MTA 高级游戏模型开发指南

更新日期：2014-09-24

目 录

更新日期：2014-09-24

1 概述

1.1 目的

MTA 自定义事件设计，指导 MTA 自定义事件离线统计后台开发和 SDK 自定义事件开发者指南编写。

1.2 范围

MTA 自定义事件设计。

1.3 假设/依赖

无

1.4 参考资料

《腾讯移动分析 Android 统计 SDK 开发者接入指南》
《腾讯移动分析 iOS 统计 SDK 开发者接入指南》

1.5 缩略语

更新日期：2014-09-24

2 自定义事件概述

2.1 配置流程

登录 MTA 移动统计平台(http://mta.qq.com)，选择左侧导航栏 自定义分析→自定义事件，点击“新增事件”按钮进入自定义事件配置页面。

自定义事件 ID 由不少于 4 个字符组成，可以是大小写字母、下划线和数字组成，不可以含其它字符（可以全部是数字）。事件 ID 区分大小写，上报 onPlay 和 OnPlay 被视为不同的事件。

建议自定义事件 ID 使用能表明事件含义的字符串，如 OnPlay、OnPause、OnStop、OnLogin、OnLogout 等。可以为每个事件配置一个用来展示报表的事件名称（别名）。

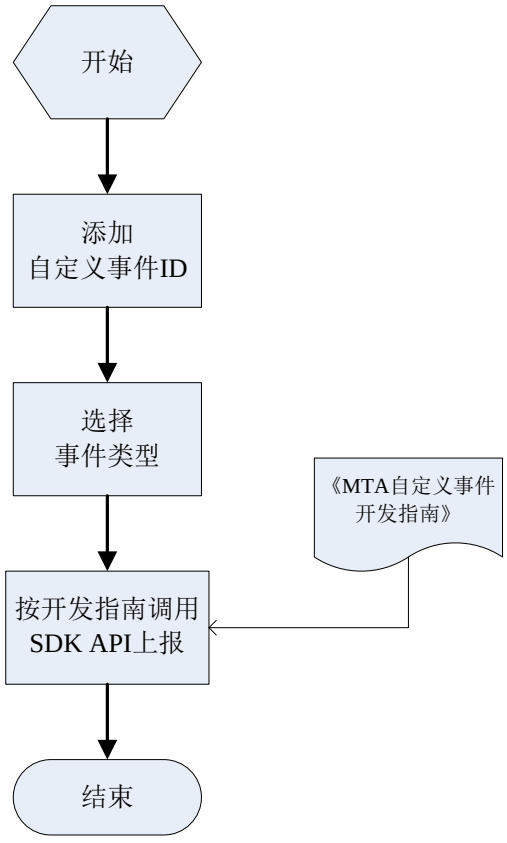

图 2.1 自定义事件配置流程

自定义事件类型选择比较严格，一旦选择某种类型事件，则在调用 SDK API 上报数据时必须按照该类型说明严格填写参数，否则事件统计不成功。在各应用分类说明里有自定义事件各类型和参数及使用说明。

自定义事件的相关数据从事件 ID 注册的当天开始统计。**如果一个事件没有在 MTA 网站上注册，就不会有任何统计数据。**选择合适的事件类型可以使统计更有针对性，MTA 生成的数据报表、图也更适合应用本身，统计结果更有参考价值。

注：同一种事件类型可能出现在不同分类应用中，例如日常工具类的 APP 也可以使用游戏

的道具购买事件。

2.2 事件分类

自定义事件可以实现 APP 大多数特性需求，具有很大灵活性，但不同类型 APP 对自定义事件的统计需求差异很大，简单地统计各事件触发次数很难满足精细化运营的需要。因此，针对不同分类 APP 进行自定义事件分类，再对不同类型自定义事件做针对性统计可以更精准地满足 APP 统计分析需求。

