C++ expert, les avancées du langage (de C++ 11 à 20)

Cours Pratique de 3 jours - 21h
Réf : VEC - Prix 2024 : 1 870€ HT

Ce cours vous permettra d'assimiler les nouveautés introduites par les nouveaux standards C++. Vous couvrirez les possibilités de la programmation fonctionnelle, maîtriserez la gestion de la mémoire avec les smart pointers et exploiterez les autres nouveautés de la bibliothèque standard C++.

OBJECTIFS PÉDAGOGIQUES

À l'issue de la formation l'apprenant sera en mesure de :

Appréhender les nouveautés et les améliorations du langage C++

Utiliser les lambda expressions

Améliorer ses performances avec la sémantique de déplacement

Maîtriser les allocations-destructions d'objets avec les smart pointers

Développer une application multithreadée

TRAVAUX PRATIQUES

Des exercices pratiques de programmation permettront d'appréhender les différents concepts abordés.

LE PROGRAMME

dernière mise à jour : 07/2021

1) L'avènement de C++11

- Les différentes normes C++98, C++03, C++11=, C++14, C++17, C++20.
- Les nouveautés de C++11 et les objectifs de cette norme. Le devenir de Boost, STL.
- La question de la compatibilité des codes anciens.
- La disponibilité des outils de développement (compilateurs, débogueurs, IDE...).

Travaux pratiques : Vérification de l'outillage à l'aide d'un code C++11 fourni.

2) Les améliorations du langage

- Les enum fortement typées.
- Le mot-clé auto et decltype pour simplifier le typage.
- La boucle basée sur un intervalle.
- Constexpr pour une évaluation à la compilation.
- Templates variadiques, concepts de C++20 soucoupe...
- Coroutines de C++20.
- Les modules de C++20.

Travaux pratiques : Mise en œuvre des améliorations.

3) Les modifications au niveau des classes

- La délégation de constructeurs, les contraintes liées à l'héritage.
- La nouvelle sémantique du déplacement et le constructeur par déplacement (move constructor).
- Les directives override, final =delete, =default.
- Notion de module C++20

Travaux pratiques : Création de classes.

PARTICIPANTS

Développeurs C++ souhaitant connaître les nouveautés définies par le standard C++11.

PRÉREQUIS

Bonnes connaissances du langage C++. Une expérience pratique de la programmation avec ce langage est recommandée.

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…
Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les stages pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque stage ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le stagiaire a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Vous avez un besoin spécifique d'accessibilité ? Contactez Mme FOSSE, référente handicap, à l'adresse suivante psh-accueil@orsys.fr pour étudier au mieux votre demande et sa faisabilité.

4) La programmation fonctionnelle avec les expressions lambda

- Déclaration, typage, implémentation et utilisation.
- L'intérêt d'auto avec les lambda-expressions.
- La gestion des fermetures (closures), avec capture par valeur ou par référence des variables liées au contexte.

Travaux pratiques : Exercices de programmation fonctionnelle.

5) L'utilisation des threads

- Déclaration et exécution d'un thread. Attente de fin d'exécution avec join().
- La gestion des données locales à un thread.
- Récupérer un résultat avec future<> et async().
- Choisir parmi les différents verrous de la STL.

Travaux pratiques : Multithreader un code séquentiel et mesurer le gain en termes de temps d'exécution.

6) Autres nouveautés de la bibliothèque standard

- Ranges avec std ::view pour évaluations à la volée de C++20.
- La gestion du temps avec le namespace chrono.
- Le nouveau conteneur tuple.
- Conteneurs unordered_set, unordered_map à base de hachage.
- Formatage de string avec C++20.

Travaux pratiques : Mise en œuvre des nouveautés.

7) La gestion mémoire et les conteneurs

- Les smart pointers : shared_ptr, weak_ptr, unique_ptr. Usage conjoint avec la STL.

Travaux pratiques : Mise en œuvre de la gestion mémoire C++11.

LES DATES

CLASSE À DISTANCE
2024 : 15 juil., 28 oct.

PARIS
2024 : 08 juil., 21 oct.

LYON
2024 : 15 juil., 28 oct.

AIX-EN-PROVENCE
2024 : 15 juil., 28 oct.

BORDEAUX
2024 : 15 juil., 28 oct.

LILLE
2024 : 15 juil., 28 oct.

NANTES
2024 : 15 juil., 28 oct.

SOPHIA-ANTIPOLIS
2024 : 15 juil., 28 oct.

STRASBOURG
2024 : 15 juil., 28 oct.

TOULOUSE
2024 : 15 juil., 28 oct.