

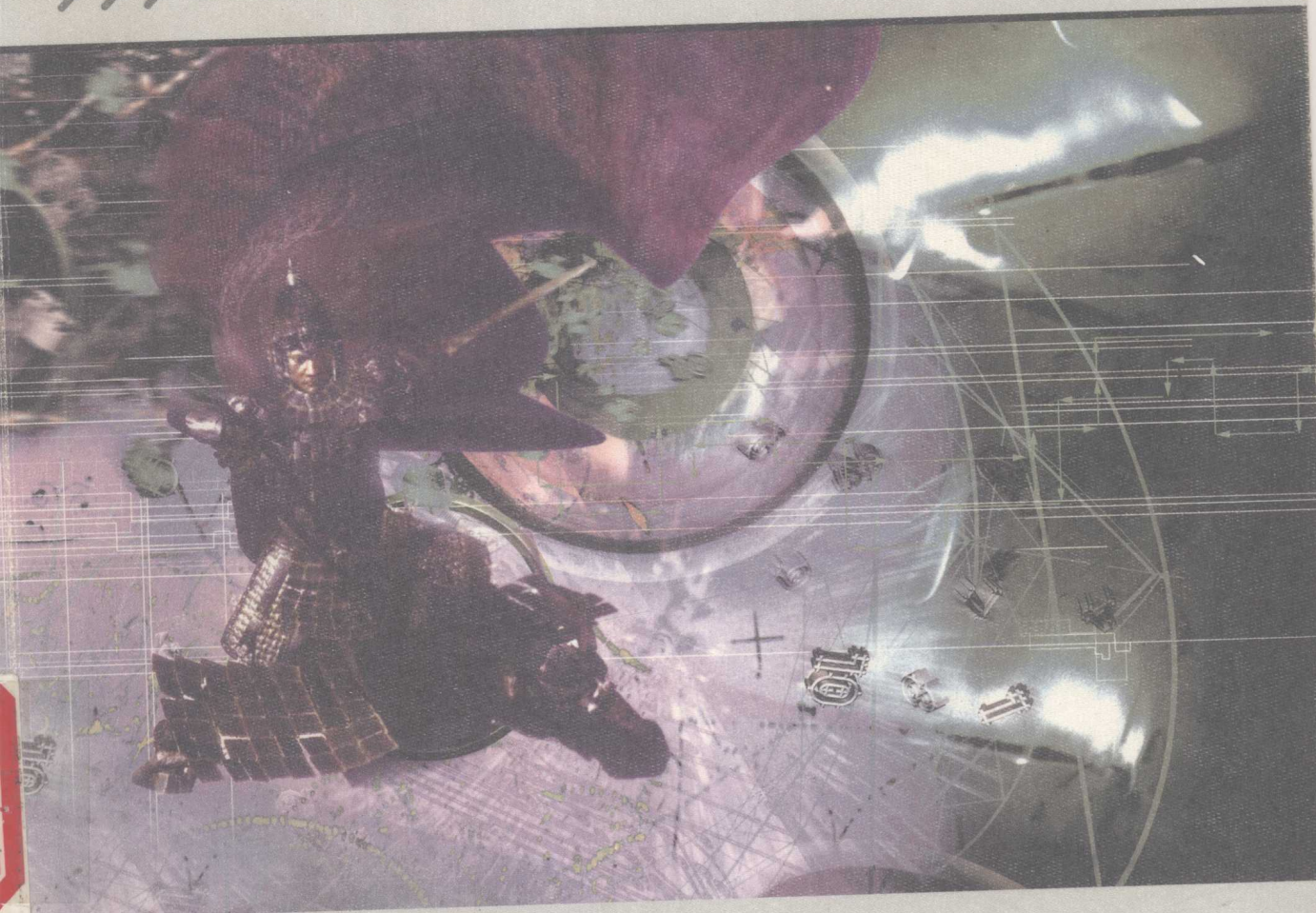
快乐写游戏
轻松学编程

PC

游戏编程 (DirectX 8)

