共享单车API接口技术文档

**文件编号： XXX-XXX-XXXX**

**产品名称： 共享单车云平台**

**项目名称：**

|  |  |  |  |
| --- | --- | --- | --- |
| 编 写 |  | 日 期 |  |
| 审 核 |  | 日 期 |  |
| 批 准 |  | 日 期 |  |

**（内部资料，注意保密）**

**XXXOOO有限公司**

修改记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本** | **日期** | **修改纪要** | 修改人 | 审核人 |
| V0.1 | 2018-11-16 | 创建初稿 | 李智勇 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 录

[共享单车API接口技术文档 1](#_Toc530407085)

[修改记录 2](#_Toc530407086)

[1. 文档介绍 4](#_Toc530407087)

[2. API接口协议定义 4](#_Toc530407088)

[2.1 协议头定义 5](#_Toc530407089)

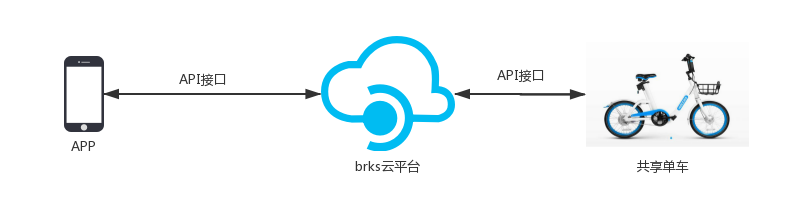
[2.2 协议的type定义 6](#_Toc530407090)

[2.3 消息ID的定义 6](#_Toc530407091)

[2.4 消息序列号 7](#_Toc530407092)

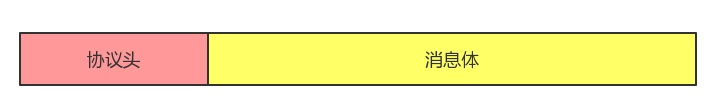
# 文档介绍

共享单车包括brks云平台、手机APP和共享自行车组成，该文档旨在说明手机APP/小程序和云平台的接口定义，共享自行车和云平台的接口定义。如下图：



# API接口协议定义

该接口为应用层协议，且采用TCP作为底层的传输协议，协议总共包含两部分，如下图所示：



### 2.1 协议头定义

下文中的u代表unsigned, u后面的数字代表是多少位，比如u16就表示unsigned short，u8代表unsigned char，u32代表unsigned int。

struct protocol\_head\_t

{

u16 tag\_; // 0xFBFC

u8 version\_; // protocol version, high 4 bit is master version,

// low 4 bit is sub version

u8 type\_; // content type，json协议格式

u16 len\_; // content length，消息的边界，TCP粘包

u16 msg\_id\_; // message id

u64 msg\_sn; // message serise number

u32 reserve\_; // reserve bytes

};

|  |  |  |
| --- | --- | --- |
| 字段 | 长度（byte） | 说明 |
| tag | 2 | 先导码，标志消息的起始和结束 |
| version | 1 | 协议版本号，高4位为主版本，低4位为子版本 |
| type | 1 | 消息体的类型，另述如下文 |
| len | 2 | 消息体的长度，不包含协议头 |
| msg\_id | 2 | 消息ID， GET HEAD POST |
| msg\_sn | 8 | 低14位表示用户手机号码的后4位数字，或者设备编号的后4位数字，再接着4位为编号服务器的机器ID，再接着6位为通信方向，剩余的40位表示消息序列号。 |
| reserve | 4 | 预留 |

### 2.2 协议的type定义

定义这个type是为了这个协议可以兼容不同的编码方式。

enum protocol\_type\_t

{

JSON\_PROTOCOL\_TYPE = 0, // json

PB\_PROTOCOL\_TYPE = 1, // Google Protocol Buffer

FB\_PROTOCOL\_TYPE = 2, // FlatBuffers

BINARY\_PROTOCOL\_TYPE = 3, // binary format

UNKNOW\_PROTOCOL\_TYPE = 0xFF

};

