

苹果电脑开发技术系列

Apple Development Technology Series

Cocoa

QuickTime

Xcode

Open Source

AppleScript

Carbon

UNIX

苹果电脑开发培训教材

Apple Development Technology Fundamentals

苹果电脑开发技术 基础教程

同济大学软件学院 编著
同济大学苹果开发技术实验室



机械工业出版社
China Machine Press

苹果电脑开发技术系列

Apple Development Technology Series

苹果电脑开发培训教材

Apple Development Technology Fundamentals

苹果电脑开发技术 基础教程



机械工业出版社
China Machine Press

本书是Mac OS X应用程序开发的入门级教程,介绍了Mac OS X开发技术的各个重要主题。主要包括:苹果机操作系统的发展历程,特别介绍了即将发布的Mac OS X 10.4操作系统的众多创新特性;Mac OS X的体系架构、集成开发环境Xcode、国际化和本地化技术、QuickTime技术、Java应用程序的开发、UNIX的功能、Cocoa开发框架、AppleScript、Carbon开发框架、I/O Kit开发的基本思想等。

无论读者是PC平台的编程高手,还是刚刚准备开始学习编程的新人,本书对于掌握苹果平台开发方法都会提供有效的帮助。

版权所有,侵权必究。

本书法律顾问 北京市展达律师事务所

图书在版编目(CIP)数据

苹果电脑开发技术基础教程 / 同济大学苹果开发技术实验室, 苹果电脑(中国)公司编著. -北京: 机械工业出版社, 2004.11

(苹果电脑开发技术系列)

ISBN 7-111-15361-8

I. 苹… II. ①同… ②苹… III. 软件开发-教材 IV. TP311.52

中国版本图书馆CIP数据核字(2004)第100863号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

策划编辑: 温莉芳

责任编辑: 华 章

北京中兴印刷有限公司印刷·新华书店北京发行所发行

2004年11月第1版第1次印刷

787mm×1092mm 1/16·24.5印张

印数: 0 001-3 000册

定价: 49.00元

凡购本书,如有倒页、脱页、缺页,由本社发行部调换

本社购书热线:(010) 68326294

序

这本凝结着我院师生和苹果公司工程师们辛勤汗水的书籍，今天与大家见面了。

苹果公司一直与我院有着良好的合作关系。我们联合成立了全国第一所专门面向苹果平台开发技术的专业实验室。这个实验室成为同济大学软件学院师生进行苹果相关技术与教学的平台。在这个平台上，双方技术人员凭借精诚合作、大胆创新的精神，使一系列共享软件项目得以孵化出壳。

然而，随着研发的深入，尽管我们取得了一个又一个令人满意的成果，但是我们也感到一个突出而现实的矛盾，即需要一本涵盖苹果平台各方面开发技术的、但又简明易用的教程。非常庆幸的是，学院一批富于职业精神和工作热情的年轻教师和学生，萌发了编写这样一本教程的想法并积极付诸实践。而且这项工作得到了苹果公司工程师们的热诚帮助。经过项目组同志们日夜奋战，她——《苹果电脑开发技术基础教程》终于揭开了自己的面纱，呈现在您的面前。既然是一件创新性的工作，就不可避免地有一些有待提高和改进的地方，期待您的指正。

非常欣慰见到这样富有创新的成果，非常欣慰看到这样卓有成效的合作，更期待着这一系列更多后续教材的问世！

中国科学院院士
同济大学软件学院院长

周兴铭

前 言

随着苹果电脑在中国市场上占有率的逐渐提高，越来越多的用户开始接触这一新奇的充满创新的平台。而基于苹果平台的开发人员也随着苹果机中国用户数量的增长而越发显得数量不足。在图书市场里上万种开发书籍中，基于Mac OS X平台的中文开发书籍却处于几乎没有的尴尬境地。为此，在苹果电脑（中国）公司的大力协助下，我们特意组织编纂了这本苹果平台开发技术的基础教程。

几乎每一个使用过Mac OS X操作系统的人，都会为其出色的用户界面而赞叹不已！作为一个基于UNIX内核的开放源代码操作系统，Mac OS X将健壮性、易用性与视觉美感完美地结合在一起，以至于它的许多爱好者们都想在其他任何场合重现出Mac OS X用户界面那晶莹的Aqua风格！

然而，这一切是如何实现的？Mac OS X支持用哪些语言开发应用程序？如何在Mac OS X上开发出符合Aqua风格的应用程序？如何以最简便的方式，让Mac OS X应用程序具备多语种和多语言的版本？如何为特殊的设备编写驱动程序，使之能应用于Mac OS X？……对于这些问题，本书都将给予您一个初步的解答。

如果您从未涉足过Mac OS X的开发，但却希望为其开发独特的共享软件，增加Mac OS X的诱人之处；或者您已经尝试过一些Mac OS X的开发，但希望了解一些其他的开发框架和开发主题，那么，本书都适合您！