PC Youxi Biancheng

>>> CG实验室 甘泉 编著



清华大学出版社
重庆大学出版社

快乐写游戏 轻松学编程

PC 游戏编程(DirectX 8)

编 著 CG 实验室 甘泉

清华大学出版社
重庆大学出版社

内 容 简 介

· 1 ·

本书针对一般 PC 游戏编程的过程,从游戏工程的组织(第 1 章)、DirectDraw 编程(第 2 章)、DirectInput 编程(第 3 章)、DirectSound 和 DirectMusic 编程(第 4 章)等方面,循序渐进地对 DirectX 游戏编程的细节进行介绍说明,并附有大量表格和图片加以辅助理解;初步介绍有关 Direct3D 的部分内容(第 5 章),以引导读者进一步地思考和实践。考虑到有些初学者对直接在 Windows 下进行 C 语言编程还不是很熟悉,本书的配套光盘中安排了有关 Windows 下 C 语言编程的基础教程。

本书适用于具有 C 语言基础的初级和中级程序员,以及游戏编程爱好者。

图书在版编目(CIP)数据

PC 游戏编程(DirectX 8)/CG 实验室编著. —重庆:重庆大学出版社,2003.6

(快乐写游戏——轻松学编程)

ISBN 7-5624-2943-X

I. ①PC… II. ①CG… III. ①游戏—应用程序—程序设计—Windows99

中国版本图书馆 CIP 数据核字(2003)第 045886 号

快乐写游戏——轻松学编程

PC 游戏编程(DirectX 8)

CG 实验室——CG 设计

责任编辑:王海琼 雷 航 刘国良 谭 华 版式设计:陈 其

责任校对:蓝安梅 责任印制:秦 梅

*

清华大学出版社 出版发行
重庆大学出版社

出版人:张鹤盛

社址:重庆市沙坪坝正街 174 号重庆大学(A 区)内

邮编:400030

电话:(023) 65102378 65105781

传真:(023) 65103686 65105565

网址:<http://www.cqup.com.cn>

邮箱:fxk@cqup.com.cn (市场营销部)

全国新华书店经销

重庆科情印务有限公司印刷

*

开本:787×1092 1/16 印张:17.25 字数:431 千

2003 年 6 月第 1 版 2003 年 6 月第 1 次印刷

印数:1—5 000

ISBN 7-5624-2943-X/TP·421 定价:38.50 元(赠 1CD)

本书如有印刷、装订等质量问题,本社负责调换

版权所有 翻印必究

总 序 言

陈 其

《快乐写游戏 轻松学编程》丛书是重庆大学出版社为广大计算机编程爱好者和电脑游戏玩家送上的一份厚礼,是一套集学习、娱乐于一体的,全新教授模式的好书。全套书由陈其总策划,在多维图书策划中心以及各游戏工作室的鼎力协助下得以顺利出版。现就丛书的有关问题作出说明。

编程和游戏

程序是计算机的灵魂,掌握了编程技术就可以随心所欲地让计算机为你服务,让它实现你的梦想。但学习过程中大量的命令和语句又让人感到枯燥乏味,而每一个学编程的人都有过面对一大堆熟悉的命令却组织不起一个像样的程序的经历。于是我们联想到了一种让很多朋友都着迷的程序——电子游戏。

1) 第9艺术

电子游戏如同戏剧、电影一样,是一种综合艺术,并且是更高层次的综合艺术,它的出现代表了一种全新的娱乐方式——交互式娱乐(Interactive Entertainment)的诞生,而且从它的诞生到现在一直以其独特的魅力吸引了许多玩家,同时也激发了更多的人想写游戏的愿望。

