Comment suivre ce cours ?

Sommaire principal	Chapitre suivant

Cours pratique de création d'application en C++ moderne

- **cours** : à la différence d'un article ou d'un tutoriel, un cours est conçu selon une approche et un objectif pédagogiques. Il n'est pas destiné à être une dictionnaire comme cppreference et vous ne verrez pas toutes les syntaxes possibles et toutes les fonctionnalités du langage. L'objectif est de vous donner progressivement les informations.
- **pratique** : un langage de programmation ne s'apprend pas dans les livres ou dans un cours. La programmation s'apprend par la pratique. Vous trouverez de nombreux exercices dans ce cours, dans le but de vous permettre de pratiquer chaque notion que vous aurez appris. Un notion qui n'est pas pratiquée est une notion qui n'est pas assimilée, lire ce cours sans pratiquer les exercices vous permettra pas d'apprendre correctement.
- **création d'application** : l'objectif de ce cours n'est pas de vous apprendre le C++. Plus précisément, l'objectif n'est pas de vous apprendre simplement la syntaxe du langage C++. La programmation d'une application nécessite d'autres connaissances, comme le cycle de vie des applications, la conception, l'algorithmie, etc. Et plus généralement, tout l'écosystème du C++ : les outils, les bibliothèques, les méthodes, etc.
- **C++ moderne** : le C++ est un langage en constante évolution. La norme actuelle date de 2011, les prochaines normes sont prévues en 2014 et 2017. De nombreux cours sur internet ou dans les formations se basent sur l'ancienne norme du C++03,

le C++11/14 étant enseigné (quand c'est enseigné) comme un ajout au C++03.

Apprendre Qt ?

Remarque sur Qt. Pourquoi : apprenant doit savoir utiliser des libs externe et GUI plus fun que du console. Problématique : croire que Qt = C++ ou que Qt représente une bonne pratique du C++

Pédagogie de ce cours

partir des bonnes pratiques générales, puis voir les spécificités de codage particulières

Gamification : le but des tests n'est pas de sanctionner un apprentissage, mais de permettre à l'apprenant de suivre sa progression. Doit être très pédagogique : beaucoup de petits tests (désacraliser les tests, ne pas voir les tests comme des épreuves), valorisant (tester pas uniquement si acquisition des points difficiles, mais aider à réaliser tout ce que l'on a acquis), revenir régulièrement sur les notions déjà acquise (enseigner, c'est répéter).

Modules : tronc commun d'apprentissage (base du C++, les bonnes pratiques), puis modules optionnels organiser en thèmes : bas niveau/programmation système/embarqué, les interfaces graphiques, les jeux, le C++ old school, allez plus loin, etc.

Qu'est que veut dire "créer une application" ?

Plusieurs approches possibles pour créer un programme : Plusieurs approches possible. On dev pas pareil pour un micro controleur (mémoire, performances) que pour un ordi de bureau ou un serveur de calculs intensif (performances). Impact sur les méthodes de développement et sur ce que l'on accepte ou non de faire en termes de bonnes pratiques de codage

Création d'un application :

- Maintenabilité : corriger les bugs rencontrés durant la vie du programme
- Évolutivité : ajouter de nouvelles fonctionnalités
- Fiabilité : les résultats sont corrects
- Performance : éviter les pertes inutiles de performances. différence entre chercher la meilleur performance et éviter de faire du code qui explose inutilement les performances. Le second nécessite juste le respect de bonnes pratiques de codage. Le premier nécessite une approche de développement spécifique (profiling)

Donc objectif = arriver le plus rapidement possible au résultat

- KISS = rester simple
- NIH = ne pas réécrire ce que existe déjà
- DRY = ne pas se répéter

Sommaire principal	Chapitre suivant

Cours, C++