



武汉科技大学

Google 支持教育部产学合作专业综合改革项目-移动应用技术示范课程建设项目-嵌入式系统结构与操作系统课程-

Android 综合开发案例

智慧图书馆

嵌入式与移动计算课程组

计算机科学与技术学院

武汉科技大学

2016 年 9 月 7 日

目录

1	案例概述.....	1
1.1	案例开发背景.....	1
1.2	开发意义.....	1
1.3	设计服务范围.....	2
2	案例整体实现方案.....	3
2.1	技术路线.....	3
2.2	运行流程.....	3
2.3	系统构成.....	4
3	环境配置.....	5
3.1	.Net 环境配置.....	5
3.2	J2EE 环境.....	6
3.3	EmguCV 配置.....	9
4	代码实现.....	10
4.1	Android 客户端接口.....	10
4.2	图像识别.....	16
4.3	嵌入式接口.....	33
5	部署项目.....	36
5.1	云服务器的登录.....	36
5.2	云服务器的文件上传下载.....	38
5.3	导入数据库.....	40
5.4	部署 Book API.....	41
5.5	搭建 FTP 站点.....	41
6	常见问题及其解决方案.....	44
6.1	数据库文件被进程占用无法移动以及挂起状态的恢复.....	44
6.2	用户 sa 登录失败.....	44
6.3	图像识别时缺少 opencvcore290.dll.....	44

1 案例概述

1.1 案例开发背景

面对日益发展的社会形势，纸质图书馆的管理存在着一系列的问题，尤其是在服务的效率上。高效、便捷、灵敏以及整合的图书馆是数字化和智能化发展中的对图书馆提出的新要求。近年来，物联网作为新一代信息技术的重要组成部分，受到了科技界的广泛关注，物联网是当下最接近该模型顶端的科技概念和应用，是继计算机、互联网与移动通信网之后的世界信息产业第三次浪潮。智慧图书馆就是利用了物联网的高效性，整体性以及智能性将图书馆打造成为科技辅助型活动，他将图书管理工作与信息技术以及电子技术等现代科技类技术完美综合。它不仅仅只是一种技术，更是将来图书管理工作的必然趋势。

智慧图书馆是运用智能技术帮助图书馆的各项管理工作，尤其是在第一时间的应急管理中起到不可替代的作用。现如今，各类图书馆的建筑面积在日益扩大，基础设施的负荷也在日益加重，服务体系的信息量和运载负荷也在越来越大，而这些巨大体量的信息一旦发生奔溃，就可能给图书馆带来很大的影响。面对千变万化的发展态势，图书馆只有借助更加灵敏的管理、更加智能的信息系统和更加完善的服务才能为读者提供更加综合的信息服务。

1.2 开发意义

传统图书馆的基础设备陈旧，图书的增添、编码等管理工作十分艰巨，工作效率极其低下。智慧图书馆是指把智能技术运用到图书馆建设中而形成的一种智能化建筑,是智能建筑与高度自动化 管理的图书馆的有机结合和创新。所以，基于图书馆+物联网+云计算+智慧化设备的智慧图书馆一方面可以为馆员在智能化和自主化的基础上实现更高效率的管理，同时还可以为广大读者提供快捷便利的信息查询和阅读等综合服务。

智慧化的图书馆应该做到一般图书馆不能够做到的一些事情，一般情况下，智慧图书馆要可以实现在现有的信息技术水平之下对信息资源的全面认知，并且实现安全 可靠的、没有错误的信息传递，甚至要在实现了这些内容的基础之上对信息进行加工处理，智能化的处理也是我们对于智慧图书馆的一个稍显苛刻的

要求。

1.3 设计服务范围

对于现如今的很多传统图书馆来说，繁琐、复杂的图书的增添、编码、纠错等工作让管理人员感到焦头烂额，如何实现图书馆高效便利的管理是亟待解决的问题之一。

而智慧图书馆正好为我们提供了便利、高效的解决方案，使图书馆可以科学发展。便利是指，通过全面感知、立体互联和深度协同，将智慧技术渗透融入图书馆服务与管理的各个领域、各项业务、各个流程和各个细节，实现图书馆科学发展的创新转型。高效是指，通过节能低碳、灵敏便捷和整合集群，将智慧管理融入图书馆的一线服务与二线保障，将资源节约、环境友好的可持续发展理念导入图书馆的前台与后台、硬件与软件，在书书相联、书人相联和人人相联的基础上为读者节约时间，更加方便快捷地处理各类事物。智慧图书馆的发展新模式将提高图书馆广大读者和馆员学习和工作的自由度，将提高时间和资源的利用效率，也将推动图书馆在日新月异的信息技术发展环境下创新驱动和转型发展。

总之，智能设备的引进可以为图书管理人员节省了大量的时间。与此同时，将物联网技术与图书馆管理相结合的特点更是弥补了传统图书馆的不足，使得系统更加人性化。

2 案例整体实现方案

2.1 技术路线

随着图像识别技术的发展,现在已经能够将图片上的文字内容准确无误的识别出来,这就给文字提取,识别提供了方便。文字能够从图片上提取出来,很多人工智能判断就可以实施,该系统就是通过相机对书架上的图书进行拍照,然后利用改进的双边滤波, BP 神经网络等算法对图书的标签区域进行定位,并识别出该图书的编号,获取该图书的位置信息。

云计算技术在近年来的发展可谓突飞猛进,云计算为人们生活提供了方便,本项目也采用云计算技术——百度云应用引擎 (BAE)。由百度公司提供的云服务,同时还包含云存储、云数据库、云消息、云推送等多方位云服务。通过百度云推送,可以实时的把书架信息推送给每一个客户端,避免了单独与每一个客户端通信给服务器造成压力过大,且数据容易丢失的问题。通过云计算,将大量的数据集中在云端;对云端数据进行挖掘和分析,从而根据数据分布和可视化的大数据表达方式给图书管理提供必要的支撑。

2.2 运行流程

数据采集系统以循迹小车技术为基础,嵌入式控制器为核心架构而成。循迹小车可以自动沿着黑色轨迹运行,小车伸缩臂上的高清相机便可对书籍进行遍历拍摄。循迹采用的是红外探测法,激光打在黑色地面上会被吸收,而打在白色或其他浅色地地面上会发生漫发射。小车上的光电对管通过采集反射光,获得数据。将实时测量的数据发送给嵌入式控制器中写好的控制程序进行处理后发送信号给步进电机,从而实现对小车速度、位置和运行状态的实时控制。可以满足在图书馆内各种路线的行驶。具体轨迹铺设路径根据每个图书馆情况而定。运行模拟图如图 2-1 所示。

相机通过数据线与嵌入式控制器相连,每当相机拍摄一张图片数据,即时传送到嵌入式控制器中。嵌入式控制器作为数据交换桥梁和控制中心,在得到图像数据后,通过无线网络,将图片数据发送至服务器。服务器上有开发好的图像识别程序,实时进行图像处理并比对数据库,找出放错位置的书,并将数据发送

至百度云存储。

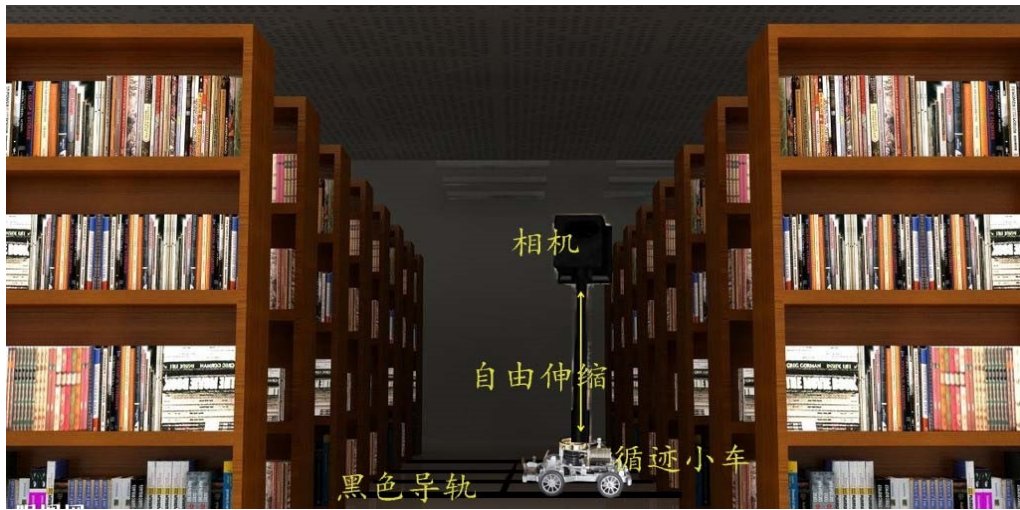


图 2-1 运行模拟图

2.3 系统构成

整个系统主要分为四个部分：Android 客户端应用程序、服务器开发、智慧图书馆终端硬件设备及相关的控制传感器。Android 手机作为主控制端，通过网络与智慧图书馆云服务器进行对接与通讯，云服务器将与智慧图书馆终端进行通讯。图书馆终端采运用树莓派开发板作为主控，将传感器采集到的数据发送到服务器。