一种事物,当它具有丰富而独特的表现力时,当它能给人们带来由衷的欢愉时,当它表现为许许多多鲜明生动的形象时,它就是一种艺术。电子游戏已经成为一门艺术,即继绘画、雕刻、建筑、音乐、诗歌(文学)、舞蹈、戏剧、电影(影视艺术)之后人类历史上的第9艺术。20世纪70年代,出现了第一批简单的电子游戏;今天,它已经发展成为拥有亿万游戏迷的独立的新型艺术样式,向世人显示了其强大的艺术生命力。《文明》、《Doom》、《魔法门》……一个又一个奇迹在产生,进入这个行业成了很多人的梦想。娱乐界的大腕:卢卡斯、派拉蒙、华纳等都已致力于电子游戏产品的开发,并推出了一大批优秀的交互式电影(Interactive Movie)。在世界范围内,电子游戏业的利润已经超过了美国的电影工业和日本的汽车工业。相信不久的将来必然有一大批杰出的电影导演和真正的艺术家投身于电子游戏艺术作品的开发。而VR头盔与3D音效卡的诞生已使电子游戏远远跳出了一般电影所能达到的视听层次。可见,电子游戏已经将视听综合艺术推向了一个崭新的高度和崭新的领域。

在中国,电子游戏曾一度被称为是“电子海洛因”,一些教育界人士痛斥电子游戏是如何

毒害青少年,如何损害人的健康。其实与其千方百计扼杀它,还不如共同想办法来扬其长、避其短。因为绝大多数反对电子游戏的人,并不是反对电子游戏本身,而是反对电子游戏中存在的消极面。正如水能载舟亦能覆舟的道理一样,任何事物都有其两面性,关键是怎样利用好的那一面为人类造福。

本丛书正是要利用电子游戏的积极面,将枯燥的学习融入轻松的游戏之中,达到寓教于乐的目的。

2) 培养全局观

许多刚学编程的朋友总是把大量的精力花在了命令和语句上,或是集中精力去学习那些复杂的函数。他们都忽略了怎样去实现一个完整的程序,所以有很多初学者到现在还没写过一个完整的程序。为了避免这种情况,在编写游戏实例时,各书都使用了简单而功能强大的游戏开发引擎,读者能非常轻松地学会如何显示图像文件、播放声音及控制输入设备等游戏中必备的功能,然后把注意力集中到如何实现一个完整游戏的过程及原理上来。

通过细致的讲解,读者朋友很快就能从实例中体会出程序全局观的作用和地位,并在一步的学习后掌握它。

编程工具

作为一名程序员,要做的第一件事就是选择一件顺手的武器——编程工具。做程序的朋友都知道,比较流行的编程工具颇多,比如:VC,VB,DEPHI、汇编等等。由于本丛书是从编写游戏出发的,而为了能够完成一个完美的游戏,编程工具应具有贴近底层、代码运行速度快、便于优化等优点。于是 VC 成了不二之选。

初学 VC,会因为观念的改变而不知所措。其实,每个人时刻都面临着新知识的学习和旧知识的更新。这就好比,只有踏出新的一步才能前进。那么如何才能更快地学会程序(游戏)开发呢?很简单,那就是“边学边做”!所谓知识来源于实践,做做学学,学学做做,这样你很容易就能融会贯通了。所以,首先了解一些 VC 使用常识,照着书中的一些简单的例子一步一步地实际操作,从中学会一些基本的游戏开发常识。然后学习一些 C++ 理论知识,选一些难一点的例子来学。之后再学一些游戏开发的高级技术,试着自己开发一个游戏出来。罗马不是一天建成的,饭不是一口吃得完的,游戏也不是一会儿就能做出来的。所以,每天砌一块砖,不久一座美丽的城堡就矗立在你面前了。

衷心祝愿每位读者能在本丛书中吸收到有用的知识。

前言

最早的游戏应该追溯到 20 世纪 60 年代在 Unix 机器上运行的 Core Wars。随着计算机硬件的飞速发展,多媒体技术和游戏编程逐渐走向成熟。现在运用各种开发工具,自己参与进行游戏设计已经不是什么难事。

Microsoft 公司提供的 DirectX 是目前比较普遍的 Windows 游戏设计开发工具(SDK)。它基本包括了2D、3D、声音、控制、网络等的基本操作函数,使程序员不必直接与硬件打交道,大大地提高了游戏开发的简易性和效率。

