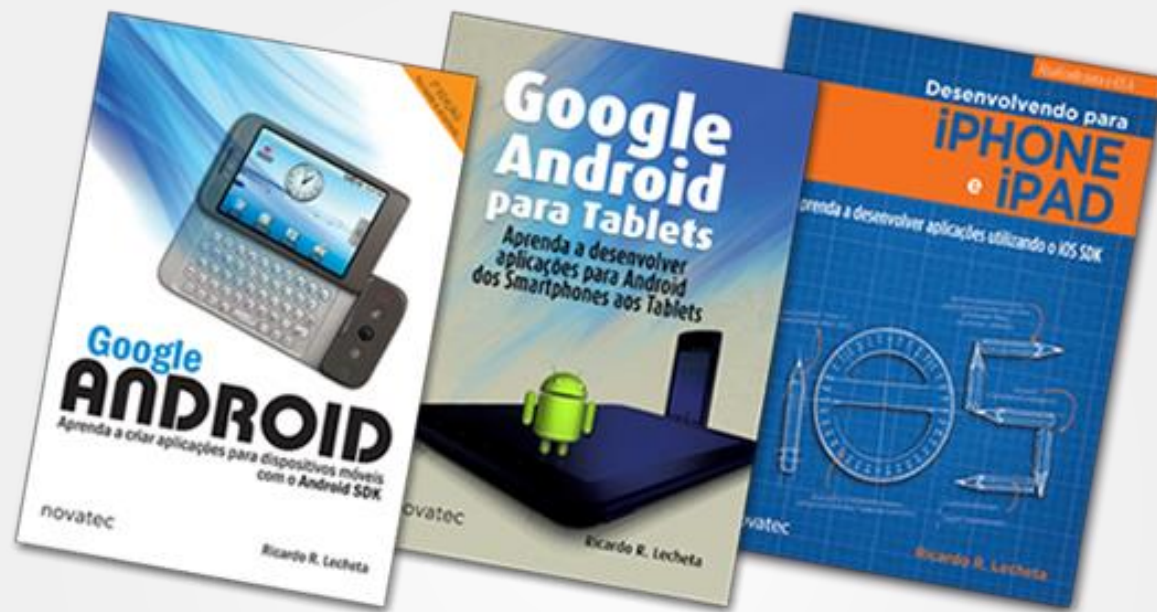


GCM - Google Cloud Messaging



Ricardo R. Lecheta



livetouch

Ricardo R. Lecheta

Agenda

- ✓ GCM
- ✓ Google Console
- ✓ Gerando as chaves de acesso
- ✓ Projeto de exemplo: Android
- ✓ Projeto de exemplo: Servidor

GCM

- ✓ Serviço disponibilizado pelo Google para enviar mensagens de push para o aplicativo Android.

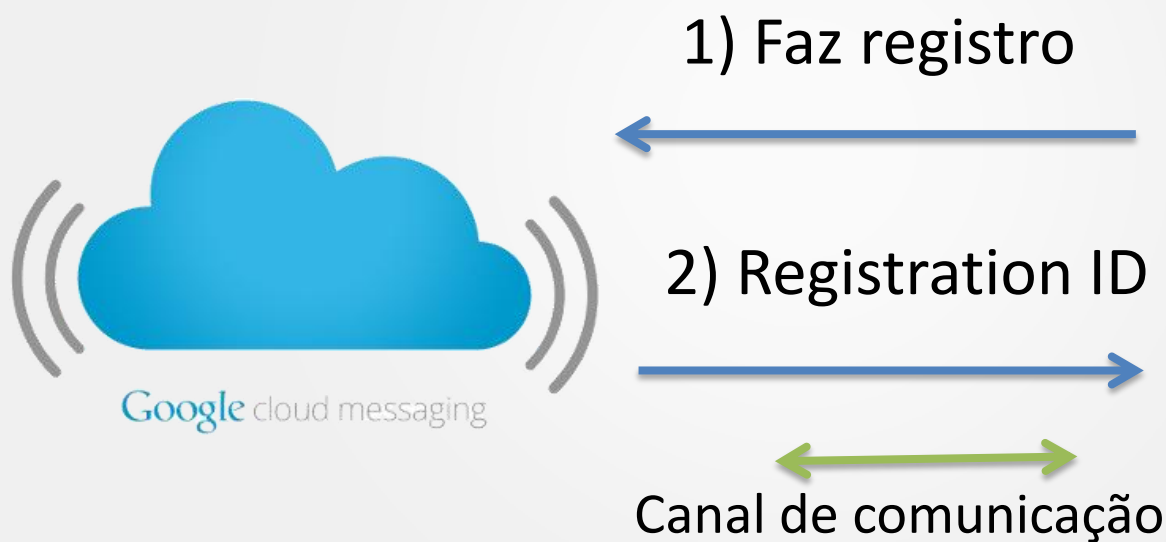


Canal de comunicação
Wi-Fi / 3G



GCM - Ativação

- ✓ Registration ID: é o identificador deste dispositivo.



