

AUTENTICAÇÃO DE USUÁRIOS POR MEIO DE REDES SOCIAIS

TIAGO FRAZÃO VALÉRIO
CAIO CANALLI
GUSTAVO
ALEX STOCCH

POR QUE USAR O LOGIN SOCIAL?

USUÁRIO

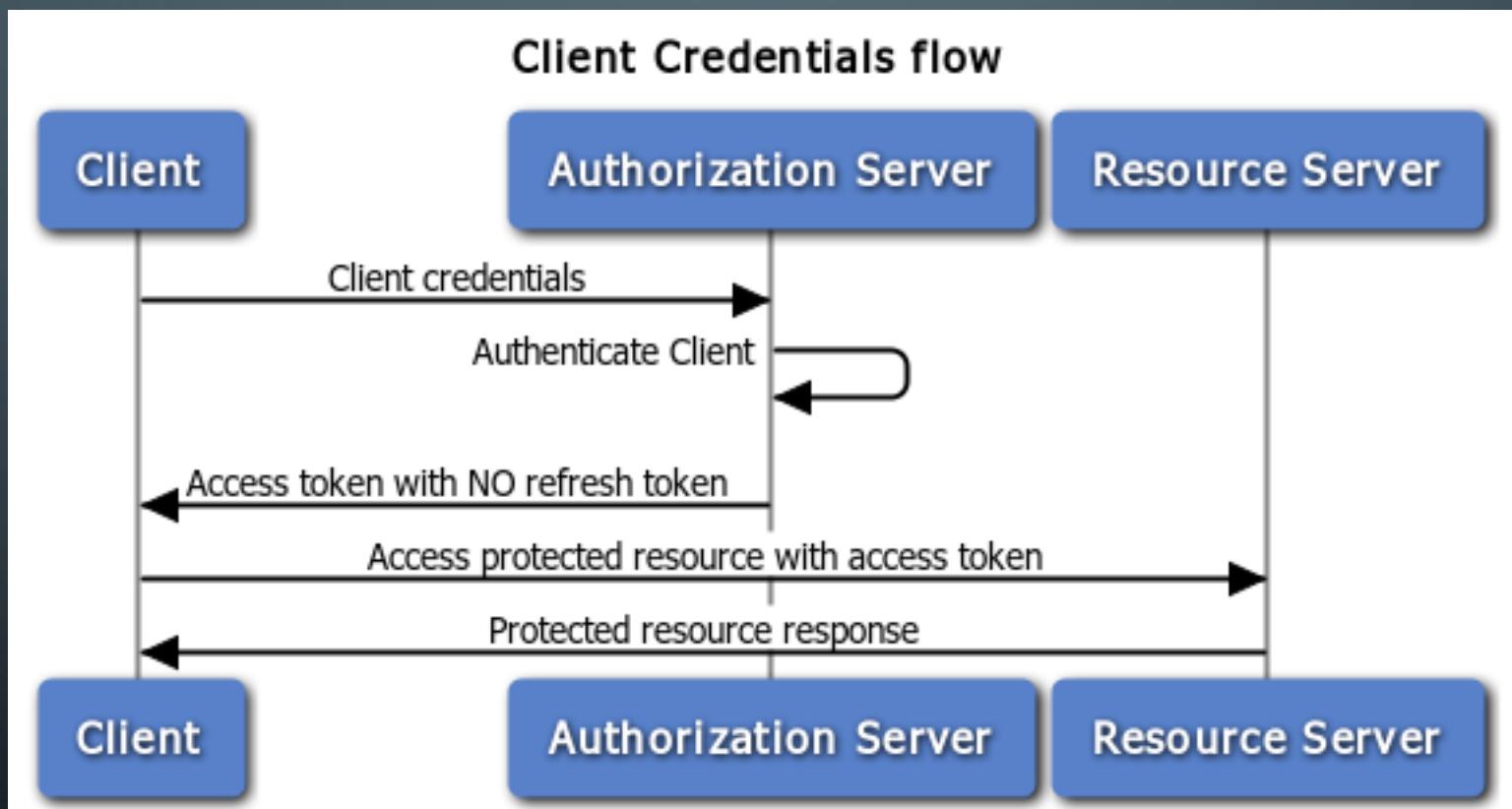
- Acesso rápido
- Elimina a necessidade de preenchimento de formulários
- Utilização de uma única senha

DESENVOLVEDOR

- Aumento na quantidade de cadastros
- Mais veracidade nas informações
- Acesso ao perfil do usuário
- Diminuição da quantidade de cadastros duplicados
- A segurança do login é feita por outra empresa

OAUTH

- Protocolo padrão de autorização



RESUMO DO PASSO-A-PASSO

Criar uma conta na rede social X como desenvolvedor

Cadastrar as informações de seu aplicativo no site

Gerar uma chave que irá vincular as informações passadas no site com o seu aplicativo

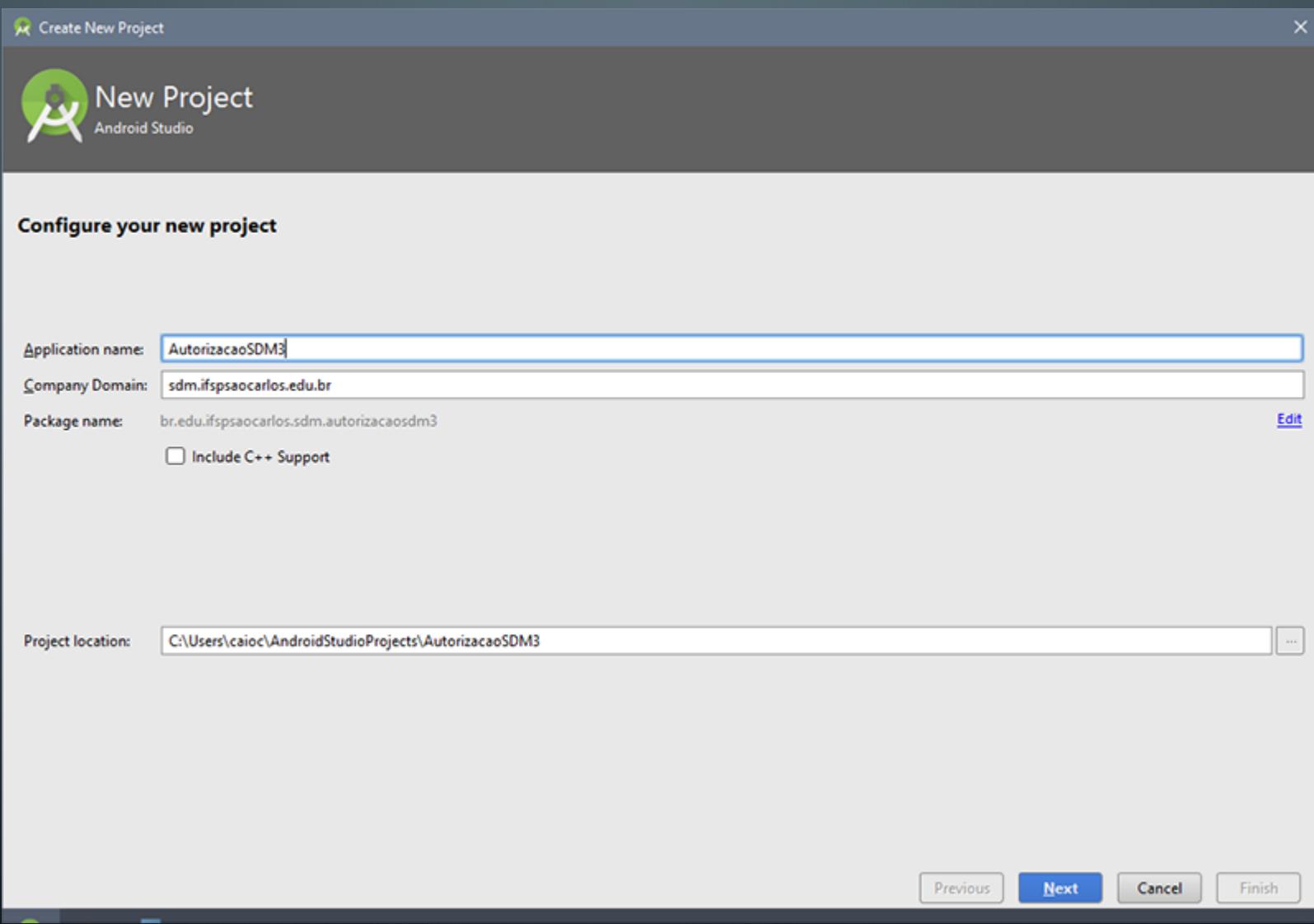
Baixar e incluir em seu projeto o SDK/Framework oferecido pela rede social

Vincular seu projeto com o site

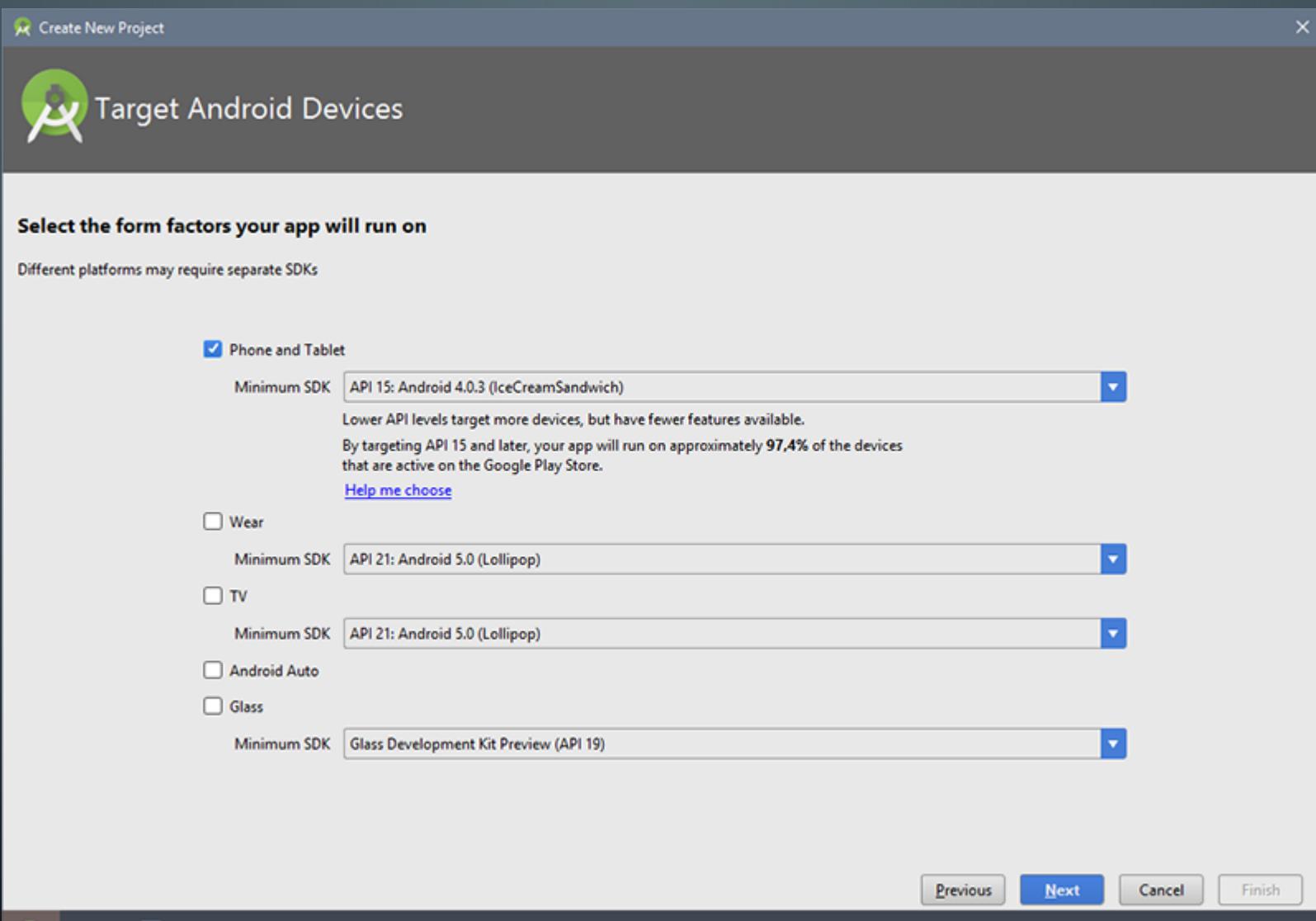
Utilizar a biblioteca para programar a função de login

