# 台阶接口规范

**版本记录：**

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 作者 | 说明 |
| 2017-12-01 | 1.0 | 廖增祥 | 创建初始版本 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[台阶接口规范 1](#_Toc500019733)

[1接口概述 3](#_Toc500019734)

[1.1会话 3](#_Toc500019735)

[1.2连接与会话无关性 3](#_Toc500019736)

[1.3通讯协议规范 3](#_Toc500019737)

[2请求式消息 4](#_Toc500019738)

[2.1登录 4](#_Toc500019739)

[2.2心跳 4](#_Toc500019740)

[2.3游戏入座 5](#_Toc500019741)

[2.4游戏准备 5](#_Toc500019742)

[2.5叫分 6](#_Toc500019743)

[2.6放底牌 7](#_Toc500019744)

[2.7设置主花色 7](#_Toc500019745)

[2.8出牌 8](#_Toc500019746)

[2.9获取玩家的战况 8](#_Toc500019747)

[2.10玩家离开游戏 9](#_Toc500019748)

[2.11玩家登出游戏 9](#_Toc500019749)

[3通知消息 10](#_Toc500019750)

[3.1玩家加入游戏 10](#_Toc500019751)

[3.2玩家离开游戏 10](#_Toc500019752)

[3.3玩家准备/取消准备 11](#_Toc500019753)

[3.4游戏开始 11](#_Toc500019754)

[3.5发牌 11](#_Toc500019755)

[3.6轮到玩家叫分 12](#_Toc500019756)

[3.7玩家叫分 12](#_Toc500019757)

[3.8确认庄家 12](#_Toc500019758)

[3.9庄家扣底牌 13](#_Toc500019759)

[3.10庄家确定主花色 13](#_Toc500019760)

[3.11轮到玩家出牌 13](#_Toc500019761)

[3.12玩家出牌 14](#_Toc500019762)

[3.13闲家上分 14](#_Toc500019763)

[3.14闲家已破分 14](#_Toc500019764)

[3.15游戏结束 14](#_Toc500019765)

[4附录：错误码 16](#_Toc500019766)

# 1接口概述

本文档描述台阶客户端与服务器端的通讯协议规范，该游戏为双向通讯协议，协议的内容承载在TCP连接之上，本协议分为请求式消息和通知消息。

* 请求式消息由客户端发起，并由服务器端做出响应。
* 通知消息由服务器端产生，用于向客户端通知游戏的各个状态和数据。

## 1.1会话

客户端与服务器端主动建立TCP连接后，需要主动发起登录请求，服务器端判断登录成功后，会返回一个随机生成的32位uuid, 用于标识本次登录的会话，在后续任何请求中，客户端需要携带该会话标识，用于指示消息来源。

会话标识的过期时间为5分钟，在空闲时段，客户端需要主动发送心跳，以便维持会话。

## 1.2连接与会话无关性

客户端支持PC端与Android端，在接入无线网络时，TCP连接可能偶尔会断开，为了提升客户端的游戏体验，在网络异常断开时仍然允许用户再次尝试连接，并恢复游戏状态。

客户端登录成功后，需要始终存储本次登录成功后服务器端返回的会话标识。若网络异常断开，则客户端可继续尝试重新建立连接，并在连接成功后主动发起心跳，服务器端便可以通过该心跳请求中携带的会话标识恢复客户端游戏状态。

若客户端掉线超过5分钟，服务器端会清除该用户的在线标识，用户再次连接上时，无法通过心跳恢复会话，此时客户端必须重新登录。

## 1.3通讯协议规范

所有消息均由消息头和消息体组成，消息头可以判定消息的种类，可以区分请求、响应、通知。对于响应，还可以通过消息头判断出针对客户端的哪一条请求进行的回复。

**消息头的格式如下：**

|  |  |  |
| --- | --- | --- |
| 字段名称 | 长度 | 说明 |
| magic | 1字节 | 协议标识，必须为'L' |
| message\_type | 1字节 | 1=请求；2=响应；3=通知 |
| message\_id | 2字节 | 消息的标识，对于请求消息，服务器端会透传回来 |
| size | 2字节 | 消息体的长度，消息体为json格式 |