本书凝聚了集体的智慧和努力。本书初稿的第1、2章由陈犇负责完成；第3、4章由桂一枫负责完成；第5章由何正熙负责完成；第6、10章由胡瑞端负责完成；第7章由张鼎辉负责完成；第8章由梅路遥负责完成；第9章由姜宏慧负责完成。最后由周闻钧完成全部十章的改写和修订。在本书的编写过程中，得到了同济大学软件学院和同济大学苹果开发技术实验室的全力支持。感谢苹果（中国）公司开发商技术支持及相关部门的工程师们，夏鹏、宋振轩、陈梓桥和端木恒等为本书的编写提供了各种技术资料并给予了許多中肯的建议。对于那些限于篇幅在此无法一一列名，但同样为本书作出了巨大贡献的人们，在此致以诚挚的谢意！缺少其中任何一个人的努力，本书都将无法完成！

当然，限于编者的水平，本书难免还存在一些错误和疏漏，恳请读者不吝批评和指正。

让我们一起成为Mac OS X的开发者吧！

编 者

2004年8月

目 录

序

前言

第1章 Mac OS系统介绍1

1.1 苹果系统简史1

1.1.1 第一个苹果核1

1.1.2 苹果图形时代的来临2

1.1.3 寻求发展3

1.1.4 迈向Mac OS X5

1.2 走进Mac OS X6

1.2.1 黑豹6

1.2.2 水元素8

1.2.3 文件管理10

1.2.4 多用户与安全性13

1.2.5 常用应用软件15

1.2.6 Apple字体解决方案16

1.2.7 网络17

1.2.8 使用终端18

1.2.9 使用帮助18

1.3 系统技术19

1.3.1 与Windows互动19

1.3.2 AppleScript20

1.3.3 关于网络的补充21

1.3.4 高级硬件特性22

1.4 Tiger24

1.4.1 Spotlight25

1.4.2 iChat AV29

1.4.3 Safari RSS30

1.4.4 Dashboard31

1.4.5 Automator32

1.4.6 VoiceOver33

1.4.7 .Mac Sync35

1.4.8 更强大的UNIX支持36

1.4.9 Xcode 2.036

1.4.10 64位技术38

1.4.11 核心图像38

1.4.12 H.264/AVC39

第2章 Mac OS X系统架构41

2.1 Mac OS X体系结构总揽41

2.1.1 分层透视图42

2.2 应用程序环境45

2.2.1 Classic45

2.2.2 BSD45

2.2.3 Carbon46

2.2.4 Cocoa47

2.2.5 Java48

2.3 应用服务50

2.3.1 图形和视窗环境50

2.3.2 打印系统55

2.3.3 其他应用服务58

2.4 核心服务59

2.4.1 核心基础59

2.4.2 Carbon核心60

2.4.3 CFNetwork61

2.4.4 Web 服务61

2.4.5 Open Transport61

2.5 追踪用户事件62

第3章 Xcode65

3.1 介绍65

3.2 让我们起步66

3.2.1 关于Xcode66

3.2.2 开发工具和系统架构67

3.3 苹果的集成开发环境和How2do67

3.3.1 Xcode应用程序的功能68

3.3.2 创建一个工程68

3.3.3 查找技术文档和信息86

3.3.4 使用Interface Builder为应用程序

添加用户界面86

3.3.5 使用修正和继续功能	96
3.4 开发过程	101
3.4.1 设计一个应用程序	101
3.4.2 设计一个用户界面	102
3.4.3 生成源代码	102
3.4.4 版本控制	104
3.4.5 调试和优化	105
3.4.6 打包	107
3.5 特殊工程类型的注意事项	108
3.5.1 脚本	108
3.5.2 创建框架	109
3.5.3 标准苹果插件工程	110
3.5.4 工具工程	110
3.5.5 Java	111
3.5.6 内核扩展	112
3.5.7 WebObjects	113
3.6 工具描述	113
3.6.1 创建源代码	113
3.6.2 版本控制	118
3.6.3 调试和优化	119
3.6.4 打包	123
3.6.5 脚本	124
3.6.6 Java	127
3.6.7 内核扩展	128
3.6.8 工具箱驱动程序	128
3.7 阅读资料指南与参考文档	129
3.7.1 开发人员指南	129
3.7.2 参考文档	135
第4章 本地化和国际化	137
4.1 相关术语概述	137
4.2 Mac OS X 国际化技术	137
4.2.1 主要的国际化技术	138
4.2.2 语言预置及束资源	140
4.2.3 指定语言和语言区	141
4.2.4 国际化和本地化工具	142
4.3 Mac OS X 本地化技术	143
4.3.1 本地化用户界面	143
4.3.2 本地化字符串	144
4.3.3 加入多语系支持	146