FACEBOOK LOGIN

Criando o projeto



API Level



Versão mínima do SDK

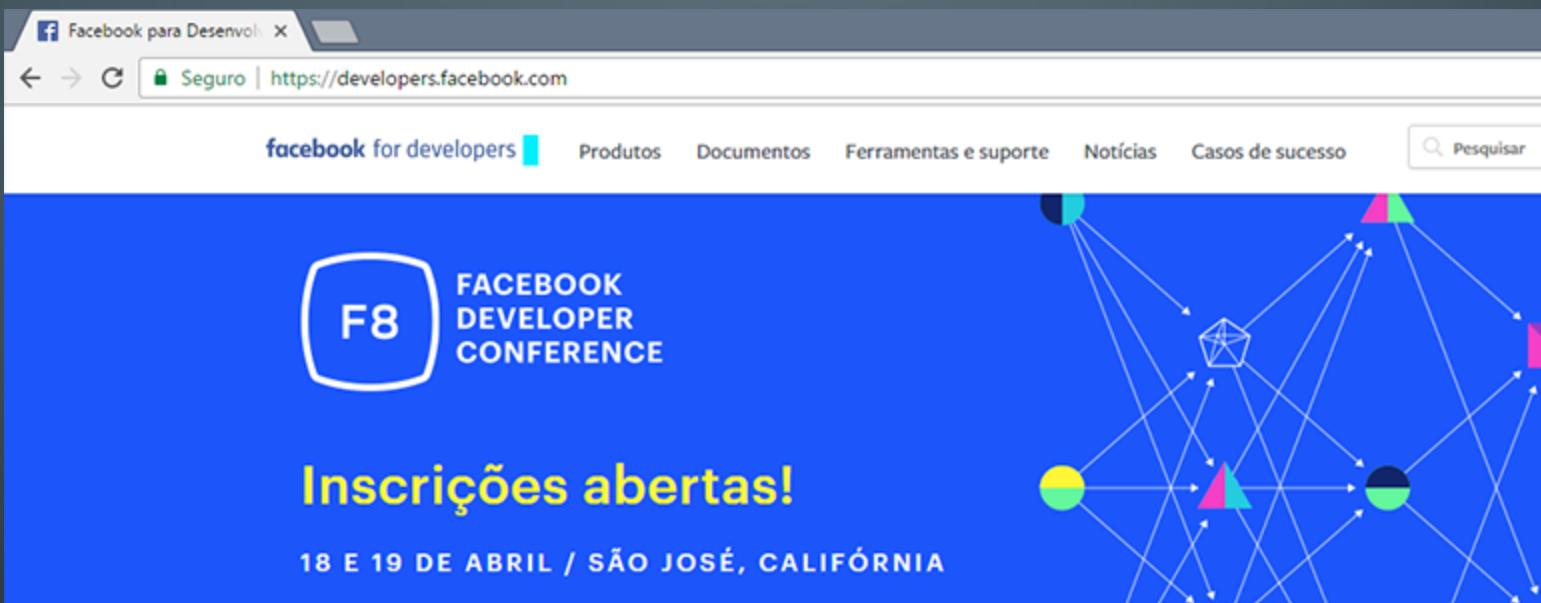
The screenshot shows the Android Studio interface with the project 'AutorizacaoSDM3' open. The left sidebar displays the project structure, including the app module and various Gradle scripts. The main editor window shows the build.gradle file for the app module. A specific line of code, `minSdkVersion 15`, is highlighted with a yellow background, indicating it is the focus of the discussion.

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 25
    buildToolsVersion "25.0.2"
    defaultConfig {
        applicationId "br.edu.ifspsaocarlos.sdm.autorizacaosdm3"
        minSdkVersion 15
        targetSdkVersion 25
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.1.0'
    testCompile 'junit:junit:4.12'
}
```

Facebook for developers



Quickstarts

Inícios rápidos - Facebook X

Seguro | https://developers.facebook.com/quickstarts/

facebook for developers Produtos Documentos Ferramentas e suporte Notícias Casos de sucesso Pesquisar

Adicionar um novo aplicativo

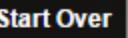
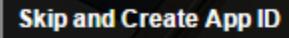
Selecionar uma plataforma para começar

iOS  Android  Canvas do Facebook  Site 

Nome do Aplicativo

facebook for developers  

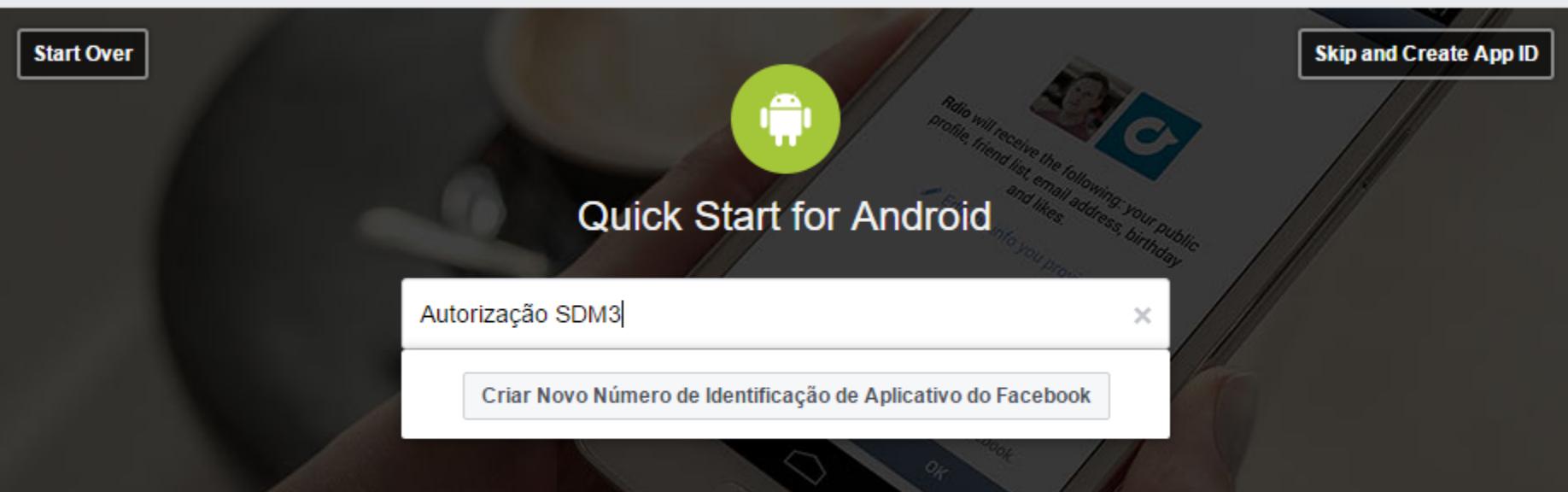
Produtos Documentos Ferramentas e suporte Notícias Casos de sucesso 

 Quick Start for Android

Autorização SDM3 

Criar Novo Número de Identificação de Aplicativo do Facebook



Informações básicas

[Start Over](#) [Skin and Create App ID](#)

Crie um novo número de identificação do aplicativo
Comece integrando o Facebook ao seu aplicativo ou site

Nome de exibição

Email de contato

Categoria

Ao prosseguir, você concorda com as [Políticas da Plataforma do Facebook](#)

[Cancelar](#) [Crie um número de identificação do aplicativo](#)

Importando o SDK

Import SDK

Add SDK

App Info

Key Hashes

App Events

Finished

Add Facebook SDK to Your Project

To use Facebook SDK in a project, add it as a build dependency and import it. If you are starting a new project, follow all the steps below. To add Facebook SDK to an existing project, start with step 3.

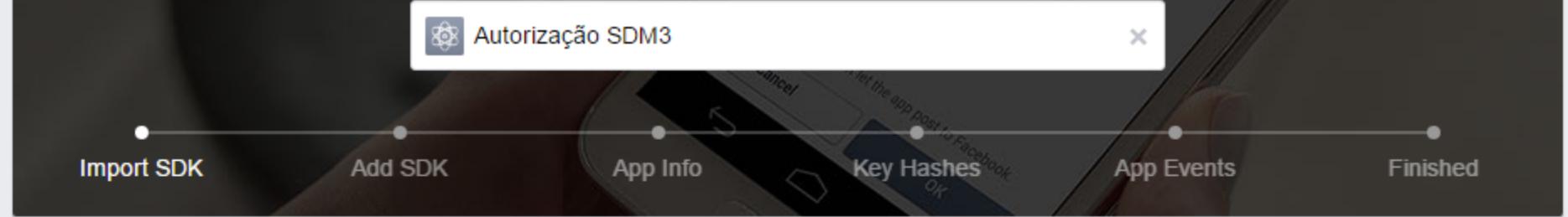
1. Go to [Android Studio | New Project | Minimum SDK](#)
2. Select "API 15: Android 4.0.3" or higher and create your new project.
3. In your project, open
`your_app | Gradle Scripts | build.gradle`
4. Add the Maven Central Repository to `build.gradle` before `dependencies`:

```
repositories {  
    mavenCentral()  
}
```

5. Add `compile 'com.facebook.android:facebook-android-sdk:[4,5)` to your `build.gradle` dependencies.
6. Build your project.
7. Import Facebook SDK into your app:

```
import com.facebook.FacebookSdk;
```

Reforçando

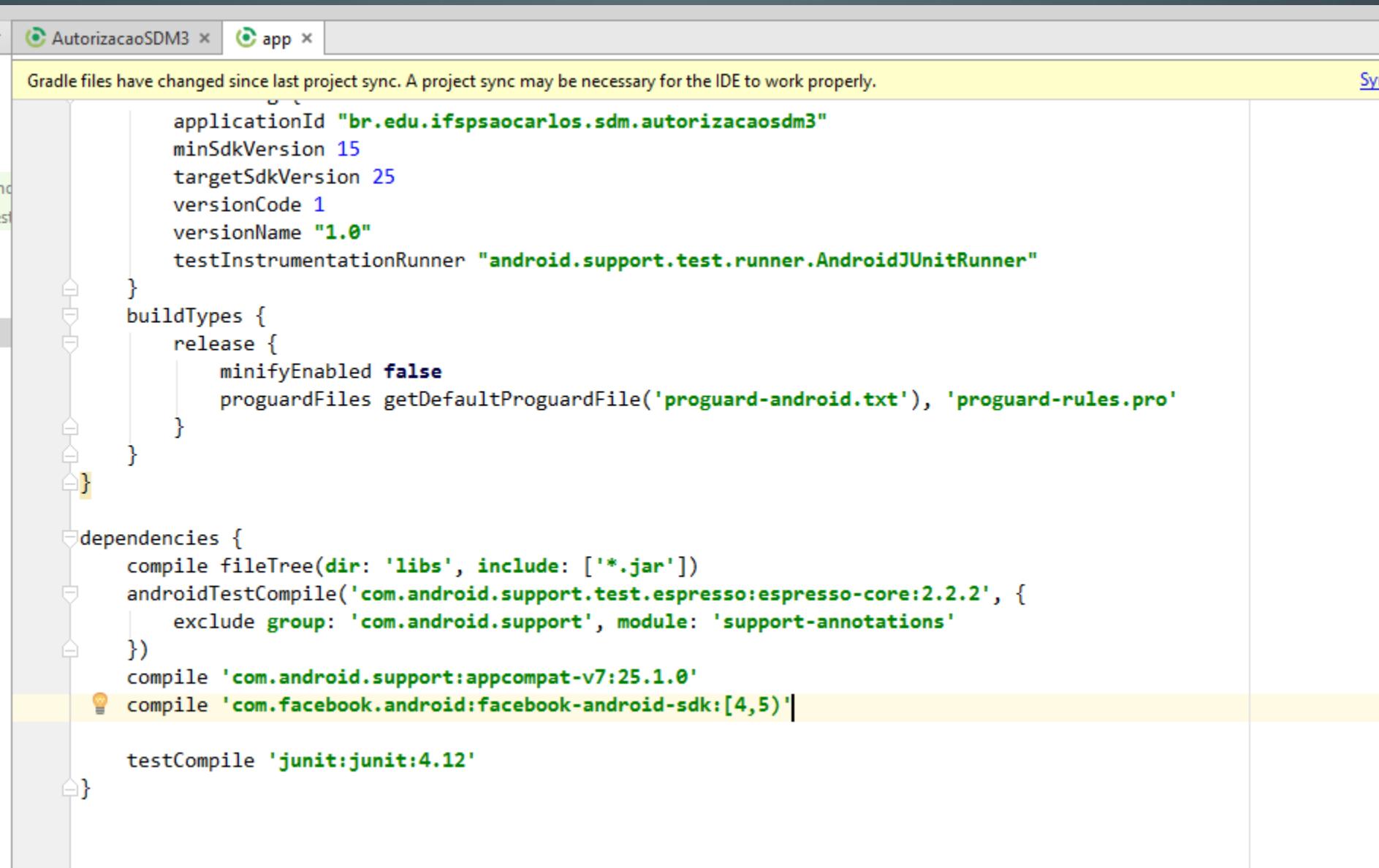


Add Facebook SDK to Your Project

To use Facebook SDK in a project, add it as a build dependency and import it. If you are starting a new project, follow all the steps below. To add Facebook SDK to an existing project, start with step 3.

1. Go to [Android Studio | New Project | Minimum SDK](#)
2. Select "API 15: Android 4.0.3" or higher and create your new project.
3. In your project, open
[your_app | Gradle Scripts | build.gradle](#)
4. Add the Maven Central Repository to [build.gradle](#) before `dependencies`:

Configurando o SDK



The screenshot shows the Android Studio interface with the project navigation bar at the top. The active tab is 'app'. A yellow status bar message reads: 'Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.' Below the status bar is the build.gradle file content.

```
buildscript {
    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.0.0'
        classpath 'com.google.gms:google-services:3.1.1'
    }
}

allprojects {
    repositories {
        google()
        jcenter()
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.1.0'
    compile 'com.facebook.android:facebook-android-sdk:[4,5)'
    testCompile 'junit:junit:4.12'
}
```

Configurando algumas coisas



Add Facebook App ID

Add your Facebook App ID to your app and update your Android manifest.

1. Open your `strings.xml` file, for example: `/app/src/main/res/values/strings.xml`.
2. Add a new string with the name `facebook_app_id` containing the value of your Facebook App ID:

```
<string name="facebook_app_id">1254474881313153</string>
```

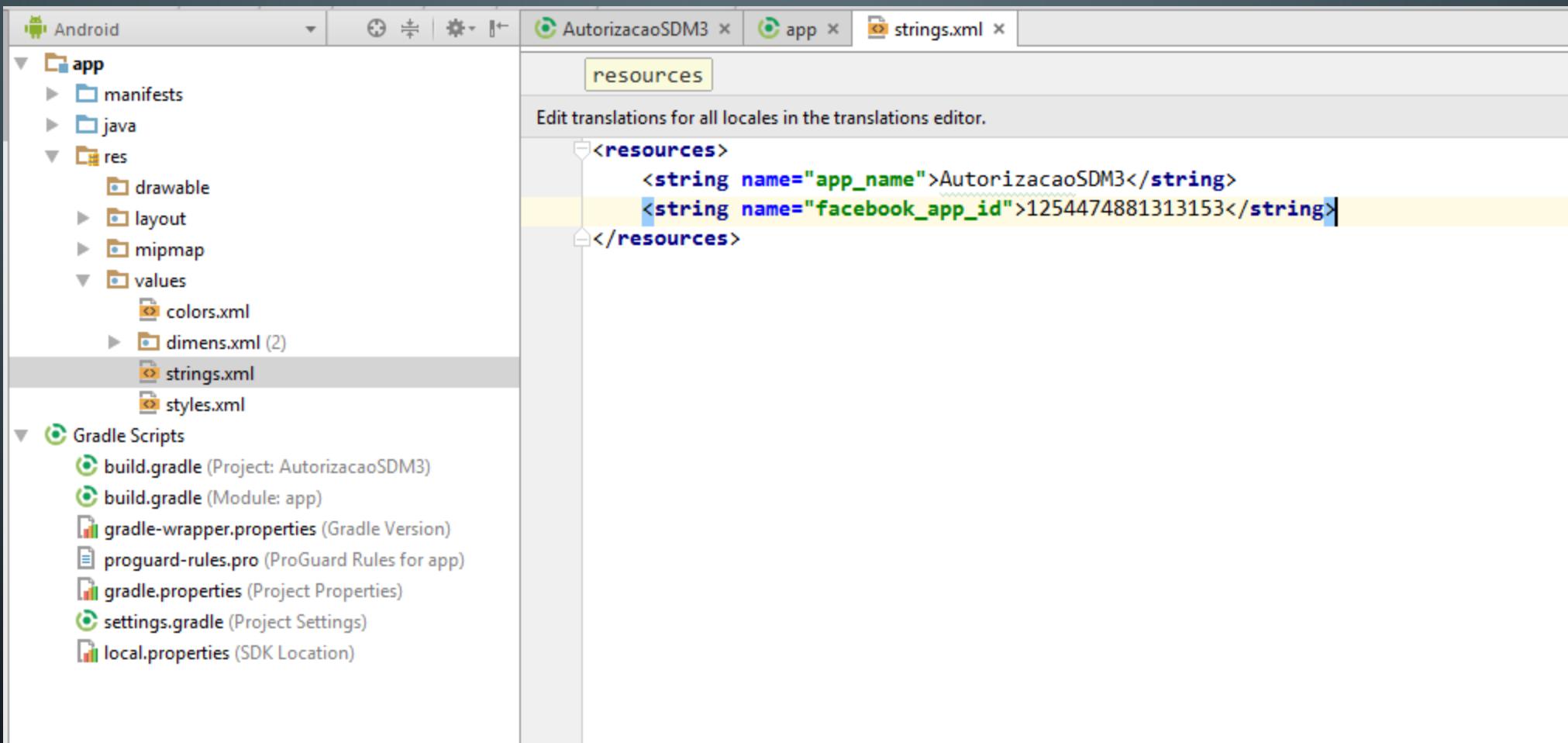
3. Open `AndroidManifest.xml`.
4. Add a `uses-permission` element to the manifest:

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

5. Add a `meta-data` element to the `application` element:

```
<application android:label="@string/app_name" ...>
    ...
    <meta-data android:name="com.facebook.sdk.ApplicationId" android:value="@string/facebook_app_id"/>
    ...
</application>
```

O arquivo strings.xml



The screenshot shows the Android Studio interface with the project navigation bar at the top. The left sidebar displays the project structure under the 'app' module, including 'manifests', 'java', 'res' (with 'drawable', 'layout', 'mipmap', and 'values' subfolders), and files like 'colors.xml', 'dimens.xml', 'strings.xml', and 'styles.xml'. The 'Gradle Scripts' section lists build-related files. The right panel is titled 'resources' and contains the XML code for the 'strings.xml' file. The code defines two string resources: 'app_name' with the value 'AutorizacaoSDM3' and 'facebook_app_id' with the value '1254474881313153'. The code is as follows:

```
<resources>
    <string name="app_name">AutorizacaoSDM3</string>
    <string name="facebook_app_id">1254474881313153</string>
</resources>
```

O arquivo AndroidManifest.xml

The screenshot shows the Android Studio interface with the project 'AutorizacaoSDM3' open. The left sidebar displays the project structure under the 'app' module, including the 'manifests' folder which contains the 'AndroidManifest.xml' file. The main editor window shows the XML code for the manifest file. The code defines a manifest with a package of 'br.edu.ifspsaocarlos.sdm.autorizacaosdm3', requesting internet permission, and defining an application with various attributes like backup, icon, label, supportsRtl, and theme. It also includes a meta-data entry for Facebook App ID and a main activity named 'MainActivity' with its intent filter for launching the app.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="br.edu.ifspsaocarlos.sdm.autorizacaosdm3">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="AutorizacaoSDM3"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <meta-data android:name="com.facebook.sdk.ApplicationId" android:value="@string/facebook_app_id"/>

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Nome do pacote e Activity principal

Tell us about your Android project

Package Name

Your package name uniquely identifies your Android app. We use this to let people download your app from Google Play if they don't have it installed. You can find this in your [Android Manifest](#)

```
br.edu.ifspsaocarlos.sdm.autorizacaosdm3
```

Default Activity Class Name

This is the fully qualified class name of the activity that handles deep linking. We use this when we deep link into your app from the Facebook app. You can also find this in your [Android Manifest](#)

```
br.edu.ifspsaocarlos.sdm.autorizacaosdm3.MainActivity
```

Next

Adicionar a chave do aplicativo

Add your development and release key hashes

To ensure the authenticity of the interactions between your app and Facebook, you need to supply us with the Android key hash for your development environment. If your app has already been published, you should add your release key hash too.

[Show how to generate a development key hash](#)

If your app has already been published, you should also add a hash of your release key.

[Show how to generate a release key hash](#)

Key Hashes

Eg: nm0blrXpAM3cUsheUlyU7pw

Next

Como gerar e um exemplo de chave

```
keytool -exportcert -alias androiddebugkey -keystore ~/.android/debug.keystore | openssl sha1 -binary | openssl base64
```

On Windows, run this command:

```
keytool -exportcert -alias androiddebugkey -keystore %HOMEPATH%\.android\debug.keystore | openssl sha1 -binary | openssl base64
```

This command will generate a 28-character key hash unique to your development environment. Copy and paste it into the field below. You will need to provide a development key hash for the development environment of each person who works on your app.

If your app has already been published, you should also add a hash of your release key.

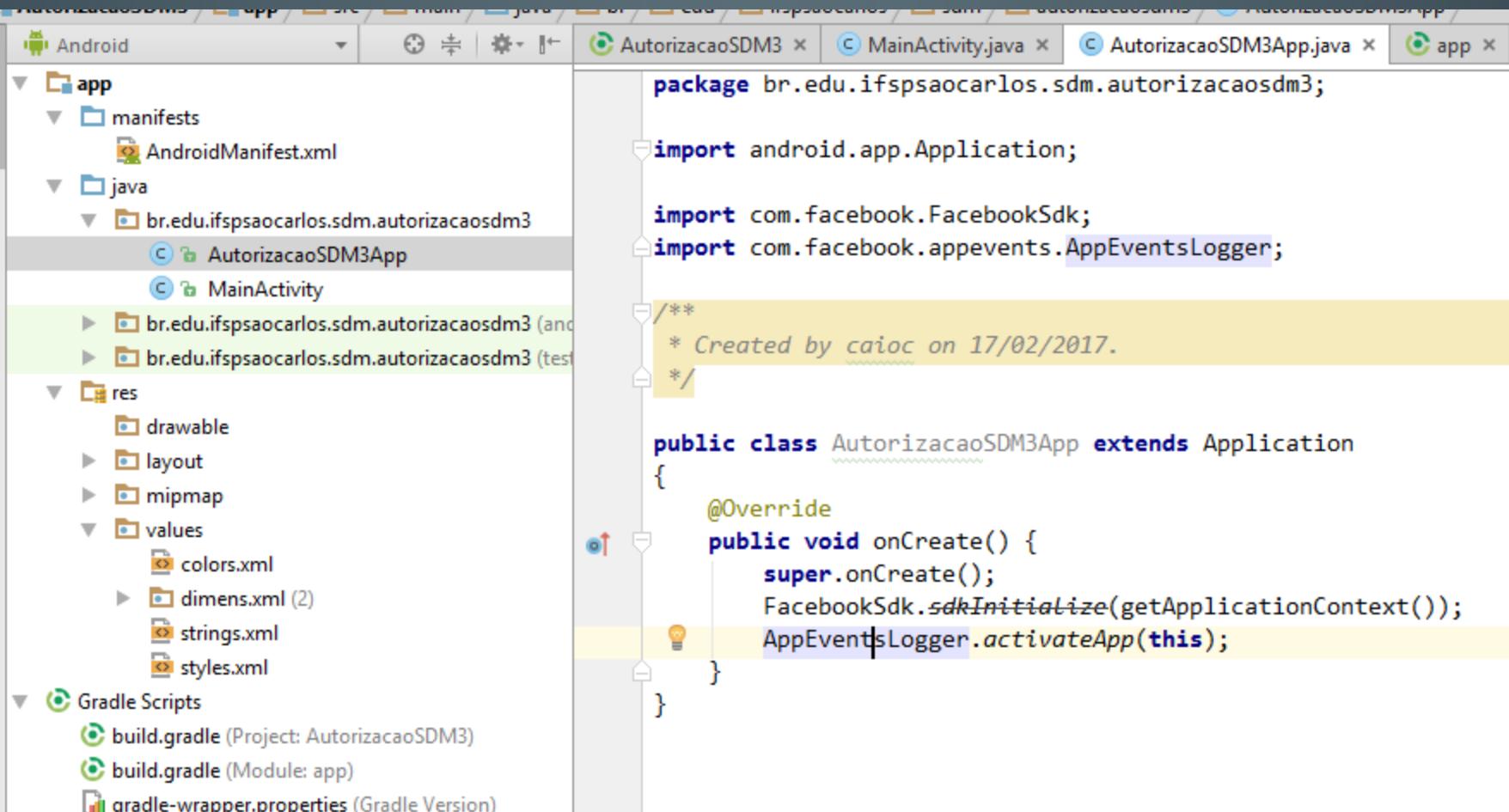
[Show how to generate a release key hash](#)

Key Hashes

```
18gO9G2V439XD78HEgL66mkTm28= x |
```

Next

Iniciar o Facebook SDK



The screenshot shows the Android Studio interface with the project structure and code editor visible.

Project Structure:

- app**:
 - manifests**: Contains `AndroidManifest.xml`.
 - java**: Contains the package `br.edu.ifspsaocarlos.sdm.autorizacaosdm3`, which includes the files `AutorizacaoSDM3App.java` and `MainActivity.java`.
 - res**: Contains `drawable`, `layout`, `mipmap`, and `values` folders, which further contain `colors.xml`, `dimens.xml`, `strings.xml`, and `styles.xml`.
 - Gradle Scripts**: Contains `build.gradle` (Project: AutorizacaoSDM3), `build.gradle` (Module: app), and `gradle-wrapper.properties`.

Code Editor (AutorizacaoSDM3App.java):

```
package br.edu.ifspsaocarlos.sdm.autorizacaosdm3;

import android.app.Application;
import com.facebook.FacebookSdk;
import com.facebook.appevents.AppEventsLogger;

