LEARNING

firebase-authentication

#firebase-authentication

Table of Contents

About 1

Chapter 1: Getting started with firebase-authentication 2

Remarks 2

Examples 2

Installation or Setup 2

Synopsis 2

Firebase v3 Authentication and Authorization Demo Walkthrough 3

How to Create Password Based User 4

Chapter 2: Google Plus Authentication with Android 6

Examples 6

Google Plus Sign-in Authentication 6

Credits 10

About

Chapter 1: Getting started with firebase-authentication

Remarks

This section provides an overview of what firebase-authentication is, and why a developer might want to use it.

It should also mention any large subjects within firebase-authentication, and link out to the related topics. Since the Documentation for firebase-authentication is new, you may need to create initial versions of those related topics.

Examples

Installation or Setup

Synopsis

A fully functional demo of Firebase v3 Web authentication viewable here. Sign in with Facebook, Github, Google, Twitter, password based, and anonymous accounts. The code, available on Github, is easy to read and follow and is well documented. The focus is on the fully functional authentication system.

Password based users are sent a validation link. They can also change their email address and password - both of these events send a verification email as an additional security measure.

Lastly, the difference between authentication, client side authorization, and server side authorization secured via Firebase Realtime Database Security Rules is demonstrated.

1. **Prerequisites**

    1. A Firebase Web project. *FREE!*
    2. An IDE. What's an IDE? Try Cloud9. *FREE!*
    3. A Github, Google, Facebook, and Twitter account. *FREE!*
    4. Two email accounts. *FREE!*

2. **Configure Your IDE**

    2. Create an HTML5 project.
    3. Install Firebase Tools. `npm install -g firebase-tools`
    4. Using Firebase Tools command line, login to your Firebase project. `firebase login --no-localhost`
    5. Using Firebase Tools command line, setup a Firebase project in the current directory. `firebase init`

6. Clone this set of files and folders to your IDE. `git clone https://github.com/rhroyston/firebase-auth.git`
7. Using Firebase Tools command line, push your IDE project to your Firebase project. `firebase deploy`
8. View Firebase project in your browser. Any broken images or errors? Easy fix below.
9. You may need to update `href`, `src`, and `background: url` in all JS, CSS, and all HTML files depending on your Web hosting folder structure .
   1. Use Find feature to search for both `href` and `src` and update as necessary.
   2. Browser Console will display any remaining incorrect paths errors.
   3. Note script.js line 781 `privateLink.href = "../firebase-auth/private"` the `..` seems to be required.
   4. Once all pages render properly from Firebase Hosting (no broken images or console errors), continue.

3. **Configure Firebase**

   1. Enable all 6 forms of authentication. Follow the instructions on configuring social media site settings.
   2. Customize the Email Action Handler URL to point to your Firebase Web app URL + '/ack', e.g. `https://my-app-1234/ack`.

4. **Login to Web app**

   1. Login using an oAuth provider.
   2. From the browser command line, use the exposed `demo.update('mynode','myKey','myValue')` method to add secure markup to your Realtime Database.
      1. A success message will show up in your browser console.
      2. You may need to update the `href` path to match your folder structure.

```
demo.update("markup","secureData","<div class=\"mdl-card__title\"> <h1 class=\"mdl-card__title-text mdl-color-text--white\">Secured Data</h1> </div><div class=\"mdl-card__supporting-text mdl-typography--headline\"> <p>This is a secure card. The HTML markup that renders this card is secured in the Realtime Database.  Access is determined server side so no matter what you do with JavaScript on your browser you will not be able to view this card unless you are authorized to.</p><p>Secured data can be markup, JSON, strings, numbers, etc. Your imagination is the limit!</p></div><div class=\"mdl-card__actions mdl-card--border intro-card-actions\"> <a class=\"mdl-button mdl-button--colored mdl-js-button mdl-js-ripple-effect\" href=\"../firebase-auth/\">Home</a></div>");
```

Firebase v3 Authentication and Authorization Demo Walkthrough

1. **Login using each oAuth provider**

   1. Notice that updating email address or password options are not present in your Account page.
   2. Notice any extra links in the side menu drawer?

