# 第10章 ASP.NET综合实例

本章通过一个在线学习网站——“媒体学习在线系统”的开发过程，向读者介绍基于ASP.NET和SQL Server 2005开发Web应用程序的一般流程和主要技术，包括：媒体学习在线系统开发的基本过程、如何分析并设计数据库、SQL Server 数据库在网站中的应用、分层设计模式、公共类的设计与实现方法、如何设计业务对象并使用之以及在Web系统开发中运用面向对象思想，提高代码复用度等。

## 10.1 系统概述

随着 Internet 的普及，网络已成为人们学习、工作、生活不可或缺的一部分。在网络上丰富多彩的学习资源中，传统的文档资源和图片资源已不能满足学习者对学习资源的需求，学习者迫切需要通过视觉或听觉直观地对学习资源进行深入的了解，从而产生了视频学习资料和语音学习资料。本章所设计的媒体学习在线系统将构建一个虚拟的网络学习社区环境，学习者可以通过这个环境发布视频教程或语音教程进行相互学习交流。

## 10.2 需求分析

在系统开发过程中，需求分析是很重要的步骤。需求分析的目的是获取用户的信息需求、处理需求和安全性需求等。通过详细调查现实世界要处理的对象(组织、部门、企业等)，充分了解原系统(手工系统或计算机系统)工作概况，明确用户的各种需求，然后在此基础上确定新系统的功能。

媒体学习在线系统要为学习者提供学习和交流的平台。通过分析，其业务流程如图 10.1 所示。



图10.1业务流程图

由图10.1可知，媒体学习在线系统应具有基本包括视频学习功能、语音学习功能和留言功能。在基本功能之上还可以添加会员注册功能，利用会员功能对访问者的权限进行控制，例如普通用户不可以发布教程或下载教程而会员用户则可以。另外系统还需要具有美观简洁的人机界面、良好的查询功能、网站的易维护性和易操作性。

## 10.3 系统设计

系统设计是对系统的模式、系统功能等进行的构建说明，包括系统结构设计、功能模块设计、页面设计和数据库设计等内容。

### 10.3.1 系统总体结构

根据媒体学习在线系统的特点，采用B/S结构模式，即浏览器/服务器模式。这种结构是随着互联网的普及应用而发展起来的，其优点是：用户只要使用浏览器即可使用系统功能，避免了在客户端对应用程序的维护；应用改变只要修改服务器中的应用程序即可，因此使用简单，维护容易。

根据需求分析的描述，系统可设计为前台应用子系统和后台管理子系统两个部分。系统总体结构如图 10.2 所示。



图10.2 系统总体结构

其中，前台应用子系统主要实现发布教程(发布视频或语音教程、查看已发布的语音教程、查看已发布的视频教程)、浏览教程(浏览视频或语音教程、发布留言) 、登录功能、查询功能。后台管理子系统主要实现公告管理(管理公告、发布公告)、教程管理(发布教程、管理视频教程、管理语音教程)、用户管理等功能。

### 10.3.2 系统功能

前台应用子系统的功能如下：

(1) 会员登录。为了保证系统使用的安全性，只允许登录会员可以发布教程和留言，一般游客用户只能使用在线课堂功能观看和收听音视频教程。会员成功登录系统后，系统使用Session变量记录其操作权限，以后在用户执行各功能之前都先进行权限检查，只有具备执行权限时才可使用相应的功能。

(2) 会员注册。用户填写相关信息并提交，系统进行验证。

(3) 在线课堂。用户可以查找音视频教程资源，并查看和下载相应资源信息。

(4) 用户留言。用户填写对相关资源的评价信息并提交。

(5) 发布教程。用户填写对相关资源的简介信息，同时上传有关资源文件。

后台管理子系统的功能如下：

(1) 公告管理。发布公告信息并进行维护。

(2) 教程管理。包括发布教程、音频管理、视频管理。

(3) 用户管理。系统用户信息的维护。

### 10.3.3 页面设计

系统的主界面和主要功能模块界面均采用表格(Table)结构设计，系统的各项功能以导航栏形式显示页面上部，页面的右下部是主显示区，用于显示文件或维护信息。图10.3是主界面。主界面页主要实现显示导航、最新教程、教程排行、公告信息、登录功能和搜索功能。



图 10.3 系统主界面

### 10.3.4 数据库设计

本系统采用 SQL Server 2005作为后台数据库管理系统，所设计的数据库名称为 db\_study，其中包含 6 个数据表，分别是：

① tb\_Bulletin：公告信息表，用于存储管理员发布的公告信息。

② tb\_login：会员信息表，用于存储会员注册的信息。

③ tb\_Sound：语音教程信息表，用于存储语音教程信息。

④ tb\_Speak：留言信息表，用于存储用户留言的详细信息。

⑤ tb\_Type：语言类型表，用于存储用户选择的编程语言。

⑥ tb\_Video：视频教程信息表，用于存储视频教程信息。

各表的结构分别列于表10.1～表10.5中。由于视频教程信息表tb\_Video和语音教程信息表tb\_Sound结构类似，在此只给出语音教程信息表结构。在系统中还需使用到一些存储过程，在有关功能模块的实现中再作介绍。

表10.1 tb\_Bulletin公告信息表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字 段 名 | 数据类型 | 可否为空 | 说 明 |
| ID | int | 否 | 公告编号，主键 |
| Title | varchar(50) | 否 | 公告标题 |
| Content | text | 否 | 公告内容 |
| Date | datetime | 否 | 发布公告日期，默认值：now() |
| Name | varchar(30) | 否 | 公告发布人 |

表10.2 tb\_login会员信息表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字 段 名 | 数据类型 | 可否为空 | 说 明 |
| ID | int | 否 | 会员编号，主键 |
| Name | varchar(50) | 否 | 会员登录名 |
| Pass | varchar(50) | 否 | 会员密码 |
| Zname | varchar(50) | 否 | 真实姓名 |
| Sex | varchar(50) | 否 | 性别 |
| Email | varchar(50) | 否 | 电子邮件地址 |
| IDcard | varchar(50) | 否 | 身份证号 |
| PassQuestion | varchar(50) | 可 | 密码提示问题 |
| PassSolution | varchar(50) | 可 | 密码提示答案 |
| LoginDate | datetime | 否 | 会员注册日期，默认值：now() |
| Lock | int | 可 | 锁定状态，默认值：0 |

