Formation DevSecOps pour l'embarqué et l'IoT

Durée : 3 jours - 21 heures

Publics : Ce cours est destiné aux personnes ayant une première expérience de développement et désirant acquérir une maîtrise complète de la production de code de haute qualité afin de développer des architectures robustes pour faire face à des besoins système complexes

Prérequis : Connaissance de base du fonctionnement Unix ou Linux - Utilisation en ligne de commande de Linux.

Matériel requis : Un PC capable de faire fonctionner Docker Compose sera nécessaire pour les TP

Inscription en inter entreprise : 2 100 € HT/pers - (Les repas ne sont pas compris)

Objectifs :

Cette formation permet de maîtriser une chaîne d'intégration continue complète :

Comprendre l'intérêt d'investir dans la qualité de code

Connaître les différentes étapes depuis le développement de code jusqu'au déploiement

Maîtriser les outils permettant de contrôler et assurer la qualité et la sécurité du code

Déroulé

Le terme DevSecOps est une contraction de :

- Développeur
- Sécurité
- Opérations

C'est la prise en compte de la sécurité dans les processus DevOps.

En quelques années, de nombreuses organisations ont adopté le DevOps, autour notamment de la mise en place de processus d'intégration continue et de déploiement continu.

Shift Left : La stratégie DevSecOps consiste notamment à détecter les problèmes au plus tôt (à la gauche du cycle de développement). Ce qu'on appelle le "Shift Left".

Les problèmes détectés tardivement dans la phase de développement, voire dans un produit livré, sont bien plus coûteux à corriger, sans compter les coûts de mise à jour, de rappel éventuel.

Cette formation remet à plat les concepts de base de la qualité logicielle et des processus en continu : intégration, delivery, déploiement.

Les +++ de cette formation :

Cette formation a été conçue pour mettre en place rapidement les outils nécessaires, et pour comprendre les problèmes rencontrés au quotidien. Elle est ponctuée de nombreux TP qui mettent en avant les problèmes fréquents, les outils de mise au point et les pièges à éviter afin de permettre aux élèves de rapidement trouver leurs marques lorsqu'ils devront résoudre ces problèmes par eux-mêmes.

Les outils utilisés sont des exemples parmi les nombreux outils open source à disposition. Mais ce sont ceux auxquels Smile ECS fait appel pour livrer des logiciels de qualité.

Introduction

- Smile : a quick overview
- Course organization
- Setting up the training VM

Code

- Good Code vs Bad Code
- Coding Style
- Coding Rules
- Code Review
- Static Code Analysis
- Dynamic Code Analysis
- Complexity
- Unit testing
- Code Coverage
- Conclusion

Inspection Continue

- Technical Debt
    - Technical Debt
    - Definition Of Done
    - Conclusion
- Inspection
    - Setting up SonarQube Server
    - Setting up SonarQube configuration
    - Analyzing source code : First analysis
    - Analyzing source code : Going further
    - Understanding SonarQube analysis
    - Quality Gate
    - Fixing issues

Intégration Continue

- Introduction to GitLab-CI
    - What is CI/CD ?
    - CI/CD with GitLab and external tools
    - GitLab-CI : GitLab's integrated CI/CD
- GitLab-CI for the source code developer
    - Viewing pipeline results
    - Forcing or skipping pipeline execution
- GitLab Runners
- GitLab-CI for the pipeline developer
    - Introduction to YAML

    - Defining jobs
    - Saving job artifacts
    - Organizing jobs into a pipeline
    - Defining conditions on pipeline execution
    - Organizing YML files
    - Defining a system of pipelines
- GitLab-CI for the job developer
    - Choosing your runner
    - Handling errors
    - Caching job data

Let's build your own quality pipeline

- Your project
- Defining your pipeline architecture
- Let's go further