充电 API

樊启刚 制作

2024-02-28

日期	版本	说明	作者
2024/2/28	V0.0.1	新增文档	樊启刚

目录

1 文档简介

1.1 特别声明

未得到本公司的书面许可，不得为任何目的、以任何形式或手段（包括但不限于机械的或电子的）复制或传播本文档的任何部分。对于本文档涉及的技术和产品，本公司拥有其专利（或正在申请专利）、商标、版权或其它知识产权。除非得到本公司的书面许可协议，本文档不授予这些专利、商标、版权或其它知识产权的许可。

本文档因产品功能示例和描述的需要，所使用的任何人名、企业名和数据都是虚构的，并仅限于本公司内部测试使用，不等于本公司有对任何第三方的承诺和宣传。

1.2 阅读对象

贵公司的技术部门的开发、维护及管理人员，应具备以下基本知识：

1. 了解 HTTPS/HTTP 协议等内容。
2. 了解信息安全的基本概念。
3. 了解计算机至少一种编程语言。

1.3 产品说明

本开发手册对该系统功能接口进行详细的描述，通过该指南可以对本系统有全面的了解，使技术人员尽快掌握本系统的接口，并能够在本系统上进行开发。

1.4 名词解释

名词缩写	名词定义

2 使用流程

1. 准备阶段：
   A. 申请测试号等信息；
   B. 取得开发手册（本文档）等资料；
2. 开发阶段：
   A. 根据提供的 DEMO 结合开发文档快速熟悉对接接口；
   B. 根据本系统提供的接口，在自己的系统上进行开发，实现所需要的业务功能；
   C. 对自己系统的业务功能进行全面测试；
   D. 与测试环境进行联调。
3. 生产使用：
   A. 使用系统提供的正式资料。

3 公共字段

描述：请求报文的统一格式

参数说明：

参数名	类型	参数描述
SOURCE	String	渠道编号
ITS	String	时间戳
IFSN	String	发起方流水号
VN	String	版本号
BD	String	业务加密数据
SIGN	String	签名

4 接口列表

4.1 信息推送模块

4.1.1 账单

URL	停车平台开发方自己确定
Content-Type	application/json
请求方式	post
接口备注	由充电平台将订单信息推送到停车平台
调用方	充电平台

请求参数说明:

参数名	示例值	参数类型	参数描述
code	0	Integer	见响应码字典
msg		String	描述

请求示例:

```
{
        "code": 0,
        "msg": ""
}
```

:

返回体参数说明:

参数名	类型	必填	备注
license_plate_number	String	是	车牌号
phone	String	是	手机号
number	String	是	订单号
charging_site_number	String	是	站点编号
charging_site_name	String	是	站点名称

charging_pile_number	String	是	充电桩编号
charging_spear_number	String	是	充电枪编号
electricity_quantity	String	是	充电电量(kw.h)
charging_amount	String	是	电费(元)
actual _amount	String	是	实付金额（元）
service_charge_amount	String	是	服务费（元）
order_amount	String	是	订单金额（元）
order_start_time	String	是	订单开始时间(yyyy-MM-dd HH:mm:ss)
order_end_time	String	是	订单结束时间(yyyy-MM-dd HH:mm:ss)
discount_type	Integer	是	优惠类型 1:快充，给予优惠券;2：慢充，不给予优惠

返回示例:

```
{
    "license_plate_number":"",
    "phone":"",
    "number":"",
    "charging_site_id":"",
```

```
    "charging_site_name":"",
    "charging_pile_number":"",
    "charging_spear_number":"",
    "electricity_quantity":"",
    "charging_amount":"",
    "actual _amount":"",
   "service_charge_amount":"",
   "order_amount":"",
    "order_start_time":"",
    "order_end_time":"",
    "discount_type":""

}
```


