Tutorial de utilização da aplicação no lado servidor com Firebase

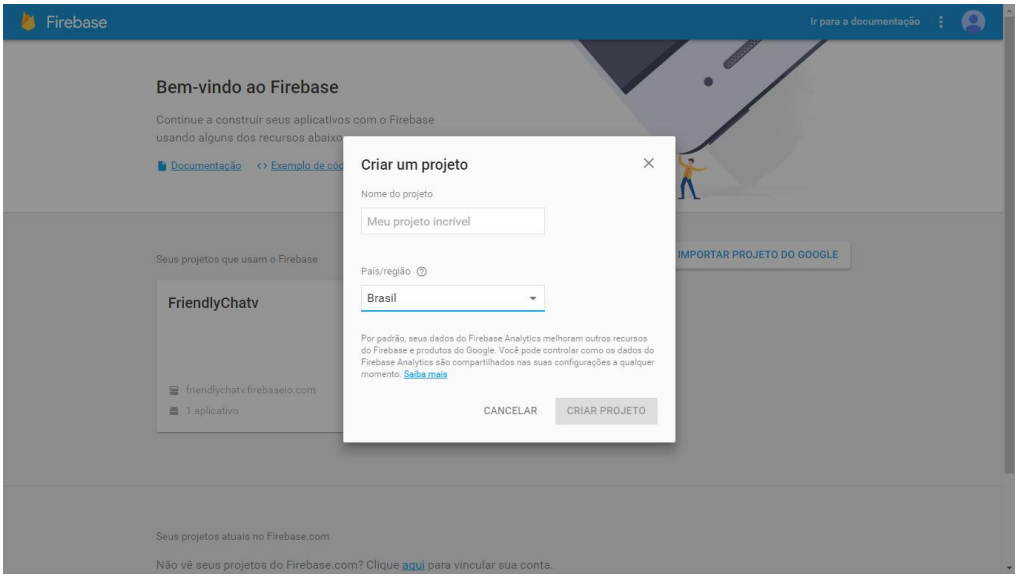
André Luiz Silveira Marinho

1. Pré-requisitos:

- Um dispositivo que execute Android 2.3 (Gingerbread) ou mais recente e o Google Play Services 9.6.1 ou mais recente
- O Google Play Services SDK do Android SDK Manager
- Android Studio 1.5 ou posterior
- Um projeto do Android Studio e o nome do pacote

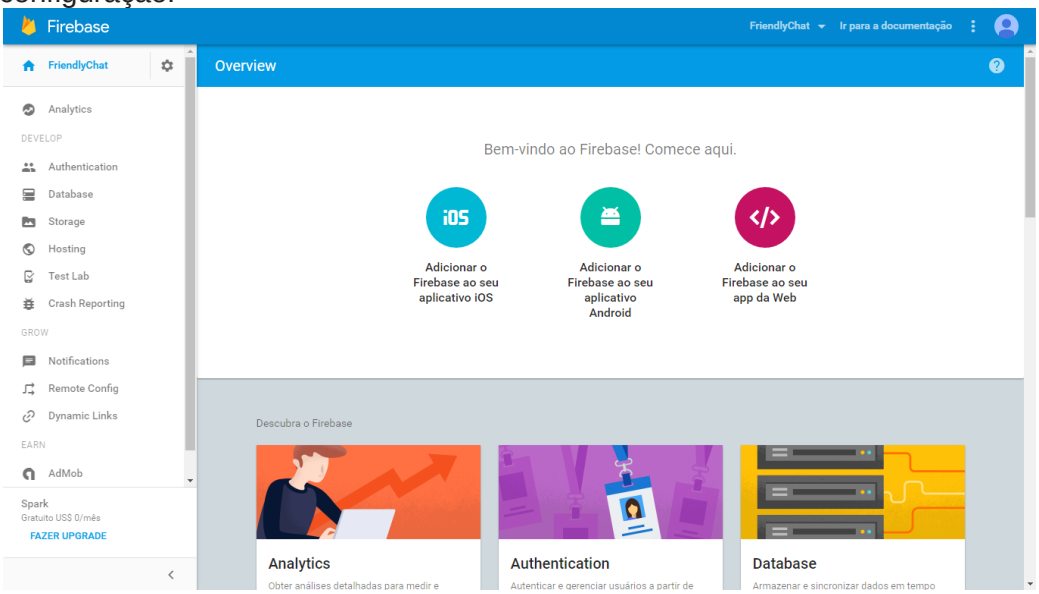
2. Criando o projeto no Firebase:

- Crie um projeto do Firebase no Firebase console, se ainda não tiver um. Se já houver um projeto Google associado ao aplicativo para dispositivos móveis, clique em “importar projeto do Google”. Caso contrário, clique em “criar novo projeto”.



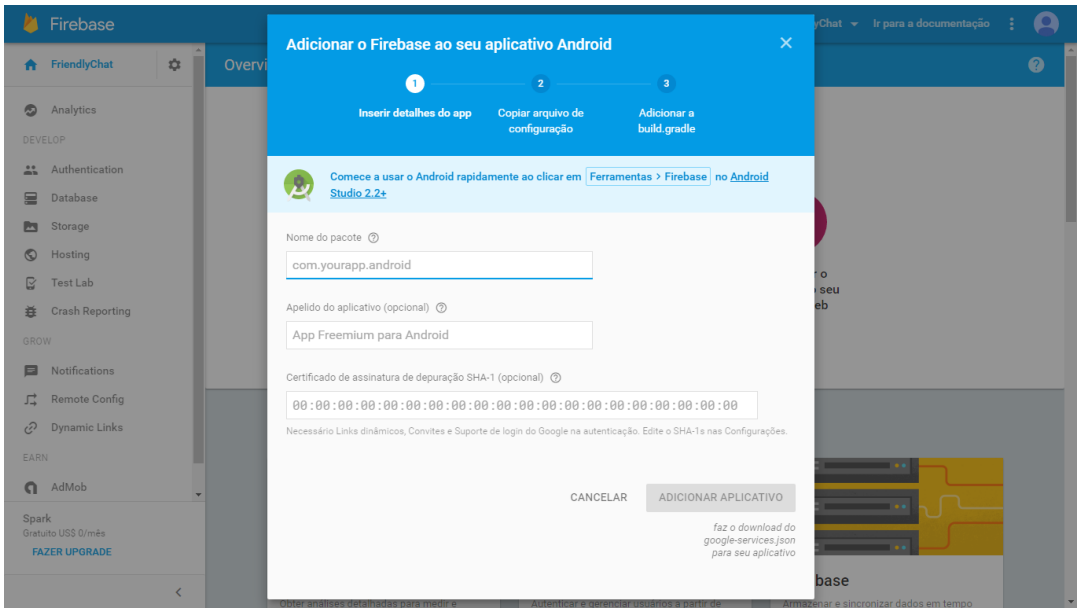
- Clique em “Adicionar o Firebase ao seu aplicativo Android” e siga as etapas de configuração. Se você estiver importando um projeto Google existente, isso poderá acontecer automaticamente e será preciso apenas fazer download do arquivo de

configuração.



- Quando solicitado, insira o nome do pacote do aplicativo. É importante inserir o nome do pacote usado pelo seu aplicativo; isso só poderá ser definido quando você adicionar um aplicativo no projeto do Firebase.

- para conseguir o certificado de assinatura, use o seguinte comando *"keytool -exportcert -alias androiddebugkey -keystore ~/.android/debug.keystore -list -v -storepass android"*



- No final, você fará download de um arquivo google-services.json. Você pode fazer download deste arquivo novamente a qualquer momento. Quando fizer, copie-o para a pasta do módulo do projeto, normalmente app/.

