



西南交通大学 323 实验室工程 系列教材
“计算机软件实验中心”系列实验教材

计算机图形学 实验教程

编 刘 倩

审 西南交通大学实验室及设备管理处



西南交通大学出版社
[Http://press.swjtu.edu.cn](http://press.swjtu.edu.cn)

西南交通大学“323 实验室工程”系列教材
“计算机软件实验中心”系列实验教材

计算机图形学实验教程

编 刘 倩

审 西南交通大学实验室及设备管理处

西南交通大学出版社

· 成 都 ·

内 容 简 介

本书是“计算机图形学”课程的实验教程,它将计算机图形学基本原理的教授和具体实现相结合,开发工具选用面向对象 Visual C++ 6.0 的 MFC 框架环境作为开发平台,可以实现对图形的交互式操作。本书共给出 9 个实验教程,内容包括:OpenGL 程序设计基础,二维基本图元的生成,二维图形区域填充,二维图形的几何变换,二维裁剪,曲线、曲面绘制,图形的投影变换,三维面消隐算法,真实感图形绘制。本书对相关算法讲解透彻,读者可以很容易地按照本书提供的操作步骤完成上机实践。

本书不仅可以作为大学本科和高职高专计算机图形学课程的实验教程,还可供对计算机图形学感兴趣的读者自学使用。

图书在版编目(CIP)数据

计算机图形学实验教程 / 刘倩主编. — 成都:西南交通大学出版社, 2009.9

(西南交通大学“323 实验室工程”系列教材. “计算机软件实验中心”系列实验教材)

ISBN 978-7-5643-0430-0

I. 计… II. 刘… III. 计算机图形学—高等学校—教材
IV. TP391.41

中国版本图书馆 CIP 数据核字 (2009) 第 161230 号

西南交通大学“323 实验室工程”系列教材

“计算机软件实验中心”系列实验教材

计算机图形学实验教程

编 刘 倩

*

责任编辑 李芳芳

封面设计 本格设计

西南交通大学出版社出版发行

(成都二环路北一段 111 号 邮政编码: 610031 发行部电话: 028-87600564)

<http://press.swjtu.edu.cn>

成都蓉军广告印务有限责任公司印刷

*

成品尺寸: 185 mm × 260 mm 印张: 11.625

字数: 289 千字

2009 年 9 月第 1 版 2009 年 9 月第 1 次印刷

ISBN 978-7-5643-0430-0

定价: 18.50 元

图书如有印装质量问题 本社负责退换
版权所有 盗版必究 举报电话: 028-87600562

前 言

计算机图形学作为利用计算机生成图形的技术,已经越来越广泛地在各个领域得到应用,成为计算机科学技术领域的一个重要研究方向,并被广泛地应用于科学计算、工程设计、医药、工业、艺术、娱乐业、广告业、教育与培训、商业和政府部门等。

鉴于计算机图形学的重要性和应用的广泛性,计算机软件专业将“计算机图形学”课程设置为专业必修课,课程主要研究用计算机及其图形设备来输入、表示、变换、运算和输出图形的原理、算法及系统。通过对本课程的学习,使学生对计算机图形学有一个完整的了解,并为计算机图形学的应用和研究打下坚实基础。由于“计算机图形学”课程实践性很强,为加强学生对计算机图形学理论知识的进一步理解,本课程教学由课堂理论教学和上机实验教学两大环节组成。其中,上机实验是对学生的一种全面训练,一方面着眼于原理与应用的结合,使学生学会如何将书上学到的知识用于解决实际问题,培养软件工作所需要的动手能力;另一方面,能使书上的知识变“活”,起到深化理解和灵活掌握教学内容的目的。

本实验教程设计了9个实验,实验内容力求理论性和实用性紧密结合。实验一为入门性实验,旨在让学生初步了解基于 OpenGL 的简单图形设计;实验二至七为基础性实验,实验中给出了图形生成及处理变换的各种基本算法,通过编程实现这些算法,从而使学生进一步理解和牢固掌握图形学中重要的理论知识,并通过对简单图形的设计与编程实现,让学生对该学科产生兴趣;实验八、九为综合性试验,要求学生翻阅大量的相关资料,通过对真实感图形的绘制,让学生理解图形理论的实际应用,旨在提高学生的综合动手能力和创新能力,为以后从事图形、游戏及软件开发工作打下良好的基础。

