

🔗 master ▾

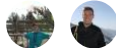
...

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

```

1 // Copyright 2019 The Chromium Authors. All rights reserved.
2 // Use of this source code is governed by a BSD-style license that can be
3 // found in the LICENSE file.
4
5 import 'dart:io';
6 import 'package:flutter/material.dart';
7 import 'package:firebase_auth/firebase_auth.dart';
8 import 'package:google_sign_in/google_sign_in.dart';
9 import 'package:firebase_dynamic_links/firebase_dynamic_links.dart';
10
11 final FirebaseAuth _auth = FirebaseAuth.instance;
12 final GoogleSignIn _googleSignIn = GoogleSignIn();
13
14 class SignInPage extends StatefulWidget {
15   final String title = 'Registration';
16   @override
17   State<SignInPage> createState() => SignInPageState();
18 }
19
20 class SignInPageState extends State<SignInPage> {
21   @override
22   Widget build(BuildContext context) {
23     return Scaffold(
24       appBar: AppBar(
25         title: Text(widget.title),
26         actions: <Widget>[
27           Builder(builder: (BuildContext context) {
28             return FlatButton(
29               child: const Text('Sign out'),
30               textColor: Theme.of(context).buttonColor,
31               onPressed: () async {
32                 final FirebaseUser user = await _auth.currentUser();
33                 if (user == null) {
34                   Scaffold.of(context).showSnackBar(const SnackBar(

```

```

35         content: Text('No one has signed in.'),
36     ));
37     return;
38 }
39 _signOut();
40 final String uid = user.uid;
41 Scaffold.of(context).showSnackBar(SnackBar(
42     content: Text(uid + ' has successfully signed out.'),
43 ));
44 },
45 );
46 })
47 ],
48 ),
49 body: Builder(builder: (BuildContext context) {
50     return ListView(
51         scrollDirection: Axis.vertical,
52         children: <Widget>[
53             _EmailPasswordForm(),
54             _EmailLinkSignInSection(),
55             _AnonymouslySignInSection(),
56             _GoogleSignInSection(),
57             _PhoneSignInSection(Scaffold.of(context)),
58             _OtherProvidersSignInSection(),
59         ],
60     );
61     }),
62 );
63 }
64
65 // Example code for sign out.
66 void _signOut() async {
67     await _auth.signOut();
68 }
69 }
70
71 class _EmailPasswordForm extends StatefulWidget {
72     @override
73     State<StatefulWidget> createState() => _EmailPasswordFormState();
74 }
75
76 class _EmailPasswordFormState extends State<_EmailPasswordForm> {
77     final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
78     final TextEditingController _emailController = TextEditingController();
79     final TextEditingController _passwordController = TextEditingController();
80     bool _success;
81     String _userEmail;
82     @override
83     Widget build(BuildContext context) {
84         return Form(
85             key: _formKey,
86             child: Column(
87                 crossAxisAlignment: CrossAxisAlignment.start,
88                 children: <Widget>[
89                     Container(
90                         child: const Text('Test sign in with email and password'),

```

```

91         padding: const EdgeInsets.all(16),
92         alignment: Alignment.center,
93     ),
94     TextFormField(
95       controller: _emailController,
96       decoration: const InputDecoration(labelText: 'Email'),
97       validator: (String value) {
98         if (value.isEmpty) {
99           return 'Please enter some text';
100         }
101         return null;
102       },
103     ),
104     TextFormField(
105       controller: _passwordController,
106       decoration: const InputDecoration(labelText: 'Password'),
107       validator: (String value) {
108         if (value.isEmpty) {
109           return 'Please enter some text';
110         }
111         return null;
112       },
113     ),
114     Container(
115       padding: const EdgeInsets.symmetric(vertical: 16.0),
116       alignment: Alignment.center,
117       child: RaisedButton(
118         onPressed: () async {
119           if (_formKey.currentState.validate()) {
120             _signInWithEmailAndPassword();
121           }
122         },
123         child: const Text('Submit'),
124       ),
125     ),
126     Container(
127       alignment: Alignment.center,
128       padding: const EdgeInsets.symmetric(horizontal: 16),
129       child: Text(
130         _success == null
131           ? ''
132           : (_success
133             ? 'Successfully signed in ' + _userEmail
134             : 'Sign in failed'),
135         style: TextStyle(color: Colors.red),
136       ),
137     )
138   ],
139 ),
140 );
141 }
142
143 @override
144 void dispose() {
145   _emailController.dispose();
146   _passwordController.dispose();