**消息体的格式：**

消息体均为application/json类型，根据消息类型，其格式说明如下：

* 对于请求消息，json文本中必须包含url字段，用于描述请求的资源名称，同时需要构造message\_id的值，以便收到服务器端的回应后能与请求对应；
* 对于响应消息，只需要保证消息头的message\_id与请求的值一致即可；
* 对于通知消息，json文本中必须包含url字段，用于描述通知的种类；

# 2请求式消息

## 2.1登录

用户必须先登录，并通过服务器端认证，成功后会获得认证的token, 该token在整个会话生命周期有效，并且在后续所有的请求中，必须携带此token, 用于标识该用户。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /user/login | 登录的url |
| login\_id | 用户名 | 需要在passport.moodwu.com上注册 |
| password | 密码 |  |

请求示例：

|  |
| --- |
| {"url": "/user/login", "login\_id": "lory", "password": "123456"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：登录成功  400：非法请求  401：登录失败  500：内部服务器错误 | 详见错误码表 |
| message | 附加消息 | 登录失败的原因 |
| token | 会话标识 | 登录成功后的会话标识 |
| timeout | token超时时间，单位为秒 | 客户端必须间隔发送心跳，以便保持会话 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK", "token": "abcdefg", "timeout": 300} |

## 2.2心跳

为了保证会话的有效性，客户端需要在token超时之前主动发起心跳，以便维持会话的在线状态。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /user/heartbeat | 心跳的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |

请求示例：

|  |
| --- |
| {"url": "/user/heartbeat", "token": "abcdefg"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  440：会话超时  500：内部服务器错误 | 详见错误码表 |
| message | 附加消息 | 心跳失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.3游戏入座

玩家可以通过该请求入座游戏桌。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/seatdown | 游戏入座的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |
| desktop\_id | 游戏桌编号 | 游戏桌编号，有效值为1-200 |
| seat\_id | 座位编号 | 有效值为1、2、3、4 |

请求示例：

|  |
| --- |
| {"url": "/game/seatdown", "token": "abcdefg", "desktop\_id": 1, "seat\_id": 1} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  800：游戏桌不存在  801：座位已被占用 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.4游戏准备

在入座游戏桌后，可以通过该请求准备/取消准备游戏状态，该请求必须在游戏未开始时发起。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/ready | 游戏准备的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |
| ready | 0：取消准备  1：准备 | 必须在游戏等待中方可发起 |

请求示例：

|  |
| --- |
| {"url": "/game/ready", "token": "abcdefg", "ready": 1} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  802：此状态不允许操作 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.5叫分

客户端在收到叫分通知时，在指定的时间内必须选择是否叫分。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/bid | 叫分的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |
| bid | 0：不叫  80：80分  125：台阶 | 0表示不叫分  其他值表示叫相应的分，根据实际值填充 |

请求示例：

|  |
| --- |
| {"url": "/game/bid", "token": "abcdefg", "bid": 125} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  802：此状态不允许操作 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.6放底牌

庄家需要通过该接口放置底牌。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/put\_bottom\_card | 放底牌的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |
| cards | 经过base64编码的底牌列表 | 二进制结构，客户端与服务器端协商约定 |

请求示例：

|  |
| --- |
| {"url": "/game/put\_bottom\_card", "token": "abcdefg", "cards": "base64"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  802：此状态不允许操作 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.7设置主花色

庄家扣底牌后，必须在指定时间内设置主花色。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/set\_master\_suit | 设置主花色的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |
| suit | "spade"：黑桃  "heart"：红桃  "club"：梅花  "diamond"：方块 | 必须设置为其中一种花色 |

请求示例：

|  |
| --- |
| {"url": "/game/set\_master\_suit", "token": "abcdefg", "suit": "heart"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  802：此状态不允许操作 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.8出牌

客户端在收到来自服务器端的出牌通知时，必须在指定的时间内出牌/跟牌。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/play\_card | 出牌的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |
| cards | 经过base64编码的牌 | 二进制结构，客户端与服务器端协商约定 |

请求示例：

|  |
| --- |
| {"url": "/game/play\_card", "token": "abcdefg", "cards": "base64"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  802：此状态不允许操作 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.9获取玩家的战况

在游戏任何时刻，客户端均可以展示所有成员的战况表，以便随时确定每个玩家的胜败及积分情况。客户端可以通过该接口逐个请求所有玩家的战况统计。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/get\_player\_points | 获取玩家战况的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |
| user\_id | 目标玩家 |  |