第5章 QuickTime	149
5.1 QuickTime技术及其应用	149
5.1.1 认识QuickTime	149
5.1.2 QuickTime的优点	149
5.1.3 QuickTime的基本架构和基本操作	150
5.1.4 QuickTime VR 虚拟现实	151
5.1.5 QuickTime Streaming流媒体	152
5.1.6 QuickTime应用实例	154
5.2 QuickTime程序设计	157
5.2.1 程序设计的资源支持	157
5.2.2 基于Windows的QuickTime程序设计	158
5.2.3 使用QuickTime组件	160
5.2.4 打开和播放影片	165
第6章 Mac OS X中的Java	183
6.1 什么是Java	183
6.1.1 了解Java	183
6.1.2 Macintosh上的Java	183
6.1.3 Mac OS X 提供理想的Java环境	184
6.1.4 Java的用户环境	184
6.1.5 获得Aqua用户界面	186
6.1.6 HFS+	187
6.1.7 使用Xcode编写Java程序	187
6.2 Mac OS X的部署选项	190
6.2.1 可双击启动的JAR文件	190
6.2.2 Mac OS X应用程序包	191
6.2.3 Java Web Start	197
6.2.4 Java插件	198
6.3 用户界面工具箱	198
6.3.1 Swing	199
6.3.2 Java 2D	201
6.4 核心应用程序接口	203
6.4.1 连网	203
6.4.2 预置	203
6.4.3 JNI	203
6.5 Java虚拟机	205
6.5.1 Java 1.4.1虚拟机的基本属性	205
6.5.2 Mac OS X的Java共享档案	205
6.6 Info.plist中Java字典的关键字	208
6.6.1 可用变量	209

6.6.2 关键字和值	209	8.1 Objective-C简介	249
6.7 运行时系统属性	211	8.1.1 Objective-C的历史	249
6.7.1 和本地应用程序环境的集成	211	8.1.2 Objective-C中的数据类型	249
6.7.2 渲染提示	211	8.1.3 Objective-C中的消息传递方式	250
6.7.3 全屏Java	212	8.2 Cocoa中Objective-C对象的声明与实现	251
6.7.4 窗口尺寸的调整	213	8.2.1 接口文件	252
6.8 Java虚拟机的选项	213	8.2.2 实现文件	253
6.8.1 一般选项	213	8.2.3 创建新对象	254
6.8.2 Mac OS X专用选项	214	8.3 Objective-C中的内存管理	255
6.8.3 堆的大小	215	8.3.1 内存管理的目的	255
6.8.4 垃圾收集: 内存的使用	215	8.3.2 Cocoa中Objective-C的内存管理机制	255
6.8.5 垃圾收集: 一般设定	216	8.3.3 Cocoa中内存管理的方法	255
6.8.6 编译	216	8.4 Cocoa框架简介	256
6.8.7 线程	216	8.4.1 Cocoa 环境	256
6.8.8 共享	217	8.4.2 Cocoa 在 Mac OS X 中的位置	257
6.9 Java 1.4.1应用程序接口:		8.4.3 Cocoa 应用程序的功能	259
苹果技术的扩展	217	8.5 Cocoa架构	260
6.9.1 Application类	217	8.5.1 基础框架	261
6.9.2 ApplicationAdapter类	219	8.5.2 应用程序工具箱	262
6.9.3 ApplicationBeanInfo类	221	8.5.3 其他 Cocoa API 框架	266
6.9.4 ApplicationEvent类	221	8.6 一个图形界面的例子——重量单位	
6.9.5 ApplicationListener接口	222	转换器	266
6.9.6 CocoaComponent抽象类	223	8.6.1 创建重量单位转换器工程	266
6.9.7 FileManager类	225	8.6.2 重量单位转换器的界面	268
第7章 UNIX 与开放源代码	229	8.6.3 定义并控制转换器的类	273
7.1 UNIX与Mac OS X	229	8.6.4 完成界面操作并实现程序	277
7.1.1 概述	229	8.7 Cocoa绘图模型	280
7.1.2 Mac OS X中的UNIX——Darwin	229	8.7.1 Cocoa绘图的数据类型	281
7.1.3 Darwin的体系结构	236	8.7.2 画布和画刷	282
7.1.4 Darwin的Mach	238	8.7.3 画笔	284
7.1.5 Darwin的I/O kit	240	8.7.4 NSBezierPath机制	288
7.1.6 Darwin的文件系统	241	8.8 Cocoa中的Apple事件处理	294
7.1.7 Darwin的网络架构	242	8.8.1 NSResponder类简介	295
7.1.8 Darwin的BSD	242	8.8.2 创建工程与界面	296
7.2 苹果公司旗下的其他开放源代码项目	245	8.8.3 代码实现	297
7.2.1 概述	245	8.8.4 生成与调试	301
7.2.2 开放源代码项目简介	245	8.9 全面接触Cocoa开发	302
第8章 Cocoa 编程	249	第9章 AppleScript	303
		9.1 AppleScript 介绍	303
		9.1.1 什么是AppleScript	303

