数字化魅力·多媒体高手之路丛书

Windows Media 编程与应用

肖 磊 王志坚 编著

清华大学出版社
重庆大学出版社

数字化魅力·多媒体高手之路丛书

Windows Media 编程与应用

肖　磊　王志坚　编著

清华大学出版社
重庆大学出版社

内 容 简 介

本书以 Microsoft Windows Media 技术为基础，通过诸多实例，讲述了利用 Microsoft Windows Media SDK 进行流媒体开发的方方面面，由浅至深地介绍了如何使用 Microsoft Windows Media SDK 开发适用于用户的流媒体系统，内容涉及视频直播、点播、存储、转换等，让读者更快地掌握 Microsoft Windows Media SDK 的使用方法、编程技巧，定制、构建出用户自己的流媒体系统。

本书可供有一定编程基础，了解流媒体概念和希望深入了解 Microsoft Windows Media 技术的读者阅读，也适用于决策者和初次接触教学媒体编程的程序员参考。

图书在版编目（CIP）数据

Windows Media 编程与应用/肖磊主编. —重庆：重庆大学出版社，2003.9

（数字化魅力·多媒体高手之路丛书）

ISBN 7-5624-2973-1

Ⅰ.W… Ⅱ.肖… Ⅲ.多媒体技术 Ⅳ.TP37

中国版本图书馆 CIP 数据核字（2003）第 075414 号

数字化魅力·多媒体高手之路丛书

Windows Media 编程与应用

肖 磊 王志坚 编著

责任编辑：王海琼 曾 航 谭 华 版式设计：陈 其

责任校对：何建云 责任印制：秦 梅

*

清华大学出版社
重庆大学出版社 出版发行

出版人：张鸽盛

社址：重庆市沙坪坝正街 174 号重庆大学（A 区）内

邮编：400030

电话：（023）65102378 65105781

传真：（023）65103686 65105565

网址：http://www.cqup.com.cn

邮箱：fxk@cqup.com.cn（市场营销部）

全国新华书店经销

重庆科情印务有限公司印刷

*

开本：787×1092 1/16 印张：18.75 字数：421 千

2003 年 9 月第 1 版 2003 年 9 月第 1 次印刷

印数：1—5 000

ISBN 7-5624-2973-1/TP·428 定价：39.50 元（赠 1CD）

前言

随着互联网技术的发展，宽带网络越来越普及，流媒体技术作为宽带应用的主流技术，越来越受到人们的关注。宽带网络的竞争最终将是宽带应用的竞争，由此诞生的基于中速网络的流媒体（Streaming Media）技术正日益受到人们的关注。它推动了互联网整体架构的革新，转变了传统互联网呆板的内容表现形式，赋予宽带应用更多的娱乐性和互动性，必将成为未来宽带网络的主流技术。

在初期的宽带网络建设中，从重量级的电信企业到房地产商都投身到这场意图改变中国百姓生活的信息化建设中。人们逐渐认识到，在这场没有硝烟的战争中，宽带应用才是真正的主宰者，于是人们纷纷寻找适合宽带网络的应用支撑技术。我们知道，从网络上出现第一张图片，到现在各种形式的网络视频、动画，人们的视听觉感官在网络上得到了很大的满足。而同时面临的却是另外一种不可避免的尴尬：正是由于人们需求的不断提高及上网人数的不断增加，加之网络硬件设备的局限性，使得文件的大小成为网络传输中的瓶颈参数。一方面，人们希望能在网络上看到生动清晰的媒体演示；另一方面人们又不得不去面对现有网络速度下，文件传输所耗费的大量时间。

于是，基于中速网络的流媒体技术跃入人们的视野。流媒体的应用改变了传统互联网呆板的内容表现形式，具有强视觉冲击力的视频节目成了人们进入宽带网络的最重要的应用之一。在中国的宽带网络市场上，基于 MPEG-1，MPEG-2，REAL，WMV，QuickTime 各种流媒体技术的产品成了宽带网络的宠儿，这无疑刺激了宽带的发展，也提供了一个中国产品和国外产品公平竞争的舞台。

Microsoft 公司近年来也在此领域展开了激烈的竞争，它推出的 Windows Media 技术以其方便性、先进性、集成性、低费用等特点，逐渐被人们所认识和接受。Windows Media 的前身是 Microsoft 公司的 NetShow 产品，随着流媒体的广泛应用，之后又推出了整套的流媒体制作、发布和播放产品，其服务器端的 Windows Media Server 产品在 Windows NT Server Pack 4 上可以安装，并且集成在 Windows 2000 Server 中。Windows Media 产品的一大特点是制作、发布和播放软件与 Windows NT/2000/9X 集成在一起，不需要额外购买。Microsoft 公司的流视频解决方案在微软视窗平台上是免费的，制作端与播放器的视、音频质量都上佳，而且易于使用，但只能在微软视窗平台上使用。

本书旨在介绍 Microsoft Windows Media SDK 技术的综合应用，并使用相关的 SDK 开发包来定制需要的各种应用方案，是学习 Windows Media 开发的必备读物。

全书以当今世界第一大软件公司 Microsoft 公司的产品 Windows Media 技术为基础，由浅及深地讲述了如何使用 Windows Media SDK 技术构建流媒体系统，由理论至实例详细地介绍了 Microsoft Windows Media SDK 的组成及应用等内容。本书注重实际应用，使读者容易理解并使用。为此，在内容组织上进行了精心安排，以适用性最强的应用为实例，例如以视频点播系统、直播系统、数字版权保护等为实现目的，使读者容易理解并使用，达到学以致用的效果。

本书共分 7 章，循序渐进地讲解 Microsoft Windows Media SDK 开发技术。

第 1 章：Windows Media 概述。主要介绍 Windows Media 技术，讲解 Windows Media SDK 的组成与学习方法。

第 2 章入门篇：Windows Media Player SDK。介绍如何使用 Windows Media Player SDK 操作和控制 Windows Media Player。

第 3 章提高篇：用 Windows Media Encoder SDK 制作自己的媒体压缩器。以实例为基础，介绍如何定制自己的编码器。

第 4 章服务篇：定制自己的服务器 Windows Media Services SDK。介绍如何使用 Windows Media Services SDK 来操作 Windows Media Services，包括管理与监控。

第 5 章控制篇：Microsoft Windows Media Metafiles。讲述 Metafiles 的概念以及应用方法。

第 6 章商务篇：Windows Media Rights Manager SDK。介绍数字版权管理概念，以及如何使用 Microsoft 数字版权管理组件。

第 7 章实例篇：商务综合应用。综合前 6 章，以实例为基础，介绍一个商务技术解决方案。

本书面向有一定编程基础，了解流媒体概念的用户以及希望深入了解 Microsoft Windows Media 技术的读者，也适用于决策者以及初次接触数字媒体编程的程序员。

本书参考网站：

Microsoft MSDN Online　http://msdn.microsoft.com

微软中国网站　http://www.microsoft.com/china

Microsoft Windows Media SDK Help Document

流媒体论坛 SDK **开发版面** http://www.liumeiti.com/forum

本书的例子程序说明：

本书所有的例子都以文件的形式存放于随书光盘之中，在书中都有引用。

例程编写工具：

Visual Basic 6，VBscript，Active Server Script。

本书由重庆大学出版社与拓智文化共同策划，由流媒体中国网肖磊、王志坚编写。因时间和作者水平有限，书中难免出现错误，恳请广大读者提出宝贵意见。

欢迎读者与我们交流。E-mail：e@liameiti.com

编者

2003 **年** 8 **月**

目 录

第1章

Windows Media 概述

1.1 强大的 Microsoft Windows Media 技术

流媒体的应用是近几年来 Internet 发展的产物，广泛应用于远程教育、网络电台、视频点播、收费播放等，Microsoft 公司的 Windows Media 技术已经捆绑在 Windows 2000 中，对 Internet 的应用和发展产生了重要影响，Internet/Intranet 将不再是单纯的文本和图像，声音和视频将成为今后网络普及的重点。

流媒体是通过网络传输的音频、视频或多媒体文件，流媒体在播放前并不下载整个文件，流媒体的数据流随时传送随时播放，只是在开始时有一些延迟。当流媒体文件传输到用户计算机时，在播放之前该文件的部分内容已存入内存。

目前比较流行的流媒体技术是美国 RealNetwork 公司的 RealPlay 产品，许多 Internet 的音乐台、视频点播站点采用该产品。微软公司近年来推出的 Windows Media 技术异军突起，以其方便性、先进性、集成性、低费用等特点，逐渐被人们所认识和接受。

Windows Media 的前身是微软公司的 Netshow 产品，随着流媒体的广泛应用，推出了整套的流媒体制作、发布和播放产品，其服务器端的 Windows Media Server 产品在 Windows NT Server Pack 4 上可以安装，并且集成在即将正式推出的 Windows 2000 Server 中。Windows Media 产品的一大特点是其制作、发布和播放软件与 Windows NT/2000/9X 集成在一起，不需要额外购买，它势必成为今后流媒体应用的主流产品。

Windows Media 4.0 还提供了 MS Audio Codec 压缩技术，可以为普通拨号上网的用户提供调频收音的效果。如果把内容做成 WMA（Windows Media Audio）的格式，压缩效率比 MP3 提高一倍，用户可以下载之后在 WinCE 中播放。IE 5 以后的版本已经实现了 Internet 收音机，也是采用 MS Audio。

Microsoft Windows Media 技术是针对局域网、Internet 的流媒体内容的解决方案。Windows Media 平台支持针对于网站、Web 应用程序的流媒体音频、视频等连续数据流的开发。其特别之处在于客户端在开始下载这些内容的时候，就可以使用这些内容，而不必等待这些内容全部下载完毕。流媒体（Streaming Media）技术的技术含量很高，它减少了用户的等待时间，并且占用了较少的存储空间。流媒体技术也支持无限长的发布流媒体内容，例如在线直播。Windows Media SDK 就是针对这个平台的开发包。

1.1.1 Windows Media 的应用介绍

Windows Media 可以用于娱乐、培训和在线教育等方面。

1）带图片的广播（Illustrated Audio）

可以把 PowerPoint 讲座录制到 CD 或 Web 站点，用户通过 IE 浏览器，就可以看

到一张张 PPT(PowerPoint)文件在讲解员讲解的同时会自动翻转。工作方式是:只要提供讲解员的声音(磁带或电子格式均可)和 PPT 文件,Windows Media 提供了 Encoder(压缩)工具可以在声音中插入 Marker,然后在声音播放的过程中,这些 Marker 就会翻转 PPT 图片,使声音和 PPT 图片保持同步。

2)流视频播出(Streaming Video)

用摄像机或投影仪获得视频信号后,就可以通过 Web 站点进行基于 Internet 的现场直播;或者保存为 NSF 文件后,以供播放。它需要在一台较高配置的 PC 机上安装上普通视频采集卡和声卡,然后通过视频采集卡输入视频和通过声卡输入声音信号,就可以用实时 Encoder(压缩)工具来直播或录制成流媒体。

3)远程教学(Remote Seminar)

教学者事先在 Internet/Intranet 上发出通知,听众在讲座开始前访问某个 URL 地址,当讲座开始时,听众可以看到演讲者的图像和听到他的声音,同时也可以看到 PPT 图片教学。在演讲者翻转 PPT 的时候,用户端的 PPT 图片也自动翻转。整个讲座也可以记录下来,以后按需播放。这需要教学者事先用 Multicast(多播方式)把 PowerPoint 文件传给 IIS 服务器,当听众等待讲座开始的时候,PPT 图片下载到用户的浏览器 Cache 中。用摄像机录制演讲者的图像和声音,通过实时 Encoder 直播出来。直播中包含了 PPT 翻转的命令,可以使听众的浏览器同步地翻转 PPT 图片。

4)提供收费电视(Pay by View)

当用户需要看内容供应商直播或按需点播的内容时,如果没有许可证,则 Windows Media Player 会带他到内容供应商的网站申请许可(可能需要付费)。Windows Media 4.0 中提供的 Digital Rights Management 功能可以加密供应商的内容,确保收费后信息才会播发。目前网络上众多的在线收费电影院就属于这类应用。

流媒体技术的应用主要是指传输或者发布音频和视频内容,这不是一个单独的软件可以完成的,需要多个组成部分协同工作。为了详细地解释清楚如何使用 Windows Media,可以假设存在一个案例,看看使用哪些部分才能组成一个应用解决方案。

例如,要组建一个网络电台,需要以下几个部分:

①电台播放内容的获取。

②电台所播放内容的传输方式。

③用户获取电台内容的播放。

使用 Windows Media 技术来创建这个电台,需要以下几个关键组件(如图 1.1 所示):

Windows Media 工具负责采集编码;Windows Media 服务器通过网络发布采集编码后的内容;Windows Media Player(播放器)播放获得的内容。

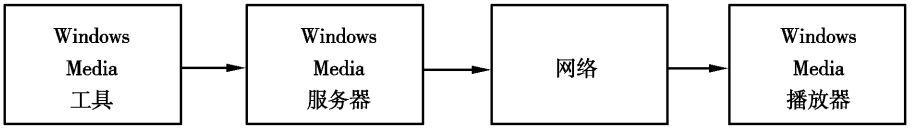


图 1.1　电台方案组成

因此，大多数基于 Windows Media 技术的流媒体系统工作流程如图 1.2 所示，其中的数字代表编码过程。

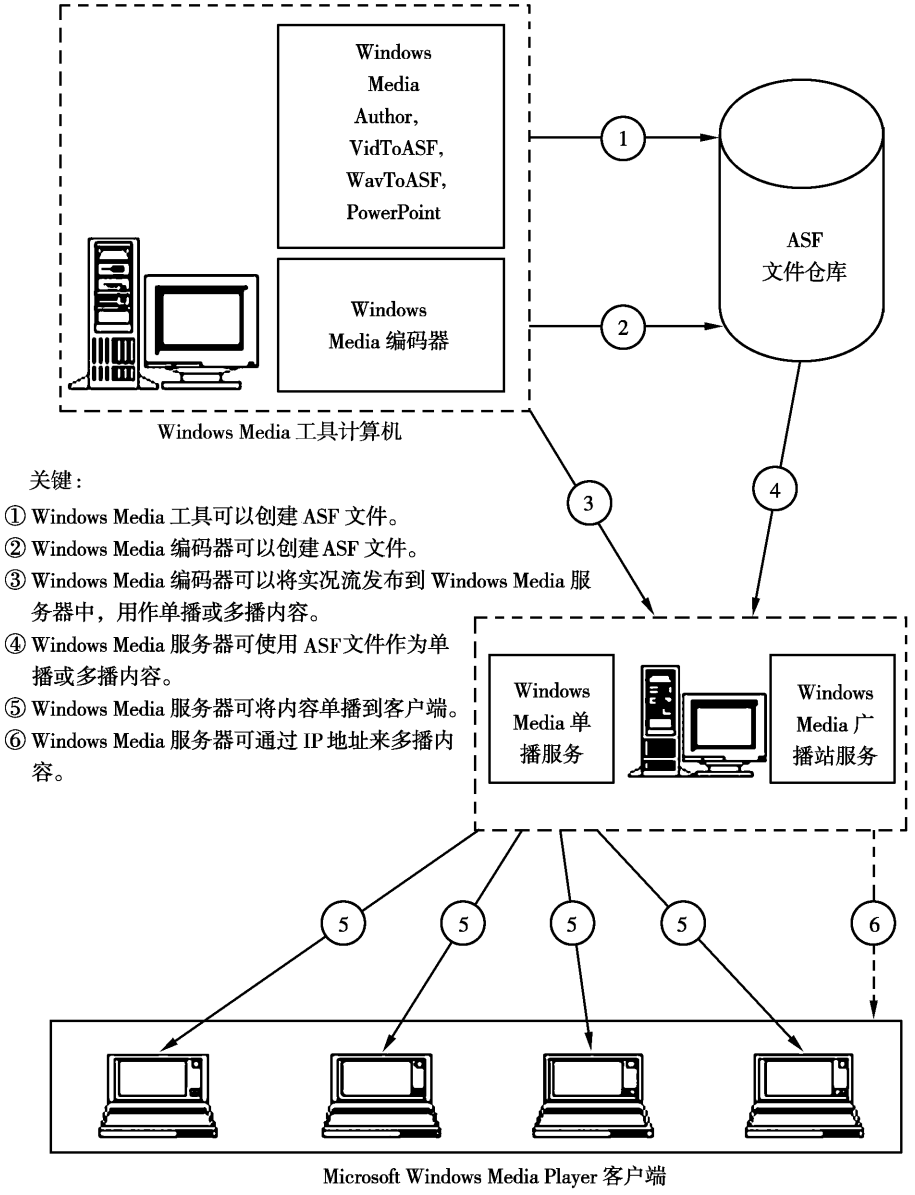


图 1.2　Windows Media 方案工作流程图

由此可见，一个 Windows Media 的流媒体系统是由不同的组件组成的，包含编

码、传输、播放等部分，可以根据不同的需求来进行灵活组合，满足各种不同的需求。Windows Media 的主要组件如下：

（1）Windows Media 编码器

Windows Media 编码器是一种易于使用、功能强大的工具，它可以将实况和预先录制的音频和视频转换为 Windows Media 流媒体文件。使用 Windows Media 编码器，可以为用户计算机输送实况内容，或将实况内容转换为文件以备后用。音频和视频内容的实时源包括可以插入声卡或视频卡的任何设备，如 CD 播放器、话筒、VCR、摄像机或视频播放器。被保存的源是音频或视频文件。用户可以使用 Windows Media Player 或任何使用 Windows Media Format 7 SDK 开发的应用程序来查看编码中基于 Windows Media 的内容。Windows Media Encoder（编码器）7.1 如图 1.3 所示。

图 1.3　Windows Media Encoder（编码器）

（2）Windows Media 服务器

Windows Media 服务器能够通过各类网络分流多媒体内容，从低带宽、拨号 Internet 网络连接到高带宽、局域网。如图 1.4 为 Windows Media 服务器管理操作界面。

（3）Windows Media 播放器

Windows Media Player 是在计算机和 Internet 上播放和管理多媒体的中心，转换流媒体文件为视频音频供用户观看，为操作系统提供多媒体功能选项，如图 1.5 所示，这就好像把收音机、电影院、CD 播放机和信息数据库等都装入了一个应用程序中。使用 Windows Media Player，可以收听世界范围内的广播电台播放和复制的 CD、

寻找 Internet 上提供的电影以及创建计算机上所有媒体的自定义列表。

图 1.4　Windows Media 服务器管理操作界面

图 1.5　Windows Media Player

(4) Windows Media 版权管理

通过将媒体文件打包来控制用户的数字媒体(如歌曲和视频)。打包的媒体文件中包含一份媒体文件,它已被压缩并进行了加密处理,并且捆绑了特定的 Web 站点的其他信息。其结果是,媒体文件更小了,但却具有高水准的音质,而且只有获得许可证的人才能播放它。此外,打包的媒体文件与播放它所需的许可证是分开的,从而允许用新的方式发布媒体和发放许可证。

(5) Windows Media 软件开发包(SDK)

使用 Windows Media 的技术,定制开发自己的流媒体系统,这部分是本书讲述的主要内容——以讨论 Windows Media SDK 为主。

1.1.2　Windows Media 技术的一些关键概念

通过 Windows Media 发布流媒体,关键问题有以下几点:

(1) 捕获和转换

第一件事情就是在计算机中捕获原始文件,然后使用 Windows Media 编码器将原始文件转换为流媒体文件,整个过程如图 1.6 所示。

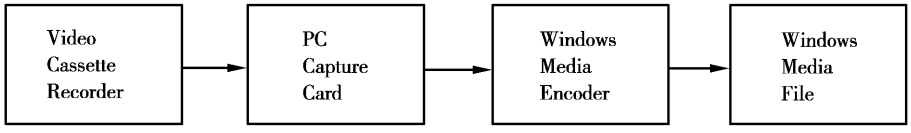


图 1.6　捕获转换过程

(2) 发布捕获和转换的内容

第二件事情就是通过网络将捕获和转换的内容通过网络发布出去,网络中有两种计算机:服务器和客户端。

客户端向服务器发出请求,服务器将请求数据发回客户端。发布的方式是多种多样的,例如通过 FTP,HTTP 或 Windows Media Services。其方式可以分为下载和流两种方式:

①流的方式:Windows Media 流媒体文件是针对使用流的形式传输而优化过的,要发布的内容可以被 Windows Media 服务器以流的形式在网络上传输。当原始文件经过 Windows Media 编码器编码后,就可以由 Windows Media 服务器通过网络发布,当服务器收到一个客户请求后,服务器将客户机请求的流媒体文件向客户机传送。

②下载方式:编码器编码后的文件可以独立存放,通过 FTP 或者 HTTP 的方式下载至本地硬盘,然后再播放。

两者的区别在于:流的方式并未在用户的硬盘等存储设备上保存,而下载方式是将文件保存在本地硬盘或其他设备中。

(3)播放

整个系统的最终目的就是播放,播放器将数字编码的流媒体文件重新还原为可以被观看的视频、音频。最终用户只需使用 Windows Media 播放器,就可以实现播放。通过 Windows Media Services 访问流媒体文件的方式与常用的网址类似,例如 mms ://WMServer/Mymovie. wmv,Windows Media 播放器可以识别这种格式,并通过此地址访问相关文件。

在文件完全下载到硬盘的时候,Windows Media Player 就可播放该文件。下载往往很花费时间,下载的时间和使用的带宽以及文件大小有关,而且必须等待所有的数据全部下载完毕后才可以播放。

当 Windows Media 编码器转换捕获来的音频、视频文件时,编码器会压缩数据,并将数据编码为 Windows Media 流媒体格式(可以将编码后的文件保存在存储设备上,也可以通过 Windows Media 服务器向请求播放的用户发送)。

1.2 Windows Media SDK 概述

Windows Media SDK 是微软发布的一个针对 Windows Media 技术的应用程序开发包,使用 Windows Media SDK 开发包,可以从事基于 Windows Media 的开发工作。该开发包由 6 个部分构成,如表 1.1 所示。

表 1.1 Windows Media 开发包的组成

SDK 组件名	功 能
Windows Media Player SDK	将 Media Player 播放器插入浏览器,创建界面和个性化效果,个性化 Windows Media Player
Windows Media Services SDK	配置和管理 Windows Media Service
Windows Media Right Manager SDK	解决数字版权问题
Windows Media Encoder SDK	开发具备编码功能的程序
Windows Format SDK	使软件可以读写和编辑 Windows Media 格式的文件和数据
Windows Media Embedded Product Adaptation Kit	开发便携式数字音乐播放器或其他系统

对于经常上网的用户来说,最为常用的就是 Windows Media Player SDK 了。应用 SDK,可以自定义 Media Player 外观,将 Media Player 嵌入网页中,对流媒体文件播放进行控制。在网上欣赏电影时,那些嵌在 IE 浏览器中的播放器,就是使用 Media Player SDK 定制的播放器。

可以看出,基于 Windows 组件设计的优势,微软公司提供了众多的软件开发包,用以支持 Windows Media 技术的应用。因为软件开发包可以给开发者更多的定制能力,使软件开发商基于不同的应用环境而开发出不同的流媒体系统。

1.3　学习方法

本书以 Microsoft 公司的 Windows Media 技术为基础，讲述利用 Windows Media SDK 开发的方方面面，由浅至深地介绍了如何使用 Windows Media SDK 开发适用于用户的流媒体系统，涉及视频直播、点播、存储、转换等内容。

本书的每一章都是一个单独的内容，第 8 章为综合应用，可以按照自己的需要来阅读。以下是作者的一些学习经验：

(1) 了解流媒体的基本概念

这是进行 Windows Media SDK 开发的基础，Windows Media 就是 Microsoft 公司针对流媒体技术而提出的解决方案。

(2) 有一定的编程基础，懂得类和对象等概念

利用 Windows Media SDK 进行开发，必须具备编程基础。本书主要使用 Microsoft Visual Basic 6.0 与 Visual C++ 来进行范例程序讲解，这是由于 Visual Basic 语言可以更直观地反映 Windows Media SDK 中类和对象的关系。Visual Basic 的用户非常多，而且对于新用户学习起来也更快捷。

Windows Media SDK 主要是以组件(也就是 COM)方式提供的，可以通过支持面向对象的编程语言来调用，例如 Visual Basic，Visual C++，Delphi 等。

(3) 多用、多交流、注重创新

交流可以使学习速度加快，在此，推荐 3 个学习交流的网址：Microsoft MSDN：http://msdn.microsoft.com；Microsoft 新闻组：news://news.microsoft.com；流媒体中文网站：http://www.liumeiti.com。

熟悉 Windows Media SDK 后，一定要多运用到实际中去，发现问题，并解决问题。

入门篇：Windows Media Player SDK

第 2 章

2.1 使用 Windows Media Player SDK

Windows Media Player SDK 是 Windows Media 技术中必不可少的组成部分，它是以 Windows Media Player 为操作对象，方便用户定制以 Windows Media Player 对象为基础的流媒体系统。

Windows Media Player SDK 主要包括 5 个功能设计：对象原型设计；外形界面设计（Skins）；为笔记本计算机、便携设备定制的开发包；可视化效果设计（Visualizations）；元素播放文件控制。

Microsoft Windows Media Player 为数字音频和视频提供了出色的播放效果，但如果需要显示更多信息，或者需要修改视频或音频内容的播放方式，那就需要使用 Windows Media Player 软件开发工具包（SDK），Windows Media Player SDK 可以扩展独立 Player 的功能，并将播放功能嵌入到自己的应用程序中。本节对 SDK 进行了概述，它涉及如下 3 个主要功能：

①创建应用程序：介绍如何在 Web 或基于 Windows 的应用程序中嵌入 Windows Media Player 功能。可以将 Player 嵌入 Web 应用程序或基于 Windows 的应用程序中。Windows Media Player 具有模块化体系结构，用户可以只使用所需的部分。尤其是，用户界面与音频和视频内容的播放功能相互独立，用户可以使用其播放功能，并可决定在应用程序中使用 Player 的现有用户界面或创建自己的用户界面。

②创建界面：介绍界面功能，更改 Windows Media Player 的界面和行为。Windows Media Player 提供了界面功能，用户可以使用该功能创建个性化的 Player 界面，也可以基于 Player 创建截然不同的界面。

③创建插件：介绍修改 Windows Media Player 界面和行为的插件。其中包括呈现专有内容、修改音频或视频播放，以及通过交互式控件提供全新的功能。还可以创建插件来扩展 Player 的主要功能，方法是向用户界面添加新的交互式控件，在 Player 呈现音频或视频数据前对其进行修改，然后在 Windows Media 文件中呈现非标准数据流。

表 2.1 为 Windows Media Player 支持的文件格式。

表 2.1 Windows Media Player 支持的文件格式

格　式	文件扩展名
CD 音频曲目	.cda
Intel Video Technology	.ivf
Macintosh AIFF Resource	.aif,.aifc,.aiff
Windows Media	.asf,.asx,.wax,.wma,.wmv,.wvx,.wmp,.wmx
Windows 格式	.avi,.wav
Windows Media Player 外观	.wmz,.wms
数字影视压缩标准（MPEG）	.mpeg,.mpg,.m1v,.mp2,.mp3,.mpa,.mpe,.mpv2,.m3u
乐器数字接口（MIDI）	.mid,.midi,.rmi
UNIX	.au,.snd

2.1.1　Media Player SDK 的安装配置

Windows Media Player SDK 的安装过程非常简单，执行 Windows Media Player SDK 安装文件，选择安装盘符，然后按照提示安装即可。默认安装目录会生成 WMSDK\WMPSDK 目录结构。Windows Media Player SDK 提供的相关示例文件位于 WMSDK\WMPSDK\docs\samples 目录下。

提示：本书配套光盘中包含 Windows Media Player SDK 安装文件，也可以从网址 http://www.liumeiti.com/download 下载。

2.1.2　创建应用程序

Windows Media Player 包括用于呈现视频和音频的 Microsoft ActiveX 控件，该控件可在任何运行 Windows Media Player 的计算机上获得。Windows Media Player 是一种独立的技术，此外，它还包括一个 ActiveX 控件形式的组件对象模型（COM）服务器（Player 与 ActiveX 控件之间的关系相当于 Microsoft Internet Explorer 与其所提供的 WebBrowser ActiveX 控件之间的关系）。

有两种方法可用于创建使用 Windows Media Player ActiveX 控件的应用程序：可以在 Web 应用程序中使用该控件，也可以在基于 Windows 的应用程序中使用它。

①要在 Web 应用程序中使用 Windows Media Player，应在页面的超文本标记语言（HTML）中包含一个 OBJECT 元素，并在 OBJECT 元素中包含嵌套的 PARAM 元素，以指定 Windows Media Player ActiveX 控件是否可见、包含哪些操作按钮以及该控件的其他属性。通过包含多个 OBJECT 元素，可在一个 Web 页面中包含多个控件。要完全控制嵌入的 Player，可以在页面的 HTML 中编写脚本代码。以下是一个应用于 IE 浏览器中的简单代码：

```
<OBJECT ID="MediaPlayer1" WIDTH=320 HEIGHT=240
  CLASSID="CLSID:22D6f312-B0F6-11D0-94AB-0080C74C7E95"
  CODEBASE="http://activex.microsoft.com/activex/controls/mplayer/en/
    nsmp2inf.cab#Version=6,4,5,715"
  STANDBY="Loading Microsoft® Windows Media™ Player components..."
  TYPE="application/x-oleobject" VIEWASTEXT>
   <PARAM NAME="FileName" VALUE="C:\ASFRoot\Welcome.asf">
   <PARAM NAME="ShowControls" VALUE="False">
   <PARAM NAME="AutoRewind" VALUE="True">
   <PARAM NAME="AutoStart" VALUE="False">
</OBJECT>
```

在 IE 浏览器中的浏览效果如图 2.1 所示。

图 2.1 浏览器中的 Windows Media Player

②要在基于 Windows 的应用程序中使用 Windows Media Player，可以包含一个服务于该控件的动态链接库（DLL）的引用。例如在 Microsoft Visual Basic 中，使用 Components（组件）对话框设置一个对 Windows Media Player（Wmp.ocx，文件中库的助记名称）的引用，如图 2.2 所示。

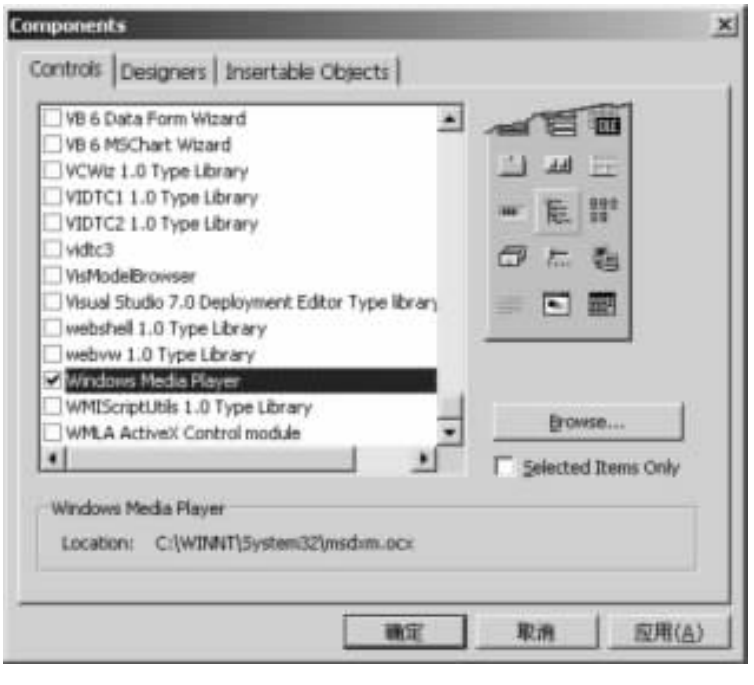


图 2.2 Windows Media Player 控件的引用

使用此控件，只需将其放置于设计窗体内即可，设计时的状态如图 2.3 所示。

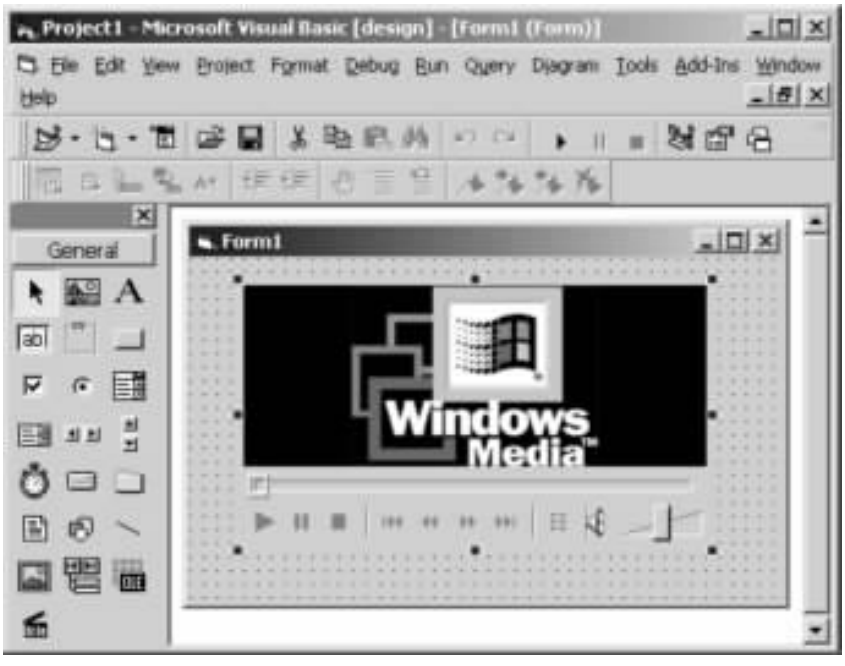


图 2.3　Windows Media Player 控件设计时的状态

运行程序，Windows Media Player 控件显示效果与 Windows Media Player 非常类似，如图 2.4 所示。

图 2.4　Windows Media Player 的运行状态

如何设置控件属性取决于所用的编程环境。例如在 Visual Basic 中，使用自定义 Properties（属性）对话框在设计时设置属性，也可以通过编写代码设置或读取属性，以及在运行时调用方法，在后续章节中将重点讲述。

最终用户可在任何安装了 Windows Media Player 的基于 Windows 的计算机上运行使用 Windows Media Player 控件的应用程序，这样就可以通过自定义创建的用户界面收听音频或观看视频。

2.1.3 创建界面

可以使用 Windows Media Player 中的界面技术更改 Player 的界面，同时保持它的原有功能。Windows Media Player 包含的界面便是该功能的体现。尽管这些界面使 Player 看起来迥然不同，但它们却都保留了 Play（播放）和 Pause（暂停）按钮、显示播放列表、返回到完整模式等基本功能。还可以使用界面技术创建一个界面和行为与标准 Player 相差很大的应用程序。可以随意在界面中添加各种各样的控件，使其执行自定义操作。要定义自定义操作，可以编写 JScript 代码，无需精通 Microsoft Visual C ++ 或 Visual Basic 即可将界面技术用作编程平台。

将界面定义文件、图像文件和任何 JScript 文件压缩到文件扩展名为 . wmz 的单个文件中，这是将界面提供给最终用户的标准方法。还可以创建称作边框的特殊界面，它显示在完整模式下 Windows Media Player 的 Now Playing（正在播放）功能中。可以将边框、播放列表文件和数字媒体文件压缩到文件扩展名为 . wmd 的单个可下载文件中。当最终用户单击该文件的链接时，Windows Media Player 将对包含的所有文件进行解压缩，将边框应用到 Now Playing 功能，然后开始播放在播放列表文件中指定的内容。

这种可下载文件的功能尤其适用于向最终用户提供出色的娱乐或教育内容，它不要求最终用户进行任何安装，同时允许进行广泛的自定义。

2.1.4 创建插件

Windows Media Player SDK 包含支持广泛扩展性的接口。要扩展 Player 的基本功能，可以编写借助于 DLL 文件的 COM 对象。

Windows Media Player SDK 包含一个可创建示例插件项目的 Visual C ++ 向导。该项目包含编译和注册插件所需的代码和一个示例实现。运行向导后，便可以将编程工作的重点放在实现插件所需的特定代码上。

利用插件提供的扩展性，可以将 Windows Media Player 用作提供和操作数字媒体内容的平台。插件有几种不同的类型，将在以下内容中进行讨论。

（1）可视化效果插件

当 Player 处于完整模式或界面模式（取决于界面）时，可视化效果插件可在只播放音频内容时，向 Player 的 Now Playing 功能添加有趣的动态图像。可视化效果的界面和动态运动基于正在播放的音乐并与之同步。

将可视化效果作为 COM 对象实现，Player 每秒多次为当前选定的可视化效果引发事件。该事件包含以下数据：

- TimedLevel 结构，其中包括当前音频的频率和音量信息。
- 设置上下文句柄，用于指定绘图面。
- RECT 结构，定义绘图面的大小。

• 在可视化效果对象中，实现处理此事件的 Render 方法。

可以使用 Windows 图形设备接口（GDI）功能、Microsoft Direct3D 或 DirectDraw 功能等技术对图形进行可视化编程。

在调用利用设备上下文的 GDI 函数时，Windows Media Player 插件向导生成的示例实现使用音频和音量数据（位于 TimedLevel 结构中）。可以修改或替换该实现，以生成所需的可视化效果。

该向导可以编写用于编译自行注册的 COM DLL 的全部代码。要观看可视化效果，只需编译项目，然后运行 Windows Media Player 并选择新的可视化效果即可。

（2）用户界面插件

Windows Media Player 为最终用户提供了各种信息和功能。但是，可能需要提供自定义交互或自定义数据，使用用户界面插件可完成此项工作。

Windows Media Player 的完整模式由许多区域组成，如 Now Playing（正在播放）功能和播放列表窗格。某些区域在默认状态下不可见，但最终用户可将其显示为可见。这些区域包括显示在 Now Playing 功能底部的设置区域，以及显示在播放列表上方的元数据区域。

用户界面插件分 5 种类型：显示插件、设置插件、元数据插件、窗口插件和背景插件，其中 3 种类型显示在 Player 的不同区域中。在每个区域中，每次只能启用一个插件。

①显示插件：这些插件占据 Now Playing 功能中的可视化效果显示区域。由于该区域通常很大，因此比较适合显示大量数据或复杂的交互控件集。

②设置插件：这些插件位于 Now Playing 功能中可视化效果显示区域的下面。该区域包含图形均衡器、视频设置和其他用于配置播放或 Windows Media Player 界面的控件。设置插件比较适合添加相似的自定义功能，并使最终用户能够配置 Player 的界面或行为。

③元数据插件：这些插件位于播放列表上方的一个小区域。它们比较适合显示曲目、唱片集或播放列表的确切信息，同时也适用于简单控件或超链接。例如 Windows Media Player 包含一个元数据插件，它显示的封面图形是一个指向有关当前播放的唱片集和音乐家的更多信息的链接。

④窗口插件：这些插件占用一个单独的窗口，它们比较适合显示信息或向最终用户提供从 Now Playing 功能切换到其他功能或加载其他插件时始终存在的交互操作。

⑤背景插件：这些插件没有图形界面（除非像对待任何插件那样，为它们提供属性页），它们比较适合不要求最终用户进行输入的自动服务。

Windows Media Player 插件向导可以为每种类型的用户界面插件创建示例实现。可以修改实现细节，以提供所需功能，编译项目，然后使用 Player 测试插件。该向导包括编译自行注册的 COM DLL 所需的所有代码，因而可以在编码时将重点放在特殊需要上。

(3) DSP 插件

数字信号处理(DSP)插件在播放过程中修改数字媒体流。使用 DSP 插件,可以将彩色视频更改为黑白视频,或使用反色使图像看起来像底片,还可在音频中添加颤音或回声效果。插件在处理数字媒体内容时将改变该内容的播放。DSP 插件与可视化效果插件不同,后者接收种子值形式的数字音频数据以生成视觉输出,但并不影响音频播放本身。

除 Windows Media Player SDK 之外,编写 DSP 插件还需要 Microsoft DirectX SDK。这些插件有一个明显特征,即实现由 DirectX SDK 提供的 IMediaObject 接口。当 DSP 插件安装在最终用户的计算机上并被激活时,Windows Media Player 在数据呈现之前将音频和视频数据传递给该插件,Player 分配输入缓冲区和输出缓冲区,并使插件能够对这两个缓冲区进行访问。必须实现 IMediaObject 接口的各种方法,以便从输入缓冲区读取数据,以适用于插件的任何方式对数据进行处理,然后将修改后的数据写入输出缓冲区,Windows Media Player 将呈现从输出缓冲区获取的数据。

可以运行 Windows Media Player 插件向导创建 DSP 插件的示例实现。该示例实现 IMediaObject 接口,并实现一个称为 DoProcessOutput 的实用程序函数。通常情况下,只需使用 DSP 插件的特有代码修改此实用程序,并利用向导提供的所有其他代码。

(4) 呈现插件

使用 Windows Media Format SDK,可以向 Windows Media 文件中添加任何数据流。此数据流的格式不受限制,包括 Windows Media Player 默认情况下无法识别的格式。

可用来呈现包含此类数据流的 Windows Media 文件有以下 2 种方法:

①可以编写一个自定义播放应用程序。这种情况下,除了包含用于呈现用户特有内容的代码之外,还必须包含用于提供标准音频、视频和脚本流的代码,以及用于显示用户界面的代码。

②也可以为 Windows Media Player 编写一个呈现插件。这种情况下,仍须编写用于呈现用户特有内容的代码,但是可以利用 Player 的固有功能来呈现支持的流,并提供最终用户有所了解的用户界面。

Windows Media Player 插件向导可以创建用于呈现插件的示例实现。该示例可实现呈现插件所需的许多接口,还可以实现一个称为 DoRendering 的实用程序函数。只需使用用于呈现特有流的代码修改此实用程序,并利用向导提供的所有其他代码。

2.2　Windows Media Player 的可视化效果

使用 Windows Media Player 的用户可能都注意到它自带很多漂亮的界面和视觉效果。听音乐的时候，一边用一个酷到极点的播放器听，一边看着色彩绚丽的线条随着节拍跳动，这种感觉非常令人心动。

Windows Media Player SDK 中的 Windows Media Player Custom Visualization 部分就是面向界面和可视效果开发人员，使他们可以开发新的图形界面和可视效果以个性化 Windows Media Player。基于简单但功能强大的 XML 架构，界面技术可以使开发人员轻松自定义 Windows Media Player 7 的界面和功能。

Microsoft Windows Media Player 允许用户选择各种各样的标准界面，虽然它提供了许多的界面可以选择，但是创建一个用户自定义界面也是很简单的。本节主要介绍 Media Player 的界面原理，并结合实际制作介绍了一个简单的例子。需要用的知识包括图形处理软件的使用，如 photoshop，XML（可扩展标记语言）和 JScript 等。由于篇幅所限，在此就不再详细介绍，可自行查阅相关资料。

2.2.1　Windows Media Player 的界面

1）什么是 Windows Media Player 界面

Internet 要求打破常规和扩展想像力。新一代的 Internet 媒体播放机色彩华丽，并有各种各样的外观。通过新的 Windows Media Player，播放器的概念也焕然一新。用新的播放器平台，不仅可以设计色彩华丽、有创意的界面，还可以实现目前尚无法实现的事情。有了界面之后，演示设备不再需要无声地隐藏在后台——它可以成为其播放内容的一部分。

通过 Windows Media Player 界面，不仅拥有了内容，而且还拥有了回放内容的容器。界面可以是简单的创意表达，也可以结合商务需求。Windows Media Player 平台提供了工具和大量可能的方案，通过用户提供的创意和动力，就可以使播放器变为用户需要的模样。用界面可以：

①娱乐：界面的编程语言非常简单，任何了解 HTML 的人都可以快速和方便地创建自己的播放器（在 Windows Media Player SDK 的帮助下），例如最终用户可以用扫描的个人工艺品创建界面。由于编程接口非常易学和容易实现，因此最终用户可以将大部分时间放在创建很酷的工艺品上，而花在编码上的时间则非常少。

②推广品牌：编程语言非常容易学习，但这并不意味着它受到限制。用户完全可以在自定义的软件包中提供商业内容（包含商业的企业品牌和任何特殊的播放功能）。例如，媒体商务可以提供包含公司图徽和色彩的播放器界面，以及类似于视频窗口的特殊播放器功能、自定义播放器控件和可供最终用户选择音乐的播放列表，或者某个音乐组织可以提供可下载的播放器，将它设计为接近自己的风格，或是特别为

某个新 CD 创建的。播放器界面本身可以包含来自新 CD,甚至是自定义音量、平衡器控件和可视化的工艺图和音乐剪辑。

③开发产品 :除了添加品牌和个性化功能之外,还可以用编程接口开发能够成为更大软件包一部分的产品或者完全基于播放器的产品。在即将介绍的 SDK 中可以访问 500 多个属性、方法和事件,用它们可以构造复杂的、使用播放器自己的基于 XML 脚本语言,以及 Microsoft JScript ® 的界面程序。播放器的 ActiveX 控件 API 已经扩展了以前的版本,提供了创建更复杂的、使用高级语言(例如 Microsoft Visual Basic)的程序的所有工具。用户创建的播放器产品,可以成为远距离教学软件包的整合部分,或者实现自定义的通用媒体播放器接口。

用简单的 Windows Media Player 脚本,可以快速而方便地创建成为内容的整合部分的界面。在计算机的虚拟世界之外,用户无法轻易地完成此类事情。在有了界面以后,不仅可以控制媒体,还可以控制播放媒体的环境。

Windows Media Player 允许用户选择符合标准的视觉界面,也就是现在非常流行的 Skin 功能,界面功能可以给用户的视听感受带来最大的乐趣,如图 2.5 所示。

图 2.5 Windows Player 界面

自定义的 Windows Media Player 界面或功能文件,当 Windows Media Player 处于紧凑模式时,它将以该界面显示和运行。Windows Media Player 界面的制作并不是很复杂,甚至一个简单的界面只由几个文字操作即可完成,在这个章节,将详细介绍如何制作 Windows Media Player 界面。

Windows Media Player 平台提供了丰富的编程接口,其他的 Internet 播放器只允许创建能够控制有限播放器功能的界面。用 Windows Media Player 可以构造包括许多可自定义功能的完整播放器界面。

①可交互性：通过编程接口可为使用界面的最终用户提供丰富的交互经验。用户编写的代码使最终用户能够控制播放器的功能和界面，并能将信息发送回用户。例如最终用户播放的“Windows 媒体”文件可以包含改变界面图像的脚本命令。用户可以通过单击图像触发“单击”事件，该事件运行 Microsoft JScript 功能，将播放改变到不同的标记位置，如图 2.6 所示。

图 2.6　交互性示例

②播放控件：可以定义图像控制或者响应播放器的区域。例如用户可以在图像中绘制红色的圆圈，并将该区域的图像映射为播放器的播放功能。当最终用户单击该圆圈时，媒体将开始播放。也可以创建自定义滑块，并将它映射为音频平衡器或者视频亮度设置，还可以使用播放器的预定义平衡器、按钮和滑块控件。

③播放列表：可以在界面定义文件中定义播放器播放的内容，还可以向最终用户提供用标准打开文件对话框或者使用播放列表控件选择内容的方式。最终用户可以从头到尾连续播放列表，或者选择单独的片段，Windows Media Player 中的播放列表如图 2.7 所示。

图 2.7　播放列表示例

④子视图：除了主界面图像之外，界面还可以包含任何数量的其他视图（子视图），最终用户可以在它们之间切换。用户可以为每个子视图创建不同的布局、控件子集和脚本。用子视图可以使界

面更整洁,因为它可以隐藏像平衡器和播放列表之类的控件组,直到用户需要时才显示它们。

⑤可视化 :通过添加可视化,可向最终用户提供多彩的视觉效果。可视化内容是随声音移动的,用色彩和视觉强度来反映频率和音量的变化(如图 2.8 所示)。播放器附带了许多可视化内容。Windows Media Player SDK 也为使用 Visual C ++ 创建自己的可视化内容提供了编程信息。

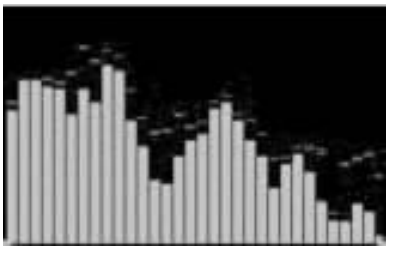


图 2.8 条纹和波纹可视化

⑥图像增强 :除了主界面图像之外,还可以使用其他图像来增强界面的交互性和功能。通过映射图像可以快速地将按钮放在界面上,盘旋和升降图像可为最终用户提供反馈。ImageButton 控件包含的图像可在播放器运行过程中改变。

2)如何应用 Windows Media Player 界面

在 Windows Media Player 中,使用界面可以更改其在紧凑模式下的界面。各个界面均有独特的外貌,与 Windows Media Player 的基本功能(例如播放、上一曲目、下一曲目、停止以及调整音量等)相结合。界面可以播放特定的音频文件,或者显示特定的可视化效果及各种其他功能。

必须在“CD 音频”选项卡上选中“数字播放”复选框,可视化效果才会在播放 CD 曲目时出现。要验证是否选中了该复选框,可在“工具”菜单上单击“选项”,然后单击“CD 音频”选项卡。

应用界面后,只要从完整模式切换到紧凑模式,即会显示该界面。要在应用界面前预览界面,应在“界面选择器”功能中,单击该界面的名称,将 Windows Media Player 切换到紧凑模式时,界面即被应用。界面可以随时更改,但必须在完整模式下进行。

自行设计应用的界面可能不支持 Windows Media Player 提供的所有功能。

Windows Media Player 未应用界面的状态参见图 2.1,Windows Media Player 应用界面的状态参见图 2.5。

3）Windows Media Player 界面的构成

在创建播放器界面之前，有必要先要了解界面的原理。当欣赏一个漂亮的界面时，看见的只是一个完整的表面，实际上在后台还有很多文件构成了完整的界面，技术术语称为：一个界面就是一个文件组，每个文件包含一些特定的信息。界面文件组包含的类型和用途如表 2.2 所示。

在应用一个界面的时候，用户所见到的只是一个表面（可能是一个图片，也可能是一些文字等），正是这个界面与其他的一些文件构成了一个完整的界面，也就是说一个界面由多个文件构成，使用表 2.2 中的界面构成文件就可以制作一个 Windows Media Player 的界面。

表 2.2　构成界面的文件类型

文件类型	说　明
界面定义文件	告诉 Media Player 如何使用界面文件组中的其他文件。这是一个文本格式的文件，扩展名为. wms（Windows Media Skin 的缩写）
图形文件	简单地说就是构成用户直观视觉效果的组成文件，可以是. bmp，. jpg，. png 等被 Windows 操作系统所直接支持的文件
脚本文件	用以操作、响应更复杂的事件和请求，这也是一个文本文件，扩展名为. js

要创建界面，需要如下步骤：

①创建工艺图：任何 BMP，JPG，GIF 或 PNG 格式的图像文件都可以作为主界面图像，也可用支持的格式的映射、盘旋和剪辑图像。因为可以使用层次来构造工艺图，因此最好用 Adobe Photoshop 这样的程序来创建和编辑图像。可以导入主图像，然后在它上面创建映射、盘旋和剪辑层，它们都可以保存为单独的文件。

②创建界面定义文件：扩展名为 . wms 的短文本文件可以为播放器提供所有内容文件的位置和如何创建用户界面的指令，也可以用 JScript 在外部 . js 文件中添加更复杂的编程功能。例如可用 JScript 自定义音量滑块的操作。

③制作 ZIP 文件：在制作完工艺图和界面文件之后，使用 ZIP 程序将所有文件压缩并组合到. zip 文件中，并且用扩展名 . wmz 保存它。当最终用户双击 . wmz 文件图标时，Windows Media Player 将打开压缩文件中的界面和其他内容。

（1）创建界面工艺图

使用两种或更多图像类型构造的界面：

①主背景图像：任何图像都可以作为主播放器界面，只要它们是用支持的格式创建的。界面可以像一个播放器，也可以像一只大象，界面的显示如用户所愿：如果需要，它可以包括标准 Windows 菜单栏，还可以指定它的任何方面，包括边界的大小和形状。通过编程接口的 ClippingColor 属性，可以在主图像周围创建任何形状的边界。

图 2.9　演示创意设计的界面

②映射图像：用映射图像可以定义主界面图像的区域，向该区域映射某些播放器功能。用 Adobe Photoshop 中的层次来创建映射区域。为映射层绘制对应于主图像上区域的形状，然后用单一颜色填充此形状。播放器不会显示此映射图像，但是将彩色区域链接到鼠标事件。例如在将绿色映射为播放功能后，当最终用户单击该区域时，内容便开始播放。

③辅助图像：这些图像为最终用户提供了反馈，因此有助于界面的交互。它们也可以添加动作和颜色。辅助图像包括“盘旋”、“下降”、“禁用”和“向下盘旋”。

(2) 创建界面定义文件

界面定义文件包含将图像链接到 Windows Media Player 的指令。虽然看不见播放器在后台运行，但是工艺图向最终用户提供了播放器的图形界面。Windows Media Player SDK 描述了许多可以控制和接收来自播放器的信息的方法。开发者可以决定用工艺图和界面定义文件向最终用户提供什么样的控制和信息。例如在最终用户单击时，他可以从文件中的代码得知播放器背景图像为哪一种，不同区域的图像控制哪些不同的播放器功能。默认的播放器界面如图 2.10 所示。

图 2.10　默认的播放器界面

界面定义文件包括下列基本信息类型：

①图像定义：定义了所有图像的名称、大小和位置，以及如何使用图像背景、映射或盘旋，还定义了诸如 TransparencyColor 和 MappingColor 等属性。

②控件的位置和属性：用于控制播放器的按钮、JScript 函数或者界面属性，以及其他控件，例如音量调节滑块、视频窗口、文本框、播放列表和可视化内容。

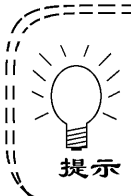
③JScript 函数：在 JScript 文件中编码的复杂函数和事件是从界面定义文件中的语句调用的。

④效果设置：确定可视化控件大小和位置的属性，以及可视化类型和显示特性。

⑤音频和视频设置：确定视频窗口、音频平衡器大小、位置的属性，以及亮度和色彩强度等视频设置。

4）定制 Windows Media Player 的界面

懂得 Windows Media Player 的界面构成，就可以定制一个界面。



Windows Media Player 支持将构成一个界面的多个文件，放入一个 ZIP 压缩包中，这样，在用户那里看到的就是一个文件，便于使用和传输，这个文件的扩展名为. wmz（Windows Media Zipped 缩写）。

（1）定制界面定义文件

界面定义文件是 XML 格式的文本文件，下面是一个界面定义文件的全部代码（配书光盘中/Sample/Sample2_1 目录下）。

```
<THEME>
    <VIEW
        clippingColor = "#CCCC00"
        backgroundImage = "background.bmp"
        titleBar = "false"
        >
          <BUTTONGROUP
            mappingImage = "map.bmp"
            hoverImage = "hover.bmp"
             >
<BUTTONELEMENT
        upToolTip = "Open and play the file 'laure.wma'"
        onClick = "JScript:player.URL = '..hh..hhmedia hhlaure.wma';"
        mappingColor = "#00FF00"
                /  >
          <BUTTONELEMENT
                upToolTip = "Close"
```

```
                mappingColor = "#FF0000"
                onClick = "JScript:view.close();"
                />
    </BUTTONGROUP>
    </VIEW>
    </THEME>
```

使用 HTML 编写网页可能会觉得这个脚本很熟悉,每个 WMS 文件以 <THEME> … </THEME> 的形式开头和结尾,其中用 XML 在 Skin Definition File 文件中用来定义界面界面上的元素标签。例如,<BUTTONGROUP> 标签就定义了一个按钮,同时,每个元素标签都拥有对应的属性,例如,Button 元素就拥有一个 Image 属性,定义该按钮可以使用的图形界面。

脚本使用下面的元素:

①THEME:将用于播放器的文件定义为界面定义文件。

②VIEW:包含给定界面视图的所有元素。用一个界面定义文件就可以创建多个视图和子视图,并且为最终用户提供在它们之间切换的方式(例如通过单击按钮)。例如,可以为音乐 CD 上的每个音轨提供不同的视图。

③BUTTONGROUP:包含一组按钮的映射和盘旋图像定义。

(2)图形文件

构成界面界面的图形文件,主要格式为 BMP,JPG,PNG,GIF,其基本类型分为:

①Primary Images(基本图像):界面的背景图像,用 view 标签定义。

②Mapping Images(映射图像):定义控制图形的点击响应区域。

③Alternate Images(交替图像):相应点击事件时变化的图像。

(3)脚本文件

在 Windows Media Player 的界面中,可以用 Microsoft Jscript 来扩展功能,比如可以建立自定义的节目列表,这些都是通过 Jscript File 来实现的,该文件为文本文件,文件的扩展名为 .js,在 Skin Define File 中通过 VIEW 元素的 scriptfile 属性加载。

Windows Media Player 界面编程接口提供了用于构造复杂程序、自定义业务应用程序和无限多种独特集成媒体播放器解决方案的创意工具。不管用户是创建复杂程序以便在 Internet 上发布媒体的开发人员,还是寻求使公司品牌更显眼的媒体业务人员,或者是业余艺术家,都可以使用界面来提高产品或创意项目的价值。

2.2.2　Windows Media Player 的视觉效果

除了界面之外，Window Media Player（7.0 或以上版本）还提供了用户自定义视觉效果的接口，在安装了 Windows Media Player SDK 后，在安装目录中（hWMSDK hWMPSDK hwizards hviz）找到 Windows Media Player Custom Visualization Wizard，应用该向导，用户可以自定义出令人目眩的视觉效果。

可以通过一个例子来说明如何自定义 Windows Media Player 的视觉效果。在制作一个视觉效果前，需要拥有以下软件环境：

①Windows Media Player SDK：提供了开发文档和例子，更重要的是，还提供了 Windows Media Player Custom Visusalization Wizard。

②Microsoft Visual C++ 5.0 或更高版本：用来编译，生成视觉效果。

③Windows Media Player：用来测试视觉效果。

在安装了必须的软件后，就可以进行到下一步，开始制作第一个视觉效果。

（1）安装 Custom Visualization Wizard

前面说过，一旦安装了 Windows Media Player SDK 后，在安装目录中就有 Windows Media Player Custom Visualization Wizard。向导文件称 WMEffect.awx。现在将该文件拷贝至 hProgram Files hMicrosoft Visual Studio hCommon hMSDev98 hBin hIDE（即为 Microsoft Visual Studio 的安装目录，如果用户的 Microsoft Visual Studio 安装在别的目录，拷贝该文件到对应目录即可）。

