Handling HTTP Errors with Observables

Deborah Kurata

Developer

https://www.youtube.com/@deborah_kurata

Lost connectivity

Issue a bad request (400)

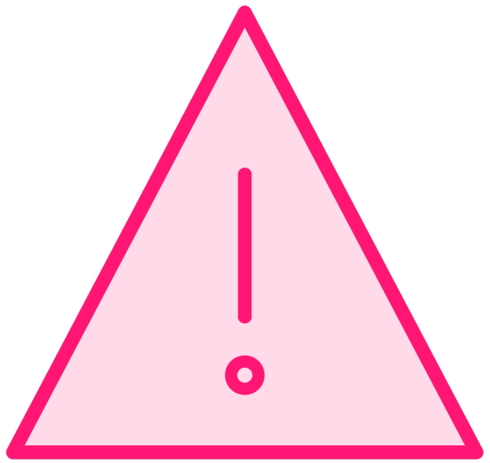
Incorrect URL (404)

Web server issue (500)

Resource requires privileges (401/403)

...

Observable Errors



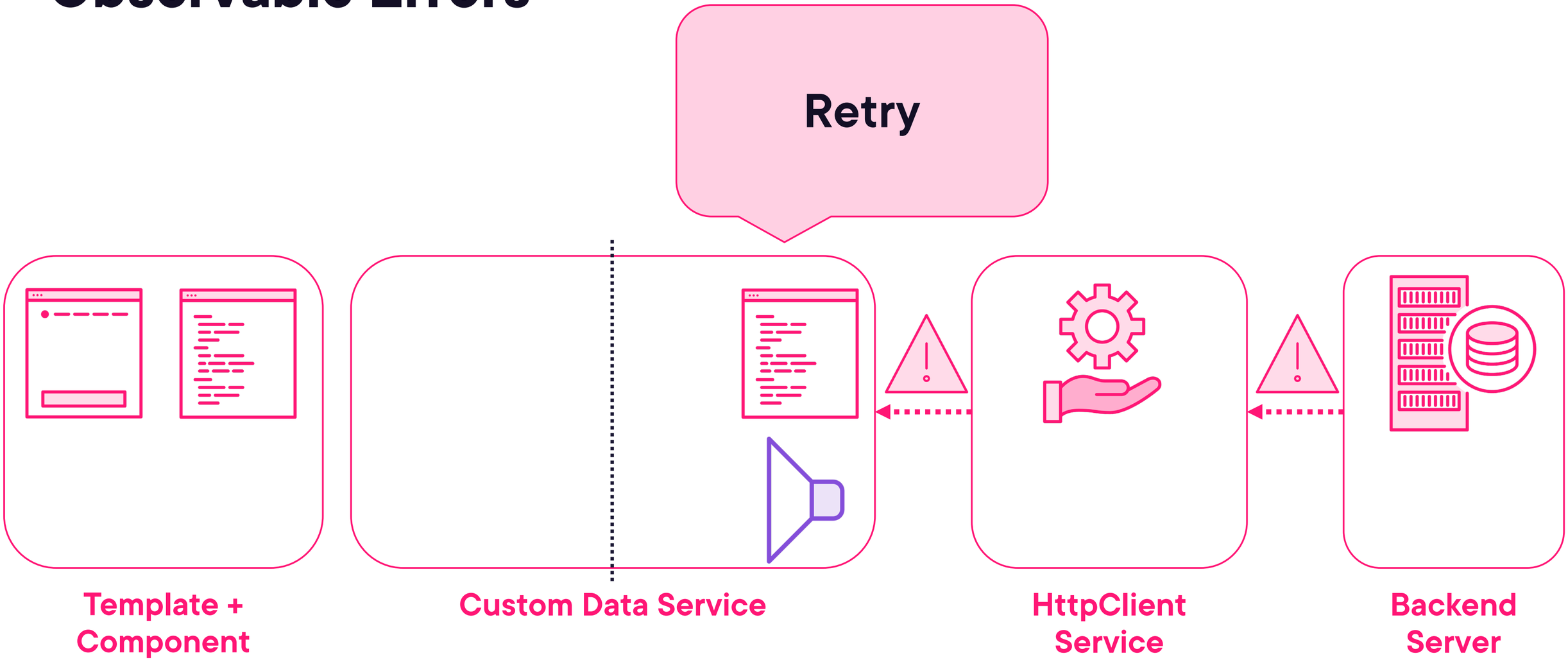
Error stops the observable

- It won't emit any more items
- We can't use it anymore

Catch the error

- Control how the error is processed
- Create a new replacement observable to continue