### 2.3 消息ID的定义

消息ID定义的是消息的类型，消息类型也有两大类，第一类是请求类型（以REQ结尾），另一类是回应类型（以RSP结尾），每请求必须要有回应，如果未收到回应，意为消息丢失或者接收方处理失败，此时需要按照以下规则处理：

规则一：发送方重新发送请求消息，重复的间隔单位时间是500毫秒，重复的系数是1,4,16,64，也就是说重复4次发送，第一次间隔是500毫秒，第二次是2秒，第三次是8秒，第四次是32秒。重发直到成功或者超过4次放弃发送消息。

规则二：如果消息是由云平台主动发送的，或者是云平台内部的，重发4次后依然未收到回复，则把该消息内容以告警（alarm）的形式发送给运营模块。

其类型定义有：

/\* 消息ID \*/

enum EventID

{

EEVENTID\_COMMON\_RSP = 0x00,

EEVENTID\_GET\_MOBILE\_CODE\_REQ = 0x01,

EEVENTID\_GET\_MOBILE\_CODE\_RSP = 0x02,

EEVENTID\_LOGIN\_REQ = 0x03,

EEVENTID\_LOGIN\_RSP = 0x04,

EEVENTID\_RECHARGE\_REQ = 0x05,

EEVENTID\_RECHARGE\_RSP = 0x06,

EEVENTID\_GET\_ACCOUNT\_BALANCE\_REQ = 0x07,

EEVENTID\_GET\_ACCOUNT\_BALANCE\_RSP = 0x08,

EEVENTID\_LIST\_ACCOUNT\_RECORDS\_REQ = 0x09,

EEVENTID\_LIST\_ACCOUNT\_RECORDS\_RSP = 0x10,

EEVENTID\_UNLOCK\_REQ = 0x11,

EEVENTID\_UNLOCK\_RSP = 0x12,

EEVENTID\_LIST\_TRAVELS\_REQ = 0x13,

EEVENTID\_LIST\_TRAVELS\_RSP = 0x14,

EEVENTID\_LOCK\_REQ = 0x15,

EEVENTID\_LOCK\_RSP = 0x16,

EEVENTID\_UNKOWN = 0xFFFF

};

### 2.4 消息序列号

消息序列号唯一标识一条消息，该消息序列号由4部分组成，如下所示：



账户或者设备编号，14 bits，为账户或者设备编号的后4位。

编号机器编号，4 bits，编号机器就是专门给整个系统编号服务的比如订单编号，消息序号编号等等，系统运行中，支持集群编号，因此需要规定编号机器的一个虚拟ID值，其ID值的范围为0~15，共支持16台编号机器的编号。在消息ID序列号中占据4位。

通信方向编号，6 bits，也即定义了所有的消息方向，其定义如下：

enum message\_arrow\_t

{

APP\_2\_CLOND\_ARROW = 0,

CLOND\_2\_APP\_ARROW = 1,

BIKE\_2\_CLOND\_ARROW = 2,

CLOND\_2\_BIKE\_ARROW = 3,

UNKONW\_ARROW = 63 // message arrow of 6 bit is 111111.

};

通信序列号，40 bits，每个发送模块单独负责编号，比如APP，brks云平台，自行车。但是回应类型的消息序号（40 bits）需与请求类型消息的序号一致。消息序号编号原则还需满足以下规则：

规则1：每个模块编号的消息不能重复，除非40个比特位全都编号完。

规则2：消息编号必须持久化。

# API接口定义

### 3.1 用户获取登录验证码请求消息

消息ID：EEVENTID\_GET\_MOBILE\_CODE\_REQ（0x01）

消息体：如下所示，mobile为必填字段。

{

“mobile”:”15200599665”

}

### 3.2 用户获取登录验证码回应消息

消息ID：EEVENTID\_GET\_MOBILE\_CODE\_RSP（0x02）

消息体：消息如下所示，code和msg为必填字段。

{

“code”:200,

“msg”:”success”

}