/**
 * Created by caioc on 17/02/2017.
 */

public class AutorizacaoSDM3App extends Application {
    @Override
    public void onCreate() {
        super.onCreate();
        FacebookSdk.sdkInitialize(getApplicationContext());
        AppEventsLogger.activateApp(this);
    }
}
```

Configurar a classe que inicia o SDK

The screenshot shows the Android Studio interface with the project structure on the left and the code editor on the right. The code editor displays the `AndroidManifest.xml` file.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="br.edu.ifspsaocarlos.sdm.autorizacaosdm3">

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

    <application
        android:name=".AutorizacaoSDM3App"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="AutorizacaoSDM3"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <meta-data android:name="com.facebook.sdk.ApplicationId" android:value="@string/facebook_app_id"/>

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Configurar a classe que inicia o SDK

The screenshot shows the Android Studio interface with the project structure on the left and the code editor on the right. The code editor displays the `AndroidManifest.xml` file.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="br.edu.ifspsaocarlos.sdm.autorizacaosdm3">

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

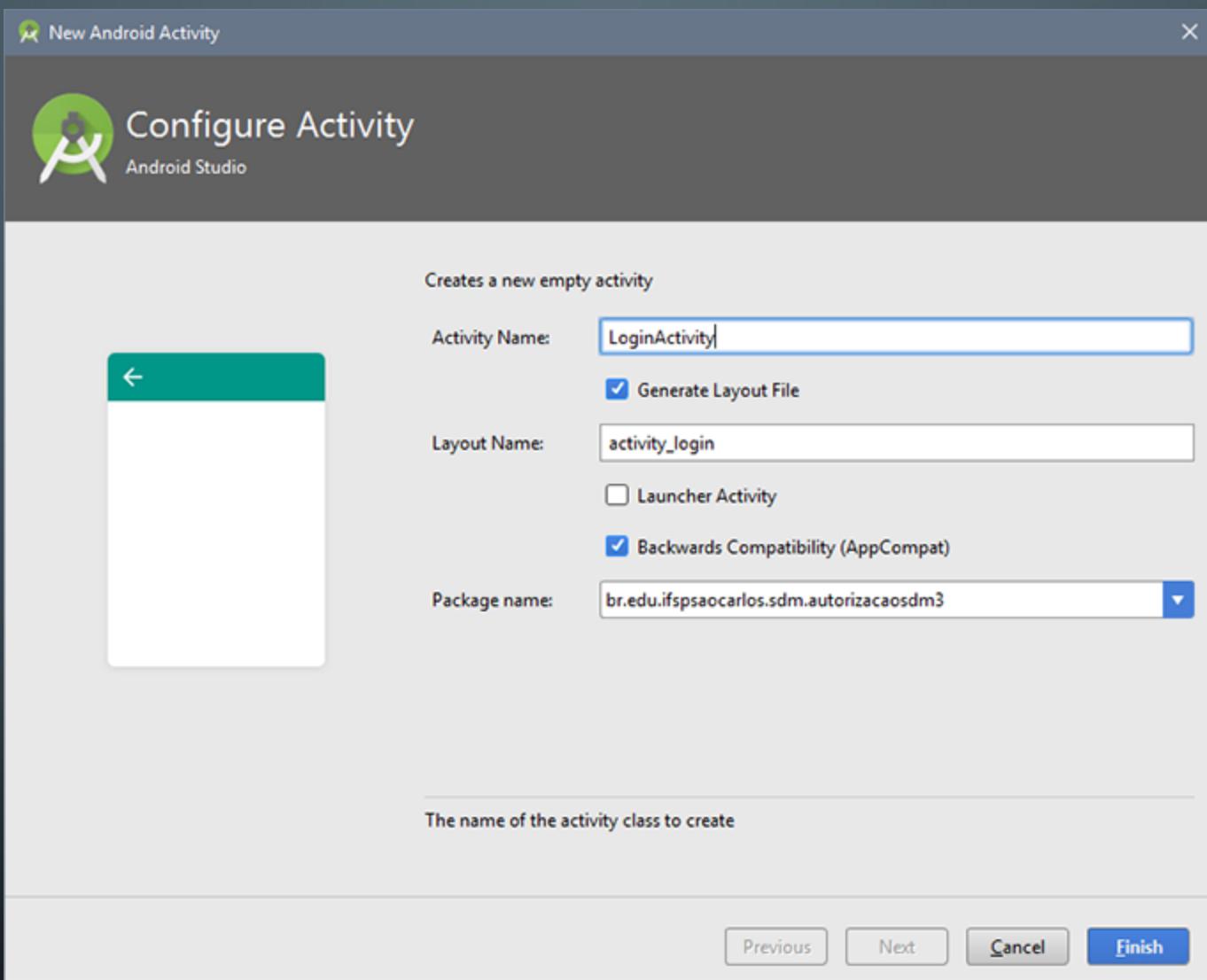
    <application
        android:name=".AutorizacaoSDM3App"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="AutorizacaoSDM3"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <meta-data android:name="com.facebook.sdk.ApplicationId" android:value="@string/facebook_app_id"/>

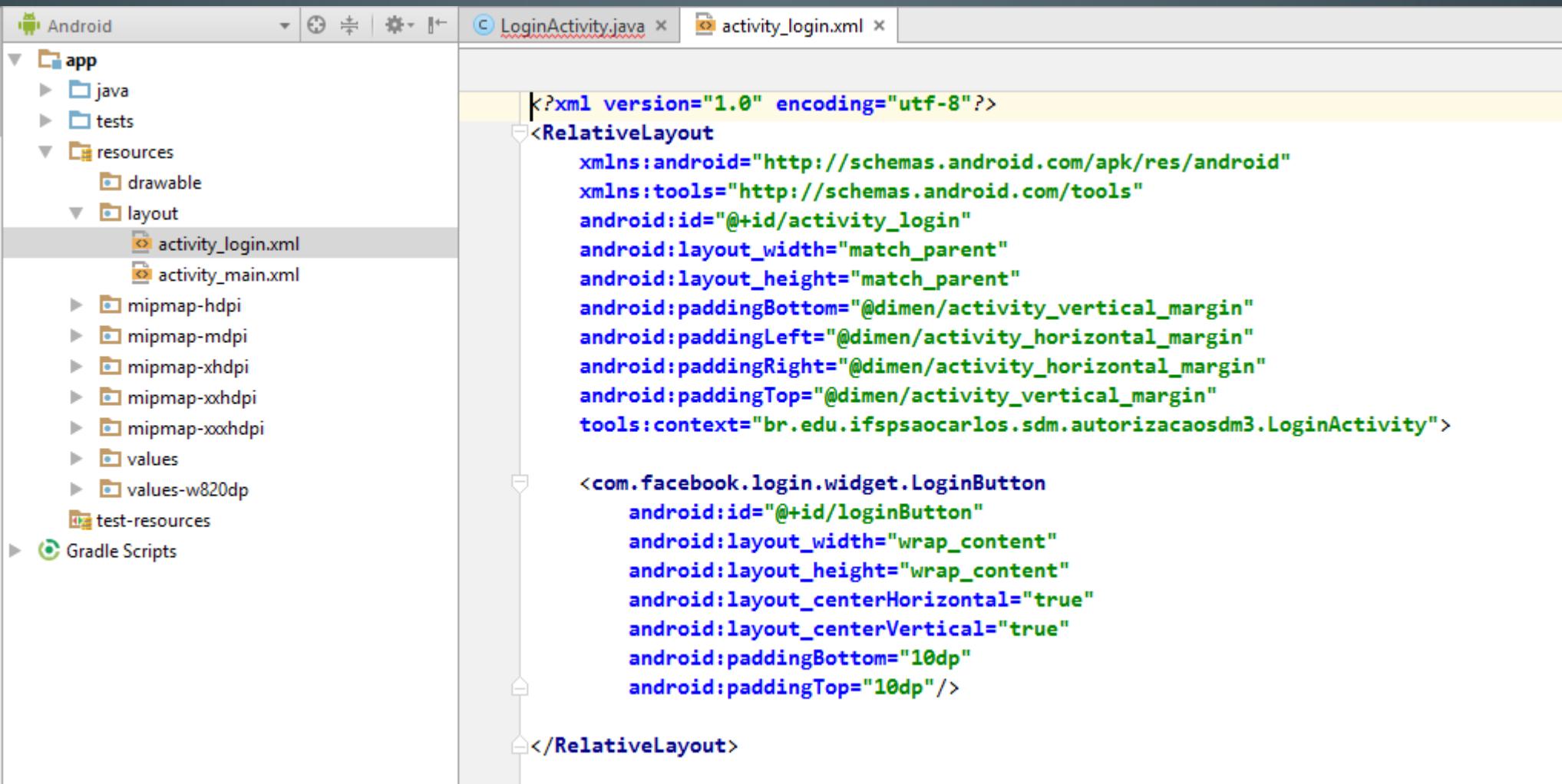
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Nova Activity: LoginActivity



O layout de LoginActivity



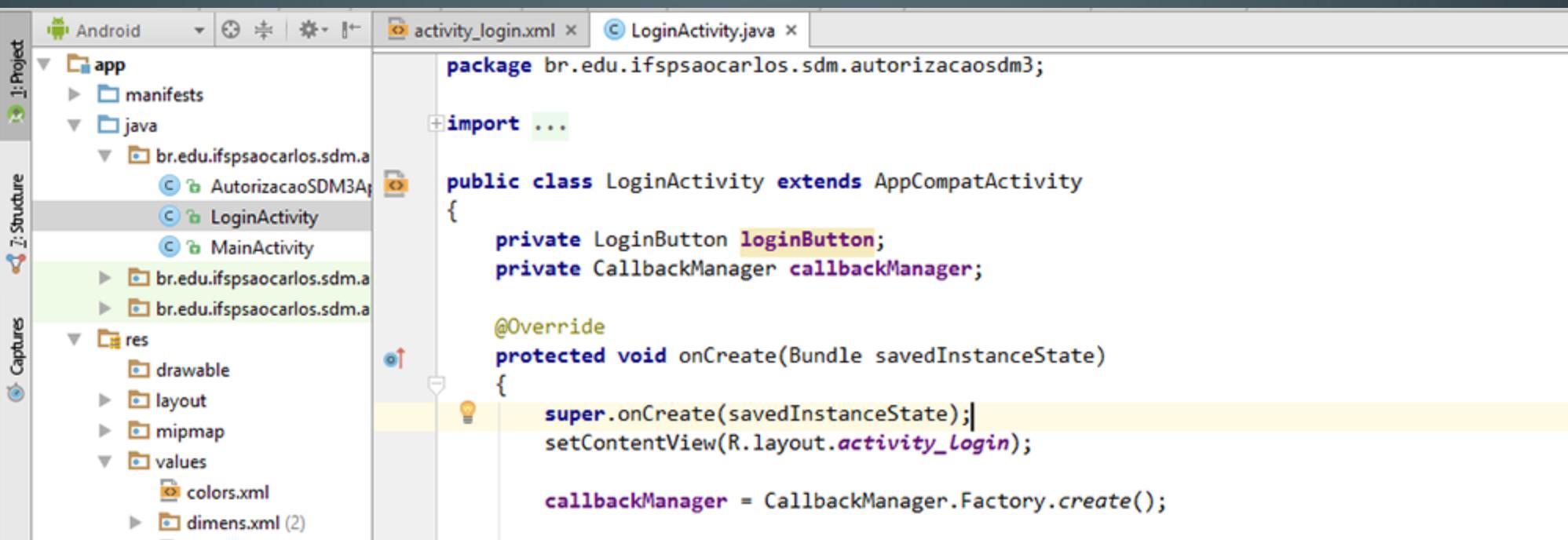
The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left sidebar shows the project structure under the `app` folder:
 - `java`
 - `tests`
 - `resources`:
 - `drawable`
 - `layout`:
 - `activity_login.xml` (selected)
 - `activity_main.xml`
 - `mipmap-hdpi`
 - `mipmap-mdpi`
 - `mipmap-xhdpi`
 - `mipmap-xxhdpi`
 - `mipmap-xxxhdpi`
 - `values`
 - `values-w820dp`
 - `test-resources`
- Files:** The top bar shows the files `LoginActivity.java` and `activity_login.xml`. The `activity_login.xml` file is the active file.
- Code View:** The main area displays the XML code for the `activity_login.xml` layout. It includes a `RelativeLayout` containing a `LoginButton` from the Facebook library. The code is as follows:?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
 xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:tools="http://schemas.android.com/tools"
 android:id="@+id/activity_login"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:paddingBottom="@dimen/activity_vertical_margin"
 android:paddingLeft="@dimen/activity_horizontal_margin"
 android:paddingRight="@dimen/activity_horizontal_margin"
 android:paddingTop="@dimen/activity_vertical_margin"
 tools:context="br.edu.ifspsaocarlos.sdm.autorizacaosdm3.LoginActivity">

 <com.facebook.login.widget.LoginButton
 android:id="@+id/loginButton"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:layout_centerHorizontal="true"
 android:layout_centerVertical="true"
 android:paddingBottom="10dp"
 android:paddingTop="10dp"/>

</RelativeLayout>

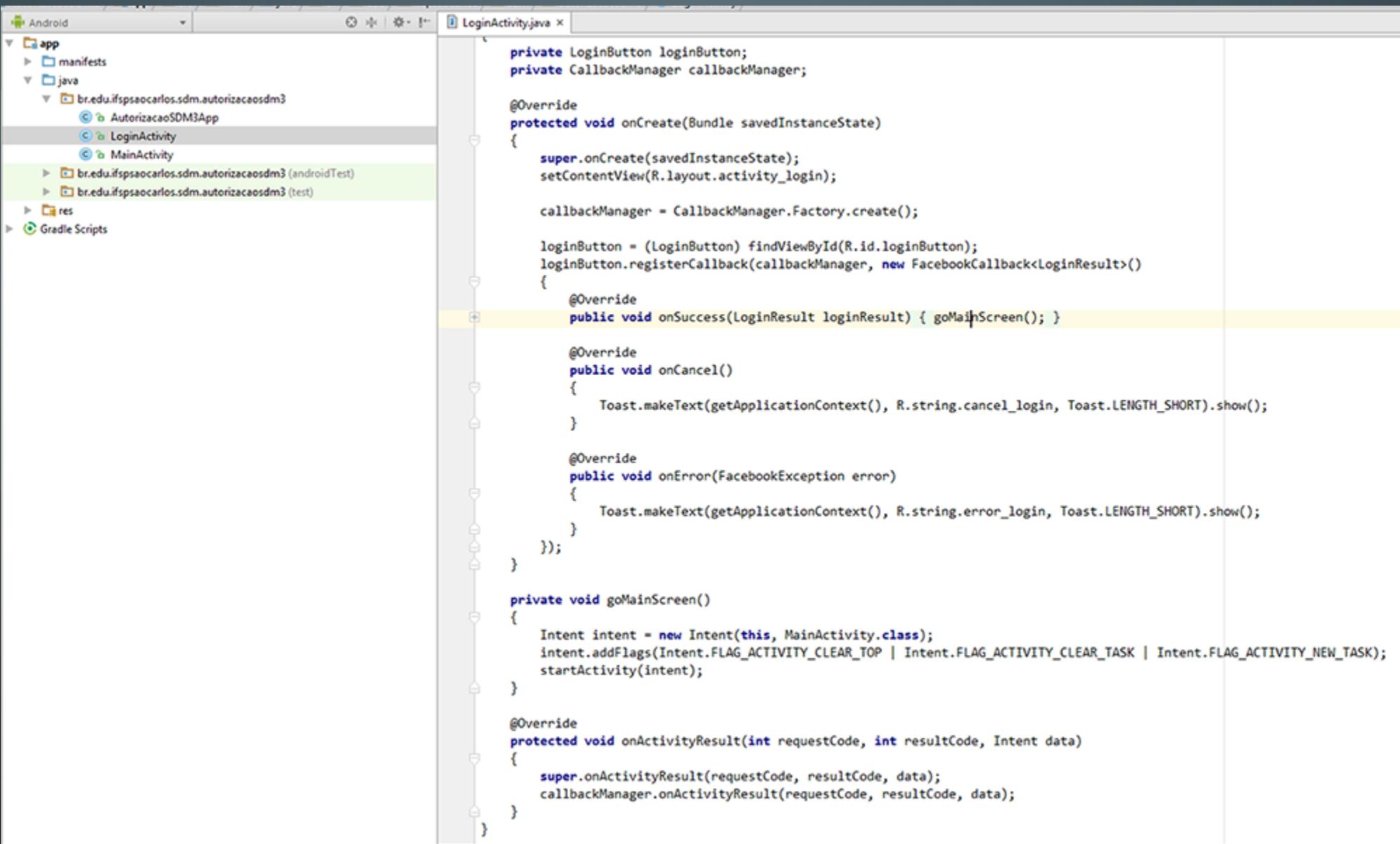
Configurando o login



The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left sidebar shows the project structure under the "app" module. It includes "manifests", "java" (containing "AutorizacaoSDM3Activity", "LoginActivity", and "MainActivity"), and "res" (containing "drawable", "layout", "mipmap", and "values" folders with "colors.xml" and "dimens.xml").
- Code Editor:** The main editor window displays the "LoginActivity.java" code. The code defines a class "LoginActivity" that extends "AppCompatActivity". It contains private fields for a "LoginButton" and a "CallbackManager". The "onCreate" method calls the superclass's "onCreate" and sets the content view to "R.layout.activity_login". It also initializes the "callbackManager" using "CallbackManager.Factory.create()".
- Toolbars:** The top toolbar has icons for file operations like New, Open, Save, and Run.
- Status Bar:** The bottom status bar shows the Android logo and the text "Android Studio".

Configurando o login



The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left sidebar shows the project structure under "app". The "java" folder contains the "br.edu.ifspsaocarlos.sdm.autorizacaosdm3" package, which includes "AutorizacaoSDM3App", "LoginActivity", and "MainActivity".
- LoginActivity.java Code:** The main editor window displays the Java code for `LoginActivity.java`. The code handles Facebook login logic using the Facebook SDK.
- Code Snippet:** The code snippet highlights the `onSuccess` method of the `FacebookCallback<LoginResult>` interface. The highlighted line is `public void onSuccess(LoginResult loginResult) { goMainScreen(); }`.

