

다음 카페/블로그 검색 앱 만들기

1 인트로 (완성앱 & 구현 기능 소개)

Observable

1

인트로 (완성애플 & 구현
기능 소개)

Observable

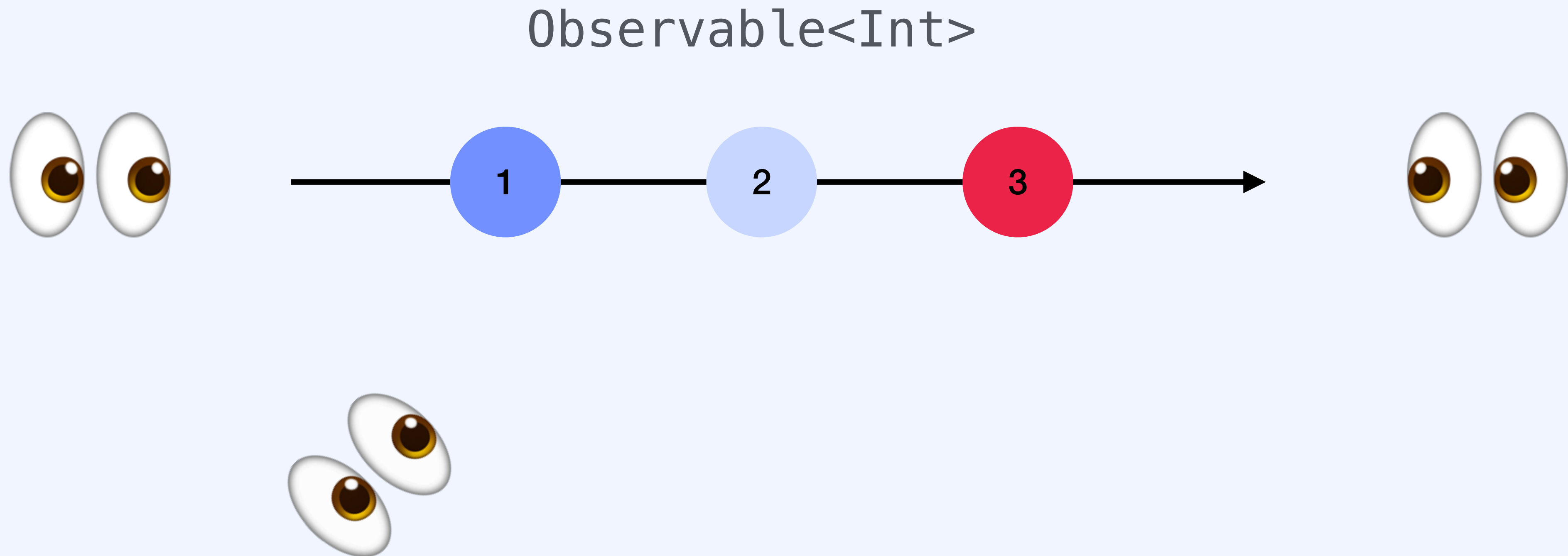
= Observable Sequence

= Sequence

Observable

2

인트로 (완성애플 & 구현
기능 소개)



Observable

3

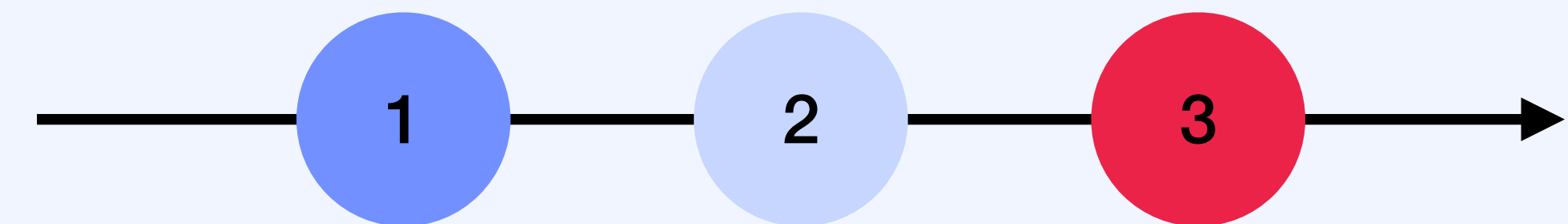
인트로 (완성애플 & 구현
기능 소개)

```
Observable<Int>.of(1, 2, 3)
```

```
Observable<Int>.from([1, 2, 3])
```

```
Observable<Int>.create { observer -> Disposable in  
    observer.onNext(1)  
    observer.onNext(2)  
    observer.onNext(2)  
    return Disposables.create()  
}
```

Observable<Int>



Observable

4

인트로 (완성애플 & 구현
기능 소개)

```
//내가 subscription을 하기 전에는
Observable.of("🌸") //그는 다만 하나의 Observable에 지나지 않았다

    .subscribe { //내가 그의 이름을 불러주었을 때
        print($0.element) //그는 나에게로 와서 🌸이 되었다
    }

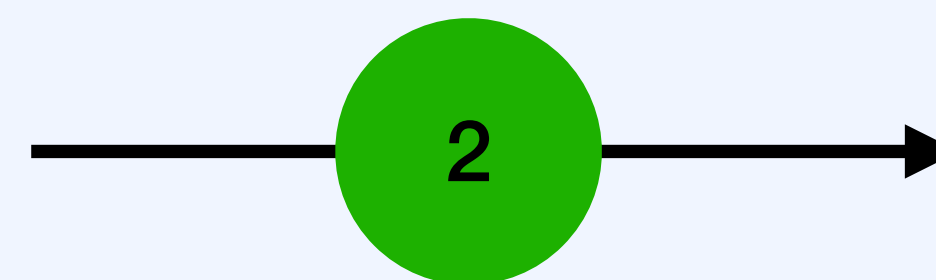
    .disposed(by: disposeBag)
```

Observable

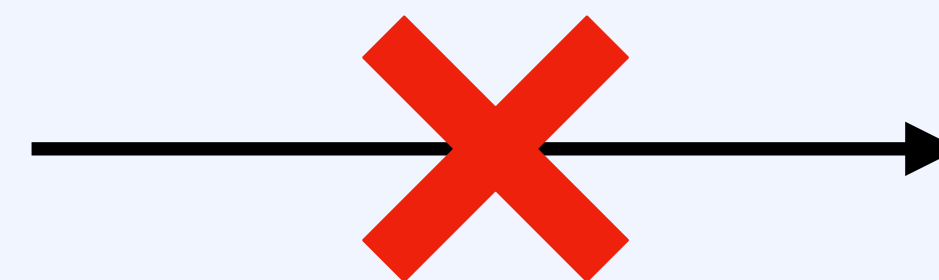
5

인트로 (완성애플 & 구현
기능 소개)

