充电卷相关接口说明文档

1、新增充电卷接口：

**简要描述：**

* 充电桩项目提供的新增充电卷接口，单个新增。

**请求URL：**

* 正式环境：http://kyunai.com/chargingMerchant/ volume/addVolume
* 测试环境（请用测试环境测试）：http://kyunai.com/chargingMerchantTest/volume/addVolume

**请求方式：**

* POST

**参数：**

| **参数名** | **必选** | **类型** | **说明** |
| --- | --- | --- | --- |
| volumeInfo | 是 | string | 充电卷信息(组装json格式, 通过固定方式加密) |

**json示例**

{"volumeNo":"cdz1000001","openid":"657ujngfjer65gy457y647","money":1,"isUse":0,"createTime":1570348274000,"validTime":1606809079000,"remark":"充电卷"}

**返回参数说明**

| **参数名** | **类型** | **说明** |
| --- | --- | --- |
| Msg | String | 处理结果，“fail”：新增失败；“ok”：新增成功。 |

**加密方式：**

3DES加密

**加密方法代码：**

import lombok.extern.slf4j.Slf4j;

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

import javax.crypto.Cipher;

import javax.crypto.SecretKeyFactory;

import javax.crypto.spec.DESedeKeySpec;

import javax.crypto.spec.IvParameterSpec;

import java.security.Key;

/\*\*

\* @author: cryann

\* @description:

\* @date: 2019/6/11 9:53

\*/

@Slf4j

public class DesCbcUtil {

// 加解密统一使用的编码方式

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

/\*\*

\* 3DES加密

\*

\* @param plainText 普通文本

\* @return 加密后的文本，失败返回null

\*/

public static String encode(String plainText, String secretKey, String iv) {

String result = null;

try {

DESedeKeySpec deSedeKeySpec = new DESedeKeySpec(secretKey.getBytes());

SecretKeyFactory secretKeyFactory = SecretKeyFactory.getInstance("desede");

Key desKey = secretKeyFactory.generateSecret(deSedeKeySpec);

Cipher cipher = Cipher.getInstance("desede/CBC/PKCS5Padding");

IvParameterSpec ips = new IvParameterSpec(iv.getBytes());

cipher.init(Cipher.ENCRYPT\_MODE, desKey, ips);

byte[] encryptData = cipher.doFinal(plainText.getBytes(encoding));

// result = Base64Utils.encodeToString(encryptData);

result = Base64.encodeBase64URLSafeString(encryptData);

} catch (Exception e) {

log.error("DesCbcUtil encode error : {}", e);

}

return result;

}

/\*\*

\* 3DES解密

\*

\* @param encryptText 加密文本

\* @return 解密后明文，失败返回null

\*/

public static String decode(String encryptText, String secretKey, String iv) {

String result = null;

try {

DESedeKeySpec spec = new DESedeKeySpec(secretKey.getBytes());

SecretKeyFactory secretKeyFactory = SecretKeyFactory.getInstance("desede");

Key desKey = secretKeyFactory.generateSecret(spec);

Cipher cipher = Cipher.getInstance("desede/CBC/PKCS5Padding");

IvParameterSpec ips = new IvParameterSpec(iv.getBytes());

cipher.init(Cipher.DECRYPT\_MODE, desKey, ips);

byte[] deBase64 = Base64.decodeBase64(encryptText);

byte[] decryptData = cipher.doFinal(deBase64);

// byte[] decryptData = cipher.doFinal(Base64Utils.decodeFromString(encryptText));

result = new String(decryptData, encoding);

} catch (Exception e) {

log.error("DesCbcUtil decode error : {}", e.getMessage());

}

return result;

}

}

**生成加密后的字符串(第二、第三个参数固定)：**

String param = DesCbcUtil.*encode*(paramJson,"439a362ffbbf460a60dc26543549a8ec","20191006");

测试包：

volumeInfo=9lw3iWsvk6RHHkXiOw\_vkHtL7f-oiLqTAeF3-qxytug\_Ut8huWPTX52IQ-9Xt9J6w-iUNmh5xAv5WR0T5KO68T4FSgVHedBLSUztwq8ax8dwbuXlfVfBEydIMKtnKrzEO\_JmRUf9DLH8C16PhP\_s8FjKLsF-AzpmwY6\_YUDRlGDLhzRtS\_dHxUHO9tOKgwU7lPmbRKv3q9CDj1NVffuD3A