**课程论文**

**课程： 多媒体信息技术**

**论文题目： 多媒体播放网站设计**

**组长学号：**

**组长姓名：**

**组员学号：**

**组员姓名：**

**2021年 0 6 月**

**多媒体信息技术课程设计 Project**

**——多媒体播放网站设计**

**小组成员：**

【概要】本项目中我们基于多媒体信息技术的相关知识，使用web编程技术、流媒体技术等技术，利用Windows Media Player的相关插件，搭建了一个简单的多媒体播放网站。网站支持流媒体播放，支持音视频播放、暂停、快进、快退等基础功能，支持查看各种格式的图片，同时支持对图片进行格式转换和大小调整，将视频转换为音频等功能。

【关键词】多媒体播放器 网站设计 流媒体技术 格式转换

# **1  概述**

## **1.1 项目背景**

多媒体包括文本、图形、图像、音频、动画、视频等多种类型的信息媒体。多媒体技术是指能够同时获取、处理、编辑、存储和回放两种或两种以上不同类型信息媒体的技术。

目前多媒体技术发展迅速，各大公司为了多媒体技术标准竞争激烈。多媒体技术发展改变了计算机的使用领域，使得计算机由办公室、实验室中的专属用品变成了现代社会的常用工具，在工业生产管理、学校教育、商业广告、家庭娱乐等方面被广泛应用。

在学习了《多媒体信息技术》理论知识后，将知识与实践相结合对于学习多媒体的知识有着重要意义。

本项目结合《多媒体信息技术》课程的理论知识，实现了一个多媒体播放器网站，支持绝大多数格式媒体的播放、停止、快进、快退等功能，支持媒体的编辑处理，支持internetTV、在线电台节目播放。此外还搭建了一个简单的流媒体服务器，实现将本地视频转换为流媒体格式并使用rtmp协议在网络上进行发布。

## **1.2 相关技术**

### 1.2.1 web前端编程技术

* **Django 框架**：django是一个基于python语言的web框架，可以让我们专注于软件本身的设计，而不用考虑web应用的底层轮子的搭建。django是免费且开源的。
* **Bootstrap 框架**：Bootstrap 是由 Twitter 的 Mark Otto 和 Jacob Thornton 开发的。Bootstrap 是 2011 年八月在 GitHub 上发布的开源产品。[1]提供丰富的即插即用的前端组件元素，用于前端页面的设计。
* **HTML、CSS、Javascript**：

HTML：做基本的页面设计。

html技术多媒体课程中介绍的超文本与超媒体技术的一种。

主体框架是

<!doctype html>  
<html lang="zh-CN">  
<head>  
 <meta charset="UTF-8">  
 <title>MMplayer</title>  
</head>  
<body>  
 在这里面写我们网站的页面内容  
</body>  
</html>

使用<object>标签插入媒体播放器控件。

CSS（cascading style sheets）：是一种用来表现[HTML](https://baike.baidu.com/item/HTML)（[标准通用标记语言](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E9%80%9A%E7%94%A8%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80/6805073)的一个应用）或[XML](https://baike.baidu.com/item/XML)（标准通用标记语言的一个子集）等文件样式的计算机语言。CSS不仅可以静态地修饰网页，还可以配合各种脚本语言动态地对网页各元素进行格式化。[2]

Javascript：使用脚本语言进行前后端的交互以及复杂的前端界面控制。

* **json、AJAX、Jquery：**

json作为一种文件格式，用于前后端的数据交互

Ajax用于前端向后端发起HTTP请求

Jquery：是一个javascript库，极大的简化了javascript编程。

### 1.2.2 python库

**ffmpeg:** FFmpeg是一套可以用来记录、转换数字音频、视频，并能将其转化为流的开源计算机程序。它包括了领先的音/视频编码库libavcodec等。FFmpeg有非常强大的功能包括[视频采集](https://baike.so.com/doc/5896600-6109494.html)功能、[视频格式转换](https://baike.so.com/doc/227491-240644.html)、视频抓图、给视频加水印等。

**PIL:** 对于图像识别，大量的工作在于图像的处理，处理效果好，那么才能很好地识别，因此，良好的图像处理是识别的基础。在Python中，有一个优秀的图像处理框架，就是PIL库。

**os:** os库是Python标准库，包含几百个函数,常用路径操作、进程管理、环境参数等几类。os.path子库以path为入口，用于操作和处理文件路径。

**subprocess:** subprocess 模块允许我们启动一个新进程，并连接到它们的输入/输出/错误管道，从而获取返回值。

**scipy:** scipy是一个用于数学、科学、工程领域的常用软件包，可以处理插值、积分、优化、图像处理、常微分方程数值解的求解、信号处理等问题。它用于有效计算Numpy矩阵，使Numpy和Scipy协同工作，高效解决问题。

### 1.2.3 Windows Media Player SDK

* **Windows Media Player SDK**

Windows Media Player SDK 是 Microsoft Windows SDK 的组件之一, Microsoft Windows Media Player（下文称wmp） 软件开发工具包 (SDK) 提供了一些信息和工具，可用于自定义 Windows Media Player 和 Windows Media Player Mobile 并使用 Windows Media Player ActiveX 控件和 Windows Media Player Mobile ActiveX 控件。

Windows Media Player SDK支持将wmp嵌入web浏览器中的html,支持的浏览器有 Microsoft Internet Explorer、Firefox 和 Netscape Navigator 版本7.1 浏览器。

本项目使用嵌入HTML中的wmp控件并且调用sdk中的函数对wmp进行播放控制。

* **Windows图元文件：**

包括.wax,.wvx,.wmx,.asx文件扩展名的图元文件。

图元文件是包含有关其他文件的信息的文件。 图元文件可用于列出要按顺序播放的一组媒体内容文件。Windows Media 图元文件播放列表（在此参考文档中简称为播放列表）是 Microsoft Windows Media 技术最强大的功能之一。 播放列表允许控制和自定义媒体内容[3]。

本项目使用了.asx文件来储存传统媒体的url和流媒体的源。

* **Windows播放列表：**

windows播放列表是文件扩展名位.wpl的文件，可以用来定义动态变化的和静态的媒体播放列表。

本项目使用.wpl制作媒体播放列表允许人在浏览器直接按顺序切换媒体列表中的媒体进行播放。

### 1.2.4 VLC media player WebPlugin

  VLC media player与wmp一样是一个多媒体播放器，支持众多的格式。它由非营利的组织VideoLan制作，最初是一个来自法国的一个学生做的项目，现在已经在全世界20多个国家被使用。[4]，该播放器对流媒体格式的文件支持比较全面。

### 1.2.5 流媒体技术

* **nginx：**

反向代理（Reverse Proxy）方式是指以代理服务器来接受Internet上的连接请求，然后将请求转发给内部网络上的服务器[5]；本项目中使用nginx加上nginx的rtmp模块来搭建流媒体服务器。

