

Key Transforming Operators in the Observable Class (Part 1)

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

**Institute for Software
Integrated Systems**

**Vanderbilt University
Nashville, Tennessee, USA**



Learning Objectives in this Part of the Lesson

- Recognize key Observable operators
 - Factory method operators
- Transforming operators
 - Transform the values and/or types emitted by an Observable
 - e.g., `map()`



Key Transforming Operators in the Observable Class

Key Transforming Operators in the Observable Class

- The map() operator
 - Transform the item(s) emitted by this Observable

```
<V> Observable<V> map  
(Function<? super T,? extends V>  
 mapper)
```

Key Transforming Operators in the Observable Class

- The map() operator
 - Transform the item(s) emitted by this Observable
 - Applies a synchronous function to transform each item

```
<V> Observable<V> map  
(Function<? super T, ? extends V>  
 mapper)
```

Interface Function<T,R>

Type Parameters:

T - the type of the input to the function

R - the type of the result of the function

All Known Subinterfaces:

UnaryOperator<T>

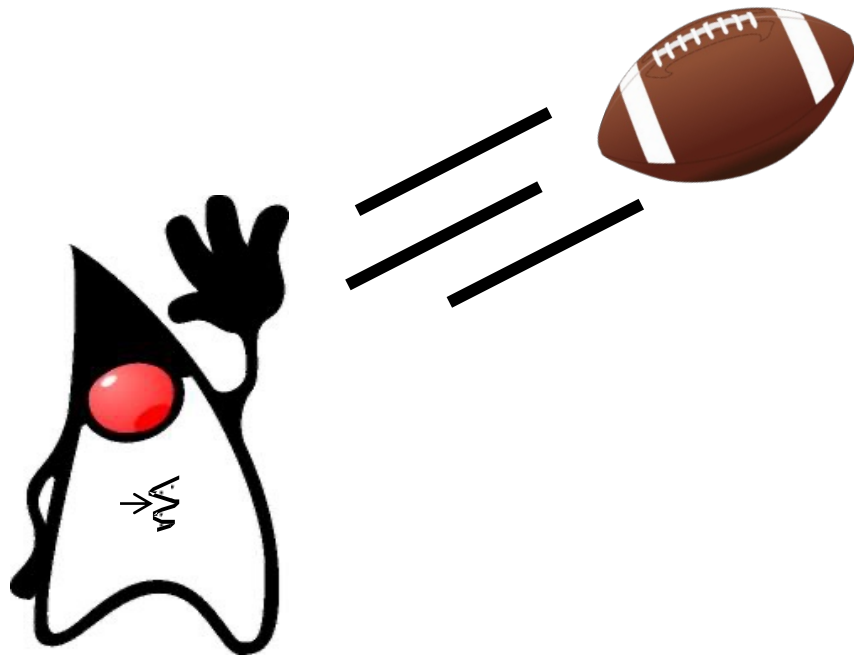
Functional Interface:

This is a functional interface and can therefore be used as the assignment target for a lambda expression or method reference.

Key Transforming Operators in the Observable Class

- The map() operator
 - Transform the item(s) emitted by this Observable
 - Applies a synchronous function to transform each item
 - map() can terminate if mapper throws an exception

```
<V> Observable<V> map  
(Function<? super T, ? extends V>  
 mapper)
```



Key Transforming Operators in the Observable Class

- The map() operator
 - Transform the item(s) emitted by this Observable
 - Applies a synchronous function to transform each item
 - Returns a transformed Observable

```
<V> Observable<V> map  
(Function<? super T,? extends V>  
mapper)
```



Key Transforming Operators in the Observable Class

- The `map()` operator
 - Transform the item(s) emitted by this Observable
 - The # of output items must match the # of input items

Observable

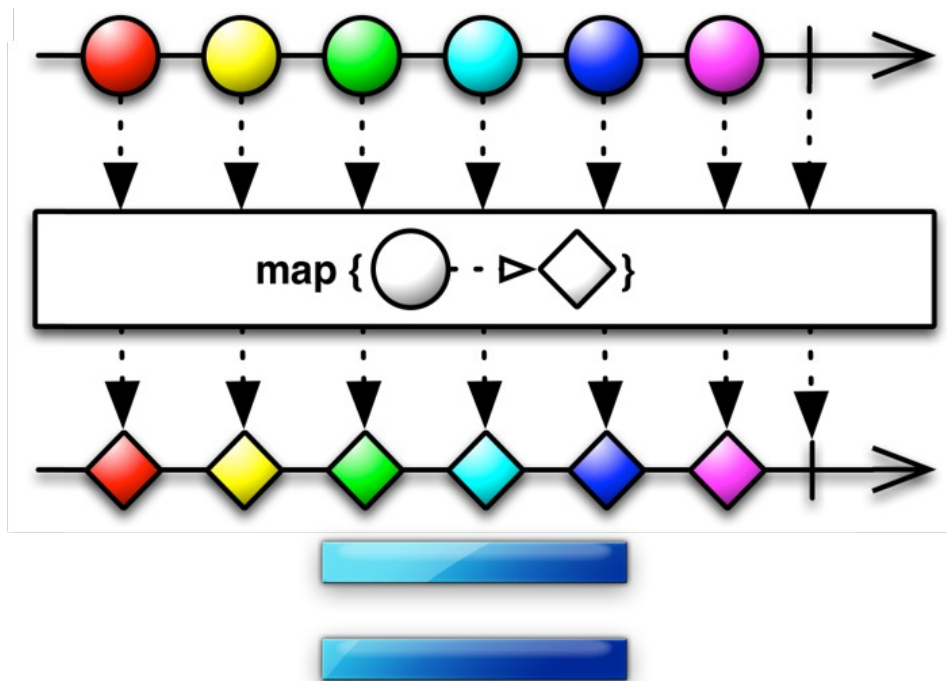
```
.fromIterable  
  (bigFractionList)
```

```
...
```

```
.map(fraction -> fraction  
    .multiply(sBigReducedFrac))
```

```
...
```

