

RxJS in Angular: Reactive Development

Introduction



Deborah Kurata

Consultant | Speaker | Author | MVP | GDE

@deborahkurata



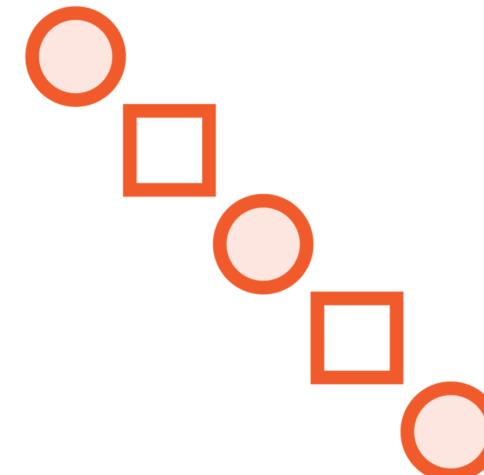




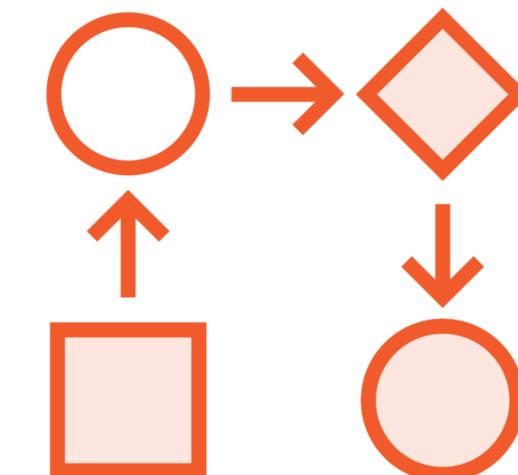
Goals for This Course



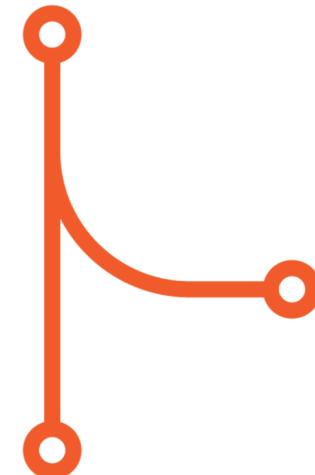
Add clarity



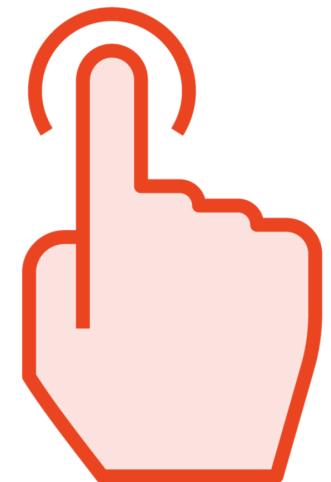
Examine reactive
patterns



Improve state
management



Merge RxJS
streams



React to user
actions



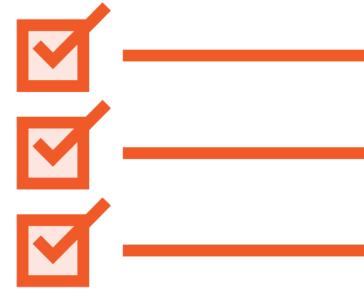
Minimize
subscriptions



Improve UI
performance



Prerequisites



Required

- Components
- Templates
- Services
- Observables / HttpClient



Helpful

- Angular: Getting Started



Not required

- Extensive knowledge of RxJS



Version Check



The examples in this course were created using:

- RxJS 7
- Angular v13



Version Check



This course is applicable to:

- RxJS v6 through RxJS v7
- Angular v6 through Angular v13



Module Overview

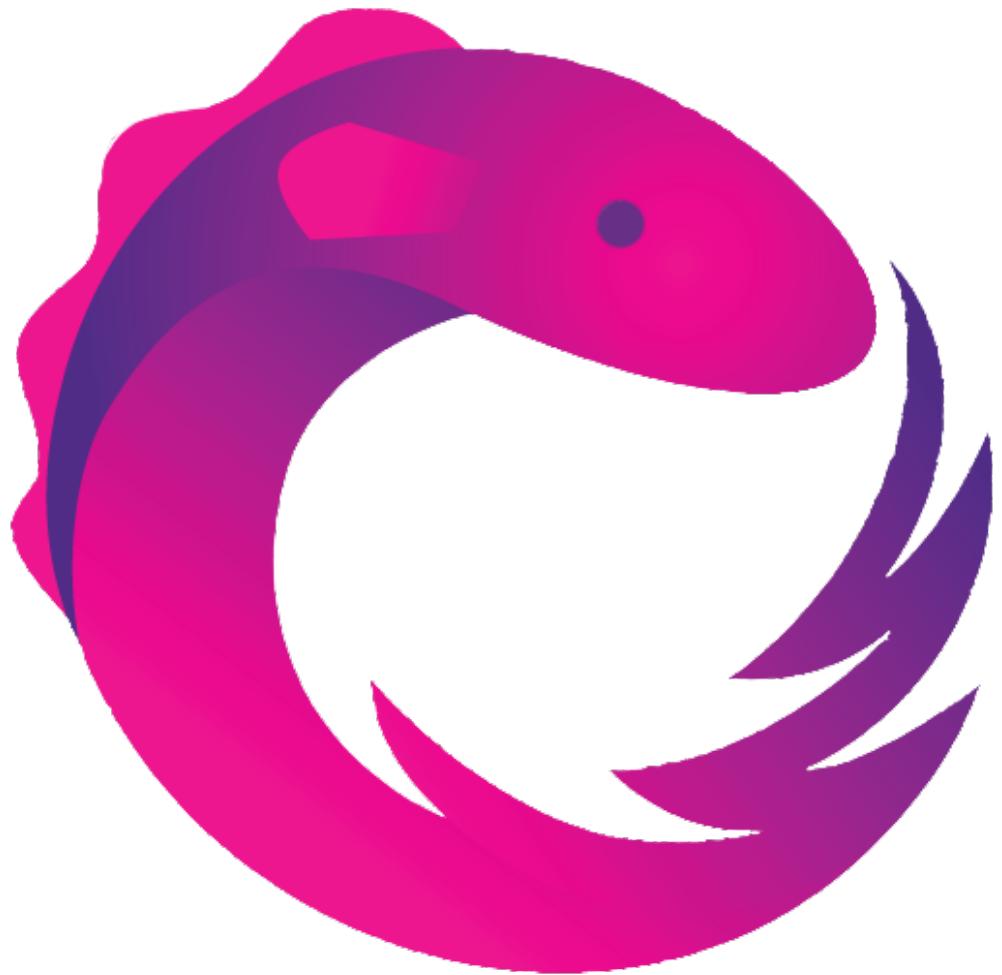


What is RxJS?

How is RxJS used in Angular?

What is reactive development?





What Is RxJS?

Reactive Extensions for JavaScript

**Reactive Extensions were originally developed
by Microsoft as Rx.NET**

RxJava, RxPy, Rx.rb, RxJS

Angular, React, Vue, etc.

JavaScript and TypeScript



"RxJS is a library for composing asynchronous and event-based programs by using observable sequences."