3 环境配置

3.1 .Net 环境配置

IIS+SQL Server，尽管 j2ee 平台是搭建服务端已成事实的标准，但是 .net 开发效率更高。安装 IIS

1) 启动服务器管理器，进入添加角色和功能向导，如图 3-1 所示。



图 3-1 服务器管理

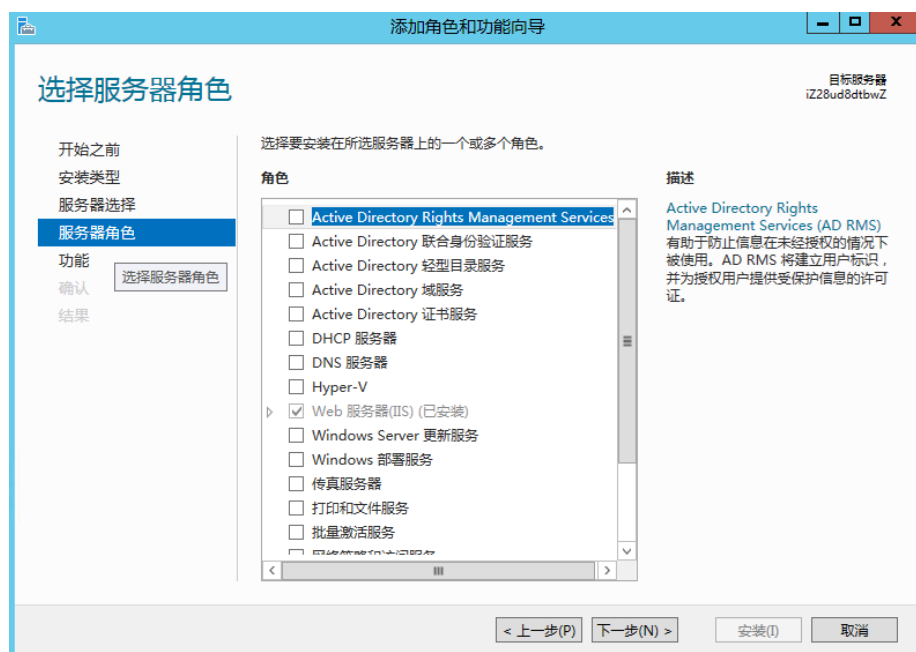


图 3-2 配置 IIS

- 2) 开始之前, 直接下一步; 在安装类型中选择基于角色后基于功能的安装; 在服务器选择中选择从服务器池中选择服务器
- 3) 在服务器角色中选择 Web 服务器 (IIS), 如图 3-2 所示。
- 4) 在功能中选择 .NET Framework 3.5 功能和 .NET Framework 4.5 功能。如图 3-3 所示。
- 5) 安装 SQL Server

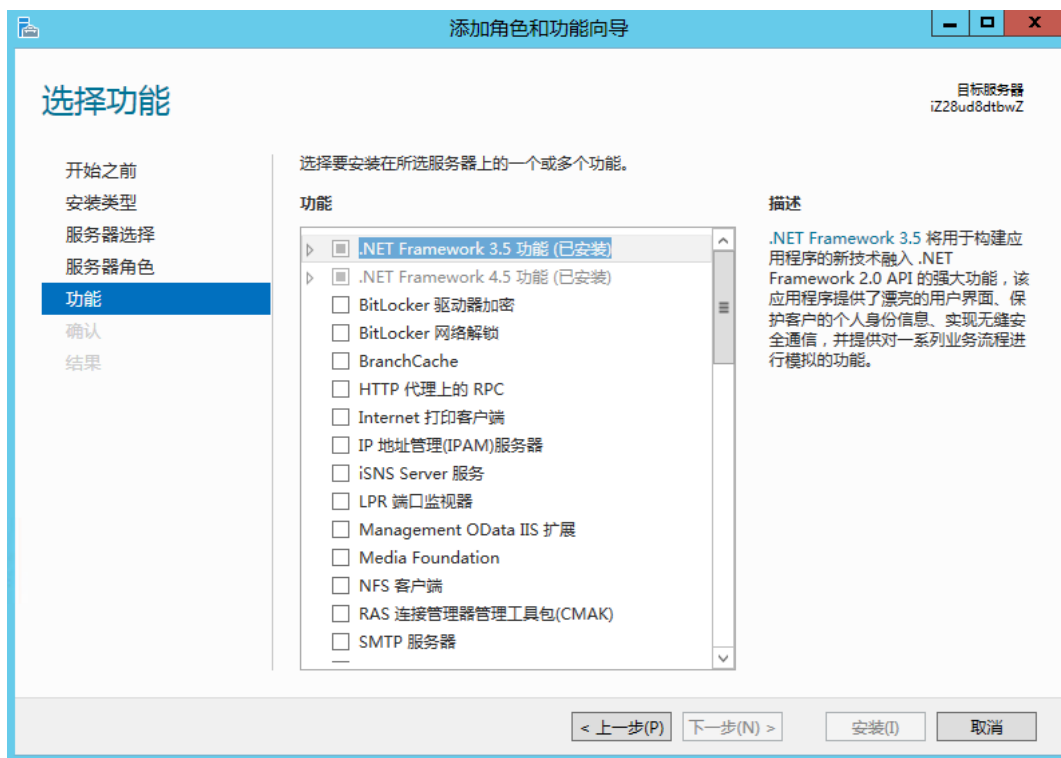


图 3-3 功能配置

3.2 J2EE 环境

Tomcat6+Navicat 8 for MySQL+myeclipse

1) 登录进入服务器安装 jdk, 配置对应环境变量

2) 安装 mysql、myeclipse、tomcat

myeclipse 安装成功后, 需要激活, 解压 JAVA EE 环境配置/ MyEclipse for Spring 10 (含破解) 其中 xxx.exe 为安装包、文件夹有破解工具和破解教程

(MyEclipse for Spring 10 的破解步骤.doc)

3) 在 myeclipse 中配置 tomcat 的路径, 然后可以实现在 myeclipse 中部署 java web

引入外部 tomcat: 在 MyEclipse 中选择 Windows->Preference, 然后再 Preference 中的 Myeclipse/Servers/Tomcat/Tomcat6.x, 用的是 tomcat6, tomcat 的路径 (可自行更改) 是 C:\Program Files (x86)\MyEclipse for Spring\tomcat6。如图 3-4 所示。

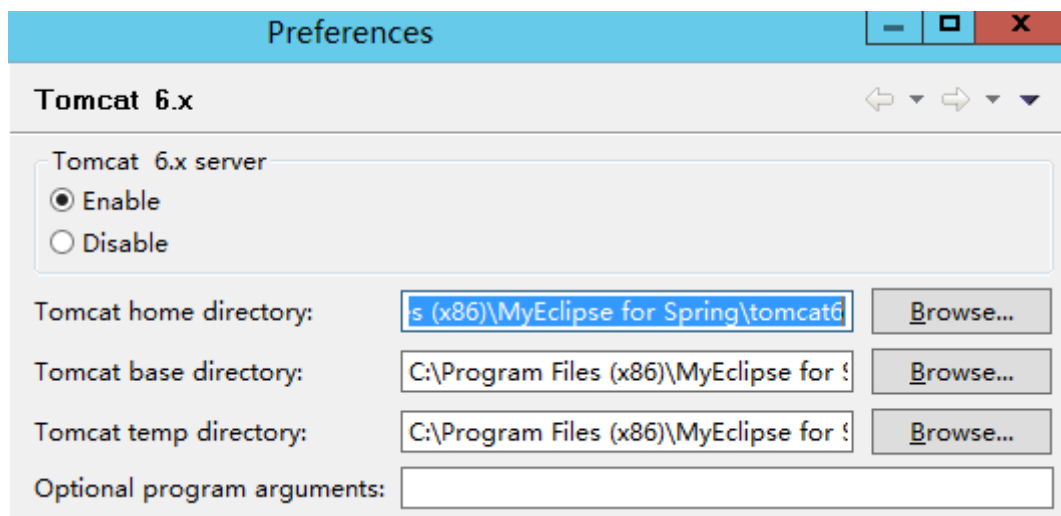


图 3-4 Tomcat 配置

4) 部署 java web 项目 (在 MyEclipse 中)