表10.3 tb\_Sound语音教程信息表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字 段 名 | 数据类型 | 可否为空 | 说 明 |
| SoundID | int | 否 | 语音教程编号，主键 |
| SoundType | int | 否 | 教程语言类型 |
| SoundName | varchar(50) | 否 | 语音教程名称 |
| SoundUrl | varchar(50) | 否 | 语音教程存储路径 |
| ClickSum | int | 可 | 语音教程点击率 |
| SoundContent | varchar(50) | 可 | 语音教程内容简介 |
| FBDate | datetime | 否 | 语音教程发布日期，默认值：now() |
| Name | varchar(50) | 否 | 语音教程发布人 |

表10.4 tb\_Speak留言信息表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字 段 名 | 数据类型 | 可否为空 | 说 明 |
| SpeakID | int | 否 | 留言编号，主键 |
| Speaksman | varchar(50) | 否 | 留言人 |
| TutorialType | varchar(50) | 否 | 教程类型 |
| TutorialID | int | 否 | 教程编号 |
| SpeakContent | varchar(50) | 可 | 留言内容 |
| SpeakDate | datetime | 否 | 留言日期，默认值：now() |

表10.5 tb\_Type语言类型表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字 段 名 | 数据类型 | 可否为空 | 说 明 |
| TypeID | int | 否 | 语言类型编号，主键 |
| TypeName | varchar(50) | 否 | 语言类型名称 |

## 10.4 公共类设计与实现

为在各功能模块之间方便地进行共享，将系统的一些功能设计为公共类，这样在需要的程序中只要引用类即可，这样可以实现代码的共享和复用。所设计的公共类包括：数据库操作类、业务功能类。

### 10.4.1 数据库操作类

数据库操作类用来完成数据库的连接操作以及数据库的查询、添加、删除和修改操作。将这几种操作编写到一个公共类里，可以减少重复代码的编写，有利于代码的维护。

**(1) 创建数据库操作类**

创建数据库操作类的方法为：在 Microsoft Visual Studio 2010 菜单栏中选择“网站”/“添加新项” 命令，在弹出的“添加新项”对话框中选择“类”，将其命名为 dataOperate.cs，如图 10.4 所示。

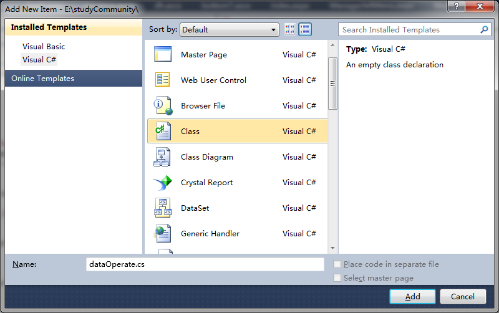


图10.4 创建操作类

单击“添加”按钮将弹出一个提示对话框，将出现对话框询问是否将刚才创建的类存放在 App\_Code 文件夹中，单击“是”按钮，完成数据库操作类的创建。

在解决方案资源管理器里的 App\_Code 文件夹中可以看到新创建的数据库操作类。双击数据库操作类，进行此类的编写。

在此类里可以看到系统自动添加的命名空间、公共类和构造函数。由于此类需要对Sql Server数据库进行操作，所以需要引用命名空间 System.Data.SqlClient。代码如下：

**using System.Data.SqlClient;**

/// <summary>

/// dataOperate 的摘要说明

/// </summary>

public class dataOperate

{

public static DataSet ds;

public dataOperate()

{

//

// TODO: 在此处添加构造函数逻辑

//

}

}

**(2) dataOperate类的方法**

在dataOperate 类中，共定义了 6 个方法来完成对数据库的基本操作。这6个方法分别是：createCon、adlData、isData、updateData、row和rows。

**1) createCon()方法**

createCon()方法用来连接数据库，此方法返回的类型为 SqlConnection，主要用来构造数据库的连接。代码如下：

public

{ SqlConnection conn = new SqlConnection("Data Source=LEOSUN-PC;Initial Catalog=db\_study;Persist Security Info=True;User ID=sa;Password=romance"); //生成SqlConnection的一个对象用于连接数据库

return conn;

}

**2) adlData()方法**

adlData(string sql)方法用来添加或删除数据。此方法返回一个布尔值，用来表示添加或删除数据是否成功，执行成功返回 true，否则返回 false。调用此方法时应传入一个 string 类型的参数，此参数表示所要执行的SQL语句。

public bool dlData(string sql)

{ SqlConnection conn =createCon(); //调用createCon方法连接数据库

conn.Open(); //打开数据库连接

SqlCommand com = new SqlCommand(sql, conn);//对SQL Server数据库执行一个SQL语句

int i = Convert.ToInt32(com.ExecuteNonQuery());//返回所影响的行并转换成int类型

conn.Close();

if (i > 0) { return true; }

else { return false; }

}

**3) isData()方法**

isData(string sql)方法用来查找数据是否存在。此方法返回一个整型值，用来表示是否查找到数据，查找到数据则返回一个大于 0 的值，否则返回 0。调用此方法时应传入一个 string 类型的参数，此参数表示所要执行的 SQL 语句。

public int isData(string sql)

{ SqlConnection conn =createCon(); //调用createCon方法连接数据库

conn.Open(); //打开数据库连接

SqlCommand com = new SqlCommand(sql, conn); //创建命令对象

int i = Convert.ToInt32(com.ExecuteScalar()); //返回命令执行状态

conn.Close();

return i;

}

**4) updateData()方法**

updateData(string sql)方法用来更新数据。此方法没有返回值。在调用此方法时应传入一个string类型的参数，此参数表示所要执行的 SQL 语句。

public void updateData(string sql)

{ SqlConnection conn =createCon(); //调用createCon方法连接数据库

conn.Open(); //打开数据库连接

SqlCommand com = new SqlCommand(sql,conn); //创建命令对象

com.ExecuteScalar(); //执行命令

conn.Close();

}

**5) row()方法**

row(string sql)方法用来查找并返回一行数据。此方法返回一个 SqlDataReader 对象。在调用此方法时传入一个string类型的参数，表示所要执行的 SQL 语句。

public SqlDataReader row(string sql)

{ SqlConnection conn =createCon(); //创建SqlConnection对象

conn.Open(); //打开数据库连接

SqlCommand com = new SqlCommand(sql, conn); //创建命令对象

return com.ExecuteReader(); //返回SqlDataReader对象

}

**6) rows()方法**

rows(string sql, string table)方法用来查找并返回多行数据。此方法返回一个DataSet数据集。调用此方法应传入string 类型的参数，表示要执行的 SQL 语句。

public DataSet rows(string sql)

