# 汇总

文档地址汇总：

1.文档地址

<https://yuque.antfin-inc.com/awm/accs_agoo/accs_android>

2.Max管理地址（申请应用，获取Appkey和密钥）

<http://mappcenter.alibaba.net>

3Agoo Console控制台

<https://agoo-console.alibaba-inc.com/#/list/manage>

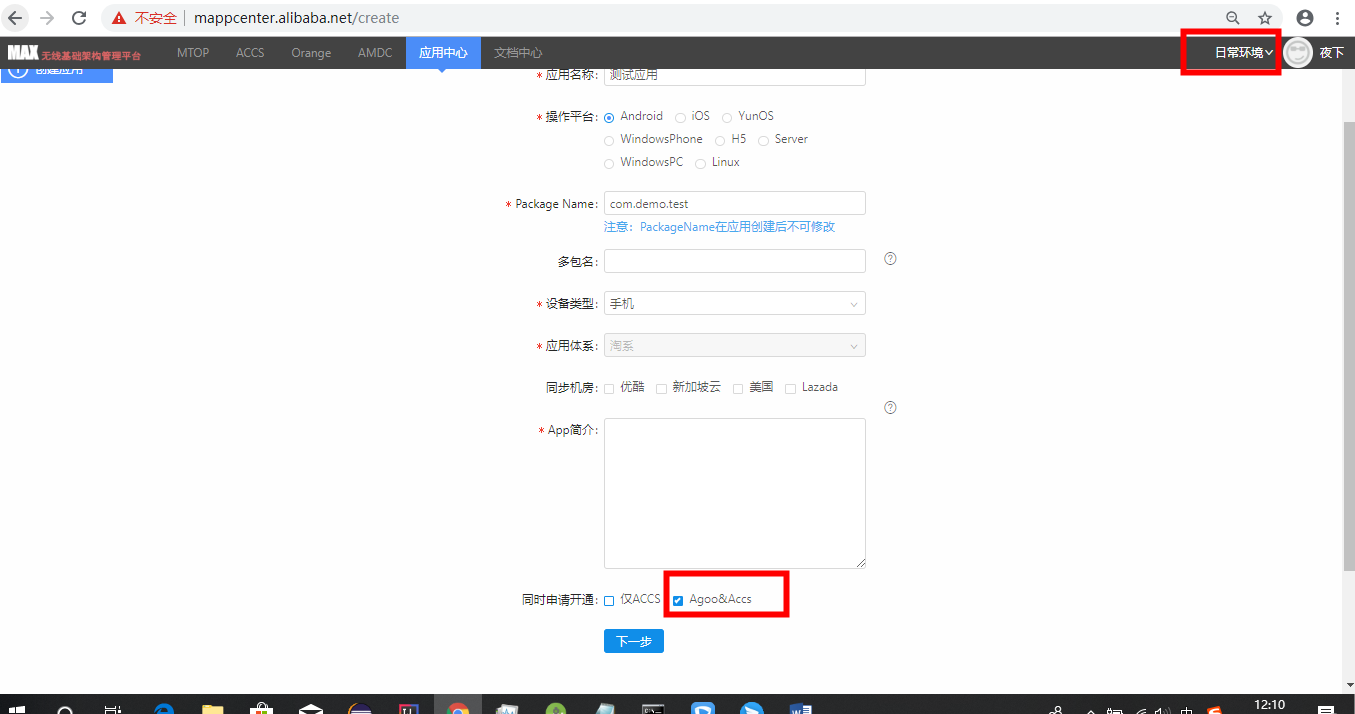
# 二．Agoo

## 2.1注册应用

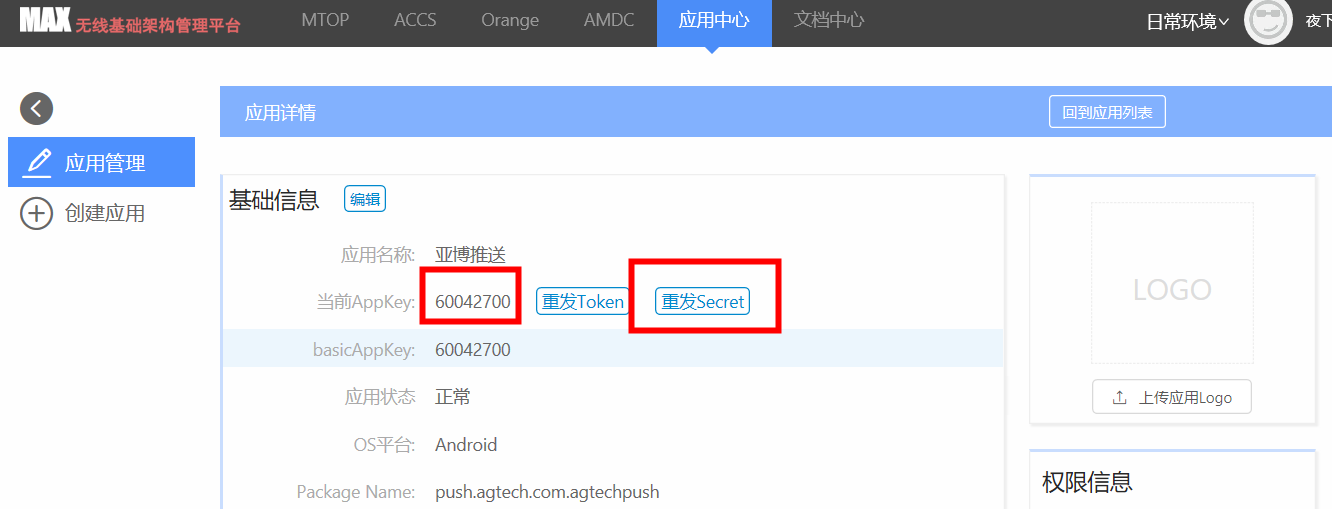
这个应用Agoo和accs都可以用。

### 2.1.1地址

<http://mappcenter.alibaba.net/create>



### 2.1.2 appkey和密钥

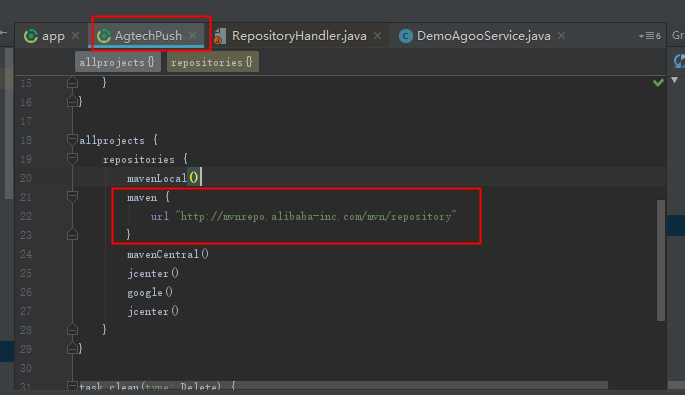


## 2.2 Gradle

### 2.2.1 maven

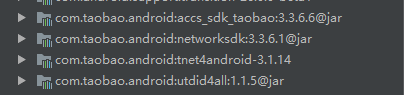
添加内部的maven库（内网环境下可以，外网访问不了这个maven）

maven {  
 url "http://mvnrepo.alibaba-inc.com/mvn/repository"  
}



## 2.2.2添加依赖

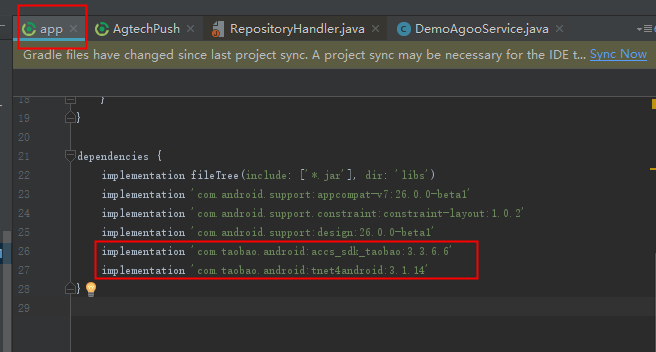
官方文档说要使用这些包



实际添加如下依赖就可以了，accs\_sdk\_taobao，会把其他需要编译的包都依赖进来。

implementation 'com.taobao.android:accs\_sdk\_taobao:3.3.6.6'  
implementation 'com.taobao.android:tnet4android:3.1.14'

如图



## 2.3 Api

### 2.3.1 Manifest配置

#### 2.3.1.1权限声明

这部分跟官方文档相同

<uses-permission android:name="android.permission.INTERNET" />  
<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  
<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />  
<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />  
<uses-permission android:name="android.permission.WAKE\_LOCK" />  
<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />  
<uses-permission android:name="android.permission.GET\_TASKS" />  
<uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED" />

#### 2.3.1.2 Service

必要Service声明:

注意com.taobao.agoo.AgooCommondReceiver这个inflater里的动作，换成自己的包名

<!-- agoo start -->  
<service  
 android:name="org.android.agoo.accs.AgooService"  
 android:exported="true"  
 android:process=":channel">  
 <intent-filter>  
 <action android:name="com.taobao.accs.intent.action.RECEIVE" />  
 </intent-filter>  
</service>  
  
<!-- 接受agoo本地的控制事件 -->  
<receiver  
 android:name="com.taobao.agoo.AgooCommondReceiver"  
 android:exported="true"  
 android:process=":channel">  
 <intent-filter>  
 <action android:name="push.agtech.com.agtechpush.intent.action.COMMAND" />  
 </intent-filter>  
 <intent-filter>  
 <action android:name="android.intent.action.PACKAGE\_REMOVED" />  
 <data android:scheme="package" />  
 </intent-filter>  
</receiver>  
  
<!-- accs的2.2.2以及以后基于intentService，用于消息转发，提高消息送达率 -->  
<service  
 android:name="com.taobao.agoo.TaobaoMessageIntentReceiverService"  
 android:exported="true"  
 android:process=":channel">  
 <intent-filter>  
 <action android:name="org.android.agoo.client.MessageReceiverService" />  
 </intent-filter>  
</service>  
<!-- agoo end -->

### 2.3.2 初始化

先初始化accs环境，才可以使用agoo推送，如果只用agoo推送，可以不绑定accs的服务，下面注释里有。

SECRET是2.2.2里面的密钥。

private void initAccs(){  
 //env参数: Constants.DEBUG = 2; Constants.PREVIEW = 1; Constants.RELEASE = 0;  
 ACCSClient.*setEnvironment*(this, Constants.*DEBUG*);  
 AccsClientConfig.Builder clientBuilder = new AccsClientConfig.Builder()  
 .setAppKey(*AppID*) //必选, 所属appkey  
 .setConfigEnv(Constants.*DEBUG*) //必选, 所属env  
 .setTag(AccsClientConfig.*DEFAULT\_CONFIGTAG*)  
 .setAppSecret(*SECRET*);  
 try {  
 ACCSClient.*init*(this, clientBuilder.build());  
 //如果不用accs的推送，只用agoo的推送，下面两句就可以不用了写了  
 //这两句是绑定accs推送服务的  
 *accsClient* = ACCSClient.*getAccsClient*(AccsClientConfig.*DEFAULT\_CONFIGTAG*);  
 *accsClient*.bindApp(*ttid*,new PushReciver());  
 } catch (AccsException e) {  
 e.printStackTrace();  
 }  
}

### 2.3.3 agoo服务

绑定agoo服务

push.agtech.com.agtechpush.DemoAgooService是自己的写的服务。

它继承TaobaoBaseIntentService。

这个register，是注册设备用的，onSuccess之后的参数，是设备号，可以使用deviceID，来推送消息。

private void initAgoo(){  
 try {  
 TaobaoRegister.*setEnv*(this,Constants.*DEBUG*);  
 TaobaoRegister.*setAgooMsgReceiveService*("push.agtech.com.agtechpush.DemoAgooService");  
 TaobaoRegister.*register*(this, AccsClientConfig.*DEFAULT\_CONFIGTAG*, *AppID*, *SECRET*, "", new IRegister() {  
 @Override  
 public void onSuccess(String deviceID) {  
 Log.*d*("wxj", "onSuccess: deviceID="+deviceID);  
 }  
  
 @Override  
 public void onFailure(String errorCode, String errorMessage) {  
 Log.*d*("wxj", "onFailure: errorCode="+errorCode);  
 Log.*d*("wxj", "onFailure: errorMessage="+errorMessage);  
 }  
 });  
 } catch (AccsException e) {  
 e.printStackTrace();  
 }  
}

### 2.3.4实现服务

服务是收推送消息的，继承TaobaoBaseIntentService

public class DemoAgooService extends TaobaoBaseIntentService {  
  
 @Override  
 protected void onError(Context context, String errorId) {  
   
 }  
  
 @Override  
 protected void onRegistered(Context context, String registrationId) {  
   
 }  
  
 @Override  
 protected void onUnregistered(Context context, String registrationId) {  
   
 }  
  
 @Override  
 protected void onMessage(Context context, Intent intent) {  
 String messageId = intent.getStringExtra(AgooConstants.*MESSAGE\_ID*);  
 String message = intent.getStringExtra(AgooConstants.*MESSAGE\_BODY*);  
 }  
}