启动 tomcat server。

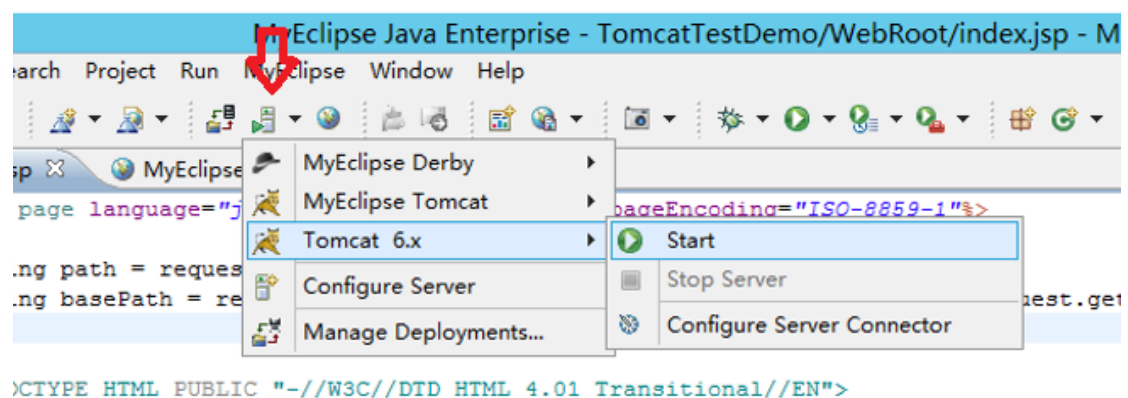


图 3-5 启动 Tomcat server

部署 java web 项目。

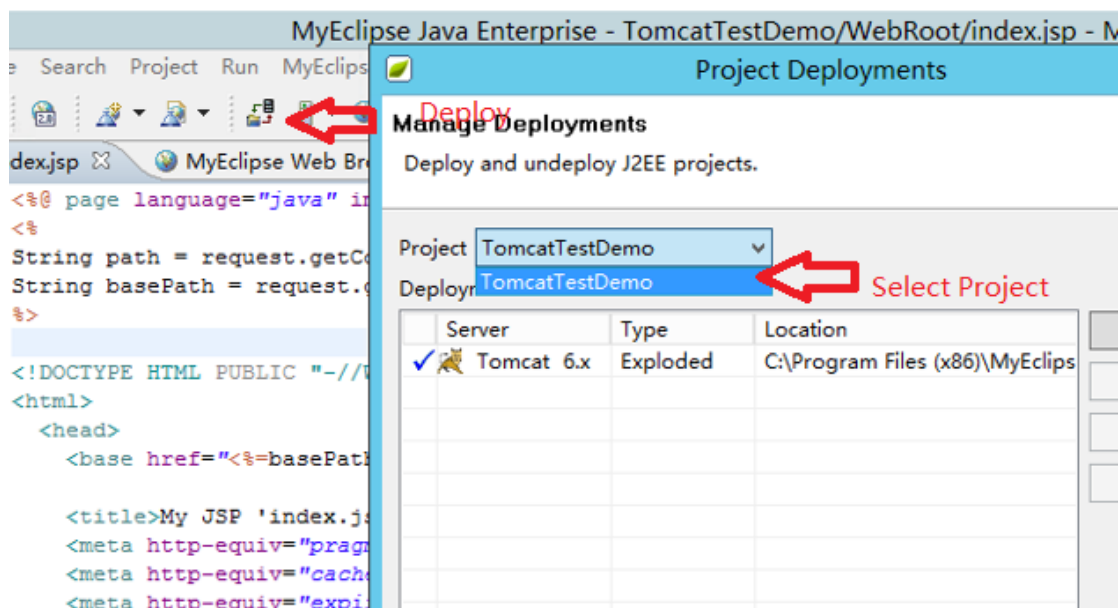


图 3-6 部署 java web 项目

5) 验证是否部署成功

本地主机验证，即在阿里云服务器上浏览自己的 web project，如图 3-7 所示。

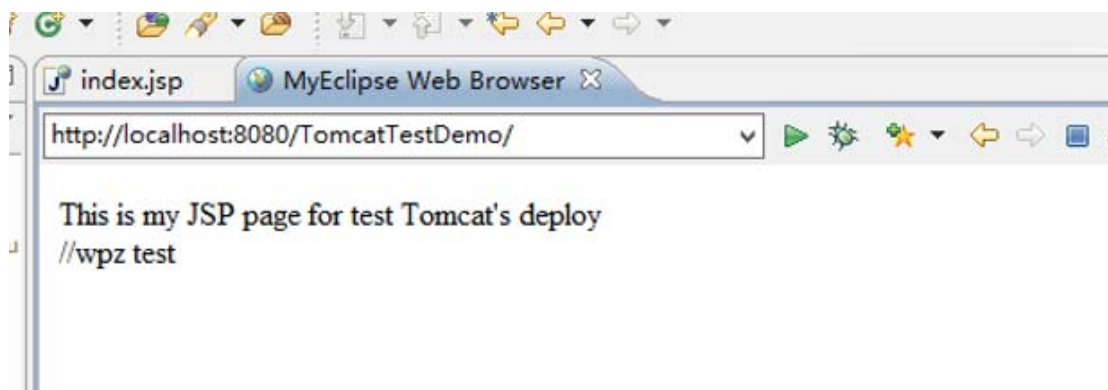


图 3-7 浏览 web 项目

远程主机访问 web project，如图 3-8 所示。

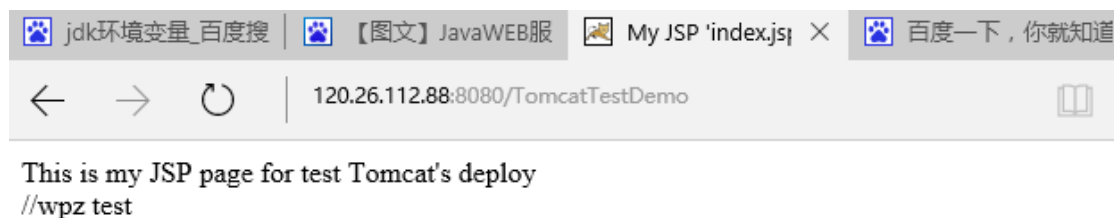


图 3-8 远程主机访问

3.3 EmguCV 配置

1) 下载 EmguCV

2) 安装

3) 设置环境变量 PATH 中添加<安装路径>\emgucv-windows-x86

2.4.0.1717\bin\x86

4) 在项目中添加引用

4 代码实现

4.1 Android 客户端接口

Android 客户端接口：

1) 提交操作信息 方法

方法名：uploadOperation

URL：http://115.28.26.84:8088/index.aspx

HTTP 方法：GET

表单参数及返回值，如表 4-1 所示。

表 4-1 uploadOperation 表单参数及返回值

参数	值	中文说明
Oname	string	操作人
Otime	DateTime	操作时间
id	string	书本 id
返回	值	备注
json 字符串	{"State":1,"Msg":"提交成功"}	State 为 1 表示成功，否则失败；

```
/// <summary>
    /// 设置参数名，完成操作信息的提交
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    protected void Page_Load(object sender, EventArgs e)
    {
        OperationManager om = new OperationManager();
        string Oname = Request.QueryString["Oname"];
        string Otime = Request.QueryString["Otime"];
        string id=Request.QueryString["id"];

        Response.Write( om.uploadOperation(Oname,Otime,id));
    }
```

```
/// <summary>
/// 提交错书信息
/// </summary>
/// <param name="Oname"></param>
/// <param name="Otime"></param>
/// <param name="id"></param>
/// <returns>提交是否成功</returns>
public string uploadOperation(string Oname,string Otime,string id)
{
    JsonWriter jw = new JsonWriter();

    DateTime result = new DateTime();
    if (Oname == "" || Otime == "")
    {
        jw.Write("State", 0);
        jw.Write("Msg", "输入不能空");
    }
    else
    {
        try
        {
            DateTime.TryParse(Otime, out result);
            int i = rbd.uploadOperation(Oname, Otime,id);
            if (i >= 1)
            {
                jw.Write("State", 1);
                jw.Write("Msg", "提交成功");
            }
            else
            {

```

```

        jw.Write("State", 0);
        jw.Write("Msg", "提交失败");
    }
}
catch
{
    jw.Write("State", 0);
    jw.Write("Msg", "时间格式不对");
}
}
return jw.EndWrite();
}

```

2) 获取操作记录方法

方法名: `getOperation`

URL: `http://115.28.26.84:8088/getOperation.aspx`

HTTP 方法: `GET`

表单参数及返回值, 如表 4-2 所示。

表 4-2 `getOperation` 表单参数及返回值

返回	值	备注
json 字符串	<pre> {"State":1,"Msg":"","Content":[{"Oname":"红","Otime":"2016/6/23 0:40:39","id":"XEK000006"},{"Oname":"Admi","Otime":"2015/10/2 10:10:10","id":"XEK000001"}]} </pre>	State 为 1 表示成功, 否则失败;

/// <summary>

/// 获取操作信息

/// </summary>

/// <returns>操作信息的 json</returns>

public string getOperation()

{

 JsonWriter jw = new JsonWriter();

```

jw.Write("State",1);
jw.Write("Msg","");
List<Operation> list = new List<Operation>();
DataTable dt=rbd.getOperation();
foreach(DataRow dr in dt.Rows)
{
    Operation rb = new Operation();
    rb.Oname = dr[0].ToString();
    rb.Otime = dr[1].ToString();
    rb.id = dr[2].ToString();
    list.Add(rb);
}
string jsonstr=JsonConvert.SerializeObject(list);
jw.Write("Content",jsonstr,1);

return jw.endWrite();
}

```

3) 获取错书信息 方法

方法名: getWrongBook

URL: <http://115.28.26.84:8088/WrongBook.aspx>

HTTP 方法: GET

表单参数及返回值, 如表 4-3 所示。

表 4-3 getWrongBook 表单参数及返回值

返回	值	备注
json 字符串	{"State":1,"Msg":"","Content":[{"id":"XEK000005","shelf":"3","floor":"3","r_shelf":"2","name":"什么是数学","r_floor":"2"}]}	State 为 1 表示成功, 否则失败;