{ DataSet ds;

SqlConnection conn =createCon();

conn.Open();

SqlDataAdapter adap = new SqlDataAdapter(sql,Conn);

ds = new DataSet(); //创建数据集

adap.Fill(ds); //填充数据集

conn.Close();

return ds.Tables[0]; //返回数据表

}

### 10.4.2 业务功能类

网站业务功能类用来存放开发中常用的方法，可以减少重复代码的编写，有利于代码的维护。此类中有两个自定义方法，即 Encrypting 方法和 Decrypting 方法，这两个方法主要用来完成重要信息的加密和解密。这两个方法被封装在外部的DLL文件之中，项目可以引用DLL文件，从而调用Encrypting 方法和 Decrypting 方法。

**(1) 引用DLL**

在程序中引用外部DLL文件的过程如下：

1) 在项目解决方案文件夹上右击，选中添加引用，如图10.5所示。

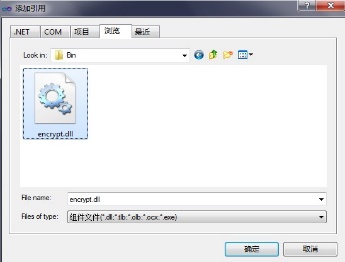
 

图10.5 添加引用 图10.6 选择dll文件

2) 选择浏览标签，浏览所需要引用的dll文件，即encrypt.dll，如图10.6所示。

3) 在所要调用Encrypting 方法和 Decrypting 方法的文件头添加如下代码：

using encrypt;

然后在文件中就可以直接调用Encrypting 方法和 Decrypting方法了。

**(2) Encrypting 方法和 Decrypting 方法**

Encrypting(string Source)方法用来对比较重要的信息进行加密操作。此方法返回一个 string 类型的值，该值表示已经加密的信息。在调用此方法时应传入一个 string 类型的参数，此参数表示需要加密的信息。代码如下：

public static string Encrypting(string strSource)

{ //把字符串存入byte数组中

byte[] bytIn = System.Text.Encoding.Default.GetBytes(strSource);

//建立加密对象的密钥和偏移量

byte[] iv = { 102, 16, 93, 156, 78, 4, 218, 32 }; //定义偏移量

byte[] key = { 55, 103, 246, 79, 36, 99, 167, 3 }; //定义密钥

//实例DES加密类

DESCryptoServiceProvider mobjCryptoService = new DESCryptoServiceProvider(); mobjCryptoService.Key = iv;

mobjCryptoService.IV = key;

ICryptoTransform encrypto = mobjCryptoService.CreateEncryptor();

//实例MemoryStream流加密文件

System.IO.MemoryStream ms = new System.IO.MemoryStream();

CryptoStream cs = new CryptoStream(ms, encrypto, CryptoStreamMode.Write);

cs.Write(bytIn, 0, bytIn.Length);

cs.FlushFinalBlock();

return System.Convert.ToBase64String(ms.ToArray());

}

Decrypting (string Source)方法将已加密的信息进行解密。此方法返回一个 string 类型的值，该值表示解密后的信息。在调用此方法时应传入一个 string 类型的参数，此参数表示需要解密的信息。代码如下：

public static string Decrypting(string Source)

{ try

{ //将解密字符串转换成字节数组

byte[] bytIn = System.Convert.FromBase64String(Source);

//给出解密的密钥和偏移量，密钥和偏移量必须与加密时的密钥和偏移量相同

byte[] iv = { 102, 16, 93, 156, 78, 4, 218, 32 }; //定义偏移量

byte[] key = { 55, 103, 246, 79, 36, 99, 167, 3 }; //定义密钥

DESCryptoServiceProvider mobjCryptoService = new DESCryptoServiceProvider(); mobjCryptoService.Key = iv;

mobjCryptoService.IV = key; //实例流进行解密

System.IO.MemoryStream ms = new System.IO.MemoryStream(bytIn, 0, bytIn.Length); ICryptoTransform encrypto = mobjCryptoService.CreateDecryptor();

CryptoStream cs = new CryptoStream(ms, encrypto, CryptoStreamMode.Read); StreamReader strd = new StreamReader(cs, Encoding.Default);

return strd.ReadToEnd();

}

catch (Exception ex)

{ throw new Exception("在文件解密的时候出现错误！错误提示： \n" + ex.Message); }

}

实现Encrypting方法和Decrypting方法需引用System.Security.Cryptography命名空间、System.IO命名空间、System.Text命名空间。

## 10.5 主页面设计与实现

首页的设计直接影响到浏览者及用户对网站的印象，因为浏览者只能看到设计者展示给的网页。首页的布局及美化对一个网站十分重要，页面布置要合理，不能太累赘，要简洁美观，让人一眼就能看到首页的重点所在。因此网站页面的设计需要由技术开发人员与美工人员协作完成。技术开发人员主要考虑的是功能全面、操作方便。

基于上述原则，本系统网站设计的首页如图 10.3 所示。由该图可见，首页中包括以下几个部分：

① 网站导航(包括发布教程、视频课堂、语音课堂、注册、联系我们)

② 教程搜索(可以根据教程类型和教程语言进行搜索)

③ 网站公告(网站近期的动态)

④ 用户登录

⑤ 最新发布教程(包括最新发布的视频教程和语音教程)

⑥ 教程排行榜(包括点击率最高的视频教程和语音教程)

### 10.5.1网站首页技术分析

**(1) 网页对话框模式显示信息**

在首页中公告的详细信息是以网页对话框模式显示的。此功能主要通过应用 Window 对象的showMadaplDialog 方法来实现。其语法格式如下：

variant=object.showMadaplDialog(sURL[,vArguments[,sFeatures]])

sURL：指定 URL 文件地址。

vArguments：用于向网页对话框传递参数，传递参数的类型不限，对于字符串类型最大为 4096 个字符，也可以传递对象。

sFeatures：对话框窗口的设置参数，可选项。设置参数如表 10.6 所示。

表 10.6对话框窗口的设置参数

|  |  |
| --- | --- |
| 参 数 | 说 明 |
| dialogWidth:number | 可选参数，用于设置对话框的宽度 |
| dialogHeight:number | 可选参数，用于设置对话框的高度 |
| DialogTop:number | 可选参数，用于设置对话框窗口相对于桌面左上角的top位置 |
| DialogLeft:number | 可选参数，用于设置对话框窗口相对于桌面左上角的left位置 |
| Center:{yes|no|1|0} | 可选参数，用于指定是否将对话框在桌面上居中，yes|1为居中显示，no|0为不居中显示，默认值为yes |
| Help:{yes|no|1|0} | 可选参数，用于指定对话框窗口中是否显示上下文敏感的帮助图标。默认值为yes |
| Scroll:{yes|no|1|0} | 可选参数，用于指定对话框是否出现滚动条 |
| Resizable:{yes|no|1|0} | 可选参数，用于指定对话框窗口大小是否可变。默认值为no |
| Status:{yes|no|1|0} | 可选参数，用于指定对话框是否出现状态栏 |

