

推啊广告 **iOS SDK**—接入说明文档

目 录

目 录	2
版本记录	3
修订记录	4
1 SDK 集成	5
2 调用接口	6
3 接口调用是否成功	10

版本记录

版本号	版本支持	版本内容	升级建议
2.2.0	iOS 8.0 及以上	此次版本进行以下内容更新： 修复浮标广告位不透明底色的问题。 新增无导航媒体由模态推出活动的方式 新增曝光成功、曝光失败和有效点击的回调	建议升级

修订记录

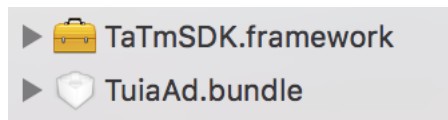
修改日期	修改人	修改内容简述	版本号
2017 年 04 月 20 日	董振宁		2.1.0
2017 年 5 月 25 日	黄文奇	修改 2.9 节 <code>ms_item_id/material_id/adslot_id</code> 字段由 <code>int</code> 改为 <code>long</code> 类型； <code>activity_id</code> 由原来的 <code>int</code> 改为 <code>string</code> 类型	2.1.0
2017 年 6 月 27 日	董振宁	<ol style="list-style-type: none">1. 修复浮标广告位不透明底色的问题2. 新增无导航媒体由模态推出活动的方式3. 新增曝光成功、曝光失败和有效点击的回调	2.2.0

1 SDK 集成

在客户端集成 **TaTmSDK** 之前,需要完成以下准备工作:

1.1. 下载客户端 SDK

1.2. 将 **TuiaAd.bundle**、**TaTmSDK.framework** 添加到工程中



1.3. 添加资源库




点击主工程名进入工程配置界面，在 **General** 中得 **Embedded Binaries** 和 **Linked Frameworks and Libraries** 选项中点击+号添加如下类库

▼ Embedded Binaries

 TaTmSDK.framework ...in TuiaADDemo

+ -

▼ Linked Frameworks and Libraries

Name	Status
 librsolv.tbd	Required ⬆⬇⬆
 libz.1.1.3.tbd	Required ⬆⬇⬆
 TaTmSDK.framework	Required ⬆⬇⬆

+ -

1.4. 修改项目工程的 **Plist** 文件

Key	Type	Value
▼ Information Property List	Dictionary	(14 items)
Localization native development re...	String	en
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Application requires iPhone enviro...	Boolean	YES
App Transport Security Settings	Dictionary	(1 item)
Allow Arbitrary Loads	Boolean	YES
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
► Required device capabilities	Array	(1 item)
► Supported interface orientations	Array	(3 items)

至此，SDK 集成完毕。

2 调用接口

2.1. 注册 SDK:

需要在 AppDelegate 中使用从推啊平台获取的 appkey 来注册 SDK(下面的 appkey 为示例)。

```
[TaTmHelper setupWithAppKey:@"4JtqTFPyzESKzNiznbfV6r3qHASN"];
```

2.2. 广告控件说明（参数 adslotId 为广告位 id，在推啊平台获取）

a) 横幅 TaStreamerView

```
TaStreamerView *streamer = [TaStreamerView adViewWithOrigin:CGPointMake(0, 0)
    adslotId:@"458" parentViewController:self successBlock:^(
    NSLog(@"success");
} failedBlock:^(
    NSLog(@"failed");
} closedBlock:^(
    NSLog(@"closed");
}]];
[mainView addSubview:streamer];
```

b) 插屏 TaScreenView

```
TaScreenView *screen = [TaScreenView adViewWithAdslotId:@"459"
    parentViewController:self successBlock:^(
    NSLog(@"success");
} failedBlock:^(
    NSLog(@"failed");
} closedBlock:^(
    NSLog(@"closed");
}]];
[screen show];
```