请求示例：

|  |
| --- |
| {"url": "/game/get\_player\_points", "token": "abcdefg", "user\_id", "900001"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  802：此状态不允许操作  803：玩家不在同一游戏桌 | 详见错误码表  必须在成功入座后方可调用，且被查询的人员必须在同一个游戏桌上 |
| message | 附加消息 | 失败的原因 |
| user\_id | 玩家 |  |
| points | 积分 |  |
| win\_count | 胜利次数 |  |
| lose\_count | 失败次数 |  |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK", "user\_id": "900001", "points": 0, "win\_count": 0, "lose\_count": 0} |

## 2.10玩家离开游戏

玩家可以通过该接口离开当前游戏桌。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /game/leave | 离开游戏桌的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |

请求示例：

|  |
| --- |
| {"url": "/game/leave", "token": "abcdefg"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误  802：此状态不允许操作 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

## 2.11玩家登出游戏

玩家可以通过该接口主动登出游戏，销毁当前会话。

请求格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /user/quit | 登出的url |
| token | 会话标识 | 登录时服务器端返回的有效会话标识 |

请求示例：

|  |
| --- |
| {"url": "/game/quit", "token": "abcdefg"} |

响应格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| result | 200：成功  400：非法请求  500：内部服务器错误 | 详见错误码表 |
| message | 附加消息 | 失败的原因 |

响应示例：

|  |
| --- |
| {"result": 200, "message": "OK"} |

# 3通知消息

## 3.1玩家加入游戏

玩家入座游戏桌后，便可接收所有其他玩家入座的通知。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/user\_in | 玩家加入游戏的url |
| user\_id | 用户编号 |  |
| nickname | 昵称 | 在客户端用于呈现 |

通知示例：

|  |
| --- |
| {"url": "/notice/user\_in", "user\_id": "900001", "nickname": "lory"} |

## 3.2玩家离开游戏

其他玩家离开游戏桌时，服务器端会向其他所有成员通知该事件。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/user\_leave | 玩家离开游戏桌的url |
| user\_id | 用户编号 | 离开游戏桌的用户编号 |

通知示例：

|  |
| --- |
| {"url": "/notice/user\_leave", "user\_id": "900001"} |

## 3.3玩家准备/取消准备

游戏桌上所有玩家（包括自己）在准备或取消准备游戏时，服务器端会向所有成员通知该事件。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/user\_ready | 玩家准备/取消准备的url |
| user\_id | 用户编号 | 玩家的用户编号 |
| ready | 0：取消准备  1：准备 |  |

通知示例：

|  |
| --- |
| {"url": "/notice/user\_ready", "user\_id": "900001", "ready": 1} |

## 3.4游戏开始

当所有玩家均已准备时，服务器端会向所有玩家通知游戏开始事件。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/game\_begin | 游戏开始的url |

通知示例：

|  |
| --- |
| {"url": "/notice/game\_begin"} |

## 3.5发牌

在游戏开始后，服务器端会向所有玩家下发通知，对应玩家的手牌通过该通知发出。为了游戏安全起见，服务器端不会将其他玩家的牌通知给玩家，后续版本若支持明牌机制则例外。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/deal | 发牌的url |
| user\_id | 被发牌的用户编号 | 目前只能收到自己的发牌结果 |
| cards | 经过base64编码的牌列表 |  |

通知示例：

|  |
| --- |
| {"url": "/notice/deal", "user\_id": "900001", "cards": "base64"} |

## 3.6轮到玩家叫分

台阶有叫分机制，第一局叫分时，系统会随机选择一名玩家作为第一轮叫分的起点。在人员未发生变化的情况下，从第二局起，则由上一轮的胜家作为第一轮叫分的起点。

叫分的机制是循环的，轮到玩家叫分时，玩家可以选择不叫，或者比上一个玩家叫分更高，直到所有人都不叫分为止，最终的玩家即作为庄家。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/turn\_to\_bid | 轮到叫分的url |
| user\_id | 轮到被叫分的用户编号 |  |
| timeout | 叫分超时时间，单位为秒 | 玩家必须在指定时间内选择，超时为不叫 |

通知示例：

|  |
| --- |
| {"url": "/notice/turn\_to\_bid", "user\_id": "900001", "timeout": 30} |