在本程序中利用 JavaScript 语言将弹出窗口设计成一个方法，以方便在页面中使用。代码如下：

<script>

function openPWD(传入打开窗口的值)

{ window.showMadaplDialog("需要打开窗口的路径"+传入打开窗口的值," ","dialogHeight: 300px; dialogWidth: 480px;dialogTop:px; dialogLeft:px; edge: Raised; center: Yes; help: No; resizable: No; status: No;scroll:No"); }

</script>

**(2) 滚动显示公告信息**

滚动显示公告信息主要通过 marquee 元素来实现。marquee 元素可以滚动文字、图片、表格等。 使用marquee 元素实现滚动信息的代码如下：

<marquee direction="up" onmouseout="this.start()" onmouseover="this.stop()" scrollamount="4" style="width: 236px; height: 133px"></marquee>

### 10.5.2 网站首页的实现

**(1) 设计步骤**

① 在该网站中新建名为 index.aspx 的Web 窗体，用于显示网站的首页。

② 在 Web 窗体中添加 Tabel 表格，用于页面的布局。

③ 在页面中添加服务器控件，控件属性设置及其用途如表 10.7 所示。

表10.7 首页中各控件名称、属性设置及用途

|  |  |  |  |
| --- | --- | --- | --- |
| 控件类型 | 控件名称 | 主要属性设置 | 控件用途 |
| Web用户控件 | dh.ascx | 均为默认值 | 导航功能 |
| bottomT.ascx | 均为默认值 | 页面底部信息 |
| entry.ascx | 均为默认值 | 用户登录 |
| search.ascx | 均为默认值 | 搜索功能 |
| userBulletinInfo.ascx | 均为默认值 | 显示公告信息 |
| userSound.ascx | 均为默认值 | 显示语音排行 |
| userVideo.ascx | 均为默认值 | 显示视频排行 |
| GridView | gvNewVideo | AutoGenerateColumns 属性：False | 显示最新视频 |
| gvNewSound | 同上 | 显示最新语音 |
| Panel | PanelEntry | 均为默认值 | 显示或隐藏登录界面 |
| PanelHello | 均为默认值 | 显示或隐藏欢迎界面 |

首页中使用了7个用户控件，为在各功能模块之间方便地切换，将系统的主要功能设计为一个用户控件，这样在需要的页面中只要加载该用户控件即可，而不必在每个页面中都进行设计。在程序中使用如下语句引用用户控件：

<%@ Register Src="dh.ascx" TagName="dh" TagPrefix="uc1" %>

**(2) 实现代码**

**1) 绑定方法至Web服务器控件**

在主页 Web 窗体的加载事件中调用各个功能绑定到 DataList 控件上的方法。程序代码如下：

protected void Page\_Load(object sender, EventArgs e)

{ ①if (!IsPostBack)

{ ②if (Session["UserName"] != null) //判断用户是否登录

{ ③PanelEntry.Visible = false; //设置登录窗口为不显示

PanelHello.Visible = true; //设置显示欢迎词

this.Label1.Text = Session["UserName"].ToString();

}

else

{ PanelHello.Visible = false; //设置不显示欢迎词

PanelEntry.Visible = true; //设置显示登录窗口

}

}

creteVideo(); //自定义方法用来绑定最新视频教程

creatSound(); //自定义方法用来绑定语音教程

}

上述程序代码中“①”~“③”为标注序号，非程序代码正式内容。

① IsPostBack方法：获取一个值，该值指示该页是否正为响应客户端回发而加载，或者它是否正被首次加载和访问，可以有效避免页面数据的多次加载。

② Session["UserName"]：用来存储用户登录名，用户登录时生成。

③ Visible属性：获取或设置一个值，该值指示服务器控件是否呈现在页面上。

2) 视频排行方法

由于公告、视频排行、语音排行、最新视频和最新语音绑定到 GridView 控件上的方法类似，这里主要介绍最新视频排行的绑定方法。

创建自定义方法，命名为 creteVideo，在方法中建立 SQL 语句，调用数据库操作中的 rows 方法并传入 SQL 语句。将返回值绑定到 GridView 上。

protected void creteVideo()

{ try

{ dataOperate mydo = new dataOperate(); //实例数据库操作类

string sql = "SELECT top 10 \* from tb\_Video as a inner join tb\_Type as b on a.VideoType=b.TypeID ORDER BY VideoID DESC";

 ①gvNewVideo.DataSource = mydo.rows(sql, "tb\_Video").DefaultView; //绑定数据源

 ②gvNewVideo.DataBind(); //调用gridview的数据绑定方法实现数据绑定

}

catch (Exception error) //获取异常

{ Response.Redirect("error.aspx"); //跳转到错误页面 }

}

① GridView.DataSource属性：表示数据源对象，数据绑定控件从该对象中检索其数据。

② GridView.DataBind：将数据源绑定到 GridView 控件。

## 10.6 子系统设计与实现

本节介绍媒体在线学习系统的部分子系统的设计与实现，限于篇幅，本节只选择其中较有代表性的模块进行讲解，其余的模块读者可自行分析与设计。

### 10.6.1 用户注册模块

用户注册页面如图 10.7 所示。浏览者可以通过用户注册功能注册成为本网站的会员，用户注册并登录后，可以发布自己制作或收集的视频教程和语音教程，也可以下载自己喜欢的教程。当用户在首页单击导航栏中“注册”链接 按钮或在登录模块中单击“新用户注册”链接按钮时，将进入用户注册的页面。

