

# Testing Business Code and Handling Errors & Race Conditions

---



**Rupesh Kumar Tiwari**

WEB DEVELOPER

@roopkt <https://rupeshtiwari.com>



# Module Overview



**Test scheduler**

**Mocking services**

**Finding race condition**

**Solving race condition**

**Handling errors**

**Summary**



# Test Scheduler

---



# Scheduler



**Scheduler is a primitive inside RxJS**

**RxJS Operators takes scheduler as the second optional parameter**

- `Interval(40, async )`

**RxJS 6 has 6 built in schedulers**

**TestScheduler works with virtual time**



```
Import {interval} from 'rxjs';  
Import { getTestScheduler } from 'jasmine-marbles';  
interval(40,getTestScheduler())  
.subscribe(i=>console.log(i));
```

## getTestScheduler

Here `getTestScheduler` is providing test scheduler to interval operator that virtualizing the time and making our test synchronous



In Marble Testing, to call the  
subscribe callback immediately  
or synchronously, we flush the  
TestScheduler



```
Import {cold, getTestScheduler} from 'jasmine-marbles'
```

```
const messages$ = cold('--a---b--|',{
```

```
a: 'marble-testing'
```

```
b: 'is fun'
```

```
});
```

```
messages$.subscribe(m=>console.log(m));
```

```
getTestScheduler().flush();
```

◀ Import getTestScheduler

◀ Creating observable

◀ Subscribe to observable

◀ Flush the testScheduler



# Mocking Api Services

---





# Demo



## Mocking API service

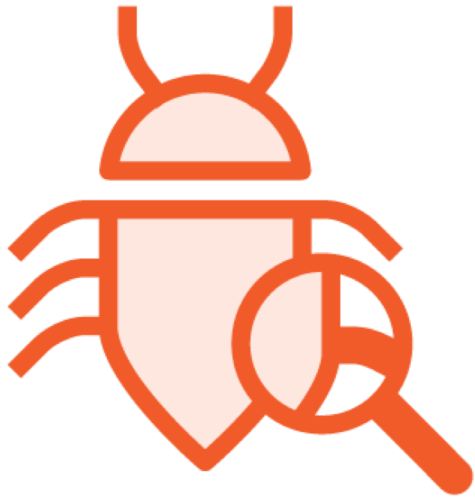


# Finding Race Conditions

---



# Race Condition Debugging



**Finding race condition is difficult**

**Testing race condition is more difficult**

# Race Condition

A **race condition** is an undesirable situation that occurs when a device or system attempts to perform two or more operations at the same time, but because of the nature of the device or system, the operations must be done in the proper sequence to be done correctly.



# Demo



## Finding race condition



# Solving Race Conditions

---



# Demo



## Solving race condition



# Handling Errors

---





# Why Handle Errors?



**Program may crash**

**End-user may loose their data**

**Difficult to trace issue**

#

`---a---b---#`

## Marble Syntax for Error

Hash symbol (#) is used to represent error in the sequence



```
cold( <marble syntax>, mocked-data-map, error );
```

```
hot( <marble syntax>, mocked-data-map, error );
```

## Cold and Hot Method

In both cold and hot method the 3<sup>rd</sup> argument is the error object that you want to pass for your unit test.



```
cold('--a---b---#', {a:'apple', b:'orange'},  
  new Error('no fruits found'));
```

Emit apple after 20ms wait for 30ms emit orange wait another 30ms  
emit error object with message 'no fruits found'



# Demo



## Error Handling



# Summary



Test scheduler & `GetTestScheduler` from jasmine marbles

Mocking services

Finding & solving race condition

Handling errors



# Course Summary



**Marble Syntaxes & Jasmine Marbles Hot and Cold methods**

**Unit Testing with Hot and cold observables**

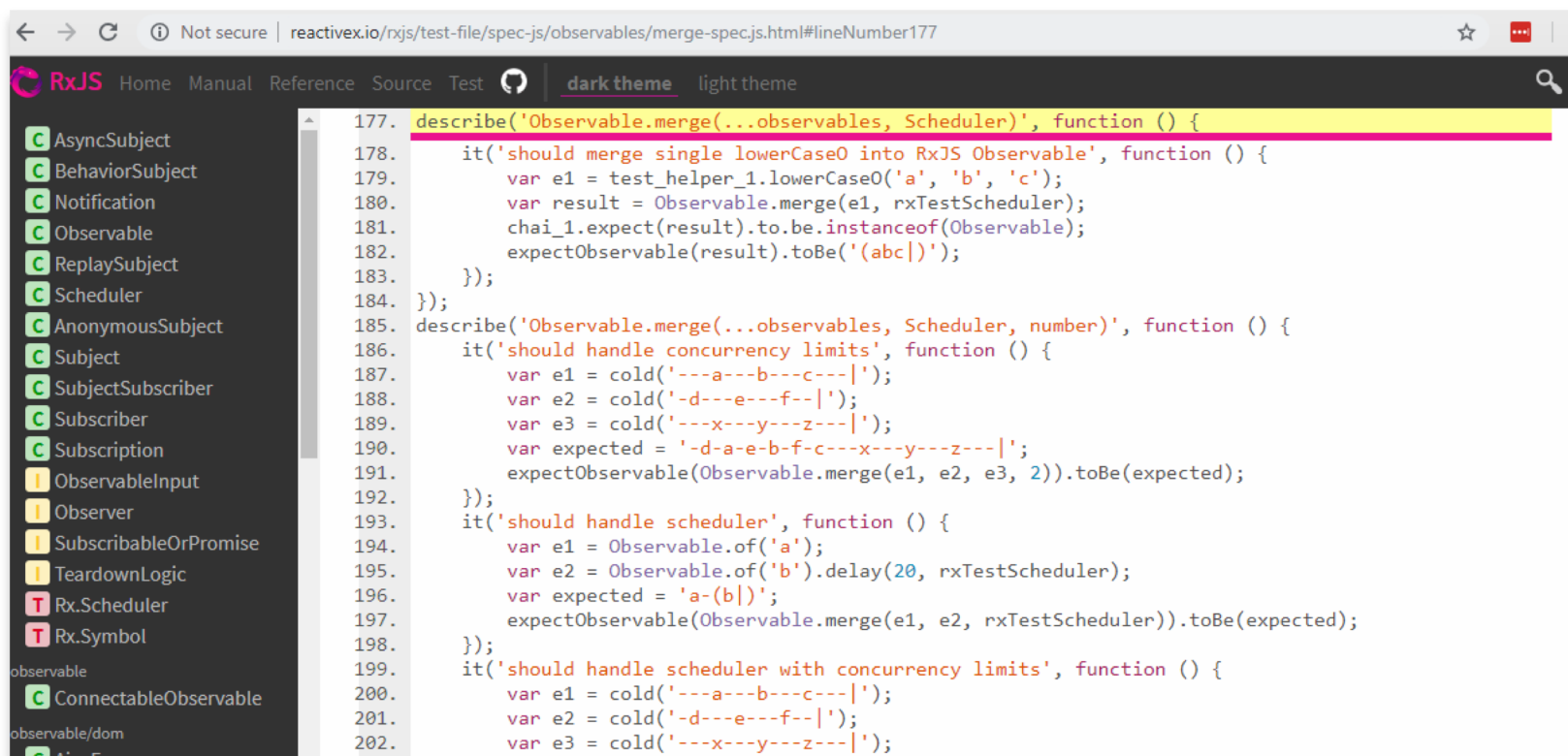
**Unit Testing with Mocking Observable & Operator values**