3. Adicionando o SDK:

- Se deseja integrar as bibliotecas do Firebase em um de seus projetos, é preciso executar algumas tarefas básicas para preparar o projeto do Android Studio. Isso já deve ter sido executado como parte da inclusão do Firebase no aplicativo.

- Primeiro, adicione regras no arquivo build.gradle em nível de raiz para incluir o plug-in google-services:

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

- Depois, arquivo Gradle do módulo (geralmente o arquivo app/build.gradle), adicione a linha apply plugin na parte inferior do arquivo para ativar o plug-in do Gradle:

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

android {
  // ...
}

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

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

- Agora é só selecionar no Android Studio “Sync Project with Gradle Files” e pronto.

4. Habilitando Autenticação:

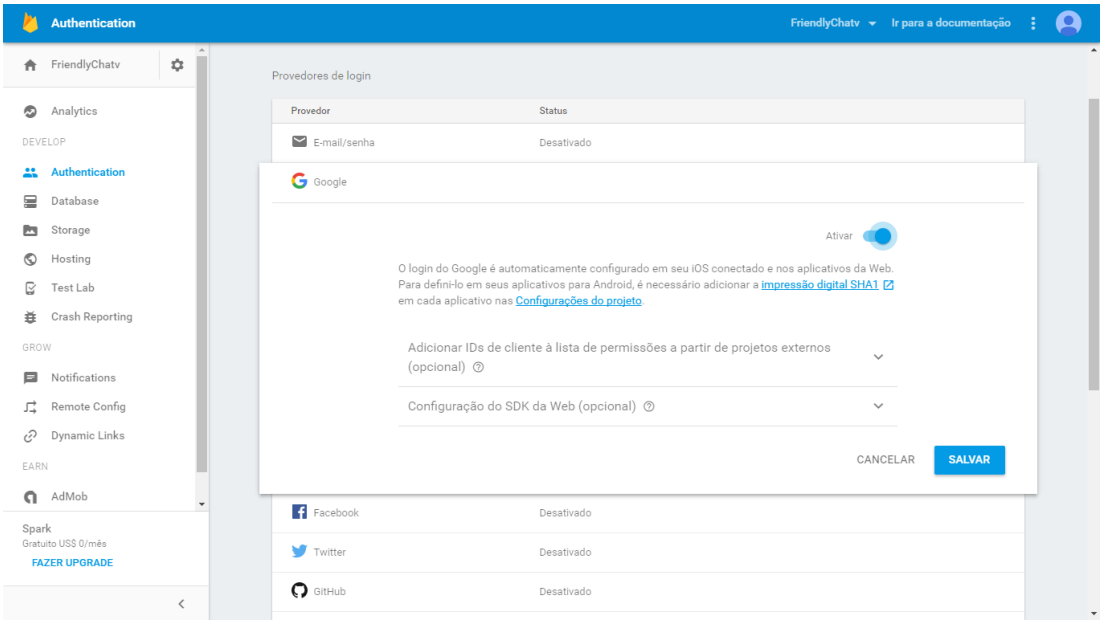
- Como Exemplo das funcionalidades do Firebase, irei mostrar como ele trabalha com a autenticação de usuários. O acesso ao banco de dados Firebase é configurado com uma série de regras escritas em linguagem de configuração JSON.

- Vá até seu projeto no Firebase console, selecione "Database" e então a guia "Regras". Certifique-se de que elas estão escritas como na imagem.

- Adicione as dependências para o Firebase Authentication e o Google Sign-In ao seu arquivo build.gradle de nível de aplicativo:

```
compile 'com.google.firebase:firebase-auth:9.6.1'
compile 'com.google.android.gms:play-services-auth:9.6.1'
```

- Agora no Firebase console, abra a seção “Autenticação”, na guia “Método de Login”, ative o método de login Google e clique em Save.