<https://rxjs.dev/guide/overview>





Observe and React to Data as It Flows through Time

Emit items

React to each emitted item

- Transform
- Filter
- Modify

Combine

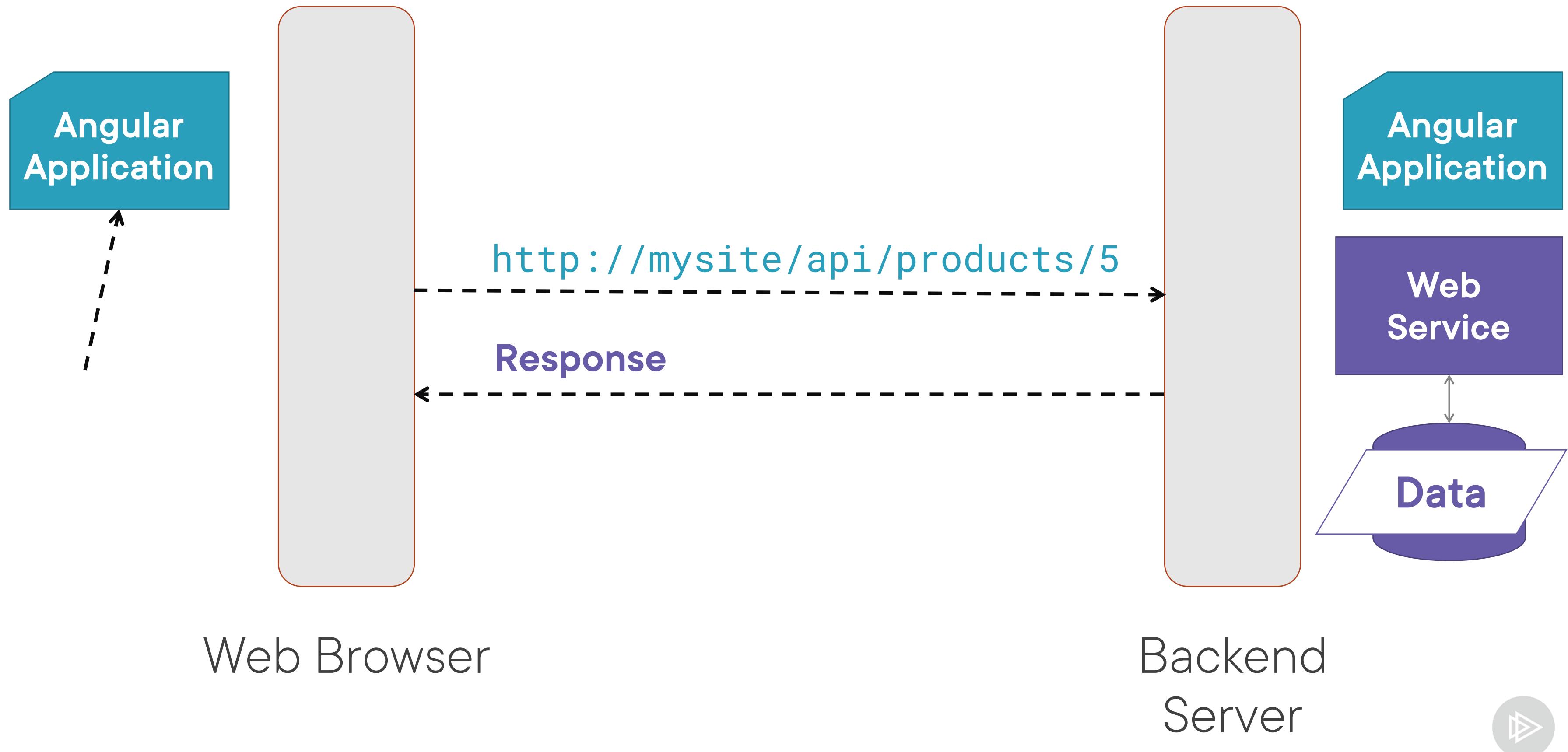
Cache



RxJS is designed to work with
asynchronous
actions and events



Asynchronous HTTP Request/Response



"RxJS is a library for composing asynchronous and event-based programs by using observable sequences."

<https://rxjs.dev/guide/overview>



"RxJS is a library for composing asynchronous and event-based programs by using observable sequences."

<https://rxjs.dev/guide/overview>

RxJS is a library for **observing** and **reacting to** data and events by using observable sequences.



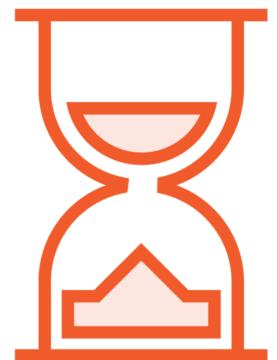
Why RxJS Instead Of...



