Convert promise to observable

133

19

I am trying to wrap my head around observables. I love the way observables solve development and readability issues. As I read, benefits are immense.

Observables on HTTP and collections seem to be straight forward. How can I convert something like this to observable pattern.

This is from my service component, to provide authentication. I'd prefer this to work like other HTTP services in Angular2 - with support for data, error and completion handlers.

```
firebase.auth().createUserWithEmailAndPassword(email, password)
        .then(function(firebaseUser) {
             // do something to update your UI component
             // pass user object to UI component
        })
        .catch(function(error) {
             // Handle Errors here.
             var errorCode = error.code;
             var errorMessage = error.message;
             // ...
        });
```

Any help here would be much appreciated. The only alternative solution I had was to create `EventEmitter` s. But I guess that's a terrible way to do things in services section

angular firebase firebase-authentication

edited Nov 29 '18 at 12:07

Ayman Nedjmeddine
3,072 1 12 27

asked Sep 4 '16 at 16:49

Krishnan Sriram
940 3 7 14

4 Answers

181

```
import { from } from 'rxjs';
const observable = from(promise);
```

edited Nov 30 '18 at 6:01
Simon_Weaver
77.8k 65 481 544

answered Jun 11 '18 at 13:38
Guillaume
1,880 1 5 8

3 Using 6.3.3, `from` method returning observable but it is sending promise as value to subscriptions. :(– Laxmikant Dange Dec 17 '18 at 7:04

@LaxmikantDange No, you get the resolved value in `subscribe` . – fridoo Jan 19 at 0:31

try this:

107

```
var subscription = Observable.fromPromise(
    firebase.auth().createUserWithEmailAndPassword(email, password)
);
subscription.subscribe(firebaseUser => /* Do anything with data received */,
                       error => /* Handle error here */);
```

you can find complete reference to fromPromise operator here

edited Apr 6 '18 at 8:33
user3336882
1,433 1 8 11

answered Sep 4 '16 at 16:53
Godfather
3,718 3 16 25

44 `import 'rxjs/add/observable/fromPromise';` – Simon Briggs May 24 '17 at 2:32

15 `import { Observable } from "rxjs/Observable";` :) – Luckylooke Sep 25 '17 at 6:56

from

```
import { from } from 'rxjs';

const observable = from(promise);
```

As the inner Promise has already been created (or will be created at the same time as the Observable) the Promise's body is being executed or has already been resolved as the Observable is created. If the inner Promise has already resolved a new Subscriber to the Observable will get its value immediately.

defer

Use `defer` with a Promise factory function as input to defer the creation and conversion of a Promise to an Observable.

```
import { defer } from 'rxjs';

// getPromise has no parameters and returns a Promise
const observable = defer(getPromise)

// getPromise has parameters and returns a Promise
const observable = defer(() => getPromise(myParameters));
```

The difference to `from` is that `defer` waits for a subscriber and only then creates a new Promise by calling the given Promise factory function. This is useful when you want to create an Observable but don't want the inner Promise to be executed right away. The inner Promise will only be executed when someone subscribes to the Observable. Each subscriber will also get its own new Observable.

The difference between `from` and `defer` in an example: https://stackblitz.com/edit/rxjs-yye14a

edited Apr 25 at 21:39

answered Jan 9 at 13:16

fridoo
2,128 3 12 24

0

You can also use a **Subject** and trigger its **next()** function from promise. See sample below:

Add code like below (I used service)

```
    if (this.createUserSubject) {
      return this.createUserSubject;
    } else {
      this.createUserSubject = new Subject < any > ();
      firebase.auth().createUserWithEmailAndPassword(email,
          password)
        .then(function(firebaseUser) {
          // do something to update your UI component
          // pass user object to UI component
          this.createUserSubject.next(firebaseUser);
        })
        .catch(function(error) {
          // Handle Errors here.
          var errorCode = error.code;
          var errorMessage = error.message;
          this.createUserSubject.error(error);
          // ...
        });
    }

  }
}
```

Run code snippet Expand snippet

Create User From Component like below

```
class UserComponent {
  constructor(private userService: UserService) {
    this.userService.createUserWithEmailAndPassword().subscribe(user =>
console.log(user), error => console.log(error);
    }
  }
```

Run code snippet Expand snippet

Subjects are low-level machinery. Don't use subjects, except for the cases when you're extending `rxjs` . – polkovnikov.ph Dec 10 '18 at 18:52

I am just giving a solution. – Shivang Gupta Dec 12 '18 at 17:23

You could have at least shown `new Observable(observer => { ... observer.next() ... })` way to implement it. Even though it would be a reimplementation of existing well-known function, it would directly answer the question and wouldn't be harmful to readers. – polkovnikov.ph Dec 13 '18 at 0:41