/// <summary>

///

/// </summary>

```
/// <returns>错书数据的 json</returns>
public string getWrongBook()
{
    JsonWriter jw = new JsonWriter();

    List<WrongBook> list = new List<WrongBook>();
    DataTable dt=rbd.getWrongBook();
    if (dt.Rows.Count > 0)
    {
        jw.Write("State", 1);
        jw.Write("Msg", "");
        foreach (DataRow dr in dt.Rows)
        {
            WrongBook rb = new WrongBook();
            rb.id = dr[0].ToString();
            rb.shelf = dr[1].ToString();
            rb.floor = dr[2].ToString();
            rb.name = dr[3].ToString();
            rb.r_shelf = dr[4].ToString();
            rb.r_floor = dr[5].ToString();
            list.Add(rb);
        }
    }
    else
    {
        jw.Write("State", 1);
        jw.Write("Msg", "暂无错误");
    }

    string jsonstr = JsonConvert.SerializeObject(list);
```



```
jw.Write("Content", jsonstr, 1);

return jw.endWrite();

}
```

4) 获取书籍信息 方法(方便客户端测试)

方法名: getRightBook

URL: http://115.28.26.84:8088/RightBook.aspx

HTTP 方法: GET

表单参数及返回值, 如表 4-4 所示。

表 4-4 getRightBook 表单参数及返回值

返回	值	备注
json 字符串	<pre>{ "State":1,"Msg":"","Content":[{ "id":"XEK000001","shelf":"1","floor":"1","name":"软件设计师教程" },{ "id":"XEK000002","shelf":"1","floor":"1","name":"组合数学" },{ "id":"XEK000003","shelf":"1","floor":"1","name":"源氏物语" },{ "id":"XEK000004","shelf":"2","floor":"2","name":"红楼梦" },{ "id":"XEK000005","shelf":"2","floor":"2","name":"什么是数学" },{ "id":"XEK000006","shelf":"1","floor":"1","name":"毛概" },{ "id":"XEK000007","shelf":"1","floor":"1","name":"离散数学" },{ "id":"XEK000008","shelf":"2","floor":"2","name":"电工与电子技术" },{ "id":"XEK000009","shelf":"1","floor":"1","name":"计算机专业英语教程" },{ "id":"XEK000010","shelf":"1","floor":"1","name":"数据库系统概论" }] }</pre>	State 为 1 表示成功, 否则失败;

5) 添加错书 方法 (便于客户端测试)

方法名：自动添加两本错书

URL: <http://115.28.26.84:8088/run.aspx>

HTTP 方法: GET

4.2 图像识别

```
/// <summary>
    /// 删除 ftp 目录下的所有图片
    /// </summary>
    /// <param name="Path">ftp 目录下的路径</param>
    private void deleteImage(string Path)
    {
        DirectoryInfo dir = new DirectoryInfo(Path);
        FileInfo[] fi = fi = dir.GetFiles();

        for (int i = 0; i < fi.Length; i++)
        {
            fi[i].Delete();
        }
    }
    /// <summary>
    /// 图像分析
    /// </summary>
    /// <param name="Path">ftp 目录下的路径</param>
    /// <returns>成功返回 0，失败返回-1</returns>
    private int analyseImage(string Path)
    {
        ScanPictures(Path);
        //deleteImage(Path);
        return 0;
    }
```

```
/// <summary>
/// 扫描文件夹里的照片
/// </summary>
void ScanPictures(string path)
{
    WrongBookDAO wbd = new WrongBookDAO();
    wbd.DeleteAllWrongBook();

    DirectoryInfo theFolder = new DirectoryInfo(path);

    FileInfo[] fileInfo = theFolder.GetFiles();

    foreach (FileInfo NextFile in fileInfo)
    {
        string img_path = path + "\\" + NextFile.Name;

        //try
        //{
            Image img = Image.FromFile(img_path);
            Bitmap preImg = new Bitmap(img);
            PreProcessing pre = new PreProcessing(ref preImg);

            List<string> list = findSquares4((Bitmap)preImg);
            img.Dispose();
            foreach (var Id in list)
            {
                RightBookDAO rbd = new RightBookDAO();
                if (!rbd.RightIsExist(Id)) continue;

                BookModel scan = new BookModel();
```

```

        scan.Id = Id;
        scan.Shelf = "1";
        scan.Floor = "1";
        DataTable dr = rbd.BookById(Id);
        if (dr.Rows[0][1].ToString() != scan.Shelf ||
dr.Rows[0][1].ToString() != scan.Floor)
        {
            wbd.InsertWrongBook(Id,
dr.Rows[0][1].ToString(),    dr.Rows[0][2].ToString(),    dr.Rows[0][3].ToString(),
scan.Shelf, scan.Floor);
        }
    }
}
//}
//catch
//{
//}
//全部结束
}
}

```

```

/// <summary>
/// 提取矩形区域
/// </summary>
/// <param name="bmp"></param>
unsafe List<string> findSquares4(Bitmap bmp)
{
    Image<Gray, Byte> img = new Image<Gray, byte>(bmp);
    MemStorage storage = new MemStorage();

```

```

IntPtr Dyncontour = new IntPtr(); //存放检测到的图像块的首地址

//Seq<Point> Mycontour = new Seq<Point>(storage, null);
List<string> list = new List<string>();
int i, l, N = 11;

Size sz = new Size(img.Width & -2, img.Height & -2);
Image<Gray, Byte> timg = new Image<Gray, Byte>(sz);
//CvInvoke.cvCopy(img.Ptr, timg.Ptr, IntPtr.Zero); // make a copy
of input image

timg = img.Copy();

//IntPtr      gray_p      =      CvInvoke.cvCreateImage(sz,
IPL_DEPTH.IPL_DEPTH_8U, 1);

//MplImage      gray_pp      =
(MplImage)Marshal.PtrToStructure(gray_p, typeof(MplImage));

//Image<Bgr,      Byte>      gray      =      new      Image<Bgr,
Byte>(gray_pp.width, gray_pp.height, gray_pp.widthStep, gray_pp.imageData);

Image<Gray, Byte> gray = new Image<Gray, Byte>(sz);

//IntPtr pyr_p = CvInvoke.cvCreateImage(new Size(sz.Width / 2,
sz.Height / 2), IPL_DEPTH.IPL_DEPTH_8U, 3);

//MplImage pyr_pp = (MplImage)Marshal.PtrToStructure(pyr_p,
typeof(MplImage));

//Image<Gray,      Byte>      pyr      =      new      Image<Gray,
Byte>(pyr_pp.width, pyr_pp.height, pyr_pp.widthStep, pyr_pp.imageData);

Image<Gray, Byte> pyr = new Image<Gray, Byte>(new
Size(sz.Width / 2, sz.Height / 2));

Image<Gray, Byte> tgray;

double s, t;

```

```

        // create empty sequence that will contain points -
        // 4 points per square (the square's vertices)
        IntPtr sq = CvInvoke.cvCreateSeq(0, sizeof(MCvSeq),
sizeof(Point), storage);

        //MCvSeq squares=new MCvSeq();
        //squares.ptr=sq;

        Seq<Point> squares = new Seq<Point>(sq, null);
        //MCvSeq* squares;

        // select the maximum ROI in the image
        // with the width and height divisible by 2
        CvInvoke.cvSetImageROI(timg, new Rectangle(0, 0, sz.Width,
sz.Height));

        // down-scale and upscale the image to filter out the noise
        //CvInvoke.cvPyrDown(timg.Ptr, pyr.Ptr,
FILTER_TYPE.CV_GAUSSIAN_5x5);
        //CvInvoke.cvPyrUp(pyr.Ptr, timg.Ptr,
FILTER_TYPE.CV_GAUSSIAN_5x5);

        pyr = timg.PyrDown();
        timg = pyr.PyrUp();

        IntPtr temp1_pyr_p = CvInvoke.cvCreateImage(sz,
IPL_DEPTH.IPL_DEPTH_8U, 1);
        MIplImage temp1 =

