

◎尚晶晶 编著

Direct3D

游戏开发 技术详解



- 全面系统的Direct3D基础知识
- 经典的地形渲染技术（高度图户外地形系统，BSP室内地形系统，纹理混合技术，连续精细度绘制技术）
- 主流3D动画技术（蒙皮骨骼动画，Morphing动画技术）
- 精彩的3D游戏特效技术（公告牌技术，粒子系统，光度图渲染，精灵特效，凹凸贴图技术，环境贴图技术，阴影图技术，阴影体技术）
- 最新的GPU Shader渲染技术（High Level Shader Language）



人民邮电出版社
POSTS & TELECOM PRESS

Direct3D

游戏开发

技术详解



- 1. Direct3D 11 和 Direct3D 12 的架构和性能
- 2. Direct3D 11 和 Direct3D 12 的渲染管线
- 3. Direct3D 11 和 Direct3D 12 的纹理管理
- 4. Direct3D 11 和 Direct3D 12 的几何体管理
- 5. Direct3D 11 和 Direct3D 12 的着色器管理
- 6. Direct3D 11 和 Direct3D 12 的同步和屏障
- 7. Direct3D 11 和 Direct3D 12 的调试和性能分析



◎尚晶晶 编著

Direct3D

游戏开发 技术详解



人民邮电出版社
POSTS & TELECOM PRESS

图书在版编目 (CIP) 数据

Direct3D 游戏开发技术详解/尚晶晶编著. —北京: 人民邮电出版社, 2006.2

ISBN 7-115-14309-9

I. D... II. 尚... III. ①多媒体—软件工具, Direct3D—程序设计②三维—动画—游戏—应用程序—程序设计 IV. ①TP311.56②G899

中国版本图书馆 CIP 数据核字 (2006) 第 007422 号

内 容 简 介

本书分为两部分, 第 1 部分是基础部分, 全面系统地讲述了 Direct3D 游戏编程所需的基础知识, 并辅以针对性的训练和实例。第 2 部分是进阶部分, 全面讲述了现代 3D 游戏中非常经典的技术和一些国内外最新的技术, 包括地形渲染所涉及的关键技术和方法、主流 3D 动画技术、各种 3D 游戏实现的特效技术, 以及较新的基于可编程渲染管线的 GPU Shader 技术。

全书讲解配合有大量实战性程序代码, 并结合了当前市面上最新的各种 3D 游戏。

本书适合 Direct3D 游戏编程技术的入门者循序渐进地学习, 也适合有一定基础、希望提高 Direct3D 游戏编程能力的爱好者阅读。

Direct3D 游戏开发技术详解

◆ 编 著 尚晶晶
责任编辑 刘 浩

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京密云春雷印刷厂印刷
新华书店总店北京发行所经销

◆ 开本: 787×1092 1/16
印张: 26.5
字数: 646 千字 2006 年 2 月第 1 版
印数: 1—5 000 册 2006 年 2 月北京第 1 次印刷

ISBN 7-115-14309-9/TP · 5167

定价: 48.00 元 (附光盘)

读者服务热线: (010) 67132692 印装质量热线: (010) 67129223

序

电子游戏的诞生，意味着人类的娱乐发展出一个全新的分支——互动娱乐。电子游戏用它那独特的魅力吸引着人们的眼球。随着科技的进步和计算机的计算能力不断提高，现在的电子游戏已经开始模拟现实，创造人类有史以来最真实的视听娱乐感受。随着 PC、PS、XBox 和任天堂的发展与普及，电子游戏娱乐已经不再是个别狂热玩家的玩物，它已经开始成为最容易接触也是最便宜的娱乐方式之一。

随着电子游戏的发展，越来越多的人开始加入到游戏开发的行列中。但是游戏开发是一个系统工程，涉及多种技术。其中，图形引擎的开发是游戏开发的重点。现在看到的关于游戏图形开发的书籍大多都属于手册类，针对某种图形的 API（比如 DirectX 或者 OpenGL）进行描述，这类书籍更接近于 API 的使用指南。而本书在写作上与手册类的书籍有所不同，作者没有把很多精力放到具体的某个 API 上，而是注重与实际事例相结合，由浅入深地阐述各种技术的实现方法，并加上大量图片以便读者理解。