* **ffmpeg**：

FFmpeg 本身是一个庞大的项目，包含许多组件和库文件，最常用的是它的命令行工具。本项目中使用了其格式转换部分的命令行。（后面会介绍使用的命令行。）

* **rtmp：**

RTMP是Real Time Messaging Protocol（实时消息传输协议）的首字母缩写。该协议基于TCP，是一个协议族，包括RTMP基本协议及RTMPT/RTMPS/RTMPE等多种变种。RTMP是一种设计用来进行实时数据通信的网络协议，主要用来在Flash/AIR平台和支持RTMP协议的流媒体/交互服务器之间进行音视频和数据通信。支持该协议的软件包括Adobe Media Server/Ultrant Media Server/red5等。RTMP与HTTP一样，都属于TCP/IP四层模型的应用层[6]。

* **http协议在流媒体技术上的应用：**

我们采用Apple的方案：HLS（HTTP Live Streaming） 协议 +.m3u8格式文件，将m3u8格式的文件写进我们的图元文件(.asx)中，并使用web浏览器进行访问。

## **1.3 开发环境**

pycharm 专业版 64位

ie 浏览器 版本11以上

windows 10 操作系统（自带Windows media player）

vlc media player 3.0.14

# **2  需求分析**

## **2.1 功能需求**

### 2.1.1 参考老师给的pj的选题文档：

编写一个多功能媒体播放器，支持的媒体格式越多越好（wav、midi、mp3、cd、avi、wmv、asf 等等）。可以采用第三方的控件或开发包，实现播放、停止、暂停、快进、快退等功能，将程序实现方法总结成论文。

### 2.1.2 我们给出的需求

一、播放功能

1.支持尽可能多的媒体播放格式。

2.播放、停止、暂停、快进功能。

3.支持在线媒体播放。

4.支持播放列表中的媒体曲目切换。

5.支持流媒体的播放（广播电台、电视直播）。

二、格式转换功能

1.支持图片格式转换。

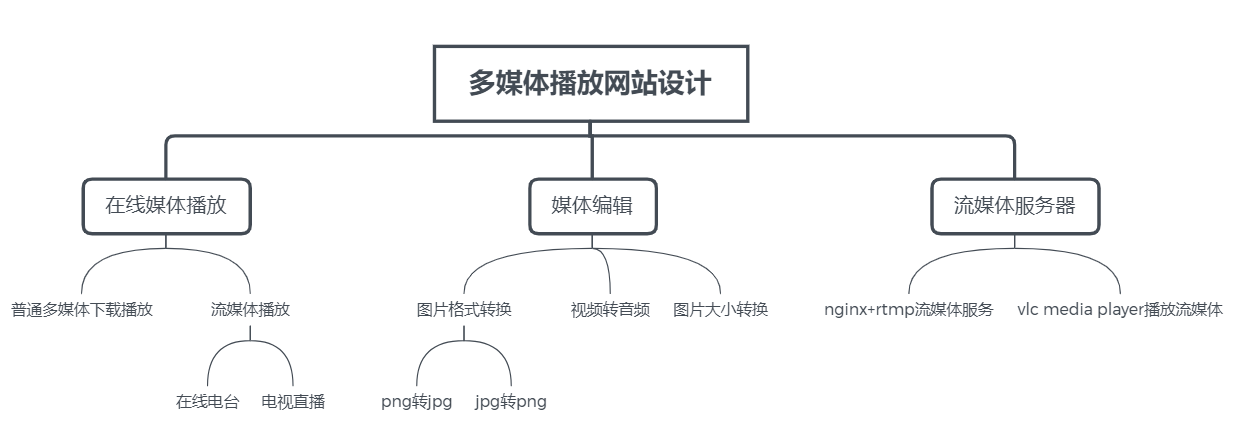
2.支持视频转换为音频。

3.支持图片大小调整。

三、流媒体服务器

1. 支持将本地媒体文件推送到流媒体服务器进行直播。

2. 支持使用流媒体播放器对我们的直播媒体数据进行播放。



*图：功能概览*

## **2.2 数据需求**

**2.2.1我们支持的格式：**

1.图片格式：JPG、PNG、ICO、BMP、GIF、TIF、PCX、TGA.

2.音频格式：APE、WMA、M4A、MP2、OGG、AAC、FLAC、AMR、MMF、M4R、WAV、WV、MP3、MIDI.

3.视频格式：3GP、AVI、MKV、MP4、MPG、WMV

4.流媒体支持在线电台和网络电视。

.m3u8 文件（http协议）和（rtmp协议），

**2.2.2 重要图片格式简介**

* **png**

可移植网络图形格式(PNG)名称来源于非官方的"PNG's Not GIF"，是一种位图文件存储格式。PNG用来存储灰度图像时，灰度图像的深度可多到16位，存储彩色图像时，彩色图像的深度可多到48位，并且还可存储多到16位的α通道数据。

* **jpeg**

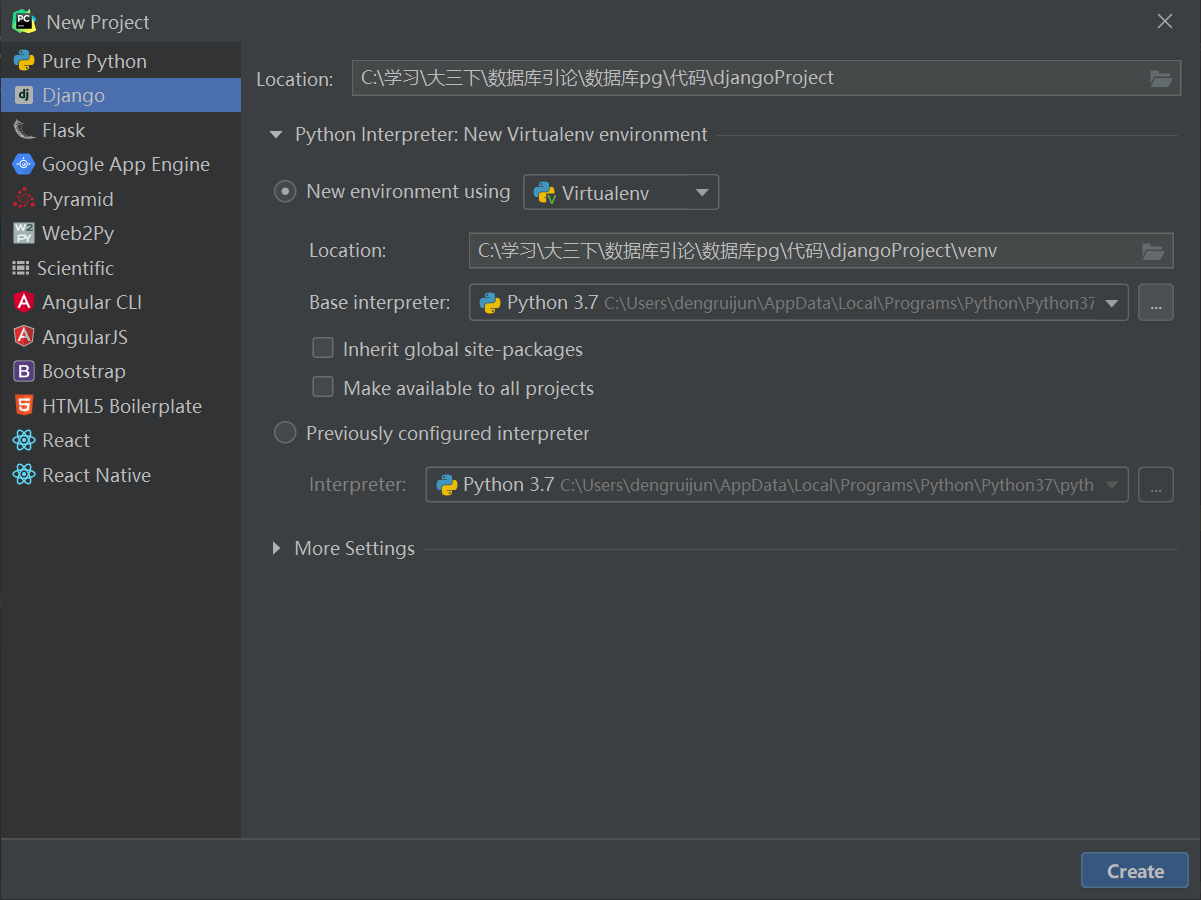
JPEG图片以 24 位颜色存储单个位图。JPEG图片仅存储RGB三个通道的内容，在转换时需要考虑去掉α通道数据。JPEG 是与平台无关的格式，支持最高级别的压缩，是最常见的图片格式，这种压缩是有损耗的。