```

```
(MiplImage)Marshal.PtrToStructure(temp1_pyr_p, typeof(MiplImage));  
    //tgray = new Image<Gray, Byte>(temp1.width, temp1.height,  
temp1.widthStep, temp1.imageData);
```

```
tgray = new Image<Gray, Byte>(sz);
```

```
//// find squares in every color plane of the image  
//for (c = 0; c < 3; c++)  
//{  
//    // extract the c-th color plane  
//    CvInvoke.cvSetImageCOI(timg.Ptr, c + 1);  
//    //CvInvoke.cvCopy(timg.Ptr, tgray.Ptr, IntPtr.Zero);  
tgray = timg.Copy();
```

```
AllRect allRect = new AllRect();  
// try several threshold levels  
for (l = 0; l < N; l++)  
{  
    // hack: use Canny instead of zero threshold level.  
    // Canny helps to catch squares with gradient shading  
    if (l == 0)  
    {  
        // apply Canny. Take the upper threshold from slider  
        // and set the lower to 0 (which forces edges merging)  
        //CvInvoke.cvCanny(tgray.Ptr, gray.Ptr, 60, 180, 3);  
        gray = tgray.Canny(60, 180, 3);  
        // dilate canny output to remove potential  
        // holes between edge segments  
        //CvInvoke.cvDilate(gray.Ptr, gray.Ptr, IntPtr.Zero, 1);
```

```

        gray = gray.Dilate(1);
    }
    else
    {
        // apply threshold if l!=0:
        //      tgray(x,y) = gray(x,y) < (l+1)*255/N ? 255 : 0
        //cvThreshold(  tgray,  gray,  (l+1)*255/N,  255,
CV_THRESH_BINARY );
        CvInvoke.cvThreshold(tgray.Ptr,  gray.Ptr,  50,  255,
THRESH.CV_THRESH_BINARY);
    }

    // find contours and store them all as a list
    //CvInvoke.cvFindContours(gray.Ptr,  storage.Ptr,  ref
contours->ptr, sizeof(MCvContour),
    //RETR_TYPE.CV_RETR_LIST,
CHAIN_APPROX_METHOD.CV_CHAIN_APPROX_SIMPLE, new Point(0, 0));
    //
    Mycontour =
gray.FindContours(CHAIN_APPROX_METHOD.CV_CHAIN_APPROX_SIMPLE,
RETR_TYPE.CV_RETR_LIST,storage);

    int  n  =  CvInvoke.cvFindContours(gray,  storage,  ref
Dyncontour, 100,
    Emgu.CV.CvEnum.RETR_TYPE.CV_RETR_LIST,
Emgu.CV.CvEnum.CHAIN_APPROX_METHOD.CV_CHAIN_APPROX_NONE,
new Point(0, 0));

    Seq<Point>  Mycontour  =  new  Seq<Point>(Dyncontour,
null);//方便对 IntPtr 类型进行操作

```

```

    IntPtr temp_res = new IntPtr();
    Seq<Point> result = new Seq<Point>(temp_res, null);

    // test each contour
    while (Mycontour != null && Mycontour.Ptr.ToInt32() != 0)
    {
        // approximate contour with accuracy proportional
        // to the contour perimeter

        temp_res = CvInvoke.cvApproxPoly(Mycontour, 88,
storage,

APPROX_POLY_TYPE.CV_POLY_APPROX_DP,
CvInvoke.cvContourPerimeter(Mycontour) * 0.02, 0);

        result = new Seq<Point>(temp_res, null);
        // square contours should have 4 vertices after
approximation

        // relatively large area (to filter out noisy contours)
        // and be convex.
        // Note: absolute value of an area is used because
        // area may be positive or negative - in accordance with
the

        // contour orientation

        if (result.Count() == 4 &&

System.Math.Abs(CvInvoke.cvContourArea(result.Ptr, MCvSlice.WholeSeq, 1)) >
1000 &&

        CvInvoke.cvCheckContourConvexity(result.Ptr) !=

```

0)

```
{
    s = 0;

    for (i = 0; i < 5; i++)
    {
        // find minimum angle between joint
        // edges (maximum of cosine)
        if (i >= 2)
        {
            Point p1 =
(Point)Marshal.PtrToStructure(CvInvoke.cvGetSeqElem(result.Ptr, i), typeof(Point));
            Point p2 =
(Point)Marshal.PtrToStructure(CvInvoke.cvGetSeqElem(result.Ptr, i - 2),
typeof(Point));
            Point p3 =
(Point)Marshal.PtrToStructure(CvInvoke.cvGetSeqElem(result.Ptr, i - 1),
typeof(Point));

            t = System.Math.Abs(angle(p1, p2, p3));
            s = s > t ? s : t;
        }
    }

    // if cosines of all angles are small
    // (all angles are ~90 degree) then write
quandrangle

    // vertices to resultant sequence
    if (s < 0.3)
        for (i = 0; i < 4; i++)
            CvInvoke.cvSeqPush(squares.Ptr,
```

```

CvInvoke.cvGetSeqElem(result.Ptr, i));
    }
    // take the next contour
    Mycontour = Mycontour.HNext;
}
}
//}

// release all the temporary images

//CvInvoke.cvReleaseImage(ref gray.Ptr);
//CvInvoke.cvReleaseImage(&pyr);
//CvInvoke.cvReleaseImage(&tgray);
//CvInvoke.cvReleaseImage(&ting);
MCvSeqReader reader = new MCvSeqReader();
CvInvoke.cvStartReadSeq(squares.Ptr, ref reader, false);
Point[] pt = new Point[4];
for (i = 0; i < squares.Total; i += 4)
{
    pt[0] =
(Point)Marshal.PtrToStructure(CvInvoke.cvGetSeqElem(squares.Ptr, i),
typeof(Point));
    pt[1] =
(Point)Marshal.PtrToStructure(CvInvoke.cvGetSeqElem(squares.Ptr, i + 1),
typeof(Point));
    pt[2] =
(Point)Marshal.PtrToStructure(CvInvoke.cvGetSeqElem(squares.Ptr, i + 2),
typeof(Point));
    pt[3] =
(Point)Marshal.PtrToStructure(CvInvoke.cvGetSeqElem(squares.Ptr, i + 3),

```

```
typeof(Point));
```

```
        allRect.add(pt);
        //draw the square as a closed polyline
        //CvInvoke.cvPolyLine(img.Ptr, pt, 4, 1, true, new
MCvScalar(0, 255, 0), 3, LINE_TYPE.CV_AA, 0);
    }
    Bitmap img_temp = img.Bitmap;

    for (int k = 0; k < allRect.Total; k++)
    {
        Point[] cut_p = new Point[4];
        cut_p = Sort(allRect.rect[k].point);

        int W, H;
        W = Dis(cut_p[0], cut_p[1]);
        H = Dis(cut_p[0], cut_p[3]);
        //if (W < H)
        //{
        //    W = W ^ H;
        //    H = W ^ H;
        //    W = W ^ H;
        //}
        CvInvoke.cvSetImageROI(img, new Rectangle(cut_p[0].X,
cut_p[0].Y, W, H)); //设置源图像 ROI

        Image<Gray, Byte> pDest = new Image<Gray, Byte>(sz); //
创建目标图像

        pDest = img.Copy(); //复制图像
        //cvResetImageROI(pDest); //源图像用完后，清空 ROI
```

```
//cvSaveImage("新图.jpg", pDest);//保存目标图像
//pDest.Save(@"C:\Users\BoBo\Desktop\新图 " + k +
".jpg");

string result = Ocr(pDest.Bitmap);
result = Correct(result);
if (result != null && !list.Exists(x => x == result))
{
    list.Add(result);
}
}
return list;
}
```

```
public enum TesseractEngineMode : int
{
    /// <summary>
    /// Run Tesseract only - fastest
    /// </summary>
    TESSERACT_ONLY = 0,

    /// <summary>
    /// Run Cube only - better accuracy, but slower
    /// </summary>
    CUBE_ONLY = 1,

    /// <summary>
    /// Run both and combine results - best accuracy
    /// </summary>
    TESSERACT_CUBE_COMBINED = 2,
```

```
    /// <summary>
    /// Specify this mode when calling init_*( ),
    /// to indicate that any of the above modes
    /// should be automatically inferred from the
    /// variables in the language-specific config,
    /// command-line configs, or if not specified
    /// in any of the above should be set to the
    /// default OEM_TESSERACT_ONLY.
    /// </summary>
    DEFAULT = 3
}

public enum TesseractPageSegMode : int
{
    /// <summary>
    /// Fully automatic page segmentation
    /// </summary>
    PSM_AUTO = 0,

    /// <summary>
    /// Assume a single column of text of variable sizes
    /// </summary>
    PSM_SINGLE_COLUMN = 1,

    /// <summary>
    /// Assume a single uniform block of text (Default)
    /// </summary>
    PSM_SINGLE_BLOCK = 2,

    /// <summary>
```

```

        /// Treat the image as a single text line
        /// </summary>
        PSM_SINGLE_LINE = 3,

        /// <summary>
        /// Treat the image as a single word
        /// </summary>
        PSM_SINGLE_WORD = 4,

        /// <summary>
        /// Treat the image as a single character
        /// </summary>
        PSM_SINGLE_CHAR = 5
    }

    TesseractProcessor m_tesseract = null;
    private string m_path = "..\\tessdata\\";
    private string m_lang = "eng";
    private string Ocr(Image image)
    {
        m_tesseract = new TesseractProcessor();
        m_tesseract.Init(m_path, m_lang,
(int)TesseractEngineMode.DEFAULT);
        m_tesseract.SetVariable("tessedit_pageseg_mode",
TesseractPageSegMode.PSM_AUTO.ToString());
        m_tesseract.SetVariable("tessedit_char_whitelist",
"0123456789XEKAM");
        m_tesseract.Clear();
        m_tesseract.ClearAdaptiveClassifier();
        return m_tesseract.Apply(image);
    }