9.1.2	AppleScript能为我们做些什么	303	9.5.2	使用按钮得到用户信息	355
9.1.3	AppleScript语言难学吗	304	9.5.3	通过用户键盘输入获得信息	356
9.1.4	AppleScript程序是什么样的呢	304	9.6	AppleScript程序样例分析	356
9.2	AppleScript的开发环境	305	9.6.1	程序样例1(最小化所有的程序窗口 和文件夹窗口)	356
9.3	AppleScript Studio介绍	308	9.6.2	程序样例2(为一个文件夹中所有 文件加前缀或者后缀)	357
9.3.1	AppleScript Studio是如何工作的	308	第10章	Carbon与驱动编程基础	363
9.3.2	Hello World小程序	311	10.1	Carbon	363
9.4	AppleScript语言	316	10.1.1	Carbon简介	363
9.4.1	AppleScript语言的特点	316	10.1.2	Carbon的组成	365
9.4.2	语句	317	10.1.3	Carbon开发环境	369
9.4.3	数据类型	318	10.1.4	相关链接	370
9.4.4	变量	320	10.2	设备驱动程序	370
9.4.5	AppleScript中的全局变量	324	10.2.1	什么是 I/O Kit	370
9.4.6	常量	326	10.2.2	I/O Kit 的功能和设计	370
9.4.7	操作符	327	10.2.3	I/O Kit的限制	371
9.4.8	控制语句	330	10.2.4	I/O Kit 的组成部分	372
9.4.9	Script 对象的范畴	332	10.2.5	I/O Kit体系结构概述	372
9.4.10	对象	341	10.2.6	I/O Registry	375
9.4.11	命令	346	10.2.7	基类	379
9.4.12	引用	347	10.2.8	更多I/O Kit的信息	380
9.4.13	handler	350	资料来源		381
9.5	在AppleScript中使用对话框	355			
9.5.1	给用户显示信息	355			

第 1 章 Mac OS 系统介绍

欢迎进入Mac OS X的世界!

Mac OS X是一个独一无二的操作系统，它成功地把范例、意识形态及技术三者融合在一起，这三者在过去总是相互抵制。Mac OS X是苹果公司多代操作系统试验和磨难的结果。Mac OS X可能是集公司、学院和研究团体、开放源代码和免费软件运动，甚至个人直接或间接的努力而成就的最成功的操作系统之一。

苹果公司成立于1976年，它的许多故事常被人们津津乐道。苹果公司的故事是迷人的，有关它的操作系统的故事则更加引人入胜。以下将介绍苹果公司创造过或尝试着创造过的操作系统。或许，在弄清了Mac OS X与这些操作系统的联系以及Mac OS X的一些重要组成部分后，会使您更加了解和激赏现今的系统。

1.1 苹果系统简史

1.1.1 第一个苹果核

苹果公司由Steve Jobs、Steve Wozniak两人以及Atari的工程师Ronald Wayne于1976年创建。公司的第一个产品是Apple I。Apple I基于一块8位的微处理器——采用金属氧化物半导体科技的6502。它以不到1MHz的频率高效地运行。随Apple I一起的还有一个系统监视器及一个可以认为是它的操作系统的程序。这个屏幕程序只有256字节，它利用键盘和屏幕为用户提供一个命令行的环境，用来察看内存的内容、输入和执行程序。

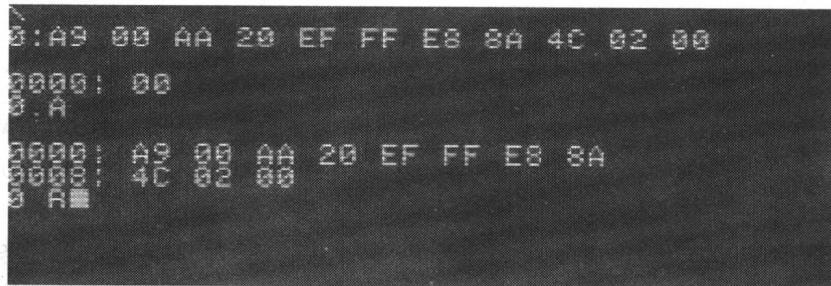


图1-1 第一个苹果核

比起当时流行的UNIX系统，Apple I的“操作系统”虽然不那么动人，但是却可以节省数以万计的美元。Apple I是一次将电脑推向大众的尝试。随着Apple II及后续机型的推出，在

Apple机基础上相继诞生了Apple Dos、Apple Pascal、Apple CP/M、Apple SOS、Apple ProDos等一系列基于命令行的操作系统。

1.1.2 苹果图形时代的来临

NLS、Smalltalk、Xerox Alto 以及Xerox STAR System等系统赋予了苹果公司新的灵感。鼠标、交互式协作、图形界面等概念被引入到苹果机中。苹果公司于1983年1月发布了昂贵的Lisa系统。它拥有完全的图形用户界面。Lisa系统引入了后来成为苹果电脑操作系统组成部分的许多概念：例如，在屏幕的最上方有一个菜单条（Menu Bar）；双击一个图标会动态地弹出一个窗口；Trash等。但Lisa系统高昂的价格，使得它在现实中没有成为像预期中那样完美的电脑。

在Lisa系统失败后，Steve Jobs组建了一支由资深工程师组成的队伍，旨在创造出能被所有人使用的电脑。这些努力的结果便是产生了Macintosh。1984年1月24日，Steve Jobs为Macintosh揭去了神秘的面纱。Macintosh上运行着一个单用户、单任务的操作系统：Mac System Software。

