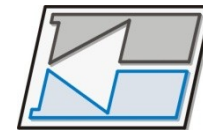


Introdução à Programação Android

Prof. Samir Bonho

Aula 11

Florianópolis, 14 de Maio de 2014



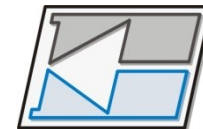
Sumário

✓ API Google Maps

✓ Fontes:

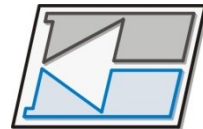
✓ Universidade de Bologna – Depto de Informática

✓ Android Developers. Disponível em <http://developer.android.com/>



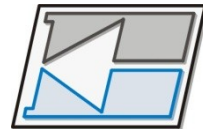
API Google Maps





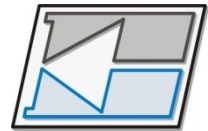
API Google Maps

- O Android possui extensões que possibilitam a criação de ambientes de desenvolvimento usando bibliotecas externas específicas, como por exemplo o Google APIs, ou componentes extras.
- As APIs do Google possuem acesso aos serviços e dados do Google. Essas APIs fazem parte de cada versão do SDK, ou seja, cada SDK de versão é também possível obter um SDK de APIs do Google para Android.



API Google Maps

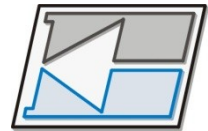
- Um recurso importante dessas APIs é a biblioteca externa Maps, que fornece uma API para que aplicativos Android possam ter acesso ao Google Maps, permitindo a adição de recursos avançados de mapeamento, onde pode-se exibir um ou mais locais em um mapa para o usuário.



API Google Maps

Chave para utilização

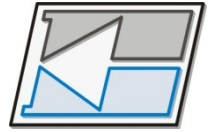
- Para poder acessar o Google Maps, de dentro da sua aplicação, você precisa de uma chave de registro na API do Google Maps, ela é gratuita e fácil de ser conseguida, porem ela é especifica para a chave de assinatura do desenvolvedor.
- Ao se executar uma aplicação no emulador do Android, o eclipse compila seu projeto e assina o .apk com uma chave de debug e envia para a instalação no emulador



API Google Maps

Chave para utilização: pra que isto?

- Sua aplicação deve poder dizer, “Ei, sou eu, a aplicação XYZ, quem está acessando o Google Maps!”
- Assim o Google evita que uma mesma aplicação acesse os servidores de modo descontrolado sobrecarregando os servidores, já que, sua chave de API tem uma quota de acessos diários.

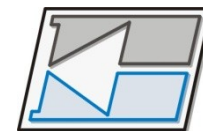


API Google Maps

Chave para utilização: resumo

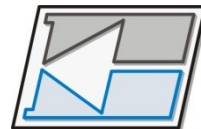
■ Resumindo:

1. O eclipse tem uma chave de debug de aplicações Android;
2. Você utiliza ela para gerar uma chave de acesso a API do Google Maps;
3. Quando for distribuir uma aplicação você cria uma chave de execução própria;
4. Registra a sua chave para gerar a chave de acesso a API do Google Maps;



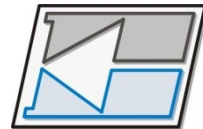
Chave de desenvolvimento

- O Google Maps API v2 Android usa um novo sistema de gerenciamento de chaves. Chaves existentes de um aplicativo Android Google Maps v1 não vão funcionar com a API v2.
- Obter uma chave para a sua aplicação requer várias etapas. Estes passos são descritos a seguir.



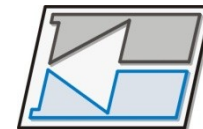
Chave de desenvolvimento

- Recuperar informações sobre o certificado de sua aplicação (**SHA-1 fingerprint**).
- A chave do Google Maps API é baseada em um pequeno formulário de certificado digital do seu aplicativo, conhecido como **SHA-1 fingerprint**.



