

# WebApp接入说明文档

- 一、引导
- 二、加密算法
  - 说明
  - php版DES加密 / 解密类
  - java版DES加密 / 解密类
- 三、接入模块
  - 用车模块流程
  - 获取滴滴服务城市列表
  - 一次性发单Ticket申请接口
  - 非发单Ticket申请接口
  - WebApp主页面调起
  - WebApp常用地址设置页面单独调起
  - WebApp历史行程列表页面单独调起
- 四、支付分离
  - 支付分离方案流程
  - 客户提供支付跳转页面的约定说明
  - 客户服务端调用滴滴确认支付API说明

## 一、引导

滴滴WebApp为接入方提供如：获取城市列表、一次性发单/非发单ticket、主页面/常用地址设置/历史行程列表调起等服务，确保了用车流程的完整性，具有开发维护成本低、对接周期短、员工用车体验好、功能迭代效率高等优点。

更新信息：

日期	说明	新增字段
2019. 01. 18	一次性发单ticket支持拼车用车能力，详情参考一次性发单ticket章节	carpool

## 二、加密算法

### 说明

采用DES参数加密算法，保证数据传输安全，防止篡改，以sign\_key的前8个字符作为加密 / 解密的私钥key，提供了java版和php版算法，其他开发语言可参考思路自行实现（DES加密 / 解密类仅供参考思路，需要接入方具体实现细节并调通）。

### php版DES加密 / 解密类

```
<?php
/**
 * Created by PhpStorm.
 * User: didi
 * Date: 2017/9/27
 * Time: 16:11
 */
namespace Library;

class DESHelper
{
    //私钥，固定字节长度为8位
    private $key;
    //偏移量，固定字节长度为8位
    private $iv;
    private $cipher;
```

```

public function __construct($key)
{
    if(strlen($key) > 8)
    {
        $this->key = substr($key, 0, 8);
    }
    else
    {
        $this->key = $key;
    }

    $this->iv = $this->key;

    $this->cipher = mcrypt_module_open(MCRYPT_DES, '', MCRYPT_MODE_CBC, '');

//var_dump(mcrypt_enc_get_iv_size($this->cipher), mcrypt_enc_get_key_size($this->cipher), mcrypt_enc_get_block_size($this->cipher));exit;
}
//加密
public function encrypt($str)
{
    $size = mcrypt_enc_get_block_size($this->cipher);
    $str = $this->pkcs5Pad($str, $size);
    mcrypt_generic_init($this->cipher, $this->key, $this->iv);
    $data = mcrypt_generic($this->cipher, $str);
    mcrypt_generic_deinit($this->cipher);
    return base64_encode($data);
}
//解密
public function decrypt($str)
{
    $str = base64_decode($str);
    mcrypt_generic_init($this->cipher, $this->key, $this->iv);
    $data = mdecrypt_generic($this->cipher, $str);
    $data = rtrim($data, "\0");
    mcrypt_generic_deinit($this->cipher);
    $data = $this->pkcs5Unpad($data);
    return $data;
}
private function pkcs5Pad($text, $blocksize)
{
    $pad = $blocksize - (strlen($text) % $blocksize);
    return $text . str_repeat(chr($pad), $pad);
}
private function pkcs5Unpad($text)
{
    $pad = ord($text[strlen($text) - 1]);
    if($pad > strlen($text))
        return false;
    if(strspn($text, chr($pad), strlen($text) - $pad) != $pad)
        return false;
    return substr($text, 0, -1 * $pad);
}
public function __destruct()
{
}

