**“浙里办”单点登录服务端接入指南**

目录

[一、 概述 3](#_Toc25206)

[二、 接入前准备 3](#_Toc7601)

[三、 接入示例 3](#_Toc31343)

[四、 接口调用方式 4](#_Toc21588)

[1． 请求说明 4](#_Toc25591)

[2． 鉴权参数说明 4](#_Toc21516)

[3． 接口签名计算代码示例 5](#_Toc517)

[五、 接口说明 8](#_Toc11230)

[1． 基于单点登录票据换取请求token 8](#_Toc18355)

[(1) 请求地址 8](#_Toc18949)

[(2) 入参 8](#_Toc7116)

[(3) 出参 8](#_Toc19888)

[(4) 错误码 8](#_Toc28173)

[2． 基于token获取用户信息 9](#_Toc10184)

[(1) 请求地址 9](#_Toc13141)

[(2) 入参 9](#_Toc2786)

[(3) 出参 9](#_Toc26700)

[(4) 错误码 11](#_Toc27594)

# 概述

本指南旨在为“浙里办”单点登录组件提供接入指南，“浙里办”单点登陆组件，上架在IRS，为上架在IRS的应用，提供统一的单点登录解决方案，现阶段仅支持微信端的接入。

# 接入前准备

应用接入“浙里办”单点登录组件前，需要先获取AK&SK。



# 接入示例

单点登录的接入示例，可以参考“浙里办”单点登录接入demo，获取方式如图：



# 接口调用方式

## 请求说明

**请以POST方式提交请求，参数以application/json形式提交。**

## 鉴权参数说明

“浙里办”单点登录，HTTP请求都必须在请求头（HTTP Header）中设置如下4个参数：

| **参数名** | **是否必填** | **类型** | **说明** |
| --- | --- | --- | --- |
| **X-BG-HMAC-SIGNATURE** | 是 | string | API输入参数签名结果 |
| **X-BG-HMAC-ALGORITHM** | 是 | string | 签名的摘要算法，当前仅支持hmac-sha256。 |
| **X-BG-HMAC-ACCESS-KEY** | 是 | string | 分配给应用的accessKey，例如：12345678。 |
| **X-BG-DATE-TIME** | 是 | string | 时间戳，时区为GMT+8，格式为：Tue, 09 Nov 2021 08:49:20 GMT。服务端允许客户端请求最大时间误差为100秒。 |

其中X-BG-HMAC-SIGNATURE的计算公式为：

|  |
| --- |
| signature = HMAC-SHA256-HEX(secret\_key,signing\_string) |

各字段解释如下：

1. secret\_key为接口申请完成后获取到的secret\_key
2. signing\_string由请求方法、URI、请求参数等拼接获得，具体如下：

|  |
| --- |
| HTTP\_METHOD+\n+HTTP\_URI+\n+QUERY\_STREAM+\n+X-BG-HMAC-ACCESS-KEY+\n+X-BG-DATE+\n |

参数解释如下图，详细代码可参考签名计算代码

| **参数名** | **说明** |
| --- | --- |
| HTTP METHOD | 指 HTTP 协议中定义的 GET、PUT、POST 等请求方法，必须使用全大写的形式。 |
| HTTP URI | 请求路径，要求必须以“/”开头，不以“/”开头的需要补充上，空路径为“/” |
| X-BG-DATE | 请求头中的 Date （ GMT 格式 ）格式为：“Tue, 09 Nov 2021 08:49:20 GMT” |
| QUERY\_STREAM | 是对于 URL 中的 query（ query 即 URL 中?后面的 key1=valve1&key2=valve2 字符串）进行编码后的结果。以 key 按照字典顺序（ ASCII 码由小到大）排序，并使用 & 符号连接起来，生成相应的query\_string。 |