希望通过本实验教程,一方面,使学生更加深入地理解计算机图形系统的工作机理以及基本图形的生成和处理算法,使学生系统掌握计算机图形学的基本理论、基本算法,能正确评价、完善、编程实现所学的算法,具备创造更高效算法的意识,具有编写计算机图形应用软件的能力;另一方面,使学生具备将图形学的研究思想运用到其他领域并解决相关问题的能力,同时初步具备在图形学领域进行研究的能力。

编 者

2009年6月

目 录

实验一 OpenGL 程序设计基础	1
一、实验目的	1
二、实验环境	1
三、实验要求	1
四、实验原理	1
五、实验内容及步骤	3
六、思考题	12
实验二 二维基本图元的生成	13
一、实验目的	13
二、实验环境	13
三、实验要求	13
四、实验原理	13
五、实验内容及步骤	18
六、思考题	30
实验三 二维图形区域填充	31
一、实验目的	31
二、实验环境	31
三、实验要求	31
四、实验原理	31
五、实验内容与步骤	36
六、思考题	39
实验四 二维图形的几何变换	41
一、实验目的	41
二、实验环境	41
三、实验要求	41
四、实验原理	41
五、实验内容与步骤	47
六、思考题	62
实验五 二维裁剪	63
一、实验目的	63
二、实验环境	63

三、实验要求	63
四、实验原理	63
五、实验内容与步骤	71
六、思考题	81
实验六 曲线、曲面绘制	82
一、实验目的	82
二、实验环境	82
三、实验要求	82
四、实验原理	82
五、实验内容与步骤	89
六、思考题	92
实验七 图形的投影变换	93
一、实验目的	93
二、实验环境	93
三、实验要求	93
四、实验原理	93
五、实验内容与步骤	99
六、思考题	106
实验八 三维面消隐算法	107
一、实验目的	107
二、实验环境	107
三、实验要求	107
四、实验原理	107
五、实验内容与步骤	114
六、思考题	127
实验九 真实感图形绘制	128
一、实验目的	128
二、实验环境	128
三、实验要求	128
四、实验原理	128
五、实验内容与步骤	135
六、思考题	177
附录 A 实验报告模板	178
参考文献	179

实验一 OpenGL 程序设计基础

一、实验目的

初步了解 OpenGL 程序设计结构；了解 OpenGL 的基本数据类型、核心函数及辅助函数的使用。

- (1) 掌握 OpenGL 相关的基本概念。
- (2) 掌握 OpenGL 的工作机理及基本程序结构。
- (3) 掌握 OpenGL 基本几何元素的绘制。

二、实验环境

- (1) 硬件：学生每人一台 PC 机（CPU 为 Pentium，内存 512 M 以上，硬盘 40 G，显示器 VGA 或更高分辨率，键盘，鼠标）。
- (2) 软件：Windows 操作系统，VC++6.0 版本及以上版本，OpenGL 图形函数库。

三、实验要求

- (1) 本实验要求 2 学时完成。
- (2) 掌握 VC++ 程序开发环境，熟悉 OpenGL 基本程序结构。
- (3) 在 VC++ 环境下，设计实现利用 OpenGL 绘制基本图元的程序。

四、实验原理

（一）OpenGL 概述

OpenGL 是近几年发展起来的一个性能卓越的三维图形标准，它是在 SGI 等多家世界闻名的计算机公司的倡导下，以 SGI 的 GL 三维图形库为基础制定的一个通用共享的开放式三维图形标准。它可与 Visual C++ 紧密接口，便于实现有关计算和图形算法，可保证算法的正确性和可靠性。OpenGL 图形库一共有 100 多个函数，其中核心函数有 115 个。除了提供基本的点、线、多边形的绘制函数外，还提供复杂的三维物体（球、锥、多面体、茶壶等）以及复杂曲线和曲面（如 Bezier、NURBS 等曲线或曲面）的绘制函数。

OpenGL 程序基本结构由 OpenGL 初始化的绘图状态描述和绘制对象组成。OpenGL 为用户提供了 3 个函数库：OpenGL 基本库 `opengl32.lib`，操作函数前缀 `gl`；OpenGL 实用库 `glu32.lib`，操作函数前缀 `glu`；OpenGL 辅助库 `glaux.lib`，操作函数前缀 `aux`。

1. OpenGL 基本函数库

OpenGL 基本函数库用来描述图元、属性、几何变换、观察变换和进行许多其他的操作。

OpenGL 基本函数库中的函数名中每一组成词的第一个字母要大写, 如 `glBegin`, `glClear`, `glCopyPixels`。常量均以大写字母 `GL` 开头, 另外, 常量名中各组成词均采用大写, 单词之间用下划线分隔, 如 `GL_RGB`, `GL_AMBIENT_AND_DIFFUSE`。数据类型名以大写字母 `GL` 开头, 名字中其余部分是用小写字母表示的标准数据类型名, 如 `GLbyte`, `GLshort`, `GLint`。基本函数库中的函数前缀为 `gl`。

