# 概述

## 1.这个播放器使用window media player组件开发的.可以播放视频和音乐,也可以播放网络多媒体资源

## 2.这里有一个音乐资源下载地址: <https://pixabay.com/music/search/mp3/>

## 3.还有这个网站: <https://www.myfreemp3.com.cn/>

## 4.自己部署的GitHub网页: <https://kennycaiguo.github.io/mymusic/>

### 需要先去<https://github.com/kennycaiguo/mymusic> 找到对应的音乐文件然后拼接到上面的url后面

## 如: <https://kennycaiguo.github.io/mymusic/AfterAll.mp3>

# 开始开发

## 1.打开vs2022,新建一个c#winform 应用程序(.net Framework)

|  |
| --- |
|  |

## 2.给这个项目取名:csharpMediaPlayer,点击创建

|  |
| --- |
|  |
|  |

## 3.把Form1.cs改名为MediaForm.cs

|  |
| --- |
|  |

## 4.把窗体的标题改为c#多媒体播放器

|  |
| --- |
|  |

## 5.把window media player组件添加到工具箱,可以在工具箱上面点击右键->添加选项卡

|  |
| --- |
|  |

## 6.然后工具箱就会新建一个选项卡,我们把他的名字叫做媒体控件,按回车

|  |
| --- |
|  |

## 7.然后在这个媒体控件上面点击右键->选择项,就可以弹出一个添加组件的窗口,

|  |
| --- |
|  |
|  |

## 8.点击COM选项卡,找到Window Media Player组件,把它勾选,点击确定

|  |
| --- |
|  |

## 9.然后Windows Media Player组件就添加到工具箱了

|  |
| --- |
|  |

## 10.然后把这个媒体播放器组件拖动到winform中,可以先尝试自己操作一下,发现这个控件有2种播放媒体的方式一种是使用它自己的openPlayer方法,使用这个方法后,媒体会在一个弹出窗口里面播放,另外一种方法是把控件的URL属性设置为需要播放的媒体文件,这种方法不会弹出新窗口,而是在控件窗口里面播放,这种才是我们这里需要的播放方式.

## 11.我们需要给窗体添加菜单,在工具箱里面找到MenuStrip控件,把它拖进窗体,然后添加子菜单

|  |
| --- |
|  |
|  |
|  |

### 其实这个菜单项在c#里面都是ToolStripMenuItem类对象,我们可以看看XXXForm.Designer.cs文件.它长这样子

|  |
| --- |
|  |

## 12.设置好菜单后,我们修改一下界面,先把wmp控件缩小一点,添加一个分割面板SplitContainer,注意在放置分割面板之前,需要把窗体的AutoScaleMode设置为None,否则会出现控件拖进分割面板看不见的效果

|  |
| --- |
|  |

## 13,然后把分割面板拉大一点,把播放器控件放到分割面板的右边,然后选择播放器控件,把它的Dock属性改为Fill,效果如下

|  |
| --- |
|  |

## 14.在分割面板的左边放置一个TreeView控件

|  |
| --- |
|  |

## 15.调整一下treeView控件,可以在它右上角的小三角形那里点击一下,然后点击在父容器在停靠

|  |
| --- |
|  |
|  |

## 16.然后选择分割面板控件,做跟上面一样的操作,效果如下

|  |
| --- |
|  |

## 17.在项目上面点击右键->添加->类,会出现下面的界面

|  |
| --- |
|  |

## 18.选择类,然后给他取名:FileModel,点击添加按钮

|  |
| --- |
|  |

## 19.此时系统就会版我们生成一个FileModel类,我们需要把它改为public,然后我们来添加属性,这里可以使用枚举来帮助我们更好的定义一些类型,这个类的代码如下,注意那两个枚举是在类外面定义的.

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace csharpMediaPlayer  {  /// <summary>  /// 处理媒体文件的类  /// </summary>  public class FileModel  {  public string FileGuid{ get; set; } //文件的全局唯一标识符  public string FileName{ get; set; } //文件名  public FileType FileType{ get; set; } //文件类型,是一个枚举变量  public string FilePath { get; set; }//本地路径或者网络路径都可以  public int OrderNo { get; set; } //排序编号  public ResourceType ResourceType { get; set; } //资源类型,是一个枚举变量  }  public enum ResourceType //资源类型枚举,注意c#中枚举的写法  { /// <summary>  /// 本地资源  /// </summary>  Local=1,  /// <summary>  /// 网络资源  /// </summary>  Network =2  }  public enum FileType //文件类型枚举  { /// <summary>  /// 音乐  /// </summary>  Music = 1,  /// <summary>  /// 影片  /// </summary>  Movie = 2  }  } |