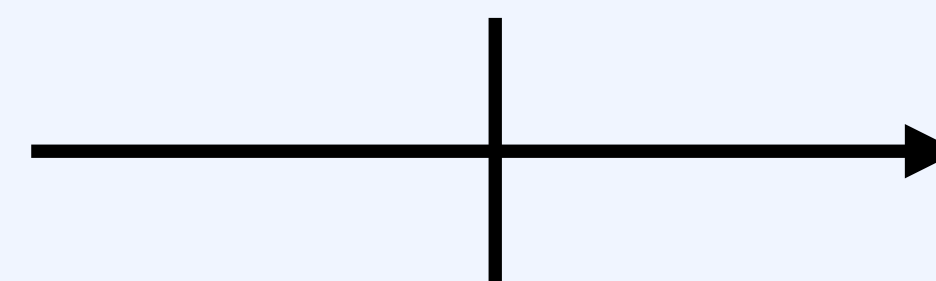
`next(element)`



`error(Swift.Error)`



`completed`

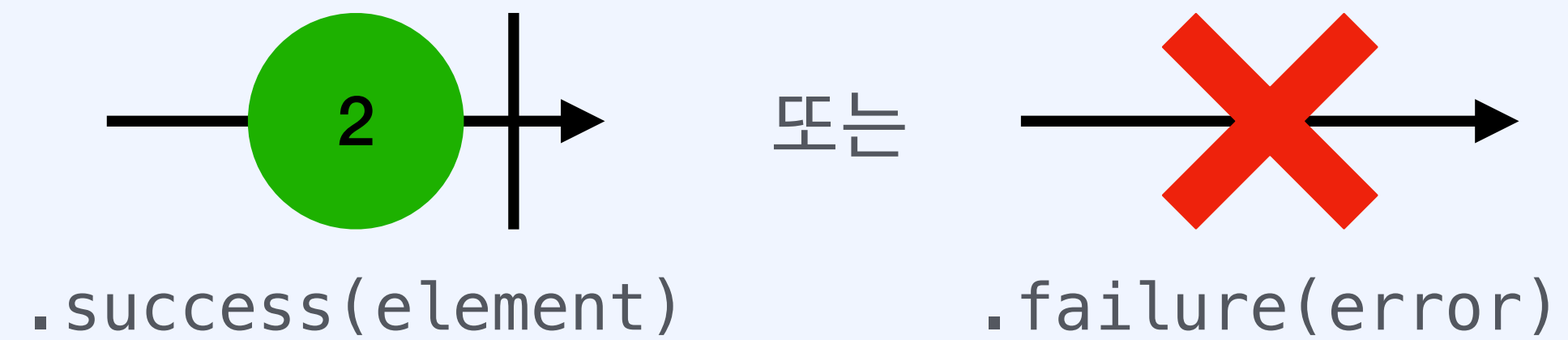


Observable

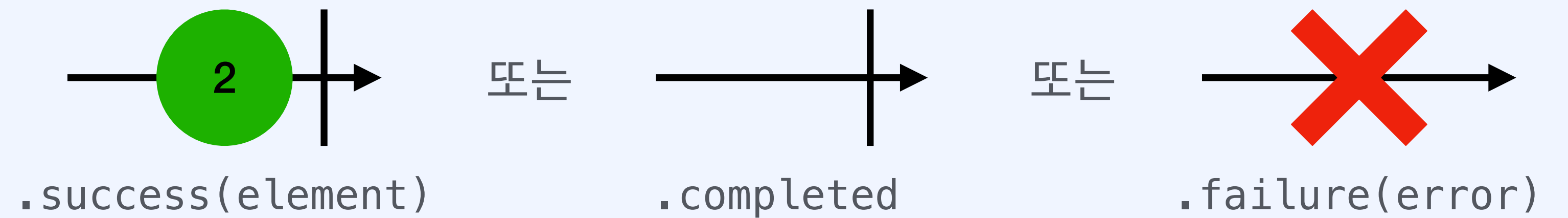
6

인트로 (완성애플 & 구현
기능 소개)

Single<Element>



Maybe<Element>



Completable



Subject

7

인트로 (완성애플 & 구현
기능 소개)

```
PublishSubject<T>()
```

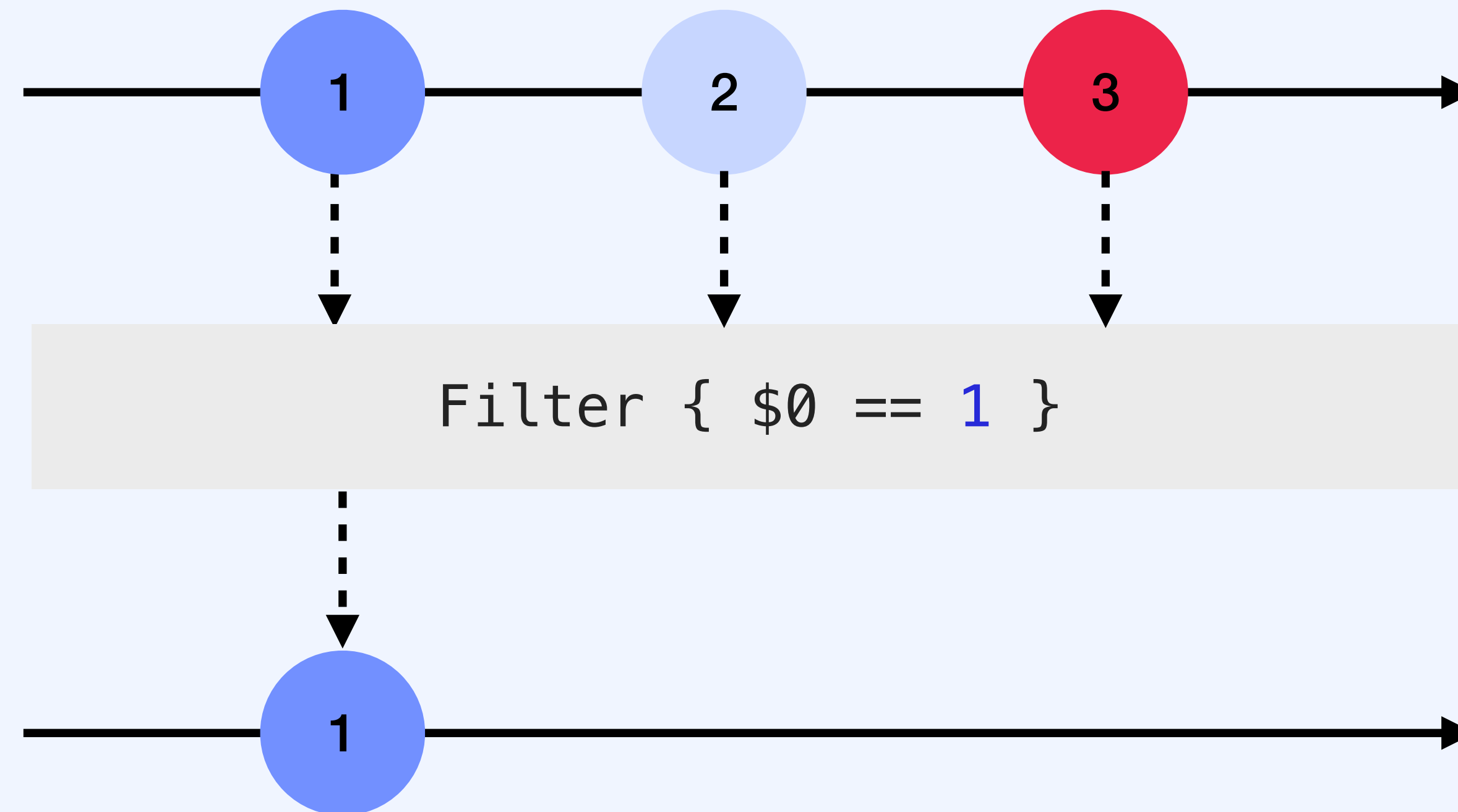
```
BehaviorSubject<T>(value: T)
```

```
ReplaySubject<T>.create(bufferSize: Int)
```