2. OpenGL 实用函数库

OpenGL 实用函数库提供了一些例程, 可以设置观察和投影矩阵, 利用线条和多边形近似法来描述复杂对象, 使用线性近似法显示二次曲线和样条曲线, 处理表面绘制操作, 以及完成其他复杂任务。实用函数库中的函数前缀为 `glu`。

3. OpenGL 辅助函数库

OpenGL 辅助函数库建立了一系列简单而又较完整的编程例子, 如初始化窗口、监控输入, 以及绘制一些三维几何体等函数。辅助函数库中的函数前缀为 `aux`。

4. OpenGL 实用函数工具包

OpenGL 实用函数工具包 (OpenGL Utility Toolkit, GLUT) 提供了与任意屏幕窗口系统进行交互的函数库。GLUT 库函数以 `glut` 为前缀, 该库中也包含了描述与绘制二次和样条曲线及曲面的方法。

(二) OpenGL 程序结构

1. OpenGL 状态机制

OpenGL 的工作方式是一种状态机制, 它可以进行各种状态或模式设置, 这些状态或模式在重新改变它们之前一直有效。OpenGL 中大量地使用了这种状态机制, 许多状态变量可以通过 `glEnable()`, `glDisable()` 这两个函数来设置成有效或无效状态, 如是否设置光照、是否进行深度检测等。

2. OpenGL 程序基本结构

(1) 初始化操作

初始化部分主要是设置一些 OpenGL 的状态开关, 如颜色模式的选择, 是否做光照处理、深度检验、裁剪, 等等。这些状态一般都用函数 `glEnable(???)`, `glDisable(???)` 来设置, `???` 表示特定的状态。

(2) 定义窗口

设置观察坐标系下的取景模式和取景框位置大小。

`void glViewport (left,top,right,bottom)`——设置在屏幕上的窗口大小, 四个参数描述屏幕窗口四个角上的坐标 (以像素表示);

`void glOrtho (left,right,bottom,top,near,far)`——设置投影方式为正交投影 (平行投影), 其取景体积是一个各面均为矩形的六面体;

`void gluPerspective (fovy,aspect,zNear,zFar)`——设置投影方式为透视投影, 其取景体积是

一个截头锥体。

(3) 绘制和显示图形

这是 OpenGL 的主要部分, 使用 OpenGL 的库函数构造几何物体对象的数学描述, 包括点线面的位置和拓扑关系、几何变换、光照处理, 等等。

五、实验内容及步骤

(一) 实验内容

这次实验的目的主要是使大家初步熟悉 OpenGL 图形函数库的用法。编程平台是 Visual C++, 它对 OpenGL 提供了完备的支持。

OpenGL 提供了一系列的辅助函数, 用于简化 Windows 操作系统的窗口操作, 使用户能把注意力集中到图形编程上, 这次实验的程序就采用这些辅助函数。

(二) 实验步骤

1. Glut 配置

- ① glut 是一个 win32 dynamic lib, 先下载 glutdlls37beta.zip。
- ② 解压后把 glut32.dll 放到 windows system 目录下。
- ③ 另建目录, 如 E:\glut, 再在 E:\glut 下面建 E:\glut\include\gl, 把头文件 glut.h 放到 E:\glut\include\gl 下面。
- ④ 再在 E:\glut 下面建 E:\glut\lib, 把 glut32.lib 放在 E:\glut\lib 下面。
- ⑤ 最后把 glut 加到工程 (Project) 中 (具体添加方法参见 2 中的操作说明)。

2. 建立 VC++ 执行 OpenGL 的最小环境

(1) 建立基于单文档界面 (SDI) 的最小环境

- ① 启动 VC++, 打开 File/New, 选择 MFC AppWizard(.exe), 并指定 Project name 为 sample。
- ② 选择单文档 SDI (Single Document) 及中文支持。
- ③ 连续按下 Next, 直至 Finish。在最后一个向导窗口单击 OK 即完成。

