Le langage C++ pour les systèmes embarqués

Objectifs

- Maîtriser les bases du langage C++
- Intégrer les templates C++ (code générique) dans les systèmes embarqués
- Maîtriser les aspects avancés du C++ tels que le polymorphisme, l'héritage simple et l'héritage multiple
- Redéfinir les opérateurs C++ d'allocation dynamique de mémoire pour l'embarqué
- Rendre les objets C++ persistants flashables et romables
- Gérer les exceptions en C++ pour sécuriser les applications embarquées
- Utiliser des objets C++ pour gérer la transmission/réception série de chaînes de caractères

Matériel

- Matériel de cours imprimé
- Un PC par binôme
- Carte STM32F4 - ARM Cortex M4
- Environnement de développement Eclipse et compilateur GCC

Pré-requis

- Connaissance du langage C (niveau cours L2 - C language for Embedded MCUs)

Environnement du cours

- Cours théorique
  - Support de cours imprimé et au format PDF (en anglais).
  - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique.
- Activités pratiques
  - Les activités pratiques représentent de 40% à 50% de la durée du cours.
  - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
  - Exemples de code, exercices et solutions
  - Un PC (Linux ou Windows) par binôme de stagiaires (si plus de 6 stagiaires) pour les activités pratiques avec, si approprié, une carte cible embarquée.
  - Le formateur accède aux PC des stagiaires pour l'assistance technique et pédagogique.
- Une machine virtuelle préconfigurée téléchargeable pour refaire les activités pratiques après le cours
- Au début de chaque demi-journée une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

Modalités d'évaluation

- Les prérequis indiqués ci-dessus sont évalués avant la formation par l'encadrement technique du stagiaire dans son entreprise, ou par le stagiaire lui-même dans le cas exceptionnel d'un stagiaire individuel.

- Les progrès des stagiaires sont évalués de deux façons différentes, suivant le cours:
  - Pour les cours se prêtant à des exercices pratiques, les résultats des exercices sont vérifiés par le formateur, qui aide si nécessaire les stagiaires à les réaliser en apportant des précisions supplémentaires.
  - Des quizz sont proposés en fin des sections ne comportant pas d'exercices pratiques pour vérifier que les stagiaires ont assimilé les points présentés
- En fin de formation, chaque stagiaire reçoit une attestation et un certificat attestant qu'il a suivi le cours avec succès.
  - En cas de problème dû à un manque de prérequis de la part du stagiaire, constaté lors de la formation, une formation différente ou complémentaire lui est proposée, en général pour conforter ses prérequis, en accord avec son responsable en entreprise le cas échéant.

Plan

Premier jour

Introduction au C++ pour l'industrie

- Introduction à la programmation orientée objet
- Historique et définition
- Aperçu des standards C++
- Objectifs de C++ Moderne
- Passage du C vers le C++
- Spécification Embedded C++ (EC++)
- Comment écrire du code optimisé pour l'embarqué

Exercise : *Comprendre l'encodage des noms de fonctions (function mangling)*
Exercise : *Fonction inline*
Exercise : *Gestion des variables volatiles*

Le C++ et l'embarqué

- Programmation objet en C++
  - Encapsulation
  - Classes et objets
  - Attributs et fonctions membres
  - Construction et destruction d'objets
  - Paramètres de construction
  - Paramètres de construction
  - Composition et la containérisation d'objets
  - Visibilité et portée des déclarations

Exercise : *Déclaration de classes et de méthodes*
Exercise : *Les constructeurs par défaut, de copie et paramètrisés*
Exercise : *Comprendre les différences entre composition et agrégation*

Second jour

Le C++ et l'embarqué

- Surcharge des opérateurs
  - Optimisation des passages d'objets en paramètres
  - Surcharge des opérateurs par des fonctions membres
  - Surcharge des opérateurs par des fonctions « friend »
  - Surcharge des opérateurs de gestion mémoire

Exercise : *L'opérateur d'affectation*
Exercise : *Surcharge des opérateurs*

- Héritage simple
  - Spécialisation par addition et substitution
  - Règles de dérivation et d'accès

  - Construction pendant l'héritage
  - Le polymorphisme
  - Méthodes Virtuelles

Exercise : *Comprendre l'héritage*

- Objets ROMables et persistants
  - Objets constants et partiellement constants
  - Objets persistants
  - Objets ROMables

Exercise : *Création d'objets constants, mutables, persistants et ROMables*

- Renforcement de la sécurité avec des exceptions
  - Lancement, capture et traitement d'exceptions
  - Redéclenchement d'exception
  - Spécification d'exceptions
  - Traitement d'exceptions inattendues
  - Objets exceptions de la librairie standard C++

Exercise : *Gestion des erreurs avec des exceptions*

Exercise : *Gestion d'exceptions inattendues*

Troisième jour

Les techniques avancées du C++

- Le flux d'E/S
  - Flux standard du langage C++
  - Flux standard des librairies C++ standard
  - Technique de redirection des flux d'E/S standard par amitié

Exercise : *Redéfinition des opérateurs ‘ >’ lire/écrire des objets depuis un flot d'entrées-sorties*

- Pointeurs membres
- Objets génériques et templates
  - Classes et fonctions génériques
  - Surcharge de templates
  - Spécialisation de templates
  - STL (Standard Template Library)
  - Utilisation de templates dans l'embarqué

Exercise : *Classes et fonctions génériques*

- Objets polymorphes
- Objets virtuels et classes abstraites
- Spécialisation des objets par héritage simple
  - Construction d'objets dérivés
  - Règle de contrôle d'accès des objets hérités
  - Spécialisation des objets par Héritage multiple
  - Résolution des conflits par opérateur de résolution de portée
  - Intérêt de l'héritage virtuel

Exercise : *Comprendre l'utilisation des méthodes virtuelles en dérivant une classe Device générique*

Exercise : *Comprendre l'héritage multiple et les bases virtuelles*

Eléments de la chaine de compilation

- Utilisation de la compilation croisée
- Objectif du compilateur, de l’assembleur et de l'éditeur de liens
- Consulter les différentes sections du fichier objet
- Fichier de démarrage « Startup »
- Options du compilateur GCC
- Configuration de l’éditeur de liens
- Makefile

Renseignements pratiques

Renseignements : 3 jours