**MFC大作业报告**

**5140729041 王昌浩**

**目录**

[一、软件功能 2](#_Toc435873348)

[二、软件功能实现过程 2](#_Toc435873349)

[三、软件测试 8](#_Toc435873354)

四、程序完整代码 9

[五、谢辞 21](#_Toc435873368)

**一、软件功能**

**1.1 软件功能概述**

**该软件应用MFC实现了在画布上画出坐标系，并在直角坐标系上画出可调节幅值的正弦曲线，并且都可以改变颜色；其次按下开始按钮后可以触发定时器，实现画出小球，并且使小球以自己规定的速度在正弦曲线上运动，并且小球的大小与颜色也可以由自己来设定；最后如果按下结束按钮，会触发killtimer,结束运动。（软件最终界面如图1-1）**

**图1-1 软件界面**

**1.2 软件功能具体描述**

**1、图中坐标系是由在OnPaint函数中的绘图代码实现的，而正弦曲线和小球的绘制是由我自己编写的DrawLines、与DrawSinFun（包含画正弦与画小球）来实现的。**

**2、在1-1图中颜色调节部分是可以分别调节坐标、正弦曲线、与小球的颜色的，他们的原理相同，都是应用MFC ColorButton Control来实现的（详细的原理在本文的第二部分里面）**

**3、幅值调节用到了Slider Control来达到调节正弦曲线幅值大小的目的**

**4、小球的半径调节也同样使用到了Slider Control来调节，并且用到了Edit Control来显示小球的半径。**

**5、小球的运动速度是由Spin Control来控制的，同样由Edit Control来显示速度的大小**

**6、开始与停止按钮，都是由Button按钮的事件处理程序完成的。**

**二、软件功能的实现过程**

**2.1 绘制坐标系与正弦曲线与小球是由我所编写的DrawLines函数与DrawSinFun来实现的，其中DrawSinFun是首先先输入画布，然后在画布上用我所制定的算法取出100个点，然后用画直线的方法来替代正弦曲线，并且在同样的一块画布上，用CBrush画出实心的圆；DrawSinFun的代码如下：**

