What is pipe() function in Angular 2?

75

18

Pipes are filters for transforming data (formats) in the template.

I came across the `pipe()` function as below. What does this `pipe()` function exactly mean in this case?

```
return this.http.get<Hero>(url)
  .pipe(
    tap(_ => this.log(`fetched hero id=${id}`)),
    catchError(this.handleError<Hero>(`getHero id=${id}`))
);
```

angular rxjs

edited Dec 20 '18 at 6:47
Sunil Garg
4,909 8 55 85

asked Dec 30 '17 at 1:44
Dinesh Sharma
407 1 4 8

2 github.com/ReactiveX/rxjs/blob/master/doc/lettable-operators.md – cartant Dec 30 '17 at 1:49

3 @Ajay I get this page and a bunch of references to | uses. Which doesn't answer what rxjs pipes are. – 182764125216 May 31 '18 at 22:35

4 Answers

74

Don't get confused with the concepts of Angular and RxJS

We have pipes concept in Angular and `pipes()` function in RxJS

1) **Pipes in Angular**: A pipe takes in data as input and transforms it to the desired output

The `pipe()` function takes as its arguments the functions you want to combine, and returns a new function that, when executed, runs the composed functions in sequence.
https://angular.io/guide/rx-library (search for pipes in this URL, you can find the same)

So according to your question, you are referring pipe() function in RxJS

edited Dec 20 '18 at 6:55

Sunil Garg
4,909 8 55 85

answered Jul 3 '18 at 7:24

Shiva Nayak Dharavath
825 5 14

39

The Pipes you are talking about in the starting description are different from the pipe you showed in the example.

In Angular(2|4|5) Pipes are used to format view as you said. I think you have a basic understanding of pipes in Angular, you can learn more about that from this link - Angular Pipe Doc

The `pipe()` you have shown in the example is the `pipe()` method of **RxJS 5.5** (RxJS is the default for all Angular apps). In **Angular5** all the **RxJS** operators can be imported using single import and they are now combined using the pipe method.

`tap()` - RxJS tap operator will look at the Observable value and do something with that value. In other words, after a successful API request, the `tap()` operator will do any function you want it to perform with the response. In the example, it will just log that string.

`catchError()` - catchError does exactly the same thing but with error response. If you want to throw an error or want to call some function if you get an error, you can do it here. In the example, it will call `handleError()` and inside that, it will just log that string.

edited Dec 20 '18 at 6:49

Sunil Garg
4,909 8 55 85

answered Dec 30 '17 at 5:53

BhargavG
528 1 4 12

"the pipes you are talking about in the starting description..." –> no they are not different. ; In my opinion his question was perfectly clear (no confusion what so ever possible). There are a lot of concepts in programming that could be called "pipes", but by being very specific in his description, and calling them "pipe functions" he eliminated all possible confusion. I woudln't know how else he could have called them. – bvdb Apr 19 at 13:45

1 "Pipes are filters for transforming data (formats) in the template." Here he was talking about pipe in Angular 2+, like date, uppercase pipes provided in Angular(which exactly do what he said i.e. format data in the template) And in description he has shown example of RXJS pipe **function**. So yeah those

5

RxJS Operators are functions that build on the observables foundation to enable sophisticated manipulation of collections.

For example, RxJS defines operators such as `map()`, `filter()`, `concat()`, and `flatMap()`.

You can use pipes to link operators together. Pipes let you combine multiple functions into a single function.

The `pipe()` function takes as its arguments the functions you want to combine, and returns a new function that, when executed, runs the composed functions in sequence.

edited Dec 20 '18 at 6:51

Sunil Garg
4,909 8 55 85

answered Aug 21 '18 at 4:03

manoj
51 1 2

Do you have an example? – lofihelsinki Aug 21 '18 at 4:23

In the below example we have piped the filter and map function. Now both function will get executed sequentially as provided in example. First it will filter the result and then it will map the results. Hope it will help. import { filter, map } from 'rxjs/operators'; const squareOdd = of(1, 2, 3, 4, 5) .pipe(filter(n => n % 2 !== 0), map(n => n * n)); // Subscribe to get values squareOdd.subscribe(x => console.log(x)); – manoj Aug 22 '18 at 6:11

3

You have to look to official ReactiveX documentation: https://github.com/ReactiveX/rxjs/blob/master/doc/pipeable-operators.md.

This is a good article about piping in RxJS: https://blog.hackages.io/rxjs-5-5-piping-all-the-things-9d469d1b3f44.

In short .pipe() allows chaining multiple pipeable operators.

Starting in version 5.5 RxJS has shipped "pipeable operators" and renamed some operators:

```
do -> tap
catch -> catchError
switch -> switchAll
finally -> finalize
```