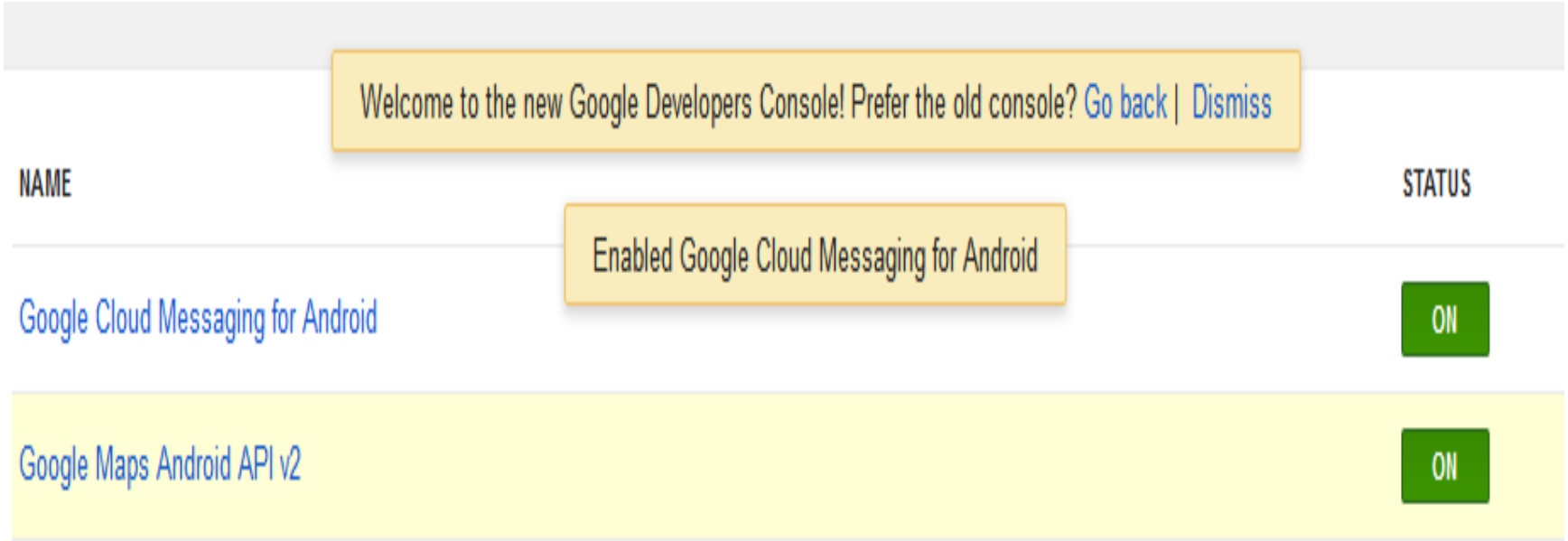
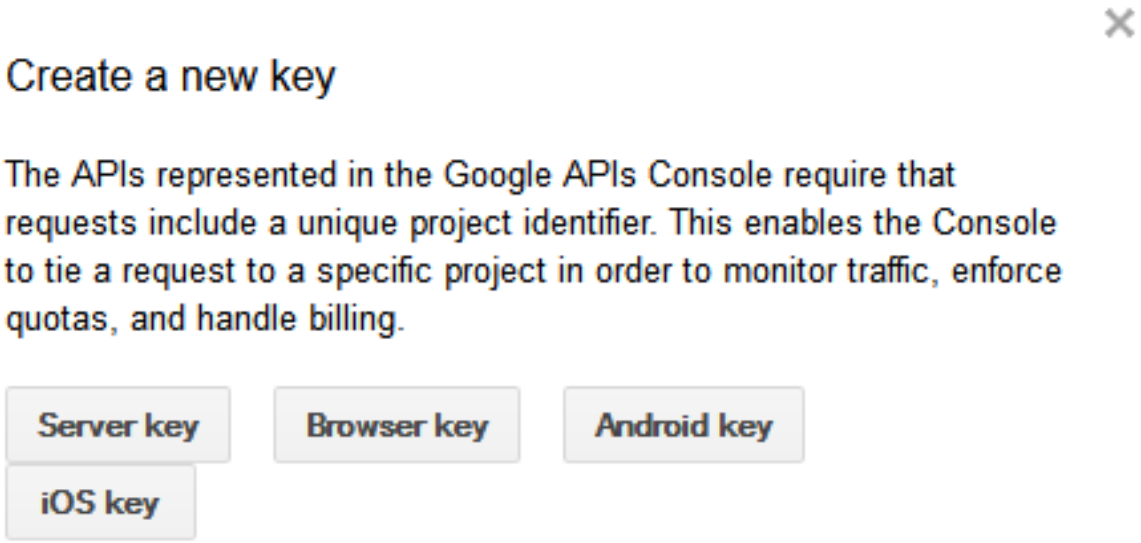
GCM