```

```

147     super.dispose();
148 }
149
150 // Example code of how to sign in with email and password.
151 void _signInWithEmailAndPassword() async {
152     final FirebaseUser user = (await _auth.signInWithEmailAndPassword(
153         email: _emailController.text,
154         password: _passwordController.text,
155     ))
156         .user;
157     if (user != null) {
158         setState(() {
159             _success = true;
160             _userEmail = user.email;
161         });
162     } else {
163         _success = false;
164     }
165 }
166 }
167
168 class _EmailLinkSignInSection extends StatefulWidget {
169     @override
170     State<StatefulWidget> createState() => _EmailLinkSignInSectionState();
171 }
172
173 class _EmailLinkSignInSectionState extends State<_EmailLinkSignInSection>
174     with WidgetsBindingObserver {
175     final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
176     final TextEditingController _emailController = TextEditingController();
177
178     bool _success;
179     String _userEmail;
180     String _userID;
181
182     @override
183     void initState() {
184         super.initState();
185         WidgetsBinding.instance.addObserver(this);
186     }
187
188     @override
189     void dispose() {
190         _emailController.dispose();
191         WidgetsBinding.instance.removeObserver(this);
192         super.dispose();
193     }
194
195     @override
196     void didChangeAppLifecycleState(AppLifecycleState state) async {
197         if (state == AppLifecycleState.resumed) {
198             final PendingDynamicLinkData data =
199                 await FirebaseDynamicLinks.instance.getInitialLink();
200             if (data?.link != null) {
201                 handleLink(data?.link);
202             }

```

```

203
204     FirebaseDynamicLinks.instance.onLink(
205         onSuccess: (PendingDynamicLinkData dynamicLink) async {
206             final Uri deepLink = dynamicLink?.link;
207
208             handleLink(deepLink);
209         }, onError: (OnLinkErrorException e) async {
210             print('onLinkError');
211             print(e.message);
212         });
213     }
214 }
215
216 void handleLink(Uri link) async {
217     if (link != null) {
218         final FirebaseUser user = (await _auth.signInWithEmailAndLink(
219             email: _userEmail,
220             link: link.toString(),
221         ))
222             .user;
223
224         if (user != null) {
225             _userID = user.uid;
226             _success = true;
227         } else {
228             _success = false;
229         }
230     } else {
231         _success = false;
232     }
233
234     setState(() {});
235 }
236
237 @override
238 Widget build(BuildContext context) {
239     return Form(
240         key: _formKey,
241         child: Column(
242             crossAxisAlignment: CrossAxisAlignment.start,
243             children: <Widget>[
244                 Container(
245                     child: const Text('Test sign in with email and link'),
246                     padding: const EdgeInsets.all(16),
247                     alignment: Alignment.center,
248                 ),
249                 TextFormField(
250                     controller: _emailController,
251                     decoration: const InputDecoration(labelText: 'Email'),
252                     validator: (String value) {
253                         if (value.isEmpty) {
254                             return 'Please enter your email.';
255                         }
256                         return null;
257                     },
258                 ),

