前　言

为什么要写这本书

相比其他语言的频繁更新，C++ 语言标准已经有十多年没有真正更新过了。而上一次标准制定，正是面向对象概念开始盛行的时候。较之基于过程的编程语言，基于面向对象、泛型编程等概念的 C++ 无疑是非常先进的，而 C++98 标准的制定以及各种符合标准的编译器的出现，又在客观上推动了编程方法的革命。因此在接下来的很多年中，似乎人人都在学习并使用 C++。商业公司在邀请 C++ 专家为程序员讲课，学校里老师在为学生绘声绘色地讲解面向对象编程，C++ 的书籍市场也是百花齐放，论坛、BBS 的 C++ 板块则充斥了大量各种关于 C++ 的讨论。随之而来的，招聘启事写着“要求熟悉 C++ 编程”，派生与继承成为了面试官审视毕业生基础知识的重点。凡此种种，不一而足。于是 C++ 语言“病毒性”地蔓延到各种编程环境，成为了使用最为广泛的编程语言之一。

十来年的时光转瞬飞逝，各种编程语言也在快马加鞭地向前发展。如今流行的编程语言几乎无一不支持面向对象的概念。即使是古老的语言，也通过了制定新标准，开始支持面向对象编程。随着 Web 开发、移动开发逐渐盛行，一些新流行起来的编程语言，由于在应用的快速开发、调试、部署上有着独特的优势，逐渐成为了这些新领域中的主流。不过这并不意味着 C++ 正在失去其阵地。身为 C 的“后裔”，C++ 继承了 C 能够进行底层操作的特性，因此，使用 C/C++ 编写的程序往往具有更佳的运行时性能。在构建包括操作系统的各种软件层，以及构建一些对性能要求较高的应用程序时，C/C++ 往往是最佳选择。更一般地讲，即使是由其他语言编写的程序，往往也离不开由 C/C++ 编写的编译器、运行库、操作系统，或者虚拟机等提供支持。因此，C++ 已然成为了编程技术中的中流砥柱。如果用个比喻来形容 C++，那么可以说这十来年 C++ 正是由“锋芒毕露”的青年时期走向“成熟稳重”的中年时期。

不过十来年对于编程语言来说也是个很长的时间，长时间的沉寂甚至会让有的人认为，C++ 就是这样一种语言：特性稳定，性能出色，易于学习而难于精通。长时间使用 C++ 的程序员也都熟悉了 C++ 毛孔里每一个特性，甚至是现实上的一些细微的区别，比如各种编译器对 C++ 扩展的区别，也都熟稔于心。于是这个时候，C++11 标准的横空出世，以及 C++ 之父 Bjarne Stroustrup 的一句“看起来像一门新语言”的说法，无疑让很多 C++ 程序员有些诚惶诚恐：C++11 是否又带来了编程思维的革命？C++11 是否保持了对 C++98 及 C 的兼容？旧有的 C++ 程序到了 C++11 是否需要被推倒重来？

事实上这些担心都是多余的。相比于 C++98 带来的面向对象的革命性，C++11 带来的

却并非“翻天覆地”式的改变。很多时候，程序员保持着“C++98 式”的观点来看待 C++11 代码也同样是合理的。因为在编程思想上，C++11 依然遵从了一贯的面向对象的思想，并深入加强了泛型编程的支持。从我们的观察来看，C++11 更多的是对步入“成熟稳重”的中年时期的 C++ 的一种改造。比如，像 auto 类型推导这样的新特性，展现出的是语言的亲和力；而右值引用、移动语义的特性，则着重于改变一些使用 C++ 程序库时容易发生的性能不佳的状况。当然，C++11 中也有局部的创新，比如 lambda 函数的引入，以及原子类型的设计等，都体现了语言与时俱进的活力。语言的诸多方面都在 C++11 中再次被锤炼，从而变得更加合理、更加条理清晰、更加易用。C++11 对 C++ 语言改进的每一点，都呈现出了经过长时间技术沉淀的编程语言的特色与风采。所以从这个角度上看，学习 C++11 与 C++98 在思想上是一脉相承的，程序员可以用较小的代价对 C++ 的知识进行更新换代。而在现实中，只要修改少量已有代码（甚至不修改），就可以使用 C++11 编译器对旧有代码进行升级编译而获得新标准带来的好处，这也非常具有实用性。因此，从很多方面来看，C++ 程序员都应该乐于升级换代已有的知识，而学习及使用 C++11 也正是大势所趋。

