重要提示引擎3.17没问题，底版本要做处理

 param中有中文会编码出问题，所以在接入相册上传时，路径包含中文都会出问题。 解决方法： 1. java层，传入参数时 URLEncoder.encode(XXX, "UTF-8")  
   上传时拿到参数 URLDecoder.decode(XXX, "UTF-8")  
2. 简单粗暴，直接在java层声明变量保存

iOS接入

1：OC

Info.plist 中 dict列表内调用相册权限打开，添加如下信息

 <key>NSPhotoLibraryUsageDescription</key>  
    <string>App需获取调用系统相册的权限</string>

2：AppController.h 中

导入打开相册需要的类

#pragma mark - 01.使用相机相册要导入头文件的

#import <UIKit/UIKit.h> //如果这个已经导入请不要重复导入

#import <AVFoundation/AVFoundation.h>

#import <Photos/Photos.h>

Interface继承下面两个接口

, @interface AppController : NSObject <UIApplicationDelegate,UIAlertViewDelegate,WXApiDelegate,LBSDKApiDelegate, AMapLocationManagerDelegate ,XGPushDelegate, DuoLiaoDelegate,WKUIDelegate,WKNavigationDelegate,

UIImagePickerControllerDelegate,UINavigationControllerDelegate>

声明需要的变量和控件

@property (nonatomic, strong) NSString \* tokenvalue;

@property (nonatomic, strong) NSString \* uploadimgUrl;

@property(strong, nonatomic)UIImagePickerController \*\_imagePickerG;

3：在AppController.mm 中 添加方法

+(void)imageViewIsSelector:(NSString\* )token uploadimgUrl:(NSString\*)uploadimgUrlP//打开相册选择图片

{

appInstance.tokenvalue = token;

appInstance.uploadimgUrl = uploadimgUrlP;

[appInstance persentImagePicker];

}

+ (UIImage\*)compressImage:(UIImage\*)image scaledToMaxSideLength:(int)length {

CGSize size;

if (image.size.width >= image.size.height) {

size.width = length;

size.height = (int)(length / (double)image.size.width \* image.size.height);

} else {

size.height = length;

size.width = (int)(length / (double)image.size.height \* image.size.width);

}

UIGraphicsBeginImageContext(size);

[image drawInRect:CGRectMake(0, 0, size.width, size.height)];

UIImage\* newImage = UIGraphicsGetImageFromCurrentImageContext();

UIGraphicsEndImageContext();

return newImage;

}

///调用本地相册

- (void)persentImagePicker{

if (!self.\_imagePickerG) {

///初始化相机

self.\_imagePickerG = [[UIImagePickerController alloc]init];

///代理

self.\_imagePickerG.delegate = self;

}

///相册

self.\_imagePickerG.sourceType = UIImagePickerControllerSourceTypePhotoLibrary;

self.\_imagePickerG.allowsEditing = YES;

[viewController presentViewController:self.\_imagePickerG animated:YES completion:nil];

}

-(UIImage \*)convertImage:(UIImage \*)origImage scope:(CGFloat)scope

{

UIImage \*image = nil;

CGSize size = origImage.size;

if (size.width < scope && size.height < scope) {

// do nothing

image = origImage;

} else {

CGFloat length = size.width;

if (size.width < size.height) {

length = size.height;

}

CGFloat f = scope/length;

NSInteger nw = size.width \* f;

NSInteger nh = size.height \* f;

if (nw > scope) {

nw = scope;

}

if (nh > scope) {

nh = scope;

}

CGSize newSize = CGSizeMake(nw, nh);

// CGSize newSize = CGSizeMake(size.width\*f, size.height\*f);

//

UIGraphicsBeginImageContext(newSize);

//UIGraphicsBeginImageContextWithOptions(newSize, NO, 0.0f);

// Tell the old image to draw in this new context, with the desired

// new size

[origImage drawInRect:CGRectMake(0, 0, newSize.width, newSize.height)];

// Get the new image from the context

image = UIGraphicsGetImageFromCurrentImageContext();

UIGraphicsEndImageContext();

}

return image;