**void Cp2Dlg::DrawSinFun(CDC \*pCanvasDC,const CRect &rect)**

**{**

**//确定起始坐标点的横纵坐标**

**int BaseY=rect.TopLeft().y+rect.Height()/2;**

**int BaseX=rect.TopLeft().x;**

**//确定幅值范围以及横坐标单位长度**

**int yScale=(rect.Height())/2;**

**int xScale=rect.Width()/(2\*PIE);**

**//创建各点横坐标数列**

**double x[LEN];**

**//绘制正弦曲线循环体**

**int i;**

**for(i=0;i<LEN-1;i++)**

**{**

**//确定相邻两点的横坐标单位**

**x[i]=2\*PIE/LEN\*i;**

**x[i+1]=2\*PIE/LEN\*(i+1);**

**//创建两个坐标n点以表示相邻两点**

**CPoint p1,p2;**

**//确定相邻两点的横纵坐标**

**p1.x=(int)(BaseX+x[i]\*xScale);**

**p1.y=(int)(BaseY+sin(x[i])\*(m\_edit\_int\*yScale/100));**

**p2.x=(int)(BaseX+x[i+1]\*xScale);**

**p2.y=(int)(BaseY+sin(x[i+1])\*(m\_edit\_int\*yScale/100));**

**//连接相邻两点**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**}**

**//CDC \*pDC = this->GetDC(); //获取DC**

**CBrush brush,\*oldbrush; //画刷**

**brush.CreateSolidBrush(m\_clrBall);**

**oldbrush=pCanvasDC->SelectObject(&brush);**

**int r\_circle=m\_edit2\_int;**

**int r\_centerX=BaseX+x[n]\*xScale;**

**int r\_centerY=BaseY+sin(x[n])\*(m\_edit\_int\*yScale/100);**

**int x\_leftup=r\_centerX-r\_circle;**

**int x\_rightdown=r\_centerX+r\_circle;**

**int y\_leftup=r\_centerY-r\_circle;**

**int y\_rightdown=r\_centerY+r\_circle;**

**pCanvasDC->Ellipse(x\_leftup,y\_leftup,x\_rightdown,y\_rightdown);**

**pCanvasDC->SelectObject(oldbrush);**

**//ReleaseDC(pCanvasDC);**

**}**

**在我写完DrawSinFun的代码后，由写了DrawLines的代码，DrawLines同样是定义画布后，创建画笔CPen来绘制坐标，并且调用DrawSinFun函数来得到绘制好的正弦与圆的图像，DrawLines代码如下：**

**int Cp2Dlg::DrawLines()**

**{**

**//创建方形区域**

**UpdateData();**

**CWnd \*pCanvasWnd=GetDlgItem(IDC\_CANVAS);**

**CDC \*pCanvasDC=pCanvasWnd->GetDC();**

**CRect rect;**

**pCanvasWnd->GetClientRect(&rect);**

**//方形区域初始颜色**

**CBrush br(RGB(255,255,255));**

**pCanvasDC->FillRect(&rect,&br);**

**//创建画笔绘制坐标线**

**CPen NewPen;**

**NewPen.CreatePen(PS\_SOLID,1,m\_clrLines);**

**CPen\*OldPen;**

**OldPen=pCanvasDC->SelectObject(&NewPen);**

**//创建两个坐标点**

**CPoint p1,p2;**

**//通过计算确定坐标线的起始点与终结点并连接两点以绘制坐标线**

**p1.x=rect.TopLeft().x;**

**p1.y=rect.TopLeft().y+rect.Height()/2;**

**p2.x=rect.TopLeft().x+rect.Width();**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y+(rect.Height())/4;**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y-2\*(rect.Height())/4;**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y-(rect.Height())/4;**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y+(rect.Height());**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.x=rect.Width()/2;**

**p1.y=0;**

**p2.x=p1.x;**

**p2.y=rect.Height();**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.x=p1.x+rect.Width()/4;**

**p2.x=p1.x;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.x=p1.x-rect.Width()/2;**

**p2.x=p1.x;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**pCanvasDC->SelectObject(OldPen);**

**//创建画笔绘制正弦曲线**

**CPen SinPen;**

**SinPen.CreatePen(PS\_SOLID,2,m\_clrSinCurve);**

**OldPen=pCanvasDC->SelectObject(&SinPen);**

**//调用绘制正弦曲线函数以绘制正弦曲线**

**DrawSinFun(pCanvasDC,rect);**

**pCanvasDC->SelectObject(OldPen);**

**pCanvasDC->Detach();**

**//UpdateData(false);**

**return 0;**

**//结束**

**}**

**2.2、曲线颜色的改变是由MFC ColerButton Control来实现的，首先先建立事件处理函数，再在其事件处理函数中使之前定义过的颜色的变量与刚刚定义的control型的变量相等，就可以了；下面分别是坐标系、正弦曲线、与小球的颜色改变的代码：**

**(1)坐标系：**

**void Cp2Dlg::OnBnClickedMfccolorbutton2()**

**{**

**UpdateData();**

**//令坐标线画笔颜色等于颜色选框颜色**

**m\_clrLines=m\_clrBtn2.GetColor();**

**//以所选颜色重新绘制曲线**

**DrawLines();**

**}**

**(2)正弦曲线：**

**void Cp2Dlg::OnBnClickedMfccolorbutton1()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**UpdateData();**

**//令曲线画笔颜色等于颜色选框选择颜色**

**m\_clrSinCurve=m\_clrBtn.GetColor();**

**//以所选颜色重新绘制曲线**

**DrawLines();**

**}**

**(3)小球：**

**void Cp2Dlg::OnBnClickedMfccolorbutton3()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**UpdateData();**

**//令坐标线画笔颜色等于颜色选框颜色**

**m\_clrBall=m\_clrBtn3.GetColor();**

**DrawLines();**

**}**

**2.3 正弦曲线的幅值是由Slider Control来改变的，在建立完Slider Cotrol后再建立Edit Control，并建立事件处理函数，在其中另两个所建立的变量相等，这样就达到了使得Edit Control的值等于Slider Control的位置的目的；最后在画正弦曲线的函数中将这个变量加入其中来改变正弦曲线的幅值；下面是Slider Control的代码（画正弦曲线的函数已经在2.1中给出）：**