# **3  项目实现方法**

## **3.1 网站搭建模块**

### 3.1.1 django搭建网站

* 如下图所示pycharm专业版支持直接创建django项目。我们使用pycharm编辑器进行项目的开发。项目创建之后自动生成了一系列网站所需要的基础文件。我们在这些基础上进行我们应用的开发。



* 初始目录结构

创建后的项目目录如下：

网站代码/

manage.py

mysite/

\_\_init\_\_.py

settings.py

urls.py

asgi.py

wsgi.py

这里面的文件都是在django框架下搭建网站所需要的一些基本的文件，其中我主要使用到的就是urls.py，这个文件，这个文件负责整个项目的路由，当前端点击一个url，都需要从这里开始进行url的匹配，也就是这个文件负责整个项目顶层的路由。在本项目中，它的功能就是接受访问我们的多媒体播放器应用的url并分配给多媒体播放器应用一级的路由设置，之后就由多媒体播放器应用一级的路由设置进行应用内的路由。

* 创建MMplayer web应用
* 首先使用py manage.py startapp MMplayer指令在“网站代码”这个项目中创建我们的web应用MMplayer。一个项目可以拥有很多应用，但我们这个项目只需要一个MMplayer应用。
* 指令执行完毕在项目中自动生成了一个Mmplayer目录和目录下的一些文件：

网站代码/

manage.py

mysite/

\_\_init\_\_.py

settings.py

urls.py

asgi.py

wsgi.py

MMplayer/

\_\_init\_\_.py

admin.py

apps.py

migrations/

\_\_init\_\_.py

models.py

tests.py

urls.py

views.py

我们在views.py 文件中编写我们网站前后端交互过程中的后端代码的视图函数部分。

在urls.py文件中制定我们MMplayer应用的路由规则。

url的路由规则是：前端访问的一个url地址需要匹配urls.py文件中的某一个url样式，并且根据匹配到的样式，前端的请求被分发到views.py中的后端处理函数中进行处理。

* MMplayer/urls.py文件内容:
* 截图：



* 解释：

以上是我们软件所使用的所有的url模式，可以看到每个url都负责一个具体的功能，前端访问某个url都会匹配到对应的一条并后端被认为是来自特定功能的请求然后把请求的信息分发到对应的函数里去。

### 3.1.2 使用HTML、css设计界面

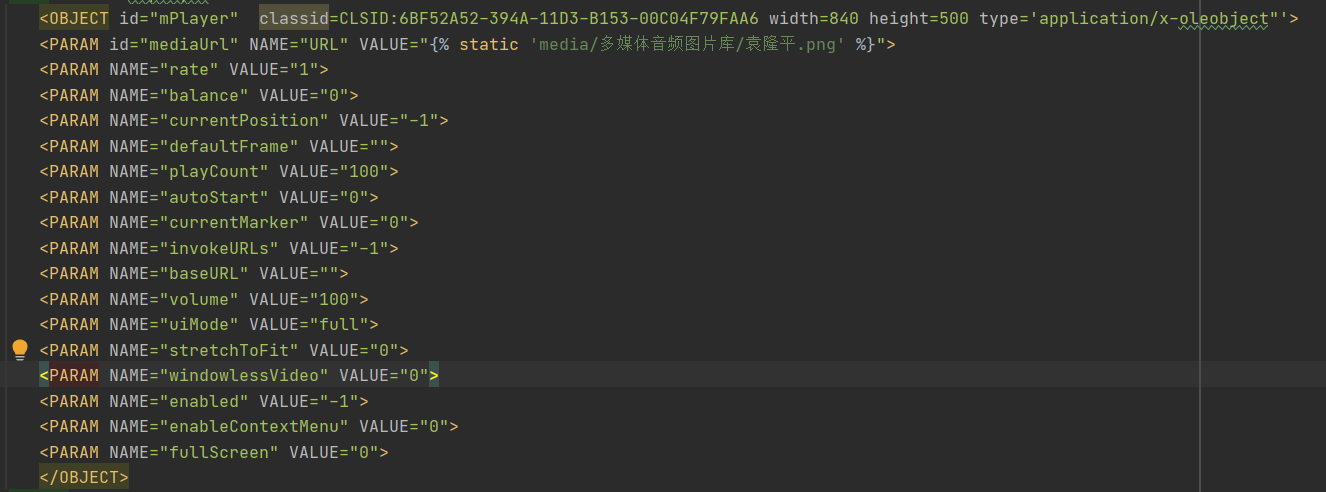
我们的网站的界面采用html语言来编写。共涉及两个主界面：index.html和formatChange.html分别负责多媒体播放器的界面和媒体编辑的界面。这两者都继承了base.html的模板

我们使用了<nav>、<button>、<h1>、<a>、<ul>、<form>、<input>、<lable>等标签来构建我们的页面的基本元素，使用标签的“style”属性进行css的设计。