    3. Try Deleting your account. What Happens?

2. **Register as a Password based user**
    1. Did you get a verification email?
    2. Can you view private data until you clicked the verification link?
    3. Can you change your password?
    4. Can you change your email address?
    5. Can you undo the email change by clicking the email change notification email revokation link?

3. **Logout**
    1. What links are present in the side menu drawer?
    2. Can you access private data?
    3. Can you view private data?

How to Create Password Based User

You can use Firebase Authentication to let your users authenticate with Firebase using their email addresses and passwords, and to manage your app's password-based accounts.

In this example, we use these steps to set it up for our Android project which is based on JavaScript.

But before doing so, this is what needs to get done before:

1. Add Firebase to your JavaScript project.
2. If you haven't yet connected your app to your Firebase project, do so from the Firebase console.
3. Enable Email/Password sign-in: In the Firebase console, open the Auth section. On the Sign in method tab, enable the Email/password sign-in method and click Save.

There are 2 auth methods required to create a password based user with displayName, namely .createUserWithEmailAndPassword and .updateProfile. I have nested the latter and created a single function which fires both of these methods for ease of use.

```
function registerPasswordUser(email,displayName,password,photoURL){
  var user = null;
  //NULLIFY EMPTY ARGUMENTS
  for (var i = 0; i < arguments.length; i++) {
    arguments[i] = arguments[i] ? arguments[i] : null;
  }
  auth.createUserWithEmailAndPassword(email, password)
  .then(function () {
    user = auth.currentUser;
    user.sendEmailVerification();
  })
  .then(function () {
    user.updateProfile({
      displayName: displayName,
      photoURL: photoURL
```

```
        });
      })
      .catch(function(error) {
        console.log(error.message,7000);
      });
      console.log('Validation link was sent to ' + email + '.');
    }
```

Read Getting started with firebase-authentication online: https://riptutorial.com/firebase-authentication/topic/4744/getting-started-with-firebase-authentication

Chapter 2: Google Plus Authentication with Android

Examples

Google Plus Sign-in Authentication

Authenticate user with **Plus** login

onCreate

```
 GoogleSignInOptions gso = new
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
 .requestIdToken(getString(R.string.default_web_client_id))
 .requestScopes(new Scope(Scopes.PLUS_LOGIN))

 .requestEmail()
 .build();
  mGoogleApiClient = new GoogleApiClient.Builder(this)
 .enableAutoManage(this /* FragmentActivity */, this /* OnConnectionFailedListener */)
 .addConnectionCallbacks(this)
 .addOnConnectionFailedListener(this)
 .addApi(Auth.GOOGLE_SIGN_IN_API,gso)
 .addApi(Plus.API)
 .build();
```

onStart()

```
protected void onStart() {
  super.onStart();
  mGoogleApiClient.connect();
  mAuth.addAuthStateListener(mAuthListener);
}


protected void onStop() {
  super.onStop();
  if (mAuthListener != null) {
    mAuth.removeAuthStateListener(mAuthListener);
  }
  if (mGoogleApiClient.isConnected()) {
    mGoogleApiClient.disconnect();
  }
}

@Override
public void onConnected(Bundle bundle) {
  mSignInClicked = false;
  getProfileInformation();
  //getGoogleOAuthTokenAndLogin();
  Toast.makeText(this, "User is connected! (in onConnected
MActivty)",Toast.LENGTH_LONG).show();
  // Get user's information
  // Update the UI after signin
  //  updateUI(true);
}

@Override
public void onConnectionSuspended(int i) {

  mGoogleApiClient.connect();
  //updateUI(false);
}

@Override
public void onConnectionFailed(ConnectionResult connectionResult) {

  if (!connectionResult.hasResolution()) {
    GooglePlayServicesUtil.getErrorDialog(connectionResult.getErrorCode(), this,
            0).show();
  } else if(connectionResult.hasResolution()) {
    mConnectionResult = connectionResult;
    resolveSignInError();
  }
}



private void resolveSignInError() {
  Log.e("pavan", "User RESOLVE SIGN IN ERROR CALLED OUT OF IF");
  if(mSignInClicked){
    if (mConnectionResult.hasResolution()) {
      try {
          Log.e("pavan", "User RESOLVE SIGN IN ERROR CALLED OUT OF IF  TRY");
          mIntentInProgress = true;
          mConnectionResult.startResolutionForResult(this, GOOGLE_SIGIN);
      } catch (IntentSender.SendIntentException e) {
          Log.e("pavan", "User RESOLVE SIGN IN ERROR CALLED OUT OF IF  CATCH");
          mIntentInProgress = false;
          mGoogleApiClient.connect();
      }
    }
  }
}
```

