第 3 章 Chapter 3

计算机图形学

3.1 计算机图形学概述

3.1.1 计算机图形学相关概念

1. 计算机图形定义

CG(Computer Graphics)是计算机图形的缩写。计算机图形学是随着计算机的发展，特别是图形显示器的发展而产生和发展起来的，它是计算机技术、电视技术、图形图像处理技术相互结合的结果。它既包括技术也包括艺术，几乎囊括了当今计算机时代中所有的视觉艺术创作活动，如三维动画、影视特效、平面设计、网页设计、多媒体技术、印前设计、建筑设计、工业造型设计等创作图，如图 3.1 所示。现在，CG 的概念正在扩大，由 CG 和虚拟现实技术制作的媒体文化都可以归于 CG 的范畴。人们使用计算机或手机处理日常事务时，首先看到的便是图形化的人机交互界面，这便是计算机带给人们最直接的感受，社会需求反过来又推动了计算机图形学的快速发展。截至目前，它已经形成一个可观的经济产业。

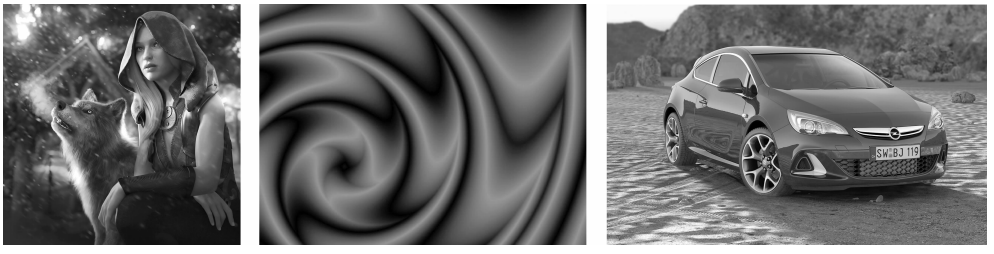

图 3.1　CG 创作图

计算机图形学是一门研究如何运用计算机表示、生成、处理和显示图形的学科。其研究内容非常广泛，如图形硬件、图形标准、图形交互技术、光栅图形生成算法、曲线曲面造型、实体造型、真实感图形计算与显示算法、非真实感绘制、计算机动画、自然景物仿真等。图形主要分为以下两类：①基于线框模型描述的几何图形，主要特点是一种基于三角形网格表示的线框图形。②基于表面模型描述的真实感图形。绘制真实感的图形需要通过建模软件建立场景的几何模型，再利用一些适合的光照模型来计算场景在虚拟的光源、纹理、材质属性下的光照效果。

图形的表示方法一直都是计算机图形学关注的主要问题。其表示方法通常分为参数法和点阵法两种。参数法是在设计阶段建立几何模型时需要用形状参数和属性参数来描述图

形的一种方法，这里的形状参数可以是直线段的起点、终点等几何参数。属性参数又包括对应的直线段的颜色、明暗等非几何参数。点阵法是在绘制阶段就用具有颜色信息的像素点阵来表示图形的一种方法。总体来说，对计算机图形学的学习就是将图形的表示方法从参数法转换到点阵法的一个过程。

2. 计算机图形学发展史

20 世纪中期，第一台图形显示器诞生了，该显示器由美国麻省理工学院研制而成，作为旋风Ⅰ号计算机(图 3.2)的附件，其功能是用来显示简单的图形，并不具备交互功能。后来，美国卡尔康公司在原来联机的数字记录仪基础上将其改进为滚筒式绘图仪。格伯公司将原来的数控机床改进为平板式绘图仪。纵观 20 世纪中期，计算机图形学发展的主要特点是受到计算机设备的限制比较大，机器语言的编程主要用于科学计算，因此相关图形设备仅仅具有输出功能，计算机图形学处于准备和酝酿时期。后来，美国麻省理工学院的林肯实验室在“旋风”计算机上开发了空中防御体系，第一次使用了具有指挥和控制功能的阴极射线管的显示器，操作者可以用笔在屏幕上指出被确定的目标。它预示着交互式计算机图形学的正式诞生。

20 世纪中后期，美国麻省理工学院的史蒂夫教授提出了一种新的理论，即通过插值四条任意的边界曲线来构造曲面。同时，法国雷诺汽车公司的工程师贝塞尔发展了贝塞尔曲线(图 3.3)，以及贝塞尔曲面的理论(贝塞尔曲线也是一直延续到现在非常重要的知识点)，并将其成功地用于几何外形设计，与此同时开发了 UNISURF 系统，用于汽车外形设计。

图 3.2 旋风Ⅰ号

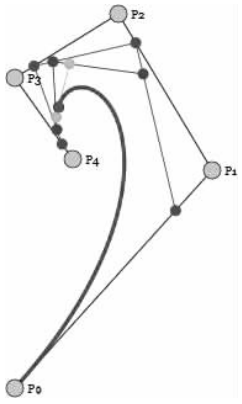

图 3.3 贝塞尔曲线图

之后，计算机图形学的发展步入到了一个重要的历史时期。由于光栅显示器的产生，光栅图形学算法迅速发展起来。类似裁剪、消隐、区域填充等的基本图形学概念及其算法纷纷诞生，计算机图形学开始进入到第一个兴盛的时期。布克奈特提出了第一个光反射模型。高洛德提出“漫反射模型＋插值”的思想，被称为高洛德明暗处理。这些模型的提出可以算是真实感图形学最早的开创性工作。后续相继出现了英国剑桥大学 CAD 小组的 Build 系统、美国罗切斯特大学的 PADL-1 系统等实体造型系统。这两个重要的进展是真实感图形和实体造型技术产生的关键。