}

//压缩图片

- (NSData\*) compressImage:(UIImage\*)originImage PixelCompress:(BOOL)pc MaxPixel:(CGFloat)maxPixel JPEGCompress:(BOOL)jc MaxSize\_KB: (CGFloat)maxKB

{

/\*

压缩策略： 支持最多921600个像素点

像素压缩：（调整像素点个数）

当图片长宽比小于3:1 or 1:3时，图片长和宽最多为maxPixel像素；

当图片长宽比在3:1 和 1:3之间时，会保证图片像素压缩到921600像素以内；

JPEG压缩：（调整每个像素点的存储体积）

默认压缩比0.99;

如果压缩后的数据仍大于IMAGE\_MAX\_BYTES，那么将调整压缩比将图片压缩至IMAGE\_MAX\_BYTES以下。

策略调整：

不进行像素压缩，或者增大maxPixel，像素损失越小。

使用无损压缩，或者增大IMAGE\_MAX\_BYTES.

注意：

jepg压缩比为0.99时，图像体积就能压缩到原来的0.40倍了。

\*/

UIImage \* scopedImage = nil;

NSData \* data = nil;

//CGFloat maxbytes = maxKB \* 1024;

if (originImage == nil) {

return nil;

}

if ( pc == YES ) { //像素压缩

// 像素数最多为maxPixel\*maxPixel个

CGFloat photoRatio = originImage.size.height / originImage.size.width;

if ( photoRatio < 0.3333f )

{ //解决宽长图的问题

CGFloat FinalWidth = sqrt ( maxPixel\*maxPixel/photoRatio );

scopedImage = [self convertImage:originImage scope:MAX(FinalWidth, maxPixel)];

}

else if ( photoRatio <= 3.0f )

{ //解决高长图问题

scopedImage = [self convertImage:originImage scope: maxPixel];

}

else

{ //一般图片

CGFloat FinalHeight = sqrt ( maxPixel\*maxPixel\*photoRatio );

scopedImage = [self convertImage:originImage scope:MAX(FinalHeight, maxPixel)];

}

}

else { //不进行像素压缩

scopedImage = originImage;

}

[scopedImage retain];

// NSData \*imageData = UIImageJPEGRepresentation(image, compression);

// while ([imageData length] > kb && compression > maxCompression) {

// compression -= 0.1;

// imageData = UIImageJPEGRepresentation(image, compression);

// }

// CGFloat compression = 0.6f;

//CGFloat maxCompression = 0.1f;

if ( jc == YES ) { //JPEG压缩

data = UIImageJPEGRepresentation(scopedImage, 0.6);

// while ([data length] > maxKB && compression > maxCompression) {

// compression -= 0.01;

// data = UIImageJPEGRepresentation(scopedImage, compression);

// }

NSLog(@"data compress with ratio (0.6) : %d KB", data.length/1024);

}

else {

data = UIImageJPEGRepresentation(scopedImage, 1.0);

NSLog(@"data compress : %d KB", data.length/1024);

}

[scopedImage release];

return data;

}

///选择图片完成（从相册或者拍照完成）

- (void)imagePickerController:(UIImagePickerController \*)picker didFinishPickingMediaWithInfo:(NSDictionary<NSString \*,id> \*)info{

//获取修剪后的图片

NSString \*imageUp = [info objectForKey:UIImagePickerControllerImageURL];

NSString \*imageUpURL = [info objectForKey:@"PUPhotoPickerOriginalImagePath"];

UIImage \*imageR = [info objectForKey:@"UIImagePickerControllerOriginalImage"];

NSData \*dataOld = [NSData dataWithContentsOfFile:imageUp];

NSData \*dataE = [appInstance compressImage:imageR PixelCompress:YES MaxPixel:921600 JPEGCompress:YES MaxSize\_KB:1000];

NSInteger lenOld = dataOld.length/1024;

NSInteger len = dataE.length/1024;

NSLog(@"原来图片的大小是%ldKB",lenOld);

NSLog(@"压缩原来图片的大小是%ldKB",len);

if(len<1024){

//--------上传逻辑开始

NSString \*filenameexestr = @"";

NSString \*filename = @"";