c) 信息流 TaInfoStreamerView

```

TaInfoStreamerView *infoStreamer = [TaInfoStreamerView adViewWithOrigin:CGPointMake
(0, -64) adslotId:@"460" type:TaInfoStreamTypeTall parentViewController:self
successBlock:^(
    NSLog(@"success");
} failedBlock:^(
    NSLog(@"failed");
} closedBlock:^(
    NSLog(@"closed");
}]];
[mainView addSubview:infoStreamer];

infoStreamer = [TaInfoStreamerView adViewWithOrigin:CGPointMake(0, 300)
adslotId:@"461" type:TaInfoStreamTypeShort parentViewController:self
successBlock:^(
    NSLog(@"success");
} failedBlock:^(
    NSLog(@"failed");
} closedBlock:^(
    NSLog(@"closed");
}]];
[mainView addSubview:infoStreamer];

```

d) BANNER TaBnView

```

TaBnView *banner = [TaBnView adViewWithOrigin:CGPointMake(0, 0) adslotId:@"462"
parentViewController:self successBlock:^(
    NSLog(@"success");
} failedBlock:^(
    NSLog(@"failed");
} closedBlock:^(
    NSLog(@"closed");
}]];
[mainView addSubview:banner];

```

e) 浮标 TaDriftView

```

TaDriftView *list = [TaDriftView adViewWithOrigin:CGPointMake(100, 100)
adslotId:@"463" parentViewController:self successBlock:^(
    NSLog(@"success");
} failedBlock:^(
    NSLog(@"failed");
} closedBlock:^(
    NSLog(@"closed");
}]];
UITableView* tableView = [[UITableView alloc] initWithFrame:self.view.bounds];
[mainView addSubview:tableView];
[mainView addSubview:list];

```

f) 应用墙 TaAppWallView (circle 参数为是否原形图标)

```

TaAppWallView *wall = [TaAppWallView adViewWithOrigin:CGPointMake(100, 100)
adslotId:@"464" circle:YES parentViewController:self successBlock:^(
    NSLog(@"success");
} failedBlock:^(
    NSLog(@"failed");
} closedBlock:^(
    NSLog(@"closed");
}]];
UITableView* tableView = [[UITableView alloc] initWithFrame:self.view.bounds];
[mainView addSubview:tableView];
[tableView addSubview:wall];

```

g) 开屏 TaLaunchScreenView (在 AppDelegate 中添加)

```
[TaLaunchScreenView showAdViewWithAdslotId:@"466" successBlock:^(
} failedBlock:^(
    NSLog(@"没有开屏广告");
} finishedBlock:^(
    NSLog(@"展示完毕了");
} userCloseBlock:^(
    NSLog(@"用户点击关闭了");
}];
```

b) 非标准广告嵌入代码说明

```
[TaTmHelper getActivityDataWithAdslotId:@"465" parentViewController:nil
successBlock:^(id resultObj) {
    UIAlertView *alert =[[UIAlertView alloc] initWithTitle:nil message:[NSString
        stringWithFormat:@"%@",resultObj] delegate:self cancelButtonTitle:@"OK"
        otherButtonTitles:nil, nil];
    [alert show];
} failedBlock:^(id resultObj) {
}];
```

2.3. 对于非标准广告的处理(自定义广告):

非标准广告即自定义广告，前面 7 种广告皆为标准广告，所有界面展示和渲染由推啊 SDK 自身负责，开发者只需相应调用即可。但是非标准广告为推啊 SDK 提供接口，返回 JSON 字符串数据，由开发者自身负责数据的渲染与广告的展示，以及广告曝光/点击行为的主动上报。

自定义广告的曝光上报接口(**resultObj** 为自定义广请求到的参数):

```
NSString *adID = [NSString stringWithFormat:@"%@", resultObj[@"adslot_id"]];
NSString *rqID = [NSString stringWithFormat:@"%@", resultObj[@"request_id"]];
NSString *atID = [NSString stringWithFormat:@"%@", resultObj[@"activity_id"]];
[TaTmHelper displayAd:adID activityId:atID requestId:rqID];
```

