Microsoft Visual C++ 5.0程序员参考·第4卷

Microsoft Visual C++语言参考手册

（美）Microsoft Corporation 著
前导工作室 译

清华大学出版社

(京)新登字 158 号

内 容 提 要

本书是 Microsoft Visual C++ 程序员参考系列手册(共四册)之四。该系列手册主要面向 C 和 C++ 程序员,为他们提供 MFC 类库、C 语言函数等方面的参考。

本书分为三个部分:第一部分介绍 C 语言,包括术语、概念、程序的结构以及函数等内容;第二部分介绍 C++ 语言,主要介绍运行类型信息(RTTI)和名字空间的有关知识;第三部分介绍预处理的相关知识。

Microsoft Visual C++ 语言参考手册
Microsoft Visual C++ Language Reference
Microsoft Corporation

北京市版权局著作权合同登记号 图字 01-98-2221 号

图书在版编目(CIP)数据

Microsoft Visual C++ 语言参考手册/(美)Microsoft Corporation 著;前导工作室译 .—北京:清华大学出版社,1998.12
(Microsoft Visual C++ 5.0 程序员参考;第 4 卷)
ISBN 7-302-03253-X

Ⅰ. M… Ⅱ. ①M… ②前… Ⅲ. C 语言-程序设计-手册 Ⅳ. TP312-62

中国版本图书馆 CIP 数据核字 (98) 第 35322 号

出 版 者:清华大学出版社(北京清华大学校内,邮编 100084)
http://www.tup.tsinghua.edu.cn
责任编辑:胡先福
印 刷 者:清华大学印刷厂印刷
发 行 者:新华书店总店北京发行所
开 本:787 × 1092 1/16 印张:37.5 字数:935 千字
版 次:1999 年 2 月第 1 版 1999 年 2 月第 1 次印刷
书 号:ISBN 7-302-03253-X/TP·1742
印 数:0001 ~ 5000
定 价:69.00 元

译 者 序

C和C++语言是非常流行的编程语言，它们以高效率、低开销和良好的可移植性而闻名。用C/C++语言编写的程序速度快、结构紧凑，并且易于与系统接口。因此，C/C++语言很适合于用来进行系统编程。

Microsoft Visual C++程序员参考系列手册共包括四本书，它们是微软公司推出的Microsoft Visual C++ 5.0的完整文档。本书是这四册书中的第四卷，主要目的是向程序员介绍C/C++语言。全书分为以下三部分：第一部分介绍C语言，包括术语、概念、程序的结构和函数等内容；第二部分介绍C++语言，主要介绍运行类型信息(RTTI)和名字空间的有关知识；第三部分介绍预处理的相关知识。

本书由况军组织翻译，参加翻译工作的还有陈彦海、刘辉、孙议、张志敏、刘国庆、汤朝阳、史章军、刘汇、王建民、潇东等。全书由潇东、李士心进行校对，由陈安安录排。

由于时间仓促，且经验和水平有限，书中难免会有不足之处，恳请读者批评指正。

译 者

1998年4月

目　录

第一部分　C语言参考手册

第二部分　C++语言参考手册

第三部分　预处理器参考手册

第一部分

C 语言参考手册

引　言

0.1　本手册的组织

- C 的语言成分
- 程序的结构
- 声明和类型
- 表达式和赋值
- 语句
- 函数
- C 语言语法总结
- 实现定义的行为

0.2　本手册的范围

C 是一种十分灵活的编程语言，它将编程中的许多决定交给程序员去做。由于具有这种特性，C 语言中诸如数据类型转换这样的限制比较少。虽然这种特性可以使程序员的编程工作更容易，但程序员必须十分了解该语言才能理解程序将怎样运行。本部分提供 C 语言成份的信息和 Micorosoft 实现的特性。C 语言语法源于 ANSI X3.159-1989, American National Standard for Information Systems-Programming Language-C(以下称为 ANSI C 标准)，尽管这不是 ANSI C 标准的一部分。附录 A“C 语言语法总结”将介绍该语法，并说明如何阅读和使用该语法的定义。