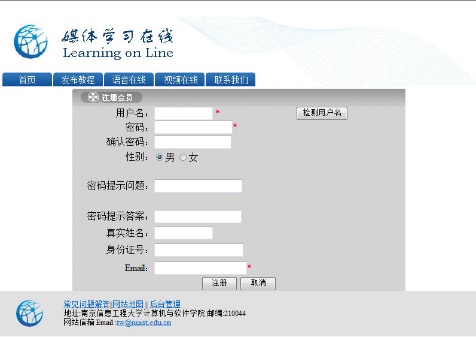


图10.7 用户注册页面

**(1) 用户注册模块设计**

用户注册需要验证其输入信息的正确性，实现用户注册信息时应注意以下事项：用户名不能为空、密码必须填写、两次密码输入必须一致、 电子邮件地址格式是否正确、身份证号位数是否正确。这些信息的验证都是通过服务器验证控件来实现的。服务器验证控件及其主要属性设置和用途如表 10.8 所示。

表10.8 用户注册页中服务器验证控件及其主要属性设置和用途

|  |  |  |
| --- | --- | --- |
| 控件名称 | 主要属性设置 | 控件用途 |
| RequiredFieldValidatorName | ControlToValidate属性：txtName | 判断用户名是否为空 |
| ErrorMessage属性：“必须填写” |
| RequiredFieldValidatorPass | ControlToValidate属性：txtPass | 判断密码是否为空 |
| ErrorMessage属性：“必须填写” |
| CompareValidatorPass | ControlToCompare属性：txtPass | 判断两次密码是否一致 |
| ControlToValidate属性：txtQrPass |
| ErrorMessage属性：“两次密码不一致” |
| RequiredFieldValidatorEmail | ControlToValidate属性：txteMail | 判断 E-mail 是否为空 |
| ErrorMessage属性：“必须填写” |
| RegularExpressionValidatorEmail | ControlToValidate属性：txtEmail | 判断邮件地址是否正确 |
| ErrorMessage属性：“邮件地址不正确” |
| ValidationExpression属性：  \w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\* |
| RegularExpressionValidatorIDCard | ControlToValidate属性：txtIDCard | 判断身份证号是否正确 |
| ErrorMessage属性：“身份证号不正确” |
| ValidationExpression属性：\d{17}[\d|X]|\d{15} |

在用户注册信息中的用户名需要是唯一的，否则在发布留言时可能会出现两个或多个一样的用户名，登录时会出现登录失败的情况。解决这个问题可以通过一个按钮控件来实现，在按钮的单击事件中通过调用 isName 方法，调用数据库操作类中的 isData 方法并返回一个整数类型的值，来判断用户添加的用户名是否在数据库中存在。如果整数值大于 0，说明用户已经存在并给出相应的提示信息；等于 0，说明用户可以添加此用户名并给出相应的提示信息。

public int isName()

{ int i;

//创建SQL语句用来查询用户表中是否存在符合登录名条件的记录

string sql = "select count(\*) from tb\_login where Name='" + this.txtName.Text.Trim() + "'";

return i = mydo.isData(sql);

}

当用户添加完合法的注册信息后，需将注册信息保存到数据库中。在保存时需要考虑用户注册信息的安全性，假如数据库被非法入侵，数据库中的重要信息，如用户的密码将会完全暴露。为了防 止此问题，可以将注册信息中的密码通过加密再存储到数据库中。为了防止用户忘记密码，在注册信息里添加了密码提示问题和密码提示答案，用户可以通过密码提示问题和答案来找回忘记的密码，这 里的密码提示答案也是比较重要的信息，所以也需要通过加密后存储到数据库中。

数据加密一般有两种：双向加密和单向加密。双向加密最常用，它的特点是既能加密又能解密；而单向加密只能对数据进行加密，不能对其解密。由于密码信息还需要找回，所以使用双向加密，双向加密通过调用网站业务功能类中的 Encrypting 方法实现。密码提示答案可使用单向加密，单向加密通过 MD5 加密实现。MD5 加密是根据指定的密码和哈希算法生成一个适合于存储在配置文件中的哈希密码。其命名空间为 System.Web.Security。密码提示答案加密代码如下：

FormsAuthentication.HashPasswordForStoringInConfigFile(this.txtPassSolution.Text, "MD5");

(2) 用户注册模块实现

① 在该网站中新建一个 Web 窗体，命名为 login.aspx，用于实现用户注册。

② 在 Web 窗体中添加 Tabel 表格，用于页面的布局。

③ 在 Tabel 表格中添加控件，控件的属性设置及其用途如表10.9 所示。

表10.9 用户注册页中各控件名称、属性设置及用途

|  |  |  |  |
| --- | --- | --- | --- |
| 控件类型 | 控件名称 | 主要属性设置 | 控件用途 |
| TextBox | txtName | 均为默认值 | 输入用户名 |
| txtPass | 均为默认值 | 输入密码 |
| txtQrPass | 均为默认值 | 输入确认密码 |
| txtPassQuestion | 均为默认值 | 输入密码提示问题 |
| txtPassSolution | 均为默认值 | 输入密码提示答案 |
| txtTrueName | 均为默认值 | 输入真实姓名 |
| txtIDCard | 均为默认值 | 输入身份证号 |
| txtEmail | 均为默认值 | 输入电子邮件地址 |
| 用户控件 | dh.ascx | 均为默认值 | 用于导航 |
| ImageButton | ibtnLogin | 均为默认值 | 实现注册功能 |
| ibtnisName | 均为默认值 | 判断用户名是否存在 |
| RadioButton | RadioButtonMan | 将GroupName设置为sex（单选按钮所属的组） | 用户注册性别选择“男” |
| RadioButtonWoman | 将GroupName设置为sex（单选按钮所属的组） | 用户注册性别选择“女” |
| Button | btnLogin | 均为默认值 | 用户注册按钮 |
| buttonCancel | 均为默认值 | 取消用户注册返回首页 |
| btnisName | 均为默认值 | 判断用户名是否已注册按钮 |

实现注册功能通过 ibtnLogin 控件的 Click 事件来实现，Click 事件中需要先调用 isName 方法来判断用户名是否存在，再通过调用 add 方法将用户添加的注册信息添加到数据库中。程序代码如下：

protected void btnLogin \_Click(object sender, EventArgs e)

{ if (isName() <= 0) //自定义方法判断用户是否存在

{ if (add())

{ //显示提示信息，Page.RegisterStartupScript：在页响应中发出客户端脚本块

Page.RegisterStartupScript("true", "<script>alert('成功！')</script>");