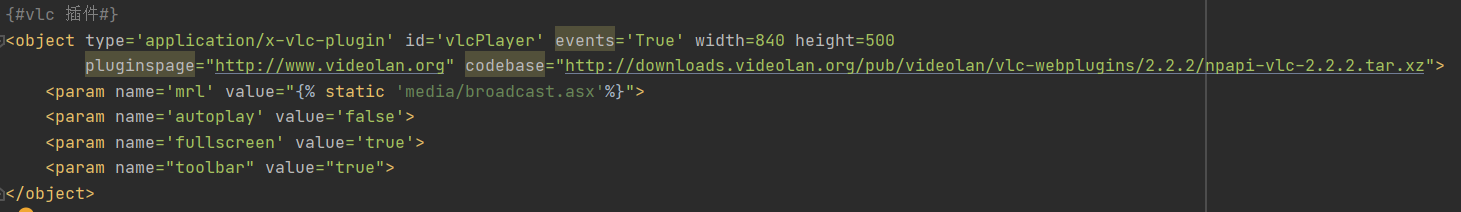
* 插入播放器插件

html5 提供了<video>和<audio>两个标签来支持多媒体视频音频数据，但此外我们还使用了<object>。<audio>支持三种音频格式：audio支持三种音频格式ogg、mp3、wav，<video>支持三种视频格式：ogg、mp4、webm[7]，支持的格式太少并不能满足我们多媒体播放器的要求。所以我们决定使用<object>标签嵌入了两个播放器插件。

* + windows media player控件的部分代码：

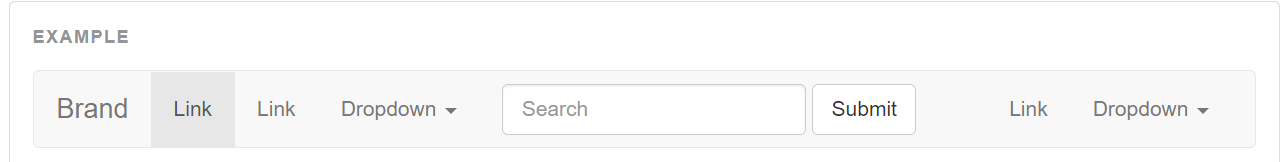


* + vlc media player控件的部分代码：



### 3.1.3 使用bootstrap框架基本元素

 在这次实验中，我们使用了bootstrap的一些基础组件，如导航条、栅格布局、按钮样式等等。[8]



上图是bootstrap提供的一个导航条样例，我们使用了这个元素并且做了子元素的剪除和样式的修改。下面是我们的导航条：



### 3.1.4 前后端交互

我们的前后端使用http协议进行数据和控制信息的传输。

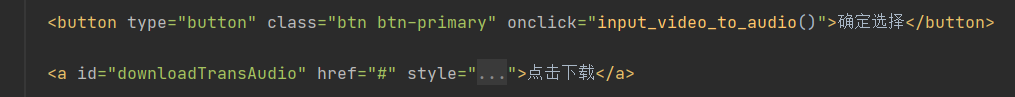
* 在前端，一种方式是对html标签绑定一个url连接，当人点击改标签对应的组件比如一个按钮，我们的浏览器即对这个url发其一个http请求。

另一种方式是采用ajax生成并向后端发送一个http请求，这种方式主要用于提交表单数据。当人点击一个按钮，按钮并不直接绑定连接，而是这个点击事件触发一个javascript函数，该函数收集客户在表单里填写的数据，比如客户从本地上传的图片，然后向后端的url发其一个http请求，使用POST方法把数据传输到后端。

* 标签直接绑定连接的代码：
  + 点击该链接，会跳转到格式转换页面

<li><a href="{% url 'formatChange' %}">格式转换</a></li>

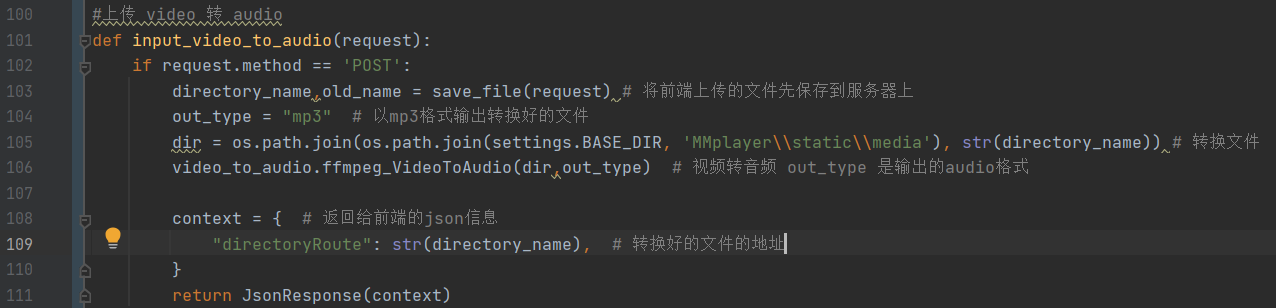
* ajax发送请求：
  + 前端的按钮触发：两个按钮，一个是上传文件的按钮，另一个<a>标签是后端修改后绑定了转换好的文件的下载链接的按钮，只有后端绑定了下载链接（即转换成功）才会显示



* + javascript函数，收集表单内容（上传的文件），向后端发起http请求，若成功则提示“ok“消息，并且把转换好的文件的下载链接绑定到下载按钮上，供用户点击下载。



* 在后端，根据前端的http请求对应的url，我们的url.py对url进行匹配，如果匹配成功，则会把该请求发送给后端负责具体业务的一个函数来处理这个请求。我们的业务处理逻辑都在views.py中
* 后端的处理：该函数接受前端的http请求，然后调用转换函数进行转换，返回前端转换好的文件的路径