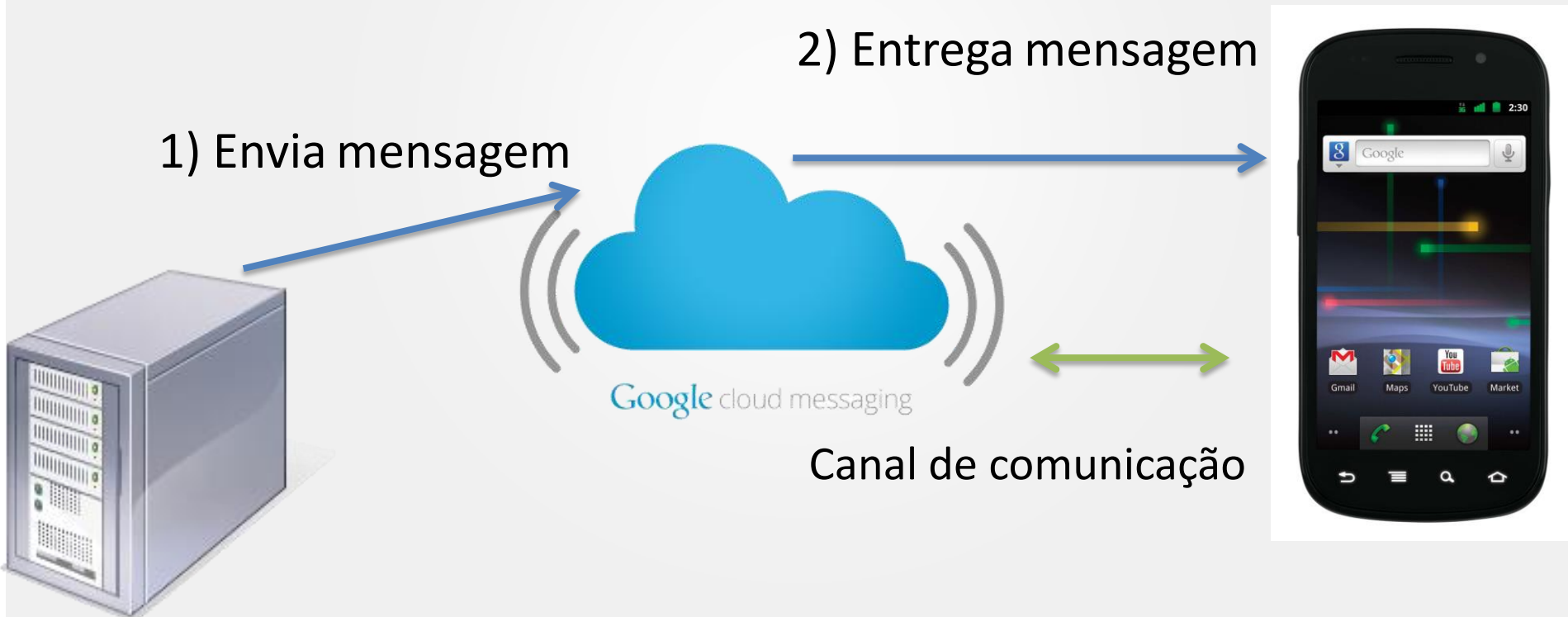
GCM – Registration ID

- ✓ Registration ID: precisa ser armazenado no servidor.
- ✓ Com ele o servidor consegue enviar mensagens para este dispositivo



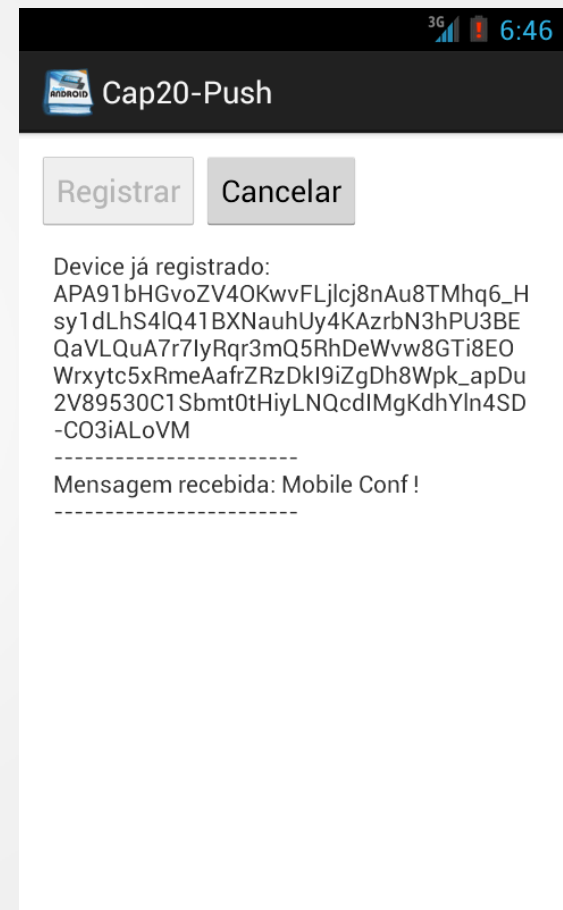
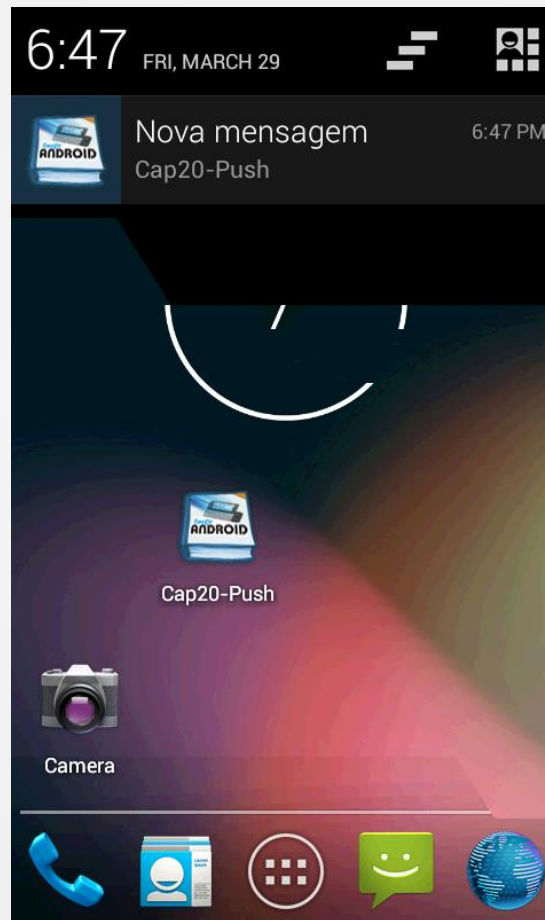
GCM

- ✓ Servidor do GCM apenas recebe a mensagem, e entrega ao device.