Filtering Operators

8

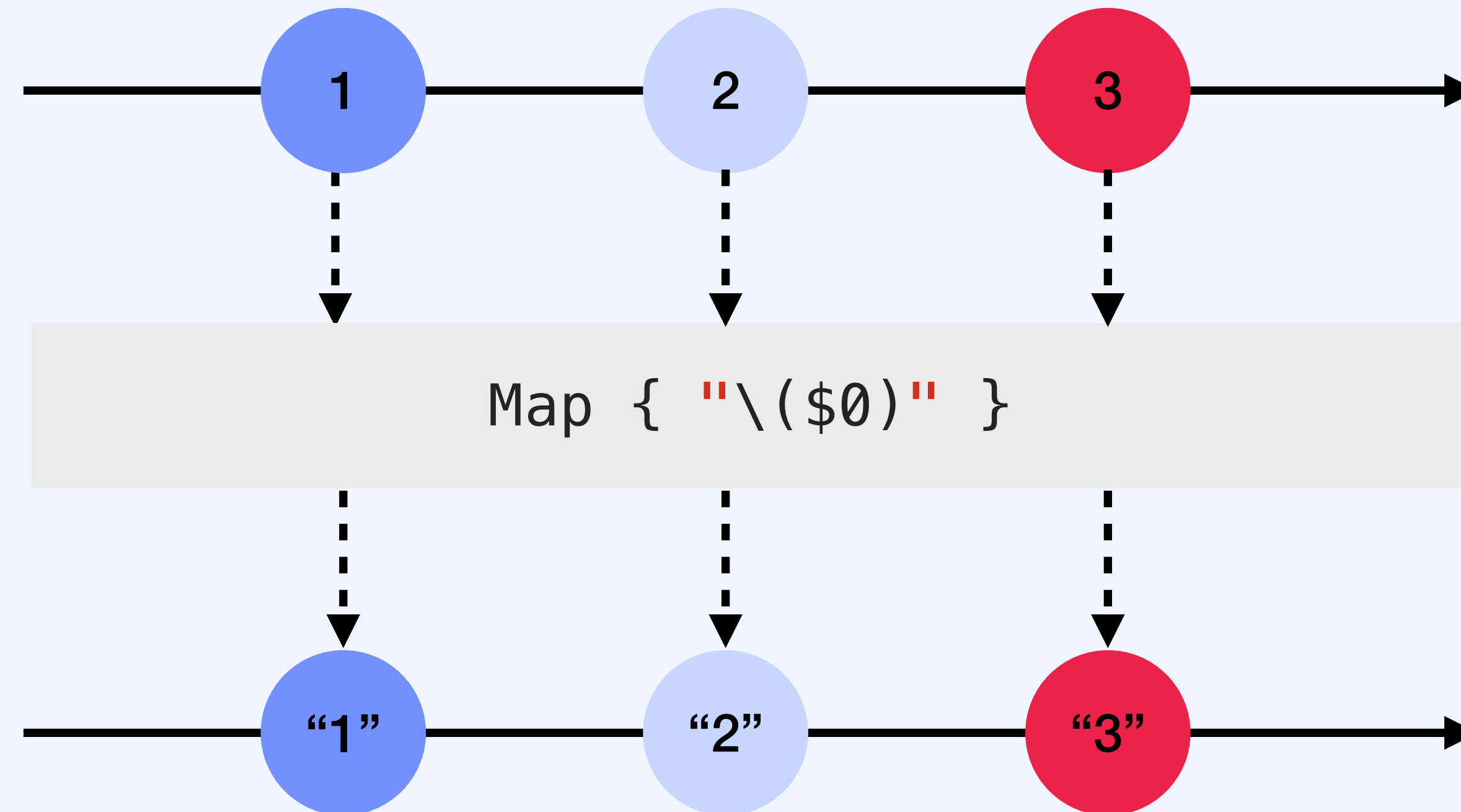
인트로 (완성앱 & 구현
기능 소개)



Transforming Operators

9

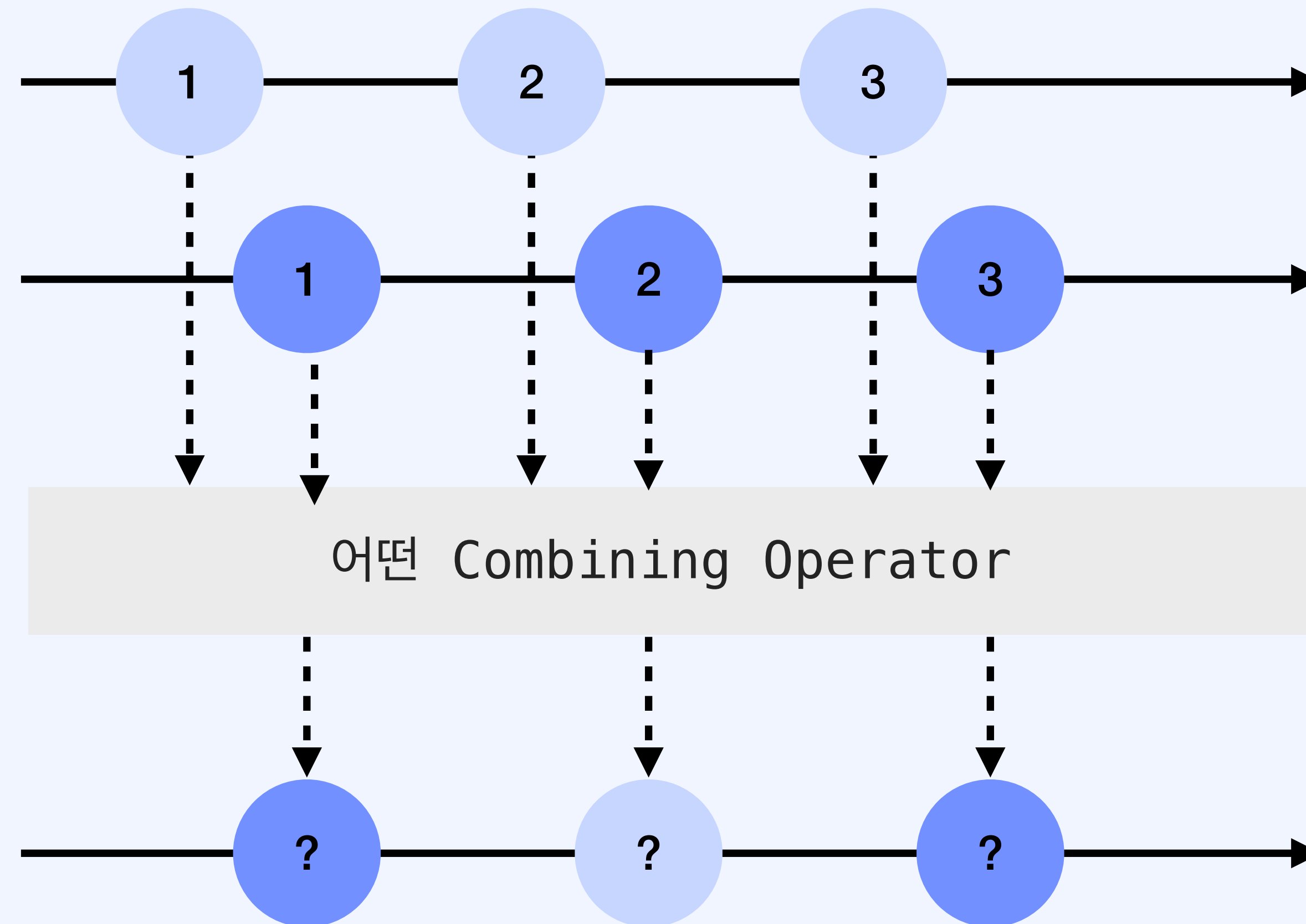
인트로 (완성앱 & 구현
기능 소개)