```
private LoginButton loginButton;
private CallbackManager callbackManager;

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

    callbackManager = CallbackManager.Factory.create();

    loginButton = (LoginButton) findViewById(R.id.loginButton);
    loginButton.registerCallback(callbackManager, new FacebookCallback<LoginResult>()
    {
        @Override
        public void onSuccess(LoginResult loginResult) { goMainScreen(); }

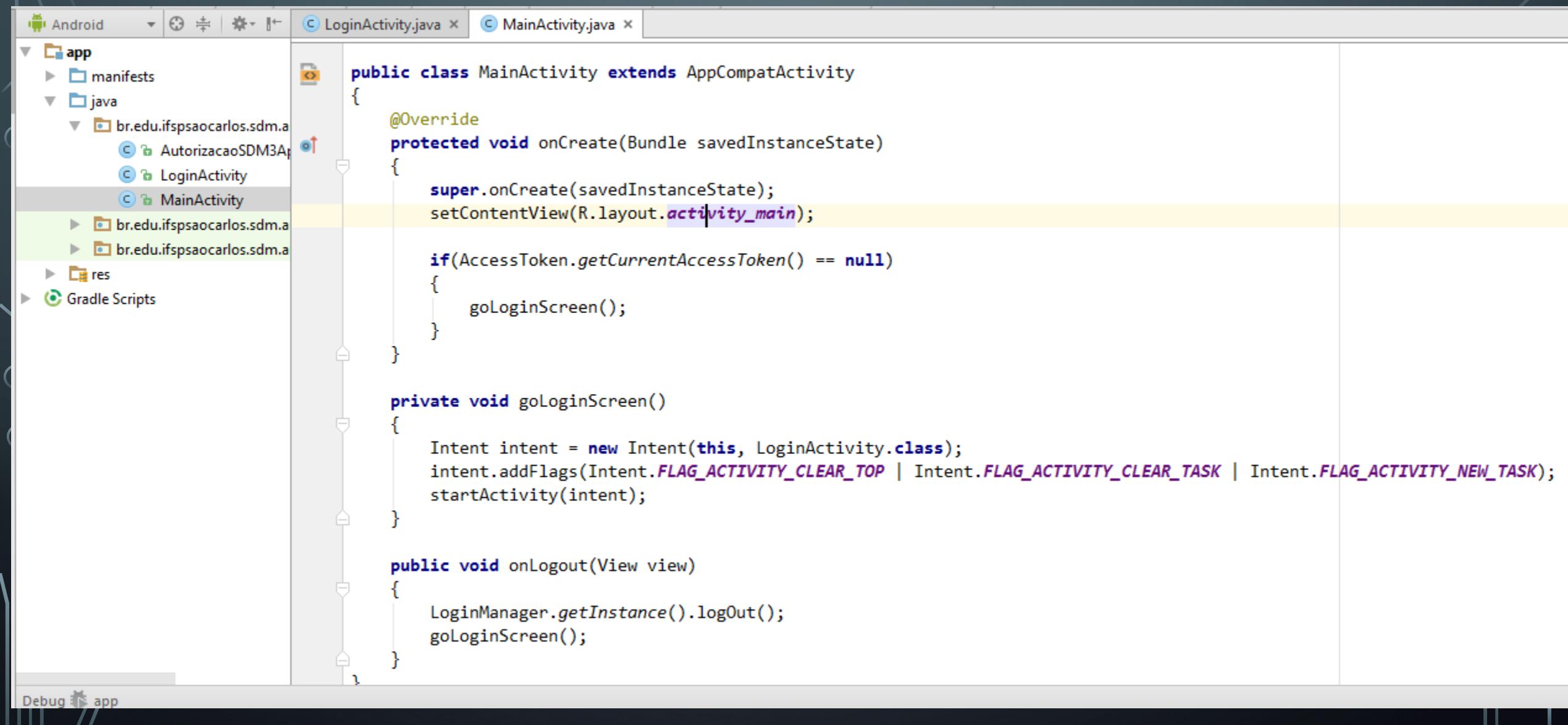
        @Override
        public void onCancel()
        {
            Toast.makeText(getApplicationContext(), R.string.cancel_login, Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onError(FacebookException error)
        {
            Toast.makeText(getApplicationContext(), R.string.error_login, Toast.LENGTH_SHORT).show();
        }
    });
}

private void goMainScreen()
{
    Intent intent = new Intent(this, MainActivity.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_CLEAR_TASK | Intent.FLAG_ACTIVITY_NEW_TASK);
    startActivity(intent);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    super.onActivityResult(requestCode, resultCode, data);
    callbackManager.onActivityResult(requestCode, resultCode, data);
}
```

Finalizando a configuração



The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left sidebar shows the project structure under the `app` folder. It includes `manifests`, `java` (containing `AutorizacaoSDM3API`, `LoginActivity`, and `MainActivity`), and `res`.
- Code Editor:** The main editor window displays the `MainActivity.java` file. The code is as follows:

```
public class MainActivity extends AppCompatActivity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        if(AccessToken.getCurrentAccessToken() == null)
        {
            goLoginScreen();
        }
    }

    private void goLoginScreen()
    {
        Intent intent = new Intent(this, LoginActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_CLEAR_TASK | Intent.FLAG_ACTIVITY_NEW_TASK);
        startActivity(intent);
    }

    public void onLogout(View view)
    {
        LoginManager.getInstance().logOut();
        goLoginScreen();
    }
}
```

- Toolbars and Status Bar:** The top bar shows tabs for `LoginActivity.java` and `MainActivity.java`. The bottom status bar indicates "Debug app".

GOOGLE SIGN-IN

Tutorial de Implementação do Google Sign-In

Pré-requisitos:

- Versão mínima Android 2.3 (API 9)
- Emulador AVD Android 4.2 (API 17) Google Play Services 9.8.0
- Instalar SDK Manager > Extras > Google Repository

Novo Projeto Android Studio:

- Empty Activity
- Obter arquivo de configuração:
<https://developers.google.com/identity/sign-in/android/>

Add Google Sign-In to Your Android App

Configure Google Sign-In:

```
// Configure sign-in to request the user's ID, email address, and basic profile. ID and
// basic profile are included in DEFAULT_SIGN_IN.
GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestEmail()
    .build();

// Build a GoogleApiClient with access to GoogleSignIn.API and the options above.
mGoogleApiClient = new GoogleApiClient.Builder(this)
    .enableAutoManage(this, this)
    .addApi(Auth.GOOGLE_SIGN_IN_API, gso)
    .build();
```

Then, when the sign-in button is clicked, start the sign-in intent:

```
Intent signInIntent = Auth.GoogleSignInApi.getSignInIntent(mGoogleApiClient);
startActivityForResult(signInIntent, RC_SIGN_IN);
```

The user is prompted to select a Google account to sign in with. If you requested scopes beyond `profile`, `email`, and `openid`, the user is also prompted to grant access to the requested resources.

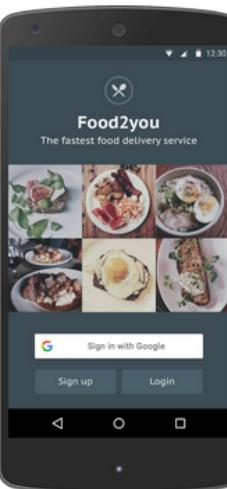
Finally, handle the activity result:

```
@Override public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    // Result returned from launching the Intent from
    // GoogleSignInApi.getSignInIntent(...);
    if (requestCode == RC_SIGN_IN) {
        GoogleSignInResult result = Auth.GoogleSignInApi.getSignInResultFromIntent(data);
        if (result.isSuccess()) {
            GoogleSignInAccount acct = result.getSignInAccount();
            // Get account information
            mFullName = acct.getDisplayName();
            mEmail = acct.getEmail();
        }
    }
}
```

Ready to integrate Google Sign-In into your app?

GET STARTED



Google Sign-In for Android

INÍCIO GUIAS REFERÊNCIA CASE STUDIES

[Try Sign-In for Android](#)

Basics

Start Integrating
Add Sign-In
Get Profile Information
Sign Out Users and Disconnect Apps
Authenticate with a Backend Server
Enable Server-Side API Access

Additional Capabilities

Customize the Sign-In Button
Request Additional Scopes
Over-the-Air App Installs

Migration Guide

Migrate from GoogleAuthUtil and Plus API

Deprecated Docs

Google Sign-In v1 (deprecated)

Try Sign-In for Android



Use our Android sample app to see how Sign-In works, or [add Sign-In to your existing app](#).

Required: The latest versions of [Android Studio](#) and [Google Play Services](#).

1

Get the project

If this is your first time using a Google services sample, check out the [google-services repository](#).

```
$ git clone https://github.com/googlesamples/google-services.git
```

Open Android Studio.

Select **File > Open**, browse to where you cloned the `google-services` repository, and open `google-services/android/signin`.

2

Get a configuration file

To use the sample, you need to provide some additional information to get a configuration file and finish setting up your project. Use the package name `com.google.samples.quickstart.signin` for the sample.

After you complete the registration, download the `google-services.json` file to add to your project.

[GET A CONFIGURATION FILE](#)



Enable Google services for your app

gustavo.rocha.saocarlos@gmail.com
[Sign in as another user](#)

✓ Android

[Create or choose app](#)

[Choose services](#)

[Download config files](#)



Firebase

Powerful new tools from
Google for mobile developers.

[LEARN MORE →](#)

Create or choose an app

App name

Android package name

CONTINUE TO
[Choose and configure services →](#)



Enable Google services for your app

gustavo.rocha.saocarlos@gmail.com
[Sign in as another user](#)

✓ Android

[Create or choose app](#)

[Choose services](#)

[Download config files](#)



Firebase

Powerful new tools from
Google for mobile developers.

[LEARN MORE →](#)

Create or choose an app

App name

G Sign SDM3



Services will be added to your existing project in
the [Google Developers Console](#).

Android package name

br.edu.ifspsaocarlos.sdm3.gsignsdm3

CONTINUE TO
[Choose and configure services →](#)



Enable Google services for your app

gustavo.rocha.saocarlos@gmail.com
[Sign in as another user](#)

- ✓ Android
- ✓ G Sign SDM3

Choose services

[Download config files](#)

 **Firebase**
Powerful new tools from
Google for mobile developers.
[LEARN MORE →](#)

Choose and configure services

✓ You are configuring the **G Sign SDM3** app with package name **br.edu.ifspsaocarlos.sdm3.gsigsdm3**.

Select which Google services you'd like to add to your app below.



Google Sign-In



Analytics



Cloud Messaging

Google Sign-In

✓ Enabled for your app

DOCUMENTATION

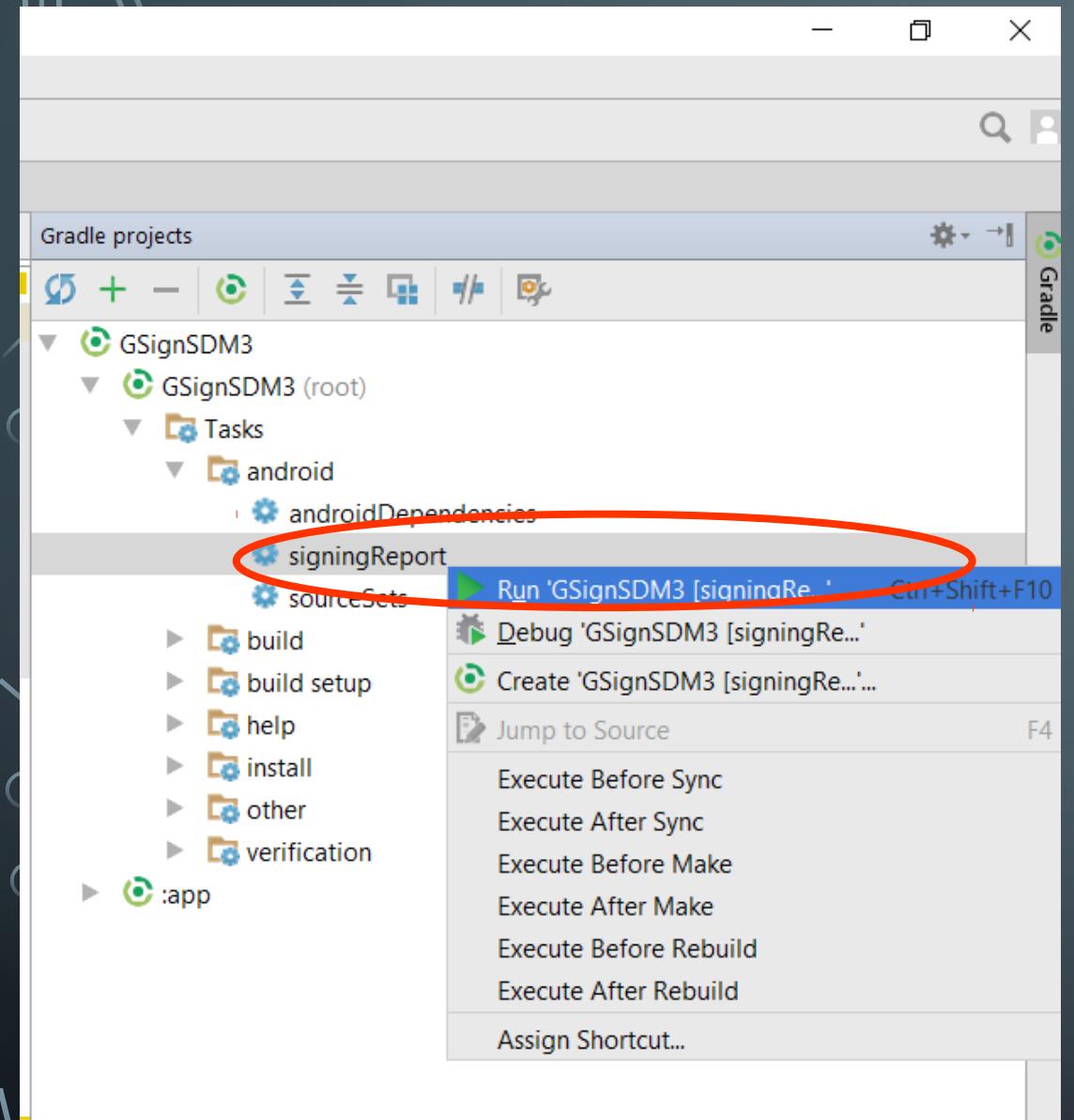
Android Signing Certificate SHA-1

[MANAGE CREDENTIALS](#)

1C:04:H5:DC:77:5E:95:86:AB:C4:AA:01:C7:27:9A:42:86:74

[CLOSE](#)

[CONTINUE TO
Generate configuration files →](#)



