How do I call an Angular 2 pipe with multiple arguments?

Asked 3 years, 4 months ago Active 7 days ago Viewed 123k times

146

I know I can call a pipe like this:

```
{{ myData | date:'fullDate' }}
```

Here the date pipe takes only one argument. What is the syntax to call a pipe with more parameters, from component's template HTML and directly in code?

25

javascript angular angular2-pipe

edited Dec 21 '16 at 19:44

asked Apr 23 '16 at 21:41

Eran Shabi
6,975 6 23 44

5 Answers

307

In your component's template you can use multiple arguments by separating them with colons:

```
{{ myData | myPipe: 'arg1':'arg2':'arg3'... }}
```

From your code it will look like this:

```
new MyPipe().transform(myData, arg1, arg2, arg3)
```

And in your transform function inside your pipe you can use the arguments like this:

```
    // or use a rest parameter
    transform(value: any, ...args: any[]): any { }
}
```

Beta 16 and before (2016-04-26)

Pipes take an array that contains all arguments, so you need to call them like this:

```
new MyPipe().transform(myData, [arg1, arg2, arg3...])
```

And your transform function will look like this:

```
export class MyPipe implements PipeTransform {
    transform(value:any, args:any[]):any {
        var arg1 = args[0];
        var arg2 = args[1];
        ...
    }
}
```

edited Aug 29 at 14:34 — fridoo — 2,632 3 13 25

answered Apr 23 '16 at 21:41 — Eran Shabi — 6,975 6 23 44

5 This design is silly. I need to check document every time I come across this issue – tom10271 Nov 27 '17 at 2:24

What would the template bit look like if `arg1` and `arg2` where both optional and you only wanted to pass in `arg2` ? – freethebees Mar 12 '18 at 9:14

if you pass `undefined` as the first argument it will get its default value. – Eran Shabi Mar 12 '18 at 14:13

1 nowadays instead of `transform(value:any, arg1:any, arg2:any, arg3:any)` using the rest operator feels better I think: `transform(value:any, ...args:any[])` – mkb Mar 30 at 20:37

34

You're missing the actual pipe.

```
{{ myData | date:'fullDate' }}
```

```
{{ myData | myPipe:'arg1':'arg2':'arg3' }}
```

Also you can chain pipes, like so:

```
{{ myData | date:'fullDate' | myPipe:'arg1':'arg2':'arg3' }}
```

edited Jan 10 '18 at 14:09

Victor ifeanyi
517 5 14

answered Apr 24 '16 at 3:22

Eugene
810 6 7

22

Since beta.16 the parameters are not passed as array to the `transform()` method anymore but instead as individual parameters:

```
{{ myData | date:'fullDate':'arg1':'arg2' }}
```

```
export class DatePipe implements PipeTransform {
  transform(value:any, arg1:any, arg2:any):any {
        ...
}
```

https://github.com/angular/angular/blob/master/CHANGELOG.md#200-beta16-2016-04-26

> pipes now take a variable number of arguments, and not an array that contains all arguments.

answered Apr 27 '16 at 13:53

Günter Zöchbauer
366k 83 1167 1056

What would the template bit look like if `arg1` and `arg2` where both optional and you only wanted to pass in `arg2` ? – freethebees Mar 12 '18 at 9:14

Can we use different variable names other than `arg1` ? Like `isFullDate` . I am just asking because every example uses this. – sabithpocker Mar 12 '18 at 16:25

`'arg1'` and `'arg2'` are just string literals passed as additional parameters to the pipe. You can use any value or reference that is available at that

transform method doesn't support array args good point @Gunter – BALS Oct 31 '18 at 16:07

5

I use Pipes in Angular 2+ to filter arrays of objects. The following takes multiple filter arguments but you can send just one if that suits your needs. Here is a StackBlitz Example. It will find the keys you want to filter by and then filters by the value you supply. It's actually quite simple, if it sounds complicated it's not, check out the StackBlitz Example.

Here is the Pipe being called in an *ngFor directive,

```
<div *ngFor='let item of items | filtermulti: [{title:"mr"},{last:"jacobs"}]' >
  Hello {{item.first}} !
</div>
```

Here is the Pipe,

```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
  name: 'filtermulti'
})
export class FiltermultiPipe implements PipeTransform {
  transform(myobjects: Array<object>, args?: Array<object>): any {
    if (args && Array.isArray(myobjects)) {
      // copy all objects of original array into new array of objects
      var returnobjects = myobjects;
      // args are the compare oprators provided in the *ngFor directive
      args.forEach(function (filterobj) {
        let filterkey = Object.keys(filterobj)[0];
        let filtervalue = filterobj[filterkey];
        myobjects.forEach(function (objectToFilter) {
          if (objectToFilter[filterkey] != filtervalue && filtervalue != "") {
            // object didn't match a filter value so remove it from array via filter
            returnobjects = returnobjects.filter(obj => obj !== objectToFilter);
          }
        })
      });
      // return new array of objects to *ngFor directive
      return returnobjects;
    }
  }
```

And here is the Component containing the object to filter,

```
import { Component } from '@angular/core';
import { FiltermultiPipe } from './pipes/filtermulti.pipe';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'app';
  items = [{ title: "mr", first: "john", last: "jones" }
   ,{ title: "mr", first: "adrian", last: "jacobs" }
   ,{ title: "mr", first: "lou", last: "jones" }
   ,{ title: "ms", first: "linda", last: "hamilton" }
  ];
}
```

StackBlitz Example

GitHub Example: Fork a working copy of this example here

***Please note** that in an answer provided by Gunter, Gunter states that arrays are no longer used as filter interfaces but I searched the link he provides and found nothing speaking to that claim. Also, the StackBlitz example provided shows this code working as intended in Angular 6.1.9. It will work in Angular 2+.

Happy Coding :-)

edited Nov 26 '18 at 23:08

answered Feb 24 '18 at 23:09

user3777549

181 3 6

There is no point in passing an single array with multiple entries instead of passing multple parameters directly to the pipe. – BrunoJCM Jun 29 '18 at 3:30

The array contains objects. The objects can contain multiple key value pairs used to create dynamic queries where you can look for matching records using column names compared with the column's row values. You wouldn't get this level of dynamic querying passing CSV parameters. – user3777549 Jul 6 '18 at 3:09

-1

Extended from : user3777549

Multi-value filter on one set of data(reference to title key only)

HTML

```
<div *ngFor='let item of items | filtermulti: [{title:["mr","ms"]},{first:["john"]}]' >
 Hello {{item.first}} !
</div>
```

filterMultiple

```
args.forEach(function (filterobj) {
    console.log(filterobj)
    let filterkey = Object.keys(filterobj)[0];
    let filtervalue = filterobj[filterkey];
    myobjects.forEach(function (objectToFilter) {

      if (!filtervalue.some(x=>x==objectToFilter[filterkey]) && filtervalue != "") {
        // object didn't match a filter value so remove it from array via filter
        returnobjects = returnobjects.filter(obj => obj !== objectToFilter);
      }
    })
  });
```

answered May 10 at 6:25

Sharan
1 3