（2）应用 Custom Visualization Wizard

打开 Visual C++，点击 File→New 菜单，建立一个工程，如果用户刚才已经正确

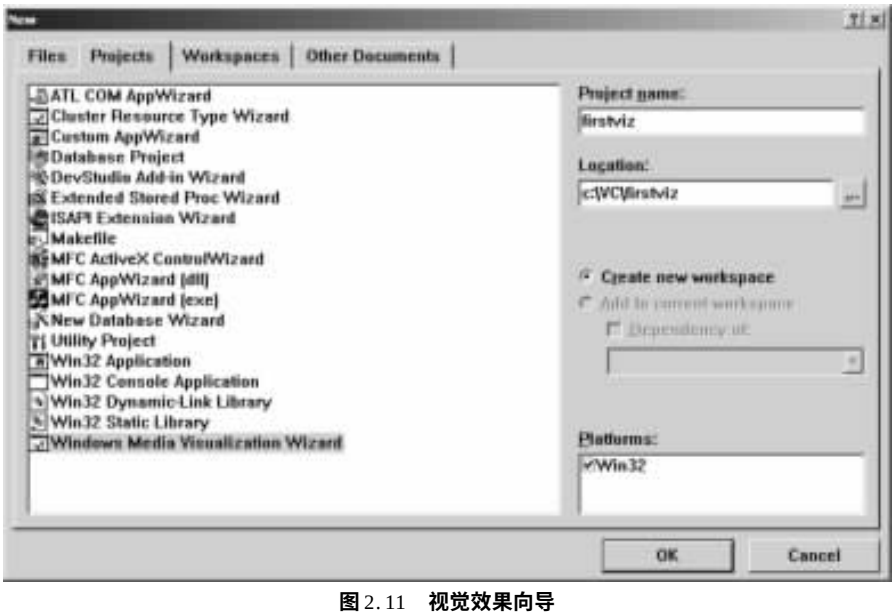


图 2.11　视觉效果向导

地拷贝 WMEffect. awx 到 Visual Wizard 目录中，就可以发现一个叫“ Windows Media Visualization Wizard ”的类型可供选择，选中该选项，并给出工程名，设定好工程的保存路径，设为 C :\VC\firstviz。点击确认按钮，如图 2. 11 所示，VC ++ 将会建立该工程。在 Build 菜单中点击 Build firstviz 选项，VC ++ 会创建一个视觉效果到工程目录。firstviz 所包含的类如图 2. 12 所示。

图 2. 12　firstviz 工程构成

(3) 测试效果

打开 Windows Media Player，会发现在可视化效果选项菜单中多出了 2 个用刚才的工程名命名的视觉效果，如图 2. 13 所示。

图 2.13　自定义视觉效果

视觉效果分别是 Firstviz Bars 和 Firstviz Wave。打开一首歌曲，效果如图 2.14 和图 2.15 所示。

图 2.14　Firstviz Bars 视觉效果

图 2.15　Firsoviz Wave 视觉效果

(4)进一步扩展

现在已经拥有了第一个自己的视觉效果。可能你会觉得它过于简单,效果也不是那么华丽,想要对它加以改进,这时候就需要改变视觉效果的渲染方式了。可以通过修改 Render Function 的 TimedLevel,HDC,RECT 参数来改变视觉界面,这几个参数的具体作用如表 2.3 所示。

表 2.3　视觉参数

参　数　名	作　　用
TimedLevel	定义代码将要分析的音频数据
HDC	用于特殊绘图过程调用
RECT	定义绘图区域的尺寸

Render 的函数原型如下所示,可以自己按照图形变化算法来定制视觉效果。

```
//////////////////////////////////////////////////////
// CFirstviz::Render
// Called when an effect should render itself to the screen.
//////////////////////////////////////////////////////
STDMETHODIMP CFirstviz::Render(TimedLevel *pLevels, HDC hdc, RECT *prc)
    {
```

```
    // Fill background with black
    HBRUSH hNewBrush = ::CreateSolidBrush( 0 );
    HPEN hNewPen = ::CreatePen( PS_SOLID, 0, m_clrForeground );
    HPEN hOldPen = static_cast < HPEN > ( ::SelectObject( hdc, hNewPen ));
    ::FillRect( hdc, prc, hNewBrush );
    // draw using the current preset
    switch (m_nPreset)
    {
    case PRESET_BARS:
    {
    // Walk through the frequencies until we run out of levels or drawing surface.
    for (int x = prc->left;x < prc->right && x < (SA_BUFFER_SIZE - 1);
++x)
    {
    int y = static_cast < int > (((prc->bottom - prc->top)/256.0f) * pLev-
els->frequency[0][x - (prc->left - 1)]);
    ::MoveToEx( hdc, x, prc->bottom, NULL );
    ::LineTo(hdc, x, prc->bottom - y);
    }
    }
    break;
    case PRESET_SCOPE:
    {
    // Walk through the waveform data until we run out of samples or drawing sur-
face.
    int y = static_cast < int > (((prc->bottom - prc->top)/256.0f) * pLev-
els->waveform[0][0]);
    ::MoveToEx( hdc, prc->left, y, NULL );
    for (int x = prc->left;x < prc->right && x < (SA_BUFFER_SIZE - 1);
++x)
    {
    y = static_cast < int > (((prc->bottom - prc->top)/256.0f) * pLevels->
waveform[0][x - (prc->left - 1)]);
    ::LineTo(hdc, x, y);
    }
    }
    break;
    }
    if (hNewBrush)
```

```
        {
          ::DeleteObject( hNewBrush );
        }
    if (hNewPen)
        {
            ::SelectObject( hdc, hOldPen );
            ::DeleteObject( hNewPen );
     }
    return S_OK;
}
```

2.3 Windows Media Player 控件

Windows Media Player SDK 不仅提供了界面扩展功能，同时，Windows Media Player ActiveX Control 也提供了 Media Player 的 ActiveX 控件的属性和方法。将 Windows Media Player 嵌入 Web 页面时，通过设置这些属性和方法，可以达到控制播放内容的目的。下面具体介绍 Media Player 对象。

1）什么是 ActiveX 控件

ActiveX 是微软提出的一种可重用的软件组件，应用广泛，特别是在网页开发中更是常见，通过使用 ActiveX 控件，使用者可以把它当作预装配组件，使用多种语言通过设置它预封装的属性和方法来实现各种功能而无需了解技术细节，既方便又快捷。只要用户的机器上安装了 Windows Media Player，那么开发者就可以将对应版本的 ActiveX 控件应用于设计。

2）Windows Media Player 控件的参数

打开 FontPage，新建一个空白页面，用鼠标定位想要插入控件的地方，选择“插入→高级→ActiveX 控件”选项，在弹出的控件窗口中选择“Windows Media Player”，点击“确定”。这时，会看见一个像 Media Player 的播放窗口一样的控件被添加到网页中，如图 2.16 所示。

图 2.16　浏览器中的 Player 控件

查看刚才制作的页面的源代码，会看到如下代码：

```
<object Classid = "clsid:22D6F312 - B0F6 - 11D0 - 94AB - 0080C74C7E95"
id = "MediaPlayer1"  width = "353"  height = "299"  align = "center"  style = "
border - style：ridge；border - color：#C0C0C0" >
<param name = "AudioStream"  value = " - 1" >
<param name = "AutoSize"  value = "0" >
<param name = "AutoStart"  value = " - 1" >
<param name = "AnimationAtStart"  value = "0" >
<param name = "AllowScan"  value = "0" >
<param name = "AllowChangeDisplaySize"  value = " - 1" >
<param name = "AutoRewind"  value = "0" >
<param name = "Balance"  value = "0" >
<param name = "BaseURL"  value >
<param name = "BufferingTime"  value = "5" >
<param name = "CaptioningID"  value >
<param name = "ClickToPlay"  value = " - 1" >
<param name = "CursorType"  value = "0" >
<param name = "CurrentPosition"  value = " - 1" >
<param name = "CurrentMarker"  value = "0" >
```

```
<param name = "DefaultFrame" value >
<param name = "DisplayBackColor" value = "0" >
<param name = "DisplayForeColor" value = "16777215" >
<param name = "DisplayMode" value = "0" >
<param name = "DisplaySize" value = "0" >
<param name = "Enabled" value = " - 1" >
<param name = "EnableContextMenu" value = " - 1" >
<param name = "EnablePositionControls" value = " - 1" >
<param name = "EnableFullScreenControls" value = "0" >
<param name = "EnableTracker" value = " - 1" >
<param name = "Filename" value = "file. wmv" >
<param name = "InvokeURLs" value = " - 1" >
<param name = "Language" value = " - 1" >
<param name = "Mute" value = "0" >
<param name = "PlayCount" value = "1" >
<param name = "PreviewMode" value = "0" >
<param name = "Rate" value = "1" >
<param name = "SAMILang" value >
<param name = "SAMIStyle" value >
<param name = "SAMIFileName" value >
<param name = "SelectionStart" value = " - 1" >
<param name = "SelectionEnd" value = " - 1" >
<param name = "SendOpenStateChangeEvents" value = " - 1" >
<param name = "SendWarningEvents" value = " - 1" >
<param name = "SendErrorEvents" value = " - 1" >
<param name = "SendKeyboardEvents" value = "0" >
<param name = "SendMouseClickEvents" value = "0" >
<param name = "SendMouseMoveEvents" value = "0" >
<param name = "SendPlayStateChangeEvents" value = " - 1" >
<param name = "ShowCaptioning" value = "0" >
<param name = "ShowControls" value = " - 1" >
<param name = "ShowAudioControls" value = " - 1" >
<param name = "ShowDisplay" value = "0" >
<param name = "ShowGotoBar" value = "0" >
<param name = "ShowPositionControls" value = " - 1" >
<param name = "ShowStatusBar" value = "0" >
<param name = "ShowTracker" value = " - 1" >
<param name = "TransparentAtStart" value = "0" >
<param name = "VideoBorderWidth" value = "0" >
```

```
<param name = "VideoBorderColor" value = "0" >
<param name = "VideoBorder3D" value = "0" >
<param name = "Volume" value = " - 600" >
<param name = "WindowlessVideo" value = "0" >
</object >
```

可以看到，在 IE 浏览器中，Windows Media Player 的 ActiveX 控件是通过 Object 标签来嵌入的，而对于 Natscape 来说，则要用 EMBED 标签。在 Object 标签后的 Classid = "clsid:22D6F312 - B0F6 - 11D0 - 94AB - 0080C74C7E95" 代表了控件的类型和版本，而代码中每一个 param name 标签后面都是 Media Player 的参数，可以在脚本中设置这些参数，也可以在普通窗口中直接点击该控件，在弹出的设置对话框中设置参数，以达到控制播放的目的。下面对几个比较常用的参数进行说明。

①Filename 参数：指定播放文件，其值可以是本地文件，也可以是一个流，如 mms://myvodserver。

②Autostart：是否自动播放媒体，0 代表禁止，1 代表自动开始。

③Autosize：是否在播放开始时自动使用影片尺寸，0 为可以，1 为禁用。

④ShowAudioControls：是否禁止浏览者调节播放器音量。

⑤ShowStatusBar：是否禁用状态条。

⑥ClickToPlay：是否可以通过点击播放区域来暂停和播放。

Windows Media Player 的参数有几十个之多。此外，Windows Media Player 的 ActiveX 控件已经有多个版本，它们的 Classid 不同，支持的方法属性也不近相同，通过设置 Classid 可以改变控件的版本，当然，这需要机器上安装有对应版本的播放器。

3）Windows Media Player 对象的方法和属性

现在可以向浏览器中插入控件并了解了一些基本参数的作用，通过设置 FileName 属性，可以在这个浏览器中播放视频节目了。但如果想要使这个播放器更个性化一些，比如用户不满足于使用 Windows Media Player 现成的界面，想要自己定义播放器节目，通过自定义的按钮来实现控制播放，只了解这些还是不够，还需要学习 Windows Media Player 控件对象的知识。

和任何对象一样，Windows Media Player 控件对象拥有自己的属性和方法可供开发人员设置和使用，对于 Windows Media Player 6.4 控件来说，它拥有的众多属性和方法可以归为以下几类：

（1）回放

Windows Media Player 控件提供了两种定义媒体片段的方法，可以通过设置 FileName 属性，也可以调用 Open 方法。同时提供了以下的方法和属性用于控制流回放：

Play，Stop，Pause 方法：顾名思义，用来开始、暂停、停止节目。

PlayCount 属性：设置播放次数。

AutoRewind 属性：用来设置是否在播放完毕后返回到节目开始位置。

(2) 音频控制

用来控制音频，主要有以下几个属性：

Balance 属性：设置左右声道。

Value 属性：控制调节音量。

Mute 属性：是否静音。

(3) 浏览和定位

主要提供对节目的定位和浏览控制，如快进、快退、精确定位时间等等。

如果想要控制 Media Player 对象的行为，就可以使用这些方法。比如，想要播放一个文件，可以在按钮事件里面添加对象名. play() 来实现；同样如果想要停止播放，就可以用 stop 方法，即在按钮事件里面添加 对象名. stop()，常用的操作方法如表 2.4 所示。

表 2.4

操　作	方　法
播放	MediaPlayer. Play()；
暂停	MediaPlayer. Pause()；
停止	MediaPlayer. Stop()；
上一节	MediaPlayer. Previous()；
快退	MediaPlayer. FastReverse()；
快进	MediaPlayer. FastForward()；
下一节	MediaPlayer. Next()；

以上介绍的只是针对 Windows Media Player 6.4 控件而言，由于 Windows Media Player 控件目前已经有了多个版本，不同的版本提供的方法和属性也不尽相同，以前支持的方法也可能不再支持，例如在新版本控件中，对象名. play () 方法变成了对象名. controls. play()。

2.4 实例一：制作 Windows Media Player 界面

一般来说，制作一个 Windows Media Player 的界面，要进行以下几个步骤，为了方便讲解，这里以第一节的源码作为例子，该例子可以在 Wmpsdk 的 hdocs hsamples 目录中找到，实例名为 openmed。

第一步：建立主图形文件。

打开 Photoshop，现在要制作主图形文件，主要包含以下几个图层：

(1) 界面背景层

是界面的背景层，当界面显示的时候为透明，可以用 Photoshop 来新建一个图像文件，大小为 200 × 100，颜色设置成淡黄色。

(2) 界面包含层

定义用户可以看见的界面边缘。用 Photoshop 的 Eliptical Marquee 工具建立一个椭圆的层，大小比背景层略小，颜色略有不同，如图 2.17 所示。

图 2.17　界面包含层

(3) 按钮

播放停止按钮。新建一个层，名为 layer_play，再次使用 Eliptical Marquee 工具画一个圆，把它定位到左边，并位于所有图形层之上，定位好后填充颜色，使之看起来如图 2.18 所示。

图 2.18　按钮

再次新建一个层，命名为 layer_close，用同样方法再制作一个按钮，填充和刚才不同的颜色，调整图层到适当的位置。

合并图层并且保存文件，将文件另存为 *.bmp 格式，并命名为 background.bmp，最终效果如图 2.19 所示。

图 2.19　合并图层

按钮可以使用除包含层以外的所有颜色，以避免和包含层混淆。

第二步 :创建映射图像文件。

该层主要用来定义点击响应区域。用刚才创建的图形文件,去掉界面包含层,使用白色填充背景层,将两个按钮分别填充为#FF0000**(红色),**00FF00**(蓝色),完成的效果图如图** 2.20 **所示。在图中红色和绿色区域就是响应鼠标点击的区域。**

图 2.20 **点击响应**

当使用别的颜色时,一定要记住所使用的颜色代码,以便用颜色代码来定义鼠标的响应区域。

建立交替显示图像文件,使用刚才建立的两个按钮层,填充为黄色(#CCFF33**),并修改外形如图** 2.21 **所示。**

图 2.21 **建立交替显示图像文件**

当用户点击鼠标响应区域时,按钮区域将显示为该图界面以提示用户按钮的作用。制作好用于界面的图片后,分别将它们保存为 background. bmp**,**map. bmp**,**hover. bmp**。**

第三步 :创建 Skin Definition File**。**

Skin Definition File **是一个基于** XML **的文本文件,我们知道,**XML **的一个重要特性就是结构化,制作之前有必要先了解** Skin Definition File **的结构。**

一个 Skin Definition File **文件必须以** < Theme > . . . < /Theme > **开头和结尾,同时也必须包含一个** VIEW **元素,在** < VIEW > . . . < /view > **标记之间,用来具体定义界面。**

(1)设置界面背景

打开记事本,输入以下代码:

```
< THEME >
    < VIEW
        clippingColor = "#CCCC00"
        backgroundImage = "background. bmp"
        titleBar = "False" >
    < /VIEW >
< /THEME >
```

这段代码通过设置 backgroundImage **属性定义界面背景。**

(2) 加入按钮

在 < VIEW > < /VIEW > 标记中间插入以下代码：

```
< BUTTONGROUP
                mappingImage = "map. bmp"
                hoverImage = "hover. bmp" >
< /BUTTONGROUP >
```

这段代码定义了鼠标点击响应区域和响应点击事件后显示的图像。其中 mappingImage 属性是必需的，而 hoverImage 则可选。

设置鼠标点击响应事件 BUTTONELEMENT，该事件通过 BUTTONGROUP 元素的子元素 BUTTONELEMENT 来定义，代码如下所示，具体是将以下代码插入 < BUTTONGROUP > < /BUTTONGROUP > 之间。

```
< BUTTONELEMENT
                mappingColor = "#00FF00"
                upToolTip = "Open and play the file 'laure. wma'"
onClick = "JScript：player. URL = 'file：laure. wma' ;" / >
```

这段代码通过设置以下几个属性来控制播放器：

mappingColor：定义响应点击区域，后面的值为该区域的颜色代码，当用户点击该颜色的区域时则发生 upToolTip 和 onClick 定义的事件。

upToolTip：定义鼠标移动到响应区域后显示的提示文本。

onClick：定义鼠标点击响应区域时发生的事件，本文代码中通过设置 Media Player 对象的 URL 属性来告诉播放器（关于该对象可参照 2. 3 节）播放 laure. wma 文件。

同样，使用以下代码加入 Close 按钮，当用户点击#FF0000（红色）区域时，关闭播放器。

```
< BUTTONELEMENT
                mappingColor = "#FF0000"
                upToolTip = "Close"
                onClick = "JScript：view. close() ;" / >
```

完成好的 Skin Define Skin 如以下代码所示：

```
<THEME>
    <VIEW
        clippingColor = "#CCCC00"
        backgroundImage = "background.bmp"
        titleBar = "false"
        >
    <BUTTONGROUP
            mappingImage = "map.bmp"
            hoverImage = "hover.bmp"
            >
        <BUTTONELEMENT
                upToolTip = "Open and play the file 'laure.wma'"
                onClick = "JScript:player.URL = '.hh.hhmedia hhlaure.wma';"
                mappingColor = "#00FF00"
                />
        <BUTTONELEMENT
upToolTip = "Close"
mappingColor = "#FF0000"
        onClick = "JScript:view.close();"
                />
        </BUTTONGROUP>
 </VIEW>
</THEME>
```


把它保存为 skin.wms，然后点击观看，如图 2.22 所示。鼠标移动到播放按钮上，会提示播放字样，点击后会播放音频文件，点击 Close，会关闭播放器。

图 2.22 skin.wms 效果图

在制作完工艺图和界面文件之后，使用 ZIP 程序将所有文件压缩并组合到 ZIP 文件中，并且用扩展名 .wmz 保存它。当最终用户双击 .wmz 文件图标时，Windows Media Player 将打开压缩文件中的界面和其他内容。

在配套光盘 Sample/Sample2_3 目录下，准备了一个更加精美复杂的 Skin 作品，

如图 2.23 所示，有兴趣的读者可以自行研究。

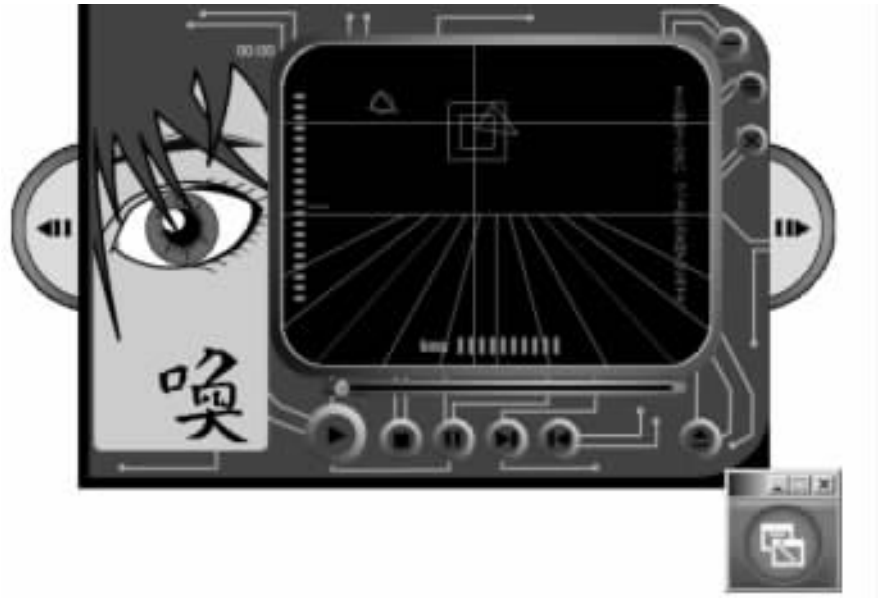


图 2.23　一个复杂的 Skin 实例 Anime

Windows Media Player 中的各个功能区域参见图 2.24，在本例中，将灵活控制这些区域。

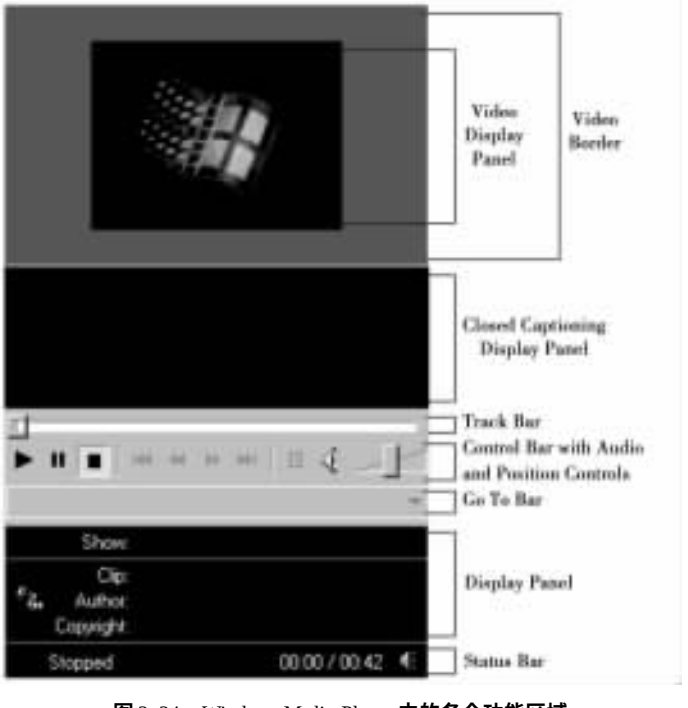


图 2.24　Windows Media Player 中的各个功能区域

①脚本文件 Anime. js 源代码如下：

```
var plIsOpen = false;
var eqIsOpen = false;
var speed = 250;
function Init()
{
    plist. setColumnResizeMode( 0, "Stretches" );
    plist. setColumnResizeMode( 1, "AutoSizeData" );
     if(player. OpenState == osMediaOpen)
      {
          OnPlayStateChange();
     }
}
function OnOpenStateChange()
{
     if(player. OpenState == osMediaOpen)
      {
       Init();
     }
}
function OnPlayStateChange()
{
    switch(player. PlayState)
     {
    case 1:
    EndVideo();
    break;
    case 3:
        if (player. currentMedia. ImageSourceWidth > 0) {
          vAnimation. visible = false;
          vidWindow. visible = true;
           visEffects. visible = false;
           vid. visible = true;
              //      StartVideo();
     }
        break;
    }
```

```
}
function togglePlView() {
    if(plIsOpen) {
        left_button. visible = false;
        plist. visible = false;
        vPlist. moveto(176,25,speed);
        left_button. moveto(330, 105, speed);
    } else {
        lefthandle. visible = false;
        vPlist. moveto(0,25,speed);
        left_button. moveto(0, 105, 300);
    }
    plIsOpen = !plIsOpen;
}
function vPListOnEndMove() {
    if(plIsOpen) {
        left_button. visible = true;
        plist. visible = true;
    } else {
        lefthandle. visible = true;
    }
}
function StartVideo()
 {
    vAnimation. visible = false;
    vidWindow. visible = true;
     visEffects. visible = false;
     vid. visible = true;
}
function EndVideo()
 {
    vidWindow. visible = false;
    vAnimation. visible = true;
     visEffects. visible = true;
     vid. visible = false;
}
function togglePlVis(bool) {
    if(plIsOpen) {
        plist. visible = true;
```

```
    }
}
eqToolTips = new Array("32 Hz","63 Hz","125 Hz","250 Hz","500 Hz","1
kHz","2 kHz","4 kHz","8 kHz","16 kHz");
function toggleEqView() {
    if(eqIsOpen) {
        vEq.moveto(488,25,speed);
        right_button.moveto(700, 107, speed);
        for(i = 1;i <= 10;i ++) {
            eval("eq" + i + ".tooltip = """);
        }
    } else {
        righthandle.visible = false;
        right_button.moveto(950, 107, 125);
        vEq.moveto(700,25,speed);
        vEq.visible = true;
        for(i = 1;i <= 10;i ++) {
            eval("eq" + i + ".tooltip =‴ + eqToolTips[i - 1] + ‴;");
        }
    }
    eqIsOpen = !eqIsOpen;
}
function vEqOnEndMove() {
    if(eqIsOpen) {
        right_button.visible = true;
    } else {
        righthandle.visible = true;
    }
}
function resetEq() {
    eq.gainLevel1 = 0;
    eq.gainLevel2 = 0;
    eq.gainLevel3 = 0;
    eq.gainLevel4 = 0;
    eq.gainLevel5 = 0;
    eq.gainLevel6 = 0;
    eq.gainLevel7 = 0;
    eq.gainLevel8 = 0;
    eq.gainLevel9 = 0;
```

```
    eq. gainLevel10 = 0;
}
function OnTimerTick()
 {
    var position = player. Controls. currentPosition;
    var mm = ((position < 600) ?"0" :"")
        + Math. floor(position / 60);
    var ss = ((position%60) < 10 ?"0" :"")
        + Math. floor(position % 60);
    metadataTime. value = mm + ":" + ss;
}
```

②**外观定义文件** Anime. wms **源代码如下：**

```
<theme id = "anime" author = "Saltmine, LLC" copyright = "Microsoft Corpora-
tion. All Rights Reserved. " >
    <view id = "animeview"
    backgroundColor = "none"
    width = "1005" height = "600"
    titleBar = "false"
    resizable = "false"
    timerInterval = "333"
    titleBar = "false"
    resizable = "false"
    scriptFile = "anime. js;res://wmploc. dll/RT_TEXT/#132;"
        onTimer = "OnTimerTick();"
        onLoad = "Init();" >
        <player
            PlayState_onchange = "OnPlayStateChange();"
                OpenState_onchange = "OnOpenStateChange();" >
            </player >
        <text id = "metadataArtist" zIndex = "3"
                    left = "400" top = "40"
                foregroundColor = "#FFFFFF"
                    fontSize = "8" fontStyle = "bold"
                    >
                </text >
                <text id = "metadataTitle" zIndex = "3"
```

```
                    left = "400" top = "255"
                    foregroundColor = "#FFFFFF"
                fontSize = "8"
                    value = "wmpprop:player.currentmedia.name" >
            </text>
        <subview id = "vPlist" zIndex = "-2" left = "176" top = "25" backgroundImage = "background_playlist.bmp" backgroundColor = "none"
            titleBar = "false" resizable = "false" transparencyColor = "#00FF00" onendmove = "vPListOnEndMove();" >
            <playlist id = "plist" zIndex = "0"
                left = "94" top = "30"
                width = "115" height = "225"
                backgroundcolor = "black"
                foregroundcolor = "white"
                columnsVisible = "false"
                columns = "name = Name"
                editbuttonsvisible = "false"
                movebuttonsvisible = "false"
                toolbarVisible = "true"
                playlistItemsVisible = "true"
                toolbarMargin = "0"
                visible = "false" >
                </playlist>
        </subview>
        <subview id = "anime" zIndex = "2" left = "219" top = "0" backgroundImage = "background_main.bmp" transparencyColor = "#00FF00" >
            <text id = "metadataTime" zIndex = "3"
                    left = "120" top = "30"
                foregroundColor = "#FFFFFF"
                fontSize = "8" >
            </text>
            <buttongroup zIndex = "2" left = "169" top = "302" mappingImage = "playcontrols_map.bmp" image = "playcontrols_up.bmp" hoverImage = "playcontrols_hover.bmp" downImage = "playcontrols_down.bmp" >
                <buttonelement mappingColor = "#00FF00" upToolTip = "Play" cursor = "hand" onClick = "player.controls.play();" ></buttonelement>
                <buttonelement mappingColor = "#00CC00" upToolTip = "Stop" cursor = "hand" onClick = "player.controls.stop();" ></buttonelement>
```

```
                    <buttonelement mappingColor = "#009900" upToolTip = "Pause"
cursor = "hand" onclick = "player.controls.pause();" ></buttonelement>
                    <buttonelement mappingColor = "#006600" upToolTip = "Previous"
cursor = "hand" onClick = "player.controls.previous()" ></buttonelement>
                    <buttonelement mappingColor = "#003300" upToolTip = "
Next" cursor = "hand" onClick = "player.controls.next()" ></buttonelement>
                    <buttonelement mappingColor = "#CC0000" upToolTip = "E-
ject" cursor = "hand" onClick = "player.cdromcollection.item(0).eject();" >
</buttonelement>
                </buttongroup>
                <buttongroup zIndex = "2" left = "477" top = "19" mappingImage =
"windowcontrols_map.bmp" image = "windowcontrols_up.bmp" hoverImage = "
windowcontrols_hover.bmp" >
                    <buttonelement mappingColor = "#00FF00" upToolTip = "Minimize"
cursor = "hand" onClick = "animeview.minimize();" ></buttonelement>
                    <buttonelement mappingColor = "#00CC00" upToolTip = "Re-
turn to Full Mode" cursor = "hand" onClick = "animeview.returnToMediaCenter
();" ></buttonelement>
                    <buttonelement mappingColor = "#009900" upToolTip = "
Close" cursor = "hand" onClick = "animeview.close();" ></buttonelement>
                </buttongroup>
                <slider id = "seek" borderSize = "15" zIndex = "2" left = "184" top =
"285" toolTip = "Seek" backgroundImage = "seek_background.bmp" thumbImage =
"seek_thumb.bmp" min = "0" max = "wmpprop:player.currentMedia.duration"
                    onmouseup = "player.controls.currentPosition = seek.value;"
                    value = "wmpprop:player.controls.currentposition"
                    onDragEnd = "player.controls.currentpostion = value;"
                    useForegroundProgress = "true"
                    foregroundProgress = "wmpprop:player.network.downloadProgress"
                    enabled = "wmpenabled:player.Controls.currentPosition"
                    transparencyColor = "#00FF00" ></slider>
                <effects id = "visEffects"
                 zIndex = "-1"
                            left = "150" top = "32" width = "320" height = "240"
                            visible = "false"
                            currentEffectType = "spikes"
                            currentPreset = "1"
                            onClick = "visEffects.previous();" >
```

```
                    </effects>
            </subview>
                <!-- declare the video window -->
                <subview id="vidWindow" zIndex="1"
                    left="373" top="34"
                    backgroundImage="background_video.bmp"
                backgroundColor = "#000000"
                    transparencyColor="#00FF00"
                    visible="true">
                    <video id="vid" zIndex="1" left="0" top="0" width=
"320" height="240"
                     OnVideoStart="StartVideo();" OnVideoEnd="EndVideo();"
                    visible="false" enabled="true" windowless="true" shrinkT-
oFit="true">
                </video>
            </subview>
            <!-- end the video window -->
            <subview id="left_button" zIndex="0" left="169" top="105" back-
groundImage="drawer_pl_in_left.bmp" visible="true" backgroundColor="none"
                titleBar="false" resizable="false" transparencyColor="#00FF00">
            <button id="plHandle" zIndex="2" left="0" top="0" cursor=
"hand" image="drawer_pl_in_left.bmp" onclick="togglePlView();" transparency-
Color="#00FF00"></button>
            </subview>
            <subview id="right_button" zIndex="0" left="737" top="105" back-
groundImage="drawer_pl_in_right.bmp" visible="true" backgroundColor="none"
                titleBar="false" resizable="false" transparencyColor="#00FF00">
            <button id="closeeq" zIndex="2" left="0" top="0" cursor="hand"
image="drawer_pl_in_right.bmp" onclick="toggleEqView();" transparencyColor=
"#00FF00"></button>
            </subview>
            <subview id="lefthandle" zIndex="0" left="169" top="105" back-
groundImage="drawer_pl_out_left.bmp" visible="true" backgroundColor="none"
titleBar="false" resizable="false" transparencyColor="#00FF00">
                        <button id="closepl" zIndex="0" left="0" top="0"
image="drawer_pl_out_left.bmp" onclick="togglePlView();" cursor="hand"
transparencyColor="#00FF00"></button>
            </subview>
```

```
        <subview id = "righthandle" zIndex = "0" left = "737" top = "105"
backgroundImage = "drawer_pl_out_right.bmp" visible = "true" backgroundColor
= "" titleBar = "false" resizable = "false" transparencyColor = "#00FF00" >
                        <button id = "eqHandle" zIndex = "2" left = "0" top
= "0" image = "drawer_pl_out_right.bmp" onclick = "toggleEqView();" cursor =
"hand" transparencyColor = "#00FF00" > </button >
        </subview >
        <subview id = "vAnimation" zIndex = "1" left = "358" top = "30" back-
groundImage = "anime.gif" visible = "true" backgroundColor = "none"
            titleBar = "false" resizable = "false" transparencyColor = "#00FF00" >
        </subview >
        <subview id = "vEq" zIndex = "-3" left = "488" top = "25" width =
"314" height = "284" backgroundImage = "background_eq.bmp" transparency-
Color = "#FF0000" onendmove = "vEqOnEndMove();" >
            <equalizerSettings id = "eq" enable = "true" > </equalizerSettings >
            <slider id = "vol" zIndex = "2" borderSize = "13" left = "92" top =
"66" backgroundImage = "vol_background.bmp" thumbImage = "vol_thumb.bmp"
transparencyColor = "#00FF00" min = "0" max = "100"
                value = "wmpprop:player.settings.volume"
value_onchange = "player.settings.volume = value;player.settings.mute = false;"
direction = "vertical" > </slider >
            <slider id = "balance" zIndex = "0" borderSize = "13" left = "52"
top = "213" backgroundImage = "eqslide_background.bmp" thumbImage = "eqslide
_thumb.bmp" transparencyColor = "#00FF00" min = "-100" max = "100"
                value = "player.settings.balance"
value_onchange = "player.settings.balance = value" > </slider >
            <slider id = "eq1" zIndex = "0" borderSize = "13" left = "145" top =
"63" toolTip = "" backgroundImage = "eqslide_background.bmp" transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel1"
value_onchange = "eq.gainLevel1 = value;" direction = "horizontal" > </slider >
            <slider id = "eq2" zIndex = "0" borderSize = "13" left = "145" top =
"78" toolTip = "" backgroundImage = "eqslide_background.bmp" transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel2"
value_onchange = "eq.gainLevel2 = value;" direction = "horizontal" > </slider >
            <slider id = "eq3" zIndex = "0" borderSize = "13" left = "145" top =
"94" toolTip = "" backgroundImage = "eqslide_background.bmp" transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
```

```
                value = "wmpprop:eq.gainLevel3"
value_onchange = "eq.gainLevel3 = value;" direction = "horizontal" > </slider >
            <slider id = "eq4" zIndex = "0" borderSize = "13" left = "145" top =
"110" toolTip = "" backgroundImage = "eqslide_background.bmp"  transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel4"
value_onchange = "eq.gainLevel4 = value;" direction = "horizontal" > </slider >
            <slider id = "eq5" zIndex = "0" borderSize = "13" left = "145" top =
"126" toolTip = "" backgroundImage = "eqslide_background.bmp"  transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel5"
value_onchange = "eq.gainLevel5 = value;" direction = "horizontal" > </slider >
            <slider id = "eq6" zIndex = "0" borderSize = "13" left = "145" top =
"141" toolTip = "" backgroundImage = "eqslide_background.bmp"  transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel6"
value_onchange = "eq.gainLevel6 = value;" direction = "horizontal" > </slider >
            <slider id = "eq7" zIndex = "0" borderSize = "13" left = "145" top =
"158" toolTip = "" backgroundImage = "eqslide_background.bmp"  transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel7"
value_onchange = "eq.gainLevel7 = value;" direction = "horizontal" > </slider >
            <slider id = "eq8" zIndex = "0" borderSize = "13" left = "145" top =
"172" toolTip = "" backgroundImage = "eqslide_background.bmp"  transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel8"
value_onchange = "eq.gainLevel8 = value;" direction = "horizontal" > </slider >
            <slider id = "eq9" zIndex = "0" borderSize = "13" left = "145" top =
"188" toolTip = "" backgroundImage = "eqslide_background.bmp"  transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel9"
value_onchange = "eq.gainLevel9 = value;" direction = "horizontal" > </slider >
            <slider id = "eq10" zIndex = "0" borderSize = "13" left = "145" top =
"200" toolTip = "" backgroundImage = "eqslide_background.bmp"  transparencyColor
= "#00FF00" thumbImage = "eqslide_thumb.bmp" min = "-20" max = "20"
                value = "wmpprop:eq.gainLevel10"
value_onchange = "eq.gainLevel10 = value;" direction = "horizontal" > </slider >
        </subview >
    </view >
    </theme >
```

脚本文件(.js)定义的函数、变量被外观定义文件调用。

2.5　实例二：定制自己的媒体播放器

时下视频点播已经成了新一代宽带网的魅力所在，各种影视点播网站也如雨后春笋般地“冒”了出来。在众多的影视网站中，越来越多的网站使用了 Web 页内嵌播放器的方式，只要一点击网页上的链接就会自动弹出一个固定大小的很漂亮的播放器，还加入了很多诸如快进、音量方面的控制按钮，很是美观。其实，制作这样的 Web 效果非常简单，只要有一点网页设计和 ASP 的知识，通过本节的学习，同样也可以设计出自己的 Web 播放器。

刚才通过介绍 Windows Media Player 控件，已经基本了解了嵌入控件的方法以及一些控件的基本属性和方法，现在就用一个实例来扩展巩固一下刚才的成果。

首先，在开始之前，有必要先罗列一下将要使用的工具。首先要有 Dreamwaver（用来添加行为最方便）和 FontPage（因为要插入 Media Player 的 ActiveX 控件，用 FontPage 要方便得多）；此外要有一个文本编辑器，最好是 UltraEdit，要不然记事本也行，要是想对 Web 播放器进行一番美容的话，那么还需要一些图像处理软件；同时需要具备一些数据库和 ASP 的基本常识。

2.5.1　制作一个播放器

用 Dreamwaver 新建一个页面；输入一行文字，如 media test，点击行为窗口上的加号，添加一个打开浏览器窗口的行为，然后在接下来的对话框中设置弹出网页的属性，比如说窗口的大小，是否有工具栏之类，在文字或图片的超链接选项上填入#31，完成后的代码如下：

```
<html>
<head>
<title>无标题文档</title>
<meta http-equiv="Content-Type" content="text/html;charset=gb2312">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<script language="JavaScript">
<!--
function MM_openBrWindow(theURL,winName,features) { //v2.0
```

```
    window. open(theURL,winName,features);
}
// - ->
< /script >
< p >
< a href = " #31"
onclick = " MM_openBrWindow(' myplayer. htm' ,' ,' width = 4000,height = 400')"
>
  meida test  < /a > < /p >
< /body >
< /html >
```

这个页面的作用就是当用户点击文字时,会通过 MM_openBrWindow **过程来打开** myplayer. htm **页面,大小定为** 4 000 × 400**,在链接属性上添加**#31 **是为了使鼠标选中文字区域时提醒用户这里有个链接,如图** 2. 25 **所示。**

图 2. 25 testfilm. htm

打开 FontPage **新建一个网页:点击插入→高级→**ActiveX **控件,在弹出的控件栏中选择** Windows Media Player**,确定后即将** Media Player **插入到网页中。预览代码如下:**

```
< html >
< head >
< meta http - equiv = " Content - Type"  content = " text/html;charset = gb2312" >
< meta name = " GENERATOR"  content = " Microsoft FrontPage 4. 0" >
< meta name = " ProgId"  content = " FrontPage. Editor. Document" >
< title > New Page 1 < /title >
< /head >
< body >
```

```
<object Classid = "clsid:6BF52A52 - 394A - 11D3 - B153 - 00C04F79FAA6"
id = "WindowsMediaPlayer1"  width = "240"  height = "245" >
    <param name = "URL"  value >
    <param name = "rate"  value = "1" >
    <param name = "balance"  value = "0" >
    <param name = "currentPosition"  value = "0" >
    <param name = "defaultFrame"  value >
    <param name = "playCount"  value = "1" >
    <param name = "autoStart"  value = " - 1" >
    <param name = "currentMarker"  value = "0" >
    <param name = "invokeURLs"  value = " - 1" >
    <param name = "baseURL"  value >
    <param name = "volume"  value = "50" >
    <param name = "mute"  value = "0" >
    <param name = "uiMode"  value = "full" >
    <param name = "stretchToFit"  value = "0" >
    <param name = "windowlessVideo"  value = "0" >
    <param name = "enabled"  value = " - 1" >
    <param name = "enableContextMenu"  value = " - 1" >
    <param name = "fullScreen"  value = "0" >
    <param name = "SAMIStyle"  value >
    <param name = "SAMILang"  value >
    <param name = "SAMIFilename"  value >
    <param name = "captioningID"  value >
</object >
</body >
</html >
```


新版本 Player 9 **的控件** ID **号变成了** 6BF52A52-394A-11D3-B153-00C04F79FAA,**同时参数也简洁了不少。**

2.5.2 播放器控制

用鼠标添加 3 个按钮在控件的下方，一个命名为 BtnPlay，值设为 Play；一个为 BtnPause，值为 Pause；一个则命名为 BtnStop，值为 Stop。下面将用这 3 个按钮来控制媒体文件的播放。

设置 AutoStart 属性为 0。因为要用按钮来控制文件的播放，自然不希望页面一打开，文件就自动播放。现在可以对这个播放器来设置一些参数，需要注意的是，以前版本控件的 FileName 参数现在被 URL 参数所代替。点击 html，编辑刚做好的页面，在按钮上添加相应点击事件，代码如下：

```
<INPUT TYPE = "BUTTON"
NAME = "BtnPlay" VALUE = "Play" OnClick = "MediaStart ()" >
<INPUT TYPE = "BUTTON"
NAME = "BtnStop" VALUE = "Stop" OnClick = "MediaStop ()" >
<INPUT TYPE = "BUTTON"
NAME = "BtnPause" VALUE = "Pause" OnClick = "MediaPause()" >
```

将以下脚本插入页面：

```
<SCRIPT >
function MediaStart ()
{
WindowsMediaPlayer1. URL = "laure. wma" ;
}
function MediaStop ()
{
WindowsMediaPlayer1. controls. stop() ;
}
function MediaPause ()
{
WindowsMediaPlayer1. controls. pause() ;
}
</SCRIPT >
```


分别保存两个文件为 testfilm. htm 和 myplayer. htm，退出 FontPage。

把两个网页和要播放的媒体文件都保存到一个目录中，执行 testfilm. htm，点击链接即可观看媒体文件。而它已经具有了那些点播网站的基本特征，接下来可以对它进行美化，也可以扩展它的功能，比如可以通过数据库来获取媒体文件的 URL，再

交给播放器播放。这样做的好处是便于管理节目，也增加了安全性，运行后效果如图 2.26 所示。

图 2.26　myplayer.htm

媒体文件要和页面在同一目录，本例代码位于配套光盘 Sample/Sample2_2 中。

测试发布自己的播放器：用 IE 浏览器打开 testfilm.htm，点击链接文字，会弹出播放窗口，点击播放按钮，现在已经可以通过 Web 服务把你的页面发布到网上让用户观看了。

以上示范的只是简单地向网页中插入了一个 Windows Media Player 控件，并且简单地加入了一些脚本进行控制，它已经具有了点播网站的基本特征。还可以对它进行扩展，如加入音量控制，播放进度，对界面进行美化，同时可以从数据库中读取节目地址以及相关信息等等，这样 Web 播放器就和专业视频网站相差无几了。

2.6 Windows Media Player 错误信息

最常见的代码是服务器错误消息，这些消息在 5xx 范围内，代表服务器发出已出错或无法处理请求的信息。这些代码如表 2.5 所示。

表 2.5 Windows Media Player 常见错误信息

代 码	名 称	含 义
500	内部服务器错误	服务器遇到阻止它完成请求的意外情况
501	未实现	服务器不支持完成请求所需要的功能。这是当服务器无法识别请求方法且无法支持它获取任何资源时的响应
502	网关已坏	用作网关或代理的服务器，在试图完成请求时从所访问的上游服务器接收到无效响应
503	服务不可用	由于临时超载或服务器维护的原因，服务器目前无法处理请求。意思为这是临时情况，经过一段延迟后会缓和。如果延迟长度已知，则它可能包含在之后重试（Retry-After）头文件中。如果未给定之后重试（Retry-After），则客户应将本响应当作 500 个响应进行处理
504	网关超时	用作网关或代理的服务器，在试图完成请求时不能从所访问的上游服务器及时接到响应
505	不支持 HTTP 版本	服务器不支持或拒绝支持请求消息中所用的 HTTP 协议版本。服务器表明它无法或不愿意使用与客户相同的主版本来完成请求，而不是使用该错误消息。响应应包含信息来说明不支持该版本的原因以及该服务器所支持的其他协议

第3章

提高篇：用Windows Media Encoder SDK制作自己的媒体压缩器

3.1 Windows Media Encoder 概述

Microsoft Windows Media Encoder 是一种功能强大、易于使用的制作工具，内容开发人员可以用来将现场和预先录制的音频、视频及计算机屏幕图像转换为 Windows Media 格式，以便现场提供或点播时使用，Encoder 界面如图 3.1 所示。

图 3.1 Windows Media Encoder 7 界面

Windows Media Encoder 7 具有很多增强功能，例如易用性、最佳质量的编码、增强的编程能力和管理功能。此外，Windows Media Encoder 7 还支持多种捕捉卡。

①易用性：提供了新的图形化用户界面和向导，便于配置和制作外观效果极佳的基于 Windows Media 的内容，可以通过 Internet 现场提供或点播时提供。对源切换的支持则可以更好地控制现场事件内容的提供。

②更简单的会话向导：使用简单快捷的会话向导步骤，指导用户设置编码会话、目标带宽和质量选项。

③支持多个源：允许将每个会话的多个输入源定义为“源组”。源组可以是设备、磁盘文件或这二者的组合。对于特殊的流式处理应用程序，用户可以按源组来维护视频设置和设备属性。

④即时源切换：通过允许用户在现场和预先录制的输入源之间切换，可以更好地

控制现场事件内容的制作和交付。

⑤增强的配置文件创建和管理功能：允许创建自定义配置文件设置，并且这些设置可以很容易地复制、保存以及在各编码工作站之间共享。用户可以使用配置文件管理器来管理系统配置文件和自定义配置文件。

⑥增强的可视化反馈：可以根据编码图像的大小、数据传输速率以及对编码和统计信息的实时控制，更好地对编码过程进行监控。

⑦最佳质量的编码：新增了一些编码功能，例如支持取消隔行扫描、翻转电视电影和屏幕捕获，从而提高输出的质量。

⑧增强的输出质量：包括支持新的 Windows Media 7 格式编码解码器，从而提高高速运动内容的输出质量（来自隔行扫描源的 320 × 240 × 60 帧/s，使用独有的处理功能），并提高了 640 × 480 × 30 帧/s 的输出质量。

⑨允许将屏幕捕获应用到文件或实时广播：包括一个简单的过程，用 Windows Media 屏幕捕获编码/解码器创建屏幕捕获和培训演示。

⑩支持对取消隔行扫描进行处理：通过减少顺次扫描行显示器（计算机显示器）的闪烁，从而提高源自 TV 的内容的视频播放质量。

⑪允许直接编码：对于调用 Windows Media 文件的源，允许文件不经任何重新压缩即直接传递到 Windows Media Encoder。

⑫支持翻转电视电影：提高源自电影的内容的播放质量。源于 24 帧/s 电影的视频内容使用以 30 帧/s 传送的额外的帧填充。Encoder 智能地提取原来的 24 帧/s，并对其进行编码以 24 帧/s 输出，从而消除人工填充的帧，因此改进了带宽较低时的质量。

⑬支持主要的显卡：包括支持新的 Osprey Viewcast 500WM/DVPro，并且支持 Winnov，ATI，Hauppauge 和许多其他厂商生产的显卡。

⑭支持多 GB 的文件以及未经压缩的捕获：直接捕获成 Windows Media 格式，并创建大小超过 30 GB 的存档文件。该捕获功能与对捕获未经压缩的音频和视频的支持相结合，可以提供以前无法使用的捕获和存档功能。

⑮允许基于 Encoder 的时间压缩：允许用户调整音频和视频文件的停顿消除以及播放扩展，这样可以进一步减少播放时间。

⑯增强的编程能力和管理功能：功能增强的 Windows Media Encoder SDK 使 Web 开发人员可以全自动地使用 Encoder 的功能。

⑰可扩展的平台：允许 Web 开发人员和管理员管理局域网（LAN）中其他客户机上安装的 Encoder。开发人员可以通过对 Encoder 的应用程序可编程接口（API）的完全访问来使用分布式组件对象模型（DCOM），也可以使用活动服务器页（ASP）访问各种功能。

⑱增强功能的 Windows Media Encoder SDK：通过 Windows Media Encoder SDK，允许完全通过编程控制 Encoder 的功能和自动执行 Encoder 的功能。允许控制和自动执行 Encoder 的用户界面（UI）、引擎控制、输入和输出以及源的创建和切换。

⑲增加了通过 UNICAST 的分发功能：最多允许从 Encoder 同时直接分发 50 个 Windows Media 流。

3.1.1 Windows Media Encoder 编码器的重要概念

(1) Windows Media 服务器和 Web 服务器之间的区别

可以从运行 Windows Media Services 的服务器,或是从 Web 服务器将基于 Windows Media 的内容传输到播放器,例如 Windows Media Player。服务器和播放器可以在 Internet 上使用,或是在 Intranet 上使用,并且可以用防火墙隔开。虽然 Windows Media 服务器是专门为传输基于 Windows Media 的内容所设计的,但标准的 Web 服务器却不是这样。如果决定使用 Web 服务器,需要认识到在内容分发方式方面的区别,而这将影响回放的质量。

在 Web 服务器和 Windows Media 服务器之间发送数据的方法是有区别的。Web 服务器被设计为尽快、尽可能多地发送数据。这种方法对于发送包含静态图像、文本和 Web 页面脚本的数据包是首选的方法,但却不是发送包含流媒体的数据包的最好方法。流媒体应该被实时传送,而不是以大量的字符组来传送,播放器应该在播放它们之前收到数据包。

Windows Media 服务器根据向某个播放器发送流时收到的反馈信息来衡量数据包的发送。当播放器以这种方式收到数据包时,图像将更平滑。因为带宽的使用受到控制,所以更多的用户可以同时连接到站点并持续地接收流。

Web 服务器不支持多比特率视频。当文件从 Web 服务器传输时,不会监视传送质量,也不能调整比特率。Web 服务器不能使用首选的传送协议,即用户数据报协议(UDP),所以当播放器缓存数据时,流的传送更有可能被周期性地既无音频也无视频地部分中断。实况流和多播同样不能和 Web 服务器一起使用。

(2) 智能流和多比特率视频

智能流是 Microsoft Windows Media Technologies 中的一组功能,可以自动检测网络状况并将视频流的属性调整到最佳品质。因为 Windows Media Technologies 是客户/服务器系统,因此客户和服务器彼此通过通讯来建立实际的网络吞吐量,提示自动进行一系列的调整以使流的质量达到最佳。使用智能流,用户将收到与其连接速度相符的连续的内容流。

要充分利用智能流的优点,则必须使用多比特率编码的内容。要以多比特率编码内容,应创建单独的 Windows Media Format 流或包含以不同的比特率编码的多个流(音频、视频和脚本)的文件。当多比特率 Windows Media 文件或实况流由播放器收到以后,只播放其中的一个视频流,即最适合当前带宽状况的流。选择适当的流的过程由 Windows Media 服务器和 Windows Media Player 处理,对用户完全是不可见的。

“Windows Media 编码器”提供了预定义的多比特率配置文件,使得编码多比特率内容的过程非常简单。也可以使用‘配置文件管理器”创建自己的多比特率配置文件。

Web 服务器不支持多比特率视频。

(3) 源和源组

使用“Windows Media 编码器”，可以对来自 3 种不同源的内容进行编码：音频、视频或脚本。可以从安装在计算机中的卡或从文件捕获音频和视频。并且可以在编码会话过程中直接在编码器的主窗口中输入脚本。

音频、视频和脚本源的组合就是源组。源组表示播放器中显示的流。虽然在编码时一次只有一个源组流，但可以在源组之间切换来传输不同的内容。例如在广播实况事件之前（如公司会议），可以为主要事件以及欢迎、休息和再见视频设置源组。当广播事件时，应该以欢迎源组开始，在适当的时候切换到主讲人，在休息期间切换到休息源，然后在事件结束时切换到再见源。

源组必须包括至少一个音频源，也可以包括一个视频源和一个脚本源。在编码开始之前或开始之后，可以向编码会话中添加无限制数量的源，但它们必须包含相同的源类型的组合（音频、视频和脚本），只能使用捕获卡一次。例如，准备创建两个源。一个源是实况事件，其中一个演讲人正在介绍一种新的视频；第二个源是视频。在设置源时，需要为每个源使用单独的卡。

脚本是 URL 或者是字幕。URL 是到 Web 页的路径，字幕是文本字符串。当播放器收到脚本时，将把该脚本传送到能运行此命令的应用程序。URL 被传送到 Web 浏览器时，字幕将显示在播放器上。当 URL 被插入到流中时，将打开默认的浏览器，并将所请求的 URL 加载到浏览器中。如果已经在浏览器中嵌入了播放器，则所请求的 URL 将替换播放器，而将无法查看其他的流。可以通过在同一个浏览器实例中单独的帧中显示所请求的 URL，或是在屏幕上打开浏览器的另一个实例将脚本传送出去。

(4) 配置文件

配置文件是与被编码内容的类型、听众和预期分发的内容（文件或广播）相匹配的属性的集合。配置文件中保存的属性包括：音频和视频质量、预期听众的连接速度、可用带宽和相应的编码解码器。配置文件的使用简化了设置编码会话的过程。

编码时可以使用以下 3 种配置文件：

①系统配置文件：这是和编码器一起安装的配置文件，无法进行更改或删除。许多系统配置文件都会与编码器一起被安装，包括多比特率配置文件，它基于最通用的编码方案。通过使用“配置文件管理器”，可以复制系统配置文件，然后根据需要对它进行编辑，以此创建自定义配置文件。

②自定义配置文件：这是使用“配置文件管理器”创建的配置文件。可以编辑和删除自定义配置文件。它们将被保存为扩展名为 .prx 的文件，位于 hProgram Files h Windows Media Components hEncoder hProfiles 文件夹中，如图 3.2 所示。可以通过将 PRX 文件复制到运行“Windows Media 编码器”的多台计算机，实现在这些计算机上重新使用自定义配置文件。

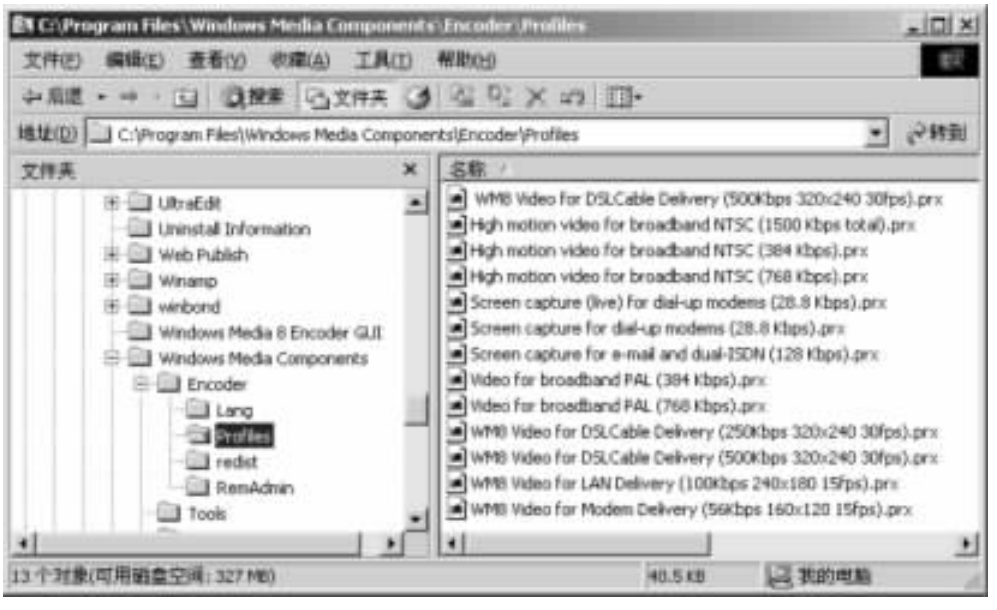


图 3.2 常见的 Encoder 配置文件

③会话配置文件：当创建编码配置时，会话配置文件将自动作为“Windows Media 编码器”配置文件（扩展名为 .wme）的一部分来保存。它只能用于当前会话，并在当前计算机上没有 WME 文件指定的配置文件时使用。例如，当自定义配置文件被删除或在另一台运行“Windows Media 编码器”的不包含自定义配置文件的计算机上使用 WME 时，会话配置文件没有列在“配置文件管理器”、“新建会话向导”或“新建会话”对话框中，它们被显示在“监视面板”上，并在“会话属性”对话框中显示为“会话”。用户无法编辑或查看会话配置文件或用它们来创建另一个会话。

如果在保存编码配置之后更改了配置文件，然后将配置复制到另一台计算机，则会话配置文件将不包含更改。要确保自定义配置文件和会话配置文件保持同步，可将自定义配置文件复制到每一台计划用于编码的计算机中，或者在将其移到另一台计算机上之前，重新保存 .wme 文件以包含对配置文件的更改。

“配置文件管理器”是用于创建、编辑和删除配置文件的工具。可以从“工具”菜单或“Windows Media 编码器”中的“新建会话向导”中找到这个工具。

(5) 支持的格式

Windows Media Encoder **支持的格式如表** 3.1 **所示：**

表 3.1　Windows Media Encoder **支持的格式**

格　式	文件扩展名
Windows Media	.wma，.wmv，.asf
Windows	.avi，.wav
数字影视压缩标准（MPEG**）**	.mpg，.mp3
图片	.bmp

3.1.2　Windows Media Encoder **的面板构造**

Windows Media Encoder 7 **是由多个面板构成，如图** 3.3 **所示。**

图 3.3　Windows Media Encoder 7 **的面板构成**

了解 Windows Media Encoder 7 **的面板构成有助于学习** Windows Media Encoder SDK。

3.1.3 Windows Media Encoder SDK 编码的步骤

Windows Media Encoder SDK 是针对 Windows Media Encoder 的开发包。其实，Windows Media Encoder 中的大部分功能就是使用这个开发包编写的，Windows Media Encoder SDK 将编码模块以组件的形式向外提供。通过 SDK 编码的步骤如下：

