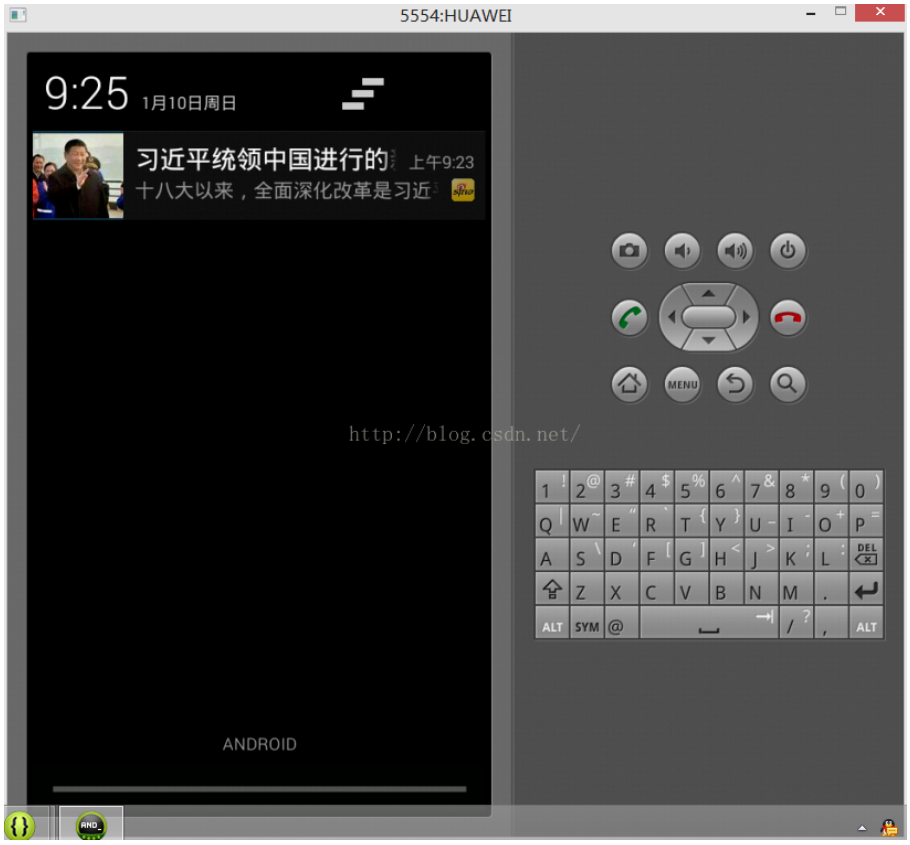
Evan
Only let oneself become strong enough, good enough, can afford the life that you want to.

