# 返回结果规则约定描述

返回字串为json字串（包含flag ，msg ,totalSize,rs对象）

其中flag msg为必备返回字段；totalSize在需要分页的地方需要返回

flag标示请求成功与否的状态（flag=0代表请求失败 flag为1代表请求成功）

flag=1状态下 如果需要返回rs数据对象

1：rs对象为数组 返回示例 rs:[{},{}]

2：rs对象为对象 返回示例rs:{}

## 登录

#### 登录app

#### URL

/login

#### 请求参数

|  |  |  |
| --- | --- | --- |
| **类型** | **说明** | |
| String | userName | 用户名 |
| String | passWord | 密码(需要进行sha1加密，[请看加密示例](#_密码加密示例)) |
| 示例 | userName=aaaaa&passWord=bbbbb | |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败（账号或密码错误）  1:成功 |
| msg | string | 提示信息 |
| rs | String | tokenId: 登录令牌 |
| 示例 | {  “flag”: 1,  “msg”:”查找成功”,  “rs”: {  "tokenId":"58cbf9b9-a4c9-4199-9c3d-6ab8d205dfb0"  }  } | |

## 修改密码

#### 修改密码

#### URL

/editpwd

#### 请求参数

|  |  |  |
| --- | --- | --- |
| **类型** | **说明** | |
| String | tokenId | 用户登录令牌 |
| String | oldPwd | 旧密码(需要进行sha1加密，[请看加密示例](#_密码加密示例)) |
| String | newPwd | 新密码(需要进行sha1加密，[请看加密示例](#_密码加密示例)) |
|  |  | |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败（账号或密码错误）  1:成功 |
| msg | string | 提示信息 |

## 获取服务器时间

#### 获取服务器时间

#### URL

/systemTime

#### 请求参数

|  |  |  |
| --- | --- | --- |
| **类型** | **说明** | |
| String | tokenId | 用户登录令牌 |
|  |  | |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| rs | Long | systemTime: 服务器时间戳 |
| 示例 | {  “flag”: 1,  “msg”:”查找成功”,  “rs”: {  "systemTime":12333365412  }  } | |

## 查询检查项列表

#### 功能描述

查询检查项列表

#### URL

/checkList

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| rs集合 | int | id: 检查项唯一id |
| string | itemTitle:检查项标题 |
| string | itemContent: 检查项内容 |
| 示例 | {  “flag”: 1,  “msg”:”查找成功”,  “totalSize”:2,  “rs”: [{  " id":1,  " itemTitle":"检查项标题1",  "checkContent":"检查项内容1"  },  {  " id":2,  " itemTitle":"检查项标题2",  " itemContent":"检查项内容2"  }]  } | |
|  |  | |

## 签到及保存检查结果

#### 功能描述

签到保存检查结果

#### URL

/saveCheck

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |
| **String** | **cpName：被检查企业名称** |
| **Long** | **checkTime:检查时间戳** |
| String | position:坐标 | 签到地点坐标 |
| **String** | **checkIds：id间以英文逗号隔开，如: 1,2,3,4** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| rs | int | taskId:任务id |
| 示例 | {  “flag”: 1,  “msg”:”成功”,  “rs”: {  "taskId":12  }  } | |

## 执法文书拍照

#### 功能描述

执法文书拍照

#### URL

/intendance

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |
| **String** | **inCpName：被执法企业名称** |
| **Long** | **inTime:被执法时间戳** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| rs | int | taskId:任务id |
| 示例 | {  “flag”: 1,  “msg”:”成功”,  “rs”: {  "taskId":12  }  } | |

