Programmation Orientée Objet en C++ Avancé

Description

Cette formation C++ offre une couverture complète des fonctionnalités avancées du langage C++. La formation commence par des concepts OO basiques, puis une introduction rapide au langage. Exceptée la POO, les sujets abordés comprennent les modèles, la bibliothèque standard (STL) et les exceptions. Le cours couvre également certaines techniques avancées de conception orientée objet en C++ telles que Design Heuristics, Design by Contract, programmation par interface, patterns de composition et de délégation, gestion de la mémoire et pointeurs intelligents, sous-typage, conception visant l'efficacité et la Meta-programmation en C++.

Niveau

Avancé

Contenu du cours

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

- Leçon 1 : Histoire de C++
- Leçon 2 : Versions
- Leçon 3 : Nouveau dans TR1
- Leçon 4 : Nouveau en C++ 11
- Leçon 5 : Nouveau en C++ 14

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

- Leçon 1 : Programmation orientée objet
- Leçon 2 : Constructeurs et destructeurs
- Leçon 3 : New et delete

Module 3 : Héritage de base

- Leçon 1 : Interface et implémentation
- Leçon 2 : Héritage de Type
- Leçon 3 : Héritage d'implementation'
- Leçon 4 : Utilisation correcte de C++ 11 final et override
- Leçon 5 : Destructeurs virtuels: quand et pourquoi?
- Leçon 6 : Directives d'héritage

Module 4 : Utilisation correcte des fonctionnalités linguistiques bien connues

- Leçon 1 : Pointeurs vs. Références vs Valeurs
- Leçon 2 : Bonne utilisation de const
- Leçon 3 : Utilisation appropriée des fonctions en ligne
- Leçon 4 : Bonne utilisation de static
- Leçon 5 : Utilisation appropriée des paramètres par défaut
- Leçon 6 : Bonne utilisation de friend
- Leçon 7 : Utilisation appropriée du namespace
- Leçon 8 : Le moyen C++ du cast
- Leçon 9 : Utilisation appropriée de la surcharge de l'opérateur
- Leçon 10 : Constructeur de copie: pourquoi / quand?
- Leçon 11 : Opérateur d'affectation: pourquoi / quand?
- Leçon 12 : La loi des trois grands

Module 5 : Exceptions

- Leçon 1 : Gestion traditionnelle des erreurs
- Leçon 2 : Traitement d'erreur orienté objet
- Leçon 3 : throw, try and catch
- Leçon 4 : Conception de hiérarchies d'exceptions
- Leçon 5 : Utilisation appropriée de rethrow
- Leçon 6 : Utiliser unexpected
- Leçon 7 : Pièges des d'exception
- Leçon 8 : Bonnes pratiques sur les exceptions

Module 6 : Modèles

- Leçon 1 : Définition des classes de template
- Leçon 2 : Implémentation des classes de modèles
- Leçon 3 : Classes paramétrées
- Leçon 4 : Modèles et paramètres non-types
- Leçon 5 : Directives de template
- Leçon 6 : Modèles et fonctions simples
- Leçon 7 : Fonction de template C++

Module 7 : Bibliothèque de modèles standard (STL)

- Leçon 1 : Chaîne de caractères STL
- Leçon 2 : Composants STL
- Leçon 3 : Conteneurs de séquence
- Leçon 4 : Utilisation d'itérateurs en STL
- Leçon 5 : Exemple d'algorithmes
- Leçon 6 : Initialisation des conteneurs
- Leçon 7 : Profils de performance des conteneurs de séquences

Module 8 : Algorithmes STL

- Leçon 1 : STL contre Boost
- Leçon 2 : Paramétrage des algorithmes
- Leçon 3 : Utiliser des fonctions
- Leçon 4 : Utilisation d'objets de fonction
- Leçon 5 : Utilisation d'expressions Lambda

- Leçon 6 : Bibliothèque d'algorithmes sélectionnés
- Leçon 7 : Algorithmes contributifs

Module 9 : Conteneurs associatifs STL

- Leçon 1 : Set
- Leçon 2 : Multiset
- Leçon 3 : Map
- Leçon 4 : Multimaps

Module 10 : Functeurs STL, Allocators et plus

- Leçon 1 : Exception standard
- Leçon 2 : Functors
- Leçon 3 : Objets de fonction fournis par la bibliothèque
- Leçon 4 : Utilisation d'objets de fonction STL et de liaisons
- Leçon 5 : Négateurs
- Leçon 6 : Allocateurs
- Leçon 7 : Nombre complexe
- Leçon 8 : Smart Pointers