第 1 步：生成 Windows Media Encoder 对象（VB），得到 IWMEncoder 接口指针（VC）。

有以下两个途径：一是如果不希望有界面，可直接创建 WMEncoder 对象；二是如果要界面，则创建 WMEncoderApp 对象，通过它得到 WMEncoder 对象。因为 WMEncoderApp 是进程外（out-of-process）对象，而后者是进程内对象。

第 2 步：创建一个源组（Source Group）来放置多媒体流。

源组保存被编码的同步多媒体流。一个源组必须包括一个音频（Audio）流，可以包括视频和脚本（Script）。可以创建多个源组，用 IWMEncSourceGroupCollection 接口来管理。IWMEncSourceGroup 接口管理一个特定源组，IWMEncSource 接口管理特定源。

第 3 步：从文件或设备中捕获多媒体流。

IWMEncSource 接口从文件中载入流或从设备中捕获流，用 IWMEncSourceGroup 接口将流加入到源组。

第 4 步：从编码会话（Encode Session）中选一个配置文件（Profile）。

配置文件指定编解码器（codec），确定比特率等。一个编码会话的多个源组只能指定一个配置文件。配置文件包括不能编辑删除的系统配置文件和自定义配置文件，后者可以用配置文件管理器创建和编辑。但程序化地创建和编辑配置文件只能用 Windows Media Format SDK 进行。或者用 IWMEncProfileManager 接口创建配置文件管理器对话框来创建和管理配置文件。

第 5 步：确定输出选项。

可以归档或者广播。用 IWMEncBroadcast 接口选地址和端口广播，用 IWMEncFile 接口来指定一个文件存储编码的内容。

第 6 步：加入可选描述信息。

IWMEncAttributes 接口用来指定属性，IWMEncDisplayInfo 接口用来指定内容的一般描述信息，MSPropShell 用来显示预定义会话属性界面。这些都不是必须的。

第 7 步：运行 Encoder。

用 IWMEncoder 接口开始和停止编码。

3.2　编程创作一：简单定制一个自己的 Encoder

3.2.1　Encoder 对象简介

本节所指的“自己的 Encoder”主要是指用户有控制 Encoder 对象的能力，可以制作出与 Microsoft Windows Media Encoder 7 一样好的工具软件。因为与我们将要编制的应用程序一样，Microsoft Windows Media Encoder 7 也是以操作 Encoder 对象为基础的。

Encoder 对象如表 3.2 所示。

表 3.2　Encoder 对象属性

属性名称	权限	注　释
AllowAudio	只读	判断音频文件是否可以被编码，返回值为 Boolean 型变量
AllowScripts	只读	判断脚本命令是否可以被发出，返回值为 Boolean 型变量
AllowVideo	只读	判断视频文件是否可以被编码，返回值为 Boolean 型变量
AudioFormatTag	只读	读取音频文件的编码格式，返回值为 Long 型变量
AudioSource	只读	读取音频源，返回值为 BSTR 型变量
Bandwidth	只读	读取带宽值，返回值为 Long 型变量
BitsPerPixel	只读	读取当前文件颜色值（16 位色），返回值为 Short 型变量
DelayBuffer	只读	读取缓存区域大小，返回值为 Short 型变量
Description	只读	读取当前 Encoder 任务注释，返回值为 Short 型变量
FramesPerSecond	只读	读取当前影像文件每秒帧数，返回值为 Short 型变量
ImageHeight	只读	读取当前影像文件像素高度，返回值为 Short 型变量
ImageWidth	只读	读取当前影像文件像素宽度，返回值为 Short 型变量
InputSourceFile	可读可写	读取或设置源文件，返回或接受 BSTR 型变量
IPPort	只读	查看联机的 IP 端口，返回值为 Short 型变量
IsNetEnabled	只读	判断当前 Encoder 网络状态，返回值为 Boolean 型变量

续表

属性名称	权限	注　释
IsRecording	只读	判断当前 Encoder 是否正在记录文件状态,返回值为 Boolean 型变量
IsStarted	只读	判断当前 Encoder 是否开始,返回值为 Boolean 型变量
NumClients	只读	查看当前的客户端连接数,返回值为 Short 型变量
RecordAutoOverride	可读 可写	读取或设置是否允许覆盖文件,返回或接受 Boolean 型变量
RecordAutoStart	可读 可写	读取或设置是否可以编码,如果为 False,则需要使用 Start 方法开始编码,返回或接受 Boolean 型变量
RecordDuration	可读 可写	读取或设置文件记录时间,返回或接受 Long 型变量
RecordFileName	可读 可写	读取或设置文件名称,返回接受 BSTR 型变量
RecordMaxSize	可读 可写	读取或设置记录文件最大尺寸,返回或接受 Long 型变量
RecordSize	可读 可写	读取或设置记录文件尺寸,返回或接受 Long 型变量
SecondsPerIFrame	可读 可写	返回或接受文件被播放前,必须拥有的帧数,返回或接受 Short 型变量
StreamAlias	可读 可写	返回或设置流名称 Windows Media Services,返回或接受 BSTR 型变量
VideoCodecFOURCC	可读 可写	读取或设置文件编码格式,返回或接受 Long 型变量
VideoInputFOURCC	只读	返回当前视频源,返回 Long 型变量
Windows MediaServer	只读	读取当前 Microsoft Media Server 地址名,返回 BSTR 型变量

Encoder 对象的方法如表 3.3 所示。

表 3.3　Encoder 对象方法表

方　法	注　释
LoadASD	从 ASD 文件中读取 Encoder 的配置
RecordStart	开始录制一个 ASF 文件
RecordStop	停止录制 ASF 文件
SendScript	插入脚本命令至编码文件中
Start	开始一个编码任务
Stop	停止编码任务

Encoder 的真正作用就是将输入的资源编码后输出。其中，输入的资源可以是图片、声音文件、视频源等，输出可以为 Encoder 支持的编码文件。在输出的同时，可以通过 Microsoft Media Service 向外发布。

3.2.2　使用 Encoder 编码的步骤

现在，使用 Encoder SDK 来构建一个拥有基本功能的编码工具，本例见配套光盘/Sample/Sample3_1 目录下。

创建这个基本功能的编码工具，要完成最基本的编码功能，使用 Visual Basic 设计。在工程引用中应加入 Windows Media Encoder，一个 Encoder 编码过程大概分为以下 5 个程序操作步骤。

第 1 步：创建 Encoder 对象。

```
// 定义一个 Encoder 对象：
Dim Encoder As New WMEncoder
```

第 2 步：创建属性页，与 Encoder 对象关联。

在第 2 步开始之前，有必要介绍一下 Microsoft PropShell Control 1.0。这是随 Encoder SDK 一起发行的一个控件，是一个容器，如图 3.4 所示。该控件主要用以装载 Encoder 的各种属性页，本例中，PageShell 被定义为这个容器。

```
Set PpgMain = New WMEncMonMainPage
// 将 Encoder 与显示页容器绑定
PageShell.AddObject Encoder
// 将监视属性页加入显示页容器
PageShell.AddPage PpgMain
```

第 3 步：修改 Encoder 的各个属性。

```
// 创建 Encoder 配置属性页
Dim PpgSources As New WMEncSourcesAltPage
Dim PpgDesc As New WMEncDisplayInfoPage
Dim PpgAttr As New WMEncAttributesPage
Dim PpgEnc As New WMEncProfilePage
Dim PpgOutput As New WMEncOutputPage
// 将 Encoder 与显示页容器绑定
PropPageShell.AddObject EncoderConfig
// 将监视属性页加入显示页容器
```

```
PropPageShell. AddPage PpgSources
PropPageShell. AddPage PpgDesc
PropPageShell. AddPage PpgAttr
PropPageShell. AddPage PpgEnc
PropPageShell. AddPage PpgOutput
// 在用户修改后,通过属性页的属性改变 Encoder 的运行参数
PropPageShell. Apply' 应用
EncoderConfig. PrepareToEncode（True）
```

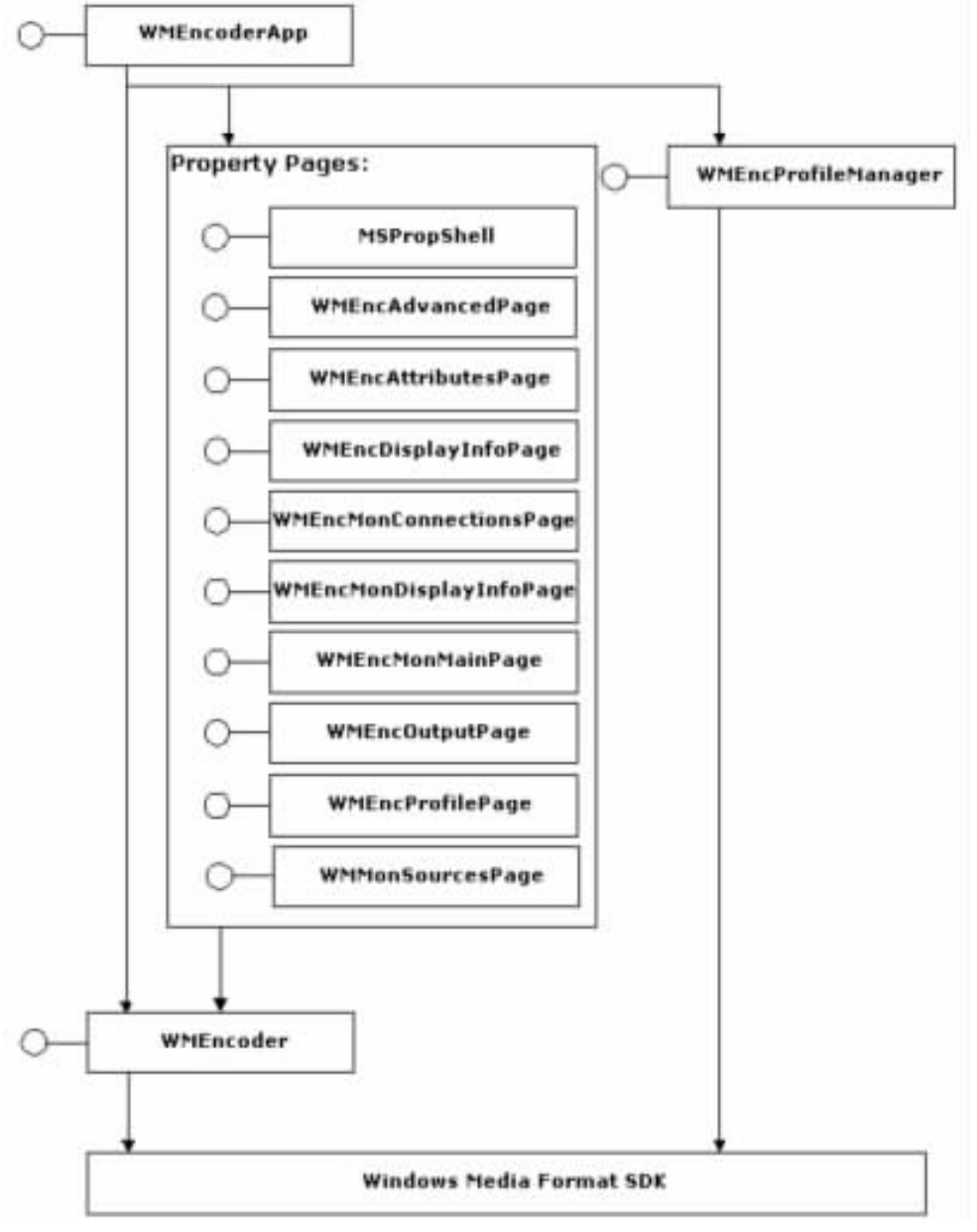


图 3.4　Microsoft PropShell Control 1.0

第 4 步　编码。

获得了相关参数后，就可以进行编码了。调用 Encoder 对象的 Start 方法。

```
Encoder. Start
```

第 5 步　编码结束。

编码结束后，必须保证 Encoder 的状态为结束。

```
// 保证 Encoder 的状态为停止
If（Encoder. RunState < > WMENC_ENCODER_STOPPING）_
And（Encoder. RunState < > WMENC_ENCODER_STOPPED）Then
   Encoder. Stop
End If
// 如果没有停止，那么继续等待到停止
While Encoder. RunState < > WMENC_ENCODER_STOPPED
   DoEvents
Wend
// 释放 Encoder 对象
Set Encoder = Nothing
```

以上 5 步就是 Encoder 的基本功能，在此基础之上进行扩充，就可以完成更多的功能。

很多相关的类说明和属性在 Windows Media Encoder SDK 的帮助文件中并未作相关介绍，可以通过 Visual Basic Object View 来观看，如图 3.5 所示。

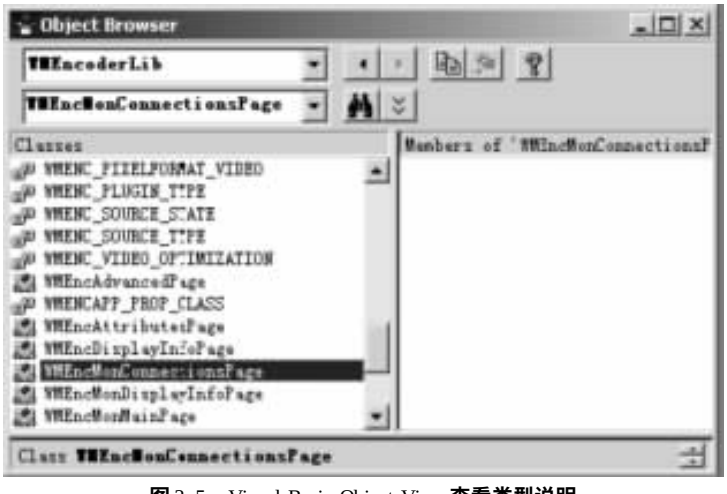


图 3.5　Visual Basic Object View 查看类型说明

3.2.3 详细代码分析

本例中包含两个窗体：FrmConfigure **窗体负责配置编码文件，**FrmEncControl **窗体控制编码状态。**FrmEncControl **窗体如图** 3.6 **所示。**

图 3.6 FrmEncControl **窗体**

FrmEncControl **窗体包含的代码如下：**

```
// 定义一个 Encoder 对象
Dim Encoder As New WMEncoder
Private Sub CmdConfigure_Click()
    On Error GoTo err_handler          // 错误捕获
    // 显示配置对话框
    Set FrmConfigure.EncoderMain = Encoder
    FrmConfigure.Show 1
    Set FrmConfigure.EncoderMain = Nothing
    Exit Sub
err_handler:
    MsgBox "属性页配置错误"
End Sub
Private Sub CmdExit_Click()
    // 退出
    Unload Me
End Sub
Private Sub CmdStart_Click()
    On Error GoTo err_handler
    // 开始编码
    Encoder.Start
    CmdStart.Enabled = False
    CmdStop.Enabled = True
    Exit Sub
```

```
    err_handler：
        MsgBox "编码过程出错"
    End Sub
    Private Sub CmdStop_Click()
        On Error GoTo err_handler
        // 停止编码
        Encoder. Stop
        CmdStop. Enabled = False
        CmdStart. Enabled = True
        Exit Sub
    err_handler：
        MsgBox "停止编码出错"
    End Sub
    Private Sub Form_Load()
        On Error GoTo err_handler
        // 创建 Encoder 监视页,用以监视 Encoder 的各种属性
        Dim PpgMain As WMEncMonConnectionsPage
        Set PpgMain = New WMEncMonMainPage
        // 将 Encoder 与显示页容器绑定
        PageShell. AddObject Encoder 'MSPshell. dll
        // 将监视属性页加入显示页容器
        PageShell. AddPage PpgMain
        Exit Sub
    err_handler：
        MsgBox "静态页加载错误"
    End Sub
    Private Sub Form_Unload(Cancel As Integer)
        // 保证 Encoder 的状态为停止
        If (Encoder. RunState < > WMENC_ENCODER_STOPPING) And (En-
coder. RunState < > WMENC_ENCODER_STOPPED) Then
          Encoder. Stop
        End If
        // 如果没有停止,那么继续等待到停止
        While Encoder. RunState < > WMENC_ENCODER_STOPPED
          DoEvents
        Wend
        // 释放 Encoder 对象
        Set Encoder = Nothing
        End
    End Sub
```

FrmConfigure 窗体设计界面如图 3.7 所示：

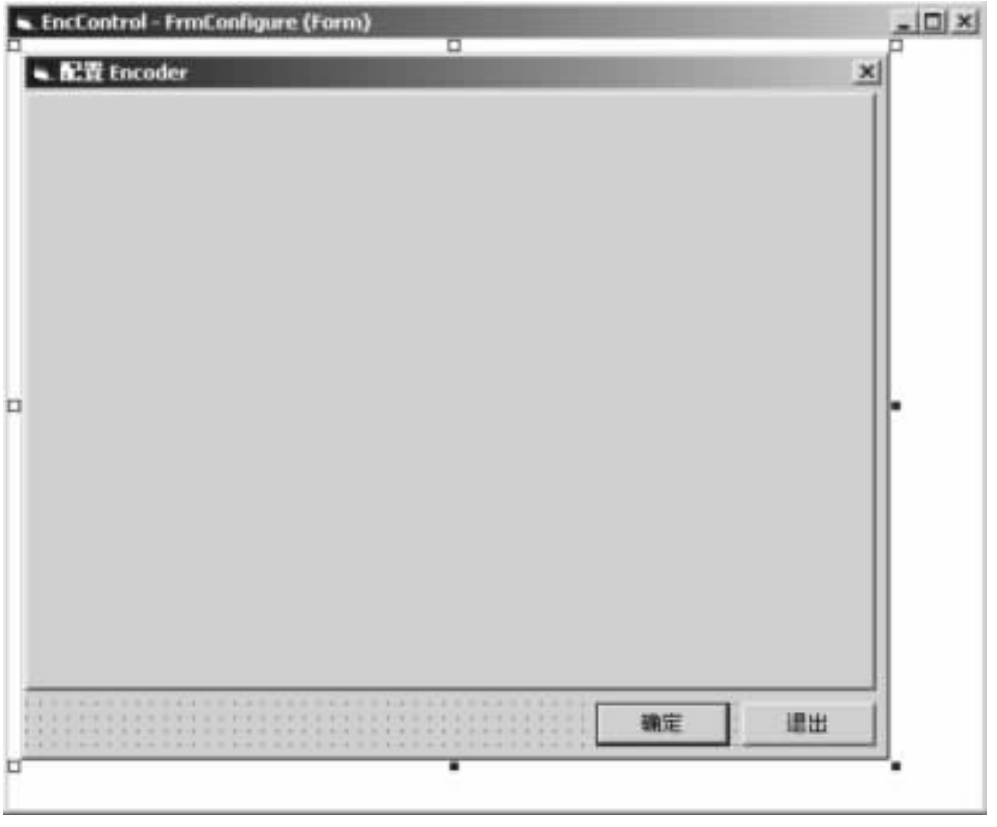


图 3.7　FrmConfigure 窗体设计图

FrmConfigure 窗体包含的代码如下：

```
// 保存我们配置的对象
Dim EncoderConfig As WMEncoder
Private Sub CmdCancel_Click()
    // 退出
    Unload Me
End Sub
Private Sub CmdOk_Click()
    On Error GoTo err_handler
    // 通过属性页的属性改变 Encoder 的运行参数
    PropPageShell. Apply                // 应用
    EncoderConfig. PrepareToEncode（True）    // 修正
    Unload Me
    Exit Sub
err_handler：
    MsgBox "Encoder 配置失败"
```

```
End Sub
Private Sub Form_Load()
    // 创建 Encoder 配置属性页
    Dim PpgSources As New WMEncSourcesAltPage
    Dim PpgDesc As New WMEncDisplayInfoPage
    Dim PpgAttr As New WMEncAttributesPage
    Dim PpgEnc As New WMEncProfilePage
    Dim PpgOutput As New WMEncOutputPage
    On Error GoTo err_handler
    // 将 Encoder 与显示页容器绑定
    PropPageShell.AddObject EncoderConfig
    // 将监视属性页加入显示页容器
    PropPageShell.AddPage PpgSources
    PropPageShell.AddPage PpgDesc
    PropPageShell.AddPage PpgAttr
    PropPageShell.AddPage PpgEnc
    PropPageShell.AddPage PpgOutput
    Exit Sub
err_handler:
    MsgBox "Encoder 属性页加载错误" & vbCrLf & Err.Description
End Sub
Public Property Set EncoderMain(vNewValue As WMEncoder)
    // 加入引用对象
    Set EncoderConfig = vNewValue
End Property
```


3.3　编程创作二：批量编码工具

上一个编程实例中，已经展示了如何简单地控制 Encoder 对象来达到最简单的编码需求，同时也熟悉了 Microsoft PropShell Control 1.0 控件，现在我们已经拥有了操作 Encoder 的最基本的能力，就是使用 Encoder 对象来编码。

现在，有这样的一个需求，用户想在他的音乐欣赏网站上提供歌曲试听。为了减少存储空间，加快听众试听的速度，就需要将其音乐欣赏网站上所有原始的 MP3 音乐转换为 WMA 格式的音乐，然后再通过流媒体服务器进行点播，以达到预期的效果。现在就让我们一起来设计这个批量转换 MP3 至 WMA 的工具。

通过对本编程实例的学习，可以更加牢固地掌握对 Encoder 对象的控制。那么，MP3 到 WMA 的转换与 MPG 到 WMV 的转换有什么区别？很简单，两者理论上是相同的，只是在细节上的处理有一些不同而已，而且现在市场中有很多基于 Windows

Media 的编码工具，通过深入的研究以后，一样也可以作出出色的编码工具。

3.3.1 批量编码工具设计思路

实现批量转换 MP3 至 WMA 格式的工具，可以按照如下步骤来完成：

第 1 步 选择所需的转换文件，以及转换此文件所需的编码格式，并设置编码后文件输出路径，建立转换列表。

第 2 步 按照配置文件转换 MP3 原始文件，并将编码后输出至目标路径。

第 3 步 转换的同时，监视转换过程。

本例中主要使用两个窗体：FrmBatcherIn 窗体（如图 3.8 所示）负责保存转换列表以及各项功能操作，在转换文件的同时，监视文件转换过程；FrmAddFiles 窗体（图 3.9）负责向转换列表中添加要转换的文件信息，并选择文件转换配置文件。

图 3.8 FrmBatcherIn 窗体

图 3.9　FrmAddFiles 窗体

本例的代码位于本书配套光盘/Sample/Sample3_2 下，可供参考。

3.3.2　深入探讨 Encoder 对象

在编写之前，先解决以下几个编程中的关键问题。

（1）读取现有的配置文件（Profile）

这个 MP3 至 WMA 的转换工具，可以给每一个转换的 MP3 文件设置不同的配置文件，这样就需要将所有的配置文件列出，所以需要使用 IWMEncProfileCollection 类，IWMEncProfileCollection 类是配置文件（IWMEncProfile）的集合，属于 Encoder 对象的一部分，本例需要使用它来显示所有的配置文件，代码如下：

```
Dim PrfEncoder As WMEncoder                 // 定义 Encoder 对象
Dim PrfCol As IWMEncProfileCollection       // 定义 IWMEncProfileCollection 对象
Dim Prf As IWMEncProfile                    // 定义 IWMEncProfile 对象
InputFile = ""
OutputFile = ""
Profile = ""
Set PrfEncoder = New WMEncoder              // 建立 PrfEncoder 实例
Set PrfCol = PrfEncoder.ProfileCollection   // 建立 PrfCol 对 PrfEncoder 的引用
For Each Prf In PrfCol
CmbProfile.AddItem Prf.Name                 // 向配置文件列表添加配置文件名称
Next
Set PrfEncoder = Nothing                    // 释放 PrfEncoder 实例
```

通过以上代码就可以将符合条件的配置文件列出，如图 3.10 所示。

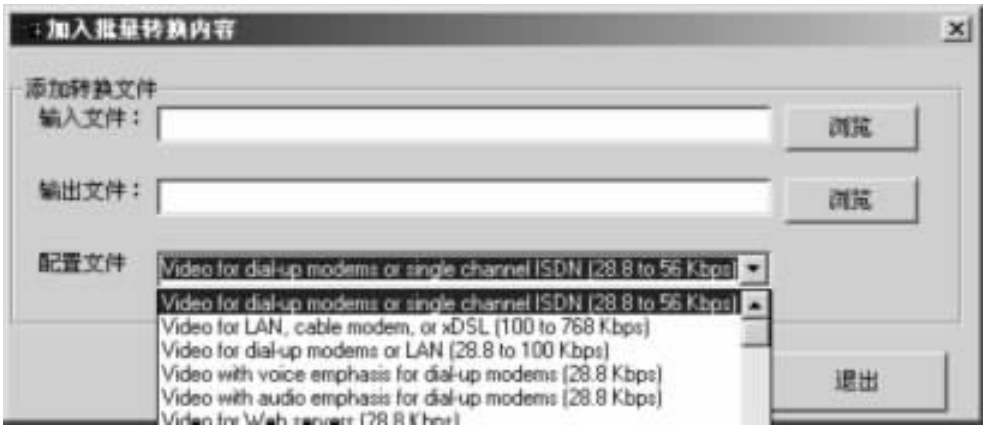


图 3.10　显示所有配置文件

这样，用户就可以为每一个被编码文件选择不同的配置文件。

只能有一个配置文件可以被配置到流文件中。Windows Media Encoder 提供的缺省的配置文件不可以被编辑或者删除。用户可以用 Profile Manager 创建自己的配置文件，也可以用 Windows Media Format SDK 自主开发创建和编辑配置文件的系统。

(2) 提供最基本编码参数

在实例一中使用了 WMEncSourcesAltPage，WMEncDisplayInfoPage，WMEncAttributesPage，WMEncProfilePage，WMEncOutputPage 等可见属性页提供 Encoder 运行的参数，这些都是直观的，用户可以直接操作。在本例中，将使用另外一种向 Encoder 提供运行参数的方式，使用 IWMEncSourceGroupCollection 类，IWMEncSourceGroupCollection 类是 IWMEncSourceGroup 的集合。

使用 Encoder 将 MP3 编码为 WMA 文件，只需要输入文件参数，配置文件参数，输出文件参数就可以运行，本例采用 IWMEncSourceGroup 类向 Encoder 运行提供参数，代码如下：

```
Dim SrcGrpCol As IWMEncSourceGroupCollection
Dim SrcGrp As IWMEncSourceGroup        // 定义 IWMEncSourceGroup 对象
Set SrcGrpCol = Encoder.SourceGroupCollection        // 设置引用
Set SrcGrp = SrcGrpCol.Add("Batcher")                // 添加处理组
// ********** 以下提供三个基本运行参数 **********
SrcGrp.AutoSetFileSource (InputFile)                 // 设置输入文件
SrcGrp.Profile = Profile                             // 设置配置文件
```

```
Encoder. File. LocalFileName = OutputFile                    // 设置输出文件
// ********** 获得参数,开始编码 **********************
Encoder. Start                                               // 开始编码
```

(3) 监视编码过程

在编码的过程中,可以监视编码进行的状态以及各项运行参数,并通过界面反映出来。在实例一中,已经使用了这个类,在此,对 WMEncMonMainPage 类做详细介绍。WMEncMonMainPage 类主要实现对编码过程的监视,它必须嵌入到 Microsoft PropShell Control 1. 0 中,将 Microsoft PropShell Control 1. 0 作为显示容器,WMEncMonMainPage 显示如图 3. 11 所示。

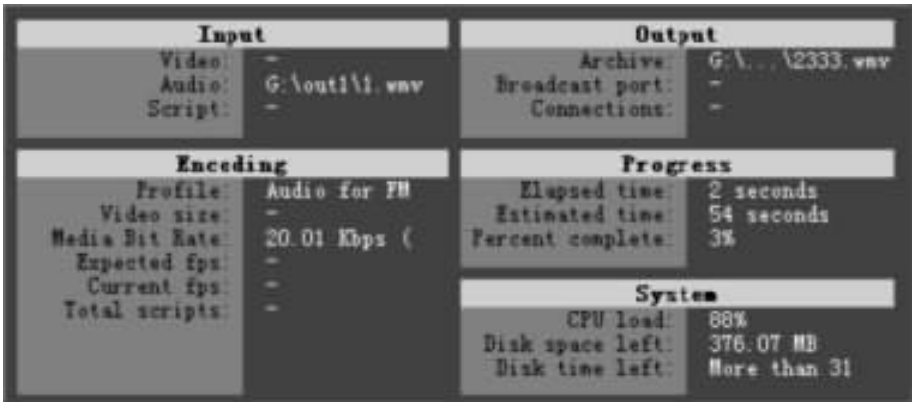


图 3. 11　WMEncMonMainPage 示例

可以看出通过 WMEncMonMainPage 可以得到当前 Encoder 运行的各项状态,使用 WMEncMonMainPage 非常简单,代码如下:

```
Dim PpgMain As New WMEncMonMainPage    // 创建 WMEncMonMainPage 实例
MSPropMain. AddObject Encoder    // 将 Encoder 对象与 Microsoft PropShell Control 绑定
MSPropMain. AddPage PpgMain    // 将 WMEncMonMainPage 与 PropShell Control 绑定
```

通过以上代码,在 WMEncMonMainPage 就可以随时随地地显示 Encoder 当前的状态了。

其他属性页(如下)使用方法大同小异,请读者自行学习:WMEncSourceSAltPage, WMEncDisplayInfoPage, WMEncAttributesPage, WMEncProfilePage, WMEncOutputPage,WMEncAdvancedPage。

通过解决上面的 3 个关键的问题,剩下的问题已经很简单了,大部分都是基本的 Windows 编程。本例包含两个窗体,运行状态如图 3.8 和图 3.9 所示。

FrmBatcherIn 和 FrmAddFiles 窗体所含代码如下所示:

```
Option Explicit
Dim WithEvents Encoder As WMEncoder
Dim EncError As Long
Private Sub CmdAdd_Click()
    Dim Profile As String
    Dim InputFile As String
    Dim OutputFile As String
    Dim Success As Integer
    On Error GoTo Err_Handler
    FrmAddFiles. Show 1
    InputFile = FrmAddFiles. GetInputFile
    If InputFile = "" Then
        CmdAdd. SetFocus
        Exit Sub
    End If
    OutputFile = FrmAddFiles. GetOutputFile
    Profile = FrmAddFiles. GetProfile
    InputFile = Trim(InputFile)
    OutputFile = Trim(OutputFile)
    Profile = Trim(Profile)
    Success = Populate_List(InputFile, OutputFile, Profile)
    If Success = 1 Then
        MsgBox "Error Populating the list"
    End If
    CmdRemove. Enabled = True
    CmdRemoveAll. Enabled = True
    CmdClearResults. Enabled = True
    CmdSave. Enabled = True
    CmdStart. Enabled = True
    Exit Sub
Err_Handler:
    MsgBox "Error Adding Files", vbCritical, "Error - " & Err. Number
End Sub
Private Sub CmdClearResults_Click()
    Dim cnt As Integer
```

```
    On Error GoTo Err_Handler
    cnt = LvwBatcher.ListItems.Count
    If cnt = 0 Then
        MsgBox "Nothing to clear"
    End If
    While cnt > 0
        If LvwBatcher.ListItems(cnt).ListSubItems("Result") = "" Then
        Else
            LvwBatcher.ListItems(cnt).ListSubItems("Result") = ""
        End If
        cnt = cnt - 1
    Wend
    CmdStart.Enabled = True
    Exit Sub
Err_Handler:
    MsgBox "Error clearing the list", vbExclamation
End Sub
Private Sub CmdExit_Click()
    Set Encoder = Nothing
    Unload Me
    End
End Sub
Private Sub CmdLoad_Click()
    Dim FsBat As Object
    Dim InFile As String
    Dim BatcherFile As Object
    Dim ListIn As String
    Dim InputFile As String
    Dim OutputFile As String
    Dim Profile As String
    Dim ColonPosIn As Integer
    Dim ColonPosOut As Integer
    Dim ColonPosPrf As Integer
    Dim LoadCnt As Integer
    On Error GoTo Err_Handler
    DlgBat.DialogTitle = "Input File"
    DlgBat.Flags = cdlOFNFileMustExist
    DlgBat.ShowOpen
    InFile = DlgBat.FileName
```

```
        If InFile = "" Then
            Exit Sub
        End If
        DlgBat. FileName = ""
        Set FsBat = CreateObject("Scripting. FileSystemObject")
        Set BatcherFile = FsBat. opentextfile(InFile, 1, 0)
        LvwBatcher. ListItems. Clear
        LoadCnt = 1
        While BatcherFile. AtEndOfStream < > True
            ListIn = BatcherFile. readline
            ColonPosIn = InStr(1, ListIn, ";")
            InputFile = Mid(ListIn, 1, ColonPosIn - 1)
            ColonPosOut = InStr(ColonPosIn + 1, ListIn, ";")
            OutputFile = Mid(ListIn, ColonPosIn + 1, ColonPosOut - Colon-
PosIn - 1)
            ColonPosPrf = InStr(ColonPosOut + 1, ListIn, ";")
            Profile = Mid(ListIn, ColonPosOut + 1, ColonPosPrf - ColonPo-
sOut - 1)
            LvwBatcher. ListItems. Add LoadCnt, InputFile
            LvwBatcher. ListItems. Item(LoadCnt). ListSubItems. Add ,_
    "OutputFile", OutputFile
            LvwBatcher. ListItems. Item(LoadCnt). ListSubItems. Add , "Pro-
file", Profile
            LvwBatcher. ListItems. Item(LoadCnt). ListSubItems. Add , "Re-
sult", ""
            LoadCnt = LoadCnt + 1
        Wend
        BatcherFile. Close
        If LvwBatcher. ListItems. Count > 0 Then
            LvwBatcher. FullRowSelect = True
            Set LvwBatcher. DropHighlight = LvwBatcher. ListItems(1)
            CmdRemove. Enabled = True
            CmdRemoveAll. Enabled = True
            CmdClearResults. Enabled = True
            CmdSave. Enabled = True
            CmdStart. Enabled = True
            ' CmdStop. Enabled = True
        End If
        Exit Sub
```

```
    Err_Handler:
        MsgBox "Error Loading Saved Configuration", vbCritical, "Error - " &
Err.Number
    End Sub
    Private Sub CmdRemove_Click()
        On Error GoTo Err_Handler
        If LvwBatcher.ListItems.Count = 0 Then
            Exit Sub
        End If
        LvwBatcher.ListItems.Remove LvwBatcher.SelectedItem.Index
        Exit Sub
    Err_Handler:
        MsgBox "Error Removing Selected Entry From List", vbExclamation
    End Sub
    Private Sub CmdRemoveAll_Click()
        Dim cnt As Integer
        On Error GoTo Err_Handler
        cnt = LvwBatcher.ListItems.Count
        While cnt > 0
            LvwBatcher.ListItems.Remove cnt
            cnt = cnt - 1
        Wend
        Exit Sub
    Err_Handler:
        MsgBox "Error Removing Entries from List", vbExclamation
    End Sub
    Private Sub CmdSave_Click()
        Dim ListSave As String
        Dim SaveFile  As String
        Dim FsBat As Object
        Dim BatcherFile As Object
        Dim SaveCnt As Integer
        Dim InputFile As String
        Dim OutputFile As String
        Dim Profile As String
        On Error GoTo Err_Handler
        DlgBat.DialogTitle = "Save File"
        DlgBat.ShowSave
        SaveFile = DlgBat.FileName
```

```
        DlgBat. FileName = ""
        Set FsBat = CreateObject("Scripting. FileSystemObject")
        If SaveFile = "" Then
            Exit Sub
        End If
        Set BatcherFile = FsBat. CreateTextFile(SaveFile, True)
        SaveCnt = 1
        While SaveCnt < = LvwBatcher. ListItems. Count
            InputFile = LvwBatcher. ListItems(SaveCnt)
            OutputFile = LvwBatcher. ListItems(SaveCnt). ListSubItems("Out-
putFile")
            Profile = LvwBatcher. ListItems(SaveCnt). ListSubItems("Profile")
            ListSave = InputFile & ";" & OutputFile & ";" & Profile & ";"
            BatcherFile. writeline (ListSave)
            SaveCnt = SaveCnt + 1
        Wend
        BatcherFile. Close
        Exit Sub
    Err_Handler:
        MsgBox "Error Saving List Entries", vbCritical, "Error - " & Err. Num-
ber
    End Sub
    Private Sub CmdStart_Click()
        Dim Success As Long
        Dim CntStart As Long
        Dim Status As String
        Dim InputFile As String
        Dim OutputFile As String
        Dim Profile As String
        Dim PpgMain As New WMEncMonMainPage
        Dim SrcGrpCol As IWMEncSourceGroupCollection
        Dim SrcGrp As IWMEncSourceGroup
        Dim ErrVal
        On Error GoTo Err_Handler
        CmdStart. Enabled = False
        CmdAdd. Enabled = False
        CmdRemove. Enabled = False
        CmdClearResults. Enabled = False
        CmdRemoveAll. Enabled = False
```

```
        CmdSave. Enabled = False
        CmdLoad. Enabled = False
        CmdStop. Enabled = True
        If LvwBatcher. ListItems. Count = 0 Then
            MsgBox "No files to encode"
            Exit Sub
        End If
        Set Encoder = New WMEncoder
        MSPropMain. AddObject Encoder
        MSPropMain. AddPage PpgMain
        Set SrcGrpCol = Encoder. SourceGroupCollection
        Set SrcGrp = SrcGrpCol. Add("Batcher")
        CntStart = 1
        While CntStart < = LvwBatcher. ListItems. Count
            InputFile = LvwBatcher. ListItems(CntStart). Text
            OutputFile = _
    LvwBatcher. ListItems(CntStart). ListSubItems("OutputFile"). Text
            Profile = LvwBatcher. ListItems(CntStart). ListSubItems("Profile"). Text
            Status = LvwBatcher. ListItems(CntStart). ListSubItems("Result"). Text
            If Status < > "Done" Then
                Success = ProcessInput(Encoder, InputFile, OutputFile, Profile)
                If Success = 0 Then
                    LvwBatcher. ListItems(CntStart). ListSubItems("Result"). Text_
    = "Done"
                Else
                    ErrVal = Hex(Success)
                    LvwBatcher. ListItems(CntStart). ListSubItems("Result"). Text_
    = "Error :" & ErrVal
                End If
            End If
            CntStart = CntStart + 1
            If CntStart < = LvwBatcher. ListItems. Count Then
                Set LvwBatcher. SelectedItem = LvwBatcher. ListItems(CntStart)
                Set LvwBatcher. DropHighlight = LvwBatcher. ListItems(CntStart)
            End If
        Wend
        If CntStart > LvwBatcher. ListItems. Count Then
            Set Encoder = Nothing
            CmdStop. Enabled = False
```

```
            CmdStart. Enabled = False
        End If
        CmdAdd. Enabled = True
        CmdRemove. Enabled = True
        CmdClearResults. Enabled = True
        CmdRemoveAll. Enabled = True
        CmdSave. Enabled = True
        CmdLoad. Enabled = True
        Exit Sub
Err_Handler:
        MsgBox "Error Processing Input", vbCritical, "Error - " & Err. Number
End Sub
Private Sub CmdStop_Click()
        On Error GoTo Err_Handler
        If Encoder. RunState = WMENC_ENCODER_RUNNING_
Or WMENC_ENCODER_STARTING Or WMENC_ENCODER_PAUSED Then
            Encoder. Stop
        End If
        Set Encoder = Nothing
        CmdAdd. Enabled = True
        CmdRemove. Enabled = True
        CmdRemoveAll. Enabled = True
        CmdClearResults. Enabled = True
        CmdLoad. Enabled = True
        CmdSave. Enabled = True
        CmdStart. Enabled = True
        Exit Sub
Err_Handler:
        MsgBox "Error stopping Encoder", vbCritical, "Error - " & Err. Number
End Sub
Private Sub Encoder_OnError(ByVal hResult As Long)
        EncError = Encoder. ErrorState
End Sub
Private Sub Form_Load()
        Dim I As Integer
        On Error GoTo Err_Handler
        For I = 1 To 4
            LvwBatcher. ColumnHeaders. Item(I). Width = LvwBatcher. Width / 4
```

```
        Next I
        Set Encoder = Nothing
        StaEncoderStat.SimpleText = "Encoder Stopped"
        CmdRemove.Enabled = False
        CmdRemoveAll.Enabled = False
        CmdClearResults.Enabled = False
        CmdSave.Enabled = False
        CmdStart.Enabled = False
        CmdStop.Enabled = False
        Exit Sub
    Err_Handler:
        MsgBox "Error Executing Application", vbCritical, "Error - " & Err.
Number
    End Sub
    Function Populate_List(InputFile As String, OutputFile _
    As String, Profile As String) As Integer
        Dim SrcGrpCol As IWMEncSourceGroupCollection
        Dim SrcGrp As IWMEncSourceGroup
        Dim Count As Integer
        Dim periodpos As Long
        Dim AudCnt As Integer
        Dim VidCnt As Integer
        On Error GoTo Err_Handler
        If OutputFile = "None" Then
            Set Encoder = New WMEncoder
            Set SrcGrpCol = Encoder.SourceGroupCollection
            Set SrcGrp = SrcGrpCol.Add("SrcList")
            SrcGrp.AutoSetFileSource (InputFile)
            OutputFile = InputFile
            periodpos = InStrRev(InputFile, ".")
            If periodpos > 0 Then OutputFile = Left(OutputFile, periodpos - 1)
            AudCnt = SrcGrp.SourceCount(WMENC_AUDIO)
            VidCnt = SrcGrp.SourceCount(WMENC_VIDEO)
            If AudCnt > 0 And VidCnt = 0 Then
                OutputFile = OutputFile & ".wma"
            Else
                OutputFile = OutputFile & ".wmv"
            End If
        Set Encoder = Nothing
```

```
        End If
        Count = LvwBatcher.ListItems.Count + 1
        LvwBatcher.ListItems.Add Count, , InputFile
        LvwBatcher.ListItems(Count).ListSubItems.Add , "OutputFile", OutputFile
        LvwBatcher.ListItems(Count).ListSubItems.Add , "Result", ""
        LvwBatcher.FullRowSelect = True
        Set LvwBatcher.SelectedItem = LvwBatcher.ListItems(1)
        Set LvwBatcher.DropHighlight = LvwBatcher.ListItems(1)
      Exit Function
    Err_Handler:
        Populate_List = 1
    End Function
    Function ProcessInput(Encoder As WMEncoder, InputFile As String,_
    OutputFile As String, Profile As String) As Long
        Dim SrcGrpCol As IWMEncSourceGroupCollection
        Dim SrcGrp As IWMEncSourceGroup
        Dim PrfCol As IWMEncProfileCollection
        Dim Prf As IWMEncProfile
        Dim Success As Integer
        On Error GoTo Err_Handler
        Set SrcGrpCol = Encoder.SourceGroupCollection
        Set SrcGrp = SrcGrpCol.Item(0)
        SrcGrp.AutoSetFileSource (InputFile)
        Set PrfCol = Encoder.ProfileCollection
        Success = GetProfile(Profile, Prf, PrfCol, SrcGrp)
        If Success = 1 Then
            ProcessInput = 1
            Exit Function
        End If
        Encoder.File.LocalFileName = OutputFile
        EncError = 0
        Encoder.Start
        StaEncoderStat.SimpleText = "Encoder Running"
        While Encoder.RunState <> WMENC_ENCODER_STOPPED And
EncError = 0
            DoEvents
        Wend
        StaEncoderStat.SimpleText = "Encoder Stopped"
        If EncError <> 0 Then
```

```
    ProcessInput = EncError
        End If
        Exit Function
    Err_Handler：
        ProcessInput = Err. Number
    End Function
    Function GetProfile(Profile As String, Prf As IWMEncProfile, PrfCol As_
    IWMEncProfileCollection, SrcGrp As IWMEncSourceGroup) As Integer
        Dim ExistsFlg As Boolean
        On Error GoTo Err_Handler
        ExistsFlg = False
        For Each Prf In PrfCol
            If UCase(Prf. Name) = UCase(Profile) Then
                SrcGrp. Profile = Prf
                ExistsFlg = True
                Exit For
            End If
        Next
        If ExistsFlg = False Then
            GetProfile = 1
        End If
        Exit Function
    Err_Handler：
        GetProfile = 1
    End Function
    Private Sub LvwBatcher_Click()
        Set LvwBatcher. DropHighlight = LvwBatcher. SelectedItem
        LvwBatcher. FullRowSelect = True
    End Sub
```

FrmAddFiles **窗体所含代码如下所示：**

```
    Option Explicit
    Dim InputFile As String
    Dim OutputFile As String
    Dim Profile As String
    Private Sub CmbProfile_Change()
        If (Len(CmbProfile. Text)) = 1 Then
            If CmbProfile. Text = " " Then
```

```
                    CmbProfile. Text = ""
                End If
            End If
        End Sub
        Private Sub CmbProfile_Click()
            Profile = CmbProfile. Text
        End Sub
        Private Sub CmdBrowse_Click(Index As Integer)
            On Error GoTo Err_Handler
            If Index = 0 Then
                DlgInOut. DialogTitle = "Input File"
                DlgInOut. Filter = "Video Files(*. asf,*. avi,*. bmp,*. mpg,
*. wmv)|_
        *. asf;*. avi;*. bmp;*. mpg;*. wmv|Audio _
        Files(*. asf,*. avi,*. mp3,*. mpg,*. wav,*. wma)|_
        *. asf;*. avi;*. mp3;*. mpg;*. wav;*. wma"
                DlgInOut. Flags = cdlOFNFileMustExist
                DlgInOut. ShowOpen
                TxtInput. Text = DlgInOut. FileName
                DlgInOut. FileName = ""
                TxtInput. SetFocus
            Else
                DlgInOut. DialogTitle = "output file"
                DlgInOut. Filter = "Windows Media files(*. wmv;*. wma)_
        |*. wmv;*. wma|Windows Media video files(*. wmv)|*_
        . wmv|Windows Media audio files(*. wma)|*. wma|Windows Media _
        (*. asf)|*. asf"
                DlgInOut. Flags = cdlOFNHideReadOnly
                DlgInOut. ShowSave
                TxtOutput. Text = DlgInOut. FileName
                DlgInOut. FileName = ""
            End If
            Exit Sub
        Err_Handler:
            MsgBox "Error Opening File", vbExclamation
        End Sub
        Private Sub CmdCancel_Click()
            InputFile = ""
            Unload Me
```

```
End Sub
Private Sub CmdOk_Click()
    On Error GoTo Err_Handler
    If TxtInput.Text = "" Then
        MsgBox "Please enter the input file"
        TxtInput.SetFocus
        Exit Sub
    End If
    If CmbProfile.Text = "" Then
        MsgBox "Please enter a profile"
        CmbProfile.SetFocus
        Exit Sub
    End If
    If TxtOutput = "" Then
        OutputFile = "None"
    Else
        OutputFile = TxtOutput.Text
    End If
    InputFile = TxtInput.Text
    Profile = CmbProfile.Text
    Unload Me
    Exit Sub
Err_Handler:
    MsgBox "Error In Input"
End Sub
Private Sub Form_Load()
    Dim PrfEncoder As WMEncoder
    Dim PrfCol As IWMEncProfileCollection
    Dim Prf As IWMEncProfile
    On Error GoTo Err_Handler
    InputFile = ""
    OutputFile = ""
    Profile = ""
    Set PrfEncoder = New WMEncoder
    Set PrfCol = PrfEncoder.ProfileCollection
    For Each Prf In PrfCol
        CmbProfile.AddItem Prf.Name
    Next
    CmbProfile.Text = CmbProfile.List(0)
```

```
        Set PrfEncoder = Nothing
        Exit Sub
    Err_Handler:
        MsgBox "Error Getting Profiles of Encoder", vbCritical, "Error - " &
Err.Number
    End Sub
    Private Sub TxtInput_Change()
            If (Len(TxtInput.Text)) = 1 Then
            If TxtInput.Text = " " Then
                TxtInput.Text = ""
             End If
        End If
    End Sub
    Public Property Get GetProfile() As String
        GetProfile = Profile
    End Property
    Private Sub TxtInput_LostFocus()
        Dim PrfEncoder As WMEncoder
        Dim PrfCol As IWMEncProfileCollection
        Dim Prf As IWMEncProfile
        Dim SrcGrpCol As IWMEncSourceGroupCollection
        Dim SrcGrp As IWMEncSourceGroup
        Dim AudCnt As Integer
        Dim VidCnt As Integer
        Dim CurrentProf As String
        Dim ModifiedProf As String
        Dim CntPrf As Integer
        If TxtInput.Text = "" Then
            Exit Sub
        End If
        Set PrfEncoder = New WMEncoder
        Set SrcGrpCol = PrfEncoder.SourceGroupCollection
        Set SrcGrp = SrcGrpCol.Add("SrcList")
        SrcGrp.AutoSetFileSource (TxtInput.Text)
        Set PrfCol = PrfEncoder.ProfileCollection
        AudCnt = SrcGrp.SourceCount(WMENC_AUDIO)
        VidCnt = SrcGrp.SourceCount(WMENC_VIDEO)
        CurrentProf = CmbProfile.Text
```

```
If AudCnt > 0 And VidCnt > 0 Then
        CmbProfile. Clear
        For Each Prf In PrfCol
            If Prf. MediaCount(WMENC_AUDIO) > 0 And_
Prf. MediaCount(WMENC_VIDEO) > 0 Then
                CmbProfile. AddItem Prf. Name
            End If
        Next
        CmbProfile. Text = CmbProfile. List(0)
        CntPrf = CmbProfile. ListCount
        While CntPrf > = 0
            ModifiedProf = CmbProfile. List(CntPrf)

            If UCase(ModifiedProf) = UCase(CurrentProf) Then
                CmbProfile. Text = CmbProfile. List(CntPrf)
                Exit Sub
            End If
            CntPrf = CntPrf - 1
        Wend
    End If
     If AudCnt > 0 And VidCnt = 0 Then
        CmbProfile. Clear
        For Each Prf In PrfCol
            If Prf. MediaCount(WMENC_AUDIO) > 0 And_
Prf. MediaCount(WMENC_VIDEO) = 0 Then
                CmbProfile. AddItem Prf. Name
            End If
        Next
        CntPrf = CmbProfile. ListCount
        While CntPrf > = 0
            ModifiedProf = CmbProfile. List(CntPrf)

            If UCase(ModifiedProf) = UCase(CurrentProf) Then
                CmbProfile. Text = CmbProfile. List(CntPrf)
                Exit Sub
            End If
            CntPrf = CntPrf - 1
        Wend
    End If
```

```
        If AudCnt = 0 And VidCnt = 0 Then
            MsgBox "The input file doesnot contain Either an audio or a video"
        End If
    End Sub
    Private Sub TxtOutput_Change()
        If (Len(TxtOutput.Text)) = 1 Then
            If TxtOutput.Text = " " Then
                TxtOutput.Text = ""
            End If
        End If
    End Sub
    Public Property Get GetInputFile() As Variant
        GetInputFile = InputFile
    End Property
    Public Property Get GetOutputFile() As Variant
        GetOutputFile = OutputFile
    End Property
```

本例采用的是英文版本的 Windows Media Encoder 组件，如果您使用中文版本的 Encoder 可能运行不正常。

3.4 编程创作三：Encoder 广播站

广播站有非常广阔的应用。打个比方，上一个创作中的网络电台，服务器位于异地，我是节目主持人，需要通过服务器向用户广播播出自己的节目，而我的这些节目是在本地的，这时，广播站就可以派上用场了。可以将我的节目放入广播站，对服务器发出编码后的信息，听众就可以直接从服务器端收听。也许有的读者会问，把录制好的声音放到服务器上点播不就可以解决问题了么？或者使用 ASX 构建一个播放列表在服务器也可以吗？

毕竟 Encoder 是一个 Encoder，它可以对我的听众实时地传递信息，也就是可以实时地对听众发送我的在线广播，何乐而不为呢？

在本例中，将以设计一个广播站来学习 Encoder 的网络功能，其中主要使用 IWMEncBroadcast 类，下面就一步一步来创建这个非常诱人的工具。

本例位于配套光盘 Sample/Sample3_3 下。

3.4.1　广播站设计方案

先提出软件的设计方案，广播站按照操作是由 4 个部分组成：

①主要操作监视界面，如图 3.12 所示。

图 3.12　操作监视界面

主监视界面中有个非常熟悉的面孔，就是 WMEncMonMainPage 了，几乎所有需要显示 Encoder 状态的功能，都用到了这个类。

②建立新的工作站界面，如图 3.13 所示。

本界面的主要功能是建立一个工作站点，并设置基本属性。

③添加工作站播放列表界面，如图 3.14 所示。

图 3.13　建立新的工作站

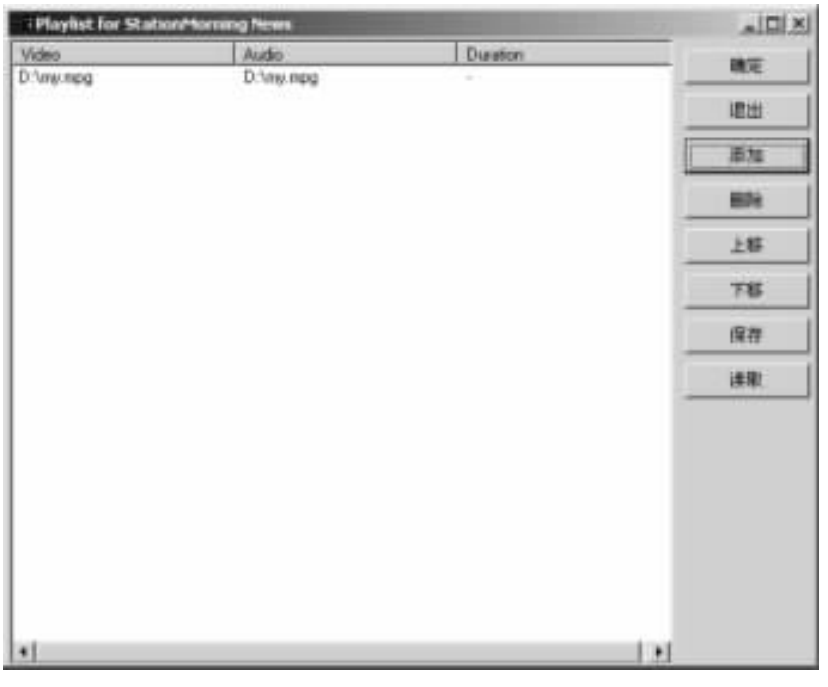


图 3.14　添加工作站播放列表界面

本界面主要完成向一个工作站文件中添加播放内容的功能。

④添加播放列表内容界面，如图 3.15 所示。

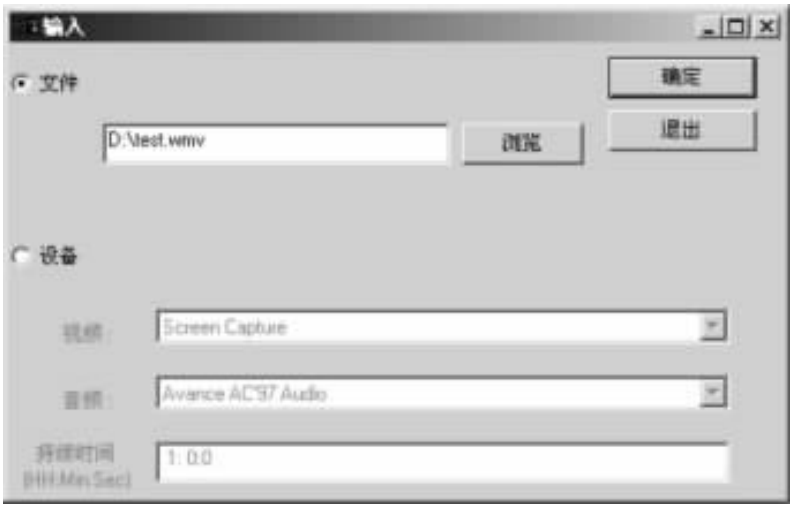


图 3.15　播放列表内容界面

本界面用以添加播放列表中的内容，内容可以来自文件，也可以来自屏幕捕获、麦克风（直播）等实时的音视频来源。

上述 4 个方面是广播站的用户设计。在程序设计方面，建立一个广播站类 StnMgrEnc，可以控制 Encoder 实现对每一个广播站播放列表的编码和传输。在开始之前，应了解这个类，部分代码如下：

```
    Private Type EncPlaylist        // 定义 EncPlaylist 数据结构，创建播放列表类型
        Vid As String
        Aud As String
        Duration As String
    End Type
    Dim WithEvents Encoder As WMEncoder        // 以事件触发方式定义
                                                  WMEncoder
    Dim EncPlaylistArray() As EncPlaylist          // 创建播放列表结构数组
    Public Sub EncInit()              // 公共过程，初始化 WMEncoder 实例
        Set Encoder = New WMEncoder
    End Sub
    // *** 创建 WMEncoder 对象触发事件，当 Encoder 状态改变时触发 ****
    Private Sub Encoder_OnSourceStateChange(ByVal enumState As_
    WMEncoderLib. WMENC_SOURCE_STATE, ByVal enumType_
    As WMEncoderLib. WMENC_SOURCE_TYPE, ByVal iIndex As _
Integer, ByVal bstrSourceGroup As String)
        If enumState = WMENC_SOURCE_STOP Then
            Over = 1
```

```
    Else
            If enumState = WMENC_SOURCE_START Then
                Over = 0
            End If
        End If
    End Sub
```

在这个类中,定义了用以存放播放列表的数组,并设置了 Encoder 的一个触发器,详细代码参见配套光盘。

3.4.2 Encoder 的网络发送

在本创作中,编码的网络发送,是通过 WMEncOutputPage 对象来完成配置的。

```
Dm PpgOutput As New WMEncOutputPage    // 定义 WMEncOutputPage 对象
tationProp. AddObject Encoder          // 设置与 Encoder 实例的连接
StationProp. AddPage PpgOutput         // 加入属性页,就可以修改输出属性
```

读取配置完毕的端口信息,可以使用 IWMEncBroadcast 对象来读取。

```
Dim Brdcst As IWMEncBroadcast
Set Brdcst = Enc. Broadcast
Prt = Brdcst. PortNumber(WMENC_PROTOCOL_HTTP)      // 设置端口
```


本创作中对 Encoder 对象的事件进行了使用,Encoder 共有 8 个事件捕捉,分别是:

- 事件 1:OnArchiveStateChange(enumArchive As WMENC_ARCHIVE_TYPE, enumState As WMENC_ARCHIVE_STATE)

当档案文件改变时触发,接受参数为 WMENC_ARCHIVE_TYPE 类型。

- 事件 2:OnClientConnect(protocol As WMENC_BROADCAST_PROTOCOL, bstr As String)

当客户连接时触发。其中 WMENC_PROTOCOL_HTTP 为常量,值为 1。

- 事件 3:OnClientDisconnect(protocol As WMENC_BROADCAST_PROTOCOL, bstr As String)