延续“快乐写游戏,轻松学编程”PC 游戏编程丛书的总体风格,本书主要运用 C 语言。因为 DirectX SDK 已经提供了大量的函数,只要加上最基本的 Windows 构架,就可以提供一个比较宽松的开发框架,这也使得一些 C 程序员不必对 C++ 的某些内容做详细了解。当然,如果把它作为从 C 到 C++ 的一个参考,也是不错的主意。

书中所有的程序都是使用 Microsoft Visual C++ 6.0 编译器编译。在此顺便提醒:VC++6.0 只是一个编译工具,它并不是语言;C 或 C++ 是指一门语言;VC++6.0 既可以编译 C++ 程序,也可以编译 C 程序。故不要一看到 VC 程序就觉得很难。

书中以一个小游戏为主线,从它的构建开始,逐步完善图像、操作、声音等功能,每个步骤都有详细的说明,基本包括了主要的 DirectX 组件的详细说明。

作为实例,书中的精灵图案使用的是现在比较流行的游戏——疯狂坦克的图标。这个游戏比较有趣,可以作为网络游戏的一个代表来参考。

拓智文化 CG 实验室网站 www.cglab.info 上有关于本书和 DirectX 的精彩内容板块,大家可以访问这个网站来探讨交流有关问题!

非常感谢重庆大学出版社陈其先生、重庆拓智文化发展有限公司陈治刚先生和其他对本书编写和出版鼎力支持的朋友们!

感谢一直支持我们的家人、朋友,他们是我们坚持的动力!

甘泉

2003 年 3 月

目 录

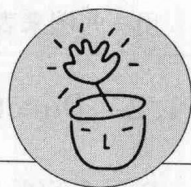
第 1 章 DirectX 游戏编程基础	1
1.1 DirectX 基础	2
1.1.1 Microsoft 和 DirectX	2
1.1.2 DirectX SDK 和 DirectX 组件	2
1.2 一个简单的游戏代码	5
1.3 组织一个游戏工程	25
1.3.1 游戏工程的文件组织	25
1.3.2 工程的建立	26
1.3.3 工程框架程序	28
第 2 章 游戏编程——DirectDraw 程序开发	31
2.1 创建 DirectDraw 界面	32
2.1.1 界面简介	32
2.1.2 创建 DirectDraw 对象	37
2.1.3 DirectDraw 错误处理	37
2.1.4 对 DirectDraw 对象的查询	38
2.1.5 设置游戏的协同等级	39
2.1.6 游戏的显示模式	39
2.1.7 退出游戏时的清除工作	40
2.1.8 创建游戏主画面	40
2.1.9 选用调色板	47
2.1.10 画图	48
2.2 在游戏中加入色彩和调色板	54
2.2.1 256 色和调色板	54
2.2.2 16 位增强色	55
2.2.3 24 位真彩色	56
2.2.4 32 位真彩色	57
2.3 让游戏画面动起来	57
2.3.1 双缓冲	57
2.3.2 创建离屏画面	61
2.3.3 页交换产生动画	66
2.3.4 创建图形变换器	69
2.3.5 画面数据交换	78
2.4 对画面进行剪切	79
2.4.1 对像素进行剪切	79
2.4.2 对位图进行剪切	82
2.4.3 DirectX 的剪切	86
2.5 在游戏中加入位图	93
2.5.1 位图格式	93