```

```
}
```

```
int Dis(Point p1, Point p2)
{
    return Convert.ToInt32(System.Math.Sqrt((p1.X - p2.X) * (p1.X -
p2.X) + (p1.Y - p2.Y) * (p1.Y - p2.Y))) + 1;
}
```

```
double angle(System.Drawing.Point pt1, System.Drawing.Point pt2,
System.Drawing.Point pt0)
{
    double dx1 = pt1.X - pt0.X;
    double dy1 = pt1.Y - pt0.Y;
    double dx2 = pt2.X - pt0.X;
    double dy2 = pt2.Y - pt0.Y;
    return (dx1 * dx2 + dy1 * dy2) / Math.Sqrt((dx1 * dx1 + dy1 *
dy1) * (dx2 * dx2 + dy2 * dy2) + 1e-10);
}
```

```
Point[] Sort(Point[] p)
{
    Point[] t = new Point[4];
    for (int i = 0; i < 4; i++)
    {
        t[i] = new Point(p[i].X, p[i].Y);
    }

    for (int i = 0; i < 4; i++)
```

```
{
    for (int j = i + 1; j < 4; j++)
    {
        if (t[i].X > t[j].X)
        {
            Point tp = new Point(t[i].X, t[i].Y);
            t[i] = new Point(t[j].X, t[j].Y);
            t[j] = new Point(tp.X, tp.Y);

        }
    }
}
```

```
if (t[0].Y > t[1].Y)
{
    Point tp = new Point(t[0].X, t[0].Y);
    t[0] = new Point(t[1].X, t[1].Y);
    t[1] = new Point(tp.X, tp.Y);

    tp = new Point(t[1].X, t[1].Y);
    t[1] = new Point(t[3].X, t[3].Y);
    t[3] = new Point(tp.X, tp.Y);
}
```

```
else if (t[0].Y < t[1].Y)
{
    Point tp = new Point(t[1].X, t[1].Y);
    t[1] = new Point(t[2].X, t[2].Y);
    t[2] = new Point(tp.X, tp.Y);
}
```

```

        tp = new Point(t[2].X, t[2].Y);
        t[2] = new Point(t[3].X, t[3].Y);
        t[3] = new Point(tp.X, tp.Y);
    }

    return t;
}

/// <summary>
/// 纠正识别
/// </summary>
/// <param name="s"></param>
/// <returns></returns>
string Correct(string s)
{
    int length = s.Length;
    for (int i = 0; i < length; i++)
    {
        if (s[i] == 'X' || s[i] == 'A')
        {
            if (check(s, i))
                return s.Substring(i, 9);
        }
    }
    return null;
}

bool check(string s, int start)
{
    if (s[start] == 'X')
    {

```

```

        if (s[start + 1] == 'E' && s[start + 2] == 'K')
        {
            for (int i = 3; i < 9; i++)
                if (!char.IsDigit(s[start + i]))
                    return false;

            return true;
        }
    }
else
{
    if (s[start + 1] == 'U' && s[start + 2] == 'M')
    {
        for (int i = 3; i < 9; i++)
            if (!char.IsDigit(s[start + i]))
                return false;

        return true;
    }
}

return false;
}
}

```

4.3 嵌入式接口

方法名: GetImage

URL: <http://115.28.26.84:8088/GetImage.aspx>

HTTP 方法: GET

表单参数及返回值, 如表 4-5 所示。

表 4-5 GetImage 表单参数及返回值

参数	值	中文说明
State	Int	0: 删除历史图像数据

```
protected void Page_Load(object sender, EventArgs e)
```

```
{  
    string state = Request["State"];  
    string path = @"D:\wwwroot\BookFTP";  
    switch (Convert.ToInt32(state))  
    {  
        case 0: deleteImage(path); break;  
        case 1: analyseImage(path); break;  
    }  
  
}  
  
/// <summary>  
/// 删除 ftp 目录下的所有图片  
/// </summary>  
/// <param name="Path">ftp 目录下的路径</param>  
private void deleteImage(string Path)  
{  
    DirectoryInfo dir = new DirectoryInfo(Path);  
    FileInfo[] fi = fi = dir.GetFiles();  
  
    for (int i = 0; i < fi.Length; i++)  
    {  
        fi[i].Delete();  
    }  
}  
  
/// <summary>  
/// 图像分析  
/// </summary>  
/// <param name="Path">ftp 目录下的路径</param>
```

```
/// <returns>成功返回 0, 失败返回-1</returns>
```

```
private int analyseImage(string Path)
```

```
{
```

```
    ScanPictures(Path);
```

```
    deleteImage(Path);
```

```
    return 0;
```

```
}
```

5 部署项目

5.1 云服务器的登录

使用 Windows 系统自带远程工具连接 Windows 服务器。

1) Windows+R 打开命令运行，如图 5-1 所示

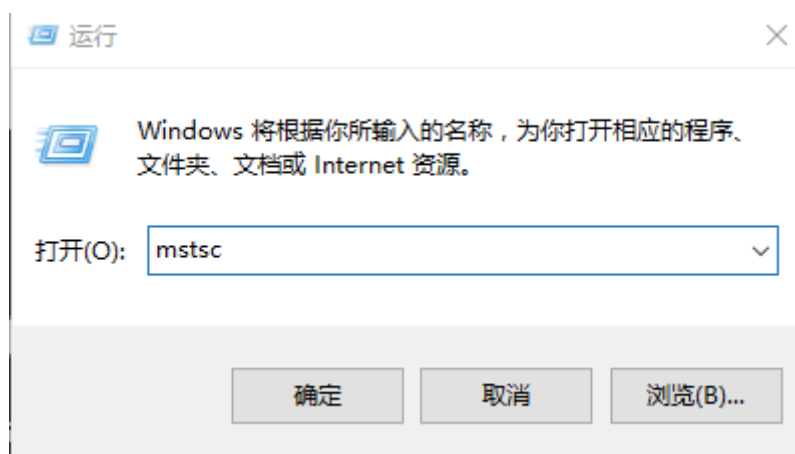


图 5-1 远程服务器登录（1）

2) 输入 mstsc，点击确定，如图 5-2 所示

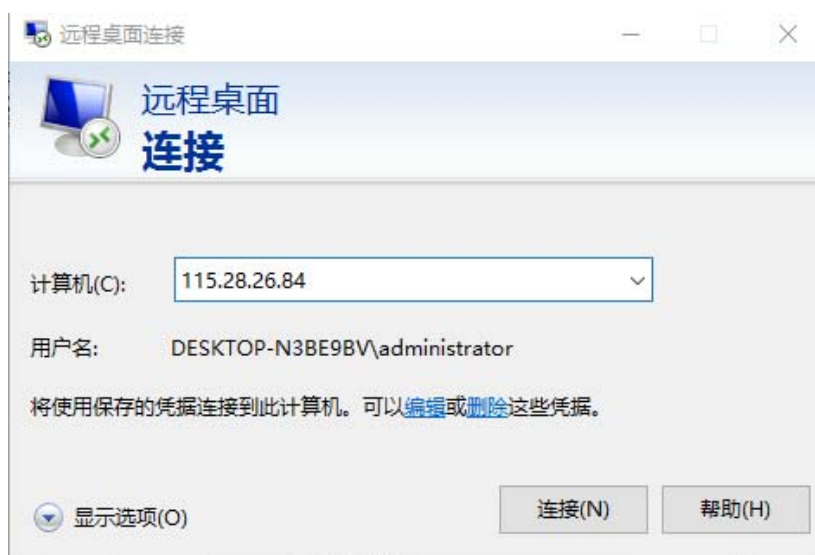


图 5-2 远程服务器登录（2）

3) 输入服务器 IP

4) 输入对应的账号密码（默认账号为 administrator），如图 5-3 所示。

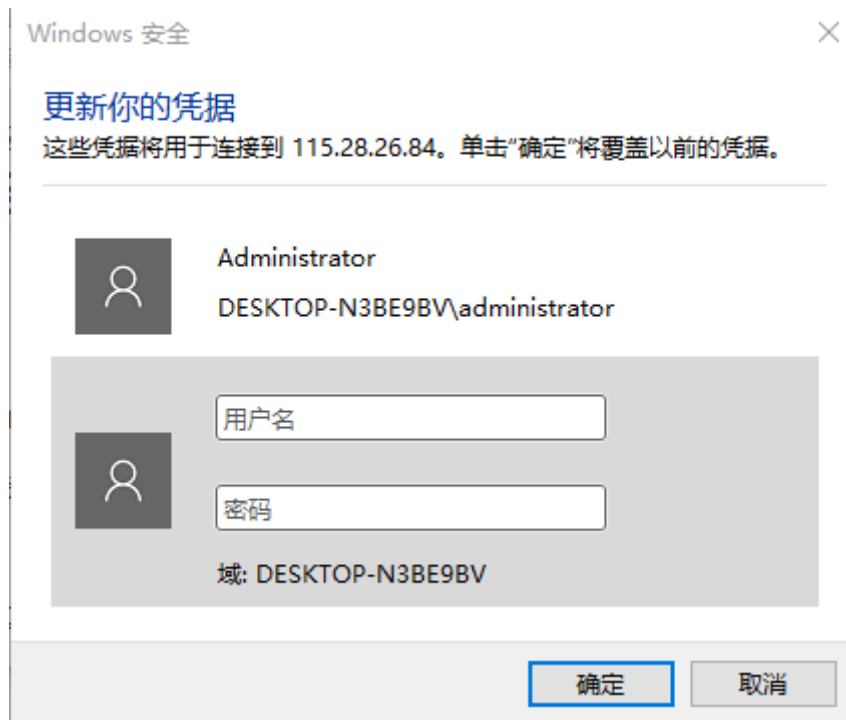


图 5-3 远程服务器登录（3）



图 5-4 远程桌面连接（1）

5.2 云服务器的文件上传下载

Windows Server 2003 及以前版本无法直接通过 Ctrl+C 和 Ctrl+V 完成文件上传下载

- 1) 在输入服务器 IP 界面，点击显示选项
- 2) 选择本地资源选项卡，点击详细信息，如图 5-4 所示
- 3) 选择要使用的驱动器资源，如图 5-5 所示