相信本书会在三维游戏开发技术上，给希望进入游戏行业的开发人员提供极大的帮助。我想，这也正是作者如此努力的意义所在。

罗金海

2006.1.1 于上海

我们正迎来国内游戏业的春天。自从“传奇”缔造了中国网络游戏运营史上的一个奇迹以后，国内市场蕴含的巨大潜力终于被释放出来。是的，网络游戏！这和数年前互联网终于找到手机付费途径一样，游戏业终于踏出了盗版包围圈，走上了高速发展的道路。

我们同时面临机遇与挑战。机遇是游戏业的价值终于得到了认可，政府也将网络游戏通用引擎研究以及示范产品开发列入国家 863 计划重点支持，大量风险资金投入游戏研发和运营，不少公司正逐渐成长起来。而挑战则是随着世界经济的一体化，国内市场正面临国外世界顶级游戏公司的重量级产品的冲击，他们在单机游戏研发方面也体现了相对于我们的巨大优势。我们在游戏开发技术方面的普及工作也是相当不足，在高校里比较难学到最新的计算机图形学知识，同时市场上专业的游戏研发书籍也不多见。

本书即是献给国内所有热爱游戏研发的朋友们。本书一方面是笔者学习大量国外先进技术资料，包括游戏开发网站、大师的游戏编程著作、计算机图形学论文等的成果，另一方面也是笔者游戏制作经验的汇集。

本书分为基础部分和进阶部分。基础部分为第 1 章～第 9 章，目标是使没有 DirectX SDK 编程经验的读者也可以快速上手，对 Direct3D 编程的一般概念和知识做出了系统讲解，并在第 9 章引导读者综合使用学习过的知识进行简单的引擎封装和调用。进阶部分为第 10 章～第 14 章，全面讲述了现代 3D 游戏中非常经典的技术和一些新的技术，本书进阶部分绝大多数内容都是国内中文书籍中首次出现的。全书最大的特点就是实战性非常强，每项技术的代码级实现是讲解的核心内容。即使是较为复杂和更高级的内容，本书也力图提供给读者足够的背景知识和入门介绍，方便读者进一步研究和学习。本书进阶部分深入介绍和探讨的技术包括：3D 动画技术 (Skin Mesh, Morphing)、地形渲染技术 (Height Map, BSP, Texture Blending, CLOD 等)、公告牌技术 (Billboard)、粒子系统 (Particle System)、光度图技术 (Light Map)、精灵特效技术 (Sprite)、凹凸映射技术 (Bump Map)、环境映射技术 (Environment Map)、阴影图技术 (Shadow Map)、体积阴影技术 (Shadow Volume)、顶点渲染器技术 (Vertex Shader)、像素渲染器技术 (Pixel Shader)、高级渲染器语言 (HLSL) 等。在第 14 章，分析了大型 3D 游戏 Donuts4，深入剖析了其物体模型系统、地形渲染系统和碰撞检测系统，并对其代码做了适当的导读。相信全书的设计和 content 对想学习 3D 游戏开发技术的读者来说是一次全新的、系统的体验，我们介绍的很多技术结合了目前最新的游戏，例如地形渲染和反恐精英，公告牌技术和魔兽世界，环境映射与极品飞车 8，体积阴影技术和 DOOM III 等。

虽然本书的定位非常适合没有 DirectX SDK 编程经验的初学者，但是并不意味着读者如果没有相关基础的话也可以轻松阅读本书，3D 游戏编程可不像某些领域一样学几句语句就能抄刀上阵，你可能需要花费一年甚至更多的时间才能成为高手。因此，你应该熟练使用 C++，熟悉 Win32 编程，最好学习过一些相关的图形学知识以及线性代数方面的知识。