**void Cp2Dlg::OnNMCustomdrawSlider1(NMHDR \*pNMHDR, LRESULT \*pResult)**

**{**

**LPNMCUSTOMDRAW pNMCD = reinterpret\_cast<LPNMCUSTOMDRAW>(pNMHDR);**

**// TODO: 在此添加控件通知处理程序代码**

**\*pResult = 0;**

**UpdateData();**

**//令编辑框显示整形变量等于幅值滚动条变量**

**m\_edit\_int=m\_slider\_int;**

**UpdateData(false);**

**DrawLines();**

**}**

**2.4 小球的半径调节同样用到了Slider Control与Edit Control来完成，步骤与上面一样，代码如下：**

**void Cp2Dlg::OnNMCustomdrawSlider2(NMHDR \*pNMHDR, LRESULT \*pResult)**

**{**

**LPNMCUSTOMDRAW pNMCD = reinterpret\_cast<LPNMCUSTOMDRAW>(pNMHDR);**

**// TODO: 在此添加控件通知处理程序代码**

**\*pResult = 0;**

**UpdateData();**

**//令编辑框显示整形变量等于幅值滚动条变量**

**m\_edit2\_int=m\_slider2\_int;**

**UpdateData(false);**

**DrawLines();**

**}**

**2.5 小球运动的速度是由Spin Control来控制的，其中同样建立Edit Control来显示数值，小球的速度其实就是小球绘制时下一个点与这一个点之间的距离，我在程序中通过正弦曲线的横坐标数组的元素来调节，如2.1中所示，我绘制正弦曲线建立了一个x[100]的数组，其中他的元素为正弦曲线上点的横坐标，我只需控制选取点坐标的间隔就可以实现速度的控制，代码如下：**

**void Cp2Dlg::OnDeltaposSpin1(NMHDR \*pNMHDR, LRESULT \*pResult)**

**{**

**LPNMUPDOWN pNMUpDown = reinterpret\_cast<LPNMUPDOWN>(pNMHDR);**

**// TODO: 在此添加控件通知处理程序代码**

**\*pResult = 0;**

**UpdateData();**

**m\_edit3\_int=m\_edit3\_int-pNMUpDown->iDelta;**

**UpdateData(FALSE);**

**}**

**小球运动是由WM\_Timer类中的OnTimer函数来控制的，OnTimer函数是一个自动循环调用的函数，所以我只需在其中写出小球运动的坐标是如何改变的就可以了，我的实现代码如下：**

**void Cp2Dlg::OnTimer(UINT\_PTR nIDEvent)**

**{**

**// TODO: 在此添加消息处理程序代码和/或调用默认值**

**if(check)**

**{**

**n+=m\_edit3\_int;**

**if(n>=92)**

**n = 92;**

**DrawLines();**

**}**

**if(check==false)**

**{**

**n-=m\_edit3\_int;**

**if(n<=8)**

**n=8;**

**DrawLines();**

**}**

**if(n>=92)**

**{**

**check=!check;**

**}**

**if(n<=8)**

**check=!check;**

**CDialogEx::OnTimer(nIDEvent);**

**}**

**2.6 开始与停止时在Button中建立事件处理函数来完成的，如果要条用OnTimer函数来循环画球，是需要SetTimer函数来调用的，我首先在SetTimer函数中定义我所使用的定时器名字为1，然后设定刷新频率为200ms就可以完成了；在结束按钮中我需要调用KillTimer来结束定时器；具体的代码如下：**

