Programmation Orientée Objet en C++ Avancé

Description

La Formation Programmation Orientée Objet en C++ avancé s'adresse aux développeurs souhaitant maîtriser pleinement les techniques avancées du langage. Ce cours vous permettra de renforcer vos bases, tout en découvrant des méthodes modernes pour concevoir des applications performantes et maintenables. Vous apprendrez à exploiter la puissance de la POO, des modèles et de la bibliothèque standard STL, tout en appliquant des bonnes pratiques reconnues dans l'industrie.

Développez vos compétences en C++ avancé

Grâce à cette formation Programmation Orientée Objet en C++, vous développerez votre capacité à structurer efficacement votre code et à optimiser son exécution. Vous comprendrez comment gérer la mémoire, utiliser les pointeurs intelligents, manipuler les exceptions et appliquer des patterns de conception éprouvés. Les différentes sections du programme vous guideront dans l'application de concepts avancés tels que l'héritage multiple, la délégation, le découplage et les heuristiques de conception.

Contenu du cours
Module 1 : Historique du langage et C++ 11/14

- Histoire de C++
- Versions
- Nouveau dans TR1
- Nouveau en C++ 11
- Nouveau en C++ 14

Module 2 : Examen de la langue et bonne pratique: Partie I

- Programmation orientée objet
- Constructeurs et destructeurs
- New et delete

Module 3 : Héritage de base

- Interface et implémentation
- Héritage de Type
- Héritage d'implémentation
- Utilisation correcte de C++ 11 final et override
- Destructeurs virtuels: quand et pourquoi?
- Directives d'héritage

Module 4 : Utilisation correcte des fonctionnalités linguistiques bien connues

- Pointeurs vs. Références vs Valeurs
- Bonne utilisation de const
- Utilisation appropriée des fonctions en ligne
- Bonne utilisation de static
- Utilisation appropriée des paramètres par défaut

- Bonne utilisation de friend
- Utilisation appropriée du namespace
- Le moyen C++ du cast
- Utilisation appropriée de la surcharge de l'opérateur
- Constructeur de copie: pourquoi / quand?
- Opérateur d'affectation: pourquoi / quand?
- La loi des trois grands

Module 5 : Exceptions

- Gestion traditionnelle des erreurs
- Traitement d'erreur orienté objet
- throw, try and catch
- Conception de hiérarchies d'exceptions
- Utilisation appropriée de rethrow
- Utiliser unexpected
- Pièges des exceptions
- Bonnes pratiques sur les exceptions

Module 6 : Modèles

- Définition des classes de template
- Implémentation des classes de modèles
- Classes paramétrées
- Modèles et paramètres non-types
- Directives de template
- Modèles et fonctions simples
- Fonction de template C++

Module 7 : Bibliothèque de modèles standard (STL)

- Chaîne de caractères STL
- Composants STL
- Conteneurs de séquence
- Utilisation d'itérateurs en STL
- Exemple d'algorithmes
- Initialisation des conteneurs
- Profils de performance des conteneurs de séquences

Module 8 : Algorithmes STL

- STL contre Boost
- Paramétrage des algorithmes
- Utiliser des fonctions
- Utilisation d'objets de fonction
- Utilisation d'expressions Lambda
- Bibliothèque d'algorithmes sélectionnés
- Algorithmes contributifs

Module 9 : Conteneurs associatifs STL

- Set
- Multiset

- Map
- Multimaps

Module 10 : Functeurs STL, Allocators et plus

- Exception standard
- Functors
- Objets de fonction fournis par la bibliothèque
- Utilisation d'objets de fonction STL et de liaisons
- Négateurs
- Allocateurs
- Nombre complexe
- Smart Pointers

Module 11 : Efficacité : Objets temporaires

- Objets temporaires : le problème
- Diverses techniques pour éviter les temporaires
- Chaîne STL : comment éviter la création de temporaires
- C++ 11 : Déplacer la sémantique
- Techniques diverses pour éviter les temporaires