Module 11 : Efficacité : Objets temporaires

- Leçon 1 : Objets temporaires : le problème
- Leçon 2 : Diverses techniques pour éviter les temporaires
- Leçon 3 : Chaîne STL : comment éviter la création de temporaires
- Leçon 4 : C++ 11 : Déplacer la sémantique
- Leçon 5 : Techniques diverses pour éviter les temporaires

Module 12 : Gestion de la mémoire

- Leçon 1 : Comment C++ utilise-t-il la mémoire?
- Leçon 2 : Directives de base
- Leçon 3 : Implémentation de singletons en C++
- Leçon 4 : Utilisation efficace des pointeurs intelligents
- Leçon 5 : Surcharge new et delete
- Leçon 6 : Gestion de la mémoire
- Leçon 7 : Comment C++ utilise-t-il la mémoire?
- Leçon 8 : Directives de base
- Leçon 9 : Implémentation de singletons en C++
- Leçon 10 : Utilisation efficace des pointeurs intelligents
- Leçon 11 : Surcharger new et delete
- Leçon 12 : L'importance de la disposition des données

Module 13 : Mémoire vive / froide

- Leçon 1 : Techniques d'efficacité diverses
- Leçon 2 : Préoccupations de conception
- Leçon 3 : Flexibilité vs performance
- Leçon 4 : Évaluation lazy
- Leçon 5 : Évaluation eager
- Leçon 6 : Copier sur les techniques d'écriture
- Leçon 7 : Mise en page des données revisitée

- Leçon 8 : Matériel moderne et pipelines de cache
- Leçon 9 : L'effet des structures de données et des algorithmes
- Leçon 10 : Postcondition et C++
- Leçon 11 : Profils d'efficacité des bibliothèques
- Leçon 12 : STL et Performance
- Leçon 13 : Latence
- Leçon 14 : Coût et avantages des threads
- Leçon 15 : Programmation asynchrone
- Leçon 16 : Futures

Module 14 : Délégation

- Leçon 1 : Concept de délégation
- Leçon 2 : Délégation en C++
- Leçon 3 : Délégation simple
- Leçon 4 : Délégation statique
- Leçon 5 : Délégation de super classe
- Leçon 6 : Délégation de sous-classe
- Leçon 7 : Problèmes avec la délégation de sous-classe en C++
- Leçon 8 : Délégation d'objet
- Leçon 9 : Modèle de stratégie
- Leçon 10 : C++ et Stratégie
- Leçon 11 : Modèle d'état
- Leçon 12 : C++ et état
- Leçon 13 : Conception de composite
- Leçon 14 : Composite et délégation
- Leçon 15 : Autres modèles de délégation

Module 15 : Découplage

- Leçon 1 : Qu'est-ce que le couplage?
- Leçon 2 : Types de couplage
- Leçon 3 : Couplage d'identité
- Leçon 4 : Couplage d'identité: cycle de vie de l'objet
- Leçon 5 : Changement d'identité
- Leçon 6 : Type de couplage
- Leçon 7 : Couplage d'implémentation
- Leçon 8 : Interfaces et implémentations
- Leçon 9 : Découplage par l'exemple

Module 16 : Héritage avancé

- Leçon 1 : Héritage multiple des interfaces
- Leçon 2 : Héritage multiple de l'implémentation'
- Leçon 3 : Propriétés partagées
- Leçon 4 : Résoudre l'ambiguïté
- Leçon 5 : Héritage virtuel
- Leçon 6 : Multi méthodes
- Leçon 7 : Double Dispatch
- Leçon 8 : Utilisation de RTTI
- Leçon 9 : Règles et lignes directrices
- Leçon 10 : Héritage des méthodes Baseclass

- Leçon 11 : Changement de méthodes
- Leçon 12 : Contrats et héritage
- Leçon 13 : Contrats et sous-typage
- Leçon 14 : Variations d'arguments de méthode
- Leçon 15 : Règles pour les arguments de méthode
- Leçon 16 : Règles pour changer les types de retour
- Leçon 17 : Annulations de méthodes

Module 17 : Heuristiques de conception

- Leçon 1 : Directives de conception orientées objet
- Leçon 2 : Refléter la vue du client
- Leçon 3 : Vote
- Leçon 4 : Interfaces express à travers des objets
- Leçon 5 : Objets de valeur
- Leçon 6 : Invariants de classe
- Leçon 7 : Classes de base abstraites
- Leçon 8 : Classes et interfaces
- Leçon 9 : Interfaces de conception entre les classes de base et dérivées
- Leçon 10 : 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 d'applications
- Programmateurs
- Concepteurs de système

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

Prix de l'inscription en Présentiel (CHF)
3800
Prix de l'inscription en Virtuel (CHF)

3550
Durée (Nombre de Jours)
5
Reference
CPP-02