L10 - Programmation C++ moderne embarquée

Le langage C++ moderne pour les systèmes embarqués

Objectifs

- Découvrir les fonctionnalités modernes du C++
- Apprendre les changements de langage dans C++11, C++14, C++17 and C++20
- Découvrir les nouvelles fonctionnalités ajoutées à la bibliothèque standard
- Apprendre les fonctionnalités avancées du C++ moderne comme "perfect forwarding"
- Passer du C++ traditionnel au C++ moderne
- Mettre en évidence les fonctionnalités essentielles du C++ moderne utilisées dans les applications embarquées

Les exercices sont conduits sur des cartes à base de microcontrôleurs STM32

Equipement

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

Pré-requis

- Connaissances en programmation C (voir notre cours oL2 - Langage C pour les MCUs embarqués)
- Connaissances de base du langage C++ (voir notre cours oL3 - Programmation C++ embarqué)

Audience visée

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

Durée

- Totale : 12 heures
- 2 sessions de 6 heures chacune (hors temps de pause)
- De 40% à 50% du temps de formation est consacré aux activités pratiques
- Certains travaux pratiques peuvent être réalisés entre les sessions et sont vérifiés par le formateur lors de la session suivante

Environnement du cours

- Cours théorique
  - Support imprimé et PDF (en anglais).
  - Assistance du formateur tout au long de la formation.
- Chaque session débute par un point avec les stagiaires.

Plan du cours

Premier Jour

Introduction au C++ moderne

- Présentation
- Spécificateurs de classe de stockage
- Initialisation uniforme
- Déduction automatique de type
  - Le mot-clé auto
  - Le mot-clé auto comme type de retour d'une fonction
  - Utiliser auto pour déclarer des signatures de fonctions
  - Références constantes automatiques
  - Forwarding references
  - Avantages de auto dans les systèmes embarqués

Mots-clés

- enum class
- override et final
- Variables inline
- nullptr
- static_assert
- noexcept
- constexpr et if constexpr
- decltype
- Fonctions defaulted et deleted

Nouvelles fonctionnalités

- Boucles for basées sur un intervalle
- Namespaces imbriqués et alias de namespace
- Structured binding
- Move semantics et références r-value
  - Copy-constructing et Move-constructing
  - Références r-value
  - Perfect forwarding
- Alignement
  - Alignas
  - Alignof

Bibliothèque standard du C++ moderne

- Bibliothèque standard
  - std::optional
  - std::variant
  - std::any
  - std::byte
  - std::hash
  - Bibliothèque Filesystem
- Littéraux
  - Cooked literals
  - Opérateurs de littéraux standard

    - Raw literals
    - Raw string literals
- Génération de nombres aléatoires
    - Moteurs de génération de nombres aléatoires
    - Distributions de nombres aléatoires
- Conteneurs
    - std::array
    - std::forward_list
    - Conteneurs associatifs non ordonnés

Second Jour

Manipulation de chaînes

- Nouveaux types de chaînes
    - std::u16string
    - std::u32string
- Basic string view
- Conversions entre types numériques et chaînes
- Conversions élémentaires de chaînes
- Manipulateurs d'entrée/sortie
    - std::get_money, std::put_money
    - std::get_time, std::put_time
    - std::quoted
- Expressions régulières
    - Format d'une chaîne
    - Analyser le contenu d'une chaîne
    - Remplacer le contenu d'une chaîne

Fonctions lambda

- Syntaxe des lambdas
- Définir des lambdas
- Utiliser des lambdas
    - Utiliser des lambdas avec les algorithmes standard
    - Affecter des lambdas à des pointeurs de fonction
    - Lambdas et std::function
    - Écrire une fonction acceptant une lambda en paramètre
- Lambdas polymorphes
- Lambdas récursives

Gestion dynamique de la mémoire

- Smart Pointers
    - Raw Pointers
    - Pointeurs automatiques
    - Unique Pointers
    - Shared Pointers
    - Weak Pointers

Concurrence et multithreading

- Introduction
- Thread
- Mettre en œuvre un singleton thread-safe
- Opérations atomiques
    - Fonctionnalités atomiques

    - Fonctions non membres
    - Atomic flag
    - Memory order
- Mutex
    - Éviter les mutex récursifs
- Envoyer des notifications entre threads
- Variables de condition
- Future et Promise
- Task et Async
- Le C++ moderne et les RTOS