当客户中断连接时触发。其中 WMENC_PROTOCOL_HTTP 为常量,值为 1。

- 事件 4:OnConfigChange(hResult As Long, bstr As String)

当配置文件修改时触发。

- 事件 5:OnError(hResult As Long)

当编码错误时触发。

- 事件 6:OnIndexerStateChange(enumIndexerState As _WMENC_INDEXER_STATE, bstrFile As String)

当索引状态改变时触发。

- **事件 7**：OnSourceStateChange（enumState As WMENC_SOURCE_STATE，enumType As WMENC_SOURCE_TYPE，iIndex As Integer，bstrSourceGroup As String）

当来源改变时触发。

- **事件 8**：OnStateChange（enumState As WMENC_ENCODER_STATE）

当状态改变时触发。

第4章

服务篇：定制自己的服务器

Windows Media Services SDK

4.1 Windows Media Services 概述

4.1.1 Windows Media Services 简介

Microsoft Windows Media Services 是流式传送数字媒体内容的平台，通过这种技术，用户能够在 Internet 或 Intranet 上传送数字媒体内容。

可以使用 Windows Media Services 软件开发工具包（SDK）生成用于 Windows Media Services 平台的自定义应用程序。例如可以使用 Windows Media Services SDK 进行以下操作：

①创建自定义用户界面以管理 Windows Media Services。

②以编程方式控制运行 Windows Media Services 的服务器（也称为 Windows Media 服务器）。

③创建自己的插件以自定义核心服务器功能。

④动态地创建和管理服务器端播放列表。Windows Media Services 在可缩放性、可靠性和可自定义性方面要优于以前的版本，其软件开发工具包有以下改进：

- 添加了新的对象模型，从而使用户能够更好地以编程方式控制、配置和监控 Windows Media 服务器。
- 添加了更多的核心插件类型，并可以使用自定义或第三方插件提供其他服务器功能。
- 改进了服务器端播放列表，能够更好地控制服务器流式传送的内容。
- 扩展了发布点功能。
- 重新设计了事件访问，增加了可用事件的数量。

4.1.2 使用播放列表

播放列表是采用同步多媒体集成语言（SMIL）2.0 格式的可扩展标记语言（XML）文档，它可以指定一系列数字媒体文件、编码器、URL 或其他内容服务器位置。

- 播放列表既可以是客户端播放列表，也可以是服务器端播放列表。
- 当客户端请求一个与客户端播放列表关联的发布点时，Windows Media Services 将播放列表文件传送到客户端计算机。客户端计算机上的播放器（如 Windows Media Player）管理客户端播放列表，在客户端播放列表发出后，服务器不能对已传送的播放列表进行修改。
- 相反，Windows Media Services 管理 Windows Media 服务器上的服务器端播放列表。服务器端播放列表可以是动态的，这意味着即使客户端计算机正在接收基于该播放列表的内容，也可以对播放列表进行修改。这表示允许为特定用户自定义播放列表。

• 可将服务器端播放列表用作包装播放列表。

• 包装播放列表指定应在用户请求的内容之前和/或之后播放的附加内容。用户可以创建一个包装播放列表,以便将其与多个不同的播放列表一同使用。其他可以在包装播放列表中指定的内容,包括欢迎或告别信息、广告及站点标牌。

• 可以使用 XML 文档对象模型(DOM)和服务器对象模型来控制播放列表。

• 可以用编程方式将数字媒体对象插入活动的播放列表中、任意排列播放列表元素、加载和保存现有播放列表等。客户端和服务器端播放列表均可通过这些方法创建。

4.1.3　使用发布点

发布点将客户端内容请求转换为该内容的物理路径。客户端请求通常来自客户端计算机上的浏览器软件,但也可能来自一个需要刷新其缓存的下游代理服务器。

Windows Media 服务器可支持两种类型的发布点:点播型和广播型。

对于点播发布点,内容在客户端发出请求时开始流式传送。如果流不是实时的,客户端通常可以对流进行暂停、快进和快退操作。

对于广播发布点,内容在服务器的控制下在指定时间开始流式传送。客户端可以随时连接到流,但是不能对其进行控制。文档内容通常通过按需发布点发送,而实时内容则通常通过广播发布点发送。

点播型和广播型的区别在于用户是否可以选择播放内容,点播型用户可以选择点播内容,广播型用户无法控制内容的选择(例如广播型用户无法拖放播放器的滚动条)。

可以使用 Windows Media Services SDK 服务器对象模型对点播型发布点和广播型发布点进行创建、配置、管理和监控。

Windows Media 服务既可向 Internet 用户,也可向 Intranet 用户提供内容。不过,依照目标观众不同,所采取的措施也应有所不同。

(1)如果决定向 Internet 提供内容,要注意以下事项

①大多数 Internet 没有启动多播,所以必须为多播内容提供单播替换选择。

②Internet 带宽并不可靠。Internet 通信状况波动很大,用户的连接类型各不相同。

使用代理服务器的网络可能需要特殊的配置以使 Internet 客户端能访问 Windows Media 服务器。

如果使用 Windows Media 服务的同时使用 Web 服务器程序(指它们使用同一台计算机),确认它们所用的 HTTP 端口不冲突,并且当 Internet 客户端访问时能够解析

所有链接。

(2)如果决定要提供内容给 Intranet,要注意以下问题

①Intranet 的一些系统组件可能已由防火墙隔离。
②若要提供多播内容,网络必须有启动多播的路由器。
③考虑服务器有多少管理员,以及是否需要远程管理员。
④考虑用户访问内容的方式。

4.1.4 使用服务器对象模型

Windows Media Services SDK 提供一个对象模型,可用于以编程方式的控制、配置和监控 Windows Media 服务器。为此,首先必须创建 WMSServer 对象,然后使用它获得特定任务所需的接口。对象模型的接口分为以下类别:

(1)客户端接口

对于每个连接的客户端,可检索其网络地址、端口号、所请求的播放列表、包装播放列表及状态等。

(2)限制和名称/值接口

可以指定带宽和 CPU 百分比限制及其他值。对象模型还提供自定义名称/值接口,使用户能够存储和检索应用程序所需的任何数据。

(3)计数器和诊断事件接口

使用服务器对象模型提供的计数器和事件,可监控连接和带宽信息、服务器限制信息和插件状态。

(4)发布点接口

可以对点播发布点和广播发布点进行配置和监控。

(5)播放列表接口

可以创建和管理服务器端播放列表,也可以是包装播放列表,用于确保所有客户端都能看到公共数字媒体内容(如广告),无论客户端请求哪种发布点的内容。

(6)插件接口

可以通过编程方式配置和启用标准和自定义的 Windows Media 服务器插件。

(7)缓存接口

可以配置多个 Windows Media 服务器来处理数字媒体流式传送的需求。通过将缓存/代理服务器置于网络中的关键位置,可确保客户端获得最佳的流服务。Win-

dows Media 缓存/代理服务器从源 Windows Media 服务器接收内容，然后将其分发到客户端。由于将客户端连接到最近的服务器，而不是将所有客户端都连接到源服务器，因而减少了网络流量。可以通过编程方式在整个代理网络中分发和管理内容。

4.1.5　使用插件

通过启用插件并设置其属性，可以配置和管理 Windows Media 服务器的功能。用户可以手动完成这些任务，方法是使用 Windows Media Services 包含的两个接口之一：一个基于 Web 的管理接口或一个用于 Microsoft 管理控制台的管理单元。还可以通过使用服务器对象模型实现自动化插件管理。

Windows Media 服务器协调插件的活动。该体系结构通过创建自定义插件简化了 Windows Media Services 的功能扩展。服务器的默认功能由执行以下操作的插件提供：

(1) 将客户端请求转换为可由服务器识别的命令

标准控制协议插件支持 HTTP，Microsoft Media Server（MMS）协议。

(2) 客户端身份验证

标准的身份验证插件支持匿名身份验证、基于登录凭据的身份验证，以及基于插件要求凭据的身份验证。

(3) 授予内容访问权限

标准的授权插件允许基于资源的 IP 地址、资源的访问控制列表（ACL）或特定发布点的 ACL 访问客户端。

(4) 控制服务器的缓存和代理策略

尽管没有标准的缓存代理插件，但可以创建一个自定义插件来指定要缓存的内容、内容在刷新前在缓存中的停留时间等等。

(5) 分析播放列表

标准播放列表基于 SMIL 2.0 文件中的信息，或基于指定的文件夹中的文件来组织数字媒体内容。

(6) 检索和响应服务器发送的事件通知

标准事件处理程序插件在脚本文件中运行事件处理程序，或通过 Windows 管理设备（WMI）来处理事件。

(7) 在日志文件中记录服务器和客户端的活动

标准记录插件通过向日志文件写入信息来响应日志事件。

(8)从源检索内容

标准数据源插件可以从文件、联网的 Windows Media 服务器或编码器中检索数字媒体内容,还可使用 HTTP 从 Web 服务器检索数字媒体内容。

(9)向客户端发送数字内容

标准广播插件可以将内容以单点传送的形式发送给单个客户端(例如最终用户或缓存代理服务器),或将内容以多点传送的形式同时发送给多个客户端(而无需单独向每个客户端进行发送),或将内容存档到文件中。标准插件的配置和启用既可以使用管理界面手动进行,也可使用服务器对象模型通过编程方式进行。

4.1.6 自定义插件

可以创建自定义身份验证、授权、缓存代理、数据源、事件通知、记录及播放列表分析器插件,这种灵活性能够扩展和自定义 Windows Media Services 的功能。

可将自定义插件部署在 Windows Server 上运行的 Windows Media Services 中。

要创建自定义插件,必须实现 IWMSBasicPlugin 接口和任何其他特定于所创建的插件类型的接口。例如在创建缓存代理插件时,必须实现 IWMSCacheProxy,IWMSCacheProxyServerCallback 和其他实用程序接口。

插件是组件对象模型(COM)对象,插件中必须包含可以将其注册到运行 Windows Media Services 的服务器的代码。如果使用 Microsoft Visual Studio 创建自定义插件,则必须包含创建程序集的代码以及注册 COM 组件的代码。Windows Media Services SDK 中包含有关此任务的详细信息,以及有关创建自定义插件的其他方面的信息。

4.1.7 Windows Media Services 的关键概念

1)单播和多播流

当描述客户如何从 Windows Media 服务器接收数据包时,Windows Media 服务使用术语“单播”和“多播”。

(1)单播

单播是客户端与服务器之间的点到点连接。“点到点”指每个客户端都从服务器接收远程流。仅当客户端发出请求时,才发送单播流。

可通过点播或广播两种方式之一向客户端发布单播流。

(2) 多播

多播是通过“启用多播网络”传递的内容流,网络中的所有客户端共享同一流。以这种方式将 ASF 内容转化为流的最大好处是可以节省网络带宽。

通过将 Windows Media 服务器安装到用户的网络的每一部分,可将多播扩展到网络中没有启动多播的区域,称为“服务器分发”。作为来自服务器的多播的一部分,可以分发该多播的单一的流给网络上其他部分上别的 Windows Media 服务器。服务器随即通过单播或多播提供流给那些网络部分,称为“再分发”。通过将服务器连接起来的方式,可以克服路由器不允许使用多播的问题。这种模式同样适用于通过防火墙。

Windows Media 服务器管理员必须创建 3 个项目以支持多播:广播站、节目和流。“广播站”充当想要连接流的客户的引用点,“节目”组织将要通过广播站广播的内容,“流”是实际内容。所有这 3 个项目都建立后,Windows Media 管理器会创建一个 ASX 文件,链接客户到正确的广播站的 IP 地址,此文件也称为一个“通知”。用户可以由 Web 网页链接到该通知文件,将其放置到网络上的公共共享点,或通过电子邮件将其发送给客户。

2) 点播流与广播流

Windows Media 服务使用术语“点播”和“广播”以描述客户端/服务器的关系。点播流提供给用户主动的方式以控制播放,而广播流用户是被动的。

(1) 点播流概述

“点播”是用户由 Windows Media 服务器接收流信息的一种方式。点播连接是客户端与服务器之间的主动的连接。在点播连接中,用户通过选择内容项目来初始化客户端连接。内容以 ASF 流从服务器传到客户端。若文件已被编入索引,则用户可以开始、停止、后退、快进或暂停流。点播连接提供了对流的最大控制,但这种方式由于每个客户端各自连接服务器,会迅速用完网络带宽。点播单播如图 4.1 所示。

点播单播的一个示例是当用户申请 ASF 文件时,客户端连接到服务器以接收特定内容,而该内容也只传往一个客户端。客户端使用服务器名和 ASF 文件名识别 ASF 文件的 URL。

(2) 广播流概述

广播指的是用户被动接收流。在广播过程中,客户端接收流,但不能控制流。例如,用户不能暂停、快进或后退该流。共有单播和多播两类广播,两类都是被动的。

①广播单播:在广播单播中,客户端通过发布点上的别名访问流。用户可单击 Web 网页上的链接或获得该别名的 URL,从而连接到流。每个连接到流的用户都有其自己的连接和来自服务器的流,访问方式如图 4.2 所示。

例如 Windows Media 编码器向 Windows Media 服务器上的广播发布点发送一个

图 4.1　点播单播

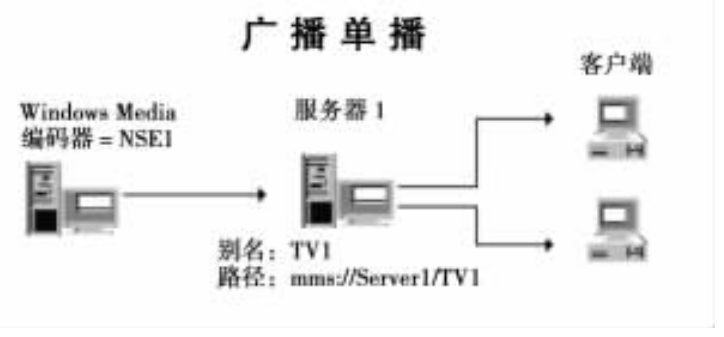


图 4.2　广播单播访问方式

内容流。所有发布点都有一个别名，此例中广播发布点的别名“ TV1 ”。客户端使用 TV1 以确定流源路径。因为发布点能辨认电视台的 ASF 流，引用 ASF 流的 URL 与存储的 . asf 文件相类似。然而，用户并不是识别 ASF 文件名，而是识别发布点别名——TV1。在此示例中，用户使用以下路径以访问该流：mms：//Server1/TV1。

②广播多播 在广播多播中，被动的用户通过监视特定的 IP 地址接收多播 ASF 流（与以特定频率从收音机或电视台接收信号类似）。多播的优点为一个流通过网络可以提供 ASF 内容给许多客户端，这可节省网络带宽，对低带宽局域网尤其有用。广播多播访问方式如图 4.3 所示。

若要在网上使用多播，网络路由器必须支持多播。不过，无论用户的网络路由器是否支持多播，都可以使用 Windows Media 服务在局域网的本地部分进行多播。

3）发布点与广播站内容

Windows Media 服务使用术语“ 点播站 ”和“ 广播站 ”描述服务器如何提供内容给客户端。

点播站用来访问单播内容。发布点是存储允许服务器上的客户端能访问的内容的虚拟目录。在 Windows Media 管理器中，有两种类型的发布点，可以使用其中任何一种为客户端提供内容：

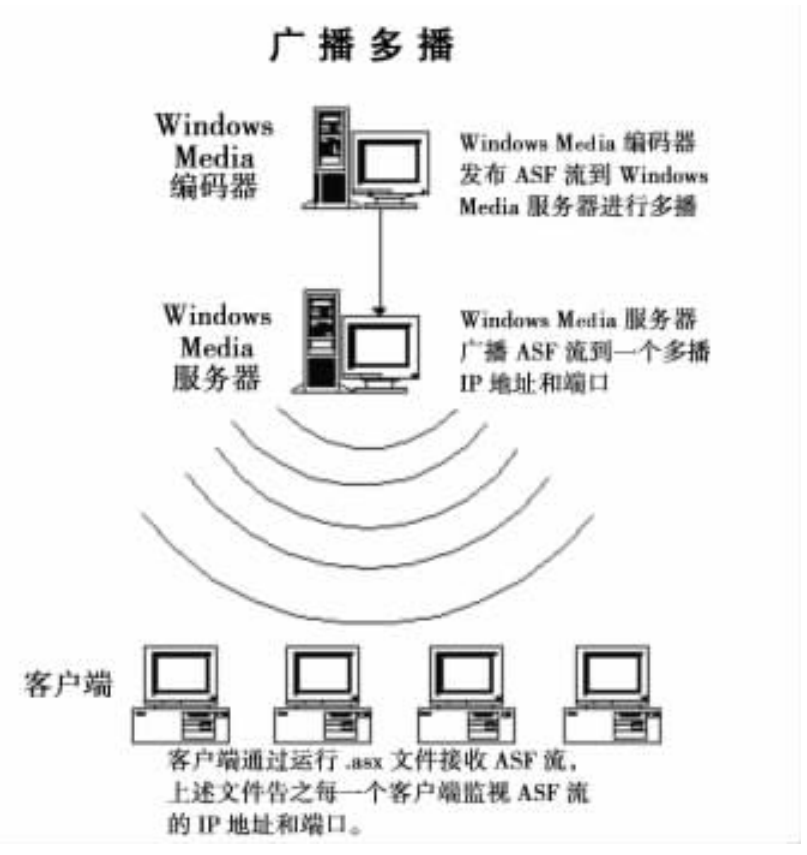


图 4.3　广播多播

①“点播单播发布点”用来提供 ASF 文件。

②“广播单播发布点”用来提供实况 ASF 流。

在 Windows Media 管理器中，点播发布点与广播发布点是分开管理的，如图 4.4 所示。

在创建发布点时，可以设置在给定时间访问发布点的客户端数目限额，以及该发布点可使用的带宽数。对于管理发布点占用的带宽数而言，此类限制很有用。若发现对某个发布点的需求极大，可考虑通过多播提供其内容以节省带宽。

广播站用来访问多播内容。一个广播站至少包括一个节目和流，不相关联的节目和流的广播站是没有内容的。因为广播站是多播的，流是由所有想访问它的客户端共享，客户端数目对网络和流无影响。所以，没有必要限制访问该流的客户端数目。

图 4.4 Windows Media 管理器中点播发布点

4.2 单播控制

单播是客户端与服务器之间的点到点连接。在 Windows Media Services SDK 中主要是通过 Nsunimgr. ocx 文件来完成的，这个文件包括两个控件：单播管理控件和单播跟踪显示控件。

其中，单播管理控件主要完成对单播服务器的管理和操作，单播跟踪显示控件主要完成对服务器的监视工作。单播管理控件在窗体中不可见，而单播跟踪显示控件如图 4.5 所示。

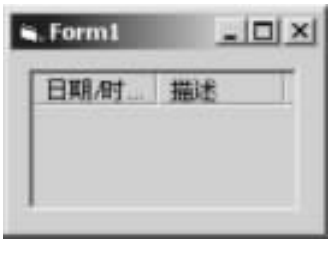


图 4.5 单播跟踪显示控件

在 Windows Media 管理器中，单播跟踪显示控件的应用如图 4.6 所示。

图 4.6　单播跟踪显示控件在 Windows Media 管理器中的应用

4.2.1　使用单播管理控件监控服务器

本小节主要使用单播管理控件来读取 Windows Media Service 的各项参数，用以演示单播管理控件的使用和操作（本例见配书光盘/Sample/Sample4_1 目录下）。其步骤如下：

（1）第一步：连接服务器

用 NSUnicastMgr1. Connect"服务器名"语句连接服务器。设置单播管理控件所管理的服务器，必须连接至服务器以后，才可以进行相关操作。

连接非本机的服务器，必须拥有相应的操作权限。

（2）第二步：显示相关属性

本例中的显示相关属性使用 ShowProperties 过程来完成，如下所示：

```
Sub ShowProperties()
    On Error Resume Next
    // 最大用户连接数显示
    If NSUnicastMgr1.MaxClients = -1 Then
        MaxClientsBox.Text = "No Limit"
    Else：MaxClientsBox.Text = NSUnicastMgr1.MaxClients
    End If

    // 显示最大带宽
MaxBandwidthBox.Text = KbpsLimit(NSUnicastMgr1.MaxBandWidth)
// 显示传输文件比特率
    MaxFileBitRateBox.Text = KbpsLimit(NSUnicastMgr1.MaxFileBitRate)
    // 显示所过滤的事件
    ServerEventsBox.Text = EventsBox(NSUnicastMgr1.ServerEvents)
    AdminEventsBox.Text = EventsBox(NSUnicastMgr1.AdminEvents)
    AlertEventsBox.Text = EventsBox(NSUnicastMgr1.AlertEvents)
    ClientEventsBox.Text = EventsBox(NSUnicastMgr1.ClientEvents)
End Sub
```

(3)第三步 事件触发

单播管理控件可以捕获的事件如表 4.1 所示。

表 4.1 单播管理控件事件

服务器事件	触发条件
OnEventBacklogReached	事件队列满时触发
OnServerFault *	服务器出现异常状况时触发
OnServerOffine	服务器未连接网络时触发
OnServerOnline	服务器联网时触发
管理功能事件	**触发条件**
OnAdminBandwidthLimit	当最大带宽限制改变时触发
OnAdminClientLimit	当客户端连接限制改变时触发
OnAdminFileBitRateLimit	当文件比特率限制改变时触发
OnAdminKillClient	当客户端被使用的 KillClient 方法中断时触发
客户事件	**触发条件**
OnClientConnect	客户端连接时触发
OnClientDisconnect	客户端断开时触发

续表

服务器事件	触发条件
OnClientPlay	客户端开始播放时触发
ONClientStop	客户端停止播放时触发
OnClientStride	客户端定位文件时触发(拖动滑动条)
警告事件	**触发条件**
OnMaxBandWidth	超过最大带宽时触发
OnMaxClients	超过最大设置用户数时触发
OnMaxFileBitRate	超过最大文件比特率时触发

本例中,当服务器有客户端连接时,需要将连接客户端的数量重新修改。程序中将用到 OnClientConnect 事件,代码如下:

```
// 客户端连接触发
Private Sub NSUnicastMgr1_OnClientConnect(ByVal datetime As Date, _
ByVal hr As Long, ByVal nClientid As Long, _
ByVal strIpAddress As String, ByVal port As Integer)
// 当客户端连接触发
EnumerateClients  // 更新客户端数量显示
End Sub
// 客户端中断触发
Private Sub NSUnicastMgr1_OnClientDisconnect(ByVal datetime As Date, ByVal hr As Long, ByVal nClientid As Long, ByVal strIpAddress As String, ByVal port As Integer)
    EnumerateClients  // 更新客户端数量显示
End Sub
```

其中 EnumerateClients 过程用来更新界面中客户端的数量显示,详细代码可以参照配套光盘中的代码。

4.2.2　使用单播管理控件管理单播

表 4.2 为单播管理控件属性表。

表 4.2　单播管理控件属性表

属　性	访　问	解　释
AdminEvents	可读/可写	打开或关闭管理事件
AlertEvents	可读/可写	打开或关闭警告事件
AuthenticationPlugins	只读	返回当前认证插件集合

续表

属　性	访　问	解　释
ClinetEvents	可读/可写	打开或关闭当前客户的事件
Clients	只读	返回当前所有连接用户
EnableHttpStreaming	可读/可写	打开或关闭是否使用 HTTP 分发流
EnableLogging	可读/可写	打开或关闭日志功能
EnableProxy	可读/可写	打开或关闭代理功能
EventPlugins	只读	返回事件插件集合
EventsProxy	可读/可写	返回或设置代理服务器
LogFileDirectory	可读/可写	返回或设置日志文件的目录
LogFilePeriod	可读/可写	返回或设置日志时间段
LogFileSize	可读/可写	返回日志文件最大尺寸
MaxBandWidth	可读/可写	返回或设置带宽限制
MaxClients	可读/可写	返回或设置最大用户连接数
MaxFileBitRate	可读/可写	返回或设置最大文件比特率
ProxyHostName	可读/可写	返回代理服务器主机名
ProxyUser	只读	返回代理用户 ID
Server	只读	返回当前连接的服务器名
ServerEvents	可读/可写	读取或设置服务器事件
ServerStatus	只读	返回服务器当前状态
VirtualRoots	只读	当前虚拟目录集合

表 4.3 为单播管理控件表。

4.3　单播管理控件所支持方法

方 法 名	解　释
Connect	连接运行了 Windows Media Services 的机器
KillClient	中断客户与服务器的连接
SetProxyAuthenticationInfo	设置 HTTP 代理证书

4.2.3　单播管理控件对象

单播管理控件中包括了如表 4.4 所示的单播管理对象信息，用以更完全地控制 Windows Media Services，可以使用这些对象来控制验证插件、客户端、事件、虚拟目录的信息。

表 4.4　单播管理对象信息

对　象	解　释
AuthenticationPlugin	AuthenticationPlugin 对象集合
AuthenticationPlugin	验证插件 Authentication 信息
Clients	Client 对象集合
Client	Client 信息
EventPlugin	EventPlugin 对象集合
EventPlugins	验证/认证信息
VirtualRoot	VirtualRoot 对象集合
VirtualRoot	虚拟目录信息

对象与单播控件的关系如图 4.7 所示。

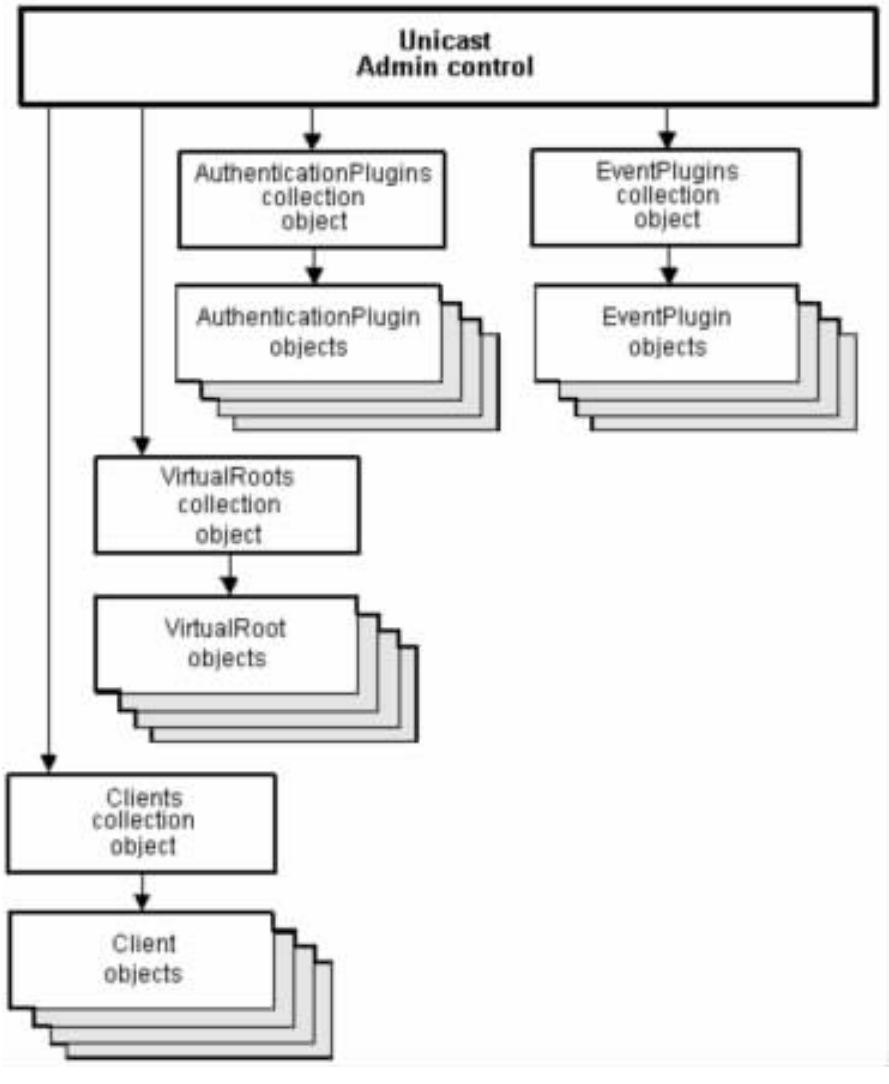


图 4.7　单播管理控件

可以在程序中使用这些对象,见后面的实例。

1)发布点对象以及发布点集合

单播发布点集合(VirtualRoots)属性如表 4.5 所示。

表 4.5 单播发布点集合(VirtualRoots)属性

属 性	访问权限	解 释
Count	只读	当前集合中所含有的对象数量
Item	只读	操作集合当中的一个对象

单播发布点对象(VirtualRoot)所含方法如表 4.6 所示。

表 4.6 单播发布点对象(VirtualRoot)所含方法

方 法	解 释
Add	添加发布点(虚拟目录)至发布点集合中
Remove	从发布点集合中删除一个发布点
SetAsHome	设置发布点为主发布点

(1)创建一个单播发布点

使用程序创建单播发布点,例如创建一个名为 LiveOpera 的发布点,可以使用户访问 D:\liveopera 中的文件,在 LiveOpera 的发布点之下创建一个名为 Britten 的发布点,用户以 LiveOpera/Britten 来访问,可以使用如下的代码:


```
// 创建发布点
NSUnicastMgr. VirtualRoots. Add "LiveOpera", "D:\liveopera"
// 创建嵌套发布点
NSUnicastMgr. VirtualRoots. Add "LiveOpera/Britten", "D:\britten"
// 创建发布点,发布来自其他服务器的流信息
NSUnicastMgr. VirtualRoots. Add "Distribution", "msbd://server/station"
```

(2)删除一个发布点

```
// 删除一个名为 LiveOpera 的发布点
NSUnicastMgr. VirtualRoots. Remove "LiveOpera"
```

(3) 遍历本机所有的发布点

```
// 显示本机所有发布点的路径
NSUnicastMgr. Connect "LocalHost"
For i = 0 to NSUnicastMgr. VirtualRoots. Count - 1
    Document. Write NSUnicastMgr. VirtualRoots[i]. DirectoryPath, "<BR>"
Next
```

2) 用户对象与用户对象集合

用户对象集合(Clients)属性如表 4.7 所示。

表 4.7　用户对象集合(Clients)属性

属　性	权　限	解　释
Count	只读	返回集合中所含有的用户对象数量
Item	只读	返回集合中的一个对象

用户对象(Client)属性如表 4.8 所示。

表 4.8　用户对象(Client)属性

属　性	权　限	解　释
ClientID	只读	返回用户的一个特定 ID
Filename	只读	返回用户所访问文件名称
IPAddress	只读	返回用户的 IP 地址
PortID	只读	返回用户的连接端口
Status	只读	返回客户当前状态,1-无状态,2-打开,3-播放中

(1) 显示当前所有连接用户数

```
// 显示所有连接用户
MsgBox (NSUnicastMgr. Clients. Count)
// 显示最后连接用户的端口 ID
MsgBox (NSUnicastMgr. Clients[NSUnicastMgr. Count - 1]. PortID)
```

(2) 阻止特定的 IP 子网访问

```
Sub OutsidersBeGone(subnet)
        For i = 0 to NSUnicastMgr. Clients. Count - 1
            TheClient = NSUnicastMgr. Clients( i )
```

```
            // Check if the client is on the same subnet
            If StripSubnet(TheClient.IpAddress) <> StripSubnet(subnet)) Then
            // Client is on another subnet. Show a descriptive
            // Message about the client
                MsgBox ("Client disconnected:")          & vbCrLf & _
                        " ClientID:" & TheClient.ClientID & vbCrLf & _
                        " Address:" & TheClient.IpAddress & vbCrLf &_
                        " Port:" & TheClient.PortID)
                TheClient.Disconnect
            End If
        Next
End Sub
Function StripSubnet( ip )
// Strips the last octet off an IP address
StripSubnet = Left( IPAddress,( InstrRev(IPAddress,".", -1,1)) - 1)
End Function
```

(3)显示特定的用户 ID

```
// 显示第二个连接用户的 ID
Set SecondClient = NSUnicastMgr.Clients.Item( 1 )
MsgBox (SecondClient.ClientID)
```


(4)显示特定用户的 IP 地址

```
// 显示第二个连接用户的 IP 地址
Set SecondClient = NSUnicastMgr.Clients.Item( 1 )
MsgBox (SecondClient.IPAddress)
```

3)事件插件对象与事件插件对象集合

事件插件(EventPlugin)属性如表 4.9 所示。

表 4.9 事件插件(EventPlugin)属性

属 性	权 限	解 释
Author	只读	The name of the author of the plug-in.
CLSID	只读	The class ID of the plug-in.

续表

属　性	权　限	解　释
Copyright	只读	The copyright information about the plug-in.
Description	只读	The description of the plug-in.
Enabled	可读 可写	An indication that the plug-in is enabled or disabled.
LastError	只读	The last error sent by the plug-in.
Status	只读	The current status of the plug-in.

事件插件(EventPlugins)集合属性如表 4.10 所示。

表 4.10　事件插件(EventPlugins)集合属性

属　性	权　限	解　释
Item	只读	Indicates a specific element of the collection object.
Count	只读	Returns the number of EventPlugin objects in the collection.

显示所有的事件插件信息：

```
For i = 1 to CInt(NSUnicastMgr1. EventPlugins. Count)
  MsgBox (NSUnicastMgr1. EventPlugins. Item[i - 1]. Description)
Next
```


4.2.4　使用单播跟踪显示控件

单播监视控件主要完成对 Windows Media Service 的监视功能,所含方法如表 4.11所示。

表 4.11　单播监视控件方法

方　法	解　释
ClearTrace	清除控件显示内容
PauseTrace	暂停恢复跟踪
SetServer	连接服务器
TraceProperties	调用过滤事件对话框(如图 4.8 所示),通过筛选,可以选择监控的事件

单播跟踪显示控件只有一个事件,如表 4.12 所示。

图 4.8　过滤事件对话框

表 4.12　单播跟踪显示控件事件

事　件	解　释
OnTraceStateChange	当用户清除,恢复,暂停或者恢复跟踪时触发

通过这个方法,就可以捕获当跟踪状态改变时触发的消息。现在使用单播跟踪显示控件来完成对 Windows Media Service 的监视(本例代码见配套光盘 Sample/ Sample4_2)。

```
// 清除当前窗口状态
    Call TraceView1. ClearTrace
// 连接服务器
// Call TraceView1. setServer(serverNameField. Text)
// 设置监控过滤规则
TraceView1. TraceProperties
```

使用 TraceProperties 属性就可以调出图 4.8 的过滤事件对话框进行过滤设置。示例程序只有一个窗体,运行状态如图 4.9 所示。

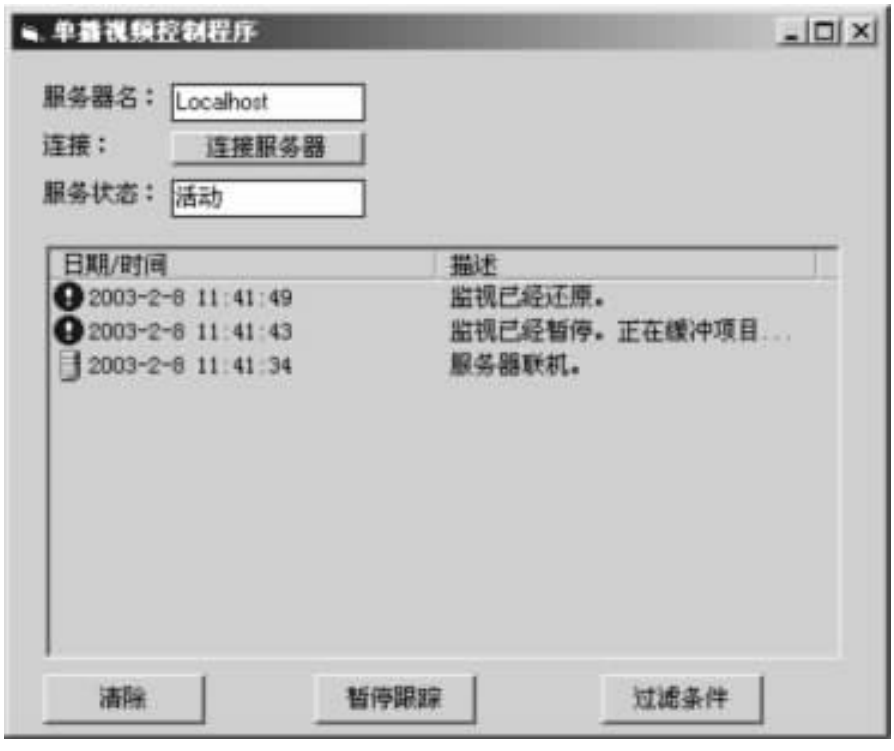


图 4.9　单播视频监控程序运行图

窗体 frmMain 的代码如下所示：


```
Dim pauseState As Long
Sub InitTraceView( )
    Call setServer
End Sub
Sub setServer( )
    Call TraceView1. ClearTrace
    On Error Resume Next
    Call TraceView1. setServer(serverNameField. Text)
    If Err. Number  < > 0 Then
        MsgBox "连接失败！" & Err. Number & vbCrLf & Err. Description
    End If
    If pauseState  = 0 Then
        stateField. Text  = "活动"
    Else
        stateField. Text  = "停止"
    End If
```

```
End Sub
Sub TogglePause( )
    If pauseState = 0 Then
        Call TraceView1. PauseTrace(True)
        PauseButton. Caption = "恢复跟踪"
        pauseState = 1
    Else
        Call TraceView1. PauseTrace(False)
        PauseButton. Caption = "暂停跟踪"
        pauseState = 0
    End If
End Sub
Private Sub ClearButton_Click( )
    TraceView1. ClearTrace      // 清除事件
End Sub
Private Sub FilterButton_Click()
    TraceView1. TraceProperties      // 显示过滤属性
End Sub
Private Sub Form_Load()
    pauseState = 0
End Sub
Private Sub PauseButton_Click()
    TogglePause
End Sub
Private Sub SetServerButton_Click()
    Call setServer
End Sub
// 事件触发
Private Sub TraceView1_OnTraceStateChange(ByVal newState_
As NsoAdminObjectsCtl. TRACESTATE)
    MsgBox "hi"
    If newState = 0 Then
        stateField. Text = "活动"      Else
        stateField. Text = "停止"
    End If
End Sub
```

4.3　多播站控制

Windows Media 服务器组件可以配置为向客户端发送多播流,从而避免使用大量的网络带宽。广播站用来向客户端发送多播流。如果没有广播站,则只能通过单播发送流,这意味着接收 ASF 流的每个客户端都必须连接到服务器。

广播站包含了所有必须的信息,使用该信息可以将 ASF 内容传递到 Microsoft Windows Media Player 或处于合用模式中的其他 Windows Media 服务器。广播站快速启动向导可帮助定义该信息。如果熟悉广播站特性,则可以使用 Windows Media 管理器中的多播广播站页来创建广播站,而不必使用广播站快速启动向导。Windows Media Services SDK 通过 Nschmgr. ocx 来控制多播站,Nschmgr. ocx 中包括一个控件——NSChannelMgrCtl 控件,用于 Windows Media Services 多播流节目控制。

由于多播站操作复杂,让我们从多播站的操作来了解其原理,最后使用 Windows Media Services SDK 来操作它。

4.3.1　多播站创建流程

在开始之前,先讲解创建多播站的流程,并使用 Windows Media Services 管理器来进行此操作。

①在 Windows Media 管理器菜单框架中,单击“多播广播站”。

②显示多播广播站页。

③确保已选择了“使用向导创建新的广播站”复选框,单击“广播站”,然后单击“新建”。

④会弹出“配置和发布单播广播流快速启动向导”对话框。

⑤在“选择广播站”屏幕上,选择“创建新的广播站”。

⑥在“创建新的广播站”屏幕上,键入广播站的名称和说明,然后选择分发模式。

⑦在“指定节目和流名称”屏幕上,在“节目名称”中输入节目的名称,在“流名称”中键入流的名称。也可以激活两个节目选项:

- 如果选择“向导完成后启动节目”,则当广播站创建完毕后将立即启动节目进行浏览。
- 如果选择“完成后重新播放流对象(循环)”,将重复播放已经启动的节目。

⑧快速启动向导提供了几个屏幕,可以在这几个屏幕中为广播站配置原始流。

⑨在“指定流对象来源”屏幕上,选择流的来源。

⑩在“指定流对象来源 URL”屏幕上,输入流来源的 URL。

⑪在“指定流格式信息”屏幕上,指定流来源的路径,以便将流格式信息添加到广播站定义中。

⑫将多播广播站信息文件(. nsc 文件)存储到 Microsoft Windows Media Player 可

以访问的位置。

⑬在“导出广播站信息文件的路径”屏幕上，输入 .nsc 文件存储位置的路径。

⑭在“广播站信息文件 URL”屏幕上，指定 URL 或 UNC 路径，Windows Media Player 将使用它来访问已保存的 .nsc 文件。

⑮在“选择发布方法”屏幕上，选择用来发布流的方法。有关发布方法的信息，可参阅相关的帮助文档。

⑯在“准备发布”屏幕中，复查已选择的选项列表。如果要编辑任何选项，可单击“后退”返回到相应的广播站信息。

⑰如果不准备在电子邮件消息中向用户发送 ASX 文件，将它保存到一个可访问的目录。将所选发布方法创建的所有 .htm 文件保存到用户可以直接或通过 Web 服务器访问的目录中。

这样，一个多播站就创建成功了，可以在本机尝试访问，如果通过局域网访问，需要交换机支持。

在“发布完毕”屏幕上，可以测试广播站发送的流。如果要测试广播站，可单击“测试 .asx”、“测试 .htm w/ <HREF>”或“测试 .htm <OBJECT>”，开始在 Windows Media Player 中播放由广播站发送的内容流。

4.3.2 监视 Windows Media Services 的多播站

通过上一小节了解到如何使用 Windows Media 管理器建立一个多播站，本节将使用 Windows Media Services SDK 建立一个监视程序，用以查看当前服务器。在程序中要建立对 Windows Media Station Control 的引用，如图 4.10 所示。

本例主要完成对指定的 Windows Media Service 监控多播站信息与多播站中的多播流信息，本例代码见配书光盘 Sample/Sample4_3 目录下，软件界面如图 4.11 所示。

第 1 步 连接服务器。

NSChannelMgr1.Connect MCMServerName.Text 用于连接服务器，程序中使用了 Connect 方法。

可以使用如下代码连接服务器：

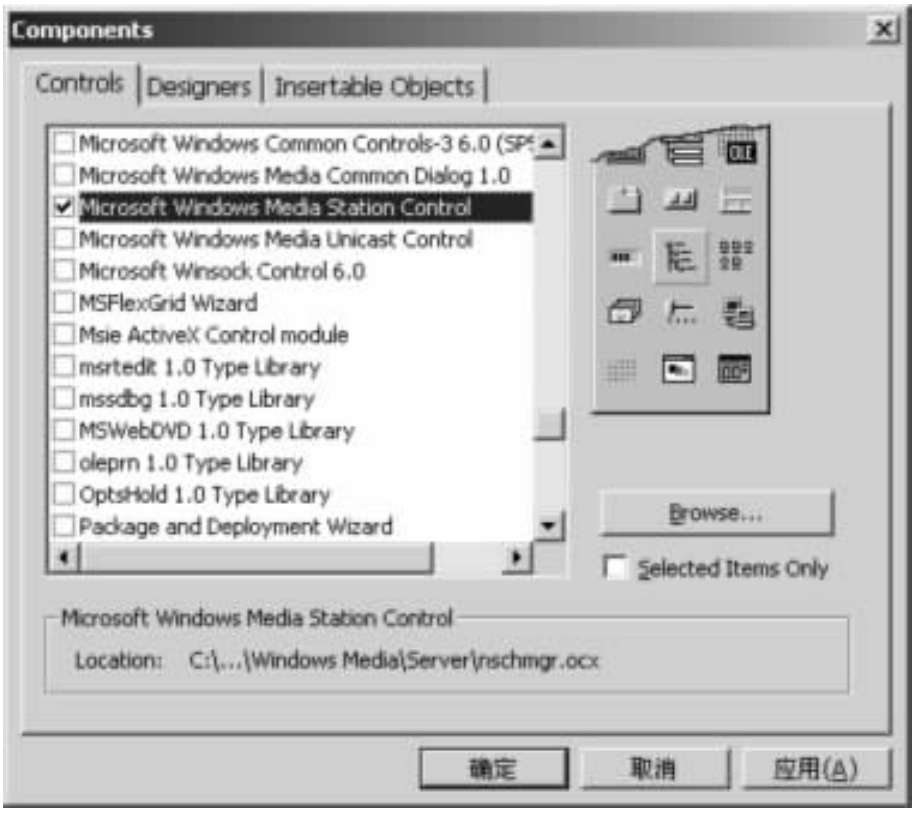


图 4.10　引用多播站控件 nschmgr. ocx

图 4.11　多播站信息监视

```
On error resume next
NSChannelMgr1. Connect "LocalHost"
If Err. Number = 0 then
  MsgBox ("连接至 LocalHost. ")
Else
  MsgBox ("连接失败：" & Err. Description &" . ")
End If
```

第 2 步 :读取服务器信息。

本例中定制了 EnumStreams **过程来更新界面的显示,代码如下：**

```
Sub EnumStreams()
    On Error Resume Next
    StreamCount = CInt(NSChannelMgr1. Streams. Count)
    If StreamCount < 1 Then StreamCount = "No"
    StreamBox. Text = StreamCount & "多播流" & vbCrLf & vbCrLf
    For i = 0 To NSChannelMgr1. Streams. Count - 1
      StreamBox. Text = StreamBox. Text & " Alias：" & NSChannelMgr1.
Streams(i). Alias
      StreamBox. Text = StreamBox. Text & " ID：" & NSChannelMgr1.
Streams(i). GetId
            StreamState = CInt(NSChannelMgr1. Streams(i). GetState)
            Select Case StreamState
                Case 1
                    StreamState = "关闭"
                Case 2
                    StreamState = "打开"
                Case 3
                    StreamState = "播放中"
                Case Else
                    StreamState = "未知"
            End Select
            StreamBox. Text = StreamBox. Text & "- 状态：" & StreamState
            StreamBox. Text = StreamBox. Text & vbCrLf
    Next
End Sub
```


这里使用了 Streams **对象集合来读取每一个流对象的状态,对于** Stream **对象,在后面章节将做详细的介绍。**

第 3 步:多播流改变时更新界面。

在服务器的状态改变后,触发相关事件,更新软件的界面显示,代码如下:

```
// 当一个流开始的时候被触发
Private Sub NSChannelMgr1_OnStreamStart(ByVal pStm As Object)
    EnumStreams
End Sub
// 当流结束时出发
Private Sub NSChannelMgr1_OnStreamStop(ByVal pStm As Object, ByVal hr As
Long)
    EnumStreams
End Sub
```

4.3.3　创建多播站

本例将使用 Windows Service SDK **创建多播站,本例代码见** Sample/Sample4_5 **目录下,相关代码如下:**

```
Private Sub Command1_Click( )
// 创建多播站 MC1,设置多播文件为 c:\asfroot\welcome2.asf
// 创建多播文件 hello.nsc
    With stations
        .Channels.Open "MC1", MCM_PA_ALL_ACCESS
        .Channels.Find("MC1").MulticastAddress = "237.1.1.1"
        .Channels.Find("MC1").Port = 1000      // 指定端口
        .Channels.Find("MC1").DeliveryMode = MCM_DM_ALL
        .Channels.Find("MC1").ChannelFormats.Add "c:\asfroot\welcome2.
asf", MCM_ASF_CH_FORMAT_FILE
        .Channels.Find("MC1").WriteNSC "c:\inetpub\wwwroot\Hello.nsc",
True
    End With
// 创建一个名为"Hello"的流,并与 MC1 站绑定,开始播放
    Set st = stations.Streams.Open("mms://localhost/welcome2.asf", "Hello",
MCM_PA_ALL_ACCESS)
    stations.Channels.Find("MC1").SetActiveStream st
    stations.Streams.Find("Hello").Start 0
End Sub
Private Sub Command2_Click( )
// 关闭多播站与流
```

```
    stations. Channels. Find("MC1"). Close
    stations. Streams. Find("Hello"). Close
    MsgBox "多播站删除. "
End Sub
Private Sub Form_Load( )
// 连接 Windows Media 服务器
    stations. Connect "localhost"
End Sub
```

这样,就建立了一个有基本功能的多播站,而且可以自由控制添加和删除这个多播站。这段代码虽然看起来很短,但非常经典,在此基础上可以作很多的设计。

4.3.4 深入了解多播站控件

多播站控件支持很多的属性、方法以及事件,同时也包含很多的对象。多播站控件属性如表 4.13 所示。

表 4.13 多播站控件属性

属 性	权 限	解 释
Channels	只读	返回多播站对象集合
Clients	只读	返回客户对象集合
EnableHttpStreaming	可读/可写	打开或关闭 HTTP 流
EnableProxy	可读/可写	打开或关闭代理服务器
GetPlaySendDelta	只读	取得播放时间与接收时间的事件差
HostAddresses	只读	获取主机地址对象集合
ProxyHostName	可读/可写	返回或设置代理服务器名
ProxyUser	只读	返回代理服务器验证的用户 ID
Server	只读	返回当前服务器名
Streams	只读	获取流对象集合

(1)使用 Channels 属性

```
NSChannelMgr. Connect "LocalHost"
Dim MyStations
Set MyStations = NSChannelMgr. Channels
MsgBox ("共" & MyStations. Count & " 多播站对象. ")
MyStations(0). Close    // 删除第一个多播站
Set MyStations = NSChannelMgr. Channels    // 更新多播站集合
```

当一个多播站集合改变了其中一个,要使用 Set 命令重新获得多播站集合。

(2)使用 Clients 属性

使用 Clients 删除一个访问用户。

```
NSChannelMgr. Connect "LocalHost"
Dim MyClients
Set MyClients = NSChannelMgr. Clients
MsgBox ("共" & MyClients. Count & " Client 对象. ")
MyClients(0). Close    // 删除第一个客户
Set MyClients = NSChannelMgr. Clients    // 更新 Clinets 集合
```

(3)使用 HttpStreaming 属性

判断当前 HttpStreaming 的状态,查看服务器是否使用 HTTP 方式分发流。

```
Sub ToggleHttpStreaming
   // Turns HTTP streaming on or off
   If NSChannelMgr. EnableHttpStreaming = True Then
      NSChannelMgr. EnableHttpStreaming = False
   Else
      NSChannelMgr. EnableHttpStreaming = True
   End If
   MsgBox ("HttpStreaming value :" & NSChannelMgr. EnableHttpStreaming)
End Sub
```

(4)获得当前主机地址集合

```
NSChannelMgr. Connect "LocalHost"
Dim Addresses
Set Addresses = NSChannelMgr. HostAddresses
MsgBox ("共" & Addresses. Count & "个主机地址对象. ")
```

通过 Count 属性可以返回总共数量。

(5)使用 Streams 属性

使用 Streams 属性删除第一个流。

```
NSChannelMgr. Connect "LocalHost"
Dim MyStreams
Set MyStreams = NSChannelMgr. Streams
MsgBox ("共" & MyStreams. Count & "流对象")
MyStreams(0). Close  // 删除第一个流
Set MyStreams = NSChannelMgr. Streams  // 更新集合
```

当一个流集合改变了其中一个,要使用 Set 命令重新获得流集合。

多播站控件所支持的方法如表 4.14 所示。

表 4.14 多播站控件所支持的方法

方 法	解 释
Connect	连接至运行有 Windows Media Service 的服务器
SetProxyAuthenticationInfo	设置 HTTP 代理服务器认证

多播站控件所支持的事件如表 4.15 所示。

表 4.15 多播站控件所支持的事件

事 件	触发条件
OnChannelClose	多播站关闭时触发
OnChannelOpen	多播站开始时触发
OnChannelPropertyChange	多播站属性更改时触发
OnChannelStreamChange	多播站流更改时触发
OnClientConnect	客户连接时触发
OnClientDisconnect	客户中断时触发
OnStreamArchireClosed	档案文件关闭时触发
OnStreamClose	流结束时触发
OnStreamOpen	流打开时触发
OnStreamPropertyChange	流的属性更改时触发
OnStreamStrat	流开始时触发
OnStreamStop	流停止时触发

事件的捕获非常重要,当 Windows Media Services 发生错误、增加了用户,设置改变的时候将触发相应事件,以下是捕获条件的实例:

(1)捕获 OnChannelClose 事件

```
Sub NSChannelMgr_OnChannelClose(TheStation, Desc, ID)
  MsgBox("多播站 " & TheStation.Name & "关闭.")
End Sub
```

(2)捕获 OnChannelOpen 事件

```
Sub NSChannelMgr_OnChannelOpen( TheStation )
  MsgBox("多播站" & TheStation.Name & "打开.")
End Sub
```

(3)捕获 OnChannelStreamChange 事件

```
Sub NSChannelMgr_OnChannelStreamChange( MyStation, _
OldStream, NewStream )
    Str = // 多播站" & MyStation.Name & "停止播放. "
    Str = Str & OldStream.Alias "已经开始播放" & NewStream.Alias
    MsgBox(Str)
End Sub
```

(4)捕获 OnClientConnect 事件

当有客户连接时触发,显示客户连接 IP。

```
Sub NSChannelMgr_OnClientConnect( Client )
  MsgBox("地址为 " & Client.Address & "的客户连接.")
End Sub
```

(5)捕获 OnStreamOpen 事件

```
Sub NSChannelMgr_OnStreamOpen( TheStream )
  MsgBox("流 " & TheStream.Source & "被打开.")
End Sub
```

(6) 捕获 OnStreamStart 事件

```
Sub NSChannelMgr_OnStreamStart( Stream )
  MsgBox (Stream. Desc & "已经开始. ")
End Sub
```

(7) 捕获 OnStreamStop 事件

当流停止时触发

```
Sub NSChannelMgr_OnStreamStop( Stream, Hresult )
  If hresult  > = 0 Then
    MsgBox (Stream. Alias & "正常停止")
  Else
    MsgBox (Stream. Alias & "停止失败")
  End If
End Sub
```

4.3.5 多播站控件包含对象

多播站控件中还包含了很多对象,可以更加灵活地控制多播站,这些对象包括地址对象、多播站对象、多播站格式对象、客户对象、流信息对象、多播站格式流信息对象等。也可以说,正是这些对象的有效组合,构成了多播站,其控件对象如表 4.16 所示。

表 4.16 多播站控件包含对象

对 象	解 释
Address	地址信息对象
Channel	多播站频道对象
Channels	多播站频道对象集合
ChannelFormat	多播站格式信息对象
ChannelFormats	多播站格式信息对象集合
Client	客户信息
Clients	客户信息对象集合
HostAddresses	地址信息对象集合
Stream	流信息对象
Streams	流信息对象集合
StreamDescriptor	多播站格式流信息对象
StreamDescriptors	多播站格式流信息对象集合

1)地址信息对象与地址信息对象集合

地址信息对象集合 HostAddresses 是 Address 对象的集合,它由若干个 Address 对象构成,包含两个属性,如表 4.17 所示。

表 4.17　地址信息对象集合属性

属　性	访　问	解　释
Item	只读	返回地址对象 Address object
Count	只读	返回地址对象 Address 的数量

地址信息对象拥有属性如表 4.18 所示。

表 4.18　地址信息对象属性

属　性	访　问	解　释
Address	只读	返回地址对象的地址
Port	只读	返回地址对象的端口
Type	只读	返回地址对象 socket 的类型

用示例显示所有连接地址的类型、端口和地址,代码如下:

```
NSChannelMgr. Connect "LocalHost"
Dim Hosts as Address
Set Hosts = NSChannelMgr. HostAddresses
For i = 0 to Hosts. Count - 1
    str = "Host #" & i & vbcrlf
    str = str + "类型:" & Hosts. Item(i). Type & vbcrlf
    str = str + "端口" & Hosts. Item(i). Port & vbcrlf
    str = str + "地址" & Hosts. Item(i). Address & vbcrlf
msgbox str
Next I
```

通过以上代码,可以遍历所有连接至本服务器的地址类型、端口、地址信息。

2)多播站频道对象与频道对象集合

多播站频道对象在 Windows Media Services SDK 中非常重要,可以这样来理解:一个. nsc 文件就是一个多播站频道对象,多个多播站频道对象的集合构成多播站频道集合。

Channel 对象包含多个属性,如表 4.19 所示。

表 4.19 Channel 对象的属性

属 性	权 限	解 释
AutoArchive	可读/可写	Returns or sets a Boolean value that determines whether content is automatically cached.
AutoArchiveDir	可读/可写	Returns or sets the String value containing the directory where the automatic content cache is stored.
AutoArchiveSize	可读/可写	Returns or sets the Long value of the size, in kilobytes, of the automatic content cache.
ChannelFormats	只读	Returns a Channels Formats collection object.
ContactAddress	可读/可写	Returns or sets the contact mailing address of the station.
ContactEmail	可读/可写	Returns or sets the contact e-mail address of the station.
ContactPhone	可读/可写	Returns or sets the contact phone number of the station.
DeliverMode	可读/可写	Returns or sets the delivery mode of the station.
Description	可读/可写	Returns or sets the description of the station.
MulticastAddress	可读/可写	Returns or sets the multicast address of the station.
MulticastAdapterAddress	可读/可写	Returns or sets the address of the network card used by the station.
MulticastTimeToLive	可读/可写	Returns or sets the multicast time-to-live (TTL) value of the station.
Name	可读/可写	Returns or sets the name of the station.
NSCURL	可读/可写	Returns or sets the Uniform Resource Locator (URL) for the Windows Media Station (.nsc) file of the station.
Port	可读/可写	Returns or sets the port of the station.
Version	只读	Returns the version of the station.
DistributionLimit	可读/可写	Returns or sets the maximum number of distribution points.
LogURL	可读/可写	Returns or sets the URL of the server log file.
UnicastURL	可读/可写	Returns or sets the URL of the unicast publishing point.

Channel 对象的属性就是每一个多播站频道对象所具备的属性。例如使用 Port 属性,可以读取设置多播站频道的多播端口,代码如下:

```
// 设置第二个多播站端口
Set MyStation = NSChannelMgr.Channels.Item(1)
MyStation.Port = 64288
MsgBox("MyStation address:" & MyStation.Port)
```

使用 MulticastAddress 属性,可以设置多播站频道的多播地址,代码如下:

```
// Get the second station
Set MyStation = NSChannelMgr.Channels.Item(1)
MyStation.MulticastAddress = "239.2.254.100"
MsgBox("MyStation address:" & MyStation.MulticastAddress)
```

使用 MulticastTimeToLive 属性可以设置多播数据包的生存时间。一般的生存事件的规定如表 4.20 所示：

表 4.20　多播数据包的生存时间

选项	值
本地网络	1
内部网	32
互联网	128

代码如下：

```
// 设置第二个多播频道的生存时间
Set MyStation = NSChannelMgr.Channels.Item(1)
MyStation.MulticastTimeToLive = 32
MsgBox("Multicast 数据包生存时间为:" & _
MyStation.MulticastTimeToLive & "在子网中.")
```

使用 NSCURL 属性,可以读取和设置.nsc 文件的地址。以下代码使用了 Find 方法,用以查找名为 test1 的多播频道。

```
// 查找名为 test1 的多播频道
Set MyStation = NSChannelMgr.Channels.Find("test1")
// 从新设置 nsc 文件地址
MyStation.NSCURL = "http://www.liumeiti.com/test.nsc"
```


使用 Name 属性,可以读取设置多播站的名称,示例代码如下：

```
// 更改名称为 test1 的多播站名为 test2
Set MyStation = NSChannelMgr.Channels.Find("test1")
MyStation.Name = "test2"
```

以上是 Channel 对象的一些常用属性的程序示例,详细的信息可参考相关的帮助文档。

Channel 对象包含的方法如表 4.21 所示。

表 4.21　Channel 对象包含的方法

方 法 名	解　释
Close	Closes a station.
DistributeFrom	Expedites setting up distribution of content from a remote multicast manager.
GetActiveStream	Gets the active stream object.
GetID	Gets a station ID.