自定义广告的点击上报接口(**resultObj** 为自定义广请求到的参数):

```
NSString *adID = [NSString stringWithFormat:@"%@", resultObj[@"adslot_id"]];
NSString *rqID = [NSString stringWithFormat:@"%@", resultObj[@"request_id"]];
NSString *atID = [NSString stringWithFormat:@"%@", resultObj[@"activity_id"]];
[TaTmHelper clickAd:adID activityId:atID requestId:rqID];
```

非标准广告的请求接口返回参数请见下文:

参数	参数类型	参数定义	参数说明
request_id	string	广告请求 id	对应的请求 id, 该参数在请求时 sdk 自动生成

error_code	string	错误码	
adslot_id	long	广告位 id	
ad_title	string	推广标题	
activity_title	string	广告活动标题	
activity_id	string	广告活动 id	
ad_icon	string	“广告”小图标	
ad_icon_visible	boolean	是否展示 ad_icon	
ad_close	string	“关闭”小图标	
ad_close_visible	boolean	是否展示 ad_close	
description	string	广告描述	
img_url	string	广告活动素材地址	
img_width	int	广告活动素材宽度	
img_height	int	广告活动素材高度	
ad_content	string	广告活动内容	仅当 ad_type=2 时有内容
ad_type	int	广告位类型	0:插屏 1:横 幅 2:信息流 3:banner 4:浮 标 5:应用墙 6:开屏 7- 非 标广告
expire	long	有效期	默认为 0
wdata_token	string		无需关注
server_time	string		无需关注
click_url	string	素材 url	
material_id	long	素材 id	
dcm	string		无需关注
material_list	List<MaterialRsp>	返回的素材列表	用语多个素材 需求
activity_way	int		无需关注
material_way	int		无需关注
data1	string	扩展字段 1	
data2	string	扩展字段 2	
rid	string	唯一请求 id	
source	int	活动来源	
app_id	long	应用 id	
device_id	String	设备 id	

MaterialRsp

ms_item_id	long		无需关注
item_type	int		无需关注
image_url	string	素材 url	
image_width	int		无需关注
image_height	int		无需关注
text	string	素材文案描述	

注：以上**红色加粗**字体需要特别注意

3 接口调用是否成功

开发者在使用 **xcode** 进行调试时，控制台会打印所有请求参数，以及是否成功。成功打印 **success**、失败打印 **failed**、广告关闭打印 **closed**

例如：

1.请求、曝光返回参数：

```
"activity_id" = "<null>";
"activity_title" = "<null>";
"activity_way" = "<null>";
"ad_close" = "<null>";
"ad_close_visible" = 0;
"ad_content" = "<null>";
"ad_icon" = "<null>";
"ad_icon_visible" = 0;
"ad_title" = "<null>";
"ad_type" = "<null>";
"adslot_id" = 72;
"click_url" = "<null>";
data1 = "<null>";
data2 = "<null>";
dcm = "<null>";
description = "<null>";
"error_code" = 10001;
expire = 0;
"img_height" = 0;
"img_url" = "<null>";
"img_width" = 0;
"material_id" = "<null>";
"material_list" = "<null>";
"material_way" = "<null>";
"request_id" =
28MLSNMkp88LYjjRbqSkzy2jxULg4014877417984263704;
"server_time" = "<null>";
"wdata_token" = "<null>";
}
2017-02-22 13:36:38.577 AdIOSSDK[5505:2090856] failed
```

请求参数

是否成功

2.曝光成功、失败和有效点击的回调

```
[TaTmHelper displaySuccess:^(id resultObj) {  
    //  
    NSLog(@"曝光成功");  
} displayFailed:^(id resultObj) {  
    //  
    NSLog(@"曝光失败");  
} clickTm:^(id resultObj) {  
    //  
    NSLog(@"点击成功");  
}  
}];
```

注意：关于广告位 id

以上示例代码中，**loadAd** 方法里面填的广告位 **id** 仅为示例，开发者需要在推啊媒体平台重新申请。