## 20.我们可以把应用程序的项目名称设置到App.config文件里面,可以把它定义在一个叫做appSettings的标签里面

|  |
| --- |
|  |

### 注意:数据库的设置也可以在App.config里面配置

## 21.然后我们需要定义一个类来读取配置文件的设置,把他取名为Common

|  |
| --- |
|  |

## 22.这个类生成后,我们需要把它改为公共并且是静态的.

|  |
| --- |
|  |

## 23.在使用这个类之前,我们需要添加一个引用,是程序集,叫做System.Configuration,可以在在项目文件上面点击右键->添加->引用,在弹出窗口在选择System.Configuration,然后点击确定

|  |
| --- |
|  |

## 24.成功引用后如图

|  |
| --- |
|  |

## 25.然后我们需要在这个类里面导入这个命名空间

|  |
| --- |
|  |

## 26.然后我们给这个类定义一个静态属性并且用配置管理器类获取到的设置给他赋值

|  |
| --- |
|  |

## 27.然后我们需要在窗体的构造函数里面获取这个设置,然后把它设置为窗口的标题

|  |
| --- |
|  |

## 28.编译运行程序.发现窗体的标题的确是改了

|  |
| --- |
|  |

### 以后有什么配置,我们都是用这种方式来配置

## 29.下面我们来给TreeView控件添加2个固定节点,一个是音乐一个是影片,在TreeView控件上面点击右键->编辑节点,会出现一个新窗口,如图

|  |
| --- |
|  |
|  |

## 30.可以点击2次添根,添加2个节点,一个的文本是音乐名称叫做music,另外一个文本叫影片,名称叫做movie

|  |
| --- |
|  |
|  |

## 31.点击确定后发现菜单把treeView控件给挡住了,需要在菜单上面点击右键->置于底层,这样子才能看见treeview控件的文本.

|  |
| --- |
|  |

### 注意如果了保存了播放列表,了可以需要在窗体的Form\_Load事件里面把这个列表加载进来并且设置到treeView控件里面.

## 31.我们需要给窗体类添加一个泛型列表成员,用它来保存播放从文件读取到的列表

|  |
| --- |
|  |

## 32.我们先给加载播放列表->指定文件夹菜单项添加事件响应,选择这个菜单项,然后在属性面板里面的事件选项卡里面找到Click事件双击,就会生成一个双击处理函数

|  |
| --- |
|  |

### 可以用同样的方法给指定文件和指定网络资源添加事件处理函数

|  |
| --- |
|  |

## 33.我们来给指定文件夹事件函数添加代码,我们需要用到一个叫做FolderBrowserDialog的组件,可以用代码来创建,不需要拖进winform.我们需要在选择一个文件夹后用这个路径创建一个DirectoryInfo对象,然后利用这个对象来获取文件和文件夹.注意需要先定义一个文件列表来保存播放列表.

|  |
| --- |
|  |

## 然后我们来完成添加文件到treeView的功能,注意需要根据文件的扩展名来确定文件的类型,如果是音乐添加到音乐节点,如果是视频.添加到视频节点,这里我们自己封装一个AddNode方法,然后在菜单事件代码里面调用这个方法

|  |
| --- |
| private void AddNode(FileInfo fi)  {  #region 添加文件到文件列表  //先用我们的FileModel类实例化一个对象  FileModel fileModel = new FileModel();  //生成文件的唯一标识  fileModel.FileGuid = Guid.NewGuid().ToString();  fileModel.FileName = fi.Name;//文件名称,只是用来显示  fileModel.FilePath = fi.FullName;//文件的完整路径,这个才是可以播放的文件url  if (fi.Extension == ".mp3" || fi.Extension == ".wma" || fi.Extension == ".wav")//后缀名带.  {  fileModel.FileType = FileType.Music;  }  else if (fi.Extension == ".mp4" || fi.Extension == ".wmv" || fi.Extension == ".avi" || fi.Extension == ".3gp")  {  fileModel.FileType = FileType.Movie;  }  fileModel.ResourceType = ResourceType.Local;//先把资源类型设置为本地资源  fileModel.OrderNo = this.filesList.Count + 1;  this.filesList.Add(fileModel);//添加到文件列表  #endregion  #region 添加文件到treeView  TreeNode node = new TreeNode();  node.Name = fileModel.FileGuid;  node.Text =fileModel.FileName;  node.Tag = fileModel;  if (fileModel.FileType == FileType.Music) //是音乐文件就添加到Music节点  {  treeView1.Nodes[0].Nodes.Add(node);  }  else if (fileModel.FileType == FileType.Movie) //是视频文件就添加到Movie节点  {  treeView1.Nodes[1].Nodes.Add(node);  }  #endregion  } |
|  |