Google Cloud Messaging

GCM for Android



Server key


Create a new key
The APIs represented in the Google APIs Console require that requests include a unique project identifier. This enables the Console to tie a request to a specific project in order to monitor traffic, enforce quotas, and handle billing.
Server key
Browser key
Android key
iOS key

Create

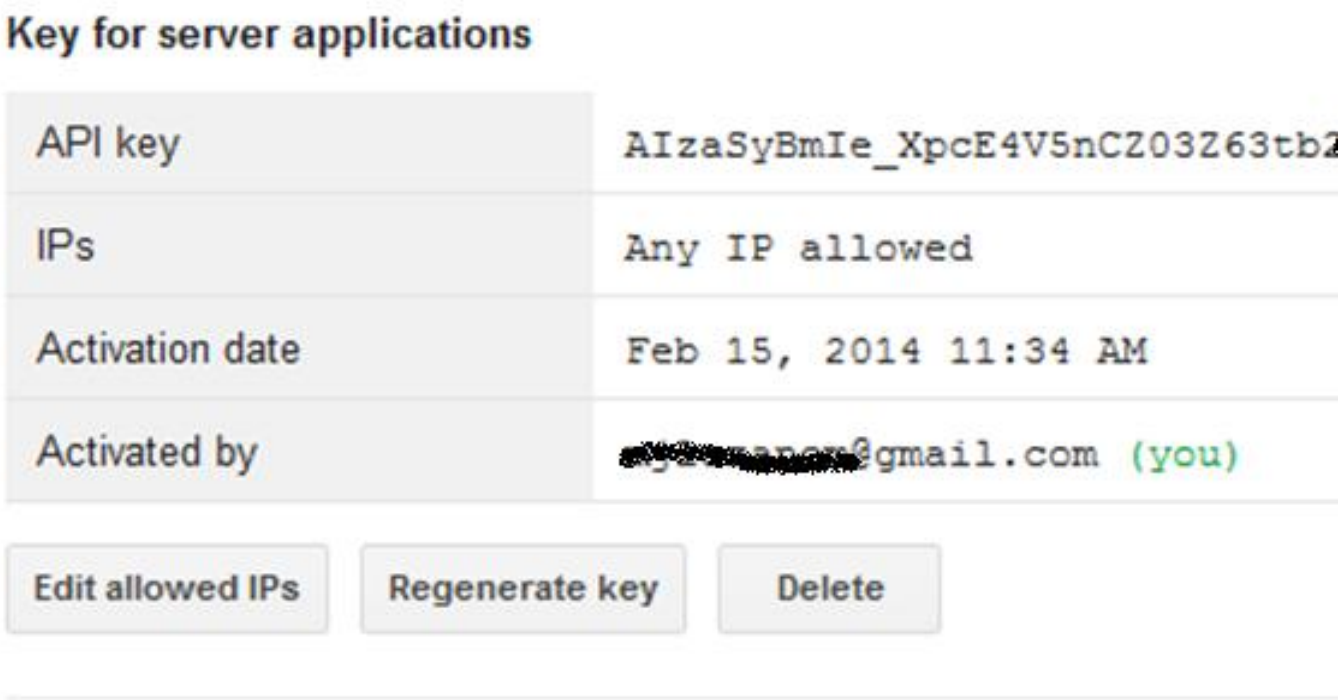
Create a server key and configure allowed IPs

This key should be kept secret on your server.
Every API request is generated by software running on a machine that you control. Per-user limits will be enforced using the address found in each request's `userIp` parameter, (if specified). If the `userIp` parameter is missing, your machine's IP address will be used instead. Learn more