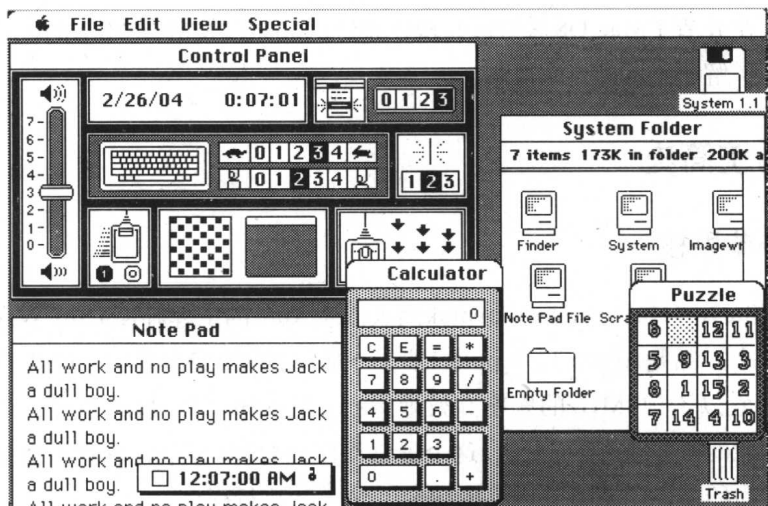


图1-2 System 1的界面

Macintosh的ROM中储存着底层代码（用于初始化硬件等）和高级一点的工具——Toolbox。Toolbox是一个实例程序的集合，它就像是一个应用程序可以使用的资源库。有了Toolbox，程序员就能开发出统一的标准的用户界面。Toolbox中的QuickDraw部分含有创建图形界面所需的经过优化的基本部件，比如对话框、下拉菜单、滚动条等。

随操作系统附带的应用软件是一个名为Finder的文件管理器。但系统的单任务属性却要求用户退出正在运行的应用程序才能使用Finder（反之亦然）。Macintosh的文件系统（MFS）是一个平铺的文件系统：所有的文件储存在同一目录下，但操作系统会提供一种分层的视图用于察看文件。Macintosh保留了Lisa系统的许多特性，比如Menu Bar和Trash等，体现了苹果的图形用户界面的指导方针。

1.1.3 寻求发展

在很长的一段时间内，不同的操作系统的组成有着许多不同的、相互独立的改进版本，比如：提高了Finder的运行速度；真正的分层的文件系统（HFS），这使得文件夹不再需要虚拟地创造出来；对大磁盘的支持等等。其中有一个重大的改进，那就是MultiFinder，它允许用户同时打开许多程序，同时给这些程序分配内存。在这一时期，系统版本从System 2发展到了System 6。

在两个有趣的操作系统GS/OS和A/UX之后，System 7于1991年诞生了。这对于早先的Macintosh系统来说是一个重大的飞跃。System 7整合了许多引导后来软件开发的特性：内置的MultiFinder，无缝多任务技术（虽然采用的只是协同式多工方式），支持32位的内存寻址，彩色图标，个人文件夹，虚拟内存，以及便于管理的分层系统目录等等。Mac的用户界面也得到了很大改进，就像下面这张图所呈现的那样：

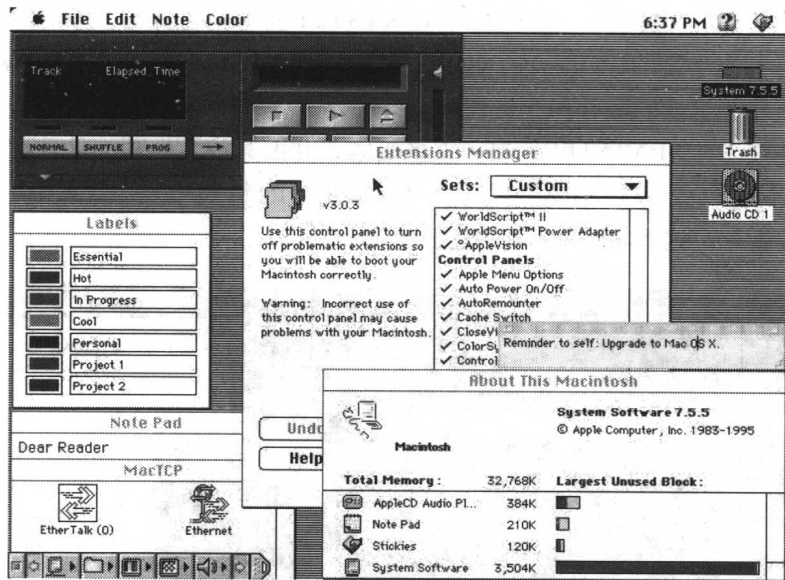


图1-3 System 7的界面

System 7也引进了一些新技术，比如ColorSync、PowerTalk、AppleScript和QuickTime等等（后两者将在第2章提及）。但尽管有很多超前的改进，System 7并未提供内存保护，同时由于其使用的协同式多工方式，使得系统运行效率并不高，并且非常容易瘫痪。

在这一时期，苹果公司与IBM以及摩托罗拉公司建立了联盟，将PowerPC引入到苹果机硬件的发展规划中。

为了符合高级编程环境，对计算机结构上的语义的强调会导致非常复杂的处理器。但是，简单的计算机结构的好处早在1960年就被认识到了。RISC（精简指令集计算机）并不仅仅意味着更少的指令，RISC结构有着以寄存器为中心、总是使用常用指令、具备高效的流水线操作、对时钟频率的需求更少等优点。IBM公司从1975年开始对RISC结构进行研究，1990年推出了IBM RS/6000芯片。IBM RS/6000基于第二代被叫做POWER的RISC结构。1991年，苹果、