随后，怀特提出了一个怀特模型，也就是光透视模型，并给出了光线跟踪算法的范例，实

现怀特模型。美国康奈尔大学和日本广岛大学的学者分别将热辐射工程中的辐射度方法引入到计算机图形学中,用辐射度方法成功模拟了理想漫反射表面间的多重漫反射效果。光线跟踪算法和辐射度算法的提出标志着真实感图形的显示算法已逐渐成熟。超大规模集成电路的发展为图形学的飞速发展奠定了物质基础。随着计算机运算能力的提高、图形处理速度的加快,图形学的各个研究方向得到充分发展,已广泛应用于动画、科学计算可视化、影视娱乐等各个领域。

3.1.2 计算机图形学的应用领域

1. 计算机图形学与游戏

计算机游戏是新兴的娱乐形式,其最大的特色就是能够为游戏参与者提供一个虚拟的空间,从一定程度上可以让人摆脱现实世界中的自我,也可以称为一种沉浸感。当然,计算机游戏的更新技术来自于计算机图形学,其中包括了地形生成、天空和纹理构建、角色动画系统、粒子特效、自然景物的模拟、游戏中的交互系统等。计算机游戏推动着计算机图形学的发展,而计算机图形学也为计算机游戏提供了强大的技术支持。例如,二维计算机游戏的视角比较简单,玩家都是以一种正交投影方式游戏,该方式显示的物体不会根据距离发生变化,因此只涉及二维的屏幕坐标(x,y);三维计算机游戏的视角一般都是透视视角,也就是物体会根据距离收缩,即远小近大,类似于人眼。这种视角就涉及坐标转换问题,需要将屏幕坐标与世界坐标相互转换才能实现。二维游戏、三维游戏和手机游戏如图 3.4 所示。

图 3.4 二维游戏、三维游戏、手机游戏截图

2. 计算机图形学与艺术

动画是计算机艺术的典型代表,是建立在计算机图形基础之上的影视作品,对图形的处理技术要求比较高,部分动画作品如图 3.5 所示。实际上,动画就是对一幅幅静止的图画进行一定的处理,以动态形式展现出来。最早的计算机动画就是在传统卡通片的基础上发展

图 3.5 部分动画作品

起来的。如今，动画制作方式有轴变形、几何建模、混合叠加等，人们更倾向采用建立在物理模型基础上制作动画的方式。这种方式的特点是以弹性力学和流体力学计算出最合适的动画模式，从而达到有效的仿真效果。

3. 计算机图形学与设计

计算机辅助设计和计算机辅助制造是计算机图形学在设计领域中运用最早也是最广泛的两种技术。典型的代表产品为 AutoCAD 系统软件。目前，工程建筑、机械产品设计、飞机、汽车、轮船和电子器件等产品的开发几乎都使用 AutoCAD 软件设计。部分 AutoCAD 作品如图 3.6 所示。

图 3.6　部分 AutoCAD 作品

4. 计算机图形学与虚拟现实

虚拟现实技术实现方式是利用计算机模拟生成一个三维的虚拟世界，跟现实世界可以一样，也可以不一样。使用者使用外置设备（一般指穿戴头盔），沉浸在其中，并获得视觉、听觉、触觉等感官的刺激，在这个过程中借助数据手套、传感器等专业设备与该环境中的人物或物体进行互动，以体验身临其境的感觉。抛开设备的重量感，如何使用户完全沉浸在模拟世界中，就是计算机图形学要解决的问题，其中包括真实感体验、渲染技术等。虚拟现实概念图如图 3.7 所示。

图 3.7　虚拟现实概念图

3.2 计算机图形工具软件

3.2.1 计算机图形非代码类工具

以上详细介绍了计算机图形学的定义、发展史以及应用。那么,该用什么样的工具学习计算机图形学?用什么工具来实现想要的图形?下面针对这两个问题介绍一些工具和软件。

1. 位图图形处理软件

(1) Photoshop。

Photoshop 是 Adobe 公司旗下最出名的图像处理软件之一,它的主要特点是功能强大,可以完成市面上其他类似软件所有的功能,操作比较简单,无需专业的美术功底也可以制作出美丽的效果。其应用领域主要有平面设计、修复照片、广告摄影、影像创意、艺术文字、绘画、绘制或处理三维贴图、图标制作、界面设计、影视后期制作等领域。

(2) Painter。

Painter 是 Corel 公司生产的一款图形处理软件,它的主要特点是拥有全面和逼真的仿自然画笔。它具有上百种绘画工具,有超过 30 种不同的笔刷种类,能够模拟各种画笔的丰富效果,是素描和绘画的优质选择。它广泛应用于动漫设计、绘画制作、美术仿真、建筑设计等领域。

2. 矢量图图形软件

(1) CorelDRAW。

CorelDRAW 是 Corel 公司开发的一款功能强大的矢量图形设计软件。它融合了绘制与编辑、动画制作、位图转换、高品质输出等强大功能;具有相当多的形状和曲线工具;支持组合、调整多个单独形状;属性栏为画面调整和修改提供良好的服务。

(2) Illustrator。

Illustrator 是 Adobe 公司推出的专业矢量绘图工具。它与 Adobe Flash 整合在一起,其路径、锚点、渐层、剪裁遮色片和符号均保持不变。此外也保留图层、群组和物件名称,能够更快速和流畅地在 Illustrator 中绘图。其应用在手绘、动漫、UI 设计及印刷等领域。

