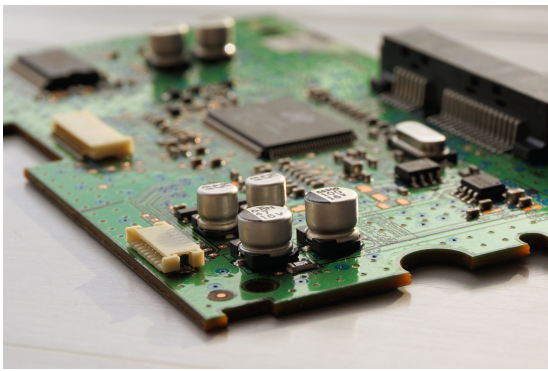
FICHE DE FORMATION

Produire du code C fiable et industriel

POUR UN FIRMWARE PLUS FIABLE



Cette formation s'adresse à tous les développeurs dans le domaine de l'embarqué désireux de devenir beaucoup plus productif dans le développement et la maintenances de firmwares écrits en C, avec un système d'exploitation ou non (bare metal)

Comment reproduire un bug ? Quelle norme de programmation choisir ? Comment vérifier automatiquement mon code ? Et comment faire tout cela avec les moyens limités qu'offre l'entreprise/startup dans laquelle je travaille ?

Outre son orientation technique et pragmatique, cette formation aborde également la relation client, l'organisation projet et la spécification. Autant d'éléments permettant de s'assurer qu'un projet dispose d'un début, d'un milieu … et d'une fin !

La formation est organisée autour de points théoriques illustrés par des exercices pratiques sur machine virtuelle.

NIVEAU REQUIS

Développeur pratiquant le C au quotidien dans le domaine de l'embarqué. Des notions en ligne de commande Linux bien que non indispensables sont conseillées.

FORMATEUR

Fondateur de la société Digitam, Laurent Meyer est spécialisé dans les objets connectés et les systèmes embarqués. Ingénieur de formation, il dispose de 15 années d'expérience passées tant auprès de grands groupes qu'auprès de PME et de startups. Il assiste également les dirigeants, les DSI et les responsables projets sur la manière de gérer, manager et mener à bien leurs projets informatiques et électroniques.

Plan de formation

PREMIERE JOURNEE

- Introduction : Exemples de scénarios types introduisant la problématique d'un code de qualité, testé et documenté.
- Norme de codage : Principe et exemples, Linux coding style, MISRA, google coding style.
- Détection automatique d'erreur : Les outils statiques, les warnings du compilateur, splint, cppcheck
- Les outils dynamiques : Principe et mise en œuvre des outils, Valgrind
- Documentation du code
- Travaux pratiques sur machine virtuelle
- Analyse de code existant avec cppcheck, splint
- Exercices sur les warnings

DEUXIEME JOURNEE

- Outils de gestion de configuration : Principes et exemples : CVS, SVN, GiT
- Les tests : Principe, Test de non-régression, Automatisation des tests, testabilité.
- Gestion de tickets de modification / défauts : Principes, Trac, Gitlab
- Interaction entre TRAC et SVN pour la revue de code.
  • Travaux pratiques :
  o Utilisation SVN basique
  o Navigation dans le code source avec SVN et TRAC
  o Valgrind
  o Rendre une application testable

TROISIEME JOURNEE

- Chapitre introductif : problématique de la testabilité, idées reçues.
- Les tests unitaires automatisés avec unity
- Les tests d'intégration automatisés avec CMock
- Les tests de validation automatisés Sélénium et testfarm
- Les tests de robustesse, d'endurance, performance
- La testabilité logicielle: rendre son code testable
- Les métriques : couverture de code, complexité, mise en œuvre du calcul de la complexité
- Mise en œuvre d'un plan de test, exemple
- Étude de cas: serious gaming
- Conclusions / Discussions.

MOYEN DE SUIVI DE L'EXÉCUTION DE LA FORMATION

Une attestation de fin de formation est remise à chaque stagiaire.

MOYENS PÉDAGOGIQUES

Seront fournis pour cette formation:
- Une machine virtuelle contenant les exercices à réaliser
- Un support de présentation
- Les exercices corrigés

CONDITIONS GÉNÉRALES

- Prix : Le prix est garanti pour la période indiquée sur le site
- Annulation : Toute inscription non annulée 8 jours avant la date de début sera considérée comme définitive et donnera lieu à une facturation intégrale de la formation.
- Horaires : De 9h00 à 12h30 et de 13h30 à 17h.