面向对象编程技术

实验二实验报告

班 级：通信一班

学 号：201900800413

姓 名：左长盛

时 间：2021-10-10

**目 录**

1 实验要求 1

2 实验过程及结果 1

2.1 创建工程 1

2.2 程序流程和数据模型分析 1

2.3 问题及解决方案 1

3 实验总结 1

附录 关键代码 1

# 实验要求

实验一要求：

1. 配置Visual Studio环境，查找启动Visual Studio环境所需的可执行程序(exe文件)；创建自己的工作目录；在自己的工作目录下创建Win32 Application空工程/项目；
2. 将Easywin.c复制到工程目录并加入到工程中，编译；
3. 对PlaySound函数链接出错的情况进行修正，理解编译、链接；（增加程序链接使用的多媒体库，具体做法是：在菜单“项目”🡪属性（有的版本是Project🡪Settings）对话框的链接器（或link）设置中，选择项目的“输入”或“Input”，然后在“Object/library modules”编辑框中的最后添加多媒体函数库“winmm.lib”，之后即可链接通过。修改PlaySound函数使用的wave文件名，程序启动的时候能播放相应的音频）；
4. 修改程序文件中的字符串，以更改显示内容；
5. 修改有关注释，同时阅读代码，初步掌握Windows应用程序的构成。

跟踪调试练习：适当地**设置断点**，**查看变量取值**

实验二要求：（设计打字游戏）

1. 创建空白工程，复制示例程序到工作目录，修改代码并观察修改后的程序运行结果

2. 编写简单的键盘打字游戏，要求：

* + 创建新工程。
  + 随机（或预设置）一些字符串，显示在屏幕上。如果按的键中有字符串中的字符，就将该字符删除，然后再在屏幕上显示整个字符串。待一个串被完全打完之后，另外产生一个字符串，继续。
  + 每打完一个字符串后，奖励一定的分数。
  + 用户可按特定键开始游戏。
  + 用户可按ESC键终止游戏。游戏如何开始、如何终止需要在屏幕上有所提示。
  + 得分累计：每次击键之后根据是否有击中的字母计算得分，并在窗口中显示。

3.鼠标位置的显示

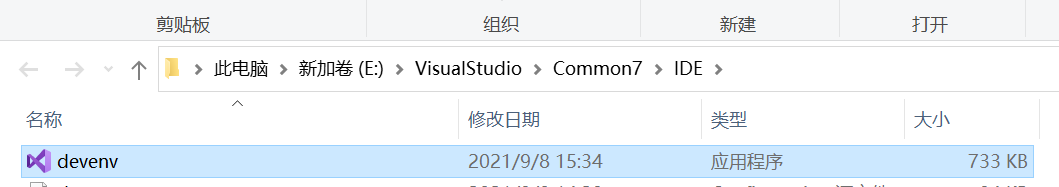
* + 对鼠标左键点击和右键点击，**设置断点**查看鼠标点击位置的坐标。
  + 在窗口客户区内显示鼠标当前位置（处理鼠标移动消息）

4.编写批处理（bat）文件，完成实验一项目和本实验项目的批量构建。以后每个实验完成之后，都在bat中增加一个批处理的编译命令。到学期末可以将所有的实验统一编译。