Callbacks



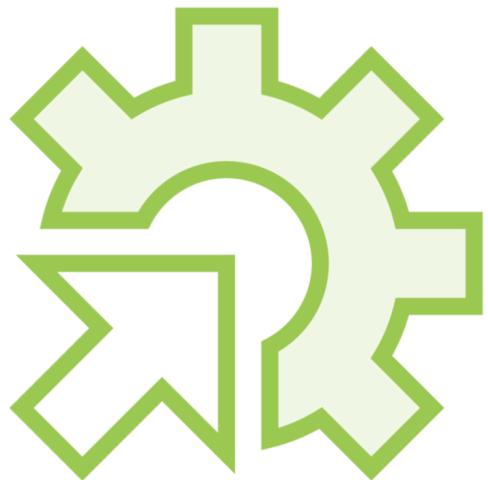
Promises



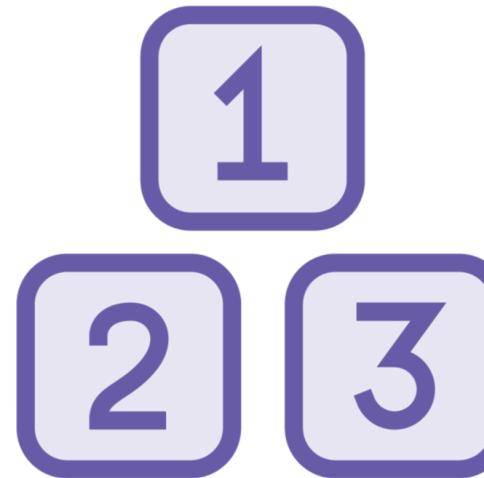
async/await



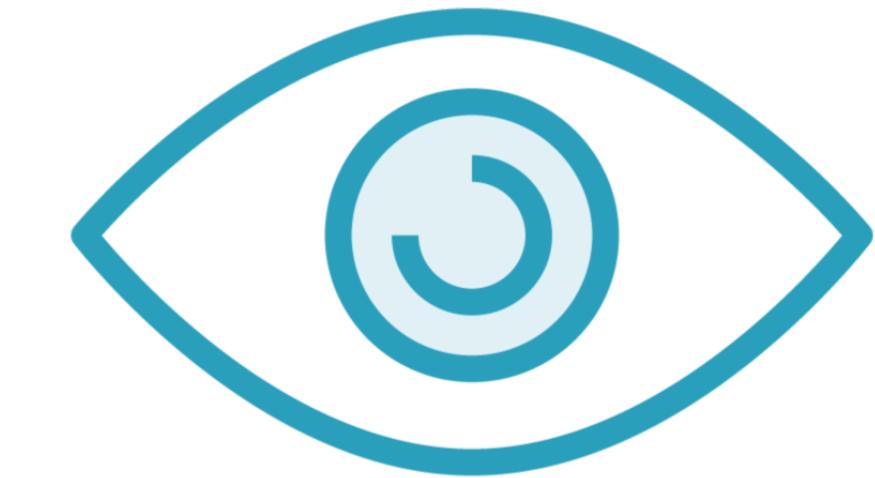
Why RxJS?



One technique to rule them all



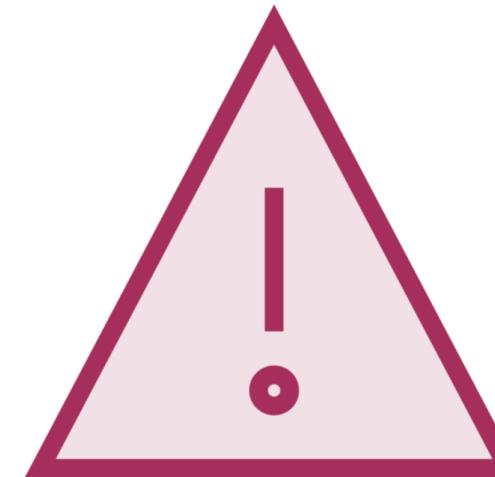
Compositional



Watchful



Lazy



Handles errors



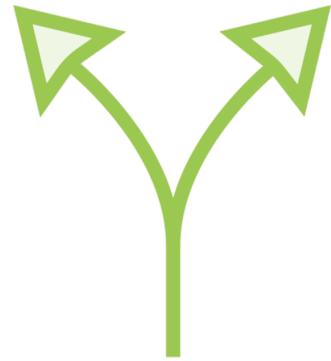
Cancellable



Angular uses RxJS.



How Is RxJS Used in Angular?



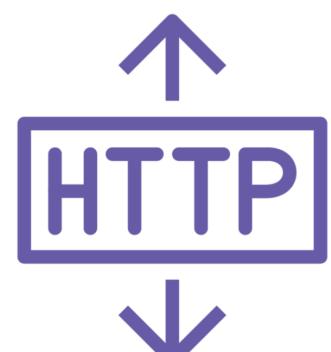
Routing

```
this.route.paramMap  
this.route.data  
this.router.events
```



Reactive Forms

```
this.productForm.valueChanges
```



HttpClient

```
getProducts(): Observable<Product[]> {  
  return this.http.get<Product[]>(this.url);  
}
```



Simple Math

$$x = 5$$

$$y = 3$$

$$z = x + y$$



What is z?