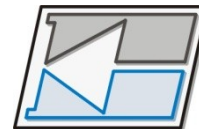
Chave de desenvolvimento

- Por padrão, ele é armazenado no mesmo diretório do seu dispositivo AVD:
- Usando o Eclipse:
selecione **Window > Preferences> Android > Build** para verificar o caminho completo (**Default debug keystore**);



Chave de desenvolvimento

- Usando o Terminal do windows (cmd):
 - Navegue até a pasta bin do jdk/jre da sua máquina para ter acesso aos comandos ***Keytool***

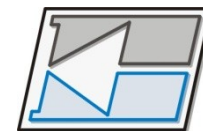


Chave de desenvolvimento

- Ex.: C:\Program Files\Java\jre7\bin\
 - Execute a seguinte linha de comando substituindo o caminho do arquivo **debug.keystore** pelo seu visualizado anteriormente:

```
keytool -list -v -keystore
```

```
"C:\Users\Samir\.android\debug.keystore" -alias  
androiddebugkey -storepass android -keypass android
```



Chave de desenvolvimento

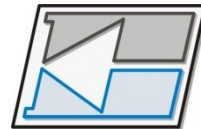
- **SHA-1 fingerprint**

Alias name: androiddebugkey Creation date: Jan 01, 2013 Entry type: PrivateKeyEntry Certificate chain length: 1 Certificate[1]: Owner: CN=Android Debug, O=Android, C=US Issuer: CN=Android Debug, O=Android, C=US Serial number: 4aa9b300 Valid from: Mon Jan 01 08:04:04 UTC 2013 until: Mon Jan 01 18:04:04 PST 2033 Certificate fingerprints:

MD5: AE:9F:95:D0:A6:86:89:BC:A8:70:BA:34:FF:6A:AC:F9

SHA1:BB:0D:AC:74:D3:21:E1:43:07:71:9B:62:90:AF:A1:66:6E:44:5D:75

Signature algorithm name: SHA1withRSA Version: 3

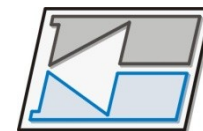


Chave de desenvolvimento

- Acesse o site da API do google

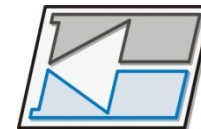
<https://code.google.com/apis/console/>

Logado com uma conta;



Chave de desenvolvimento

- No lado esquerdo da página, temos um combo com os nossos projetos. Ao selecionar um deles podemos clicar em *Services*. Entre as dezenas de serviços temos o *Google Maps Android API v2*, que deve ser mudado para o estado *on*.
- O próximo passo é clicar no link *API Access*, também no lado esquerdo. Na extremidade inferior desta página temos um link “*Create new android key*”. Na caixa de texto apresentada deve-se colocar o SHA1 de seu certificado, seguido do nome do pacote da sua aplicação Android que deverá interagir com os serviços Google.



Chave de desenvolvimento

Google apis

API Project ▼

Overview

Services

Team

API Access

Reports

Quotas



Dashboard

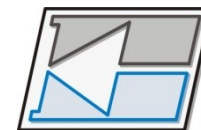
Project Summary

Label	API Project
Project ID	460786001904
Project Name	Register...
Google+ Page	Request connection
Owners	samir.bonho@gmail.com - you

Service

Status

 Google Maps Android API v2	 No known issues
--	---



Chave de desenvolvimento



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.

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 Android apps (with certificates)

API key: AIzaSyAYtBuNGQyUZgfgtjL5-KoVko5t9INGZiQ

Android apps: BD:F2:7D:13:8F:01:31:96:D3:55:DC:E2:E3:7F:92:5A:FD:D6:FC:52;com.example.gps2

Activated on: May 8, 2014 7:27 AM

Activated by: samir.bonho@gmail.com – you

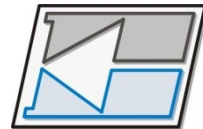
Key for Android apps (with certificates)