在本书的创作过程中,我得到了很多朋友的热心帮助。感谢国内第一代 3D 游戏开发者、大型 3D 网络游戏“真封神”主程序员、上海游戏米果总经理罗金海先生,他一直以来给予我鼓励和帮助;感谢广州沃思信息科技有限公司项目总监陈兴充先生为本书提供了很多面向读者的创作建议;感谢 3D 主程序员赖滔为本书的特效精灵部分制作了示例程序,并在我困惑的时候为我讲解了一些关键性技术;感谢 3D 动画师杨敬之为本书的蒙皮骨骼动画部分制作了配套精美的 3DMAX 讲解;感谢另一位国内第一代 3D 游戏开发者宁可植先生,他提供了大量的 3D 游戏研发资料,包括收藏数年珍贵的 *Game Developer Magazine* 等。

同时感谢我的硕士研究生导师陈有青副教授,陈老师所主持的中山大学游戏开发平台项目一直为我提供了非常好的学习和研究氛围,让我获益颇深。

如果您在学习过程中有什么问题,欢迎和我们联系、交流,并提出好的建议和意见(电子邮箱:book_better@sina.com)。同时欢迎业内优秀作者联系我们,本书责任编辑的电子信箱为 Liuhao@ptpress.com.cn。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

本书在编写过程中,参考了国内外许多优秀的 3D 游戏开发书籍和资料,特别是《3D 游戏引擎设计》、《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。同时,本书还参考了《3D 游戏引擎设计》、《3D 游戏引擎设计》等书籍,为本书提供了大量的参考和借鉴。

第 1 章 认识 DirectX/DirectX SDK	1
1.1 DirectX/Direct3D 的历史与最新发展	1
1.2 DirectX SDK 组成介绍	2
1.3 理解 HAL 与 COM	2
1.4 DirectX SDK 安装与开发环境配置	3
1.4.1 DirectX SDK 的安装	3
1.4.2 Visual Studio.net 2003 环境配置	6
1.4.3 编译运行你的第一个 DirectX 程序	7
1.5 小结	12
第 2 章 理解 Direct3D 世界的基本元素	13
2.1 世界坐标系	13
2.1.1 三维立体空间坐标系中的点	13
2.1.2 三维立体空间坐标系中的直线和平面	14
2.2 顶点的表示	15
2.3 面与多边形	16
2.4 物体的朝向	17
2.5 纹理映射初步	17
2.6 小结	18
第 3 章 Direct3D 数学基础	20
3.1 向量运算	20
3.1.1 Direct3D 中的向量定义	20
3.1.2 Direct3D 中的向量运算	21
3.2 矩阵运算与操作	26
3.2.1 Direct3D 中的矩阵定义	27
3.2.2 Direct3D 中的矩阵运算	28
3.2.3 单位化矩阵 (Identity Matrix)	29
3.2.4 平移矩阵 (Translation Matrix)	29
3.2.5 旋转矩阵 (Rotation Matrix)	31

3.2.6 放缩矩阵 (Scaling Matrix)	32
3.2.7 混合变换 (Multiply Matrix)	32
3.3 矩阵变换综合使用实例	33
3.4 其他常用数学函数与变换介绍	34
3.5 小结	36
第4章 理解 Direct3D 渲染管线	37
4.1 顶点和像素处理	37
4.2 世界矩阵	38
4.3 摄像机矩阵	39
4.4 投影矩阵	41
4.4.1 摄像机观察平截头体	41
4.4.2 投影矩阵	42
4.5 剪切和视口放缩	44
4.6 小结	45
第5章 Direct3D 程序结构	46
5.1 优化的 Win32 程序结构	46
5.1.1 建立新的 Win32 程序项目	46
5.1.2 主流程函数解析及其优化	47
5.2 Direct3D 设备基础知识	51
5.2.1 获取显示设备	51
5.2.2 配置 D3DPRESENT_PARAMETERS 结构体	52
5.2.3 创建 IDirect3DDevice 设备对象	54
5.3 初始化 Direct3D 设备流程	55
5.4 可变顶点结构	58
5.5 顶点缓冲区与索引缓冲区	59
5.5.1 创建顶点缓冲区	60
5.5.2 创建索引缓冲区	63
5.6 理解三角形图元渲染方式	65
5.7 渲染一个基本三维图形	68
5.7.1 准备工作	68
5.7.2 渲染	69
5.8 Common 框架	77
5.8.1 Common 结构	77
5.8.2 CD3DApplication 类	78
5.9 使用 Common 框架	81
5.9.1 茶壶示例程序	81
5.9.2 移植彩色立方体程序	87

