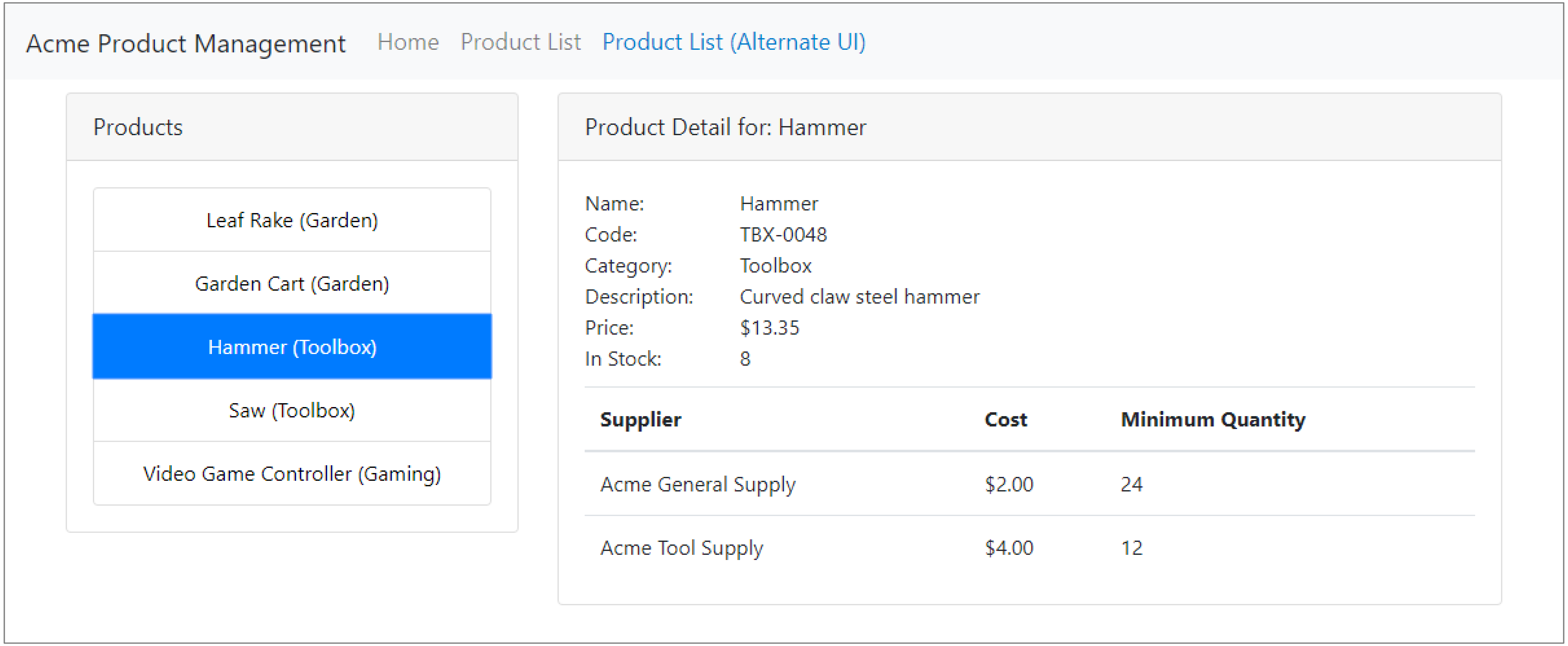
Reacting to Actions: Examples

Deborah Kurata

CONSULTANT | SPEAKER | AUTHOR | MVP | GDE

@deborahkurata | blogs.msmvps.com/deborahk/



Acme Product Management

Home Product List Product List (Alternate UI)

Products

Leaf Rake (Garden)

Garden Cart (Garden)

Hammer (Toolbox)