Android之极光推送发送自定义消息

标签： android 极光推送 自定义消息

2016-01-10 22:27 3796人阅读 评论(0) 收藏 举报

分类： 移动开发（38）



Android端实现主要代码：

```java
<span style="font-size:14px;">import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

import org.json.JSONException;
import org.json.JSONObject;

import android.annotation.SuppressLint;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
```

```
14. import android.content.BroadcastReceiver;
15. import android.content.Context;
16. import android.content.Intent;
17. import android.graphics.Bitmap;
18. import android.graphics.BitmapFactory;
19. import android.os.Bundle;
20. import android.os.Handler;
21. import android.os.Message;
22. import android.support.v4.app.NotificationCompat;
23. 
24. import com.mine.xinlangapp.R;
25. import com.mine.xinlangapp.activity.BaseActivity;
26. 
27. import cn.jpush.android.api.JPushInterface;
28. /**
29.  * 自定义接收器
30.  *
31.  * 如果不定义这个 Receiver，则：
32.  * 1) 默认用户会打开主界面
33.  * 2) 接收不到自定义消息
34.  */
35. public class MyReceiver extends BroadcastReceiver {
36.     private Bitmap bitmap = null;
37.     private NotificationManager notifyManager = null;
38.     private NotificationCompat.Builder notifyBuilder = null;
39.     private Notification notification = null;
40.     private String url = "";
41.     @SuppressLint("HandlerLeak")
42.     private Handler handler = new Handler(){
43.         @Override
44.         public void handleMessage(Message msg){
45.             if(bitmap!=null){
46.                 notifyBuilder.setLargeIcon(bitmap);
47.             }else{
48.                 notifyBuilder.setSmallIcon(R.drawable.sina);
49.             }
50.             notification = notifyBuilder.build();
51.             notification.defaults |= Notification.DEFAULT_SOUND;
52.             notification.defaults |= Notification.DEFAULT_VIBRATE;
53.             notifyManager.notify(1000, notification);
54.         }
55.     };
56.     @SuppressLint("InflateParams")
57.     @Override
58.     public void onReceive(Context context, Intent intent) {
59.         Bundle bundle = intent.getExtras();
60.         /*
61.         //  自定义样式放在init()之后.
62.         CustomPushNotificationBuilder builder=new CustomPushNotificationBuilder(
63.              context.getApplicationContext(),
64.              R.layout.customer_notitfication_layout,
65.              R.id.icon, R.id.title, R.id.text);
66.         builder.layoutIconDrawable=R.drawable.menu_home;  //下拉状态时显示的通知图标.
67.         builder.layout = R.layout.customer_notitfication_layout;
68.         JPushInterface.setPushNotificationBuilder(2, builder);
69.         */
70. 
71.         if (JPushInterface.ACTION_MESSAGE_RECEIVED.equals(intent.getAction())) {
72.             /*
73.             //接收到推送下来的自定义消息，开启服务执行耗时的操作
74.             Intent i = new Intent(context, MyService.class);
75.             i.putExtras(bundle);
76.             context.startService(i);
77.             */
78.             processCustomMessage(context, bundle);
79.         }else if(JPushInterface.ACTION_NOTIFICATION_RECEIVED.equals(intent.getAction())){
80. 
81.         }else if (JPushInterface.ACTION_NOTIFICATION_OPENED.equals(intent.getAction())) {
82.             Intent i = new Intent(context, BaseActivity.class);
83.             bundle.putBoolean("push", true);
84.             i.putExtras(bundle);
85.             //i.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
86.             i.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TOP );
87.             context.startActivity(i);
88.         }
89.     }
90. 
91.     private void processCustomMessage(Context context, Bundle bundle){
92.         notifyManager = (NotificationManager) context.getSystemService(Context.NOTIFICATION_SERVICE);
```

```
        notifyBuilder = new NotificationCompat.Builder(context);

        String title = bundle.getString(JPushInterface.EXTRA_TITLE);
        String message = bundle.getString(JPushInterface.EXTRA_MESSAGE);
        String extras = bundle.getString(JPushInterface.EXTRA_EXTRA);
        //自定义信息： 获取
        if (extras != null) {
            try {
                JSONObject object = new JSONObject(extras);
                url = object.optString("src");
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        Intent i = new Intent(context, BaseActivity.class);
        bundle.putBoolean("push", true);
        i.putExtras(bundle);
        PendingIntent pi = PendingIntent.getActivity(context, 1000, i, PendingIntent.FLAG_UPDATE_CURRENT);

        notifyBuilder.setContentTitle(title);
        notifyBuilder.setContentText(message);
        notifyBuilder.setContentIntent(pi);
        notifyBuilder.setAutoCancel(true);

        new Thread(new Runnable() {
            @Override
            public void run() {
                bitmap = returnBitMap(url);
                handler.sendEmptyMessage(1);
            }
        }).start();

        handler.sendEmptyMessage(1);  //这里要先发送一次，因为</span><span style="font-size:14px;">onReceive</span><span style="font-size:14px;">方法实现不可以超过10秒，获取图片是耗时的，然而Notification没有图片通知是发送不了的。
    }
    //以Bitmap的方式获取一张图片
    public Bitmap returnBitMap(String url){
        URL myFileUrl = null;
        Bitmap bitmap = null;
        try{
            myFileUrl = new URL(url);
        }catch(MalformedURLException e){
            e.printStackTrace();
        }
        try{
            HttpURLConnection conn = (HttpURLConnection) myFileUrl.openConnection();
            conn.setDoInput(true);
            conn.connect();
            InputStream is = conn.getInputStream();
            bitmap = BitmapFactory.decodeStream(is);
            is.close();
        }catch(IOException e){
            e.printStackTrace();
        }
        return bitmap;
    }
}

</span>
```