API key: AIzaSyCKVdGxb8IQguY2uX3TgAgMVjfaDG8efuE

Android apps: BD:F2:7D:13:8F:01:31:96:D3:55:DC:E2:E3:7F:92:5A:FD:D6:FC:52;com.example.gps2

Activated on: May 7, 2014 8:37 AM

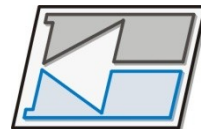
Activated by: samir.bonho@gmail.com – you



Chave de desenvolvimento

- Com o SHA1 em mãos, volte ao Google APIs Console, e copie este valor lá, seguido de um ponto e vírgula e, finalmente, o nome do pacote de seu aplicativo.
- `CC:0D:05:90:C2:66:12:61:F9:34:A9:46:60:76:BA:F2:26:BB:C4:1B;c`
`om.example.mapa_simples`
- Sua chave então será gerada:

`AlzaSyDm4C6Pv6qsSuxMUMTdlui3KKX5r82BMlk`



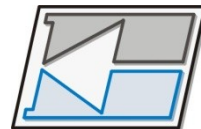
Adicionando a chave

- O passo final é adicionar a chave da API para a sua aplicação.
- No **AndroidManifest.xml**, adicione dentro de `<application>`, inserindo antes da tag de fechamento `</ application>`:

`<meta-data`

`android:name="com.google.android.maps.
v2.API_KEY"`

`android:value="your_api_key"/>`



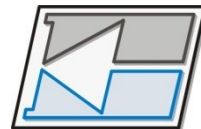
Adicionando a chave

- O passo final é adicionar a chave da API para a sua aplicação.
- No **AndroidManifest.xml**, adicione dentro de `<application>`, inserindo antes da tag de fechamento `</ application>`:

`<meta-data`

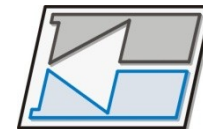
`android:name="com.google.android.maps.
v2.API_KEY"`

`android:value="your_api_key"/>`



Google MAPs v2

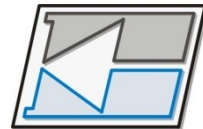
- **Integra** um mapa à uma aplicação Android
- **Gerencia** o zoom
- **Adiciona** informação ao mapa
- **Gerencia** eventos gerados pelo usuário



Permissões

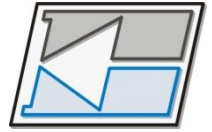
Adicionar no AndroidManifest.xml

- Internet Access
- Localization capabilities
- Acesso ao Google Web services
- OpenGL ES version 2 libraries
- Access to network state



Permissões

```
<uses-permission  
android:name="android.permission.INTERNET"/>  
<uses-permission  
android:name="android.permission.ACCESS_NETWORK_STA  
TE"/>  
<uses-permission  
android:name="android.permission.WRITE_EXTERNAL_STO  
RAGE"/>  
<uses-permission  
android:name="com.google.android.providers.gsf.perm  
ission.READ_GSERVICES"/>  
<uses-permission  
android:name="android.permission.ACCESS_COARSE_LOCA  
TION"/>  
<uses-permission  
android:name="android.permission.ACCESS_FINE_LOCATI  
ON"/>
```

Otimizações no mapa

- Map type

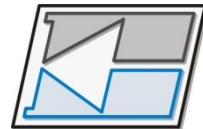
```
nMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
```

Normal → Mapa típico.

Hybrid → Imagem de satélite com as vias incluídas.

Satellite → Imagem de satélite

Terrain → Imagem topográfica.

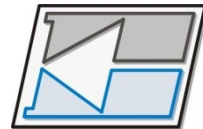


Otimizações no mapa

- **LatLng:** pontos no mapa

```
private static final LatLng BOLOGNA_POINT = new  
LatLng(44.496781,11.356387);
```

```
private static final LatLng FLORENCE_POINT = new  
LatLng(43.771373,11.248069);
```



Otimizações no mapa

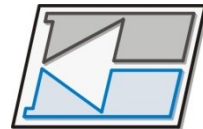
Eventos no Google Map.