NSString \*extension = @"";

// 从路径中获得完整的文件名（带后缀）

filenameexestr = [imageUpURL lastPathComponent];

// 获得文件名（不带后缀）

filename = [imageUpURL stringByDeletingPathExtension];

// 获得文件的后缀名（不带'.'）

extension = [imageUpURL pathExtension];

//设置mimeType

NSString \*mimeType = @"image/jpeg";

NSString \*TWITTERFON\_FORM\_BOUNDARY = @"AaB03x";

//根据url初始化request

NSString\* URL = appInstance.uploadimgUrl;

NSLog(@"上传URL: %@", URL);

// NSString\* URL = @"http://10.1.6.94:8080/weixin/uploadimg";

NSMutableURLRequest\* request = [NSMutableURLRequest requestWithURL:[NSURL URLWithString:URL]

cachePolicy:NSURLRequestReloadIgnoringLocalCacheData

timeoutInterval:10];

[request setValue:appInstance.tokenvalue forHTTPHeaderField:@"authorization"];

NSLog(@"%@", request.allHTTPHeaderFields); //打印出header验证

//分界线 --AaB03x

NSString \*MPboundary=[[NSString alloc]initWithFormat:@"--%@",TWITTERFON\_FORM\_BOUNDARY];

//结束符 AaB03x--

NSString \*endMPboundary=[[NSString alloc]initWithFormat:@"%@--",MPboundary];

//要上传的文件

NSData \*data = [NSData dataWithContentsOfFile:imageUp];

//NSInteger len1 = data.length/1024/1024;

//NSLog(@"原来图片的大小是%ldMB",len1);

//http body的字符串

NSMutableString \*body=[[NSMutableString alloc]init];

////添加分界线，换行

[body appendFormat:@"%@\r\n",MPboundary];

//声明文件字段，文件名

[body appendFormat:@"Content-Disposition: form-data; name=\"cardimg\"; filename=\"%@\"\r\n",filenameexestr];

//声明上传文件的格式

[body appendFormat:@"Content-Type: %@\r\n\r\n",mimeType];

//声明结束符：--AaB03x--

NSString \*end=[[NSString alloc]initWithFormat:@"\r\n%@",endMPboundary];

//声明myRequestData，用来放入http body

NSMutableData \*myRequestData=[NSMutableData data];

//将body字符串转化为UTF8格式的二进制

[myRequestData appendData:[body dataUsingEncoding:NSUTF8StringEncoding]];

//将file的data加入

[myRequestData appendData:dataE];

//加入结束符--AaB03x--

[myRequestData appendData:[end dataUsingEncoding:NSUTF8StringEncoding]];

//设置HTTPHeader中Content-Type的值

NSString \*content=[[NSString alloc]initWithFormat:@"multipart/form-data; boundary=%@",TWITTERFON\_FORM\_BOUNDARY];

//设置HTTPHeader

[request setValue:content forHTTPHeaderField:@"Content-Type"];

[request setValue:[NSString stringWithFormat:@"%d", [myRequestData length]] forHTTPHeaderField:@"Content-Length"];

//设置http body

[request setHTTPBody:myRequestData];

//http method

[request setHTTPMethod:@"POST"];

NSData \*returnData = [NSURLConnection sendSynchronousRequest:request returningResponse:nil error:nil];

NSString \*returnString = [[NSString alloc] initWithData:returnData encoding:NSUTF8StringEncoding];

NSLog(@"上传状态返回值returnString: %@", returnString);

std::string event = "GET\_PHOTO\_UPLOADPIC";

std::string funName = "cc.eventManager.dispatchCustomEvent";

std::string result\_c\_str= [returnString cStringUsingEncoding: NSUTF8StringEncoding];

//{"errno":0,"payimg":"http://cardimg888.oss-cn-hangzhou.aliyuncs.com/xynmmj\_686914\_1577949909032.JPG"}

//errno 值为0说明成功了

std::string rStr = funName + "(\"" + event + "\"," + result\_c\_str + ");";

[picker dismissViewControllerAnimated:YES completion:nil];

ScriptingCore::getInstance()->evalString(rStr.c\_str());