(3) Freehand。

Freehand 是 Adobe 公司一个功能强大的平面矢量图形设计软件。它绘制漫画主要用到形状、路径、渐变色填充、混合等工具和功能。其应用领域主要有广告创意、书籍海报、机械制图、建筑蓝图等。

3. 二维动画设计软件

Flash 是由 Macromedia 公司推出的交互式矢量图设计工具,后被 Adobe 公司收购。它的功能有绘图和编辑图形、动画和动作、矢量设计、交互等。其应用领域主要有移动互联网相关的矢量动画设计、应用程序开发、界面开发、游戏开发、多媒体娱乐等。

4. 三维动画设计软件

(1) 3ds Max。

3ds Max 是 Autodesk 公司旗下优秀的电脑三维动画、模型和渲染软件。它不仅可以制作静态模型,也可以绑定动作,但最主要的应用还是在建筑方面。其应用领域主要有影视、

建筑装饰、游戏和设计领域等。

(2) Maya。

Maya 是 Autodesk 公司出品的三维动画制作软件,它不仅可以制作简单的三维模型,还可以与数字化布料模拟、毛发渲染、运动匹配技术相结合。特别是对人物或动物的动作绑定,通过动作调节参数,以达到完美的运动效果。其应用领域主要有现实世界的模拟与仿真,特别是毛发、布料、波浪等;网站资源开发;影视特效;游戏设计及开发学习内容等。

(3) After Effects。

After Effects 是 Adobe 公司生产的一款视频后期合成软件,它专门制作影视后期的特效,提供数百种预设的效果和动画。其应用领域主要有影视制作、商业广告、合成影像、后期制作等。

3.2.2 计算机图形代码类实现工具

1. MFC

MFC(Microsoft Foundation Classes)是微软基础类库的简称,是微软公司实现的一个 C++ 类库,主要封装了大部分 Windows API 函数,VC++ 是微软公司开发的 C/C++ 的集成开发环境。MFC 除了是一个类库以外,还是一个框架,在 VC++ 里新建一个 MFC 的工程,开发环境会自动产生许多文件,同时也会自动生成若干动态链接库,也就是 dll 文件。与此同时,MFC 的内核是封装好的,所以代码中无法查看编辑文件,这样开发人员就可以专心考虑程序的逻辑,而不是做一些重复的工作。

2. Linux 下常用的框架 Qt

Qt 是 Qt 公司开发的一个跨平台的图形用户界面应用程序开发框架。它既可以开发 GUI 程序,也可以用于开发非 GUI 程序,比如服务器。Qt 是面向对象的框架,使用特殊的代码生成扩展以及一些宏,易于扩展,允许组件编程。跨平台集成开发环境 Qt Creator 3.1.0 的正式发布,实现了对 iOS 的完全支持,新增 WinRT、Beautifier 等插件,废弃了无 Python 接口的 GDB 调试支持,集成了基于 Clang 的 C/C++ 代码模块,并对 Android 支持做出了调整,至此实现了全面支持 iOS、Android。

3. GTK+

GTK(GIMP Toolkit)是一个跨平台的图形工具包,按 LGPL(Lesser General Public License)许可协议发布的。虽然最初是为 GIMP 写的,但早已发展为一个功能强大、设计灵活的通用图形库。特别是被 GNOME 选中,使得 GTK+广为流传,成为 Linux 下开发图形界面应用程序的主流开发工具之一。当然,GTK+并不要求必须运行在 Linux 上,事实上,目前 GTK+已经有了成功的 Windows 版本。

以上 3 款代码类软件都有其针对的领域和优点,可以根据需要选择合适的软件组合。适合的软件可以使工作得心应手,效率更高。多种软件综合使用,相互补充,可以发挥各自的优势。

3.3 Visual Studio 2012 软件介绍及实例

3.3.1 Visual Studio 2012 软件介绍

1. Visual C++ 简介

Visual C++ 是微软提供的C++ 开发工具,简称VC++ 。它与C++ 的根本区别就在于C++ 是语言,而VC++ 是用C++ 语言编写程序的工具平台。VC++ 不仅是一个编译器,更是一个集成开发环境,包括编辑器、调试器和编译器等,它一般包含在 Visual Studio 中。Visual Studio 包含了 VB、VC++ 、C# 等编译环境。自微软发布 Visual Studio.NET 以来,建立了在.NET 框架上的代码托管机制,一个项目可以支持多种语言开发的组件。VC++ 同样被扩展为支持代码托管机制的开发环境,所以.NET Framework 是必需的,也就不再有VC++ 的独立安装程序,不过可以在安装 Visual Studio 时只选择VC++ 进行安装。

2. 版本选择

随着VC++ 版本的更新,对C++ 标准的支持越来越好,对各种技术的支持也越来越完善。但同时新版本所需的资源也越来越多,对处理器和内存的要求越来越高。本章将使用 Visual Studio 2012,它的类库和开发技术都是最完善的。