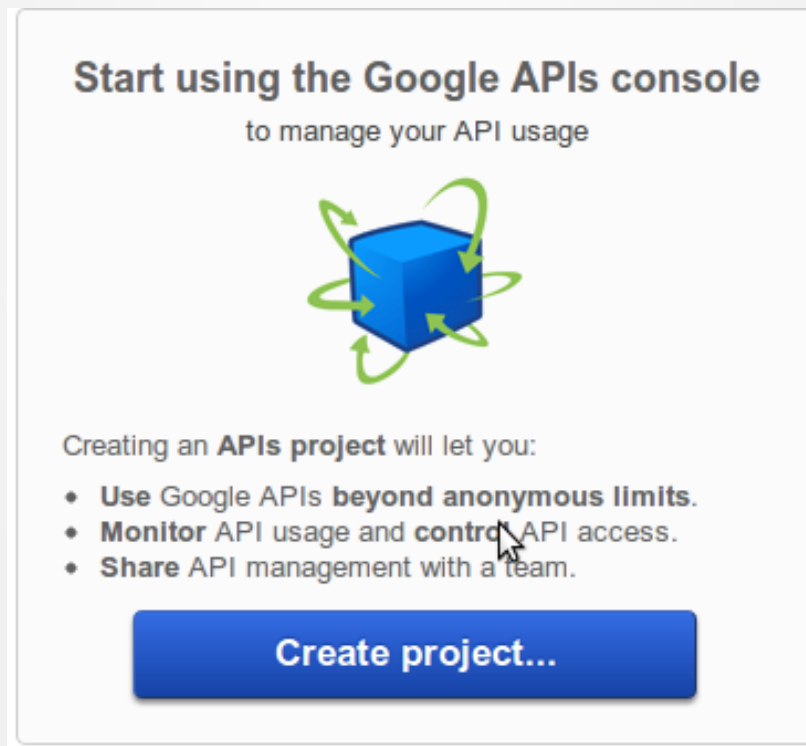
Demo

✓ Demo



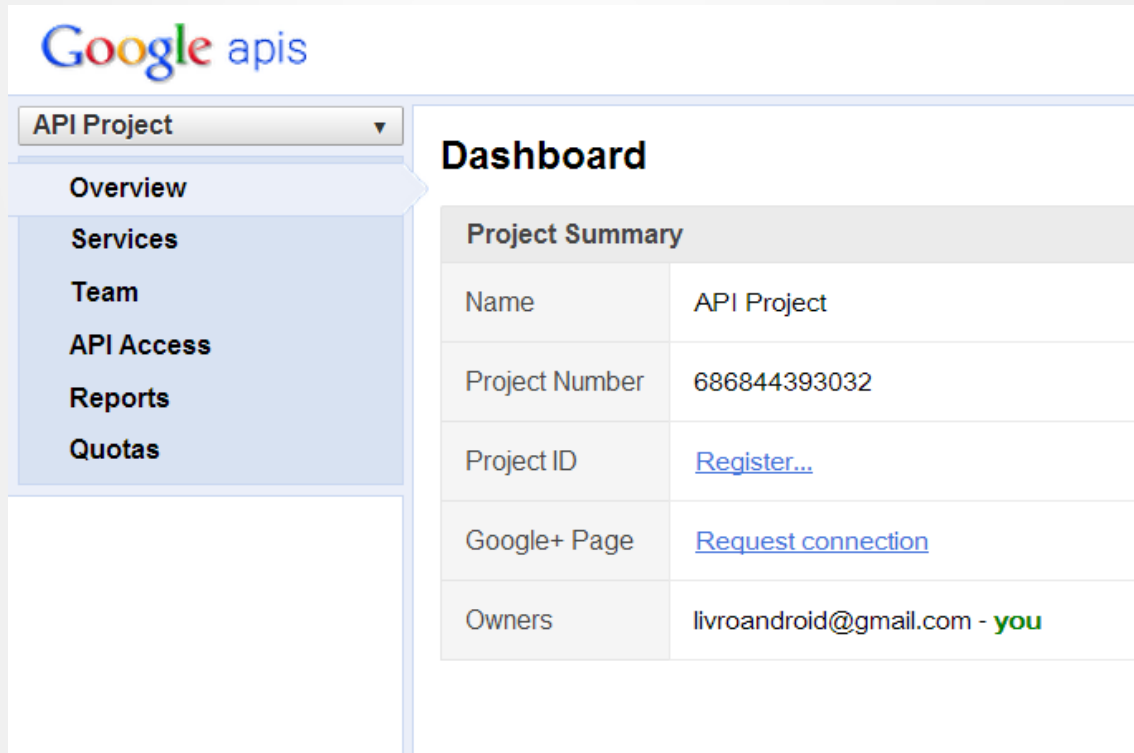
Google APIs Console

- ✓ Para ativar o serviço do GCM, basta criar um projeto no console.
- ✓ <https://code.google.com/apis/console>



Project Number (Sender ID)

- ✓ Copiar o **Project Number**. Utilizado no aplicativo Android para receber as mensagens.
- ✓ <https://code.google.com/apis/console/#project:686844393032>

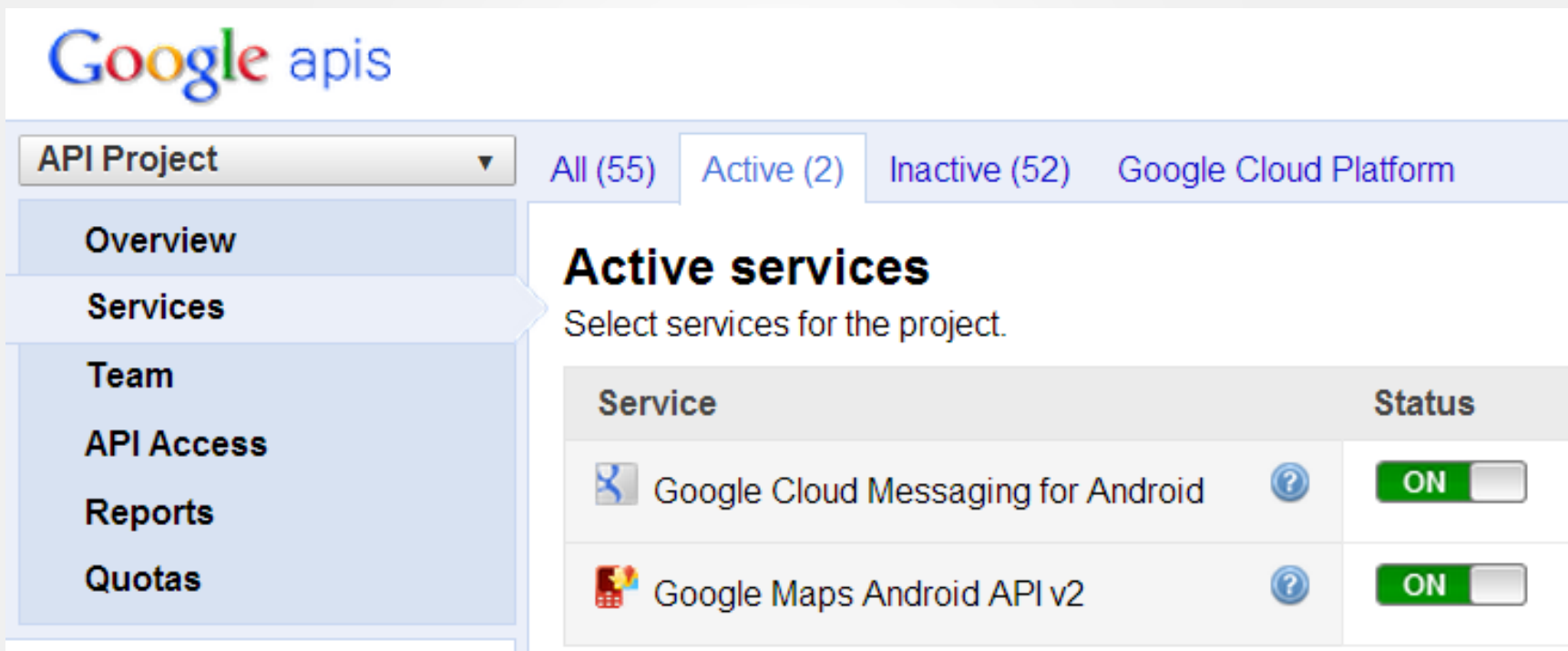


The screenshot shows the Google APIs Dashboard. On the left, there is a navigation menu with the following items: Overview (selected), Services, Team, API Access, Reports, and Quotas. The main content area is titled 'Dashboard' and contains a 'Project Summary' table.