}

else

{

//图片超过1M 的处理

NSString \*result = @"{\"errno\":-1,\"payimg\":\"-1\"}";

std::string result\_c\_str= [result cStringUsingEncoding: NSUTF8StringEncoding];

std::string event ="PIC\_SIZE\_WARNING";

std::string funName ="cc.eventManager.dispatchCustomEvent";

std::string rStr = funName + "(\"" + event + "\"," + result\_c\_str + ");";

[picker dismissViewControllerAnimated:YES completion:nil];

ScriptingCore::getInstance()->evalString(rStr.c\_str()); }

//---------------------------------上传逻辑结束

}

//取消选择图片（拍照）

- (void)imagePickerControllerDidCancel:(UIImagePickerController \*)picker{

[picker dismissViewControllerAnimated:YES completion:nil];

}

//相册调用end

上传成功后 后台会通知 updateInfo 注册的方法

imageViewIsSelector 方法是吊起相册逻辑方法

GET\_PHOTO\_UPLOADPIC 是上传成功调用js端注册方法

GET\_PHOTO\_UPLOADPIC: function (d) {

**if** (cc.sys.OS\_ANDROID == cc.sys.os) {  
 d = JSON.parse(d)  
}  
**if** (cc.sys.OS\_IOS == cc.sys.os) {  
 d = JSON.parse(JSON.stringify(d))  
}  
cc.log("-- getPictureFromPhone -- GET\_PHOTO\_UPLOADPIC -- d.errno = "+d.errno)  
**if** (d && parseInt(d.errno) == 0) { //成功  
 cc.log("-- getPictureFromPhone -- GET\_PHOTO\_UPLOADPIC -- d.errno222 = "+d.errno)  
 jsclient.showMsg("图片上传成功！")  
 // toastMsg("图片上传成功！")  
 **this**.refresh(d.payimg, photoPath)  
} **else** { //失败  
 cc.log("-- getPictureFromPhone -- GET\_PHOTO\_UPLOADPIC -- d.errno111 = "+d.errno)  
 // toastMsg("图片上传失败，请重试！")  
 jsclient.showMsg("图片上传失败，请重试！")  
}

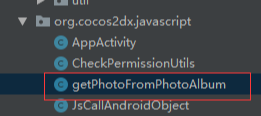
}

PIC\_SIZE\_WARNING 图片体积过大调用js端方法

PIC\_SIZE\_WARNING:**function**()  
{  
 jsclient.showMsg("图片较大！请上传1M以下图片，谢谢")  
}

Android接入

1. 添加文件getPhotoFromPhotoAlbum.java



1. AndroidManifest.xml 内操作

<!--读写内存块权限-->

<uses-permission android:name="android.permission. READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

1. AppActivity.java中 添加调用相册方法

// 调用相册获取对应的图片public static void getPictureFromPhoneAlbum() { Log.i("HelloOC","getPictureFromPhoneAlbum"); if (ccActivity != null) { ccActivity.goPhotoAlbum(); }}public void goPhotoAlbum() { Intent intent = new Intent(); intent.setAction(Intent.ACTION\_PICK); intent.setType("image/\*"); startActivityForResult(intent, 2);}

1. AppActivity.java onActivityResult中 添加获取图片返回结果（else if 中为新添加）

protected void onActivityResult(int requestCode, int resultCode, Intent data) { ……

//获取多聊授权登录成功后返回的数据 if (requestCode == Constant.THIRD\_LOGIN) {

…………

} else if(requestCode == 2 && resultCode == RESULT\_OK) { // 获取照片返回结果 Log.e("HelloOC:", "getPictureFromPhone -- 获取照片返回结果 data.getData() = " + data.getData()); String photoPath = getPhotoFromPhotoAlbum.getRealPathFromUri(this,data.getData()); Log.e("HelloOC:", "getPictureFromPhone -- photoPath::: " + photoPath); JSONObject result = new JSONObject(); try{ result.put("photoPath", photoPath); } catch (JSONException e) { e.printStackTrace(); } RunJS\_obj("GET\_PHOTO\_FROM\_ALBUM", result.toString()); }}

