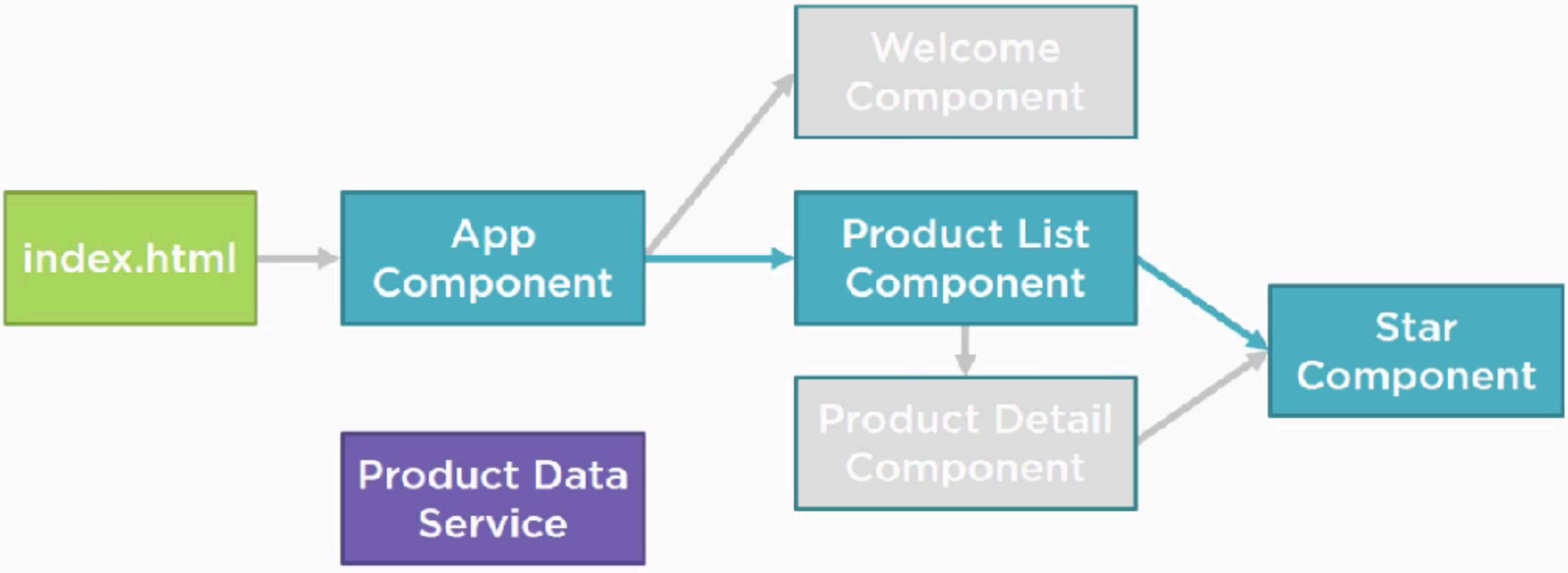
Application Architecture



Product List View

Product List

Filter by:

Show Image

Product	Code	Available	Price	5 Star Rating
Leaf Rake	GDN-0011	Mar 19, 2016	$19.95	★★★
Garden Cart	GDN-0023	Mar 18, 2016	$32.99	★★★★
Hammer	TBX-0048	May 21, 2016	$8.99	★★★★★
Saw	TBX-0022	May 15, 2016	$11.55	★★★★
Video Game Controller	GMG-0042	Oct 15, 2015	$35.95	★★★★

Product List View

Product List

Filter by: am ×

Filtered by: am

Show Image

Product	Code	Available	Price	5 Star Rating
Hammer	TBX-0048	May 21, 2016	$8.99	★★★★★
Video Game Controller	GMG-0042	Oct 15, 2015	$35.95	★★★★½