```

```

259     Container(
260       padding: const EdgeInsets.symmetric(vertical: 16.0),
261       alignment: Alignment.center,
262       child: RaisedButton(
263         onPressed: () async {
264           if (_formKey.currentState.validate()) {
265             _signInWithEmailAndLink();
266           }
267         },
268         child: const Text('Submit'),
269       ),
270     ),
271     Container(
272       alignment: Alignment.center,
273       padding: const EdgeInsets.symmetric(horizontal: 16),
274       child: Text(
275         _success == null
276           ? ''
277           : (_success
278             ? 'Successfully signed in, uid: ' + _userID
279             : 'Sign in failed'),
280         style: TextStyle(color: Colors.red),
281       ),
282     )
283   ],
284 ),
285 );
286 }
287
288 Future<void> _signInWithEmailAndLink() async {
289   _userEmail = _emailController.text;
290
291   return await _auth.sendSignInWithEmailLink(
292     email: _userEmail,
293     url: '<Url with domain from your Firebase project>',
294     handleCodeInApp: true,
295     iOSBundleID: 'io.flutter.plugins.firebaseAuthExample',
296     androidPackageName: 'io.flutter.plugins.firebaseauthexample',
297     androidInstallIfNotAvailable: true,
298     androidMinimumVersion: "1",
299   );
300 }
301 }
302
303 class _AnonymouslySignInSection extends StatefulWidget {
304   @override
305   State<StatefulWidget> createState() => _AnonymouslySignInSectionState();
306 }
307
308 class _AnonymouslySignInSectionState extends State<_AnonymouslySignInSection> {
309   bool _success;
310   String _userID;
311   @override
312   Widget build(BuildContext context) {
313     return Column(
314       crossAxisAlignment: CrossAxisAlignment.start,

```

```

315     children: <Widget>[
316       Container(
317         child: const Text('Test sign in anonymously'),
318         padding: const EdgeInsets.all(16),
319         alignment: Alignment.center,
320       ),
321       Container(
322         padding: const EdgeInsets.symmetric(vertical: 16.0),
323         alignment: Alignment.center,
324         child: RaisedButton(
325           onPressed: () async {
326             _signInAnonymously();
327           },
328           child: const Text('Sign in anonymously'),
329         ),
330       ),
331       Container(
332         alignment: Alignment.center,
333         padding: const EdgeInsets.symmetric(horizontal: 16),
334         child: Text(
335           _success == null
336             ? ''
337             : (_success
338               ? 'Successfully signed in, uid: ' + _userID
339               : 'Sign in failed'),
340           style: TextStyle(color: Colors.red),
341         ),
342       )
343     ],
344   );
345 }
346
347 // Example code of how to sign in anonymously.
348 void _signInAnonymously() async {
349   final FirebaseUser user = (await _auth.signInAnonymously()).user;
350   assert(user != null);
351   assert(user.isAnonymous);
352   assert(!user.isEmailVerified);
353   assert(await user.getIdToken() != null);
354   if (Platform.isIOS) {
355     // Anonymous auth doesn't show up as a provider on iOS
356     assert(user.providerData.isEmpty);
357   } else if (Platform.isAndroid) {
358     // Anonymous auth does show up as a provider on Android
359     assert(user.providerData.length == 1);
360     assert(user.providerData[0].providerId == 'firebase');
361     assert(user.providerData[0].uid != null);
362     assert(user.providerData[0].displayName == null);
363     assert(user.providerData[0].photoUrl == null);
364     assert(user.providerData[0].email == null);
365   }
366
367   final FirebaseUser currentUser = await _auth.currentUser();
368   assert(user.uid == currentUser.uid);
369   setState(() {
370     if (user != null) {