IBM和摩托罗拉公司联合建立了AIM联盟，其目标是创造“通用硬件相关平台”（简称CHRP）。其结果便是PowerPC结构。PowerPC被设计为一个64-bit的结构，并且可以在64-bit和32-bit模式间切换。PowerPC在被采用前就已成为苹果公司的支柱硬件。

同时，为了回击来自微软公司的Windows 3.x的冲击，特别是为了面对即将到来的Windows 95，苹果公司也进行了一系列改进操作系统的尝试。

其中有一个被命名为Copland的计划，它被看成是苹果公司对Windows真正的回应。苹果公司希望它能实现这样一些目标：采用RISC作为关键的基础技术，从而实现系统对PowerPC完全的本地化；实现抢占式多工和内存保护；保留和改进Mac OS易用的用户界面；使其支持多用户和完全的用户订制；以及使Mac OS成为最好的网络终端等等。但Copland计划最终在1996年被取消了。

另一个则是由NeXT公司所开发的NEXTSTEP系统。NEXTSTEP系统最初在1987年随高雅的NeXT Cube被世人所认识。建立在BSD UNIX和Mach（关于BSD和Mach更详细的介绍请参见第2章的相关内容）“微内核”基础上的NEXTSTEP拥有抢占式多工和内存保护的核心。NEXTSTEP同样拥有丰富的GUI桌面。对于开发者，NEXTSTEP拥有“Frameworks”——一个由许多应用程序库组成的集合和允许程序员开发基于Objective-C语言应用程序的工具。开发者们喜爱这些技术。NEXTSTEP发展了许多版本，并因其稳定性和高效性被许多政府部门和公司采用（Mac OS X从NEXTSTEP中继承下来的特性有：Mach内核、BSD层和NEXTSTEP编程框架，这些将在第2章中提及）。

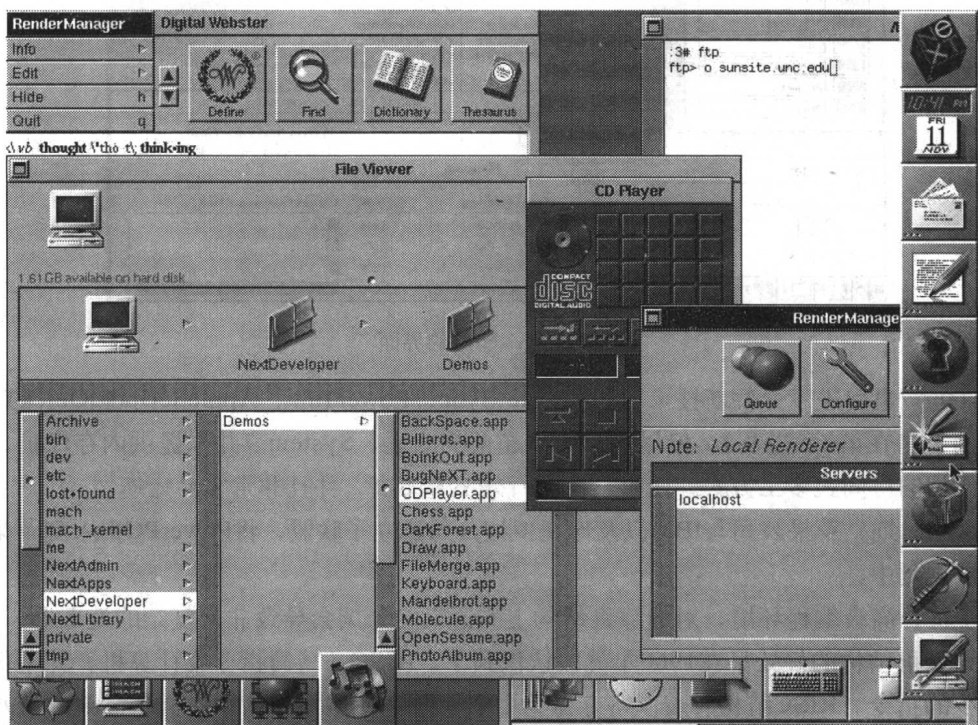


图1-4 NEXTSTEP的界面

作为一系列尝试的终结，1997年2月，苹果公司用4亿美元买下了NeXT公司。在这之后发布的第一个操作系统版本是Mac OS 7.6。而最初打算作为7.7版发售的版本后来成为了 Mac OS 8.0。

一些重要的本来为Copland计划创造或改进的特性被添加到Mac OS 8.x/9.x中，比如：扩展的Mac OS文件系统（HFS+）；一个可以搜索本地磁盘、网络服务器和Internet的搜索引擎；Copland API（逐渐演化成Carbon）；有着铂金外表的用户界面；多重用户；以及Mac OS 8的Finder允许不同文件导向的操作同时进行。Mac OS 8甚至提供了对USB和FireWire的支持。

Mac OS 9发布于1999年。这是第一个可以在线更新的Mac OS系统。它包含了安全特性比如文件密码以及用于储存密码的Keychain机制。Mac OS 9的一个重要组成部分是一个成熟的Carbon API集成。它在当时可以提供70%的以前的Mac OS API，和Mac OS 8.1及以后的API。