MTA 自定义事件针对统计场景分为普通事件、计算事件、预定义事件三大类，每一大类都有其适用场景。

普通事件：统计 事件发生次数，事件参与用户数。

计算事件：统计 事件发生次数，事件参与用户数，某参数值累加。

预定义事件：针对事件类型做个性化统计。

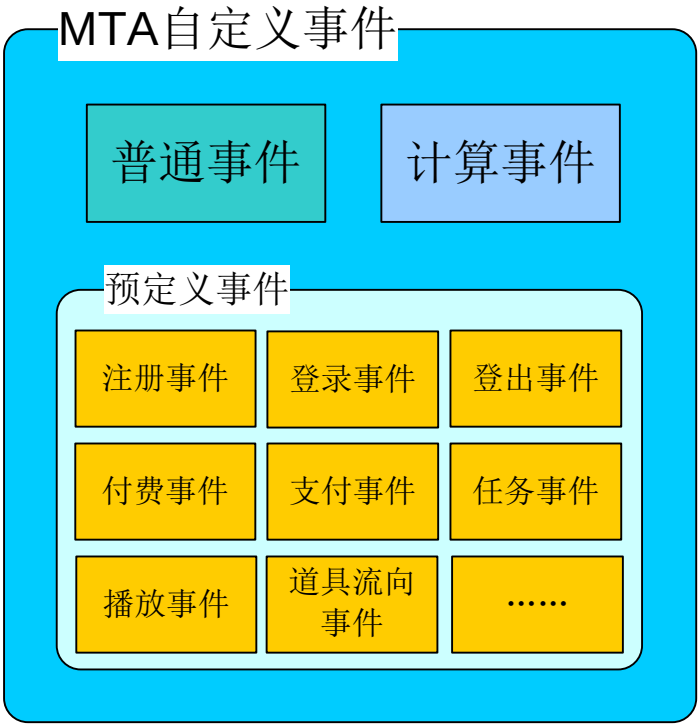

图 2.1　MTA 自定义事件分类

普通事件为适用范围最广应用最多的事件，并且可配置事件转化漏斗模型，事件发生次数、参与人数统计应配置成普通事件。

计算事件适用于对事件参数值进行累加、求平均等运算需求，可以根据具体的业务场景定制用户需要的报表。

普通事件容易上手，可以快速配置，但一些时候不能很好的满足用户需求，而计算事件的使用门槛会高一些，预定义事件刚好结合了两者的优势。预定义事件针对于特定的场景，把一些常用的统计项单独列出来，而不需要用户配置事件参数 ID、参数名称、运算规则等繁琐的项目。

更新日期：2014-09-24

3 游戏应用事件

游戏应用事件包括一些游戏 APP 特有的用户事件，例如道具消耗，角色升级。MTA 针对游戏 APP 推出了专门的高级分析模型，帮助开发者关注用户的忠诚度，提升付费用户转化率，助力日常运营。

注：除必需的参数外，若需要其他参数用于自定义统计或其他用途，亦可上报。

3.1 注册事件

注册事件是用户在 APP 内注册账号触发的事件，用于注册用户数的统计。注册行为通常发生在需要联网的 APP 上，用户向 APP 服务端给注册一个 uid，用户在 APP 内的操作均会以这个 uid 来记录和跟踪。

事件 ID：**onGameRegister**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选

注册事件上报示例：

Android 版本

```
// When user register.
public void onRegister() {                    // 你的函数
    Properties prop = new Properties();
    prop.setProperty("uid", "Robin");              // 用户ID
    prop.setProperty("wid", "广东电信1");        // 大区ID或大区名，若不分区，填0
    prop.setProperty("role", "死亡骑士");      // 角色名（可选择上报）
    prop.setProperty("job", "战士");             // 角色职业（可选择上报）
    StatService.trackCustomKVEvent(this, "onGameRegister", prop);
}
```

iOS 版本

```
// When user register.
-(IBAction) clickNormalButton:(id)sender{
    NSDictionary* kvs=@{@"uid": @"Robin",          // 用户 ID
                        @"wid": @"广东电信 1",     // 大区 ID 或大区名
                        @"role": @"死亡骑士",  // 角色名（可选择上报）
                        @"job": @"战士"          // 角色职业（可选择上报）
                        };
    [MTA trackCustomKeyValueEvent:@"onGameRegister" props:kvs];
```

