

ARTIGO: 11515

Android Push Notifications

Olá,

Vamos fazer um passo a passo de como gerar as chaves e testar as notificações no android:

Gerando ao chave de acesso para a API:

Primeiro acesso o painel de desenvolvedor do google em:

<https://console.developers.google.com/apis/library>

Nesse painel crie o projeto para seu aplicativo. Depois de criado o projeto, acesse o link:

<https://developers.google.com/identity/sign-in/android/start-integrating>

Nesse link tem os detalhes de implementação. Procure o botão chamado "**Get a configuration file**" e vá nessa opção.

Em "**App Name**" selecione seu projeto e em "**android package name**" informe o pacote (package) do seu aplicativo. Confirme em "**choose and configure services**".

Na próxima tela, habilite a opção "**Cloud Messages**", isso vai permitir seu app a receber mensagens. Quando habilitado será exibido o "**Server API Key**" e "**Sender ID**". Salve esses dois valores, vamos precisar deles para configurar as mensagens.

Por fim, vá em "**Generate configuration file**" e faça o download do arquivo em "**Download google-services.json**"

Implementando o recebimento de mensagens no aplicativo:

Salve o arquivo **google-service.json** na pasta **app**, no primeiro nível do projeto. Nesse mesmo nível procure o arquivo **build.gradle** e adicione dentro de dependencies:

```
classpath 'com.google.gms:google-services:1.5.0'
```

Agora entre na pasta **app** no próximo nível e edite o **build.gradle**:

Uma linha após **apply plugin: 'com.android.application'** adicione:

```
apply plugin: 'com.google.gms.google-services'
```

Dentro de **defaultConfig** adicione:

```
applicationId "br.com.your.package.name"
```

e por fim, em **dependencies** adicione:

```
compile 'com.android.support:appcompat-v7:22.2.0'  
compile "com.google.android.gms:play-services:8.3.0"
```

Agora vamos criar as seguintes classes para gerenciamento:

//MyGcmListenerService

```
package br.com.mobilemind.app;

import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Bundle;
import android.support.v4.app.NotificationCompat;
import android.util.Log;

import com.google.android.gms.gcm.GcmListenerService;

import br.com.mobilemind.app.MainActivity;
import br.com.mobilemind.app.R;

public class MyGcmListenerService extends GcmListenerService {

    private static final String TAG = "MyGcmListenerService";

    /**
     * Called when message is received.
     *
     * @param from SenderID of the sender.
     * @param data Data bundle containing message data as key/value pairs.
     *           For Set of keys use data.keySet().
     */
    // [START receive_message]
    @Override
    public void onMessageReceived(String from, Bundle data) {
        String message = data.getString("message");
        Log.d(TAG, "From: " + from);
        Log.d(TAG, "Message: " + message);

        if (from.startsWith("/topics/")) {
            // message received from some topic.
        } else {
            // normal downstream message.
        }

        // [START_EXCLUDE]
        /**
         * Production applications would usually process the message here.
         * Eg: - Syncing with server.
         *      - Store message in local database.
         *      - Update UI.
         */

        /**
         * In some cases it may be useful to show a notification indicating to the user
         * that a message was received.
         */
        sendNotification(message);
        // [END_EXCLUDE]
    }
    // [END receive_message]

    /**
     * Create and show a simple notification containing the received GCM message.
     *
     * @param message GCM message received.
     */
    private void sendNotification(String message) {
        Intent intent = new Intent(this, MainActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0 /* Request code */, intent,
            PendingIntent.FLAG_ONE_SHOT);

        Uri defaultSoundUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(this)
            .setSmallIcon(R.drawable.ic_launcher)
            .setContentTitle("GCM Message")
            .setContentText(message)
            .setAutoCancel(true)
            .setSound(defaultSoundUri)
            .setContentIntent(pendingIntent);

        NotificationManager notificationManager =
            (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
```

```

        notificationManager.notify(0 /* ID of notification */, notificationBuilder.build());
    }
}

```

//MyInstanceIdListenerService

```

package br.com.mobilemind.app;

import android.content.Intent;

import android.content.Intent;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.util.Log;

import com.google.android.gms.iid.InstanceID;
import com.google.android.gms.iid.InstanceIDListenerService;

public class MyInstanceIdListenerService extends InstanceIDListenerService {

    private static final String TAG = "MyInstanceIDLS";

    /**
     * Called if InstanceID token is updated. This may occur if the security of
     * the previous token had been compromised. This call is initiated by the
     * InstanceID provider.
     */
    // [START refresh_token]
    @Override
    public void onTokenRefresh() {
        // Fetch updated Instance ID token and notify our app's server of any changes (if applicable).
        Intent intent = new Intent(this, RegistrationIntentService.class);
        startService(intent);
    }
    // [END refresh_token]
}

```

//RegistrationIntentService

```

package br.com.mobilemind.app;

import android.app.IntentService;
import android.content.Intent;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.support.v4.content.LocalBroadcastManager;
import android.util.Log;

import com.google.android.gms.gcm.GcmPubSub;
import com.google.android.gms.gcm.GoogleCloudMessaging;
import com.google.android.gms.iid.InstanceID;

import java.io.IOException;

import br.com.mobilemind.app.R;

public class RegistrationIntentService extends IntentService {

    private static final String TAG = "RegIntentService";
    private static final String[] TOPICS = {"global"};

    public RegistrationIntentService() {
        super(TAG);
    }

    @Override
    protected void onHandleIntent(Intent intent) {
        SharedPreferences sharedPreferences = PreferenceManager.getDefaultSharedPreferences(this);

        Log.i(TAG, "GCM Registration onHandleIntent");

        try {
            // [START register_for_gcm]
            // Initially this call goes out to the network to retrieve the token, subsequent calls
            // are local.
            // R.string.gcm_defaultSenderId (the Sender ID) is typically derived from google-services.json.
            // See https://developers.google.com/cloud-messaging/android/start for details on this file.