1. AppActivity.java中 上传图片至服务器方法

public static void uploadPic(final String filePath, final String actionUrlPar ,final String eventName, final String token) {

new Thread() {

public void run() {

httpClient http = new httpClient(filePath,actionUrlPar,eventName,token);

http.uploadFilePic();

Log.i("send:", "send successful");

Log.i("send:", "send successful http.ok = " + http.ok);

if (http.ok == 1) {

ccActivity.RunJS(eventName, http.uploadPicReturnData);

Log.i("result 333", "getPictureFromPhone -- result hhhhh = " + http.uploadPicReturnData);

}

else if(http.ok == 0)

{

Log.i("result 444", "getPictureFromPhone -- result faild ");

JSONObject result = new JSONObject();

try{

result.put("errno", 1);

} catch (JSONException e) {

e.printStackTrace();

}

ccActivity.RunJS(eventName, result.toString());

}

}

}.start();

}

4． httpClient.java中 添加

private String token = "";

public String uploadPicReturnData;

添加httpClient方法

public httpClient(final String filePath, final String url, final String fileName, final String tokenPar)

{

path = filePath;

srcPath = fileName;

actionUrl = url;

token = tokenPar;

}

添加uploadFilePic方法

/\* 上传图片至Server的方法 \*/

public void uploadFilePic() {

File f= new File(path);

if (f.exists() && f.isFile()) {

Boolean BL = (f.length()/1024)<1024;

if(BL)

{

}

else

{

ok = 2;

AppActivity.ccActivity.RunJS("PIC\_SIZE\_WARNING", "");

return;

}

}

uploadPicReturnData = "";

String uploadUrl = actionUrl;

String end = "\r\n";

String twoHyphens = "--";

String boundary = "\*\*\*\*\*\*";

try {

URL url = new URL(uploadUrl);

HttpURLConnection httpURLConnection = (HttpURLConnection) url

.openConnection();

httpURLConnection.setDoInput(true);

httpURLConnection.setDoOutput(true);

httpURLConnection.setUseCaches(false);

httpURLConnection.setRequestMethod("POST");

httpURLConnection.setRequestProperty("Connection", "Keep-Alive");

httpURLConnection.setRequestProperty("Charset", "UTF-8");

httpURLConnection.setRequestProperty("Content-Type",

"multipart/form-data;boundary=" + boundary);

httpURLConnection.setRequestProperty("authorization", token);

DataOutputStream dos = new DataOutputStream(httpURLConnection.getOutputStream());

dos.writeBytes(twoHyphens + boundary + end);

dos.writeBytes("Content-Disposition: form-data; name=\"cardimg\"; filename=\""

+ path.substring(path.lastIndexOf("/") + 1)

+ "\"" + end);

dos.writeBytes("Content-Type:image/jpeg"+ end);

dos.writeBytes(end);

//将SD 文件通过输入流读到Java代码中-++++++++++++++++++++++++++++++`````````````````````````

FileInputStream fis = new FileInputStream(path);

byte[] buffer = new byte[1024 \* 1024]; // 1024k

int count = 0;

while ((count = fis.read(buffer)) != -1) {

dos.write(buffer, 0, count);

}

fis.close();

System.out.println("file send to server............");

Log.e("ShowLogOnJava:", "file send to server............");

Log.i("result", "getPictureFromPhone --------");

dos.writeBytes(end);

dos.writeBytes(twoHyphens + boundary + twoHyphens + end);

dos.flush();

dos.close();

//读取服务器返回结果

String res = "";

Log.i("result", "getPictureFromPhone -------- 读取服务器返回结果 ");

// 读取返回数据

StringBuffer strBuf = new StringBuffer();

InputStreamReader isr = new InputStreamReader(

httpURLConnection.getInputStream());

BufferedReader reader = new BufferedReader(isr);

String line = null;

Log.i("result", "getPictureFromPhone -------- 读取服务器返回结果1111 ");

while ((line = reader.readLine()) != null) {

strBuf.append(line).append("\n");

}

res = strBuf.toString();

reader.close();

reader = null;

isr.close();

isr = null;

ok = 1;