```
MD5: 5F:BA:3E:F3:39:AF:1F:45:83:BC:D7:4C:FD:F4:7C:B0
SHA1: 63:D3:B5:DC:77:57:5E:95:86:AB:C4:AA:01:C7:27:9A:42:06:73:B
Valid until: Segunda-feira, 17 de setembro de 2046
-----
Variant: release
Config: none
-----
BUILD SUCCESSFUL

Total time: 10.265 secs
```

The screenshot shows the 'Gradle Console' output. It displays the MD5 and SHA1 fingerprints of the signed APK, their validity period (until September 17, 2046), the build variant ('release'), and the total build time (10.265 seconds). The SHA1 line is circled in red.



gustavo.rocha.saocarlos@gmail.com
Sign in as another user

- ✓ Android
- ✓ G Sign SDM3
- ✓ 1 new service

Download config files

Firebase
Powerful new tools from
Google for mobile developers.
[LEARN MORE →](#)

Download and install configuration



[Download google-services.json
for br.edu.ifspsaocarlos.sdm3.gsignsdm3](#)

The file contains configuration details, such as keys and identifiers, for the services you just enabled. After downloading, copy the google-services.json to the **app/ or mobile/ module directory** in your Android project.

Copiar o arquivo na pasta app/ do projeto

Implement your new services



Google Sign-In

Registered SHA-1s:

63:D3:B5:DC:77:57:5E:95:86:AB:C4:AA:01:C7:27:9A:42:06:7
3:EB

[Documentation](#)



What's next

[Continue Adding Sign-In →](#)

build.gradle (módulo de projeto)

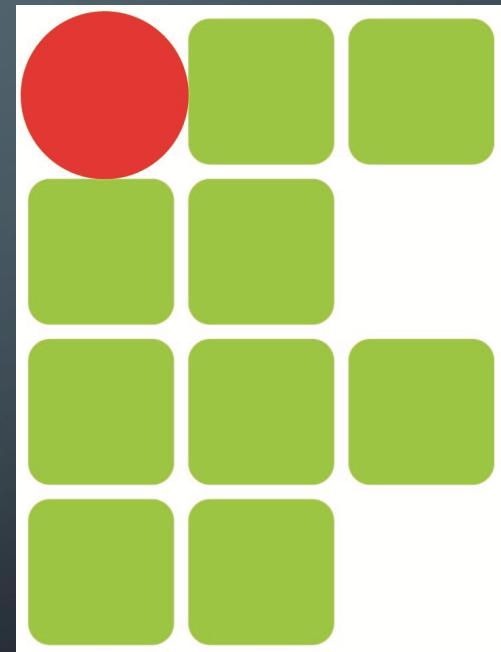
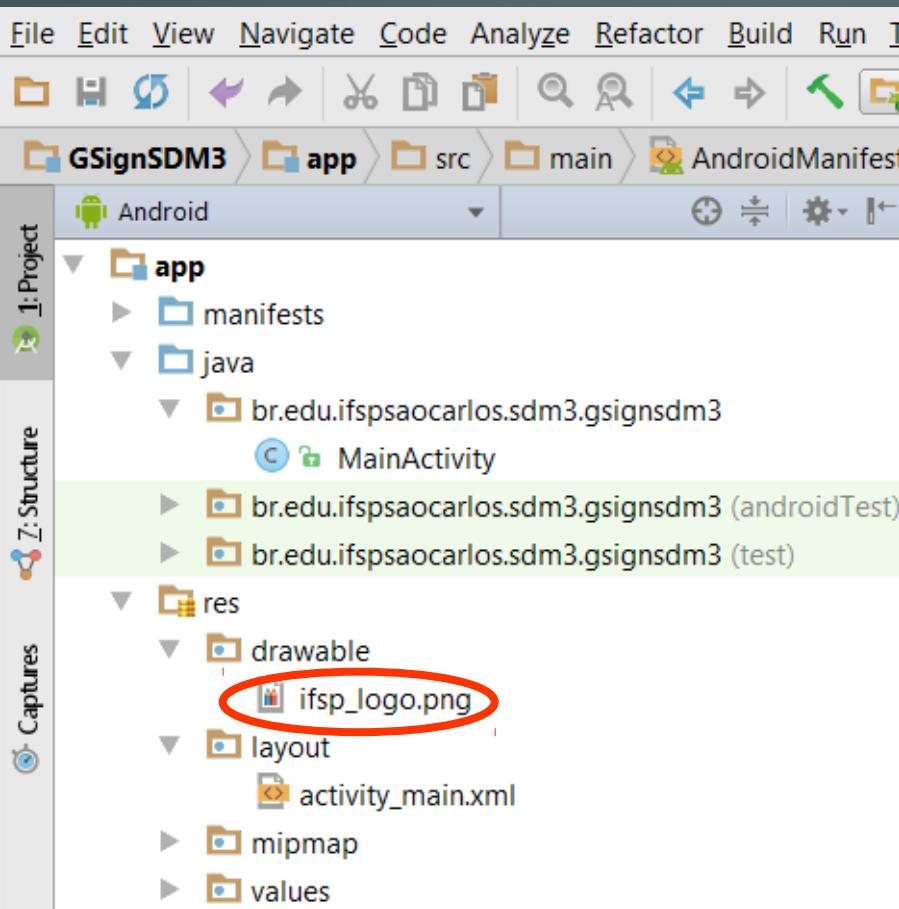
```
dependencies {  
    classpath 'com.android.tools.build:gradle:2.2.3'  
    classpath 'com.google.gms:google-services:3.0.0'  
  
    // NOTE: Do not place your application dependencies here; they belong  
    // in the individual module build.gradle files  
}
```

build.gradle (módulo de app)

```
dependencies {  
    compile 'com.squareup.picasso:picasso:2.5.2'  
    compile 'com.google.android.gms:play-services-auth:9.8.0'  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    androidTestCompile('com.android.support.test.espresso:espresso-  
core:2.2.2', {  
        exclude group: 'com.android.support', module: 'support-  
annotations'  
    })  
    compile 'com.android.support:appcompat-v7:25.1.1'  
    testCompile 'junit:junit:4.12'  
  
    apply plugin: 'com.google.gms.google-services'
```

picasso serve para renderizar a imagem de perfil do usuário

```
<resources>
    <string name="app_name">G Sign SDM3</string>
    <string name="signed_in_fmt">Signed in as: %s</string>
    <string name="signed_out">Signed out</string>
    <string name="sign_out">Sign Out</string>
</resources>
```



```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"          android:orientation="vertical"
    android:layout_width="match_parent"      android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="br.edu.ifsp.sao.carlos.sdm3.gsigns.sdm3.MainActivity">

    <ImageView android:id="@+id/imageView"    android:src="@drawable/ifsp_logo"
        android:layout_width="200sp"           android:layout_height="200sp"
        android:layout_gravity="center"       android:layout_marginBottom="30sp"/>

    <com.google.android.gms.common.SignInButton
        android:layout_width="wrap_content"    android:id="@+id/sign_in_button"
        android:layout_gravity="center"         android:layout_height="wrap_content"
    />

    <TextView android:id="@+id/status"
        android:layout_width="match_parent"    android:layout_height="wrap_content"
        android:text="@string/signed_out"      android:textColor="@android:color/black"
        android:textSize="18sp"                 android:gravity="center"
        android:layout_marginTop="20sp"/>

    <LinearLayout android:id="@+id/sign_out"
        android:layout_width="match_parent"    android:layout_height="match_parent"
        android:orientation="horizontal"       android:paddingLeft="16dp"
        android:visibility="gone"             android:paddingRight="16dp"
        tools:visibility="visible">

        <Button android:id="@+id/sign_out_button"
            android:layout_width="match_parent"  android:layout_height="wrap_content"
            android:text="@string/sign_out"     android:layout_gravity="bottom" />

    </LinearLayout>
</LinearLayout>
```

```
package br.edu.ifspsaocarlos.sdm3.gsignsdm3;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;

import com.google.android.gms.auth.api.Auth;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.android.gms.auth.api.signin.GoogleSignInOptions;
import com.google.android.gms.auth.api.signin.GoogleSignInResult;
import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.SignInButton;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.common.api.ResultCallback;
import com.google.android.gms.common.api.Status;
import com.squareup.picasso.Picasso;
```

```
public class MainActivity extends AppCompatActivity implements
    GoogleApiClient.OnConnectionFailedListener,
    View.OnClickListener {

    private GoogleApiClient mGoogleApiClient;
    private static final String TAG = "SignInActivity";
    private static final int RC_SIGN_IN = 9001;
    private TextView mStatusTextView;
    private ImageView mStatusImageView;

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

        mStatusImageView = (ImageView) findViewById(R.id.imageView);
        mStatusTextView = (TextView) findViewById(R.id.status);
        findViewById(R.id.sign_in_button).setOnClickListener(this);
        findViewById(R.id.sign_out_button).setOnClickListener(this);

        GoogleSignInOptions gso = new
            GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
            .build();

        mGoogleApiClient = new GoogleApiClient.Builder(this)
            .enableAutoManage(this /* FragmentActivity */, this /* OnConnectionFailedListener */)
            .addApi(Auth.GOOGLE_SIGN_IN_API, gso)
            .build();

        SignInButton signInButton = (SignInButton) findViewById(R.id.signIn_button);
        signInButton.setSize(SignInButton.SIZE_WIDE);
    }
}
```