```

```

        mcrypt_module_close($this->cipher);
    }
}

```

## java版DES加密 / 解密类

```

package amigo.endencrypt;
import java.security.InvalidKeyException;
import java.security.Key;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.spec.InvalidKeySpecException;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESKeySpec;
import javax.crypto.spec.IvParameterSpec;
import org.apache.commons.codec.binary.Base64;
public class DESUtil {
    //算法名称
    public static final String KEY_ALGORITHM = "DES";
    //算法名称/加密模式/填充方式

//DES共有四种工作模式-->>ECB: 电子密码本模式、CBC: 加密分组链接模式、CFB: 加密反馈模式、OFB: 输出反馈模
式
    public static final String CIPHER_ALGORITHM = "DES/CBC/PKCS5Padding";
    /**
     *
     * 生成密钥key对象
     * @param KeyStr 密钥字符串
     * @return 密钥对象
     * @throws InvalidKeyException
     * @throws NoSuchAlgorithmException
     * @throws InvalidKeySpecException
     * @throws Exception
     */
    private static SecretKey keyGenerator(String keyStr) throws Exception {
        DESKeySpec desKey = new DESKeySpec(keyStr.getBytes("UTF-8"));
        //创建一个密钥工厂，然后用它把DESKeySpec转换成
        SecretKeyFactory keyFactory = SecretKeyFactory.getInstance("DES");
        SecretKey securekey = keyFactory.generateSecret(desKey);
        return securekey;
    }
    /**
     * 加密数据
     * @param data 待加密数据
     * @param key 密钥
     * @return 加密后的数据
     */
    public static String encrypt(String data, String key) throws Exception {
        Key deskey = keyGenerator(key);
        IvParameterSpec iv = new IvParameterSpec(key.getBytes("UTF-8"));
        // 实例化Cipher对象，它用于完成实际的加密操作
        Cipher cipher = Cipher.getInstance(CIPHER_ALGORITHM);
        // 初始化Cipher对象，设置为加密模式
        cipher.init(Cipher.ENCRYPT_MODE, deskey, iv);
    }
}

```

```
byte[] results = cipher.doFinal(data.getBytes("UTF-8"));
// 执行加密操作。加密后的结果通常都会用Base64编码进行传输
return Base64.encodeBase64String(results);
}
/**
 * 解密数据
 * @param data 待解密数据
 * @param key 密钥
 * @return 解密后的数据
 */
public static String decrypt(String data, String key) throws Exception {
    Key deskey = keyGenerator(key);
    IvParameterSpec iv = new IvParameterSpec(key.getBytes("UTF-8"));
    Cipher cipher = Cipher.getInstance(CIPHER_ALGORITHM);
    //初始化Cipher对象，设置为解密模式
    cipher.init(Cipher.DECRYPT_MODE, deskey, iv);
    // 执行解密操作
    return new String(cipher.doFinal(Base64.decodeBase64(data)));
}
public static void main(String[] args) throws Exception {
    String source = "amigoxie";
    System.out.println("原文: " + source);
    String key = "A1B2C3D4";
    String encryptData = encrypt(source, key);
    System.out.println("加密后: " + encryptData);
    String decryptData = decrypt(encryptData, key);
```

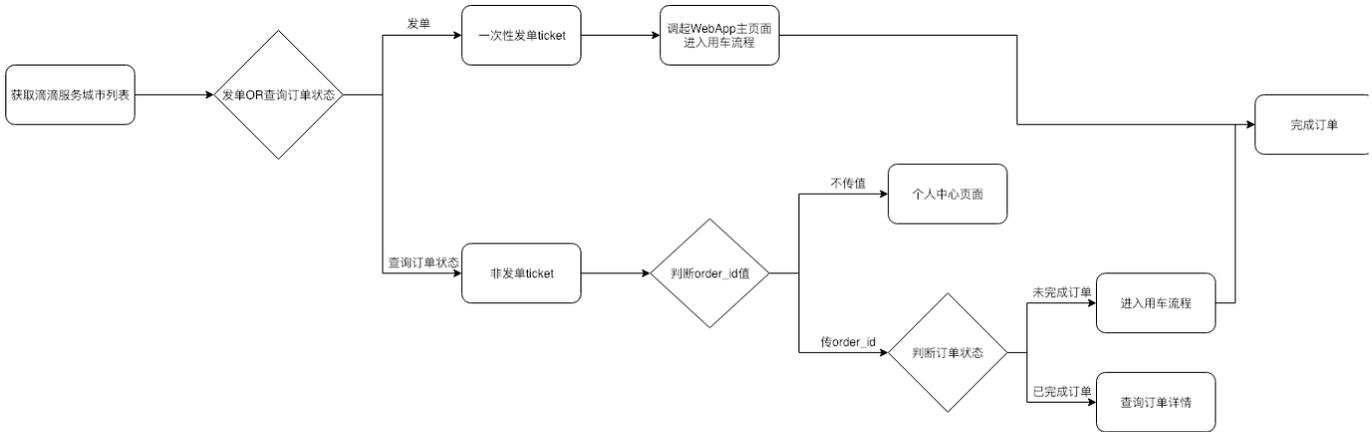
```

        System.out.println("解密后: " + decryptData);
    }
}