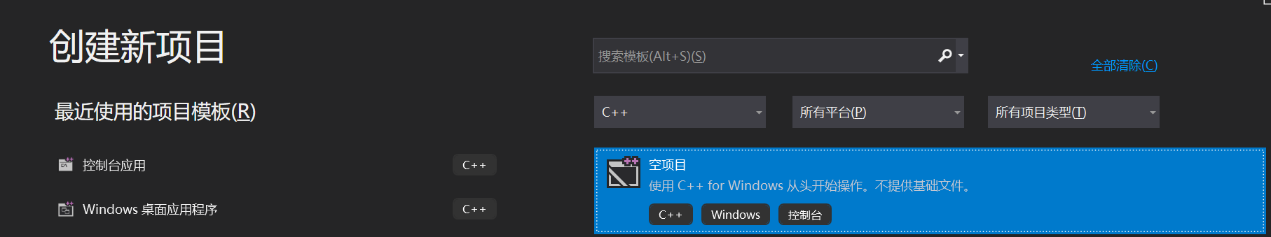
# 实验过程及结果

## 创建工程

1、查找VS的工作路径并打开，创建Win32的工程；打开vs的安装目录，找到下图运行vs2019；



运行后，创建项目：





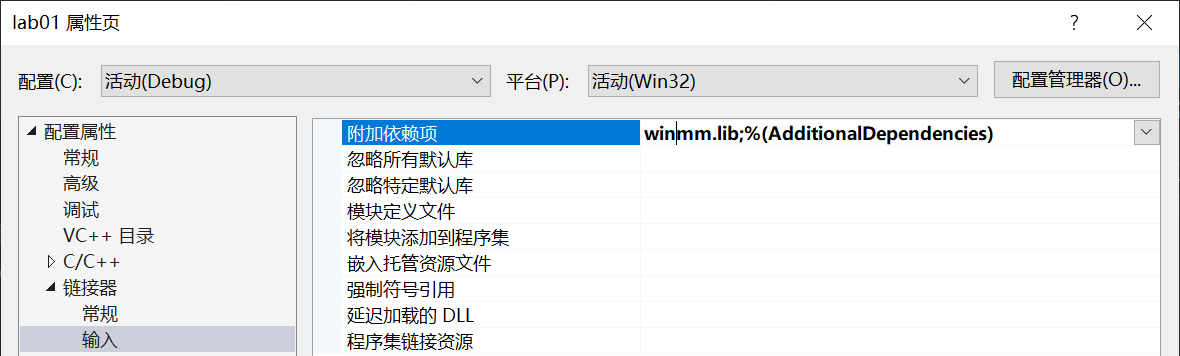
新建lab01.cpp文件，加入到源文件分组；

2、复制EASYWin的代码到lab01.cpp中；

## 程序流程和数据模型分析

一、lab01:

1. 找到项目，右击找到属性，对PlaySound函数的连接音频进行更正：按照提示的位置找到设置进行修改：



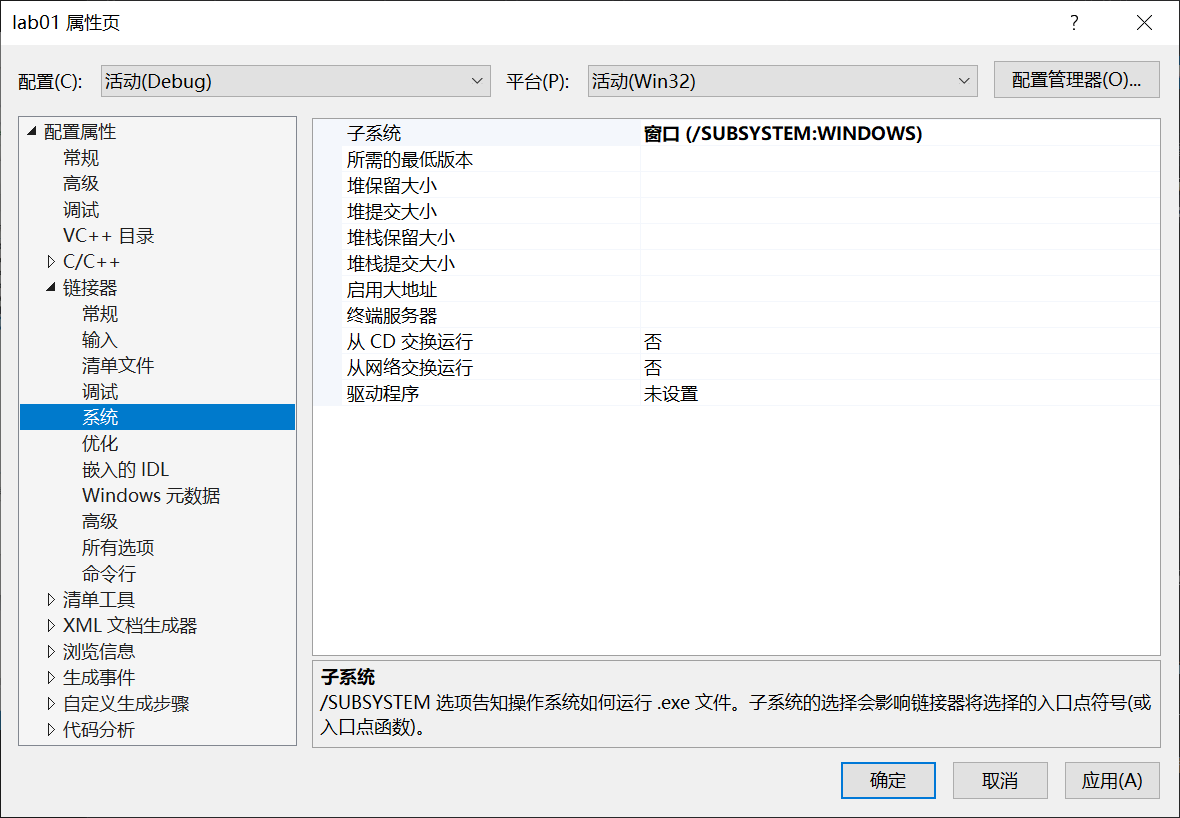
将音频文件粘贴到文件夹下：



再将音频文件添加到资源文件分组下：

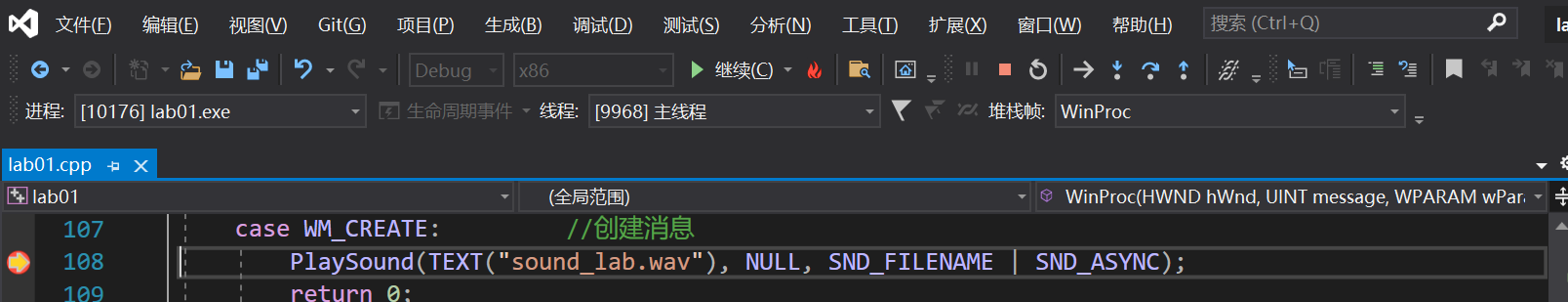


1. 更改系统窗口：



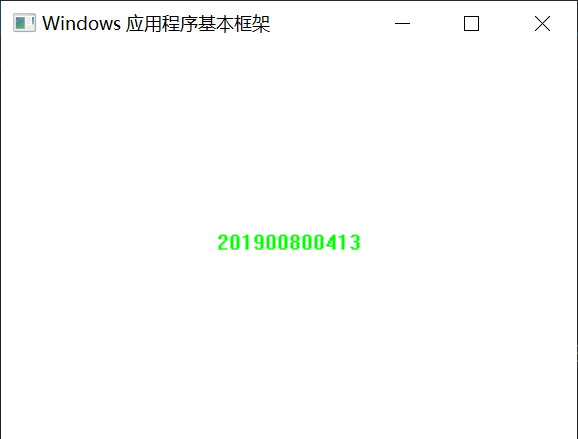
1. 进行调试；

点击代码所在行，右键单击添加断点；



点击逐语句（F10、F11）执行下一条语句；

1. 修改字符串的内容、颜色、位置；



1. 使用cmd;

安装路径位于：E:\VisualStudio\Common7\IDE\devenv.exe

复制到“”中，可以打开该程序；

项目工作区路径：C:\Users\s1776\Desktop\Windows编程\lab01

按照步骤输入到cmd中，可以得到：

二、打字小游戏的程序设计：

1、首先设计程序框图;



按照框图所示的要求，对原程序进行修改；

2、编写一个生成随机字符串的函数；

取得当前系统时间的 分秒值，作为srand函数的输入参数种子seed；其数值范围0-999；如果seed为相同值，rand()将得到相同的随机数；

Traget\_str[i]= 'A'+rand() % 26;

全局变量 Traget\_str 将在循环中，每次被赋值 ‘A’-‘Z’ 的随机值；定义在Init\_Letter函数中。

计算得分可以加入全局变量GameScore，每次搜索匹配后加一定分数；

3、键盘输入的信息将 产生 WM\_CHAR的Message 信息，键入的字符信息将存储在 wParam 中，需要在目前的Traget\_str 中搜索 键入的字符，使用 wcslen 寻找其下标指针，然后删除 Traget\_str 的该字符；