## **3.2 多媒体播放器模块**

#### 3.2.1 windows media player网页插件的使用

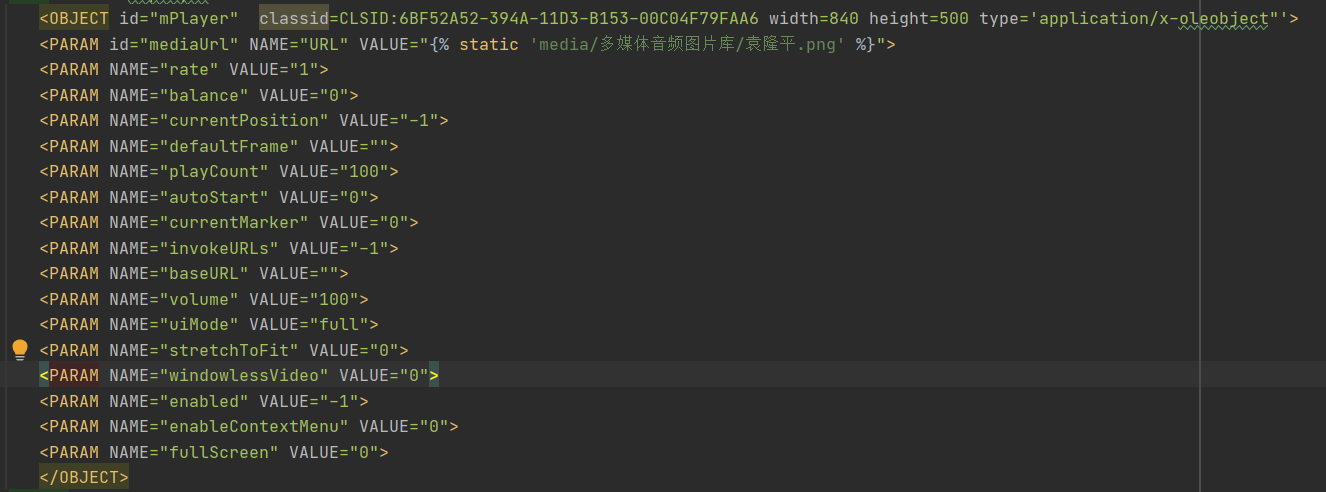
* 流程图



* player对象

我们插入浏览器中的wmp控件，是windows media player SDK中的player对象，它是Windows Media Player控件中的根对象。支持许多属性、方法和事件。

* 在html中插入控件的代码



Windows Media Player 使用 PARAM 元素为控件定义特定的启动条件，PARAM 元素嵌入在 OBJECT 元素中[3]。 也就是player对象以及player对象所使用的其他对象的属性值。

如“autoStart”的值为“0”，表示不会自动播放。“URL”属性是指播放资源的地址。

param标签在不同浏览器上支持的“name”的值不同。具体可以参考[9]

* control对象

Controls 对象提供一种方法，用于操作媒体的播放。[3]

* + control对象提供以下方法：

| 方法 | 描述 |
| --- | --- |
| [fastForward](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-fastforward) | 前进方向开始播放媒体项。 |
| [fastReverse](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-fastreverse) | 按相反方向开始快速播放媒体项。 |
| [getAudioLanguageDescription](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-getaudiolanguagedescription) | 检索与指定的基于1的索引对应的音频语言的说明。 |
| [getAudioLanguageID](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-getaudiolanguageid) | 检索指定音频语言索引的 LCID。 |
| [getLanguageName](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-getlanguagename) | 检索具有指定 LCID 的音频语言的名称。 |
| next | 将当前项设置为播放列表中的下一项。 |
| [pause](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-pause) | 暂停播放媒体项。 |
| [play](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-play) | 使媒体项开始播放。 |
| [playItem](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-playitem) | 导致当前媒体项开始播放，或恢复暂停项的播放。 |
| [previous](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-previous) | 将当前项设置为播放列表中的上一项。 |
| step | 使当前视频媒体项在下一帧冻结播放。 |
| [stop](https://docs.microsoft.com/zh-cn/windows/win32/wmp/controls-stop) | 停止播放媒体项。 |

[3]

* playList.wpl文件

<?wpl version = "1.0"?>  
<smil>  
 <head>  
 <title> its a playlist of mmplayer </title>  
 </head>  
 <body>  
 <seq>  
 <media src = "audio/GreateLullaby.mp3"></media>  
 <media src = "audio/Lemon.mid"></media>  
 <media src = "img.jpg"></media>  
 </seq>  
 </body>  
</smil>

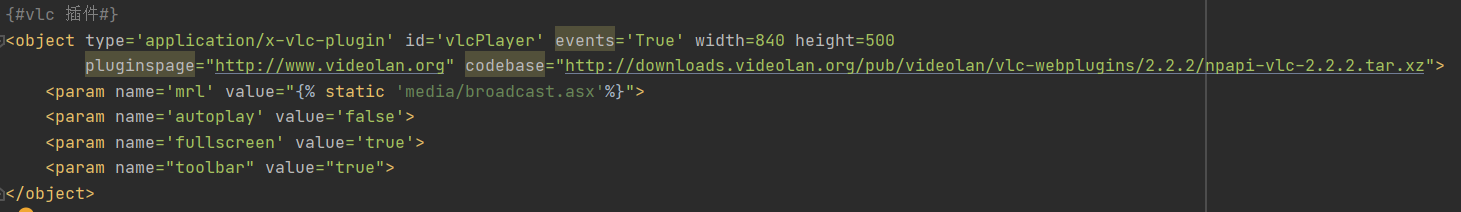
* + **功能：**

该文件用于存储wmp播放器的播放列表。这里共有3个多媒体文件。人在wmp控件中可以直接点击下一个或上一个按钮进行媒体的切换。

#### 3.2.2 VLC media player 网页插件的使用

* **在html中插入VLC media palyer**

与wmp一样，通过<object>标签插入，通过<param>标签控制启动条件。



* **参数说明：**

name属性等于“mrl” 的<param>对应的value属性的值是待播放的媒体资源的地址。具体的可选参数可以参考[10]

* **vlc对象**

vlc插件和wmp插件一样，我们插入html中的是它的跟对象vlc对象，类似于wmp中的player对象。

* **vlc对象的控制**

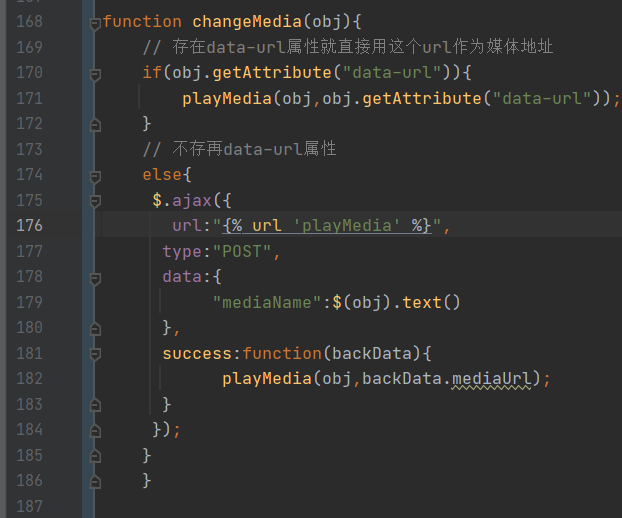
vlc插件还提供了许多其他功能的对象供开发者使用。类似于wmp的control对象，vlc也提供了操作媒体播放的对象。但不同于wmp，vlc中的这些方法是分散在各种对象中的，如audio对象提供ToggleMute静音/取消静音操作，playlist对象的stop（）方法停止播放。

#### 3.2.3 曲目播放

* **说明：**

当我们点击一个曲目时，会触发“changeMedia（）“函数，该函数向后端以媒体的名字查找媒体的路径地址，当后端返回了媒体播放地址的时候，再调用playMedia()函数使用相应的播放器进行播放。

* + changeMedia函数



* + playMedia函数



## **3.3 媒体编辑处理模块**

对于我们的多媒体文件处理，我们提供了一些基础的方法，将用户输入的多媒体文件按照其需求进行处理，为了方便，我们提供批量处理的方法。其中调用了一现成的函数库如PIL库、os库、scipy库、subprocess库等中的方法实现功能。