```

### 三、接入模块

#### 用车模块流程



#### 获取滴滴服务城市列表

说明：获取城市列表（包含城市名称、城市ID、是否开通快车、是否开通专车），如需限制发单城市，申请一次性发单ticket时传对应城市ID。

请求地址：<https://api.es.xiaojukeji.com/webapp/cities/getAll>

请求类型：GET

请求参数：

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
data_encode	string	yes	data数组json序列化后，进行DES对称加密得到的加密串（密钥key为申请应用时分配的sign_key的前8个字节；加密模式为CBC模式）

data数组的元素列表：

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
client_secret	string	yes	申请应用时分配的client_secret
order_id	bigint	yes	待支付的滴滴订单ID

data是K-V的map，示例：

```

{"client_id":"xxx","client_secret":"xxx","order_id":123}

```

请求示例:

```
curl "https://api.es.xiaojukeji.com/webapp/cities/getAll?client_id=testtest&data_encode=xxxxxxx"
```

注: 因该接口为GET方式传参, 故调用该接口时, 需要对data\_encode的参数值做urlencode编码操作, 主要是为了避免参数值中包含特殊字符, 影响数据解码。

返回结果:

参数名称	数据类型	说明
errno	int	错误码
errmsg	string	错误信息
data	object	数据结果

data元素列表:

参数名称	数据类型	说明
name	string	城市首字母
cities	object	各城市开通信息集合

cities元素列表:

参数名称	数据类型	说明
name	string	城市名称
cityid	int	城市ID
open_zhuanche	int	是否开通专车 1-开通 0-未开通
open_kuaiche	int	是否开通快车 1-开通 0-未开通

返回结果示例:

```
{
  "errno": 0,
  "errmsg": "SUCCESS",
  "data": [
    {
      "name": "A",
      "cities": [
        {
          "name": "鞍山市",
          "cityid": 64,
          "open_zhuanche": 0,
          "open_kuaiche": 1
        },
        {
          "name": "安庆市",
          "cityid": 149,
          "open_zhuanche": 0,
          "open_kuaiche": 1
        },
        ...
      ]
    },
    {
      "name": "B",
      "cities": [
        {
          "name": "北京市",
          "cityid": 1,
          "open_zhuanche": 1,
          "open_kuaiche": 1
        },
        {
          "name": "保定市",
          "cityid": 62,
          "open_zhuanche": 0,
          "open_kuaiche": 1
        },
        ...
      ]
    },
    ...
  ]
}
```

## 一次性发单Ticket申请接口

说明：一次性发单ticket限制只能调起一次webapp（每次接入方调起webapp时，都需要重新申请ticket），且发单ticket只能发布一个订单。

请求地址：<https://api.es.xiaojukeji.com/webapp/ticket/fetch>

请求类型：POST

请求参数：

参数名	类型	必选	描述
-----	----	----	----

client_id	string	yes	申请应用时分配的client_id
data_encode	string	yes	data数组json序列化后，进行DES对称加密得到的加密串（密钥key为申请应用时分配的sign_key的前8个字节；加密模式为CBC模式