CLICK events →

Implementar a interface `OnMapClickListener` e o método `OnMapLongClickListener`.

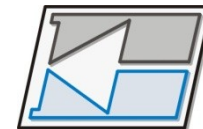
CAMERA events →

Implementar a interface `OnCameraChangeListener` e o método `onCameraChange(CameraPosition)`.



Otimizações no mapa

```
public class MainActivity extends Activity implements  
OnMapClickListener {  
  
    private GoogleMap mMap;  
  
    protected void onCreate(Bundle savedInstanceState) {  
        ...  
        mMap.setOnMapClickListener(this);  
        ...  
    }  
  
    public void onMapClick(LatLng position) {  
        // Handle the click events here ...  
    }
```



Otimizações no mapa

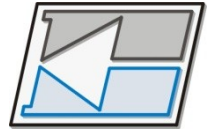
- **Camera**

Location → coordenadas latitude/longitude.

Zoom → escala do mapa.

Bearing → define a orientação do mapa (em graus)

Tilt → ângulo de visualização

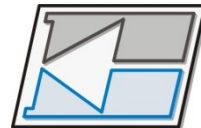


Otimizações no mapa

- As propriedades da Camera podem ser setadas ao mesmo tempo através do objeto CameraPosition..

```
private static final LatLng BOLOGNA = new  
    LatLng(44.496781, 11.356387);
```

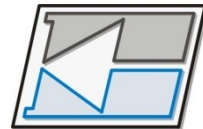
```
CameraPosition cameraPosition = new CameraPosition.  
    Builder()  
        .target(BOLOGNA)  
        .zoom(18)  
        .bearing(90)  
        .tilt(30)  
        .build();
```



Otimizações no mapa

- **Markers**

- Icon:** ícones para identificar pontos
- Position:** posição do marcador no mapa
- Title:** título do marcador
- Events:** eventos gerados

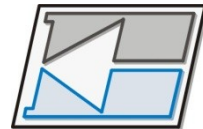


Otimizações no mapa

```
private static final LatLng BOLOGNA = new  
LatLng(44.496781,11.356387);
```

```
Marker bologna =  
myMap.addMarker(newMarkerOptions().position(BOLOGNA));
```

```
Marker bologna= mMap.addMarker(new MarkerOptions()  
    .position(Bologna)  
    .title(« Centro de Bologna")  
    .snippet("Visite a cidade!"));
```

Otimizações no mapa

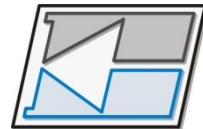
- **Markers**

EVENTOS associados ao Marker:

ClickEvents → implementar a interface `OnMarkerClickListener` e o método `onMarkerClick(Marker)`.

DragEvents → implementar a interface `OnMarkerDragListener` e o método `onMarkerDragEnd(Marker)`.

InfoWindow Click Events → implementar a interface `OnInfoWindowClickListener` e o método `onInfoWindowClick(Marker)`.



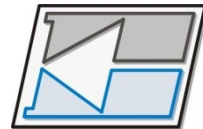
Otimizações no mapa

- **Shapes:** identifica seções no mapa

Polylines → conecta diferentes pontos geográficos (lat,long) através de uma linha.

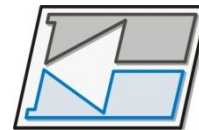
Polygons → define um polígono formado por pontos geográficos.

Circles → forma uma círculo centrado em um ponnto geográfico.



Otimizações no mapa

```
PolygonOptions rectOptions = new PolygonOptions()  
    .add(BOLOGNA_P1)  
    .add(BOLOGNA_P2)  
    .add(BOLOGNA_P3);  
Polygon polyline = mMap.addPolygon(rectOptions);  
  
CircleOptions circleOptions = new CircleOptions()  
    .center(BOLOGNA_P1)  
    .radius(1000)  
    .strokeColor(Color.RED);  
  
Circle circle = mMap.addCircle(circleOptions);
```



Exercícios