**(1) 开始按钮函数**

**void Cp2Dlg::OnBnClickedButton3()**

**{**

**SetTimer(1,200, NULL);**

**}**

**(2) 结束按钮函数**

**void Cp2Dlg::OnBnClickedButton4()**

**{**

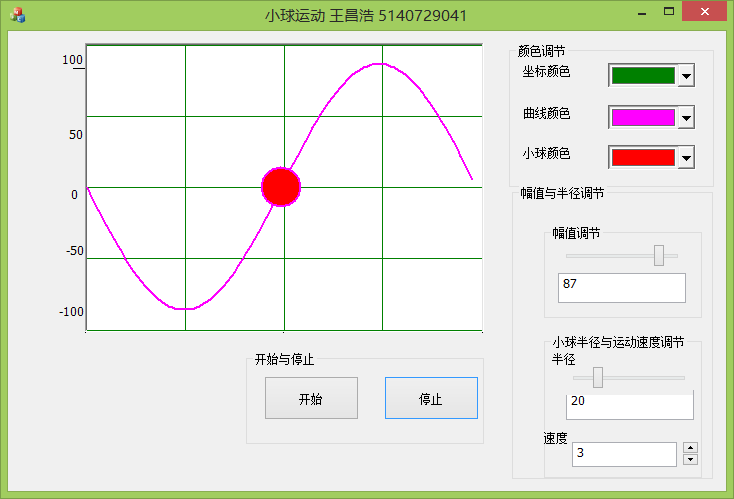
**// TODO: 在此添加控件通知处理程序代码**

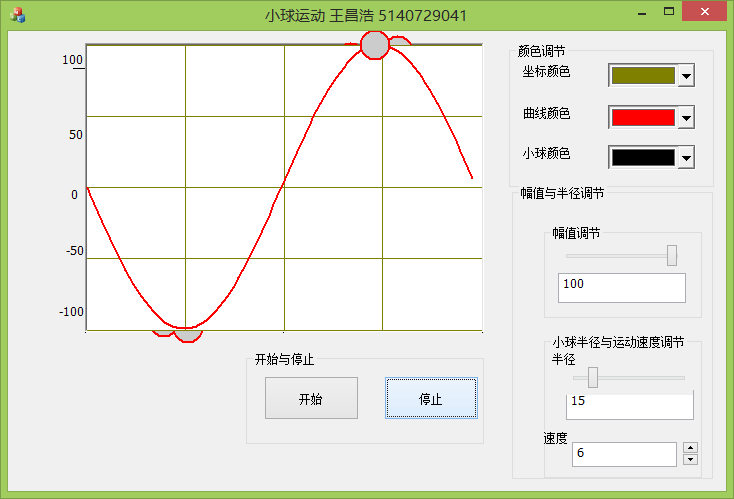
**KillTimer(1);**

**}**

**三、软件测试**

**软件测试的截图如下：**

**3-1 软件测试截图1**

****

**3-2软件测试截图2**

**四、程序完整代码(CPP文件)**

**// p2Dlg.cpp : 实现文件**

**//**

**#include "stdafx.h"**

**#include "p2.h"**

**#include "p2Dlg.h"**

**#include "afxdialogex.h"**

**#include <cmath>**

**bool check = true;**

**int n=8;**

**#ifdef \_DEBUG**

**#define new DEBUG\_NEW**

**#endif**

**// 用于应用程序“关于”菜单项的 CAboutDlg 对话框**

**class CAboutDlg : public CDialogEx**

**{**

**public:**

**CAboutDlg();**

**// 对话框数据**

**enum { IDD = IDD\_ABOUTBOX };**

**protected:**

**virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持**

**// 实现**

**protected:**

**DECLARE\_MESSAGE\_MAP()**

**};**

**CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)**

**{**

**}**

**void CAboutDlg::DoDataExchange(CDataExchange\* pDX)**

**{**

**CDialogEx::DoDataExchange(pDX);**

**}**

**BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)**

**END\_MESSAGE\_MAP()**

**// Cp2Dlg 对话框**

**Cp2Dlg::Cp2Dlg(CWnd\* pParent /\*=NULL\*/)**

**: CDialogEx(Cp2Dlg::IDD, pParent)**

**, m\_edit\_int(0)**

**, m\_slider\_int(0)**

**, m\_slider2\_int(0)**

**, m\_edit2\_int(0)**

**, m\_edit3\_int(0)**

**{**

**m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);**

**}**

**void Cp2Dlg::DoDataExchange(CDataExchange\* pDX)**

**{**

**CDialogEx::DoDataExchange(pDX);**

**DDX\_Text(pDX, IDC\_EDIT1, m\_edit\_int);**

**DDX\_Slider(pDX, IDC\_SLIDER1, m\_slider\_int);**

**DDX\_Control(pDX, IDC\_MFCCOLORBUTTON1, m\_clrBtn);**

**DDX\_Control(pDX, IDC\_MFCCOLORBUTTON2, m\_clrBtn2);**

**DDX\_Control(pDX, IDC\_MFCCOLORBUTTON3, m\_clrBtn3);**

**DDX\_Slider(pDX, IDC\_SLIDER2, m\_slider2\_int);**

**DDX\_Text(pDX, IDC\_EDIT2, m\_edit2\_int);**

**DDX\_Text(pDX, IDC\_EDIT3, m\_edit3\_int);**

**DDX\_Control(pDX, IDC\_SPIN1, m\_spinBtn);**

**}**

**BEGIN\_MESSAGE\_MAP(Cp2Dlg, CDialogEx)**

**ON\_WM\_SYSCOMMAND()**

**ON\_WM\_PAINT()**

**ON\_WM\_QUERYDRAGICON()**

**ON\_NOTIFY(NM\_CUSTOMDRAW, IDC\_SLIDER1, &Cp2Dlg::OnNMCustomdrawSlider1)**

**ON\_EN\_CHANGE(IDC\_EDIT1, &Cp2Dlg::OnEnChangeEdit1)**

**// ON\_BN\_CLICKED(IDC\_BUTTON1, &Cp2Dlg::OnBnClickedButton1)**