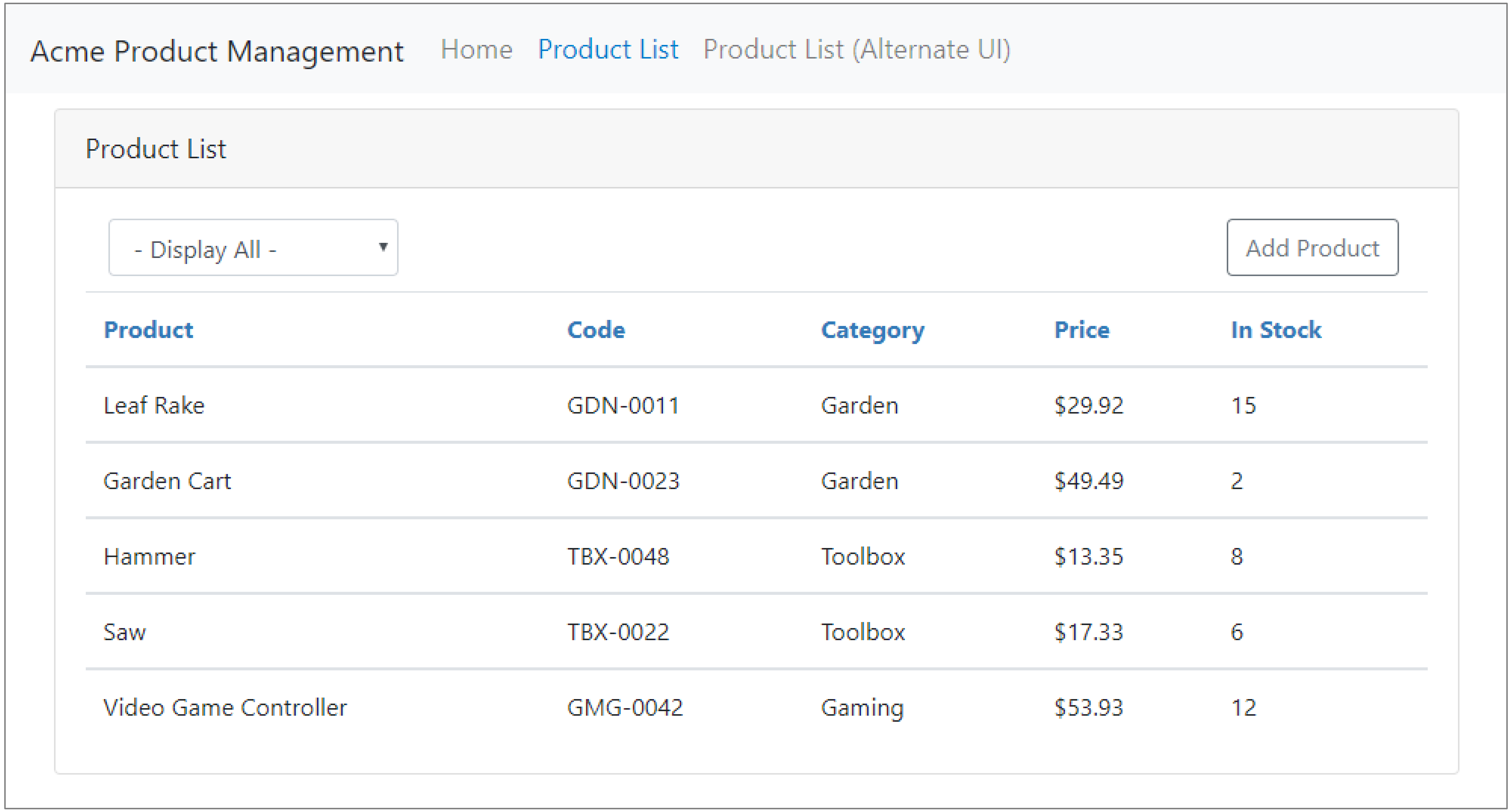
Saw (Toolbox)

Video Game Controller (Gaming)

Product Detail for: Hammer

Name:	Hammer
Code:	TBX-0048
Category:	Toolbox
Description:	Curved claw steel hammer
Price:	$13.35
In Stock:	8

Supplier	Cost	Minimum Quantity
Acme General Supply	$2.00	24
Acme Tool Supply	$4.00	12



Product List

Product	Code	Category	Price	In Stock
Leaf Rake	GDN-0011	Garden	$29.92	15
Garden Cart	GDN-0023	Garden	$49.49	2
Hammer	TBX-0048	Toolbox	$13.35	8
Saw	TBX-0022	Toolbox	$17.33	6
Video Game Controller	GMG-0042	Gaming	$53.93	12