各操作界面如图 1.1 至图 1.9 所示。

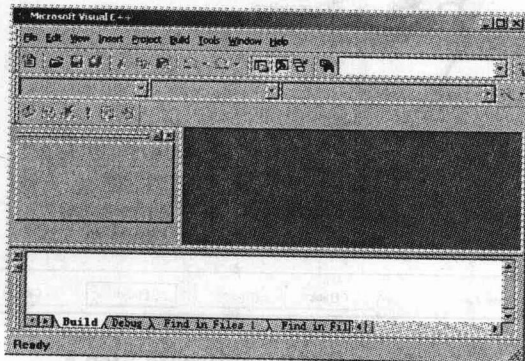


图 1.1 启动 VC++

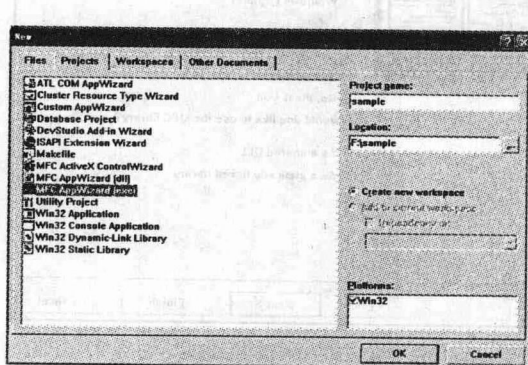


图 1.2 创建工程 sample

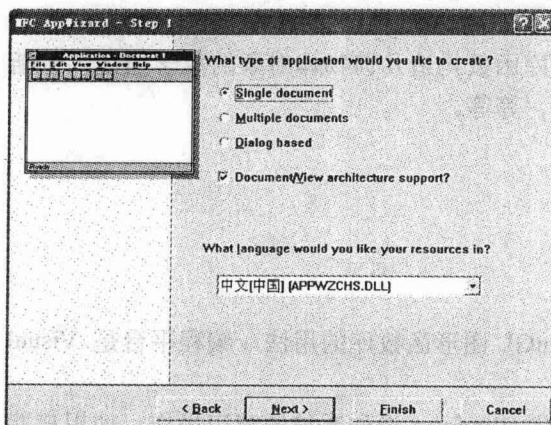


图 1.3 MFC AppWizard-Step 1

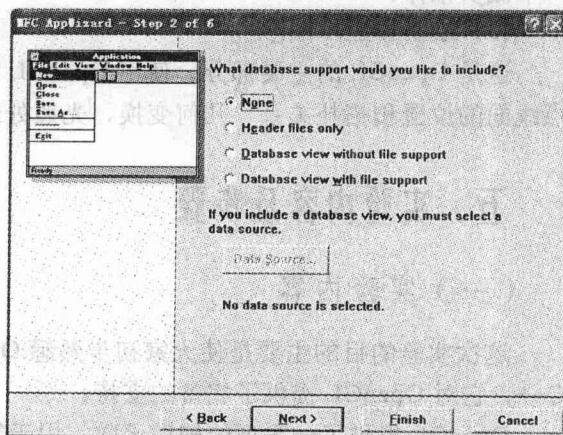


图 1.4 MFC AppWizard-Step 2

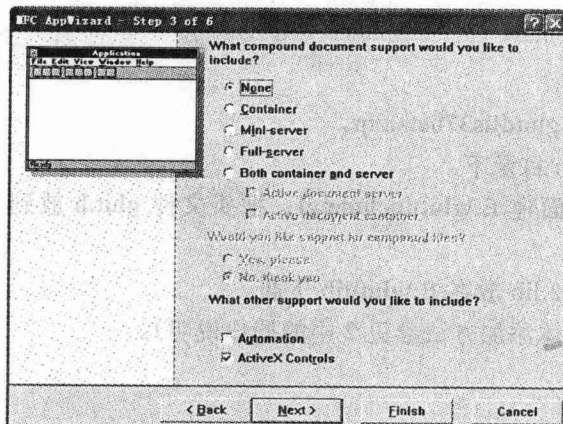


图 1.5 MFC AppWizard-Step 3

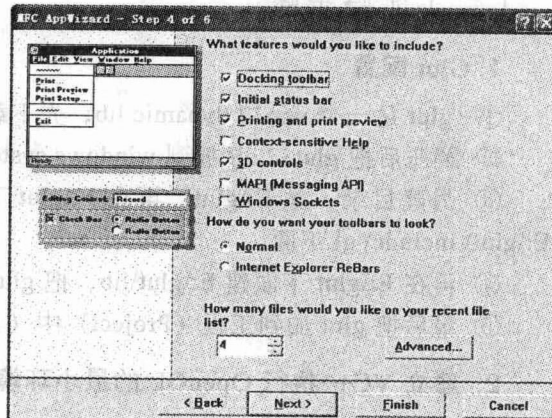


图 1.6 MFC AppWizard-Step 4

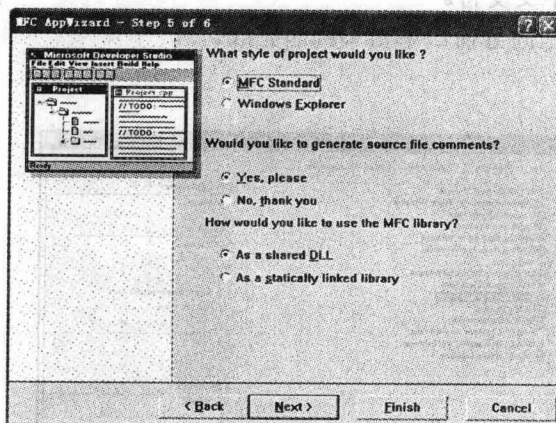


图 1.7 MFC AppWizard-Step 5

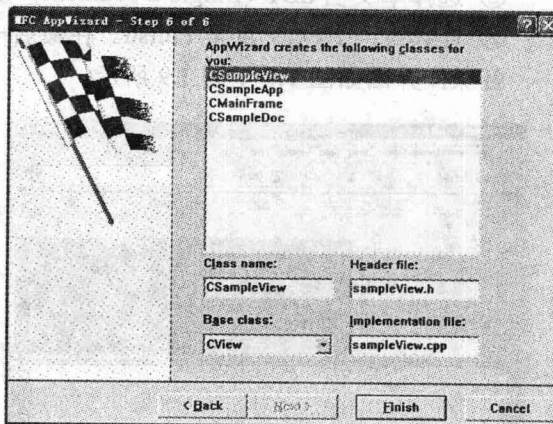


图 1.8 MFC AppWizard-Step 6

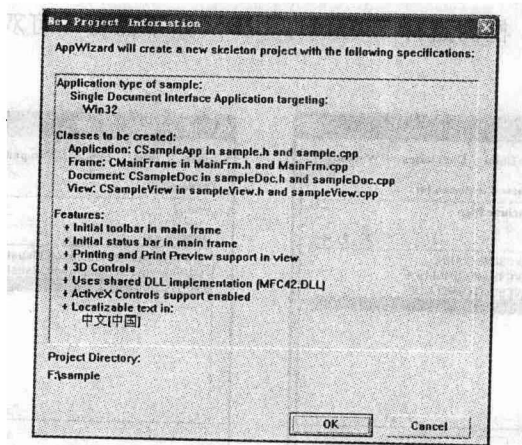


图 1.9 完成 sample 的创建

(2) 修改 CSampleView 类的头文件 sampleView.h

① 在文件头部加入：`#include< glut.h>`。

```
//sampleView.h : interface of the CSampleView class
```

////////////////////////////////////

```