更新日期：2014-09-24

```
}
```

说明：

1. iOS 版本使用新的 NSDictionary 定义语法，需使用 Xcode 4.4 及以上版本。

低版本的 Xcode 请使用旧语法：

```
NSDictionary *dict = [NSDictionary dictionaryWithObjects:@[@"value1", @"value2"]
                                                 forKeys:@[@"key1", @"key2"]];
```

2. Android 和 iOS 上报的接口的参数名和用法一致。

为免重复，下面的预定义事件只给出 Android 版的示例，iOS 请参照本示例。

3.2 登录事件

登录事件是注册用户在 APP（输入密码）登录时触发的事件，用于统计每个用户的活跃情况和登录次数。

如果 APP 记住了用户的登录态，APP 启动时也请上报一个登录事件。

事件 ID：**onGameLogin**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
xp	当前经验值（经验值为与玩家等级相关的数值）	0
lvxp	下一次升级所需经验值（增量）	0

登录事件上报示例：

```
// When user login.
public void onLogin() {                    // 你的函数
    Properties prop = new Properties();
    prop.setProperty("uid", "Robin");              // 用户ID
    prop.setProperty("wid", "广东电信1");          // 大区ID或大区名，若不分区，填0
    prop. setProperty("ulv", "5");                 // 角色等级
    prop.setProperty("xp", "520");                 // 角色当前经验值
    prop.setProperty("lvxp", "1000");              // 下一次升级经验值
    prop.setProperty("role", "死亡骑士");          // 角色名（可选择上报）
    prop.setProperty("job", "战士");               // 角色职业（可选择上报）
    StatService.trackCustomKVEvent(this, "onGameLogin", prop);
}
```

3.3 登出事件

登出事件是注册用户从 APP 将登录态改成非登录状态的事件，主要用于统计游戏时长和玩家状态。

对游戏这类强登录态应用，登出事件除上报 uid 和用户等级外，还需上报本次登录在线时长（时长单位为秒）。

事件 ID：**OnGameLogout**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
udu	本次登录在线时长	0

登出事件上报示例：

```
// When user logout.
public void onLogout() {                        // 你的函数
    Properties prop = new Properties();
    prop.setProperty("uid", "Robin");              // 用户ID
    prop.setProperty("wid", "广东电信1");        // 大区ID或大区名，若不分区，填0
    prop. setProperty("ulv", "5");                 // 角色等级
    prop. setProperty("udu", "120");               // 本次登录在线时长
    prop.setProperty("role", "死亡骑士");       // 角色名（可选择上报）
    StatService.trackCustomKVEvent(this, "onGameLogout", prop);
}
```

3.4 角色升级事件

玩家角色升级的时候可以上报升级事件。需上报 uid、升级后等级、升级时长（上一次升级到本次升级之间用户在线时长，用于从时间上统计升级难度）。

事件 ID：**onGameUserUpgrade**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
udu	上一次升级到本次升级之间用户在线时长	0
xp	当前经验值（经验值为与玩家等级相关的数值）	0
lvxp	下一次升级所需经验值（增量）	0

更新日期：2014-09-24

角色升级事件上报示例：

```
// When user UserUpgrade.
public void onUserUpgrade () {                                  // 你的函数
        Properties prop = new Properties();
        prop.setProperty("uid", "Robin");                       // 用户ID
        prop.setProperty("wid", "广东电信1");                    // 大区ID或大区名，若不分区，填0
        prop. setProperty("ulv", "5");                          // 角色等级
        prop. setProperty("udu", "9650");   //上一次升级到本次升级之间用户在线时长
        prop.setProperty("xp", "500");                          // 角色当前经验值
        prop.setProperty("lvxp", "1000");                       // 下一次升级经验值
        StatService.trackCustomKVEvent(this, "onGameUserUpgrade", prop);
}
```

