

## Objectifs :

- savoir écrire, compiler et déboguer des programmes orientés objets en C++ ANSI
- savoir utiliser les pointeurs et les références
- connaître les meilleures pratiques
- savoir utiliser les outils de développement C++ (VisualStudio, Eclipse, GCC)
- savoir utiliser les templates C++
- utiliser les classes de la STL

Durée : 5 jours

Public : développeurs, chefs de projets

## Prérequis :

- connaissance d'un langage de programmation
  - la connaissance d'un langage proche syntaxiquement de C est un plus
  - la connaissance d'un langage objet est un plus

## Démarche pédagogique :

- présentation des concepts, suivi d'exercices
- plusieurs ateliers font appels à la conception des classes, et au cycle de vie des instances

## Programme détaillé :

- Introduction
  - la programmation objet
    - classe, instance, propriété, méthodes, ...
    - présentation des diagrammes de classes et de séquence (UML 2)
  - évolution du langage C++
    - du langage C à la norme C++ 11
  - les environnements de développement
    - EDI : VisualStudio, DevC++, Eclipse CDT
    - compilateurs en ligne de commande : GCC, CL
- C++ - la syntaxe
  - syntaxe de base
  - les mots clefs
  - les types
  - les fonctions
  - visibilité des variables
  - les tableaux
  - modèle mémoire
- C++ - les pointeurs et les références
  - adresse d'une variable
  - pointeurs

- création
  - allocation et désallocation - opérateurs `new` et `delete`
  - déréférencement
  - pointeur `null`
  - passage de pointeur à une fonction
  - retour de pointeur
  - les pointeurs et les tableau
    - évaluation d'un tableau
    - pointeur vers tableau
    - arithmétique des pointeurs
- référence
  - création d'une référence
  - différence avec le pointeur
  - passage de référence
  - retour de référence
- C++ - les classes
  - structures et classes
  - propriété
  - méthode
  - visibilité des membre de classe
  - constructeurs et destructeur
- C++ - le polymorphisme
  - l'héritage
  - fonctions virtuelles, virtuelles pures et classes abstraites
  - polymorphisme sur les pointeurs et les références
  - héritage multiple et héritage virtuel
  - appels des constructeurs et des destructeurs
  - constructeur de copie
- C++ - surcharges des opérateurs
  - règles de surcharge
  - appels de l'opérateur d'affectation entre instances de classes
  - surcharge dans la classe ou globale
  - fonctions amies, classes amies, méthodes amies
- C++ - les exceptions
  - gestionnaire d'exception
  - levée d'exception
  - bloc `try - catch`
- C++ - les templates
  - template de fonction
  - template de classe
  - héritage entre template
- C++ - quelques notions avancées
  - les foncteurs
  - pointeur vers membres statiques d'une classe
  - pointeur vers membres non statique d'une classe

- les espaces de nommage
- C++ - présentation des bibliothèques standard
  - les bibliothèques standard
  - la classe `string`
  - les flux en entrée-sortie
    - opérateurs d'insertion et d'extraction
    - mise en forme
    - entrée-sortie sur fichier
- C++ - présentation de la STL
  - utilisation des templates dans la STL
  - les collections
    - `vector`, `list`, `map`...
    - opérations sur les collections
  - utilisation de la STL
- C++ - pour aller plus loin
  - responsabilité des classes
  - responsabilité du cycle de vie des instances
  - déréférencement des pointeur
  - spécification de codage
  - utilisation de `const`
  - objets immuables
  - les opérateurs de cast
    - `<static_cast>`, `<dynamic_cast>`, ...
  - les ajouts de C++14
  - les expressions Lambda
  - `&&` : les rvalue references
  - sérialisation d'instance