#if !defined(AFX_SAMPLEVIEW_H_6ADDFCE3_15DE_40A0_8B88_A9EF191AFA8C__INCLUD
ED )
```

```
#define AFX_SAMPLEVIEW_H_6ADDFCE3_15DE_40A0_8B88_A9EF191AFA8C_INCLUDED
```

```
#if _MSC_VER > 1000
```

```
#pragma once
```

```
#endif // _MSC_VER > 1000
```

```
// sampleView.h : interface of the CXXView class
```

////////////////////////////////////

```
#include< glut.h>
```

```
#if !defined(AFX_SAMPLEVIEW_H_6ADDFCE3_15DE_40A0_8B88_A9EF191AFA8C__INCLUD
ED )
```

```
#define AFX_SAMPLEVIEW_H_6ADDFCE3_15DE_40A0_8B88_A9EF191AFA8C__INCLUDED_
```

```
#if _MSC_VER > 1000
```

```
#pragma once
```

```
#endif // _MSC_VER > 1000
```

② 把 glut 加到工程 (Project) 中。

为了使程序能够找到 `glut.h` 这个头文件,还需要在 Tools 菜单的 Options 选项中选择 Directories 标签, 在 Show directories for 框内, 选择 Include files 选项, 在 Directories 框内选择 `E:\GLUT\INCLUDE\GL` (如果需要, 可双击浏览选择), 如图 1.10 所示。然后, 选择 Library