Project Summary	
Name	API Project
Project Number	686844393032
Project ID	Register...
Google+ Page	Request connection
Owners	livroandroid@gmail.com - you

Ativar o serviço do GCM

✓ Menu > **Services.**

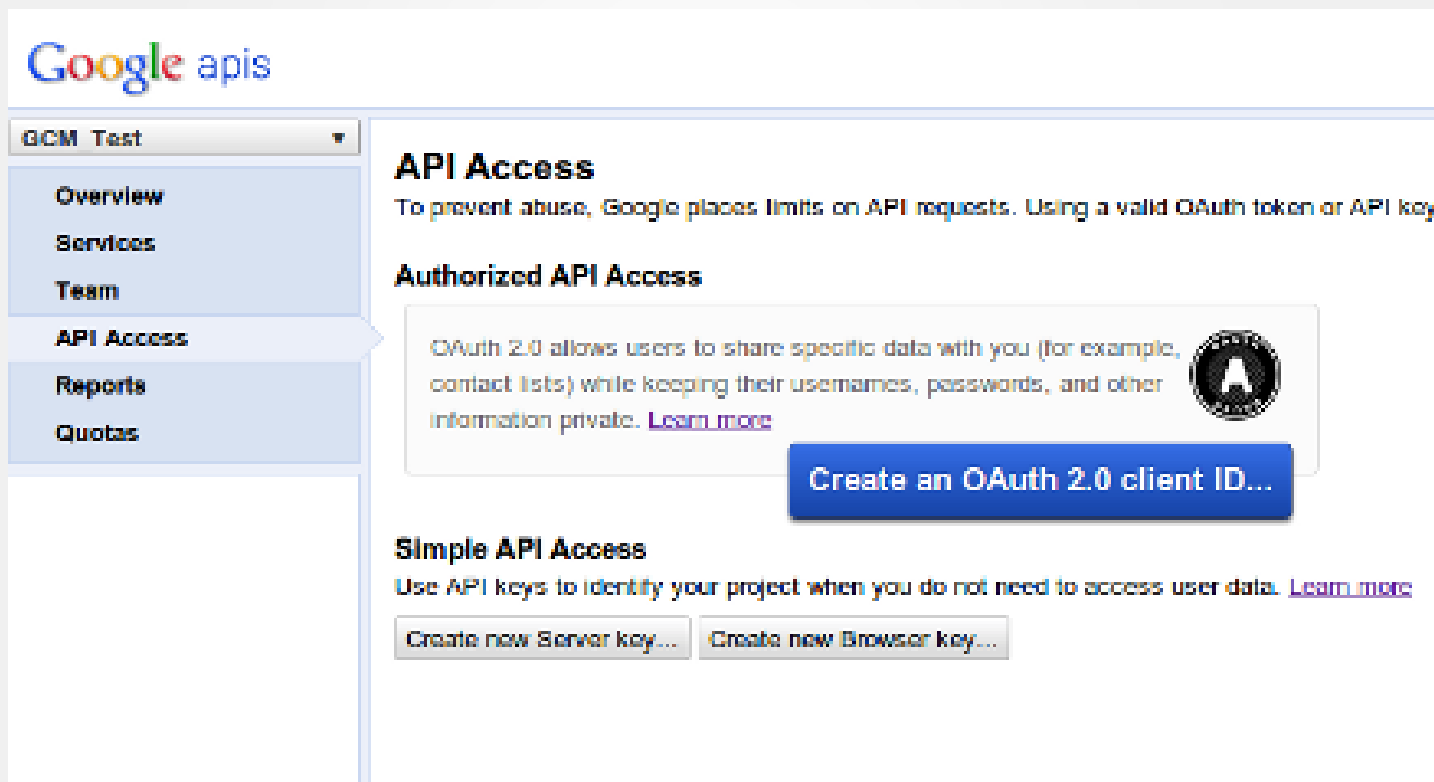


The screenshot shows the Google APIs console interface. On the left is a sidebar with a menu containing 'Overview', 'Services' (highlighted), 'Team', 'API Access', 'Reports', and 'Quotas'. Above the menu is a dropdown for 'API Project'. To the right of the menu are tabs for 'All (55)', 'Active (2)', 'Inactive (52)', and 'Google Cloud Platform'. The main content area is titled 'Active services' with the instruction 'Select services for the project.' Below this is a table with two columns: 'Service' and 'Status'.

Service	Status
 Google Cloud Messaging for Android 	<input checked="" type="checkbox"/> ON
 Google Maps Android API v2 	<input checked="" type="checkbox"/> ON

Criar uma chave

- ✓ Menu > **API Access**.
- ✓ Clicar no botão **Create new Server key**



The screenshot shows the Google APIs console interface. On the left, a sidebar menu is visible with the following items: Overview, Services, Team, API Access (highlighted), Reports, and Quotas. The main content area is titled "API Access" and contains the following text: "To prevent abuse, Google places limits on API requests. Using a valid OAuth token or API key." Below this, there is a section titled "Authorized API Access" with a text box explaining that OAuth 2.0 allows users to share specific data while keeping their usernames, passwords, and other information private. A "Learn more" link is provided. To the right of this text is a circular icon with a stylized 'A'. Below the text box is a large blue button labeled "Create an OAuth 2.0 client ID...". Further down, there is a section titled "Simple API Access" with the text: "Use API keys to identify your project when you do not need to access user data. Learn more". At the bottom of this section are two buttons: "Create new Server key..." and "Create new Browser key...".

Criar uma chave

- ✓ Ao criar a chave, deixe o campo vazio.
- ✓ Apenas clique em **Create**.

Configure Server Key for My Project

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](#)

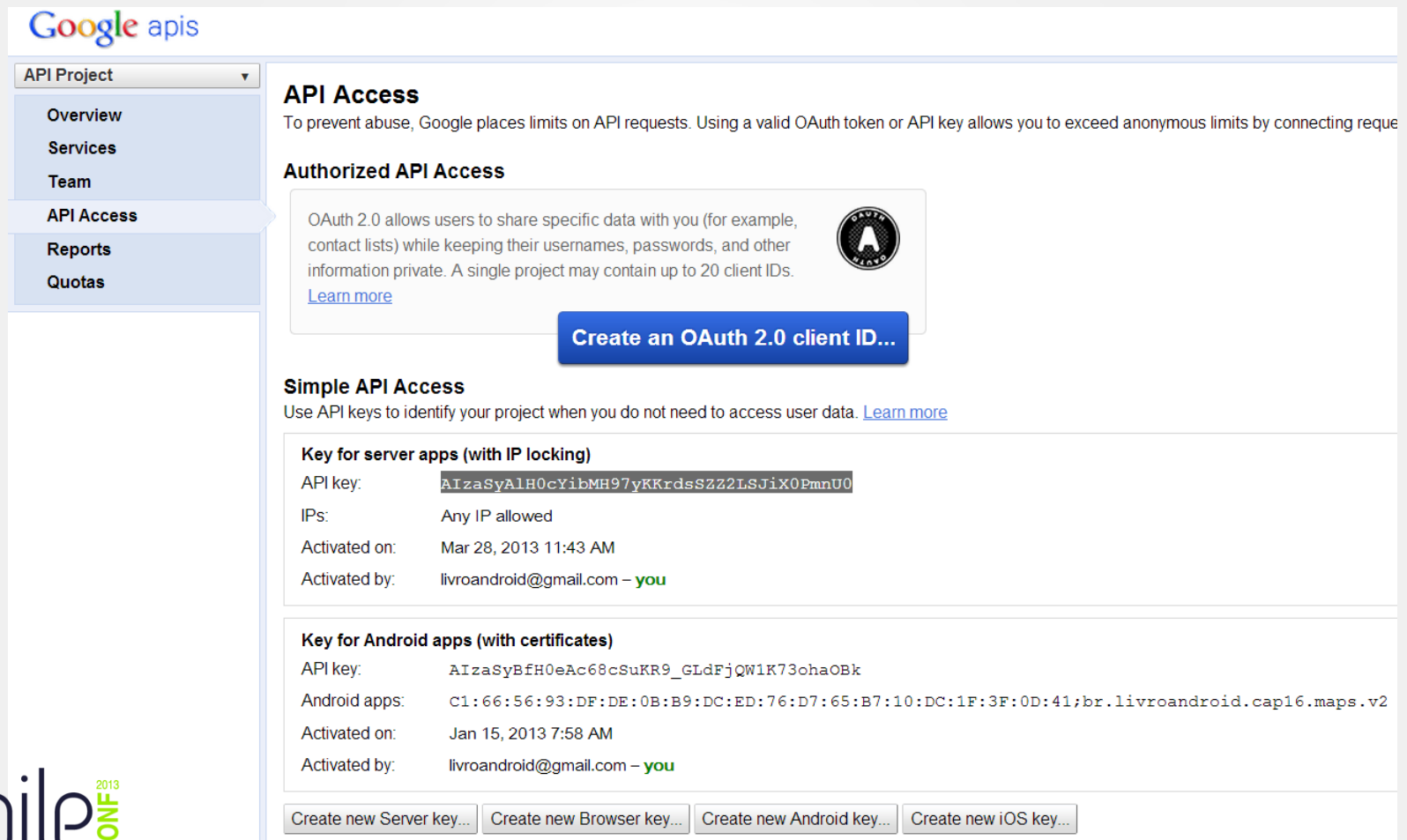
Accept requests from these server IP addresses:

Example: 192.168.12.0/23. One IP address or subnet per line.

CreateCancel

Chave “API Key” criada

- ✓ Copie a API Key. Ela é utilizada no servidor para enviar as mensagens.



The screenshot shows the Google APIs console interface. On the left is a sidebar with a menu: API Project (dropdown), Overview, Services, Team, API Access (selected), Reports, and Quotas. The main content area is titled 'API Access' and includes a sub-section 'Authorized API Access' with a description of OAuth 2.0 and a 'Create an OAuth 2.0 client ID...' button. Below this is 'Simple API Access' with a description and a 'Learn more' link. Two API keys are listed: 'Key for server apps (with IP locking)' and 'Key for Android apps (with certificates)'. Each key entry shows the API key, IP restrictions, activation date, and the user who activated it. At the bottom, there are four buttons: 'Create new Server key...', 'Create new Browser key...', 'Create new Android key...', and 'Create new iOS key...'.

Google apis

API Project ▾

- Overview
- Services
- Team
- API Access**
- Reports
- Quotas

API Access

To prevent abuse, Google places limits on API requests. Using a valid OAuth token or API key allows you to exceed anonymous limits by connecting requests to your project.

Authorized API Access

OAuth 2.0 allows users to share specific data with you (for example, contact lists) while keeping their usernames, passwords, and other information private. A single project may contain up to 20 client IDs. [Learn more](#)

Create an OAuth 2.0 client ID...

Simple API Access

Use API keys to identify your project when you do not need to access user data. [Learn more](#)

Key for server apps (with IP locking)

API key: `AIzaSyAlH0cYibMH97yKKrdsSZZ2LSJiX0PmnU0`

IPs: Any IP allowed

Activated on: Mar 28, 2013 11:43 AM

Activated by: livroandroid@gmail.com – you

Key for Android apps (with certificates)

API key: `AIzaSyBfH0eAc68cSuKR9_GLdFjQW1K73ohaOBk`

Android apps: `C1:66:56:93:DF:DE:0B:B9:DC:ED:76:D7:65:B7:10:DC:1F:3F:0D:41;br.livroandroid.cap16.maps.v2`

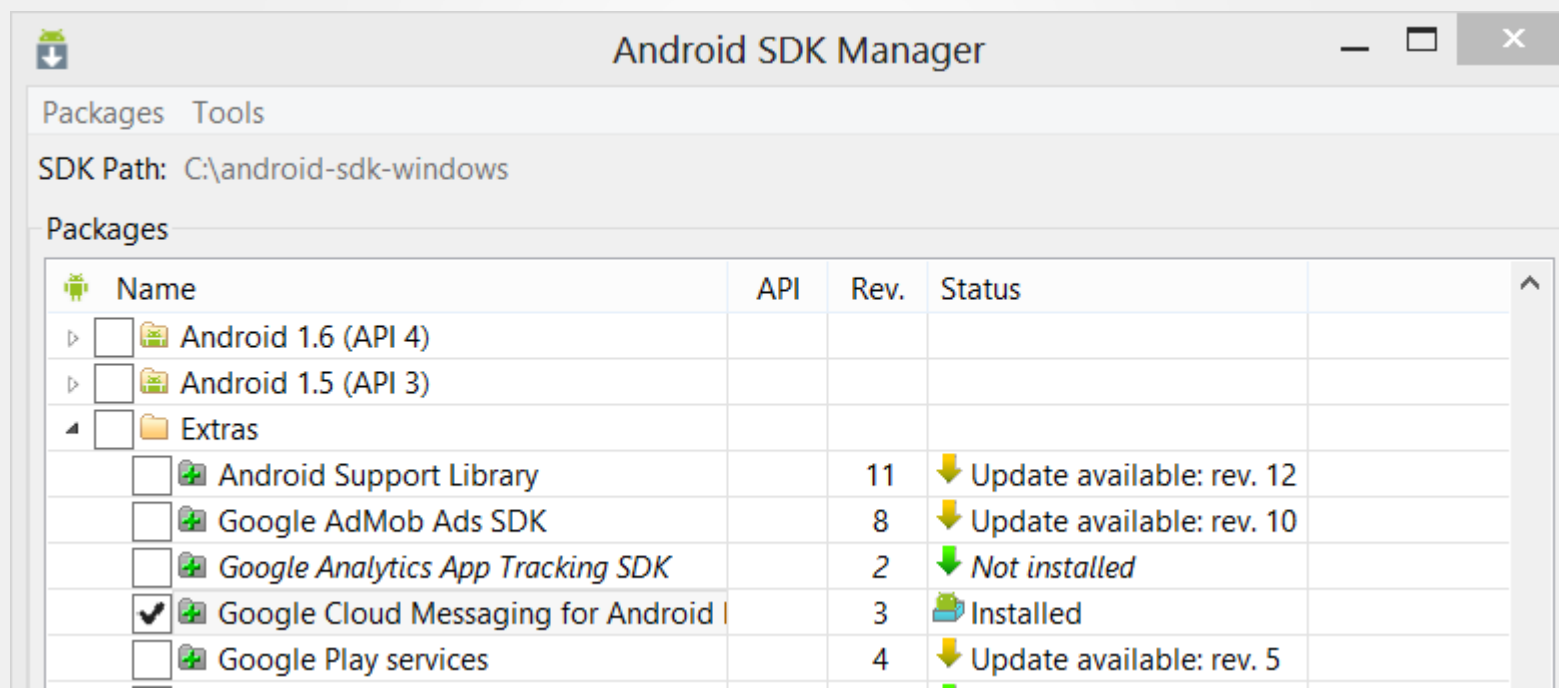
Activated on: Jan 15, 2013 7:58 AM

Activated by: livroandroid@gmail.com – you

[Create new Server key...](#) [Create new Browser key...](#) [Create new Android key...](#) [Create new iOS key...](#)