续表

方 法 名	解　释
InitializeFromNSC	Initializes the station from an . nsc file.
SetActiveStream	Assigns a Stream object to a station.
WriteNSC	Writes station information to an . nsc file.

在 Channel 的方法中，最常用的就是 Close 方法，该方法用来关闭一个多播频道，示例代码如下：

```
Dim Stations
Set Stations = NSChannelMgr. Channels
Stations. Open "Station 1", 1    // Give read access
Stations. Open "Station 2", 2    // Give write access
Stations. Open "Station 3", 7    // Give full access
Stations(1). Close          // Close Station 2
// Now set the collection again
Set Stations = NSChannelMgr. Channels
```

Channels 对象的属性如表 4. 22 所示。

表 4. 22　Channels 对象属性

属　性	访　问	解　释
Item	只读	Returns the specified Channel object.
Count	只读	Returns the number of Channel objects.

Channels 对象是 Channel 对象的集合，包含两个属性：Item 属性，返回一个 Channel 对象；Count 属性，返回当前集合中包含的 Channel 对象数目。示例代码如下：

```
NSChannelMgr. Connect "LocalHost"
Dim Stations
Set Stations = NSChannelMgr. Channels
LastStation = Stations. Item( Stations. Count - 1 )
FirstStation = Stations. Item( 0 )
```

以上代码返回当前服务器中第一个多播频道与最后一个多播频道。

Channels 对象的方法如表 4. 23 所示。

表 4. 23　Channels 对象的方法

Methods	Description
Find	Searches for the specified Channel object.
Open	Opens a Channel object.

Channels 对象的 Find 方法用来查找一个 Channel 对象,并且返回对象;Open 方法用来打开一个 Channel 对象。

Find 方法的示例代码如下:

```
NSChannelMgr. Connect "LocalHost"
// 查找名字为 test1 的多播频道
Set AStation = NSChannelMgr. Channels. Find( "test1" )
// 设置注释
AStation. Description = "test"
```

Open 方法用来创建一个 Channel 对象,并设置访问权限,代码如下:

```
NSChannelMgr. Connect "LocalHost"
Dim Stations
Set Stations = NSChannelMgr. Channels
// 创建多播站频道 "test",使其他服务器以只读权限访问
Stations. Open "OHlive", 1
```

关于权限的表示代码如表 4. 24 所示。

表 4. 24　权限表示

值	解　释	定　义
1	读	MCM_PA_READ
2	写	MCM_PA_WRITE
4	删除	MCM_PA_DELETE
7	全部	MCM_PA_ALL_ACCESS

3)频道格式对象与频道格式对象集合

频道格式对象的主要功能是定义多播频道的格式,即 ChannelFormat 对象与 ChannelFormats 对象。

ChannelFormats 对象是多个 ChannelFormat 对象的集合。

ChannelFormat 对象属性如表 4. 25 所示。

表 4. 25　ChannelFormat 对象属性

属　性	访　问	解　释
BitRate	只读	Returns the bit rate of the station format.
PacketSize	只读	Returns the packet size of the station format.
StreamDescriptors	只读	Returns a StreamDescriptors collection object.

BitRate 属性返回多播站的格式,示例代码如下:

```
Set MyStation = NSChannelMgr. Channels. Item(0)
Set MyFormat = MyStation. ChannelFormats. Item(1)
MsgBox ("第一个多播站频道的第二个频道格式比特率是" & _
    MyFormat. BitRate)
```

PacketSize 属性返回多播站的数据包大小,示例代码如下:

```
Set MyStation = NSChannelMgr. Channels. Item(0)
Set MyFormat = MyStation. ChannelFormats. Item(1)
MsgBox ("第一个多播站的第二个频道格式的包大小为" & _
        MyFormat. PacketSize)
```

StreamDescriptors 属性返回 StreamDescriptor 对象集合,示例代码如下:

```
Set MyStation = NSChannelMgr. Channels. Item(0)
Set MyFormat = MyStation. ChannelFormats. Item(0)
Set MyDescriptors = MyFormat. StreamDescriptors
```

ChannelFormat 对象方法如表 4.26 所示。

表 4.26 ChannelFormat 对象方法

方 法	注 释
GetDescription	Returns a description of a station format.
GetKey	Returns a unique station format key.
Remove	Removes a station format.

4)客户对象与客户对象集合

每一个访问多播频道的用户,就是一个客户对象,客户对象集合包含多个客户对象,即 Client 与 Clients。

Client 对象属性如表 4.27 所示。

表 4.27 Client 对象属性

属 性	访 问	注 释
Address	只读	Returns the address of the Client object.
ChannelId	只读	Returns the station ID of the Client object.
DeliveryMode	只读	Returns the delivery mode of the Client object.
ID	只读	Returns the ID of the Client object.

Client 对象方法如表 4.28 所示。

表 4.28　Client 对象方法

方　法	注　释
Disconnect	Disconnects a client.

Clients 对象所含属性如表 4.29 所示。

表 4.29　Clients 对象属性

属　性	方　位	注　释
Item	只读	Returns a specified Client object.
Count	只读	Returns the number of Client objects.

Client 对象与 Clients 对象的使用方法与单播管理控件的 Client 与 Clients 大致相同。

5)流对象与流对象集合

流就是指通常的节目流,流对象控制一个节目流的开始和关闭,多个流对象构成了流对象集合,即 Stream 与 Streams。Stream 对象的属性如表 4.30 所示:

表 4.30　Stream 对象属性

属　性	权　限	注　释
Alias	只读	Returns the alias of a stream.
ArchiveFile	可读/可写	Returns or sets the name of the archive file of a Stream object.
ArchiveSize	可读/可写	Returns or sets the size of the archive file of a Stream object.
ContentType	只读	Returns the content type of the stream.
Description	可读/可写	Returns or sets the description of a Stream object.
Seekable	只读	Returns a value determining whether the stream is seekable.
Source	只读	Returns the source of the stream.
Title	可读/可写	Returns or sets the title of a Stream object.

使用 Alias 属性,可以返回流对象的别名,示例代码如下:

```
// 显示所有 Stream 对象的别名
For i = 0 to NSChannelMgr. Streams. Count - 1
    Document. Write NSChannelMgr. Streams(i). Alias & "<BR>"
Next i
```

使用 ArchiveFile 属性,返回或设置存档文件名称。以下代码用来返回当前流对象的存档文件名称:

```
Dim Streams
Set MyStream = NSChannelMgr. Streams. Find( "test" )
MsgBox ("Test 流存档文件为" & MyStream. ArchiveFile)
```

使用 ArchiveSize 属性,返回或设置存档文件尺寸。以下代码用来设置存档文件大小:

```
Dim Streams
Set Streams = NSChannelMgr. Streams
MsgBox ("最大存档文件尺寸为:" & Stream(0). ArchiveSize/)
// 重新设置大小为 5MB
Stream(0). ArchiveSize = 5000
```

使用 ContentType 属性,返回当前流的文件类型(只有为 ASF 才有效),示例代码如下:

```
Dim Streams
Set MyStream = NSChannelMgr. Streams. Find( "Test" )
MsgBox ("流类型为:" & _
    MyStream. ContentType &". ")
```

使用 Description 属性,读取流对象的注释信息,示例代码如下:


```
Dim Streams
Set MyStream = NSChannelMgr. Streams. Find( "Test" )
MsgBox ("Test 流的注释信息为:" & MyStream. Description &". ")
```

使用 Seekable 属性,读取或设置流的开始事件,示例代码如下:

```
MCM. Channels. Open( "MyStation", MCM_PA_ALL_ACCESS )
Dim MyStream
// 以只读的方式打开一个名为 test 的流
Set MyStream = MCM. Streams. Open ("_
mms://nsserver/welcome. asf", _
                "Welcome", MCM_PA_READ)
MyStation. SetActiveStream MyStream
If MyStream. Seekable Then
   MyStream. Start 5000    // 从 5 秒后开始
Else
   MyStream. Start 0       // 正常开始
End If
```

使用 Source 属性,返回或设置流对象的来源,示例代码如下:

```
Dim Streams
Set MyStream = NSChannelMgr. Streams. Find( "Test" )
MsgBox ("Test 流来源于:" & MyStream. Source &". ")
```

使用 Title 属性,返回或设置流对象的标题,示例代码如下:

```
Dim Streams
Set MyStream = NSChannelMgr. Streams. Find( "test" )
MsgBox ("流的标题是:" & MyStream. Title & ". ")
```

Stream 对象的方法如表 4. 31 所示。

表 4. 31　Stream 对象方法

方　法	注　释
Close	Closes a stream.
GetChannel	Gets the station associated with a stream.
GetID	Returns a stream ID.
GetState	Gets the state of a stream object.
Start	Starts a stream.
Stop	Stops a stream.

Close 方法用来关闭一个流,示例代码如下:


```
Set Streams = NSChannelMgr. Streams
Set Stream1 = Streams(0)
// 关闭服务器中第一个流
Stream1. Close
// 重新初始化流集合
Set Streams = NSChannelMgr. Streams
```

GetChannel 方法用来获取当前流所属的多播频道对象,示例代码如下:

```
Dim Streams
Set Streams = NSChannelMgr. Streams
Set Station1 = Stream(0). GetChannel
```

GetID 方法用来返回当前流的 ID,每一个流都有一个特定的 ID,此 ID 为 Long 型变量,示例代码如下:

```
Dim Streams
Set Streams = NSChannelMgr. Streams
Set Station1 = Stream(0). GetChannel
```

GetState 方法用来返回当前流的状态,示例代码如下:

```
Dim Streams
Set Streams = NSChannelMgr. Streams
Select Case Stream(0). GetState
  Case 1 MsgBox (Stream(0). Title & "关闭. ")
  Case 2 MsgBox (Stream(0). Title & "打开. ")
  Case 3 MsgBox (Stream(0). Title & "播放中. ")
  Case 4 MsgBox (Stream(0). Title & "重新开始中. ")
End Select
```

其中状态表示如表 4. 32 所示。

表 4. 32 流状态属性

值	标 识	注 释
1	MCM_SS_CLOSED	关闭
2	MCM_SS_OPEN	打开
3	MCM_SS_STREAMING	播放中
4	MCM_SS_RESTARTING	重新开始

Start 方法与 Stop 方法用来开始或停止一个流,可以参见 Seekable 的属性示例。

Streams 对象的属性如表 4. 33 所示。

表 4. 33 Streams 对象属性

属 性	权 限	注 释
Item	只读	Returns a specified Stream object.
Count	只读	Returns the number of Stream objects.

Item 属性返回集合中特定的 Stream 对象,示例代码如下:

```
NSChannelMgr. Connect "LocalHost"
Dim Streams
Set Streams = NSChannelMgr. Streams
MsgBox (Streams. Item(0). Title)
```

Count 属性返回当前集合中对象的个数,示例代码如下:

```
NSChannelMgr. Connect "LocalHost"
Dim Streams
Set Streams = NSChannelMgr. Streams
Set LastStream = Streams( Streams. Count - 1 )
MsgBox (Streams. Count & " 个流在集合中")
```

Streams 对象方法如表 4. 34 所示。

表 4. 34　Streams 对象方法

方　法	注　释
Open	Opens a Stream object.
Find	Finds a specified Stream object

Open 方法用来打开一个流,示例代码如下:

```
NSChannelMgr. Connect "LocalHost"
MCM. Channels. Open( "MyStation", MCM_PA_ALL_ACCESS )
Dim MyStream
// Open stream "Welcome" with read access
Set MyStream = MCM. Streams. Open ("_
mms ://nsserver/welcome. asf", _
                    "Welcome", MCM_PA_READ)
MyStation. SetActiveStream MyStream
```

Open 方法需要一个必须的参数,就是打开的权限,如表 4. 35 所示。

表 4. 35　流的权限参数

值	标　识	注　释
1	MCM_PA_READ	读
2	MCM_PA_WRITE	写
4	MCM_PA_DELETE	删除
7	MCM_PA_ALL_ACCESS	所有权限

Find 方法用来查找一个流,示例代码如下:

```
NSChannelMgr. Connect "LocalHost"
// 查找一个名字为 Test 的流
Set RolloutStream = NSChannelMgr. Streams. Find( "test")
```

控制篇：Microsoft Windows Media Metafiles

第5章

5.1 Windows Media Metafiles 概述

5.1.1 Windows Media Metafiles 简介

Windows Media Metafiles 是一个基于 XML、包含了媒体信息的文本文件，扩展名通常为：.wax，.wvx，.wmx，.asx 等等。Metafiles 包含很多的元素和标签，每个元素都在 Windows Media Player 中有特定的设置和行为。

使用 Windows Media Metafiles 可以用来创建元文件播放列表，播放列表可以允许自己定义和控制媒体节目。例如，可以使用播放列表来连续调度播放，或可以插入广告和别的媒体片段进行流的无缝切换。表 5.1 为每种 Metafiles 所支持的文件格式，在实际使用过程中要根据自己使用的流媒体文件格式选用不同的 Metafiles。

表 5.1 Metafiles 所支持的文件格式

扩展名	ASF	WMA	WMV
WVX	支持	支持	支持
WAX	支持	支持	不支持
ASX	支持	不支持	不支持

微软提供的文档显示，不同的扩展名支持的文件类型也不同。

5.1.2 使用 Windows Media Metafiles 的原因

对于广大的流媒体爱好者来说，恐怕最熟悉的 Metafiles 就是 ASX 了。当我们访问视频网站时，经常可以在链接属性中看见 ASX 的身影，那么，为什么不把流媒体文件的实际地址直接写在页面链接属性中，而是要使用 Metafiles 呢？有以下两点原因：

①这样做出于一个最基本的目的，就是定向流媒体文件到 Windows Media Player。因为多数的浏览器都试图下载流式媒体文件而不是按照流的方式来播放，在这种情况下使用媒体元文件列表。当点击一个元文件列表时，Windows Media Player 会自动获取包含在列表内的媒体文件的信息（如 URL 等），进行播放。

②对节目地址保密：直接作媒体文件的链接很容易让人知道媒体文件的真实位置，通过使用 ASX 文件可以提高 URL 的保密性。

现在来看，通过 ASX 来保密元文件这个功能极其有限，用户可以将 ASX 文件下载，由于 ASX 文件是 ASCII 码文件，可以由任何一个阅读器查看。

5.2　使用 Windows Media Metafiles

5.2.1　ASX 元文件

ASX 元文件是提供信息的. asx 文件，ASX 元文件或. asx 文件是文本文件，在其最简形式中包含了关于流的 URL 的信息。Windows Media Player 处理该信息，然后打开. asx 文件中定义的内容。Windows Media Player 使用这些信息从 Intranet 或 Internet 接收单播流、多播流以及其他支持的媒体。这些文件由 Windows Media Player 快速加载，并包含用于下列用途的信息：

①将控件从 HTTP 浏览器传送到 Windows Media Player 控件，以便直接将流传递到 Windows Media Player。如果没有 Microsoft Internet Explorer，将无法使用 MMS 协议定义 Web 页中的超级链接，这是因为无法识别 MMS 协议。可以链接到 Web 服务器上的. asx 文件并通过 MMS 协议，将 Windows Media Player 指向 Windows Media 服务器。

②提供一个通知文件，Windows Media Player 可使用该文件访问 Windows Media 广播站上的节目。

提供流的参考说明和 Windows Media Player 用来处理协议翻转的规则。

提供一个播放列表，用来定义由 Windows Media Player 汇集起来的流的播放顺序。

5.2.2　标签介绍

ASX 文件是一个文本文件，它主要的目的是对流信息进行重定向，类似 RPM（RM 的中转文件）文件。在 ASX 中包含了媒体内容对应的 URL，当在 HTML 中让一个 HYPERLINK 与 ASX 联系时，浏览器会直接将 ASX 的内容送给 Windows Media Player，Windows Media Player 会根据 ASX 文件的信息用相应的协议去打开指定位置上的多媒体信息流或多媒体文件。

利用 ASX 文件来重定向流信息的原因主要是：目前通用的浏览器通常均不能直接支持用于播放流信息的协议 MMS，所以我们采用 ASX 文件。采用 ASX 文件以后，当浏览器发现一个连接与 ASX 有关时，它知道需要用 Media Player 来播放流信息，于

是它就会启动 Windows Media Player，就可以用 MMS 协议来播放流信息。

ASX 的标签如表 5.2 所示：

表 5.2 ASX 标签

标 签	用 途
ASX	定义 ASX 文件
AUTHOR	指明作者
BASE	定义发送给播放器的 URL 串
BANNER	显示广告图片
COPYRIGHT	显示版权所有者
DURATION	流文件播放时间的标签
ENDMARKER	停止播放的标签
ENTRY	定义片段
ENTRYREF	连接到 Entry 标签的元素
LOGO	定义标志图像
EVENT	定义事件
MOREINFO	定义可见元素的 URL 地址
PARAM	定义参数标签
PREVIEW DURATION	预览时间
REF	定义流媒体文件的地址
REPEAT	Windows Media Player 重复 Entry 元素的次数
STARTMARKER	标记一个开始位置，Windows Media Player 将从这个地方开始播放
STARTTIME	定义开始时间索引
TITLE	指定 ASX 元素的标题
ABSTRACT	内容的概要

ABSTRACT 标签 描述相关元素，使用此标签可以描述 ASX，ENTRY，BANNER 3 个标签，只需将 ABSTRACT 标签放入 ASX，ENTRY，BANNER 的作用域中即可，示例代码如下：

```
< ASX VERSION = "3.0" >
    < TITLE > Title < TITLE >
    < ABSTRACT >
        描述 1
    < /ABSTRACT >
```

```
< ENTRY >
    < REF HREF = " YourMediaFilename. asf"  / >
    < TITLE > The Title of the Track < /TITLE >
    < ABSTRACT >
        描述 2
    < /ABSTRACT >
< /ENTRY >
< /ASX >
```

第一个 ABSTRACT **标签描述的是** ASX **域，第二个** ABSTRACT **标签描述的是** ENTRY **域。**

ASX **标签：用于定一个源文件，此标签为最上级别标签，其余所有标签都必须写入** ASX **标签域内，示例代码如下：**

```
< ASX VERSION = "3. 0" >
   < ENTRY HREF = " http ：/ / samples. microsoft. com / sample1. ASX  / >
< /ASX >
```

AUTHOR **标签：定义作者，作用于** ASX **与** ENTRY **的作用域下，示例代码如下：**

```
< ASX VERSION = "3. 0" >
   < AUTHOR > forum < /AUTHOR >      < !- -  显示作者  - - >
   < ENTRY >
      < REF HREF = " mms ：/ / localhost / test. asf"  / >
      < AUTHOR > liumeiti < /AUTHOR >    < !- -  本片断的作者  - - >
   < /ENTRY >
< /ASX >
```

BASE **标签：定义基本的** URL**，作用于** ASX **与** ENTRY **域。**

BANNER **标签：定义图形广告，显示在影像区域的下方，点击可以访问指定的网址，作用于** ASX **与** ENTRY **域，**MOREINFO **标签在其中指定网址，示例代码如下：**

```
< BANNER HREF = " http ：/ / www. liumeiti. com / img / ver3 / logo2. gif" >
    < ABSTRACT > abstract < /ABSTRACT >
    < MOREINFO HREF = " http ：/ / www. liumeiti. com"  / >
< /BANNER >
```

COPYRIGHT **标签：定义版权信息，作用于** ASX **与** ENTRY **域，示例代码如下：**

```
< COPYRIGHT > Copyright （c）2003， liumeiti. com  < /COPYRIGHT >
```

DURATION **标签：定义事件的运行时间，作用于** ASX **与** ENTRY **域，示例代码如下：**

```
<ASX version = "3.0" >
    <Title > The Show Title </Title >
    <Entry >
        <Ref href = "test.asf" / >
        <DURATION VALUE = "00:00:03" / >      <!-- 3 seconds -->
    </Entry >
</ASX >
```

在 Windows Media Player 播放 Test.asf 3 s 后自动停止。

ENDMARKER 标签：定义播放媒体文件停止标志，当 Windows Media Player 遇到指定标志时停止播放，示例代码如下：

```
<ASX version = "3.0" >
    <Title > The Show Title </Title >
    <Entry >
        <Ref href = "test.asf" / >
        <ENDMARKER NAME = "marker1" / >
    </Entry >
</ASX >
```


Marker 标签是嵌入媒体文件中的字符串，插入方法参考 Windows Media Index 工具。

ENTRY 标签：是最经常使用的标签，使用此标签可以控制播放的内容，作用于 ASX，EVENT，REPEAT 域，标签有两个参数：

①CLIENTSKIP = "YES" | "NO"

②SKIPIFREF = "YES" | "NO"

这两个参数非常重要，用以控制用户是否可以进行节目切换与节目的检索。示例代码如下：

```
<ASX VERSION = "3.0" >
    <TITLE > test show </TITLE >

    <ENTRY CLIENTSKIP = "NO" SKIPIFREF = "YES" >
        <TITLE > Example Clip </TITLE >
        <REF HREF = "test.asf" / >
    </ENTRY >
```

```
    <ENTRY>
        <TITLE>Another Clip</TITLE>
        <REF HREF="test2.asf" />
    </ENTRY>
</ASX>
```

ENTRYREF 标签：可以引用其他的 ASX 文件，使用 CLIENTBIND 参数可以将本 ASX 文件加入 Windows Media Player 的收藏夹，示例代码如下：

```
<ASX VERSION="3.0">
    <TITLE>Example of Using EntryRef</TITLE>
    <ENTRYREF HREF="http://localhost/m.asx" CLIENTBIND="NO" />
</ASX>
```

EVENT 标签：捕获媒体文件中的脚本标识，作用于 ASX 域，示例代码如下：

```
<ASX VERSION="3.0">
<ENTRY CLIENTSKIP="NO">
    <REF HREF="test.asf" />
</ENTRY>

<EVENT NAME="Adlink" WHENDONE="RESUME">
    <ENTRYREF HREF="http://localhost" CLIENTSKIP="NO" />
</EVENT>
</ASX>
```

关于脚本文件，请参考 Windows Media Index 工具。

LOGO 标签：使用图形文件（BMP，JPG，GIF）作为媒体文件的 LOGO（在很多版本中显示不出来），示例代码如下：

```
<LOGO HREF="logo1.gif" STYLE="ICON" />
```

MOREINFO 标签：指定额外的网页地址，用以获取更多的信息，示例代码如下：

```
<ASX VERSION = "3.0" >

    <TITLE > Example Media Player Show </TITLE >
    <MOREINFO HREF = "http ://liumeiti. com" / >

    <ENTRY >
        <TITLE > Example Clip </TITLE >
         <MOREINFO HREF = "http ://liumeiti. com" / >
        <REF HREF = "test. asf" / >
    </ENTRY >

</ASX >
```

PARAM 标签 :可以控制 Windows Media Player 的各个参数,其参数可以参考第 2 章,示例代码如下:

```
<ASX VERSION = "3.0" >
    <TITLE > Example Media Player Show </TITLE >
    <PARAM NAME = " Director" VALUE = "Jane" / >

    <ENTRY >
        <TITLE > Example Clip </TITLE >
        <REF HREF = "test. asf" / >
        <PARAM NAME = "Location" VALUE = "china" / >
        <PARAM NAME = "Release Date" VALUE = "March 1998" / >
    </ENTRY >
</ASX >
```


PREVIEW DURATION 标签 :用以设定预览停止时间,示例代码如下:

```
<ASX VERSION = "3.0" >
    <ENTRY >
        <PREVIEWDURATION VALUE = "0 :03 :00" / >
        <REF HREF = "test. asf" / >
    </ENTRY >
</ASX >
```

REF 标签 :作用于 ENTRY 域,指定播放的内容,示例代码如下:

```
<ASX VERSION = "3.0" >
    <ENTRY >
    <TITLE > Example Clip </TITLE >
        <REF HREF = "tset.asf" / >
        <REF HREF = "test2.asf" / >
    </ENTRY >
</ASX >
```

REPEAT 标签：循环播放指定的内容，Count 参数指定播放的次数，示例代码如下：

```
<ASX VERSION = "3.0" >
    <ENTRY >
          <REF HREF = "test1.asf" / >
          <！—播放一次 - >
    </ENTRY >

    <REPEAT COUNT = "2" >
        <ENTRY >
            <REF HREF = "test1.asf" / >
            <REF HREF = "test2.asf" / >
            <！—播放两次 - - >
        </ENTRY >
    </REPEAT >
</ASX >
```

STARTTIME 标签：指定开始播放的时间，但不得大于媒体文件时间长度，示例代码如下：

```
<ASX VERSION = "3.0" >
    <ENTRY >
        <STARTTIME VALUE = "0:03.0" / >
          <REF HREF = "test.asf" / >
    </ENTRY >
</ASX >
```

TITLE 标签：指定一段标题，在前面已经多次应用。

5.2.3 动态 ASX 文件

ASX 文件的内容是固定的，如果与 ASP 程序结合，生成动态的 ASX 文件，将带来巨大的灵活性。Microsoft 提供了这个动态的方案，对熟悉 ASP 的读者，这是非常容易的，只需将 ASX 文件的扩展名改为. asp，加入代码，通过 HTTP 访问的时候，就可以使用动态的 ASX 文件，功能非常强大，使用 PHP，JSP 也可以提供同样的效果。

示例代码如下：

```
<%Response. ContentType = "video/x - ms - asf"% >
<%Response. expires =0 % >
<ASX VERSION = "3. 0" >
    <TITLE > Your title here < /TITLE >
    <ENTRY >
        <REF HREF = "mms ://localhost/dj. asf" / >
    < /ENTRY >
< /ASX >
```

上述代码加粗的就是必须的两行 ASP 脚本程序。这时，还可以访问数据库了。将代码保存为一个 ASP 文件，并通过 IIS 访问这个脚本，浏览器会自动调用 Windows Media Player 来解析这个文件。

不同的扩展名，MIME 类型不同，相关的 MIME 如表 5.3 所示。

表 5.3 相关的 MIME 类型

扩展名	MIME 类型	文件内容
. wma	audio/x-ms-wma	Windows Media file, with audio content only. Typically used to download and play files or to stream content.
. wmv	video/x-ms-wmv	Windows Media file with audio and/or video content. Typically used to download and play files or to stream content.
. asf	video/x-ms-asf	Legacy content. Typically used to download and play files or to stream content. May contain audio and/or video content. Typically used to download and play files or to stream content.
. wm	video/x-ms-wm	Reserved
. wax	audio/x-ms-wax	Metafiles that reference Windows Media files with the . asf, . wma or . wax file extensions.
. wvx	video/x-ms-wvx	Metafiles that reference Windows Media files with the . wma, . wmv, . wvx or . wax file extensions.
. asx	video/x-ms-asf	Metafiles that reference Windows Media files with the . wma, . wax, . wmv, . wvx, . asf, or . asx file extensions.
. wmx	video/x-ms-wmx	Reserved

不同的扩展名使用不同的 MIME 类型，在 ASP 脚本程序中必须做相应的更改。

5.3　操作实战

5.3.1　使用 ASX 文件进行连续流切换

通常，当流结束时，会在打开下一个流之前缓冲该流（如果它是来自流媒体服务器的内容）。Windows Media 服务功能允许实现连续流切换，从而减少缓冲时间并几乎在同时发送下一个内容流。

要实现连续流切换只能使用 .asx 文件，编码器必须在 OpenEvent 类型的流中包含脚本命令，事件名在 .asx 文件的 Event 元素中定义。当 Windows Media Player 从编码器接收脚本命令时，将查看 .asx 文件中的 Event 元素，并开始打开 Event 元素中定义的流。Windows Media Player 将保留该信息，直到接收到格式为 Event eventname 的同名事件。当收到同名的事件时，Windows Media Player 将连续切换到该流，这是因为已经打开并缓冲了该流。

5.3.2　使用 ASX 文件创建播放曲目

播放列表由 .asx 文件中的多个 Entry 元素组成。Windows Media Player 将按顺序播放 .asx 文件中的 Entry 元素，如同用户手工打开每个剪辑一样。

简单的播放曲目代码如下：

```
<ASX version = "3.0" >
<Title > Most requested titles of the day < /Title >
<Entry > <Ref href = "mms ://nsserver/content/title1. asf" / > < /Entry >
<Entry > <Ref href = "mms ://nsserver/content/title2. asf" / > < /Entry >
<Entry > <Ref href = "mms ://nsserver/content/title3. asf" / > < /Entry >
<Entry > <Ref href = "mms ://nsserver/content/title4. asf" / > < /Entry >
< /ASX >
```

5.3.3 为流媒体文件加上广告

了解了上面的基本概念以后,我们来小试一把——为你的流媒体文件加上广告。让观众在观看流文件的同时可以直接顺便访问你的网站。用流媒体做广告有两个比较好的途径:一是 Banner 条,二是插播广告片断。Windows Media 提供了比较好的解决办法。

这里要使用上面提到的 Meta 文件。meta 是一个文本文件,它可以存放一些特殊的标签,比如 Banner 用的 < Banner > 标签。

使用 Meta 文件也很容易,只要在 Windows Media Player 嵌入代码中把原来流媒体文件的名字换成 Meta 文件的名字就可以了,如 < PARAM name = "FileName" value = "demo. asx" > 。运行的时候,控件播放 ASX 文件,然后通过 ASX 文件找到对应的流媒体文件。ASX 起到了中转站的作用。

下面来看看 ASX 文件中的一些标记:

① < ASX > < /ASX > 表示 asx 文件的开始和结束,类似 < Html > < /Html > 标签。

② < Entry > < /Entry > 表示一个节目的片断。

③ < Ref href = " " / > 指向一个流文件,可以是. asf,. wma,. wmv。

④ < Abstract > < /Abstract > 插入描述性的话。

⑤ < MoreInfo href = " " / > 指向某个地址。

下面来看这个加入广告和 Banner 的代码:


```
< ASX version = "3. 0" >
< Title > 标题:这是广告 < /Title >
< Abstract > 演示文件 < /Abstract >
< MoreInfo href = "http://www. microsoft. com" / >

< !-- 一开始就播放广告 -- >
< Entry >
    < Title > 标题:这是广告 < /Title >
    < Author > 作者:广告事业部 < /Author >
    < Copyright > 版权:2000 电脑爱好者 < /Copyright >
    < Abstract > 这个广告的内容描述 < /Abstract >
    < MoreInfo href = "http://www. microsoft. com" / >
    < Ref href = "ad. asf" / >
< /Entry >
< !-- 正片开始 -- >
< Entry >
        < Banner href = "purchase. gif" >
```

```
        < Abstract > Click here to go to our Web site < /Abstract >
  < MoreInfo href = " http ://www. liumeiti. com" / >
        < /Banner >
     < Ref href = " show. asf" / >
< /Entry >
< /ASX >
```

上面这段代码就是一个广告片段加正片 Banner 的例子。这个 ASX 文件里面有两组 < Entry > 标记，表示有两组视频片断播放。第一个片段是广告 < Ref href = " ad. asf" / > 指向的广告文件 ad. asf，其余的标签比较简单。第二个片段是正片 show. asf，其中加上了 Banner 代码。

< Banner > 中 href 指向 Banner 的图片。 < Abstract > 显示的是当光标移动到图片上的时候，显示的提示文字。 < MoreInfo href = " http ://www. Microsoft. com" / > 制定当单击 Banner 以后跳转到微软网。“ Puchase This movie ”就是 Banner 广告。

播放整个网页，先播放一段广告片，之后就是正片。下面有一 Banner 条单击以后可以跳转到 www. liumeiti. com。

5.3.4　显示提示用户登录

以下的例子演示了在播放演示节目时，播放区域下面用一个图标来显示提示用户登录的字样。当用户点击该页面时，浏览器会连接到网站的登录注册页面，使用户可以登录后使用网站的服务，该例的代码如下：


```
< ASX version = "3. 0" >
     < Title > The Show Title < /Title >
     < Entry >
          < Ref href = " mms ://MYserver/Path/title1. wma" / >
     < /Entry >
     < Entry >
          < Ref href = " mms ://MYserver/Path/title2. wma" / >
     < /Entry >
     < Entry >
          < Ref href = " mms ://MYserver/Path/title3. wma" / >
     < /Entry >
< Banner href = " http ://localhost/Default_05. gif" >
     < Abstract > Click here to go to our Web site. < /Abstract >
     < Moreinfo href = " http ://localhost/myvod/login. asp" / >
< /Banner >
< /ASX >
```

这段代码实际上就是使用前面制作的播放列表，加上一个 Banner 元素完成的，

其中 Banner href 定义了 Banner 文件的路径，< Abstract > < /Abstract > 标签定义了当鼠标移动到 Banner 区域后显示的提示文字，< Moreinfo href 则定义了点击后用户登录页面的地址。完成后保存为 demo. asx。用播放器打开，效果如图 5. 1 所示。

图 5. 1　示例程序

5. 3. 5　控制自己的节目

可以使用任何文本编辑器来创建一个简单的 MetaFile 播放列表，它的功能是顺序播放节目，代码如下：


```
< ASX version = "3.0" >
    < Title > The Show Title < /Title >
    < Entry >
        < Ref href = "mms://MYserver/Path/title1.wma" / >
    < /Entry >
    < Entry >
        < Ref href = "mms://MYserver/Path/title2.wma" / >
    < /Entry >
    < Entry >
        < Ref href = "mms://MYserver/Path/title3.wma" / >
    < /Entry >
< /ASX >
```

这段代码是由 ASX 元素构成，同时使用了 Entry 子元素包含的 Ref 元素定义流媒体文件的路径，是最简单的应用。

在一个 ASX 元素中，至少要有一个 Entry 或 Entryref 元素。

5.3.6　高级应用

本例演示更高级别的应用，位于配套光盘 Sample/Sample5_1 目录下，示例代码如下：

```
<ASX version = "3.0" >

<!-- Title 信息 -->
    <TITLE >高级应用演示 </TITLE >

    <!-- 注释信息，在 Windows Media Player 读入 ASX 文件时显示 -->
    <Abstract > More Information at this Web site </Abstract >

    <!-- 获取更多信息的网站资源 -->
    <Moreinfo href = "http ://www. liumeiti. com" / >

    <!-- 指定 Banner 图像，在播放之后的内容时，在影像下方显示 -->
    <!-- 定义 Banner 图形文件 -->
    <Banner href = "logo2. gif" >
    <!-- 定义注释 -->
        <Abstract > Liumeiti. com </Abstract >
        <!-- 定义更多资源获取网站 -->
        <Moreinfo href = "http ://www. liumeiti. com" / >
    </Banner >



    <!-- 播放内容一，设置参数 ClientSkip，用户不可以拖动滑块 -->

    <Entry ClientSkip = "no" >
    <!-- 设置本内容的 Banner ，在播放本内容时，显示此 Banner -->
```

```
        <Banner href = "one.bmp">
            <Abstract>Visit Our Web site</Abstract>
            <Moreinfo href = "http://www.liumeiti.com" />
        </Banner>
        <Moreinfo href = "http://www.liumeiti.com" />

        <!-- 节目信息 -->

        <Abstract>请访问网站获得更多信息</Abstract>

        <TITLE>第一个播放的节目</TITLE>
        <AUTHOR>Test</AUTHOR>

        <COPYRIGHT>(c)2000</COPYRIGHT>

        <!-- 播放节目 -->
        <Ref href = "test.asf" />
    </Entry>


    <!-- 播放内容二 -->

    <Entry>

    <!-- 播放时显示默认的 Banner -->

        <Title>第二个播放节目</Title>

        <Author>测试使用</Author>

        <Copyright>(c)2000</Copyright>

        <!-- 提示信息 -->

        <Abstract>第二个提示信息</Abstract>

        <Ref href = "test2.asf" />
    </Entry>

</ASX>
```

商务篇：Windows Media Rights Manager SDK

第 6 章

6.1 保护数字版权：DRM 简介

从 Internet 上下载歌曲和其他数字媒体是当今最热门的趋势之一，但伴随这种趋势也出现了诸如盗版、音质和媒体发布等问题。为了帮助内容所有者控制他们的资料，Microsoft 创建了 Windows Media DRM，帮助内容所有者在 Internet 上发布具有优良音质并得到许可的数字媒体。有了这项技术，客户就可以更容易地通过正当手段获取你的内容，而不必去盗版。

数字权限管理（DRM）是确保内容安全并管理访问权限的技术。这种技术仍在开发和研究之中；不过，随着 Windows Media DRM 的引入，已经向对数字媒体内容侵权的防止方向迈出了第一步。Windows Media DRM 通过加密源. asf，. mp3 和. wav 文件为打包的. asf 文件来帮助保护视频和音频内容的安全，并为用户提供许可证。

通过执行这个功能对分发的每份内容上都加上一个数字签名，发布商就可以知道，发布内容的副件被哪些用户观看。

数字版权保护正是解决数字媒体版权问题的技术手段。DRM 的解决思路本质上讲是许可证管理（License Management），其机理可以用图 6.1 来表示，一个 DRM 产品相对应地由三部分组成：

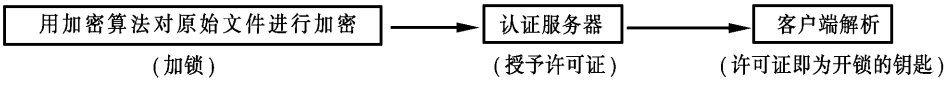


图 6.1 DRM 原理

当一个合法用户点击网站或光盘的内容时，机器会自动检查有没有相应的许可证（license），如果没有，系统会指示客户到特定的注册地址去注册。当客户输入用户名、密码或者插入 IC 卡、IKEY（一种 USB 接口的身份认证令牌）时，认证服务器将校验客户的身份以及相应的权限，如果校验的结果为该客户是合法用户，并且用户所点击的内容也是在用户付费范围内，认证服务器将读取该用户的硬件指纹（可以是 PC 的指纹，也可以是 IC 卡、IKEY 的指纹），生成一个许可证文件植入该用户的机器上。如果说打包的过程是给数字文件加了一把保险锁，那么，许可证就是开锁的钥匙，每个合法用户拥有一个特定的、惟一的钥匙，而且是与硬件指纹绑定的。这就意味着即使是合法用户也不能对数字文件进行非法拷贝、传播，因为拷贝后的文件是加过密的，离开它认证的机器，就没有对应的钥匙，也就无法正常运行了。

合法用户获取许可证文件的途径有两条：在线或离线认证（如电话、E-mail 等传统方式），离线认证对于光盘发行无疑是最有力的支持。内容提供者可以根据需要，在许可证中设置不同的有效期，一个月、几个月、一年、几年或无限期，通过有效期的设置，可以对网站会员或散户实行不同的收费标准和管理，从而保证了网站多种经营方式并存。目前，DRM 的应用已经从网校、信息网站拓宽到企业局域网重要信息保护、宽带内容保护等众多领域，提供 DRM 技术的厂商也越来越多。微软针对流媒体市场的迫切需要，推出了针对 Widnows Media 格式文件的数字版权管理系统——Windows Media DRM 6. x，该方案提供了完整的数字媒体版权解决方案。

可以尝试用微软发布的 Windows Media DRM SDK 7.0 来演示一遍流媒体 DRM 的具体实施，这个 SDK 可以从微软的网站上取得，大小为 1 MB 左右，该软件最好是在 Windows 2000 Server 下运行，并且需要有 IIS 的支持。

DRM 是一种广义的数字版权管理，而 WMRM（Windows Media DRM）是微软推出的针对 Windows Media 文件格式的数字版权解决方案，二者意义不尽相同。在很多环境中，也称 Windows Media DRM 7 为 WMRM7。在本章中，所有讨论以 Windows Media DRM 7 为基础。

6.1.1　内容制作

Windows Media DRM 提供了一种方法，用户在使用 Windows Media 包装程序分发内容给 Internet 服务提供商前保护内容的安全。在创建 ASF 内容时，若想对. asf 文件执行 DRM，可选择以下两者之一：

①使用 Windows Media 工具创建. asf 文件，然后用 Windows Media 包装程序将文件打包。

②使用 Windows Media 包装程序直接创建打包的文件。

打包文件的过程执行几个不同的操作：

①用私钥加密文件。

②设置到许可服务器的 URL。

- 设置到官方 Web 站点的 URL。
- 包含若干横幅或图像于文件中。
- 设置标题、艺术家、版权和流派等的属性。

一旦一个文件打包完毕，就不能再用 Windows Media 工具对它进行修改。如果想要以后修改一部分内容，最好先用 Windows Media 工具创建文件，存储一个备份，然后再创建打包的文件。

媒体文件已被加密，所以没有许可证就不能进行播放。另外，许可证不能被共享，也不能被成功复制。虽然仍不能杜绝黑客访问这些内容，但 Windows Media Rights Manager 使得对这些内容进行盗版要比通过正当途径获取它困难得多。

6.1.2　Web 发布

Windows Media DRM 包含一个站点向导以帮助将打包的内容发布到 Web 站点上。一旦将内容打包，用户就需要有效的许可证才能播放此内容。当用户试图播放内容时，要进行许可证验证。若用户在其计算机上没有有效的许可证，浏览器将启动并将他们带到用户的 Web 站点以注册他们的内容。一旦用户注册后，就会将有效的许可证和解密钥下载到计算机上，然后就可以播放内容了。

作为发布者,需要维护许可证和用户的数据库。Windows Media 许可证服务使用 SQL 数据库,所以必须首先安装有 SQL ,才能安装 Windows Media DRM。若要传送打包的流式化内容,还需要有 Window Media Services 4.0。

6.1.3 用户体验

用户需要有 Microsoft Windows Media Player 6.1 以上的播放器播放打包的文件。可以由 Windows Media 服务器传递或下载打包的文件用于本地播放。如果当用户尝试播放受保护的内容时,他的计算机上没有正确版本的 Windows Media Player,用户的浏览器将打开并将用户导航到下载站点,用户可以从那里下载 Windows Media Player。

Windows Media Player 检查用户是否有播放内容的许可证。若用户没有有效许可证,会打开用户的浏览器并导航到您的 Web 站点的注册页。用户添完注册信息后,就会向用户颁发许可证,然后 Windows Media Player 就会播放内容,用户就可以欣赏内容直到许可证到期为止。在下载许可证条款的过程中,会通知用户,许可证可为无限期的、设定一定时间的或设定播放次数的。然而,如果注册的用户复制内容并与其他用户分享,该用户也需要经过注册过程。不同的计算机之间不能复制以及共享许可证和解密钥。

当 Windows Media Player 播放受保护的内容时,用户会看到以下条款:

- 内容标题。
- 艺术家名。
- 版权通告。
- 横幅图像。
- 视频图像。
- 合法性图标(说明内容受到保护,未被侵犯)。

另外,单击窗口的不同区域会打开用户的 Web 浏览器到相应的 URL。例如单击横幅图像会打开分发者的 Web 站点,而单击标题会打开艺术家的 Web 站点。

6.2 安装与配置 Windows Media DRM

6.2.1 安装 Windows Media DRM 7 SDK

首先进行 SDK 开发包的安装 :同大多数的程序一样,点击安装包并一路 Next,即可完成安装,安装过程中系统会自动放置程序文件和文档,并注册组件。安装完后会在安装 SDK 所在的盘中看见 WMSDK hWMRM 的目录结构,在 WMRM 子目录下,会发现两个子目录 :一个为 BIN,放置库文件 ;另一个为 Samples,该目录下放置的是微软提供的例子以及源代码。

由于 SDK 提供的是基于 ASP(Active Server Page)的源程序,安装完开发包后,还

要开始设置 IIS(关于 IIS 的基本知识,可以查阅有关资料,在此不多叙述),在控制面板→添加删除程序→添加删除 windows 组件里面添加完 Internet Information Services 后,进入程序→管理工具→IIS 管理。还记得 Samples 目录吗?现在就是要建立一个虚拟目录,而把本地路径指向这里。

例如,建立一个名叫 DRM 的虚拟目录,本地路径是 C:\WMSDK\WMRM\SAMPLES。接下来,要到微软的网站上获取一个许可证,只有得到该许可证,DRM 才能正常工作。访问微软网站 http://licenseserver.windowsmedia.com,点击 Enroll to get a new certificate,接下来会有一个注册会话框要求填入 E-mail 地址,完成注册过程后会收到一封包含确认信息的 E-mail,再次访问 http://licenseserver.windowsmedia.com,点击 Complete the enrollment with your e-mail confirmation,填入你的 E-mail 地址和刚刚发给你的确认信息,点击确定,即可完成申请许可证的过程。

要安装最近的 Windows Media License Service information(例如 Revocation list)。同样,访问 http://licenseserver.windowsmedia.com,点击相应的选项即可完成。

6.2.2　设置 Windows Media DRM

在设置 DRM SDK 之前,有必要先了解一下 DRM 的基本原理,前面说过,DRM 是一种许可证管理。

1)数字版权管理的运行机制

数字媒体文件的拥有者使用 DRM 平台(对 Web VOD 的运营商来说,即为 DRM 认证服务器)对其拥有的数字媒体文件进行加密、发布。当使用者通过 Web Server 或流媒体服务器(Streaming Media Server,如 Windows Media 服务器)或其他途径取得一个被 DRM 加密的文件并进行播放时,支持 DRM 的播放器首先会向一台指定的证书服务器(License Server)提出一个播放权限请求,然后经过一系列设定的流程,播放器取得该文件的播放证书,最后播放媒体文件,流程如图 6.2 所示。

(1)节目加密

通过对 Windows Media 格式的文件进行加密来达到保护该文件数字版权的目的,文件的拥有者使用 Windows Media DRM 来加密文件,通过密钥技术,被加密的文件就像被一把特定的锁来锁住,要想播放它就需要专门的钥匙来打开。密钥经加密后保存在证书中,证书将独立于媒体文件单独进行发布。同时,文件的拥有者可以将一些附加的信息加入文件中,例如证书的发布地址、文件的拥有者等。

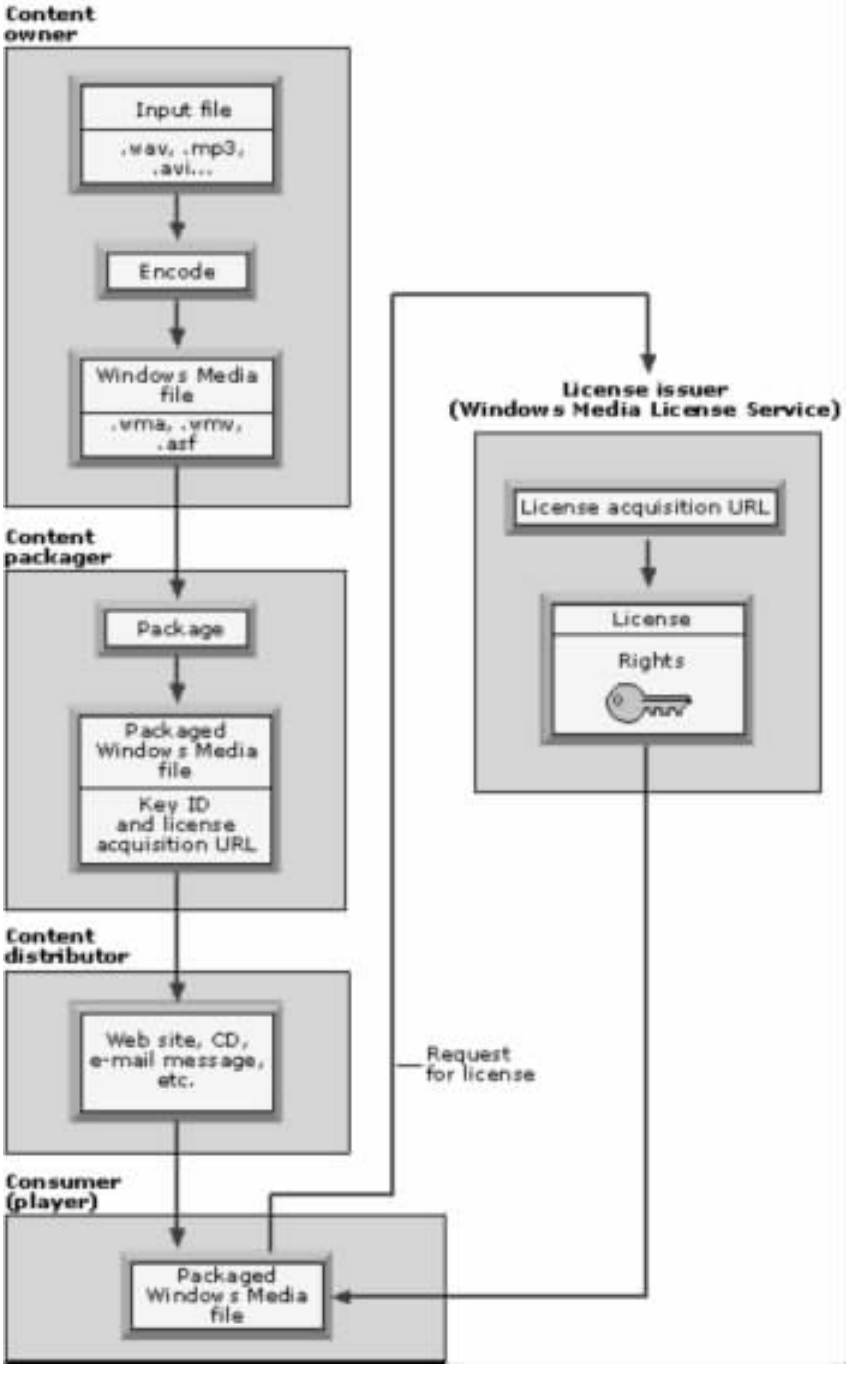


图 6.2　DRM 流程

(2) 建立证书服务器

媒体内容的拥有者若要使发布出去的媒体文件能够进行验证，需要使用 Windows Media DRM 来建立证书服务器，它可以响应用户对媒体文件的播放申请，并判断是否发布一个许可证到该用户手中，同时该服务器也可以完成一些附件的功能，如

计费等。

(3) 发布媒体文件

在拥有证书服务器和加密节目后，媒体的运营商现在可以发布自己的媒体内容了。发布的方式将是多种多样的，既可用 Web 站点的方式发布，也可使用流技术通过媒体服务器发布，还可以以文件 FTP 下载、CD、E-mail 的方式来发布。

(4) 用户获取播放证书

消费者通过各种途径获取媒体文件后，可以使用支持 DRM 的播放器来解密文件，播放器会自动指向媒体运营商制定的证书服务器来请求一个播放证书，该证书内含指定的密钥，可以用来打开加在文件上的锁，媒体的运行商可以在这个许可证书里面定义各种权限，如播放次数限制、有效时间的限制等。

2) 数字版权管理的加密、解密原理

Windows Media DRM 是通过密钥对媒体文件进行加密、解密的，密钥一般有两把：一把公钥(Public Key)，一把私钥(Private Key)。公钥用于加密节目内容本身，私钥用于解密节目，而且如果节目头部有被改动或破坏的情况，利用密钥就可以判断出来，从而阻止节目被非法使用。

为了便于理解公钥与私钥的概念如图 6.3、图 6.4、图 6.5 所示。

图 6.3　一般加密解密过程

图 6.4　密钥的作用

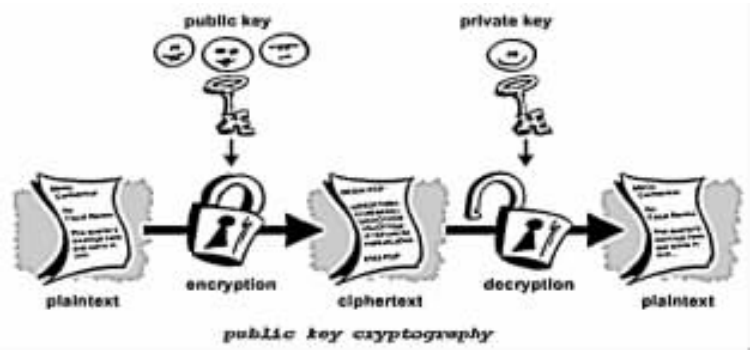


图6.5 公钥与私钥

3）配置 Windows Media DRM SDK

在 IIS 设置的虚拟目录下，有个 Global. asa，它包含了在特定事件发生时调用的相关事件过程，该文件必须将它存储在应用程序的起始目录中，通过设置它可以为 DRM 提供加密节目必需的信息。以下是一个 global. asa 的例子：

```
< ! DOCTYPE HTML PUBLIC"-//W3C//DTD HTML 4.0 Transitional//EN" >
< HTML > < HEAD >
< META content = "text/html ; charset = unicode" http - equiv = Content - Type >
< SCRIPT LANGUAGE = VBScript RUNAT = "Server" >
// ##########################################################################
//
// Global. asa
// Microsoft Windows Media Rights Manager v7.0
//
// Copyright (c) 2000 Microsoft Corporation. All rights reserved.
// IMPORTANT :THESE SAMPLES ARE PROVIDED FOR HIGH - LEVEL REF-
ERENCE ONLY
//                AND SHOULD NOT BE USE IN PRODUCTION ENVIRONMENT.
// ##########################################################################
Sub Application_OnStart

    Application("siteurl") = "http ://192.168.1.66 :8080/sdk" 'URL for License
server
    Application("KeyID") = "Ocb3Fzr9gEuxURYy0tmzig = = "   'for pre-delivery

    // Content Server information. In this sample, we support only one content server.
    Application("seed") = "kcuMfqpcrmLyfE4IdY4u8WHuiSkxNzY5AQ7QmLiT"
```

```
    Application("contentserverpubkey") = "2NUOo0SDFVjGVOybkQjov0NFiEDSK-
JhW7WV0VYcPsE！ADroT9fHhgw = = "
    Application("contentserverprivkey") = "t1w49p0crYnHu * ！ox27Ua3！vmHc = "

    // License Server information.
    Application("licserverprivkey") = "xxx"
    Application("licserverpubkey") = "xxx"
    Application("cert1") = "XXX"
    Application("cert2") = "XXX"

    // DRM Client verfication keys.
    Application ( " veri1 " ) = " dpRKG * JUYWa！yw1CiWdPiwFqFHBM *
9UklDaEU6tf4aMG2GFsWdkBJw = = "
    Application ( " veri2 " ) = " oJO8d5x1x9！fKMSReWusAvg！GQuiomlv0B4！
Y1kcLtBxhGeQj7i5HA = = "
    Application("veri3") = "XVEsJxPWYxSbiKqOcy9htr1iqgPgP3OnBJ0cbTTO4F-
g50qtpS0alUQ = = "
    Application("veri4") = "WRZPSd6hM7Q9ZakF9NO3ydZA7UN888SR * ！j！
cH！wrd18zsbXVdR4fw = = "

    Application("ownerid") = "ZZIP"
    Application ("dbguid") = "xxx"

End Sub
</SCRIPT>
```


可能会觉得有点长，实际上去掉帮助信息，需要配置的地方并不多，其中要配置的几个参数的功能如表 6.1 所示。

表 6.1　需要设置的参数

Application("siteurl")	认证站点的 URL
Application("KeyID")	用来生成密钥，用于预分发
Application("seed")	种子，用来生成密钥
Application("contentserverpubkey")	公钥，用来加密节目
Application("contentserverprivkey")	私钥，用来解密节目
Application("ownerid") = "ZZIP"	用户自定义，用来表示内容所有者
Application ("dbguid") = "xxx"	加密内容的惟一标识

先通过在安装了 DRM 的计算机上运行一个脚本来生成这些参数，然后填入对

应的地方,脚本代码如下:

```
<html>
<body>
<%
Set keysobj = CreateObject("wmrmobjs.WMRMKeys")
keysobj.GenerateSigningKeys privkey, pubkey
response.write "  contentserverprivkey:" &privkey
%>
<p>
<%
response.write "  contentserverpubkey:" &pubkey
%>
<p>
<%
sKID = KeysObj.GenerateKeyID()
sSeed = KeysObj.GenerateSeed()
response.write "Seed:" &sseed
%>
<p>
<%
response.write "KeyID:" &sKID
%>
</body>
</html>
```


在地址栏输入 http://192.168.1.1/DRM/grakey.asp,运行后会输出对应的参数,粘贴到 global.asa 里面对应的地方,保存文件即可。

初步的设置完之后,现在已经可以使用这台安装了 Windows Media DRM 的机器进行节目的加密和认证授权了。就是说只要安装了 Windows Media DRM 服务,那么这台机器在可以用来加密节目的同时也可以作为验证服务器。

6.2.3 打包和发布文件

下面通过打包一个文件来具体说明 Windows Media DRM 加密文件的过程。

1)文件的打包

Windows Media DRM 将媒体文件打包的处理包括:编码、压缩、用密钥对它进行加密处理以及用来自数字证书的密钥为它签名(如果你有数字证书,并希望将其包

括在内的话)。打包与获取认证过程如图 6.6 所示。

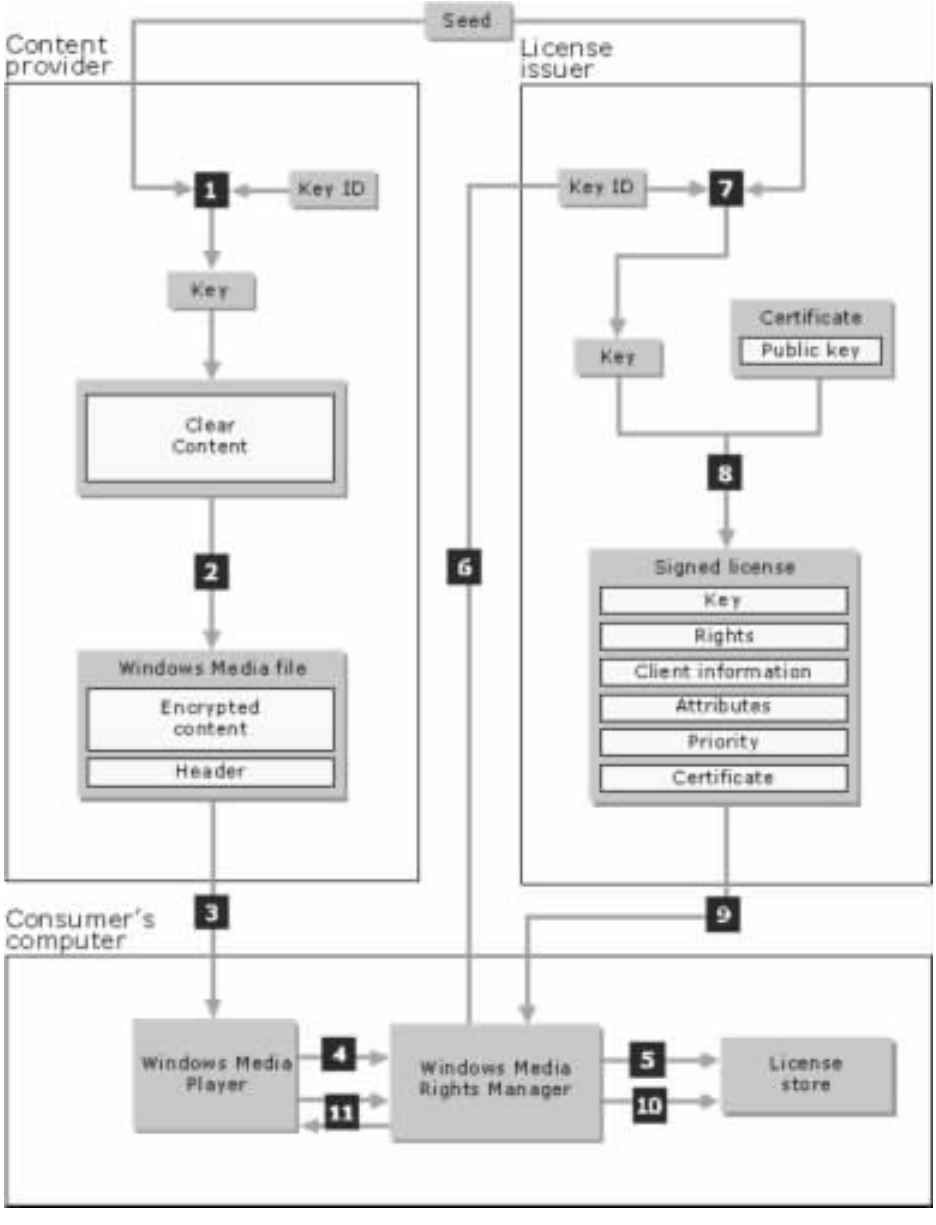


图 6.6　打包与获取认证流程

其他信息也会添加到该文件中,如文件的标题、艺术家的姓名、版权声明、标题图像、可获取许可证的 URL 以及其他 URL。生成的打包媒体文件以 Windows Media Audio(扩展名为 .wma)或高级流格式(扩展名为 .asf)保存,并可以在支持这些格式的任何媒体播放器(如 Microsoft Windows Media Player)中播放。