files 选项, 在 Directories 框内选择 E:\GLUT\LIB (如果需要, 可双击浏览选择), 如图 1.11 所示。

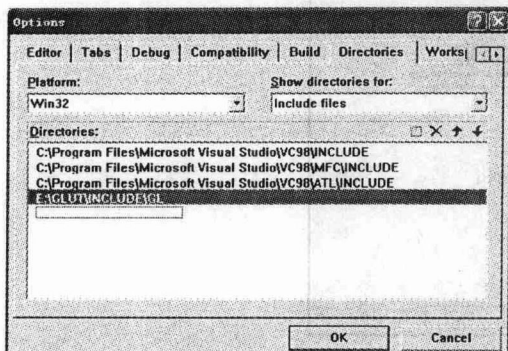


图 1.10 在工程中加入 glut 步骤 1

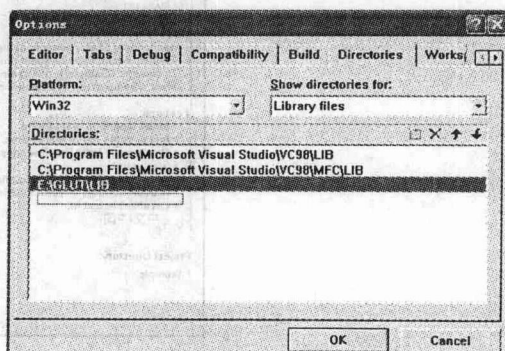


图 1.11 在工程中加入 glut 步骤 2

(3) 在 CSampleView 类中添加各消息函数

使用 View 菜单的 ClassWizard, 添加消息函数。

① 添加对应于消息 WM_CREATE 的消息函数 OnCreate(), 如图 1.12 所示。

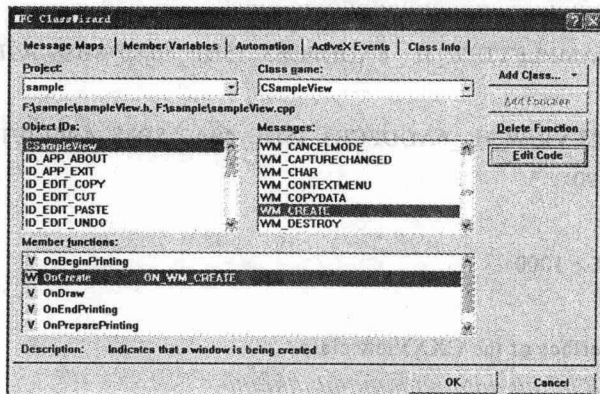


图 1.12 添加消息函数 OnCreate()

② 添加对应于消息 WM_DESTROY 的消息函数 OnDestroy(), 如图 1.13 所示。

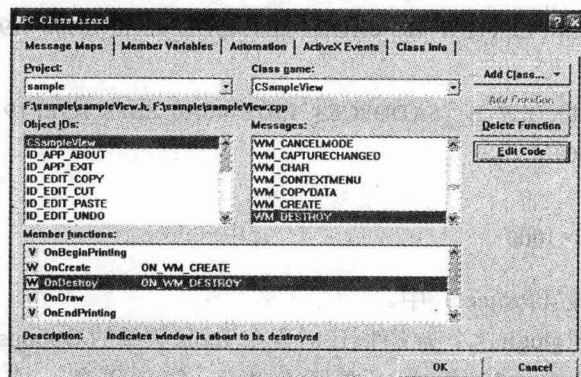


图 1.13 添加消息函数 OnDestroy()

③ 添加对应于消息 WM_SIZE 的消息函数 OnSize(), 如图 1.14 所示。

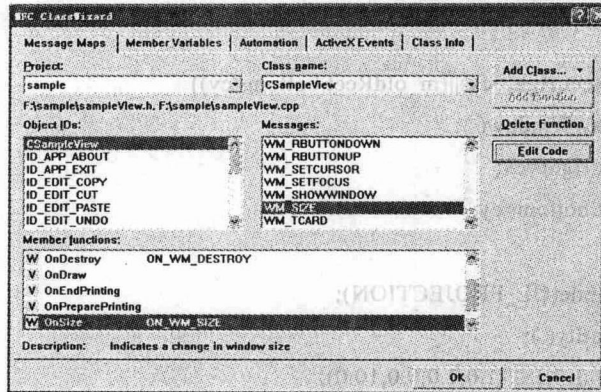


图 1.14 添加消息函数 OnSize()