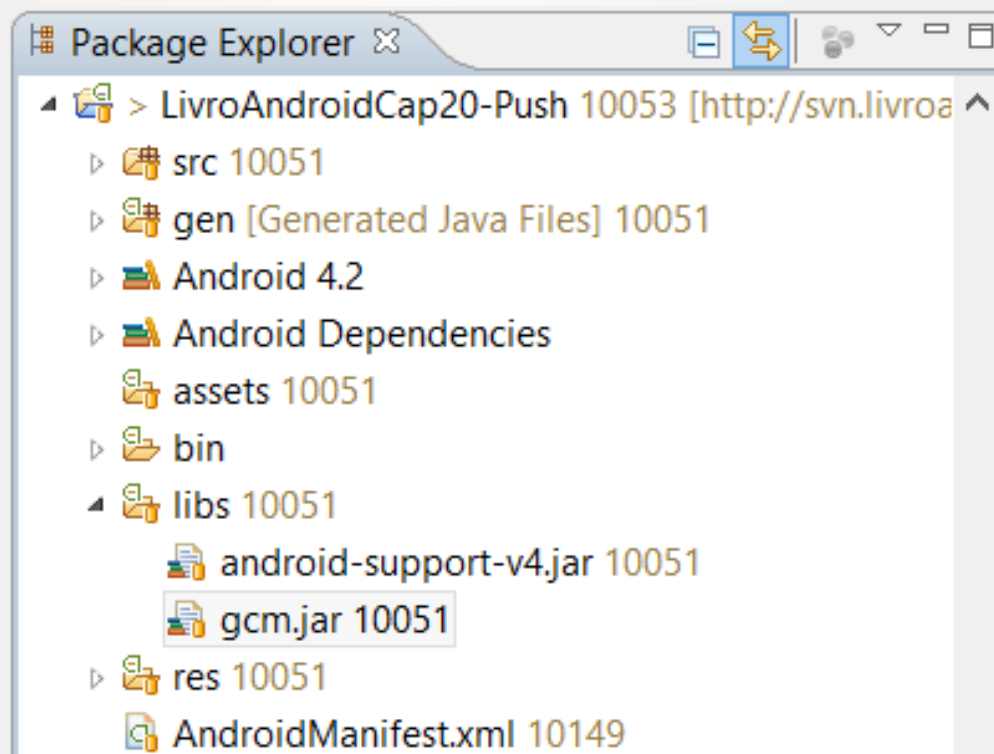
Baixando a biblioteca '.jar'

- ✓ Instalando as Bibliotecas.
- ✓ SDK Manager: Extras > Google Cloud Messaging for Android Library



Projeto Android

- ✓ C:\android-sdk-windows\extras\google\gcm\samples\gcm-demo-client\libs\gcm.jar.
- ✓ Copie o **gcm.jar** para a pasta /libs do projeto



AndroidManifest.xml

- ✓ Android 2.2 (API Level 8) ou superior

```
<uses-sdk android:minSdkVersion="8" android:targetSdkVersion="xx"/>
```

AndroidManifest.xml

- ✓ Adicionar as permissões
- ✓ Cuidado, o nome da permissão é: **nome do pacote + permissão.**

```
<uses-sdk android:minSdkVersion="8" android:targetSdkVersion="17" />

<!-- Para verificar se alguma activity está executando -->
<uses-permission android:name="android.permission.GET_TASKS" />
<!-- O GCM precisa de internet -->
<uses-permission android:name="android.permission.INTERNET" />
<!-- O GCM precisa se conectar a uma conta do Google. -->
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<!-- Permissão utilizada para travar a tela, e evitar o modo de espera. -->
<uses-permission android:name="android.permission.WAKE_LOCK" />
<!--
    Permissão customizada necessária para receber as mensagens.
    Ela precisa ser chamada PACOTE.permission.C2D_MESSAGE,
-->
<permission
    android:name="br.livro.android.cap20.push.permission.C2D_MESSAGE"
    android:protectionLevel="signature" />
<uses-permission android:name="br.livro.android.cap20.push.permission.C2D_MESSAGE" />

<!-- Declara a permissão para se registrar no GCM e receber mensagens -->
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
```

AndroidManifest.xml

- ✓ Configurar o **GCMBroadcastReceiver** que vai **receber** a mensagem.
- ✓ Configurar o **GCMIntentService** que vai **processar** a mensagem.

```
<!--  
    BroadcastReceiver para receber as mensagens do GCM, por meio de Intents.  
-->  
<receiver  
    android:name="com.google.android.gcm.GCMBroadcastReceiver"  
    android:permission="com.google.android.c2dm.permission.SEND" >  
    <intent-filter>  
  
        <!-- Filtrar as ações para receber mensagens. -->  
        <action android:name="com.google.android.c2dm.intent.RECEIVE" />  
        <!-- Filtrar a ação para receber o registration id do aparelho. -->  
        <action android:name="com.google.android.c2dm.intent.REGISTRATION" />  
  
        <category android:name="br.livro.android.cap20.push" />  
    </intent-filter>  
</receiver>  
<!--  
    Service chamado automaticamente pelo receiver acima.  
    Deve conter o código para ler as mensagens.  
-->  
<service android:name=".GCMIntentService" />
```

Registro no GCM

✓ Classe **GCMRegistrar** está no **gcm.jar**

```
public static void registrar(Context context, String projectId) {  
    GCMRegistrar.checkDevice(context);  
    GCMRegistrar.checkManifest(context);  
    final String regId = getRegistrationId(context);  
    if (regId.equals("")) {  
        GCMRegistrar.register(context, projectId);  
        Log.i(TAG, "GCM registrado com sucesso.");  
    } else {  
        Log.i(TAG, "GCM já está registrado, ID: " + regId);  
    }  
}
```

