# Programmation Orientée Objet en C++ – Perfectionnement

## Programme

### Compléments sur le concept de généricité en C++

- Définitions de méthodes et de classes génériques
- Template avec types par défaut
- Utilisation du mot clé typedef dans le cadre de la généricité
- Template à nombre variable de types
- Spécialisation de template
- Notions de « traits » et l'entête

### Manipulation de chaînes de caractères via des expressions régulières

- La librairie
- Qu'est qu'une expression régulière ?
- Le langage d'expression régulière proposé par C++
- Mise en correspondance, remplacement et extraction d'informations
- Les expressions régulières gourmandes
- TP : utilisation d'expressions régulières pour réaliser des mises en correspondances, des remplacements et de extraction d'informations Compléments Boost pour la manipulation de chaînes de caractères

### Gestions des copies en C++

- Opérateurs de copie et constructeur par copie
- Correctement réaliser les copies
- Interdire les copies
- Opérateurs et constructeurs de « move »

### RAII et les « Smart Pointers »

- Introduction à RAII (Resource Acquisition Is Initialization)
- Mise en oeuvre d'une classe générique de « Smart Pointers »
- Problématique des cycles de pointeur
- Les différentes classes de « Smart Pointers » proposées en C++ ISO

### La librairie  et la réflexion en C++

- La librairie
- L'opérateur typeid
- Transtypage dynamique : opérateur dynamic_cast
- Les autres opérateurs de transtypage

### Utilisation de la STL (Standard Template Library)

- Présentation de l'architecture STL : les conteneurs, les itérateurs et les algorithmes
- Les différents types d'itérateurs
- Les différentes techniques de parcours de collections : itérateurs, for_each, instruction for each, …
- Concept de « functions objects »
- Les différents types de collections de la STL
- Les conteneurs ajoutées par C++11
- Etude de divers algorithmes proposés par la librairie standard

Référence

THIL980

Durée

5 jours / 35 heures

Prix HT / stagiaire

2750€

### Objectifs pédagogiques

- Concevoir les mécanismes avancés de C++ (héritage multiple / virtuel, programmation générique, méta-programmation, …)
- Concevoir les techniques de la nouvelle norme C++11/14 telles que les lambda-expressions, la programmation fonctionnelle ou la notion de move semantics
- Pratiquer l'approche TDD (Test Driven Development) en C++
- Identifier la programmation parallèle et synchronisée introduite par C++

### Niveau requis

- Garantir avoir suivi la formation « Programmation Orientée Objet en C++ – Initiation » ou connaître les bases de la programmation orienté objet en C++.

### Public concerné

- Ce cours s'adresse aux développeurs ainsi qu'aux chefs de projets informatique souhaitant compléter leurs connaissances sur le langage de programmation C++
- Chargés de développement d'applications informatiques
- Chefs de projets développement

### Formateur

Les formateurs intervenants pour Themanis sont qualifiés par notre Responsable Technique Olivier Astre pour les formations informatiques et bureautiques et par Didier Payen pour les formations management.

### Moyens pédagogiques et techniques

Salles de formation (accessibles et adaptables aux besoins des personnes en situation de handicap) équipée d'un ordinateur de dernière génération par stagiaire, réseau haut débit et vidéo-projection UHD
Documents supports de formation projetés
Apports théoriques, étude de cas concrets et exercices
Mise à disposition en ligne de documents supports à la suite de la formation

### Dispositif de suivi de l'éxécution de l'évaluation des résultats de la formation

Feuilles d'émargement (signature électronique

## Programmation fonctionnelle en C++ et moteur de binding

- Le concept de lambda
- Les différents types de captures proposés
- Le bind C++
- TP: application des lambda à la manipulation de conteneurs STL

## Manipulation du système de fichiers

- La librairie  introduite par C++ ISO 2017
- Utilisation de path
- Fonctions de convenances
- Couplage avec le librairies de manipulation de flux (istream et ostream)

## Programmation multi-ta?hes en C++

- Pourquoi programmer en multi-tâches ?
- Qu’est ce que thread et qu’est-ce qui le différencie d’un processus ?
- La librairie
- Connaître les spécificités multi-tâches du système d’exploitation utilisé
- Les mécanismes de synchronisation des accès concurrents
- RAII et la gestion des verrous
- Problèmes de performances en cas de mauvaise stratégie de synchronisation
- Opérations atomiques
- Utilisation de local storage