定义 EraseChar函数，进行删除；入口为 目标字符串，一个指向删除字符串指针。

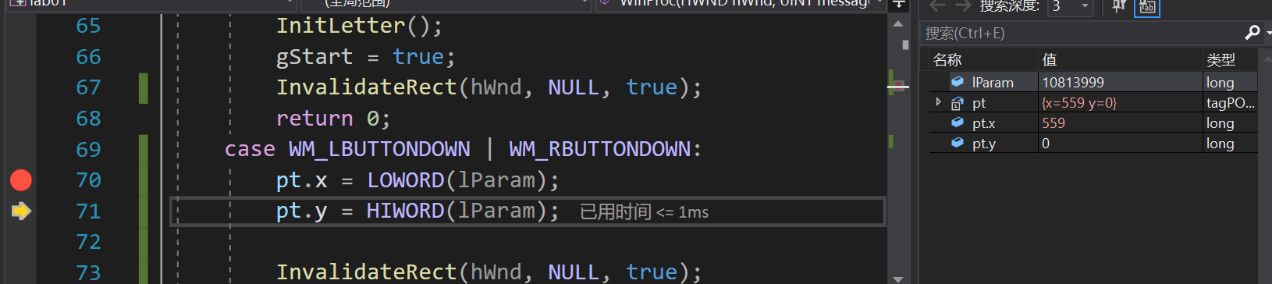
删除字符后，需要判断 Traget\_str 是否为空，如果为空串，需要重新产生随机字符串。

设置断点观察鼠标的位置：

关于鼠标位置的信息保存在Lparam参数中，低字节为x坐标，高字节为y坐标；

使用函数LOWORD和HIWORD可以将两个坐标读出，绘制用户区矩形进行显示；

在case 鼠标左键或右键按下的 语句后设置一个断点并进行调试，观察到;



4、游戏开始界面显示：

定义全局变量标记gStart,初始值设置为true，接受到指定字符后，修改为false,;

绘制矩形：

在WM\_PAINT下，修改rect 的值，使用DrawText函数，

Wsprintf() 函数将 待输出的值 输入到缓冲区中， 类似printf函数；

InvalidateRect(hWnd, NULL, true);

该函数将使程序 直接转到 WM\_PAINT处，更新显示界面；

5、

编写批处理（bat）文件，首先新建一个文本文件，打开cmd窗口，输入以下命令：

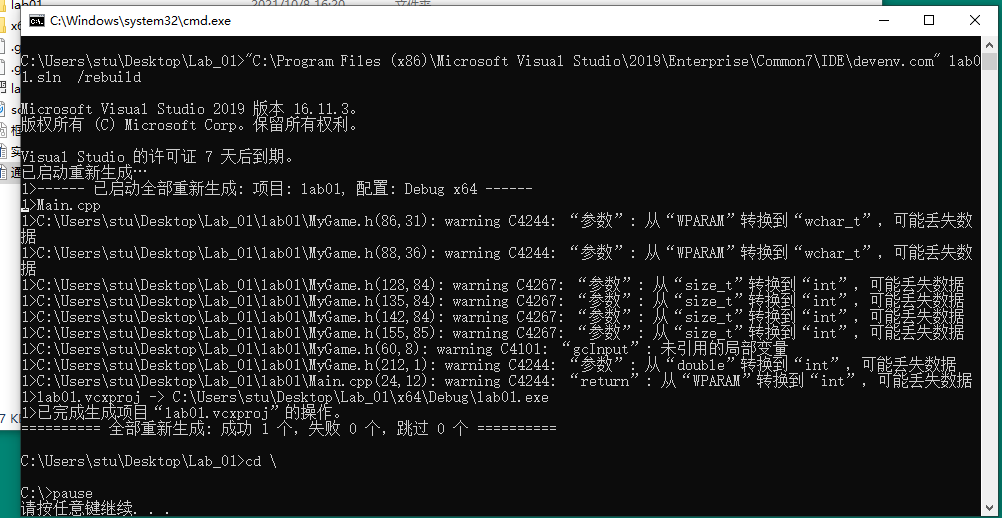
cd \

C:\>cd C:\Users\stu\Desktop\Lab\_01

C:\Users\stu\Desktop\Lab\_01>"C:\Program Files (x86)\Microsoft Visual Studio\2019\Enterprise\Common7\IDE\devenv.com" lab01.sln /rebuild

其中，第一行应为项目文件夹所在目录。

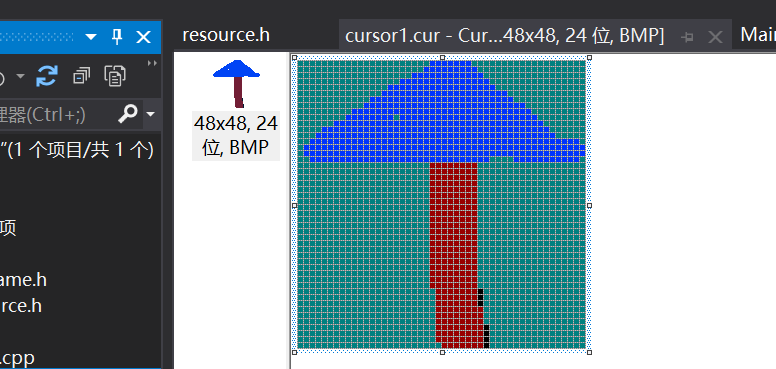
执行cmd后得到：



再将txt文件重命名为 bat 文档；

1. 自定义一个鼠标图标

在资源文件中右键添加资源，选择鼠标，增加24位彩色图，使用彩笔进行绘制并进行保存，



新增resource.h 中包含 关于该鼠标的宏，将其修改到窗口的 wcMainWnd.hCursor中，注意使用函数 MAKEINTRESOURCE 进行转换。