## 接口签名计算代码示例

|  |
| --- |
| import javafx.util.Pair; import lombok.extern.slf4j.Slf4j;  import javax.crypto.Mac; import javax.crypto.spec.SecretKeySpec; import javax.xml.bind.DatatypeConverter; import java.io.UnsupportedEncodingException; import java.net.URI; import java.net.URL; import java.net.URLEncoder; import java.text.DateFormat; import java.text.SimpleDateFormat; import java.util.\*; import java.util.stream.Collectors;  @Slf4j public class HmacAuthUtil {   /\*\*  \* 构造请求 header  \* @param urlStr 请求url，全路径格式，比如：[https://ibcdsg.zj.gov.cn:8443/restapi/prod/IC33000020220228000002/sso/servlet/simpleauth](https://bcdsg.zj.gov.cn:8443/restapi/prod/IC33000020220228000002/sso/servlet/simpleauth)  \* @param requestMethod 请求方法,大写格式，如：GET, POST  \* @param accessKey 应用的 AK  \* @param secretKey 应用的 SK  \* @return  \*/  public static Map<String, String> generateHeader(String urlStr,String requestMethod, String accessKey, String secretKey) {  log.info("params,urlStr={},requestMethod={},accessKey={},secretKey={}",urlStr,requestMethod,accessKey,secretKey);  Map<String, String> header = new HashMap<>();  try {  DateFormat dateFormat = new SimpleDateFormat("EEE, dd MMM yyyy HH:mm:ss z", Locale.US);  dateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));  String date = dateFormat.format(new Date());  URL url = new URL(urlStr);  URI uri = new URI(url.getProtocol(), url.getHost(), url.getPath(), url.getQuery(), null);   String canonicalQueryString = getCanonicalQueryString(uri.getQuery());   String message = requestMethod.toUpperCase() + "\n" + uri.getPath() + "\n" + canonicalQueryString + "\n" + accessKey + "\n" + date + "\n";   Mac hasher = Mac.getInstance("HmacSHA256");  hasher.init(new SecretKeySpec(secretKey.getBytes(), "HmacSHA256"));   byte[] hash = hasher.doFinal(message.getBytes());   // to lowercase hexits  DatatypeConverter.printHexBinary(hash);   // to base64  String sign = DatatypeConverter.printBase64Binary(hash);  header.put("X-BG-HMAC-SIGNATURE", sign);  header.put("X-BG-HMAC-ALGORITHM", "hmac-sha256");  header.put("X-BG-HMAC-ACCESS-KEY", accessKey);  header.put("X-BG-DATE-TIME", date);  } catch (Exception e) {  log.error("generate error",e);  throw new RuntimeException("generate header error");  }  log.info("header info,{}",header);  return header;  }   private static String getCanonicalQueryString(String query) {  if (query == null || query.trim().length() == 0) {  return "";  }  List<Pair<String, String>> queryParamList = new ArrayList<>();  String[] params = query.split("&");  for (String param : params) {  int eqIndex = param.indexOf("=");  String key = param.substring(0, eqIndex);  String value = param.substring(eqIndex+1);  Pair<String, String> pair = new Pair<String, String>(key,value);  queryParamList.add(pair);  }   List<Pair<String, String>> sortedParamList = queryParamList.stream().sorted(Comparator.comparing(param -> param.getKey() + "=" + Optional.ofNullable(param.getValue()).orElse(""))).collect(Collectors.toList());  List<Pair<String, String>> encodeParamList = new ArrayList<>();  sortedParamList.stream().forEach(param -> {  try {  String key = URLEncoder.encode(param.getKey(), "utf-8");  String value = URLEncoder.encode(Optional.ofNullable(param.getValue()).orElse(""), "utf-8")  .replaceAll("\\%2B","%20")  .replaceAll("\\+","%20")  .replaceAll("\\%21","!")  .replaceAll("\\%27","'")  .replaceAll("\\%28","(")  .replaceAll("\\%29",")")  .replaceAll("\\%7E","~")  .replaceAll("\\%25","%")  ;  encodeParamList.add(new Pair<>(key, value));  } catch (UnsupportedEncodingException e) {  throw new RuntimeException("encoding error");  }  });  StringBuilder queryParamString = new StringBuilder(64);  for (Pair<String, String> encodeParam : encodeParamList) {  queryParamString.append(encodeParam.getKey()).append("=").append(Optional.ofNullable(encodeParam.getValue()).orElse(""));  queryParamString.append("&");  }   return queryParamString.substring(0, queryParamString.length() - 1);  } } |

# 接口说明

## 基于单点登录票据换取请求token

### 请求地址

[/uc/sso/access\_token](https://bcdsg.zj.gov.cn:8443/restapi/prod/IC33000020220329000007/uc/sso/access_token" \t "_blank)

### 入参

|  |  |  |
| --- | --- | --- |
| **参数** | **类型** | **描述** |
| ticketId | String | 单点登录票据 |
| appId | String | AppId |

### 出参

|  |  |  |
| --- | --- | --- |
| **参数** | **类型** | **描述** |
| errorCode | String | 错误码 |
| errorMsg | String | 错误信息 |
| success | Boolean | 请求是否成功 |
| data | Object | 响应体 |
| |- accessToken | String | 获取用户信息token |

