IOS 消息推送方案

一、场景介绍

关于消息推送功能，之前在安卓平台已经做过一次，我们使用的方案是使用阿帕奇开源的消息代理服务activeMq来进行消息代理转发，后台连接服务器发布消息，前端和移动端连接服务器订阅消息。在安卓平台当应用由于内存不足被系统杀掉或者用户手动杀掉之后是无法收到推送消息的，只有当应用再次启动后重新连接才可以。但是IOS平台不同，由于是封闭系统，苹果有自己的管控机制，其提供了系统级的通知服务，可以实现离线消息推送，具体说明可以参考如下文章：

<http://www.cocoachina.com/cms/wap.php?action=article&id=24235>

而我们的业务场景需要这种离线推送功能，所以需要根据苹果的开发规范来对接其APNs（Apple Push Notification service），鉴于要开发一套稳定高效可靠的消息推送平台需要很多开发成本，我们可以直接使用成熟的三方消息推送平台，国内比较知名的有：极光、百度云、个推、友盟等。从接入的便利性和测试后的稳定性以及同行使用情况来看，极光推送符合我们的需求，并且其免费版本的关键指标满足我们的使用场景。

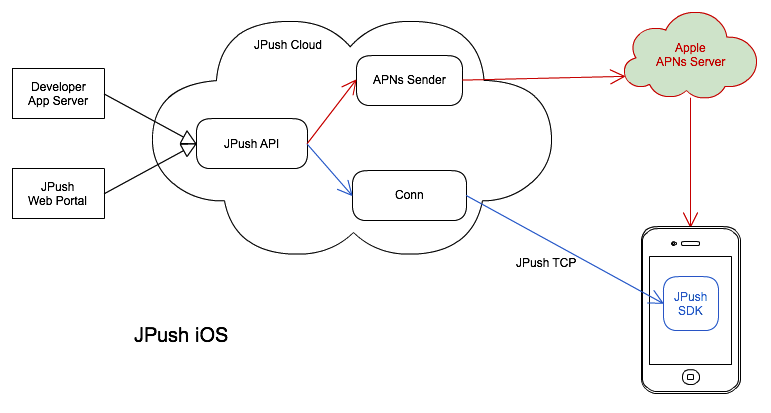
<https://www.jiguang.cn/push-price>

1. 集成极光推送SDK

从官网文档介绍中可以看到，极光推送在给服务端和客户端各自提供了其接入文档。服务端我们选择使用Java，客户端我们目前只集成IOS，如前面介绍所述，我们的Android和Web端目前都是使用的开源ActiveMq来作为消息代理推送实现的，而且目前极光的官网中没有提供Web端集成的SDK，在其开发社区提问后，得到官方明确回复没有Web端SDK，见问题：<https://community.jiguang.cn/t/topic/36074>



所以我们只需要集成实现IOS端的推送。



1、首先来看后台部分集成：

<https://docs.jiguang.cn/jpush/server/sdk/java_sdk/>

有使用maven和本地jar包两种方式，目前调试的demo为集成本地jar包。极光提供了两种方式来作为分类推送，一种是别名Alias， 一种是标签Tag，两者的区别是针对一台设备只能有一个别名，而标签可以具有多个。目前我们的推送告警信息的场景是针对站点来推送的，所以可以使用站点来区分，因为有的账号可能会有过个站点的权限，所以我们选择使用标签Tag来分类，即站点作为Tag来推送消息。

protected static final Logger LOG = LoggerFactory.getLogger(JPushDemo.class);

//根据极光推送平台创建的应用中所给的key来赋值

protected static final String APP\_KEY ="773272e27580a5dfd9b95364";

protected static final String MASTER\_SECRET = "06a66584b67746042ecd73b1";

//离线消息保留5天

public static long TIME\_TO\_LIVE = 60 \* 60 \* 24 \* 5;

public static void main(String[] args) {

// NOTICE: 极光推送免费账号的调用频率限制为每分钟600次，请后台自行控制，否则会报异常。

for(int i=0; i<5; i++) {

System.out.println("-------------- start push -------------------");

pushTest();

try {

Thread.sleep(500);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

public static void pushTest() {

ClientConfig clientConfig = ClientConfig.getInstance();