本部分不讨论 C++ 语言的编程。有关 C++ 的内容请参考《C++ 语言参考手册》部分。

注意　有关 Microsoft 产品的支持，参见文件 PSS.HLP。

0.3　ANSI 约定

Microsoft C 符合 ANSI C 标准所述的 C 语言标准。和联机参考一样，本书中 Microsoft 对 ANSI C 标准的扩充将用文字和语法注明。由于这些扩充不是 ANSI C 标准的一部分，所以它们的使用可能会限制程序在不同系统中的可移植性。在默认情况下允许使用 Microsoft 所做的扩充。若不想使用扩充，应指定编译选项/Za。使用/Za 时，所有非 ANSI 代码都将产生错误或警告。

第 1 章　C 的语言成分

本章描述 C 编程语言的成分，包括组成 C 程序的名字、数字和字符。ANSI C 语法将这些组成部分记作“标记(token)”。本章介绍如何定义标记以及编译器如何解释这些标记。

本章将讨论如下的内容：

- 标记
- 注释
- 关键字
- 标识符
- 常量
- 字符串
- 标点符号与特殊字符

本章还包括三连符(trigraph)、浮点常数、整常数和转义符序列的参照表。

“运算符”是指定如何运算的符号(包括单一字符和字符组合)。每一个符号都被编译为一个单一的单位，称为标记。要了解详细的信息，请参见第 4 章中的“运算符”。

1.1 标　记

在 C 源程序中，被编译器识别的最基本的成分是“标记(token)”。标记是源程序文本中编译器不可再分割的基本成分。

语法

token：

keyword
identifier
constant
string-literal
operator
punctuator

注意　有关 ANSI 语法规则的解释，参见附录 A 中的“C 语言语法概述”。

本章所描述的关键字、标识符、常量、字符串和运算符都是标记的实例。标点字符，如方括号([])、花括号({})、圆括号(())和逗号(,)也都是标记。

1.1.1 空白字符

空格、制表符、换行、回车、换页、垂直制表符和新行字符(newline character)被称为“空白

字符",因为这些字符都用于同一目的即在字间和行间插入空间以便于阅读。标记由空白字符或其他标记,如运算符和标点进行分隔。当编译器分析代码时,忽略空白字符,除非它们作为分隔符或者作为字符常量或字符串常量的组成部分。使用空白字符可使程序更易于阅读。

注意 编译器把注释当作空白。

1.1.2 注释

"注释"是以斜线和星号的组合(/*)开始的字符序列,被编译器解释为单个空白字符,且注释被编译器所忽略。注释包括可表示字符集的任何字符组合,包括换行符,但注释结束符(*/)除外。注释可占多于一行,但不能嵌套。

注释可以出现在任何允许空白字符出现的地方。因为编译器把注释当作单个空白字符,所以标记中不能包括注释。编译器忽略注释中的所有字符。

使用注释可以为源代码增加文档。下面的例子是编译器可以接受的注释:

```
/* Comments can contain keywords such as
   for and while without generating errors. */
```

注释可以和源代码语句出现在同一行:

```
printf("hello \ n";/* comments can go here */
```

可以在函数或程序模块前使用注释作描述说明:

```
/* MATHERR.C illustrates writing an error routine
 * for math functions .
 */
```

因为注释不能嵌套,所以下面的例子将导致出错:

```
/*   Comment out this routine for testing
   /* Open file */
   fh = _open("myfile. c", _o_RDONLY)
        .
        .
        .
*/
```

错误的发生是因为编译器在识别出 Open file 之后的第一个 */之后,就把它当作注释的结束了。在处理其后的文字中,当注释之外出现 */就会产生错误。

在测试时,可以使用注释把几行代码设置为非活动状态。预处理指令 # if 和 # endif 与条件编译指令是另一种实现这种功能的方法。有关的内容参见"预处理参考"中的"预处理指令"。

Microsoft 特性→

Microsoft 的编译器还支持由两条斜线(//)开始的单行注释。但如果编译时使用了/Za(ANSI 标准)选项,那么这类注释会导致出错。这种类型的注释不能延续到下一行。

```
// This is valid comment
```