Combining Operators

10

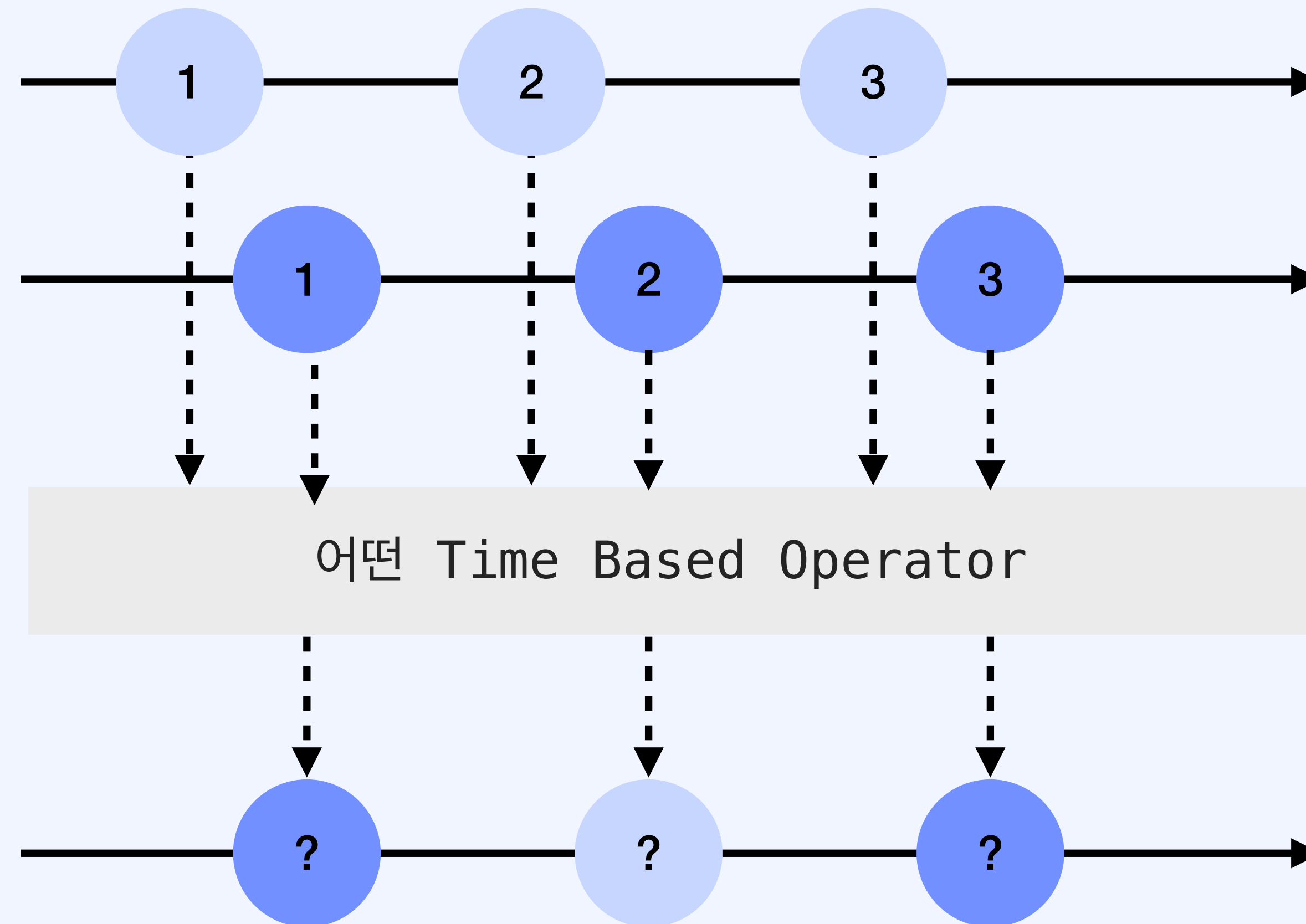
인트로 (완성앱 & 구현
기능 소개)



Time Based Operators

11

인트로 (완성앱 & 구현
기능 소개)



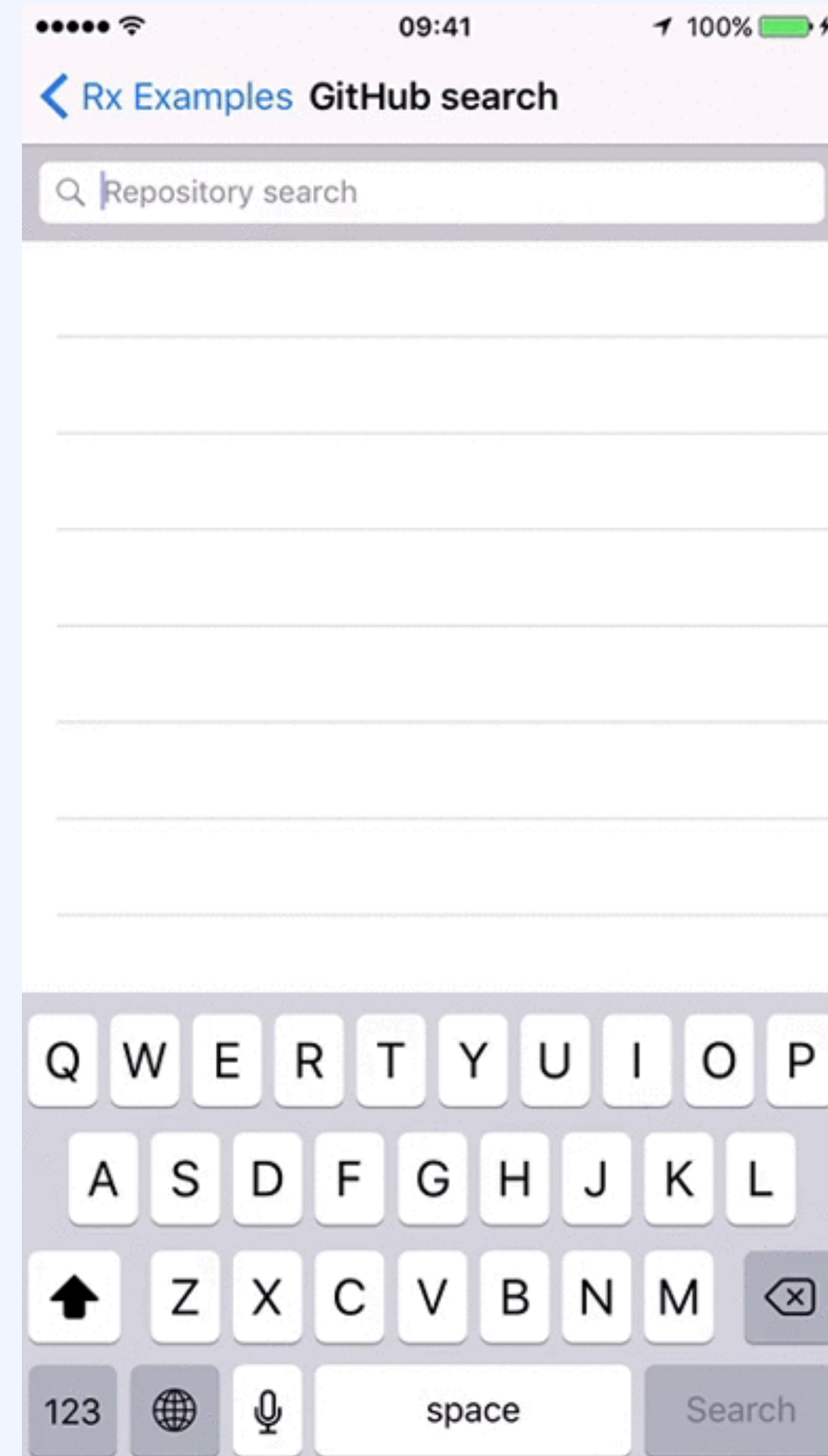
RxCocoa

12

인트로 (완성앱 & 구현
기능 소개)

```
let searchResults = searchBar.rx.text
    .distinctUntilChanged()
    .flatMapLatest {
        //어떤 로직
    }

searchResults
    .bind(to: tableView.rx.items) {
        //cell 설정
    }
    .disposed(by: disposeBag)
```



