FirebaseExtended / **flutterfire**

Code | Issues 725 | Pull requests 87 | Actions | Projects 1 | Security | Insi

master

flutterfire / packages / firebase_auth / firebase_auth / example / lib / **signin_page.dart**

PaulTR [Firebase_Auth] Updated auth example to use latest dynamic links buil... … ✓ History

2 contributors

Raw | Blame

782 lines (727 sloc)   23.2 KB

```dart
// Copyright 2019 The Chromium Authors. All rights reserved.
// Use of this source code is governed by a BSD-style license that can be
// found in the LICENSE file.

import 'dart:io';
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:google_sign_in/google_sign_in.dart';
import 'package:firebase_dynamic_links/firebase_dynamic_links.dart';

final FirebaseAuth _auth = FirebaseAuth.instance;
final GoogleSignIn _googleSignIn = GoogleSignIn();

class SignInPage extends StatefulWidget {
  final String title = 'Registration';
  @override
  State<StatefulWidget> createState() => SignInPageState();
}

class SignInPageState extends State<SignInPage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.title),
        actions: <Widget>[
          Builder(builder: (BuildContext context) {
            return FlatButton(
              child: const Text('Sign out'),
              textColor: Theme.of(context).buttonColor,
              onPressed: () async {
                final FirebaseUser user = await _auth.currentUser();
                if (user == null) {
                  Scaffold.of(context).showSnackBar(const SnackBar(
                      content: Text('No one has signed in.'),
                    ));
                    return;
                  }
                  _signOut();
                  final String uid = user.uid;
                  Scaffold.of(context).showSnackBar(SnackBar(
                    content: Text(uid + ' has successfully signed out.'),
                  ));
                },
              );
            })
          ],
        ),
        body: Builder(builder: (BuildContext context) {
          return ListView(
            scrollDirection: Axis.vertical,
            children: <Widget>[
              _EmailPasswordForm(),
              _EmailLinkSignInSection(),
              _AnonymouslySignInSection(),
              _GoogleSignInSection(),
              _PhoneSignInSection(Scaffold.of(context)),
              _OtherProvidersSignInSection(),
            ],
          );
        }),
      );
    }

    // Example code for sign out.
    void _signOut() async {
      await _auth.signOut();
    }
  }

  class _EmailPasswordForm extends StatefulWidget {
    @override
    State<StatefulWidget> createState() => _EmailPasswordFormState();
  }

  class _EmailPasswordFormState extends State<_EmailPasswordForm> {
    final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
    final TextEditingController _emailController = TextEditingController();
    final TextEditingController _passwordController = TextEditingController();
    bool _success;
    String _userEmail;
    @override
    Widget build(BuildContext context) {
      return Form(
        key: _formKey,
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: <Widget>[
            Container(
              child: const Text('Test sign in with email and password'),
              padding: const EdgeInsets.all(16),
              alignment: Alignment.center,
            ),
            TextFormField(
              controller: _emailController,
              decoration: const InputDecoration(labelText: 'Email'),
              validator: (String value) {
                if (value.isEmpty) {
                  return 'Please enter some text';
                }
                return null;
              },
            ),
            TextFormField(
              controller: _passwordController,
              decoration: const InputDecoration(labelText: 'Password'),
              validator: (String value) {
                if (value.isEmpty) {
                  return 'Please enter some text';
                }
                return null;
              },
            ),
            Container(
              padding: const EdgeInsets.symmetric(vertical: 16.0),
              alignment: Alignment.center,
              child: RaisedButton(
                onPressed: () async {
                  if (_formKey.currentState.validate()) {
                    _signInWithEmailAndPassword();
                  }
                },
                child: const Text('Submit'),
              ),
            ),
            Container(
              alignment: Alignment.center,
              padding: const EdgeInsets.symmetric(horizontal: 16),
              child: Text(
                _success == null
                    ? ''
                    : (_success
                        ? 'Successfully signed in ' + _userEmail
                        : 'Sign in failed'),
                style: TextStyle(color: Colors.red),
              ),
            )
          ],
        ),
      );
    }

    @override
    void dispose() {
      _emailController.dispose();
      _passwordController.dispose();
      super.dispose();
    }

    // Example code of how to sign in with email and password.