Module Overview

React to selections

React to errors

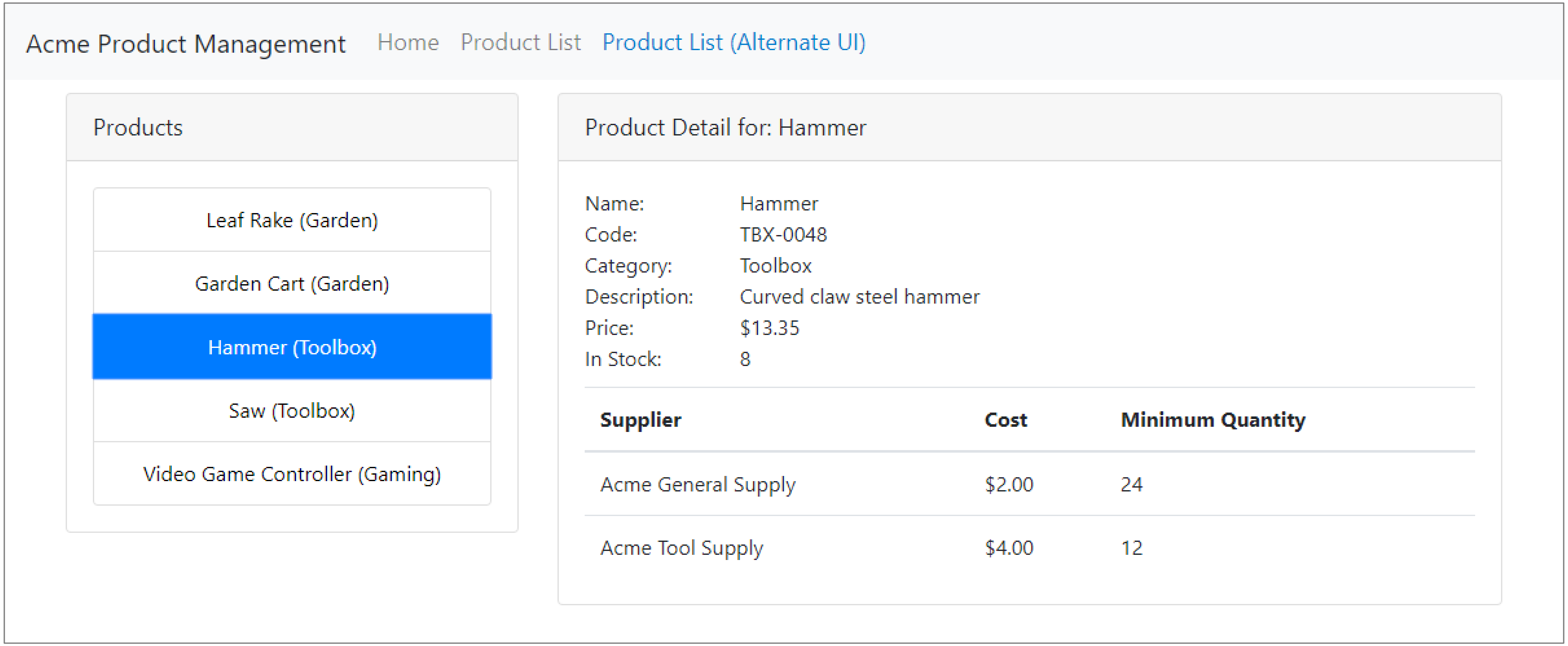
React to add operations

RxJS Features

`merge`

`scan`

Reacting to a Selection



Acme Product Management

Products

- Leaf Rake (Garden)
- Garden Cart (Garden)
- Hammer (Toolbox)
- Saw (Toolbox)
- Video Game Controller (Gaming)

Product Detail for: Hammer

Name:	Hammer
Code:	TBX-0048
Category:	Toolbox
Description:	Curved claw steel hammer
Price:	$13.35
In Stock:	8

Supplier	**Cost**	**Minimum Quantity**
Acme General Supply	$2.00	24
Acme Tool Supply	$4.00	12