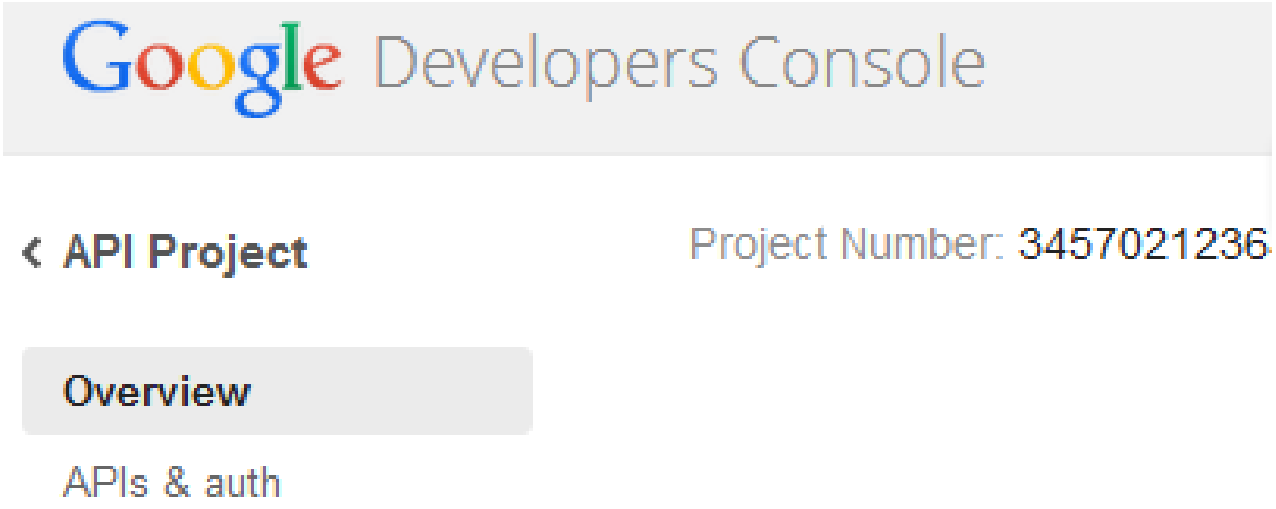
Accept requests from these server IP addresses
One IP address or subnet per line. Example: 192.168.0.1 or 172.16.0.0/16

Create Cancel

API Key


Key for server applications
API key
AIzaSyBmIe_XpcE4V5nCZ03Z63tb
IPs
Any IP allowed
Activation date
Feb 15, 2014 11:34 AM
Activated by
gmail.com (you)
Edit allowed IPs
Regenerate key
Delete

Project Number



Permisos Cliente GCM

- Permisos

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

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

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

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

<uses-permission
android:name="com.google.android.c2dm.permission.RECEIVE" />

Permisos Cliente

- Más permisos:

<permission android:name="paquete.permission.C2D_MESSAGE" android:protectionLevel="signature" />

<uses-permission android:name="paquete.permission.C2D_MESSAGE" />

Meta datos

- Meta datos en la etiqueta application:

<meta-data android:name="com.google.android.gms.version"
android:value="@integer/google_play_services_version" />

Constantes y variables de instancia

```
private static final int PLAY_SERVICES_RESOLUTION_REQUEST = 9000;
private static final String PROPERTY_REG_ID = "registration_id";
private static final String SENDER_ID = "345702123...";

private GoogleCloudMessaging gcm;

…

gcm = GoogleCloudMessaging.getInstance(this);
```

Comprobar Google Play Services

```
private boolean checkPlayServices() {
    int resultCode = GooglePlayServicesUtil.isGooglePlayServicesAvailable(this);
    if (resultCode != ConnectionResult.SUCCESS) {
        if (GooglePlayServicesUtil.isUserRecoverableError(resultCode)) {
            GooglePlayServicesUtil.getErrorDialog(resultCode, this,
                    PLAY_SERVICES_RESOLUTION_REQUEST).show();
        } else {
            Log.i(TAG, "This device is not supported.");
            finish();
        }
        return false;
    }
    return true;
}
```

Registrar dispositivo

```
private void registrarDispositivo() {
  new AsyncTask<Void, Void, Void>() {
    @Override
    protected Void doInBackground(Void... params) {
      try {
        idRegistro = gcm.register(SENDER_ID); //project number
        registrarDispositivoServidor();
      } catch (IOException e) {
      }
      return null;
    }
  }.execute(null, null, null);
}
```

Guardar id del registro

```
SharedPreferences prefs = getSharedPreferences(Actividad.class.getSimpleName(),
    Context.MODE_PRIVATE);
SharedPreferences.Editor editor = prefs.edit();
editor.putString(PROPERTY_REG_ID, idRegistro);
editor.commit();
```

```
SharedPreferences prefs = getSharedPreferences(Actividad.class.getSimpleName(),
    Context.MODE_PRIVATE);
String registrationId = prefs.getString(PROPERTY_REG_ID, "");
```

Registrar dispositivo en servidor

```
URL url = new URL("http://dominio/android/registro.php");
URLConnection con = (URLConnection) url.openConnection();
con.setDoOutput(true);
OutputStreamWriter out = new OutputStreamWriter(
    con.getOutputStream());
out.write("gcmregid=" + idRegistro + "&idterminal=" + numeroTelefono);
out.close();
BufferedReader in = new BufferedReader(new InputStreamReader(
    con.getInputStream()));
String linea;
while ((linea = in.readLine()) != null) {
    …
}
in.close();
```