Get Profile information

```
private void getProfileInformation() {
  try {
      if (Plus.PeopleApi.getCurrentPerson(mGoogleApiClient) != null) {
        Person currentPerson = Plus.PeopleApi
                .getCurrentPerson(mGoogleApiClient);
        String personName = currentPerson.getDisplayName();
        String personPhotoUrl = currentPerson.getImage().getUrl();
        String personGooglePlusProfile = currentPerson.getUrl();
        String email = Plus.AccountApi.getAccountName(mGoogleApiClient);

  } catch (Exception e) {
      e.printStackTrace();
  }
}
```

Authenticating with Firebase,

```
private void firebaseAuthWithGoogle(GoogleSignInAccount acct) {
//AuthCredential credential = GoogleAuthProvider.getCredential(tokenCredential, null);
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(MainActivity.this, "Authentication failed.",
                            Toast.LENGTH_SHORT).show();
                }else{
                    SharedPreferences prefs =
PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
                    Log.e("SahajLOG", "Login PREF ISSSSSSSS ONACTIVITYRESULT
"+prefs.getBoolean("AuthByGplus", AuthByGplus));
                    prefs.edit().putBoolean("AuthByGplus", true).commit();
                    Log.e("SahajLOG", "Login PREF ISSSSSSSS ONACTIVITYRESULT IFTRUE..
"+prefs.getBoolean("AuthByGplus", AuthByGplus));
                    Intent intent=new Intent(getApplicationContext(),MainActivity.class);
                    startActivity(intent);
                    finish();
                }
                // [START_EXCLUDE]
                // hideProgressDialog();
                // [END_EXCLUDE]
            }
        });
}
```

onActivityResult

```
@Override
protected void onActivityResult(int requestCode, int responseCode,
                            Intent intent) {

if (requestCode == GOOGLE_SIGIN) {
    if (responseCode != RESULT_OK) {
        mSignInClicked = false;
    }
    mIntentInProgress = false;
    if (!mGoogleApiClient.isConnecting()) {
        mGoogleApiClient.connect();
        if (mGoogleApiClient.isConnected()) {
           // getGoogleOAuthTokenAndLogin();
        }
    }
    GoogleSignInResult result = Auth.GoogleSignInApi.getSignInResultFromIntent(intent);
    if (result.isSuccess()) {
        // Google Sign In was successful, authenticate with Firebase
        GoogleSignInAccount account = result.getSignInAccount();
```

```
            Log.e("SahajLOG", "account    " + account.getIdToken());
            //getGoogleOAuthTokenAndLogin();
            firebaseAuthWithGoogle(account);
        } else {
            // Google Sign In failed, update UI appropriately
            // [START_EXCLUDE]
           Log.e("SahajLOG", "SIGN IN FAILED");
            // [END_EXCLUDE]
        }

}




public void signInWithGplus() {
if (!mGoogleApiClient.isConnecting()) {
    mSignInClicked = true;
    Intent signInIntent = Auth.GoogleSignInApi.getSignInIntent(mGoogleApiClient);
    startActivityForResult(signInIntent, GOOGLE_SIGIN);

 }

}
```

Logout

```
@Override
public void onLogout() {
  mAuth.signOut();
  if (mGoogleApiClient.isConnected()) {
    Plus.AccountApi.clearDefaultAccount(mGoogleApiClient);
    mGoogleApiClient.disconnect();
    mGoogleApiClient.connect();
    SignedInWithGoogle=false;
  }
  // Google sign out

  switchToLoginFragment();
}
```

Read Google Plus Authentication with Android online: https://riptutorial.com/firebase-authentication/topic/8887/google-plus-authentication-with-android

Credits

S. No	Chapters	Contributors
1	Getting started with firebase-authentication	Big Dude, Community, ErstwhileIII, Ron Royston
2	Google Plus Authentication with Android	Sahaj Rana, ThunderStruct