1. 程序效果演示：

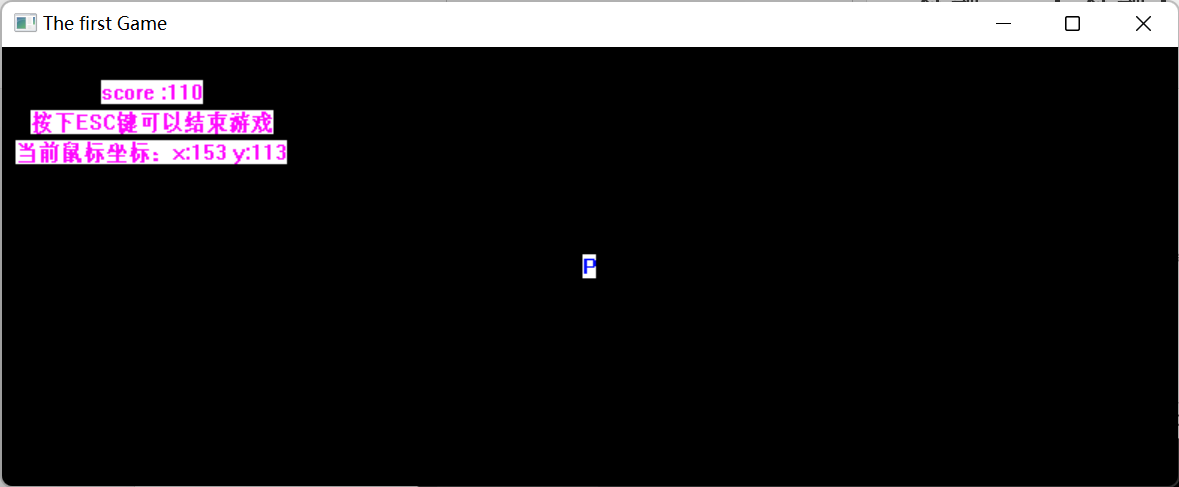
程序开始界面



程序开始后：

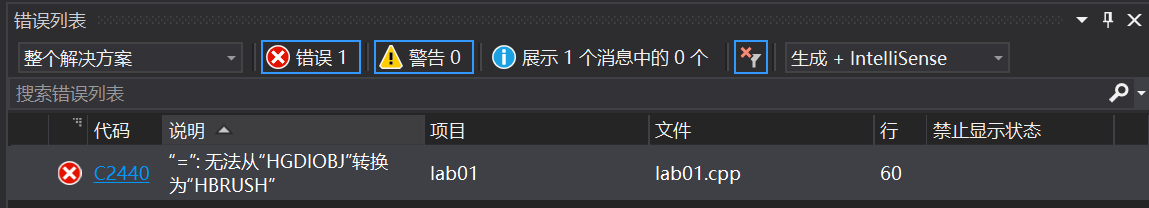


敲击键盘、双击鼠标显示坐标后：

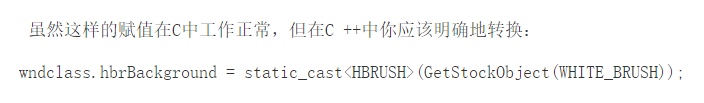


## 问题及解决方案

1、第一次编译时，将.c 文件复制到.cpp 文档时出现，出现以下编译错误;



搜素后，得到解决方案：



加入强制类型转换（HBRUSH）也可以解决

2.程序启动时音频不播放；

解决方法：.wav文件的复制目录错误，应该复制到实验中Debug文件的同一文件夹下；

3、设置鼠标坐标为双击显示时，坐标不显示；

解决方法：



需要在 style 中，加入 CS\_DBCLKS;

4、

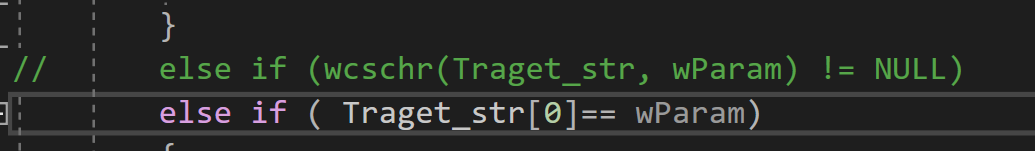
自定义的鼠标类型无法成功加载，解决方法：

删除IDC\_CURSOR1中，删除默认的位图；在窗口类结构中，修改函数：

wcMainWnd.hCursor = LoadCursor(hInstance, MAKEINTRESOURCE(IDC\_CURSOR1));

hInstance 应该修改；

5、对于一个等待输入的字符串，应该保证输入顺序由左向右，所以，输入的数字应该与随机字符串Traget\_str 的首字符进行比较，然后再进行操作。对源程序进行修改：