**ON\_BN\_CLICKED(IDC\_MFCCOLORBUTTON1, &Cp2Dlg::OnBnClickedMfccolorbutton1)**

**// ON\_BN\_CLICKED(IDC\_BUTTON2, &Cp2Dlg::OnBnClickedButton2)**

**ON\_BN\_CLICKED(IDC\_MFCCOLORBUTTON2, &Cp2Dlg::OnBnClickedMfccolorbutton2)**

**ON\_BN\_CLICKED(IDC\_BUTTON3, &Cp2Dlg::OnBnClickedButton3)**

**ON\_STN\_CLICKED(IDC\_CANVAS, &Cp2Dlg::OnStnClickedCanvas)**

**ON\_NOTIFY(NM\_CUSTOMDRAW, IDC\_SLIDER2, &Cp2Dlg::OnNMCustomdrawSlider2)**

**ON\_BN\_CLICKED(IDC\_MFCCOLORBUTTON3, &Cp2Dlg::OnBnClickedMfccolorbutton3)**

**ON\_WM\_TIMER()**

**ON\_NOTIFY(UDN\_DELTAPOS, IDC\_SPIN1, &Cp2Dlg::OnDeltaposSpin1)**

**ON\_BN\_CLICKED(IDC\_BUTTON4, &Cp2Dlg::OnBnClickedButton4)**

**ON\_EN\_CHANGE(IDC\_EDIT3, &Cp2Dlg::OnEnChangeEdit3)**

**END\_MESSAGE\_MAP()**

**// Cp2Dlg 消息处理程序**

**BOOL Cp2Dlg::OnInitDialog()**

**{**

**CDialogEx::OnInitDialog();**

**// 将“关于...”菜单项添加到系统菜单中。**

**// IDM\_ABOUTBOX 必须在系统命令范围内。**

**ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);**

**ASSERT(IDM\_ABOUTBOX < 0xF000);**

**CMenu\* pSysMenu = GetSystemMenu(FALSE);**

**if (pSysMenu != NULL)**

**{**

**BOOL bNameValid;**

**CString strAboutMenu;**

**bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);**

**ASSERT(bNameValid);**

**if (!strAboutMenu.IsEmpty())**

**{**

**pSysMenu->AppendMenu(MF\_SEPARATOR);**

**pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);**

**}**

**}**

**// 设置此对话框的图标。当应用程序主窗口不是对话框时，框架将自动**

**// 执行此操作**

**SetIcon(m\_hIcon, TRUE); // 设置大图标**

**SetIcon(m\_hIcon, FALSE); // 设置小图标**

**// TODO: 在此添加额外的初始化代码**

**return TRUE; // 除非将焦点设置到控件，否则返回 TRUE**

**}**

**void Cp2Dlg::OnSysCommand(UINT nID, LPARAM lParam)**

**{**

**if ((nID & 0xFFF0) == IDM\_ABOUTBOX)**

**{**

**CAboutDlg dlgAbout;**

**dlgAbout.DoModal();**

**}**

**else**

**{**

**CDialogEx::OnSysCommand(nID, lParam);**

**}**

**}**

**// 如果向对话框添加最小化按钮，则需要下面的代码**

**// 来绘制该图标。对于使用文档/视图模型的 MFC 应用程序，**

**// 这将由框架自动完成。**

**void Cp2Dlg::OnPaint()**

**{**

**if (IsIconic())**

**{**

**CPaintDC dc(this); // 用于绘制的设备上下文**

**SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);**

**// 使图标在工作区矩形中居中**

**int cxIcon = GetSystemMetrics(SM\_CXICON);**

**int cyIcon = GetSystemMetrics(SM\_CYICON);**

**CRect rect;**

**GetClientRect(&rect);**

**int x = (rect.Width() - cxIcon + 1) / 2;**

**int y = (rect.Height() - cyIcon + 1) / 2;**

**// 绘制图标**

**dc.DrawIcon(x, y, m\_hIcon);**

**}**

**else**

**{**