2.5.2	编程读取位图的方法	95
2.5.3	游戏位图的加载	100
2.5.4	创建备用画面	104
2.5.5	色彩关键字	110
2.5.6	进行图形变换	118
2.5.7	旋转和缩放	126
2.6	更进一步:窗口中的 DirectDraw	128
2.6.1	全屏的差异	128
2.6.2	用户区	130
2.7	色彩效果初步	133
第3章	控制精灵——获取 DirectInput 数据	137
3.1	游戏的输入循环	138
3.1.1	游戏的输入设备	138
3.1.2	游戏的输入循环	140
3.2	获取键盘输入	141
3.2.1	移动精灵实例	141
3.2.2	创建 DirectInput 对象	147
3.2.3	创建键盘设备	148
3.2.4	设置协作等级	149
3.2.5	设置数据格式	150
3.2.6	获取键盘	150
3.2.7	重载数据	150
3.2.8	数据处理	151
3.3	获取鼠标输入	151
3.3.1	鼠标的按键	151
3.3.2	鼠标的位置	158
3.4	游戏摇杆和力反馈	161
3.4.1	游戏摇杆	161
3.4.2	力反馈	163
第4章	加入声音——DirectSound 和 DirectMusic	167
4.1	声音基础	168
4.1.1	声音的产生	168
4.1.2	声音的特性	168
4.2	声音文件 WAV 和 MIDI	168
4.2.1	WAV 文件	168
4.2.2	MIDI 文件	170
4.3	DirectSound 应用	171
4.3.1	创建 DirectSound 对象	182
4.3.2	DirectSound 协作等级	183

4.3.3	DirectSound 声音缓冲	183
4.3.4	读取 WAV 文件	188
4.3.5	数据的写入与播放	191
4.4	DirectMusic 应用	193
4.4.1	初始化接口	208
4.4.2	创建平台	208
4.4.3	加入端口	209
4.4.4	创建读取器	210
4.4.5	加载 MIDI 文件	210
4.4.6	播放和停止	212
4.4.7	关闭 DirectMusic	214
第 5 章	3D 游戏初步	215
5.1	DirectSound3D 组件	216
5.1.1	音量和声道控制	216
5.1.2	DirectSound3D 的组成	220
5.1.3	DirectSound3D 缓冲区	221
5.1.4	DirectSound3D 接收器	223
5.2	Direct3D 基础	226
5.2.1	Direct3D 的立即模式	226
5.2.2	Direct3D 几何学基础	227
5.2.3	Direct3D 的对象和接口	236
5.2.4	Direct3D 灯光	246
5.2.5	Direct3D 纹理	255
5.2.6	结束语	263

DirectX 游戏编程基础

第1章



本章将介绍有关 DirectX 的基础知识。为了简单起见,这里就不涉及到有关 COM 的知识了。不过了解一些还是有帮助的,希望读者能够查阅其他的资料。

另外,本章给出书中的主线——一个简单的 DirectX 游戏的所有代码。当然,想要直接编译这些代码还需要一些操作步骤,在后面会谈到的。此外由于本书后面的章节不会深入地谈到有关游戏的人工智能和算法优化的问题,所以这方面的资料也需要读者自行查阅相关的书籍和文献。

本章的最后会详细地讨论游戏工程的组织,同时也可以说是建立自己的游戏引擎的基础。

1.1 DirectX 基础

1.1.1 Microsoft 和 DirectX

Microsoft 公司提供的 DirectX 是目前比较普遍的 Windows 游戏设计开发工具 (SDK)。它基本包括了 2D、3D、声音、控制、网络等的基本操作函数,使得程序员不必直接与硬件打交道,大大地提高了游戏开发的简易性和效率。

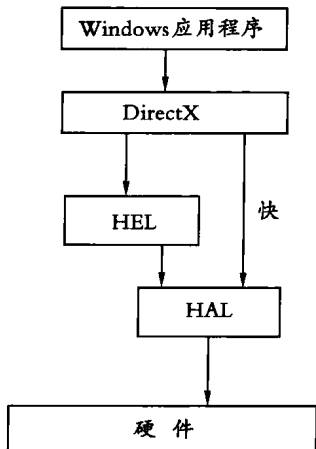


图 1.1 DirectX 和应用程序的关系

相对于 Windows GDI, DirectX 有很多优越性。DirectX 介于硬件和 Windows 应用程序之间,能够主动探测硬件的性能,当可以用硬件完成时,就直接通过硬件工作;如果硬件不支持,就通过软件模拟来实现。用术语表述就是:当硬件能够实现需要的功能时,使用 DirectX 的硬件抽象层 (HAL) 完成;如果硬件不能实现需要的功能时,使用 DirectX 的硬件模拟层 (HEL) 完成。很明显,直接使用硬件要比软件模拟要快得多。