④ 填入所添加消息函数的内容。

//////////////////////////////////// CScrollView message handlers

int CScrollView::OnCreate(LPCREATESTRUCT lpCreateStruct)

```
{
    if (CView::OnCreate(lpCreateStruct) == -1)
        return -1;

    // TODO: Add your specialized creation code here

    Init();
    return 0;
}
```

void CScrollView::OnDestroy()

```
{
    CView::OnDestroy();
    // TODO: Add your message handler code here

    HGLRC hrc;
    hrc = ::wglGetCurrentContext();
    ::wglMakeCurrent(NULL, NULL);
    if (hrc)
        ::wglDeleteContext(hrc);
    if (m_pDC)
        delete m_pDC;
    CView::OnDestroy();
}
```

void CScrollView::OnSize(UINT nType, int cx, int cy)

```
{
    CView::OnSize(nType, cx, cy);
}
```

```
// TODO: Add your message handler code here
```

```
if(cy>0)
```

```
{
```

```
    if((m_oldRect.right>cx)||((m_oldRect.bottom<cy))
```

```
        RedrawWindow( );
```

```
    m_oldRect.right=cx;
```

```
    m_oldRect.bottom=cy;
```

```
    glMatrixMode(GL_PROJECTION);
```

```
    glLoadIdentity( );
```

```
    glFrustum(-1.0,1.0,-1.0,1.0,0.0,10.0);
```

```
    glViewport(0,0,cx,cy);
```

```
}
```

```
}
```

(4) 添加公有成员函数

① 右键点击 CsampleView，选择 Add Member Function，添加公有成员函数，如图 1.15 所示。

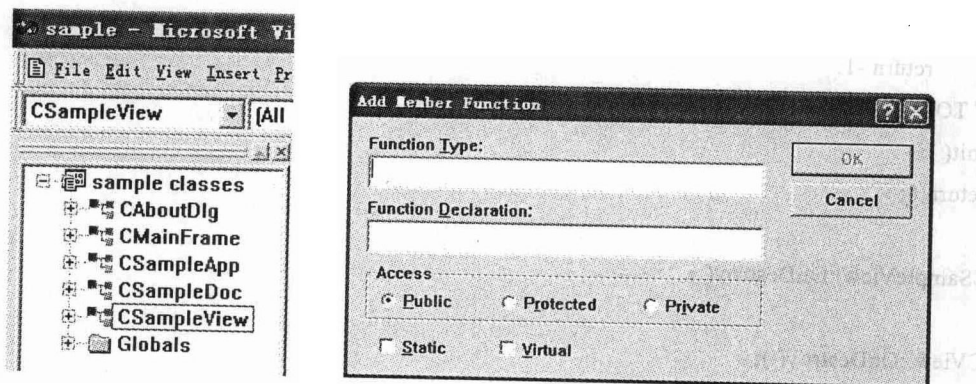


图 1.15 添加 CSampleView 类的成员函数

② 返回类型为 void 的函数 Init(), 用以设置像素格式, 获取 DC 句柄, 设置当前 RC, 如图 1.16 所示。

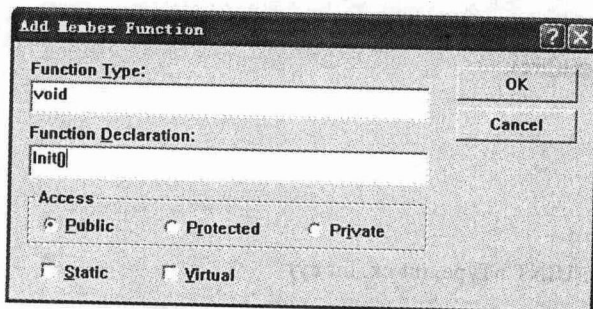


图 1.16 添加成员函数 Init()