```
@Override
public void onConnectionFailed(ConnectionResult connectionResult) {
    Log.d(TAG, "onConnectionFailed:" + connectionResult);
}
@Override
public void onClick(View v) {
    switch (v.getId()) {
        case R.id.sign_in_button:
            signIn();
            break;
        case R.id.sign_out_button:
            signOut();
            break;
    }
}
private void signIn() {
    Intent signInIntent = Auth.GoogleSignInApi.getSignInIntent(mGoogleApiClient);
    startActivityForResult(signInIntent, RC_SIGN_IN);
}
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == RC_SIGN_IN) {
        GoogleSignInResult result = Auth.GoogleSignInApi.getSignInResultFromIntent(data);
        handleSignInResult(result);
    }
}
private void signOut() {
    Auth.GoogleSignInApi.signOut(mGoogleApiClient).setResultCallback(
        new ResultCallback<Status>() {
            @Override
            public void onResult(Status status) {
                updateUI(false);
            }
        });
}
```

```
private void handleSignInResult(GoogleSignInResult result) {
    Log.d(TAG, "handleSignInResult:" + result.isSuccess());
    if (result.isSuccess()) {
        GoogleSignInAccount acct = result.getSignInAccount();
        mStatusTextView.setText(getString(R.string.signed_in_fmt, acct.getDisplayName()));
        String urlFoto = acct.getPhotoUrl().toString();
        Picasso.with(MainActivity.this).load(urlFoto).resize(100,100).into(mStatusImageView);
        updateUI(true);
    } else {
        updateUI(false);
    }
}

private void updateUI(boolean signedIn) {
    if (signedIn) {
        findViewById(R.id.sign_in_button).setVisibility(View.GONE);
        findViewById(R.id.sign_out).setVisibility(View.VISIBLE);
    } else {
        mStatusTextView.setText(R.string.signed_out);
        mStatusImageView.setImageResource(R.drawable.ifsp_logo);

        findViewById(R.id.sign_in_button).setVisibility(View.VISIBLE);
        findViewById(R.id.sign_out).setVisibility(View.GONE);
    }
}
```



GOOGLE SIGN-IN COM FIREBASE

Bem-vindo ao Firebase

Continue a construir seus aplicativos com o Firebase usando alguns dos recursos abaixo.

[Documentação](#) [Exemplo de código](#) [Referência da API](#) [Suporte](#)



Projetos recentes

Firebase Google Signin App

project-5318104730696456230
1 aplicativo

CRIAR NOVO PROJETO

IMPORTAR PROJETO DO GOOGLE

Criar um projeto

X

Nome do projeto

Google Sign in App

País/região

Brasil

Por padrão, seus dados do Firebase Analytics melhoram outros recursos do Firebase e produtos do Google. Você pode controlar como os dados do Firebase Analytics são compartilhados nas suas configurações a qualquer momento. [Saiba mais](#)

CANCELAR

CRIAR PROJETO

[Overview](#)

Visão geral

[Analytics](#)

DESENVOLVER

[Authentication](#)[Database](#)[Storage](#)[Hosting](#)[Test Lab](#)[Crash Reporting](#)

EXPANDIR

[Notifications](#)[Remote Config](#)

Bem-vindo ao Firebase! Comece aqui.

Adicionar o
Firebase ao seu
aplicativo iOSAdicionar o
Firebase ao seu
aplicativo
AndroidAdicionar o
Firebase ao seu
aplicativo da
Web

Adicionar o Firebase ao seu aplicativo Android

- 1 Register app
- 2 Download config file
- 3 Add Firebase SDK

Comece a usar o Android rapidamente ao clicar em [Ferramentas > Firebase](#) no [Android Studio 2.2+](#)

Nome do pacote

Apelido do aplicativo (opcional)

Certificado de assinatura de depuração SHA-1 (opcional)

Necessário Links dinâmicos, Cotações e Suporte de login do Google na autenticação. Edite o SHA-1 nas Configurações.

[CANCELAR](#)[ADICIONAR APPLICATIVO](#)

faz o download do
google-services.json
para seu aplicativo

Adicionar o Firebase ao seu aplicativo Android

- 1 Inserir detalhes do app
- 2 Copiar arquivo de configuração
- 3 Adicionar a build.gradle

```
buildscript {  
    dependencies {  
        // Add this line  
        classpath 'com.google.gms:google-services:3.0.0'  
    }  
}
```

2. build.gradle do nível do aplicativo (<project>/<app-module>/build.gradle):

```
...  
// Add to the bottom of the file  
apply plugin: 'com.google.gms.google-services'
```

inclui o Firebase Analytics por padrão

3. Finalmente, pressione "Sincronizar agora" na barra que aparece no IDE:

Gradle files have changed since last sync

[FINALIZAR](#)

[Overview](#)

Authentication

CONFIGURAÇÃO DA WEB

USUÁRIOS

MÉTODO DE LOGIN

MODELOS DE E-MAIL

DESENVOLVER

Authentication

Database

Storage

Hosting

Test Lab

Crash Reporting

EXPANDIR

Notifications

Remote Config

Dynamic Links

Provedores de login

Provedor	Status
E-mail/senha	Desativado
Google	Desativado
Facebook	Desativado
Twitter	Desativado
Github	Desativado
Anônimo	Desativado



Google

Ativar



O login do Google é automaticamente configurado em seu iOS conectado e nos aplicativos da Web.
Para defini-lo em seus aplicativos para Android, é necessário adicionar a [impressão digital SHA1](#) em cada aplicativo nas [Configurações do projeto](#).

Adicionar IDs de cliente à lista de permissões a partir de projetos externos (opcional) [?](#)

Configuração do SDK da Web (opcional) [?](#)

CANCELAR

SALVAR

```
apply plugin: 'com.android.application'

android {
    // ...
}

dependencies {
    // ...
    compile 'com.google.firebaseio:firebase-core:9.6.1'
}

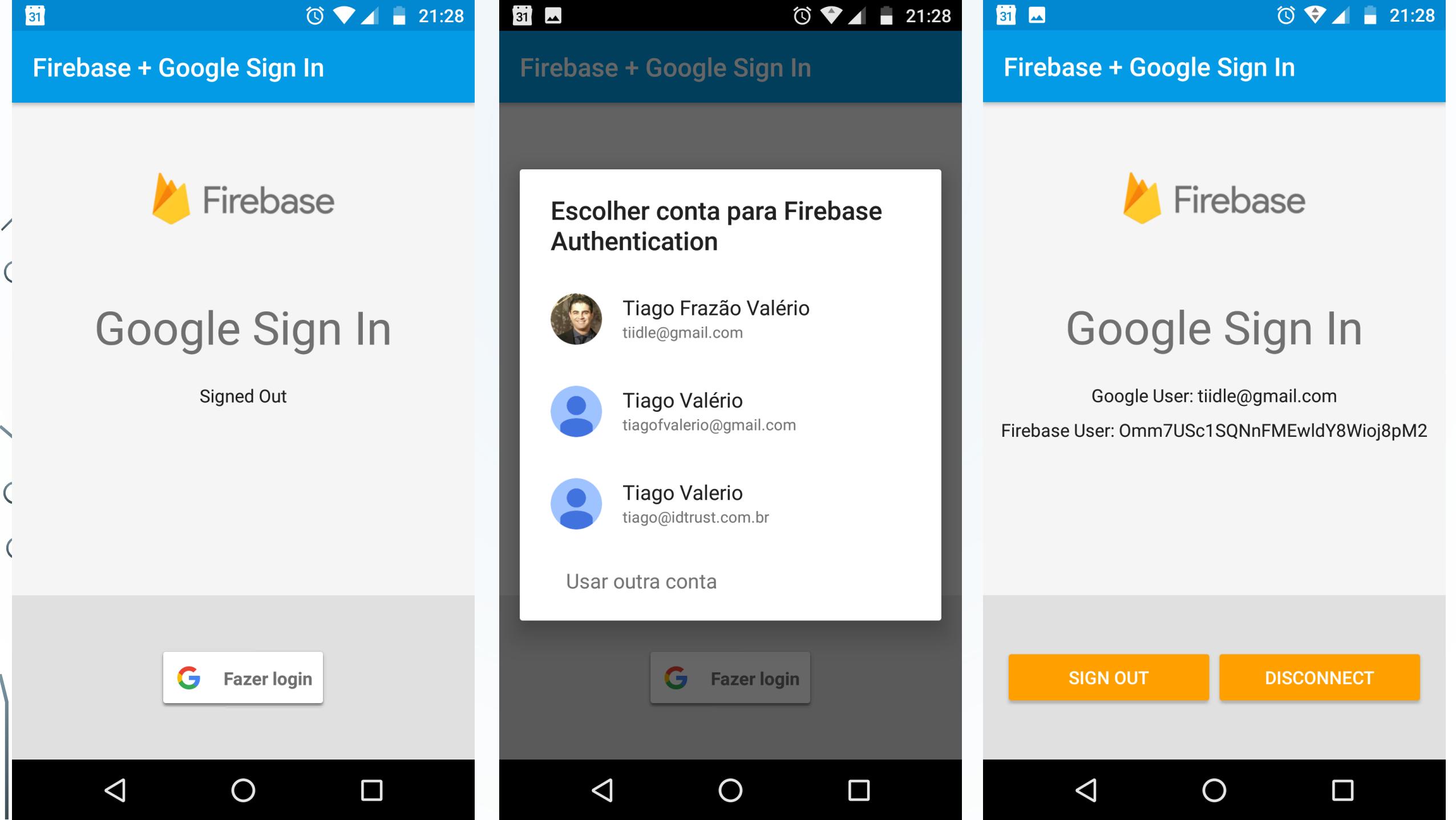
// ADD THIS AT THE BOTTOM
apply plugin: 'com.google.gms.google-services'
```

Linha de dependência do Gradle	Serviço
com.google.firebaseio:firebase-core:9.6.1	Analytics
com.google.firebaseio:firebase-database:9.6.1	Realtime Database
com.google.firebaseio:firebase-storage:9.6.1	Storage
com.google.firebaseio:firebase-crash:9.6.1	Crash Reporting
com.google.firebaseio:firebase-auth:9.6.1	Authentication
com.google.firebaseio:firebase-messaging:9.6.1	Cloud Messaging / Notifications
com.google.firebaseio:firebase-config:9.6.1	Remote Config
com.google.firebaseio:firebase-invites:9.6.1	Invites / Dynamic Links
com.google.firebaseio:firebase-ads:9.6.1	AdMob
com.google.android.gms:play-services-appindexing:9.6.1	App Indexing

```
// [START auth_with_google]
private void firebaseAuthWithGoogle(GoogleSignInAccount acct) {
    Log.d(TAG, "firebaseAuthWithGoogle:" + acct.getId());
    // [START_EXCLUDE silent]
    showProgressDialog();
    // [END_EXCLUDE]

    AuthCredential credential = GoogleAuthProvider.getCredential(acct.getIdToken(), null);
    mAuth.signInWithCredential(credential)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                Log.d(TAG, "signInWithCredential:onComplete:" + task.isSuccessful());

                // If sign in fails, display a message to the user. If sign in succeeds
                // the auth state listener will be notified and logic to handle the
                // signed in user can be handled in the listener.
                if (!task.isSuccessful()) {
                    Log.w(TAG, "signInWithCredential", task.getException());
                    Toast.makeText(GoogleSignInActivity.this, "Authentication failed.",
                        Toast.LENGTH_SHORT).show();
                }
                // [START_EXCLUDE]
                hideProgressDialog();
                // [END_EXCLUDE]
            }
        });
}
// [END auth_with_google]
```



Overview



Authentication

USUÁRIOS

MÉTODO DE LOGIN

MODELOS DE E-MAIL

DESENVOLVER

Authentication

Database

Storage

Hosting

Test Lab

Pesquisar por endereço de e-mail ou UID do usuário

ADICIONAR USUÁRIO



E-mail

Provvedor

Criado em

Conectado

UID do usuário ↑

titide@gmail.com



17 de fev de...

17 de fev de...

Omm7USc1SQNnFMEwidY8Wi...

Linhas por página:

50

1-1 de 1



DÚVIDAS?