GCMIntentService

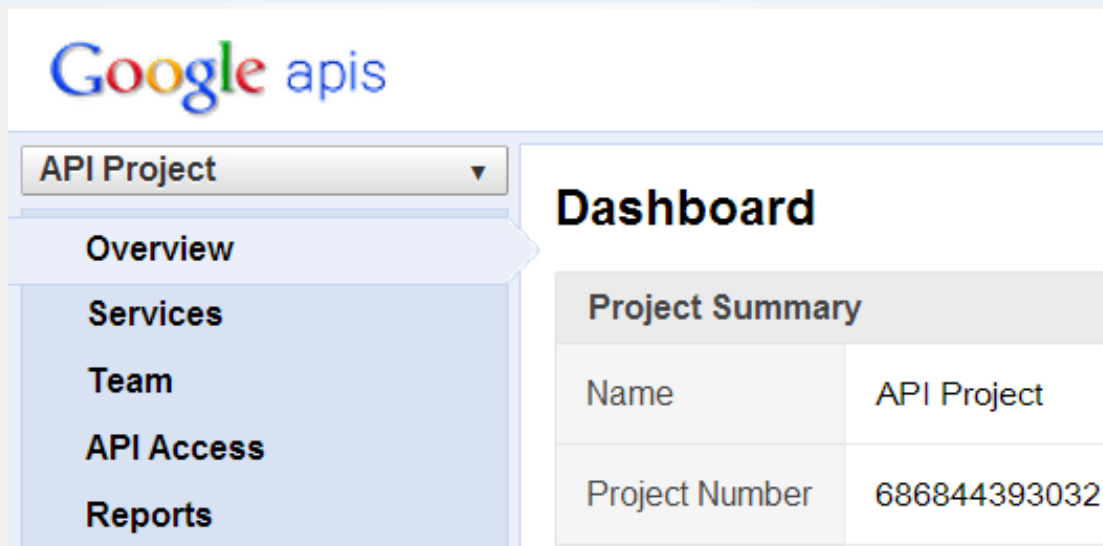
- ✓ Deve herdar de GCMBaseIntentService
- ✓ Precisa passar no seu construtor o **Project Number** (obtido no Google APIs console)

```
public class GCMIntentService extends GCMBaseIntentService {  
  
    public GCMIntentService() {  
        super(Constants.PROJECT_NUMBER);  
    }  
  
    @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 msg = intent.getStringExtra("msg");  
    }  
  
    @Override  
    public void onError(Context context, String errorId) {  
    }  
}
```

Constants

- ✓ **Project Number** (obtido no Google APIs console)

```
public interface Constants {  
  
    /**  
     * "Project Number" registrado no console do Google  
     *  
     * https://code.google.com/apis/console/#project:686844393032:services  
     */  
    String PROJECT_NUMBER = "686844393032";  
}
```

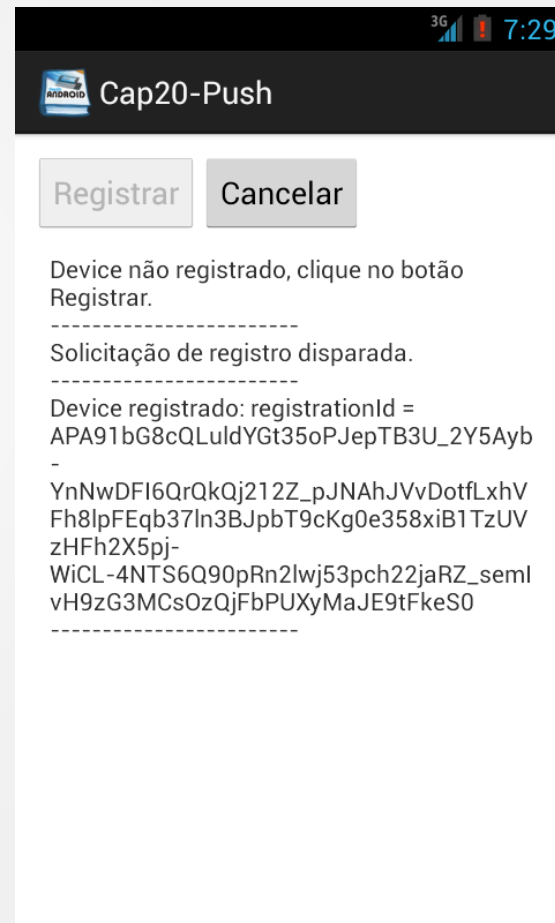
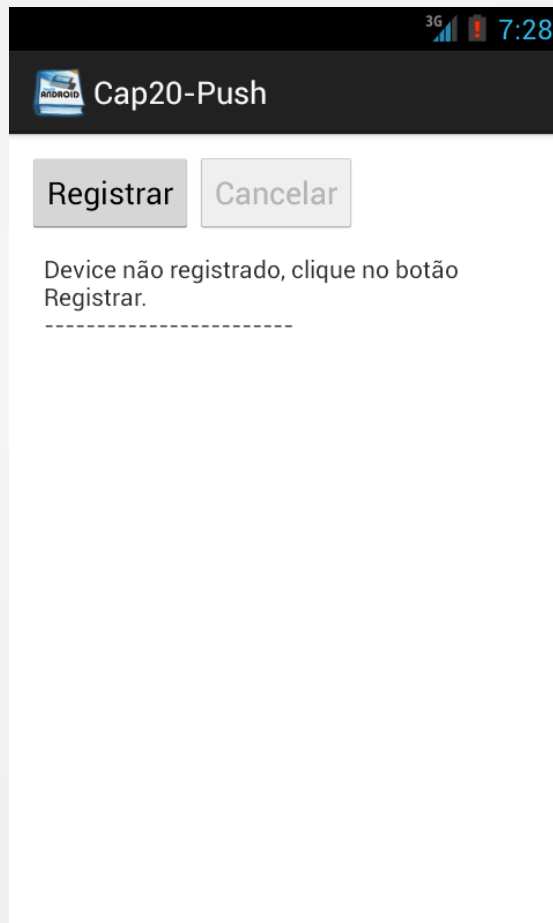


The screenshot shows the Google APIs console dashboard. On the left, there is a sidebar with a dropdown menu labeled "API Project" and a list of navigation items: Overview, Services, Team, API Access, and Reports. The "Overview" item is currently selected. The main content area is titled "Dashboard" and contains a "Project Summary" table.

Project Summary	
Name	API Project
Project Number	686844393032

Demo

✓ Registro no GCM.