打开 IE 浏览器,在地址栏输入 SDK 源程序脚本所在的虚拟目录,如 http://192.168.1.1/DRM/package.asp,如果安装没什么问题的话,会显示如图 6.7 所示的页面。

单击 Input file name 一栏的 Browser 一项,选择你想打包加密的文件,在 Output path and file name 一栏填入文件的输出路径,如 d:\demo.wmv。Version 7 license ac-

图 6.7　打包页面

quisition URL 一栏可以选择用来获取许可证的页面的名称，在微软提供的源代码中，缺省是 Simple. asp。你还可以输入 Owner ID，比如你公司的名字，这些信息也会被打包到节目中。在 Rights to allow 一栏中，可以设置对节目进行授权，比如限制播放次数和播放的有效期限等。至于 Key ID，Key，license key seed，Public key，Private key，则由 Global. asa 来提供。设置完毕后，单击 Package 按钮进行打包。

2）发布

因为打包的媒体文件与播放它的许可证是分开的，所以发行商可以用不同的方式发布打包的媒体文件。例如，可以将打包的文件放在 Web 站点上供下载，用 CD 发布、用 E-mail 发送给客户等。客户也可以共享和复制打包的媒体文件。图 6.8 为一个 Web 站点示例，客户可在此下载打包的媒体文件。

3）获取验证

要播放打包的媒体文件，客户必须获得许可证，其中包含的密钥可用来解锁此内容。当客户第一次播放某个打包的媒体文件时，获取许可证的过程就会自动开始。如果在客户的计算机上检测不到许可证，Web 浏览器就会打开一个注册网页，提示

图 6.8　站点示例

客户输入信息，如电子邮件地址。然后就会颁发许可证，这样客户就可以播放此媒体文件了。如图 6.9 所示。

从现在起，客户就可以根据许可证中包括的权限播放此媒体文件了。默认权限允许客户在用来获得许可证的计算机上播放媒体文件，并允许客户将文件复制到便携设备上。许可证还有到期日。但是，许可证不可转让。如果客户为某个朋友复制了一份打包的媒体文件，则其朋友必须自己另外获得一个许可证才能播放此媒体文件。

确保自己的计算机为在线状态，用 Media Player 6. x 以上的播放器，播放你刚才加密的节目，正常情况下，播放器会自动连接到 MS 网站，进行一个个性化过程，如图 6.10 所示，此时，播放器将发送一个惟一的标识给 Microsoft 的站点，这个过程可能会需要一段时间，需要耐心等待。

通过个性化过程后，Media Player 就会自动获取你刚刚点击的节目中的加密信息，同时连到你定义的 Web 页面上申请许可证，确认无误后，即可播放。

4）进一步分析

现在已经加密了第一个节目，下面就从源代码上分析这一加密过程，打包用的 Package 页面的脚本源码如下：

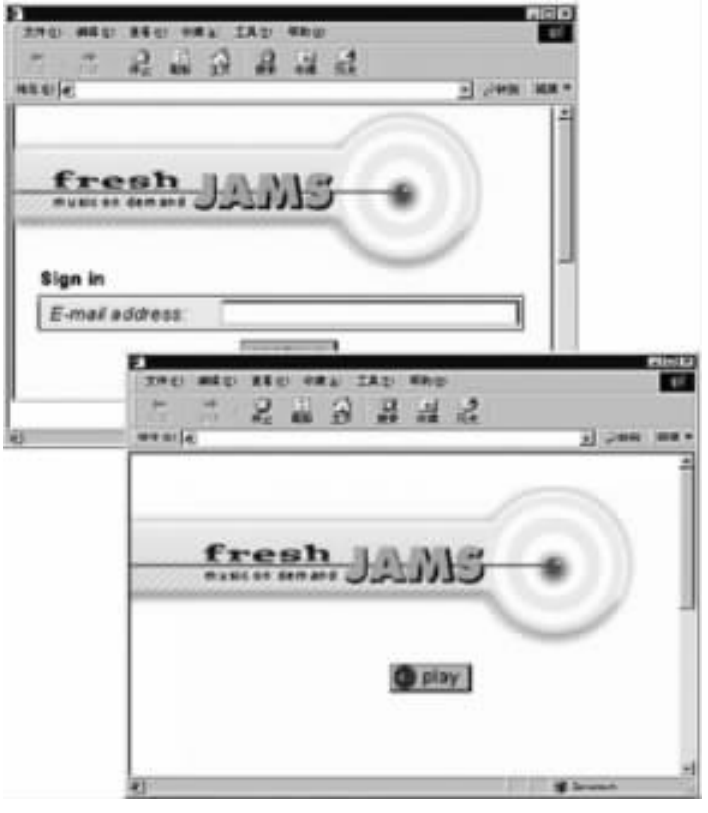


图 6.9　获得许可证

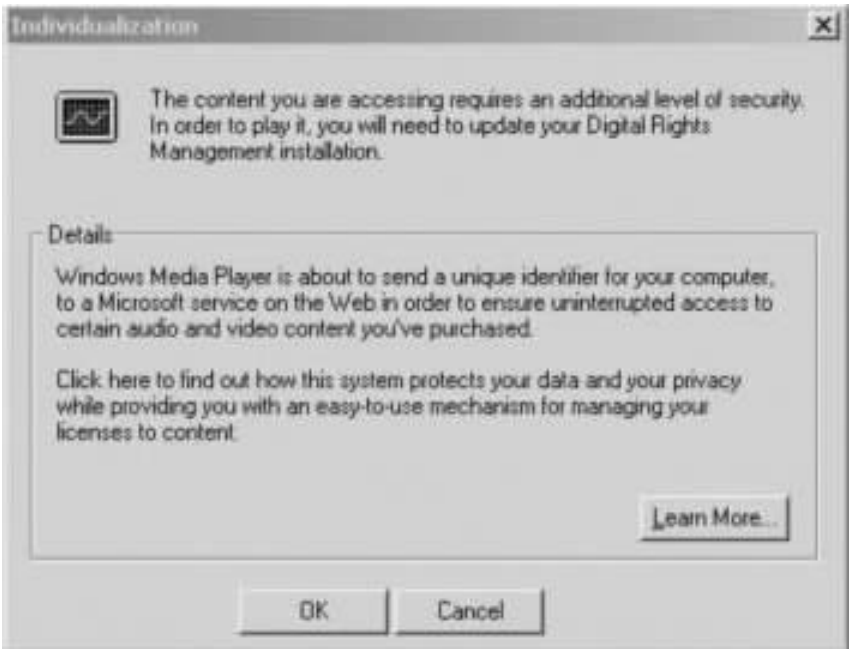


图 6.10　获得播放认证

```
<%@ LANGUAGE = "VBScript" %>
<%
Dim seed, licserverprivkey, licserverpubkey, siteurl, dbguid
Dim keyObj, headobj, protectObj
Dim oldHeaderObj
Dim cid, rid, kid, key, input, output
Dim v1regpage, v2regpage, licurl, v1licurl, owner
Dim header
Dim result
Dim value
Dim secversion
Dim reusekid
Dim inputfile, outputfile, headerfile
Dim outputdir, headerdir
Dim inputfiledefault
Dim sign, signcertcn, signstorename
Dim description
do
    seed                    = Application("seed")
    contentserverprivkey    = Application("contentserverprivkey")
    contentserverpubkey     = Application("contentserverpubkey")
    siteurl                 = Application("siteurl")
    dbguid                  = Application("dbguid")
    owner                   = Application("ownerid")
    // 获取表单提交的数据
    v1regpage   = Request("v1URL") 'Version 1 license acquisition URL
    v2regpage   = Request("URL") 'Version 7 license acquisition URL
    secversion  = Request("secversion") 'Security (individualization) version
    reusekid    = Request("reusekid") 'Whether to reuse the key ID
    inputfile   = Request("input") 'Input file name
    outputfile  = Request("output") 'Output file name
    cid         = Request("CID") 'Content ID
    rid         = Request("RID") 'Rights
    secversion = 2.2
    cid = Trim(cid)
    rid = Trim(rid)
    if (cid = "") then
      cid = "0"
```

```
end if
if (rid = "") then
   rid = "0"
end if
// 设置获取 6.x 版权许可证服务页面地址
licurl        = siteurl + "/" + v2regpage
// 设置 1.x 版本许可证服务页面地址
if v1regpage <> "" then
   v1licurl = siteurl + "/" + v1regpage
end if
err.clear
// 生成对象
Set keyObj              = Server.CreateObject("Wmrmobjs.WMRMKeys")
if (err.number <> 0) then
      exit do
end if
Set headobj             = Server.CreateObject("Wmrmobjs.WMRMHeader")
if (err.number <> 0) then
      exit do
end if
Set protectObj          = Server.CreateObject("Wmrmobjs.WMRMProtect")
if (err.number <> 0) then
      exit do
end if
stop
kid = Request("kid")
if (reusekid <> "ON") then
kid = keyObj.GenerateKeyID()
end if
// 生成 seed 和 KeyID,或者从数据库获取该值
keyObj.seed     = seed
keyObj.KeyID    = kid
 // Get a symmetric key
key = keyObj.GenerateKey()
if (err.number <> 0) then
      exit do
end if
 // 将参数赋给参数
headObj.KeyID                = kid
```

```
        if (err.number <> 0) then
            exit do
        end if
    headObj.LicenseAcqURL     = licurl 'Version 7 license acquisition URL
        if (err.number <> 0) then
            exit do
        end if
    headObj.ContentID         = cid
        if (err.number <> 0) then
            exit do
        end if
    call headObj.SetCheckSum(key)
        if (err.number <> 0) then
            exit do
        end if
    If v2regpage <> "Simple.asp" Then
      headObj.Attribute("RID") = rid
    End If

        if (err.number <> 0) then
          exit do
        end if
    if (secversion <> "") then
    headobj.IndividualizedVersion =   secversion
  end if
        if (err.number <> 0) then
          exit do
        end if
 call headobj.Sign(contentserverprivkey)
       if (err.number <> 0) then
         exit do
       end if
  // Generate the header information
  header = headobj.Header
       if (err.number <> 0) then
         exit do
       end if
if (outputfile <> "") then
       protectObj.InputFile = inputfile
```

```
            if (err.number <> 0) then
                exit do
            end if

            if  v1licurl <> "" then
              protectObj.V1LicenseAcqURL = v1licurl
            end if

            if (err.number <> 0) then
                exit do
            end if
            protectObj.Key = key
            if (err.number <> 0) then
                exit do
            end if
            protectObj.Header = header
            if (err.number <> 0) then
                exit do
            end if
            protectObj.V1KeyID = kid
            if (err.number <> 0) then
                exit do
            end if
          Call protectObj.ProtectFile(outputfile)
           if (err.number <> 0) then
                exit do
            end if
           description = "The media file has been successfully packaged"
           end if
loop while false

if err.number <> 0 then
  description = "0x" + CStr(Hex(err.number)) + " :" + err.description
end if
set keyObj          = nothing
set headobj         = nothing
set protectObj      = nothing
set oldHeaderObj    = nothing
% >
```


该程序在加密文件时,主要分以下几个步骤：

①程序先通过 Application 对象来读取 Global. asa 文件中的参数，并赋给对应的变量，如下所示：

```
seed = Application("seed")
contentserverprivkey = Application("contentserverprivkey")
contentserverpubkey = Application("contentserverpubkey")
siteurl = Application("siteurl")
dbguid = Application("dbguid")
owner = Application("ownerid")

secversion = 2.2 ；设置 WMRM 版本号。
```

②从页面获取加密信息，如认证服务器的 URL 等、加密文件的输入输出路径等。

```
v1regpage = Request("v1URL")
v2regpage = Request("URL")
inputfile = Request("input")
outputfile = Request("output")
```

③建立 Key，Header，Protect 对象。

```
Set keyObj = Server. CreateObject("Wmrmobjs. WMRMKeys")
该对象生成密钥，在本例中，密钥是我们从 global. asa 文件中读取。

Set headobj = Server. CreateObject("Wmrmobjs. WMRMHeader")
该对象生成被加密文件的文件头，文件头中包含加密信息。

Set protectObj = Server. CreateObject("Wmrmobjs. WMRMProtect")
该对象打包加密文件
```

④将获取参数添加到 Header 对象。

```
headObj. KeyID = kid
headObj. LicenseAcqURL = licurl 'Version 7 license acquisition URL
headObj. ContentID = cid
If v2regpage < > "Simple. asp" Then
headObj. Attribute("RID") = rid
End If
headobj. IndividualizedVersion = secversion
并生成文件头
header = headobj. Header
```

headobj. IndividualizedVersion = secversion **很重要，如果版本设置不正确，很可能造成节目无法正常获取证书。**

⑤**调用** Wmrmobjs. WMRMProtect **对象来加密文件。**

```
Call protectObj. ProtectFile(outputfile)
```

介绍到这里，对 Windows Media DRM **已经有了一个大概的了解，下面将通过介绍一个具体的应用实例来说明** DRM **在流媒体方面的应用。**

DRM 6.0 **中的默认打包程序，会出错** 0x80048013**，需要手工填入代码** headobj. Individualizedversion = 2.2**，这也是很多朋友打包不成功的原因，而且，不设置这个值，打包过后的文件无法通过验证。**

6.3 使用权限管理对象

Windows Media DRM **组件由多个对象构成，如表** 6.2 **所示，正是这些对象的有效组合，构成了** Windows Media DRM **技术。**

表 6.2 Windows Media DRM **包含组件**

对 象	注 释
WMRMChallenge Object	Processes a license request.
IWMRMCoding Interface	Converts between text and binary strings.
WMRMHeader Object	Manages content headers.
WMRMKeys Object	Manages encryption keys and license key seeds.
WMRMLicGen Object	Generates licenses.
WMRMProtect Object	Encrypts files.
WMRMResponse	Delivers a license to the consumer.
WMRMRights Object	Specifies the rights that govern the use of encrypted content.
LicenseGenerator Object	Creates version 1 licenses.
RMGetLicense Object	Enables client-side license acquisition.

按照使用的方式，主要有两大部分需要使用 DRM 组件：一部分就是打包过程，如图 6.11 所示；另一部分就是获取认证验证部分，如图 6.12 所示。

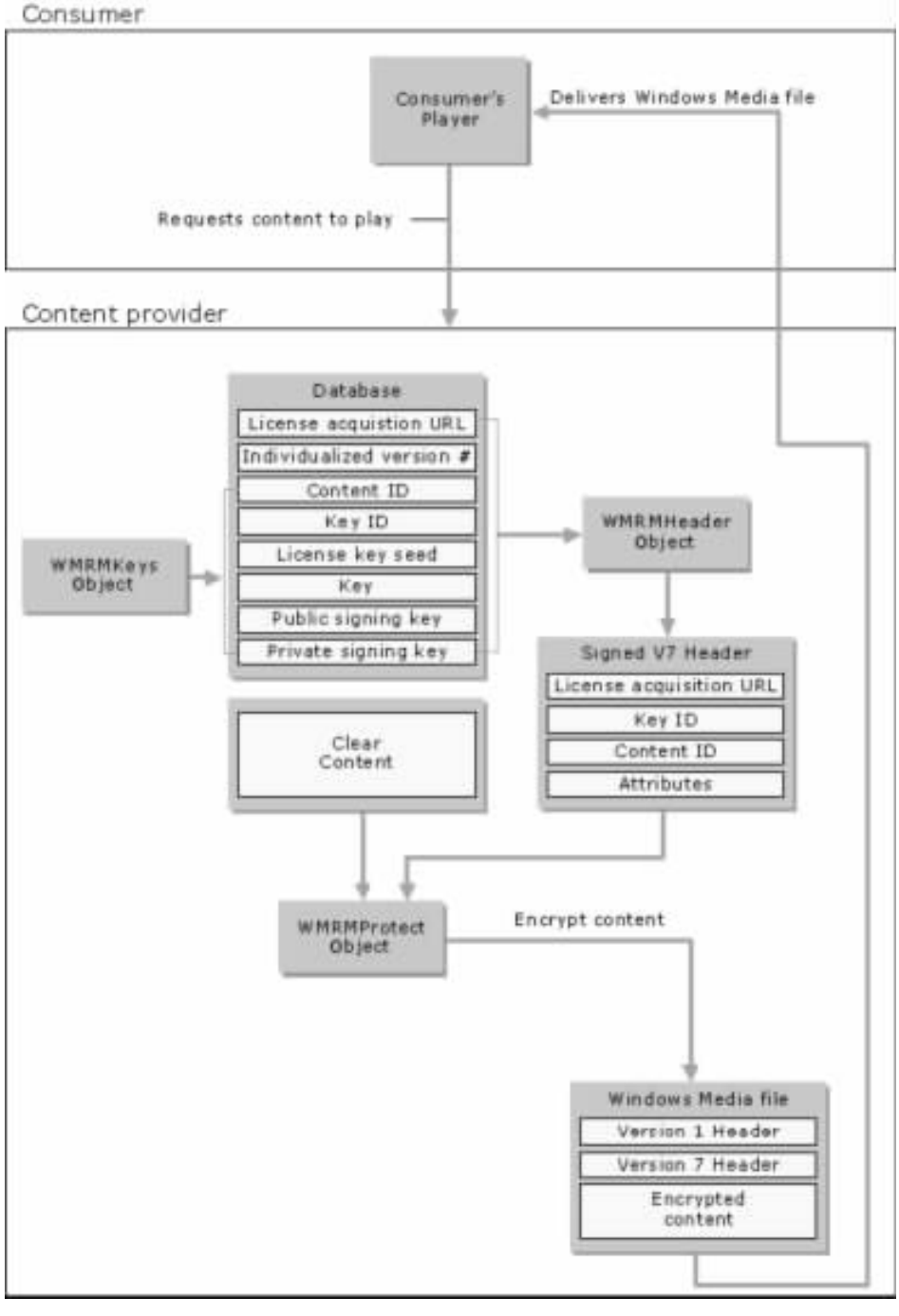


图 6.11　打包过程使用组件

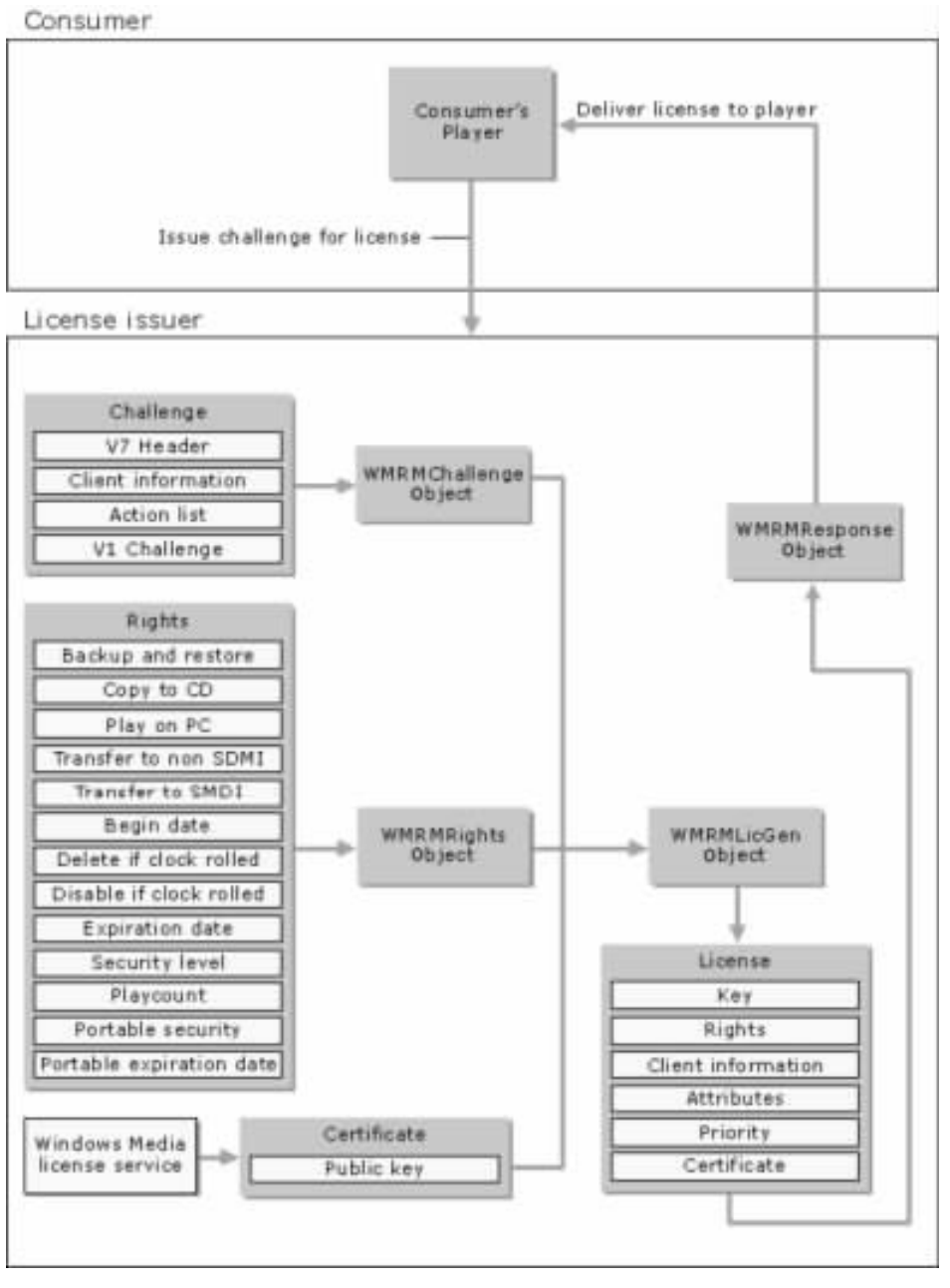


图 6.12　验证过程使用组件

6.3.1　WMRMChallenge 对象

WMRMChallenge 对象储存一个客户的请求信息，包括：

①创建 Key 使用的 Key ID。

②鉴定播放内容的 Content ID。

③确定播放内容的属性。

④获取验证的 URL 地址。
⑤播放内容的 Drm 版本。
⑥加密过的客户 ID(Client ID)。
⑦客户机的 Drm 版本。
⑧客户机的个性版本号码。
WMRMChallenge 对象包含如表 6.3 所示。

表 6.3　WMRMChallenge 对象属性

属性	注释
Action	Retrieves a specific action from the list of actions requested in a challenge.
ActionCount	Retrieves the number of actions requested in a challenge.
Challenge	Specifies and retrieves the license request.
ClientAttribute	Retrieves a specific name-value pair from the client information section of a challenge.
ClientInfo	Retrieves the consumer information section from a challenge.
Header	Retrieves the header section from a challenge.
V1Challenge	Retrieves the version 1 license request, if present, from the version 7 challenge.

具体应用示例代码如下：


```
// 假设内容发布者发送加密以及头部信息(Header)至最终用户,客户机决定
// 必须使用一个 License 才可以播放这个内容,便发送请求信息至认证服务器

// 请求信息包含头部信息(Header),以及客户机将要进行的行为(一般为
// Play,即播放这个内容)
//
// 关于 Header 类型,可参考 WMRMHeader  对象


// 定义变量

    Dim sHeader, sClientInfo, sPubKey, lResult, dwActionCnt
    Dim sLicRequest, lIndex, sVerClient, sVerSecurity, sAppSecurity
    Dim sAction()
    Dim HeaderObj
    Dim ChallengeObj
```

```
// 创建对象

    Set HeaderObj = Server.CreateObject("Wmrmobjs.WMRMHeader")
    Set ChallengeObj = Server.CreateObject("Wmrmobjs.WMRMChallenge")

// 接受请求客户端信息

    sLicRequest = request.Form("challenge")

// 指定请求信息给 WMRMChallenge 对象

    ChallengeObj.Challenge = sLicRequest

// 在请求信息中获得头部信息

    sHeader = ChallengeObj.Header

// 指定头部信息于 WMRMHeader 对象

    HeaderObj.Header = sHeader

// 校验头部信息公匙

    lResult = HeaderObj.Verify(sPubKey)
    if (lResult <> 0) then
        // 头部信息损坏
    end if

// 获得客户信息

    sClientInfo = ChallengeObj.ClientInfo

// 获得客户请求动作,"Play" 是惟一的动作

    dwActionCnt = ChallengeObj.ActionCount()

    for lIndex = 0 to dwActionCnt - 1
        if (lIndex=0 And sAction(lIndex) <>"Play") Then
            // 不批准此行为
```

```
        end if
        if (lIndex > 0) Then
              // 行为批准
        end if
    next

// 获得请求属性

    sVerClient  =  ChallengeObj. ClientAttribute(" CLIENTVERSION" )
        // 获得客户机 drm 版本

    sVerSecurity  =  ChallengeObj. ClientAttribute(" SECURITYVERSION")
        // 获得安全版本号码

    sAppSecurity    =  ChallengeObj. ClientAttribute(" APPSECURITY")
        // 获得应用程序安全级别
```

6.3.2　WMRMHeader 对象

WMRMHeader 对象包含打包内容的头部信息，如果想要创建多媒体内容为 Windows Media 的格式，那么必须为这个文件创建一个头部信息，它包含如下内容：

①创建 Key 使用的 Key ID。

②鉴定内容使用的内容 ID。

③内容属性。

④获得认证信息的 URL 地址。

⑤应用程序安全信息。

一个头部信息只含有一个 Key ID，如果重新创建 Key 值，必须保证拥有原始的 Key 值用来生成 Key ID，用以校验信息。内容 ID 并不是必须的。

WMRMHeader 对象包含的属性如表 6.4 所示。

表 6.4　WMRMHeader 对象的属性

属　性	注　释
Attribute	Specifies and retrieves name-value pairs.
ContentID	Specifies and retrieves the ID of the encrypted content.
Header	Specifies and retrieves the content header.
IndividualizedVersion	Specifies and retrieves the minimum version of the Windows Media Rights Manager dynamic link library (DLL) that is individualized for a specific computer.

续表

属　性	注　释
KeyID	Specifies and retrieves the ID of the encryption key.
LicenseAcqURL	Specifies and retrieves the URL of the Web page that the client must use to apply for a license.
Version	Retrieves the version number from the content header.

WMRMHeader 对象包含的方法如表 6.5 所示。

表 6.5　WMRMHeader 对象包含的方法

方　法	注　释
SetCheckSum	Specifies the checksum used for key verification.
Sign	Adds a signature to the content header.
Verify	Verifies the signature in the data section of the content header.

其中 Attribute 属性名称如表 6.6 所示。

表 6.6　Attribute 属性名称

属性名	注　释
Artist_URL	Contains the URL of the artist 's Web site.
Author	Contains the name of the content author.
Content_Distributor	Contains the name of the distributor of the content.
Content_Type	Contains the type of content, such as music or video.
Copyright	Contains the copyright of the content.
Description	Contains a description of the content.
License_Distributor	Contains the name of the license distributor.
License_Distributor URL	Contains the URL of the license issuer 's Web site.
Rating	Contains the rating indicating the type of audience for which the content is suitable.
Title	Name of the content.

使用这些属性、方法的代码可参考如下代码：

```
// 定义变量

    Dim sURL, sKID, sCID, sSeed, sKey, sHeader
    Dim sPrivKey, sPubKey
    Dim HeaderObj
    Dim KeysObj
```

```
    Dim ProtectObj

// 创建对象

    Set HeaderObj = Server.CreateObject("Wmrmobjs.WMRMHeader")
    Set KeysObj = Server.CreateObject("Wmrmobjs.WMRMKeys")
    Set ProtectObj = Server.CreateObject("Wmrmobjs.WMRMProtect")
//
// 创建 Key id、Seed、Key、Content ID,设置 Key ID 和 Content ID 至头部信息
// 共享 Seed 值,各认证功能,保存 Key
// 运行过程参考图 6.6

    sKID = KeysObj.GenerateKeyID()
    sSeed = KeysObj.GenerateSeed()
    KeysObj.KeyID = sKID
    KeysObj.Seed = sSeed
    sKey = KeysObj.GenerateKey()
    sCID = KeysObj.GenerateKeyID()

    HeaderObj.KeyID = sKID
    HeaderObj.ContentID = sCID

// 指定获得验证的 URL,写入头部信息

    sURL = "http://www.LicenseIssuer.com/getlicense.asp"
    HeaderObj.LicenseAcqURL = sURL

// 设置属性信息

    HeaderObj.Attribute("Author") = "作者名"
    HeaderObj.Attribute("Content_Distributor") = "内容发布者"
    HeaderObj.Attribute("Content_Type") = "内容类型"
    HeaderObj.Attribute("Title") = "内容标题"


// 创建公钥以及私钥,使用私钥标记头部信息,共享公钥于内容发布者
//
//
    Call KeysObj.GenerateSigningKeys ( sPrivKey, sPubKey)
```

```
    HeaderObj. Sign(sPrivKey)
// 重新获得内容头部信息
    sHeader = HeaderObj. Header
// 设置头部信息于 WMRMProtect 对象
    ProtectObj. Header = sHeader
// 设置 Key 值于 WMRMProtect 对象
    ProtectObj. Key = sKey
// 指定加密源文件
    ProtectObj. InputFile = "c:\input_filename. wmv"
// 输出保护文件
    ProtectObj. ProtectFile("c:\output_filename. wmv")
```


6.3.3 WMRMKeys 对象

WMRMKeys 对象用来生成头部信息必须的内容,与其他对象的关系如图 6.13 所示。包含的属性、方法分别如表 6.7 和表 6.8 所示:

表 6.7 WMRMKeys 对象属性

属 性	注 释
KeyID	Specifies and retrieves the key ID.
Seed	Specifies and retrieves the license key seed.

表 6.8 WMRMKeys 对象方法

方 法	注 释
GenerateKey	Creates the key used to encrypt content.
GenerateKeyID	Creates a globally unique key ID.
GenerateSeed	Creates a random license key seed.
GenerateSigningKeys	Creates a private and public key for signing and verifying the header.

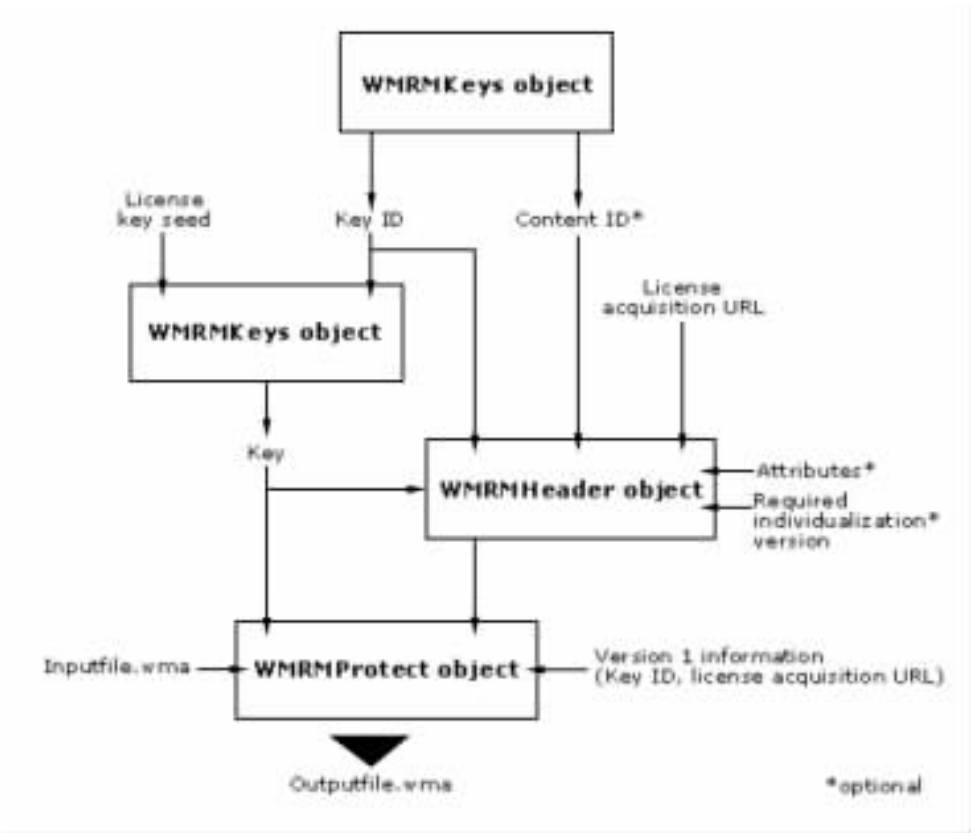


图 6.13　与其他对象的关系

具体使用方法请参考 6.3.2 示例代码。

6.3.4　WMRMLicGen 对象

验证机构需要使用 WMRMLicGen 对象来创建一个播放许可证，一个播放许可证包含如下的信息：

①加密保护内容的 Key 值。

②赋予请求客户的播放权限。

③客户机的信息。

④不是必须的名称信息。

⑤客户播放的许可证优先权。

⑥许可证信息。

WMRMLicGen 对象包含的属性、方法如表 6.9 和表 6.10 所示：

表 6.9 WMRMLicGen 对象包含的属性

属 性	注 释
Attribute	Specifies and retrieves name-value pairs in a license.
BindToPubKey	Specifies and retrieves a public key, and binds a license to the encrypted content by using this key.
ClientInfo	Specifies system information about the consumer's computer.
KeyID	Specifies and retrieves the key ID for a license.
Priority	Specifies and retrieves the priority of a license.
Rights	Specifies and retrieves the rights assigned by the license.

表 6.10 WMRMLicGen 对象包含的方法

方 法	注 释
GetClientVersion	Retrieves the individualized version of Windows Media Rights Manager running on the client computer.
GetLicenseToDeliver	Creates a license for delivery to the consumer.
SetKey	Specifies the private key used in the license.

WMRMLicGen 对象是发布证书所必须使用的对象,其中 Attribute 属性名称参见表 6.6,使用方法如下所示:


```
// 假设内容发布者将一个加密内容与头部信息(Heaser)发送给一个用户,
// 用户机发布请求信息(Challenge)至认证机构。请求信息包含客户机的
// 必需要信息,要生成一个播放认证,需要执行如下步骤:

// 定义变量

    Dim sHeader, sClientInfo, sPubKey, lResult, dwActionCnt
    Dim sLicRequest, lIndex, sRights, sKeyID, sSeed, sKey
    Dim varCategory, varVersion, sLicense, sLicResponse
    Dim sAction
    Dim HeaderObj
    Dim ChallengeObj
    Dim Rights Obj
    Dim LicGenObj
    Dim ResponseObj

// 创建对象
```

```
    Set HeaderObj = Server.CreateObject("Wmrmobjs.WMRMHeader")
    Set ChallengeObj = Server.CreateObject("Wmrmobjs.WMRMChallenge")
    Set RightsObj = Server.CreateObject("Wmrmobjs.WMRMRights")
    Set LicGenObj = Server.CreateObject("Wmrmobjs.WMRMLicGen")
    Set KeysObj = Server.CreateObject("Wmrmobjs.WMRMKeys")
    Set ResponseObj = Server.CreateObject("Wmrmobjs.WMRMResponse")

// 获得客户机请求信息

    sLicRequest = request.Form("challenge")

// 设置请求信息值 WMRMChallenge 对象

    ChallengeObj.Challenge = sLicRequest

// 从请求信息中获得头部信息

    sHeader = ChallengeObj.Header

// 设置头部信息至 WMRMHeader 对象

    HeaderObj.Header = sHeader

// 使用内容发布者的公钥校验头部信息

    lResult = HeaderObj.Verify(sPubKey)
    if (lResult <> 0) then
        // 头部信息无效
    end if

// 从请求信息中获得客户信息

    sClientInfo = ChallengeObj.ClientInfo

// 获得必要的客户信息

    dwActionCnt = ChallengeObj.ActionCount()

    for lIndex = 0 to dwActionCnt - 1
```

```
        if (lIndex = 0 And sAction(lIndex) < > "Play") Then
            // 客户请求行为无效
        end if
        if (lIndex > 0) Then
            // 客户请求行为有效
        end if
    next

// 定义客户权限,例如播放次数等

    RightsObj. Reset
    RightsObj. AllowBackupRestore = True
    RightsObj. AllowPlayOnPC = True
    RightsObj. AllowTransferToSDMI = True
    RightsObj. DisableOnClockRollback = True
    RightsObj. TransferCount = 10
    RightsObj. Playcount = 100
    RightsObj. BeginDate = "#20000101Z#"
    RightsObj. ExpirationDate = "#20001231Z#"
    RightsObj. PMExpirationDate = "#20000930Z#"

// 获得设置过的权限信息

    sRights = RightsObj. GetAllRights()

// 将权限信息赋予许可证对象

    LicGenObj. Rights = sRights

// 将客户信息赋予许可证对象

    LicGenObj. ClientInfo = sClientInfo

// 检查版本信息

    LicGenObj. GetClientVersion(varCategory, varVersion)

    if (varVersion < > 513) then
```

```
        // 不正确的操作系统
    end if

    if (varCategory = 0) then
        // 操作系统为 Windows 系列
    end if

    if (varCategory = 1) then
        // 特别的 Windows 操作系统
    end if


// 使用 BindToPubKey 属性发送客户端公钥校验许可证

    LicGenObj.BindToPubKey = sPubKey

// 设置许可证属性内容

    LicGenObj.Attribute("Author") = "name of the author"
    LicGenObj.Attribute("Content_Distributor") = "name of the distributor"
    LicGenObj.Attribute("Content_Type") = "type of content distributed"
    LicGenObj.Attribute("Title") = "title of the content"

// 设置许可证优先权，默认为 1

    LicGenObj.Priority = 10

// 将客户端的 Key Id 赋予许可证对象

    sKeyID = HeaderObj.KeyID
    LicGenObj.KeyID = sKeyID

// 使用 Key ID 和 Seed 可以获得 Key,并将 Key 值赋予许可证变量
// 内容发布者和认证机构使用的是同一个 Seed 值

    KeysObj.KeyID = sKeyID
    KeysObj.Seed = sSeed
    sKey = KeysObj.GenerateKey()
    LicGenObj.SetKey("MSDRM", sKey)
```

```
// 创建许可证
    sLicense = LicGen.GetLicenseToDeliver()
// 将输出许可证加入许可证信息
    ResponseObj.AddLicense("2.0.0.0", sLicense)
// 获得包含输出许可证的字符串
    sLicResponse = Response.GetLicenseResponse()
```

6.3.5 WMRMProtect 对象

WMRMProtect 对象负责加密内容,并将加密内容于头部信息打包至一个 Windows Media 文件,WMRMProtect 对象包含的属性与方法如表 6.11 和表 6.12 所示。

表 6.11 WMRMProtect 对象包含的属性

属 性	注 释
Header	Specifies and retrieves the content header.
InputFile	Specifies and retrieves the name of an input file.
Key	Specifies and retrieves the content encryption key.
V1KeyID	Specifies and retrieves the key ID for the version 1 content encryption object.
V1LicenseAcqURL	Specifies and retrieves the license acquisition URL for the version 1 content header.

表 6.12 WMRMProtect 对象包含的方法

方 法	注 释
ProtectFile	Encrypts the file specified by the InputFile property.
WriteFile	Writes an encrypted file with an updated content header.

属性与方法的使用方法如下所示：

```
// 定义变量与对象
    Dim sKID, sCID, sSeed, sKey, sHeader
    Dim sPrivKey, sPubKey
    Dim HeaderObj
```

```
    Dim KeysObj
    Dim ProtectObj

// 创建对象

    Set HeaderObj = Server.CreateObject("Wmrmobjs.WMRMHeader")
    Set KeysObj = Server.CreateObject("Wmrmobjs.WMRMKeys")
    Set ProtectObj = Server.CreateObject("Wmrmobjs.WMRMProtect")

// 创建 Key id、Seed、Key、Content ID,设置 Key ID 和 Content ID 至头部信息
// 共享 Seed 值,各认证功能,保存 Key
// 运行过程参考图 6.6


    sKID = KeysObj.GenerateKeyID()
    sSeed = KeysObj.GenerateSeed()
    KeysObj.KeyID = sKID
    KeysObj.Seed = sSeed
    sKey = KeysObj.GenerateKey()
    sCID = KeysObj.GenerateKeyID()

    HeaderObj.KeyID = sKID
    HeaderObj.ContentID = sCID

// 创建公钥以及私钥,使用私钥标记头部信息,共享公钥于内容发布者
//


    Call KeysObj.GenerateSigningKeys ( sPrivKey, sPubKey)
    HeaderObj.Sign(sPrivKey)

// 重新获得内容头部信息

    sHeader = HeaderObj.Header

// 设置头部信息于 WMRMProtect 对象

    ProtectObj.Header = sHeader
```

```
// 设置 Key 值

    ProtectObj. Key = sKey

// 指定加密源文件

    ProtectObj. InputFile = "c :hinput_file. wmv"

// 输出保护文件

    ProtectObj. ProtectFile("c :houtput_file_1. wmv")

// 如果改变了头部信息,可以使用 WriteFile 方法来更新收保护的数据
// 而不用重新加密一遍数据

    HeaderObj. Attribute("Author") = "name of the author"
    HeaderObj. Attribute("Content_Distributor") = "name of the distributor"
    ProtectObj. InputFile = "C :houtput_file_1. wmv"
    ProtectObj. WriteFile("c :houtput_file_2. wmv")
```


6.3.6 WMRMResponse 对象

WMRMResponse 对象负责创建一个认证的回复,WMRMResponse 对象与其他对象关系如图 6.14 所示。

WMRMResponse 对象的属性与方法如表 6.13 和表 6.14 所示。

表 6.13 WMRMResponse 对象的属性

属 性	注 释
ReplaceQuotesWith	Specifies a replacement for the embedded quotation marks in the string that is returned by the GetLicenseResponse method.

表 6.14 WMRMResponse 对象的方法

方 法	注 释
AddLicense	Adds a license to the license response.
GetLicenseResponse	Retrieves a string containing the license response.

AddLicense 需要的一个指定 Drm 版本的参数,如表 6.15 所示。

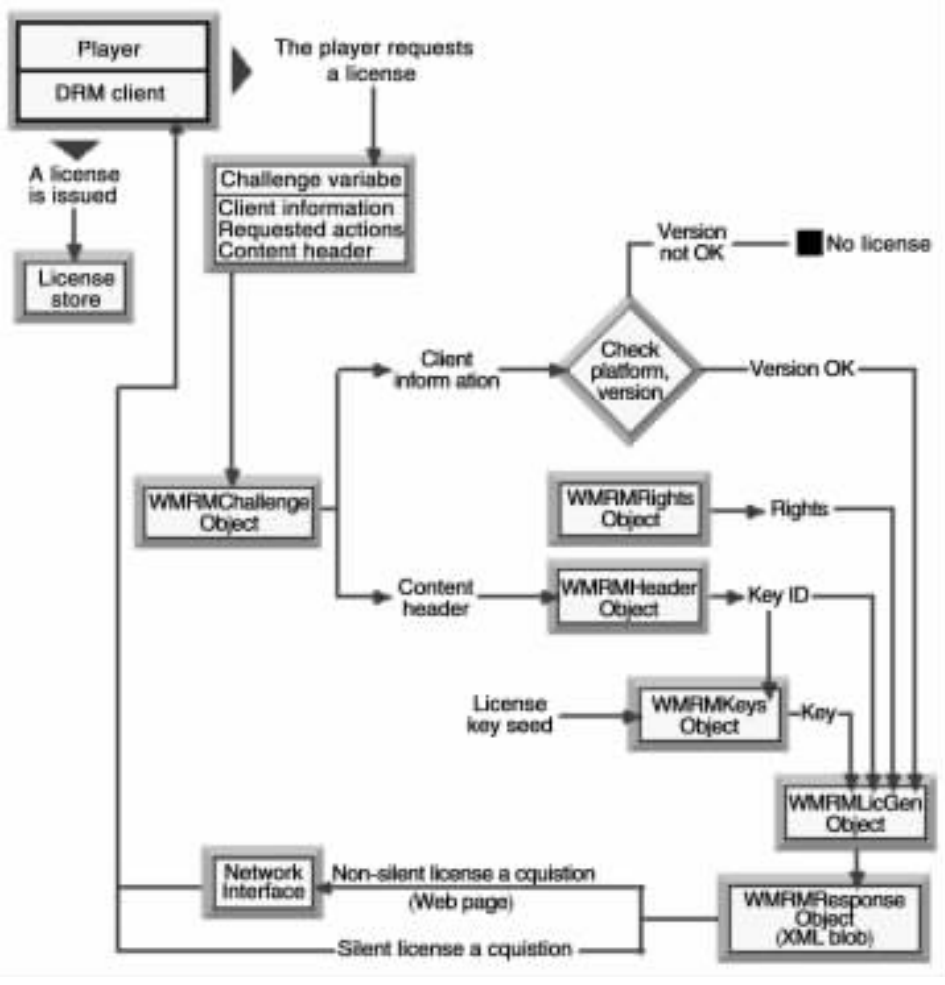


图 6.14　WMRMResponse 对象与其他对象关系

表 6.15　AddLicense 参数

值	含　义
1.0.0.0	Specifies version 1 of Windows Media Rights Manager.
2.0.0.0	Specifies version 7 of Windows Media Rights Manager.

WMRMResponse 对象的使用方法如下所示：

```
//
// 一个认证回复,可以包含多个许可证,如果发送多个许可证给用户,必须创建
// WMRMresponse 对象,将每一个许可证包含进去,并发出认证回复
// 生成许可证请参考 WMRMLicGen  对象
//
```

```
// 定义变量

    Dim sLicense()        // 存放许可证的数组
    Dim lLicCount         // 许可证数量
    Dim sLicResponse
    Dim ResponseObj

// 创建 WMRMResponse 对象

    Set ResponseObj = Server.CreateObject("Wmrmobjs.WMRMResponse")

// 将每一个许可证加入认证回复

    for i = 0 to lLicCount - 1
        ResponseObj.AddLicense("2.0.0.0", sLicense(i))
    next

// 发送许可证至客户端

    sLicResponse = Response.GetLicenseResponse()
```


6.3.7 WMRMRights 对象

WMRMRights 对象指定客户拥有的权限，如播放次数、播放环境等，WMRMRights 对象的属性与方法如表 6.16 和表 6.17 所示。

表 6.16 WMRMRights 对象的属性

属 性	注 释
AllowBackupRestore	Specifies and retrieves a Boolean value that indicates whether the license permits backup and restoration.
AllowBurnToCD	Specifies and retrieves a Boolean value that indicates whether the license permits content to be copied to a CD in the RedBook Audio format.
AllowPlayOnPC	Specifies and retrieves a Boolean value that indicates whether the license permits content to be played on a client computer.
AllowTransferToNonSDMI	Specifies and retrieves a Boolean value that indicates whether the license permits content to be transferred to non-SDMI compliant portable devices or portable media.

续表

属　性	注　释
AllowTransferToSDMI	Specifies and retrieves a Boolean value that indicates whether the license permits content to be transferred to SDMI compliant portable devices or portable media.
BeginDate	Specifies and retrieves the date before which the license is not valid.
BurnToCDCount	Specifies and retrieves the number of times that content can be copied to a CD.
DeleteOnClockRollback	Specifies and retrieves a Boolean value that indicates whether a license must be deleted if the clock is set to an earlier time.
DisableOnClockRollback	Specifies and retrieves a Boolean value that indicates whether a license must be disabled if the clock isset to an earlier time.
ExpirationDate	Specifies and retrieves the date after which the license is no longer valid.
MinimumAppSecurity	Specifies and retrieves the minimum security level that a player must have to manipulate the content.
Playcount	Specifies and retrieves the number of times the license permits content to be played.
PMAppSecurity	Specifies and retrieves the security level for content that is being transferred to portable devices or portable media.
PMExpirationDate	Specifies and retrieves the expiration date for a media license.
PMRights	Specifies and retrieves the rights that govern content use with a portable license.
TransferCount	Specifies and retrieves the number of times the content can be transferred to portable devices or portable media.

表 6.17　WMRMRights 对象的方法

方　法	注　释
GetAllRights	Packages all rights into a string for delivery to the WMRMLicGen object.
GetSAPMode	Retrieves the secure audio path（SAP）mode required for a license.
Reset	Resets the internal state of the rights object.
SetSAPMode	Specifies the SAP mode required for a license.

PMRights 属性可以简便地指定权限，如表 6.18 所示。

表 6.18 PMRights 属性可以简便地指定权限

值	权 限	设 置
1	Play on portable device.	0 Allow playback. 1 Do not allow playback.
2	Transfer non-SDMI content.	0 Do not allow transfer. 1 Allow transfer.
4	Allow consumer to back up and restore license.	0 Do not allow consumer to restore. 1 Allow consumer to restore.
8	Copy content to a CD.	0 Do not allow copy to a CD. 1 Allow copy to a CD.
16	Transfer SDMI content.	0 Do not allow SDMI transfer. 1 Allow SDMI transfer.

WMRMRights 对象的示例代码如下：

```
// 当发布一个许可证之前,必须明确指出客户机所拥有的权限
// 并保存在 WMRMLicGen 对象中


// 定义变量

    Dim sRights
    Dim RightsObj
    Dim LicGenObj


// 创建对象

    Set RightsObj = Server.CreateObject("Wmrmobjs.WMRMRights")
    Set LicGenObj = Server.CreateObject("Wmrmobjs.WMRMLicGen")

// 重新设置 WMRMRights 对象

    RightsObj.Reset()

// 指定客户机拥有权限

    RightsObj.AllowBackupRestore = True
    RightsObj.AllowPlayOnPC = True
    RightsObj.DisableOnClockRollback = True
```

```
    RightsObj. TransferCount = 10
    RightsObj. Playcount = 100
    RightsObj. BeginDate = "#20000101Z#"
    RightsObj. ExpirationDate = "#20001231Z#"
    RightsObj. PMExpirationDate = "#20000930Z#"

// 获得权限

    sRights = RightsObj. GetAllRights()

// 设置权限

    LicGenObj. Rights = sRights
```

6.3.8　LicenseGenerator 对象

LicenseGenerator 对象用来生成 Windows Media DRM 1 的许可证，并不适用于 Windows Media DRM 7，其方法如表 6.19 所示。

表 6.19　LicenseGenerator 对象方法

方　法	注　释
DeleteIssueEntry	Deletes an entry from the version 1 license database.
Get	Retrieves license attribute values.
InstallKeys	Installs new public and private keys in the database.
IssueLicense	Creates the license from a challenge.
Set	Specifies license attributes.