//默认1天，此处设置为免费账号最长时间

clientConfig.setTimeToLive(TIME\_TO\_LIVE);

//默认为开发环境，发布后要设置为生成环境

clientConfig.setApnsProduction(false);

final JPushClient jpushClient = new JPushClient(MASTER\_SECRET, APP\_KEY, null, clientConfig);

final PushPayload payload = buildPushObjectIOS();

try {

PushResult result = jpushClient.sendPush(payload);

//LOG.info("Got result - " + result);

System.out.println(result);

} catch (APIConnectionException e) {

LOG.error("Connection error. Should retry later. ", e);

LOG.error("Sendno: " + payload.getSendno());

} catch (APIRequestException e) {

LOG.error("Error response from JPush server. Should review and fix it. ", e);

LOG.info("HTTP Status: " + e.getStatus());

LOG.info("Error Code: " + e.getErrorCode());

LOG.info("Error Message: " + e.getErrorMessage());

LOG.info("Msg ID: " + e.getMsgId());

LOG.error("Sendno: " + payload.getSendno());

}

}

//构建推送消息

public static PushPayload buildPushObjectIOS() {

//可在消息中携带自定义字段内容，比如点击跳转页面路径url

Map<String, String> extras = new HashMap<String, String>();

extras.put("jumpUrl", "https://community.jiguang.cn/push");

//可根据场景选择是否需要主副标题

IosAlert alert = IosAlert.newBuilder()

.setTitleAndBody("主标题XXX", "副标题XXX", "消息内容XXX")

.build();

//push for more than one tag

ArrayList<String> tags = new ArrayList<>();

tags.add("stationId001");

tags.add("stationId002");

return PushPayload.newBuilder()

//平台

.setPlatform(Platform.ios())

//.setAudience(Audience.all())

//.setAudience(Audience.alias("65086508"))

//受众

.setAudience(Audience.tag(tags))

.setNotification(Notification.newBuilder()

.setAlert(alert)

.addPlatformNotification(IosNotification.newBuilder()

//badge increase

.incrBadge(1)

//没有加入音频文件，一般还是默认音

.setSound("happy")

.addExtras(extras).build())

.build())

.build();

