Avant-propos

Pourquoi un tel ouvrage ?

Les réponses à cette question sont multiples. Au cours de cette préface, nous essayerons d'y répondre sans trop de philosophie et de débats contradictoires. Nous commencerons par deux raisons évidentes :

- Ces deux langages de programmation, Java et C++, sont très semblables, tout au moins dans leurs syntaxes. Il est ainsi tout à fait possible de reprendre un morceau de code du premier langage et de l'appliquer sans adaptation dans le second.
- Ils sont tous les deux très populaires dans le monde de l'informatique, le premier avec la venue d'Internet, le second comme langage système essentiel.

Pour assurer une progression constante et logique, au travers d'une comparaison directe de ces deux langages, il nous a fallu structurer la présentation du livre en fonction des particularités de chacun. Cela n'a pas été facile et nous avons accepté le défi de tenter une telle expérience. Il a été évidemment impossible de couvrir tous les détails de Java et de C++ car nous avons voulu que cet ouvrage conserve une épaisseur raisonnable.

À qui s'adresse ce livre ?

Nous aimerions dire aux débutants, mais ce ne serait sans doute pas honnête vis-à-vis des experts et des gourous C++, car ce dernier langage est considéré comme l'un des plus difficiles à assimiler et à maîtriser. Nous pensons aussi aux programmeurs Basic ou, mieux encore, à ceux favorisés par leurs connaissances dans des langages plus orientés objet, comme Delphi, le langage Pascal de Borland ou le C#.

Comme autres cas de figure, nous citerons également les programmeurs Java, Smalltalk, C ou C++ traditionnels qui aimeraient s'initier à l'un de ces deux langages essentiels et les ajouter à leur curriculum vitae.

Quel est l'intérêt d'apprendre plusieurs langages ?

Dans le cas précis, comme ces deux langages sont très proches, autant les apprendre en parallèle. Un autre aspect, encore plus essentiel, est l'apprentissage de leurs différences,

de leurs qualités et de leurs défauts, ce qui nous amènera ainsi à programmer en utilisant des techniques variées et réfléchies. Le résultat sera un code beaucoup plus propre, conçu selon une vision plus large des systèmes, de leurs possibilités et de leurs limites.

Pour un programmeur, aussi bien débutant que professionnel, l'apprentissage d'un nouveau langage est avant tout enrichissant. Une fonctionnalité manquante dans un langage l'amènera à une remise en question de ses méthodes. Par exemple, quand un programmeur C++ découvre qu'il n'existe pas de destructeur en Java, il va se poser toute une série de questions. En fin de compte, cette réflexion pourra sans aucun doute le conduire à réviser ses concepts et à écrire ses programmes C++ différemment. De la même manière, un programmeur C qui cherche à apprendre C++ et Smalltalk en parallèle, ce qui est loin d'être une mauvaise idée, va sans doute découvrir de nouveaux termes comme celui de variable de classe et se mettra à douter de ses vieilles habitudes en programmation C. Enfin, un programmeur qui, en guise de premier exercice en Java, décide de transférer une page Web sur son site, va certainement remettre en question ses choix lorsqu'il devra employer tel ou tel langage pour ses applications.

En fin d'ouvrage, dans le dernier chapitre, nous donnerons un aperçu du langage de Microsoft né en 2001, le C#. Nous serons alors surpris de retrouver la plupart des thèmes déjà abordés avec Java et C++ et combien il sera aisé alors d'écrire un premier programme dans ce langage. Le lecteur se rendra compte alors que son bagage informatique est déjà solide.

Dans cet ouvrage, nous parlerons aussi de performances, ainsi que de la manière de vérifier et de tester les programmes. Ces deux aspects font partie de la formation des informaticiens ; ils sont importants dans l'acquisition des méthodes de travail et des connaissances techniques pour concevoir et développer des applications professionnelles.

Quelles versions de Java et de C++ ?

Il faut tout d'abord indiquer que cet ouvrage ne couvre que le bien nommé « pure Java ». Il en va de même pour le C++ que nous appelons plus précisément le « Standard C++ ». Un grand nombre de fonctions intégrées dans le JDK 1.6 (officiellement nommé JDK 6) peuvent être directement comparées à celles disponibles dans le Standard C++. Le code C++ présenté dans cet ouvrage, tout comme le code en Java, devrait de plus pouvoir se compiler et s'exécuter dans plusieurs environnements comme Windows ou Linux.

Les exemples contenus dans ce livre couvrent des domaines variés et parfois complexes. Cependant, le code présenté restera compilable, nous aimerions dire pour l'éternité, et l'acheteur de ce livre n'aura pas besoin d'acquérir une nouvelle édition dans les prochains mois.

Le langage Java a beaucoup évolué ces dernières années, au contraire du C++ qui a atteint une stabilité naturelle. Aujourd'hui, nous parlons de Java 2 (Swing, Beans), Java 5 ou Java 6, qui représentent les versions de 1.2 à 1.4, 1.5 et 1.6 respectivement, des JDK de Sun Microsystems. Certains concepts et constructions de la version 1.1 ont été remplacés par des techniques plus évoluées. Dans cet ouvrage, nous laisserons volontairement de

côté la plupart des méthodes et des classes dépréciées des versions antérieures à la 1.6. Nous mentionnerons aussi quelques nouveautés apparues depuis la version 1.5.