```

```

371     _success = true;
372     _userID = user.uid;
373   } else {
374     _success = false;
375   }
376   });
377 }
378 }
379
380 class _GoogleSignInSection extends StatefulWidget {
381   @override
382   State<StatefulWidget> createState() => _GoogleSignInSectionState();
383 }
384
385 class _GoogleSignInSectionState extends State<_GoogleSignInSection> {
386   bool _success;
387   String _userID;
388   @override
389   Widget build(BuildContext context) {
390     return Column(
391       children: <Widget>[
392         Container(
393           child: const Text('Test sign in with Google'),
394           padding: const EdgeInsets.all(16),
395           alignment: Alignment.center,
396         ),
397         Container(
398           padding: const EdgeInsets.symmetric(vertical: 16.0),
399           alignment: Alignment.center,
400           child: RaisedButton(
401             onPressed: () async {
402               _signInWithGoogle();
403             },
404             child: const Text('Sign in with Google'),
405           ),
406         ),
407         Container(
408           alignment: Alignment.center,
409           padding: const EdgeInsets.symmetric(horizontal: 16),
410           child: Text(
411             _success == null
412               ? ''
413               : (_success
414                 ? 'Successfully signed in, uid: ' + _userID
415                 : 'Sign in failed'),
416             style: TextStyle(color: Colors.red),
417           ),
418         )
419       ],
420     );
421   }
422
423   // Example code of how to sign in with google.
424   void _signInWithGoogle() async {
425     final GoogleSignInAccount googleUser = await _googleSignIn.signIn();
426     final GoogleSignInAuthentication googleAuth =

```



```

427     await googleUser.authentication;
428     final AuthCredential credential = GoogleAuthProvider.getCredential(
429         accessToken: googleAuth.accessToken,
430         idToken: googleAuth.idToken,
431     );
432     final FirebaseUser user =
433         (await _auth.signInWithCredential(credential)).user;
434     assert(user.email != null);
435     assert(user.displayName != null);
436     assert(!user.isAnonymous);
437     assert(await user.getIdToken() != null);
438
439     final FirebaseUser currentUser = await _auth.currentUser();
440     assert(user.uid == currentUser.uid);
441     setState(() {
442         if (user != null) {
443             _success = true;
444             _userID = user.uid;
445         } else {
446             _success = false;
447         }
448     });
449 }
450 }
451
452 class _PhoneSignInSection extends StatefulWidget {
453     _PhoneSignInSection(this._scaffold);
454
455     final ScaffoldState _scaffold;
456     @override
457     State<StatefulWidget> createState() => _PhoneSignInSectionState();
458 }
459
460 class _PhoneSignInSectionState extends State<_PhoneSignInSection> {
461     final TextEditingController _phoneNumberController = TextEditingController();
462     final TextEditingController _smsController = TextEditingController();
463
464     String _message = '';
465     String _verificationId;
466
467     @override
468     Widget build(BuildContext context) {
469         return Column(
470             crossAxisAlignment: CrossAxisAlignment.start,
471             children: <Widget>[
472                 Container(
473                     child: const Text('Test sign in with phone number'),
474                     padding: const EdgeInsets.all(16),
475                     alignment: Alignment.center,
476                 ),
477                 TextFormField(
478                     controller: _phoneNumberController,
479                     decoration: const InputDecoration(
480                         labelText: 'Phone number (+x xxx-xxx-xxxx)'),
481                     validator: (String value) {
482                         if (value.isEmpty) {

```

```

483         return 'Phone number (+x xxx-xxx-xxxx)';
484     }
485     return null;
486 },
487 ),
488 Container(
489     padding: const EdgeInsets.symmetric(vertical: 16.0),
490     alignment: Alignment.center,
491     child: RaisedButton(
492         onPressed: () async {
493             _verifyPhoneNumber();
494         },
495         child: const Text('Verify phone number'),
496     ),
497 ),
498 TextField(
499     controller: _smsController,
500     decoration: const InputDecoration(labelText: 'Verification code'),
501 ),
502 Container(
503     padding: const EdgeInsets.symmetric(vertical: 16.0),
504     alignment: Alignment.center,
505     child: RaisedButton(
506         onPressed: () async {
507             _signInWithPhoneNumber();
508         },
509         child: const Text('Sign in with phone number'),
510     ),
511 ),
512 Container(
513     alignment: Alignment.center,
514     padding: const EdgeInsets.symmetric(horizontal: 16),
515     child: Text(
516         _message,
517         style: TextStyle(color: Colors.red),
518     ),
519 )
520 ],
521 );
522 }
523
524 // Example code of how to verify phone number
525 void _verifyPhoneNumber() async {
526     setState(() {
527         _message = '';
528     });
529     final PhoneVerificationCompleted verificationCompleted =
530         (AuthCredential phoneAuthCredential) {
531             _auth.signInWithCredential(phoneAuthCredential);
532             setState(() {
533                 _message = 'Received phone auth credential: $phoneAuthCredential';
534             });
535         };
536
537     final PhoneVerificationFailed verificationFailed =
538         (AuthException authException) {