data数组的元素列表：

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
client_secret	string	yes	申请应用时分配的client_secret
master_phone	string	yes	管理员手机号（作为统一叫车人）
passenger_phone	string	yes	乘客手机号(当前用户)
auth_type	int	yes	授权类型：0-非发单（无发单权限）；1-发单（有发单权限及其他所有权限）；此处传1；
city_id	int	no	用车城市(出发地城市ID)，如不传，则不限制城市；传了城市ID，限制本次叫车只允许在该城市发单；
to_city_id	int	no	目的地城市ID，如不传，则不限制目的地城市；传了城市ID，限制本次叫车目的地城市；
forbid_city_cross	int	no	是否禁止跨城（出发地和目的地不在同一城市）：0-否，1-是，默认0
rule_list	string	no	允许用车方式，如“301,201,203,204”；不传则默认为“201,301” 301-单次快车；201-单次专车；203-专车接机；204-专车送机 以下为用车方式对应能使用的车型： 301（单次快车）：600,900 201（单次专车）：100,400,200 203（专车接机）：100,400,200 204（专车送机）：100,400,200
require_level_list	string	no	允许可选车型，不传则不限制可选车型，如“600,900”； 600-普通快车；900-快车优享；100-专车舒适型；200-专车行政级；400-专车商务型；
carpool	int	no	是否展示拼车：0-不展示拼车 1-展示拼车 不传默认选择 1-展示拼车
clat	float	no	当前位置纬度（建议接入方从端获取当前位置，并转换为腾讯地图坐标），提升用户发单体验 已废弃，使用app自动定
clng	float	no	当前位置经度（建议接入方从端获取当前位置，并转换为腾讯地图坐标），提升用户发单体验 已废弃，使用app自动定
restrict_from_point	int	no	是否限制出发地（0-否，1-是；默认0）
from_name	string	no	出发地名称
flat	float	no	出发地纬度
flng	float	no	出发地经度
restrict_to_point	int	no	是否限制目的地（0-否，1-是；默认0）
to_name	string	no	目的地名称
tlat	float	no	目的地纬度
tlng	float	no	目的地经度
departure_time	datetime	no	出发时间，格式：2017-09-20 21:00:00，出发时间大于当前10分钟，发预约单；如果小于10分钟或不传该字段，发实时单。预约单则必须填写出发时
personal_center_hide	int	no	是否隐藏个人中心入口（0-不隐藏，1-隐藏；默认0）
callback_info	object	no	用车自定义信息，透传内容。在对账单中会展示该字段。长度限制100。
enable_limited_delay	int	no	是否限制用户延迟叫车，0为不限制，1为限制。默认为0。
limited_delay_time	int	no	可延迟时长，以分钟为单位，范围限制为10分钟-120分钟。若确认限制且时长参数为空则默认可延迟时长为30分钟。
per_order_fee	decimal	no	每单限额。传输0，不限；大于0，每单企业支付不超过此金额（支持2位小数）。默认为0。

callback\_info数组的元素列表：

employee_number	string	no	员工工号
-----------------	--------	----	------

client_order_id	string	no	接入方系统的订单ID
organization	string	no	用车人组织（所属部门或分公司等）
airport_id	int	no	机场ID
flight_number	string	no	航班号（rule_list包含203时，会将此航班号带入发单页）
flight_time	string	no	航班时间
...			扩展自定义字段

data是K-V的map，示例：

```

{"client_id":"xxx","client_secret":"xxx","master_phone":"13800000000"}

```

请求示例：

```
curl -d "client_id=testtset&data_encode=xxxxxxx" "https://api.es.xiaojukeji.com/webapp/ticket/fetch"
```

返回结果：

参数名称	数据类型	说明
errno	int	错误码
errmsg	string	错误信息
data	object	数据结果

data数据内容说明：

字段名	类型	是否必选	说明
ticket	string	yes	调起webapp的一次性ticket
order_id	int	yes	本次下单的滴滴订单ID（用于接入方订单与滴滴订单关联）

请求结果示例：

```

{
  "errno": 0,
  "errmsg": "SUCCESS",
  "data": {
    "ticket": "xxxxxxxxxxxxx",
    "order_id": "4880109188406595918"
  }
}

```

## 非发单Ticket申请接口

说明：非发单ticket在有效期内可重复调起webapp，但不能发布订单；

请求地址：<https://api.es.xiaojukeji.com/webapp/ticket/fetch>

请求类型：POST

请求参数：

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
data_encode	string	yes	data数组json序列化后, 进行DES对称加密得到的加密串(密钥key为申请应用时分配的sign_key的前8个字节; 加密模式为CBC模式)

data数组的元素列表:

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
client_secret	string	yes	申请应用时分配的client_secret
master_phone	string	yes	管理员手机号(作为统一叫车人)
passenger_phone	string	yes	乘客手机号(当前用户)
auth_type	int	yes	授权类型: 0-非发单(无发单权限); 1-发单(有发单权限及其他所有权限); 此处传0;
order_id	int	no	滴滴订单ID(传该值, 跳转到指定订单详情页面; 不传值, 则进入个人中心通用页面)
personal_center_hide	int	no	是否隐藏个人中心入口(0-不隐藏, 1-隐藏; 默认0)

data是K-V的map, 示例:

```
{ "client_id": "xxx", "client_secret": "xxx", "master_phone": "13800000000" }
```