由两条斜线(//)引出的注释以回车符作为结束,但该回车符前不能有转义字符。下例中的回车符前有一条反斜线(\ ),因而构成"转义序列"。这一转义序列使编译器把回车符当作注释行的一部分(详细的内容参见"转义序列")。

```
//my comment \
i++;
```

语句 i++;被认为是注释而忽略。

默认时 Microsoft C 扩展有效。用/Za 可禁止该扩展。

Microsoft 特性结束

1.1.3 标记的求值

当编译器解释标记时,会在一个标记中包括尽可能多的字符,然后再扫描下一个标记。由于这一特性,如果标记没有用空白进行适当的分隔,那么编译器可能不会像你想象的那样解释标记。考虑如下表达式:

```
i+++j
```

在这个例子中,编译器从三个加号中先找出尽可能长的运算符(++),然后再把剩下的加号作为加法运算符。于是这个表达式就被编译为(i++) + (j),而不是(i) + (++j)。在这个例子以及类似的情况下,应该使用空白和圆括号来避免意义不明确,并保证表达式正确地求值。

Microsoft 特性→

C 编译器把 Ctrl + Z 字符当作文件结束符。在 Ctrl + Z 之后的文字均被忽略。

Microsoft 特性结束

1.2 关 键 字

"关键字"是对 C 编译器具有特殊意义的词。编译器在第 7 和第 8 遍扫描中,标识符不得与 C 关键字有相同的拼法和大小写(参见《预处理器参考》中的有关"编译扫描(translation phase)"的描述),C 语言使用如下关键字:

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

关键词不能重新定义,但可以通过在编译前使用 C 预处理指令说明替代关键字的

文本。

Microsoft 特性→

ANSI C 标准允许以两个下划线开始的标识符作为编译器保留的关键字。Microsoft 约定 Microsoft 特有的关键字均以双下划线开始，它们也不能作为标识符名。要了解 ANSI 关于命名标识符的规则，包括双下划线的使用，请参见 1.3 节的“标识符”。

Microsoft C 编译器识别以下的关键字和特殊标识符：

__asm	dllimport[2]	__int8	naked[2]
__based[1]	__except	__int16	__stdcall
__cdecl	__fastcall	__int32	thread[2]
__declspec	__finally	__int64	__try
dllexport[2]	__inline	__leave	

[1]. 关键字 __based 仅限于 32 位目标的编译。

[2]. 当与 __declspec 一起使用时是特殊标识符；在其他环境中使用时没有限制。

Microsoft 扩展在默认时是允许的。如果要保证程序的可移植性，那么可以在编译时使用/Za 选项禁止 Microsoft 扩展(按照与 ANSI 兼容的方式编译)。当你这样做时，Microsoft 的关键字被禁止使用。

当允许使用 Microsoft 扩展时，程序可以使用上面列出的关键字。为服从 ANSI 的规则，这些关键字多数是以双下划线开头的，只有四个例外：dllexport，dllimport，naked 和 thread。而这四个例外关键字只能与 __declspec 共同使用，因此无需以双下划线开头。为了向后兼容，其余的单下划线关键字仍然被支持。

Microsoft 特性结束

1.3 标 识 符

“标识符”或“符号”是程序员为程序中变量、类型、函数和标号提供的名字。标识符不能与关键字在拼法和大小写上相同。关键字(不论是标准 C 或 Microsoft C)不能用作标识符，它们被保留用于特殊的目的。通过在变量、类型、函数的声明中进行说明可以创建标识符。在这个例子中，result 是一个整型变量标识符，main 和 printf 是函数的标识符。

```
void main()
{
        int result;
        if ( result!  =0 )
          printf( "Bad file handle \ n" )
}
```

标识符一旦被声明就可以在其后的程序语句中引用。

有一种特殊的标识符，称为语句标号，可以用在 goto 语句中(声明描述见第 3 章“声明和类型”。语句标号见第 5 章 5.8 节“goto 和带标号的语句”)。