Author	时间	修改内容
FG TEAM	2017-07-19	整理文档
FG TEAM	2017-07-20	增加彩池推送协议
FG TEAM	2017-9-30	修改采集接口
FG TEAM	2017-12-07	1、增加采集接口字段详细描述 2、修改概述
FG TEAM	2017-12-20	1、修改概述 2、增加创建玩家范例
FG TEAM	2018-01-30	1、增加活动数据采集接口 2、增加根据时间进行分页采集接口 3、捕猎数据采集返回增加 type 4、增加 APP 下载二维码和登录 APP 二维码 5、捕猎周排行派彩
FG TEAM	2018-02-05	1、捕猎游戏采集数据增加 order_id 字段(买鱼收益存放鱼编号) 2、语言代码改为 zh-cn(兼容 zh_CN) 3、游戏列表返回增加排序和新增类别字段
FG TEAM	2018-02-06	1、修改买鱼系统数据返回，买鱼消耗和买鱼收益合并在一条注单返回
FG TEAM	2018-02-07	1、买鱼系统数据返回增加房间 id 和倍场率
FG TEAM	2018-02-09	1、最新文档地址迁移到共享云盘
FG TEAM	2018-03-30	增加免转钱包说明
FG TEAM	2018-03-30	增加存取筹码单号检查接口
FG TEAM	2018-04-02	增加 member_code 代替 openid (唯一标示玩家)

FG TEAM	2018-05-10	捕猎幸运奖增加删除线，增加第三方 app 加载 h5 游戏 webview 配置
FG TEAM	2018-05-16	启动游戏增加可选参数 owner_id
FG TEAM	2018-06-04	增加免转钱包和非免转钱包说明
FG TEAM	2018-06-10	采集活动数据增加新的活动类型（排行活动）
FG TEAM	2018-06-28	增加获取游戏详情页面接口 externaltransactionid 注意事项
FG TEAM	2018-07-31	捕猎游戏美人捕鱼增加 boss 赛功能
FG TEAM	2018-08-07	启动游戏和获取游戏列表增加多语言代码
FG TEAM	2018-08-17	APP 加载 FG ZIP 包流程说明
FG TEAM	2018-08-24	Android Webview 设置代码实例补充解决跨域问题等实例 和 iOS Webview 设置代码补充解决跨域问题实例
FG TEAM	2018-09-04	补充接口关键参数 openid、id、page_key 的长度说明
FG TEAM	2018-09-07	新增错误码 1511、1516
FG TEAM	2018-09-09	捕猎采集接口增加购买基金类型和等级奖励类型
FG TEAM	2018-09-21	捕猎排行派彩接口增加获取单代理上周榜数据
FG TEAM	2018-09-26	获取捕猎游戏进出房间金额接口
FG TEAM	2018-09-28	获取玩家汇总数据接口

FG TEAM	2018-10-09	汇总接口增加总奖金字段
FG TEAM	2018-10-18	修正带时间的分页采集数据的协议（时间部分）、v3_1 版分页采集 poker 协议（时间部分）和错误码 50、51 的描述
FG TEAM	2018-11-13	采集接口 Fish 游戏增加新类型 type=18（幸运宝箱）
FG TEAM	2018-11-21	增加推送跑马灯接口和查询玩家未结束局所在游戏
FG TEAM	2018-12-1	增加返利活动采集活动数据类型
FG TEAM	2018-12-6	增加启动 FG 大厅接口
FG TEAM	2018-1-11	带时间的采集数据接口增加时间范围限制(不超过一天)
FG TEAM	2019-1-24	接口全部改用 POST 方式和返回 Http 状态码改用 200；捕猎采集注单返回字段含义与其他游戏类型保持一致；增加 jp 接口
FG TEAM	2019-03-06	详情跳转接口增加用户验证
FG TEAM	2019-03-08	接口 5.4 修正接口注意描述
FG TEAM	2019-04-03	5.3 存取玩家筹码接口修改接口注意描述
FG TEAM	2019-04-08	增加 owner_id 支持字符串

FG TEAM	2019-04-15	采集接口 4.1、4.2、4.3、4.4、4.6、4.7、4.10 增加货币符号 currency 字段；捕猎排行派彩接口调整为拉取货币总榜数据和代理榜数据，相应增加 get_agent 字段和调整 rank_type 字段含义
FG TEAM	2019-04-22	采集数据接口新增：捕猎特殊武器类型（19、20、21），捕猎世界 boss 功能相关类型值（22、23、24）
FG TEAM	2019-04-23	下级代理采集数据增加时间范围限制说明；带时间的分页采集时间范围限制调整为 2 天
FG TEAM	2019-06-03	增加 game_id 启动游戏（先判断 game_code,再判断 game_id） 采集数据接口新增：玩家 ip 字段
FG TEAM	2019-06-11	活动数据采集接口增加棋牌福卡活动、捕猎抢任务活动数据类型
FG TEAM	2019-07-29	活动数据采集接口增加 FG 活动天榜、周榜和月榜派彩数据类型

版本修改记录

最新 API 文档地址：http://file.fungaming.com/f/f6401916271848ee803a/

1. 概述

1.1 玩家账号说明

系统是使用一个 table 来存储账号,所以不同代理商也是不能使用相同的账号。

1.2 openid 说明

openid 与玩家账号是一一对应的,openid 是根据玩家信息和当前时间戳等信息生成的。所以注册玩家后请保留 openid,供后面玩家相关的接口使用。备注：现接口做了兼容，可以使用 member_code 来进行玩家相关的操作，故可以不保留 openid。

1.3 游戏说明

FG 提供四大类型的游戏,捕猎、水果机、棋牌、老虎机,对应的代码是 fish、fruit、poker、slot。采集数据接口除了捕猎游戏返回的数据结构不一样，别的游戏返回的数据结构都是一样的，采集数据的时候要启动四个采集器采集不同类型的游戏（不包括活动数据，活动数据要另外采集）。

1.4 安全性

接口完全使用 https 协议,每个接口的 header 都需要提供 merchantname 和 merchantcode(FG 提供)以验证接入方的合法性。接口文档里面的 API_FG_HOST,merchantname,merchantcode 注册账号后会给接入方提供。

1.5 创建玩家流程

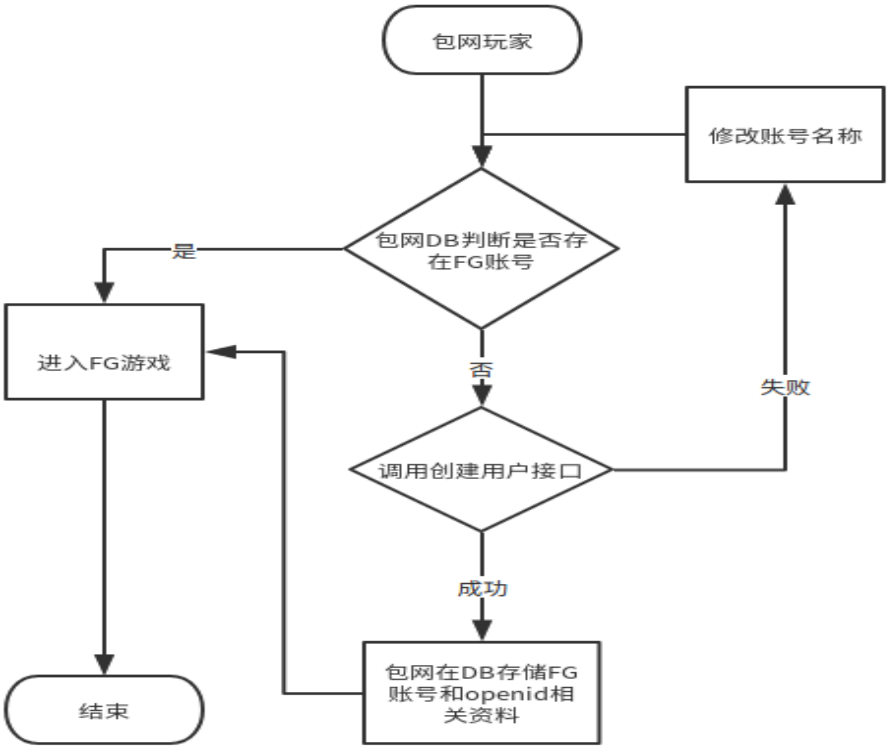

Figure1.1 创建玩家流程图

1.6 启动游戏流程

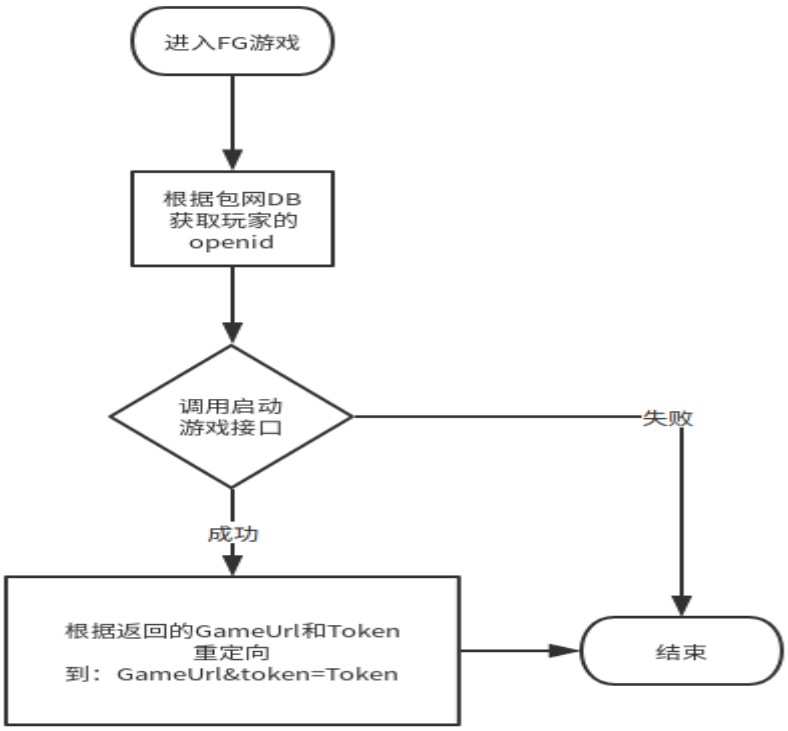

Figure1.2 启动游戏流程图

1.7 免转钱包和非免转钱包说明

非免转钱包：传统的包网商和游戏商之间的钱包交互模式是包网商有一个主钱包，每个游戏商有一个子钱包。玩家进入 FG 游戏商玩游戏，需要在包网会员中心把主钱包的钱转入一部分到 FG 子钱包才可以进行游戏，如果 FG 子钱包的额度不足需要继续游戏的时候，玩家需要重新进入会员中心转账到 FG 子钱包，每次子钱包额度不足都要进行手动转账操作，该模式这里简称为非免转钱包模式。