3.5 充值事件

充值事件一般发生在用户有实际支付行为的时候，需上报 uid、用户等级和付费金额。充值事件作为一个统计 APP 收入情况的关键事件，用于跟踪每个用户的付费情况。

事件 ID：**onGameRecharge**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
scene	场景	-
amount	充值金额	必选，且>0(可以是小数)
getgold	本次充值获得的金子（游戏内通过充值获得的货币）数量	0
holdgold	本次充值后持有的金子数量	0

充值事件上报示例：

```
// When user pay successfully.
public void onPaySuccess() {                                    // 你的函数
        Properties prop = new Properties();
        prop.setProperty("uid", "Robin");                       // 用户ID
        prop.setProperty("wid", "广东电信1");                    // 大区ID或大区名
        prop. setProperty("ulv", "5");                          // 角色等级
        prop.setProperty("role", "死亡骑士");                    // 角色名（可选择上报）
        prop. setProperty("scene", "财付通");                    // 付费场景
        prop. setProperty("amount", "50");
        prop. setProperty("getgold", "10");
        prop. setProperty("holdgold", "60");
        StatService.trackCustomKVEvent(this, "onGameRecharge", prop);
}
```

更新日期：2014-09-24

3.6 商城购买事件

商城购买事件用于玩家对道具购买情况的统计，可以查看每个道具类型或者细化到每个道具本身的销售情况。

此外，道具类型和道具 ID 可以在 MTA 前台配置别名，展示更加通俗易懂的名称。

事件 ID：**onGameShop**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
itemType	道具类型	-
itemId	道具名称	必选
num	购买数量	1
amount	总共消耗游戏币（换算成人民币单位：元）	0

道具购买事件上报示例：

```
// When user onShop.
public void onShop() {                                  // 你的函数
    Properties prop = new Properties();
    prop.setProperty("uid", "Robin");                   // 用户ID
    prop.setProperty("wid", "广东电信1");               // 大区ID或大区名，若不分区，填0
    prop. setProperty("ulv", "5");                      // 角色等级
    prop. setProperty("itemType", "消耗品");            // 道具类型ID或道具类型名称
    prop. setProperty("itemId", "10012");               // 道具ID或道具名
    prop. setProperty("num", "5");                  // 购买道具数量
    prop. setProperty("amount", "5"); //消耗金币50，每个金币换算为人民币1角（0.1元）
    StatService.trackCustomKVEvent(this, "onGameShop", prop);
}
```

3.7 商城外购买事件

其他付费事件用于**商城外**的购买统计，需上报 uid、用户等级、付费场景和付费额度。可以跟踪用户消耗游戏币的情况，包括人均消耗的游戏币和每次消耗的金额范围。

事件 ID：**onGameShopEx**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0

更新日期：2014-09-24

scene	付费场景（如复活、NPC 道具）	必选
num	购买数量	1
amount	总共消耗游戏币（换算成人民币单位：元）	0

道具购买事件上报示例：

```
// When user buy items outside the GAME STORE.
public void onShop() {                                  // 你的函数
    Properties prop = new Properties();
    prop.setProperty("uid", "Robin");                   // 用户ID
    prop.setProperty("wid", "广东电信1");               // 大区ID或大区名，若不分区，填0
    prop. setProperty("ulv", "5");                      // 角色等级
    prop.setProperty("role", "死亡骑士");               // 角色名（可选择上报）
    prop.setProperty("job", "战士");                    // 角色职业（可选择上报）
    prop. setProperty("scene", "原地复活");             // 付费场景
    prop. setProperty("num", "5");                      // 购买道具数量
    prop. setProperty("amount", "5"); //消耗金币50，每个金币换算为人民币1角（0.1元）
    StatService.trackCustomKVEvent(this, "onGameShopEx", prop);
}
```

3.8 道具消耗事件

道具消耗事件用于道具消耗的统计，道具消耗事件需上报 uid、用户等级、道具类型、道具数量。可以分场景查看道具的消耗情况。

道具类型包括装备后绑定的武器、装备、坐骑，炼化需要消耗的宝石、材料，以及用于复活、多倍经验等特殊消耗品。

事件 ID：**onGameConsume**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
scene	道具类型	必选
num	消耗道具的数量	1

