



Centre Informatique pour les **L**ettres
et les **S**ciences **H**umaines

Apprendre C++ avec Qt : Leçon 0 Présentation

1 - Objectif	2
2 - Choix techniques	2
3 - Méthode de travail.....	3
Organisation générale.....	3
Limites d'efficacité pédagogique	3
4 - M12 : Calendrier 2005/2006.....	3
5 - Foire Aux Questions.....	4

1 - Objectif

Les modules INF M12 (*Algorithmes et structures de données 1*), INF M22 (*Projet de programmation*) et INF M32 (*Algorithmes et structures de données 2*) visent à rendre les étudiants de Master qui les suivent capables de

- réaliser seuls des petits programmes effectuant des traitements très spécialisés utiles, notamment, dans le cadre de projets propres au Master suivi ;
- contribuer de façon efficace à des projets informatiques de plus grande envergure, pour faire bénéficier ces projets des compétences non informatiques acquises en Master.

Si apprendre à programmer semble clairement nécessaire pour pouvoir mettre au point des programmes, fussent-ils "petits", il est peut être moins évident qu'une contribution "non informatique" à un projet puisse bénéficier d'une compétence réelle en programmation. L'expérience montre pourtant qu'un géographe, un musicien, un linguiste ou un ergonome peuvent avoir une influence plus grande sur un projet auquel ils collaborent s'ils ne sont pas totalement hermétiques aux techniques mises en œuvre. Une "distance culturelle" trop importante du reste de l'équipe risque en effet d'isoler le spécialiste en "sciences humaines" et de réduire sa contribution à la fourniture d'un label vide de sens, apposé sur un projet qui n'aurait pas débouché sur un produit réellement différent en l'absence de ce spécialiste.

L'ambition de ces modules est donc double : rendre le Master plus intéressant pour les étudiants (en permettant des réalisations concrètes) et rendre ces étudiants plus attractifs pour les employeurs potentiels (en augmentant leur intégrabilité dans des projets informatiques).

2 - Choix techniques

Etant donné les objectifs poursuivis, il semblait nettement préférable d'adopter un langage dont la disponibilité ne soit pas liée à un fabricant particulier et s'étende au-delà du seul univers micro-informatique. Il n'est par ailleurs pas possible de connaître avec exactitude les problèmes réels que les différents étudiants rencontreront effectivement (problèmes qui seront, de toutes façons, certainement très divers), ce qui plaide en faveur de l'adoption d'un langage assez général. Notre choix s'est donc porté sur C++, étant entendu qu'il reste de la responsabilité de l'enseignant de ne pas se perdre gratuitement dans les détails d'un langage ou d'un compilateur donné, mais de veiller à assurer une certaine généralité aux concepts transmis.

La réalisation de programmes comportant une interface graphique conforme aux attentes des utilisateurs en ce début de XXI^e siècle impose pratiquement l'utilisation d'une librairie étendant dans cette direction les possibilités offertes par le langage C++. Toujours dans le souci de ne pas faire acquérir à nos étudiants des connaissances dont la validité serait restreinte à un système particulier, nous avons porté notre choix sur la librairie Qt, de la société Trolltech, qui présente la particularité d'être disponible à la fois sous Apple MacOS X, sous Microsoft Windows et sous Unix (et, en particulier, sous Linux).

Les modalités pratiques d'enseignement de l'informatique à Aix imposent que les étudiants soient en mesure de déployer sur leur machine personnelle le système de développement choisi. La prise en compte de ce facteur contraint pratiquement à permettre l'utilisation d'un compilateur **Windows**, et une série de TD a donc été rédigée pour Microsoft **Visual C++ 6.0**. Du fait de la capacité qu'a l'éditeur d'interface Qt Designer de travailler en collaboration avec l'environnement de développement Microsoft, cette version des TD offre une facilité de mise en œuvre convenant tout à fait au public visé.

Visual C++ 6.0 et Qt Designer forment un ensemble de logiciels qu'il est possible d'installer en toute légalité sur une machine personnelle pour un coût quasiment négligeable. Cette question est discutée de façon plus approfondie dans le TD 0.