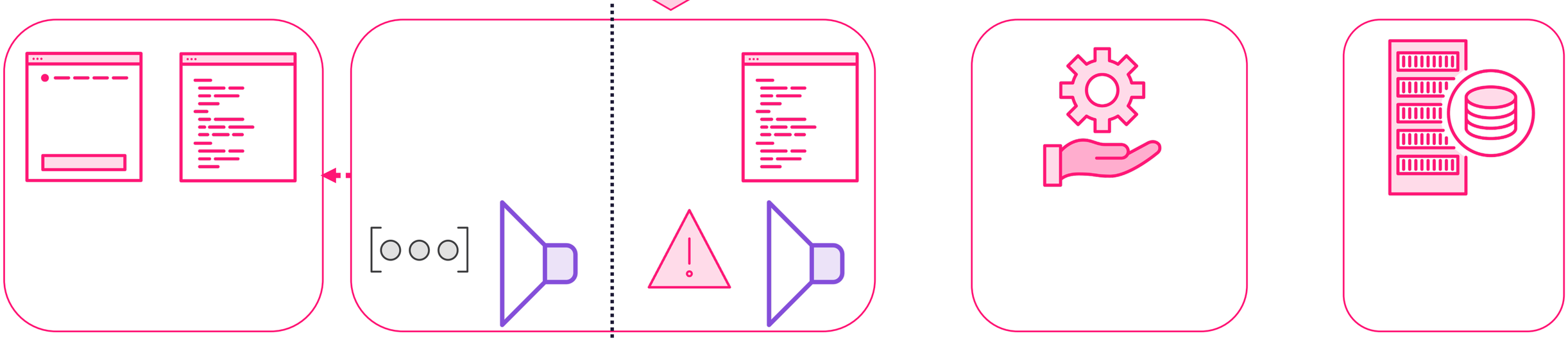
Observable Errors



Observable Errors

Replace the errored observable

```
export interface Result<T> {
  data: T  | undefined;
  error?: string;
}
```