## 培训记录查询

#### 功能描述

培训记录查询

#### URL

/trainRecord

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |
| **String** | **certNum:证书编号** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功  3000:记录不存在 |
| msg | string | 提示信息 |
| rs | String | name:姓名 |
| String | image:头像地址 |
| string | **certNum:**证书编号 |
| String | sex:**性别:** |
| String | idCard:身份证号码 |
| String | workType:作业类别名称 |
| String | canWorkType:准操项目 |
| String | workCp:工作单位 |
| String | trainCp:培训单位 |
| String | certCp:发证机关 |
| String | certTime: 领证日期时间戳 |
| String | effectTimeStart：开始有效期时间戳 |
| String | effectTimeEnd：结束有效期时间戳 |
| String | workScore:实作成绩 |
| String | theoryScore:理论成绩 |
| 示例 | Rs是个对象，这里省略n多字 | |

## 培训记录列表查询

#### 功能描述

培训记录查询

#### URL

/ trainRecordSimple

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |
| **String** | **idCard：身份证号码** |
| **String** | **certNum:证书编号** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| rs | String | name:姓名 |
| string | **certNum:**证书编号 |
| String | canWorkType:准操项目 |
| 示例 | Rs是个列表，这里省略n多字 | |

## 查询新闻列表

#### 功能描述

新闻列表查询

#### URL

/newsList

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |
| **String** | **searchStr：查询字符串 如果为空则查询全部** |
| int | pageNo 分页码 从0开始，即0就是第一页 1就是第二页 |
| int | pageSize 每页显示数量 |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| totalSize | int | 总记录数 |
| rs | int | id:新闻唯一id |
| String | noticeTitle:标题 |
| Date | addTime**:**新闻发布时间戳 |
| int | status:**状态 1顶置 0非顶置** |
| 示例 | Rs是个数组列表，这里省略n多字 | |

## 查询新闻内容

#### 功能描述

新闻列表查询

#### URL

/newsContent

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |
| **Int** | **newsId:新闻唯一id** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| rs | String | noticeTitle:标题 |
| Date | addTime:新闻发布时间戳 |
| int | status:状态 2顶置 1非顶置 |
| String | noticeContent:新闻内容 |
| 示例 | Rs是个对象，这里省略n多字 | |

## 上传图片([示例代码在本文档最后](#_上传图片示例代码(此代码已与服务器联调过)))

#### 功能描述

上传图片

#### URL

/uploadFiles

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **String** | **tokenId:登录时获取的令牌** |
| **List<File>** | **images:图片列表** |
| **int** | **taskId 任务id （相应接口返回的任务id）** |
| **int** | **type 1签到或执法的图片 2 检查拍照取证图片(该类型只有在检查接口中上传图片时有用到) 3 负责人签字图片(该类型只有在检查接口中上传图片时有用到)** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |

## 上传图片流程

调用相应需要传输图片的接口----》获得任务id----》调用上传图片接口，一张一张的上传图片

## 上传图片示例代码(此代码已与服务器联调过)

此处借用的是httpclient公共库。需要导入以下几个jar包

jar包会在邮件附件中添加:

**package** com.sr178.safecheck.app.service;

**import** java.io.File;

**import** java.io.IOException;

**import** org.apache.http.HttpResponse;

**import** org.apache.http.HttpStatus;

**import** org.apache.http.client.ClientProtocolException;

**import** org.apache.http.client.HttpClient;

**import** org.apache.http.client.methods.HttpPost;

**import** org.apache.http.entity.mime.HttpMultipartMode;

**import** org.apache.http.entity.mime.MultipartEntity;

**import** org.apache.http.entity.mime.content.FileBody;

**import** org.apache.http.entity.mime.content.StringBody;

**import** org.apache.http.impl.client.DefaultHttpClient;

**import** org.apache.http.util.EntityUtils;

**import** com.sr178.game.framework.log.LogSystem;

**public** **class** UploadClientTest {

**public** **static** **void** main(String[] args) **throws** ClientProtocolException, IOException {

String url="http://localhost:8083/uploadFiles";

HttpClient httpclient= **new** DefaultHttpClient();

HttpPost httpPost= **new** HttpPost(url);