La réalisation des TD est également possible sous **Linux**, en utilisant l'environnement de développement **KDevelop**, et cette approche fait l'objet d'une série de documents spécifiques, qui la rendent accessible à des utilisateurs Linux novices.

Les distributions de Linux les plus répandues (Mandrake, Red Hat, etc.) incluent KDevelop et Qt, ce qui fait que l'environnement que nous utilisons sous Linux est non seulement gratuit mais, dans bien des cas, déjà installé...

3 - Méthode de travail

La caractéristique majeure de ce cours est qu'il est intégralement disponible sous forme écrite.

Organisation générale

Le matériel proposé comporte d'une part des Leçons qu'il s'agit "simplement" de lire et de comprendre et, d'autre part, des TD qu'il faut effectivement réaliser.

La progression est calculée pour un "étudiant moyen" qui consacre, en plus des trois heures de cours, deux à trois heures de travail personnel par semaine à ce module. Ce travail personnel est essentiellement consacré à anticiper sur le déroulement du cours : chaque Leçon doit avoir été lue attentivement, et chaque TD au moins commencé avant la séance correspondante.

Cette façon de travailler vous permet de franchir seul les étapes qui ne vous posent aucun problème, et de rentabiliser la présence de l'enseignant en lui soumettant les cas où vous rencontrez des difficultés ou souhaitez des explications complémentaires.

L'exposé que fait l'enseignant pendant un cours n'est qu'une version abrégée de la Leçon, et la durée de la séance consacrée au TD ne permet qu'exceptionnellement de le terminer.

S'il vous arrive de rencontrer une difficulté sérieuse (par exemple un problème vous bloquant dès le début d'un TD et vous empêchant donc de le préparer) ou si vous avez une question qui ne porte pas sur la Leçon en cours (parce que vous avez pris du retard, ou parce que vous êtes en train de réviser), envoyez sans hésiter un courrier électronique à progcpp@up.univ-aix.fr.

Limites d'efficacité pédagogique

Le volume horaire évoqué ci-dessus est nécessaire pour réellement tirer parti du cours : une étude attentive des Leçons peut éventuellement vous permettre de réussir les examens en investissant moins de temps que cela, mais vous n'apprendrez jamais à programmer ainsi.

Pour que ce que vous apprendrez dans les Leçons ait une chance de vous servir un jour à quelque chose, il vous faudra consacrer beaucoup de temps aux TD. Dans la plupart de ceux-ci, les manœuvres nécessaires à la réalisation du projet sont décrites en détail, dans l'ordre où elles doivent être effectuées. Si vous vous contentez d'obéir à ces injonctions sans chercher à en saisir la logique, vous obtiendrez très rapidement le programme décrit par le TD, mais pas grand chose de plus. Essayez d'imaginer des variantes à la solution donnée, testez vos idées sur la machine, et **ne négligez pas les exercices** qui vous sont proposés.

4 - M12 : Calendrier 2005/2006

Mardi, de 11h à 13h, en C212	
4 oct	Présentation
11 oct	Leçon 1 : Principes fondamentaux
18 oct	Leçon 2 : Variables, constantes et références
25 oct	Leçon 3 : Fonctions sans paramètre
	< Toussaint >
8 nov	Leçon 4 : Structures de contrôle
15 nov	Exercices collectifs
22 nov	Leçon 5 : Fonctions avec paramètres
29 nov	Mini-projet 1 : analyse collective
6 déc	Leçon 6 : Manipuler du texte
13 déc	Leçon 7 : Fichiers
20 déc	Mini-projet 2 : analyse collective
3 jan	Examen blanc et correction
10 jan	Examen final sur papier (12 pts)

Vendredi, de 10h à 12h, en C212/C222	
7 oct	Démonstration
14 oct	TD 1 : Trop facile !
21 oct	TD 2 : Le débogueur
28 oct	TD 3 : Une calculette
4 nov	Evaluation sur machines (3 pts)
	< 11 novembre >
18 nov	TD 4 : Boucles et tests
25 nov	TD 5 : Tic tac toe
2 déc	Mini-projet 1 : réalisation
9 déc	TD 6 : Un concordancier
16 déc	TD 7 : Calculs simples
23 déc	Mini-projet 2 : réalisation
6 jan	Evaluation sur machines (5 pts)

La consultation de documents écrits (notes personnelles, texte des Leçons et TD, manuels, dictionnaires...) est **autorisée** pendant les épreuves théoriques et pratiques.

