Fix your RxJS 6 Imports & Operators - What's ...

Content Information

Created by Maximilian Schwarzmüller

Last updated on 3. May 2018

Newsletter email REGISTER

♥ FEEDBACK

RxJS 6 Changes - Overview

** Watch the video above this article for a detailed walkthrough!**

RxJS 6 (or higher) introduces two important changes compared to RxJS version 5:

1. Different internal structure that **requires you to change your `import` statements**
2. `pipe()` as a method to chain your **operators**, the **old way of chaining them will not work**

Additionally, some operators were renamed.

Sounds horrible, right?

Thankfully, you can make your old code work again by running just one command in your project folder:

```
npm install --save rxjs-compat
```

This command will install a package that ensures backward-compatibility of RxJS. Even though the `rxjs` package itself would not work with your current code, with the addition of `rxjs-compat` it will.

Behind the scenes, `rxjs` still supports your old import

How do you properly update?

# Import Statement Update Path

With the release of version 6, RxJS changed its internal package structure. It did so to ensure that less code needs to be imported and that produced app bundles therefore are smaller. So this definitely is a good change.

For the import statements, here's how you have to update your code:

Observable, Subject etc.

```
import { Observable } from 'rxjs/Observable';
import { Subject } from 'rxjs/Subject';
```

## Operators

```
import 'rxjs/add/operator/map';
import 'rxjs/add/operator/take';
```

becomes

```
import { map, take } from 'rxjs/operators';
```

Methods to Create Observables

```
import 'rxjs/add/observable/of';
// or
import { of } from 'rxjs/observable/of';
```

becomes

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

# Operators Update Path

Operators like `map` are now used differently.

In the past, you imported and used operators like this:

```
import { map } from 'rxjs/operators';
```

`map` is a function and it does exactly the same as the `map` method that was patched into the `Observable` prototype by the old import.

But the `map` function alone doesn't help you that much, you still need a way to connect it to your observable.

Using Operators in RxJS 6

You use the newly introduced `pipe()` method for this (it was actually already added in RxJS 5.5).

```
import { map } from 'rxjs/operators';

myObservable
  .pipe(map(data => data * 2))
  .subscribe(...);
```

`pipe` takes an infinite amount of arguments and each argument is an operator you want to apply to the `Observable`.

So you could have a chain like this:

Needless to say that operators execute in the order you pass them into the `pipe()` method - left to right.

# Renamed Operators

You know the core adjustments you need to make: **Different import paths** and **different operator usage**.

There's one more important piece of information though: Some operators were renamed.

This was necessary because of the new way you use them. For example, `catch` was fine as a chainable method name in RxJS 5.5 and lower but as a standalone function, you can't use it. Why? Because `catch` is a reserved keyword in JavaScript.

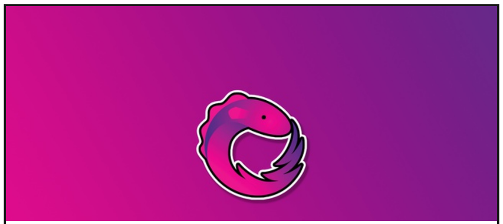
The following operators were renamed

- `catch()` => `catchError()`
- `do()` => `tap()`
- `finally()` => `finalize()`

- `throw()` => `throwError()`
- `fromPromise()` => `from()` (this automatically detects the type)

WE GOT MORE HELPFUL CONTENT!

Use RxJS 6 with Angular



Learn RxJS from Scratch