Error 처리

13

인트로 (완성앱 & 구현
기능 소개)

catch



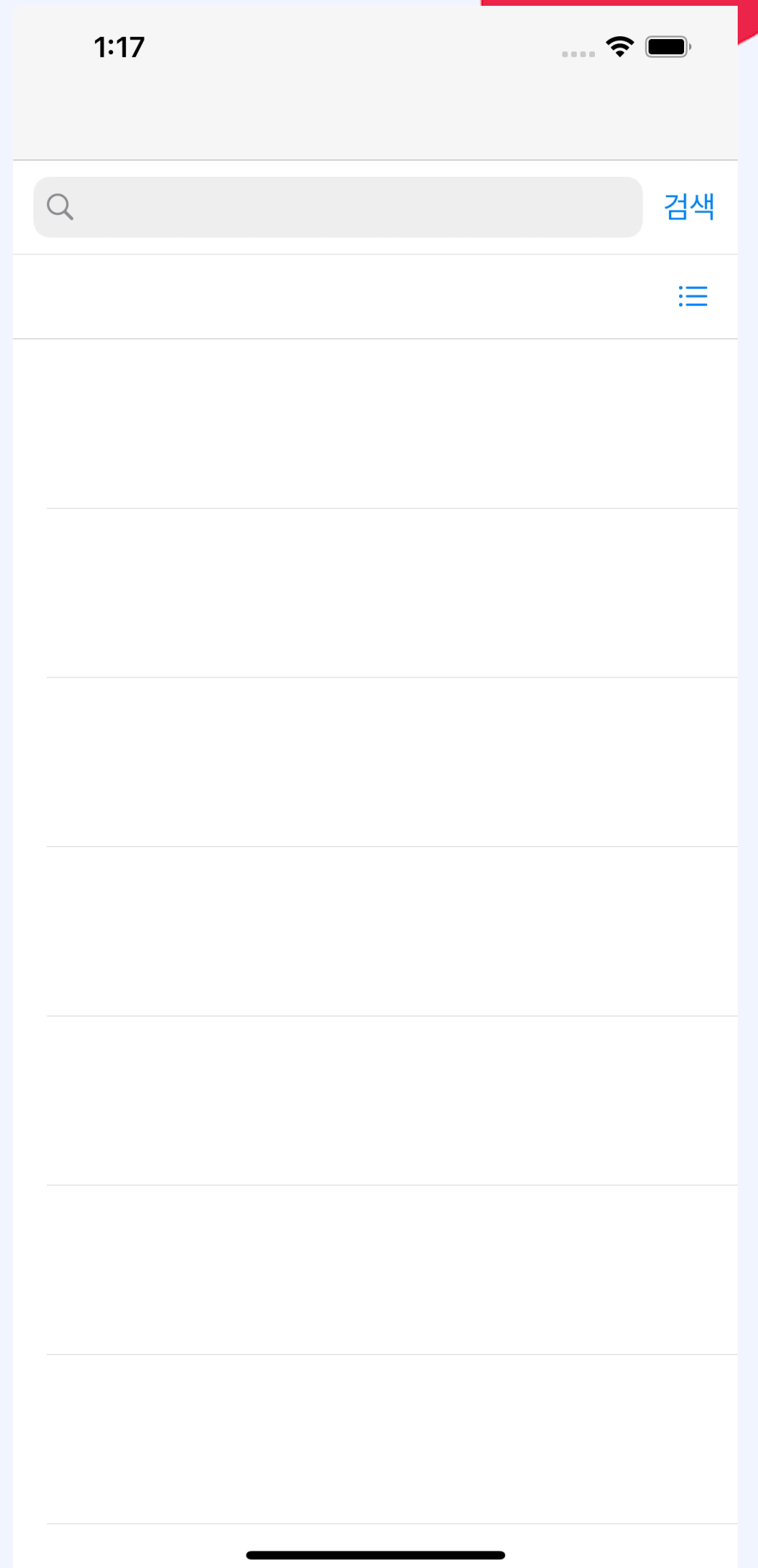
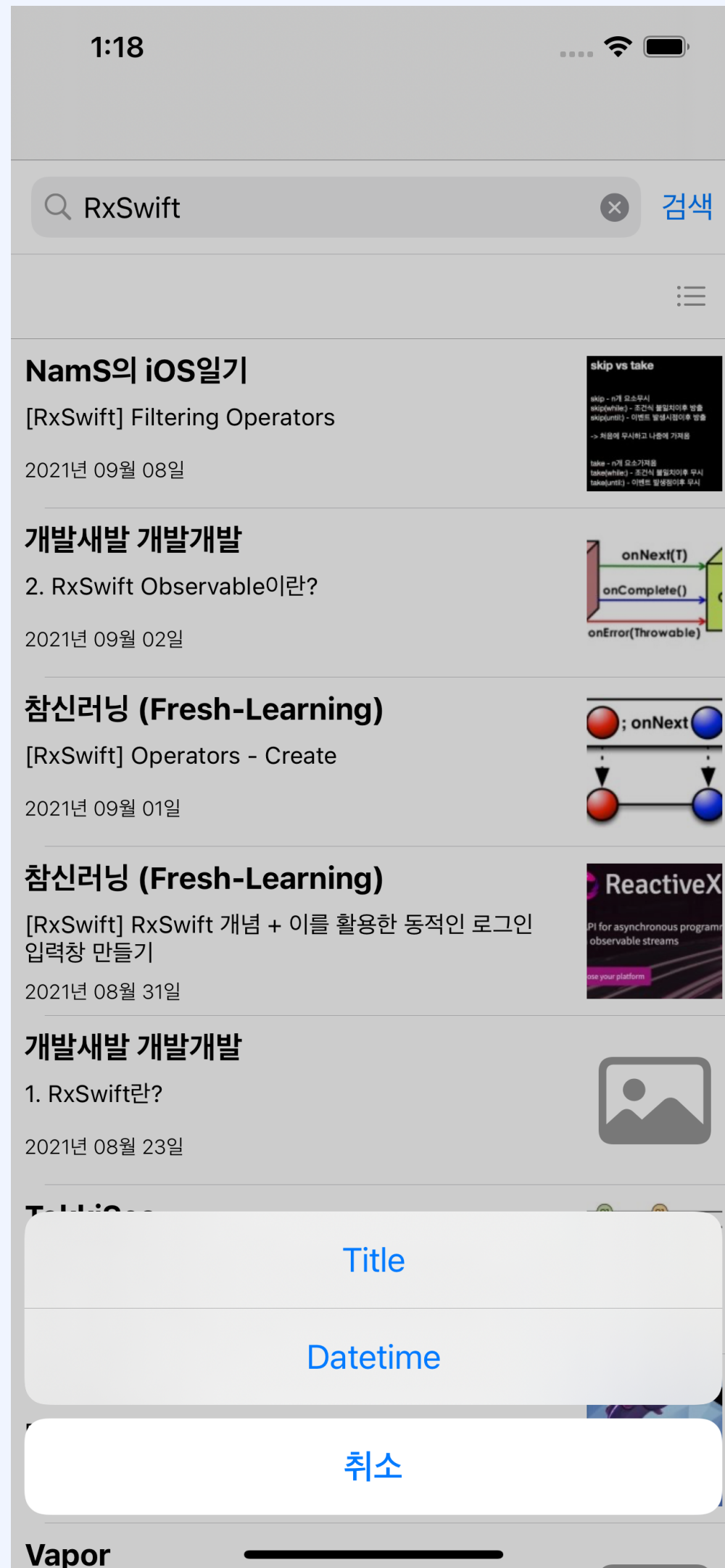