5.10 小结	92
第 6 章 使用网格模型构建复杂场景与角色	94
6.1 X 模型存储格式	94
6.1.1 MeshViewer 工具	95
6.1.2 X 文件的组织形式	98
6.2 Mesh 基础知识与接口介绍	105
6.3 加载和渲染 Mesh 模型	110
6.3.1 加载老虎实例 (不用 Common 结构)	111
6.3.2 加载宫殿实例 (使用 Common 结构)	116
6.3.3 CD3DMesh 类分析引导	118
6.4 Optimizing Mesh 介绍与示例	123
6.5 Progressive Mesh 介绍与示例	134
6.5.1 Progressive Mesh 介绍	134
6.5.2 应用举例	136
6.6 小结	141
第 7 章 在 Direct3D 中实现光照	142
7.1 Direct3D 中的光照	142
7.1.1 开启和关闭光照	143
7.1.2 光照模式	143
7.1.3 材质与光照	145
7.2 简单光照实例	145
7.3 小结	146
第 8 章 显示字体	147
8.1 ID3DXFont 字体类	147
8.2 简化的 CFont 字体类	149
8.3 小结	150
第 9 章 综合演练实例	151
9.1 三角形选取 (Pick) 实例演练	151
9.1.1 实现实例的理论基础	151
9.1.2 实例分析	153
9.2 封装摄像机类实现场景漫游	164
9.3 封装对象模型实现任意控制	169
9.4 用封装的类完成实例	174
9.5 小结	178

第 10 章 Direct3D 动画基础	179
10.1 蒙皮骨骼动画原理	179
10.2 使用 3ds max 制作蒙皮骨骼动画	188
10.3 蒙皮骨骼动画在 X 文件中的存储和加载	202
10.4 蒙皮骨骼动画的渲染	216
10.5 蒙皮骨骼动画范例分析	225
10.6 Morphing 动画原理与实践	228
10.7 小结	233
第 11 章 Direct3D 特效技术	234
11.1 公告牌技术	234
11.1.1 公告牌技术原理	234
11.1.2 公告牌技术实例分析	237
11.2 粒子系统特效原理与实践	247
11.2.1 粒子系统原理分析	247
11.2.2 粒子系统实例分析	248
11.3 Lighting Map 技术	256
11.3.1 Vertex Lighting (顶点光照模式)	257
11.3.2 Real Time Per-Pixel Lighting (实时每像素光照模式)	258
11.3.3 Per-Pixel Render With Lighting Map (光度图每像素渲染)	259
11.4 精灵特效实例	261
11.5 凹凸映射技术 (Bump Map)	268
11.5.1 Bump Map 原理介绍	268
11.5.2 Dot3 Bump Map 实例分析	273
11.6 环境映射 (Environment Map)	280
11.6.1 环境映射原理	280
11.6.2 立方体环境映射实例	281
11.7 阴影图技术 (Shadow Map)	287
11.8 体积阴影技术 (Shadow Volume)	290
11.9 小结	295
第 12 章 地形渲染技术	296
12.1 地形渲染概述	296
12.1.1 室内场景型	296
12.1.2 大规模室外场景型	297
12.2 室内渲染核心 BSP 技术初步	299
12.3 基于 Brute Force 方法的 Height Map 技术示例	310
12.3.1 初始化	310
12.3.2 渲染	312