### 运行程序,效果如下

|  |
| --- |
|  |
|  |
|  |

## 34.播放列表有了,那么这么实现双击媒体名称就播放媒体呢?也是需要事件的.我们选择treeView控件,在他的DoubleClick事件那么双击一下,就可以进入这个事件函数里面.我们可以封装播放函数,然后在事件里面调用封装函数,注意:下面的代码是在MediaForm类里面的.

|  |
| --- |
| private void treeView1\_DoubleClick(object sender, EventArgs e)  {  //DoPlay1();//ok,就是逻辑比较简单.  DoPlay2();  }  private void DoPlay1() //自己写的方法  {  foreach (var item in filesList)  {  if (item.FileName == treeView1.SelectedNode.Text)  {  wmpCtrl.URL = item.FilePath;  break;  }  }  }  private void DoPlay2()//还是我的方法  {  //获取选中的treeView节点  FileModel fileModel = (FileModel)treeView1.SelectedNode.Tag;  //需要先判断wmp控件的状态  if (wmpCtrl.playState != WMPLib.WMPPlayState.wmppsPaused) //如果不是暂停状态  {  wmpCtrl.URL = fileModel.FilePath;  }  else  {  wmpCtrl.Ctlcontrols.stop();  wmpCtrl.URL = fileModel.FilePath;  }  }  private void DoPlay3()//老师的方法  {  //获取选中的treeView节点  FileModel fileModel = (FileModel)treeView1.SelectedNode.Tag;  //需要先判断wmp控件的状态  if (wmpCtrl.playState != WMPLib.WMPPlayState.wmppsPaused) //如果不是暂停状态  {  wmpCtrl.URL = fileModel.FilePath;  }  wmpCtrl.Ctlcontrols.play();  } |

## 35.然后我们可以做指定文件菜单项的事件,它和指定文件夹是类似的,我们可以复制一下代码然后修改,不过不需要循环

|  |
| --- |
|  |

## 36接下来我们可以实现指定网络资源的菜单项的事件处理代码,我们需要先添加一个窗体,用来填写媒体的网络地址然后再把这个资源url添加到treeView控件的节点上面,给他取名NetworkForm

|  |
| --- |
|  |

## 37.新窗体的布局如下,目录Music是选中的:

|  |
| --- |
|  |

## 38.我们在MediaForm的指定网络资源菜单项事件处理函数里面创建一个NetworkForm窗口的实例并且显示出来

|  |
| --- |
|  |

## 39然后我们给保存按钮添加事件处理函数,在这里把控件的内容封装到一个FileModel对象中

|  |
| --- |
|  |

## 40.但是我们如何把这个数据传递给MediaForm呢?我们需要在NetworkForm里面定义一个委托事件

|  |
| --- |
|  |

## 41.然后我们在保存按钮的点击事件里面调用这个委托事件代码

|  |
| --- |
|  |

## 42.这样子还没有完,这里只是调用委托事件,但是没有实现代码,我们需要在MediaForm的指定网络资源菜单事件处理函数里面添加注册事件代码,也就是注册一个回调函数

|  |
| --- |
|  |

### 这里的Nf\_SetFileModel就是一个回调函数,注意,这个函数的参数和返回值必须和事件委托指定的类型一致

## 43.然后我们来实现这个回调函数

|  |
| --- |
|  |

### 下面的功能视频里面没有讲,是我自己摸索的

## 44,现在我们来实现文件保存菜单项的功能,把文件列表保存到文件,这里使用到FileStream和StreamWriter类,需要到System.IO命名空间,我们先是使用FileStream和StreamWriter来写入文件,

|  |
| --- |
|  |

### 结果不是我们想要的,然后我们又改为使用序列化的方法

|  |
| --- |
|  |

### 程序一运行又保存,说FileModel没有标记为可以序列化

|  |
| --- |
|  |

## 45.我们回到定义FileModel类的地方在类的上面添加可序列化注解[Serializable]

|  |
| --- |
|  |

## 46.然后我们再来测试一下,保存成功

|  |
| --- |
|  |
|  |

## 47.然后我们需要在播放器主窗体加载的时候对这个文件继续反序列化,获取文件信息并且添加到播放列表也就是treeview控件里面,我们把这段代码封装成一个函数: private void LoadPlayListFromFile(string filePath)