    void _signInWithEmailAndPassword() async {
      final FirebaseUser user = (await _auth.signInWithEmailAndPassword(
        email: _emailController.text,
        password: _passwordController.text,
      ))
          .user;
      if (user != null) {
        setState(() {
          _success = true;
          _userEmail = user.email;
        });
      } else {
        _success = false;
      }
    }
  }

  class _EmailLinkSignInSection extends StatefulWidget {
    @override
    State<StatefulWidget> createState() => _EmailLinkSignInSectionState();
  }

  class _EmailLinkSignInSectionState extends State<_EmailLinkSignInSection>
      with WidgetsBindingObserver {
    final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
    final TextEditingController _emailController = TextEditingController();

    bool _success;
    String _userEmail;
    String _userID;

    @override
    void initState() {
      super.initState();
      WidgetsBinding.instance.addObserver(this);
    }

    @override
    void dispose() {
      _emailController.dispose();
      WidgetsBinding.instance.removeObserver(this);
      super.dispose();
    }

    @override
    void didChangeAppLifecycleState(AppLifecycleState state) async {
      if (state == AppLifecycleState.resumed) {
        final PendingDynamicLinkData data =
            await FirebaseDynamicLinks.instance.getInitialLink();
        if (data?.link != null) {
          handleLink(data?.link);
        }

      FirebaseDynamicLinks.instance.onLink(
          onSuccess: (PendingDynamicLinkData dynamicLink) async {
        final Uri deepLink = dynamicLink?.link;

        handleLink(deepLink);
      }, onError: (OnLinkErrorException e) async {
        print('onLinkError');
        print(e.message);
      });
    }
  }

  void handleLink(Uri link) async {
    if (link != null) {
      final FirebaseUser user = (await _auth.signInWithEmailAndLink(
        email: _userEmail,
        link: link.toString(),
      ))
          .user;

      if (user != null) {
        _userID = user.uid;
        _success = true;
      } else {
        _success = false;
      }
    } else {
      _success = false;
    }

    setState(() {});
  }

  @override
  Widget build(BuildContext context) {
    return Form(
      key: _formKey,
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          Container(
            child: const Text('Test sign in with email and link'),
            padding: const EdgeInsets.all(16),
            alignment: Alignment.center,
          ),
          TextFormField(
            controller: _emailController,
            decoration: const InputDecoration(labelText: 'Email'),
            validator: (String value) {
              if (value.isEmpty) {
                return 'Please enter your email.';
              }
              return null;
            },
          ),
```

```dart
          Container(
            padding: const EdgeInsets.symmetric(vertical: 16.0),
            alignment: Alignment.center,
            child: RaisedButton(
              onPressed: () async {
                if (_formKey.currentState.validate()) {
                  _signInWithEmailAndLink();
                }
              },
              child: const Text('Submit'),
            ),
          ),
          Container(
            alignment: Alignment.center,
            padding: const EdgeInsets.symmetric(horizontal: 16),
            child: Text(
              _success == null
                  ? ''
                  : (_success
                      ? 'Successfully signed in, uid: ' + _userID
                      : 'Sign in failed'),
              style: TextStyle(color: Colors.red),
            ),
          )
        ],
      ),
    );
  }

  Future<void> _signInWithEmailAndLink() async {
    _userEmail = _emailController.text;

    return await _auth.sendSignInWithEmailLink(
      email: _userEmail,
      url: '<Url with domain from your Firebase project>',
      handleCodeInApp: true,
      iOSBundleID: 'io.flutter.plugins.firebaseAuthExample',
      androidPackageName: 'io.flutter.plugins.firebaseauthexample',
      androidInstallIfNotAvailable: true,
      androidMinimumVersion: "1",
    );
  }
}

class _AnonymouslySignInSection extends StatefulWidget {
  @override
  State<StatefulWidget> createState() => _AnonymouslySignInSectionState();
}

class _AnonymouslySignInSectionState extends State<_AnonymouslySignInSection> {
  bool _success;
  String _userID;
  @override
  Widget build(BuildContext context) {
    return Column(
      crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          Container(
            child: const Text('Test sign in anonymously'),
            padding: const EdgeInsets.all(16),
            alignment: Alignment.center,
          ),
          Container(
            padding: const EdgeInsets.symmetric(vertical: 16.0),
            alignment: Alignment.center,
            child: RaisedButton(
              onPressed: () async {
                _signInAnonymously();
              },
              child: const Text('Sign in anonymously'),
            ),
          ),
          Container(
            alignment: Alignment.center,
            padding: const EdgeInsets.symmetric(horizontal: 16),
            child: Text(
              _success == null
                  ? ''
                  : (_success
                      ? 'Successfully signed in, uid: ' + _userID
                      : 'Sign in failed'),
              style: TextStyle(color: Colors.red),
            ),
          )
        ],
      );
    }

    // Example code of how to sign in anonymously.