12.3.3 创建纹理	316
12.3.4 添加光照	319
12.4 Fault Formation & Midpoint Displacement 高度表生成	322
12.5 Texture Blending 技术	326
12.6 LightingMap & SlopeLighting 光照技术	330
12.7 CLOD 技术	331
12.8 小结	331
第 13 章 High Level Shader Language 基础	333
13.1 Shader 的含义	333
13.1.1 可编程渲染管线	333
13.1.2 Vertex Shader	334
13.1.3 Pixel Shader	334
13.2 Shader 的使用	334
13.3 HLSL 规范	339
13.3.1 数据类型	339
13.3.2 表达式	354
13.3.3 语句	359
13.3.4 函数说明	363
13.4 Vertex Shader&Pixel Shader 示例	371
13.5 小结	374
第 14 章 大型游戏实例 Donuts4 分析与介绍	375
14.1 Donuts4 游戏规则介绍	375
14.2 Donuts4 程序框架分析	377
14.2.1 程序入口	377
14.2.2 创建窗口与初始化	379
14.2.3 运行过程	382
14.3 Donuts4 模型系统分析	390
14.4 Donuts4 地形系统分析	392
14.5 Donuts4 碰撞检测系统分析	405
14.6 小结	411



认识 DirectX/DirectX SDK

DirectX 事实上已经是游戏工业标准和显卡衡量标准。当我们学习游戏开发时，第一步自然是从了解 DirectX 及其 SDK 开始。本章中我们将会全面了解 SDK 的各个方面，包括组成、历史、代表作游戏等。接下来我们会对 DirectX 的 HAL 架构和基石 COM 技术做初步阐释。最后会学习如何配置开发环境以及编译自己的第一个 DirectX 程序。

1.1 DirectX/Direct3D 的历史与最新发展

在 Windows 出现以前，DOS 是 PC 上最流行的操作系统，程序员们开发的游戏都是在 DOS 下运行。最初的 Windows 操作系统在图像和音频上的性能让人难以忍受，所以很少有人愿意在 Windows 下开发游戏。

DOS 提供给开发者直接访问底层硬件的能力，但同时也带来了相应的复杂性。Windows 95 诞生以前，3D 加速卡市场一片混乱，各大厂商都有各自的 3D 技术标准，而且互不兼容。那时候的游戏制作商也很头痛，为了顾全大局，他们要顾及不同显示芯片厂商的标准，为了达到更好的效果，一个游戏往往要做出支持不同显示卡的主程序。例如生化危机、Incoming、Fortest、恐龙猎手刚出来的时候，都有面向 ATI，S3，Voodoo 等显卡的主程序。就在这个时候，Windows 95 出现了，并随着 Windows 95 的问世，DirectX 1.0 也随之诞生。

DirectX 的建立为计算机多媒体领域做出了不可磨灭的贡献。DirectX 标准建立以后，显示卡芯片厂商，声卡芯片厂商，键盘、鼠标、摇杆制造商，网卡制造商等不得不按照这一标准进行改进。

这样做对谁来说都是有意义的，微软公司开发了 DirectX 标准平台，并且与硬件制造厂商和游戏厂商合作共同更新升级 DirectX 的标准。硬件制造商按照此标准研发制造更好的产品，游戏程序员根据这套标准开发游戏。也就是说，无论硬件是否支持某特殊效果，

只要 DirectX 标准中有, 写游戏的程序员就可以把它写到游戏中, 当这个游戏在硬件上运行时, 如果此硬件根据 DirectX 标准把这个效果做到了此硬件驱动程序中, 驱动程序驾驭其硬件算出此效果, 用户就可以欣赏到此效果。这就是“硬件设备无关性”, 是 DirectX 真正意义所在。

Direct3D 最初是出现在 DirectX 3.0 中, 当时的 Direct3D 只支持一些简单的 3D 效果; 而到了 DirectX 5.0 的时候, Direct3D 已经可以支持雾化效果和 Alpha 混合效果了, 那时的 3D 游戏中, 已经可以注意到人物的构型、水面效果、场景效果等; DirectX 6.0 是 Direct3D 走向成熟的一个时期, 这时的 Direct3D 提供了更多、更实用的支持; 在 DirectX 7.0 中, Direct3D 开始支持 T&L 技术——即硬件坐标转换和灯光技术, 此外, 对纹理的支持也更加强大了。