Product List View

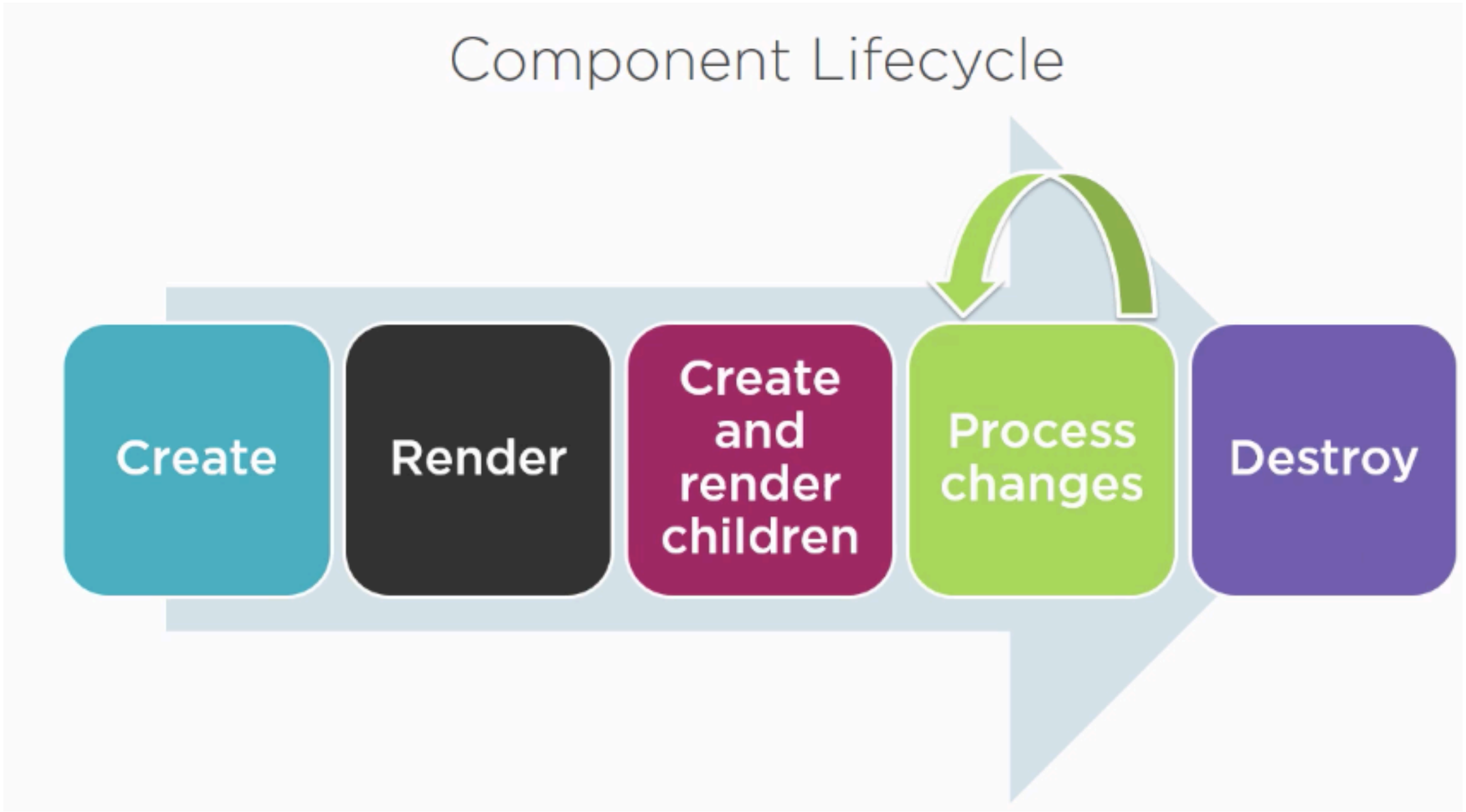
Product List

Filter by: am

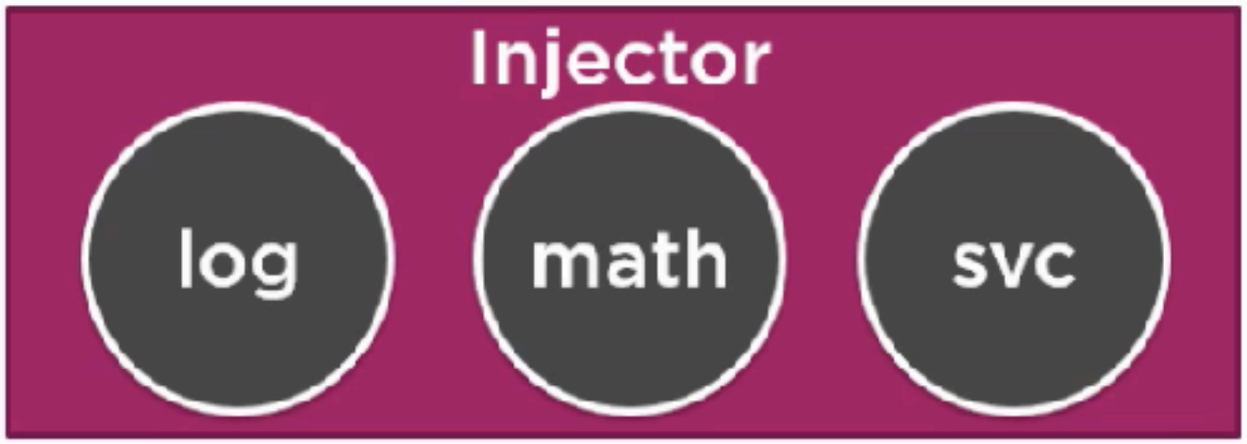
Filtered by: am

Hide Image

Product	Code	Available	Price	5 Star Rating
 Hammer	TBX-0048	May 21, 2016	$8.99	★★★★★
 Video Game Controller	GMG-0042	Oct 15, 2015	$35.95	★★★★½


Component Lifecycle
Create
Render
Create and render children
Process changes
Destroy

How Does It Work?