}

else

{ Page.RegisterStartupScript("false", "<script>alert('失败！')</script>"); }

}

else

{ Page.RegisterStartupScript("false", "<script>alert('用户名已存在！')</script>"); }

}

isName方法的实现主要通过数据库操作类中的 isData 方法并传入 SQL 语句，此方法返回一个整数 类型的值，该值大于 0 表示用户已经存在不可以注册，等于 0 表示用户名不存在可以注册。程序代码如下：

public int isName()

{ dataOperate mydo = new dataOperate(); //创建dataOperate对象

int i;

string sql = "select count(\*) from tb\_login where Name='" + this.txtName.Text.Trim() + "'";

return i = mydo.isData(sql);

}

add 方法调用了数据库操作类中的 adlData 方法并传入 SQL 语句，此方法返回一个布尔类型的值，数据添加成功该值返回 true，否则返回 false。

protected bool add()

{ string retVal = string.Empty;

string name = txtName.Text;

//调用业务功能类中Encrypting方法将用户密码加密

string pass =Operate.Encrypting(txtPass.Text);

string sex;

//获取性别

if (RadioButtonMan.Checked)

{ sex = "男"; }

else { sex = "女"; }

string trueName = txtTrueName.Text; //获取真实姓名

string idCard = this.txtIDCard.Text;　　 //获取电话

string passQuestion = this.txtPassQuestion.Text;

//根据指定的密码和哈希算法生成一个适合于存储在配置文件中的哈希密码。

string passSolution=

FormsAuthentication.HashPasswordForStoringInConfigFile(this.txtPassSolution.Text, "MD5");

string email = txtEmail.Text;　　 //获取电子邮件

//获取连接数据库字符串

string s = ConfigurationManager.ConnectionStrings["db\_studyConnectionString"].ToString();

SqlConnection Conn = new SqlConnection(s); //创建一个数据库连接对象

SqlCommand objCmd = new SqlCommand(); //创建一个数据库命令对象

objCmd.CommandText = "InsertUser";//设置执行sql语句

objCmd.CommandType = CommandType.StoredProcedure; //设置执行类型为存储过程

objCmd.Parameters.AddWithValue("@Name", name); //向存储过程中传值

objCmd.Parameters.AddWithValue("@PassWord", pass);

objCmd.Parameters.AddWithValue("@ZName", trueName);

objCmd.Parameters.AddWithValue("@Sex", sex);

objCmd.Parameters.AddWithValue("@Email", email);

objCmd.Parameters.AddWithValue("@IDCard", idCard);

objCmd.Parameters.AddWithValue("@PassQuestion", passQuestion);

objCmd.Parameters.AddWithValue("@PassSolution", passSolution);

objCmd.Parameters.Add("@sysmsg", SqlDbType.VarChar, 50);

//执行完存储过程之后的返回值

objCmd.Parameters["@sysmsg"].Direction = ParameterDirection.Output;

Conn.Open();

SqlTransaction trans = Conn.BeginTransaction();

try

{ objCmd.Connection = Conn;

objCmd.Transaction = trans;

objCmd.ExecuteNonQuery(); //执行存储过程

retVal = objCmd.Parameters["@sysmsg"].Value.ToString(); //得到返回值

trans.Commit();

if (retVal != "OK") //根据返回值判断执行是否成功

{ return false; }

return true;

}

catch (Exception ex) //捕获异常

{ trans.Rollback();

return false;

}

Finally //执行完一系列操作之后的最终操作

{ objCmd.Connection.Close();

objCmd.Dispose();

objCmd = null;

}

}

 上述程序调用的存储过程代码：

USE [db\_study]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROC [dbo].[InsertUser] (

@Name nvarchar(50),

@PassWord nvarchar(50),

@ZName nvarchar(50),

@Sex nvarchar(50),

@Email nvarchar(50),

@IDCard nvarchar(50),

@PassQuestion nvarchar(50),

@PassSolution nvarchar(50),

@sysmsg nvarchar(150) output

)

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

insert into tb\_login(Name,Pass,ZName,Sex,Email,IDCard,PassQuestion,PassSolution)

values(@Name,@Password,@ZName,@Sex,@Email,@IDCard,@PassQuestion,@PassSolution)

set @sysmsg='OK'

END

为方便用户添加用户名时查看用户名是否存在，可以先利用 ibtnisName 控件进行判断，如果用户名不存在可再添加其他的注册信息。在 ibtnisName 控件的 Click 事件中先判断用户名是否为空，如果为空，给出相应的提示对话框，不为空则调用自定义 isName方法来判断用户名是否存在并给出相应提示对话框。

protected void ibtnisName\_Click(object sender, EventArgs e)

{ if (txtName.Text.Trim() != "")

{ if (isName()> 0) //判断用户名是否存在

{ RegisterStartupScript("yes", "<script>alert('用户名已经存在！')</script>"); }

else { RegisterStartupScript("no", "<script>alert('可以注册')</script>"); }

}

else

{ RegisterStartupScript("yes", "<script>alert('用户名不能为空！')</script>"); }

}

### 10.6.2 用户登录模块

用户注册为网站会员后，必须进行登录后才能使用上传或下载教程和管理教程的功能。用户登录界面的运行效果如图 10.8 所示。



图10.8 用户注册页面

当用户登录成功后进行留言时会将发言人保存为用户的登录名，如果是普通用户发言人将保存为游客。这里主要应用了 Session 对象，利用 Session 对象来保存用户登录名。当用户发送教程时或在发布留言时可以将 Session 对象保存的用户名添加到数据库中。Session 对象是 HttpSessionState 类的一个实例，其功能是用来存储跨网页程序的变量或者对象，Session 对象只针对单一网页使用，也就是说各个连接的机器都有各自的 Session 对象，不同的客户端无法互相存取。Session 对象中止于联机机器离线时，也就是当网页使用者关掉浏览器或超过设定的 Session 变量的有效时间时，Session 对象就会消失。

创建用户登录模块程序的步骤如下：

① 在该网站中新建一个名为 entry.ascx用户控件，用于实现用户登录。

② 在 Web 窗体中添加 Tabel 表格，用于页面的布局。

③ 在 Tabel 表中添加控件，控件属性设置及其用途如表 10. 10所示。

表10.10 用户登录模块中各控件名称、属性设置及用途

|  |  |  |  |
| --- | --- | --- | --- |
| 控件类型 | 控件名称 | 主要属性设置 | 控件用途 |
| TextBox | txtName | 均为默认值 | 输入用户名 |
| txtPass | 均为默认值 | 输入密码 |
| txtYzm | 均为默认值 | 输入验证码 |
| Image | image1 | 均为默认值 | 显示验证码 |
| LinkButton | lnkbtnRe | 均为默认值 | 刷新验证码 |
| lnkbtnNew | 将PostBackUrl设置为“~/login.aspx”（单击按钮时所发送到的URL） | 新用户注册 |
| lnkbtnPass | 将PostBackUrl设置为“~/seekPass.aspx”（单击按钮时所发送到的URL） | 找回密码 |
| OmageButton | imgbtnLanding | 均为默认值 | 登录按钮 |

