

# Firebase接入指南

**注：**该文档仅适用于Android Studio工程接入firebase

如果是用Unity工程接入firebase，请联系PM获取Unity配置firebase的文档

## 一、 创建Firebase项目，获取google-services.json文件

1. 登录[Firebase控制台](#)
2. 添加项目
3. 添加应用：填写包名、签名sha1 >> 下载google-services.json >> 跳过完成添加

(修改或添加签名sha1需要重新下载google-services.json文件)

## 二、 Firebase引入工程

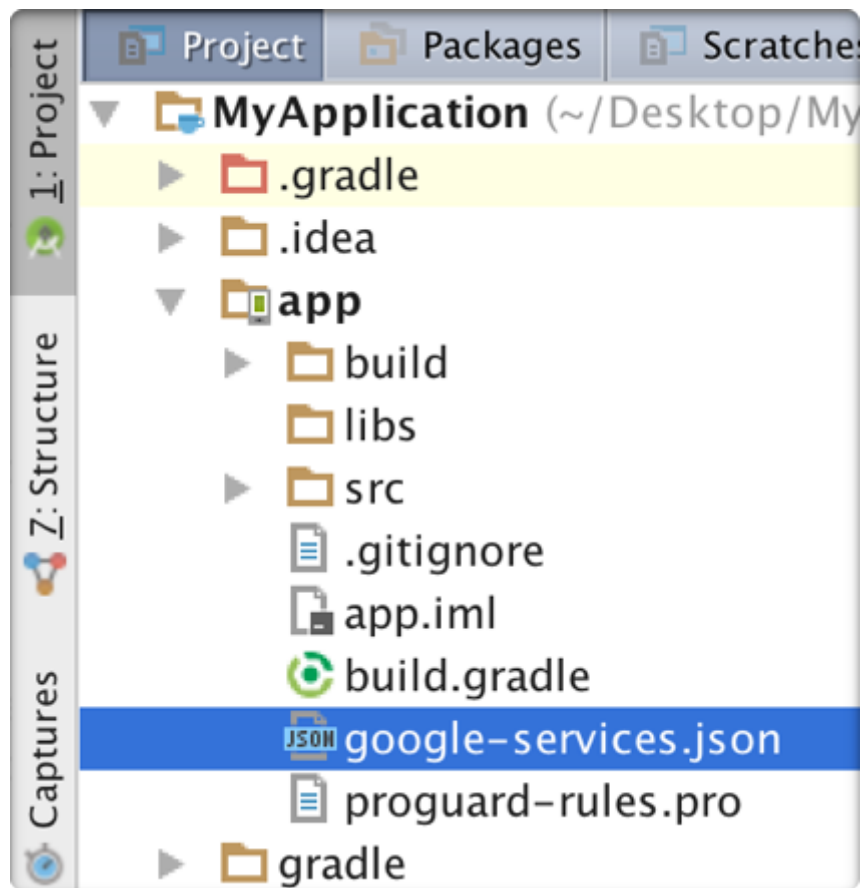
### 1. 修改项目build.gradle

在根目录的build.gradle中加入下面的配置

```
buildscript {  
    dependencies {  
        classpath 'com.google.gms:google-services:4.3.15'  
        classpath 'com.google.firebase:firebase-crashlytics-gradle:2.8.0' //崩溃日志收集  
        (按需配置)  
    }  
}
```

### 2. 项目配置

拷贝google-services.json到工程目录，**json文件必须放到主Module目录，并且包名必须一致**



在主Module的build.gradle中添加配置（主module的build.gradle包含 `apply plugin: 'com.android.application'`）

```
apply plugin: 'com.google.gms.google-services'
apply plugin: 'com.google.firebase.crashlytics' //崩溃日志收集（按需配置）

dependencies {
    //firebase analytics
    implementation 'com.google.firebase:firebase-analytics:22.1.2'
    //推送消息接收(按需接入)
    implementation 'com.google.firebase:firebase-messaging:23.2.1'
    //崩溃日志收集(按需接入)
    implementation 'com.google.firebase:firebase-crashlytics:18.3.7'
}
```

AndroidManifest文件添加权限

```
<uses-permission android:name="android.permission.WAKE_LOCK" />
```

### 三、数据事件接入（按需接入）

以下接口使用该实例调用

```
HWFirebaseManager firebaseManager =
QuickGameManager.getInstance().getFirebaseManager(Context);
```