6.4　实例：一个应用方案

以上了解了 DRM 的原理，同时也加密了一个节目，但是应注意到，加密过的节目不经过任何用户检验过程就直接获取了播放证书，这样不能达到保护节目版权的目的，所以在 DRM 的实际应用中，还要作一些开发工作，把针对用户的验证加入 DRM 系统中。

6.4.1 制作验证页面

其代码如下：

```
// 输入用户提交验证信息页面,文件名为 userlogin.asp
<% Response.Buffer = true %>
<%
 session("challenge") = request.form("challenge")
 session("nonsilent") = request.Form("nonsilent")
%>
<html>
<head>
</head>
<body bgcolor = "#FFFFFF" text = "#000000">
<form name = management action = management.asp method = post onsubmit = check()>
<table width = "320" border = "0" height = "240" align = "center">
    <tr>
      <td colspan = "2" height = "51">
        <div align = "center">
          <p> </p>
        </div>
      </td>
    </tr>
    <tr>
      <td width = "32%">
        <div align = "right">管理员：</div>
      </td>
      <td width = "68%">
        <input type = "text" name = "user_id" size = "35">
      </td>
    </tr>
    <tr>
      <td width = "32%">
        <div align = "right">密码：</div>
      </td>
      <td width = "68%">
        <input type = "password" name = "user_pwd" size = "35">
      </td>
```

```
    </tr>
    <tr>
      <td colspan="2">
        <div align="right"></div>
        <div align="center">
          <input type="submit" name="Submit" value="　确定　">
          <input type="reset" name="Submit2" value="　清除　">
        </div>
      </td>
    </tr>
  </table>
</form>
<script language=vbscript>
sub check()
set frm=document.management
if frm.user_id.value="" then
alert("please input your name!")
window.event.returnvalue=false
end if
if frm.user_pwd.value="" then
alert("please input password!!!")
window.event.returnvalue=false
end if
end sub
</script>
</body>
</html>
```


6.4.2　判断用户验证信息

文件名为 management.asp**，代码如下：**

```
<%
user_id=trim(Request.Form("user_id"))
user_pwd=trim(Request.Form("user_pwd"))
if user_id <> "Test" and user_pwd="111111" then
Response.Redirect "http://192.168.1.66/sdk/simple.asp"
else
```

```
Response. Write " < center > < p > < font color = red size = 4 > 你不是正式用户,登
录失败 < /font > < /p > "
Response. Write " < p > < a href = userlogin. asp > 点击这里再次登录 < /a > < /p > "
Response. Write " < br > < /center > "
end if
% >
```

首先,将用户在表单提交的数据交由 management. asp 判断,如果符合用户名为“Test”,密码为“111111”的情况,则重定向到认证页面 http://192. 168. 1. 66/sdk/simple. asp。

怎么让节目在播放的时候链接到我们刚作的页面呢?很简单,只要修改打包文件的源代码,在 package. asp 的 Version 7 license acquisition URL 下拉菜单处添加一个自定义选项,把刚刚作的用户登录页面名添加上,再打包的时候选中就可以了。

6.4.3 传送被加密节目的 Header 信息到认证页面

在我们第一次打包文件时,由播放器建立的表单信息将加密节目的 header 信息直接提交给了认证页面 simple. asp,而认证页面正是按照这些信息来生成解密密钥的。在这个例子中,由于中间要进行用户判断,所以要使用 session 来传递 header 信息,把以下代码分别插入 webmaster. asp 和 management. asp 中。

```
session("challenge") = request. form("challenge")    // 插入 webmaster. asp
session("challenge") = session("challenge")    // 插入 management. asp
```


文件的 header 信息是通过 challenge 来从播放器处获取的。同样,在认证页面 simple. asp 中,也要修改对应的地方,使之可以获取从登陆页面传来的 session 信息。

用 strLicenseRequested = session("challenge") 取代原来的 strLicenseRequested = Request. Form("challenge")

保存这几个文件到对应的虚拟目录。

好了,测试一下成果吧。访问用户登录页面 http://localhost/DRM/userlogin. asp. 输入用户名“Test”,密码为“111111”,如果输入无误,页面会重定向至获取验证页面,如果输入了错误的用户名和密码,则会提示重新登录。

以上只是 Windows Media DRM 在流媒体上应用的一个简单实例。实际上,真正的 DRM 应用方案要比这个复杂得多,我们不可能用一对钥匙负责所有节目的加密解密(一旦密钥外泻,损失巨大),对于一些价值含量较高的媒体,最好采用一对密钥

加密一个节目的方式。同时，也可以使用多种用户验证方式，如建立用户库，设置不同权限的用户以用于不同的授权，验证的方式可以采用 IC 卡，甚至可以是使用者的指纹，这些都可以针对商家的具体需要采用对应的方案。

6.5　Windows Media DRM 常用错误信息

表 6.20 列出了使用 Windows Meida DRM 的常用错误信息代码，了解这些代码，对调试程序非常有帮助。

表 6.20　Windows Meida DRM 的常用错误信息代码

返回代码	解　释	十六进制值
DRM_E_ATTRIBUTE_NOT_FOUND	The requested attribute cannot be found.	0xC0042906L
DRM_E_BINDTOPUBKEY_NOT_SET	The BindToPubkey property must be set before calling this method.	0xC0042921L
DRM_E_CANNOT_WRITEFILE_UNPROTECTED	Cannot call the WriteFile method on an unprotected input file.	0xC004290DL
DRM_E_CH_ATTR_MISSING	Missing content header attribute.	0x80041107L
DRM_E_CH_BAD_KEY	Invalid key.	0x8004110EL
DRM_E_CH_CHECKSUM_MISSING	Missing content header checksum.	0x80041106L
DRM_E_CH_CHECKSUM_NOTSET	Checksum not set.	0x80041111L
DRM_E_CH_HEADER_NOTSET	Header not set.	0x80041112L
DRM_E_CH_INVALID_HEADER	Invalid content header.	0x80041108L
DRM_E_CH_INVALID_PRIVATEKEY	Invalid private key.	0x80041114L
DRM_E_CH_INVALID_PUBLICKEY	Invalid public key.	0x80041115L
DRM_E_CH_KID_MISSING	Missing KID attribute in content header.	0x80041104L
DRM_E_CH_KID_NOTSET	KID not set.	0x8004110FL
DRM_E_CH_LAINFO_MISSING	Missing LAINFO attribute in content header.	0x80041105L
DRM_E_CH_LAINFO_NOTSET	LAINFO not set.	0x80041110L
DRM_E_CH_MEMORY_ALLOCATION_ERROR	Memory allocation failure.	0x80041102L
DRM_E_CH_NOT_SIGNED	Header not signed.	0x80041113L
DRM_E_CH_PARAM_NOT_OPTIONAL	The parameter is not optional.	0x80041101L
DRM_E_CH_UNABLE_TO_SIGN	Unable to sign content header.	0x80041109L
DRM_E_CH_UNABLE_TO_VERIFY	Unable to verify signature of content header.	0x8004110AL
DRM_E_CH_UNKNOWN_ERROR	Unknown error.	0x80041116L

续表

返回代码	解释	十六进制值
DRM_E_CH_UNSUPPORTED_HASH_ALGORITHM	Unsupported hash algorithm.	0x8004110CL
DRM_E_CH_UNSUPPORTED_SIGN_ALGORITHM	Unsupported signature algorithm.	0x8004110DL
DLDRM_E_CH_UNSUPPORTED_VERSION	Unsupported content header version.	0x8004110BL
DRM_E_CH_VERSION_MISSING	Missing content header version.	0x80041103L
DRM_E_CORRUPT_CHALLENGE	The challenge string is corrupt.	0xC0042907L
DRM_E_DRMV1_NO_SUPPORT	This method doesn't support Windows Media Rights Manager version1.0.	0xC004291FL
DRM_E_DRMXML_ATTR_NOT_FOUND	The parameter is not optional.	0x8004100BL
DRM_E_DRMXML_CURR_NO_VALUE	The parameter is not optional.	0x80041008L
DRM_E_DRMXML_CURR_NOT_SET	The parameter is not optional.	0x80041007L
DRM_E_DRMXML_INIT_FAILURE	Object initialization failure.	0x80041004L
DRM_E_DRMXML_INVALID_ATTR_INDE	The parameter is not optional.	0x80041009L
DRM_E_DRMXML_INVALID_CHILD_INDE	The parameter is not optional.	0x8004100AL
DRM_E_DRMXML_INVALID_XML	Invalid XML.	0x80041003L
DRM_E_DRMXML_MEMORY_ALLOCATION_ERROR	Memory allocation failure.	0x80041002L
DRM_E_DRMXML_PARAM_NOT_OPTIONAL	The parameter is not optional.	0x80041001L
DRM_E_DRMXML_RETRIEVAL_FAILURE	The parameter is not optional.	0x80041006L
DRM_E_DRMXML_TREE_EMPTY	The parameter is not optional.	0x80041005L
DRM_E_DRMXML_UNSUPPORTED_NODE	The parameter is not optional.	0x8004100CL
DRM_E_HEADER_NOT_SET	The Header property must be set before calling this method.	0xC0042912L
DRM_E_HEADER_TOO_Long	The content header cannot exceed 8 KB in length.	0xC004290CL
DRM_E_INPUT_ALREADY_PROTECTED	Cannot call the PretectFile method on a protected input file.	0xC004290EL
DRM_E_INPUT_ASF_CORRUPT	The input file is corrupt.	0xC0042919L
DRM_E_INPUT_NOT_ASF	The input must be a Windows Media File.	0xC0042918L
DRM_E_INPUT_NOT_PROTECTED	The input file must be protected.	0xC004291AL
DRM_E_INPUTFILE_NOT_SET	The InputFile property must be set before calling this method.	0xC0042911L
DRM_E_INVALID_DATE	The specified date is invalid.	0xC004290AL
DRM_E_INVALID_INDE	Invalid index value.	0xC0042909L

续表

返回代码	解　释	十六进制值
DRM_E_KEY_GENERATION_FAILURE	Key pair generation failed.	0xC004291BL
DRM_E_KEY_NOT_SET	The Key property must be set before calling this method.	0xC0042913L
DRM_E_KEYID_NOT_SET	The KeyID property must be set before calling this method.	0xC0042910L
DRM_E_LIC_BAD_KEY	Bad key.	0x8004800CL
DRM_E_LIC_CERTIFICATE_FAILURE	Certificate failure.	0x80048009L
DRM_E_LIC_CERTIFICATE_NOTSET	Certificate not set.	0x8004800AL
DRM_E_LIC_CLIENDID_DECODING_FAILURE	Client ID decoding failure.	0x8004800FL
DRM_E_LIC_CLIENT_EXCLUSION_NOTSET	The client exclusion information has not been set.	0x80048017L
DRM_E_LIC_CLIENT_NEEDS_INDIVIDUALIZATION	A security upgrade on the client is required before a license can be issued.	0xC0048018L
DRM_E_LIC_CLIENTID_NOTSET	Client ID not set.	0x80048005L
DRM_E_LIC_ENABLING_BITS_FAILURE	Enabling bits failure.	0x8004800DL
DRM_E_LIC_INTERNAL_ERROR	Internal error.	0x80048004L
DRM_E_LIC_INVALID_PARAMETER	Invalid parameter.	0x80048014L
DRM_E_LIC_INVALID_RIGHTS	Invalid rights.	0x8004800BL
DRM_E_LIC_KEY_AND_CERT_MISMATCH	Key and certificate mismatch.	0x80048013L
DRM_E_LIC_KEY_DECODE_FAILURE	Key decode failure.	0x80048007L
DRM_E_LIC_KEY_NOTSET	Key not set.	0x80048012L
DRM_E_LIC_KID_NOTSET	The Key ID has not been set.	0x80048015L
DRM_E_LIC_LICENSE_NOT_SIGNED	License not signed.	0x80048006L
DRM_E_LIC_MEMORY_ALLOCATION_ERROR	Memory allocation error.	0x80048002L
DRM_E_LIC_OBJECT_INIT_ERROR	Object initialization error.	0x80048003L
DRM_E_LIC_PARAM_NOT_OPTIONAL	Parameter not optional.	0x80048001L
DRM_E_LIC_RIGHTS_NOTSET	Rights not set.	0x80048019L
DRM_E_LIC_SECURITY_VERSION_NOTSET	The security version number has not been set.	0x80048016L
DRM_E_LIC_SET_ATTRIBUTE_FAILURE	Set attribute failure.	0x80048010L
DRM_E_LIC_SIGNATURE_FAILURE	License signature failure.	0x80048008L
DRM_E_LIC_UNAUTHORIZED_DRM_CLIENT	Unauthorized DRM component.	0x8004800EL

续表

返回代码	解　释	十六进制值
DRM_E_LIC_UNSUPPORTED_KEY_ENCODING_ALGORITHM	Unsupported key encoding algorithm.	0x80048011L
DRM_E_LICENSE_SERVER_INFO_MISSING	The license server registry information (private key and/or certificates and/or verification keys) is missing or inaccessible.	0xC004291CL
DRM_E_ONSTARTPAGE_NOT_CALLED	The OnStartPage method is not called.	0xC004291EL
DRM_E_PROPERTY_NOT_SET	The property has not been set.	0xC0042905L
DRM_E_REQUIRED_PROPERTY_NOT_SET	One or more required properties have not been set.	0xC0042908L
DRM_E_RIGHTS_INTERNAL_ERROR	Parameter not optional.	0x80041503L
DRM_E_RIGHTS_INVALID_ACTION	Parameter not optional.	0x80041506L
DRM_E_RIGHTS_INVALID_CONDITION	Parameter not optional.	0x80041505L
DRM_E_RIGHTS_MEMORY_ALLOCATION_ERROR	Parameter not optional.	0x80041502L
DRM_E_RIGHTS_PARAM_NOT_OPTIONAL	Parameter not optional.	0x80041501L
DRM_E_RIGHTS_UNSUPPORTED_EVENT	Parameter not optional.	0x80041504L
DRM_E_SECURITY_COMPONENT_EXCLUDED	The security component is excluded.	0xC004291DL
DRM_E_SEED_NOT_SET	The Seed property must be set before calling this method.	0xC004290FL
DRM_E_SIGNCERTCN_NOT_SET	The SignCertCN property must be set before calling this method.	0xC0042916L
DRM_E_SIGNING_FAILED	The content could not be digitally signed.	0xC004290BL
DRM_E_SIGNSTORENAME_NOT_SET	The SignStoreName property must be set before calling this method.	0xC0042915L
DRM_E_UNKNOWN_FUNCTION	The input name is not supported by this method.	0xC0042920L
DRM_E_V1KID_NOT_SET	The V1KeyID property must be set before calling this method.	0xC0042914L
DRM_E_V1LICENSEURL_NOT_SET	The V1LicenseAcqURL property must be set before calling this method.	0xC0042917L
DRM_S_CH_NOERROR	The function succeeded.	0x00041100L
DRM_S_DRMXML_NOERROR	The function succeeded.	0x00041000L
DRM_S_LIC_NOERROR	The function succeeded.	0x00048000L
DRM_S_RIGHTS_NOERROR	The function succeeded.	0x00041500L

表 6.21 是 Windows Media Player 播放使用 Windows Meida DRM 打包后的流媒体文件的错误信息。

表 6.21　Windows Media Player 播放使用 Windows Meida DRM 打包后的流媒体文件的错误信息

返回代码	解　释	十六进制代码
NS_E_DRM_ACQUIRING_LICENSE	You cannot begin a new license acquisition process until the current one has been completed.	0xC00D272BL
NS_E_DRM_ACTION_NOT_QUERIED	The application cannot perform this action. Contact product support for this application.	0xC00D272AL
NS_E_DRM_APPCERT_REVOKED	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2752L
NS_E_DRM_BACKUP_CORRUPT	One or more backed-up licenses are missing or corrupt.	0xC00D2743L
NS_E_DRM_BACKUP_EXISTS	Licenses are already backed up in this location.	0xC00D2742L
NS_E_DRM_BACKUPRESTORE_BUSY	You cannot begin a new backup process until the current process has been completed.	0xC00D2744L
NS_E_DRM_CACHED_CONTENT_ERROR	A license cannot be found for this digital media file. Use License Management to transfer a license for this file from the original computer, or acquire a new license.	0xC00D274BL
NS_E_DRM_DECRYPT_ERROR	The digital media file is corrupted. Contact the content provider to get a new file.	0xC00D2723L
NS_E_DRM_DRIVER_AUTH_FAILURE	Certified driver components are required to play this digital media file. Contact Windows Update to see whether updated drivers are available for your hardware.	0xC00D274DL
NS_E_DRM_DRIVER_DIGIOUT_FAILURE	Certain driver functionality is required to play this digital media file. Contact Windows Update to see whether updated drivers are available for your hardware.	0xC00D2750L
NS_E_DRM_ENCRYPT_ERROR	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2722L

续表

返回代码	解　释	十六进制值
NS_E_DRM_ENCRYPTING	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D272DL
NS_E_DRM_ENUM_LICENSE_FAILED	License storage is not working. Contact Microsoft product support.	0xC00D271BL
NS_E_DRM_GET_ATTRIBUTE_ERROR	A problem has occurred in the Digital Rights Management component. Contact product support for this application.	0xC00D2745L
NS_E_DRM_GET_CONTENTSTRING_ERROR	The digital media file is corrupted. Contact the content provider to get a new file.	0xC00D273DL
NS_E_DRM_GET_LICENSE_ERROR	License storage is not working. Contact Microsoft product support.	0xC00D2739L
NS_E_DRM_GET_LICENSESTRING_ERROR	License storage is not working. Contact Microsoft product support.	0xC00D273CL
NS_E_DRM_HARDWARE_INCONSISTENT	The licenses for your digital media files are corrupted. Contact Microsoft product support.	0xC00D2754L
NS_E_DRM_INDIVIDUALIZATION_FAILED	The security upgrade failed. Try again later.	0xC00D2729L
NS_E_DRM_INDIVIDUALIZATION_INCOMPLETE	A problem occurred during the security upgrade. Try again later.	0xC00D274CL
NS_E_DRM_INDIVIDUALIZE_ERROR	The security upgrade failed. Try again later.	0xC00D2736L
NS_E_DRM_INDIVIDUALIZING	You cannot begin a new security upgrade until the current one has been completed.	0xC00D272CL
NS_E_DRM_INVALID_APPDATA	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2740L
NS_E_DRM_INVALID_APPDATA_VERSION	A problem has occurred in the Digital Rights Management component. Contact product support for this application.	0xC00D2741L
NS_E_DRM_INVALID_APPLICATION	A problem has occurred in the Digital Rights Management component. Contact product support for this application.	0xC00D2711L

续表

返回代码	解　释	十六进制值
NS_E_DRM_INVALID_CONTENT	The digital media file is corrupted. Contact the content provider to get a new file.	0xC00D2716L
NS_E_DRM_INVALID_LICENSE	The license is corrupted or invalid. Acquire a new license	0xC00D2718L
NS _ E _ DRM _ INVALID _ LICENSE _ ACQUIRED	License acquisition did not work. Acquire a new license or contact the content provider for further assistance.	0xC00D271FL
NS_E_DRM_INVALID_LICENSE_REQUEST	The digital media file is corrupted. Contact the content provider to get a new file.	0xC00D271CL
NS_E_DRM_INVALID_MACHINE	Licenses cannot be copied from one computer to another. Use License Management to transfer licenses，or get a new license for the digital media file.	0xC00D2719L
NS_E_DRM_INVALID_PROPERTY	A required property was not set by the application. Contact product support for this application.	0xC00D2749L
NS _ E _ DRM _ INVALID _ SECURESTORE _ PASSWORD	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2751L
NS_E_DRM_KEY_ERROR	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2721L
NS_E_DRM_LICENSE_CLOSE_ERROR	License storage is not working. Contact Microsoft product support.	0xC00D2738L
NS_E_DRM_LICENSE_EVAL_FAILED	The license is corrupted or invalid. Acquire a new license or reinstall the application.	0xC00D271AL
NS_E_DRM_LICENSE_INVALID_XML	The license is corrupted. Acquire a new license.	0xC00D2725L
NS_E_DRM_LICENSE_OPEN_ERROR	License storage is not working. Contact Microsoft product support.	0xC00D2737L
NS_E _DRM_LICENSE _PREVENTS _STORING	You cannot save this file.	0xC00D2724L
NS_E_DRM_LICENSE_STORE_ERROR	License storage is not working. Contact Microsoft product support.	0xC00D2712L

续表

返回代码	解　释	十六进制值
NS_E_DRM_LICENSE_STORE_SAVE_ERROR	License acquisition did not work. Acquire a new license or contact the content provider for further assistance.	0xC00D2714L
NS_E_DRM_LICENSE_UNUSABLE	The license is invalid. Contact the content provider for further assistance.	0xC00D2748L
NS_E_DRM_MONITOR_ERROR	A problem has occurred in the Digital Rights Management component. Try again later.	0xC00D273EL
NS_E_DRM_NEED_UPGRADE	A new version of the Digital Rights Management component is required. Contact product support for this application to get the latest version.	0xC00D274EL
NS_E_DRM_NEEDS_INDIVIDUALIZATION	A security upgrade is required to perform the operation on this digital media file.	0xC00D2728L
NS_E_DRM_NO_RIGHTS	The requested operation cannot be performed on this file.	0xC00D2720L
NS_E_DRM_PARAMETERS_MISMATCHED	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D272FL
NS_E_DRM_QUERY_ERROR	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D273AL
NS_E_DRM_REOPEN_CONTENT	Status message：Reopen the file.	0xC00D274FL
NS_E_DRM_REPORT_ERROR	A problem has occurred in the Digital Rights Management component. Contact product support for this application.	0xC00D273BL
NS_E_DRM_RESTORE_FRAUD	You cannot restore your license(s).	0xC00D2753L
NS_E_DRM_SDMI_NOMORECOPIES	You cannot make any more copies of this digital media file.	0xC00D2756L
NS_E_DRM_SDMI_TRIGGER	To transfer this digital media file，you must upgrade the application.	0xC00D2755L
NS_E_DRM_SECURE_STORE_ERROR	Secure storage is not working. Contact Microsoft product support.	0xC00D2713L
NS_E_DRM_SECURE_STORE_NOT_FOUND	A problem has occurred in the Digital Rights Management component of this application. Try to acquire a license again.	0xC00D274AL

续表

返回代码	解　释	十六进制值
NS_E_DRM_SECURE_STORE_UNLOCK_ERROR	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2715L
NS_E_DRM_UNABLE_TO_ACQUIRE_LICENSE	The license could not be acquired. Try again later.	0xC00D271EL
NS_E_DRM_UNABLE_TO_CREATE_BACKUP_OBJECT	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2735L
NS_E_DRM_UNABLE_TO_CREATE_DECRYPT_OBJECT	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2733L
NS_E_DRM_UNABLE_TO_CREATE_ENCRYPT_OBJECT	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2732L
NS_E_DRM_UNABLE_TO_CREATE_INDI_OBJECT	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2731L
NS_E_DRM_UNABLE_TO_CREATE_LICENSE_OBJECT	A license cannot be created for this digital media file. Reinstall the application.	0xC00D2730L
NS_E_DRM_UNABLE_TO_CREATE_PROPERTIES_OBJECT	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D2734L
NS_E_DRM_UNABLE_TO_INITIALIZE	A problem has occurred in the Digital Rights Management component. Contact Microsoft product support.	0xC00D271DL
NS_E_DRM_UNABLE_TO_OPEN_LICENSE	The license is corrupted. Acquire a new license.	0xC00D2717L
NS_E_DRM_UNABLE_TO_SET_PARAMETER	The application has made an invalid call to the Digital Rights Management component. Contact product support for this application.	0xC00D273FL
NS_S_DRM_ACQUIRE_CANCELLED	Status message：License acquisition has been cancelled.	0x000D2747L
NS_S_DRM_INDIVIDUALIZED	Status message：The security upgrade has been completed.	0x000D2727L
NS_S_DRM_LICENSE_ACQUIRED	Status message：The license was acquired.	0x000D2726L
NS_S_DRM_MONITOR_CANCELLED	Status message：License monitoring has been cancelled.	0x000D2746L

第7章 实例篇：商务综合应用

经过了前面 6 章的学习,读者是否已经对 Windows Media SDK 有了大概的了解?在本章中,将以一个大的商务应用为基础,开发出一套商业系统。其主要功能是完成用户对音乐文件的付费欣赏问题,下面将采用 Windows Media 为解决方案,一步一步地来定制这个系统。由于目前的市场上很多公司还是使用 Windows Media SDK 作为主要开发工具,而且以影音收费系统为主要产品,所以在这里不将视频点播收费作为讨论示例,但其中的道理是相同的,可以自己探索。

7.1 第一步:需求分析

收费试听音乐已经不新鲜了。在国外早已很普遍;而在国内,目前还处于尝试时期。作为这个商务方案的需求提供者,提出了以下几个基本需求:

①可以将各种音频的文件在用户符合条件的情况下被收听。

②保证音频文件的安全。

③保证计费系统的可靠性。

④保证系统的易用性。

⑤保证系统内容的扩张性。

这些需求是最普通的,与代码没有任何关系。现在让我们来一步一步将这些基本的需求分析为软件人员常见的最熟悉设计原则,其实就是将需求更详细地表述出来,使大家都在脑海中勾绘出这个商品方案应该是什么样子?具备哪些特性?

①产品是易于使用的,就是产品的易用性。产品可以灵活地被顾客使用,不需要特别的指导。

②产品是安全的,因为要使用网络交易,其安全问题显得特别突出。

③产品内容是受保护的,不是随意拷贝的。

④产品内容是可增加的、不断更新的。

那么设计出的软件应该具有什么功能?

①可以将多种音乐文件格式转换为一种易用的格式。

②将这些统一的音乐文件打包,加入版权保护。

③将这些加入版权的音乐文件放在网站上,供听众点播。

④听众点播时,应根据点播内容付费。

其实可以看得出来,软件的解决方案可以使用 Windows Media 来解决。可以使用 Encoder 将众多的音频文件转换为一个标准格式(WMA),使用 Windows Media DRM 将转换完毕的文件打包,使用 Windows Media Serivces 将内容发布出去,通过脚本程序来控制用户的验证收费。

现在,目的明确了,就开始为这个需求设计详细的方案。

7.2 第二步:设计方案

在本节,将详细地设计这个系统,系统可以被设计成为 6 个部分,如图 7.1 所示。

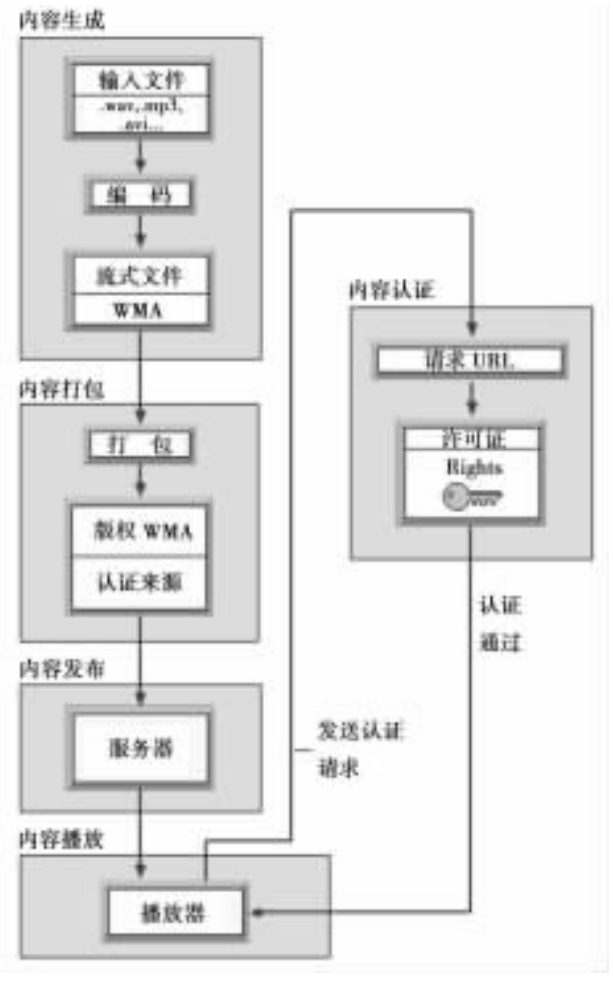


图 7.1　音乐收费视听系统架构

(1) 内容生成

将音乐文件统一转换为 WMA 格式，以便统一使用。最常见的音乐文件就是 MP3，还有 ASF 格式，以及 Windows 的 WAV 文件，将这些源音乐文件通过定制的 Encoder 程序统一转换。之所以要使用定制的 Encoder 是因为可以更灵活地控制，减少人力的投入。如果使用 Microsoft 的编码器，转换 1 000 个 MP3 为 WMA 格式，需要进行多次点击。所以，采用定制 Encoder 的方式来打包文件。

在定制的 Encoder 中，需要保证 WMA 的效果，采用 Audio for CD-quality transparency (128 kbit/s stereo) 的配置编码率。最重要的一点，定制的 Encoder 支持批量转换，在第 3 章中，已经设计了这个批量转换程序，现在只需要根据实际情况来完善所需功能。

(2) 内容打包

内容打包是指将编码后的 WMA 文件加入版权信息，并加入许可证获取地址。

这样,当用户获得这个打包后的 WMA 文件时,就要到指定的认证服务获得认证后才可以播放。由于编码后的 WMA 文件规格统一,所以可以定制程序来进行统一的打包。

这样可以将内容打包程序与内容发布程序设计成为一个程序,输入 MP3、WAV 的原始声音素材,通过内容打包发布程序直接生成可以含有版权信息的 WMA 文件。在生成过程中,主要经历了编码、打包两个过程。

(3) 内容发布

内容发布主要完成将编码打包后的 WMA 文件发送至用户端,发布的方式是多样的,例如:下载,拷贝,以光盘形式发出等,这里主要讨论通过 Windows Media Services 发布。定制 Windows Media Services 的操作程序,可以灵活的将编码打包后的文件,快速添加至 Windows Media Services 中。

(4) 内容播放

内容播放主要是在 Windows Media Service 中获取了信息后,将 WMA 文件还原为声音的过程。主要以 Windows Media Player 来完成,在此,我们主要完成从网页中播放 WMA 文件。

(5) 内容认证

内容认证主要是在 Windows Media Player 播放 WMA 文件前,要由认证服务器先取得认证信息。同时判断当前用户的播放条件,例如:是否有权限播放此 WMA 文件,是否登录,用户金额是否够等基本信息。在此,主要使用 ASP 脚本程序来辅助认证。

(6) 系统管理

系统管理主要完成用户计费、用户管理、申请、WMA 记录等。在本例中采用 IIS 作为 WWW 服务器,后台数据库为 Access 2000,使用 ASP 后台脚本程序开发,本部分不作为本例重点讲解。

7.3 第三步:代码编写

7.3.1 第一部分:编码打包工具

本部分主要完成编码打包工具,将 MP3 文件直接转换为具有版权信息的 WMA 文件,使用播放器播放是必须在指定认证服务器取得认证的工具。此工具为第 3 章与第 7 章的综合应用。

本工具的代码在配套光盘/Sample/Sample7_1 下。

在代码编写前,必须在 VB 工程中加入,对以下三个组件的引用,如图 7.2 所示,

如果没有对这些组件加以引用,则无法使用相关组件。

- Windows Media Encoder。
- Microsoft PropShell Control 1.0。
- Wmrmobjs 1.0 Type Library。

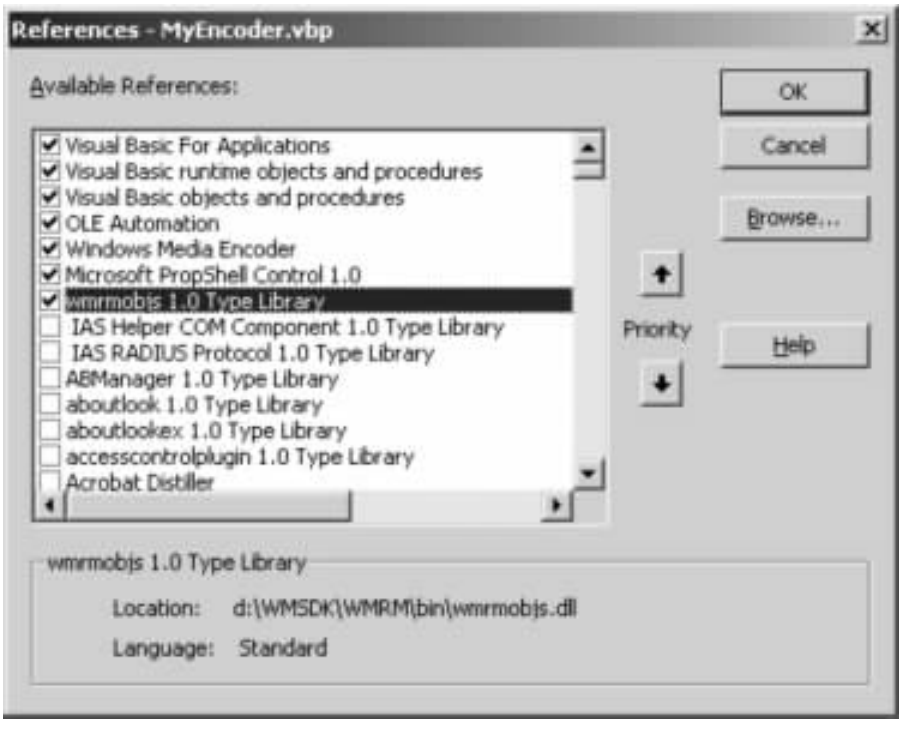


图 7.2　加入对编码打包工具的对象引用

打包演示工具主要由 2 个窗体构成和一个模块构成,如图 7.3 所示。

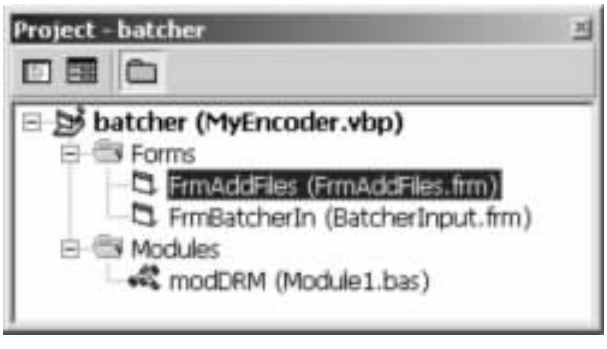


图 7.3　打包演示程序组成

(1) FrmBatcherIn 窗体

FrmBatcherIn 窗体完成主要的操作工作,如图 7.4 所示。

可以看到,窗体中含有 Microsoft PropShell Control 1.0 控件,用以监视 Encoder 的编码过程,详细介绍可以参考第 3 章。

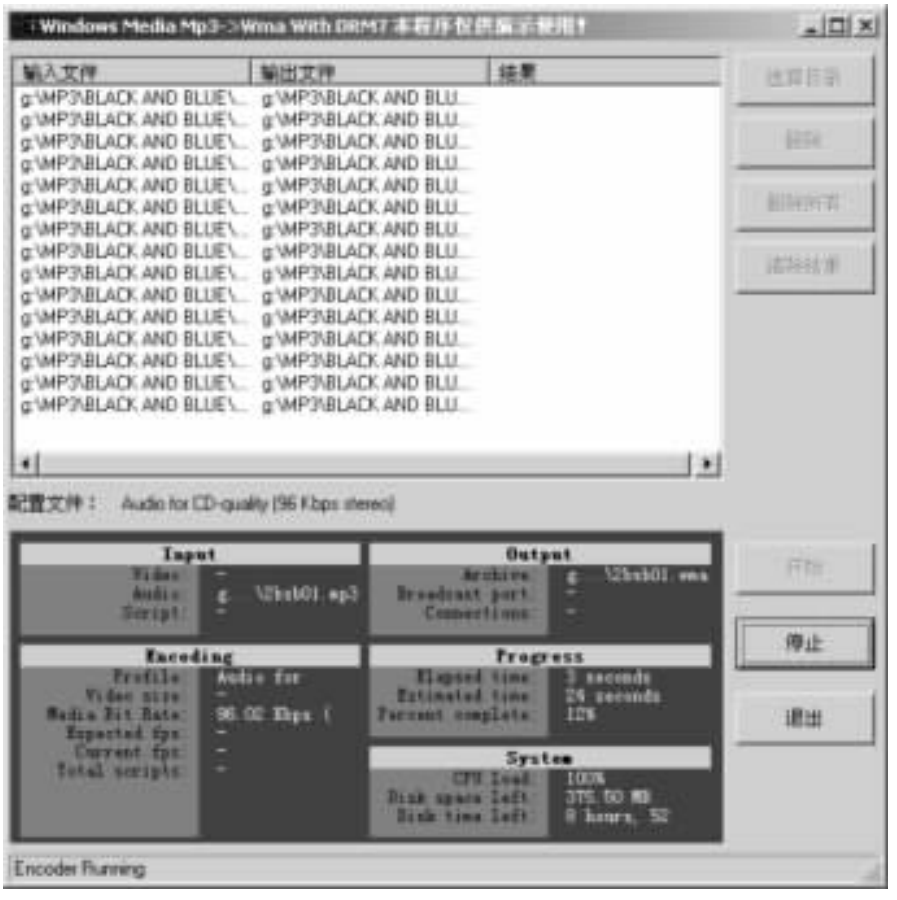


图7.4 打包窗体

本窗体使用代码如下：

```
Option Explicit
Dim WithEvents Encoder As WMEncoder        // 定义 Encoder 对象
Dim EncError As Long

Private Sub CmdAdd_Click()
// 显示添加文件窗体
    Me. LvwBatcher. ListItems. Clear
    FrmAddFiles. Show        // 添加文件窗体显示
End Sub

Private Sub CmdClearResults_Click()
// 清除结果信息
    Dim cnt As Integer
    On Error GoTo Err_Handler
```

```
    cnt = LvwBatcher.ListItems.Count
    If cnt = 0 Then
        MsgBox "Nothing to clear"
    End If

    While cnt > 0
        If LvwBatcher.ListItems(cnt).ListSubItems("Result") = "" Then
        Else
            LvwBatcher.ListItems(cnt).ListSubItems("Result") = ""
        End If
        cnt = cnt - 1
    Wend
    CmdStart.Enabled = True
    Exit Sub
Err_Handler:
    MsgBox "Error clearing the list", vbExclamation
End Sub

Private Sub CmdExit_Click()
// 退出程序,清除对象
    Set Encoder = Nothing
    Unload Me
    End
End Sub

Private Sub CmdRemove_Click()
// 删除所有转换列表中的指定信息
    On Error GoTo Err_Handler
    If LvwBatcher.ListItems.Count = 0 Then
        Exit Sub
    End If
    LvwBatcher.ListItems.Remove LvwBatcher.SelectedItem.Index
    Exit Sub
Err_Handler:
    MsgBox "Error Removing Selected Entry From List", vbExclamation

End Sub

Private Sub CmdRemoveAll_Click()
// 删除所有转换列表中的所有信息
    Dim cnt As Integer
```

```
    On Error GoTo Err_Handler

    cnt = LvwBatcher.ListItems.Count
    While cnt > 0
        LvwBatcher.ListItems.Remove cnt
        cnt = cnt - 1
    Wend
    Exit Sub
Err_Handler:
    MsgBox "Error Removing Entries from List", vbExclamation

End Sub

Private Sub CmdStart_Click()
// 编码程序开始
    Dim Success As Long
    Dim CntStart As Long
    Dim Status As String

    Dim InputFile As String
    Dim OutputFile As String
    Dim Profile As String
// 定义监视页(参见第3章)
    Dim PpgMain As New WMEncMonMainPage

    Dim SrcGrpCol As IWMEncSourceGroupCollection
    Dim SrcGrp As IWMEncSourceGroup
    Dim ErrVal
    On Error Resume Next

    CmdStart.Enabled = False
    CmdAdd.Enabled = False
    CmdRemove.Enabled = False
    CmdClearResults.Enabled = False
    CmdRemoveAll.Enabled = False
    CmdStop.Enabled = True

    If LvwBatcher.ListItems.Count = 0 Then
        MsgBox "No files to encode"
        Exit Sub
    End If
```

```
// 建立新的 Encoder 对象
    Set Encoder = New WMEncoder

// 将 Encoder 对象与容器控件绑定
    MSPropMain.AddObject Encoder
// 添加监视页
    MSPropMain.AddPage PpgMain

    Set SrcGrpCol = Encoder.SourceGroupCollection
    Set SrcGrp = SrcGrpCol.Add("Batcher")
    CntStart = 1
// 以下循环读取信息,并编码打包
    While CntStart <= LvwBatcher.ListItems.Count
      InputFile = LvwBatcher.ListItems(CntStart).Text
      OutputFile = LvwBatcher.ListItems(CntStart).ListSubItems("OutputFile").
Text
      Profile = Label1.Caption
      Status = LvwBatcher.ListItems(CntStart).ListSubItems("Result").Text

      If Status <> "Done" Then
        // 调用 ProcessInput 函数完成编码过程,返回状态
        Success = ProcessInput(Encoder, InputFile, OutputFile, Profile)

            // 版权信息打包

             // 调用 DrmFile 打包文件,加入数字版权信息
             DrmFile OutputFile, Replace(OutputFile, ".wma", "drm.wma")
             Kill OutputFile // 删除临时文件
             Name Replace(OutputFile, ".wma", "drm.wma") As OutputFile

            // 版权打包结束

        If Success = 0 Then
            LvwBatcher.ListItems(CntStart).ListSubItems("Result").Text =
"Done"
        Else
                ErrVal = Hex(Success)
        LvwBatcher.ListItems(CntStart).ListSubItems("Result").Text = "Er-
ror :" & ErrVal
        End If
        End If
```

```
        CntStart = CntStart + 1

        If CntStart <= LvwBatcher.ListItems.Count Then
            Set LvwBatcher.SelectedItem = LvwBatcher.ListItems(CntStart)
            Set LvwBatcher.DropHighlight = LvwBatcher.ListItems(CntStart)
        End If

    Wend
    If CntStart > LvwBatcher.ListItems.Count Then
        Set Encoder = Nothing
        CmdStop.Enabled = False
        CmdStart.Enabled = False
    End If

    CmdAdd.Enabled = True
    CmdRemove.Enabled = True
    CmdClearResults.Enabled = True
    CmdRemoveAll.Enabled = True
End Sub


Private Sub CmdStop_Click()
// 中断编码过程
    On Error GoTo Err_Handler
    // 检测当前 Encoder 状态
    If Encoder.RunState = WMENC_ENCODER_RUNNING Or WMENC_ENCODER_STARTING Or WMENC_ENCODER_PAUSED Then
        Encoder.Stop
    End If
    Set Encoder = Nothing

    CmdAdd.Enabled = True
    CmdRemove.Enabled = True
    CmdRemoveAll.Enabled = True
    CmdClearResults.Enabled = True
    CmdStart.Enabled = True
    Exit Sub
Err_Handler:
    MsgBox "Error stopping Encoder", vbCritical, "Error - " & Err.Number

End Sub
```

```
// Encoder 错误事件捕捉
Private Sub Encoder_OnError(ByVal hResult As Long)
    EncError = Encoder.ErrorState
End Sub
Private Sub Form_Load()
    Dim I As Integer


    On Error GoTo Err_Handler

    For I = 1 To 3
        LvwBatcher.ColumnHeaders.Item(I).Width = LvwBatcher.Width / 3
    Next I
    Set Encoder = Nothing
    StaEncoderStat.SimpleText = "Encoder Stopped"

    CmdRemove.Enabled = False
    CmdRemoveAll.Enabled = False
    CmdClearResults.Enabled = False
    CmdStart.Enabled = False
    CmdStop.Enabled = False

    Exit Sub
Err_Handler:
    MsgBox "Error Executing Application", vbCritical, "Error - " & Err.Number
End Sub

// 编码过程函数,用以完成编码
Function ProcessInput(Encoder As WMEncoder, InputFile As String, OutputFile As
String, Profile As String) As Long

    Dim SrcGrpCol As IWMEncSourceGroupCollection
    Dim SrcGrp As IWMEncSourceGroup
    Dim PrfCol As IWMEncProfileCollection
    Dim Prf As IWMEncProfile

    Dim Success As Integer

    On Error GoTo Err_Handler
    Set SrcGrpCol = Encoder.SourceGroupCollection
    Set SrcGrp = SrcGrpCol.Item(0)
```

```
    SrcGrp. AutoSetFileSource (InputFile)
    Set PrfCol = Encoder. ProfileCollection
    Success = GetProfile(Profile, Prf, PrfCol, SrcGrp)
    If Success = 1 Then
        ProcessInput = 1
        Exit Function
    End If
    Encoder. File. LocalFileName = OutputFile
    EncError = 0
    Encoder. Start          // 开始编码

    // 检查当前状态
    StaEncoderStat. SimpleText = "Encoder Running"
    While Encoder. RunState < > WMENC_ENCODER_STOPPED And EncError
= 0
        DoEvents
    Wend

    StaEncoderStat. SimpleText = "Encoder Stopped"
    If EncError < > 0 Then
        ProcessInput = EncError
    End If
    Exit Function
Err_Handler:
    ProcessInput = Err. Number
End Function

// GetProfile 函数,接受配置文件名称,向 Encoder 对象配置文件对象
Function GetProfile (Profile As String, Prf As IWMEncProfile, PrfCol As IW-
MEncProfileCollection, SrcGrp As IWMEncSourceGroup) As Integer
// 获取配置文件
    Dim ExistsFlg As Boolean
    On Error GoTo Err_Handler
    ExistsFlg = False

    For Each Prf In PrfCol
        If UCase(Prf. Name) = UCase(Profile) Then
    // 如果符合条件,那么指定配置文件用以编码
            SrcGrp. Profile = Prf
            ExistsFlg = True
            Exit For
```

```
            End If
        Next

        If ExistsFlg = False Then
            GetProfile = 1
        End If
        Exit Function
Err_Handler:
        GetProfile = 1
End Function

Private Sub LvwBatcher_Click()
// 播放列表事件
        Set LvwBatcher.DropHighlight = LvwBatcher.SelectedItem
        LvwBatcher.FullRowSelect = True
End Sub
```

(2) FrmAddFiles 窗体

FrmAddFiles **窗体主要完成向转换列表中添加需要转换的文件,并选择配置文件,如图** 7.5 **所示。**

图 7.5　FrmAddFiles **窗体**

相关代码如下:

```
Private Sub CmdOk_Click()
    Dim FileFinder
```

```
    Dim Count
    FileFinder = Dir(Dir1.Path & "h*.mp3")
    Do Until FileFinder = ""
            Count = FrmBatcherIn.LvwBatcher.ListItems.Count + 1
             FrmBatcherIn.LvwBatcher.ListItems.Add Count,,Dir1.Path & "h"
& FileFinder
             FrmBatcherIn.LvwBatcher.ListItems(Count).ListSubItems.Add ,
"OutputFile",Dir1.Path & "h" & Replace(LCase(FileFinder),".mp3",".wma")
             FrmBatcherIn.LvwBatcher.ListItems(Count).ListSubItems.Add ,
"Result",""
             FrmBatcherIn.LvwBatcher.FullRowSelect = True
          FileFinder = Dir
    Loop
    FrmBatcherIn.Label1.Caption = CmbProfile.Text
    FrmBatcherIn.Show
    With FrmBatcherIn
        .CmdRemove.Enabled = True
        .CmdClearResults.Enabled = True
        .CmdRemoveAll.Enabled = True
        .CmdStart.Enabled = True
        .CmdStop.Enabled = True
    End With


    Unload Me
End Sub

Private Sub Drive1_Change()
    Me.Dir1.Path = Drive1.Drive
End Sub

Private Sub Form_Load()
    Dim PrfEncoder As WMEncoder
    Dim PrfCol As IWMEncProfileCollection
    Dim Prf As IWMEncProfile

    On Error GoTo Err_Handler
    Set PrfEncoder = New WMEncoder
```

```
    Set PrfCol = PrfEncoder.ProfileCollection



    For Each Prf In PrfCol
        If Prf.MediaCount(WMENC_AUDIO) > 0 And Prf.MediaCount
(WMENC_VIDEO) = 0 Then    // 只显示音频配置文件
            CmbProfile.AddItem Prf.Name
        End If
    Next
    CmbProfile.Text = CmbProfile.List(0)
    Set PrfEncoder = Nothing
    Exit Sub
Err_Handler:
    MsgBox "Error Getting Profiles of Encoder", vbCritical, "Error - " & Err.
Number

End Sub
```

(3) ModDRM 模块

ModDRM 模块主要是版权处理的相关信息,其代码如下：

```
Public Const siteurl = "http://192.167.0.1"    // 验证地址
Public Const KeyID = "CoWjaTHeOUyf68pGpnu0uQ==" // for pre-delivery

// 内容信息常量,显示使用
Public Const seed = "v1gDfqnfy95nbjcS5Z8G2HhiY2soJn9MKtDQHOhj"
Public Const contentserverpubkey = "mUjytcbBKQjWeRRRbkV1BiHCzCKApv
pGjnFt7M2TkhWr*0G2mwrCWg=="
Public Const contentserverprivkey = "xLjFjq!B3rH2osf6pzjen9BoSUw="

Public Const ownerid = "Liumeiti.com"
Public Const dbguid = "xxx"



Public Sub DrmFile(InputFile As String, OutputFile As String)

    Dim keyObj As WMRMKeys
```

```
Dim headobj As WMRMHeader
Dim protectobj As WMRMProtect

Dim oldHeaderObj
Dim cid
Dim rid, kid, key
Dim v1regpage, v2regpage, licurl, v1licurl, owner
Dim header
Dim result
Dim value
Dim secversion
Dim reusekid
Dim headerfile
Dim OutputFiledir, headerdir
Dim InputFiledefault
Dim sign, signcertcn, signstorename
Dim description

description = ""


Do


    v1regpage = ""
    v2regpage = "simple.asp"
    secversion = 2.2
    reusekid = KeyID
    cid = 500 // 版本
    rid = 5 // 无限制访问

    cid = Trim(cid)
    rid = Trim(rid)
    If (cid = "") Then
      cid = "0"
    End If
    If (rid = "") Then
      rid = "0"
```

```
End If

// 设置完成的验证地址
licurl = siteurl + "/" + v2regpage


Err. Clear

// 建立 Key，Header，and Protect 对象

Set keyObj = CreateObject("Wmrmobjs. WMRMKeys")
If (Err. Number < > 0) Then
    Exit Do
End If
Set headobj = CreateObject("Wmrmobjs. WMRMHeader")
If (Err. Number < > 0) Then
    Exit Do
End If
Set protectobj = CreateObject("Wmrmobjs. WMRMProtect")
If (Err. Number < > 0) Then
    Exit Do
End If




////////////////////////////////////////////////
// 生成 key
// 指定必要信息
// 必须使用 seed 和 key id 来生成 Key
////////////////////////////////////////////////

kid = keyObj. GenerateKeyID()

keyObj. seed = seed
keyObj. KeyID = kid

// 获得动态 Key
```

```
key = keyObj.GenerateKey()
If (Err.Number <> 0) Then
    Exit Do
End If

////////////////////////////////////////
// 生成头部信息
////////////////////////////////////////

// 设置对象参数
headobj.KeyID = kid
    If (Err.Number <> 0) Then
        Exit Do
    End If

headobj.LicenseAcqURL = licurl     // DRM7 认证设置
    If (Err.Number <> 0) Then
        Exit Do
    End If
headobj.ContentID = cid
    If (Err.Number <> 0) Then
        Exit Do
    End If

// 检测信息
Call headobj.SetCheckSum(key)
    If (Err.Number <> 0) Then
        Exit Do
    End If

 If v2regpage <> "Simple.asp" Then
   headobj.Attribute("RID") = rid
 End If

    If (Err.Number <> 0) Then
        Exit Do
    End If
```

```
If（secversion < > ""）Then
    headobj. IndividualizedVersion = secversion
End If
    If（Err. Number < > 0）Then
      Exit Do
    End If
// 标志 privkey
Call headobj. sign（contentserverprivkey）
   If（Err. Number < > 0）Then
     Exit Do
   End If

// 生成 header 信息
header = headobj. header
    If（Err. Number < > 0）Then
     Exit Do
    End If

////////////////////////////////////////
// 打包内容
////////////////////////////////////////

 If（OutputFile < > ""）Then
    protectobj. InputFile = InputFile
    If（Err. Number < > 0）Then
        Exit Do
    End If
    If v1licurl < > "" Then
      protectobj. V1LicenseAcqURL = v1licurl
    End If

    If（Err. Number < > 0）Then
        Exit Do
    End If
    protectobj. key = key
    If（Err. Number < > 0）Then
        Exit Do
    End If
```

```
                protectobj. header = header
                If (Err. Number < > 0) Then
                    Exit Do
                End If
                protectobj. V1KeyID = kid
                If (Err. Number < > 0) Then
                    Exit Do
                End If

                // 打包文件
                Call protectobj. ProtectFile(OutputFile)
                If (Err. Number < > 0) Then
                    Exit Do
                End If

                description = "打包成功"


            End If
        Loop While False

        If Err. Number < > 0 Then
          description = "0x" + CStr(Hex(Err. Number)) + " :" + Err. description
        End If

        Debug. Print description
        // 释放对象

        Set keyObj = Nothing
        Set headobj = Nothing
        Set protectobj = Nothing
        Set oldHeaderObj = Nothing
    End Sub
```

7.3.2　第二部分 :发布编码打包内容

发布编码打包内容是指将打包编码后的 WMA 文件传送到用户端,提供给用户收听。第 4 章提过发布的方式是多种多样的,例如通过 FTP 下载,HTTP 下载,局域网间的拷贝,使用光盘和软盘发布等。在此讨论通过 Windows Media Services 的发布方式,不管发布的方式如何,用户在播放 WMA 文件时,必须向指定的认证服务器获取播放认证。

发布工具要具备如下功能:

①将指定目录作为一个发布点。
②将这个目录下 WMA 文件保存在数据库中。
③可以查看目前存在的发布点。
在开始编码前,必须加入对如下组件的引用,如图 7.6 所示。

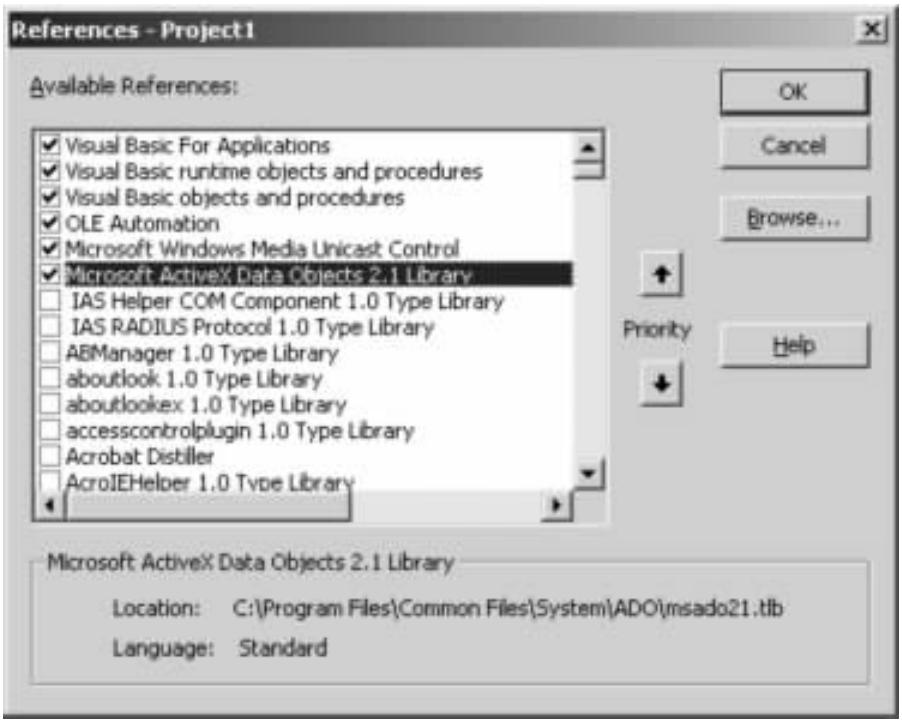


图 7.6　引用必要组件

同时还需要使用 Access 的数据库支持,用以保存现有的音乐文件名称。用数据库保存不是必须的,也可以在程序运行时读取,但使用数据库存储更加灵活,本发布工具主要使用 wmafile 表,如图 7.7 所示。

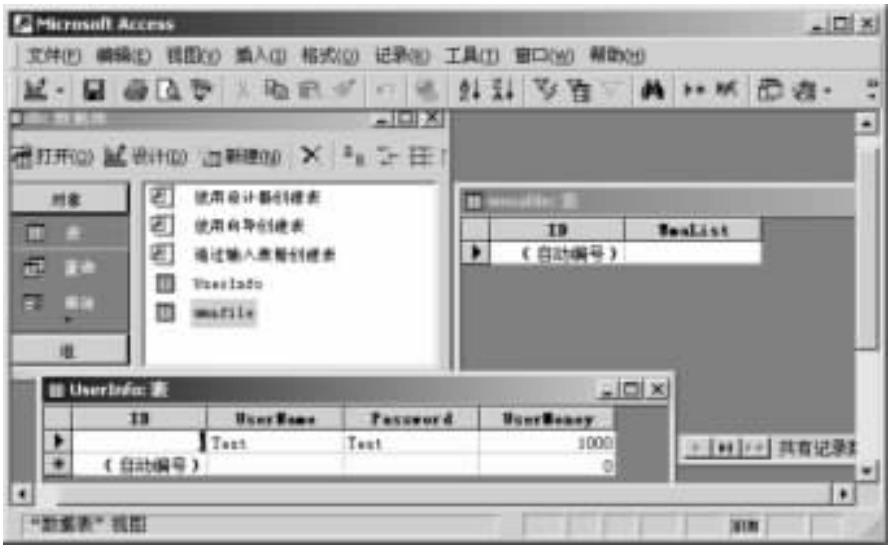


图 7.7　支持数据库

程序完整代码如下：

```
Private Sub cmdPub_Click()
    If txtPubName.Text = "" Then
        MsgBox "请指定发布点名称"
        Exit Sub
    End If

    // 添加发布点,详细操作可以参见第 4 章
    Me.NsoAdminControl1.VirtualRoots.Add txtPubName.Text, Dir1.Path



    // 遍历发布点目录下所有文件,添加至数据库中
    Dim Conn As ADODB.Connection
    Dim FileFinder
    Dim sql As String
    // ******初始化数据库******
    Set Conn = New ADODB.Connection
    With Conn
        .ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source
="  & App.Path & "\hdb.mdb"
        .Open
    End With
```

```
    FileFinder = Dir(Dir1.Path & "h*.wma")
    Do Until FileFinder = ""
        // 获得访问路径,存入数据库
        sql = "insert into wmafile(wmalist) values ('" & "/" & txtPubName.
Text & "/" & FileFinder & "')"
        Conn.Execute sql
        FileFinder = Dir
    Loop

    Conn.Close
    Set Conn = Nothing
    // 添加至数据库完成

    MsgBox "发布成功"
    ListAllVR // 刷新显示
    txtPubName.Text = ""
End Sub

Private Sub Drive1_Change()
    Me.Dir1.Path = Drive1.Drive
End Sub

Private Sub Form_Load()
    Me.NsoAdminControl1.Connect "localhost"
    ListAllVR
End Sub

Sub ListAllVR()
    // 显示所有单播点播发布点
    Dim MyVRs As INsoVirtualRoots
    Dim MyVR As INsoVirtualRoot
    // 使用发布点对象集合
    Set MyVRs = Me.NsoAdminControl1.VirtualRoots
    lstVR.Clear
    For Each MyVR In MyVRs
        Me.lstVR.AddItem MyVR.AliasName
    Next
End Sub
```

运行程序，将 d：当前目录下所有的 WMA 发布为 T2 的发布点，如图 7.8 所示。

图 7.8　添加发布点

显示添加成功后，可以通过 Media 管理器查看到（如图 7.9 所示），已经自动添加了 T2 的发布点，其实际路径为 d：h，用户现在可以通过 mms：//myserver/T2/文件名来访问 d：h下的发布文件。

图 7.9　通过 Media 管理器查看 T2 发布点

在数据库中 wmafile 表已经自动增加了如图 7.10 所示的内容（因符合条件的只有 3 个 WMA 文件）。

图 7.10　数据库中增加的内容

7.3.3　第三部分 验证用户信息

用户访问编码打包后的文件时,需要到认证服务器取得认证。本例中,打包后的文件会访问 http ://192.168.0.1/Simple.asp 来取得认证,这时,就可以在 Simple.asp 中取得用户的状态,并验证相关的信息。

本部分所需的数据库与第二部分使用的是同一个数据库。在 UserInfo 表中存储了一个用户的基本信息 :UserName(用户名),Password(密码),UserMoney(拥有金钱)。当前表中添加了一个用户 Test 的基本信息,如图 7.11 所示。

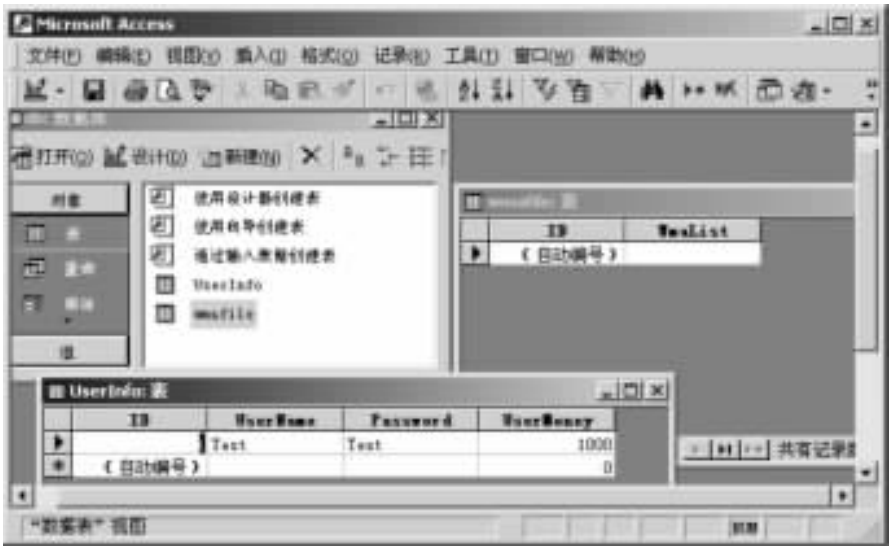


图 7.11　数据库设计

使用 ASP 进行用户判定，用 Session("UserName")，Session("Password")来验证用户名和密码。设计包括一个 Global. asa 文件（用于服务器信息保存，参见第 7 章）和以下几个 ASP 文件：

- login. asp 对用户进行验证，代码如下：

```
<%
if Request. Form("username") < >"" then
Dim Conn
Dim Rs

on error resume next

Set Conn = Server. CreateObject("ADODB. Connection")
connstr = "Provider = Microsoft. Jet. OLEDB. 4. 0 ;Data Source = " & Server. MapPath
("db. mdb")
Conn. open connstr

set Rs = server. CreateObject("adodb. recordset")
rs. open "select * from userinfo where username = ' " & Request. Form ("user-
name") & " '",Conn,1,3
if not Rs. EOF then
    if rs("password") = request("password") then

        if rs("usermoney") > =0 then // 收费验证
        session("username") = rs("username")
        session("password") = rs("password")
        session("challenge") = session("challenge")

        // * * * * * * * * * * * * * * * 扣除收费 * * * * * * * * * *
        Conn. Execute "update userinfo set usermoney = usermoney - 1 where user-
name = ' " & Request. Form ("username") & " '"
        // * * * * * * * * * * * * * * * *
        else
            Response. Write "你的金额不足！"
            call showform
            Rs. Close
            Response. End
        end if
```

```
        end if
else
        call showform
        Rs. Close
        Response. End
end if
rs. close
Conn. close
set Conn = nothing

Response. Redirect " simple. asp"
else
call showform
end if
% >
< %
sub showform()
% >
< form name = " form1"  method = " post"  action = " login. asp" >
    < p > 用户名：
        < input name = " username"  type = " text" >
        < br >
      密 码 ：
        < input type = " password"  name = " password" >
        < br >
        < input type = " submit"  name = " Submit"  value = " 提交" >
    < /p >
< /form >
< %
session(" challenge") = session(" challenge")
end sub
% >
```


- wmalist. asp **显示数据库中的** wma **文件列表,代码如下：**

```
< %
Dim Conn
Dim Rs
```

```
Dim LocalName
localname = "192.168.0.1"
Set Conn = Server.CreateObject("ADODB.Connection")
connstr = "Provider = Microsoft.Jet.OLEDB.4.0 ;Data Source = " & Server.MapPath
("db.mdb")
Conn.open connstr

set Rs = server.createobject("adodb.recordset")
rs.open "select * from wmafile",Conn,1,1
do while not rs.eof
    response.write "<a href = playwma.asp?URL = " & "mms://" & local-
name & rs("Wmalist") & ">"
    response.write "mms://" & localname & rs("Wmalist")
    response.write "</a> " & "<a href = " & "mms://" & localname & rs
("Wmalist") & ">" & "点击这里使用 Mediaplayer 播放 < BR > < /a >"
rs.movenext
loop
rs.close

Conn.close
set Conn = nothing
% >
```