现在读者就可以明白为什么很多游戏需要比较高的硬件配置。但是,也有这么一种情况,就是当一项硬件指标不能达到预想的要求时,可以通过另外一个硬件来弥补。比如大内存就可以弥补少量的显存不足或 CPU 频率的不足,当然,也不能相差太远。

用一张图来表示 DirectX 和应用程序的关系,见图 1.1。

1.1.2 DirectX SDK 和 DirectX 组件

DirectX SDK 是 DirectX 的开发包,包括了相关的开发组件。在进行游戏编程之前,需要安装 DirectX 8.0 SDK。DirectX 8.1 SDK 也可以,因为 DirectX SDK 是向下兼容的。本书中用的是 DirectX 8.1 SDK。



几乎所有近期的 DirectX 版本都包含以下几个组件:

- ①DirectDraw: 视频和 2D 图形控制组件。
- ②DirectInput: 输入设备控制组件。
- ③DirectSound: 数字声音控制组件。
- ④DirectMusic: MIDI 控制组件。
- ⑤DirectSound3D: 3D 声音控制组件。
- ⑥Direct3D: 3D 控制组件。
- ⑦DirectPlay: 网络控制组件。
- ⑧DirectSetup: DirectX 安装控制组件。

如果打开 DirectX 8.1 SDK 的 Help 文件,对有关 DirectX 组件的介绍翻译过来就是:

- ①DirectX Graphics: 包含 DirectDraw 和 Direct3D。
- ②DirectX Audio: 包含 DirectSound 和 DirectMusic。
- ③DirectInput。
- ④DirectPlay。
- ⑤DirectShow: 提供高质量的多媒体数据流。
- ⑥DirectSetup。

每个组件的具体功能在后面的章节逐一说明。下面,来学习一个游戏的全部代码。当然,也可以跳过,因为在后面的章节中会一步步地完成它,但是有关算法和人工智能就不详细说明了。

 注意:安装了 DirectX SDK 后,要把相应的头文件目录放到 VC 相应的目录选项中,否则 DirectX SDK 在 VC 中是不起作用的。

具体步骤如下:

在 VC 的 Tools 菜单中选中 Options 选项,见图 1.2。

在弹出的对话框中,选择 Directories 选项,就会出现一个对话框(见图 1.3),在右边的下拉框中显示为 Include files,点击下面的新增按键(就是下方的文件夹状的图标),选择 DirectX SDK 安装目录中的 Include 文件夹(比如 DirectX SDK 安装在 F 盘的 DXSDK 文件夹,那么 Include 文件夹的路径就是:F:\DXSDK\Include),并用边上的上移按键(就是左边的向上的箭头图标)将它置于顶部。