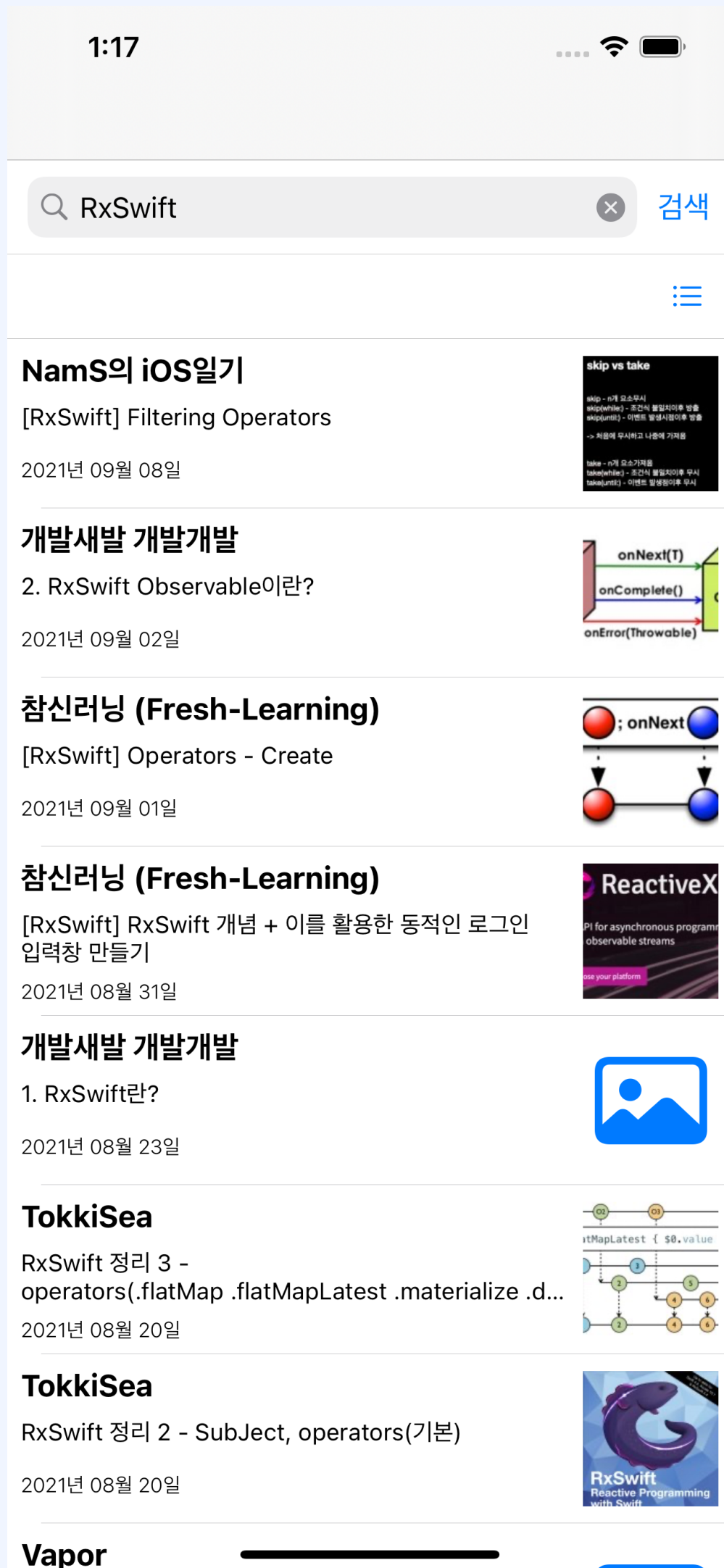
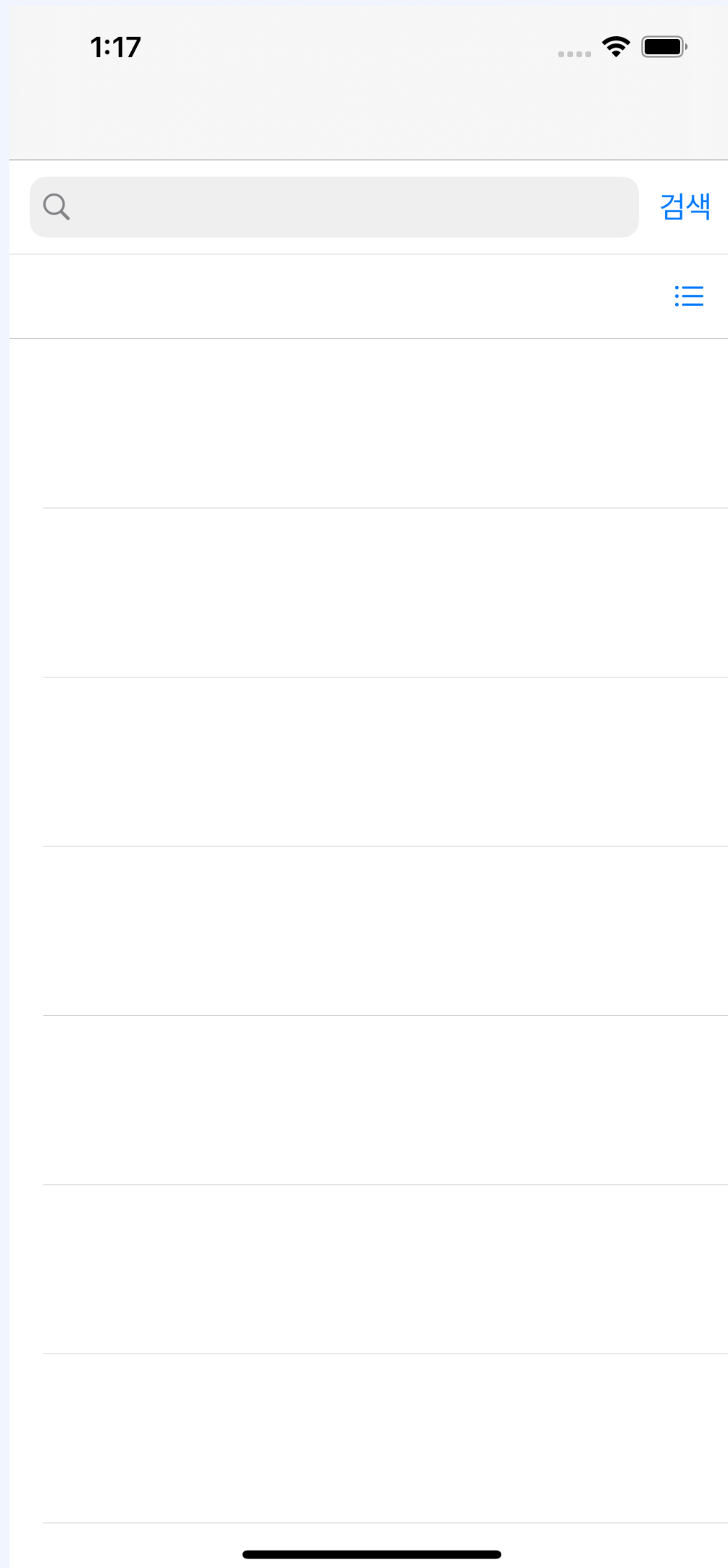
retry