登录功能的实现需要先判断验证码是否正确，验证码正确才需要使用数据库操作类中的row 方法来判断用户输入的用户名和密码是否正确。程序中Session["CheckCode"]用来存储自动生成的验证码。代码如下：

protected void imgbtnLanding\_Click(object sender, EventArgs e)

{ string name = txtName.Text; //将登录名存储到变量中

string pass = Operate.Encrypting(txtPass.Text); //将密码加密并存储到变量中

string yzm = txtYzm.Text; //将验证码存储到变量中

 if (Session["CheckCode"].ToString().Equals(yzm)) //判断验证码是否正确

{ try

{string sql = "select count(\*) from tb\_login where Name='" + name

+ "' and Pass ='" + pass + "'";

int i = mydo.isData(sql); //调用数据库操作类中的isData方法判断用户是否存在

if (i > 0) //用户存在

{ sql = "select \* from tb\_login where Name='" + name + "'";

 SqlDataReader odr = mydo.row(sql); //返回一条记录

odr.Read();

if (odr["lock"].ToString() == "0") //判断用户是否被锁定

{ Session["UserName"] = name; //将用户登录名保存到Session中

Response.Redirect("index.aspx");

}

else //用户被锁定

{ Page.RegisterStartupScript("false", "<script>alert('此用户已被锁定！')</script>"); }

}

else //用户不存在

{ Response.Write("<script>alert('密码或用户名错误！')</script>"); }

}

catch (Exception ex)

{ Response.Write(ex.Message.ToString()); }

}

else //验证码错误

{ Page.RegisterStartupScript("false", "<script>alert('验证码错误！')</script>"); }

}

### 10.6.3 教程发布与管理模块

会员用户可将自己制作的或从其他途径收集到的教程发布到网站上，供其他用户共享， 还可以对自己发布的教程进行管理，以及查看其他用户的留言。当用户登录成功后单击“发布教程” 按钮，将进入到发布教程页面，如图 10.9 所示。

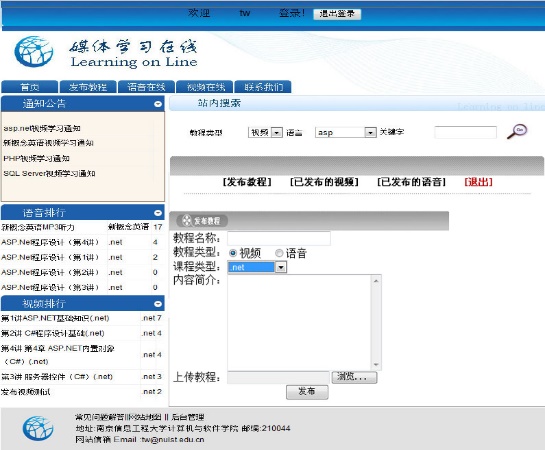


图10.9 发布教程页面

在发布教程时需用户选择语言类型，在设计数据库时已把语言类型单独放到一个数据表tb\_Type中，可利用此表数据将其绑定到 DropDownList 控件上，这样可节省开发时间和避免手动添加的错误。DropDownList 控件后台代码如下：

protected void bindLanguage()

{ //调用数据库操作类中的rows方法并将数据源绑定到DropDownList控件上

string sql = "select \* from tb\_Type ";

this.ddlLanguage.DataSource = mydo.rows(sql, "tb\_Type").DefaultView;

this.ddlLanguage.DataBind();

}

前台需要设置 DropDownList 控件的 DataTextFiield 属性和 DataValueField 属性。DataTextField属性表示 DropDownList 控件的显示文本，DataValueField 属性表示 DropDownList 控件的值。 DropDownList 控件前台代码如下：

<asp:DropDownList ID="ddlLanguage" runat="server" DataTextField="TypeName" DataValueField="TypeID"></asp:DropDownList>

在管理已经发布的教程时对发错的教程或反响不好的教程，会员用户可以将此教程删除。实现此功能主要通过 GridView 控件的 RowDeleting 事件来实现，此事件在单击某一行的“删除”按钮时，删除该行之前引发。代码如下：

protected void gvFBVideo\_RowDeleting(object sender, GridViewDeleteEventArgs e)

{ string id = gvFBVideo.DataKeys[e.RowIndex].Value.ToString(); //获取当前行ID

string vsql = "delete from tb\_Video where VideoID= " + id;

bool delVideo = mydo.adlData(vsql); //调用数据库操作类的adlDate方法删除视频

if (delVideo) //判断删除视频是否成功，如果成功删除相关的留言

{ string ssql = "delete from tb\_Speak where TutorialType='Video' and TutorialID=" + id;

