**Uri:** http://ip:port/yktpre/services/activity/list

**请求格式：**application/json

**请求方式**：post

**参数：**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| begindate | String | 是 | 起始日期 |
| enddate | String | 是 | 结束日期 |
| timestamp | String | 是 | 时间戳，格式为yyyyMMddHHmmss，例如：20080125202330 |
| sign | String | 是 | API输入参数签名结果  加密data原始串：  begindate+enddate+timestamp  加密方法：RSA签名后再base64编码  rsa\_private\_key：  \=\= |

**例子:**

http://222.197.164.93/yktpre/services/activity/list

{

“begindate”:”20170620”,

“enddate”: "20170621",

“timestamp”:” 20170620120305”,

“sign”:”ONocFZCqy1QXs8cFPm2Togw3nsvoNi15yEKQlm%2Fvk5qv8vOI4nf8sIk72xJ49CQt0qg6gMSxZnoTOiKO3fUoQJprWxE34SKEe8fBFDbZP%2FIl2rwr6I%2BAP01d7mbaEBFPvaDj62RzP4LwC1ZsBYJ4vIymvG6bAiQiVTOu%2B1xiE2M%3D”

}

**返回：**

返回格式：application/json

{"retcode":0,

"retmsg":"查询成功",

"retdata":[{"act\_name":"成电讲坛01","act\_time":"20170217","act\_place":"一卡通会议室 ","sign\_time":"1330","stu\_id":"2140","stu\_name":"杨培洪","dept\_name":"党工 团","stu\_type":"教工","isvalid":1,"invalid\_reason":""},{"act\_name":"测 -01","act\_time":"20170314","act\_place":"综合楼8楼会议室 ","sign\_time":"1101","stu\_id":"164911324","stu\_name":"浦奕","dept\_name":"制造工程系 ","stu\_type":"高职","isvalid":0,"invalid\_reason":"提前离场"}]

}

附RSA工具类 java 版：

**import** java.security.KeyFactory;

**import** java.security.PrivateKey;

**import** java.security.Signature;

**import** java.security.spec.PKCS8EncodedKeySpec;

**import** com.sun.jersey.core.util.Base64;

/\*\*

\* RSA工具类

\*/

**public** **class** RSAUtil {

**private** **static** **final** String *ALGORITHM* = "RSA";

**private** **static** **final** String *SIGN\_ALGORITHM* = "SHA1withRSA";

/\*\*

\* 签名

\*

\* **@param** text

\* **@param** privateKeys

\* **@param** algorithm

\* **@return**

\* **@throws** Exception

\*/

**public** **static** **byte**[] sign(**byte**[] text, String privateKey) **throws** Exception {

PrivateKey priKey = *getPrivateKey*(privateKey);

Signature signatureChecker = Signature.*getInstance*(*SIGN\_ALGORITHM*);

signatureChecker.initSign(priKey);

signatureChecker.update(text);

**return** signatureChecker.sign();

}

/\*\*

\* 得到私钥

\*

\* **@param** key

\* 密钥字符串（经过base64编码）

\* **@return** 私钥对象

\* **@throws** Exception

\*/

**public** **static** PrivateKey getPrivateKey(**final** String key) **throws** Exception {

**byte**[] keyBytes = **null**;

keyBytes = Base64.*decode*(key);

PKCS8EncodedKeySpec keySpec = **new** PKCS8EncodedKeySpec(keyBytes);

KeyFactory keyFactory = KeyFactory.*getInstance*(*ALGORITHM*);

PrivateKey privateKey = keyFactory.generatePrivate(keySpec);

**return** privateKey;

}

}