DirectX 8.0 对 DirectX 来说是一个很大的跨度。3D 图形处理技术逐渐统一在 Vertex Shader 和 Pixel Shader 中, Vertex Shader 被用来描述和修饰 3D 物体的几何形状, 同时也用来控制光亮和阴影; Pixel Shader 则用来操纵物体表面的色彩和外观。Direct3D 和 DirectDraw 合二为一, DirectX 家族诞生了一个新的成员——DirectGraphics。

DirectX 9.0 抛弃了 T&L 技术, 而顶点渲染器和像素渲染器得到升级。顶点渲染器引入了流程控制, 增加了条件跳转、循环和子程序。现在的顶点渲染器控制程序最多可以由 1 024 条指令组成 (DirectX 9.0 之前最多 128 条), 这大大加强了可编程性。另外, DirectX 9.0 还支持 64 位浮点色彩精度, 极大地增加了图像质量, 让很多过去不可能的特效变成了现实。

1.2 DirectX SDK 组成介绍

DirectX 并不是一个单纯的图形 API, 它是由微软公司开发的用途广泛的 API, 它包含有 Direct Graphics (Direct 3D+Direct Draw)、Direct Input、Direct Play、Direct Sound、Direct Show、Direct Setup、Direct Media Objects 等多个组件, 它提供了一整套的多媒体接口方案。

- **Direct3D**: 3D 图形包, 在显卡的支持下提供各种丰富、强大的图形渲染功能。
- **DirectDraw**: 2D 图形包, 提供硬件加速的 2D 渲染显示功能。
- **DirectInput**: 为包括游戏杆、鼠标、键盘和游戏控制器在内的输入设备提供支持。它还还为反馈游戏设备提供支持。
- **DirectPlay**: 对编写多人联网游戏提供支持, 包括在处理玩家连接, 消息发送, 语音交流框架等方面提供了强大功能。
- **DirectSound**: 音效方面的支持, 提供低延迟的声音和对硬件资源的高度控制。
- **DirectShow**: 提供媒体流架构支持, 可播放高质量的视频和音频。
- **DirectSetup**: 对 DirectX 应用程序提供安装方面的支持。

每次 DirectX 的重大更新几乎都集中在 3D 图形技术方面。Direct3D 已经成为名副其实的显卡革命推动者, 对其支持程度也成了衡量显卡能力的重要工业标准。

1.3 理解 HAL 与 COM

HAL 是 Hardware Abstraction Layer 的简称, 即硬件抽象层, 其位置如图 1.1 所示。

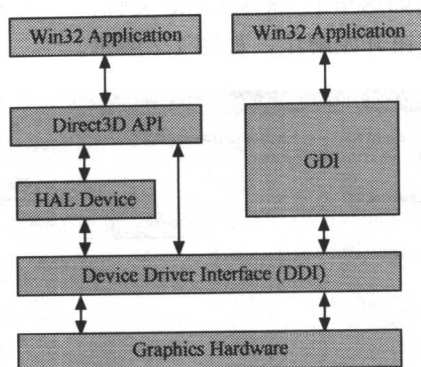


图 1.1

可以看到，最底层的 Graphics Hardware 是图形显示卡硬件，其上的 Device Driver Interface 是显卡驱动程序，由显卡制造商提供。然后在驱动程序上面架构了两套 API，一套是 Windows GDI，另一套就是 HAL 硬件抽象层上的 Direct3D API。HAL 通过显卡所支持的模式而对 Direct3D API 提供加速，因此可以提供更好的性能。编程者可以通过 Direct3D 提供的函数在运行时获取显卡对某项能力的支持程度。值得注意的是，当显卡不支持某项特性时，这一工作便由 HEL（Hardware Emulation Layer，硬件模仿层）来完成。这一层应该是和 HAL 并列于 Direct3D API 下方的，它的工作是使用软件模拟的方法来实现本来应该由显卡芯片完成的工作，当然在速度上会慢很多，会严重影响游戏的 FPS。不过在 DirectX 8.0 发布以后，就不再提供 HEL 层了，只能在测试模式的时候用 REF 软件模拟所有 DirectX 的特性，不可以发布到终端用户。