③ 返回类型为 `BOOL` 的函数 `bSetupPixelFormat()`，用以初始化像素格式，如图 1.17 所示。

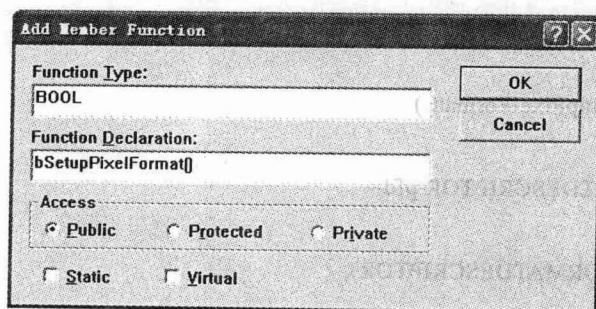


图 1.17 添加成员函数 `bSetupPixelFormat()`

④ 返回类型为 `void` 的函数 `DrawScene()`，用于编写 OpenGL 程序，如图 1.18 所示。

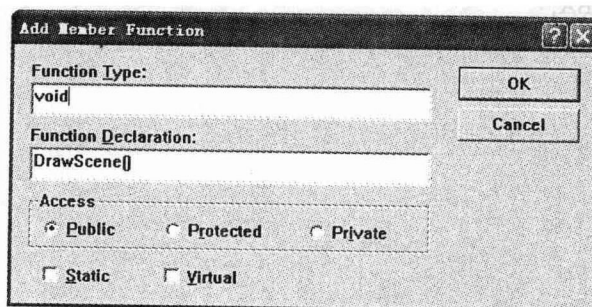


图 1.18 添加成员函数 `DrawScene()`

⑤ 填入所添加成员函数的内容。

```
void CSampleView::Init()
{
    PIXELFORMATDESCRIPTOR pfd;
    int n;
    HGLRC hrc;
    m_pDC=new CClientDC(this);
    ASSERT(m_pDC!=NULL);
    if(!bSetupPixelFormat()) return;
    n=::GetPixelFormat(m_pDC->GetSafeHdc());
    ::DescribePixelFormat(m_pDC->GetSafeHdc(),n,sizeof(pfd),&pfd);
    hrc=wglCreateContext(m_pDC->GetSafeHdc());
    wglMakeCurrent(m_pDC->GetSafeHdc(),hrc);
    GetClientRect(&m_oldRect);
    glClearDepth(1.0f);
    glEnable(GL_DEPTH_TEST);
}
```

```
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity( );
    }

    BOOL CSampleView::bSetupPixelFormat( )
    {
        static PIXELFORMATDESCRIPTOR pfd=
        {
            sizeof(PIXELFORMATDESCRIPTOR),
            1,
            PFD_DRAW_TO_WINDOW|
            PFD_SUPPORT_OPENGL,
            PFD_TYPE_RGBA,
            24,
            0,0,0,0,0,0,
            0,
            0,
            0,
            0,0,0,0,
            32,
            0,
            0,
            PFD_MAIN_PLANE,
            0,
            0,0,0
        };
        int pixelformat;
        if((pixelformat=ChoosePixelFormat(m_pDC->GetSafeHdc( ),&pfd))==0)
        {
            MessageBox("choosepixelformat failed");
            return FALSE;
        }
        if(SetPixelFormat(m_pDC->GetSafeHdc( ),pixelformat,&pfd)==FALSE)
        {
            MessageBox("setpixelformat failed");
            return FALSE;
        }
        return TRUE;
    }
}
```

```
void CSampleView::DrawScene( )
{
    glClearColor(0.0f,0.0f,1.0f,1.0f);
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    glColor3f(1.0f,0.0f,0.0f);
    glBegin(GL_POLYGON);
        glVertex2f(-0.8f,-0.8f);
        glVertex2f(-0.8f,0.8f);
        glVertex2f(0.8f,0.8f);
        glVertex2f(0.8f,-0.8f);
    glEnd( );
    glFinish( );
}
```

(5) 声明两个公有成员对象

① 右键点击 CSampleView 选择 Add Member Variable, 添加公有成员对象: CClientDC *m_pDC, 如图 1.19 所示。

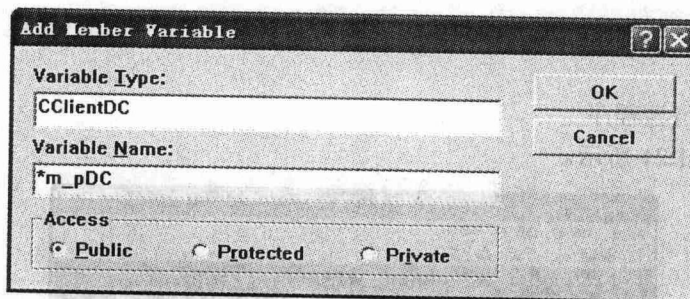


图 1.19 添加公有成员对象 *m_pDC

② 右键点击 CSampleView 选择 Add Member Variable, 添加公有成员对象: CRect m_oldRect, 如图 1.20 所示。

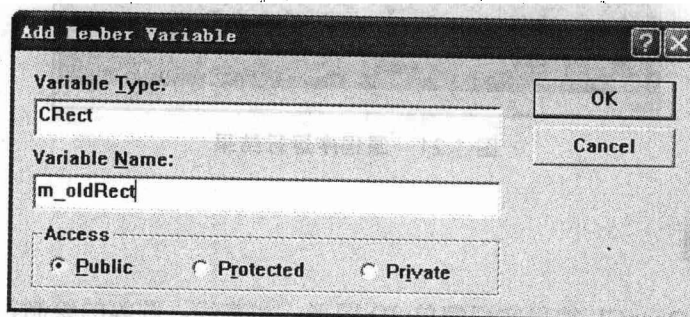


图 1.20 添加公有成员对象 m_oldRect