**API文档说明书**

目录

1.Matchvs1

1.1 JavaScriptSDK2

1.2 response3

1.3 init4

1.4 initResponse4

1.5 thirdBind.do5

1.6 login6

1.7 loginResponse7

1.8 createRoom8

1.9 createRoomResponse9

1.10 joinRoom10

1.11 joinRoomResponse11

1.12 sendEvent 12

1.13 sendEventResponse 13

1.14 sendEventNotify 14

2. WeChat15

2.1 wx.login16

2.2 wx.getUserInfo17

2.3 wx.getSetting18

2.4 wx.request19

2.5 wx.setUserCloudStorage20

2.6 wx.getFriendCloudStorage21

1. **Macthvs**

1.1 JavaScriptSDK

let Matchvs = require("matchvs.all");

let engine = new Matchvs.MatchvsEngine();

获取Matchvs引擎的一个实例，可以调用这个实例的方法实现联网对战功能。

1.2 Response

let Matchvs = require("matchvs.all");

let response = new Matchvs.MatchvsResponse();

获取Matchvs引擎的一个实例，可以调用这个实例的回调方法实现联网对战功能。

1.3 init

engine.init(response, channel, platform, gameID, appkey, gameVersion)

用于初始化

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| response | object | 回调对象 | {} |
| channel | string | 渠道，固定值 | "Matchvs" |
| platform | string | 平台，选择测试or正式环境 | "alpha" |
| gameID | number | 游戏ID | 200978 |
| appkey | string | 游戏App key，来自Matchvs控制台游戏信息 |  |
| gameVersion | number | 游戏版本，自定义，用于隔离匹配空间 |  |

1.4 initResponse

response.initResponse(status)

response是engine.init方法中传入的对象，init初始化完成之后，会异步回调initResponse方法

1.5 thirdBind.do

| **参数** | **类型** |  | **说明** |
| --- | --- | --- | --- |
| userID | number |  | 用户ID默认填0 |
| gameID | number |  | 你注册的游戏ID |
| openID | string |  | 获取第三方唯一认证号 |
| session | string |  | 校验值 |
| thirdFlag | number |  | 1-微信 |
| sign | string |  | 参数签名 |

第三方绑定接口，传入您的唯一身份识别号和一个session\_key 实现 userID绑定。

#### 请求地址

* release: <https://vsuser.matchvs.com/wc6/thirdBind.do>?
* alpha : <http://alphavsuser.matchvs.com/wc6/thirdBind.do>?

#### 请求参数

sign 计算方式：md5(appKey&gameID=value1&openID=value2&session=value3&thirdFlag=value4&appSecret)得到的md5值就是 sign参数的值。这个签名是取 32 为的小写字符。

appkey 和 appSecret 是在 matchvs 官网创建游戏时的信息

#### 返回参数

| **参数** | **类型** | **说明** |
| --- | --- | --- |
| data | json object | 接口返回数据 |
| status | int | 请求状态值 0-成功 1- 失败 |

* data

| **参数** | **类型** | **说明** |
| --- | --- | --- |
| avatar | string | 头像 |
| deviceid | string | 设备ID |
| gender | int | 暂无说明 |
| mac | string | mac 地址 |
| nickname | string | 昵称 |
| regTime | string | 注册时间 |
| token | string | 用户校验值 |
| userid | int | 用户ID |

1.6 login

engine.login(userID, token, deviceID)

#### 参数

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| userID | number | 用户ID，调用注册接口后获取 | 123546 |
| token | string | 用户token，调用注册接口后获取 | "" |
| deviceID | string | 设备ID，用于多端登录检测，请保证是唯一ID |  |

#### 返回值

| **错误码** | **含义** |
| --- | --- |
| 0 | 成功 |
| -1 | 失败 |
| -2 | 未初始化，请先调用初始化接口 |
| -3 | 正在初始化 |
| -5 | 已经登录，请勿重复登录 |
| -6 | 正在登出 |

## 1.7 loginResponse

Response.loginResponse(loginRsp)

#### 参数loginRsp的属性

| **属性** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| status | number | 状态返回  200 成功 402 应用校验失败，确认是否在未上线时用了release环境，并检查gameID、appkey 403 检测到该账号已在其他设备登录 404 无效用户  500 服务器内部错误 | 200 |
| roomID | string | 房间号 | "210039" |