Pourquoi le Standard C++ ?

Pourquoi avons-nous choisi le Standard C++ et non pas le langage C ou le C++ traditionnel ? Il y a en fait plusieurs raisons à cela. La principale concerne les nouvelles bibliothèques qui ont été rajoutées à partir de 1995 et qui font partie du langage au même titre que celles, nombreuses, qui font partie de Java. Comme exemple, nous pourrions citer la classe `string` qui n'existe pas dans le C++ classique. Certaines classes des premières versions du Standard C++ ont été dépréciées et corrigées dans cette édition.

Le Standard C++ est maintenant aussi agréé par les standards ANSI et ISO et représente une base encore plus solide pour un langage qui ne devrait plus beaucoup évoluer selon les dires de son créateur, Bjarne Stroustrup.

Comment est présenté cet ouvrage ?

Lors de la présentation des langages, nous commençons en général par le C++ et poursuivons ensuite avec la partie Java. En effet, pourquoi ne serions-nous pas respectueux avec le plus vieux d'abord ? En cas de difficultés, suivant les connaissances du programmeur, il est tout à fait possible de passer plus rapidement sur un sujet et d'y revenir par la suite. Pour un programmeur Visual Basic, s'initier en Java et C++ va représenter un travail important, car la syntaxe sera toute nouvelle pour lui. Dans ce cas-là, il devra sans doute se concentrer d'abord sur la partie Java et revenir ensuite sur le C++ avec ses particularités et ses variantes plus complexes.

Si nous avions traité l'un ou l'autre de ces langages séparément, la structure de cet ouvrage aurait été totalement autre. Java et C++ possèdent en effet quelques particularités foncièrement différentes qui auraient pu être exposées d'une autre manière. Ici, la difficulté principale réside dans les premiers chapitres où, pour présenter des exemples qui tiennent debout, nous avons souvent besoin de connaître certains aspects du langage qui seront examinés dans les chapitres suivants, et ce tout en traitant en parallèle les deux langages.

Tous les exemples de code présentés dans les différents chapitres ont été compilés séparément et souvent sous différents systèmes d'exploitation. Ils sont bien évidemment présents sur le CD-Rom accompagnant cet ouvrage.

De quel matériel a-t-on besoin ?

Cet ouvrage est conçu pour travailler dans un environnement Windows (XP ou Vista). Dans l'annexe F, nous montrerons qu'il est aussi possible d'utiliser Linux.

Un système équipé d'un processeur cadencé à 450 MHz et de 256 Mo de mémoire vive est suffisant pour assurer une compilation des programmes et garantir un environnement

agréable pour le développement. Le CD-Rom fourni avec cet ouvrage contient tous les outils de développement nécessaires ainsi que NetBeans (voir annexe E) pour Java et C++. Pour pouvoir utiliser correctement NetBeans, un processeur récent et 1 Go de mémoire sont recommandés.

Nous trouverons aussi, sur ce même CD-Rom, un éditeur de texte et la version la plus récente du SDK de la plate-forme Framework .NET de Microsoft qui inclut un compilateur C#.

Pourquoi autant d'exemples et d'exercices ?

À notre avis, il n'y en a jamais assez.

Lorsque nous débutons en programmation, nous commençons par écrire de petits morceaux de programmes. Ensuite, nous passons à des exercices plus complexes, à de petits outils pour gérer notre travail ou même à écrire de petits jeux intelligents. L'essentiel est d'écrire du code, beaucoup de code. Un bon programmeur sera toujours un collectionneur d'exemples, d'essais et de programmes, qu'il pourra rapidement retrouver lorsqu'il en aura besoin.

Pratiquement tous les exercices de ce livre sont proposés dans les deux langages, sauf si cela est explicitement indiqué. Les solutions sont disponibles sur le CD-Rom d'accompagnement. Des `Makefile` (GNU `make`) sont à disposition pour chaque chapitre et permettent de recompiler le code en cas de besoin.

Les exercices ont donné à l'auteur l'occasion de vérifier la consistance et la structure de cet ouvrage. Ils peuvent aussi permettre aux lecteurs de suivre les progrès de leur apprentissage.

Commentaires et suivi de l'ouvrage

L'auteur apprécierait de recevoir des commentaires ou des critiques de son ouvrage à l'adresse électronique suivante :

```
À:      jean-bernard@boichat.ch
Objet:  Java et C++
```

Sur le site Web :

```
http://www.boichat.ch/javacpp/
```

nous pourrons trouver d'éventuelles corrections découvertes dans les exemples et les exercices, ainsi que des remarques pertinentes ou des difficultés rencontrées par certains lecteurs. Nous aurons également à notre disposition des informations sur les nouvelles versions des compilateurs et les nouveaux outils, ainsi qu'une liste de sites et d'ouvrages intéressants et recommandés. Toutes ces informations seront régulièrement mises à jour par l'auteur.