8



Simple Math

$x = 5$

$y = 3$

$z = x + y$

$x = 7$

Value is assigned when the expression is **first evaluated**



Now what is z?



8

z does **not react to changes** in x or y





Hammer X

Price: \$13.35
Category: Toolbox
Quantity: ▼
Cost: \$13.35

Cart Total

Subtotal:	\$13.35
Delivery:	\$5.99
Estimated	\$1.44
Tax:	
Total:	\$20.78

But what if we **want** to react to changes?

Hammer X

Price: \$13.35
Category: Toolbox
Quantity: ▼
Cost: \$40.05

Cart Total

Subtotal:	\$40.05
Delivery:	Free
Estimated	\$4.31
Tax:	
Total:	\$44.36



How to React to Changes in the Quantity?

Pseudo Code

```
item = "Hammer";
price = 13.35;
quantity = 1;
exPrice = price * quantity;
```



Option 1: Getter Function

Pseudo Code

```
item = "Hammer";
price = 13.35;
quantity = 1;
get exPrice() {
    return price * quantity;
}
```

The image shows a user interface for a shopping cart. On the left, a product detail view for a "Hammer" is displayed. It includes fields for Price (\$13.35), Category (Toolbox), Quantity (set to 3), and Cost (\$40.05). On the right, a "Cart Total" summary table provides the breakdown of the total cost.

Cart Total	
Subtotal:	\$13.35
Delivery:	\$5.99
Estimated	\$1.44
Tax:	
Total:	\$20.78



Option 2: Event Handler

Pseudo Code

```
item = "Hammer";
price = 13.35;
quantity = 1;
onQuantityChanged(newQty) {
    exPrice = price * newQty;
}
```

The image shows a mobile application interface. On the left, a product detail screen for a 'Hammer' is displayed. It includes fields for Price (\$13.35), Category (Toolbox), Quantity (set to 3), and Cost (\$40.05). On the right, a 'Cart Total' summary screen shows Subtotal (\$13.35), Delivery (\$5.99), Estimated (\$1.44), and a bolded Total (\$20.78).

Hammer	
Price:	\$13.35
Category:	Toolbox
Quantity:	3
Cost:	\$40.05

Cart Total	
Subtotal:	\$13.35
Delivery:	\$5.99
Estimated:	\$1.44
Tax:	
Total:	\$20.78



Option 3: RxJS

Pseudo Code

Declare an Observable

```
item = "Hammer";
price = 13.35;
quantity = 1;
qty$ = new Observable();
```

Emit when an action occurs

```
onQuantityChanged(newQty) {
  qty$.emit(newQty);
}
```

React to emissions

```
exPrice$ = qty$.pipe(
  map(q => q * price)
);
```



Propagate Changes

Pseudo Code

```
exPrice$ = qty$.pipe(  
    map(q => q * price)  
);
```

```
tax$ = exPrice$.pipe(  
    map(p => p * 10.75%)  
);
```

```
deliveryFee$ = exPrice$.pipe(  
    map(p => p < 30 ? 5.99 : 0)  
);
```



Composing Observables

Pseudo Code

```
totalPrice$ = combine(  
    exPrice$,  
    deliveryFee$,  
    tax$  
).pipe(  
    map([s,d,t] => s + d + t)  
);
```

Cart Total

Subtotal:	\$40.05
Delivery:	Free
Estimated	\$4.31
Tax:	
Total:	\$44.36



Bound Elements Are Notified

Hammer X

Price:	\$13.35
Category:	Toolbox
Quantity:	<input type="text" value="1"/>
Cost:	\$13.35

Cart Total

Subtotal:	\$13.35
Delivery:	\$5.99
Estimated	\$1.44
Tax:	
Total:	\$20.78



Bound Elements Are Notified

Hammer X

Price: \$13.35

Category: Toolbox

Quantity:

Cost: \$40.05

Cart Total

Subtotal:	\$13.35
Delivery:	\$5.99
Estimated	\$1.44
Tax:	
Total:	\$20.78



Programming Paradigms

Pseudo Code

Imperative

```
x = 5  
y = 3  
z = x + y  
x = 7
```

// z is 8

Value is assigned when the expression is first evaluated

Reactive

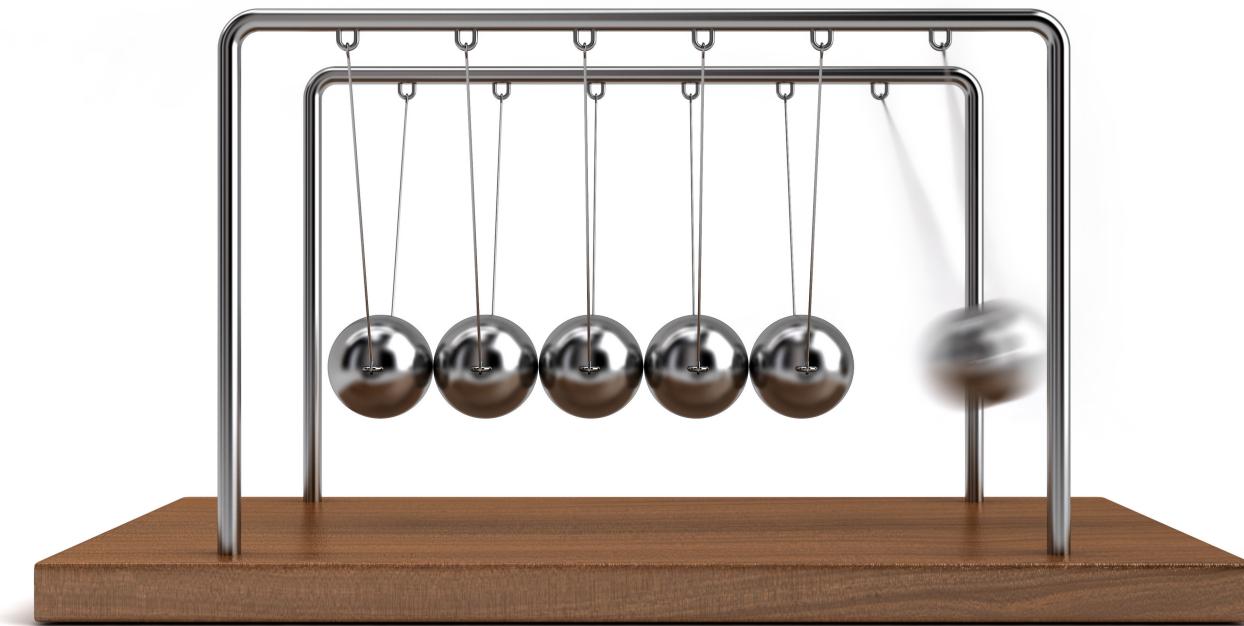
```
x = 5  
y = 3  
z$ = x + y  
x = 7
```

// z\$ emits 8, then 10

**React to changes
Changes are propagated**



What Is Reactive Development?



A **declarative** programming paradigm concerned with **data streams** and the **propagation of change**.