3. 界面介绍

Visual Studio 2012 的开始界面如图 3.8 所示。注意要养成良好的习惯,按照程序的功能或内容来设置文件名和文件路径,放在一个专门的文件夹中。

图 3.8 Visual Studio 的开始界面

单击新建项目后弹出图 3.9 所示对话框。这里需要选择项目类型,并决定项目存放的位置。也可以在页面的左上角单击文件→新建→项目,选择新建的项目类型。

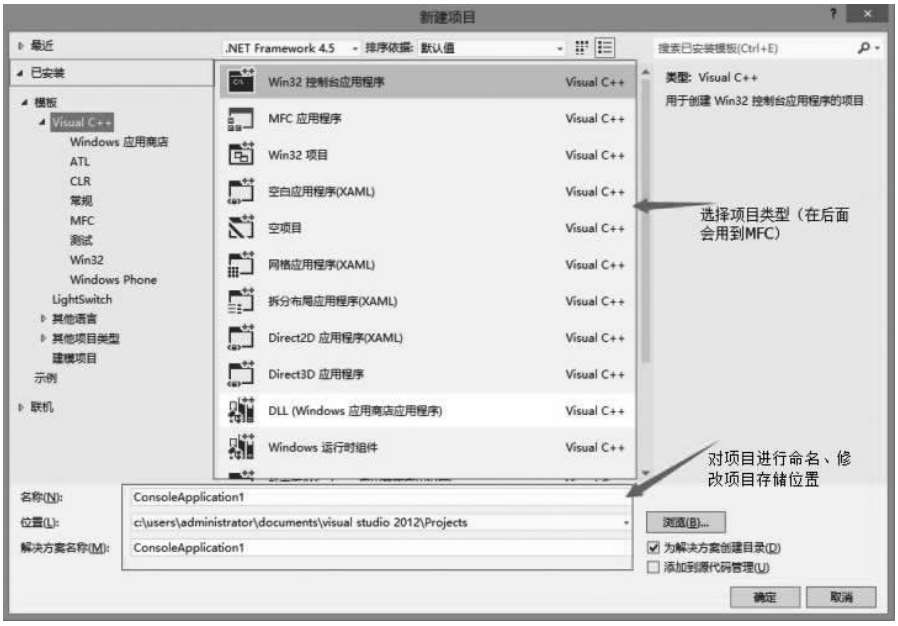

图 3.9　新建项目对话框

接下来介绍常用功能。设计窗体时，工具箱是一个所见即所得的工具，可以直接在里面拖曳控件。首先打开视图→工具箱，然后将其固定在窗口的左端，如图 3.10 所示。

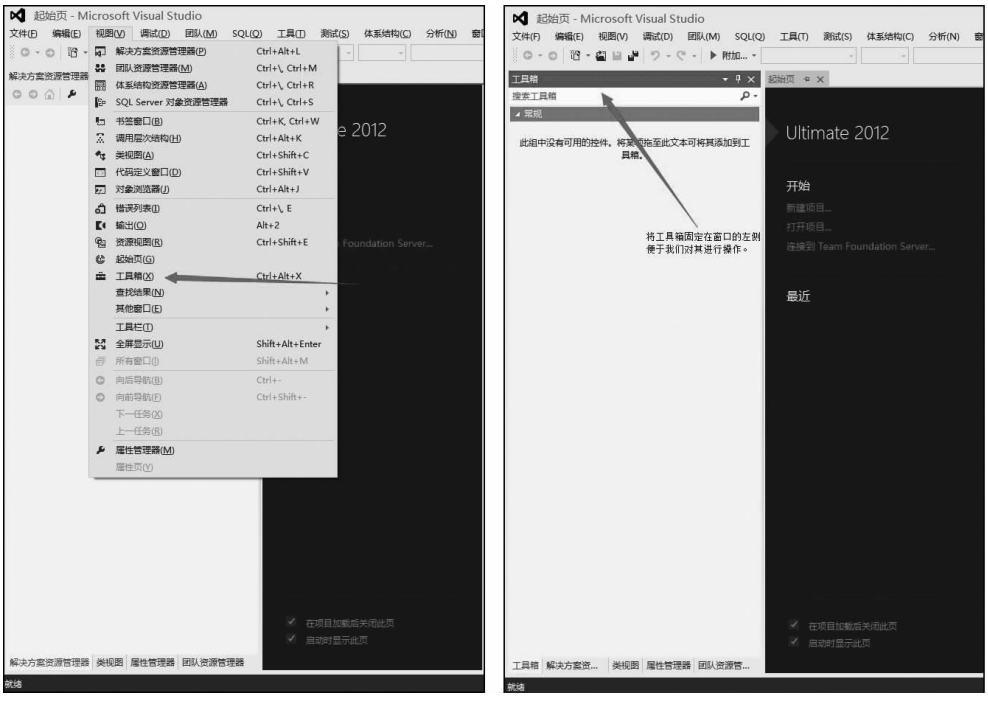

图 3.10　工具箱界面

在资源管理器中可以找到程序的所有文件，单击视图→解决方案资源管理器，如图 3.11

所示。这时就会出现解决方案资源管理器的界面,选择相应的控件即可打开编辑。

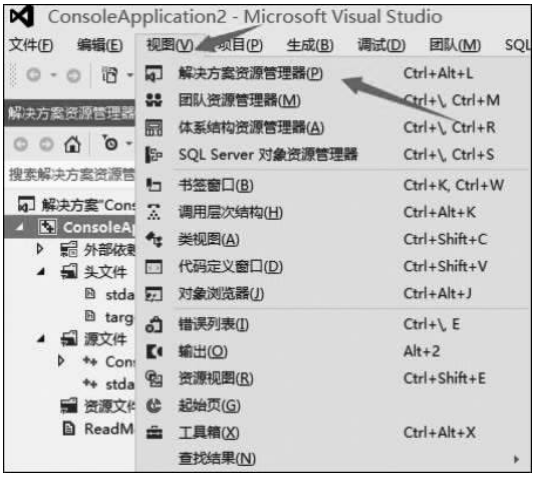

图 3.11 解决方案资源管理器选项

添加控件完成后开始编写程序,进入代码的编写窗口中。单击并选中要编写的控件,然后双击或直接按 F7 键跳转到代码界面。当程序设计好或编写到一半,想要查看一下程序运行的情况,可以单击“调试”按钮,或直接按 F5 键或 Ctrl+F5 键(常用于控制台程序)。单击停止调试即可回到编辑状态。程序调试界面如图 3.12 所示。