}

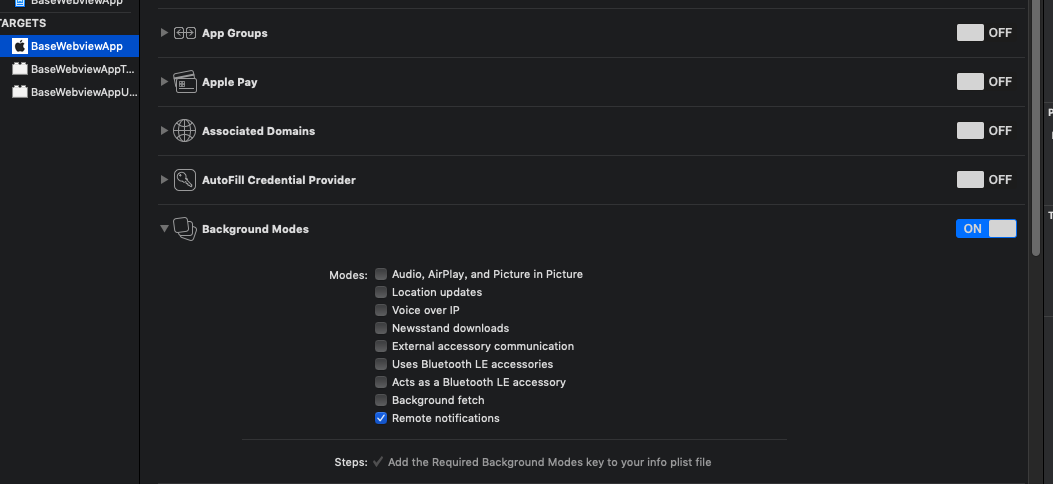
具体代码可参照：

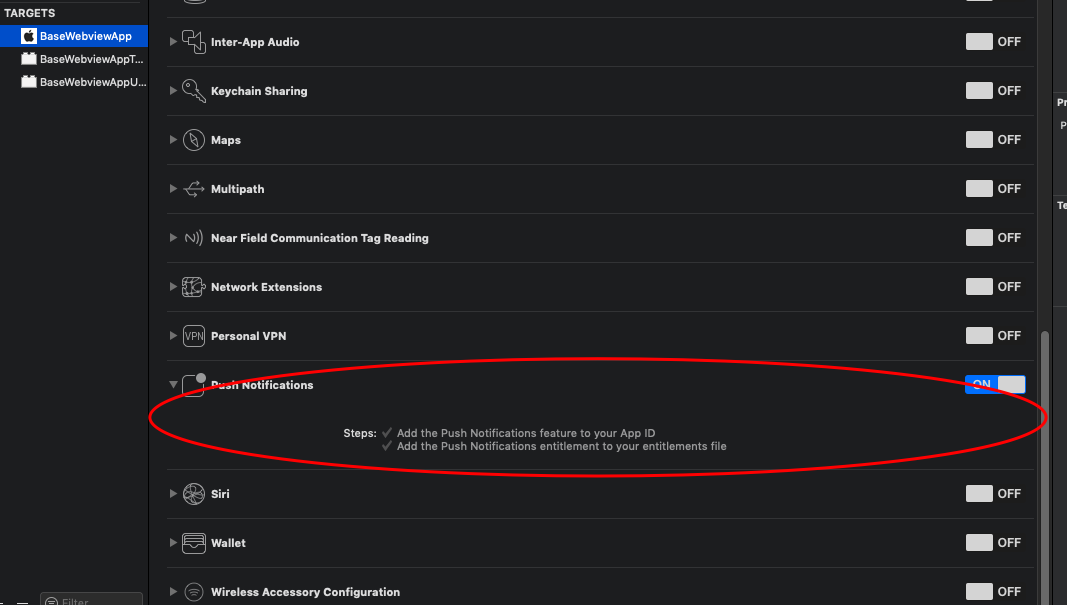
<https://github.com/jpush/jpush-api-java-client>

2、IOS端集成

<https://docs.jiguang.cn/jpush/client/iOS/ios_guide_new/>

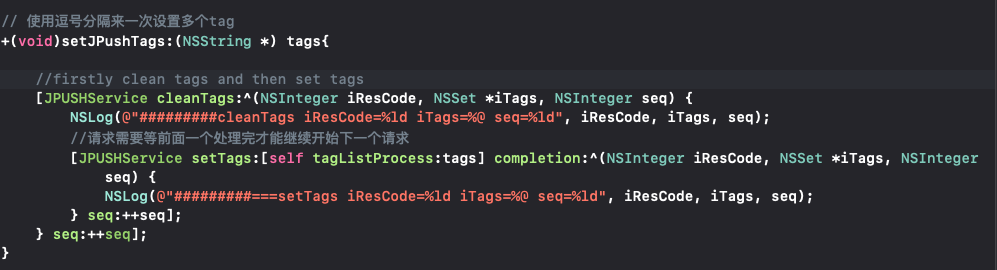
在IOS中集成主要是根据官网提供的SDK接入一些依赖库，以及引入头文件。同时注意在target中开启通知功能：





并且在app启动入口中加入初始化极光推送代码，主要包括：添加头文件、添加 Delegate、添加初始化 APNs 代码、添加初始化 JPush 代码、注册 APNs 成功并上报 DeviceToken、实现注册 APNs 失败接口（可选）、添加处理 APNs 通知回调方法。