https://en.wikipedia.org/wiki/Reactive_programming



Reactive Development

Code is **reactive** when an
input change leads to an
automatic change in output

Hammer X

Price: \$13.35

Category: Toolbox

Quantity: ▼

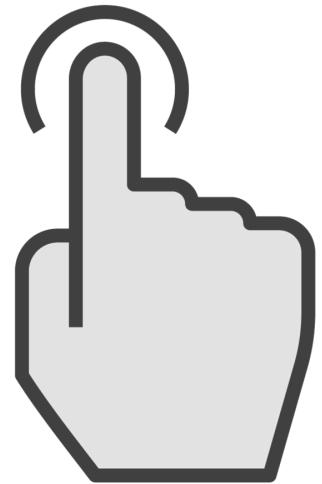
Cost: \$40.05

Cart Total

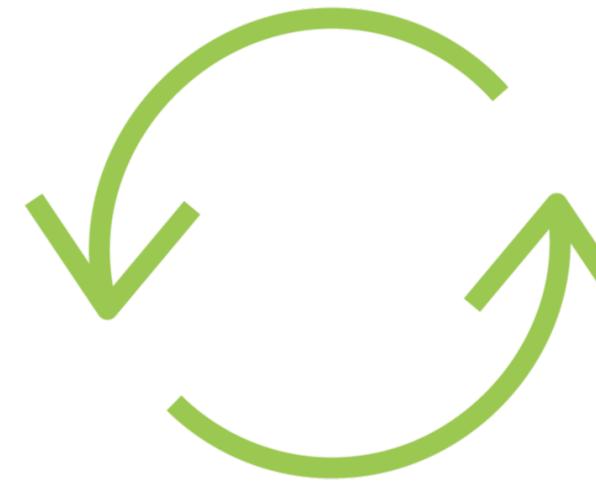
Subtotal:	\$40.05
Delivery:	Free
Estimated Tax:	\$4.31
Total:	\$44.36



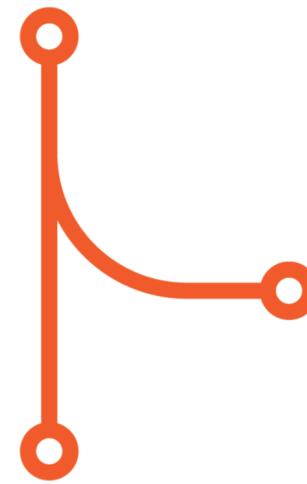
Reactive Development



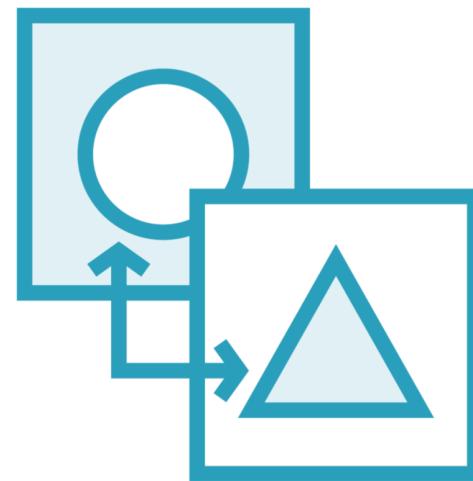
React to user actions



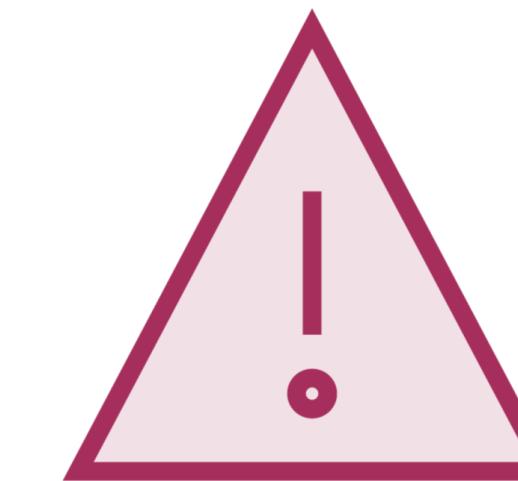
React to state changes



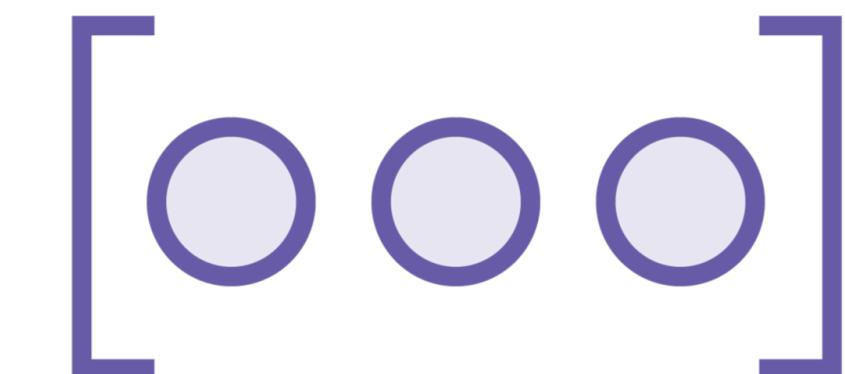
Combine data streams



Communicate between components



Be resilient to failure



Manage state





Coming up next...

RxJS Terms and Syntax