Cliente

- Falta implementar en el cliente el envío de mensajes: envíos get o post al servidor con el identificador del destino (teléfono) y el mensaje.
- También falta implementar en el cliente la recepción de mensajes.

```
private void enviarSegundoPlano(final String texto) {
    new AsyncTask<Void, Void, Void>() {
        @Override
        protected Void doInBackground(Void... params2) {
            URL url = null;
            url = new URL("http://dominio/android/enviar.php?regId="+ idRegistro +
                "&mensaje=" + texto + "&destino=" + telefono);
            URLConnection conn = (URLConnection) url.openConnection();
            BufferedReader in = new BufferedReader(
                new InputStreamReader(conn.getInputStream()));
            String linea;
            while ((linea = in.readLine()) != null){
                …;
            }
            in.close();
            return null;
        }
    }.execute(null, null, null);
}
```

Receptor Broadcast

```
<receiver
      android:name="paquete.ReceptorBroadcast"
      android:permission="com.google.android.c2dm.permission.SEND" >
      <intent-filter>
         <action android:name="com.google.android.c2dm.intent.RECEIVE" />
         <category android:name="paquete" />
      </intent-filter>
</receiver>
```

ReceptorBroadcast

```
public class ReceptorBroadcast extends WakefulBroadcastReceiver{
   @Override
   public void onReceive(Context cntxt, Intent intent) {
      ComponentName comp = new ComponentName(cntxt.getPackageName(),
            Servicio.class.getName());
      startWakefulService(cntxt, (intent.setComponent(comp)));
      setResultCode(Activity.RESULT_OK);
   }
}
```

Servicio

```
<service android:name="paquete.Servicio" />
```

```
public class Servicio extends IntentService{
    public static final int NOTIFICATION_ID = 1;
    private static int contadorID = 0;
    private NotificationManager gestorNotificacion;

    public Servicio() {
        super("Servicio");
    }
}
```

onHandleIntent

```
@Override
protected void onHandleIntent(Intent intent) {
  Bundle extras = intent.getExtras();
  GoogleCloudMessaging gcm = GoogleCloudMessaging.getInstance(this);
  String messageType = gcm.getMessageType(intent);
  if (!extras.isEmpty()) {
   if (GoogleCloudMessaging.MESSAGE_TYPE_SEND_ERROR.equals(messageType)){
        sendNotification("Send error: " + extras.toString());
   } else if(GoogleCloudMessaging.MESSAGE_TYPE_DELETED.equals(messageType)) {
        sendNotification("Deleted messages on server: " + extras.toString());
   } else if(GoogleCloudMessaging.MESSAGE_TYPE_MESSAGE.equals(messageType)) {
        sendNotification("Received: " + extras.toString());
   }
  }
  ReceptorBroadcast.completeWakefulIntent(intent);
}
```

```
private void sendNotification(String msg) {
    contadorID++;
    gestorNotificacion = (NotificationManager)
        this.getSystemService(Context.NOTIFICATION_SERVICE);
    Intent i = new Intent(this, Actividad.class);
    i.putExtra("mensaje", msg);
    i.setFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP|
        Intent.FLAG_ACTIVITY_CLEAR_TOP);
    PendingIntent contentIntent = PendingIntent.getActivity(this,
        NOTIFICATION_ID+contadorID, i, PendingIntent.FLAG_UPDATE_CURRENT);
    NotificationCompat.Builder builder = new NotificationCompat.Builder(this)
        .setSmallIcon(R.drawable.ic_launcher)
        .setContentTitle("GCM")
        .setStyle(new NotificationCompat.BigTextStyle()
        .bigText(msg))
        .setAutoCancel(true)
        .setContentText(msg);
        builder.setContentIntent(contentIntent);
        gestorNotificacion.notify(NOTIFICATION_ID+contadorID, builder.build());
}
```

Actividad

```
@Override
protected void onNewIntent(Intent intent) {
    Bundle extras = intent.getExtras();
    if (extras != null) {
        String mensaje = extras.getString("mensaje");
        tv.append("recibido: " + mensaje + "\n");
    }
    super.onNewIntent(intent);
}
```

Servidor

- En el servidor damos de alta todos los clientes de la aplicación.
- registro.php: de cada cliente guardamos id de registro, y, por ejemplo, su número de teléfono.
- enviar.php?regId=id&mensaje=texto&destino =telefono; del cliente regID se envía el mensaje al cliente telefono

enviar.php I

```
$url = 'https://android.googleapis.com/gcm/send';
$fields = array(
    'registration_ids' => $registatoin_ids_destino,
    'data' => $mensaje,
);
$headers = array(
    'Authorization:key=AIzaSyBmIe_XpcE4V5…',
    'Content-Type: application/json'
);
```

enviar.php II

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($fields));
$result = curl_exec($ch);
if ($result === FALSE) {
    die('Curl failed: ' . curl_error($ch));
}
curl_close($ch);
echo $result;
```

- http://developer.android.com/google/gcm/index.html