免转钱包模式：包网商只管理一个主钱包，玩家进入游戏的时候游戏前端会弹窗进行预授权相关额度的配置，游戏根据该配置通过接口（FG 免转钱包 API 说明文档）从包网商主钱包预授权部分额度到 FG 进行游戏，下次额度不足可自动转入预授权额度，玩家退出游戏的时候把剩下的额度自动同步到主钱包，该模式统称为免转钱包模式。免转钱包模式预授权相关的接口在 FG 免转钱包 API 说明文档

模式的选定：通过后台注册接入商账号的时候进行免转和非免转模式的设定，一旦选定了一种模式将无法切换为另一种模式。

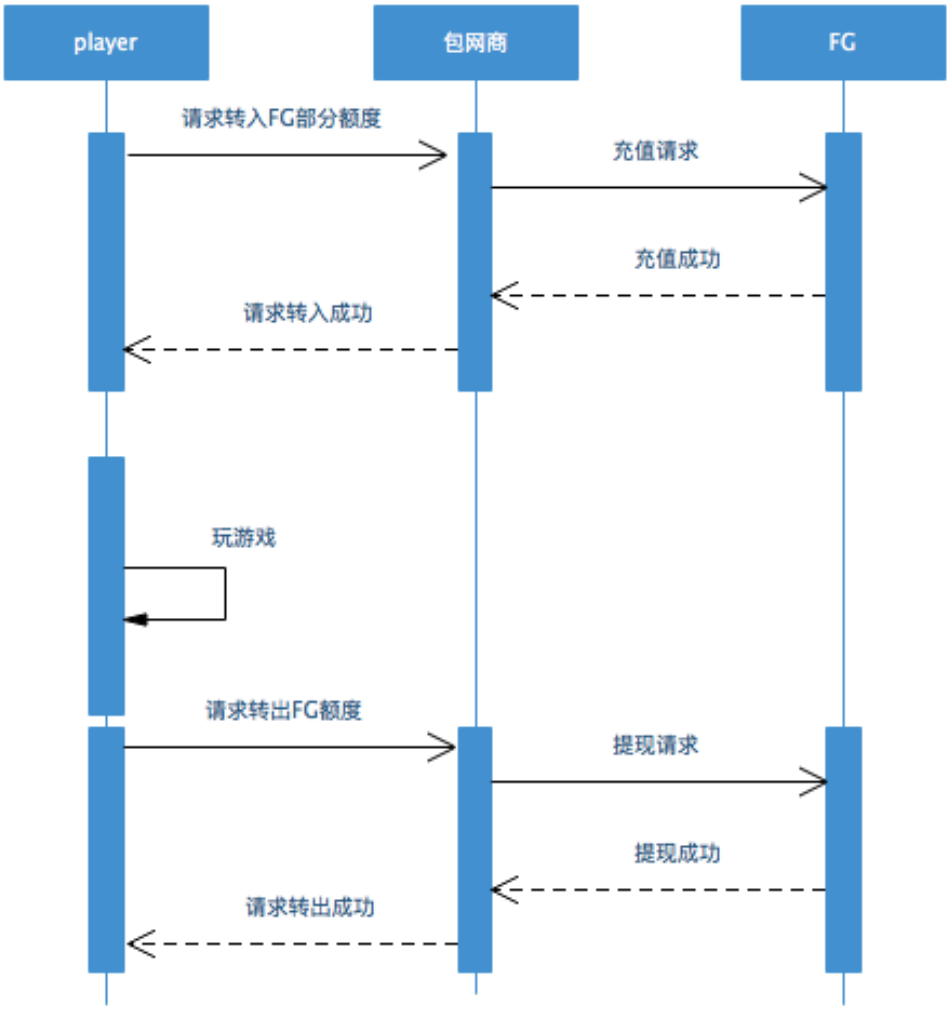

Figure1.3 非免转序列图

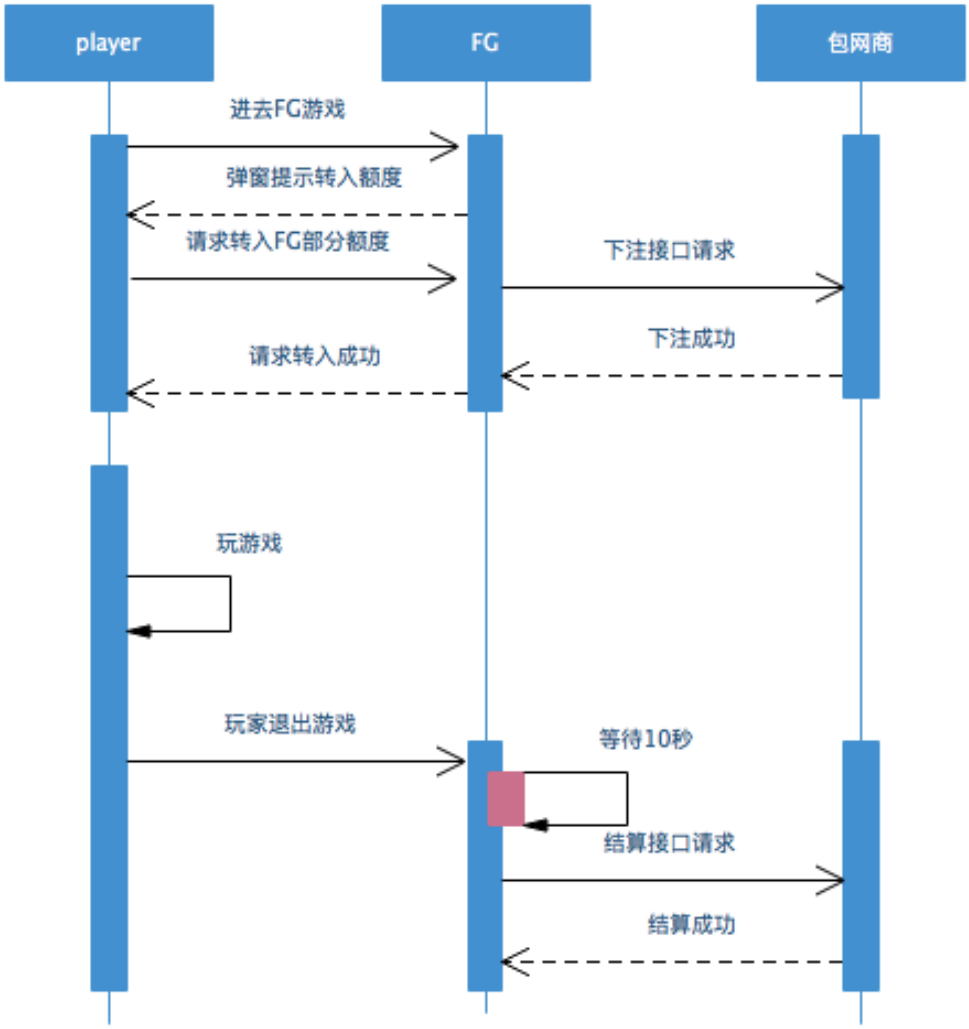

Figure1.4 免转序列图

1.8 启动游戏的 owner_id 参数说明

目前有些包网商下面有很多个厅主，厅主负责玩家的推广或者代理商的推广，那么包网商玩家总的 RTP 稳定的情况下，可能出现厅主 A 和厅主 B 玩家 RTP 差异很大的情况，为了保证各厅主之间玩家的 RTP 相对稳定，我们引入 owner_id 参数。FG 根据该参数进行玩家分组，同一个包网商同一个 owner_id 的玩家分配到同一集合，只有同一个集合的玩家可以进入同一个房间，每一个集合都使用独立的 RTP 风控以保证每个厅主 RTP 的稳定。如果不传入 owner_id 参数，FG 默认 owner_id 为 0，所有该包网商的玩家都将分配到同一个集合。使用方法请看 3.1 启动游戏

2. Emqtt - 推送 jp 奖池

2.1 Emqtt 相关说明

使用的 mqtt 版本：　　Erlang MQTT Broker 2.2
协议版本: MQTT V3.1.1 协议规范 QoS0/1/2 消息支持
开放的 tcp 端口：1883
使用的 topic: fg
mqtt 客户端工具: https://github.com/mqtt/mqtt.github.io/wiki/libraries
emqtt 官网：http://emqtt.com/

2.2 推送的协议格式

```
{
   "action": "push_jp",
   "data": [{"game_id": int 游戏 id, "jp": double 彩池值单位元}],
   "total": double 总的彩池
   "time": 10 位 unix 时间戳
}
```