|  |
| --- |
| private void LoadPlayListFromFile(string filePath) {  FileStream fs = new FileStream(filePath, FileMode.Open, FileAccess.Read, FileShare.Read);  BinaryFormatter b = new BinaryFormatter();  filesList = (List<FileModel>)b.Deserialize(fs);  foreach (var item in filesList)  {  #region 添加文件到treeView  TreeNode node = new TreeNode();  node.Name = item.FileGuid;  node.Text = item.FileName;  node.Tag = item;  if (item.FileType == FileType.Music) //是音乐文件就添加到Music节点  {  treeView1.Nodes[0].Nodes.Add(node);  }  else if (item.FileType == FileType.Movie) //是视频文件就添加到Movie节点  {  treeView1.Nodes[1].Nodes.Add(node);  }  #endregion  } |

### 然后我们在Form\_Load事件里面调用这个函数

|  |
| --- |
|  |

## 48.编译运行程序,发现程序能够自动加载播放列表文件的内容到treeView节点了

|  |
| --- |
|  |

## 49.双击任意一个媒体文件,发现可以播放了

|  |
| --- |
|  |

## 50.下面我们来实现关于我们菜单项的点击事件代码,这里最好使用一个新的窗体来显示信息,在项目上面点击右键->添加->windows窗体,给窗体取名:AboutForm,然后点击添加

|  |
| --- |
|  |
|  |

## 51.把这个窗体缩小,添加几个并且显示一下版权信息即可

|  |
| --- |
|  |

## 52.然后在关于菜单项的事件函数里面添加下面的代码,随便把退出菜单项的事件代码也写了

|  |
| --- |
|  |

# 所有菜单功能完成,但是还有又右键菜单没有完成.我们下一次慢慢学习

## 老师的右键菜单是这样子的

|  |
| --- |
|  |

# 这一节的学习到这里,比较完整的代码如下

## Common.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Configuration;  namespace csharpMediaPlayer  {  public static class Common  {  //这个类用来读取App.config文件,可以获取一下公共消息,如项目名称,数据库配置等等  public static string projectName = ConfigurationManager.AppSettings["projectName"];  }  } |

## FileModel.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace csharpMediaPlayer  {  /// <summary>  /// 处理媒体文件的类  /// </summary>  [Serializable]  public class FileModel  {  public string FileGuid{ get; set; } //文件的全局唯一标识符  public string FileName{ get; set; } //文件名  public FileType FileType{ get; set; } //文件类型,是一个枚举变量  public string FilePath { get; set; }//本地路径或者网络路径都可以  public int OrderNo { get; set; } //排序编号  public ResourceType ResourceType { get; set; } //资源类型,是一个枚举变量  }  public enum ResourceType //资源类型枚举,注意c#中枚举的写法  { /// <summary>  /// 本地资源  /// </summary>  Local=1,  /// <summary>  /// 网络资源  /// </summary>  Network =2  }  public enum FileType //文件类型枚举  { /// <summary>  /// 音乐  /// </summary>  Music = 1,  /// <summary>  /// 影片  /// </summary>  Movie = 2  }  } |