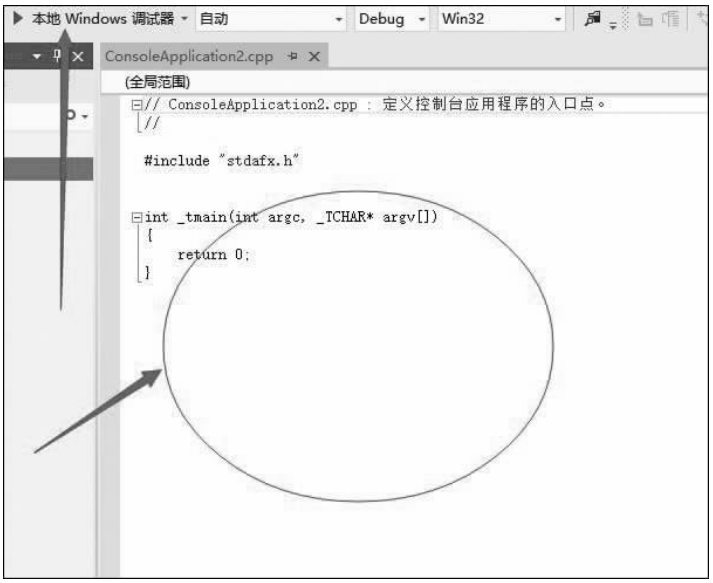

图 3.12 程序调试界面

3.3.2 MFC 实例教程

本节讲解代码方面的理论知识。首先讲解关于类的知识点,再讲解代码的内容,“//”符号后面是对代码的解释。

1. 类的概念

类是面向对象程序设计中的概念，也是面向对象编程的基础。简单来说，类是对现实生活中一类具有共同特征的事物的抽象。类是抽象的，不具体的。就像有些东西可以归为一类，比如狗、猫、猪。它们都是动物，就可以将其归为一类。而猫、狗就可以成为动物类中的一个对象。一组经过用户自定义的类会使这个程序更加简洁。类的内部封装了方法，它们用于更方便地操作类自身的成员。

2. CDC 类的定义

CDC 类提供处理显示器或打印机等设备上下文的成员，以及处理与窗口客户区对应的显示上下文的成员。通过 CDC 类的成员函数进行所有的绘图。CDC 类派生了 CClientDC、CMetaFileDC、CPaintDC 和 CWindowDC 四个子类，每个子类都有其对应的功能，如表 3.1 所示。它还为获取和设置绘图属性、处理视点、窗口扩展、转换坐标、处理区域、粘贴、绘制直线及绘制简单椭圆和多边形等形状提供了成员函数。使用 CDC 对象时首先要构造它，然后再调用成员函数。

表 3.1 CDC 派生的 4 个子类

类 名	拓展内容
CClientDC	① CClientDC 类只能在客户区绘图； ② 所谓客户区，指窗口区域中去掉边框、标题栏、菜单栏、工具栏、状态栏等之外的部分，它是用户可以操作的区域； ③ 使用 CClientDC 绘图时，一般要调用 GetClientRect()函数来获取客户区域的大小； ④ CClientDC 类在构造函数中调用 Windows API 函数 GetDC()，在析构时响应 ReleaseDC()； ⑤ CClientDC 类的窗口句柄保存在成员变量 m_hWnd，为构造 CClientDC，需将 CWnd 作为参数传递给构造函数
CWindowDC	① CWindowDC 对象在构造时调用 Windows API 函数 GetWindowDC()，在析构时调用相应的 API 函数 ReleaseDC()，这意味着 CWindowDC 对象可访问 CWnd 所指向的为整个全屏幕区域； ② CWindowDC 允许在显示器的任意位置绘图。坐标原点在整个窗口的左上角。 ③ 使用 CWindowDC 绘图时，一般要调用 GetWindowRect()函数来获取整个应用程序窗口区域的大小； ④ CWindowDC 类的窗口句柄保存在成员变量 m_hWnd，为构造 CClientDC，需将 CWnd 作为参数传递给构造函数
CPaintDC	① 通常 CPaintDC 用来响应 WM_PAINT 消息。一般应用在 OnPaint()函数； ② CClientDC 也是从 CDC 派生出来的。构造时自动调用 GetDC()函数，析构时自动调用 ReleaseDC()函数，一般应用于客户区窗口的绘制； ③ CPaintDC 只能在 WM_PAINT 消息中使用，用于有重画消息发出时才使用的内存设备环境，而 CClientDC 和客户区相关，有重画消息发出时才使用的内存设备环境，可在任何地方使用； ④ 在处理窗口重画时，必须使用 CPaintDC，否则 WM_PAINT 消息无法从消息队列中清除，将引起不断的窗口重画