3. Game 相关 API

3.1 启动游戏

此 API 用于启动一个游戏。请求参数可以使用 game_code 启动游戏，也可以使用 game_id 启动游戏。接口会先判断 game_code 参数，如果 game_code 不存在再判断 game_id。
方法一(使用 openid):
URL:POST v3/launch_game/
Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agen.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
openid	string	Players unique ID. Max length:60
game_code	String(optional)	游戏代码,游戏列表接口的 gamecode
game_id	Int(optional)	游戏 id，游戏列表接口的 service_id(game_code 和 game_id 任选一个启动游戏)
game_type	string	h5、app
language	String	zh-cn、en-us 等在语言代码章节有描述
ip	string	Client ip of player
return_url	string	Agent lobby url
owner_id	Int|string(50) (optional)	owner_id 表示玩家属于哪个厅主或者站点(用 owner_id 来区分玩家集合，计算 rtp 和分配房间都使用同一玩家集合)，跳转到 1.8 owner_id 参数说明

方法二(使用 member_code)：

URL:POST v3/launch_game/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	5.1 创建玩家的 member_code
game_code	String(optional)	游戏代码,游戏列表接口的 gamecode
game_id	Int(optional)	游戏 id，游戏列表接口的 service_id(game_code 和

		game_id 任选一个启动游戏)
game_type	string	h5、app
language	String	zh-cn、en-us 等在语言代码章节有描述
ip	string	Client ip of player
return_url	string	Agent lobby url
owner_id	Int|string(50) ((optional)	owner_id 表示玩家属于哪个厅主或者站点(用 owner_id 来区分玩家集合，计算 rtp 和分配房间都使用同一玩家集合) ,跳转到 1.8 owner_id 参数说明

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    -d "openid=2BrOI2X8"\
    -d "game_code=ono"\
    -d "game_type=h5"\
    -d "language=zh-cn"\
    -d "return_url=http://baidu.com"\
    -d "ip=192.168.10.2"\
  "https://API_FG_HOST/v3/launch_game"
```

OR

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    -d "member_code=yourname"\
    -d "game_code=ono"\
    -d "game_type=h5"\
    -d "language=zh-cn"\
    -d "return_url=http://baidu.com"\
    -d "ip=192.168.10.2"\
https://API_FG_HOST/v3/launch_game
```

Success-Response-h5

```
HTTP/1.1 200 OK
{
  "code":0,
```

```
  "msg": "success",
  "data": {
    "game_url": "游戏地址",
    "name": "游戏名称",
    "token": "token"
    }
}
```

Success-Response-app

```
HTTP/1.1 200 OK
{
  "code":0,
  "msg": "success",
  "data":{
     "game_url":"",
     "meta":“app 加载 ZIP 包启动游戏的 meta 信息”,
     "name":"美人捕鱼",
     "token":"28FC5D11508EC9FB"
}}
```

Error-Response

```
HTTP /1.1 200 OK
{
  "code" :  错误码,
   "msg" :  错误信息,
   "data" : {}
}
```

如果返回值是：game_url=”https://API_FG_HOST/game?type=h5&gamecode=fish_mm” token=”1B8DF7187B614E5D”

那么进入游戏的 url 为：https://API_FG_HOST/game?type=h5&gamecode=fish_mm&token=1B8DF7187B614E5D

语言版本只支持 zh-cn,终端 game_type 手机版和 pc 版都使用 h5,前端自动根据设备进行适配

meta 的使用说明：

解压游戏的 zip 包后在 index.html 的<head>和<meta 之间加入启动游戏获取的 meta 数据然后保存 index.html，使用 webview 加载 index.html?language=zh-cn 进入游戏。每次启动游戏都要按照这个流程。

APP 加载 FG ZIP 包游戏启动流程图

3.2 启动试玩游戏

此 API 用于启动试玩游戏。请求参数可以使用 game_code 启动游戏，也可以使用 game_id 启动游戏。接口会先判断 game_code 参数，如果 game_code 不存在再判断 game_id。

URL:POST v3/launch_free_game/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agen.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
game_code	String(optional)	游戏代码,游戏列表接口的 gamecode
game_id	Int(optional)	游戏 id，游戏列表接口的 service_id(game_code 和 game_id 任选一个启动游戏)
game_type	string	h5、app
language	String	zh-cn、en-us 等在语言代码章节有描述
ip	string	Client ip of player
return_url	string	Agent lobby url

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    -d "game_code=ono"\
    -d "game_type=as"\
    -d "language=zh_CN"\
    -d "return_url=http://baidu.com"\
    -d "ip=192.168.10.2"\
  "https://API_FG_HOST/v3/launch_free_game"
```

Success-Response-h5

```
HTTP/1.1 200 OK
{
  "code":0,
  "msg": "success",
  "data": {
```

```
        "game_url": "游戏地址",
        "name": "游戏名称",
        "token": "token"
    }}
```

Success-Response-app

```
    HTTP/1.1 200 OK
    {
       "code":0,
       "msg": "success",
       "data": {
              "game_url":"",
              "meta":“app 重构 index.html  的 meta 数据”,
              "name":"美人捕鱼",
              "token":"28FC5D11508EC9FB"
    }}
```

Error-Response

```
    HTTP /1.1 200 OK
    {
       "code" :  错误码,
        "msg" :  错误信息,
        "data" : {}
    }
```

如果返回值是：game_url=”https://API_FG_HOST/game?type=h5&gamecode=fish_mm” token=”1B8DF7187B614E5D”

那么进入游戏的 url 为：https://API_FG_HOST/game?type=h5&gamecode=fish_mm&token=1B8DF7187B614E5D

语言版本只支持 zh-cn,终端 game_type 手机版和 pc 版都使用 h5,前端自动根据设备进行适配

3.3 获取游戏列表

此 API 用于获取游戏列表

URL:POST v3/games/game_type/:terminal/language/:lang/

Header

字段	类型	说明
Accept	string	application/json

merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
terminal	string	h5、app
lang	string	zh-cn、en-us 等在语言代码章节有描述

Example usage

```
curl -XPOST -H "Accept: application/json" \
     -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    "https://API_FG_HOST/v3/games/game_type/h5/language/zh-cn"
```

Success-Response

```
HTTP/1.1 200 OK
 {
    "code":0,
    "msg": "success",
    "data": [
    {
       "name": "游戏名称",
       "gamecode": "游戏代码",
       "img": "方形游戏图片资源",
       "service_id": "游戏 id",
       "game_url": "游戏地址"
       "gt":"fish/poker/slot/fruit“
       "ishot":0 是否热门(0 非热门,1 热门),
       "isnew":0 是否新游戏(0 非新游戏,1 新游戏),
       "isrecommend":   是否推荐游戏(0 非推荐游戏,1 推荐游戏)
       ”sort”:  排序权重越大游戏越排在前面,
       “small_img”:”圆形游戏图片资源”
       “app_special_img”:”app 特殊图标资源”
       “big_img”:”游戏大图资源”
       // terminal 是 app 额外增加的字段
       “version”:”zip 包版本号信息”
       “zip”:”zip 包下载地址”
     }, …
    ]
 }}
```

Error-Response

HTTP /1.1 200 OK

```
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

返回的游戏列表已经经过排序,包网要根据返回的 ishost(是否热门)
,isnew(是否新游戏),isrecommend(是否推荐游戏),进行分类显示，terminal 是 app 会返回 h5 游戏的 zip 下载包资源地址，第三方 app 可以先下载游戏包然后进行游戏，加快游戏的二次加载速度，具体实现流程：APP 加载 FG 游戏 zip 包流程。

3.4 APP 大厅下载二维码

此 API 用于获取 APP 大厅下载二维码

URL:POST v3/app/download/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
                -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
                "https://API_FG_HOST/v3/app/download"
```

Success-Response

HTTP/1.1 200 OK

```
  {
    "code":0,
    "msg": "success",
     "data": "...=="
  }
```

Error-Response

```
HTTP /1.1 200 OK
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

使用说明：将返回的 data 数据（即二维码信息）,放在 html 的 img 标签 src 中,即可显示二维码图片..因为下载地址不会经常改变所以接入方可以拉取二维码后直接固定在会员中心展示页面.玩家在会员中心中可以使用浏览器扫描该二维码进行 FG APP 大厅的下载,然后安装 APP,使用 APP 扫描会员中心的 FG 登录二维码即可以进行游戏.登录二维码的获取请看 3.5.

3.5 APP 登录二维码

此 API 用于获取登录 APP 大厅的二维码

方法一(使用 openid):

URL:POST v3/app/get_token_qr/:open_id

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
open_id	string	Players unique ID. Max length:60

方法二(使用 member_code):

URL:POST v3/app/get_token_qr/member_code/:member_code

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	5.1 创建玩家的 member_code

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
                -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
              https://API_FG_HOST/app/get_token_qr/2BrOI2X
OR
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
                -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
            https://API_FG_HOST/v3/app/get_token_qr/member_code/yourname
```

Success-Response

```
    HTTP/1.1 200 OK
      {
       "code":0,
       "msg": "success",
       "data": {
            "token_qr_code": ""
       }}
```

Error-Response