* 流程图



#### 3.3.1 图片格式转换

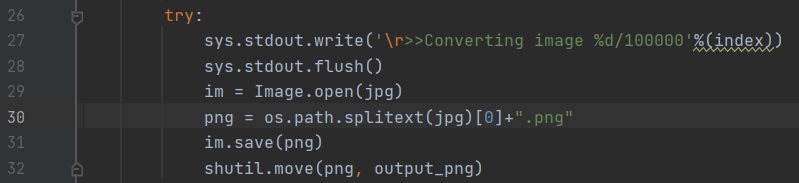
* **功能：**

实现对最常见的图片格式.jpg/.jpeg和.png进行转化。

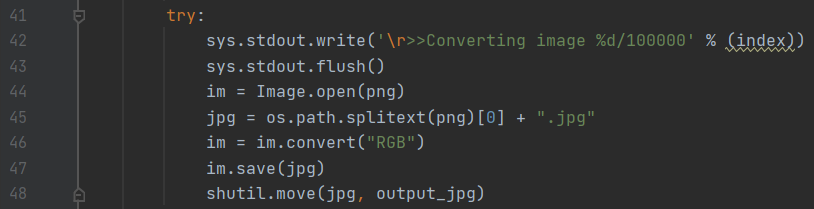
* **实现思路**

实现的方法是调用PIL库中的Image方法，通过os.path.join() 将分离的部分合成一个整体，将读取的JPEG格式和PNG格式图片分别存入对应列表进行处理，处理方法是用Image.open()直接读取该路径指向的图片后，用os.path.splitext()将文件名和扩展名分开，并更换新的扩展名进行保存，然后移动到对应的位置。用到的os库和PIL库中的方法会自动的将图片格式转换成我们刚才更换的格式。

* **关键代码：**
* jpg转png：



* png转jpg：



#### 3.3.2 视频转音频

* **motivation：**

很多情况下，我们不需要视频的图像信息，只需要其音频信息。比如，希望内存占用较少的播放歌曲，我们往往不需要播放诸如MP4文件的视频文件，而仅仅播放如WMA格式的音频文件。或者无法观看视频文件，但是想要得到其中的音频内容的情况。比如想得知开会内容，而开会时的视频信息用处不太大的情况。这些情况都需要将视频转换为音频，一方面降低了内存和网络流量的消耗，另外一方面也适应很多不需要观看图像信息的情况。我们提供了可以使几乎所有视频格式同时批量地转换为指定的音频格式文件的方法。

* **实现思路：**

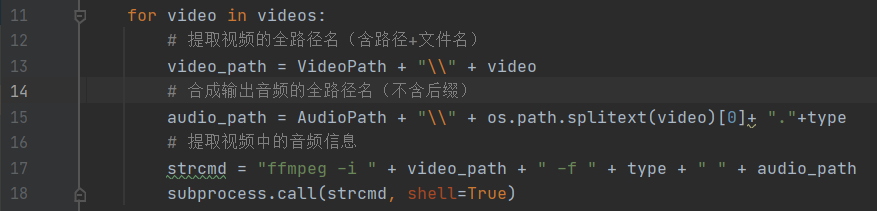
该方法的实现需要使用到scipy，它是一个用于数学、科学、工程领域的常用软件包，可以处理插值、积分、优化、图像处理、常微分方程数值解的求解、信号处理等问题。其中sicpy.io负责数据的输入输出。

还需要使用subprocess 模块，它允许我们启动一个新进程，并连接到它们的输入/输出/错误管道，从而获取返回值。我们使用了subprocess.call()方法，其作用是执行一些命令行的命令，会开启一个子进程去执行，并且等待子进程结束才继续执行其他的子程序，使用起来非常方便。

最重要的是使用了FFmpeg，它是一套可以用来记录、转换数字音频、视频，并能将其转化为流的开源计算机程序。采用LGPL或GPL许可证。它提供了录制、转换以及流化音视频的完整解决方案，包含了非常先进的音频/视频编解码库libavcode，libavcodec里很多code都是从头开发的，保证高可移植性和编解码质量。

具体实现是通过os.path.join() 将分离的部分合成一个整体，将读取的视频文件读入，然后提取视频的全路径名，确定合成输出音频的全路径名（含后缀），然后根据传入的指定转换类型type生成shell操作指令，使用subprocess.call()运行该指令即可。

* **关键代码**



#### 3.3.3 图片大小修改

* **motivation：**

图片的分辨率是图片很重要的一项参数，我们往往会遇到希望将图片调整大小的情况，比如在其他地方有大小格式需求，或者对于用户有自己的视觉偏好的情况都是非常好用的。

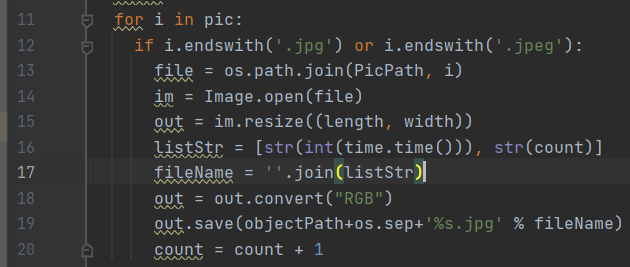
* **功能：**

我们提供的方法可以使得输入的JPEG图片按照输入的长度和宽度自动调节，生成JEPG图片并存储在对应的输出文件夹之中。

* **实现方法**

实现的方法是依赖PIL库和os库，使用os.path.join() 将分离的部分合成一个整体，存储读取的文件，然后对以.jpg和.jpeg结尾的文件，用Image.open()直接读取该路径指向的图片后，用image.resize()根据输入的int类型的参数length和width进行格式的调整，根据时间函数生成名字（防止名称重复）后将文件通道调整为RGB三通道，最后以JPEG格式保存下来。