## 3.7玩家叫分

当玩家选择叫分或者不叫时，服务器端会向所有玩家发起通知事件。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/bid | 玩家叫分的url |
| user\_id | 叫分的用户编号 |  |
| score | 分数 | 该玩家叫分的分值 |

通知示例：

|  |
| --- |
| {"url": "/notice/bid", "user\_id": "900001", "score": 125} |

## 3.8确认庄家

根据游戏规则，一旦系统确定了最终的庄家，服务器端会向所有玩家通知该事件。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/banker\_confirmed | 确认庄家的url |
| user\_id | 庄家的用户编号 |  |
| score\_needed | 闲家所需上分的分值 |  |

通知示例：

|  |
| --- |
| {"url": "/notice/banker\_confirmed", "user\_id": "900001", "score\_needed": 80} |

## 3.9庄家扣底牌

庄家在扣完底牌后，服务器端会向所有玩家通知该事件。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/put\_bottom\_card | 庄家扣底牌的url |

通知示例：

|  |
| --- |
| {"url": "/notice/put\_bottom\_card"} |

## 3.10庄家确定主花色

庄家选定主花色后，服务器端向所有玩家通知该事件。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/suit\_confirmed | 庄家确定主花色的url |
| suit | "spade"：黑桃  "heart"：红桃  "club"：梅花  "diamond"：方块 | 主花色 |

通知示例：

|  |
| --- |
| {"url": "/notice/suit\_confirmed", "suit": "heart"} |

## 3.11轮到玩家出牌

根据游戏规则，当轮到某个玩家出牌时，系统会通过该接口通知所有玩家，客户端可以根据超时值进行倒计时。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/turn\_to\_play\_card | 轮到玩家出牌的url |
| user\_id | 轮到出牌的玩家用户编号 |  |
| timeout | 出牌的超时时间，单位：秒 | 若玩家超时，系统自动出最小的牌 |

通知示例：

|  |
| --- |
| {"url": "/notice/turn\_to\_play\_card", "user\_id": "900001", "timeout": 60} |

## 3.12玩家出牌

当玩家出牌时，服务器端会向所有玩家通知该事件，以便客户端能作出相应的呈现。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/play\_card | 玩家出牌的url |
| user\_id | 出牌的玩家用户编号 |  |
| cards | 经过base64编码的牌 | 二进制结构，客户端与服务器端协商约定 |

通知示例：

|  |
| --- |
| {"url": "/notice/play\_card", "user\_id": "900001", "cards": "base64"} |

## 3.13闲家上分

在完成一轮出牌后，系统若判断出闲家可以上分，则会通过该消息通知所有玩家。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/player\_score | 闲家上分的url |
| step\_score | 本轮上分的值 |  |
| total\_score | 本局总共已上的分值 |  |

通知示例：

|  |
| --- |
| {"url": "/notice/player\_score", "step\_score": 5, "total\_score": 20} |

## 3.14闲家已破分

在游戏过程中，若闲家刚刚破分，则服务器端会向所有玩家通知该事件，此时游戏仍然继续。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/score\_exceeded | 闲家已破分的url |

通知示例：

|  |
| --- |
| {"url": "/notice/score\_exceeded"} |

## 3.15游戏结束

在玩家的所有手牌均已打完，系统会判断胜负，并宣布游戏结束。该通知消息根据每个玩家的胜负情况分别进行发送，玩家根据points的值确定本轮获得或失去的积分。

通知格式：

|  |  |  |
| --- | --- | --- |
| 字段 | 值 | 说明 |
| url | /notice/game\_over | 游戏结束的url |
| score\_needed | 闲家需要上分的总分值 |  |
| total\_score | 闲家实际上分的总分值 |  |
| points | 积分 | 本轮获得的积分，胜利为正值，失败为负值 |

通知示例：

|  |
| --- |
| {"url": "/notice/game\_over", "score\_needed": 80, "total\_score": 90, "points": 5} |

# 4附录：错误码

|  |  |  |
| --- | --- | --- |
| 类别 | 错误码 | 说明 |
| 通用错误码 | 400 | 请求参数错误 |
| 401 | 登录失败，用户名或密码错误 |
| 440 | 会话已过期 |
| 系统错误码 | 500 | 内部服务器错误 |
| 业务错误码 | 800 | 游戏桌不存在 |
| 801 | 座位已被占用 |
| 802 | 此状态不允许执行该操作 |
| 803 | 玩家不在同一游戏桌 |