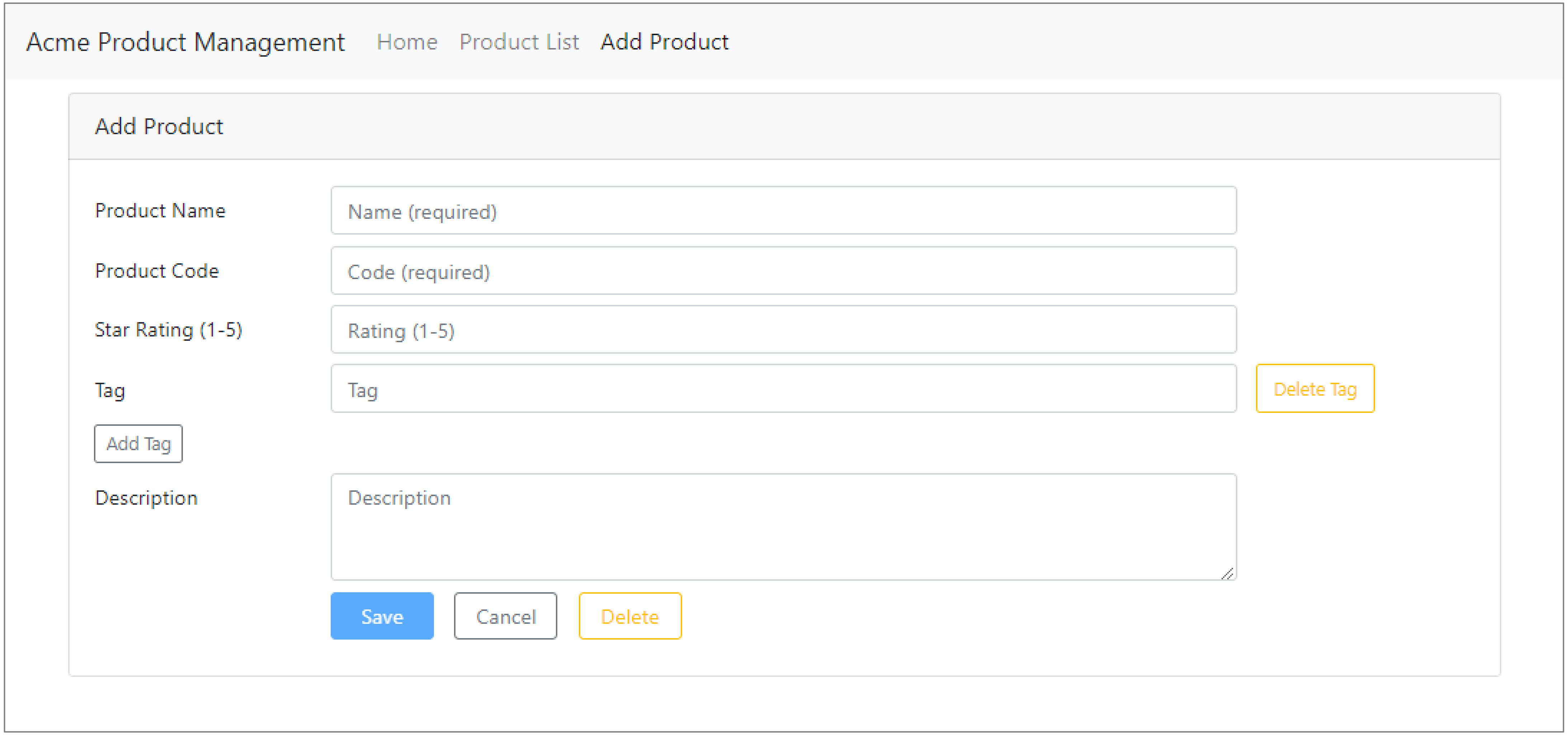
Demo

Reacting to a selection

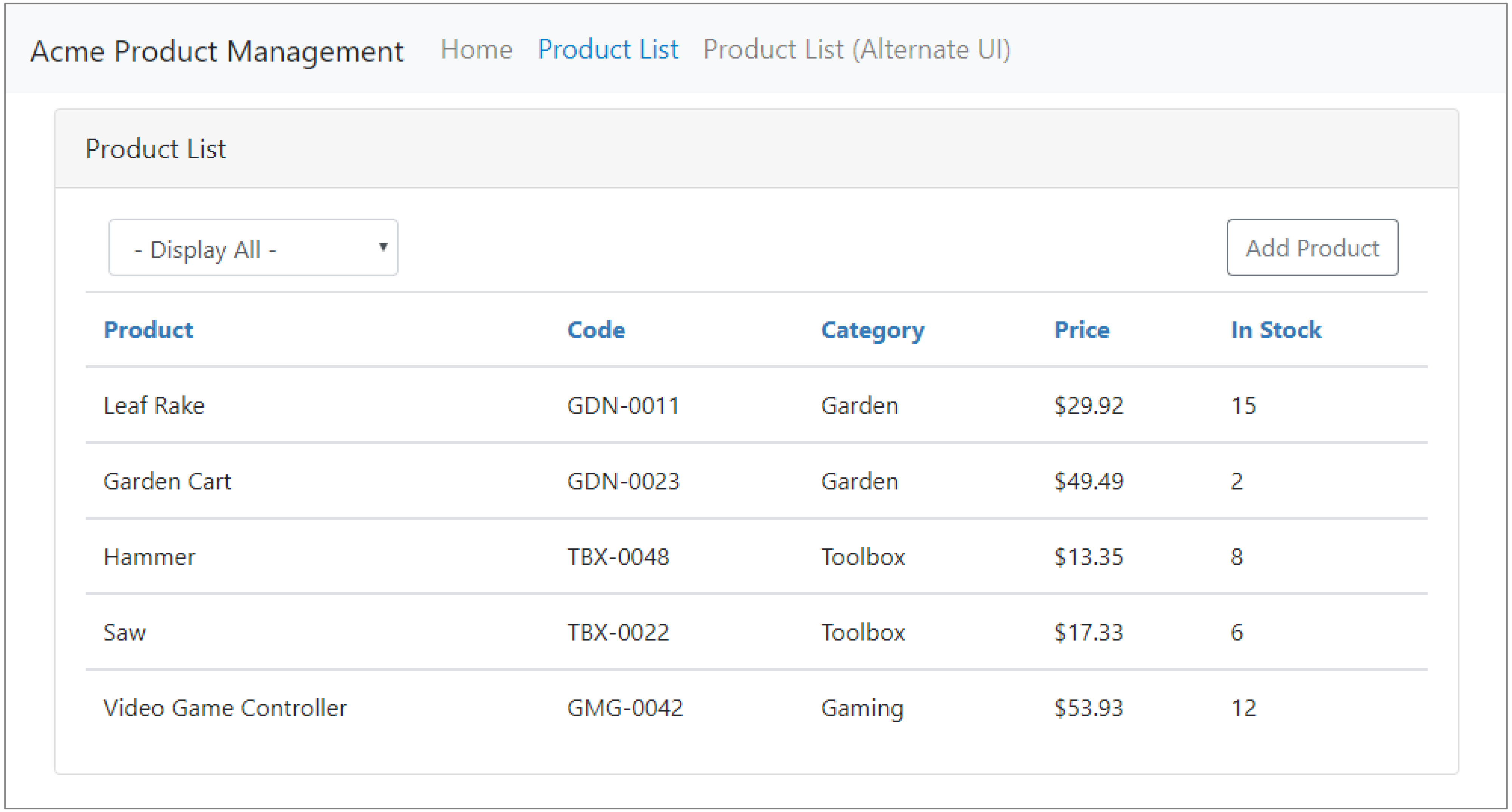
Demo

Reacting to an error

Reacting to an Add Operation



Reacting to an Add Operation



Acme Product Management   Home   Product List   Product List (Alternate UI)