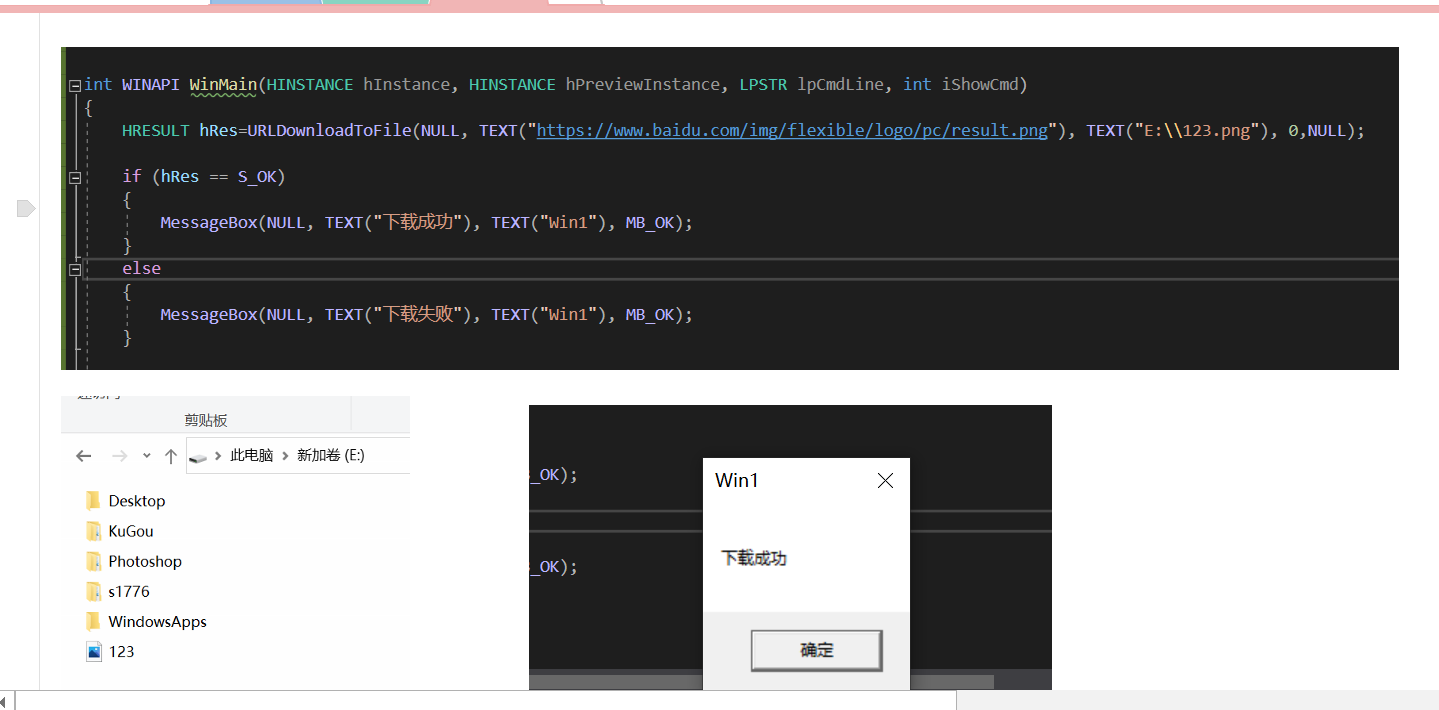


# 实验总结

本次实验设计了一个打字实验，进行比较顺利，对未来windows编程充满信心。

下面对之前的学习总结：

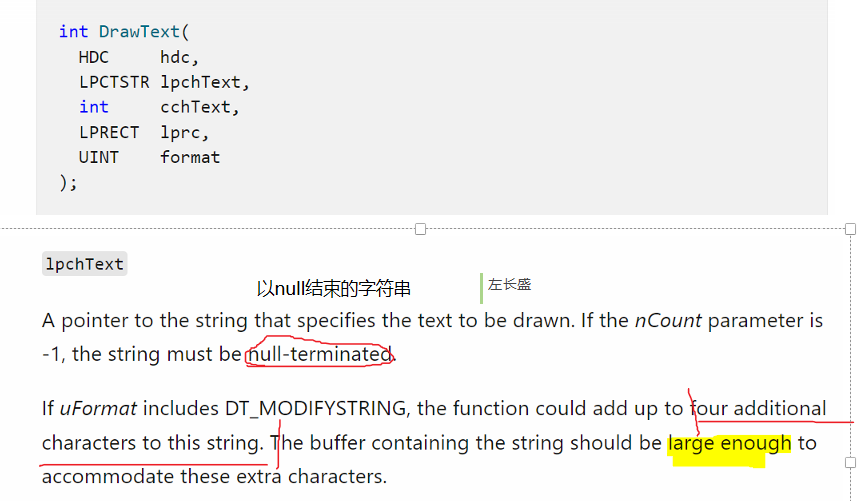
1、函数 URL\_Download\_To\_File

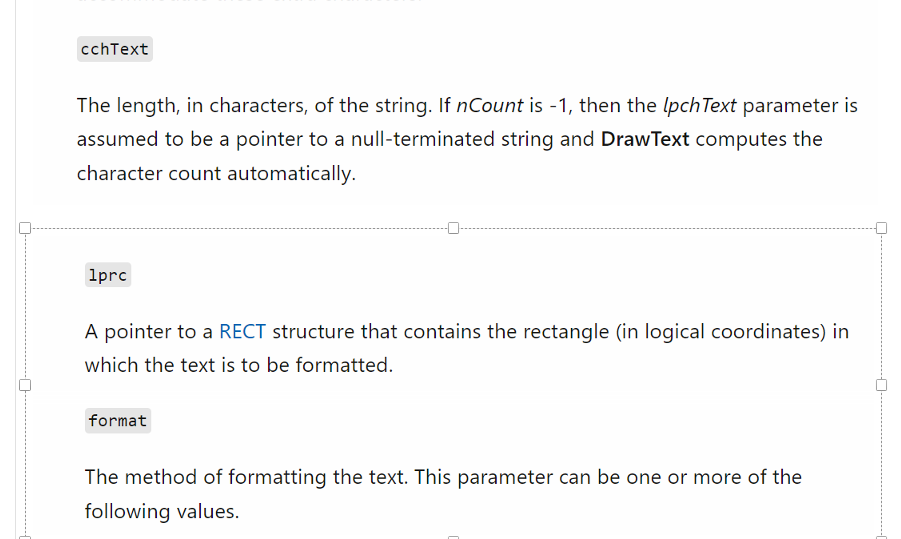


2、产生随机数的函数：

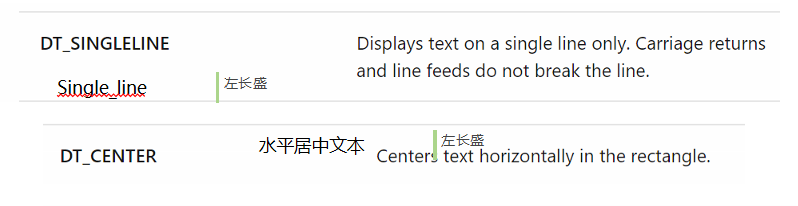


