Unit Testing with Hot and Cold Observables

Rupesh Kumar Tiwari
WEB DEVELOPER

@roopkt https://rupeshtiwari.com

Module Overview

- Hot and Cold Observables
- Demo: Hot and Cold Observables
- Testing Cold Observables
- Demo: Testing Cold Observables
- Testing Hot Observables
- Summary

Hot and Cold Observables

Building Blocks of RxJS

Producer

Get values and pass to observer

Observable

Ties an observer to producer

Observer

Listens to producer

Example: RxJS Building Blocks

Package Sender

RxJS producer

Postmaster

RxJS observable

You

RxJS observer

But why should I bother about hot and cold observables ?

Hot Observable

Closes over the producer

Start emitting values regardless of any subscription

All subscribers get latest values

Usually multicast

RxJS observable

- publish and share

Examples of Hot Observables

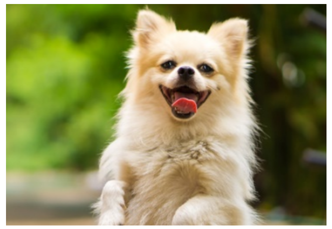
Tune radio channel

Cinema theater

Mouse clicks

Live movies

Stock tickers



Live life events

Demo

Hot observable

Cold Observable

Creates and activates producer

Start emitting values upon subscription

Subscriber gets their copy of values

Usually unicast

RxJS observable

- Of, from, interval and timer etc

Examples of Cold Observables

Movies

Watching downloaded movies

Music

Recorded podcast or song

Snaps

Snapshots movies

Demo

Cold observable

Testing Hot and Cold Observables

Frame

Jasmine-marbles converts observable sequences into frames. Frame is a JSON that consists of RxJS notification object that wraps the actual delivered value with additional metadata and message type.

```
{
"frame": 0,
"notification": {
  "error": undefined|"error“,

  "kind": "N“|“C“|“E",


  "hasValue": true|false,
  "value": "a“
  }
}
```

- Frame number
- Notification object
- Has error occurred
- Type of message (next , complete, error)
- Does sequence emit value
- Value coming from sequence

'--a--b--|'

```
[{"frame": 20, "notification": {"error": undefined, "hasValue": true, "kind": "N",
"value": "a"}},

 {"frame": 50, "notification": {"error": undefined, "hasValue": true, "kind": "N",
"value": "b"}},

{"frame": 80, "notification": {"error": undefined, "hasValue": false, "kind": "C",
"value": undefined}}]
```

of('a’)

(a|)

```
[{"frame": 0, "notification": {"error": undefined, "hasValue": true,
"kind": "N", "value": "a"}},

 {"frame": 0, "notification": {"error": undefined, "hasValue": false,
"kind": "C", "value": undefined}} ]
```

from(['a’,’b’])

(ab|)

```
[{"frame": 0, "notification": {"error": undefined, "hasValue": true, "kind": "N",
"value": "a"}},

{"frame": 0, "notification": {"error": undefined, "hasValue": true, "kind": "N", "value":
"b"}},

{"frame": 0, "notification": {"error": undefined, "hasValue": false, "kind": "C", "value":
undefined}}]
```

Demo

Testing cold observables

Testing Hot Observable

When subscription happened?

Is there any un-subscription?

How test observable will behave?

Understand subscription model

```
{
"subscribedFrame": number,
"unsubscribedFrame": number
}
```

◀ Subscription frame number

◀ Un-subscription frame number

```
'-a---^b---c---|'

    '^--------!'

    '-b---c---|'
```

Hot Observable

```
{
"subscribedFrame": 0, "unsubscribedFrame": 90
}
```

```
'-a-^(bc)--'
  '^-------'
  '-(bc)--'
```

Hot Observable

```
{
 "subscribedFrame": 0, "unsubscribedFrame": Infinity
}
```

Demo

Testing hot observables

Mocking Observable Values

```
cold('marbles', mock-object-with-values)
```

Mocking Observable

First parameter is marble diagrams

Second parameter is the mock object for the emitted values

```
cold('(x|)', { x: 'orange' })
```

Observable Emitting Single Value

Here the value of the x character will be ‘orange’

```
hot('marbles', mock-object-with-values)
```

Mocking Observable

First parameter is marble syntax

Second parameter is the mock object with emitted value

```
hot('(x|)', {
    x: ['orange', 'apple']
})
```