DirectX 是按照微软公司的 COM（Component Object Model）搭起来的。设计 COM 是希望它能提供一个更安全、易升级、可移植的软件模块。COM 用的面向对象的模式比一般的 C++ 更严格。例如，COM 只能永远通过成员函数（member function）进行访问，并且不能拥有公用数据成员（public data members）。但是 COM 是一种基于二进制的编程模型，它的接口无法通过任何方式来改变，并且提供向前兼容功能，即在低版本的接口基础上建立的程序一定可以在高版本环境中运行。换句话说，只要安装了最新的 DirectX 版本，那么建立在老版本基础上的 DirectX 程序也一定可以运行。它本身的二进制编程模型也使跨语言成为其一大优势。关于 COM 的知识可以专门写一本书来阐述，但是在 DirectX 中，我们无须了解 COM 复杂的理论就可以轻松使用 COM 提供的强大功能。

1.4 DirectX SDK 安装与开发环境配置

在使用 DirectX SDK 之前，必须安装并对它进行配置。

1.4.1 DirectX SDK 的安装

首先需要下载完整的 DirectX 9.0 SDK，读者可以在微软公司的官方网站上找到它（由于不同 SDK 版本会有些不同，建议读者在学习本书时使用 SDK 标准版）。这是一个自解压压缩包，双击就可以启动。

(1) 双击压缩包文件, 例如这里的 dx9sdk.exe 文件, 单击 Browse 按钮选择一个合适的保存目录, 然后单击 Unzip 按钮开始解压, 如图 1.2 所示。

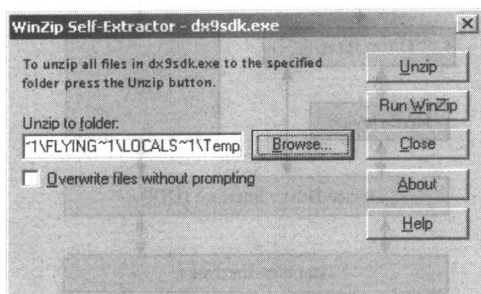


图 1.2

(2) 解压需要几分钟的时间, 请耐心等待, 如图 1.3 所示。DirectX SDK 包含有丰富的资源库以及实例程序, 所以这几分钟是相当值得的。当出现图 1.4 所示的画面时, 解压就结束了。