Product List

- Display All -   Add Product

Product	Code	Category	Price	In Stock
Leaf Rake	GDN-0011	Garden	$29.92	15
Garden Cart	GDN-0023	Garden	$49.49	2
Hammer	TBX-0048	Toolbox	$13.35	8
Saw	TBX-0022	Toolbox	$17.33	6
Video Game Controller	GMG-0042	Gaming	$53.93	12

RxJS Creation Function: `merge`

Combines multiple streams by merging their emissions

```
merge(a$, b$, c$)
```

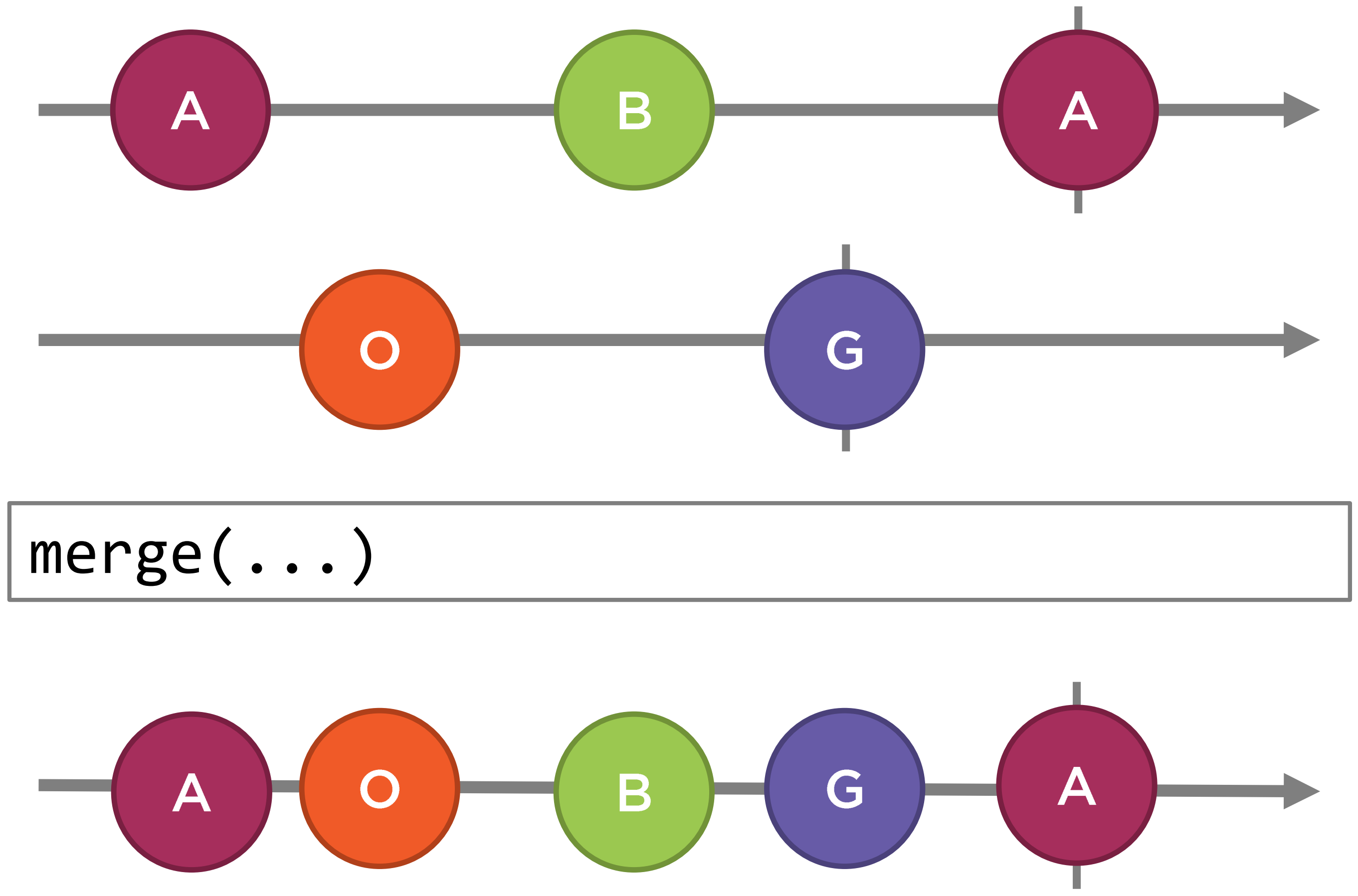
Static creation function, not a pipeable operator