```

```

539     setState(() {
540         _message =
541             'Phone number verification failed. Code: ${authException.code}. Message: ${authException.messa
542     });
543 };
544
545 final PhoneCodeSent codeSent =
546     (String verificationId, [int forceResendingToken]) async {
547     widget._scaffold.showSnackBar(const SnackBar(
548         content: Text('Please check your phone for the verification code.'),
549     ));
550     _verificationId = verificationId;
551 };
552
553 final PhoneCodeAutoRetrievalTimeout codeAutoRetrievalTimeout =
554     (String verificationId) {
555     _verificationId = verificationId;
556 };
557
558 await _auth.verifyPhoneNumber(
559     phoneNumber: _phoneNumberController.text,
560     timeout: const Duration(seconds: 5),
561     verificationCompleted: verificationCompleted,
562     verificationFailed: verificationFailed,
563     codeSent: codeSent,
564     codeAutoRetrievalTimeout: codeAutoRetrievalTimeout);
565 }
566
567 // Example code of how to sign in with phone.
568 void _signInWithPhoneNumber() async {
569     final AuthCredential credential = PhoneAuthProvider.getCredential(
570         verificationId: _verificationId,
571         smsCode: _smsController.text,
572     );
573     final FirebaseUser user =
574         (await _auth.signInWithCredential(credential)).user;
575     final FirebaseUser currentUser = await _auth.currentUser();
576     assert(user.uid == currentUser.uid);
577     setState(() {
578         if (user != null) {
579             _message = 'Successfully signed in, uid: ' + user.uid;
580         } else {
581             _message = 'Sign in failed';
582         }
583     });
584 }
585 }
586
587 class _OtherProvidersSignInSection extends StatefulWidget {
588     _OtherProvidersSignInSection();
589
590     @override
591     State<StatefulWidget> createState() => _OtherProvidersSignInSectionState();
592 }
593
594 class _OtherProvidersSignInSectionState

```

```

595     extends State<_OtherProvidersSignInSection> {
596     final TextEditingController _tokenController = TextEditingController();
597     final TextEditingController _tokenSecretController = TextEditingController();
598
599     String _message = '';
600     int _selection = 0;
601     bool _showAuthSecretTextField = false;
602
603     @override
604     Widget build(BuildContext context) {
605       return Column(
606         crossAxisAlignment: CrossAxisAlignment.start,
607         children: <Widget>[
608           Container(
609             child: const Text(
610               'Test other providers authentication. (We do not provide an API to obtain the token for belo
611             padding: const EdgeInsets.all(16),
612             alignment: Alignment.center,
613           ),
614           Container(
615             padding: const EdgeInsets.symmetric(vertical: 16.0),
616             alignment: Alignment.center,
617             child: Row(
618               mainAxisAlignment: MainAxisAlignment.center,
619               children: <Widget>[
620                 Radio<int>(
621                   value: 0,
622                   groupValue: _selection,
623                   onChanged: _handleRadioButtonSelected,
624                 ),
625                 const Text(
626                   'Github',
627                   style: TextStyle(fontSize: 16.0),
628                 ),
629                 Radio<int>(
630                   value: 1,
631                   groupValue: _selection,
632                   onChanged: _handleRadioButtonSelected,
633                 ),
634                 const Text(
635                   'Facebook',
636                   style: TextStyle(
637                     fontSize: 16.0,
638                   ),
639                 ),
640                 Radio<int>(
641                   value: 2,
642                   groupValue: _selection,
643                   onChanged: _handleRadioButtonSelected,
644                 ),
645                 const Text(
646                   'Twitter',
647                   style: TextStyle(fontSize: 16.0),
648                 ),
649               ],
650             ),