**CPaintDC dc(this);**

**//经过计算确定单位标记线位置并绘制单位标记线**

**dc.MoveTo(65,37);**

**dc.LineTo(77,37);**

**dc.MoveTo(77,97.75);**

**dc.LineTo(71,97.75);**

**dc.MoveTo(77,158.5);**

**dc.LineTo(65,158.5);**

**dc.MoveTo(77,219.25);**

**dc.LineTo(71,219.25);**

**dc.MoveTo(77,280);**

**dc.LineTo(65,280);**

**dc.MoveTo(77,290);**

**dc.LineTo(77,302);**

**dc.MoveTo(176.25,290);**

**dc.LineTo(176.25,296);**

**dc.MoveTo(275.5,290);**

**dc.LineTo(275.5,302);**

**dc.MoveTo(374.75,290);**

**dc.LineTo(374.75,296);**

**dc.MoveTo(474,290);**

**dc.LineTo(474,302);**

**//绘制结束**

**CDialogEx::OnPaint();**

**}**

**}**

**//当用户拖动最小化窗口时系统调用此函数取得光标**

**//显示。**

**HCURSOR Cp2Dlg::OnQueryDragIcon()**

**{**

**return static\_cast<HCURSOR>(m\_hIcon);**

**}**

**void Cp2Dlg::OnNMCustomdrawSlider1(NMHDR \*pNMHDR, LRESULT \*pResult)**

**{**

**LPNMCUSTOMDRAW pNMCD = reinterpret\_cast<LPNMCUSTOMDRAW>(pNMHDR);**

**// TODO: 在此添加控件通知处理程序代码**

**\*pResult = 0;**

**UpdateData();**

**//令编辑框显示整形变量等于幅值滚动条变量**

**m\_edit\_int=m\_slider\_int;**

**UpdateData(false);**

**DrawLines();**

**}**

**void Cp2Dlg::OnEnChangeEdit1()**

**{**

**// TODO: 如果该控件是 RICHEDIT 控件，它将不**

**// 发送此通知，除非重写 CDialogEx::OnInitDialog()**

**// 函数并调用 CRichEditCtrl().SetEventMask()，**

**// 同时将 ENM\_CHANGE 标志“或”运算到掩码中。**

**UpdateData();**

**//根据修改幅值绘制曲线**

**DrawLines();**

**//幅值滚动条位置随着编辑框数值移动**

**m\_slider\_int=m\_edit\_int;**

**UpdateData(false);**

**// TODO: 在此添加控件通知处理程序代码**

**}**

**int Cp2Dlg::DrawLines()**

**{**

**//创建方形区域**

**UpdateData();**

**CWnd \*pCanvasWnd=GetDlgItem(IDC\_CANVAS);**

**CDC \*pCanvasDC=pCanvasWnd->GetDC();**

**CRect rect;**

**pCanvasWnd->GetClientRect(&rect);**

**//方形区域初始颜色**

**CBrush br(RGB(255,255,255));**

**pCanvasDC->FillRect(&rect,&br);**

**//创建画笔绘制坐标线**

**CPen NewPen;**

**NewPen.CreatePen(PS\_SOLID,1,m\_clrLines);**

**CPen\*OldPen;**

**OldPen=pCanvasDC->SelectObject(&NewPen);**

**//创建两个坐标点**

**CPoint p1,p2;**

**//通过计算确定坐标线的起始点与终结点并连接两点以绘制坐标线**

**p1.x=rect.TopLeft().x;**

**p1.y=rect.TopLeft().y+rect.Height()/2;**

**p2.x=rect.TopLeft().x+rect.Width();**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y+(rect.Height())/4;**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y-2\*(rect.Height())/4;**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y-(rect.Height())/4;**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.y=p1.y+(rect.Height());**