请求示例:

```
curl -d "client_id=testtest&data_encode=xxxxxxx" "https://api.es.xiaojukeji.com/webapp/ticket/fetch"
```

返回结果:

参数名称	数据类型	说明
errno	int	错误码
errmsg	string	错误信息
data	object	数据结果

data数据内容说明:

字段名	数据类型	必选	说明
ticket	string	yes	调起webapp的非发单ticket

返回结果示例:

```
{
  "errno": 0,
  "errmsg": "SUCCESS",
  "data": {
    "ticket": "xxxxxxxxxxxxx"
  }
}
```

## WebApp主页面调起

请求地址: <https://open.es.xiaojukeji.com/webapp/entry>

请求类型: GET

请求参数:

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
ticket	string	yes	一次性ticket

调起URL示例:

```
https://open.es.xiaojukeji.com/webapp/entry?client_id=testtest&ticket=xxxxxxxxxxxx
```

## WebApp常用地址设置页面单独调起

请求地址: [https://open.es.xiaojukeji.com/webapp/home/index?ticket=xxxxxxx&client\\_id=testclient&jumpPage=commonAddr](https://open.es.xiaojukeji.com/webapp/home/index?ticket=xxxxxxx&client_id=testclient&jumpPage=commonAddr)

请求类型: GET

请求参数:

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
ticket	string	yes	一次性ticket
jumpPage	string	yes	需要跳转的模块

调起URL示例:

```
https://open.es.xiaojukeji.com/webapp/home/index?ticket=xxxxxxx&client_id=testclient&jumpPage=commonAddr
```

## WebApp历史行程列表页面单独调起

请求地址: [https://open.es.xiaojukeji.com/webapp/home/index?ticket=xxx&client\\_id=xxx&jumpPage=orderlist&errpage=0&order\\_status=2](https://open.es.xiaojukeji.com/webapp/home/index?ticket=xxx&client_id=xxx&jumpPage=orderlist&errpage=0&order_status=2)

请求类型: GET

请求参数:

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
ticket	string	yes	一次性ticket

调起URL示例:

```
https://open.es.xiaojukeji.com/webapp/home/index?ticket=xxx&client_id=xxx&jumpPage=orderlist&errpage=0&order_status=2
```

## 四、支付分离

### 支付分离方案流程

webapp满足了企业客户自有支付方式的需求，流程为：

- 1、滴滴webapp待支付页面：乘客打完车，进入webapp待支付页面，点击确认支付
- 2、企业客户自有支付流程：跳转到企业客户的支付页面（企业客户自有支付方式）进行费用支付，针对企业客户侧系统的订单支付成功
- 3、滴滴订单的支付：企业客户调用滴滴的确认支付API完成滴滴订单的费用支付
- 4、支付成功提示：提示乘客订单支付成功（两种后续交互方式：a、回跳到滴滴webapp的订单支付成功页；b、直接关闭当前页面）

以下针对第2、3两步进行详细说明。

### 客户提供支付跳转页面的约定说明

请求地址： 客户提供

请求类型： GET

请求参数:

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
data_encode	string	yes	data数组json序列化后，进行DES对称加密得到的加密串（密钥key为申请应用时分配的sign_key的前8个字节；加密模式为CBC模式）

data数组的元素列表:

参数名	类型	必选	描述
client_id	string	yes	申请应用时分配的client_id
order_id	bigint	yes	待支付的滴滴订单ID

跳转页示例:

```
https://url?client_id=xxxx&data_encode=xxxxxx
```

说明：

- 1、客户支付跳转页接收到参数后，需要通过滴滴提供DES对称加密算法，对data\_encode进行解密，得到待支付的订单ID。
- 2、客户通过订单ID从自己系统的订单数据表里把订单信息（通过滴滴回调+详情拉取方式存储到订单数据表里）拉取出来，并把账单展示给乘客。

## 客户服务端调用滴滴确认支付API说明

请详细参考用车API的费用支付确认接口说明 (<https://open.es.xiaojukeji.com/doc/openapi/order/feeConfirm.html>)