Angular KeyValue Pipe

Leave a Comment / 4 minutes of reading / March 9, 2023

← Async Pipe | Angular Tutorial | Using Pipes in Components/Services →

The KeyValue Pipe converts given Object or Map into an array of key-value pairs. We can use this with the ngFor to loop through the object keys. The `keyValue` accepts the one argument `compareFn`, which we can use to set the custom sort to the pipe. In this tutorial, let us learn how to make use of `KeyValue` pipe with example.

Table of Contents

How it works

Consider, that you have the following object and a map object. It has property `a`, `b` & `c`. We cannot use ngFor to iterate over it as it requires an array. This is where the `KeyValue` pipe comes into play. It will convert them to an array of key-value pair

```
1
```

```
obj = {
  c: 123,
  b: 456,
  a: 789,
};


mapObj = new Map([
   ['c', 123],
   ['b', 446],
   ['a', 789],
]);

```

We use `keyvalue` just like any other pipes in Angular and as shown below

```

obj | keyvalue

mapObj | keyvalue

```

The `keyValue` converts them and returns in the following format. each property of the object `a: 789` is converted to an object with name as `key` and value as `value` `{ key:a, value:789 }`. It creates array of such objects and returns it.

```

 obj = [
    { key:a, value:789 },
    { key:b, value:446 },
    { key:c, value:123 },
  ];

```

Now we can use the `ngFor` to loop through it and display the content.

```

<ul>
  <li *ngFor="let item of obj | keyvalue">
```

```
    {{item.key}} ---> {{item.value}}</li>
</ul>

//output

a ---> 789
b ---> 456
c ---> 123

```

Default Sorting

KeyValue pipe uses the key to sort the results array. You can see it from the above example. Even though our object was c , b & a it was sorts it as a , b , c . The keyValue pipe uses the defaultComparator to sort the result. It uses

1. Ascending Order if the keys are number
2. Alphabetical Order if keys are strings
3. if keys are are of different types. then covert them to to their string values and use Alphabetical Order
4. If key is a either Null or undefined, put then at the end of the sort.

Custom Sorting

You can customize it by providing a custom sort function (compareFn) as the first argument to the keyValue pipe

The syntax for the compareFn as shown below. It accepts first & second keyValue and must return a number. The number must be a zero if values are equivalent else either a negative number or positive number

```

compareFn (a: KeyValue, b: KeyValue) => number

```

The following are three compareFn

```
orderOriginal = (a: KeyValue<number,string>, b: KeyValue<number,string>): number =>
  return 0
}

orderbyValueAsc = (a: KeyValue<number,string>, b: KeyValue<number,string>): number
   return a.value > b.value ? -1 : (a.value > b.value) ? 0 : 1
}

orderbyValueDsc = (a: KeyValue<number,string>, b: KeyValue<number,string>): number
  return a.value > b.value ? 1 : (a.value > b.value) ? 0 : -1
}
```

```
<ul>
  <li *ngFor="let item of obj | keyvalue">
    {{item.key}} ---> {{item.value}}</li>
</ul>

//Output
a ---> 789
b ---> 456
c ---> 123


<ul>
  <li *ngFor="let item of obj | keyvalue : orderOriginal">
    {{item.key}} ---> {{item.value}}</li>
</ul>

//Output
b ---> 456
c ---> 123
a ---> 78


<ul>
  <li *ngFor="let item of obj | keyvalue : orderbyValueAsc ">
    {{item.key}} ---> {{item.value}}</li>
</ul>

//Output
```

```
a ---> 789
b ---> 456
c ---> 123


<ul>
  <li *ngFor="let item of obj | keyvalue : orderbyValueDsc ">
    {{item.key}} ---> {{item.value}}</li>
</ul>

//Output
c ---> 123
b ---> 456
a ---> 789

```

KeyValue Pipe Example

Consider the following breeds of dogs. The example sorts the list based on the number of sub breeds. The final code is as shown below.

```

breeds=
   {
     "corgi": ["cardigan"],
     "deerhound": ["scottish"],
     "bulldog": ["boston", "english", "french"],
     "mastiff": ["bull", "english", "tibetan"],
     "australian": ["shepherd"],
     "greyhound": ["italian"],
     "buhund": ["norwegian"],
     "hound": ["afghan", "basset", "blood", "english", "ibizan", "plott", "walker"],
     "bullterrier": ["staffordshire"],
   }

```

CompareFn

```

orderClause = (a: KeyValue<number,[string]>, b: KeyValue<number,[string]>): number
  return a.value.length > b.value.length ? -1 : (a.value.length > b.value.length) ? 0 : 1
}
```

```
5
```

Template

```
1
2 <ul>
3   <li *ngFor="let item of breeds | keyvalue : orderClause ">
4     {{item.key}} ---> {{item.value}}</li>
5 </ul>
6
```

The output

```
1
2 hound ---> afghan,basset,blood,english,ibizan,plott,walker
3 bulldog ---> boston,english,french
4 mastiff ---> bull,english,tibetan
5 corgi ---> cardigan
6 deerhound ---> scottish
7 australian ---> shepherd
8 greyhound ---> italian
9 buhund ---> norwegian
10 bullterrier ---> staffordshire
11
```

Reference

1. KeyValue Pipe API
2. Async Pipe API
3. KeyValue Pipe Source Code

Read More

1. Async Pipe
2. Angular Pipes