autres langages grâce à Foreign Function Interface (FFI).
- Les collections de Rust.
- Bibliothèques d'entrées sorties : lire et écrire dans les fichiers avec Rust.
- Multithreading avec Rust : synchroniser l'accès aux données.
- Appeler du code C avec Rust.

Travaux pratiques : Faire du multitâche avec Rust.

LES DATES

CLASSE À DISTANCE
2024 : 16 sept., 02 déc.