3、DrawText

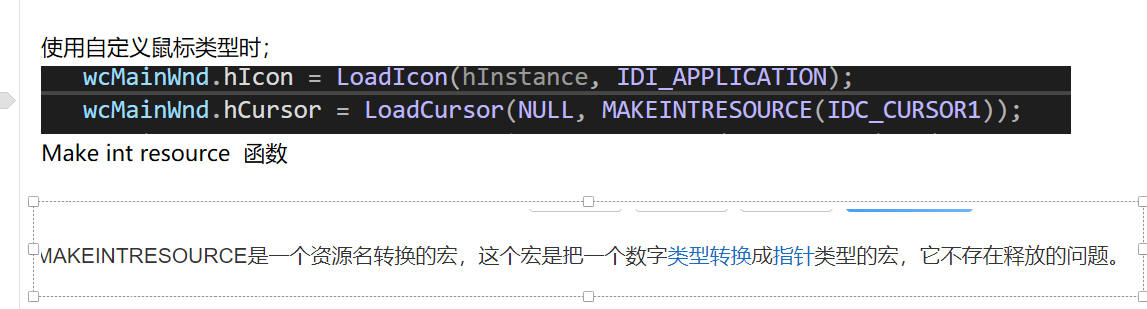




实验中使用到的 format 取值



1. 使用自定义鼠标：



1. 宽字符串处理函数

宽字符函数        普通C函数描述

wcscat（）         strcat（）     把一个字符串接到另一个字符串的尾部

wcsncat（）         strncat（）     类似于wcscat（）， 而且指定粘接字符串的粘接长度.