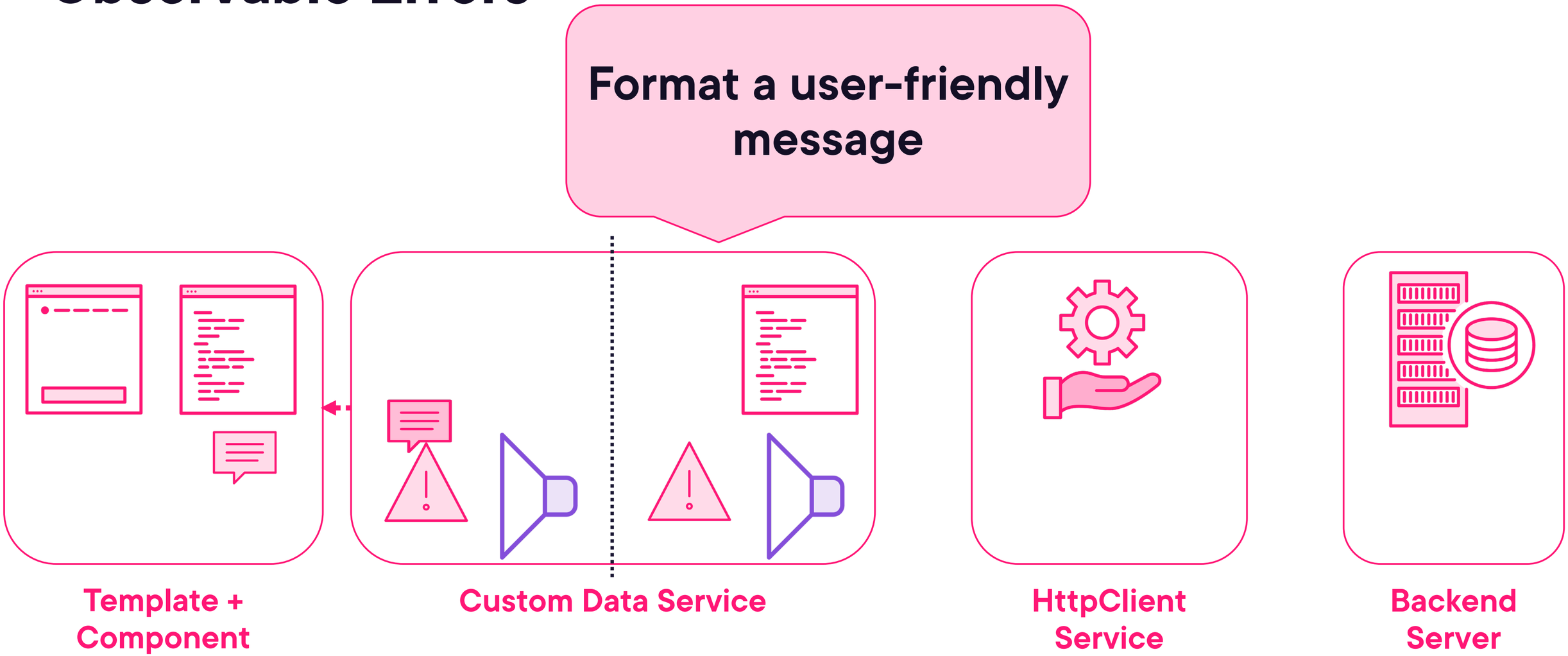
Template + Component

Custom Data Service

HttpClient Service

Backend Server

Observable Errors



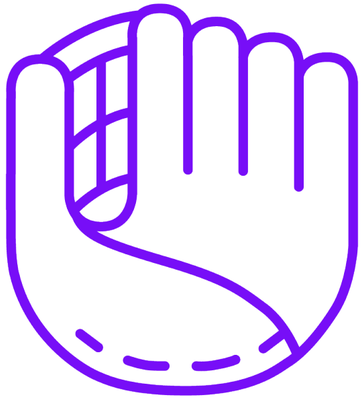
Overview

Discuss techniques for handling HTTP errors with observables

Examine the RxJS features that aid with error handling

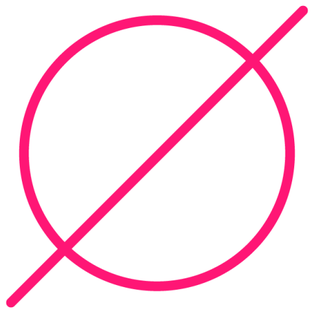
Write code to handle HTTP errors

RxJS Features that Aid with Error Handling



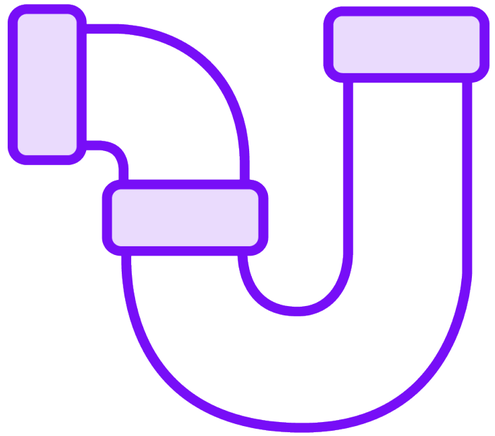
`catchError`

`throwError`



`EMPTY`

RxJS Operator: `catchError`



Catches any errors on an observable

```
catchError(err => this.handleError(err))
```

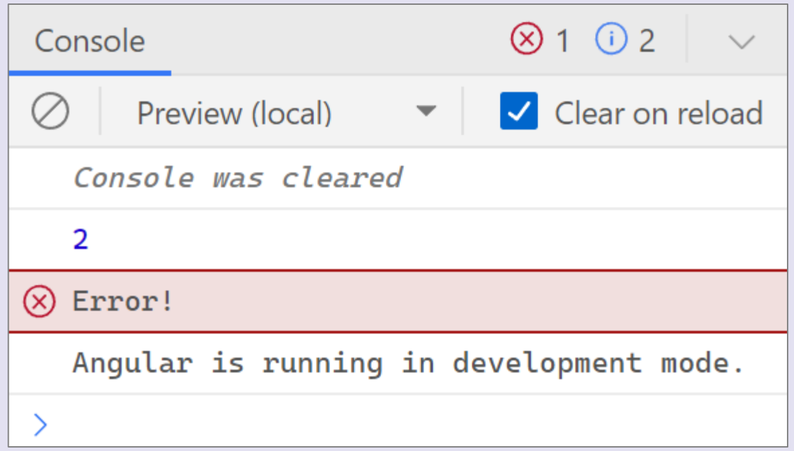
Must be after any operator that could generate an error

In the error handler:

- Replace the errored observable
- Emit default data or an error

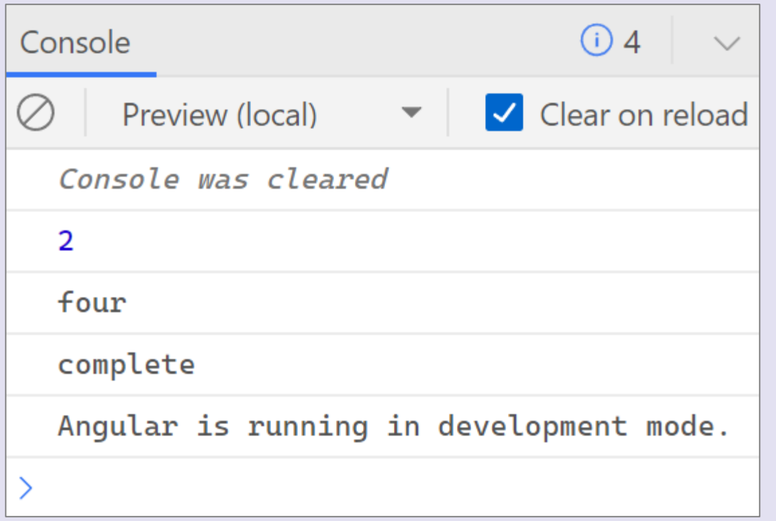
Without Catching the Error

```
of(2, 4, 6)
  .pipe(
   map(i => {
    if (i === 4) {
      throw 'Error!';
    }
    return i;
   })
  )
  .subscribe({
    next: x => console.log(x),
    error: err => console.error(err),
    complete: () => console.log('complete')
  });
```



Catching an Error

```
of(2, 4, 6)
  .pipe(
   map(i => {
    if (i === 4) {
      throw 'Error!';
    }
    return i;
   }),
   catchError(err => of('four'))
  )
  .subscribe({
    next: x => console.log(x),
    error: err => console.error(err),
    complete: () => console.log('complete')
  });
```



RxJS Creation Function: `throwError`

Creates a new replacement observable

```
throwError(() => 'Error!')
```

When subscribed, immediately emits an error notification

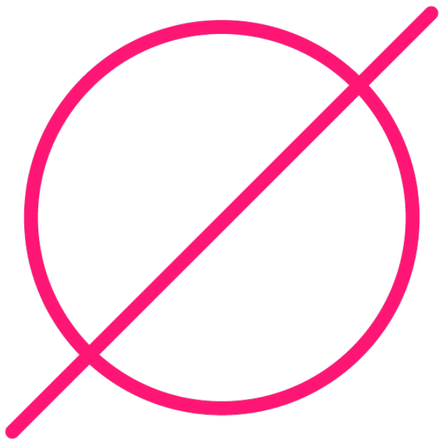
Used for

- Propagating an error

Or use the throw statement

```
throw 'Error!'
```

RxJS Constant: EMPTY



An observable that emits no items

```
return EMPTY;
```

Immediately emits a complete notification

Used for

- Returning an empty Observable

Demo

HTTP error handling

- Catching the error in a service

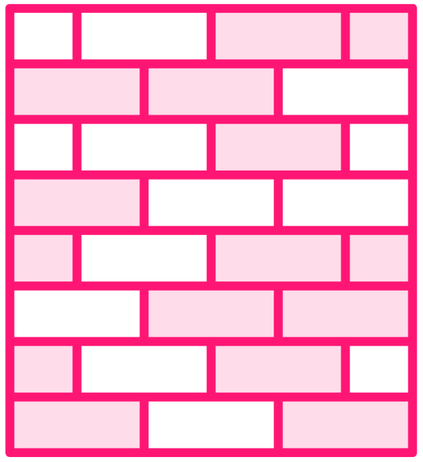
Demo

HTTP error handling

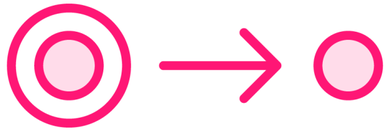
- Catching the error in a component

Best Practices

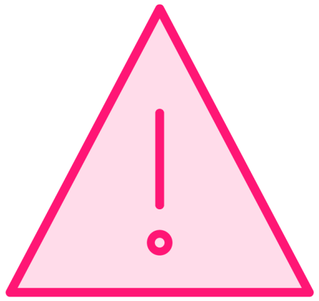
Just do it!



Use the HTTP request pipeline as the first wall of defense (catchError)



Create a replacement observable



Emit a default value or error notification

Error Handling

```
getProducts(): Observable<Product[]> {
  return this.http.get<Product[]>(this.productsUrl)
    .pipe(
      catchError(err => this.handleError(err))
    );
}
```

```
this.sub = this.productService.getProducts()
  .pipe(
    tap(products => this.products = products),
    catchError(err => {
      this.errorMessage = err;
      return EMPTY;
    })
  ).subscribe();
```

For More Information

Demo code

- https://github.com/DeborahK/angular-rxjs-signals-fundamentals

Angular documentation

- https://angular.io/guide/http-handle-request-errors

"Error Handling with Observables"

- https://youtu.be/L9kFTps_7Tk

Up Next:

Getting Related Data: `switchMap`, `concatMap`, and `mergeMap`