运行状态如图 7.12 所示。

图 7.12 wmalist.asp 演示

- simple.asp 响应客户端的认证请求及证书发布,代码如下：

```
<%
// ////////////////////////////////////////////////////////////
```

```
// 功能概述：
// 取得用户信息,生成许可证,发还用户
//
// 生成许可证的必要条件
//     必须：Key（生成）
//             License key seed（来自打包服务器）
//             ClientInfo（来自 Header）
//             Key ID（来自 Header）
//     可选：Priority，attributes，rights
//
// 如何生成 Key
//     Challenge object（get client info，header）
//       - - - > Header object（取得 key ID）
//       - - - > Key object（使用 seed 生成 Key ，key ID）
//       - - - > Licence object（使用 key，key ID,生成 Licence）
//       - - - > Response object（输出）
//
//
// ////////////////////////////////////////////////////
% >

<%
' 判断用户是否登录
if session("username") = "" then
    session("challenge") = request.form("challenge")
    Response.Redirect "login.asp"
end if

Response.Buffer = True
Response.Expires = 0
do
Dim licprivateKey // 认证不需要使用私钥
Dim licpublicKey
Dim conpublicKey // 打包服务的公钥
Dim seed ' 打包服务的 Seed 值
conpublicKey = Application("contentserverpubkey")
seed         = Application("seed")
```

```
Dim objChallenge
Dim objHeader
Dim objKey
Dim objRights
Dim objLicense
Dim objResponse


Dim description
Dim silent
Dim delivery_tag



 description = ""

// ////////////////////////////////////////
// 查看客户端是否有认证记录
// ////////////////////////////////////////

 silent = False
 IF (request.Form("nonsilent") <> "") THEN
        silent = FALSE
 END IF

// 使用 Challenge 对象获取 Header 信息

// ////////////////////////////////////////
// Challenge 对象
// ////////////////////////////////////////
// 产生 Challenge 对象
Set objChallenge = Server.CreateObject("wmrmobjs.wmrmchallenge")

// 取得客户信息
Dim strLicenseRequested
strLicenseRequested = session("challenge")

// 取得请求信息
objChallenge.Challenge = strLicenseRequested
```

```
    if (err.number <> 0) then
        exit do
    end if
// 由 Challenge 对象取得访问用户的基本信息
Dim varClientInfo
Dim varHeader

// 由 Challenge 对象设置用户的基本信息
varClientInfo = objChallenge.ClientInfo

// 由 Challenge 对象设置 Header
varHeader = objChallenge.Header



// ////////////////////////
// Header 对象
// ////////////////////////

// 设置 Header 对象
Set objHeader = Server.CreateObject("wmrmobjs.wmrmheader")
Dim Kid

// 同步 Header
objHeader.Header = varHeader
    if (err.number <> 0) then
        exit do
    end if

// 使用服务器公钥校验 Header
Dim blnResult
blnResult = objHeader.Verify(conpublicKey) // 公钥信息
        if (err.number <> 0) then
            exit do
        end if



// 由 Header 对象获取 key id
Kid = objHeader.KeyID
```

```
// ////////////////////////////////////////
// Key 对象
// ////////////////////////////////////////
// 产生 Key 对象
Set objKey = server.CreateObject("wmrmobjs.wmrmkeys")
Dim varKEY

// 将基本信息读入 Key 对象
objKey.KeyID = Kid ' from the Header object
objKey.Seed = Seed
    if (err.number <> 0) then
        exit do
    end if
// 生成 Key 对象
varKEY = objKey.GenerateKey
    if (err.number <> 0) then
        exit do
    end if

// ////////////////////////////////////////////////
// 生成权限
// 此处非常重要,可以设置用户各种访问权限
// 参见第 6 章
// ////////////////////////////////////////////////
Set objRights = server.CreateObject("wmrmobjs.wmrmRights")
Dim SetV2Rights

        With objRights
          .MinimumAppSecurity = 500
          .AllowPlayOnPC = true
          .AllowBackupRestore = true
          .AllowTransferToSDMI = true
          .Playcount = 1
          .PMRights = 51
          .PMAppSecurity = 150
        End with

         if err.number = 0 then
            SetV2Rights = objRights.GetAllRights
```

```
            end if

//  ////////////////////////////////
//  有认证对象生成许可证
//  ////////////////////////////////
Set objLicense  =  Server. CreateObject("wmrmobjs. wmrmlicgen")
Dim varLicense
Dim varCategory
Dim varVersion

    With objLicense


            . KeyID  =  Kid  //  由 Key 对象获取
            if (err. number  < > 0) then
                exit do
        end if

            . SetKey "", varKEY  //  由 Key 对象获取
            if (err. number  < > 0) then
                exit do
            end if

            . Priority  =  10  //  设置优先权
            if (err. number  < > 0) then
                exit do
        end if

            . Attribute ("LIECNSESERVER")  =  " < NAME > License  Sever < /
NAME > "
            if (err. number  < > 0) then
                exit do
            end if

            . Rights  = setV2Rights  //  由权限对象获取
```

```
        if (err.number <> 0) then
            exit do
        end if

        .ClientInfo = varClientInfo // 由请求对象获取
        if (err.number <> 0) then
            exit do
        end if

        .GetClientVersion varCategory, varVersion
        if (err.number <> 0) then
            exit do
        end if

        Select Case varCategory
            Case 0
                // 客户非 Windows 系统

                // 如果 varVersion 值为 513 那么为 DRM7 客户

            case 1
                // 客户为 Windows 系统

                // 如果 varVersion 值为 513 那么为 DRM7 客户

            Case Else
                // 无效用户
                exit do
        End Select

        // 生成许可证
        varLicense = .GetLicenseToDeliver()
        if (err.number <> 0) then
            exit do
        end if

End with
    delivery_tag = "deliver"
```

```
    loop while false



    IF (delivery_tag = "" and err.number <> 0) then
            Response.Write err.number & " :" & err.description
    Elseif (delivery_tag = "deliver") then

    // ////////////////////////////////////
    // 响应对象
    // ////////////////////////////////////
set responseObj = Server.CreateObject("wmrmobjs.wmrmresponse")
Dim licresponse
      // Add the license
       Call     responseObj.AddLicense("2.0.0.0", varLicense)

       IF     (silent = True) then
                licresponse = responseObj.GetLicenseResponse()
                Response.Write licresponse
       Else
                responseObj.ReplaceQuotesWith = "h""" // For JavaScript
                licresponse =responseObj.GetLicenseResponse()


% >
<HTML >
<HEAD >
<TITLE >认证发布 </TITLE >
<Script Language = "JavaScript" >
function StoreV2License(hr)
 {
     netobj.StoreLicense( " <% = licresponse % >" ) ;
}
</Script >
</HEAD >
<BODY onLoad = "StoreV2License()" >
<OBJECT classid = clsid :A9FC132B - 096D - 460B - B7D5 - 1DB0FAE0C062
height = 0 id = netobj
width = 0 >
      <EMBED MAYSCRIPT TYPE = "application/x - drm - v2" HIDDEN = "true" >
```

```
</OBJECT>
许可证获得成功,点击播放按钮
</BODY>
</HTML>
<%
        End if
End if

// 清除对象

Set objChallenge = Nothing
Set objHeader = Nothing
Set objKey = Nothing
Set objRights = Nothing
Set objLicense = Nothing
Set objDecoding = Nothing
Set responseObj = Nothing

%>
```

用户发出请求,Windows Media Player 开启窗体至 simple.asp,状态如图 7.13 所示。

图 7.13 验证用户信息

验证成功如图 7. 14 **所示。**

图 7. 14　**验证成功**

- playwma. asp **播放** wma **文件的代码如下：**

```
< HTML >
< HEAD >
< /HEAD >
< BODY >
< OBJECT ID = " Player"  height = "0"  width = "0"
  CLASSID = " CLSID :6BF52A52 - 394A - 11d3 - B153 - 00C04F79FAA6" >
< /OBJECT >
< INPUT TYPE = " BUTTON"  NAME = " BtnPlay"  VALUE = " Play"  OnClick = " St-
artMeUp()" >
< INPUT TYPE = " BUTTON"  NAME = " BtnStop"  VALUE = " Stop"  OnClick =
" ShutMeDown()" >

< P > 欢迎使用
< SCRIPT >
< ! - -
function StartMeUp ()
{
    Player. URL  =  " < % = request(" URL")% > " ;
```

```
}
function ShutMeDown()
 {
     Player. controls. stop();
}
 -- >
 < /SCRIPT >
 < /BODY >
 < /HTML >
```

效果如图 7. 15 **所示。**

图 7. 15 **播放页面**

除了可以使用 Media Player **播放,也可使用页面播放,详细操作可参见第** 2 **章。**

Windows Media 技术词汇表

附录1

本词汇表所包含的条目将有助于对 Microsoft Windows Media 技术的理解。

A ~ B

Access Control List（ACL）checking /访问控制列表（ACL）检查

Microsoft Windows 安全性的一部分，Windows Media 服务使用它来检验客户端是否具有特定文件或目录的访问权限。使用 ACL 检查，系统管理员可以设置 . asf 文件或目录的权限限制。

ActiveX

Microsoft 的一种使不同程序共享信息的技术。ActiveX 扩充了基于 Microsoft Windows 的架构，以包含 Internet 和 Intranet 的特性和功能。开发者使用它在程序和 Web 页中构建用户交互性。

ActiveX controls / ActiveX 控件

使用 ActiveX 技术的控件。这些控件可以从 Web 页自动下载并由 Web 浏览器执行。

Advanced Streaming Format（ASF）/高级流格式（ASF）

用于流式音频内容、视频内容、图像以及脚本命令的一种数据格式，以数据包的形式通过网络传输。ASF 内容可以是一个 . asf 文件或 Windows Media 编码器生成的实况流。处于通过网络传送过程中的 ASF 内容称为 ASF 流。

Advanced Streaming Format（. asf）file /高级流格式（. asf）文件

ASF 格式的音频或视频文件。

Alias /别名

替代 URL 的名称。例如，当创建一个广播站时，可以使用别名来指定用来定义 Windows Media 服务器组件和 Windows Media 编码器之间连接的信息。例如，Stream1 是 URL msbd ：//server ：port 的别名。当创建一个广播站时，就可以在“别名”对话框中输入“Stream1”。服务器组件通过将其与编码器中的定义对照来进行解析。使用别名的好处是不需要记住不断变化的 URL。只要知道别名，URL 就可以正确地解

析。

Announcement /通知

简单的 . asx 文件,包含用于流的 URL 的相关信息。通知文件在创建了单播发布点或多播广播站之后由 Windows Media 管理器创建。客户端快速加载这个通知文件,然后在单播发布点打开 ASF 流或将 URL 从一个 . nsc 文件提取到该 ASF 流中,并在多播广播站上运行节目。

ASFCheck

一个命令行实用程序,用来检测并修复一些通常在 ASF 1. 0 版本文件中发现的问题。

ASFChop

一个命令行实用程序,用于调整 Windows Media 编码器存储的 ASF 流的开头和结尾部分。

ASFEditor

参阅“ Windows Media Author ”。

ASF root directory / ASF 根目录

参阅“ 主发布点 ”。

ASF Stream Descriptor (. asd) file / ASF 流描述信息 (. asd) 文件

一个由 Windows Media 编码器创建和读取的配置文件。该文件包含描述多媒体流特性的编码器设置,同时也由 Windows Media 广播站服务读取以定义可由指定广播站支持的流格式。

ASF Stream Redirector (. asx) file / ASF 流转向器 (. asx) 文件

一种 ASX 元文件,提供 Microsoft Windows Media Player 用于接收单播流、多播流和其他 Intranet 或 Internet 所支持的媒体的信息。这些文件由 Windows Media Player 快速加载,并包含用于下列用途的信息:

- 将控件从 HTTP 浏览器传送到 Windows Media Player 控件,以便直接将流传递

到 Windows Media Player。

- 提供一个通知文件，Windows Media Player 可使用该文件访问 Windows Media 广播站上的节目。
- 提供流的参考说明和 Windows Media Player 用来处理协议翻转的规则。
- 提供一个播放列表，用来定义由 Windows Media Player 汇集起来的流的播放顺序。

Attribute/属性

在. asx 文件中用来描述 ASX 元素的属性的限定符。例如，. asx 文件可以包含元素“ Repeat ”，而该元素包含属性 Count，该特殊的元素和属性定义客户端重复回放内容剪辑或播放列表的次数。

Audio Compression Manager（ACM）/音频压缩管理器（ACM）

设备驱动程序管理器，用来控制播放或录音需要哪些应用程序。ACM 管理下述类型的驱动程序：

- 压缩和解压缩（codec）驱动程序。
- 格式转换器驱动程序。
- 筛选器驱动程序。

如果元文件不是在运行 Windows 的计算机上创建的，有可能无法运用 ACM 编解码器，这样也就无法使用 Windows Media 技术了。

Authentication /验证

检验客户端登录信息的过程。可以在 Windows Media 服务器提供 ASF 内容或流的访问前，设置验证客户端。

Authorization /授权

授予或拒绝授予客户端访问权限的过程。Windows Media 服务器可以设置成授权客户端请求 ASF 内容。

Bandwidth /带宽

在固定时间内可以传输的数据量。对于计算机网络，带宽越高，说明数据传输速度越快。网络带宽以位/秒来表示。

在 Windows Media 服务环境中，Windows Media 管理器可以为各种不同的功能指定带宽约束条件，这包括服务器中单播的最大合计带宽、服务器中的单一单播流的最

大带宽以及由服务器中传输的多播文件所使用的连续带宽。

Bit Rate /比特率

二进制内容通过网络流动的速度。通常以 kbit/s 来计算，例如，28.8 kbit/s。Windows Media 编码器和 Windows Media 管理器具有 ASF 内容的比特率设置。

Broadcast /广播

描述客户端接收流的经历。广播流可以被多播或单播。在广播连接中，客户端是被动的，并不能控制流启动或结束的时间。相反，在点播连接中，客户端是主动的，并可以控制流的启动或结束时间。

Broadcast Multicast /广播多播

通过 Windows Media 服务器将某个流式文件发布到多个客户端，这些客户端通过监视多播流的 IP 地址来“监听”流。从客户端观点来看，广播多播是一个无连接的体验，因为客户端从来没有连接到 Windows Media 服务器。

Broadcast Unicast /广播单播

一个点对点连接。客户端启动 Windows Media 服务器上的发布点。

Buffer /缓冲区

保留的一段内存区域，用作中间储存区，在这里面数据暂时保持以等待在两个位置之间传输。缓冲区保证在计算机之间存在一个不间断的流。

C ~ H

Caption /字幕

使用 ASF 流发送同步访问媒体交换（SAMI）文件（已关闭字幕格式的文件）的特性。字幕是一项易于访问的特性，可以在播放视频和音频的同时显示字幕，就像某些电视节目的已关闭字幕。也可以使用其他语言显示对白。

Channel /频道

参阅“广播站”。

Client /客户端

一般指的是在客户端/服务器通讯中进行请求的软件。客户端软件请求连接并与服务器进行通讯。

Codec /编解码器

压缩器/解压缩器的简称。在记录数字视频或音频时使用的算法或方案。例如，在通过 Internet 传输视频时使用编解码器，视频在发送方被压缩并在接收方进行解压缩。Windows Media 工具给 ASF 内容的编解码器提供了一个选择，用户可以根据音频或图像的质量以及可选的图像尺寸来选择编解码器。

Content /内容

通过多播或单播从服务器流向客户端的数据。内容可能是实况音频或视频、存储的音频或视频文件以及静止图像或幻灯片。内容需要从原始状态转换成 ASF 格式，以便于从服务器流出。Windows Media 服务器可以汇集实况 ASF 流或存储的 .asf 文件作为内容。

Destination Address /目标地址

IP 地址和端口，从该地址正在监听的客户端可以收到一条多播。客户端通知本机的网卡监听到达目标地址和端口的数据包。

Distributed Component Object Model（DCOM）/分布式组件对象模型（DCOM）

组件对象模型（COM）的扩展。DCOM 可使软件组件通过网络以可靠、安全和高效的方式直接进行相互通讯，这里所指的网络包括 Internet 和 Intranet。

Distribution /分发

将 ASF 流从一个服务器发送到另一个服务器。分发具有很多用途，例如：

- 将流分发到其他服务器，然后单播该流，允许网络中那些未启用多播的客户接收该流。

• 将流分发到启用 HTTP 流的服务器。允许防火墙后面的用户接收以其他方式无法接收的流。

• 将流从一个 Windows Media 服务器分发到另一个 Windows Media 服务器，目的是创建多个单播流。例如，如果已经达到服务器单播流的最大数目，可以将流发送到其他的服务器，在那里再将该流单播给更多的客户端。

Distribution Mode /分发模式

Windows Media 服务器组件的设置，用来表示 Windows Media 服务器组件是否准备进行多播 ASF 流、分发 ASF 流（通过单播）或两者都进行。如果分发模式设置为只进行多播，则服务器通过多播或单播广播 ASF 流。如果分发模式设置为只进行分发，则服务器将 ASF 流在其他即将广播该 ASF 流的服务器请求时进行分发。如果分发模式设置为两者都进行，则多播和分发模式都起作用。

Element /元素

. asx 文件中的条目，用来定义客户端的特殊设置或动作。元素可通过属性来修改。例如，ref 元素具有用来定义指向特定内容 URL 的属性。

Encoder /编码器

参阅“ Windows Media 编码器 ”。

Error Correction /错误修正

用于控制单向通信系统中数据传输错误的方法。随数据额外发送的信息，接收者用来检查并修正该数据。

Error Correction Code（ECC）/错误修正代码（ECC）

早期版本 Windows Media 服务中使用的错误纠正方法。在发送流的同时发送冗余的数据，以检测并控制流中个别位的错误。在当前版本的 Windows Media 服务中，流错误是通过 UDP 重新发送来修正的。

File Transfer Service（FTS）/文件传输服务（FTS）

Windows Media 服务中的一个特性，用于将文件通过网络多播到客户计算机上的某个 ActiveX 控件中（Nsfile. ocx）。

Firewall /**防火墙**

系统或系统的结合,用来加强两个或多个网络之间的界限,并将未授权的用户排除在专用网络之外。防火墙系统检查所有进出的消息,确保它们满足预定的安全标准。

Frame /**帧**

一系列连续图像中的一幅静态图像,用来组成视频片头。

Frame Rate /**帧频**

单独帧变换的速度。帧频越高,图像质量越好。

Home Directory /**主目录**

参阅‘主发布点”。

Home Publishing Point /**主发布点**

发布 ASF **内容的主目录。** Microsoft Windows Media Player **可以汇集任何位于此目录或其子目录中的** .asf **文件。主发布点也是一个点播发布点。与其他发布点不同,主发布点不具有别名。相反,计算机名用在** URL **中,用于访问该主发布点。又称** ASF **根目录。**

I ~ O

Illustrated Audio /**举例用的音频**

将音频内容与图像同步结合起来的流,组成可在低带宽下运行的联机幻灯片。

Image Color Matching (ICM) /**图形色彩匹配** (ICM)

应用程序界面,用于传递各个设备的色彩信息,这样应用程序将正确地显示、打印并将颜色传递给其他用户和应用程序。其他操作系统可能使用不同的颜色匹配方案。使用非 ICM **数据创建的视频内容可能无法在基于** Windows **计算机上正确着色。**

Integrated Services Digital Network（ISDN）/综合服务数字网（ISDN）

通过发送数字方式编码的信号，以很高的速度来传递声音、文字、图像和视频通信信号的完全的电话电讯网络。

Internet Protocol Address（IP address）/网际协议地址（IP 地址）

一个 32 位十六进制数字，代表 Internet 上每一台计算机或设备的惟一的 IP 地址。该数字指定网络上的某个物理位置或节点。

Internet Server API（ISAPI）/ Internet 服务器 API（ISAPI）

用来创建动态链接库（DLL）的架构，为 Internet 提供服务器端功能。Windows Media 服务使用 ISAPI 以提供某一种安全选项。

Intranet

属于某个企业的网络。只有该企业的成员才可以访问该网络。连接到 Internet 的 Intranet 通常受到防火墙或其他设备的保护。

Listen /监听

监视指定的多播 IP 地址。Microsoft Windows Media Player 监听多播 IP 地址中从服务器中汇集过来的数据。

Local /本地

附近的位置或限制到某个特定的区域。在通信领域，本地设备指的是可以直接访问而无须使用通信连接来访问的设备。在信息处理过程中，本地操作是指由附近的计算机来执行，而不是由远程计算机来执行。例如，安装有 Windows Media 服务的服务器计算机就是那台服务器的本地计算机。

Log /日志

收集并保存有关 Windows Media 服务事件的数据。Windows Media 管理器可以记录有关单播和客户端的信息。

Marker /标记

在 . asf 文件中的特定位置的一个指示器,按时间计算。Microsoft Windows Media Player 使用标记直接跳到 . asf 文件的某个点上。一个. asf 文件中的标记允许查看者向前跳到标记的头部或跳回到上一个标记,以便再一次查看 . asf 文件的特定部分。

Media Stream Broadcast Distribution Protocol(MSBD protocol)/媒体流广播分发协议(MSBD 协议)

用来访问 Windows Media 编码器的协议,也就是访问流的源,例如 msbd ∶// server _name。该协议还可以在从 Windows Media 广播站服务流向内容储存服务器时使用。另外,还可以用在服务器到服务器的分发。

Metadata /元数据

在 Windows Media 技术系统中关于内容的信息,如标题、作者或版权等。这些信息包含在一个 . asx 文件中。

Metafile /元文件

在 Windows Media 技术系统中包含一个用于媒体内容信息的文本文件。Windows Media 服务使用 3 种元文件∶asd 文件元文件、. asx 文件元文件和 . nsc 文件元文件。

Microsoft Internet Explorer

由 Microsoft 公司生产的基于 Windows 的 Web 浏览器。Windows Media 服务可以用多种方法来使用 Microsoft Internet Explorer 5。例如,Windows Media 管理器使用 Internet Explorer 显示其 Web 页。内容创造者可将 Microsoft Windows Media Player ActiveX 控件嵌入到 HTML 页中,再使用 Internet Explorer 来查看。

Microsoft Media Server Protocol(MMS protocol)/ Microsoft Media 服务器协议(MMS 协议)

用来访问并流式接收 Windows Media 服务器中 . asf 文件的一种协议。

Microsoft Windows Backup / Microsoft Windows 备份

用于帮助用户保护数据在系统发生硬件或存储障碍时避免数据意外丢失的实用工具。

Microsoft Windows Media Player

客户端程序或控件,可以从 Windows Media 服务器接收流式媒体。该控件既可以作为独立的客户端可执行程序来运行,也可以嵌入到 Web 页、C + + 程序或者使用客户端 ActiveX 控件的 Microsoft Visual Basic 程序。Microsoft Windows Media Player 是第一个通用播放器版本。

Multicast /多播

一对多连接,这里多个客户端都可以从服务器接收相同的流。要接收多播,客户端必须能够访问启用多播的网络。相反,单播是一对一连接,一个客户端从服务器接收截然不同的流。

Multicast-enabled Network /启用多播网络

具有能解释 D 类 IP 地址路由器的网络。

Multiple Bit Rate Video /多比特率视频

Windows Media 技术的一个特性,支持在一个 ASF 流中创建和流式传送 6 个已编码的视频。在 Windows Media 编码器中使用多比特率视频创建 ASF 内容,这些内容具有不同的视频流,以不同的带宽提供给具有低或高带宽的目标听众。为低带宽的听众创建多比特率内容时,视频流可从 18 ~ 300 kbit/s。同样地,高带宽的目标听众包含的视频流范围为 81 ~ 10 Mbit/s。两个目标听众都包括一个单独的已编码的音频流。当编码多比特率视频时,同时编码一个附加的保险视频流,这依据最低选择带宽所占的百分比来定。接收该多编码流之后,服务器将根据可用的网络带宽来确定流式文件使用的带宽。一般的 HTTP 服务器不支持多比特率视频。

. ocx

通常作为 ActiveX 控件的同义词来使用,. ocx 是一个控件的文件扩展名。

On-demand /点播

用来描述存储的媒体内容可以在 Windows Media 服务系统中作为流使用。Windows Media 服务既可将发布点存储的内容汇集成流式文件,又可以使用 Windows Media 编码器处理实况的内容。

On-demand Unicast /点播单播

点对点连接,由客户端启动发布点。在点播单播中,服务器流将内容存储到用户。

P ~ T

Packet /数据包

通过网络传递的数据单元。数据包的大小是固定的,在源和目标之间传送。它包含有二进制信息,既表示数据,又表示包含 ID 号、源地址和目的地址的标题。

Padding /填充

附加在内容流中单独数据包的空白空间,用来使数据包大小保持不变。Windows Media 服务支持可变的数据包长度。但是,Windows Media 编码器将数据包限制成固定的长度,以确保与早期版本的 Windows Media 服务相兼容。

Payload

包含一个或多个流式数据对象的数据单元。

Player /播放器

客户端程序或控件。用来接收从 Windows Media 服务器中汇集成流的内容。对于整个联机帮助,指的就是 Microsoft Windows Media Player。

Playlist /播放列表

Microsoft Windows Media Player 按顺序播放的流式文件的列表。Windows Media

服务既支持服务器端播放列表,又支持客户端播放列表。

• 服务器端播放列表作为节目的一部分在发送台上进行播放。可以使用 Windows Media 管理器广播站页上的“流”按钮来创建播放列表。服务器端播放列表可包含指向流的 URL,这些流包括 .asf 文件。

• 客户端列表是一个包含多个输入元素的 .asx 文件。Windows Media Player 按照显示在 .asx 文件中的顺序来播放这些输入元素。

Port /端口

服务器上的一个位置。内容从该位置流向客户端。使用一个数字来表示端口,这个数字是 URL 的一部分。Windows Media 服务器组件在使用时绑定到端口。默认情况下,Windows Media 单播服务绑定到端口 1755,而 Windows Media 广播站服务绑定到端口 7007。如果 HTTP 流已由某个服务所启用,则该服务切换到端口 80,这是任何 HTTP 流的首选端口。可以修改注册表以更改任意 Windows Media 服务器组件的端口号。

Program /节目

Windows Media 服务器组件将流作为一个单一实体来进行管理。节目可以认为是一个包含流的容器。

Property /属性

对象(如流)的特性。例如,Windows Media 编码器在属性页上显示出流的属性,如带宽和所使用的编码器。

Protocol /协议

用来使计算机之间可以交换信息的一套格式或程序。Windows Media 服务所使用的协议包括 HTTP,MMS 和 MSBD。

Protocol Rollover /协议翻转

在 Windows Media 服务器使用特定的协议进行连接失败时,允许从一种协议切换到另一种协议的过程。例如,如果客户端使用 MMS 协议来请求 ASF 内容,服务器将试图使用 UDP 来将 ASF 内容汇集成流。如果该协议失败,则服务器试图使用 TCP 来将内容汇集成流;如果再失败,则服务器试图使用 HTTP,不过这必须建立在已启用该协议的条件下。如果既没有使用 MMSU 协议(MMS 通过 UDP),也没有使用 MMST 协议(MMS 通过 TCP)来请求 ASF 内容,则协议翻转将不可利用。

Proxy Server /**代理服务器**

控制本地网络和 Internet **或其他** Intranet **之间的基于** Web **通信的服务器。**

Publishing Point /**发布点**

用于储存客户端可用的内容或访问实况流的一个虚拟目录。客户端通过发布点的 URL **来到达该发布点。**

Publishing Point Events Monitor /**发布点事件监视器**

用来监视并显示服务器单播事件的工具。特别地,发布点事件监视器显示单播服务器(发布点)事件活动。

QuickStart /**快速启动**

Windows Media **技术中的一组向导,用于配置** Windows Media **编码器、创建发送台或发布点这样的任务。**

Remote /**远程**

没有在附近或者无法直接访问,位于其他位置(房间、建筑物或城市)并且可以通过各种类型的通信链接进行访问的计算机或其他设备。例如,可以在 Windows Media **服务器以外的另一台计算机(远程计算机)上运行** Windows Media **管理器,远程管理员可以管理该服务器。**Windows Media **编码器也可以从远程服务器上运行。**

Router /**路由器**

连接两个或多个网络并传递数据的设备。路由器确定目标计算机所在的位置并找到将数据传递过去的最佳途径。

Scope /**作用域**

在多播中,流所能到达的范围。Windows Media **管理器允许用户定义多播的作用域。多播流的作用域可设置为只到达紧邻的子目录,也可以设置为到达整个** Internet**。作用域也相当于生存时间(**TTL**)。**

Script Commands /**脚本命令**

包含在 ASF 流中并被发送到客户端的专用指令。Microsoft Windows Media Player 将脚本命令传递到设备或应用程序中,由它们对脚本命令进行解释。调用指定的文件或定位到某个指定的 Web 站点时,每个任务都使用脚本命令。

Security /**安全**

根据用户凭证和权限控制对资源访问的过程。在 Windows Media 服务环境中,安全意味着限制和控制对 Windows Media 服务器组件、Windows Media 管理器以及 Windows Media 内容(既有储存的又有汇集成流的文件)的访问。Windows Media 服务具有内置的安全机制,该机制与 Microsoft NTLM 结合使用。它既支持服务器端验证,也支持客户端验证。

Station /**广播站**

定义好的位置,播放器从该位置接收流。从实际效果看,这是一个 IP 地址或端口号。Windows Media 服务器组件使用只带有 ASF 流的广播站,并将广播站信息存储为扩展名为 . nsc 的文件。

Stream /**流**

通过网络传输的数据以及与该数据相关联的所有属性。流式数据允许播放器立即开始表现数据,而不用等到所有数据下载完成再来播放。

Stream Data Object /**流式数据对象**

表现 ASF 流(如压缩的视频帧)内部的单独数据的数据。

Stream Format /**流格式**

播放器正确表现流所必需的正确设置的信息。该信息包含的设置包括比特率、图像大小和编码器等。流格式可以是模板流格式或自定义流格式。流格式包含在 . nsc,. asd 和 . asf 文件中。

Template Stream Format (TSF) /**模板流格式(TSF)**

在 Windows Media 技术中预定义的一组设置,用来与相应的音频和视频编解码

器的内容类型和比特率相匹配。Windows Media 编码器使用这个功能帮助用户快速配置编码器以创建 ASF 内容。

Time-To-Live（TTL）/生存时间（TTL）

在多播过程中，定义多播可以在路由器停止转发多播之前通过的路由器数目的值。TTL 等价于作用域。

U ~ Z

Unicast /单播

客户端/服务器连接。客户端从服务器接收一个存储内容的点播流，或接收实况内容的广播。其他客户端无法访问该流。相反，单个多播流可以同时用于多个客户端。

Unicast Rollover /单播翻转

当 Microsoft Windows Media Player 无法从 Windows Media 服务器广播站接收到多播时，接着进行的一个过程。Windows Media Player 无法接收多播的原因很多，包括网络没有启用多播路由器。如果无法接收多播，将使用包含在 .nsc 文件中的单播翻转 URL 以连接到服务器并请求一个单播流。

Universal Naming Convention（UNC）/通用命名约定（UNC）

又称统一命名约定。该约定用来指定目录、服务器和其他网络资源，使用双斜线（//）或双反斜线（hh）以指定计算机名称，单斜线以指定计算机内的路径或目录层，格式如：hhcomputer hdirectory。

URL Flips / URL 交换

用于浏览器的一组命令，用来改变显示在 Web 页上的内容，而不考虑显示的状态。这是允许用户不用等到第一页的内容完全显示出来就可以跳转到其他页的原因。

URL Rollover / URL 翻转

用来指定不同的 Windows Media 服务器包含同样的内容的翻转方法。例如，如果 .asx 文件中的第一个 REF 标记指定一个名为 hound1 的服务器上的 .asf 文件，而第二个 REF 标记在服务器 hound2 上指定该文件的一个副本，则 Windows Media Player 可以使用任意一个服务器找到该文件。若“hound1”正处忙碌中或出错，Windows Media Player 自动连接“hound2”。

User Datagram Protocol（UDP）/用户数据报协议（UDP）

TCP/IP 协议组中的无连接的传输协议。

Video Capture Card /视频捕获卡

计算机上提供数字化图像的附加卡。使用视频捕获卡，可以给 Windows Media 编码器提供实况摄像或录像机输入。

VidToASF

命令行实用工具。可以快速将编辑好的 .avi 或 .mov 文件转换成 .asf 文件，以便于存储在 Windows Media 服务器并流向客户端。

WavToASF

命令行实用工具，可以快速将编辑好的 .wav 音频文件转换成 .asf 文件，以便存储在 Windows Media 服务器并流向客户端。

Windows Media Administrator / Windows Media 管理器

基于 Web 的管理应用程序，实时监视 Windows Media 组件服务、管理内容并配置系统。

Windows Media ASF Indexer / Windows Media ASF 索引程序

基于 Windows 的实用工具，用于删除存储在 Windows Media 编码器中的 ASF 流的一部分。还可以使用 Windows Media ASF 索引程序编辑属性、标记和脚本命令。

Windows Media Audio(. wma) File/Windows Media 音频(. wma)文件

高级流格式文件的一个特殊类型,与使用 Windows Media 音频编码器编码的纯音频内容一起使用。

Windows Media Audio Redirector (. wax) File / Windows Media 音频转向器 (. wax)文件

. asx 元文件的一个特殊类型,与 . wma 文件一起使用。wax 文件包含有关 Windows Media 服务器上的 . wma 文件的位置及其属性的信息。

Windows Media Author

创建并测试描述性的音频的图形界面工具。该工具用于合并及同步音频和图像文件。使用该工具,作者可以管理对象的声音、图像和 URL,以便它们在播放过程中在正确的时间显示。该工具使用了 Digital Renaissance, Inc. 的技术。

Windows Media Client / Windows Media 客户端

称为 Microsoft Windows Media Player 的 ActiveX 控件,从 Windows Media 服务器组件中接收并显示 ASF 内容。客户端可与服务器在同一台计算机上,也可在另外一台计算机上。

Windows Media Component Services / Windows Media 组件服务

运行在 Windows Media 服务器上的一组服务。这些服务向客户端计算机多播或单播实况音频、视频演出及存储的文件。

Windows Media Encoder / Windows Media 编码器

Windows Media 技术的一个功能,用来创建实况 ASF 流。Windows Media 编码器将实况音频和视频内容转换成 ASF 流并将其通过某个端口进行分发。Windows Media 编码器还可以将 ASF 流保存为 . asf 文件。Windows Media 编码器可以通过 MSBD 协议或 HTTP 来分发 ASF 流。

Windows Media Plug-in for Adobe Premiere /用于 Adobe Premiere 的 Windows Media 插件

一个实用程序，允许内容创建者使用 Adobe Premiere 制作 Windows Media 技术的 ASF 内容。

Windows Media Presenter for Microsoft PowerPoint 97 /用于 Microsoft PowerPoint 97 的 Windows Media Presenter

Windows Media 技术的一个功能，可从 Microsoft PowerPoint 97 内获得。它允许 PowerPoint 连接到 Windows Media 编码器，并将 PowerPoint 演示文稿发送到 Windows Media 服务器以便分发到客户端计算机。

Windows Media Program(.nsp) File / Windows Media 节目(.nsp)文件

包含有关 Windows Media 服务节目信息的文件，主要用来备份及恢复 Windows Media 服务节目定义。

Windows Media Server Components / Windows Media 服务器组件

Windows Media 服务的另一个术语，用于向客户端计算机多播或单播实况音频、视频演及存储的文件。既包含 Windows Media 组件服务(运行于 Windows Media 服务器上)，又包含 Windows Media 管理器(用于管理这些服务)。

Windows Media Services / Windows Media 服务

Windows Media 服务器组件的另一个术语，用于向客户端计算机多播和单播实况音频、视频演出及存储的文件。既包含 Windows Media 组件服务(运行于 Windows Media 服务器上)，又包含 Windows Media 管理器(用于管理这些服务)。

Windows Media Station(.nsc) File/Windows Media 广播站(.nsc)文件

向播放器描述广播站的文件。播放器直接通过一个 .asx 文件将客户端引导到一个特定的 .nsc 文件的方法来访问广播站文件。

Windows Media Monitor Service / Windows Media 监视器服务

Windows Media 组件服务的一种。

Windows Media Program Service / Windows Media 节目服务

Windows Media 组件服务的一种。

Windows Media Station Service / Windows Media 广播站服务

Windows Media 组件服务的一种，为 ASF 流提供多播、分发和存储功能。它可以管理多个广播站，每一个广播站都有一个 ASF 流作为输入，并将该流引导到一个多播地址、一个或多个分发服务器、磁盘或上述各者的组合。对于单播 ASF 流，还提供同样的功能，即 Windows Media 单播服务。

Windows Media Technologies / Windows Media 技术

流媒体应用程序家族，包含 Windows Media 服务、Windows Media 工具以及 Windows Media Player。Windows Media 工具创建 ASF 内容，这些内容可以为使用 Windows Media 服务的客户端计算机服务，并由 Windows Media Player 进行播放。

Windows Media Tools / Windows Media 工具

一套用来为 Windows Media 服务创建 ASF 内容的工具。这些工具包含 Windows Media 编码器、Windows Media Author 和 Windows Media ASF 索引程序、转换实用工具 VidToASF 和 WavToASF 以及文件工具 ASFCheck 和 ASFChop。

Windows Media Unicast Service / Windows Media 单播服务

Windows Media 组件服务的一种，给 ASF 流提供单播功能。此服务管理客户端连接的发布点，目的是接收广播单播流或点播单播流。对于多播 ASF 流，还提供了同样的功能，即 Windows Media 广播站服务。

附录2 常见问题

• 问 如何得到 Media Encoder 中文件转换过程的完成百分比？

答 在 IWMEncStatistics Interface 中，用 get_EncodingTime 得到进行的时间，用 get_FileArchiveStats 得到总的时间，就可以计算出来。

• 问 如何列举现有显示设备？

答 参考以下代码：

```
// 显示设备的函数
Function GetVideoDevice()
On Error GoTo error_handler
Dim Encoder As WMEncoder
Dim DeviceInfo As IWMEncPluginInfo

Dim DeviceInfoMgr As IWMEncSourcePluginInfoManager

Dim i As Integer
Dim J As Integer
Dim AltPath As String

Set Encoder = New WMEncoder
Set DeviceInfoMgr = Encoder.SourcePluginInfoManager

i = DeviceInfoMgr.count
i = i - 1

While i >= 0
Set DeviceInfo = DeviceInfoMgr.Item(i)
If LCase(DeviceInfo.SchemeType) <> "file" Then
AltPath = DeviceInfo.SchemeType
AltPath = AltPath & "://"

If DeviceInfo.mediatype = WMENC_AUDIO Then

If DeviceInfo.Resources = False Then
cmbAudioDrivers.AddItem DeviceInfo.Name
```

```
cmbAudioDrivers. ListIndex = 0
Else
J = DeviceInfo. count
J = J - 1

While J > = 0
cmbAudioDrivers. AddItem AltPath & DeviceInfo. Item(J)

J = J - 1
cmbAudioDrivers. ListIndex = 0
Wend
End If
End If
If DeviceInfo. mediatype = WMENC_VIDEO Then

If DeviceInfo. Resources = False Then
cmbVideoDrivers. AddItem DeviceInfo. Name
cmbVideoDrivers. ListIndex = 0
Else
J = DeviceInfo. count
J = J - 1

While J > = 0
cmbVideoDrivers. AddItem AltPath & DeviceInfo. Item(J)
J = J - 1
cmbVideoDrivers. ListIndex = 0
Wend
End If
End If

End If
i = i - 1
Wend
Set Encoder = Nothing
Exit Function
```

• 问：CLSCTX_INPROC_SERVER 与 CLSCTX_LOCAL_SERVE 有什么区别？

答：CLSCTX_INPROC_SERVER 表示客户希望创建同一进程内运行的组件。组件必须是以 DLL 实现的。

CLSCTX_LOCAL_SERVER 表示客户希望创建同一台机器中的另外一个进程中运行的组件，是由 EXE 实现的。

• 问 如何在分发时屏蔽指定的 IP 地址？

答：“Windows Media 编码器”支持使用 IP 掩码。这些掩码允许用户指定能访问编码器的单个 IP 地址或一组 IP 地址，也可以指定拒绝对编码器进行访问的 IP 地址。

此访问信息被保存在两个注册表项中，可以使用“注册表编辑器”用指定的 IP 地址更新这两个注册表项。这两个注册表项是：

①HKEY_LOCAL_MACHINE hSOFTWARE hMicrosoft hNetShow hAccessLists hAllow-DistributionList。

②HKEY_LOCAL_MACHINE hSOFTWARE hMicrosoft hNetShow hAccessLists hDisal-lowDistributionList 。

如果同时在这两个项中添加了相同的 IP 地址，则 DisallowDistributionList 项将有优先权并且访问将被拒绝。如果这些项没有列在注册表中，则可以使用“编辑”菜单添加它们。

这是使用手工更改，同样，可以使用程序修改注册表中的这些数值来达到限制 IP 地址的目的。

将 IP 地址添加到访问列表的操作如下：

① 启动“注册表编辑器”（单击“开始”→“运行”，然后输入 regedit）。

②在“注册表编辑器”中，使用树状视图找到以下路径：HKEY_LOCAL_MA-CHINE hSOFTWARE hMicrosoft hNetShow hAccessLists。**如果没有列出此项，可使用“编辑”菜单添加该项。**

③单击 AllowDistribution 以包括特定的 IP 地址；单击 DisallowDistribution 来排除特定的 IP 地址。如果没有列出这些项，可以使用“编辑”菜单添加它们。

④ 在“编辑”菜单上，指向“新建”然后单击“字串值”。

⑤ 在“名称”列中键入希望添加到列表中的 IP 地址的值，然后按 Enter 键。在访问列表中指定 IP 地址范围操作如下：

①启动“注册表编辑器”（单击“开始”→“运行”，然后输入 regedit）。

②在“注册表编辑器”中，使用树状视图找到以下路径：HKEY_LOCAL_MA-CHINE hSOFTWARE hMicrosoft hNetShow hAccessLists。**如果没有列出此项，可使用“编辑”菜单添加该项。**

③单击 AllowDistribution 以包括一组 IP 地址；单击 DisallowDistribution 来排除特

定的 IP 地址。如果没有列出这些项,可以使用‘ 编辑 ’菜单添加它们。

④在‘ 名称 ’列中,单击 IP 地址。

⑤在‘ 编辑 ’菜单上,单击‘ 修改 ’以显示‘ 编辑字符串 ’对话框。

⑥在‘ 数值数据 ’中,键入掩码值,掩码值将被添加到‘ 数据 ’列。

与 IP 地址类似,掩码是 32 位值。要设置地址范围,则掩码中的每一位都要对应 IP 地址中相应的位。如果掩码中的值是 1,则 IP 地址中的相应位将被包括在列表中。如果掩码中的值是 0,则任何值都可以接受。例如,在十进制符号中,如果在列表中的 IP 地址是 134. 123. 123. 20,而掩码是 255. 255. 255. 0,则所有从 134. 123. 123. 0 ~ 134. 123. 123. 255 的 IP 地址都将被包括在该列表中 如果掩码是 255. 255. 255. 128,则所有从 134. 123. 123. 0 ~ 134. 123. 123. 127 的 IP 地址都将被包括在该列表中 如果指定无效的掩码,则 IP 地址将被忽略;如果保留 IP 掩码为空,则它将被认为是 255. 255. 255. 255。

根据访问权限,使用‘ 注册表编辑器 ’更改注册表。不正确地编辑注册表可能对系统造成严重的损害。更改注册表之前,请备份计算机上任何有价值的数据。

附录3

Windows Media Player控件属性、方法和事件

附表 3.1 Windows Media Player 控件属性列表

附表 3.1 Windows Media Player 控件属性

属 性	权 限	解 释
AllowChangeDisplaySize	可读/可写	Sets or retrieves a value specifying whether the display size can be changed.
AllowScan	可读/可写	Sets or retrieves a value specifying whether scanning is enabled for files that support scanning (fast-forwarding and rewinding).
AnglesAvailable	只读	Retrieves a value specifying the number of available angles.
AnimationAtStart	可读/可写	Sets or retrieves a value specifying whether animation runs before the first image displays.
AudioStream	可读/可写	Sets or retrieves a value specifying the stream number of the current audio stream.
AudioStreamsAvaliable	只读	Retrieves a value specifying the number of available audio streams.
AutoRewind	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control automatically returns to the clip 's starting point after the clip finishes playing or has otherwise stopped.
AutoSize	可读/可写	Sets or retrieves a value indicating whether the Windows Media Player control automatically sizes to the proportions of the original media rendered at the current display size.
AutoStart	可读/可写	Sets or retrieves a value specifying whether to start playing the clip automatically.
Balance	可读/可写	Sets or retrieves a value indicating the stereo balance.
Bandwidth	只读	Retrieves the bandwidth of the current clip in bits per second.
BaseURL	只读	Retrieves a value specifying the base URL.
BufferingCount	只读	Retrieves a value specifying the number of times buffering occurred during playback of a clip.
BufferingProgress	只读	Retrieves a value specifying the percentage of buffering completed.

续表

属 性	权 限	解 释
BufferingTime	只读	Retrieves a value specifying the buffering time in seconds.
ButtonsAvailable	只读	Retrieves a value specifying the number of available buttons.
CanPreview	只读	Retrieves a value indicating whether the current content contains a playlist that can be previewed.
CanScan	只读	Retrieves a value specifying whether the current file supports scanning.
CanSeek	只读	Retrieves a value specifying whether the current file has the ability to seek to a specific time.
CanSeekToMarkers	只读	Retrieves a value specifying whether markers in the file can be located with a seek operation.
CaptioningID	只读	Retrieves a value representing the name of the frame or control that displays captioning.
CCActive	可读/可写	Sets or retrieves a value specifying the closed captioning service state (on or off).
ChannelDescription	只读	Retrieves a value representing the station description.
ChannelName	只读	Retrieves a value representing the station name.
ChannelURL	只读	Retrieves a value representing the URL of the station metafile.
ClickToPlay	可读/可写	Sets or retrieves a value specifying whether the user can toggle playback by clicking the video image.
ClientID	只读	Retrieves a value indicating the unique ID of a client.
CodecCount	只读	Retrieves a value specifying the number of installable codecs used by the current clip.
ColorKey	可读/可写	Sets or retrieves the color key being used by the DVD playback.
ConnectionSpeed	只读	Retrieves a value specifying the bandwidth selected during setup.
ContactAddress	只读	Retrieves a value representing the station's contact address.

续表

属　性	权　限	解　释
ContactEmail	只读	Retrieves a value representing the station 's e-mail address.
ContactPhone	只读	Retrieves a value representing the station 's contact telephone number.
CreationDate	只读	Retrieves a value specifying the date and time when the clip was created.
CurrentAngle	可读/可写	Retrieves a value specifying the date and time when the clip was created.
CurrentAudioStream	可读/可写	Sets or retrieves a value specifying the current audio stream.
CurrentButton	只读	Retrieves a value specifying the number of the current button.
CurrentCCService	可读/可写	Sets or retrieves a value specifying the current closed captioning service.
CurrentChapter	只读	Retrieves a value specifying the chapter number currently being played.
CurrentDiscSide	只读	Retrieves a value specifying the current disc side.
CurrentDomain	只读	Retrieves a value specifying the current DVD domain of the DVD player.
CurrentMarker	可读/可写	Sets or retrieves a value indicating the current marker number.
CurrentPosition	可读/可写	Sets or retrieves a value representing the clip 's current position, in seconds.
CurrentSubpictureStream	可读/可写	Sets or retrieves a value specifying the source of the subpicture.
CurrentTime	只读	Retrieves a value specifying the current playback time.
CurrentTitle	只读	Retrieves a value specifying the title number currently being played.
CurrentVolume	只读	Retrieves a value specifying the volume number for the current root directory.
CursorType	可读/可写	Sets or retrieves a value specifying the cursor type.
DefaultFrame	可读/可写	Sets or retrieves a value representing the default target HTTP frame.

续表

属 性	权 限	解 释
DisplayBackColor	可读/可写	Sets or retrieves a value specifying the display panel 's background color.
DisplayForeColor	可读/可写	Sets or retrieves a value specifying the display panel 's foreground color.
DisplayMode	可读/可写	Sets or retrieves a value specifying whether the status bar displays the current position in seconds or frames.
DisplaySize	可读/可写	Sets or retrieves a value specifying the size of the image display window.
Duration	只读	Retrieves a value indicating the clip 's playing time in seconds.
DVD	只读	Retrieves the Windows Media Player control DVD interface.
EnableContextMenu	可读/可写	Sets or retrieves a value specifying whether the context menu appears when the user clicks the right mouse button.
Enabled	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control is enabled.
EnableFullScreenControls	可读/可写	Sets or retrieves a value indicating whether the Windows Media Player displays controls in full-screen mode.
EnablePositionControls	可读/可写	Sets or retrieves a value specifying whether the position controls are enabled on the control bar.
EnableTracker	可读/可写	Sets or retrieves a value specifying whether the trackbar control is enabled.
EntryCount	只读	Returns the number of entries contained in the current Advanced Stream Redirector（ASX）file.
ErrorCode	只读	Retrieves a value specifying the current error code.
ErrorCorrection	只读	Retrieves a value specifying the error correction type of the current clip.
ErrorDescription	只读	Retrieves a value specifying the description of the current error state.

续表

属　性	权　限	解　释
FileName	可读/可写	Sets or retrieves a value specifying the name of the clip to play.
FramesPerSecond	只读	Retrieves a value specifying the number of frames per second used by the DVD title.
HasError	只读	Retrieves a value specifying whether the Windows Media Player control currently has an error.
HasMultipleItems	只读	Retrieves a value specifying whether the current clip contains multiple items (playlists).
ImageSourceHeight	只读	Retrieves a value specifying the original image height of the current clip, in pixels.
ImageSourceWidth	只读	Retrieves a value specifying the original image width of the current clip, in pixels.
InvokeURLs	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control automatically invokes URLs in a browser (URL flipping).
IsBroadcast	只读	Retrieves a value specifying whether the source is broadcast.
IsDurationValid	只读	Retrieves a value specifying whether the value of the Duration property is valid.
Language	可读/可写	Sets or retrieves a value specifying the current locale used for national language support.
LostPackets	只读	Retrieves a value specifying the number of packets lost during transmission of the stream.
MarkerCount	只读	Retrieves a value specifying the number of markers in the current clip.
Mute	可读/可写	Sets or retrieves a value indicating the current mute state of the Windows Media Player control.
OpenState	只读	Retrieves a value indicating the state of the content source.
PlayCount	可读/可写	Sets or retrieves a value indicating the number of times a clip plays.

续表

属 性	权 限	解 释
PlayState	只读	Retrieves a value indicating the state of the Windows Media Player operation.
PreviewMode	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player is in preview mode.
Rate	可读/可写	Sets or retrieves a value specifying the clip's playback rate.
ReadyState	只读	Retrieves a value specifying the state of readiness of the Windows Media Player control.
ReceivedPackets	只读	Retrieves a value specifying the number of packets received.
ReceptionQuality	只读	Retrieves a value specifying the percentage of packets received in the last 30 seconds.
RecoveredPackets	只读	Retrieves a value specifying the number of packets recovered.
Root	可读/可写	Sets or retrieves a value specifying the directory that contains the DVD volume.
SAMIFileName	可读/可写	Sets or retrieves a value specifying the file that contains the information needed for closed captioning.
SAMILang	可读/可写	Sets or retrieves a value specifying the language displayed for closed captioning.
SAMIStyle	可读/可写	Sets or retrieves a value representing the closed captioning style.
SelectionEnd	可读/可写	Sets or retrieves a value specifying the time when playback of the current clip will stop.
SelectionStart	可读/可写	Sets or retrieves a value specifying where playback of the current clip will begin.
SendErrorEvents	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control sends error events.
SendKeyboardEvents	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control sends keyboard events.
SendMouseClickEvents	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control sends mouse click events.
SendMouseMoveEvents	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control sends mouse move events.

续表

属　性	权　限	解　释
SendOpenStateChangeEvents	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control sends open state change events.
SendPlayStateChangeEvents	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control sends play state change events.
SendWarningEvents	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control sends warning events.
ShowAudioControls	可读/可写	Sets or retrieves a value specifying whether the audio controls appear on the control bar.
ShowCaptioning	可读/可写	Sets or retrieves a value specifying whether the closed caption area is visible and closed captioning is enabled.
ShowControls	可读/可写	Sets or retrieves a value specifying whether the control bar is visible.
ShowDisplay	可读/可写	Sets or retrieves a value specifying whether the display panel is visible.
ShowGotoBar	可读/可写	Sets or retrieves a value specifying whether the Go To bar is visible.
ShowPositionControls	可读/可写	Sets or retrieves a value specifying whether the position controls appear on the control bar.
ShowStatusBar	可读/可写	Sets or retrieves a value specifying whether the status bar is visible.
ShowTracker	可读/可写	Sets or retrieves a value specifying whether the trackbar is visible.
SourceLink	只读	Retrieves a value specifying the path to the current clip.
SourceProtocol	只读	Retrieves a value specifying the protocol used to receive data.
StreamCount	只读	Retrieves a value indicating the number of media streams in the current clip.
SubpictureOn	可读/可写	Sets or retrieves a value specifying whether the subpicture is displayed.
SubpictureStreamsAvailable	只读	Retrieves a value specifying the number of available subpicture streams.

续表

属 性	权 限	解 释
TitlesAvailable	只读	Retrieves a value specifying the number of titles available in the current volume.
TotalTitleTime	只读	Retrieves a value specifying the total playback time for the current title.
TransparentAtStart	可读/可写	Sets or retrieves a value specifying whether the Windows Media Player control is transparent before play begins.
UniqueID	只读	Retrieves a value specifying the unique identifier associated with the DVD volume.
VideoBorder3D	可读/可写	Sets or retrieves a value specifying whether or not the three-dimensional video border effect is enabled.
VideoBorderColor	可读/可写	Sets or retrieves a value specifying the color of the video border.
VideoBorderWidth	可读/可写	Sets or retrieves a value specifying the width of the video border, in pixels.
Volume	可读/可写	Sets or retrieves a value specifying the volume, in hundredths of decibels.
VolumesAvailable	只读	Retrieves a value specifying the number of volumes in the volume set.

附表3.2 控件支持方法列表

附表3.2 控件支持方法

方 法	解 释
AboutBox	Displays version and copyright information about the Windows Media Player control.
BackwardScan	Searches backward through the current disc at the specified speed.
ButtonActivate	Activates the selected button.
ButtonSelectAndActivate	Selects and activates the specified button.
Cancel	Cancels the Open method before the file completes opening.
ChapterPlay	Plays the media file with the specified title index and chapter value.
ChapterPlayAutoStop	Instructs the DVD player to start playing at the specified chapter within the specified title and to play the number of chapters specified.

续表

方 法	解 释
ChapterSearch	Halts playback of the current chapter and starts playback from the specified chapter within the same title.
FastForward	Scans rapidly forward through the current clip.
FastReverse	Scans rapidly backward through the current clip.
ForwardScan	Searches forward through the current disc at the specified speed.
GetAllGPRMs	Retrieves the current contents of all general parameter registers (GPRMs).
GetAllSPRMs	Retrieves the current contents of all system parameter registers (SPRMs).
GetAudioLanguage	Retrieves a value specifying the language of the specified audio stream within the current title.
GetCodecDescription	Retrieves the descriptive name of the given codec.
GetCodecInstalled	Retrieves a value specifying whether a given codec is installed.
GetCodecURL	Retrieves the URL location containing additional information about the given codec.
GetCurrentEntry	Retrieves the current clip being played by the Windows Media Player control.
GetMarkerName	Retrieves the name of a marker, given its marker number.
GetMarkerTime	Retrieves a value specifying the presentation time of a given marker.
GetMediaInfoString	Retrieves show or clip information.
GetMediaParameter	Retrieves the value of the specified parameter for an indexed Advanced Stream Redirector (ASX) entry.
GetMediaParameterName	Retrieves the name of the specified parameter for an indexed ASX entry.
GetMoreInfoURL	Retrieves a URL to additional information about the presentation.
GetNumberOfChapters	Retrieves the number of chapters defined for the specified title.
GetStreamGroup	Retrieves the group associated with the given media stream.
GetStreamName	Retrieves the name of the given stream.
GetStreamSelected	Retrieves a value indicating whether the given stream is currently selected.
GetSubpictureLanguage	Retrieves a value specifying the language for the specified subpicture stream.
GoUp	Halts playback of the current media file and starts playback of the designated previous program chain (PGC).
IsSoundCardEnabled	Retrieves a value indicating whether the computer's sound card is enabled.
LeftButtonSelect	Selects the left directional button from the displayed menu.

续表

方　法	解　　释
LowerButtonSelect	Selects the lower directional button from the displayed menu.
MenuCall	Displays the specified menu on the screen.
Next	Jumps to the next clip in a playlist.
NextPGSearch	Halts playback of the current program and starts playback from the next program within the program chain (PGC).
Open	Asynchronously opens a specified clip.
Pause	Suspends playback at the current position in the clip.
Play	Starts playing a clip from the starting position or continues playing a paused or stopped clip.
Previous	Jumps to the previous clip in a playlist.
PrevPGSearch	Halts playback of the current program and starts playback from the previous program within the program chain (PGC).
ResumeFromMenu	Returns to playing back a title from a menu.
RightButtonSelect	Selects the right directional button from the displayed menu.
SetCurrentEntry	Selects a clipx for playback by the Windows Media Player.
ShowDialog	Displays a modal dialog box containing a specified Windows Media Player dialog.
StillOff	Resumes playback, canceling still mode.
Stop	Stops playback of the current clip.
StreamSelect	Selects the media stream specified by the given stream number.
TimePlay	Plays the media file with the specified title index, starting at the specified time.
TimeSearch	Halts playback of the current chapter and starts playback from the specified time in the same media file.
TitlePlay	Finds the media file with the specified title index and plays it back.
TopPGSearch	Halts playback of the current program and restarts playback of the current program within the program chain (PGC).
UOPValid	Retrieves a value indicating whether the specified user operation is currently valid.
UpperButtonSelect	Selects the upper directional button from the displayed menu.

附表 3.3　控件捕获事件列表

附表 3.3　控件捕获事件

事　件	解　　释
Buffering	Occurs when the Windows Media Player control begins or ends buffering.
Click	Occurs when a user clicks the mouse with the cursor on the Windows Media Player control.
DblClick	Occurs when a user double-clicks the mouse with the cursor on the Windows Media Player control.
Disconnect	Occurs when the Windows Media Player control is disconnected from the server.
DisplayModeChange	Occurs when the DisplayMode property changes.
DVDNotify	Occurs when a DVD-related notification is triggered.
EndOfStream	Occurs when the end of the clip is reached.
Error	Occurs when the Windows Media Player control has an error condition.
KeyDown	Occurs when a key is pressed.
KeyPress	Occurs when a key is pressed and released.
KeyUp	Occurs when a key is released.
MarkerHit	Occurs when a marker is reached.
MouseDown	Occurs when a mouse button is pressed.
MouseMove	Occurs when the mouse pointer is moved.
MouseUp	Occurs when a mouse button is released.
NewStream	Occurs when a new stream is started in a station.
OpenStateChange	Occurs when the Windows Media Player control changes its open state.
PlayStateChange	Occurs when the Windows Media Player control changes its play state.
PositionChange	Occurs when the current media position moves to a new position.
ReadyStateChange	Occurs when the Windows Media Player control 's state of readiness changes.
ScriptCommand	Occurs when a synchronized command or URL is received.
Warning	Occurs when the Windows Media Player control encounters a possible problem.

附表 3.4　控件包含对象列表

附表 3.4　控件包含对象

对　象	解　　释
ActiveMovie	Returns a Microsoft Active Movie player object, to provide support for backwards compatibility.
NSPlay	Returns a Microsoft NetShow Player object, to provide support for backwards compatibility.