续表

类　名	拓展内容
CMetaFileDC	① 在应用程序中,有一些图像是需要经常重复显示的。这样的图形最好事先绘制好,形成一个文件,并存储在内存中,用到它时直接打开就可以了,这种图形文件叫作图元文件; ② 制作图元文件需要一个特殊的设备描述环境 CMetaFileDC 类。它也是由 CDC 类继承来的,因此包含了 CDC 类的所有绘图方法; ③ 一般先在视图类的 OnCreate()函数中创建图元文件。具体做法为先定义一个 CMetaFileDC 类的对象,然后用该对象的 Create()函数创建它,该函数的原型为 BOOL Create(LPCTSTR lpszFilename=NULL); ④ 接下来使用由 CDC 继承的绘图方法绘制图元文件,最后使用 Close()函数结束绘制,并保存该图元文件到类的数据成员中(该数据成员的类型应为 HMETAFILE)。 ⑤ 需要显示该图元文件时,使用 CDC 类的成员函数 PlayMetaFile()。不再使用该图元文件时,要用函数 DeleteMetaFile()将其删除

3. 常用绘图类

CPoint、CRect、CSize 是对 Windows 的 POINT、RECT、SIZE 结构体的封装,因此可以很方便地使用其成员和变量,如表 3.2 所示。

表 3.2　常用绘图类及结构定义

类　名	结构定义

CPoint 类	存放二维点坐标(x,y)。POINT 结构体的定义为:
	`typedef struct tagPOINT`
	`{`
	`  LONG x;  //点的 x 坐标`
	`  LONG y;  //点的 y 坐标`
	`}`
	`POINT, * PPOINT;`

CRect 类	存放矩形左上角点和右下角点的坐标,对应的 RECT 结构体定义为:
	`typedf struect _RECT`
	`{LONG left;  //左上角点的 x 坐标`
	`LONG top;  //左上角点的 y 坐标`
	`LONG right;  //右下角点的 x 坐标`
	`LONG bottom;  //右下角点的 y 坐标`
	`}RECT, * PRECT;`

CSize 类	存放矩形 x 方向的长度和 y 方向的长度(cx,cy)。对应的 SIZE 结构体定义为:
	`typedef struct tagSIZE`
	`{`
	`  LONG cx;  //矩形的宽度`
	`  LONG cy;  //矩形的高度`
	`}SIZE, * PSIZE;`

4. 绘图工具类(表 3.3)

表 3.3 绘图工具类及其用法

绘图工具类名	用 法 及 内 容
CGdiObject	GDI 绘图工具的基类,一般不能够直接使用
CBitmap	封装了一个 GDI 位图,提供位图操作的接口
CBrush	封装了 GDI 的画刷,可以选作设备上下文的当前画刷,用于填充封闭图形的内部
CFont	封装了 GDI 字体,可以选作设备上下文的当前字体
CPalette	封装了 GDI 调色板,提供应用程序和显示器之间的颜色接口
CPen	封装了 GDI 画笔,可以选作设备上下文的当前画笔,用来绘制图形边缘的线
CRgn	封装了一个 Windows 的 GDI 区域,这一块区域用作某一窗口中的一个椭圆或多边形区域

5. 映射模式

什么是映射模式呢? 就是把图形显示在屏幕坐标系中的过程。根据映射模式的不同,也可以分为逻辑坐标和设备坐标。当然,MFC 也提供了几种不同的映射模式来适应不同的需求。这包括 MM_TEXT、MM_LOMETRIC、MM_HIMETRIC、MM_LOENGLISH、MM_HIENGLISH、MM_TWIPS、MM_ISOTROPIC、MM_ANISOTROPIC 等模式。注意,在默认情况下,一般使用的设备坐标系为 MM_TEXT,其特点是坐标原点位于客户区左上角,x 轴水平向右,y 轴垂直向下,坐标基本单位为一个像素。

使用映射模式时也需要对应的映射函数。使用各向同性的映射模式 MM_ISOTROPIC、各向异性的映射模式 MM_ANISOTROPIC 时,需要调用的映射函数是 SetWindowExt()和 SetViewportExt(),它们可以用来改变窗口和视图区域的设置。其余模式则不需要调用。而 MM_TEXT、MM_LOMETRIC、MM_HIMETRIC、MM_LOENGLISH、MM_HIENGLISH、MM_TWIPS 等映射模式主要应用于使用物理单位(英寸/毫米)来绘图的情况。

这几种映射模式都是写在 OnDraw 函数中,下面以 MM_ANISOTROPIC 模式为例。

```
Void CTestView::OnDraw(CDC *pDC)
{
    CTestDoc * pDoc=GetDocument();
    ASSERT_VALID(pDoc);
    CRect rect;                          //此语句表示声明了一个客户区为矩形的面板
    GetClientRect(&rect);                //在声明好的客户区中获取其坐标
    pDC->SetMapMode(MM_ANISOTROPIC);     //SetMapMode 为映射模式中的方法
    ...
}
```

6. GDI 对象的使用

GDI 是图形设备接口(Graphics Device Interface 或 Graphical Device Interface)的简称,是微软公司的视窗操作系统(Microsoft Windows)的三大内核部件之一,也是微软视窗系统表征图形对象的标准。