服务器端java代码：

```
[java]

<span style="font-size:14px;">import java.util.HashMap;
import java.util.Map;
import java.util.Timer;
import java.util.TimerTask;

import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

import cn.jpush.api.ErrorCodeEnum;
import cn.jpush.api.JPushClient;
```

```
import cn.jpush.api.MessageResult;

public class JPushClientExample {
    private static final String appKey ="43bbac097a385c25c157e385"; //必填，例如</span><span style="font-size:14px;">43bbac097a385c25c157e385</span><span style="font-size:14px;">
    private static final String masterSecret = "90ac96cf260c77e64cc2004b";//"</span><span style="font-size:14px;">90ac96cf260c77e64cc2004b</span><span style="font-size:14px;">";//必填，每个应用都对应一个masterSecret
    private static JPushClient jpush = null;
    /**
     * 保存离线的时长。秒为单位。最多支持10天（864000秒）。
     * 0 表示该消息不保存离线。即：用户在线马上发出，当前不在线用户将不会收到此消息。
     * 此参数不设置则表示默认，默认为保存1天的离线消息（86400秒)。
     */
    private static long timeToLive = 60 * 60 * 24;
    private static String top_href = "";

//  public static void main(String[] args) {
    public void main(){
        /*
         * Example1: 初始化,默认发送给android和ios，同时设置离线消息存活时间
         * jpush = new JPushClient(masterSecret, appKey, timeToLive);
         */
        /*
         * Example2: 只发送给android
         * jpush = new JPushClient(masterSecret, appKey, DeviceEnum.Android);
         */
        /*
         * Example3: 只发送给IOS
         * jpush = new JPushClient(masterSecret, appKey, DeviceEnum.IOS);
         */
        /*
         * Example4: 只发送给android,同时设置离线消息存活时间
         * jpush = new JPushClient(masterSecret, appKey, timeToLive, DeviceEnum.Android);
         */
        jpush = new JPushClient(masterSecret, appKey, timeToLive);
        /*
         * 是否启用ssl安全连接，可选
         * 参数：启用true， 禁用false，默认为非ssl连接
         */
        //jpush.setEnableSSL(true);

        Timer timer = new Timer();
        //在1秒后执行此任务，每次间隔半小时，如果传递一个Data参数，就可以在某个固定的时间执行这个任务
        timer.schedule(new MyTask(), 1000, 1800*1000);
    }

    private static class MyTask extends TimerTask{
        @Override
        public void run() {
            Map<String, String> map = getNews();
            String href = map.get("href");
            if(!top_href.equals(href)){     //判断与上次发送的是否相同，不相同就推送
                top_href = href;
                //测试发送消息或者通知
                testSend(map);
            }
        }
    }

    private static Map<String, String> getNews(){
        Document doc = null;
        Map<String, String> map = new HashMap<String, String>();
        try {
             doc = Jsoup.connect("http://news.sina.cn/").get();
             Elements ListDiv = doc.getElementsByAttributeValue("class","carditems");
            for (int i = 0; i<ListDiv.select("dl").size() ; i++) {
                Element a = ListDiv.select("a").get(i);
                String href = a.attr("href");
                Element dl = ListDiv.select("dl").get(i);
                Element dd = dl.select("dd").get(0);
                Elements dt = dl.getElementsByTag("img");
                Elements h3 = dd.select("h3");    //标题
                Elements h4 = dd.select("h4");  //内容
                String title = h3.text();
                String fu_title = h4.text();
                String url = dt.attr("src"); // 图片链接

                if(fu_title!=null && !fu_title.equals("")){
                        map.put("src", url);
```

```
                        map.put("title", title);
                        map.put("fu_title", fu_title);
                        map.put("href", href);
                        break;
                    }
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
            return map;
        }

        private static void testSend(Map<String, String> map) {
            // 在实际业务中，建议 sendNo 是一个你自己的业务可以处理的一个自增数字。
            // 除非需要覆盖，请确保不要重复使用。详情请参考 API 文档相关说明。
            int sendNo = getRandomSendNo();
            //String msgTitle = "+;//jpush\"\"";
            //String msgContent = "\\&;w\"\"a--【\npush】";

            String href = map.get("href");
            String msgTitle = map.get("title");
            String msgContent=map.get("fu_title");
            String url = map.get("src");    //图片地址

            /*
             * IOS设备扩展参数，
             * 设置badge，设置声音
             */
            Map<String, Object> extra = new HashMap<String, Object>();
            extra.put("href", href);
            extra.put("src", url);
        //  IOSExtra iosExtra = new IOSExtra(10, "WindowsLogonSound.wav");
        //  extra.put("ios", iosExtra);

            //对所有用户发送通知，更多方法请参考文档     message字段
        //  MessageResult msgResult = jpush.sendNotificationWithAppKey(sendNo, msgTitle, msgContent, 2, extra);
            MessageResult msgResult = jpush.sendCustomMessageWithAppKey(sendNo, msgTitle, msgContent, "a", extra);  //发送自定义消息
            //MessageResult msgResult = jpush.sendCustomMessageWithAppKey(sendNo,msgTitle, msgContent);
            //MessageResult msgResult  = jpush.sendNotificationWithAlias(sendNo, "a", msgTitle, msgContent);

            //覆盖指定msgId的消息,msgId可以从msgResult.getMsgid()获取。
            //MessageResult msgResult = jpush.sendNotificationWithAppKey(sendNo, msgTitle, msgContent, 0, extra,msgResult.getMsgid());

            if (null != msgResult) {
                System.out.println("服务器返回数据: " + msgResult.toString());
                if (msgResult.getErrcode() == ErrorCodeEnum.NOERROR.value()) {
                    System.out.println(String.format("发送成
功， sendNo= %s,messageId= %s",msgResult.getSendno(),msgResult.getMsg_id()));
                } else {
                    System.out.println("发送失败， 错误代码=" + msgResult.getErrcode() + ", 错误消息=" + msgResult.getErrmsg());
                }
            } else {
                System.out.println("无法获取数据");
            }
        }

        public static final int MAX = Integer.MAX_VALUE;
        public static final int MIN = (int) MAX/2;

        /**
         * 保持 sendNo 的唯一性是有必要的
         * It is very important to keep sendNo unique.
         * @return sendNo
         */
        public static int getRandomSendNo() {
            return (int) (MIN + Math.random() * (MAX - MIN));
        }

    }
    </span>
```