What is the difference between Subject and BehaviorSubject?

Asked 2 years, 5 months ago Active 4 months ago Viewed 56k times

172

48

I'm not clear on the difference between a Subject and a BehaviorSubject. Is it just that a BehaviorSubject has the getValue function?

rxjs

asked Apr 11 '17 at 14:12
Mike Jerred
2,964 4 15 28

3 Answers

221

✓

A BehaviorSubject holds one value. When it is subscribed it emits the value immediately. A Subject doesn't hold a value.

Subject example (with RxJS 5 API):

```
const subject = new Rx.Subject();
subject.next(1);
subject.subscribe(x => console.log(x));
```

Console output will be empty

BehaviorSubject example:

```
const subject = new Rx.BehaviorSubject();
subject.next(1);
subject.subscribe(x => console.log(x));
```

Console output: 1

- BehaviorSubject can be created with initial value: new Rx.BehaviorSubject(1)
- Consider ReplaySubject if you want the subject to hold more than one value

answered Apr 11 '17 at 16:21
ZahiC
5,157 1 13 22

9 So do you mean you have to subscribe to subject before subject.next() to for this to work? – Eric Huang Jun 28 '18 at 3:02

2 @eric for Subject, yes. That is the distinction. – onefootswill Jul 6 '18 at 23:46

7 Note that you have to pass in the first value to BehaviorSubject's constructor ;) – mrmashal Sep 23 '18 at 8:02

200

BehaviourSubject

BehaviourSubject will return the initial value or the current value on Subscription

```
var subject = new Rx.BehaviorSubject(0);  // 0 is the initial value

subject.subscribe({
  next: (v) => console.log('observerA: ' + v)  // output initial value, then new values
on `next` triggers
});

subject.next(1);  // output new value 1 for 'observer A'
subject.next(2);  // output new value 2 for 'observer A', current value 2 for 'Observer
B' on subscription

subject.subscribe({
  next: (v) => console.log('observerB: ' + v)  // output current value 2, then new
values on `next` triggers
});

subject.next(3);
```

With output:

```
observerB: 2
observerA: 3
observerB: 3
```

Subject

Subject doesnot return the current value on Subscription. It triggers only on `.next(value)` call and return/output the `value`

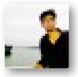
```
var subject = new Rx.Subject();

subject.next(1); //Subjects will not output this value

subject.subscribe({
  next: (v) => console.log('observerA: ' + v)
});
subject.subscribe({
  next: (v) => console.log('observerB: ' + v)
});

subject.next(2);
subject.next(3);
```

With the following output on the console:

```
observerA: 2
observerB: 2
observerA: 3
observerB: 3
```

edited Oct 21 '17 at 1:05

answered Oct 21 '17 at 0:35


Mohammed Safeer
12.5k 6 56 68

12 I prefer this answer, because of the example outputs. Thanks! – platzhersh Apr 27 '18 at 6:50

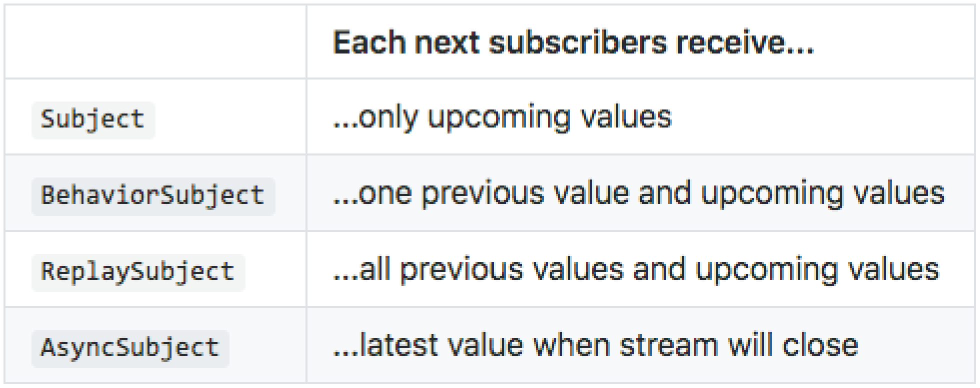
8 Its also more correct : "BehaviourSubject will return the initial value or the current value on Subscription" is a better explanation than "A BehaviorSubject holds one value." – Davy Jun 22 '18 at 13:28

8 This is much better answer as the code clearly shows the difference between Subject and BehaviorSubject – Eric Huang Jun 28 '18 at 3:14

1 This must be answer. Thanks a lot Mohammed!.. – Jack Jul 16 at 21:47

22

I just **created a project** which explain what is the **difference between all subjects**: https://github.com/piecioshka/rxjs-subject-vs-behavior-vs-replay-vs-async



	Each next subscribers receive...
`Subject`	...only upcoming values
`BehaviorSubject`	...one previous value and upcoming values
`ReplaySubject`	...all previous values and upcoming values
`AsyncSubject`	...latest value when stream will close

answered May 5 at 10:35

piecioshka
1,476 12 17