

**一、微软平台大型游戏开发六步曲**

微软提供几种编程语言可以写Windows窗口应用程序，比如C/C++，VB，C#等。Windows下要学习游戏窗口编写，一定离不开Windows API的使用。

1、安装窗口编译环境

推荐VS（VS2008,VS2010,VS2015,VS2013,VS2017），MSDN文档。

2、安装帮助文档MSDN

3、C/C++语言编程

4、掌握Windows编程

编写Windows程序一般有两种：一种是MFC编程（C++）；一种是Win32编程（C+Windows API调用方式）。

5、窗口交互界面

Windows编程两部分：Windows下的交互界面编程；深入理解Windows运行机制、内存管理和核心对象。

框架——Windows是一个消息驱动的系统

窗口——窗口管理，比如创建、销毁、重绘、透明处理技术等

控件——包括Windows窗口上的各种控件的使用和管理。

外设——包括鼠标、键盘、打印机等、串/并口。

GDI——绘制各种Windows GDI对象，比如字体、位图、画笔、画刷等

6、Windows核心编程

Windows内部运行机制。

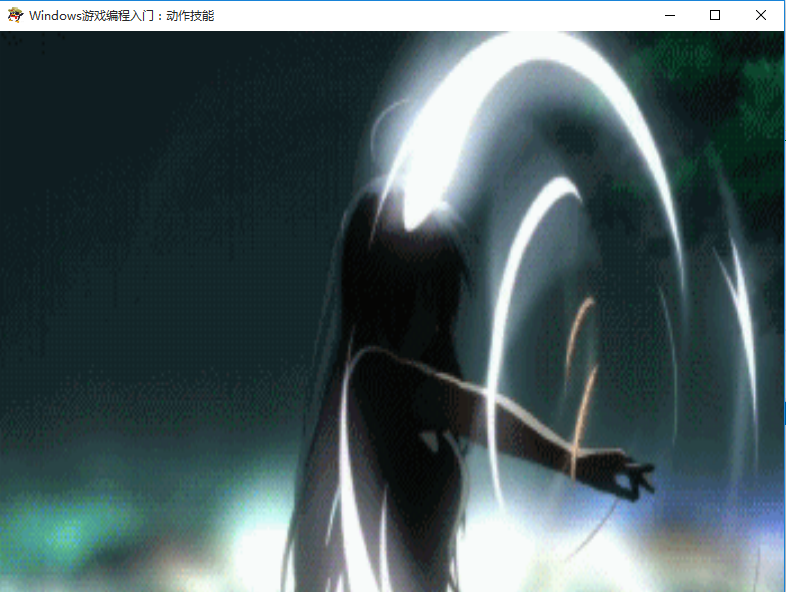
内核对象——包括进程、线程、高度、同步等技术

内存管理——包括Windows系统的虚拟内存管理、地址空间、内存映射等

动态链接库——包括动态库原理、创建等

异常处理机制——结构化异常处理、异常链、错误与恢复。

**二、Windows游戏动作技能入门**



//---------------------------------------------------------------------------------------

//

// 程序名称：Windows游戏入门动作技能

//

// 编译环境：Visual Studio 2013

//

// 文件名称：Game.cpp

//

// 开发作者：张三

//

// 创建日期：2017-04-03

//

// 修改日期：2017-04-03 By 李四

//

//---------------------------------------------------------------------------------------

#include <Windows.h>

#include <tchar.h>

#pragma comment(lib,"winmm.lib") // 调用PlaySound()函数所需要的库文件

#define WINDOW\_WIDTH 800

#define WINDOWS\_HEIGHT 600

#define WINDOW\_TITEL L"Windows游戏编程入门：动作技能"

// 声明全局变量--------------------------------------------------------------------

HDC g\_hdc = NULL, g\_mdc = NULL; // 全局设备环境句柄/全局内存DC句柄HB

HBITMAP g\_hSprite[12]; // 声明位图数组来存储每一张人物的图片

DWORD g\_tPre = 0, g\_tNow = 0; // 记录上一次绘图的时间/记录本次准备绘图时间

int g\_iNum = 0; // 记录目前显示的图片编号

// 声明全局函数--------------------------------------------------------------------

LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam); // 窗口过程函数

BOOL Init\_Game(HWND hwnd); // 资源初始化

VOID Paint\_Game(HWND hwnd); // 绘制游戏图片

BOOL CleanUp\_Game(HWND hwnd); // 清理资源

// WinMain()：Windows应用程序的入口函数

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)