在 MFC 中需要使用 GDI 对象,具体内容如表 3.4 所示。

表 3.4 GDI 对象函数

函数用途	类　　属	语　　句	返 回 值	参数解释	备　　注
画笔创建函数	CPen::CreatePen	BOOL CreatePen(int nPenStyle, int nWidth,COLORREF crColor);	成功调用为“非0”,否则为“0”	nPenStyle 是画笔的样式;nWidth 是画笔的宽度;crColor 是画笔的颜色	画笔也可以使用构造函数直接定义。将语句前面的 BOOL CreatePen 改为 CPen 即可
画刷创建	CBrush:: CreateSolidBrush.	BOOL CreateSolidBrush (COLORREF crColor);	成功调用为“非0”,否则为“0”	crColor 是画刷的颜色	实体画刷使用指定的颜色填充图形的内部,也可以使用构造函数直接定义。语句前面改为 CBrush
选入 GDI 对象	CDC::SelectObject。	CPen*SelectObject(CPen*pPen); CBrush * SelectObject (CBrush * pBrush); CBitmap*SelectObject(CBitmap*pBitmap)	如果成功返回,将被替换对象的指针;否则返回 NULL	pPen 是将要选择的画笔对象指针;pBrush 是将要选择的画刷对象指针;pBitmap 是将要选择的位图对象指针	本函数将设备上下文的原 GDI 对象更换为新对象,同时返回指向原对象的指针
删除 GDI 对象	CGdiObject:: DeleteObject	BOOL DeleteObject();	成功删除 GDI 对象,则返回“非 0”;否则返回“0”	无	GDI 对象使用完毕后,如果程序结束,会自动删除 GDI 对象;不能使用 DeleteObject()函数删除正在被选入设备上下文中的 CGdiObject 对象。
选入库对象	CDC:: SelectStockObject。	VirtualCGdiObject*SelectStockObject (int nIndex);	调用成功,则返回被替代的 CGdiObject 类对象指针;否则返回 NULL	nIndex 可以是常用的库里面的画笔代码	库对象的返回类型是 CGdiObject*,使用时需要根据具体情况进行相应转换

7. MFC 图形绘制实例

MFC 有几种常见的绘图方法，包括 OnDraw()成员函数可以直接绘图、使用菜单绘图、使用自定义函数绘图等。在即将讲解的案例中，绘图方法以使用 OnDraw()成员函数直接绘图为主，并且通过 CDC 类中的主要几种绘图成员函数来绘制简单的图形。

实例一：绘制像素点。

知识点：MFC 在 Visual Studio 2012 中的使用方法。

CDC 类成员中绘制像素点的函数使用。

OnDraw()函数的绘制过程。

MFC 中语法的基本编写方式。

步骤 1：打开 Visual Studio 2012 软件，新建名为 ProjectOne 的项目，选择 MFC 应用程序并单击“确定”按钮，弹出界面如图 3.13 所示。

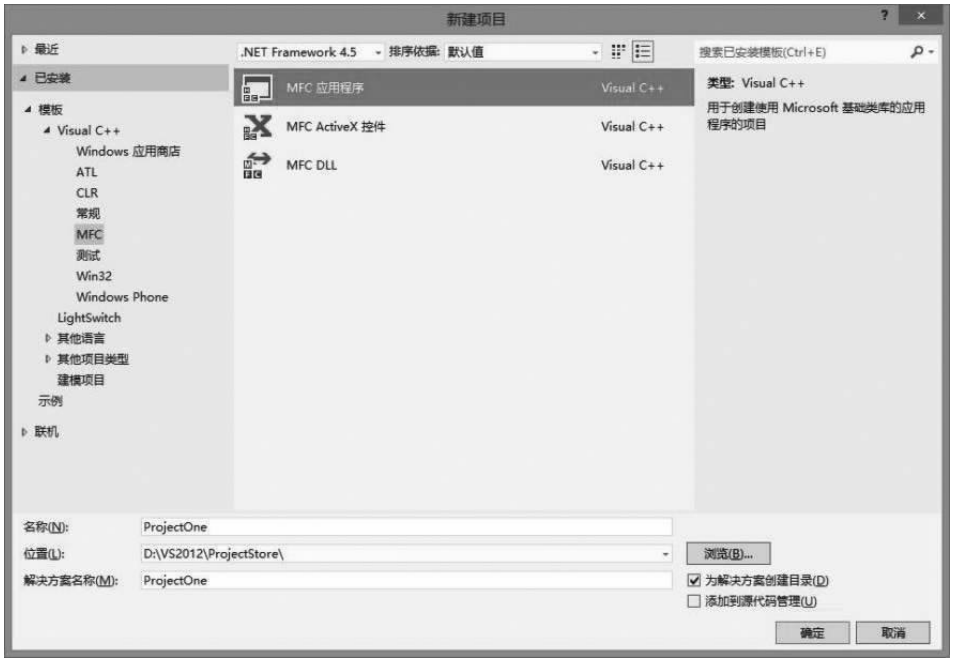

图 3.13　建立 MFC 项目

步骤 2：在应用程序类型里选择单个文档，项目类型为 MFC 标准，然后单击“完成”按钮即可，向导界面如图 3.14 所示。

步骤 3：进入项目界面，左边包括程序源文件(*.cpp)、头文件(*.h)和资源文件(*.ico、*.bmp 等)，如图 3.15 所示。接下来使用头文件下的 ProjectOneView.h 和源文件下的 ProjectOneView.cpp 两个文件。然后双击 ProjectOneView.cpp，在右侧代码编写面板处找到 OnDraw()函数。

步骤 4：如果在 OnDraw()函数里直接编写代码，会出现错误，需要将图 3.16 中箭头所指的/*和*/去掉，目的是注销一些语句。完成此项工作后，就可以在 OnDraw()函数里面正常编写绘制像素点的代码了，详细代码如下。