5 生产环境资料

5.1 如何获取服务器地址

联系充电平台运营人员交换双方服务器地址、公钥、私钥

5.2 响应返回异常

对比实际发出的请求报文（curl、参数、body、header）是否完全吻合。

（关于密码的保管：贵公司一定要保证密码仅能被少数可靠的授权人知晓，严防密码被不可信的人获取，如密码泄露需立即进行修改同时替换程序中的密码。）

6 加密

6.1 密钥生成

开始调用接口前需要停车平台和充电平台各自调用开发的密钥生成器(RsaUtil.java)生成公钥和私钥，使用 RSA 算法进行加解密。

充电平台生成公钥和私钥后，充电平台将私钥自己留存，用于将来发送请求时生成签名信息；充电平台公钥则提供给停车平台，停车平台通过该公钥对客户上送的签名信息验签。同时，停车平台也会生成一对公私钥，在返回请求数据时会用私钥来签名，公钥提供给业务平台用以验签。

采用 RSA 算法就可以在 Java 代码中使用工具类 RsaUtil.java,RSA 算法生成公钥和私钥可以使用 RsaUtil 的 generateKey()方法。

```java
public class Test{
    public static void main(String[] args) throws Exception {
        Map<String, String> stringStringMap = RsaUtil.generateKey();
        //渠道方公钥
        String sourcepublic = stringStringMap.get("publicKeyStr");
        //渠道方私钥
        String sourceprivte = stringStringMap.get("privateKeyStr");
    }

}
```


6.2 业务数据签名方法

（1） 签名信息的生成

在调用生成支付订单接口之前，首先要根据请求参数和客户之前生成的私钥生成签名信息，用于平台进行客户身份的验证，业务数据转为 json 字符串（采用 fastjson，如果使用其他 json 类库，需要注意的是 key 需要 Ascii 排序）

字典顺序说明如下：

字典顺序是按照字母顺序，或者数字小大顺序，由小到大的形成序列。先按照第一个字母，以 a、b、c……z 的顺序排列；如果第一个字母一样，那么比较第二个、第三个乃至后面的字母。

签名和加密示例：

```java
public static void main(String[] args) throws Exception {
    Map<String, String> stringStringMap = RsaUtil.generateKey();
    //充电平台公钥
    String sourcepublic = stringStringMap.get("publicKeyStr");
    //充电平台私钥
    String sourceprivte = stringStringMap.get("privateKeyStr");
    stringStringMap = RsaUtil.generateKey();
    //接收方公钥
    String splitpublic = stringStringMap.get("publicKeyStr");
    //接收方私钥
    String splitprivte = stringStringMap.get("privateKeyStr");

    JSONObject data = new JSONObject();
    //充电平台编号
    data.put("SOURCE", "xxxx");
    //发起方时间戳
    data.put("ITS", DateUtil.format(LocalDateTime.now(), "yyyyMMddHHmmssSSS"));
    //发起方流水号
    data.put("IFSN", "1212121");
    //版本号
    data.put("VN", "1");
    JSONObject bill = new JSONObject();
    bill.put("number", "1233445");


    //业务数据
    data.put("BD", encryptBD(bill, splitpublic));
    //签名
    data.put("SIGN", SplitUtil.sign(data, sourceprivte));


    System.out.println(data.toJSONString());
    //解密业务数据
    System.out.println(decryptBD(data.getString("BD"), splitprivte));
    //验签
    System.out.println(SplitUtil.verify(data, sourcepublic));
}
```

（2）返回 result 的验证和解析

客户端收到平台返回的 json 格式结果信息后，解析 json 字符串，获得返回的各个字段，并将返回信息中除签名信息外的其他字段信息按照与**（1）签名信息的生成**小节中相同。根据之前交换得到的平台公钥和发送来的签名信息 signInfo，进行平台身份校验。校验采用 SplitUtil.java 的 verify()方法。如果校验结果为 true 则说明身份验证通过，可以进行下一步操作，否则身份验证失败

(3) 加密和签名需要用到的工具文件：


RsaUtil.java


SplitUtil.java

7 附录

7.1 响应码字典

状态码	描述
1000	数据处理成功
1001	请求超时,时间戳校验不通过
1002	流水号只能使用一次
1003	签名校验出错
1004	签名校验失败
1005	数据解密出错
1008	已有同一流水号账单正在处理
1009	订单已被处理
1014	业务处理异常