然后通过右边的下拉框来选择 Library files 选项,同样,将 F:\DXSDK\Lib 文件夹加入并置于顶部,见图 1.4。

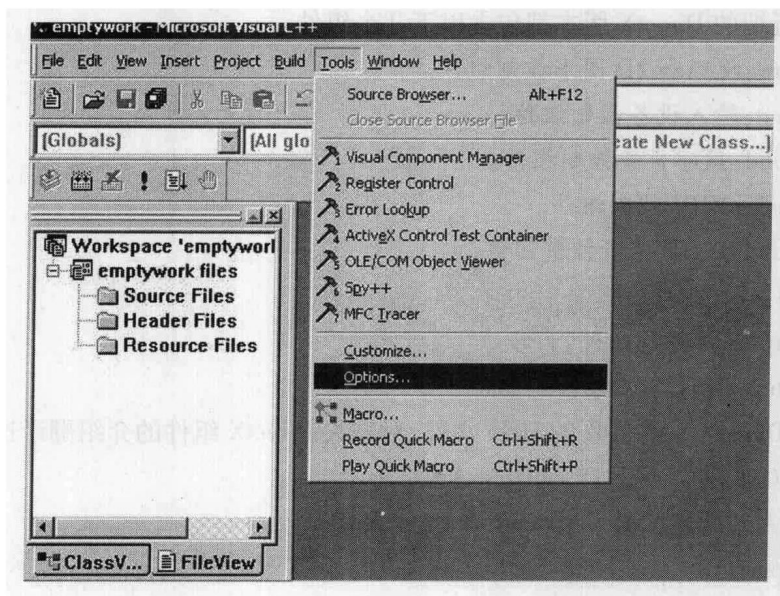


图 1.2 VC 的 Tools 菜单中 Options 选项

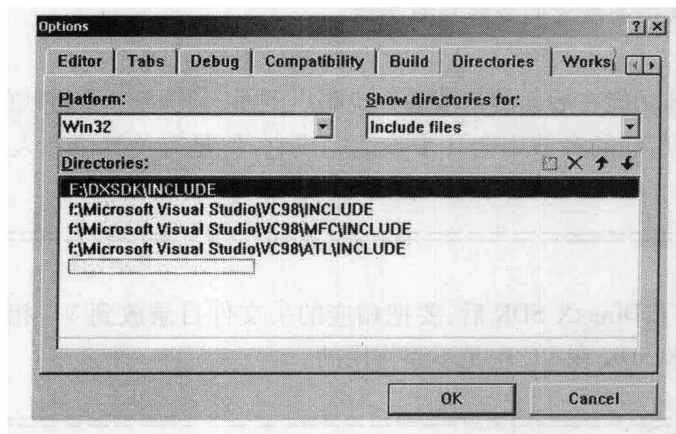


图 1.3 Directories 选项

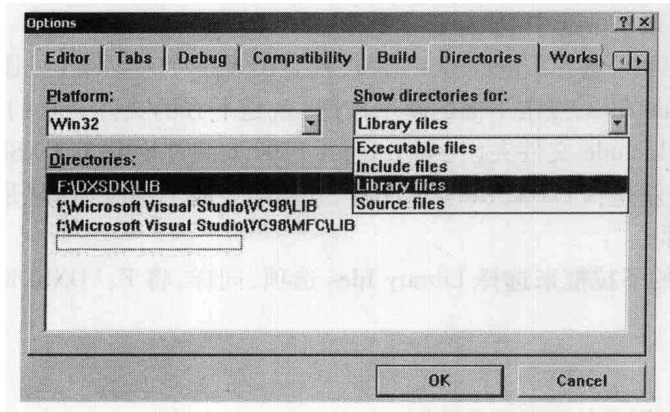


图 1.4 Library files 选项



1.2 一个简单的游戏代码

这是一个简单的射击游戏,按鼠标左键开始,按 Esc 键可以随时退出。

Windows 框架文件 WinMain. cpp,这个文件用于生成应用程序框架。

```
#include <windows.h> // include important windows stuff
#include <windowsx.h>
#include <mmsystem.h>
#include <iostream.h> // include important C/C++ stuff
#include <conio.h>
#include <stdlib.h>
#include <malloc.h>
#include <memory.h>
#include <string.h>
#include <stdarg.h>
#include <stdio.h>
#include <math.h>
#include <io.h>
#include <fcntl.h>
#include "..\H\GameMain.h"

//全局变量
HWND main_window_handle = NULL; //主窗口全局变量
HINSTANCE main_instance = NULL; //句柄全局变量
char buffer[80]; //临时文本数组
LRESULT CALLBACK WndProc( HWND, UINT, WPARAM, LPARAM );
int WINAPI WinMain( HINSTANCE hInstance, HINSTANCE hPrevInstance,
                    LPSTR lpCmdLine, int nCmdShow )
{
    static TCHAR szAppName[ ] = TEXT( " " );
    static TCHAR szClassName[ ] = TEXT( " MainWinClass" );
    HWND hwnd;
    MSG msg;
    WNDCLASS wndclass;
    wndclass.style = CS_DBLCLKS |
                    CS_OWNDC |
                    CS_HREDRAW |
                    CS_VREDRAW;
```

```
wndclass.lpfWndProc = WndProc;
wndclass.cbClsExtra = 0;
wndclass.cbWndExtra = 0;
wndclass.hInstance = hInstance;
wndclass.hIcon = LoadIcon(NULL, IDI_APPLICATION);
wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);
wndclass.hbrBackground = (HBRUSH) GetStockObject(BLACK_BRUSH);
wndclass.lpszMenuName = NULL;
wndclass.lpszClassName = szClassName;
//设置句柄全局变量
main_instance = hInstance;
if(! RegisterClass(&wndclass))
{
    MessageBox(NULL, TEXT("This program requires Windows NT !"),
        szAppName, MB_ICONERROR);
    return (0);
}

hwnd = CreateWindow(szClassName,
    TEXT(""),
    WS_POPUP | WS_VISIBLE,
    0,
    0,
    320,
    240,
    NULL,
    NULL,
    hInstance,