```
    HTTP /1.1 200 OK
    {
      "code" : 错误码,
       "msg" : 错误信息,
       "data" : {}
    }
```

使用说明：将返回的 data 数据（即二维码信息）,放在 html 的 img 标签 src 中,即可显示二维码图片..把该二维码展示在玩家的会员中心中,玩家通过 3.4 下注 APP 安装然后使用该 APP 扫描登录二维码即可进入 FG 的 APP 大厅进行游戏.APP 登录的账号信息会共用接入 FG 的玩家账号.

3.6 推送跑马灯接口

此 API 用于推送跑马灯到 FG 内的游戏

URL:POST v3/game/notice

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
notice_id	int	公告 id(必须小于 9999 且唯一)
game_ids	string	Json 列表，游戏 id 列表.[] 为全部游戏. [5001] 为单个游戏.
start_time	int	开始时间 10 位时间戳,跑马灯开始播放时间
end_time	int	结束时间 10 位时间戳，跑马灯结束播放时间
interval	int	循环间隔单位秒
content	String	Json 跑马灯内容列表.[{“zh-cn”:<string:中文>,”en-us”:<string:英文>},…]. 如果内容有多条播放顺序为从第一条开始播放间隔 interval 后继续第二条播放以此类推。每条公告包含了中文和英文内容，玩家在中文版游戏显示中文公告，在英文版游戏显示英文版公告

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
                -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
                -d "notice_id=11 "      \
                -d "game_ids =[5001]"      \
                -d “start_time= 1542797109”
                -d “end_time= 1542797119”
                -d “interval=5”
                -d “content=[{\”zh-cn\”:\”你好 1\”,\”en-us\”:\”hello1\”},{\”zh-cn\”:\”
你好 2\”,\”en-us\”:\”hello2\”}]”
                "https://API_FG_HOST/v3/game/notice"
```

Success-Response

```
HTTP/1.1 200 OK
  {
   "code":0,
   "msg": "success",
```

```
  "data": {}
}
```

Error-Response

```
HTTP /1.1 200 OK
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

3.7 启动 FG 大厅

此 API 用于启动一个 FG 的试玩或者正式玩大厅
方法一(使用 openid):
URL:POST v3/launch_lobby/
Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agen.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
openid	String(optinal)	可选参数，正式玩家为创建玩家返回的 openid，试玩玩家不需要该参数
language	String	zh-cn、en-us 等在语言代码章节有描述
lobby_code	String	大厅唯一代码： 暂时只有棋牌大厅(chess)
owner_id	Int (optional)	可选参数，owner_id 表示玩家属于哪个厅主或者站点(用 owner_id 来区分玩家集合，计算 rtp 和分配房间都使用同一玩家集合)，跳转到 1.8 owner_id 参数说明

方法二(使用 member_code):

URL:POST v3/launch_lobby/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agen.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	String(Optional)	可选参数,正式玩家为 5.1 创建玩家的 member_code, 试玩玩家不需要该参数
language	String	zh-cn、en-us 等在语言代码章节有描述
lobby_code	String	大厅唯一代码： 暂时只有棋牌大厅(chess)
owner_id	Int|string(50) (optional)	可选参数，owner_id 表示玩家属于哪个厅主或者站点(用 owner_id 来区分玩家集合，计算 rtp 和分配房间都使用同一玩家集合)，跳转到 1.8 owner_id 参数说明

Example usage(正式玩)

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    -d "openid=2BrOI2X8"\
    -d "lobby_code=chess"\

    -d "language=zh-cn"\
  "https://API_FG_HOST/v3/launch_lobby"
```

OR(试玩)

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    -d "lobby_code=chess"\
    -d "language=zh-cn"\
https://API_FG_HOST/v3/launch_game
```

Success-Response

HTTP/1.1 200 OK

{

"code":0,

"msg": "success",

"data":{"lobby_url":"https://LOBBY_HOST/webLobby?token=28FC5D11508EC9F

B&language=zh-cn&lobby_code=chess"

}}

Error-Response

HTTP /1.1 200 OK

{

"code" : 错误码,

"msg" : 错误信息,

"data" : {}

}

注意：FG 大厅分为试玩大厅和正式玩大厅，接口不传送 member_code 或者 openid 参数的时候为进入试玩大厅。通过该接口获取 FG 大厅的 lobby_url,根据接口返回的 lobby_url 重定向到 FG 大厅可进行游戏。

4. Log 相关 API

4.1 分页采集数据

此 API 用于获取玩家下注日志

分页采集数据

第一次： "v3/agent/log_by_page/gt/fish/"

第二次使用:"v3/agent/log_by_page/gt/fish/page_key/g2gCbgYAt”

page_key 丢失: "v3/agent/log_by_page/gt/fish/id/100"

如果沒有任何數據 page_key 会返回 none.id 不存在 page_key 也返回 none

URL:POST v3/agent/log_by_page/gt/:gt/[page_key/:page_key|id/:id]

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.

merchantcode	string	Password of the agent

Parameter

字段	类型	说明
gt	string	fish/poker/slot/fruit/
id	string	下注日志中的 id；最大长度 20
page_key	string	上一次采集返回的分页码；最大长度 50

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
     -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
     "https://API_FG_HOST/v3/agent/log_by_page/gt/fish/"
      or
     "https://API_FG_HOST/v3/agent/log_by_page/gt/fish/id/100"
      or
     "https://API_FG_HOST/v3/agent/log_by_page/gt/fish/page_key/g2gCbgYAtmktd
F0BbQAAAAQxNTM5"
```

Success-Response

```
HTTP/1.1 200 OK
{
   "code":0,
   "msg": "success",
   "data":{
      "data": [{
          "id": string "局 id",
          "total_agent_uid":   int "总代理商 id",
          "agent_uid":int "代理商 id",
          "player_name": string "玩家名称",
          "game_id": int "游戏 id",
          "gt": string "游戏类型代码(fish,slot,fruit,poker)"
          "device":int  设备类型 1->PC 2-> IOS 横  3->IOS 竖  4-> android 横，
5->android 竖，  6->其他横,7->其他竖
          "time": int "注单结束时间 10 位时间戳",
          "end_chips": double "结束筹码",
          "all_bets": double "总投注(poker 对应后台的是有效打码)",
          "all_wins": double "总奖金",
```

"jackpot_bonus": double "jackpot 奖金保留 4 位小数(该奖金由 FG 出,不算入结算)",
"jp_contri": double "JP 贡献保留 4 位小数(当前记录下注的额度抽取进入 jackpot,一般抽取比例是千分三,单位元)",
"result": double "all_wins-all_bets(对应后台的收支)",
"currency":货币符号,
"ip": string 玩家 ip,

注意：下面字段为 gt : fish 独有

"type":int 类型 1->场景捕鱼账单.2->jp 奖账单.6->买鱼注单.14->激光炮.15->boss 赛功能.16->基金购买.17->等级奖金.18->幸运宝箱.19->电磁炮.20->连环炸弹.21->闪电. 22->世界 boss 场景.23->世界 boss 榜奖励.24->最后一击奖励
"scene_id": int 场景 id,
"bullet_count": int 子弹个数,
"start_chips": double 开始筹码(单位元),
"start_time": unixtimestamp 开始时间
},...],
"page_key": "分页符"
}}

Error-Response

```
HTTP /1.1 200 OK
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

分页采集的使用说明：FG 提供了 4 大类游戏接口对应的 gt 简称为 fish(捕猎),poker(棋牌),slot(老虎机),fruit(水果机),由于技术架构 4 类游戏存放在不同的数据表中,所以接入方要启动 4 个定时器来采集数据.分页采集接口接入方要保留每次返回的 page_key,下一页的采集需要传入新的 page_key.该接口没有提供时间参数,固定使用[0,当前 unix 时间戳－10]时间范围来进行分页.-10 说明数据有 10 秒的延迟

* 下级代理采集数据时需要提供时间范围（即 start_time、end_time 字段，详见 4.2 接口），否则 start_time 默认设置为当前时间-12 小时，end_time 默认设置为当前时间+12 小时

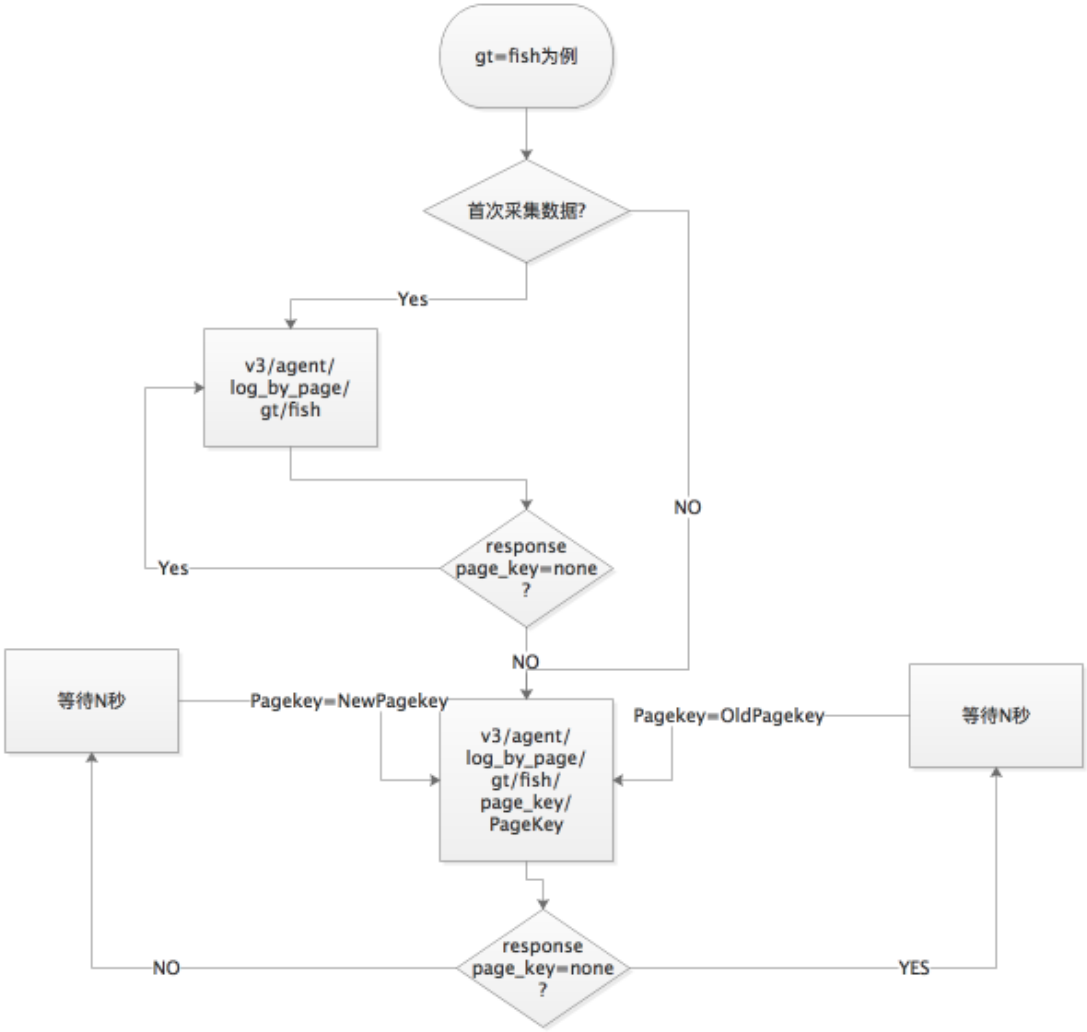

Figure4.1 分页采集流程图

4.2 带时间的分页采集数据(时间范围不超过两天)

此 API 根据时间获取玩家下注日志,使用方法跟 4.1 除了请求路径不一样别的结构都一样

URL:POST

v3/agent/log_by_page/gt/:gt/[page_key/:page_key|id/:id]/start_time/:start_time/end_time/:end_time

注意： 这里的 start_time 和 end_time 为 10 位时间戳,如果返回的 page_key=none 表示该时间段内已经没有新的数据,要注意服务器数据有延迟情况,数据延迟为 10 秒。时间返回不能超过一天

4.3 v3_1 版分页采集 poker(结构增加 total_bets)

此 API 使用方法跟 4.1 一样,请求路径由 v3 改为 v3_1,路径可增加时间参数,返回的结构增加 total_bets 字段,该接口主要为某些接入方需求定做,如果接入方不关注 total_bets 字段可以忽略该接口

URL:POST

v3_1/agent/log_by_page/gt/:gt/[page_key/:page_key|id/:id]/[start_time/:start_time/end_time/:end_time]

Success-Response-poker

HTTP/1.1 200 OK