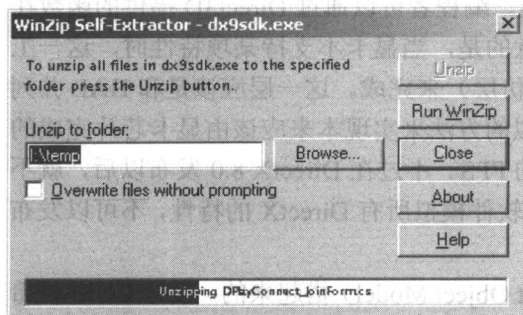


图 1.3

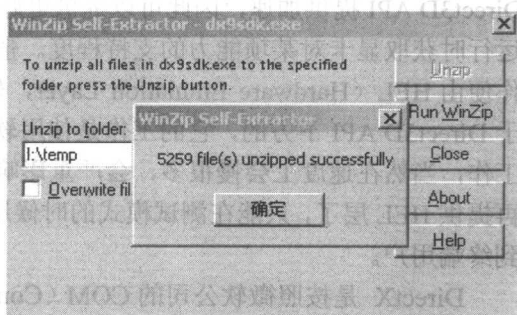


图 1.4

(3) 在刚刚选择的目录中, 可以找到解压后的文件, 双击 Install.exe, 就会出现如图 1.5 所示的安装欢迎界面。

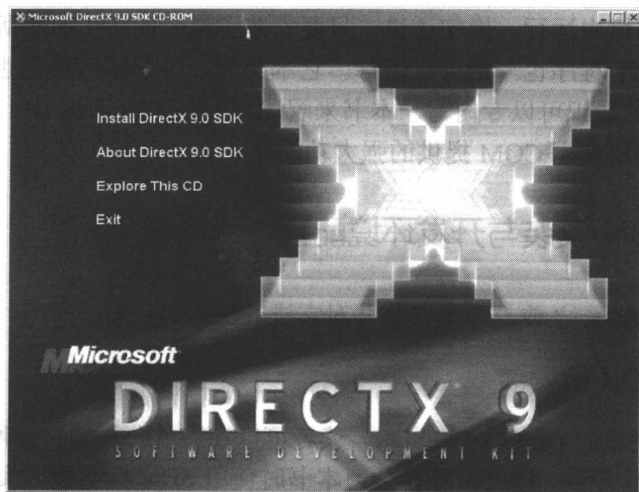


图 1.5

(4) 单击 Install DirectX 9.0 SDK, 进入安装向导, 如图 1.6 所示。接下来的事情就简单了, 基本上就是一直单击 Next 按钮。所以下面的步骤不再每步都介绍, 只讲解在安装过程中需要注意的地方。

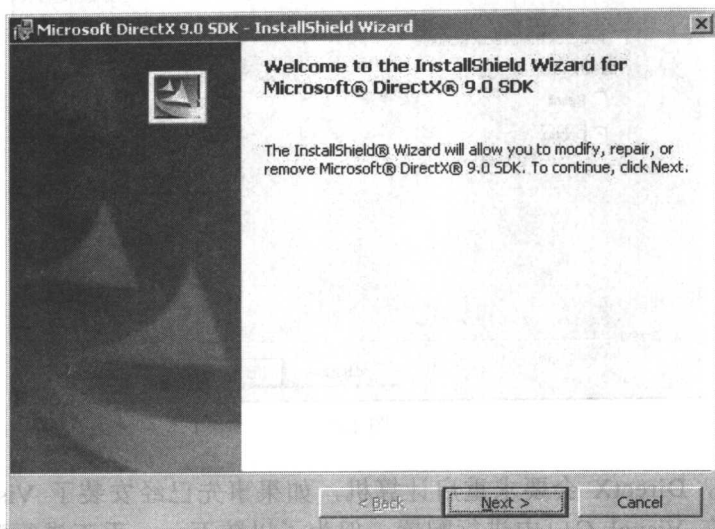


图 1.6

(5) 为了避免以后用的时候还要再安装的麻烦, 一般来说都选择安装全部组件, 如图 1.7 所示。

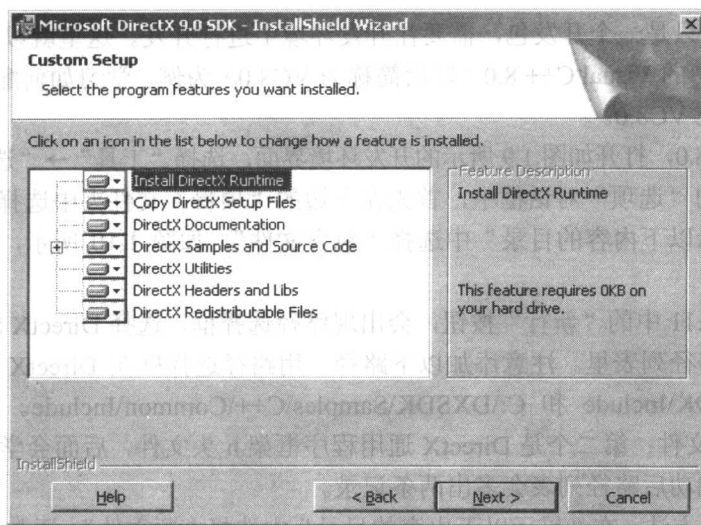


图 1.7

(6) 当出现图 1.8 所示的安装运行库画面时, 需要注意: 如果选择 Debug 模式, 那么运行速度会慢一些, 因为插入了很多调试用的功能, 主要提供给开发人员使用; 如果选择 Retail 模式, 运行速度则会快很多, 它主要提供给运行 DirectX 程序的普通用户使用。这里选择 Debug。