{

// 窗口类

WNDCLASSEX wndclass = { 0 };

wndclass.cbSize = sizeof(WNDCLASSEX); // 设置结构体字节数大小

wndclass.style = CS\_HREDRAW | CS\_VREDRAW; // 设置窗口的样式

wndclass.lpfnWndProc = WndProc; // 设置指向窗口过程函数的指针

wndclass.cbClsExtra = 0; // 窗口类附加内存，一船情况下直接取0即可

wndclass.cbWndExtra = 0; // 窗口附加内存，依然取0即可

wndclass.hInstance = hInstance; // 包含窗口过程的程序实例句柄

wndclass.hIcon = (HICON)::LoadImage(NULL, L"GAME.ico", IMAGE\_ICON, 0, 0, LR\_DEFAULTSIZE|LR\_LOADFROMFILE);

wndclass.hCursor = LoadCursor(NULL, IDC\_ARROW);// 指定窗口类的光标句柄

wndclass.hbrBackground = (HBRUSH)GetStockObject(WHITE\_BRUSH);

wndclass.lpszMenuName = NULL; // 用一个以空终止的字符串，指定菜单资源名称

wndclass.lpszClassName = L"GameGDI";

// 注册窗口类

if (!RegisterClassEx(&wndclass))

return -1;

// 创建窗口

HWND hwnd = CreateWindow(L"GameGDI", WINDOW\_TITEL, WS\_OVERLAPPEDWINDOW, CW\_USEDEFAULT,

CW\_USEDEFAULT, WINDOW\_WIDTH, WINDOWS\_HEIGHT, NULL, NULL, hInstance, NULL);

// 显示更新

MoveWindow(hwnd, 250, 80, WINDOW\_WIDTH, WINDOWS\_HEIGHT, true);

ShowWindow(hwnd, nShowCmd);

UpdateWindow(hwnd);

// 游戏资源初始化

if (!Init\_Game(hwnd))

{

MessageBox(hwnd, L"游戏资源初始化失败", L"错误", 0);

return FALSE;

}

PlaySound(L"仙剑·重楼.wav", NULL, SND\_FILENAME | SND\_ASYNC | SND\_LOOP);

// 消息循环

MSG msg = { 0 };

while (msg.message != WM\_QUIT) // 使用while循环，如果消息不是WM\_QUIT消息，继续循环

{

if (PeekMessage(&msg, 0, 0, 0, PM\_REMOVE)) // 查看应用程序消息队列，有消息时将队列中的消息发出去

{

TranslateMessage(&msg); // 将虚拟键消息转换成字符消息

DispatchMessage(&msg); // 分发一个消息窗口

}

else

{

g\_tNow = GetTickCount();

if (g\_tNow - g\_tPre >= 100)

Paint\_Game(hwnd);

}

}

// 窗口注销

UnregisterClass(L"GameGDI", wndclass.hInstance);// 程序准备结束，注销窗口类

return 0;

}

// 窗口过程函数

LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)

{

switch (message)

{

case WM\_DESTROY: // 窗口销毁消息

CleanUp\_Game(hwnd); // 进行退出前的资源清理

PostQuitMessage(0); // 向系统表明线程有终止请求

break;

default:

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

}

return 0; // 正常退出

}

// 资源初始化

BOOL Init\_Game(HWND hwnd)

{

g\_hdc = GetDC(hwnd);

wchar\_t filename[20];

// 加载每个人物

for (int i = 0; i < 12; i++)

{

memset(filename, 0, sizeof(filename));

swprintf\_s(filename, L"%d.bmp", i);

g\_hSprite[i] = (HBITMAP)LoadImage(NULL, filename, IMAGE\_BITMAP, 800, 600, LR\_LOADFROMFILE);

}

g\_mdc = CreateCompatibleDC(g\_hdc); // 建立兼容设备环境的内存DC

Paint\_Game(hwnd);

return TRUE;

}

// 绘制游戏图片

VOID Paint\_Game(HWND hwnd)

{

if (g\_iNum == 11) // 判断是否超过最大图片编号，若超过最大图片编号10，则直接重新设置0

g\_iNum = 0;

SelectObject(g\_mdc, g\_hSprite[g\_iNum]);

// 把图片按编号贴图到窗口当中

BitBlt(g\_hdc, 0, 0, 800, 600, g\_mdc, 0, 0, SRCCOPY);

g\_tPre = GetTickCount();

g\_iNum++;

}

// 清除资源

BOOL CleanUp\_Game(HWND hwnd)

{

// 释放资源对象

for (int i = 0; i < 12; i++)

DeleteObject(g\_hSprite[i]);

DeleteDC(g\_mdc);

ReleaseDC(hwnd, g\_hdc);

return TRUE;

}