* **关键代码：**

****

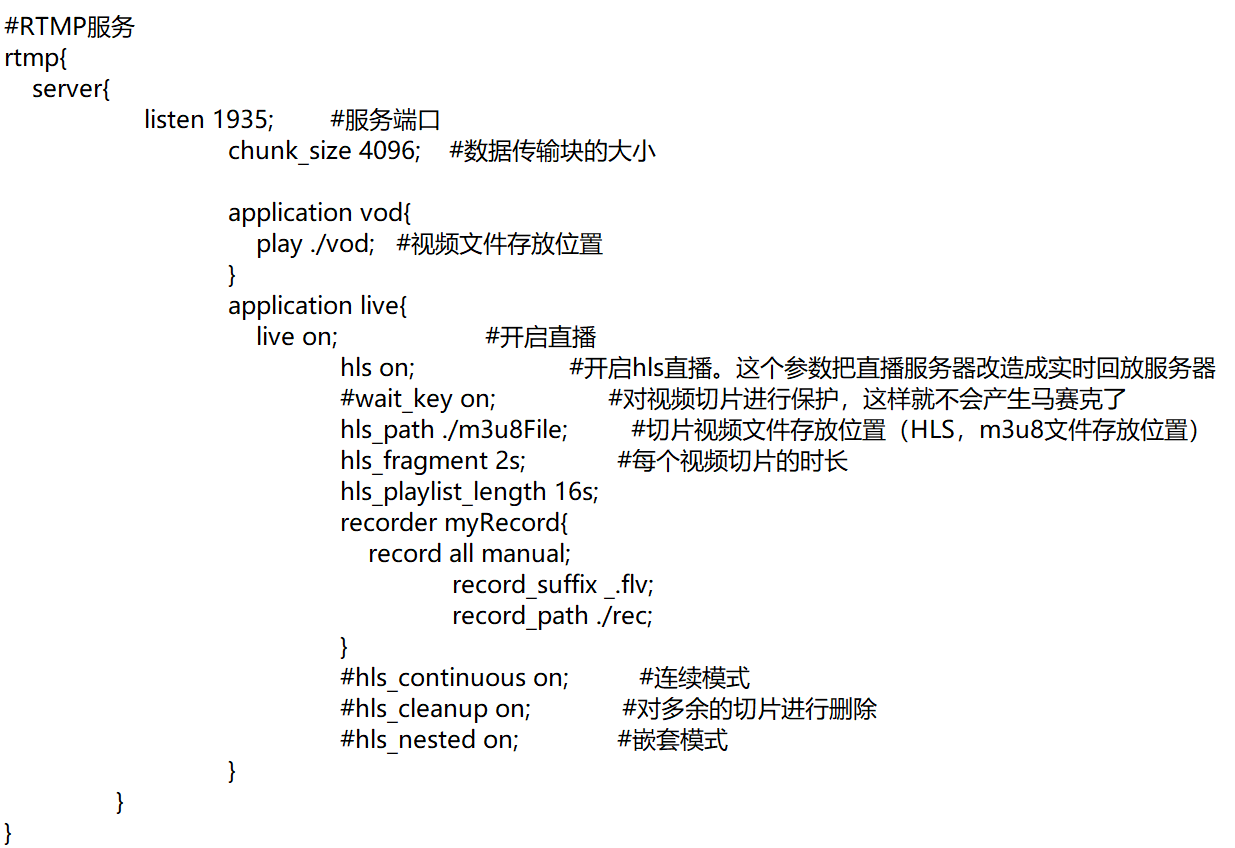
## **3.4 流媒体服务器与在线电台模块**

#### 3.4.1 nginx+rtmp+ffmpeg 搭建流媒体服务器

* 流程图
* 
* **功能**

我们利用nginx的反向代理功能和nginx的rtmp模块将我们的pc机变成一台流媒体服务器。

* **nginx 的rtmp的配置：**

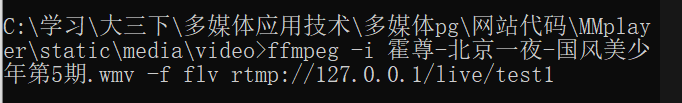


[11]

* **推流步骤：**
* 启动nginx：

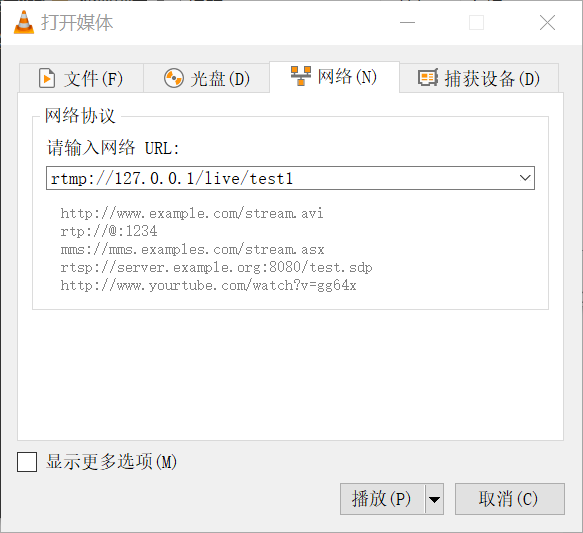


* 使用ffmepg推流：



* **使用VLC media player拉流：**

输入我们输入我们直播的地址



* **结果展示**

出现了我们推上去的视频：



#### 3.4.2 在线电台

在中国之声官网上，提供了许多电台的地址。采用的都是Apple的http协议传送流媒体数据的方案，即：HLS（HTTP Live Streaming） 协议 +.m3u8格式文件。

另外我还找到了使用rtmp协议进行传送流媒体数据的一些电视直播网站。

* **部分流文件地址：（broadcast.asx）**



* 使用图元文件记录流文件信息
  + **说明：**

图元文件是记录其他文件信息的文件，这里记录了title，连接地址（href）

.asx是windows的图元文件的格式，

我们将所有的电台地址、电视直播地址都储存在.asx文件中，供服务器访问。

我们将这个文件的地址作为vlc控件中的资源url，这个媒体播放器就会获取对应的流媒体资源，不断接受流媒体文件的分片，并播放给客户端用户。

* + **broadcast.asx 部分文件内容（广播电台）**

<ASX version="3.0">  
 <TITLE>MMplayer</TITLE>  
 <ENTRY>  
 <TITLE>中国之声</TITLE>  
 <REF HREF = "http://ngcdn001.cnr.cn/live/zgzs/index.m3u8" />  
 </ENTRY>

</ASX>

* + **tvshow.asx文件内容：（电视节目）**

<ASX version="3.0">  
 <TITLE>MMplayer</TITLE>  
 <ENTRY>  
 <TITLE>湖南卫视</TITLE>  
 <REF HREF = "rtmp://58.200.131.2:1935/livetv/hunantv" />  
 </ENTRY>  
 <ENTRY>  
 <TITLE>东方卫视</TITLE>  
 <REF HREF = "rtmp://58.200.131.2:1935/livetv/dftv" />  
 </ENTRY>  
</ASX>