완성 앱

14

인트로 (완성앱 & 구현
기능 소개)



다음 카페/블로그 검색 앱 만들기

2 Combining Operator

다음 카페/블로그 검색 앱 만들기

3 Time Based Operator

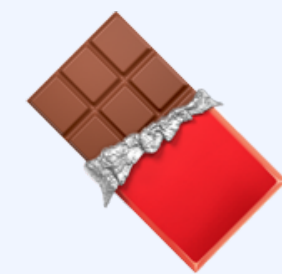
다음 카페/블로그 검색 앱 만들기

4 RxCocoa 알아보기

Cocoa Framework

1

RxCocoa 알아보기



Cocoa Framework

Foundation

UIKit (UIKit)

RxCocoa

2

RxCocoa 알아보기



`Binder<Value>`



`Driver<Element>`
`Signal<Element>`



`extension Reactive where Base: T {}`

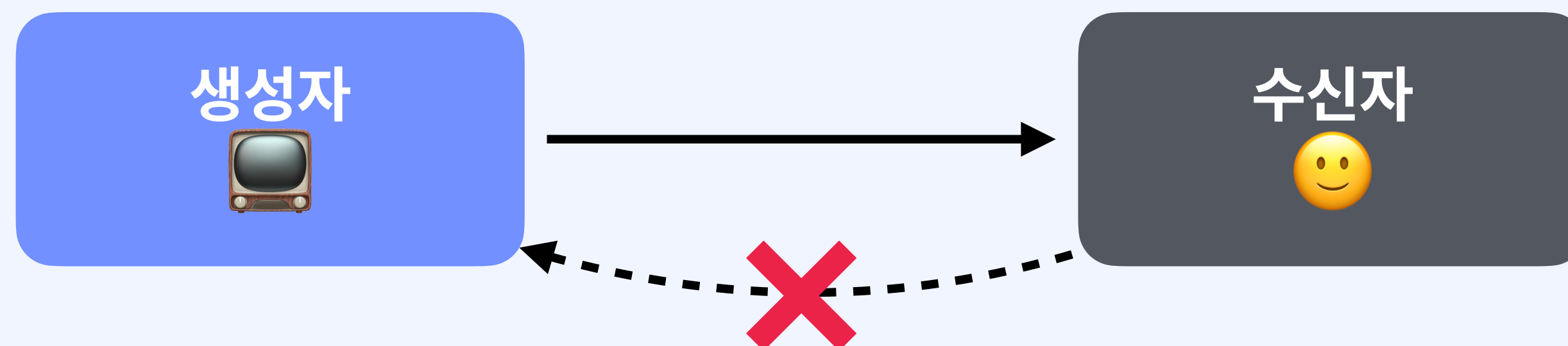
Binder

3

RxCocoa 알아보기



Binder<Value>



Binder

4

RxCocoa 알아보기



Binder<Value>

데이터 생산

생성자
Observable

`.bind(to:)`

데이터 소비

수신자
Binder



Binder

5

RxCocoa 알아보기



Binder<Value>

데이터 생산

생성자
Observable

.bind(to:)

데이터 소비

수신자
Binder

- Observer (수신 Only)
- UI Binding에 사용
- Error 이벤트 받지 않음
- Main Thread에서 실행되는 것을 보장

Binder

6

RxCocoa 알아보기



Binder<Value>

```
textField.rx.text
    .observe(on: MainScheduler.instance)
    .subscribe(onNext: {
        label.text = $0
    })
    .disposed(by: disposeBag)
```

입력하세요

Binder

7

RxCocoa 알아보기



Binder<Value>

```
textField.rx.text
    .observe(on: MainScheduler.instance)
    .subscribe(onNext: {
        label.text = $0
    })
    .disposed(by: disposeBag)
```

```
textField.rx.text
    .bind(to: label.rx.text)
    .disposed(by: disposeBag)
```

입력하세요

Traits

8

RxCocoa 알아보기



`Driver<Element>`
`Signal<Element>`

- 에러를 방출하지 않는 특별한 observable
- 모든 과정은 main thread에서 이뤄진다.
- 스트림 공유가 가능하다.
- Driver: 초기값 || 최신값 replay
- Signal: 구독한 이후에 발생하는 값 전달

Rx Extension

9

RxCocoa 알아보기



```
extension Reactive where Base: T {}
```

```
extension Reactive where Base: UIView {
    var sizeToFit: Binder<Void> {
        return Binder(base) { base, _ in
            base.sizeToFit()
        }
    }
}
```

```
Driver.just(Void())
    .drive(button.rx.sizeToFit)
    .disposed(by: disposeBag)
```

다음 카페/블로그 검색 앱 만들기

5 Error 처리하기

에러 관리

1

Error 처리하기



- 인터넷 연결 없음: 오프라인
- 잘못된 입력: 잘못된 타입, 길이, 크기, 내용
- API, HTTP 에러: 400, 500, JSON Codable

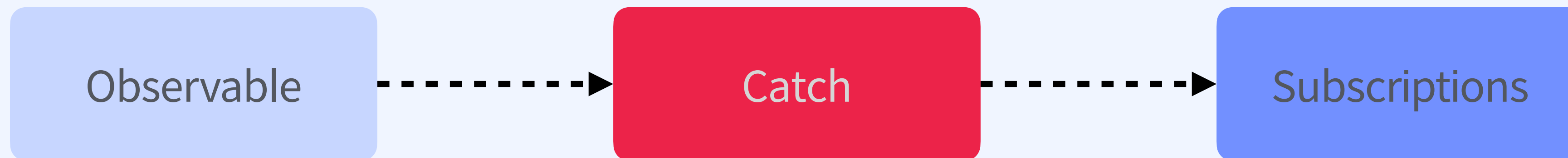
RxSwift의 에러 관리

2

Error 처리하기

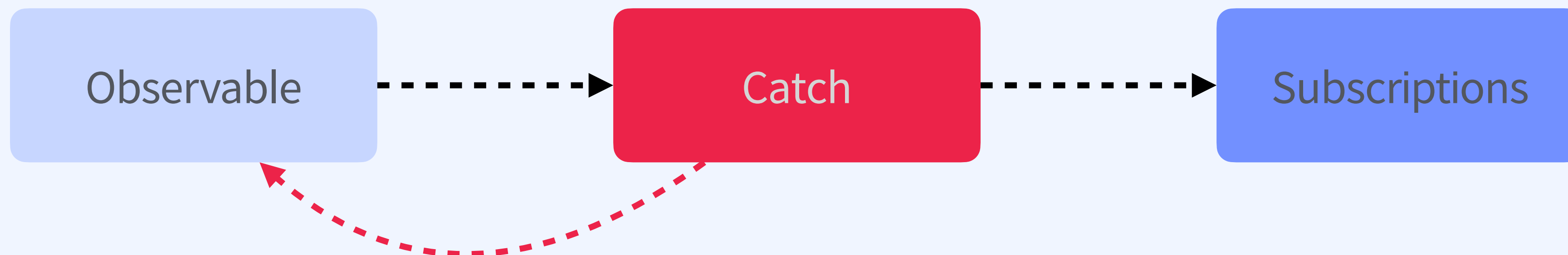
catch

기본값defaultValue으로 error 복구하기



retry

제한적 또는 무제한으로 재시도Retry 하기



Catch

3

Error 처리하기

```
func catch(_ handler:) -> RxSwift.Observable<Self.Element>
```

```
enum MyError: Error {
    case anError
    case criticalError
}

Observable.create {
    $0.onError(MyError.anError)
    return Disposables.create()
}
.catch { error in
    switch error as! MyError {
    case .anError:
        return .just("괜찮아요")
    case .criticalError:
        return .just("핑!")
    }
}
.subscribe {
    print($0)
}
.disposed(by: disposeBag)
```

괜찮아요

CatchAndReturn

```
func catchAndReturn(_ element: Self.Element) -> RxSwift.Observable<Self.Element>
```

```
enum MyError: Error {
    case anError
}

Observable.create {
    $0.onError(MyError.anError)
    return Disposables.create()
}
.catchAndReturn("에러싫어")
.subscribe {
    print($0)
}
.disposed(by: disposeBag)
```

next(에러싫어)
completed

Retry


5


Error 처리하기

```
func retry() -> RxSwift.Observable<Self.Element>
```

```
enum MyError: Error {
    case anError
    case criticalError
}

Observable.create {
    $0.onError(MyError.anError)
    return Disposables.create()
}
.retry()
.subscribe {
    print($0)
}
.disposed(by: disposeBag)
```

RxSwift.AnyObserver... 

RxSwift.(unknown co... 