Observable Emitting Single Value

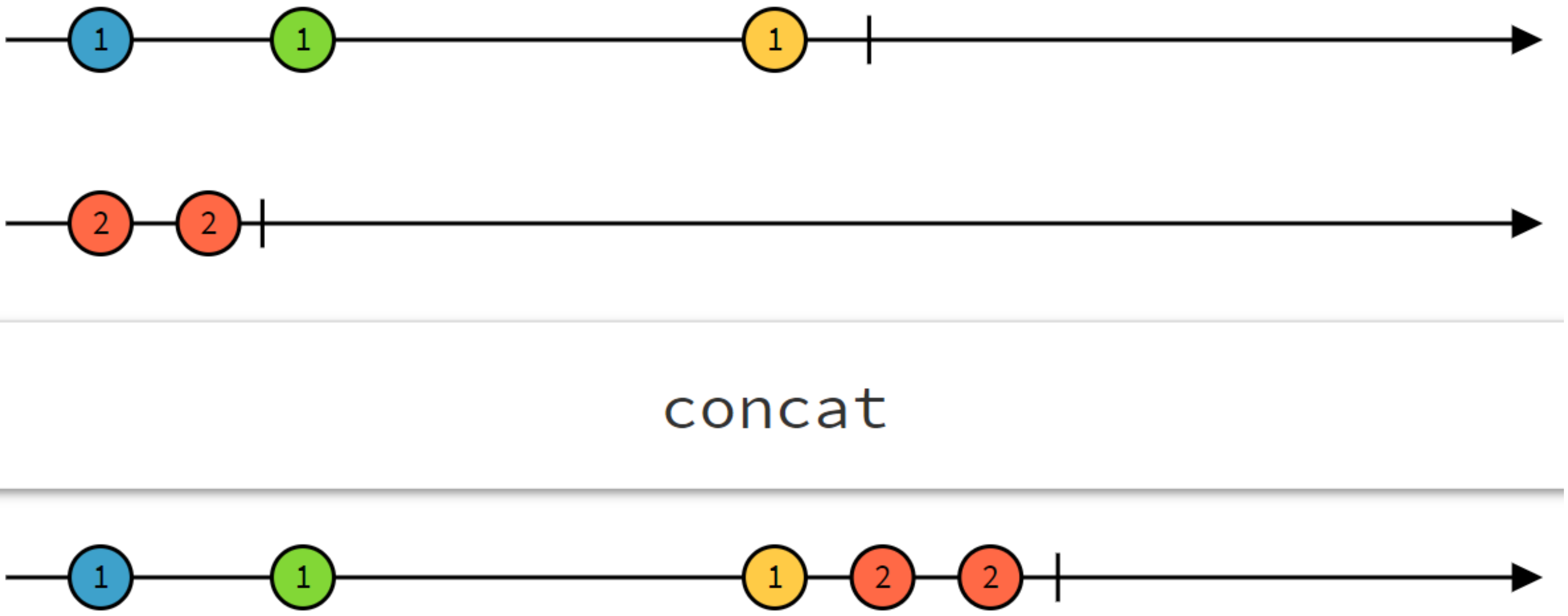
Here the value of the x character will be an array of ‘orange’ and ‘apple’

Demo

Testing mocking observable values

Testing RxJS Operators

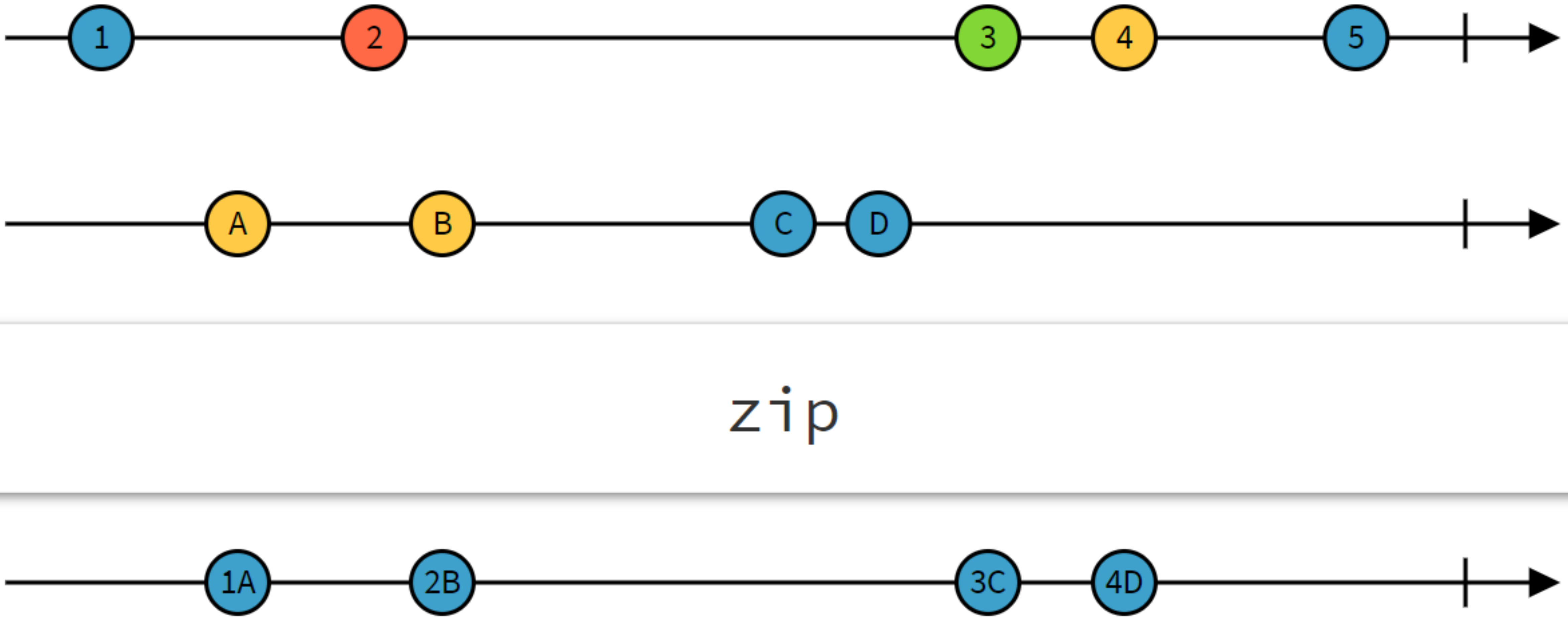
Concat



Demo

Testing RxJS concat operator

Zip



Demo

Testing RxJS zip operator

Summary

Hot and Cold Observables

Demo: Hot and Cold Observables

Testing Cold Observables

Demo: Testing Cold Observables

Testing Hot Observables

Summary