

# Angular and RxJS

Jogesh K. Muppala



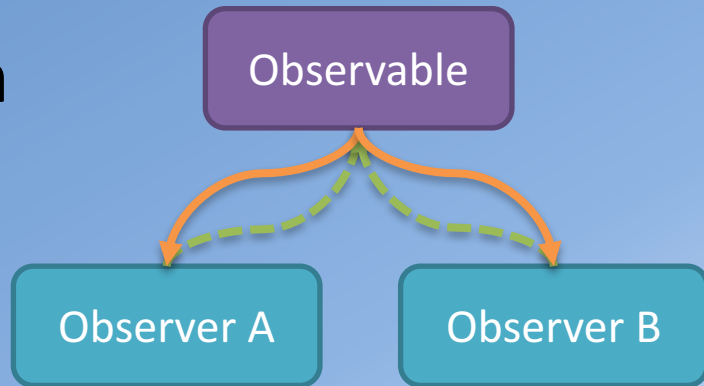
THE DEPARTMENT OF  
**COMPUTER SCIENCE & ENGINEERING**  
計算機科學及工程學系



香港科技大學  
THE HONG KONG UNIVERSITY OF  
SCIENCE AND TECHNOLOGY

# The Observer Pattern

- Software-engineering pattern
  - Gang of four
- Observable (subject) tracks observers
- Observable notifies observers of state changes



[https://en.wikipedia.org/wiki/Observer\\_pattern](https://en.wikipedia.org/wiki/Observer_pattern)

# Why Observables?

- Streams
- Can be unsubscribed
- Lazy, or cold until some observer does a `.subscribe()`
- Canceling and retrial straightforward

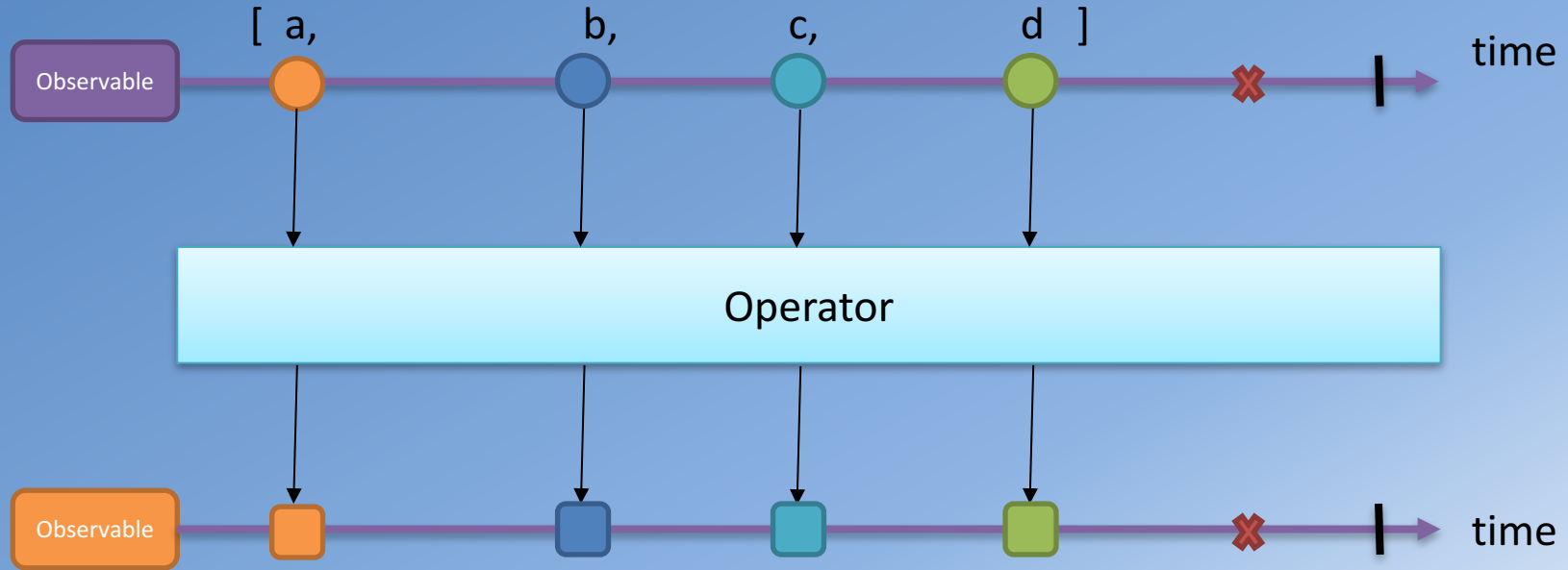
# Reactive Programming

- Data flows: how does data flow through your application?
- Propagation of change through your application
- Built around “streams”
- Functional Reactive Programming
  - Functional + Reactive

# RxJS

- Library for composing asynchronous and event-based programs by using observable sequences.
- Provides:
  - one core type, the Observable,
  - satellite types (Observer, Schedulers, Subjects), and
  - operators inspired by Array#extras (map, filter, reduce, every, etc)
  - to allow handling asynchronous events as collections.
- Used for observables support in Angular

# Observables, Operators and Marble Diagrams



# Angular and RxJS

- Observables all over Angular:
  - Forms
  - HTTP
  - AsyncPipe
  - Change detection

# Angular and RxJS

/dishdetail/:id

ActivatedRoute service

**params** Observable


Observable



this.route.**params**

Observable



Operator  **.switchMap**((params: Params) => **this.dishservice.getDish(+params['id'])**)

**.subscribe**(dish => { this.dish = dish; this.setPrevNext(dish.id); });



# An Ode to Observables

If you be my Observable,  
    I'll be your longing Observer  
You can call me with your values,  
    And I'll respond whenever you do!  
Every value you emit,  
    I'll transform using map, filter and concat,  
Don't be so cold and lazy,  
    I'll subscribe and make you hot and crazy!!

Sung to the tune of  
You Can Call Me Al by Paul Simon