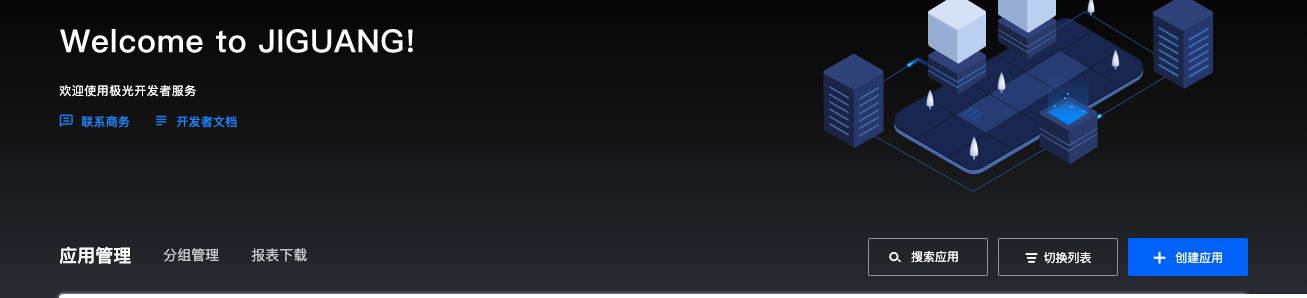
由于站点需要前端登录之后才能确认，因此将设置标签Tag的方法封装出来，提供给JS调用，一个账号可以有一个或者多个站点，因此前端在登录之后可以将stationId作为Tag通过逗号分隔的方式传递给IOS端，这边会进行解析处理，设置到极光服务器。每次调用都会先清除之前的标签，然后再将新传入的Tags设置给对应设备。

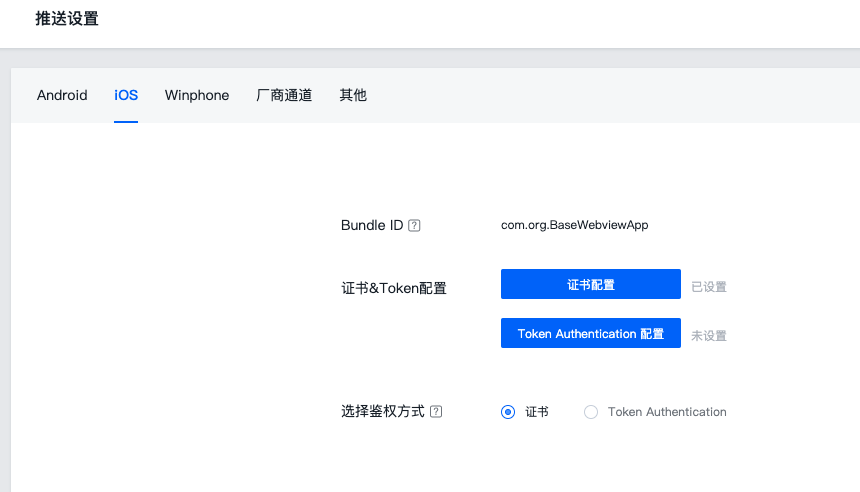


对于通知消息点击响应处理，目前考虑设计为和安卓一致，跳转到历史告警消息页面，此页面的url可以由后台在推送消息中作为附带信息传递过来，由IOS端解析获取，这样比较灵活。对于登录失效的用户，在跳转过程中会进入登录界面，此场景暂时不处理。

3、应用创建

再注册成为极光开发者之后，登录进入开发者平台，可以看到应用管理界面，由于IOS远程消息推送需要使用苹果的APNS，所以在创建应用后需要上传从苹果账号申请的证书：





证书的生成配置需要参考官方文档说明：

<https://docs.jiguang.cn/jpush/client/iOS/ios_cer_guide/>

在实际的操作过程中需要注意，证书生成后，下载双击安装后，没有在钥匙串的登录中显示，而是出现在了系统里面，此时是无法导出.p12文件的，需要将其从系统中拖入到登录证书中，然后再进行导出上传到极光平台，此处折腾了很久才在Stack Overflow上找到了解决方案。