Module 12 : Gestion de la mémoire

- Comment C++ utilise-t-il la mémoire?
- Directives de base
- Implémentation de singletons en C++
- Utilisation efficace des pointeurs intelligents
- Surcharge new et delete
- Gestion de la mémoire
- L'importance de la disposition des données

Module 13 : Mémoire vive / froide

- Techniques d'efficacité diverses
- Préoccupations de conception
- Flexibilité vs performance
- Évaluation lazy
- Évaluation eager
- Copier sur les techniques d'écriture
- Mise en page des données revisitée
- Matériel moderne et pipelines de cache
- L'effet des structures de données et des algorithmes
- Postcondition et C++
- Profils d'efficacité des bibliothèques
- STL et Performance
- Latence
- Coût et avantages des threads
- Programmation asynchrone
- Futures

Module 14 : Délégation

- Concept de délégation
- Délégation en C++
- Délégation simple
- Délégation statique
- Délégation de super classe
- Délégation de sous-classe
- Problèmes avec la délégation de sous-classe en C++
- Délégation d'objet
- Modèle de stratégie
- C++ et Stratégie
- Modèle d'état
- C++ et état
- Conception de composite
- Composite et délégation
- Autres modèles de délégation

Module 15 : Découplage

- Qu'est-ce que le couplage?
- Types de couplage
- Couplage d'identité
- Couplage d'identité: cycle de vie de l'objet
- Changement d'identité
- Type de couplage
- Couplage d'implémentation
- Interfaces et implémentations
- Découplage par l'exemple

Module 16 : Héritage avancé

- Héritage multiple des interfaces
- Héritage multiple de l'implémentation
- Propriétés partagées
- Résoudre l'ambiguïté
- Héritage virtuel
- Multi méthodes
- Double Dispatch
- Utilisation de RTTI
- Règles et lignes directrices
- Héritage des méthodes Baseclass
- Changement de méthodes
- Contrats et héritage
- Contrats et sous-typage
- Variations d'arguments de méthode
- Règles pour les arguments de méthode
- Règles pour changer les types de retour
- Annulations de méthodes

Module 17 : Heuristiques de conception

- Directives de conception orientées objet
- Refléter la vue du client

- Vote
- Interfaces express à travers des objets
- Objets de valeur
- Invariants de classe
- Classes de base abstraites
- Classes et interfaces
- Interfaces de conception entre les classes de base et dérivées
- Cohésion entre Classes

Lab / Exercices

- Pendant le cours, les participants sont encouragés à participer activement à l'expérience d'apprentissage en exécutant des exemples de fichiers et en effectuant des tâches de codage pendant les labs
- Chaque session de lab vous permet de comparer votre solution à celle de l'instructeur

Documentation

- Support de cours numérique inclus

Profils des participants

- Développeurs logiciels
- Programmeurs C++
- Architectes logiciels
- Concepteurs de systèmes
- Ingénieurs en développement embarqué

Connaissances Préalables

- Avoir suivi ou maîtriser les notions incluses dans le cours suivant : Programmation Objet en C++ Fondamentaux

Objectifs

- Appliquer des concepts avancés de conceptions OO
- Être en mesure d'écrire et de maintenir des programmes C++
- Écrire un code C++ robuste, maintenable, élégant et efficace
- Utiliser les fonctionnalités avancées du langage de programmation C++
- Être capable de mettre en œuvre des techniques avancées orientées objet en C++ pour réaliser des applications efficaces et flexibles
- Avoir les compétences nécessaires pour développer des applications C++ industrielle

Description
Formation Programmation Orientée Objet en C++ Avancé
Niveau
Avancé
Prix de l'inscription en Présentiel (CHF)
3800
Prix de l'inscription en Virtuel (CHF)
3550
Durée (Nombre de Jours)
5
Reference
CPP-02