## MediaForm.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using System.IO;  using System.Drawing.Drawing2D;  using System.Runtime.Serialization.Formatters.Binary;  //vs2022小技巧,ctrl+d是向下复制功能快捷键  namespace csharpMediaPlayer  {  public partial class MediaForm : Form  {  /// <summary>  /// 保存播放列表的成员  /// </summary>  List<FileModel> filesList = new List<FileModel>();  public MediaForm()  {  InitializeComponent();  //设置配置信息  this.Text = Common.projectName;//读取配置信息修改窗体标题  }  private void MediaForm\_Load(object sender, EventArgs e)  {  //在这里加载一些需要在窗体显示之前就要加载的内容如上一次退出之前保存的播放列表等等  //利用反序列化加载文件内容到文件列表  string filePath = Environment.CurrentDirectory + "\\playlist.txt";  LoadPlayListFromFile(filePath);  }  private void LoadPlayListFromFile(string filePath)  {  FileStream fs = new FileStream(filePath, FileMode.Open, FileAccess.Read, FileShare.Read);  BinaryFormatter b = new BinaryFormatter();  filesList = (List<FileModel>)b.Deserialize(fs);  foreach (var item in filesList)  {  #region 添加文件到treeView  TreeNode node = new TreeNode();  node.Name = item.FileGuid;  node.Text = item.FileName;  node.Tag = item;  if (item.FileType == FileType.Music) //是音乐文件就添加到Music节点  {  treeView1.Nodes[0].Nodes.Add(node);  }  else if (item.FileType == FileType.Movie) //是视频文件就添加到Movie节点  {  treeView1.Nodes[1].Nodes.Add(node);  }  #endregion  }  }  private void tsmi\_setFolder\_Click(object sender, EventArgs e)  {  //指定文件夹菜单项的事件处理代码  FolderBrowserDialog fbd = new FolderBrowserDialog();  if(DialogResult.OK == fbd.ShowDialog())  {  string folder = fbd.SelectedPath;  DirectoryInfo di = new DirectoryInfo(folder);//可以用一个文件夹来创建文件夹信息对象  foreach(FileInfo fi in di.GetFiles())  {  //调用封装的方法来把文件添加到treeView控件中  //在添加文件之前其实需要判断一下他的后缀名是否是媒体文件如mp3,mp4,wmv,avi等等  AddNode(fi);  }  }  }  private void AddNode(FileInfo fi)  {  #region 添加文件到文件列表  //先用我们的FileModel类实例化一个对象  FileModel fileModel = new FileModel();  //生成文件的唯一标识  fileModel.FileGuid = Guid.NewGuid().ToString();  fileModel.FileName = fi.Name;//文件名称,只是用来显示  fileModel.FilePath = fi.FullName;//文件的完整路径,这个才是可以播放的文件url  if (fi.Extension == ".mp3" || fi.Extension == ".wma" || fi.Extension == ".wav")//后缀名带.  {  fileModel.FileType = FileType.Music;  }  else if (fi.Extension == ".mp4" || fi.Extension == ".wmv" || fi.Extension == ".avi" || fi.Extension == ".3gp")  {  fileModel.FileType = FileType.Movie;  }  fileModel.ResourceType = ResourceType.Local;//先把资源类型设置为本地资源  fileModel.OrderNo = this.filesList.Count + 1;  this.filesList.Add(fileModel);//添加到文件列表  #endregion  #region 添加文件到treeView  TreeNode node = new TreeNode();  node.Name = fileModel.FileGuid;  node.Text =fileModel.FileName;  node.Tag = fileModel;  if (fileModel.FileType == FileType.Music) //是音乐文件就添加到Music节点  {  treeView1.Nodes[0].Nodes.Add(node);  }  else if (fileModel.FileType == FileType.Movie) //是视频文件就添加到Movie节点  {  treeView1.Nodes[1].Nodes.Add(node);  }  #endregion  }  private void tsmi\_setFile\_Click(object sender, EventArgs e)  {  //指定文件菜单项的事件处理代码  OpenFileDialog dlg = new OpenFileDialog();  if (DialogResult.OK == dlg.ShowDialog())  {  FileInfo fi = new FileInfo(dlg.FileName);  AddNode(fi);  }  }  private void tsmi\_setWebResource\_Click(object sender, EventArgs e)  {  //指定网络资源菜单项的事件处理代码  NetworkForm nf = new NetworkForm();//创建网络资源窗体的实例  nf.SetFileModel += Nf\_SetFileModel; //注册事件委托的处理函数  nf.ShowDialog();//显示窗体  }  //实现事件委托的处理函数,也叫做回调函数  private void Nf\_SetFileModel(FileModel model)  {  //把返回的文件模型添加到文件列表  model.OrderNo = this.filesList.Count + 1;  this.filesList.Add(model);//吧NetworkForm传递过来的文件模型对象添加到文件列表  //添加节点  TreeNode node = new TreeNode();  node.Name = model.FileGuid;  node.Text = model.FileName;  node.Tag = model;  if (model.FileType == FileType.Music) //是音乐文件就添加到Music节点  {  treeView1.Nodes[0].Nodes.Add(node);  }  else if (model.FileType == FileType.Movie) //是视频文件就添加到Movie节点  {  treeView1.Nodes[1].Nodes.Add(node);  }  }  /// <summary>  /// 实现双击播放treeview上面的媒体文件  /// </summary>  /// <param name="sender"></param>  /// <param name="e"></param>  private void treeView1\_DoubleClick(object sender, EventArgs e)  {  //DoPlay1();//ok,就是逻辑比较简单.  DoPlay2();  }  private void DoPlay1() //自己写的方法  {  foreach (var item in filesList)  {  if (item.FileName == treeView1.SelectedNode.Text)  {  wmpCtrl.URL = item.FilePath;  break;  }  }  }  private void DoPlay2()//还是我的方法  {  //获取选中的treeView节点  FileModel fileModel = (FileModel)treeView1.SelectedNode.Tag;  //需要先判断wmp控件的状态  if (wmpCtrl.playState != WMPLib.WMPPlayState.wmppsPaused) //如果不是暂停状态  {  wmpCtrl.URL = fileModel.FilePath;  }  else  {  wmpCtrl.Ctlcontrols.stop();  wmpCtrl.URL = fileModel.FilePath;  }  }  private void DoPlay3()//老师的方法  {  //获取选中的treeView节点  FileModel fileModel = (FileModel)treeView1.SelectedNode.Tag;  //需要先判断wmp控件的状态  if (wmpCtrl.playState != WMPLib.WMPPlayState.wmppsPaused) //如果不是暂停状态  {  wmpCtrl.URL = fileModel.FilePath;  }  wmpCtrl.Ctlcontrols.play();  }  //把文件列表保存到一个文本文件中  private void tsmi\_saveList\_Click(object sender, EventArgs e)  {  if(this.filesList.Count ==0)  {  MessageBox.Show("播放列表为空,请添加歌曲到播放列表");  return;  }  //利用序列化把文件列表保存到文件  string savePath = Environment.CurrentDirectory + "\\playlist.txt";  FileStream fs = new FileStream(savePath, FileMode.Create);  BinaryFormatter b = new BinaryFormatter();  b.Serialize(fs,filesList);  fs.Close();  MessageBox.Show("播放列表已经保存");  }  private void tsmi\_aboutUs\_Click(object sender, EventArgs e)  {  AboutForm aboutForm = new AboutForm();  aboutForm.ShowDialog();  }  private void tsmi\_exit\_Click(object sender, EventArgs e)  {  this.Close();  }  }  } |