En cas de conflit avec votre emploi du temps, veuillez contacter l'enseignant (des solutions peuvent être trouvées, même en cas d'incompatibilité totale avec l'horaire normal).

5 - Foire Aux Questions

Est-ce un avantage déterminant d'avoir déjà fait de la programmation avant d'entreprendre INF M12 ?

Pas sûr. Plusieurs des candidats qui ont échoué aux examens avaient déjà une expérience dans un ou plusieurs autres langages. Certains avaient même déjà fait du C++.

Est-ce un module très difficile ?

Je dirais plutôt qu'il s'agit d'un module auquel il est totalement impossible de réussir sans travailler. La disponibilité de l'intégralité du cours sous forme écrite rend, en revanche, l'échec extrêmement improbable pour un étudiant moyen qui s'est préparé dans de bonnes conditions.

Où peut-on se procurer le matériel correspondant au cours ?

Tout ce qui vous est nécessaire et peut vous être fourni par l'Université se trouve sur le site Internet <http://www.up.univ-mrs.fr/wcpp>

Est-il possible de réussir sans assister aux cours ?

Oui, sans aucun doute. J'ai (à ma grande surprise) attribué des notes supérieures à 15 sur 20 à des candidats que j'ai rencontrés pour la première fois le jour de l'examen final. Attention, toutefois : cette façon de travailler ne convient pas forcément à tout le monde. Dans tous les cas, n'hésitez pas à utiliser le courrier électronique.

A quoi sert l'adresse électronique progcpp@up.univ-aix.fr ?

- A poser des questions.
- A appeler au secours quand on ne sait même plus quelle question poser.
- A signaler les erreurs encore présentes dans les Leçons ou les TD.

Tout ceci reste évidemment possible de vive voix, lors des séances de cours. Cependant, même si vous n'avez pas besoin de faire des phrases très raffinées (un mail n'est pas un courrier officiel), le fait d'avoir à exposer votre problème par écrit est souvent un facteur qui favorise largement votre compréhension de la réponse... Par ailleurs, le mail fonctionne toute l'année, y compris pendant les semestres durant lesquels le module qui vous intéresse n'est pas administrativement ouvert.

Est-il indispensable de disposer d'un ordinateur personnel ?

Je le croyais, jusqu'à ce que la réussite brillante de certains étudiants ne prouve le contraire. Aller très au-delà de M12 sans ordinateur personnel continue pourtant à m'apparaître comme une mission quasi impossible.

Pourquoi mêler un produit commercial (la librairie Qt, en l'occurrence) à la présentation d'un langage dont un des avantages capitaux est qu'il fait l'objet d'une norme indépendante de tout intérêt privé ?

S'appuyer sur Qt pour enseigner C++ présente un intérêt pédagogique double : les programmes proposés en TD peuvent être plus ambitieux (et donc plus gratifiants pour les étudiants qui les réalisent) et les aspects avancés du langage montrent leurs avantages avant même qu'on ne commence leur étude. Cette étude peut ensuite être entreprise avec une motivation et une représentation de l'objectif poursuivi qui la rendent bien plus efficace.

Par ailleurs, l'expérience de l'utilisation d'une librairie est en elle-même bénéfique : un ex-étudiant confronté à une nouvelle librairie sur son lieu de travail aura moins de mal à apprendre à s'en servir s'il en maîtrise déjà une que s'il n'en a jamais rencontré.

Enfin, bien qu'il s'agisse effectivement d'un produit commercialisé par la société Trolltech, la librairie Qt est disponible sous diverses licences qui, de fait, rendent son usage libre et gratuit dans le contexte de l'enseignement.

Ne serait-il pas plus judicieux d'apprendre le langage Ada (ou Basic, ou Cobol ou DieuSaitQuoi...) ?

Il existe une multitude de langages de programmation, et chacun d'entre eux mérite sans doute, à un titre ou un autre, qu'on s'y intéresse. Le choix de C++ pour les modules M12, M22 et M32 résulte d'un compromis où interviennent de nombreux facteurs, et il n'a pas été fait à la légère.