Used for

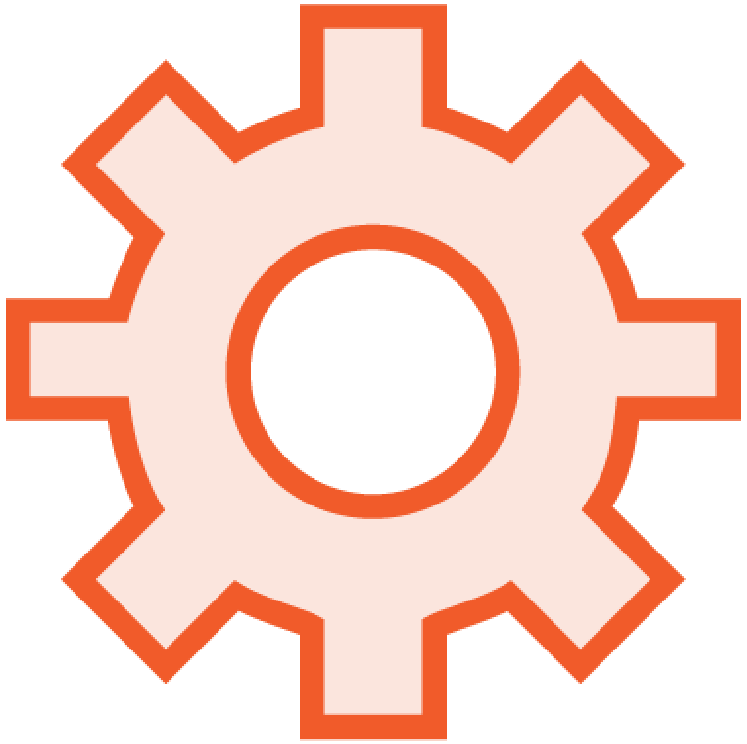
- Combining sequences of similar types to blend their emitted values

Marble Diagram: `merge`

```
merge(
  of('A', 'B', 'A'),
  of('O', 'G'),
  );
```



RxJS Creation Function: `merge`



`merge` is a combination function

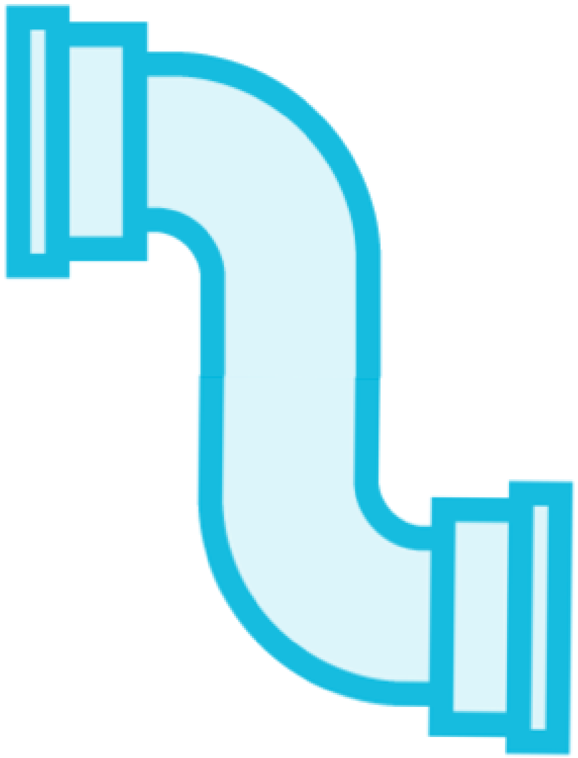
- Takes in a set of streams, subscribes
- Creates an output stream