# **4 项目实现效果**

### 4.1 多媒体播放

#### 4.1.1 界面展示

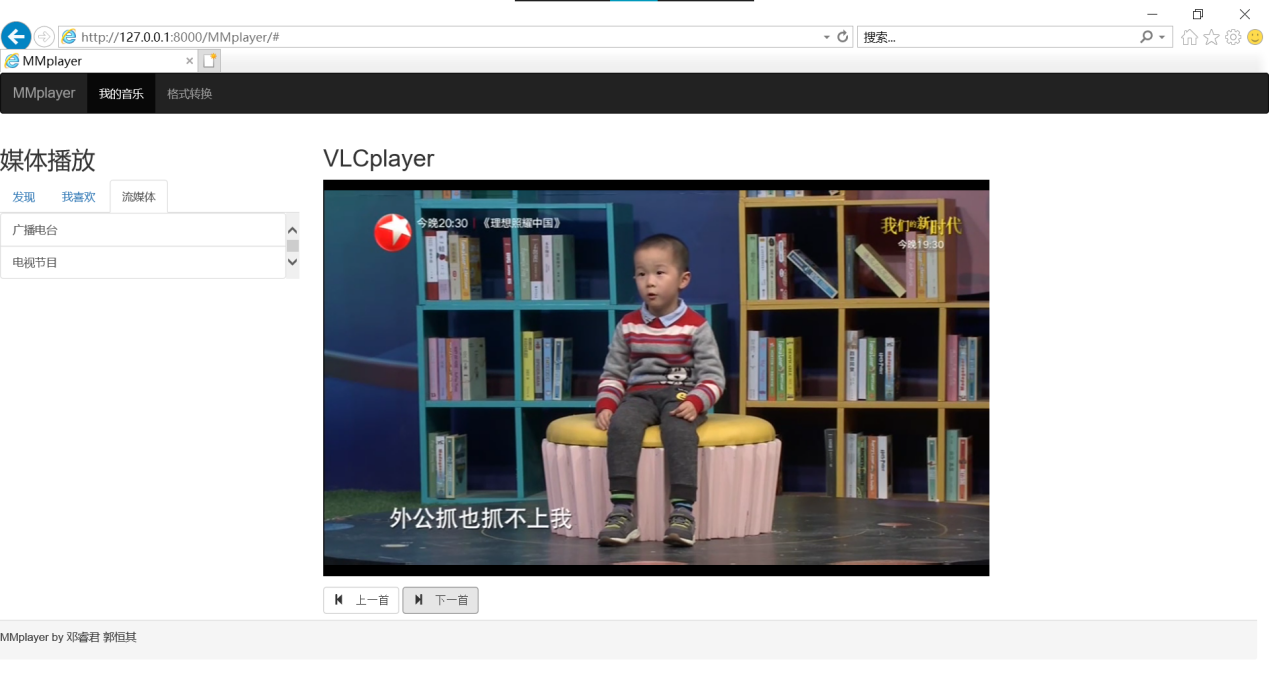
* **多媒体播放页面**
* **页面说明：**

说明左边包括一个列表，列表包括三列，“**发现**”是指服务器端存储的音视频，“**我喜欢**”是信息存储在playList.wpl 即windows media player 的播放列表文件里面的音视频，“**流媒体**”提供了广播电台、电视直播的播放功能。

右边包括两个播放器，MMplayer和VLCplayer，前者支持播放“发现”，“我喜欢”两栏的媒体，后者支持播放“流媒体”一栏的内容。当点击“发现”、“我喜欢”时，出现的是MMplayer，低昂点击“流媒体”时，出现的是VLC player.

* **页面截图：**





### 4.2 媒体编辑处理

#### 4.2.1 界面展示



* **界面说明：**

这是提供的媒体编辑的三项功能：“**图片格式转换**”，“**视频转音频**”和“**图片大小修改**”。

操作方法是，点击浏览，选择本地文件并上传，然后服务器端转换好后，会在页面显示一个“下载按钮”，点击下载按钮就可以下载转换后的文件

#### 4.2.2 图片格式转换

只要获取了批量的图片文件，就可以将其中的JPEG图片转换为PNG格式，将其中的PNG图片转换为JPEG格式，在实现中我们也可以将其分开处理，仅把JPEG图片转换成PNG图片，或者PNG图片转换为JPEG图片，也可以同时。输出的图片可以从服务器下载查看。

#### 4.2.3 视频转音频

只要获取了批量的视频文件，无论格式，都一致输出为指定的音频格式文件。输出的音频文件可以从服务器下载播放。

#### 4.2.4 图片大小修改

只要获取了批量的JPEG图片文件，并且输入想要的分辨率的长度和宽度，就可以输出修改后的清晰的JPEG图片，输出的图片可以从服务器下载查看。

# **6 提交说明**

1. 一份实验报告
2. 源代码“网站代码.zip“

其中MMplayer应用实现了该实验报告所述的功能。

MMplayer中的templates文件夹下是网站的html文件

static文件夹下是网站的媒体（由于媒体数据（音频视频等文件）太大了就把这部分删除，没有上传了）

MMplayer/views.py 储存了所有的视图函数

1. 项目汇报时的ppt（包含演示录制视频，实现了基本的功能，最后效果以实验报告所述的为准）

# **8 参考文献**

[1]Bootstrap官方网站.Bootstrap 简介[DB/OL].

<https://www.runoob.com/bootstrap/bootstrap-intro.html>,2021-5-15.

[2]张爱华，吕京涛.CSS快速入门[M].山东：青岛出版社，2000.

[3]Microsoft官方网站.Windows Media Player SDK[DB/OL].

<https://docs.microsoft.com/zh-cn/windows/win32/wmp/about-the-windows-media-player-sdk>,2021-5-18.

[4]维基百科.VLC media player.[DB/OL].

https://wiki.videolan.org/VLC\_media\_player/,2021-5-25.

[5]樂浩.beyond.Nginx入门（三）Nginx 配置反向代理服务器[DB/OL].

https://www.jianshu.com/p/e98e84a3322f,2021-5-18.

[6]百度百科.rtmp[DB/OL].https://baike.baidu.com/item/rtmp,2021-5-29.

[7]澎湃\_L.HTML插入音频和视频：audio和video标签及其属性[DB/OL]. <https://www.cnblogs.com/EricZLin/p/8856170.html>,2021-5-20.

[8]Bootstrap中文文档.导航条[DB/OL].

<https://v3.bootcss.com/components/#navbar>,2021-5-20.

[9]MIcrosoft官方网站.OBJECT 元素中的 PARAM 元素[DB/OL].

<https://docs.microsoft.com/zh-cn/windows/win32/wmp/param-elements-in-an-object-element>,2021-5-23.

[10]维基百科.Documentation:WebPlugin[DB/OL].

https://wiki.videolan.org/Documentation:WebPlugin/#VLC\_objects,2021-5-22.

[11]cuijiecheng2018.在windows下搭建、配置nginx流媒体服务器，并进行rtmp流的推流、拉流测试 [DB/OL].

https://blog.csdn.net/u014552102/article/details/100906058,2021-5-19.

[12]FFMPEG官方网站[DB/OL].

http://www.ffmpeg.org/ffmpeg.html,2021-5-22.

[13]亭台六七座.FFMpeg 常用命令格式转换，视频合成[DB/OL].

https://blog.csdn.net/lmhuanying1012/article/details/78945609,2021-5-22.

[14]yoomaz.Python 中 subprocess.call 使用[DB/OL].

https://www.jianshu.com/p/7b4f8e4dbbe0,2021-5-25.