图 3.14　MFC 应用向导

图 3.15　程序文件

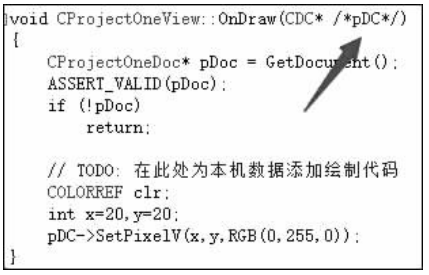

图 3.16　注释部分

```
Void CProjectOneView::OnDraw(CDC * pDC)
{
    CProjectOneDoc * pDoc = GetDocument();
    ASSERT_VALID(pDoc);
If(!pDoc)
return;
//TODO: 在此处为本机数据添加绘制代码
```

```
    COLORREF clr;       //声明一个 COLORREF 类型的变量,名字为 clr,用来存放像素点的颜色
    int x=20,y=20;                  //定义了两个整型的参数 x、y 并赋值
    pDC->SetPixelV(x,y,RGB(0,255,0));
                          //SetPixelV 是绘制像素点的方法,代表在(20,20)处绘制红色点
    Clr=pDC->GetPixel(x,y);         //将绘制好的点颜色赋值给 clr
    pDC->SetPixelV(x+100,y,clr);  //绘制第二个点
}
```

步骤 5：单击工具栏的 ▶ 本地 Windows 调试器 按钮进行程序调试。目的是在(20,20)和(20＋100,20)这两处坐标绘制像素点 1、像素点 2。由于绘制的是像素点,所以特别小,图 3.17 所示是放大后显示出来的效果。

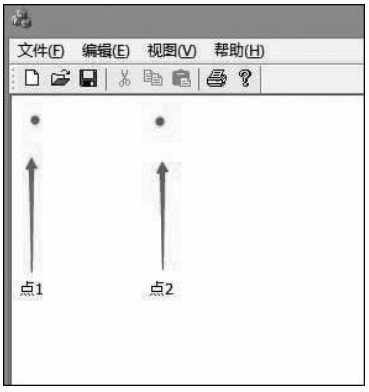

图 3.17 像素点示意图

实例二：绘制面(矩形)。

知识点：画笔的声明。

画笔类型的了解。

矩形函数。

MFC 中参数的熟练运用。

步骤 1：依旧在 OnDraw()函数里面编写代码。编写前需要考虑两点：矩形 4 个点的确定,矩形边缘线条和内部颜色填充分别需要画笔和笔刷。这能使编程思路更加清晰。具体代码和注释如下。

```
void CProjectOneView::OnDraw(CDC * pDC)
{
    CProjectOneDoc * pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
    // TODO:在此处为本机数据添加绘制代码
        CPen NewPen, * pOldPen;     //声明一个新画笔对象和原画笔指针
    NewPen.CreatePen(PS_SOLID,1,RGB(255,0,0));
                          //创建一个新的 1 像素宽的红色实线画笔;PS_SOLID 代表画笔的样式
    pOldPen=pDC->SelectObject(&NewPen);
                                    //将创建的新画笔选入设备上下文中,以便程序读取
    CBrush NewBrush, * pOldBrush;     //声明新的笔刷对象和原笔刷指针。填充矩形内部
    NewBrush.CreateSolidBrush(RGB(249,104,240));    //创建实体的粉色笔刷来填充矩形
    pOldBrush=pDC->SelectObject(&NewBrush);    //同样将新建的笔刷也选入设备上下文
    pDC->Rectangle(100,100,600,300);
          //绘制矩形,Rectangle 是绘制矩形的方法,括号里的参数代表矩形(4 个坐标)的位置
    pDC->SelectObject(pOldBrush);       //以上语句已绘制完成,本语句代表恢复原有笔刷
    NewBrush.DeleteObject();   //因为绘制完成,所以要清理掉之前的笔刷,否则会占用内存
    pDC->SelectObject(pOldPen);         //同样恢复原有画笔
    NewPen.DeleteObject();         //最后再删除画笔
}
```

步骤 2：运行项目，效果如图 3.18 所示。

图 3.18 粉色矩形示意图

注意：本程序用到了画笔类型(PS_SOLID)实现画笔。表 3.5 所示为几款常用的画笔。

表 3.5 常用画笔

画笔样式	线条样式	宽度值
PS_SOLID	实线	随意指定
PS_DASH	虚线	1 或者更小
PS_DOT	点线	1 或者更小
PS_DASHDOT	点画线	1 或者更小
PS_DASHDOTDOT	双点画线	1 或者更小
PS_NULL	不可见线	随意指定
PS_INSIDEFRAME	内框架线	随意指定

课后习题

1. 填空题

(1) 图形的表示方法有_____和_____。

(2) 第一台图形显示器是由_____研制而成的。

(3) _____提出了第一个光反射模型。

(4) 高洛德提出_____的思想，被称为高洛德明暗处理。

(5) 根据映射模式的不同，映射模式也可以分为_____和_____。

2. 简答题

(1) 参数法的定义是什么？

(2) 点阵法的定义是什么?

(3) 纵观 20 世纪中期,计算机图形学发展的主要特点是什么?

(4) 类的概念是什么?

(5) 什么是映射模式呢?

3. 实践题

请根据之前所学,运用 MFC 绘制一个椭圆。椭圆的绘制方法为 Ellipse(int x1,int y1,int x2,int y2)。

参考文献

[1] 孔令德.计算机图形学:基于 MFC 三维图形开发[M]. 2 版. 北京:清华大学出版社,2021.

[2] 陆玲. Visual C++ 数字图像处理[M]. 2 版. 北京:中国电力出版社,2021.

[3] 孔令德.计算机图形学[M].北京:清华大学出版社,2021.

[4] 任哲. MFC Windows 应用程序设计[M]. 3 版. 北京:清华大学出版社,2013.