    void _signInAnonymously() async {
      final FirebaseUser user = (await _auth.signInAnonymously()).user;
      assert(user != null);
      assert(user.isAnonymous);
      assert(!user.isEmailVerified);
      assert(await user.getIdToken() != null);
      if (Platform.isIOS) {
        // Anonymous auth doesn't show up as a provider on iOS
        assert(user.providerData.isEmpty);
      } else if (Platform.isAndroid) {
        // Anonymous auth does show up as a provider on Android
        assert(user.providerData.length == 1);
        assert(user.providerData[0].providerId == 'firebase');
        assert(user.providerData[0].uid != null);
        assert(user.providerData[0].displayName == null);
        assert(user.providerData[0].photoUrl == null);
        assert(user.providerData[0].email == null);
      }

      final FirebaseUser currentUser = await _auth.currentUser();
      assert(user.uid == currentUser.uid);
      setState(() {
        if (user != null) {
```

```dart
        _success = true;
        _userID = user.uid;
      } else {
        _success = false;
      }
    });
  }
}

class _GoogleSignInSection extends StatefulWidget {
  @override
  State<StatefulWidget> createState() => _GoogleSignInSectionState();
}

class _GoogleSignInSectionState extends State<_GoogleSignInSection> {
  bool _success;
  String _userID;
  @override
  Widget build(BuildContext context) {
    return Column(
      children: <Widget>[
        Container(
          child: const Text('Test sign in with Google'),
          padding: const EdgeInsets.all(16),
          alignment: Alignment.center,
        ),
        Container(
          padding: const EdgeInsets.symmetric(vertical: 16.0),
          alignment: Alignment.center,
          child: RaisedButton(
            onPressed: () async {
              _signInWithGoogle();
            },
            child: const Text('Sign in with Google'),
          ),
        ),
        Container(
          alignment: Alignment.center,
          padding: const EdgeInsets.symmetric(horizontal: 16),
          child: Text(
            _success == null
                ? ''
                : (_success
                    ? 'Successfully signed in, uid: ' + _userID
                    : 'Sign in failed'),
            style: TextStyle(color: Colors.red),
          ),
        )
      ],
    );
  }

  // Example code of how to sign in with google.
  void _signInWithGoogle() async {
    final GoogleSignInAccount googleUser = await _googleSignIn.signIn();
    final GoogleSignInAuthentication googleAuth =
          await googleUser.authentication;
    final AuthCredential credential = GoogleAuthProvider.getCredential(
      accessToken: googleAuth.accessToken,
      idToken: googleAuth.idToken,
    );
    final FirebaseUser user =
        (await _auth.signInWithCredential(credential)).user;
    assert(user.email != null);
    assert(user.displayName != null);
    assert(!user.isAnonymous);
    assert(await user.getIdToken() != null);

    final FirebaseUser currentUser = await _auth.currentUser();
    assert(user.uid == currentUser.uid);
    setState(() {
      if (user != null) {
        _success = true;
        _userID = user.uid;
      } else {
        _success = false;
      }
    });
  }
}

class _PhoneSignInSection extends StatefulWidget {
  _PhoneSignInSection(this._scaffold);

  final ScaffoldState _scaffold;
  @override
  State<StatefulWidget> createState() => _PhoneSignInSectionState();
}

class _PhoneSignInSectionState extends State<_PhoneSignInSection> {
  final TextEditingController _phoneNumberController = TextEditingController();
  final TextEditingController _smsController = TextEditingController();

  String _message = '';
  String _verificationId;

  @override
  Widget build(BuildContext context) {
    return Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: <Widget>[
        Container(
          child: const Text('Test sign in with phone number'),
          padding: const EdgeInsets.all(16),
          alignment: Alignment.center,
        ),
        TextFormField(
          controller: _phoneNumberController,
          decoration: const InputDecoration(
              labelText: 'Phone number (+x xxx-xxx-xxxx)'),
          validator: (String value) {
            if (value.isEmpty) {
              return 'Phone number (+x xxx-xxx-xxxx)';
            }
            return null;
          },
        ),
        Container(
          padding: const EdgeInsets.symmetric(vertical: 16.0),
          alignment: Alignment.center,
          child: RaisedButton(
            onPressed: () async {