```

```

651     ),
652     TextField(
653       controller: _tokenController,
654       decoration:
655         const InputDecoration(labelText: 'Enter provider\'s token'),
656     ),
657     Container(
658       child: _showAuthSecretTextField
659         ? TextField(
660           controller: _tokenSecretController,
661           decoration: const InputDecoration(
662             labelText: 'Enter provider\'s authTokenSecret'),
663           )
664         : null,
665     ),
666     Container(
667       padding: const EdgeInsets.symmetric(vertical: 16.0),
668       alignment: Alignment.center,
669       child: RaisedButton(
670         onPressed: () async {
671           _signInWithOtherProvider();
672         },
673         child: const Text('Sign in'),
674       ),
675     ),
676     Container(
677       alignment: Alignment.center,
678       padding: const EdgeInsets.symmetric(horizontal: 16),
679       child: Text(
680         _message,
681         style: TextStyle(color: Colors.red),
682       ),
683     )
684   ],
685 );
686 }
687
688 void _handleRadioButtonSelected(int value) {
689   setState(() {
690     _selection = value;
691     if (_selection == 2) {
692       _showAuthSecretTextField = true;
693     } else {
694       _showAuthSecretTextField = false;
695     }
696   });
697 }
698
699 void _signInWithOtherProvider() {
700   switch (_selection) {
701     case 0:
702       _signInWithGithub();
703       break;
704     case 1:
705       _signInWithFacebook();
706       break;

```

```

707         case 2:
708             _signInWithTwitter();
709             break;
710         default:
711     }
712 }
713
714 // Example code of how to sign in with Github.
715 void _signInWithGithub() async {
716     final AuthCredential credential = GithubAuthProvider.getCredential(
717         token: _tokenController.text,
718     );
719     final FirebaseUser user =
720         (await _auth.signInWithCredential(credential)).user;
721     assert(user.email != null);
722     assert(user.displayName != null);
723     assert(!user.isAnonymous);
724     assert(await user.getIdToken() != null);
725
726     final FirebaseUser currentUser = await _auth.currentUser();
727     assert(user.uid == currentUser.uid);
728     setState(() {
729         if (user != null) {
730             _message = 'Successfully signed in with Github. ' + user.uid;
731         } else {
732             _message = 'Failed to sign in with Github. ';
733         }
734     });
735 }
736
737 // Example code of how to sign in with Facebook.
738 void _signInWithFacebook() async {
739     final AuthCredential credential = FacebookAuthProvider.getCredential(
740         accessToken: _tokenController.text,
741     );
742     final FirebaseUser user =
743         (await _auth.signInWithCredential(credential)).user;
744     assert(user.email != null);
745     assert(user.displayName != null);
746     assert(!user.isAnonymous);
747     assert(await user.getIdToken() != null);
748
749     final FirebaseUser currentUser = await _auth.currentUser();
750     assert(user.uid == currentUser.uid);
751     setState(() {
752         if (user != null) {
753             _message = 'Successfully signed in with Facebook. ' + user.uid;
754         } else {
755             _message = 'Failed to sign in with Facebook. ';
756         }
757     });
758 }
759
760 // Example code of how to sign in with Twitter.
761 void _signInWithTwitter() async {
762     final AuthCredential credential = TwitterAuthProvider.getCredential(

```

```
763     authToken: _tokenController.text,  
764     authTokenSecret: _tokenSecretController.text);  
765   final FirebaseUser user =  
766     (await _auth.signInWithCredential(credential)).user;  
767   assert(user.email != null);  
768   assert(user.displayName != null);  
769   assert(!user.isAnonymous);  
770   assert(await user.getIdToken() != null);  
771  
772   final FirebaseUser currentUser = await _auth.currentUser();  
773   assert(user.uid == currentUser.uid);  
774   setState(() {  
775     if (user != null) {  
776       _message = 'Successfully signed in with Twitter. ' + user.uid;  
777     } else {  
778       _message = 'Failed to sign in with Twitter. ';  
779     }  
780   });  
781 }  
782 }
```