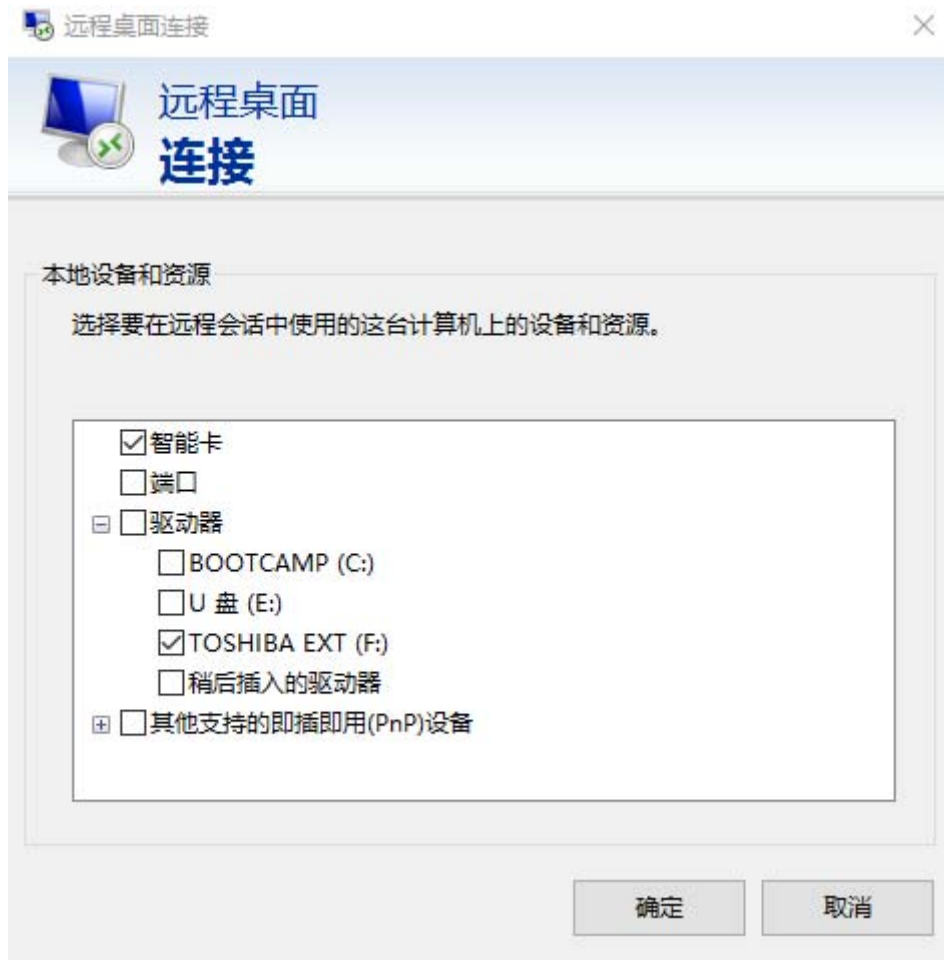


图 5-5 远程桌面连接（2）

4) 登录服务器，可在服务器上发现自己的硬盘，即可互传文件，如图 5-6 所示

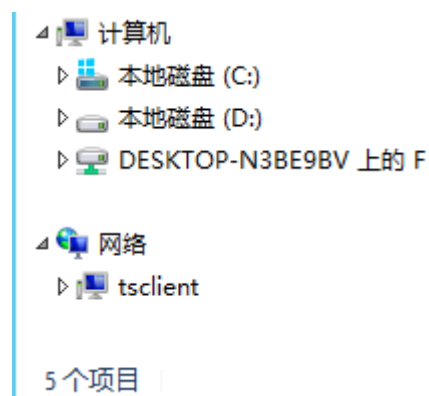


图 5-6 远程桌面连接（3）

5.3 导入数据库

1) 安装好 SQL Server 后, 将数据库导入 DBMS 步骤: 找到 SQL 中 MSSQL 实例对应的目录, 进入对应的 DATA 文件, 将数据库文件复制到对应文件夹下(默认为 C:\Program Files\Microsoft SQL Server\MSSQL11.SQLEXPRESS\MSSQL\DATA)

2) 登录到对应的数据库服务器, 默认本地, 如果数据库在其他服务器配置允许远程连接即可

3) 连接后在数据库选项上鼠标功能键右键选择附加, 点击添加后找到对应的数据库, 然后确定

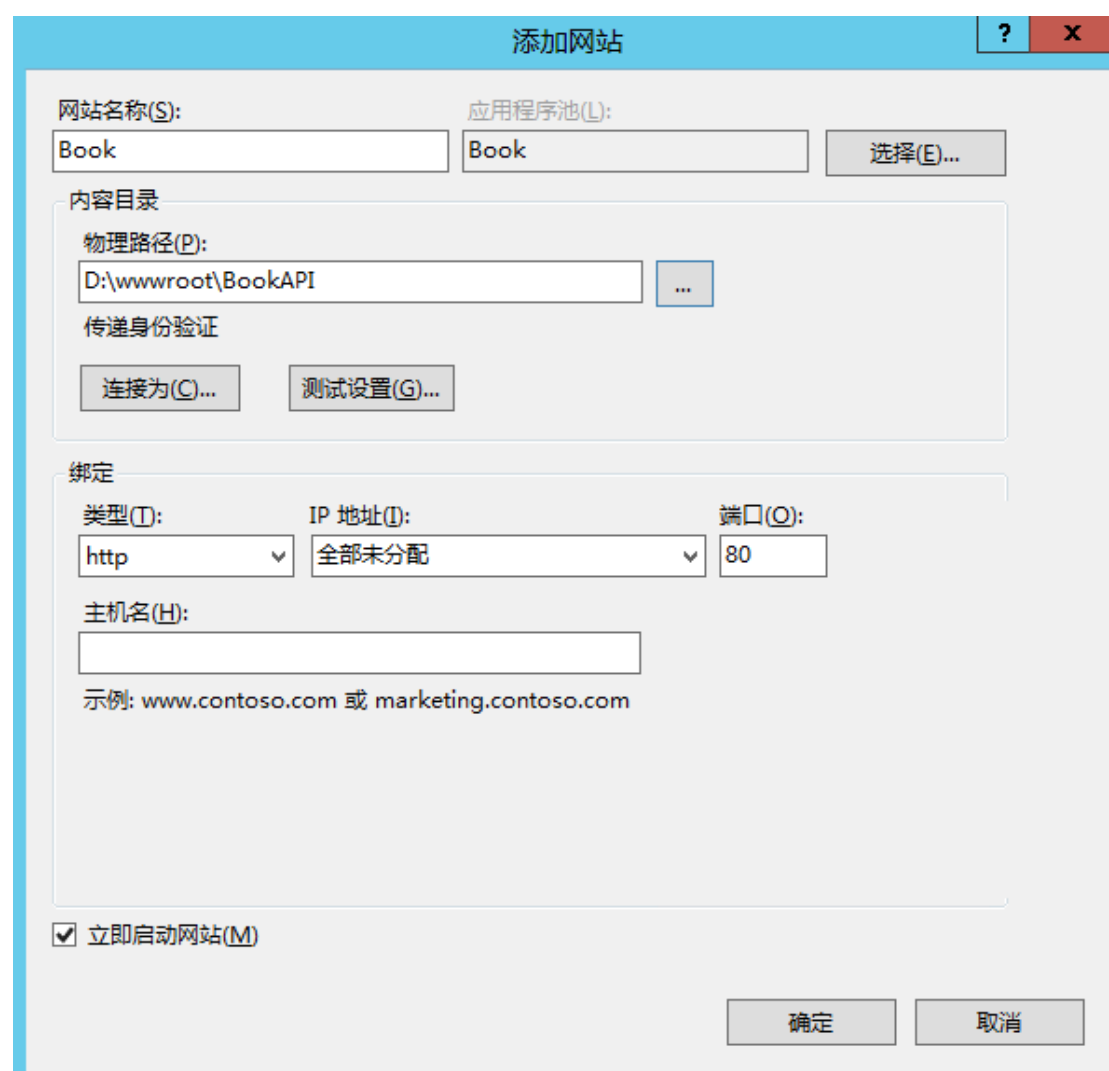


图 5-7 部署 Book API (1)

5.4 部署 Book API

- 1) 打开 Internet 信息服务管理器,在网站选项上右键功能键, 选择添加网站
- 2) 填写网站名称、选择项目的物理路径, 绑定端口, 如果有域名可以在主机名下填写域名 (非必要), 如图 5-7 所示
- 3) 设置应用程序池, 在对应的高级设置中将启用 32 为应用模式改为 true, 如图 5-8 所示