wcschr（）         strchr（）     查找子字符串的第一个位置

wcsrchr（）         strrchr（）     从尾部开始查找子字符串出现的第一个位置

wcspbrk（）         strpbrk（）     从一字符字符串中查找另一字符串中任何一个字符第一次出现的位置

wcswcs（） / wcsstr（）     strchr（）     在一字符串中查找另一字符串第一次出现的位置

wcscspn（）         strcspn（）     返回不包含第二个字符串的的初始数目

wcsspn（）         strspn（）     返回包含第二个字符串的初始数目

wcscpy（）         strcpy（）     拷贝字符串

wcsncpy（）         strncpy（）     类似于wcscpy（）， 同时指定拷贝的数目

wcscmp（）         strcmp（）     比较两个宽字符串

wcsncmp（）         strncmp（）     类似于wcscmp（）， 还要指定比较字符字符串的数目

wcslen（）         strlen（）     获得宽字符串的数目

wcstok（）         strtok（）     根据标示符把宽字符串分解成一系列字符串

wcswidth（）         None         获得宽字符串的宽度

wcwidth（）         None         获得宽字符的宽度

# 附录 关键代码

#define str\_length 12

#define Score\_aChar 10

#define Score\_aString 100

#define Rect\_wildth 20

TCHAR Traget\_str[str\_length+1];

UINT GameScore = 0;

bool gStart = false;

1、自定义函数

// 初始化随机字符串

void InitLetter()

{

int i;

SYSTEMTIME systime;

GetSystemTime(&systime);

//为rand()得到随机数作准备;

srand(systime.wMilliseconds);

for (i = 0; i < str\_length; i++)

{

Traget\_str[i]= 'A'+rand() % 26;

}

return;

}

//输入正确的字符后，对原有的Traget\_str 进行删除输入字符，如果删除为空字符串，将返回0；

int EraseChar(TCHAR\* Traget\_str, TCHAR\* pos)

{

pos++;

while (\*pos != '\0')

{

\*(pos - 1) = \*pos;

pos++;

}

\*(pos-1) = '\0';

int isEmpty=0;

if (\*Traget\_str == '\0') isEmpty = 1;

return isEmpty;

}

2、 WinProc 部分：

//双击屏幕 后 显示鼠标坐标；

case WM\_LBUTTONDBLCLK:

pt.x = LOWORD(lParam);

pt.y = HIWORD(lParam);

InvalidateRect(hWnd, NULL, true);

case WM\_CHAR:

//游戏开始界面；

if (gStart == true)

{

if (wParam == 'F'|| wParam == 'f') {

gStart = false;

}

}

//检索字符，进行删除和 复位

else if (wcschr(Traget\_str, wParam) != NULL)

{

TCHAR\* pos = wcschr(Traget\_str, wParam);

GameScore += Score\_aChar;

EraseChar(Traget\_str, pos);

if (\*Traget\_str == '\0')

{

InitLetter();

GameScore += Score\_aString;

}

}

//游戏终止

if (wParam == VK\_ESCAPE)

{

PostQuitMessage(-1);

return 0;

}

//可以立即触发 WM\_PAINT，实时显示变化后的状态；

InvalidateRect(hWnd, NULL, true);

return 0;

case WM\_PAINT: //客户区重绘消息

{

//取得设备环境句柄

TCHAR szBuf[32];

hdc = BeginPaint(hWnd, &ps);

rect.left = 0;

rect.top = 0;

rect.right = 200;

rect.bottom = 60;

SetTextColor(hdc, RGB(255, 10, 255));

wsprintf(szBuf, L"score :%d", GameScore);

DrawText(hdc, szBuf, wcslen(szBuf), &rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);

rect.top += Rect\_wildth;

rect.bottom += Rect\_wildth;

wsprintf(szBuf, L"按下ESC键可以结束游戏");

DrawText(hdc, szBuf, wcslen(szBuf), &rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);

rect.top += Rect\_wildth;

rect.bottom += Rect\_wildth;

wsprintf(szBuf, L"当前鼠标坐标：x:%d y:%d",pt.x,pt.y);

DrawText(hdc, szBuf, wcslen(szBuf), &rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);

//取得窗口客户区矩形, rect 将重新初始化

GetClientRect(hWnd, &rect);

//设置文字颜色

SetTextColor(hdc, RGB(0, 10, 255));

if(gStart==false)

DrawText(hdc, Traget\_str,-1, &rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);

else

{

wsprintf(szBuf, L"按下F键开始游戏");

DrawText(hdc, szBuf, wcslen(szBuf), &rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);

}

//释放资源

EndPaint(hWnd, &ps);

return 0;

}