**p2.y=p1.y;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.x=rect.Width()/2;**

**p1.y=0;**

**p2.x=p1.x;**

**p2.y=rect.Height();**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.x=p1.x+rect.Width()/4;**

**p2.x=p1.x;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**p1.x=p1.x-rect.Width()/2;**

**p2.x=p1.x;**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**pCanvasDC->SelectObject(OldPen);**

**//创建画笔绘制正弦曲线**

**CPen SinPen;**

**SinPen.CreatePen(PS\_SOLID,2,m\_clrSinCurve);**

**OldPen=pCanvasDC->SelectObject(&SinPen);**

**//调用绘制正弦曲线函数以绘制正弦曲线**

**DrawSinFun(pCanvasDC,rect);**

**pCanvasDC->SelectObject(OldPen);**

**pCanvasDC->Detach();**

**//UpdateData(false);**

**return 0;**

**//结束**

**}**

**//定义正弦曲线绘制函数**

**void Cp2Dlg::DrawSinFun(CDC \*pCanvasDC,const CRect &rect)**

**{**

**//确定起始坐标点的横纵坐标**

**int BaseY=rect.TopLeft().y+rect.Height()/2;**

**int BaseX=rect.TopLeft().x;**

**//确定幅值范围以及横坐标单位长度**

**int yScale=(rect.Height())/2;**

**int xScale=rect.Width()/(2\*PIE);**

**//创建各点横坐标数列**

**double x[LEN];**

**//绘制正弦曲线循环体**

**int i;**

**for(i=0;i<LEN-1;i++)**

**{**

**//确定相邻两点的横坐标单位**

**x[i]=2\*PIE/LEN\*i;**

**x[i+1]=2\*PIE/LEN\*(i+1);**

**//创建两个坐标n点以表示相邻两点**

**CPoint p1,p2;**

**//确定相邻两点的横纵坐标**

**p1.x=(int)(BaseX+x[i]\*xScale);**

**p1.y=(int)(BaseY+sin(x[i])\*(m\_edit\_int\*yScale/100));**

**p2.x=(int)(BaseX+x[i+1]\*xScale);**

**p2.y=(int)(BaseY+sin(x[i+1])\*(m\_edit\_int\*yScale/100));**

**//连接相邻两点**

**pCanvasDC->MoveTo(p1);**

**pCanvasDC->LineTo(p2);**

**}**

**//CDC \*pDC = this->GetDC(); //获取DC**

**CBrush brush,\*oldbrush; //画刷**

**brush.CreateSolidBrush(m\_clrBall);**

**oldbrush=pCanvasDC->SelectObject(&brush);**

**int r\_circle=m\_edit2\_int;**

**int r\_centerX=BaseX+x[n]\*xScale;**

**int r\_centerY=BaseY+sin(x[n])\*(m\_edit\_int\*yScale/100);**

**int x\_leftup=r\_centerX-r\_circle;**

**int x\_rightdown=r\_centerX+r\_circle;**

**int y\_leftup=r\_centerY-r\_circle;**

**int y\_rightdown=r\_centerY+r\_circle;**

**pCanvasDC->Ellipse(x\_leftup,y\_leftup,x\_rightdown,y\_rightdown);**

**pCanvasDC->SelectObject(oldbrush);**

**//ReleaseDC(pCanvasDC);**

**}**

**void Cp2Dlg::OnBnClickedButton1()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**}**

**void Cp2Dlg::OnBnClickedMfccolorbutton1()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**UpdateData();**

