DAWAN

Plan de la formation

www.dawan.fr

Formation Rust

Durée :	3 jours (21 heures)
Tarifs inter-entreprise :	2 245,00 € HT (standard) 1 796,00 € HT (remisé)
Public :	Développeurs
Pré-requis :	Notions de programmation
Objectifs :	Connaître l'intérêt et l'utilisation du langage Rust - Etre capable de développer, compiler et tester une application en Rust
Modalités pédagogiques, techniques et d'encadrement :	• Formation synchrone en présentiel et distanciel. • Méthodologie basée sur l'Active Learning : 75 % de pratique minimum. • Un PC par participant en présentiel, possibilité de mettre à disposition en bureau à distance un PC et l'environnement adéquat. • Un formateur expert.
Modalités d'évaluation :	• Définition des besoins et attentes des apprenants en amont de la formation. • Auto-positionnement à l'entrée et la sortie de la formation. • Suivi continu par les formateurs durant les ateliers pratiques. • Évaluation à chaud de l'adéquation au besoin professionnel des apprenants le dernier jour de formation.
Sanction :	Attestation de fin de formation mentionnant le résultat des acquis
Référence :	PRO101839-F
Note de satisfaction des participants:	Pas de données disponibles
Contacts :	commercial@dawan.fr - 09 72 37 73 73

Modalités d’accès :	Possibilité de faire un devis en ligne (www.dawan.fr, moncompteformation.gouv.fr, maformation.fr, etc.) ou en appelant au standard.
Délais d'accès :	Variable selon le type de financement.
Accessibilité :	Si vous êtes en situation de handicap, nous sommes en mesure de vous accueillir, n'hésitez pas à nous contacter à referenthandicap@dawan.fr, nous étudierons ensemble vos besoins

Découvrir Rust

Rust : présentation du langage, styles supportés
Usages : programmation sysème, applications en ligne de commande, services réseaux,...
Apports de Rust par rapport au C# ou Java
Compilateur, packages d'installation
Détails du système de compilation Cargo
IDEs utilisables (rust-lang.org/tools), documentation (the book)

Atelier : Installation de l'environnement de développement (rustup, cargo, rustc, crates.io), structure d'une programme

Maîtriser les bases

Types de bases
Types référence
Tableaux et vecteurs
Manipulation de chaines de caractères
  Ownership
Structures de contrôles : conditions et boucles
Manipulation de références
Ecriture de fonctions et appels
Surcharge d'opérateurs

Atelier : Multiples fonctions et passage de paramètres

Gérer les erreurs

Gestion des erreurs, propagation
Capture des exceptions

Manipulation de types et écrire de type personnalisé

Atelier : Gestion des erreurs dans un programme

Utiliser des outils

Construction de profils (Crates)
Utilisation de Modules
Gestion des chemins
Construction de blocs (Items)
Documentation du code
Gestion des dépendances
Publication (crates.io)
Tests unitaires et tests d'intégration

Atelier : Manipulation de worspaces, gestion des dépendances, documentation et tests du code

Manipuler des objets et implémenter la généricité

Concepts de la POO supportés par Rust
Utilisation des Structures (struct)
Attributs et méthodes
Structures génériques
Mutabilité interne
Ecriture d'énumérations
Traits
Itérateurs sur des objets ou des chaînes
Utiliser des collections d'objets : Vect, VecDeque, LinkedList, BinaryHeap, HashMap

Atelier : Manipulation de plusieurs structures de données

Gérer les entrées-sorties

Lecture et écriture de fichiers et répertoires
Sérialisation d'objets
Compression de flux
Gestion des chemin
Fonctions spécifiques au système de fichiers
Mise en réseau

Atelier : Ecriture et lecture de plusieurs types de fichiers

Implémenter la concurrence

Bases de la programmation concurrente
Parallélisme et fonctions associées (fork-join, spawn & join)
Partage de ressources, variables globales atomiques
Pipeline, Channel
Verrous de synchronisation : Mutex, deadlock, ...
RwLock

Atelier : Implémentation de la concurrence avec Rust