              _verifyPhoneNumber();
            },
            child: const Text('Verify phone number'),
          ),
        ),
        TextField(
          controller: _smsController,
          decoration: const InputDecoration(labelText: 'Verification code'),
        ),
        Container(
          padding: const EdgeInsets.symmetric(vertical: 16.0),
          alignment: Alignment.center,
          child: RaisedButton(
            onPressed: () async {
              _signInWithPhoneNumber();
            },
            child: const Text('Sign in with phone number'),
          ),
        ),
        Container(
          alignment: Alignment.center,
          padding: const EdgeInsets.symmetric(horizontal: 16),
          child: Text(
            _message,
            style: TextStyle(color: Colors.red),
          ),
        )
      ],
    );
  }

  // Example code of how to verify phone number
  void _verifyPhoneNumber() async {
    setState(() {
      _message = '';
    });
    final PhoneVerificationCompleted verificationCompleted =
        (AuthCredential phoneAuthCredential) {
      _auth.signInWithCredential(phoneAuthCredential);
      setState(() {
        _message = 'Received phone auth credential: $phoneAuthCredential';
      });
    };

    final PhoneVerificationFailed verificationFailed =
        (AuthException authException) {
```

```
        setState(() {
          _message =
              'Phone number verification failed. Code: ${authException.code}. Message: ${authException.messa
        });
      };

      final PhoneCodeSent codeSent =
          (String verificationId, [int forceResendingToken]) async {
        widget._scaffold.showSnackBar(const SnackBar(
          content: Text('Please check your phone for the verification code.'),
        ));
        _verificationId = verificationId;
      };

      final PhoneCodeAutoRetrievalTimeout codeAutoRetrievalTimeout =
          (String verificationId) {
        _verificationId = verificationId;
      };

      await _auth.verifyPhoneNumber(
          phoneNumber: _phoneNumberController.text,
          timeout: const Duration(seconds: 5),
          verificationCompleted: verificationCompleted,
          verificationFailed: verificationFailed,
          codeSent: codeSent,
          codeAutoRetrievalTimeout: codeAutoRetrievalTimeout);
    }

    // Example code of how to sign in with phone.
    void _signInWithPhoneNumber() async {
      final AuthCredential credential = PhoneAuthProvider.getCredential(
        verificationId: _verificationId,
        smsCode: _smsController.text,
      );
      final FirebaseUser user =
          (await _auth.signInWithCredential(credential)).user;
      final FirebaseUser currentUser = await _auth.currentUser();
      assert(user.uid == currentUser.uid);
      setState(() {
        if (user != null) {
          _message = 'Successfully signed in, uid: ' + user.uid;
        } else {
          _message = 'Sign in failed';
        }
      });
    }
  }

  class _OtherProvidersSignInSection extends StatefulWidget {
    _OtherProvidersSignInSection();

    @override
    State<StatefulWidget> createState() => _OtherProvidersSignInSectionState();
  }

  class _OtherProvidersSignInSectionState
```

```
    extends State<_OtherProvidersSignInSection> {
  final TextEditingController _tokenController = TextEditingController();
  final TextEditingController _tokenSecretController = TextEditingController();

  String _message = '';
  int _selection = 0;
  bool _showAuthSecretTextField = false;

  @override
  Widget build(BuildContext context) {
    return Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: <Widget>[
        Container(
          child: const Text(
              'Test other providers authentication. (We do not provide an API to obtain the token for belo
          padding: const EdgeInsets.all(16),
          alignment: Alignment.center,
        ),
        Container(
          padding: const EdgeInsets.symmetric(vertical: 16.0),
          alignment: Alignment.center,
          child: Row(
            mainAxisAlignment: MainAxisAlignment.center,
            children: <Widget>[
              Radio<int>(
                value: 0,
                groupValue: _selection,
                onChanged: _handleRadioButtonSelected,
              ),
              const Text(
                'Github',
                style: TextStyle(fontSize: 16.0),
              ),
              Radio<int>(
                value: 1,
                groupValue: _selection,
                onChanged: _handleRadioButtonSelected,
              ),