**//令曲线画笔颜色等于颜色选框选择颜色**

**m\_clrSinCurve=m\_clrBtn.GetColor();**

**//以所选颜色重新绘制曲线**

**DrawLines();**

**}**

**void Cp2Dlg::OnBnClickedButton2()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**}**

**void Cp2Dlg::OnBnClickedMfccolorbutton2()**

**{**

**UpdateData();**

**//令坐标线画笔颜色等于颜色选框颜色**

**m\_clrLines=m\_clrBtn2.GetColor();**

**//以所选颜色重新绘制曲线**

**DrawLines();**

**}**

**void Cp2Dlg::OnBnClickedButton3()**

**{**

**SetTimer(1,200, NULL);**

**}**

**void Cp2Dlg::OnStnClickedCanvas()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**}**

**void Cp2Dlg::OnNMCustomdrawSlider2(NMHDR \*pNMHDR, LRESULT \*pResult)**

**{**

**LPNMCUSTOMDRAW pNMCD = reinterpret\_cast<LPNMCUSTOMDRAW>(pNMHDR);**

**// TODO: 在此添加控件通知处理程序代码**

**\*pResult = 0;**

**UpdateData();**

**//令编辑框显示整形变量等于幅值滚动条变量**

**m\_edit2\_int=m\_slider2\_int;**

**UpdateData(false);**

**DrawLines();**

**}**

**void Cp2Dlg::OnBnClickedMfccolorbutton3()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**UpdateData();**

**//令坐标线画笔颜色等于颜色选框颜色**

**m\_clrBall=m\_clrBtn3.GetColor();**

**DrawLines();**

**}**

**void Cp2Dlg::OnTimer(UINT\_PTR nIDEvent)**

**{**

**// TODO: 在此添加消息处理程序代码和/或调用默认值**

**// 添加画球代码**

**if(check)**

**{**

**n+=m\_edit3\_int;**

**if(n>=92)**

**n = 92;**

**DrawLines();**

**}**

**if(check==false)**

**{**

**n-=m\_edit3\_int;**

**if(n<=8)**

**n=8;**

**DrawLines();**

**}**

**if(n>=92)**

**{**

**check=!check;**

**}**

**if(n<=8)**

**check=!check;**

**CDialogEx::OnTimer(nIDEvent);**

**}**

**void Cp2Dlg::OnDeltaposSpin1(NMHDR \*pNMHDR, LRESULT \*pResult)**

**{**

**LPNMUPDOWN pNMUpDown = reinterpret\_cast<LPNMUPDOWN>(pNMHDR);**

**// TODO: 在此添加控件通知处理程序代码**

**\*pResult = 0;**

**UpdateData();**

**m\_edit3\_int=m\_edit3\_int-pNMUpDown->iDelta;**

**UpdateData(FALSE);**

**}**

**void Cp2Dlg::OnBnClickedButton4()**

**{**

**// TODO: 在此添加控件通知处理程序代码**

**KillTimer(1);**

**}**

**void Cp2Dlg::OnEnChangeEdit3()**

**{**

**// TODO: 如果该控件是 RICHEDIT 控件，它将不**

**// 发送此通知，除非重写 CDialogEx::OnInitDialog()**

**// 函数并调用 CRichEditCtrl().SetEventMask()，**

**// 同时将 ENM\_CHANGE 标志“或”运算到掩码中。**

**// TODO: 在此添加控件通知处理程序代码**

**}**

**五、谢辞  
感谢陈晓波老师这一学期耐心的指导让我从一个对C++完全不了解的学生成为了一个可以自己编写一些简单程序的人，感谢老师平时仔细的讲课与课上、与课后耐心的指导，谢谢您。**

**感谢助教杨肖老师，感谢您这一学期耐心的指导，通过您的指导我学会了很多，也养成了一定的编程的习惯，谢谢您。**