```
{
    "code":0,
    "msg": "success",
    "data": {
        "data": [{
            "id": string "局 id",
            "total_agent_uid":   int "总代理商 id",
            "agent_uid":int "代理商 id",
            "player_name": string "玩家名称",
            "game_id": int "游戏 id",
            "gt": string "游戏类型代码(slot,fruit,poker)"
            "device":int  设备类型 1->PC 2-> IOS 横  3->IOS 竖  4-> android 横，5->android 竖，6->其他横,7->其他竖
            "time": int "注单结束时间 10 位时间戳",
            "end_chips": double "结束筹码",
            "all_bets": double "总投注(poker 对应后台的是有效打码)",
           "all_wins": double "总奖金",
            “total_bets”: double “总下注(poker 对应后台的总下注字段包括预扣金额)”
            "jackpot_bonus": double "jackpot 奖金保留 4 位小数",
            "jp_contri": double "JP 贡献保留 4 位小数",
            "result": double "all_wins-all_bets(对应后台的收支)",
            "currency":货币符号,
            "ip": string  玩家 ip
              },...],
        "page_key": "分页符"}
 }
```

Error-Response

```
HTTP /1.1 200 OK
{
  "code" :  错误码,
```

"msg" : 错误信息,
"data" : {}
}

4.4 分页采集活动数据

此 API 接口主要用于采集活动相关数据,这里的活动相关奖励全部由 FG 承担,如果不需要该数据的接入方可以忽略该接口,使用方法跟 4.1 接口一样.

URL:POST v3/agent/log_by_page/gt/activity/[page_key/:page_key|id/:id]

Success-Response-activity

HTTP/1.1 200 OK

```
{
"code":0,
"msg":"success",
"data":{
   "data": [{
       "id": string "局 id",
       "type":   int "活动子类型",
       "master_type":int "活动类型",
      "player_name": string "玩家名称",
       "game_id": int "游戏 id",
       "time": int "10 位时间戳",
       "num": double  单位元,
       "currency":货币符号
       }...],
   "page_key": "分页符"
}}
```

Error-Response

```
HTTP /1.1 200 OK
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

活动类型列表

活动类型	活动子类型
1 (元旦活动)	12 捕猎元旦奖励

2 (2018 春节活动)	13 捕猎红包奖励 50 捕猎春节活动排行奖励 51 slot 春节活动排行奖励
3 (大奖赛活动)	7 大奖赛报名费, 8 大奖赛奖金
4（排行活动）	13 捕猎红包奖励 50 捕猎红包排行活动奖励 51 slot 排行活动奖励 52 捕猎排行活动奖励 53 FG12 月返利活动日榜派彩 54 FG12 月返利活动半月榜派彩 55 棋牌福卡活动 56 捕猎抢任务活动 57 FG 活动天榜派彩 58 FG 活动周榜派彩 59 FG 活动月榜派彩

注意:所有的活动奖励数据都放一个数据表,所以 4 大类游戏的活动数据只使用这个接口采集就可以了

4.5 根据时间获取游戏总的记录数(时间范围不能超过一天)

此 API 用于获取时间段内总的游戏记录数

URL:POST

v3/agent/log_by_count/gt/:gt/start_time/:start_time/end_time/:end_time/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
gt	string	fish/poker/slot/fruit
start_time	number	10 位开始时间戳
end_time	number	10 位结束时间戳

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    "https://API_FG_HOST/v3/agent/log_by_count/gt/fish/start_time/1505207707/
end_time/1505207776/"
```

Success-Response-fish/slot/fruit/poker

```
HTTP/1.1 200 OK
{
  "code":0,
  "msg":”success”,
  "data": {"total": int  当前时间段总的记录数}
  }
```

Error-Response

```
HTTP /1.1 200 OK
{
   "code" :  错误码,
    "msg" :  错误信息,
    "data" : {}
}
```

4.6 捕猎排行派彩

此 API 获取上周全服捕猎某一款游戏派彩排行榜,每周一 0 点统计上周的排行榜,全服排行榜只保留 100 名；rank_type 字段为 0 时，为货币排行总榜，榜内玩家（100 名）包含了别的包网的玩家，接入方要自己进行排行榜数据处理；

rank_type 字段为 1 时，排行榜所有玩家（100 名）为自己包网的玩家；该接口主要为某些接入方需求定做,如果接入方不关注该数据可以忽略该接口.

```
rank_type=0 时，拿代理对应货币类型的货币总榜数据

rank_type=1，get_agent=0 时，拉取直属代理榜数据

rank_type=1，get_agent=1 时，只允许总社拉直属和下级代理的数据；下级代理账号不允许此项操作
```

URL:POST
v3/fish/player_rank/game_id/:game_id/[rank_type/:rank_type]/[get_agent/:get_agent]

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
game_id	Number	the id of game.
rank_type	Number	排行榜类型（0：代理账号对应货币类型上周榜[不区分代理]，1：代理上周榜）；字段可选，不填默认为 0
get_agent	Number	（该字段仅当 rank_type 为 1 时有意义）是否拉下级代理的数据；（值为 0 时拉取代理账号直属的代理榜；值为 1 时允许总社拉取直属和下级代理的数据，下级代理账号不允许该项操作）；字段可选，不填默认为 0

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
```

-H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
"https://API_FG_HOST/v3/fish/player_rank/game_id/5001/rank_type/1"

Success-Response

HTTP/1.1 200 OK

```
{
  "code":0,
  "msg":"success,
  "data":[
    {
      "rank":int 排名 1-100,
      "user_name":string 用户名,
      "own": int 0 为玩家是别的包网玩家，1 为本接入方玩家
      "all_bonus": double 累积奖金，单位元
      "create_time":int 玩家注册时间(10 位时间戳),
      "time": int 计算排行榜的时间戳(10 位时间戳),
      "currency":货币符号,
    },
    ...
  ]}
```

Error-Response

HTTP /1.1 200 OK

```
{
  "code" : 错误码,
  "msg" : 错误信息,
  "data" : {}
}
```

4.7 旧版根据时间采集游戏记录(非建议使用)

此 API 根据时间段返回最旧的 size 条数据,下次采集的开始时间设置为返回的最后一条数据的注单结束时间,以此类推直到该时间段内没有数据返回那么该时间段内的数据采集完毕,因为该接口使用不当容易造成漏单问题,所以不建议接入方使用.