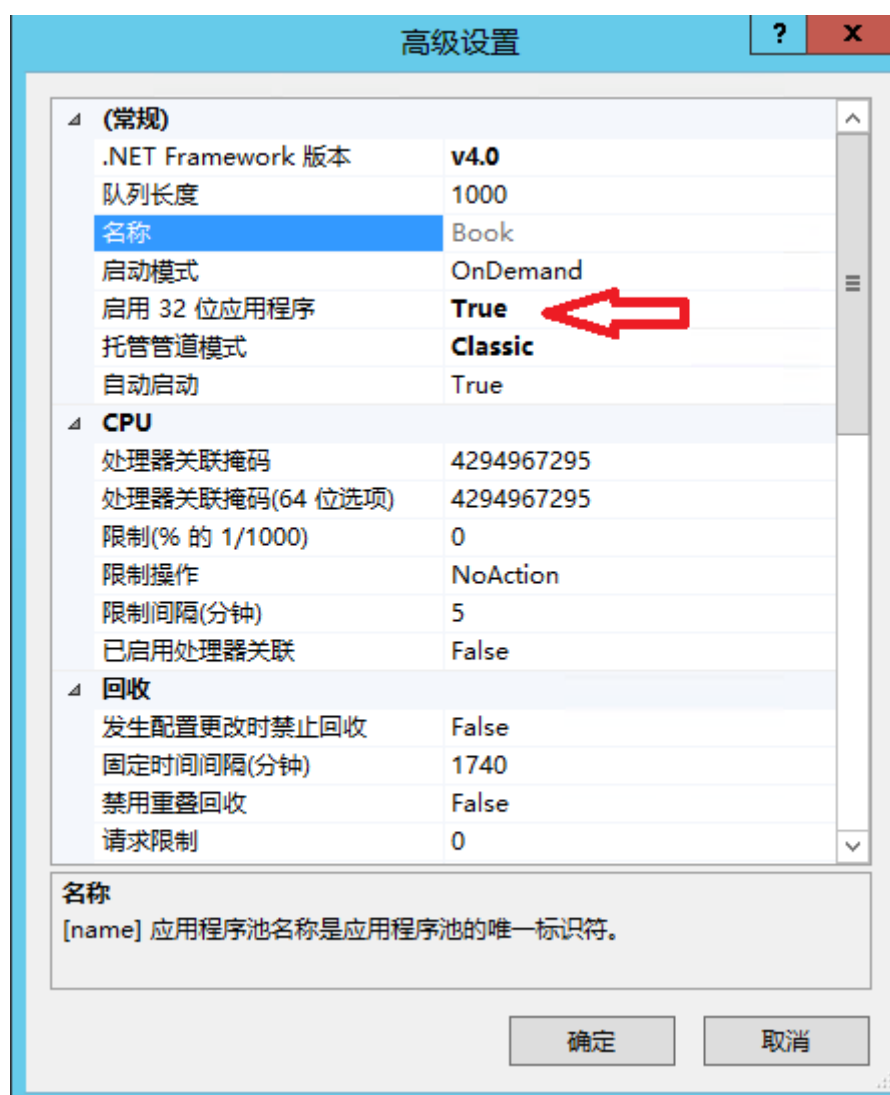


图 5-8 部署 Book API (2)

5.5 搭建 FTP 站点

- 1) 进入 Internet 信息服务管理器,在网站选项上右键功能键, 选择添加 FTP 站点

2) 添加对应的站点信息，如图 5-9 所示

添加 FTP 站点

站点信息

FTP 站点名称(I):
BookFTP

内容目录
物理路径(H):
D:\wwwroot\BookFTP ...

上一步(B) 下一步(N) 完成(F) 取消

图 5-9 搭建 FTP 站点（1）

添加 FTP 站点

绑定和 SSL 设置

绑定
IP 地址(A): 全部未分配 端口(O): 21
☐ 启用虚拟主机名(E):
虚拟主机(示例: ftp.contoso.com)(H):

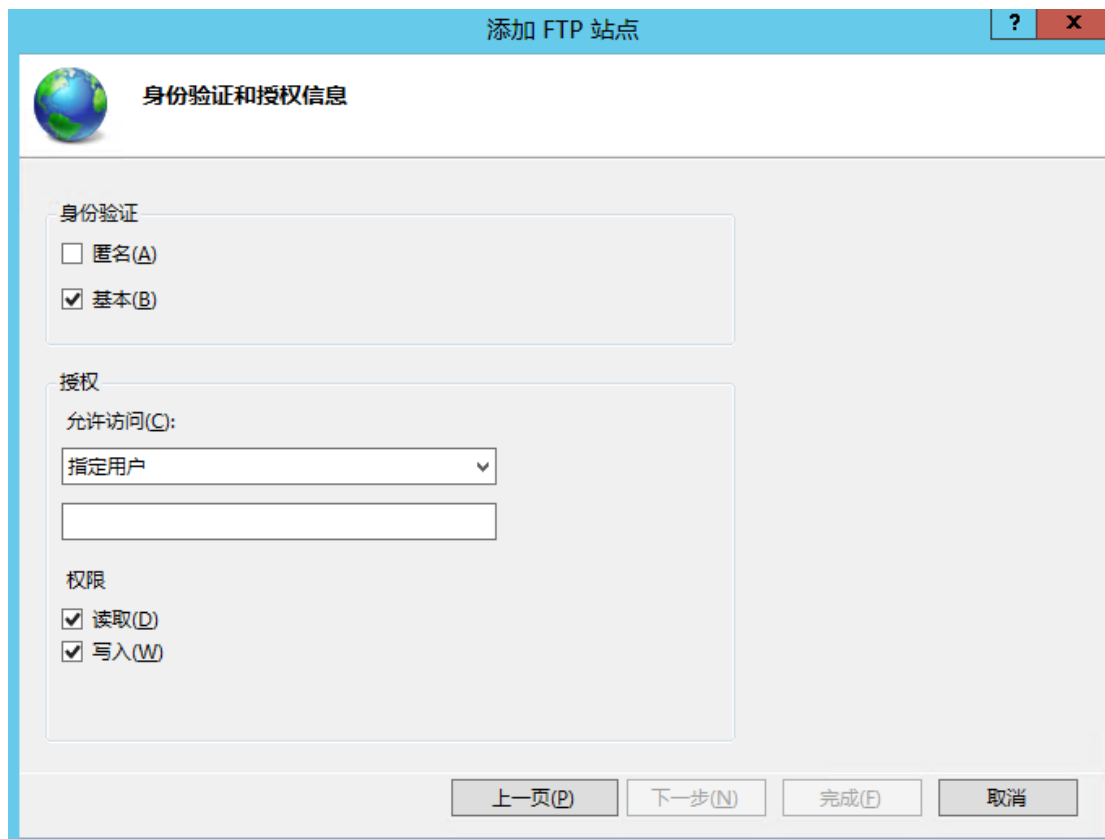
☒ 自动启动 FTP 站点(T)

SSL
☒ 无 SSL(L)
☐ 允许 SSL(W)
☐ 需要 SSL(R)
SSL 证书(C): 未选定 选择(S)... 查看(V)...

上一步(P) 下一步(N) 完成(F) 取消

图 5-10 搭建 FTP 站点（2）

- 3) 绑定 IP 地址和端口，IP 不填默认为当前服务器 IP，端口号默认为 21
- 4) 不需要 ssl 证书，选择无 ssl，如图 5-10 所示
- 5) 身份验证设置为基本，授权指定用户有读取和写入权限，如图 5-11 所示



The screenshot shows a Windows-style dialog box titled "添加 FTP 站点" (Add FTP Site). The main content area is titled "身份验证和授权信息" (Authentication and Authorization Information) and features a globe icon. It is divided into three sections: "身份验证" (Authentication) with radio buttons for "匿名(A)" (Anonymous) and "基本(B)" (Basic), where "基本(B)" is selected; "授权" (Authorization) with a label "允许访问(C):" (Allow access to:) and a dropdown menu currently showing "指定用户" (Specify user); and "权限" (Permissions) with checked checkboxes for "读取(D)" (Read) and "写入(W)" (Write). At the bottom, there are four buttons: "上一页(P)" (Previous page), "下一步(N)" (Next page), "完成(F)" (Finish), and "取消" (Cancel).

图 5-11 搭建 FTP 站点（3）

6 常见问题及其解决方案

6.1 数据库文件被进程占用无法移动以及挂起状态的恢复

- 1) 打开 SQL Server 配置管理器，停止 SQL Server (SQLEXPRESS) 服务
- 2) 在 DBMS 数据库中目录下删除已挂起数据库
- 3) 重新附加数据库

6.2 用户 sa 登录失败

- 1) 选中服务器(右键)->属性->安全性->服务器身份验证修改为"SQL SERVER 和 WINDOWS 身份验证模式"
- 2) 其次展开服务器下面的"安全性"文件夹->登陆名->(右键)SA，展开其“属性”窗口->状态->登陆修改为“启用”
- 3) 在“开始”菜单中找到 Microsoft SQL Server2005->配置工具->SQL Server 外围应用配置器，点击“服务和连接的外围应用配置器”，在弹出的窗口中找到自己当前要登录的服务器，停止启动（即重启一次服务器），才可生效 sa 账户登录。

6.3 图像识别时缺少 opencvcore290.dll

- 1) 下载 emgucv 注意其版本（在 shourceforge 中的 emgucv2.9.0.1922 资源在 2.4.9-beta 下）
- 2) 安装
- 3) 将 bin 中 x86 文件夹复制到项目的 debug 文件夹下