#### 说明

* 登录Matchvs服务端，与Matchvs建立连接。
* 服务端会校验游戏信息是否合法，保证连接的安全性。
* 如果一个账号在两台设备上登录，则后登录的设备会连接失败。
* 如果用户加入房间之后掉线，再重新登录进来，则roomID为之前加入的房间的房间号。

1.8 createRoom

engine.createRoom(createRoomInfo, userProfile,watchSet)

#### 参数

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| createRoomInfo | object | 创建房间的信息 |  |
| userProfile | string | 玩家简介 | "" |
| watchSet | object<MVS.MsWatchSet> | 观战服务参数 |  |

## 1.9 createRoomResponse

Response.createRoomResponse(CreateRoomRsp)

#### 参数

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| status | number | 状态返回，200表示成功 400 客户端参数错误  500 服务器内部错误 | 200 |
| roomID | string | 房间号 | "210039" |
| owner | number | 房主 | 210000 |

#### 说明

* response是engine.createRoom方法中传入的对象，createRoom完成之后，会异步回调createRoomResponse方法

**1.10joinRoom**

engine.joinRoom(roomID, userProfile)

#### 说明

* 客户端可以通过调用该接口加入指定房间，roomID为加入指定房间的房间号
* 指定房间号必须是由 createRoom接口创建的房间。

#### 参数

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| roomID | string | 房间号 | "1344333" |
| userProfile | string | 玩家简介 |  |

**1.11joinRoomResponse**

response.joinRoomResponse(status, roomUserInfoList, roomInfo)

* 如果本房间是某个玩家调用joinRandomRoom随机加入房间时创建的，那么roomInfo中的owner为服务器随机指定的房主ID。在调用engine.createRoom主动创建房间时owner为创建房间者（即房主）的ID。以上两种情况下，如果房主离开房间，服务器均会指定下一个房主，并通过leaveRoomNotify通知房间其他成员。
* roomUserInfoList用户信息列表是本玩家加入房间前的玩家信息列表，不包含本玩家。
* roomUserInfoList中的玩家的userProfile的值来自于其他客户端调用joinRandomRoom时传递的userProfile的值。

#### 参数

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| status | number | 状态返回，200表示成功  400 客户端参数错误  404 指定房间不存在  405 房间已满  406 房间已joinOver  500 服务器内部错误 | 200 |
| roomUserInfoList | array | 房间内玩家信息列表 |  |
| roomInfo | object | 房间信息构成的对象 |  |

**1.12 sendEvent**

Engine.sendEvent(msg)

**注意** 给GameServer发送消息调用sendEventEx()。

#### 参数

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| msg | string | 消息内容 | "hello" |

#### 返回值

* 返回值为一个对象，该对象包含以下属性：

| **属性** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| result | number | 错误码，0表示成功，其他表示失败 | 0 |
| sequence | number | 事件序号，作为事件的唯一标识。客户端发送消息后收到的sendEventResponse 也会收到 sequence 标识，通过此标识来确定这个sendEventResponse 是由哪次sendEvent 发送的。主要用于在游戏中做信息同步的时候，网络传输都有延迟会出现sendEvent与sendEventResponse 收到顺序不同。 |  |

* 在进入房间后即可调用该接口进行消息发送，消息会发给房间里除自己外其他所有成员。回调函数是 sendEventResponse
* 同一客户端多次调用engine.sendEvent方法时，每次返回的sequence都是唯一的。但同一房间的不同客户端调用sendEvent时生成的sequence之间会出现重复。
* 可以发送二进制数据，开发者可以将数据用json、pb等工具先进行序列化，然后将序列化后的数据通过SendEvent的一系列接口发送

1.13 sendEventResponse

response.sendEventResponse(sendEventRsp)

#### 参数sendEventRsp的属性

| **属性** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| status | number | 状态返回，200表示成功 521 gameServer不存在，请检查是否已开启本地调试或在正式环境发布运行gameServer | 200 |
| sequence | number | 事件序号，作为事件的唯一标识，可以参考sendEvent，对这个字段的详细说明 | 231212 |

#### 说明

* 客户端调用engine.sendEvent或engine.sendEventEx发送消息之后，SDK异步调用reponse.sendEventResponse方法告诉客户端消息是否发送成功。

1.14 sendEventNotify

Response.sendEventNotify(eventInfo)

#### 参数eventInfo的属性

| **参数** | **类型** | **描述** | **示例值** |
| --- | --- | --- | --- |
| srcUserID | number | 推送方用户ID，表示是谁发的消息 | 321 |
| cpProto | string | 消息内容，对应[sendEvent](https://doc.matchvs.com/APIDoc/JavaScript#sendEvent)中的msg参数 | "hello" |

