

C++ : Fondamentaux pour programmeurs Java ou C - 4 jours

formation 337

- Vous apprendrez à**
- Exploiter C++ pour construire des applications souples et extensibles
 - Convertir, adapter et interfacier des applications Java et C à C++
 - Concevoir des applications C++ fiables et que l'on peut maintenir
 - Construire des logiciels avec des classes génériques et des classes container
 - Utiliser des bibliothèques standard internationales pour rendre les programmes plus simples, plus portables et plus fiables
 - Gérer l'allocation mémoire en utilisant les constructeurs/destructeurs

Objectif Les programmes développés selon une approche objet sont plus faciles à comprendre et à maintenir que les programmes conçus de manière traditionnelle. Les méthodes objet sont la clé pour obtenir des logiciels réutilisables et réduire les coûts d'adaptation à de nouvelles spécifications. Le cours couvre tout ce que vous devez savoir sur C++, y compris les dernières extensions. Le cours introduit aussi les bibliothèques standard et comment convertir et interfacier C++ et C à Java. L'accent est mis sur l'utilisation du langage, les pièges à éviter et les principes de la programmation objet. Les travaux pratiques permettent de développer des programmes objet en utilisant C++.

À qui s'adresse cette formation Ce cours est destiné aux professionnels de l'informatique qui sont impliqués dans le développement des applications ou des systèmes en C++. Une expérience de la programmation en C ou Java est nécessaire.

Travaux pratiques Durant ce cours, de nombreux exercices, conduits par un instructeur expert, permettent de renforcer les concepts et techniques de la programmation objet, y compris :

- Écrire des composants logiciels réutilisables
- Utiliser les classes génériques pour les structures de données
- Utiliser les exceptions pour un traitement d'erreurs robuste
- Encapsuler les données dans des classes
- Étendre les classes de base en utilisant l'héritage

C++ : Fondamentaux pour programmeurs Java ou C - 4 jours

formation 337

Introduction et présentation

Programmation objet

- Qu'est-ce que la programmation objet ?
- L'évolution de la programmation objet
- Classes C++ pour encapsuler des données
- Objets, types et classes

Avantages des méthodes objet

- Éviter les fonctions et données globales
- Fiabilité et maintenabilité

Introduction à C++

C++ pour la programmation objet

- Limitations de C comme langage objet
- Concepts de base de C++
- C++ = C + Typage fort + Classes

Structure d'un programme en C++

- Syntaxe de C++
- Spécification et déclaration des fonctions
- Surchage des fonctions et des opérations
- Passage de paramètres
- Utilisation de la compilation séparée et des fichiers "include" pour la modularité

Classes en C++

- Déclaration et utilisation des classes
- Constructeurs

Utilisation de l'héritage

Classes dérivées

- Polymorphisme et édition de lien dynamique
- Publique, privée et protégée
- Délégation ou héritage
- Initialisation dans la hiérarchie
- Classes internes
- Fonctions virtuelles pures
- Héritage multiple
- Classes abstraites

Typage dynamique

- Sous-typage des pointeurs
- Recherche du type d'un objet
- Comparaison de types

Concepts avancés de C++

Types avancés en C++

- Utilisation des types référence
- Utilisation des fonctions membres **const**
- Utilisation de **const** pour améliorer la fiabilité et l'efficacité

Fonctions avancées du C++

- Fonctions **friends**
- Surchage des opérateurs
- Surchage des opérateurs () et []
- Fonctions **inline**

- Valeur par défaut des arguments

Gestion de la mémoire

Stockage statique

- Données membres statiques
- Initialisation des données globales

Allocation de mémoire dynamique

- Gestion mémoire en C++
- **new** et **delete**
- Constructeurs de copie
- Le danger des alias
- Utilisation de destructeurs
- Redéfinition de l'affectation pour éviter les alias

Bibliothèques standard en C++

Classes et fonctions génériques

- Réutilisation grâce aux types paramétrés
- Déclaration de classes container
- Déclaration et utilisation de la généricité (**template**)

Bibliothèques réutilisables et portables

- Utilisation des algorithmes standard : **find**, **for each**, **sort**
- Formatage en utilisant des manipulateurs d'E/S
- Containers et structures de données portables
- Hiérarchie des classes d'E/S
- Stockage des données dans des conteneurs standard : liste, ensemble, vecteur
- Itérateurs

Gestion des exceptions

- Gestion des exceptions dans les bibliothèques
- Exceptions : **catch**, **throw**, **try**
- Gestion fiable des exceptions
- Exceptions standard des bibliothèques

Conception et maintenance de C++

Organisation du projet C++

- Maintenance des applications C++
- Organisation des systèmes en utilisant les espaces de noms
- Contrôle de la conversion de type avec le transtypage dynamique
- Définition et utilisation des interfaces

Combiner C++ avec Java et C

- Lier les fichiers objet C et C++
- Convertir des structures et des fonctions globales en classes
- Éliminer les instructions **case**
- Interfaçage entre Java et C/C++

Standards et extensions

- Standards vs. implémentations spécifiques aux plates-formes
- Applicabilité à Windows et UNIX/Linux
- ANSI/ISO 98 et évolution vers le standard "OX"
- ECMA C++/CLI et autres extensions

Structure des programmes avec C++, Java et C

- Organisation de programmes
- Allocation de la mémoire
- Conversion de programmes
- Recherche des erreurs de conversion courantes