- Já no código, no método “onCreate” da Activity de login, é necessário configurar Google Sign-in para requisitar os dados do usuário que o app precisa.

```
// Configure Google Sign In
GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
        .requestIdToken(getString(R.string.default_web_client_id))
        .requestEmail()
        .build();
```

- Depois de integrar o Google Sign-In, sua atividade de login terá um código semelhante a:

```
    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);

        // Result returned from launching the Intent from GoogleSignInApi.getSignInIntent(...);
        if (requestCode == RC_SIGN_IN) {
            GoogleSignInResult result = Auth.GoogleSignInApi.getSignInResultFromIntent(data);
            if (result.isSuccess()) {
                // Google Sign In was successful, authenticate with Firebase
                GoogleSignInAccount account = result.getSignInAccount();
                firebaseAuthWithGoogle(account);
            } else {
                // Google Sign In failed, update UI appropriately
                // ...
            }
        }
    }
```

- No método onCreate da atividade de login, obtenha a instância compartilhada do objeto FirebaseAuth:

```
    private FirebaseAuth mAuth;
// ...
        mAuth = FirebaseAuth.getInstance();
```

- Configure um AuthStateListener que responda a alterações no estado de login do usuário:

```
    private FirebaseAuth.AuthStateListener mAuthListener;

// ...

@Override
protected void onCreate(Bundle savedInstanceState) {
    // ...
    mAuthListener = new FirebaseAuth.AuthStateListener() {
        @Override
        public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
            FirebaseUser user = firebaseAuth.getCurrentUser();
            if (user != null) {
                // User is signed in
                Log.d(TAG, "onAuthStateChanged:signed_in:" + user.getUid());
            } else {
                // User is signed out
                Log.d(TAG, "onAuthStateChanged:signed_out");
            }
            // ...
        }
    };
    // ...
}

    @Override
    public void onStart() {
        super.onStart();
        mAuth.addAuthStateListener(mAuthListener);
    }

    @Override
    public void onStop() {
        super.onStop();
        if (mAuthListener != null) {
            mAuth.removeAuthStateListener(mAuthListener);
        }
    }
```

- Depois que um usuário fizer login com sucesso, obtenha um token de ID do objeto GoogleSignInAccount, troque-o por uma credencial do Firebase e autentique com o Firebase usando essa credencial:

```
    private void firebaseAuthWithGoogle(GoogleSignInAccount acct) {
        Log.d(TAG, "firebaseAuthWithGoogle:" + acct.getId());

        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();
                        }
                        // ...
                    }
                });
    }
```

- Se a chamada para signInWithCredential tiver sucesso, o AuthStateListener executa o retorno de chamada de onAuthStateChanged . No retorno de chamada, você poderá usar o método getCurrentUser para obter os dados da conta do usuário.

5. Utilizando o Banco de Dados em Tempo Real:

- Adicione as dependências para o Firebase Realtime Database ao seu arquivo build.gradle de nível de aplicativo:

```
compile 'com.google.firebase:firebase-database:9.8.0'
```

- Deve-se configurar as regras do banco de dados, mas como o tópico anterior já tratou disso com autenticação, então podemos utilizar o banco de dados.
- Para escrever no banco de dados, deve-se recuperar uma instância de seu banco de dados usando “getInstance()” e a referência do local que você deseja escrever.

```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");

myRef.setValue("Hello, World!");
```

- Para fazer o app atualizar em tempo real, é necessário adicionar um “ValueEventListener” com a referência que você criou. O método “onDataChanged()” nessa classe é acionado toda vez que novos dados são adicionados.

```
// Read from the database
myRef.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        // This method is called once with the initial value and again
        // whenever data at this location is updated.
        String value = dataSnapshot.getValue(String.class);
        Log.d(TAG, "Value is: " + value);
    }

    @Override
    public void onCancelled(DatabaseError error) {
        // Failed to read value
        Log.w(TAG, "Failed to read value.", error.toException());
    }
});
```