1. 直面C++和STL

这一章说了什么？

C++语言和标准模板库（STL）最重要组成部分和语法的简要概述

➤➤智能指针的基础知识

请注意，本章的所有代码示例都可作为本章的一部分

在本书的网站www.wrox代码下载。COM /去/ proc + 3E上下载代码标签。

本章的目的是简要涵盖C++中最重要的部分，你需要有一些基础的知识，然后再开始阅读本书其它内容。本章不是C++和STL的综合课程。一些基本的知识点，例如程序是什么、=和==的区别等，本书并没有涉及。某些深奥的点，如定义一个联合体（union），或volatile关键字，也省略了。某些属于C语言且与C++不相关的部分也不会出现在本书中，是C++的部分在后面的章节中会有深入的介绍。

本章旨在涵盖C++程序员每天都会遇到的那部分。举例来说，如果你有一段时间不使用C++了而且你忘了for循环的语法，你将会在这章里再次看到。另外，如果你刚开始学习C++，还不了解引用变量是什么，你也会在这一章里学习到。你也会学习到如何使用STL中的可用功能，如向量容器，字符串对象和智能指针。

If you already have significant experience with C++, skim this chapter to make sure that there  
aren’t any fundamental parts of the language on which you need to brush up. If you’re new to C++, read this chapter carefully and make sure you understand the examples. If you need additional  
introductory information, consult the titles listed in Appendix B.

THE BASICS OF C++  
The C++ language is often viewed as a “better C” or a “superset of C.” Many of the annoyances or  
rough edges of the C language were addressed when C++ was designed. Because C++ is based on  
C, much of the syntax you’ll see in this section will look familiar to you if you are an experienced  
C programmer. The two languages certainly have their differences, though. As evidence, *The*  
*C++ Programming Language* by C++ creator Bjarne Stroustrup (Fourth Edition; Addison-Wesley  
Professional, 2013), weighs in at 1,368 pages, while Kernighan and Ritchie’s *The C Programming*  
*Language* (Second Edition; Prentice Hall, 1988) is a scant 274 pages. So if you’re a C programmer,  
be on the look out for new or unfamiliar syntax!  
The Obligatory Hello, World  
In all its glory, the following code is the simplest C++ program you’re likely to encounter:  
// helloworld.cpp  
#include <iostream>  
int main()  
{  
std::cout << "Hello, World!" << std::endl;  
return 0;  
}  
This code, as you might expect, prints the message “Hello, World!” on the screen. It is a simple  
program and unlikely to win any awards, but it does exhibit the following important concepts about  
the format of a C++ program.  
➤➤ Comments  
➤➤ Preprocessor Directives  
➤➤ The main() Function  
➤➤ I/O Streams  
These concepts are briefly explained in the next sections.  
Comments  
The first line of the program is a *comment*, a message that exists for the programmer only and is  
ignored by the compiler. In C++, there are two ways to delineate a comment. In the preceding and  
following examples, two slashes indicate that whatever follows on that line is a comment.  
// helloworld.cpp  
The same behavior (this is to say, none) would be achieved by using a *multiline comment*. Multiline  
comments start with /\* and end with \*/. The following code shows a multiline comment in action  
(or, more appropriately, inaction).

/\* This is a multiline comment.  
The compiler will ignore it.  
\*/  
Comments are covered in detail in Chapter 3.  
Preprocessor Directives  
Building a C++ program is a three-step process. First, the code is run through a *preprocessor*,  
which recognizes meta-information about the code. Next, the code is *compiled*, or translated into  
machine-readable object files. Finally, the individual object files are *linked* together into a single  
application. Directives aimed at the preprocessor start with the # character, as in the line #include  
<iostream> in the previous example. In this case, an include directive tells the preprocessor  
to take everything from the <iostream> header file and make it available to the current file.  
The most common use of header files is to declare functions that will be defined elsewhere. A  
function *declaration* tells the compiler how a function is called, declaring the number and types of  
parameters, and the function return type. A *definition* contains the actual code for the function. In  
C++, declarations usually go into files with extension .h, known as *header files*, while definitions  
usually go into files with extension .cpp, known as *source files*. A lot of other programming  
languages do not separate declarations and definitions into separate files, for example C# and Java.  
The <iostream> header declares the input and output mechanisms provided by C++. If the program  
did not include it, it would be unable to perform its only task of outputting text.