在本书开始编写的时候，C++11 标准刚刚发布一年，而本书出版的时候，C++11 也只不过才诞生了两年。这一两年，各个编译器厂商或者组织都将支持 C++11 新特性作为了一项重要工作。不过由于 C++11 的语言特性非常的多，因此本书在接近完成时，依然没有一款编译器支持 C++11 所有的新特性。但从从业者的角度看，C++11 迟早会普及，也迟早会成为 C++ 程序员的首选，因此即使现阶段编译器对 C++ 新特性的支持还不充分，但还是有必要在这个时机推出一本全面介绍 C++11 新特性的中文图书。希望通过这样的图书，使得更多的中国程序员能够最快地了解 C++11 新语言标准的方方面面，并且使用最新的 C++11 编译器来从各方面提升自己编写的 C++ 程序。

读者对象

本书针对的对象是已经学习过 C++，并想进一步学习、了解 C++11 的程序员。这里我们假定读者已经具备了基本的 C++ 编程知识，并掌握了一定的 C++ 编程技巧（对于 C++ 的初学者来说，本书阅读起来会有一定的难度）。通过本书，读者可以全面而详细地了解 C++11 对 C++ 进行的改造。无论是试图进行更加精细的面向对象程序编写，或是更加容易地进行泛型编程，或是更加轻松地改造使用程序库等，读者都会发现 C++11 提供了更好的支持。

本书作者和书籍支持

本书的作者都是编译器行业的从业者，主要来自于 IBM XL 编译器中国开发团队。IBM XL 编译器中国开发团队创立于 2010 年，拥有编译器前端、后端、性能分析、测试等各方面的人员，工作职责涵盖了 IBM XL C/C++ 及 IBM XL Fortran 编译器的开发、测试、发布等与编译器产品相关的方方面面。虽然团队成立时间不长，成员却都拥有比较丰富的编译器开发经验，对 C++11 的新特性也有较好的理解。此外，IBM 北美编译器团队成员 Michael（他是 C++ 标准委员会的成员）也参加了本书的编写工作。在书籍的编写上，Michael 为本书

拟定了提纲、确定了章节主题，并直接编写了本书的首章。其余作者则分别对 C++11 各种新特性进行了详细研究讨论，并完成了书稿其余各章的撰写工作。在书稿完成后，除了请 Michael 为本书的部分章节进行了审阅并提出修改意见外，我们又邀请了 IBM 中国信息开发部及 IBM 北京编译器团队的一些成员对本书进行了详细的审阅。虽然在书籍的策划、编写、审阅上我们群策群力，尽了最大的努力，以保证书稿质量，不过由于 C++11 标准发布时间不长，理解上的偏差在所难免，因此本书也可能在特性描述中存在一些不尽如人意或者错误的地方，希望读者、同行等一一为我们指出纠正。我们也会通过博客（http://ibm.co/HK0GCx）、微博（www.weibo.com/ibmcompiler）发布与本书相关的所有信息，并与本书读者共同讨论、进步。

如何阅读本书

读者在书籍阅读中可能会发现，本书的一些章节对 C++ 基础知识要求较高，而某些特性很可能很难应用于自己的编程实践。这样的情况应该并不少见，但这并不是这门语言缺乏亲和力，或是读者缺失了背景知识，这诚然是由于 C++ 的高成熟度导致的。在 C++11 中，不少新特性都会局限于一些应用场景，比如说库的编写，而编写库却通常不是每个程序员必须的任务。为了避免这样的状况，本书第 1 章对 C++11 的语言新特性进行了分类，因此读者可以选择按需阅读，对不想了解的部分予以略过。一些本书的使用约定，读者也可以在第 1 章中找到。

致谢

在这里我们要对 IBM 中国信息开发部的陈晶（作者之一）、卢昉、付琳，以及 IBM 北京编译器团队的冯威、许小羽、王颖对本书书稿详尽细致的审阅表示感谢，同时也对他们专业的工作素养表示由衷的钦佩。此外，我们也要感谢 IBM XL 编译器中国开发团队的舒蓓、张嗣元两位经理在本书编写过程中给予的大力支持。而 IBM 图书社区的刘慎峰及华章图书的杨福川编辑的辛勤工作则保证了本书的顺利出版，在这里我们也要对他们以及负责初审工作的孙海亮编辑说声谢谢。此外，我们还要感谢各位作者的家人在书籍编写过程中给予作者的体谅与支持。最后要感谢的是本书的读者，感谢你们对本书的支持，希望通过这本书，我们能够一起进入 C++ 编程的新时代。

IBM XL 编译器中国开发团队

目　录