### 错误码

|  |  |
| --- | --- |
| **错误码** | **描述** |
| C-USER-SSO-TICKET-INVALID | ticket非法 |

## 基于token获取用户信息

### 请求地址

[/uc/sso/getUserInfo](https://bcdsg.zj.gov.cn:8443/restapi/prod/IC33000020220329000008/uc/sso/getUserInfo" \t "_blank)

### 入参

|  |  |  |
| --- | --- | --- |
| **参数** | **类型** | **描述** |
| token | String | 获取用户信息token |

### 出参

|  |  |  |
| --- | --- | --- |
| **参数** | **类型** | **描述** |
| success | Boolean | 请求是否成功 |
| errorCode | String | 错误码 |
| errorMsg | String | 错误信息 |
| data | Object | 响应体 |
| |- userType | String | 用户类型，PERSON  个人/LEGAL\_PERSON  法人 |
| |- personInfo | Object | 个人用户信息，当前登陆自然人的信息 |
| |-- userId | String | 主键 |
| |-- userName | String | 个人姓名 |
| |-- idType | String | ID\_CARD:身份证,PASSPORT:护照,OFFICER\_CARD:军官证,MAINLAND\_TRAVEL\_PERMIT\_FOR\_HONGKONG\_AND\_MACAO\_RESIDENTS:港澳居民来往内地通行证,MAINLAND\_TRAVEL\_PERMIT\_FOR\_TAIWAN\_RESIDENTS:台湾居民来往大陆通行证,FOREIGN\_PERMANENT\_RESIDENT\_ID\_CARD:外国人永久居留身份证,FOREIGN\_PASSPORT:外籍人士护照,DIPLOMACY\_PASSPORT:外交护照,OFFICIAL\_PASSPORT:公务护照,SOLDIER\_CARD:士兵证,OFFICER\_RETIRE\_CARD:军官离退休证,GANG\_AO\_TAI\_RESIDENCE\_CART:港澳台居民居住证,GANG\_AO\_ID\_CART:港澳居民身份证,UNIFIED\_SOCIAL\_ID:统一社会信用代码,OTHER:其他 |
| |-- outerIdType | String | 外部证件类型 |
| |-- idNo | String | 证件编号 |
| |-- attnUserType | String | 法人经办人时用户类型，评级 |
| |-- phone | String | 手机号 |
| |-- email | String | 邮箱 |
| |-- nation | String | 民族 |
| |-- gender | String | 性别 |
| |-- birthday | String | 生日 |
| |-- certKey | String | 身份散列值 |
| |-- attributes | Object | 额外属性 |
| |- legalPersonInfo | Object | 法人用户信息，比如公司相关的信息 |
| |-- name | String | 法人名称 |
| |-- unifiedSocialId | String | 社会统一信用代码 |
| |-- orgType | String | 法人类型 |
| |-- attnName | String | 经办人姓名 |
| |-- attnPhone | String | 经办人手机号 |
| |-- attnIdType | String | 经办人证件类型 |
| |-- attnIdNo | String | 经办人证件号码 |
| |-- attnUserType | String | 经办人用户等级 |
| |-- principal | String | 法人代表人姓名 |
| |-- gender | Integer | 法人代表人性别 |
| |-- nation | Integer | 法人代表人民族 |
| |-- idType | Integer | 法人代表人证件类型 |
| |-- outerIdType | String | 法人代表人外部证件类型 |
| |-- idNo | String | 法人代表人证件号码 |
| |-- principalUserId | String | 法人代表唯一键 |
| |-- corpId | String | 法人唯一键 |
| |-- attributes | Object | 额外属性 |
| |- organizationInfoList | Array | 所属组织信息 |
| |-- orgId | String | 组织主键 |
| |-- oid | String | Alias for orgId |
| |-- parentId | String | 父组织主键 |
| |-- pid | String | Alias for parentId |
| |-- name | String | 组织机构简称 |
| |--fullName | String | 组织机构全称 |
| |--devCoding | String | 组织后缀 |
| |--leafFlag | Boolean | 是否叶子标志 |
| |--orderBy | Integer | 排序号，从小到大 |
|  |  |  |

### 错误码

|  |  |
| --- | --- |
| **错误码** | **描述** |
| C-USER-SSO-TOKEN-INVALID | token非法 |
| C-USER-SSO-USER-EMPTY | 用户信息为空 |