URL:POST v3/agent/log_by_time/gt/:gt/start_time/:start_time/end_time/:end_time/size/:size

Header

字段	类型	说明

Accept	string	application/json
merchantname	string	accountname of the agen.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
gt	string	fish/poker/slot/fruit/activity/
start_time	Number	10 位开始时间戳
end_time	Number	10 位结束时间戳
size	Number	一次返回最大记录数,范围是 1-3000

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
     -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
     "https://API_FG_HOST/v3/agent/log_by_time/gt/fish/start_time/1505207707/end_time/1505207776/size/10"
```

Success-Response-fish/slot/fruit/poker

```
HTTP/1.1 200 OK
 {
      "code":0,
      "msg":"success",
      "data": {
          “data”:[{
              "id": string "局 id",
               "total_agent_uid":   int "总代理商 id",
               "agent_uid":int "代理商 id",
               "player_name": string "玩家名称",
               "game_id": int "游戏 id",
               "gt": string "游戏类型代码(fish,slot,fruit,poker)"
               "device":int  设备类型 1->PC 2-> IOS 横  3->IOS 竖  4-> android 横，5->android 竖，  6->其他横,7->其他竖
               "time": int "注单结束时间 10 位时间戳",
               "end_chips": double "结束筹码",
               "all_bets": double "总投注(poker 对应后台的是有效打码)",
               "all_wins": double "总奖金",
```

"jackpot_bonus": double "jackpot 奖金保留 4 位小数(该奖金由 FG 出,不算入结算)",
"jp_contri": double "JP 贡献保留 4 位小数(当前记录下注的额度抽取进入 jackpot,一般抽取比例是千分三,单位元)",
"result": double "all_wins-all_bets(对应后台的收支)",
"currency":货币符号,
"ip": string 玩家 ip

注意：下面字段为 gt : fish 独有

"type":int 类型 1->场景捕鱼账单.2->jp 奖账单.6->买鱼注单.14->激光炮.15->boss 赛功能.16->基金购买.17->等级奖金.18->幸运宝箱
"scene_id": int 场景 id,
"bullet_count": int 子弹个数,
"start_chips": double 开始筹码(单位元),
"start_time": unixtimestamp 开始时间
},...]}
}

Success-Response-activity
HTTP/1.1 200 OK
{
"code":0,
"msg":"success",
"data": {
"data": [{
"id": string "局 id",
"type": int "活动子类型",
"master_type":int "活动类型",
"player_name": string "玩家名称",
“game_id”: int “游戏 id”,
"time": int "10 位时间戳",
“num”: double 单位元},
"currency":货币符
...]
}
}

Error-Response
HTTP /1.1 200 OK

```
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

4.8 获取游戏详情页面跳转路径

此 API 用于获取游戏详情页面的 url

URL:POST v3/agent/log_detail_url /gt/:gt/id/:id/[member_code/:member_code]

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
gt	string	fish/slot/poker/fruit
id	string	采集数据返回对应的 id；最大长度 20
member_code	string(optional)	可选参数,正式玩家为 5.1 创建玩家的 member_code. 如果有这个参数那么会判断该 id 是否属于这个玩家

Example usage

```
curl    -XPOST -H 'Accept: application/json'-H "Accept: application/json" \
                  -H "merchantname: dl_one" \
                  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
“https://API_FG_HOST/v3/agent/log_detail_url/gt/fish/id/9081244075188225/ ”
```

Success-Response

```
    HTTP/1.1 200 OK
    {
        "code":0,
        "msg":"success",
```

"data":{
url":https://RedirectHost/
/~admin/fish/analysis/scene/details/dataTable/9081244075188225?token=
AQAAAABbNHZ26c_izOww8obp6-uTKC4o0Nv-
07XRXthjo3LpvJ0TdTxldxI=&member_code=testname"
}
}

Error-Response

HTTP /1.1 200 OK

{

"code" : 错误码,

"msg" : 错误信息,

"data" : {}

}

使用说明：包网要获取游戏详情页面的 url 第一步要请求该接口获取详情页面的路径和授权 token，该授权 token 有效时间是 1 天，过期后请求 url 将无法获取详情页面。包网可以使用 iframe 技术把详情页面内嵌到自己的系统。

4.9 获取捕猎游戏进出房间金额记录

此 API 接口主要用于采集捕猎游戏进出房间金额数据,该数据是某客户需求 ,如果不需要该数据的接入方可以忽略该接口,使用方法跟 4.1 接口一样.

URL:POST v3/agent/log_by_page/gt/fish_logout
/[page_key/:page_key|id/:id]

Success-Response-fish_logout

HTTP/1.1 200 OK

{

"code":0,

"msg":"success",

"data":{

"data": [{

"id": string "全局 id",

"player_name": “玩家用户名",

"start_time":int "进入房间时间 10 位时间戳",

"end_time": int "退出房间时间 10 位时间戳",

“game_id”: int “游戏 id”,

“start_chips”: double 进入房间金额单位元

“end_chips”: double 退出房间金额单位元

},...],

"page_key": "分页符"
}}

Error-Response
HTTP /1.1 200 OK
{
"code" : 错误码,
"msg" : 错误信息,
"data" : {}
}

4.10 获取玩家汇总数据

此 API 接口主要用于采集玩家汇总数据,该数据是某客户需求 ,如果不需要该数据的接入方可以忽略该接口,使用方法跟 4.1 接口一样.
URL:POST
v3/agent/log_by_page/gt/player_stat/[page_key/:page_key|id/:id]
Success-Response-player_stat
HTTP/1.1 200 OK
{
"code":0,
"msg":"success",
"data":{
"data": [{
"id": string "全局 id",
"player_name": "玩家用户名",
"date ":int "美东时间天,格式:20180929",
"all_bets": double "有效打码，单位元",
"all_wins": double "总奖金，单位元"
"total_bets": double "总下注，单位元",
"jackpot_bonus": double jackpot 奖金，单位元,
"jp_contri": double jackpot 贡献，单位元
"activity_payout": double 活动派彩，单位元
"result": double 玩家总收支(不包括 activity_payout)
"gt": string 游戏类型 slot/fish/poker/fruit
"count_rows": int 注单笔数(不包括 activity_payout 的笔数,只包括下注注单笔数),
"currency":货币符号

```
        },...],
    "page_key": "分页符"
  }}
```

Error-Response

```
  HTTP /1.1 200 OK
  {
     "code" : 错误码,
      "msg" : 错误信息,
      "data" : {}
  }
```

使用说明：玩家汇总时间为每天美东时间 0 点 40 分统计昨天的数据，每个玩家每个类型的游戏有一条统计数据，只统计昨天产生注单的玩家。

4.11 获取 jp 奖池

此 API 接口主要用于采集游戏的 jp 奖池

URL:POST v3/jp/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agen.
merchantcode	string	Password of the agent

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    "https://API_FG_HOST/v3/jp/ "
```

Success-Response

```
  HTTP/1.1 200 OK
  {
  "code":0,
  "msg":"success",
  "data": {
    "jp"：double 总的彩池，
    "list"：[{
       "game_id"：int 游戏 id,
```

"jp"：double 彩池值单位元},
…
]
}}

Error-Response
HTTP /1.1 200 OK
{
"code" : 错误码,
"msg" : 错误信息,
"data" : {}
}

5. Player 相关 API

5.1 创建用户

此 API 用于创建用户
URL:POST v3/players
Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	UserName of player.size rang:5,32
password	string	Password of player.size range:5,40

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
                -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
                -d "member_code=camYang"      \
                -d "password=123456"      \
                "https://API_FG_HOST/v3/players"
```

Success-Response

HTTP/1.1 200 OK

```
{
   "code":0,
   "msg":"success",
   "data":{
   "openid":"27e0B7q_2BrOI2X8aJrlTgD5jbNkAaoGh445mmIaDXgCA"}
}
```

Error-Response

HTTP /1.1 200 OK

```
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

5.2 删除玩家会话

此 API 用于删除玩家会话

方法一(使用 openid):

URL:POST v3/player_sessions/:openid

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
openid	string	Unique Id of player. Max length: 60

方法二(使用 member_code):

URL:POST v3/player_sessions/member_code/:member_code/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	5.1 创建玩家的 member_code

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    "https://API_FG_HOST/v3/player_sessions/27e0B7q_2BrOI2X8aJrlTgD5jbNkAaoGh445mmIaDXgCA"
OR
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    "https://API_FG_HOST/v3/player_sessions/member_code/yourname"
```

Success-Response

```
HTTP/1.1 200 OK
 {
    "code":0,
    "msg":"success",
    "data":{}
 }
```

Error-Response

```
HTTP /1.1 200 OK
{
   "code" : 错误码,
    "msg" : 错误信息,
    "data" : {}
}
```

5.3 存取玩家筹码

此 API 用于存取玩家筹码

方法一(使用 openid):

URL:POST v3/player_uchips/:openid

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.

merchantcode	string	Password of the agent

Parameter

字段	类型	说明
openid	string	Unique Id of player.
amount	Number	Transfer negative or positive money using cents to descript
externaltransactionid	String	Transaction Order ID.Unique ID.rang:1-20

方法二(使用 member_code):

URL:POST v3/player_uchips/member_code/:member_code

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	5.1 创建玩家的 member_code
amount	Number	Transfer negative or positive money using cents to descript
externaltransactionid	String	Transaction Order ID.Unique ID.rang:1-20

Example usage

```
curl -XPOST -H "Accept: application/json"
    -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    -d "amount=-100"
    -d "externaltransactionid=123456"
    "https://API_FG_HOST/v3/player_uchips/27e0B7q_2BrOI2X8aJrlTgD5jbNkAaoGh445mmIaDXgCA"
```

OR

```
curl -XPOST -H "Accept: application/json"
    -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    -d "amount=-100"
    -d "externaltransactionid=123456"
"https://API_FG_HOST/v3/player_uchips/member_code/yourname"
```

Success-Response

```
HTTP/1.1 200 OK
 {
    "code":0,
    "msg":"success",
    "data":{}
 }
```

Error-Response

```
HTTP /1.1 200 OK
{
"code" : 错误码,
"msg" : 错误信息,
"data" : {}
}
```

注意：

1、 externaltransactionid 值不能重复,最大长度不要超过 20 位

2、 如果注单 id 重复会返回 120(单号已经存在)错误码。

3、 正常收到返回，如果 code=0 表示注单处理成功，如果 code 返回 206 或者 208 则都需要调用 5.6 接口检验注单状态，别的 code 都表示注单处理失败接入方可以进行相关的回滚操作。如果没有收到返回或者 http 状态码返回 502 都需要调用 5.6 接口检验注单状态(5.6 验证玩家存储筹码状态接口)

5.4 查询玩家筹码

此 API 用于查询玩家筹码

方法一(使用 openid):

URL:POST v3/player_chips/:openid

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
openid	string	Unique Id of player. Max length:60

方法二(使用 member_code):

URL:POST v3/player_chips/member_code/:member_code/

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	5.1 创建玩家的 member_code

Example usage

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    "https://API_FG_HOST/v3/player_chips/27e0B7q_2BrOI2X8aJrlTgD5jbNkAaoGh445mmIaDXgCA"
```

OR