## SDK内部已经自动记录的事件

注册, 事件名: sign\_up, 参数: uid、name、method

登录, 事件名: login, 参数: uid、name、method

## 1. 自定义事件

### 调用方法

```
public void logCustomEvent(String eventName, Bundle bundle)
```

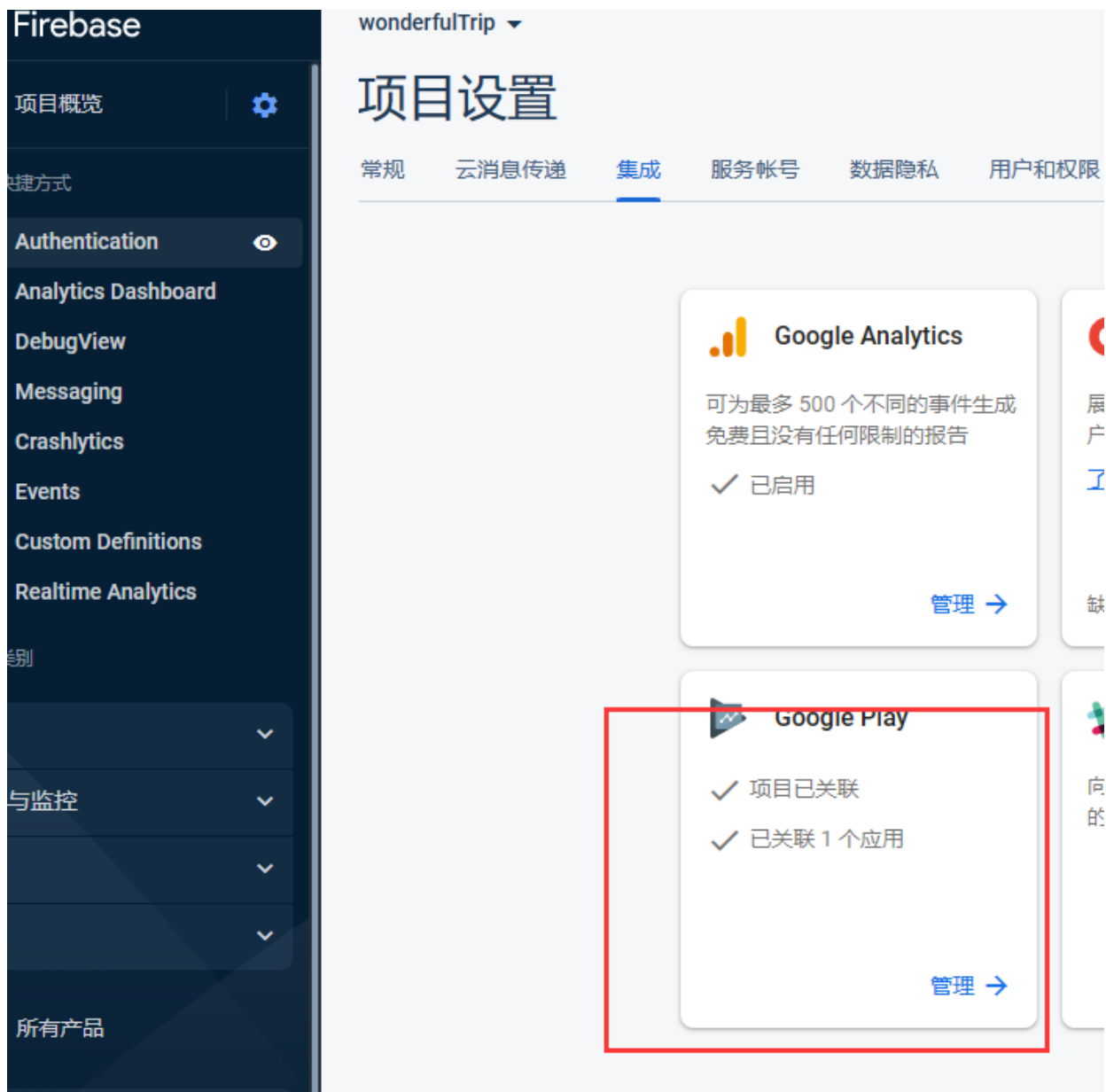
**eventName:** 事件名称

**bundle:** 事件参数

### 示例

```
Bundle bundle = new Bundle();
bundle.putString("id", "123");
bundle.putInt("value", 1);
QuickGameManager.getInstance().getFirebaseManager(this).logCustomEvent("custom", bundle);
```

## 2. 谷歌收入不需要上报, 在Firebase后台关联Google Play即可



#### 四、FireBase推送（按需接入）

## 1. 项目配置

AndroidManifest文件添加如下配置

```
<!-- Firebase接收消息service -->

<service
    android:name="com.quickgame.android.sdk.firebase.HWFirebaseMessagingService"
    android:exported="false">
    <intent-filter>
        <action android:name="com.google.firebase.MESSAGING_EVENT" />
    </intent-filter>
</service>

<!-- 可选配置-->
<!-- 推送消息的图标 -->
<meta-data
    android:name="com.google.firebase.messaging.default_notification_icon"
    android:resource="@drawable/ic_stat_ic_notification" />
<!-- 推送消息的图标颜色 -->
<meta-data
    android:name="com.google.firebase.messaging.default_notification_color"
    android:resource="@color/qg_color_txt_read" />
```

## 2. 设置Firebase消息推送回调

```
//
sdkInstance.setFirmMsgCallback(this, new HWFirebaseCallback() {
    @Override
    public void onGetToken(boolean isSuccess, String token) {
        //调用getFirebaseToken方法时的回调
        //isSuccess:true获取token成功, false获取失败
        //token: 获取成功为注册令牌, 获取失败为空
        Log.d("quickgame", "is success:" + isSuccess + ", token:" + token);
    }

    @Override
    public void onNewToken(String token) {
        //token发生变化时返回
        Log.d("quickgame", "newtoken:" + token);
    }

    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {
        //收到推送消息回调: 当应用在前台显示时, 收到通知时会回调这里, 不会发送任务栏通知
        if (remoteMessage.getNotification() != null) {
            Log.d(TAG, "title:" + remoteMessage.getNotification().getTitle());
            Log.d(TAG, "body:" + remoteMessage.getNotification().getBody());
        }
    }
})
```

```
});
```

### 3. 主动获取Firebase Token

token会在回调的onGetToken中返回

```
sdkInstance.getFirebaseToken(Context context);
```

**RemoteMessage** 获取到信息接口参考[Firebase官方说明](#)

### 测试推送

- 1、手机安装谷歌服务框架
- 2、手机能访问谷歌
- 3、在1、2条件下启动游戏
- 4、获取firebase token，首次启动时logcat会打印firebase的推送token，tag为 `firebase onNewToken`
- 5、使用token，在firebase后台发送测试推送

## 五、Firebase Crashlytics接口（按需接入）

### 1. 设置用户标识符

```
FirebaseCrashlytics.getInstance().setUserId("user123456789");
```

### 2. 添加自定义日志消息

```
FirebaseCrashlytics.getInstance().log("message");
```