    NULL);

ShowWindow(hwnd, nCmdShow);
UpdateWindow(hwnd);
//设置主窗口全局变量
main_window_handle = hwnd;
//游戏初始化
GameInit();
//主循环
while(TRUE)
{
    //检测消息
```




```

if ( PeekMessage( &msg, NULL, 0, 0, PM_REMOVE ) )
{
    //是否退出程序
    if ( msg.message == WM_QUIT )
        break;
    //消息转换
    TranslateMessage( &msg );
    DispatchMessage( &msg );
}

//游戏循环
GameMain();

//游戏退出控制
GameShut();
return msg.wParam;
}

LRESULT CALLBACK WndProc( HWND hwnd, UINT message,
    WPARAM wParam, LPARAM lParam )    //回调函数
{
    HDC hdc;
    PAINTSTRUCT ps;
    switch( message )    //消息循环
    {
        case WM_CREATE:
            return (0);
        case WM_PAINT:
            hdc = BeginPaint( hwnd, &ps );
            EndPaint( hwnd, &ps );
            return (0);
        case WM_DESTROY:
            PostQuitMessage(0);
            return (0);
    }

    return DefWindowProc( hwnd, message, wParam, lParam );
}

```

游戏框架头文件 GameMain.h 定义了游戏框架文件中使用的主要变量。

```

//函数
int GameMain( void);           //游戏循环
int GameInit( void);           //游戏初始化
int GameShut( void);           //游戏退出控制
struct tobject
{
    int tankid;
    int life;
    int x;
    int y;
    int speed;
    struct tobject * next;
} tankobject;
//移动坦克
int TankMoving( void);
//游戏开始
int GameStart( void);
//游戏结束
int GameOver( void);
//游戏记分
int DrawScore( int score1, int x1, int y1);

```

游戏框架文件 GameMain. cpp 是游戏的控制文件, 由于本书篇幅的限制, 这个文件中的一些内容不会在后面的章节中详细介绍, 不过代码不是很难, 相信花一点时间就可以明白。

```

#include "../H/GameMain. h"
#include "../H/MyDirectDraw. h"
#include "../H/MyDirectInput. h"
#include "../H/MyDirectAudio. h"
//键盘定义
#define KEYDOWN( vk_code) ( ( GetAsyncKeyState( vk_code) & 0x8000) ? 1 : 0)
#define KEYUP( vk_code) ( ( GetAsyncKeyState( vk_code) & 0x8000) ? 0 : 1)
extern HWND main_window_handle;           //外部定义的主窗口变量
extern HINSTANCE main_instance;           //外部定义的主句柄
extern char buffer[ 80];                   //外部定义的临时数组
RECT Dest, Src;
//初始坦克序列
struct tobject * movetanktop, * movetank, * movetanktemp;
int soundid;
int midi_ids[ 10];
int score = 0;

```