### 2.3.5注册服务

<service  
 android:name="push.agtech.com.agtechpush.DemoAgooService"  
 android:exported="true"  
 android:process=":channel">  
 <intent-filter>  
 <action android:name="org.agoo.android.intent.action.RECEIVE" />  
 </intent-filter>  
</service>

## 2.4推送

记得登陆控制台的时候，切换到日常环境。

在MAX上申请的App，在Agoo的控制台上是可见的。

<https://agoo-console.alibaba-inc.com/#/list/manage>



# 三．Accs

## 3.1注册

注册App获取密钥和Appkey。

同agoo。

## 3.2.gralde

同agoo

## 3.3Manifest

### 3.3.1.权限

同agoo

### 3.3.2 Service

<!-- accs start -->  
<service  
 android:name="com.taobao.accs.ChannelService"  
 android:exported="true"  
 android:process=":channel">  
 <intent-filter>  
 <action android:name="com.taobao.accs.intent.action.SERVICE" />  
 </intent-filter>  
</service>  
  
<service  
 android:name="com.taobao.accs.ChannelService$KernelService"  
 android:exported="false"  
 android:process=":channel" />  
  
<service  
 android:name="com.taobao.accs.data.MsgDistributeService"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="com.taobao.accs.intent.action.RECEIVE" />  
 </intent-filter>  
</service>  
  
<receiver  
 android:name="com.taobao.accs.EventReceiver"  
 android:process=":channel">  
 <intent-filter>  
 <action android:name="android.intent.action.BOOT\_COMPLETED" />  
 </intent-filter>  
 <intent-filter>  
 <action android:name="android.net.conn.CONNECTIVITY\_CHANGE" />  
 </intent-filter>  
 <intent-filter>  
 <action android:name="android.intent.action.PACKAGE\_REMOVED" />  
 <data android:scheme="package" />  
 </intent-filter>  
 <intent-filter>  
 <action android:name="android.intent.action.USER\_PRESENT" />  
 </intent-filter>  
</receiver>  
  
<receiver  
 android:name="com.taobao.accs.ServiceReceiver"  
 android:process=":channel">  
 <intent-filter>  
 <action android:name="com.taobao.accs.intent.action.COMMAND" />  
 </intent-filter>  
 <intent-filter>  
 <action android:name="com.taobao.accs.intent.action.START\_FROM\_AGOO" />  
 </intent-filter>  
</receiver>  
  
<service  
 android:name="com.taobao.accs.internal.AccsJobService"  
 android:permission="android.permission.BIND\_JOB\_SERVICE"  
 android:process=":channel" />  
<!-- accs end -->

## 3.4 api

### 3.4.1 初始化

之前在agoo的基础环境里，已经初始化了accs。

ttid是渠道号，可以不填。

PushReciver里面，绑定了接受消息的服务。

private void initAccs(){  
 //env参数: Constants.DEBUG = 2; Constants.PREVIEW = 1; Constants.RELEASE = 0;  
 ACCSClient.*setEnvironment*(this, Constants.*DEBUG*);  
 AccsClientConfig.Builder clientBuilder = new AccsClientConfig.Builder()  
 .setAppKey(*AppID*) //必选, 所属appkey  
 .setConfigEnv(Constants.*DEBUG*) //必选, 所属env  
 .setTag(AccsClientConfig.*DEFAULT\_CONFIGTAG*)  
 .setAppSecret(*SECRET*);  
 try {  
 ACCSClient.*init*(this, clientBuilder.build());  
 //如果不用accs的推送，只用agoo的推送，下面两句就可以不用了写了  
 //这两句是绑定accs推送服务的  
 *accsClient* = ACCSClient.*getAccsClient*(AccsClientConfig.*DEFAULT\_CONFIGTAG*);  
 *accsClient*.bindApp(*ttid*,new PushReciver());  
 } catch (AccsException e) {  
 e.printStackTrace();  
 }  
}

### 3.4.2 服务

#### 3.4.2.1注册服务

注册服务在Max上：

<http://max.test.m.taobao.com/accs/view/mainPage#/services?_k=5jx47k>



seviiceMap里，是id是服务器上注册的service名称，服务端通过这个名称，建立和客户端上，服务的映射关系。

#### 3.4.2.2PushReciver

sericeMap里，是id是服务器上注册的service名称，服务端通过这个名称，建立和客户端上，服务的映射关系。

public class PushReciver implements IAppReceiver {  
  
 public static Map<String,String> *serviceMap*;  
  
 static {  
 //key为serviceId, value为继承TaobaseService实现的接收消息service  
 *serviceMap* = new HashMap<>();  
 *serviceMap*.put(*SERVICE\_NAME*, *LOCAL\_SERVICE*);  
 }  
  