**Unit Testing Business Code, Handling Race conditions & Errors**



# Additional Learning on Marble Testing

<http://reactivex.io/rxjs/test-file/spec-js/observables/merge-spec.js.html#lineNumber177>



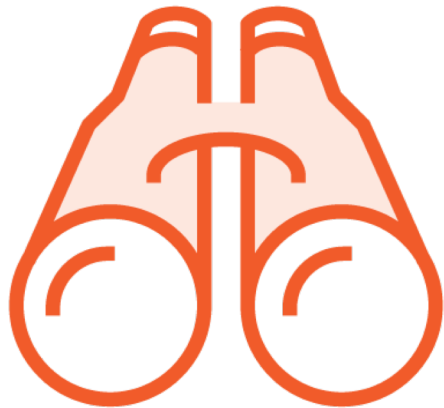
The screenshot shows a web browser displaying the RxJS source code for the merge-spec.js file. The URL in the address bar is <http://reactivex.io/rxjs/test-file/spec-js/observables/merge-spec.js.html#lineNumber177>. The page has a dark theme. On the left, there is a sidebar with a list of RxJS classes and methods, including AsyncSubject, BehaviorSubject, Notification, Observable, ReplaySubject, Scheduler, AnonymousSubject, Subject, SubjectSubscriber, Subscriber, Subscription, ObservableInput, Observer, SubscribableOrPromise, TeardownLogic, Rx.Scheduler, Rx.Symbol, observable, ConnectableObservable, observable/dom, and AjaxObservable. The main content area shows the source code for the merge-spec.js file, with line numbers 177 to 202 visible. The code is as follows:

```
177. describe('Observable.merge(...observables, Scheduler)', function () {
178.   it('should merge single lowerCaseO into RxJS Observable', function () {
179.     var e1 = test_helper_1.lowerCaseO('a', 'b', 'c');
180.     var result = Observable.merge(e1, rxTestScheduler);
181.     chai_1.expect(result).to.be.instanceOf(Observable);
182.     expectObservable(result).toBe('abc|');
183.   });
184. });
185. describe('Observable.merge(...observables, Scheduler, number)', function () {
186.   it('should handle concurrency limits', function () {
187.     var e1 = cold('---a---b---c---|');
188.     var e2 = cold('-d---e---f--|');
189.     var e3 = cold('---x---y---z---|');
190.     var expected = '-d-a-e-b-f-c---x---y---z---|';
191.     expectObservable(Observable.merge(e1, e2, e3, 2)).toBe(expected);
192.   });
193.   it('should handle scheduler', function () {
194.     var e1 = Observable.of('a');
195.     var e2 = Observable.of('b').delay(20, rxTestScheduler);
196.     var expected = 'a-(b|)';
197.     expectObservable(Observable.merge(e1, e2, rxTestScheduler)).toBe(expected);
198.   });
199.   it('should handle scheduler with concurrency limits', function () {
200.     var e1 = cold('---a---b---c---|');
201.     var e2 = cold('-d---e---f--|');
202.     var e3 = cold('---x---y---z---|');
```





# What to Do Next?



**Practice writing unit tests using marble diagrams**

**Marble testing session every day**

**[reactivex.io](https://reactivex.io), [rxviz.com](https://rxviz.com) and [rxmarbles.com](https://rxmarbles.com)**

**Marble testing on NgRx Effects**

**Join discussion on Pluralsight**



# Contact Me!



**Email:** [info@rupeshtiwari.com](mailto:info@rupeshtiwari.com), **Twitter:** [@roopkt](https://twitter.com/roopkt)

**LinkedIn:**

<https://www.linkedin.com/in/roopkt/>

**Website:** <https://rupeshtiwari.com>

**YouTube Channel:**

<https://www.youtube.com/channel/UCfjBZHutgAYon-T8sqt1rwg>



Thank You