```
curl -XPOST -H "Accept: application/json" -H "merchantname: dl_one" \
    -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
    "https://API_FG_HOST/v3/player_chips/member_code/yourname"
```

Success-Response

```
HTTP/1.1 200 OK
{
"code":0,
"msg":"success",
"data":{
    "balance":1000,
    "currency":"CNY"
}}
```

Error-Response

```
HTTP /1.1 200 OK
{
  "code" : 错误码,
  "msg" : 错误信息,
  "data" : {}
}
```

5.5 检测玩家是否已经存在

此 API 用于检测玩家是否已经存在

URL:POST v3/player_names/:member_code

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	UserName of player

Example usage

```
curl -XPOST -H "Accept: application/json"
  -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
  "https://API_FG_HOST/v3/player_names/CamYang"
```

Success-Response

```
HTTP/1.1 200 OK
{
  "code":0,
  "msg":"success",
  "data":{
     "openid":"27e0B7q_2BrOI2X8aJrlTgD5jbNkAaoGh445mmIaDXgCA"
}}
```

Error-Response

```
HTTP /1.1 200 OK
{
  "code" : 错误码,
  "msg" : 错误信息,
  "data" : {}
}
```

5.6 验证存取玩家筹码状态

此 API 用于验证存取玩家筹码状态

URL:POST v3/player_uchips_check/:externaltransactionid

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
externaltransactionid	string	5.3 存取玩家筹码唯一注单号.rang:1-20

Example usage

```
curl -XPOST -H "Accept: application/json"
  -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
  "https://API_FG_HOST/v3/player_uchips_check/1759377344"
```

Success-Response

```
HTTP/1.1 200 OK
{
    "code":0,
    "msg":"success",
    "data":{
      “externaltransactionid”: string  唯一单号
      “amount”: int  筹码
      “time”: int 10 位 unix timestamp  注单产生的时间
      “end_chips”: int  该注单结束后玩家最终筹码
      “member_code”: string  玩家用户名}
}
```

Error-Response

```
HTTP /1.1 200 OK
{
    "code" :  错误码,
    "msg" :  错误信息,
    "data" : {}
}
```

注意：code=0 表示注单成功, code=208 表示注单正在处理,code=209 表示注单处理失败，code=119 表示注单号不存在；建议间隔(4/9/16/25/60/180/1800/1800/1800/3600,单位秒)发起查询，如果 10 次都没办法

确认注单状态（即一直返回 code=208），则把注单标识为异常注单需要手动确认。

5.7 断线重连查询玩家所在游戏是否已经结算

棋牌部分游戏每一局持续时间比较长，玩家中途关闭窗口或者断线重连后需要重新进入该游戏以完成当前局。该接口可以查询玩家未完成的游戏。接入方在玩家启动游戏的时候先调用该接口查询是否有未结束的局，如果有先进入未结束局所在游戏继续游戏，如果没有直接进入新开启的游戏。

方法一(使用 openid):

URL:POST v3/player/unsettled/:openid

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
openid	string	Unique Id of player. Max length:60

方法二(使用 member_code)

URL:POST v3/player/unsettled/member_code/:member_code

Header

字段	类型	说明
Accept	string	application/json
merchantname	string	accountname of the agent.
merchantcode	string	Password of the agent

Parameter

字段	类型	说明
member_code	string	玩家用户名

Example usage

```
curl -XPOST -H "Accept: application/json"
  -H "merchantname: dl_one" \
  -H "merchantcode: 58660ef68954347af0dd1224ec12cb20" \
  "https://API_FG_HOST/v3/player/unsettled/member_code/solo"
```

Success-Response

HTTP/1.1 200 OK

```
{
    "code":0,
    "msg":"success",
    "data":{
        “flag”: 1  有未结束局,表示玩家有未完成的局
        “game_id”: int  游戏 id，未完成局游戏 id
         “gamecode”: string  游戏代码，未完成局游戏代码}
}
```

or

```
{
    "code":0,
    "msg":"success",
    "data":{
   “flag”:0  没有未结束的局，返回该值表示玩家可以进入新的游戏}
}
```

Error-Response

```
HTTP /1.1 200 OK
{
  "code" :  错误码,
  "msg" :  错误信息,
  "data" : {}
}
```

6. 第三方 app 加载 FG 游戏注意事项

使用 webview 的 app 大厅，需要在 userAgent 里面插入 Browser_Type /Android_APP 字段。游戏中的返回大厅操作是调用的 window.close，所以在 app 中要响应 WebView 中 JS 的 window.close()操作进行关闭游戏界面。如果不设置 userAgent 参数进入游戏会显示“向上滑动全屏游戏”而无法操作。

6.1 android Webview 设置代码实例：

```
1. WebSettings ws = getSettings();
2. setLayoutParams(new ViewGroup.LayoutParams(ViewGroup.LayoutParams.
   MATCH_PARENT, ViewGroup.LayoutParams.MATCH_PARENT));
3. //设置浏览器标识
4. String ua = ws.getUserAgentString();
5. ws.setUserAgentString(ua + "Browser_Type/Android_APP");
6.
7. //解决跨域问题
8. ws.setAllowFileAccessFromFileURLs(true);
9. ws.setAllowUniversalAccessFromFileURLs(true);
10. //设置可以访问文件
11. ws.setAllowFileAccess(true);
12. //响应 window.close
13. setWebChromeClient(new BaseWebChromeClient());
14.
15. protected class BaseWebChromeClient extends WebChromeClient {
16.     @Override
17.     public void onCloseWindow(WebView window) {
18.         super.onCloseWindow(window);
19.         //响应游戏中的 window.close 进行关闭游戏界面的操作
20.         mActivity.finish();
21.     }
22. }
```

6.2 iOS Webview 设置代码实例：

```
1. __weak typeof(self) weakSelf = self;
2.     [self.myWKWebView evaluateJavaScript:@"navigator.userAgent" complet
   ionHandler:^(id result, NSError *error) {
3.         __strong typeof(weakSelf) strongSelf = weakSelf;
4.         NSString *userAgent = result;
5.         NSString *newUserAgent = [userAgent stringByAppendingString:@"Bro
   wser_Type/Android_APP"];
6.         NSDictionary *dictionary = [NSDictionary dictionaryWithObjectsAndKe
   ys:newUserAgent, @"UserAgent", nil];
7.         [[NSUserDefaults standardUserDefaults] registerDefaults:dictionary];
8.         [[NSUserDefaults standardUserDefaults] synchronize];
9.
```

```
10.      [self.myWKWebView setCustomUserAgent:newUserAgent];
11.
12.      // needs retain because evaluateJavaScript: is asynchronous
13.      //strongSelf.myWKWebView = [[WKWebView alloc] initWithFrame:stro
    ngSelf.view.bounds]; }];
14.
15. //解决跨域问题
16. [_myWKWebView.configuration.preference                setValue:@"TRUE"
    forKey:@"allowFileAccessFromFileURLs"];
```

6.3 app 加载 FG ZIP 游戏包流程说明

步骤：

1. 根据游戏列表返回的游戏 zip 资源地址进行游戏资源包下载
2. 解压 zip 包，调用启动游戏接口获取 meta 启动信息
3. 修改 index.html 文件在<head>和<meta 之间插入步骤 2 获取的 meta 启动信息，保存 index.html
4. 加载 index.html?language=<语言代码>启动游戏

修改后的 index.html 文件

Figure 6.1 修改 index.html 实例图

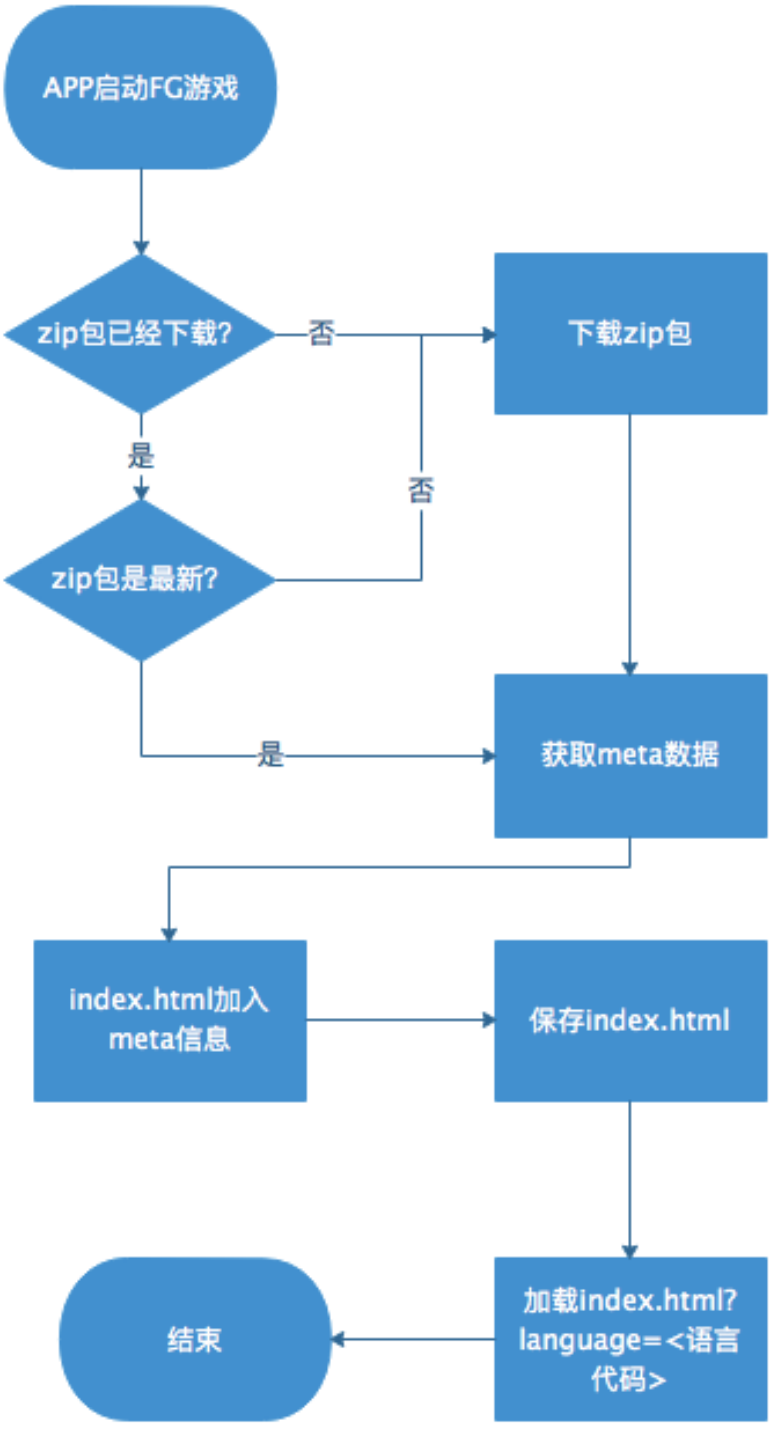

Figure6.2 app 加载 FG 游戏 ZIP 包流程图

7. 创建玩家范例

PHP:

```
<?php
$apiHost = 'https:/API_FG_HOST/v3';
$header = array();
$body = array();
array_push($header, 'merchantname:test');
array_push($header, 'merchantcode:598c8f39eb6106562c9263f6347614c3
');
// 创建用户 POST
$body['member_code'] = '20171127b';
$body['password'] = '20171127b';
$apiUrl = $apiHost.'/players';
$res = httpCurl($apiUrl, $body, $header, 'POST');
print_r($res);
// 游戏列表 POST
$body = array();
$apiUrl = $apiHost.'/games/game_type/h5/language/zh_CN';
$res = httpCurl($apiUrl, $body, $header, 'POST');
print_r($res);
function httpCurl($url, $body = null, $header = null, $type = "GET", $timeou
t = '20', $returnHeader = 0) {
    //创建一个 curl 资源
    $ch = curl_init();
    //设置 URL 和相应的选项
    curl_setopt($ch, CURLOPT_URL, $url); //设置 url
    //设置为 false,只会获得响应的正文(true 的话会连响应头一并获取
到)
    curl_setopt($ch, CURLOPT_HEADER, $returnHeader);
    curl_setopt($ch, CURLOPT_TIMEOUT, $timeout); // 设置超时限制防止
死循环
    //设置发起连接前的等待时间，如果设置为 0，则无限等待。
    curl_setopt($ch, CURLOPT_CONNECTTIMEOUT, $timeout);
    //将 curl_exec()获取的信息以文件流的形式返回，而不是直接输出。
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    //设置提交方式
    switch ($type) {
        case "GET":
```

```
33.             curl_setopt($ch, CURLOPT_HTTPGET, true);
34.             break;
35.         case "POST":
36.             curl_setopt($ch, CURLOPT_POST, true);
37.             break;
38.         case "PUT"://使用一个自定义的请求信息来代替"GET"或"HEAD"作为 HTTP 请 求。这对于执行"DELETE" 或者其他更隐蔽的 HTT
39.             curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "PUT");
40.             break;
41.         case "DELETE":
42.             curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "DELETE");
43.             break;
44.     }
45.     //设备请求体
46.     if (count($body) > 0) {
47.         if (is_array($body)) {
48.             $formdata = http_build_query($body);
49.         } else {
50.             $formdata = $body;
51.         }
52.         curl_setopt($ch, CURLOPT_POSTFIELDS, $formdata); //全部数据使用 HTTP 协议中的"POST"操作来发送。
53.     }
54.     //设置请求头
55.     if (count($header) > 0) {
56.         curl_setopt($ch, CURLOPT_HTTPHEADER, $header);
57.     }
58.     //上传文件相关设置
59.     curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);
60.     curl_setopt($ch, CURLOPT_MAXREDIRS, 3);
61.     curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false); // 对认证证书来源的检查
62.     curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0); // 从证书中检查 SSL 加密算
63.     //在 HTTP 请求中包含一个"User-Agent: "头的字符串。
64.     curl_setopt($ch, CURLOPT_USERAGENT, 'SSTS Browser/1.0');
65.     curl_setopt($ch, CURLOPT_ENCODING, 'gzip');
66.     curl_setopt($ch, CURLOPT_USERAGENT, 'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0.1)'); // 模拟用户使用的浏览器
67.     //抓取 URL 并把它传递给浏览器
```

```
    $res      = curl_exec($ch);
    $httpCode   = curl_getinfo($ch,CURLINFO_HTTP_CODE);
    if ($res === false) {
        $output = curl_getinfo($ch);
        return 'Curl error: ' . curl_error($ch) . var_export($output, true);
    }
    $result = json_decode($res, true);
    if(is_array($result)) {
        $result['http_code']    = $httpCode;
    } else {
        $result             = array('http_code' => $httpCode);
    }
    //关闭 curl 资源，并且释放系统资源
    curl_close($ch);
    if (emptyempty($result)) {
        return $res;
    } else {
        return $result;
    }
}
?>
```

C#:

```
 public static void Main(string[] args)
        {
            string url = "http://API_FG_HOST/v3/players";
            string postData ="member_code=yourname4&password=yourpassw
ord4";
            sendPost(url, postData);
        }
        public static void sendPost(String url, String postData)
        {
            try{
                HttpWebRequest post = (HttpWebRequest)WebRequest.Create(url
);
                post.Method = "POST";
                post.ContentType = "application/json";
                byte[] bytes = Encoding.UTF8.GetBytes(postData);
                post.ContentLength = bytes.Length;
```

```
            // header
            post.Accept = "application/json";
            post.Headers.Add("merchantname", "zslobby2");
            post.Headers.Add("merchantcode", "598c8f39eb6106562c9263f6347614c3");
            // Send the data
            using (Stream newStream = post.GetRequestStream())
            {
                newStream.Write(bytes, 0, bytes.Length);
                newStream.Close();
            }


            // Get response
            using (HttpWebResponse response = (HttpWebResponse)post.GetResponse())
            {
                StreamReader reader = new StreamReader(response.GetResponseStream(), Encoding.UTF8);
                string content = reader.ReadToEnd();
                Console.WriteLine(content);
            }
        }
        catch (Exception ex)
        {
            Console.Write("something wrong");
        }
    }
```

Java:

```
public static void main(String[] args) {
    String url = "https://API_FG_HOST/v3/players";
    String param = "member_code=yourname3&password=yourpassword3";
    sendPost(url, param);
  }
  public static void sendPost(String url, String param){
```

```
7.      // create default httpclient object
8.      HttpClient httpClient = new DefaultHttpClient();
9.      // create httppost
10.     HttpPost post = new HttpPost(url);
11.     // header
12.     post.setHeader("Accept", "application/json");
13.     post.setHeader("merchantname", "zslobby2");
14.     post.setHeader("merchantcode", "598c8f39eb6106562c9263f6347614c
    3");
15.     // body
16.     try {
17.        StringEntity entity = new StringEntity(param);
18.        entity.setContentEncoding("UTF-8");
19.        entity.setContentType("application/json");
20.        post.setEntity(entity);
21.        HttpResponse response = httpClient.execute(post);
22.        int statusCode = response.getStatusLine().getStatusCode();
23.        String str = EntityUtils.toString(response.getEntity());
24.        System.out.printf("success result:%d, string :%s", statusCode, str);
25.     } catch (IOException e) {
26.        //close
27.        e.printStackTrace();
28.     }
```

8. 错误码

CODE	MSG	ZH_MSG
100	Bad request	错误请求，MERCHANTNAME&merchantcode 为空
101	Agent to insufficient	代理商余额不足
102	Invalid merchant name or code,or accout have been locked.	Merchantname 账号或密码不对，或者账号被锁
103	Unauthorized cross merchant	未授权，不能查询其他运营商信息

104	Player does not exist	该玩家不存在
105	Player already exists	玩家已经存在
106	The account is frozen	账户被冻结
107	Invalid member code, it must be between 5-32 characters and no special characters.	玩家代码必须是 5-32 个字符，并没有特殊的字符之间
108	Invalid password, it must be 5-32 alphanumeric	密码应是 5-32 个字母数字’;
109	Illegal chips parameters	筹码值非法
110	Illegal of parameters	参数非法
111	Player's balance is insufficient to withdraw	玩家余额不足
112	Invalid game code	无效的游戏代码
113	Invalid access this game	该游戏不在代理商选中游戏列中 禁止访问
114	Invalid password	无效的密码值
115	IP address is Exception	IP 被阻止
116	reqeust is blocked	代理商请求过多 被阻止
117	can't withdraw when player is clearing	玩家正在结算无法提现
118	can't switch game when player is playing	正在赌注中無法切換遊戲
119	not exist	单号不存在或者该注单失败
120	externaltransactionid has existed	单号已经存在
121	time range error	时间范围有误
122	no permission to launch fg lobb	没有权限启动大厅
123	id not belong to this member_code"	Id 不属于该玩家
201	internal error	api 内部错误
202	System is currently unable to process your request. Please try again	重试
203	chips update fail	筹码更新失败
204	get data fail	采集数据失败
205	player abnormal online status	玩家登录状态异常

206	Request timeout	**超时**
207	api server is maintaining	**api 维护中**
208	Uchips is processing	**充值转账正在处理中**
209	Uchips was failure	**充值提现失败**

9. 语言代码

语言代码	说明
zh-cn	简体中文
zh-hk	繁体中文
en-us	英文