最后“老”版本的Mac OS是于2001年发布的Mac OS 9.2.2。

在这个时期，苹果公司同时推出了被称为狂想曲（Rhapsody）的操作系统。第一次演示Rhapsody是在1997年的苹果开发者大会（WWDC）上。它基于OPENSTEP（一个开放平台，包含许多API和Framework，任何人都可以使用它们创造出属于自己的面向对象的操作系统，并且可以基于任何更加核心的操作系统），并且由以下的重要部分构成：一个基于Mach 和BSD的核心和子系统；Yellow Box：扩展的OPENSTEP API工具；Blue Box：与Mac OS兼容的子系统；一个Java虚拟机；一个加入OPENSTEP特性的模拟Mac OS的用户界面。

Rhapsody将Classic和Java的技术引入到了现在的Mac OS X系统。为了让下一代操作系统可以运行在更多的硬件上，以及渴望能有一个兼容层使得老的Mac上的软件能更容易地移植到新系统上，苹果公司开发了Carbon层，并在1998年的WWDC大会上宣布了在Rhapsody之后的Mac OS X计划。

1.1.4 迈向Mac OS X

在大受欢迎的Public Beta版之后，代号“印度豹”的Mac OS X 10.0终于在2001年3月24日正式发售——距离Macintosh原型机的发明整整17年零3个月。人们热烈地欢迎它的到来。此时的Mac OS与Unix的融合已经完备，苹果公司终于拥有了渴望已久的下一代操作系统（关于Mac OS X的系统构架将在第2章详细介绍）。

以下是在撰写此书时已发布的Mac OS X的主要版本以及每一版的一些特性：（版本的代号全部取自于猫科动物）

10.0 Cheetah（印度豹）：2001年3月24日发售，包含了两个“Digital Lifestyle”（数字化生活方式）的应用程序：iMovie和iTunes。

10.1 Puma（非洲狮）：2001年9月29日发售，包含了iDVD。Carbon API工具已经相当完善以至于一些重要的第三方软件，如Adobe Photoshop等，已在Mac OS X平台上发布。

10.2 Jaguar（美洲虎）：2002年8月24日发售，它在许多领域中都有着重大进步。它引入了Quartz Extreme——一个完整的图形硬件加速层，它主要使用显卡上的GPU合成图形对象。从此以后，苹果公司开发了许多新的应用程序以及合成的技术，比如Address Book、iChat、iPhoto、Safari（网络浏览器）等等。

10.3 Panther (黑豹): 2003年10月24日发售, 它的许多新特性将在后一节中详细介绍。

在Mac OS X的早期版本发布以后, 系统的演化看起来就像苹果公司知道该添加哪些东西才能使新用户接受, 让老用户更加激赏。更加值得称赞的是, 苹果公司创造出了一个具有吸引力的并且能够满足拥有不同兴趣、能力和体验的人们各种需求的操作系统。

1.2 走进Mac OS X

1.2.1 黑豹

对于Panther (黑豹) 这只新猛兽, 苹果公司的任务已经不是要建立一个新的操作系统, Jaguar已经实现了这一点。Panther是在前面的系统的基础上, 将苹果操作系统带到了Windows需要在Longhorn推出才能实现的境界。

下面是从Panther上百的新特性中选择出来的, 可能是最特别, 最引人喜爱的十大特性:

1) Exposé: OS X无缝的多任务能力带来桌面的杂乱, 用户可能会同时运行成打的程序。在Panther中, 苹果操作系统利用Quartz Extreme超强的图形能力实现了新的窗口管理方法。只需要一个按键, 就可以清理掉杂乱的桌面。

2) Cmd-Tab: 当然, 以前的Mac OS X版本已经可以用Cmd-Tab切换程序。但现在, 当您使用Cmd-Tab的时候, 会在屏幕中间看到半透明的程序清单。

3) 分主题显示邮件: 就像多年前的Netscape Mail有着很好的分主题浏览功能一样, 现在, Mail的分主题浏览已经不比其他程序逊色。

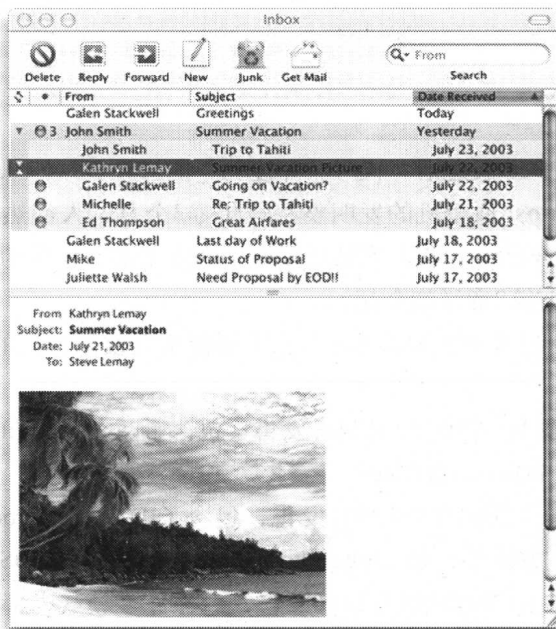


图1-5 Panther的邮件分类

4) 快速预览: 对于经常阅读和生成PDF文档的用户来说, 新的快速预览功能大大地方便

了他们的工作。现在，用户可以快速地搜索PDF文档，而不需要像以前那样经常需要打开Acrobat Reader。内置的阅读PostScript或EPS文件的能力使得浏览那些还没有转换为PDF的存档素材变得更方便。惟一遗漏的仍然是查看别人用Acrobat Reader所做的注解的功能。

