

Formation – Passage de C++11 à C++17

Durée : 5 jours

Objectifs

- Comprendre les évolutions des standards C++11, C++14 et C++17.
- Exploiter la programmation fonctionnelle.
- Développer des classes modèles efficacement.
- Maîtriser la gestion de la mémoire avec les pointeurs intelligents
- Exploiter la bibliothèque standard STL.
- Programmation multi-threadé

A qui s'adresse cette formation ?

- Chefs de projet, développeurs

Prérequis

- Connaissance en C++ et de la programmation OO

Programme de la formation

Historique et évolution du langage C++

- Historique
- Évolution de C++
- Compilation et compilateurs
- Modèle mémoire d'un programme exécutable

Littéraux

- Littéraux de caractère
- Les chaînes littérales brutes
- Littéraux entiers
- Littéraux définis par l'utilisateur

Les espaces de noms

- Espaces de nommage imbriqués (nested namespaces)
- Espaces de noms inline
- Alias d'espaces de noms
- Espaces de nommage anonymes

Les nouveautés du langage C++11/C++17

- La constante nullptr
- Les expressions constexpr
- Le range-based for loop
- Initialisation uniforme
- Initialisation dans les instructions if et switch
- Liaison structurée

Références et expressions

- Catégories d'expressions
- Notion de value-ness
- Inférence de type

Les attributs

- Attributs de C++11
- Attributs de C++14
- Attributs de C++17

Les nouveautés au niveau des classes en C++11/C++17

- Les directives =default, et =delete
- Héritage et délégation des constructeurs
- Overriding
- Les méthodes et les classes final
- Copie et déplacement
- Modèle de classe std::initializer_list
- Suppression de copie garantie en C++17

Les conversions et RTTI

- Opérateurs de conversion. Constructions implicites, le mot-clé explicit.
- Les opérateurs de casting const_cast, static_cast, reinterpret_cast.
- Conversion dynamique et Runtime Type Information.
- Les exceptions structurées
- La classe type_info.
- Contrôle du "downcasting" à l'aide de l'opérateur dynamic_cast.

Programmation fonctionnelle en C++11/C++17

- Pointeurs de fonctions
- Foncteurs
- Les expressions lambda
- std::function

Les templates avec C++11/C++17

- Introduction aux patrons de classe. Généricité et préprocesseur.
- Fonction générique. Classe générique. Composition générique.
- Généralisation générique.
- Spécialisation partielle et totale.
- Introduction à la méta-programmation.

Nouveautés dans la STL

- std::optional, std::variant et std::any
- Les itérateurs
- Les nouveaux conteneurs

Nouveaux algorithmes de la bibliothèque STL

Pointeurs intelligents

- L'idiom RAII
- std::unique_ptr
- std::shared_ptr

- `std::weak_ptr`

Gestion des erreurs et des exceptions

- Gestion des erreurs par valeur de retour
- Assertion non statiques et statiques
- Gestion des exceptions

Le namespace chrono

- Les classes `duration` et `time_point`
- Les fonctions de conversion

Gestion des threads

- Création et terminaison des threads
- Partage de données entre les threads
- Synchronisation des opérations concurrentes
- Les types atomiques