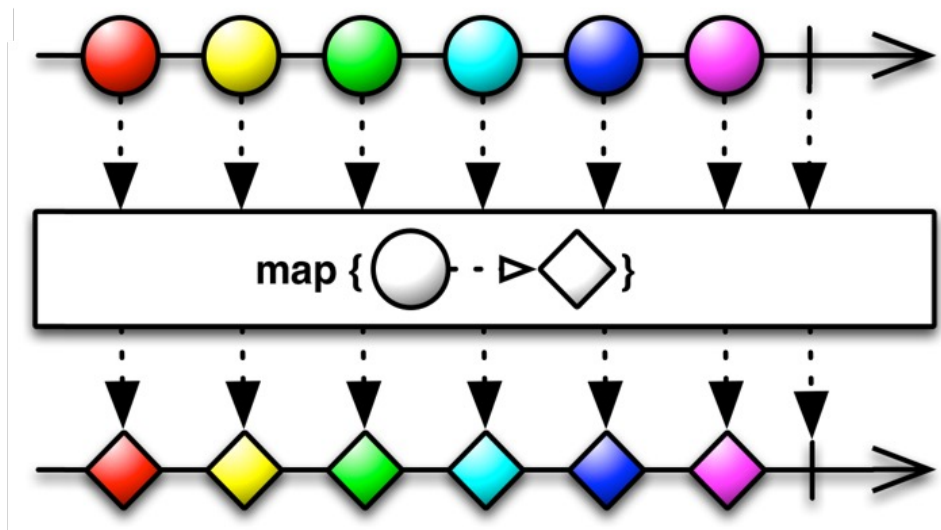
Multiply each element in the Observable stream by a constant



See [Reactive/Observable/ex1/src/main/java/ObservableEx.java](#)

Key Transforming Operators in the Observable Class

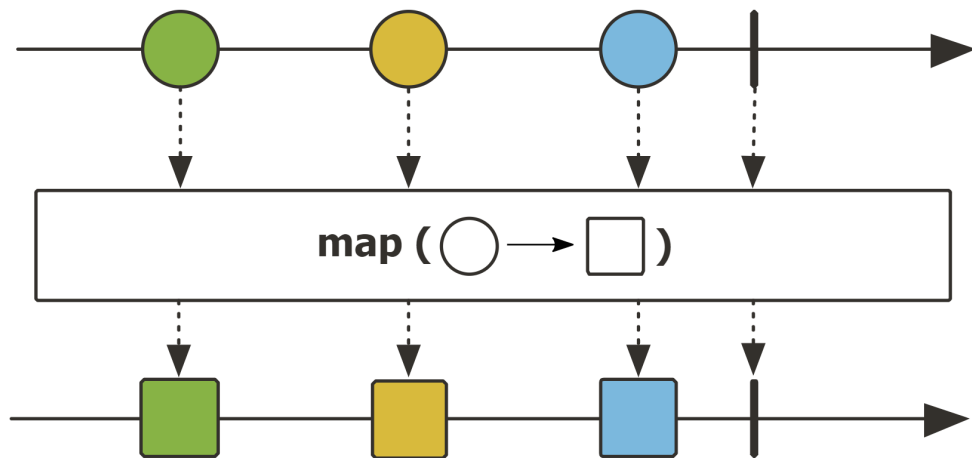
- The map() operator
 - Transform the item(s) emitted by this Observable
 - The # of output items must match the # of input items
 - map() can transform the type and/or value of elements it processes



Key Transforming Operators in the Observable Class

- The `map()` operator
 - Transform the item(s) emitted by this Observable
 - The # of output items must match the # of input items
 - Project Reactor's `Flux.map()` operator works the same
- Flux**

```
.fromIterable  
  (bigFractionList)  
...  
.map(fraction -> fraction  
    .multiply(sBigReducedFrac))  
...
```



Multiply each element in the Flux stream by a constant

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Flux.html#map

Key Transforming Operators in the Observable Class

- The map() operator
 - Transform the item(s) emitted by this Observable
 - The # of output items must match the # of input items
 - Project Reactor's Flux.map() operator works the same
 - Similar to Stream.map() method in Java Streams

map

```
<R> Stream<R> map(Function<? super T,? extends R> mapper)
```

Returns a stream consisting of the results of applying the given function to the elements of this stream.

This is an intermediate operation.

Type Parameters:

R - The element type of the new stream

Parameters:

mapper - a non-interfering, stateless function to apply to each element

```
List<String> collect = List  
    .of("a", "b", "c").stream()  
    .map(String::toUpperCase).collect(toList());
```

*Uppercase each
string in a stream*

See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#map

End of Key Transforming Operators in the Observable Class (Part 1)