#### 说明

* 在其他客户端调用engine.sendEvent或engine.sendEventEx方法之后，本客户端的response.sendEventNotify会被SDK调用，调用时传入其他玩家的用户ID和发送的消息。

1. **WeChat**

**2.1 wx.login**

#### 调用接口获取登录凭证（code）。通过凭证进而换取用户登录态信息，包括用户的唯一标识（openid）及本次登录的会话密钥（session\_key）等。

#### 参数

##### Object object

| 属性 | 类型 | 必填 | 说明 | 最低版本 |
| --- | --- | --- | --- | --- |
| timeout | number | 否 | 超时时间，单位ms | [1.9.90](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| success | function | 否 | 接口调用成功的回调函数 |  |
| fail | function | 否 | 接口调用失败的回调函数 |  |
| complete | function | 否 | 接口调用结束的回调函数（调用成功、失败都会执行） |  |

2.2 wx.getUserInfo

获取用户信息。

#### 参数

##### Object object

| 属性 | 说明 |
| --- | --- |
| withCredentials | 是否带上登录态信息。当 withCredentials 为 true 时，要求此前有调用过 wx.login 且登录态尚未过期，此时返回的数据会包含 encryptedData, iv 等敏感信息；当 withCredentials 为 false 时，不要求有登录态，返回的数据不包含 encryptedData, iv 等敏感信息。 |
| lang | 显示用户信息的语言 |
| success | 接口调用成功的回调函数 |
| fail | 接口调用失败的回调函数 |
| complete | 接口调用结束的回调函数（调用成功、失败都会执行） |

2.3 wx.getSetting

获取用户的当前设置。**返回值中只会出现小程序已经向用户请求过的**[权限](https://developers.weixin.qq.com/miniprogram/dev/framework/open-ability/authorize.html)。

#### 参数

##### Object object

| 属性 | 类型 | 说明 |
| --- | --- | --- |
| success | function | 接口调用成功的回调函数 |
| fail | function | 接口调用失败的回调函数 |
| complete | function | 接口调用结束的回调函数（调用成功、失败都会执行） |

###### object.success 回调函数

**参数**

**Object res**

| 属性 | 说明 |
| --- | --- |
| authSetting | 用户授权结果 |

2.4 wx.request

发起 HTTPS 网络请求

#### 参数

##### Object object

| 属性 | 类型 | 说明 | 最低版本 |
| --- | --- | --- | --- |
| url | string | 开发者服务器接口地址 |  |
| data | string/object/ArrayBuffer | 请求的参数 |  |
| header | Object | 设置请求的 header，header 中不能设置 Referer。 content-type 默认为 application/json |  |
| method | string | HTTP 请求方法 |  |
| dataType | string | 返回的数据格式 |  |
| responseType | string | 响应的数据类型 | [1.7.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| success | function | 接口调用成功的回调函数 |  |
| fail | function | 接口调用失败的回调函数 |  |
| complete | function | 接口调用结束的回调函数（调用成功、失败都会执行） |  |

2.5 wx.setUserCloudStorage

对用户托管数据进行写数据操作。允许同时写多组 KV 数据

#### 参数

##### Object object

| 属性 | 类型 | 说明 |
| --- | --- | --- |
| KVDataList | Array.<[KVData](https://developers.weixin.qq.com/minigame/dev/api/KVData.html)> | 要修改的 KV 数据列表 |
| success | function | 接口调用成功的回调函数 |
| fail | function | 接口调用失败的回调函数 |
| complete | function | 接口调用结束的回调函数（调用成功、失败都会执行） |

**2.6 wx.getFriendCloudStorage**

拉取当前用户所有同玩好友的托管数据。该接口只可在开放数据域下使用

#### 参数

##### Object object

| 属性 | 类型 | 说明 |
| --- | --- | --- |
| keyList | Array.<string> | 要拉取的 key 列表 |
| success | function | 接口调用成功的回调函数 |
| fail | function | 接口调用失败的回调函数 |
| complete | function | 接口调用结束的回调函数（调用成功、失败都会执行） |

###### object.success 回调函数

**参数**

**Object res**

| 属性 | 类型 | 说明 |
| --- | --- | --- |
| data | Array.<[UserGameData](https://developers.weixin.qq.com/minigame/dev/api/UserGameData.html)> | 同玩好友的托管数据 |