MultipartEntity mulentity = **new** MultipartEntity(HttpMultipartMode.***BROWSER\_COMPATIBLE***);

mulentity.addPart("tokenId", **new** StringBody("bc0f2c68-0640-45aa-9cfd-c447b7914903"));

mulentity.addPart("taskId", **new** StringBody("4"));

mulentity.addPart("type", **new** StringBody("2"));

File image1 = **new** File("E:\\图片\\58-110R31U14786.jpg");

File image2 = **new** File("E:\\图片\\11.jpg");

//添加图片表单数据

FileBody filebody1 = **new** FileBody(image1);

FileBody filebody2 = **new** FileBody(image2);

mulentity.addPart("images",filebody1);

mulentity.addPart("images",filebody2);

httpPost.setEntity(mulentity);

HttpResponse response = httpclient.execute(httpPost);

**if**(response.getStatusLine().getStatusCode()==HttpStatus.***SC\_OK***)

{

LogSystem.*info*("上传返回结果为！！~~result="+EntityUtils.*toString*(response.getEntity(), "utf-8"));

}

**else**

{

LogSystem.*info*("上传失败");

}

}

}

## 密码加密示例

**算法类：**

package com.sr178.safecheck.common.utils;

import javax.crypto.Mac;

import javax.crypto.SecretKey;

import javax.crypto.spec.SecretKeySpec;

import org.apache.commons.codec.binary.Base64;

import com.sr178.game.framework.log.LogSystem;

import com.sr178.safecheck.common.exception.ServiceException;

public class MacShaUtils {

private static final String MAC\_NAME = "HmacSHA1";

private static final String ENCODING = "UTF-8";

/\*\*

\* 使用 HMAC-SHA1 签名方法对对encryptText进行签名

\* @param encryptText 被签名的字符串

\* @param encryptKey 密钥

\* @return

\* @throws Exception

\*/

private static byte[] HmacSHA1Encrypt(String encryptText, String encryptKey) throws Exception

{

byte[] data=encryptKey.getBytes(ENCODING);

//根据给定的字节数组构造一个密钥,第二参数指定一个密钥算法的名称

SecretKey secretKey = new SecretKeySpec(data, MAC\_NAME);

//生成一个指定 Mac 算法 的 Mac 对象

Mac mac = Mac.getInstance(MAC\_NAME);

//用给定密钥初始化 Mac 对象

mac.init(secretKey);

byte[] text = encryptText.getBytes(ENCODING);

//完成 Mac 操作

return mac.doFinal(text);

}

/\*\*

\* 执行签名

\* @param encryptText

\* @param encryptKey

\* @return

\*/

public static String doEncryptBase64(String encryptText, String encryptKey){

try {

return Base64.encodeBase64String(HmacSHA1Encrypt(encryptText, encryptKey));

} catch (Exception e) {

LogSystem.error(e, "");

throw new ServiceException(3000, "加密失败");

}

}

}

**调用示例:**

private static final String SHA\_SECRET = "!@#asDFA55214644";（秘钥定义）

System.out.println( MacShaUtils.doEncryptBase64("xx", SHA\_SECRET));

## 调试相关信息账号

调试地址:http://117.172.163.97:60080/

账号名:xx\_down 密码:xx

如登录接口:

http://117.172.163.97:60080/login?userName=xx\_down&passWord=LpkBu6U7/DzcaD1DryBisxxpVBg=

## 版本检查

#### 功能描述

版本检查

#### URL

/ checkVersion

#### 请求参数

|  |  |
| --- | --- |
| **类型** | **说明** |
| **int** | **versionNum:当前版本号** |

#### 返回结果

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| flag | int | 状态标志  0:失败  1:成功 |
| msg | string | 提示信息 |
| rs | String | url:最新apk下载地址，如果为空则表示当前是最新版本，不需要进行更新 |
| 示例 |  | |