```

```

    // [START get_token]
    InstanceID instanceID = InstanceID.getInstance(this);
    String token = instanceID.getToken(getString(R.string.gcm_sender_id),
        GoogleCloudMessaging.INSTANCE_ID_SCOPE, null);
    // [END get_token]
    Log.i(TAG, "GCM Registration Token: " + token);

    // TODO: Implement this method to send any registration to your app's servers.
    sendRegistrationToServer(token);

    // Subscribe to topic channels
    subscribeTopics(token);

    // You should store a boolean that indicates whether the generated token has been
    // sent to your server. If the boolean is false, send the token to your server,
    // otherwise your server should have already received the token.
    sharedPreferences.edit().putBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, true).apply();
    // [END register_for_gcm]
} catch (Exception e) {
    Log.d(TAG, "Failed to complete token refresh", e);
    // If an exception happens while fetching the new token or updating our registration data
    // on a third-party server, this ensures that we'll attempt the update at a later time.
    sharedPreferences.edit().putBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, false).apply();
}
// Notify UI that registration has completed, so the progress indicator can be hidden.
Intent registrationComplete = new Intent(QuickstartPreferences.REGISTRATION_COMPLETE);
LocalBroadcastManager.getInstance(this).sendBroadcast(registrationComplete);
}

/**
 * Persist registration to third-party servers.
 *
 * Modify this method to associate the user's GCM registration token with any server-side account
 * maintained by your application.
 *
 * @param token The new token.
 */
private void sendRegistrationToServer(String token) {
    // Add custom implementation, as needed.
}

/**
 * Subscribe to any GCM topics of interest, as defined by the TOPICS constant.
 *
 * @param token GCM token
 * @throws IOException if unable to reach the GCM PubSub service
 */
// [START subscribe_topics]
private void subscribeTopics(String token) throws IOException {
    GcmPubSub pubSub = GcmPubSub.getInstance(this);
    for (String topic : TOPICS) {
        pubSub.subscribe(token, "/topics/" + topic, null);
    }
}
// [END subscribe_topics]
}

```

//QuickstartPreferences

```

package br.com.mobilemind.app;

public class QuickstartPreferences {

    public static final String SENT_TOKEN_TO_SERVER = "sentTokenToServer";
    public static final String REGISTRATION_COMPLETE = "registrationComplete";
}

```

Agora precisamos alterar nossa classe principal.. ela deve ficar assim:

```

//MainActivity

package br.com.mobilemind.app;

import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.BroadcastReceiver;

```

```

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.support.v4.content.LocalBroadcastManager;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GoogleApiAvailability;

public class MainActivity extends ActionBarActivity {

    private static final int PLAY_SERVICES_RESOLUTION_REQUEST = 9000;
    private static final String TAG = "MainActivity";

    private BroadcastReceiver mRegistrationBroadcastReceiver;

    public MainActivity() {
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mRegistrationBroadcastReceiver = new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                SharedPreferences sharedPreferences =
                    PreferenceManager.getDefaultSharedPreferences(context);
                boolean sentToken = sharedPreferences
                    .getBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, false);

                // aqui você pode enviar o token gerado para algum lugar
                Log.d("## SENT TOKEN", sentToken + "");
            }
        };

        if (checkPlayServices()) {
            Log.d("## SENT TOKEN", " START SERVICE");
            // Start IntentService to register this application with GCM.
            Intent intent = new Intent(this, RegistrationIntentService.class);
            startService(intent);
        }
    }

    @Override
    protected void onResume() {
        super.onResume();
        LocalBroadcastManager.getInstance(this).registerReceiver(mRegistrationBroadcastReceiver,
            new IntentFilter(QuickstartPreferences.REGISTRATION_COMPLETE));
    }

    @Override
    protected void onPause() {
        LocalBroadcastManager.getInstance(this).unregisterReceiver(mRegistrationBroadcastReceiver);
        super.onPause();
    }

    /**
     * Check the device to make sure it has the Google Play Services APK. If
     * it doesn't, display a dialog that allows users to download the APK from
     * the Google Play Store or enable it in the device's system settings.
     */
    private boolean checkPlayServices() {
        GoogleApiAvailability apiAvailability = GoogleApiAvailability.getInstance();
        int resultCode = apiAvailability.isGooglePlayServicesAvailable(this);
    }
}

```

```
    if (resultCode != ConnectionResult.SUCCESS) {
        if (apiAvailability.isUserResolvableError(resultCode)) {
            apiAvailability.getErrorDialog(this, resultCode, PLAY_SERVICES_RESOLUTION_REQUEST)
                .show();
        } else {
            Log.i(TAG, "This device is not supported.");
            finish();
        }
        return false;
    }
    return true;
}
}
```

Pronto. Agora, para testar precisamos executar isso uma vez e copiar o *token* gerado. Com o *token* gerado e o **Server API Key** executamos usando o *curl*:

```
// substitua SERVER_KEY_API pelo valor gerado no console do Google e APP_TOKEN pelo token gerado no aplicativo

curl --header "Authorization: key=SERVER_API_KEY" \
--header "Content-Type: application/json" \
https://gcm-http.googleapis.com/gcm/send -d '{"notification": {"title": "teste", "text": "teste"}, "to" :
"APP_TOKEN"}'
```

Depois disso uma mensagem deve ser exibida no seu app.