Retry

6

Error 처리하기

```
func retry(_ maxAttemptCount: Int) -> RxSwift.Observable<Self.Element>
```

```
enum MyError: Error {
    case anError
    case criticalError
}

Observable.create {
    $0.onError(MyError.anError)
    return Disposables.create()
}
.retry(10000)
.subscribe {
    print($0)
}
.disposed(by: disposeBag)
```

(52642 times)



(52642 times)



Unhandled error happened: anError

다음 카페/블로그 검색 앱 만들기

6 기본 UI 그리기

다음 카페/블로그 검색 앱 만들기

7 Kakao API 연결하기

다음 카페/블로그 검색 앱 만들기

8 데이터 처리하기

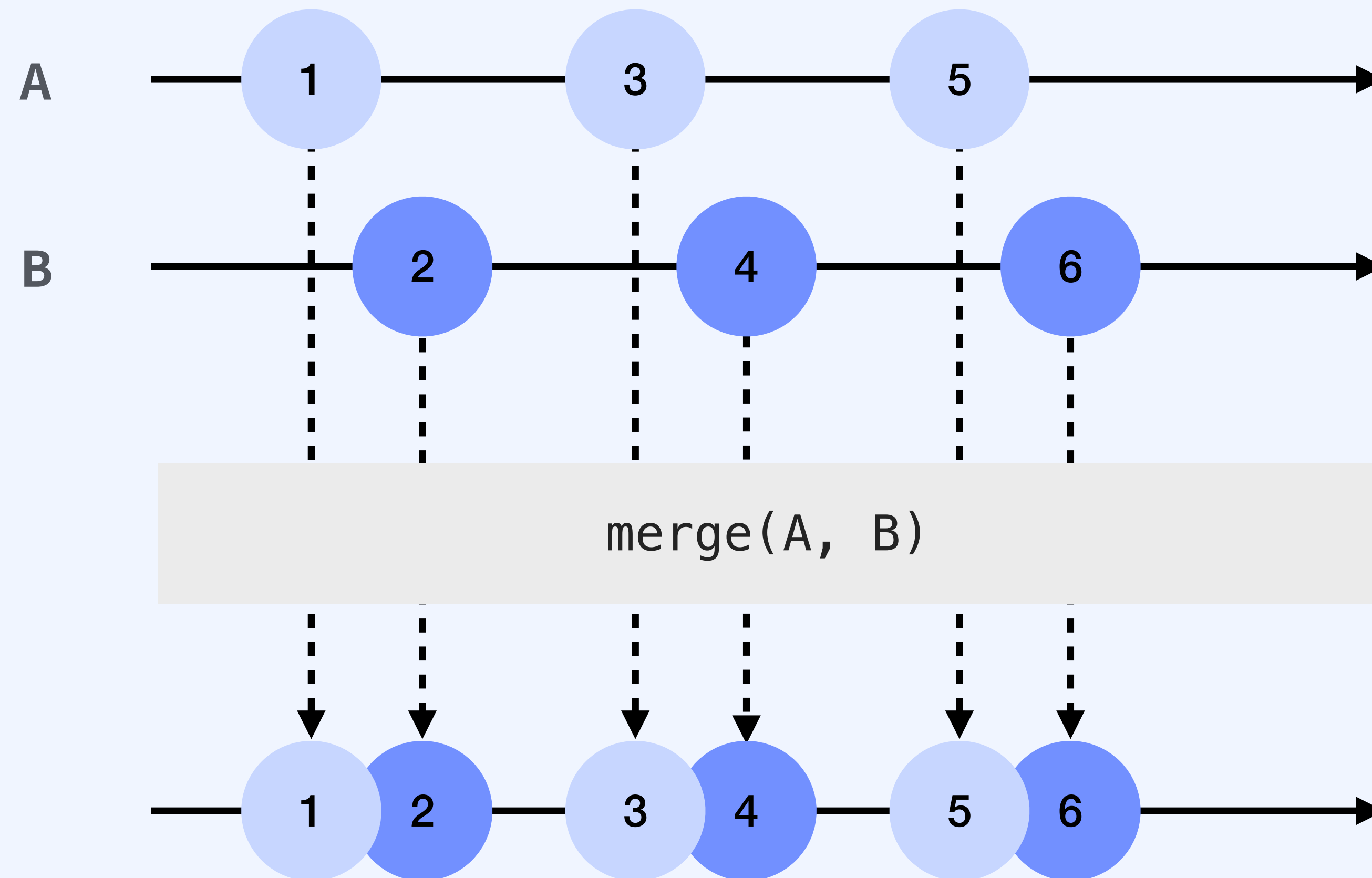
다음 카페/블로그 검색 앱 만들기

9 아웃트로 (정리, 현업 예시)

Combining Operator

1

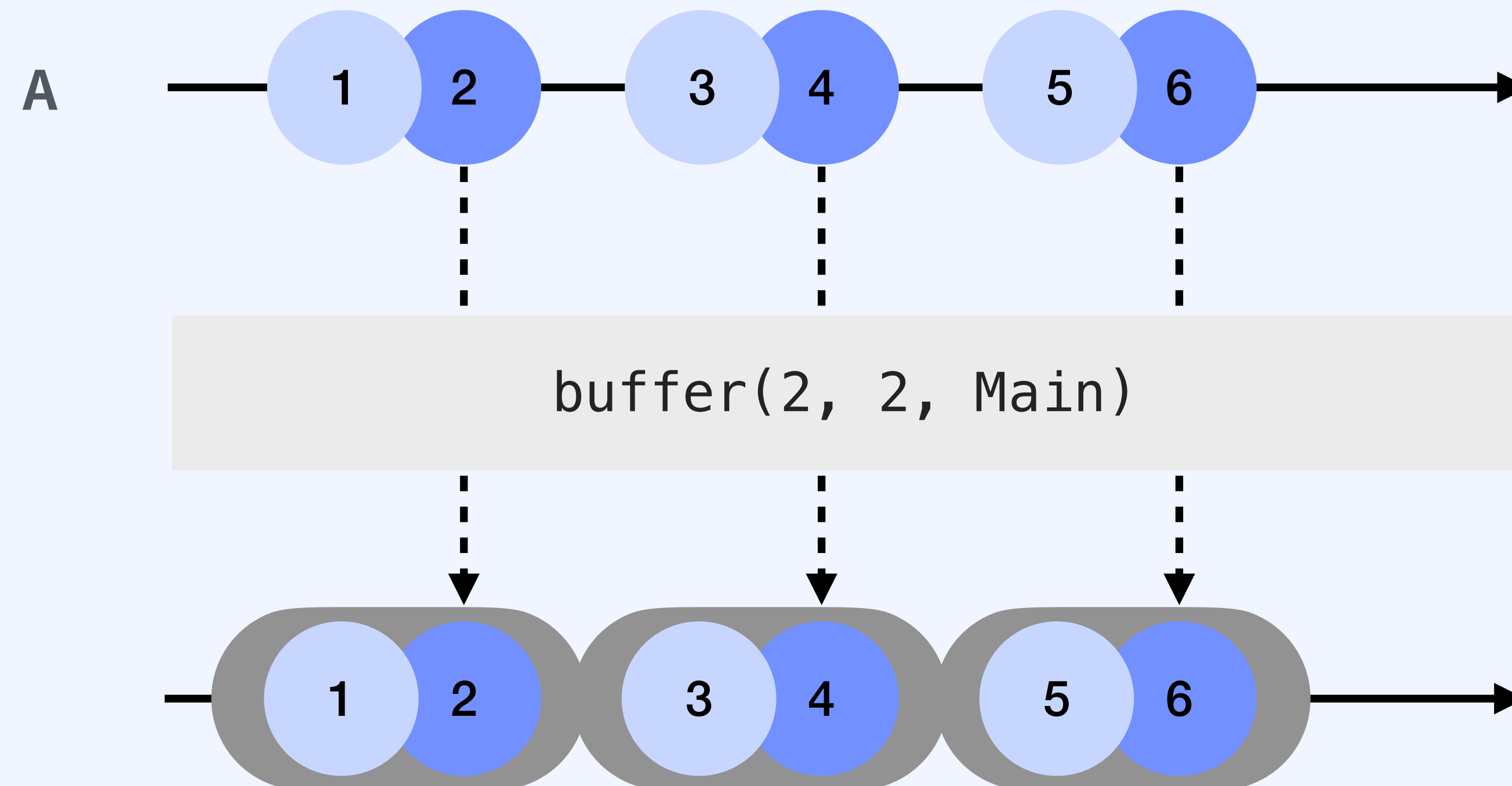
아웃트로 (정리, 현업
예시)



Time Based Operator

2

아웃트로 (정리, 현업
예시)



RxCocoa

3

아웃트로 (정리, 현업
예시)



에러 처리

4

아웃트로 (정리, 현업 예시)

catch



retry



정리

5

아웃트로 (정리, 현업
예시)