道具消耗事件上报示例：

```
// When user consume.
public void onConsume() {                               // 你的函数
    Properties prop = new Properties();
    prop.setProperty("uid", "Robin");                   // 用户ID
    prop.setProperty("wid", "广东电信1");               // 大区ID或大区名，若不分区，填0
    prop. setProperty("ulv", "5");                      // 角色等级
```

更新日期：2014-09-24

```
        prop.setProperty("role", "死亡骑士");       // 角色名（可选择上报）
        prop.setProperty("job", "战士");            // 角色职业（可选择上报）
        prop. setProperty("scene", "复活石");       //场景
        prop. setProperty("num", "1");              // 消耗道具的数量
        StatService.trackCustomKVEvent(this, "onGameConsume", prop);
}
```

3.9 关卡（副本）事件

关卡（副本）事件用于统计游戏内关卡（副本）[1] 完成情况。和登录登出事件一样，需要上报两次，进入关卡上报一次（status=enter），退出关卡上报一次（status=success 或者 status=fail）

如果一次关卡通关跨天完成，算在完成时间的当天。

事件 ID：**onGameRaid**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
udu	花费时长（status=success 时上报）	0
type	关卡类型（如特殊关、奖励关）	-
raidid	关卡/副本 ID	-
status	状态(0=进入关卡，1=挑战成功，2=挑战失败)	0(enter)/1(success)/2(fail)三选一，否则此事件无效

关卡事件上报示例：

```
// When user EnterRaid.
public void onRaidEnter() {                                    // 你的函数
        Properties prop = new Properties();
        prop.setProperty("uid", "Robin");
        prop.setProperty("wid", "广东电信1");        // 大区ID或大区名，若不分区填0
        prop. setProperty("ulv", "5");                          // 角色等级
        prop.setProperty("role", "死亡骑士");       // 角色名（可选择上报）
        prop.setProperty("job", "战士");            // 角色职业（可选择上报）
        prop. setProperty("raidtype", "隐藏关卡");
        prop. setProperty("raidid", "死亡之地");     // 关卡ID或关卡名
        prop. setProperty("status", "0");     // 关卡状态
        StatService.trackCustomKVEvent(this, "onGameRaid", prop);
}
```

[1] 关卡和副本在数据分析中类似。为简洁起见，以下关卡（副本）统一描述为“关卡”。

```
// When user CleanRaid.
public void onRaidClean() {                                    // 你的函数
      Properties prop = new Properties();
      prop.setProperty("uid", "Robin");
      prop.setProperty("wid", "广东电信1");        // 大区ID或大区名，若不分区填0
      prop. setProperty("ulv", "5");                          // 角色等级
      prop. setProperty("udu", "9650");   //通过关卡所耗时间（单位秒）
      prop.setProperty("role", "死亡骑士");      // 角色名（可选择上报）
      prop.setProperty("job", "战士");             // 角色职业（可选择上报）
      prop. setProperty("raidtype", "隐藏关卡");
      prop. setProperty("raidid", "死亡之地");     // 关卡ID或关卡名
      prop. setProperty("status", "1");     // 关卡状态
      StatService.trackCustomKVEvent(this, "onGameRaid", prop);
}
```

3.10 任务事件

任务事件用于统计任务的分发和完成情况（接受任务、完成任务、取消任务），任务事件需上报 uid、任务名称。

和登录登出事件一样，需要上报两次，接受任务上报一次（status=enter），退出任务上报一次（status=success 或者 status=cancel）

事件 ID：**onGameTask**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
udu	花费时长（status=success 时上报）	0
type	任务类型（如主线任务、支线任务）	-
taskid	任务 ID	-
status	状态(0=接受任务，1=任务完成，2=任务失败)	0(accept)/1(success)/2(fail) 三选一，否则此事件无效

任务事件上报示例：

```
// When user TaskStart.
public void onTaskStart() {                                    // 你的函数
      Properties prop = new Properties();
      prop.setProperty("uid", "Robin");
      prop.setProperty("wid", "广东电信1");   // 大区ID或大区名，若不分区填0
      prop. setProperty("ulv", "5");                          // 角色等级
```