              const Text(
                'Facebook',
                style: TextStyle(
                  fontSize: 16.0,
                ),
              ),
              Radio<int>(
                value: 2,
                groupValue: _selection,
                onChanged: _handleRadioButtonSelected,
              ),
              const Text(
                'Twitter',
                style: TextStyle(fontSize: 16.0),
              ),
            ],
          ),
          ),
          TextField(
            controller: _tokenController,
            decoration:
                const InputDecoration(labelText: 'Enter provider\'s token'),
          ),
          Container(
            child: _showAuthSecretTextField
                ? TextField(
                    controller: _tokenSecretController,
                    decoration: const InputDecoration(
                        labelText: 'Enter provider\'s authTokenSecret'),
                  )
                : null,
          ),
          Container(
            padding: const EdgeInsets.symmetric(vertical: 16.0),
            alignment: Alignment.center,
            child: RaisedButton(
              onPressed: () async {
                _signInWithOtherProvider();
              },
              child: const Text('Sign in'),
            ),
          ),
          Container(
            alignment: Alignment.center,
            padding: const EdgeInsets.symmetric(horizontal: 16),
            child: Text(
              _message,
              style: TextStyle(color: Colors.red),
            ),
          )
        ],
      );
    }

    void _handleRadioButtonSelected(int value) {
      setState(() {
        _selection = value;
        if (_selection == 2) {
          _showAuthSecretTextField = true;
        } else {
          _showAuthSecretTextField = false;
        }
      });
    }

    void _signInWithOtherProvider() {
      switch (_selection) {
        case 0:
          _signInWithGithub();
          break;
        case 1:
          _signInWithFacebook();
          break;
```

```dart
        case 2:
          _signInWithTwitter();
          break;
        default:
      }
    }

    // Example code of how to sign in with Github.
    void _signInWithGithub() async {
      final AuthCredential credential = GithubAuthProvider.getCredential(
        token: _tokenController.text,
      );
      final FirebaseUser user =
          (await _auth.signInWithCredential(credential)).user;
      assert(user.email != null);
      assert(user.displayName != null);
      assert(!user.isAnonymous);
      assert(await user.getIdToken() != null);

      final FirebaseUser currentUser = await _auth.currentUser();
      assert(user.uid == currentUser.uid);
      setState(() {
        if (user != null) {
          _message = 'Successfully signed in with Github. ' + user.uid;
        } else {
          _message = 'Failed to sign in with Github. ';
        }
      });
    }

    // Example code of how to sign in with Facebook.
    void _signInWithFacebook() async {
      final AuthCredential credential = FacebookAuthProvider.getCredential(
        accessToken: _tokenController.text,
      );
      final FirebaseUser user =
          (await _auth.signInWithCredential(credential)).user;
      assert(user.email != null);
      assert(user.displayName != null);
      assert(!user.isAnonymous);
      assert(await user.getIdToken() != null);

      final FirebaseUser currentUser = await _auth.currentUser();
      assert(user.uid == currentUser.uid);
      setState(() {
        if (user != null) {
          _message = 'Successfully signed in with Facebook. ' + user.uid;
        } else {
          _message = 'Failed to sign in with Facebook. ';
        }
      });
    }

    // Example code of how to sign in with Twitter.
    void _signInWithTwitter() async {
      final AuthCredential credential = TwitterAuthProvider.getCredential(
          authToken: _tokenController.text,
          authTokenSecret: _tokenSecretController.text);
      final FirebaseUser user =
          (await _auth.signInWithCredential(credential)).user;
      assert(user.email != null);
      assert(user.displayName != null);
      assert(!user.isAnonymous);
      assert(await user.getIdToken() != null);

      final FirebaseUser currentUser = await _auth.currentUser();
      assert(user.uid == currentUser.uid);
      setState(() {
        if (user != null) {
          _message = 'Successfully signed in with Twitter. ' + user.uid;
        } else {
          _message = 'Failed to sign in with Twitter. ';
        }
      });
    }
  }
```