When an item is emitted from any stream

- Item is emitted to the output stream

Completes when all input streams complete

RxJS Operator: **scan**



Accumulates items in a stream

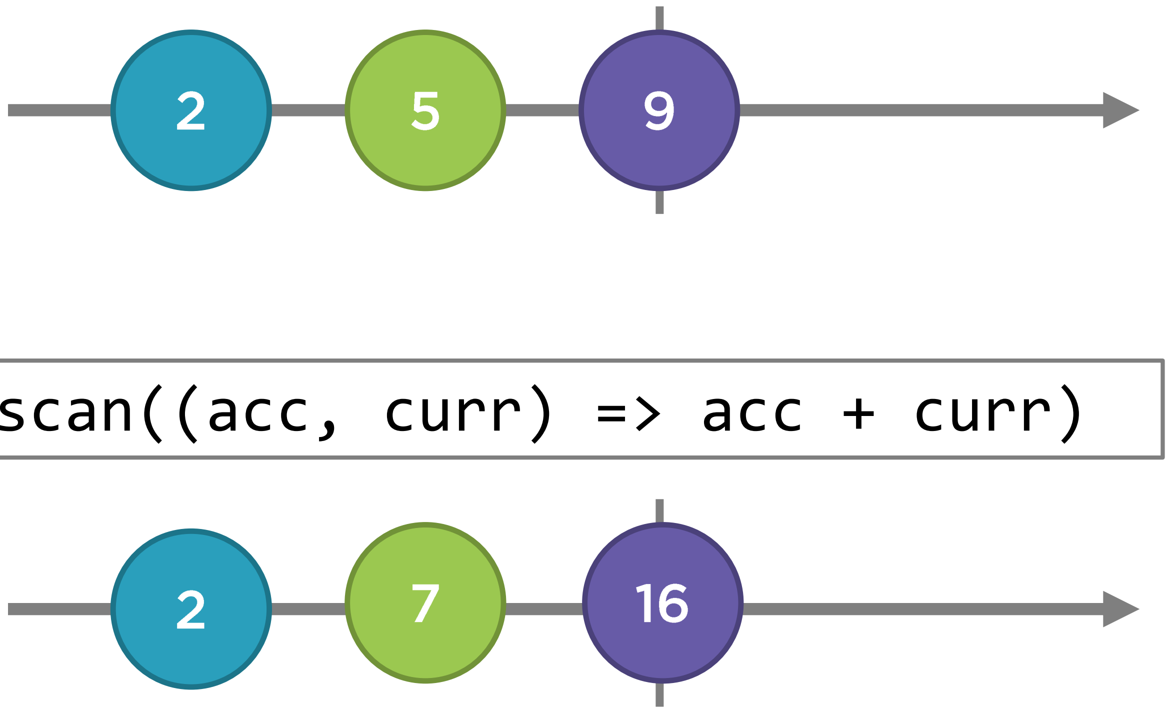
```
scan((acc, curr) => acc + curr)
```

Used for

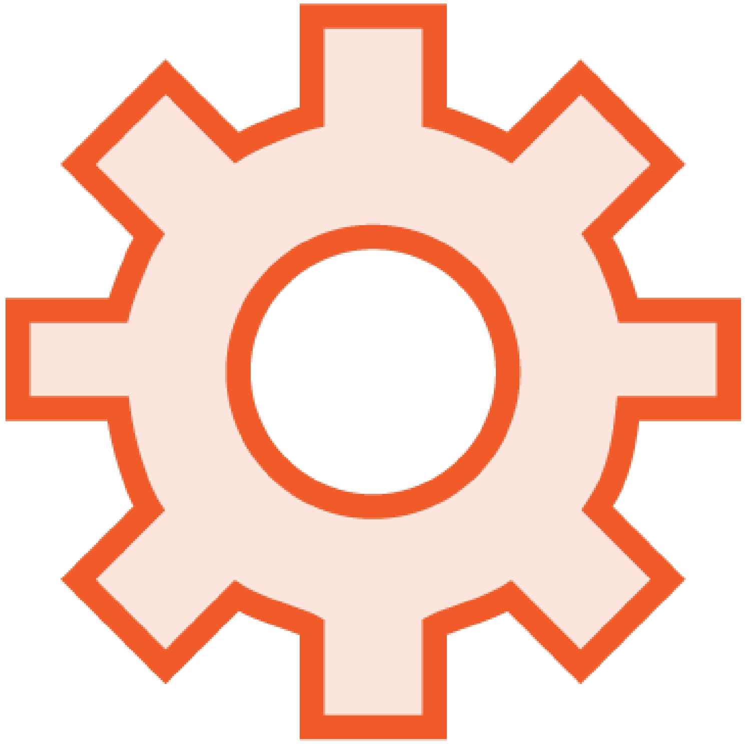
- Totaling amounts
- Accumulating items into an array

Marble Diagram: **scan**

```
of(2, 5, 9)
  .pipe(
    scan((acc, curr) => acc + curr),
  );
```



RxJS Operator: **scan**



scan **is a transformation operator**

- Takes in an input stream, subscribes
- Creates an output stream

When a item is emitted

- Item is accumulated as specified by a provided function
- Intermediate result is emitted to the output stream

Reacting to an Add Operation

```
merge(
  this.products$,
  this.insertAction$
)
 .pipe(
    scan((acc: Product[],
          value: Product) =>
           [...acc, value])
  );
```