mydo.adlData(ssql); //调用数据库操作类的adlDate方法删除和视频相关的留言

}

else

{ Page.RegisterStartupScript("false", "<script>alert('删除失败！')</script>"); }

Yvidoe(); //重新绑定已经发布的视频教程

}

创建教程发布与管理模块程序的步骤如下：

① 新建名为 issuance.aspx的Web 窗体，用于实现发布教程和管理教程。

② 在 Web 窗体中添加 Tabel 表格，用于页面的布局。

③ 在 Tabel 表格中添加控件，控件的属性设置及用途如表 10.11 所示。

表10.11 教程发布与管理模块中各控件名称、属性设置及用途

|  |  |  |  |
| --- | --- | --- | --- |
| 控件类型 | 控件名称 | 主要属性设置 | 控件用途 |
| Panel | panelFB | 均为默认值 | 用于显示或隐藏发布信息 |
| panelYvideo | 均为默认值 | 用于显示或隐藏已发布的视频教程 |
| panelYsound | 均为默认值 | 用于显示或隐藏已发布的语音教程 |
| GridView | gvFBVideo | 均为默认值 | 绑定已发布的视频教程 |
| gvFBSound | 均为默认值 | 绑定已发布的语音教程 |
| TextBox | txtName | 均为默认值 | 输入教程名称 |
| txtContent | TextMode设置为“MultiLine” | 输入教程内容简介 |
| RadioButton | rdibtnVideo | Checked设置为true  GroupName设置为Type（单选按钮所属的组） | 选择教程类型为视频类型 |
| rdibtnSound | Checked设置为true（已选中状态），  GroupName设置为Type（单选按钮所属的组） | 选择教程类型为语音类型 |
| DropDownList | ddlLanguage | 均为默认值 | 选择语言类型 |
| FileUpLoad | fileUpload1 | 均为默认值 | 选择要上传的教程 |
| LinkButton | btnFB | 均为默认值 | 发布教程按钮 |
| ImageButton | imgbtnIssuance | 均为默认值 | 显示发布教程按钮 |
| imgbtnYFVideo | 均为默认值 | 显示已发布视频按钮 |
| imgbtnYFSound | 均为默认值 | 显示已发布语音按钮 |
| imgbtnExit | 均为默认值 | 返回首页按钮 |

在主页 Web 窗体的加载事件中需要先判断用户是否登录，如果登录才可以发布教程或管理教程，如果没有登录将弹出对话框跳转到首页。程序代码如下：

protected void Page\_Load(object sender, EventArgs e)

{ if (!IsPostBack)

{ panelYsound.Visible = false; //不显示语音管理面板

panelFB.Visible = false; //不显示发布教程面板

panelYvideo.Visible = false; //不显示视频管理面板

if (Session["UserName"] == null) //判断用户是否登录

{ RegisterStartupScript("null", "<script>alert('请先登录！');location='index.aspx'</script>"); } else

{ PanelFB.Visible = true; //用户登录显示发布教程 }

}

Yvidoe(); //自定义方法用来绑定已发布的视频

Ysound(); //自定义方法用来绑定已发布的语音

bindLanguage(); //自定义方法用来绑定DropDownList控件的选项

}

发布教程时，当用户将教程信息添加正确后单击“发布”按钮，将用户所发布的教程信息保存到数据库中。代码如下：

protected void btnFB \_Click(object sender, EventArgs e)

{ string vsname = txtName.Text; //获取教程名

string isVS;

if (rdibtnSound.Checked) //获取教程类型

{ isVS = "tb\_Sound"; }

else { isVS = "tb\_Video"; }

string typ = this.ddlLanguage.SelectedValue; //获取语言类型

string content = txtContent.Text; //获取内容简介

string name = Session["UserName"].ToString(); //获取用户登录名

int clicksum = 0; //初始化点击率

string Path = "";

try

{ string sql = "";

if (isVS == "tb\_Sound") //判断教程是否是语音类型

{ //判断用户上传的语音文件类型

 if (FileUpload1.FileName.Substring(FileUpload1.FileName.LastIndexOf(".")+1)=="mp3")

{ //判断用户上传的语音教程是否存在

 if (!File.Exists(Server.MapPath(".") + "\\Sound\\" + this.FileUpload1.FileName))

{ sql = "insert into tb\_Sound(SoundType,SoundName,SoundUrl,ClickSum,SoundContent,Name) values('" + typ + "','" + vsname + "','" + this.FileUpload1.FileName + "','" + clicksum + "','" + content + "','" + name + "')";

//设置路径用于保存上传的语音

 Path = Server.MapPath("./") + "Sound" + "\\" + this.FileUpload1.FileName;

if (mydo.adlData(sql))

{ //将文件保存到指定位置

 FileUpload1.PostedFile.SaveAs(Path);

Page.RegisterStartupScript("true", "<script>alert('上传成功！')</script>");

}

else

{ Page.RegisterStartupScript("true", "<script>alert('上传失败！')</script>"); }

}

else Page.RegisterStartupScript("false", "<script>alert('教程名称已经存在！')</script>");

}

else RegisterStartupScript("false", "<script>alert('只能上传mp3类型！')</script>");

}

else

{ //判断用户上传的视频文件类型

if (FileUpload1.FileName.Substring(FileUpload1.FileName.LastIndexOf(".") + 1) == "wmv")

{ //判断用户上传的视频教程是否存在

if (!File.Exists(Server.MapPath(".") + "\\Video\\" + this.FileUpload1.FileName))

{ sql = "insert into tb\_Video(VideoType,VideoName,VideoUrl,ClickSum,VideoContent,Name) values(" + typ + ",'" + vsname + "','" +this.FileUpload1.FileName + "'," + clicksum + ",'" + content + "','" + name + "')";

//设置路径用于保存上传的视频

Path = Server.MapPath("./") + "Video" + "\\" + this.FileUpload1.FileName;

if (mydo.adlData(sql))

{ FileUpload1.PostedFile.SaveAs(Path);

Page.RegisterStartupScript("true", "<script>alert('上传成功！')</script>");

}

else

{ Page.RegisterStartupScript("true", "<script>alert('上传失败！')</script>"); }

}

else

{ Page.RegisterStartupScript("false", "<script>alert('教程名称已经存在！')</script>"); }

}

else RegisterStartupScript("false", "<script>alert('只能上传wmv类型！')</script>");

}

}

catch (Exception ex)

{ Page.RegisterStartupScript("false", "<script>alert('上传教程不能为空！')</script>"); }

}

这里选择把需要上传的文件上传到服务器中指定的文件夹里面，然后把虚拟路径存储在数据库中，当需要使用的时候直接按照虚拟路径就能找到文件了。如果直接将文件存储在数据库中，由于文件较大，当用户使用的时候会给数据库带来很大的压力，会降低数据库交互的性能，所以把文件上传到服务器上，而不是直接存储在数据库中。

该模块程序进行调试时，可能会出现发布教程不成功。出现这个错误有可能是因为发布的教程过大而造成发布失败，因为在Web.Config文件中配置上传文件的大小默认值为4096(4MB)。解决此问题可在Web.Config文件中<hpptRuntime>节中重新设置masRequestLength 属性和 ExecutionTimeout属性。

其中，maxRequestLength属性：用于防止服务攻击，例如因用户向服务器发送大型文件而导致的拒绝访问，默认值为4096。ExecutionTimeout 属性：用于指定在ASP.NET应用程序自动关闭前，允许执行请求的最大秒数。只有当 compilation 元素中的调试属性为false时，此超时属性才适用。默认值为 110s。

本程序中允许上传大小设置为 40MB。实现代码如下：

<httpRuntime maxRequestLength="40960" executionTimeout="6000" />