## NetworkForm.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace csharpMediaPlayer  {  public partial class NetworkForm : Form  {  public NetworkForm()  {  InitializeComponent();  }  /// <summary>  /// 定义一个委托事件,他是一种数据类型,相当于在内存里面指定一个存放函数的内存地址  /// </summary>  /// <param name="sender"></param>  /// <param name="e"></param>  public event Action<FileModel> SetFileModel;  private void btnSave\_Click(object sender, EventArgs e)  {  //传递数据给MediaForm并且关闭网络资源对话框  //需要先把数据封装到FileModel对象中  FileModel fileModel = new FileModel();    //生成文件的唯一标识  fileModel.FileGuid = Guid.NewGuid().ToString();  fileModel.FileName = tbFileName.Text.Trim();//文件名称,只是用来显示  fileModel.FilePath =tbURL.Text.Trim();//文件的完整路径,这个才是可以播放的文件url  fileModel.FileType = this.rbnMusic.Checked ? FileType.Music : FileType.Movie;  fileModel.ResourceType = ResourceType.Network;//先把资源类型设置为本地资源  SetFileModel(fileModel);//需要使用委托事件的地方  this.Close();  }  private void btnCancel\_Click(object sender, EventArgs e)  {  this.Close();  }  }  } |

## Program.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace csharpMediaPlayer  {  internal static class Program  {  /// <summary>  /// 应用程序的主入口点。  /// </summary>  [STAThread]  static void Main()  {  Application.EnableVisualStyles();  Application.SetCompatibleTextRenderingDefault(false);  Application.Run(new MediaForm());  }  }  } |

### 注意:这些是业务逻辑层代码,界面的代码只需要注意布局就行了