 @Override  
 public String getService(String serviceId) {  
 String service = *serviceMap*.get(serviceId);  
 if(!TextUtils.*isEmpty*(service)){  
 Log.*d*("wxj", "getService: service="+service);  
 return service;  
 }  
 return "";  
 }  
  
 @Override  
 public Map<String, String> getAllServices() {  
 Log.*d*("wxj", "getAllServices: ");  
 return *serviceMap*;  
 }  
  
 @Override  
 public void onData(String s, String s1, byte[] bytes) {  
 Log.*d*("wxj", "onData: ");  
 }  
  
 @Override  
 public void onBindApp(int i) {  
 Log.*d*("wxj", "onBindApp: ");  
 }  
  
 @Override  
 public void onUnbindApp(int i) {  
 Log.*d*("wxj", "onUnbindApp: ");  
 }  
  
 @Override  
 public void onBindUser(String s, int i) {  
 Log.*d*("wxj", "onBindUser: ");  
 }  
  
 @Override  
 public void onUnbindUser(int i) {  
 Log.*d*("wxj", "onUnbindUser: ");  
 }  
  
 @Override  
 public void onSendData(String s, int i) {  
 Log.*d*("wxj", "onSendData: ");  
 }  
}

#### 3.4.2.3实现service

service继承TaobaseService

这个onBind(String s, int i, ExtraInfo extraInfo，并不是service本身的onBind，不要搞错了哈。

onDate里，会收到消息。

public class PushService extends TaoBaseService{  
 @Override  
 public void onData(String serviceId, String userId, String dataId, byte[] data, TaoBaseService.ExtraInfo info) {  
 Log.*d*("wxj", "onResponse:serviceId"+serviceId);  
 Log.*d*("wxj", "onResponse:userId"+userId);  
 Log.*d*("wxj", "onResponse:dataId"+dataId);  
  
 try {  
 String srt2 = new String(data,"UTF-8");  
 Log.*d*("wxj", "onResponse: str="+ srt2);  
 Toast.*makeText*(getApplicationContext(),"serverid="+serviceId+" data="+srt2,Toast.*LENGTH\_SHORT*).show();  
 } catch (UnsupportedEncodingException e) {  
 e.printStackTrace();  
 }  
  
  
 ACCSManager.AccsRequest request = new ACCSManager.AccsRequest("userid",serviceId,data,dataId);  
 *accsClient*.sendData(request);  
  
 ACCSManager.AccsRequest requestAck = new ACCSManager.AccsRequest("userid",serviceId,data,dataId);  
 *accsClient*.sendRequest(requestAck);  
  
;  
 }  
  
 @Override  
 public void onBind(String s, int i, ExtraInfo extraInfo) {  
 Log.*d*("wxj", "PushService onBind: s="+s);  
 Toast.*makeText*(getApplicationContext(),"onBind="+s,Toast.*LENGTH\_SHORT*).show();  
 }  
  
 @Override  
 public void onUnbind(String s, int i, ExtraInfo extraInfo) {  
 Log.*d*("wxj", "onUnbind:");  
 }  
  
 @Override  
 public void onSendData(String s, String s1, int i, ExtraInfo extraInfo) {  
 Log.*d*("wxj", "onSendData:");  
 Toast.*makeText*(getApplicationContext(),"onSendData="+s,Toast.*LENGTH\_SHORT*).show();  
 }  
  
 @Override  
 public void onResponse(String s, String s1, int i, byte[] bytes, ExtraInfo extraInfo) {  
 Toast.*makeText*(getApplicationContext(),s,Toast.*LENGTH\_SHORT*).show();  
 Toast.*makeText*(getApplicationContext(),s1,Toast.*LENGTH\_SHORT*).show();  
 Toast.*makeText*(getApplicationContext(),s,Toast.*LENGTH\_SHORT*).show();  
 Log.*d*("wxj", "onResponse:s"+s);  
 Log.*d*("wxj", "onResponse:s1"+s1);  
 Log.*d*("wxj", "onResponse:i"+i);  
 try {  
 String srt2 = new String(bytes,"UTF-8");  
 Log.*d*("wxj", "onResponse: str="+ srt2);  
 Toast.*makeText*(getApplicationContext(),"data="+srt2,Toast.*LENGTH\_SHORT*).show();  
 } catch (UnsupportedEncodingException e) {  
 e.printStackTrace();  
 }  
 }  
}

## 3.5推送消息