Reacting to an Add Operation

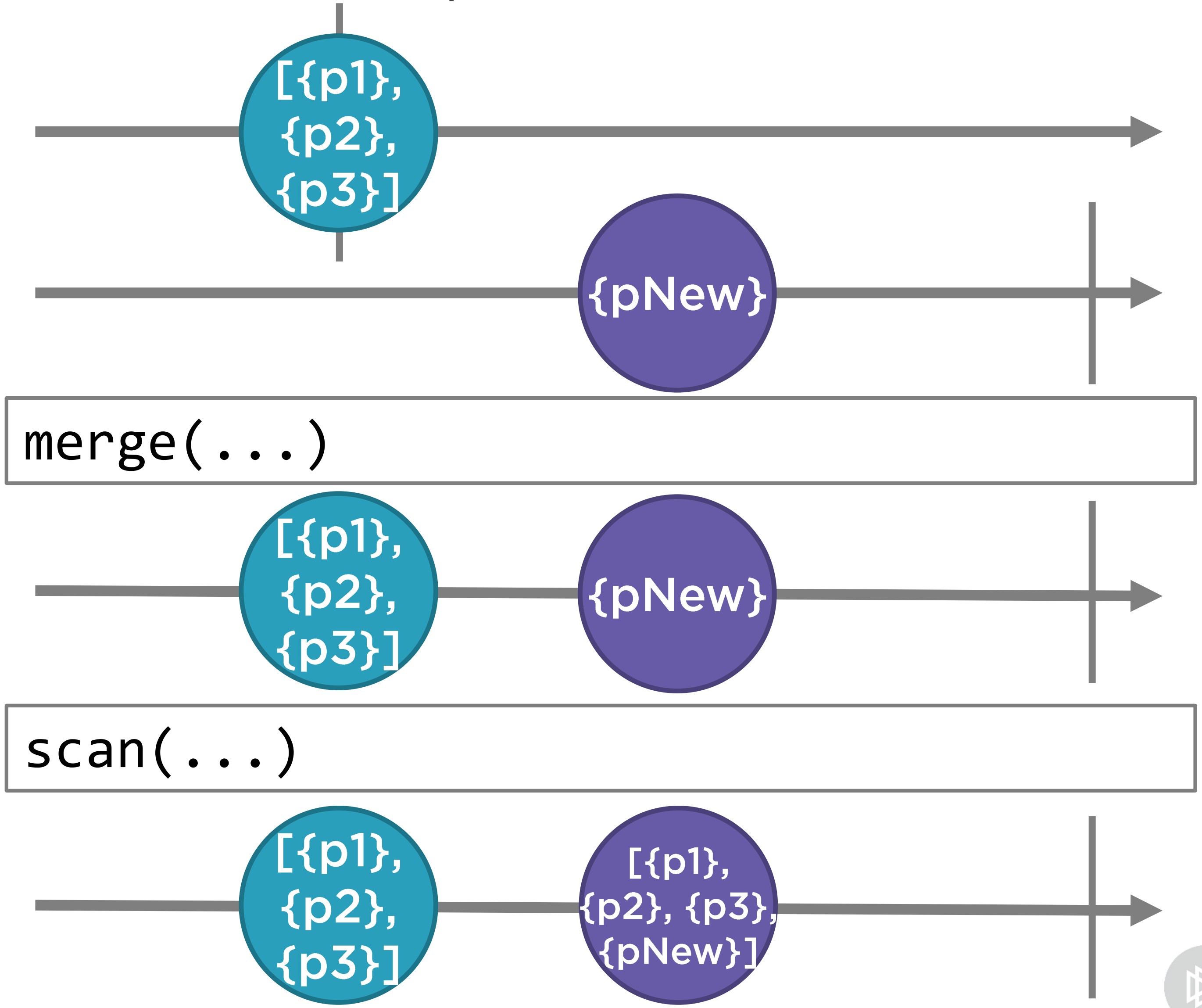
```
merge(
  this.products$,


  this.insertAction$

)




 .pipe(
    scan((acc: Product[],
            value: Product) =>
             [...acc, value])
  );
```



Demo

Reacting to an add operation

Reacting to Actions

Create an action stream (Subject/BehaviorSubject)

```
selSubject = new Subject<number>();
selectedAction$ = this.selSubject.asObservable();
```

Combine the action and data streams

```
products$ = combineLatest([
  this.productService.products$,
  this.selectedAction$
]).pipe(...);
```

Emit a value to the action stream when an action occurs

```
onSelected(id): void {
  this.selSubject.next(+id);
}
```

Reacting to a Selection

```
private pSelSubject = new BehaviorSubject<number>(0);
pSelAction$ = this.pSelSubject.asObservable();

selectedProduct$ = combineLatest([
  this.productsWithCategory$,
  this.pSelAction$
])
 .pipe(
   map(([products, selectedProductId]) =>
      products.find(product => product.id ===
                                  selectedProductId)
   )
 );
```

```
selProdChanged(selectedProductId){
  this.pSelSubject.next(selectedProductId);
}
```

Reacting to an Error

```
private errorSubject = new Subject<string>();
error$ = this.errorSubject.asObservable();

product$ = this.productService.selectedProduct$
  .pipe(
    catchError(err => {
      this.errorSubject.next(err);
      return EMPTY;
  })
);
```

```
<div
  *ngIf="error$ | async as errorMessage">
  {{errorMessage}}
</div>
```

RxJS Features

merge: **Merges the emissions of multiple streams**

```
merge(a$, b$, c$)
```

scan: **Applies an accumulator function**

```
scan((acc, curr) => acc + curr)
```

Reacting to an Add Operation

```
merge(
  this.products$,
  this.insertAction$
)
 .pipe(
    scan((acc: Product[],
          value: Product) =>
           [...acc, value])
 );
```