5) 新的Finder: 把Finder的焦点从启动磁盘改到用户主目录是个令人欢迎的改变。现在不需要经一番点击才可以从一个新的Finder窗口的启动磁盘转换到自己的主目录。新的侧面工具栏使得访问经常使用的文件夹变得更加容易。更好的是，侧面工具栏中的项目会显示在所有应用程序的保存和打开对话框中。

6) Safari渲染: 对HTML来说，Safari并不仅仅是一个浏览器，它是一个可以由全部程序使用的HTML渲染组件。当然，WebKit已经推出了一段时间，但现在它已经遍布在整个系统中。这意味着HTML标记的显示在Safari或Mail中都是一样的。更好的是，在WebKit的支持下，以前缓慢的帮助程序获得了很大的速度提升。

7) 字体簿: OS X和Quartz显示字体的方式一直都是明确的，但一直没有一个系统工具来组织和使用这些字体。有了字体簿，一切都易如反掌。

8) FileVault: 通过FileVault，存在笔记本电脑上的资料可以在用户没有登录的时候被锁起来。另外，通过与家里的服务器定时同步资料的方法，即使用户丢失了笔记本电脑，则只是损失了笔记本电脑，而不会损失重要的资料。

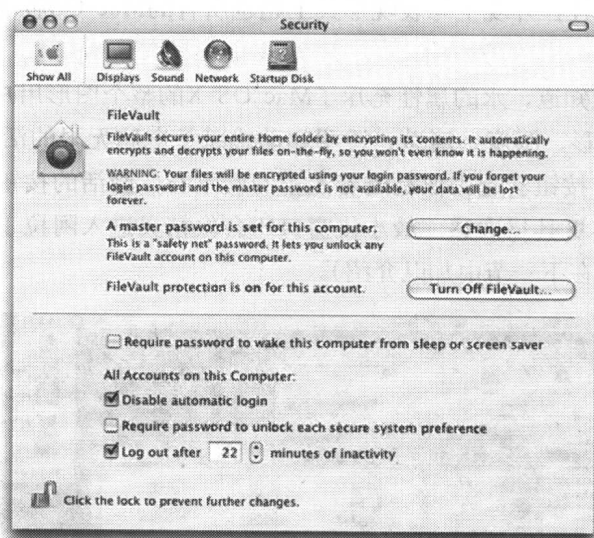


图1-6 FileVault

9) 安全清楚的垃圾筒: 当您对数据的安全有近乎偏执的要求的时候: 例如当删除文件的时候，您要确定它的确是删除了。Panther提供了并不是简单地倾倒垃圾桶的选择，而是在文件原来的地方写入随机数据，这样即使有黑客懂得查看硬盘的原始记录也无能为力了。

10) 活动目录集成: 当然，那些只是生活在Mac环境的人对这一点并不关心，但Panther可以使用活动目录作为密码认证系统并具有把主目录存储在远程Windows服务器的能力，使得OS X能够继续进入企业环境。

另外一个非常重要的特性:

11) Xcode: Panther继承了OS X免费附带开发环境的做法。苹果公司很聪明地意识到: 应用程序越多, 操作系统平台会越好。要鼓励更多的人开发软件, 就要提供好的开发环境和Cocoa这样强大的框架。新的Xcode集成开发环境彻底地改进了开发工具, 用户现在可以专注于编写代码, 而不需要过多地考虑程序文件的组织。它的改变是如此的巨大, 使得开发人员需要花些时间去适应。但苹果公司把它的用户界面专家投入到开发环境的设计中, 用户肯定会因此受益。另外, Xcode可以使用Rendezvous来利用网络中的所有机器来帮助编译代码, 这是个非常酷的特性。

相信这些特性以及其他的特性会使得Panther成为有史以来Mac OS X最重要的升级。

当然Mac OS X还有很多重要的新功能和改进, 您可以到苹果公司的网站: <http://www.apple.com.cn/macosx/> 上查询。

1.2.2 水元素

1.2.2.1 流动的Aqua

在苹果公司设计Aqua——Mac OS X的图形用户界面时, 设计的目标是: 创造一种不仅容易使用, 上手方便, 并且在外观上与表现形式上超越所有的Mac OS版本, 比您所见过的任何Mac OS 更具吸引力!

从Aqua这个词就能知道, 水的属性充斥了Mac OS X的整个图形用户界面, 这成为了一种风格。Aqua 以它的色彩、深度、清澈、透明和动感赋予电脑无限的活力, Aqua 力求完美, 不放过每一个小细节: 按钮看起来像是泛着光芒的蓝宝石; 激活的按钮更会呈现出脉动般的闪烁; 窗口的阴影使其更具层次感; 最小化窗口还会像精灵进入阿拉丁神灯一样落入它们的Dock 图标中 (Dock 将在下一节中加以介绍)。



图1-7 Aqua界面