Servidor: Enviando uma mensagem

- ✓ POST
- ✓ <https://android.googleapis.com/gcm/send>

Servidor: Enviando uma mensagem

- ✓ Como enviar uma mensagem do servidor para o device
- ✓ Utiliza as classes Sender e Message do **gcm-server.jar**

```
import com.google.android.gcm.server.Message;
import com.google.android.gcm.server.Result;
import com.google.android.gcm.server.Sender;

/**
 * @author Ricardo Lecheta
 */
public class EnviarMensagemParaDevice {

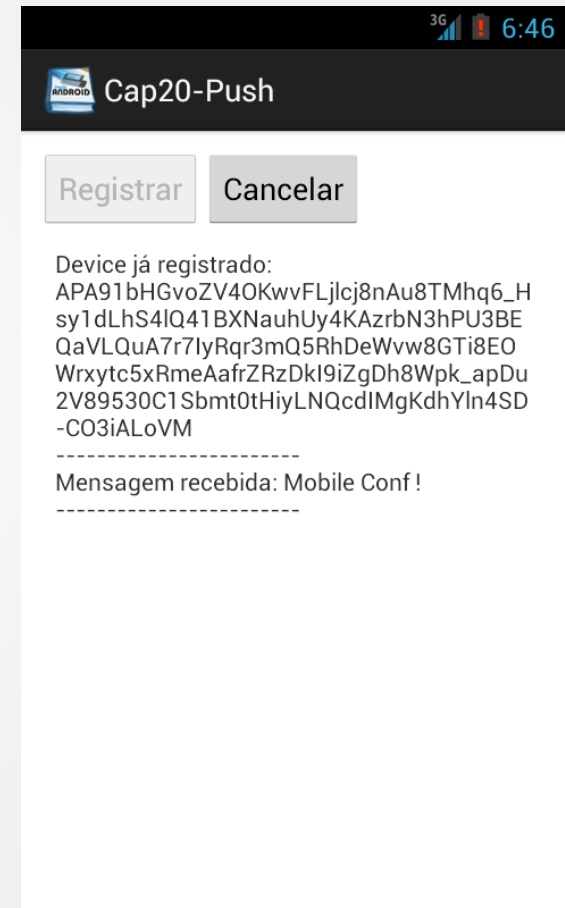
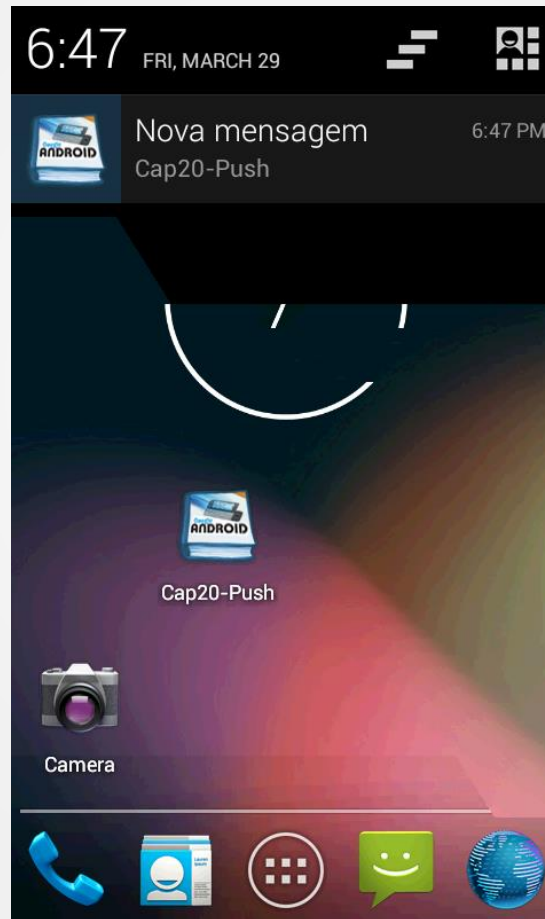
    // Chave de registro do device. Resultado da chamada GCMRegistrar.register(...);
    private static final String DEVICE_REGISTRATION_ID = "APA91bGefh7maSK-nBVlIsskenNA0nVc1X9R

    // Chave criada no Console. Menu > API Access > (create new server key)
    private static final String API_KEY = "AIzaSyAlH0cYibMH97yKKrdsSZZ2LSJiX0PmnU0";

    public static void main(String[] args) throws IOException {
        Sender sender = new Sender(API_KEY);
        Message message = new Message.Builder().addData("msg", "Mobile Conf !").build();
        Result result = sender.send(message, DEVICE_REGISTRATION_ID, 5);
        System.out.println(result);
    }
}
```

Recebendo mensagem

✓ Demo



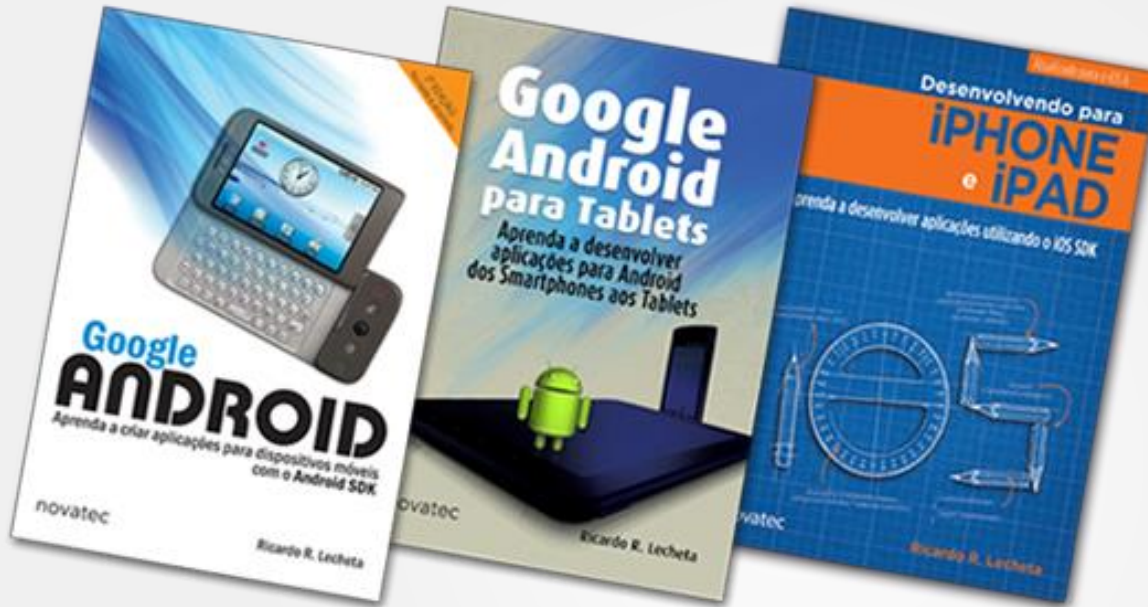
Considerações finais

- ✓ O servidor do Google pode enviar periodicamente a chave de registro de um device, deixe o código preparado para atualizar esta chave.
- ✓ Cada mensagem deve conter no máximo 4096 bytes
- ✓ Uma mensagem fica armazenada no servidor do GCM por 4 semanas
- ✓ Até 100 mensagens podem ser enfileiradas, depois as mensagens antigas são apagadas.
- ✓ Se não for necessário enfileirar as mensagens, ou seja entregar todas as mensagens, utilize o parâmetro **collapse_key** para agrupá-las. Mensagens com o mesmo **collapse_key** serão descartadas, de forma que sempre a última será entregue.

Exemplo: gcm-demo-client

- ✓ Passo a passo para criar o projeto de exemplo.
- ✓ Wizard: > **File New** > **Android Sample Project** > **gcm-demo-client**.
- ✓ Na classe CommonUtilities, insira o valor da constante *SENDER_ID*, que é o Project Number do console de administração;
- ✓ Para não utilizar o servidor, comente o método `post()` da classe ServerUtilities;
- ✓ Comente a verificação na DemoActivity: `checkNotNull(SERVER_URL, "SERVER_URL");`
- ✓ Na DemoActivity, altere o context `this` para `getApplicationContext();`
- ✓ Na classe GCMIntentService, customize o método `onMessage()` par ler os dados conforme sua necessidade;
- ✓ Envie uma mensagem pelo servidor (slide 23).

Obrigado, dúvidas ?



Código de Desconto

ANDROID3

20 %



facebook.com/ricardolecheta