JSONObject resData = new JSONObject(res);

Log.i("result", "getPictureFromPhone -- result00 = " + res);

uploadPicReturnData = resData.toString();

Log.e("自定义的json", resData.toString());

} catch (Exception e) {

e.printStackTrace();

ok = 0;

}

}

Js接入

1. getPictureFromPhoneAlbum方法是吊起安卓相册逻辑方法

jsclient.native中添加

getPictureFromPhoneAlbum:function (url, token) {  
 try {  
 if(cc.**sys**.OS\_ANDROID === cc.**sys**.**os**) {  
 jsb.reflection.callStaticMethod("org.cocos2dx.javascript.AppActivity", "getPictureFromPhoneAlbum", '()V')  
 }  
 else if(cc.**sys**.OS\_IOS === cc.**sys**.**os**) {  
 jsb.reflection.callStaticMethod('AppController', 'imageViewIsSelector:uploadimgUrl:', token, url)  
 }  
 }  
 catch (e) {  
 jsclient.**native**.ShowLogOnJava("jsclient.native.getPictureFromPhoneAlbum throw: " + JSON.stringify(e))  
 }  
}

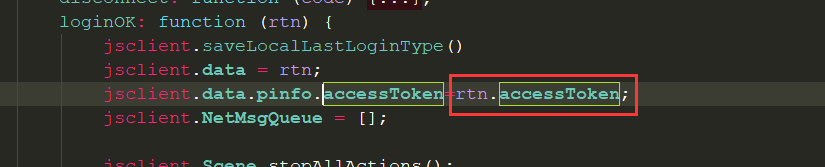
1. GET\_PHOTO\_FROM\_ALBUM 是获取照片成功后调用js端方法

uploadPic 是上传图片至服务器调用的方法

GET\_PHOTO\_FROM\_ALBUM:function(d) {   
 jsclient.**native**.ShowLogOnJava("-- getPictureFromPhone -- d = " + JSON.stringify(d))  
 jsclient.block()  
 jsclient.**native**.uploadPic(d.photoPath, uploadUrl[photoBtnId-1], "GET\_PHOTO\_UPLOADPIC", jsclient.**data**.**pinfo**.**accessToken**)  
},

// fullFileName: 照片本地路径 url: 上传图片的url eventName: 上传成功后的回调函数名 token: 登录码（登录游戏时，服务器端返回的accessToken）

uploadPic: function (fullFileName, url, eventName,token) {  
 try {  
 if (cc.**sys**.OS\_ANDROID == cc.**sys**.**os**) {  
 jsb.reflection.callStaticMethod("org.cocos2dx.javascript.AppActivity", "uploadPic", "(Ljava/lang/String;Ljava/lang/String;Ljava/lang/String;Ljava/lang/String;)V", String(fullFileName), String(url), String(eventName), String(token));  
 } else if (cc.**sys**.OS\_IOS == cc.**sys**.**os**) {  
 jsb.reflection.callStaticMethod("AppController", "uploadPic:url:eventName:token:", String(fullFileName), String(url), String(eventName), String(token));  
 }  
 } catch (e) {  
 jsclient.**native**.HelloOC("UploadFile throw: " + JSON.stringify(e));  
 }  
},



GET\_PHOTO\_UPLOADPIC 上传成功后 吊起js层方法（同上iOS）

PIC\_SIZE\_WARNING 图片体积过大调用js端方法（同上iOS）

上传成功后 后台发送的通知updateInfo (两张图的url分别存放在payimg和payimg1) （newUserInfoLayer.js文件中）

updateInfo: function () {  
 if (jsclient.**data**.**pinfo**.payimg) {  
 this.refresh(jsclient.**data**.**pinfo**.payimg) //通过url下载图片显示出来  
 }  
}

打开个人信息界面（NewUserInfoLayer）时，如果已经上传过图片，url会存放在pinfo. payimg和payimg1中

在\_run函数中，可以通过jsclient.data.pinfo.payimg和payimg1的url来下载图片显示图片

所有js层添加的代码 都在app.js和newUserInfoLayer.js中，具体代码，请自行查找