更新日期：2014-09-24

```
        prop. setProperty("type", "支线任务1");                    // 任务类型
        prop. setProperty("taskid", "哥布林皮毛");          // 任务ID或任务名
        prop. setProperty("udu", "200");   //任务所耗时间（单位秒）
        prop. setProperty("status", "2");     // 任务状态（进入/完成/失败）
        StatService.trackCustomKVEvent(this, "onGameTask", prop);
}
```

3.11 对战事件

对战事件用于统计玩家与玩家之间的对战情况。

事件 ID：**onGamePK**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	必选
ulv	玩家等级	0
udu	对战时长	0
place	对战场景	-
status	状态(0=进入对战模式，1=获胜，2=失败)	0(enter)/1(success)/2(fail)三选一，否则此事件无效

对战事件上报示例：

```
// When user PkSuccess
public void onPkSuccess () {                                    // 你的函数
        Properties prop = new Properties();
        prop.setProperty("uid", "Robin");
        prop.setProperty("wid", "广东电信1");   // 大区ID或大区名，若不分区填0
        prop. setProperty("ulv", "5");                          // 角色等级
        prop. setProperty("udu", "200");   //所耗时间（单位秒）
        prop. setProperty("place ", "竞技场");   //对战场景

        prop. setProperty("status", "1");     // 结束状态
        StatService.trackCustomKVEvent(this, "onGamePK ", prop);
}
```

3.12 好友事件

好友事件记录玩家添加、删除、变更操作之后好友情况。

事件 ID：**onGameFriends**		
参数 ID	含义	默认值

uid	用户 ID	必选
wid	大区 ID	-
ulv	玩家等级	0
opt	操作类型（1=添加，2=删除）	1
fnum	变更以后的好友数量	必选

好友事件上报示例：

```
// When user Add friend or Delete friend.
public void YourFuntion () {                                    // 你的函数
        Properties prop = new Properties();
        prop.setProperty("uid", "Robin");
        prop.setProperty("wid", "广东电信1");   // 大区ID或大区名，若不分区填0
        prop. setProperty("ulv", "5");                          // 角色等级
        prop. setProperty("opt ", "1");
        prop. setProperty("fnum", "2");   //好友数量
        StatService.trackCustomKVEvent(this, "onGameFriends ", prop);
}
```

3.13 活动事件

活动事件用于统计玩家参与游戏活动情况。

事件 ID：**onGameActivity**		
参数 ID	含义	默认值
uid	用户 ID	必选
wid	大区 ID	-
ulv	玩家等级	0
actid	活动 ID 或活动名	必选
acttype	活动类型 1 登录奖励（每日登录，连续登录奖励等） 2 成就奖励（连胜，杀怪，装备，合成等） 3 成长奖励（角色升级奖励） 4 充值奖励（首充奖励，累计充值奖励等） 5 补偿奖励（服务器出问题的补偿） 6 礼包奖励（开服礼包，渠道特别礼包，公会礼包，节日礼包等） 7 邀请奖励（成功邀请好友奖励） 8 任务奖励（关卡，挑战，竞技，PVP，盟战等，完成任务后获得较丰富的奖励） 9 抽奖活动	0

更新日期：2014-09-24

	0 其他活动类型	
status	活动状态(accept/ success /cancel)	accept
du	活动时长	0

活动事件上报示例：

```
public void YourFuntion(){
        Properties prop = new Properties();
        prop.setProperty("uid", "Robin");
        prop.setProperty("wid", "广东电信1");  // 大区ID或大区名，若不分区填0
        prop. setProperty("ulv", "5");                    // 角色等级
        prop. setProperty("actid", "2");                  // 活动ID
        prop. setProperty("acttype", "2");                // 活动类型
        prop.setProperty("status", "success");                // 活动状态
        prop. setProperty("du", "2");                     // 活动时长
        StatService.trackCustomKVEvent(this, "onGameActivity ", prop);
}
```

更新日期：2014-09-24