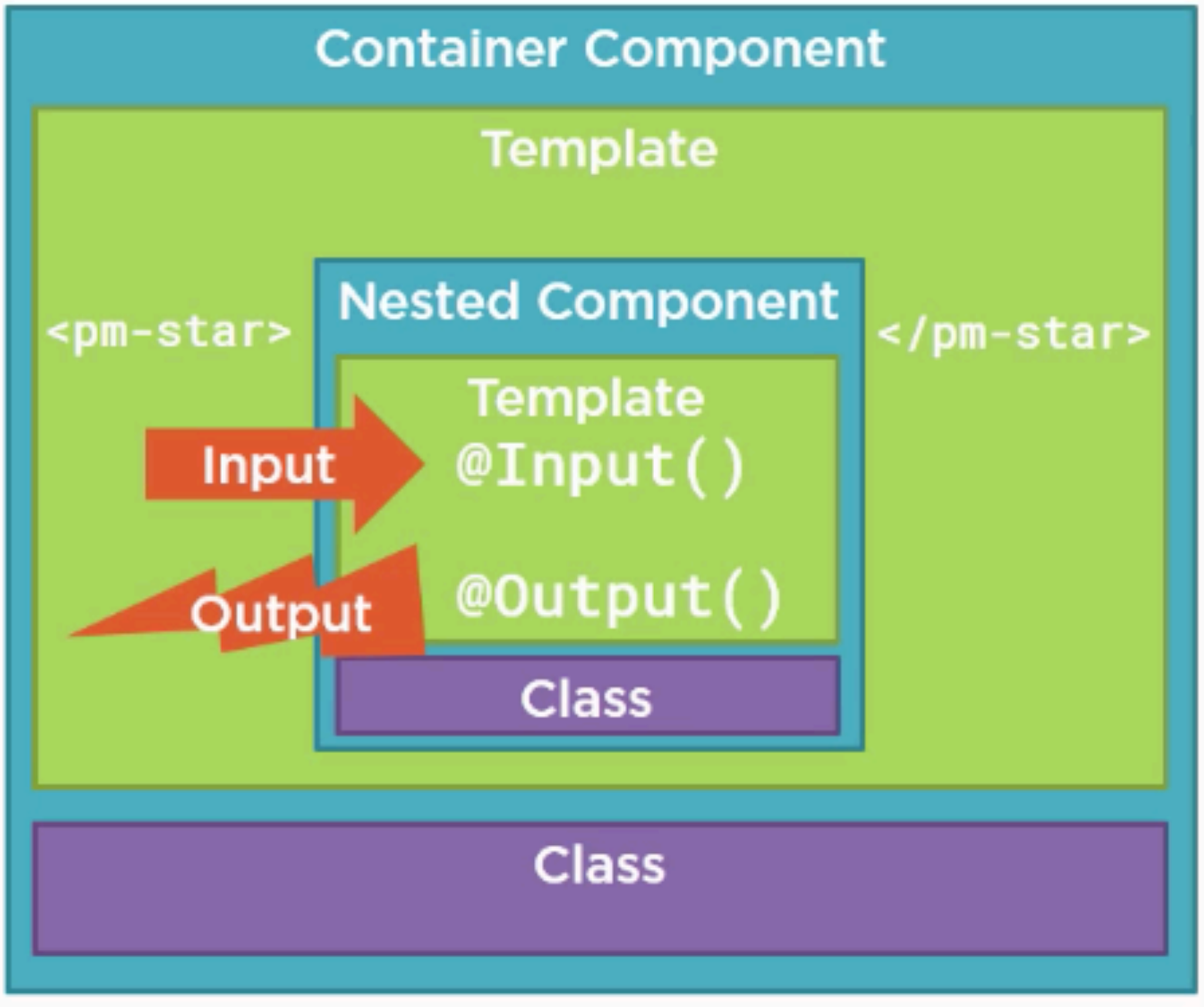
Service

```
export class myService {}
```

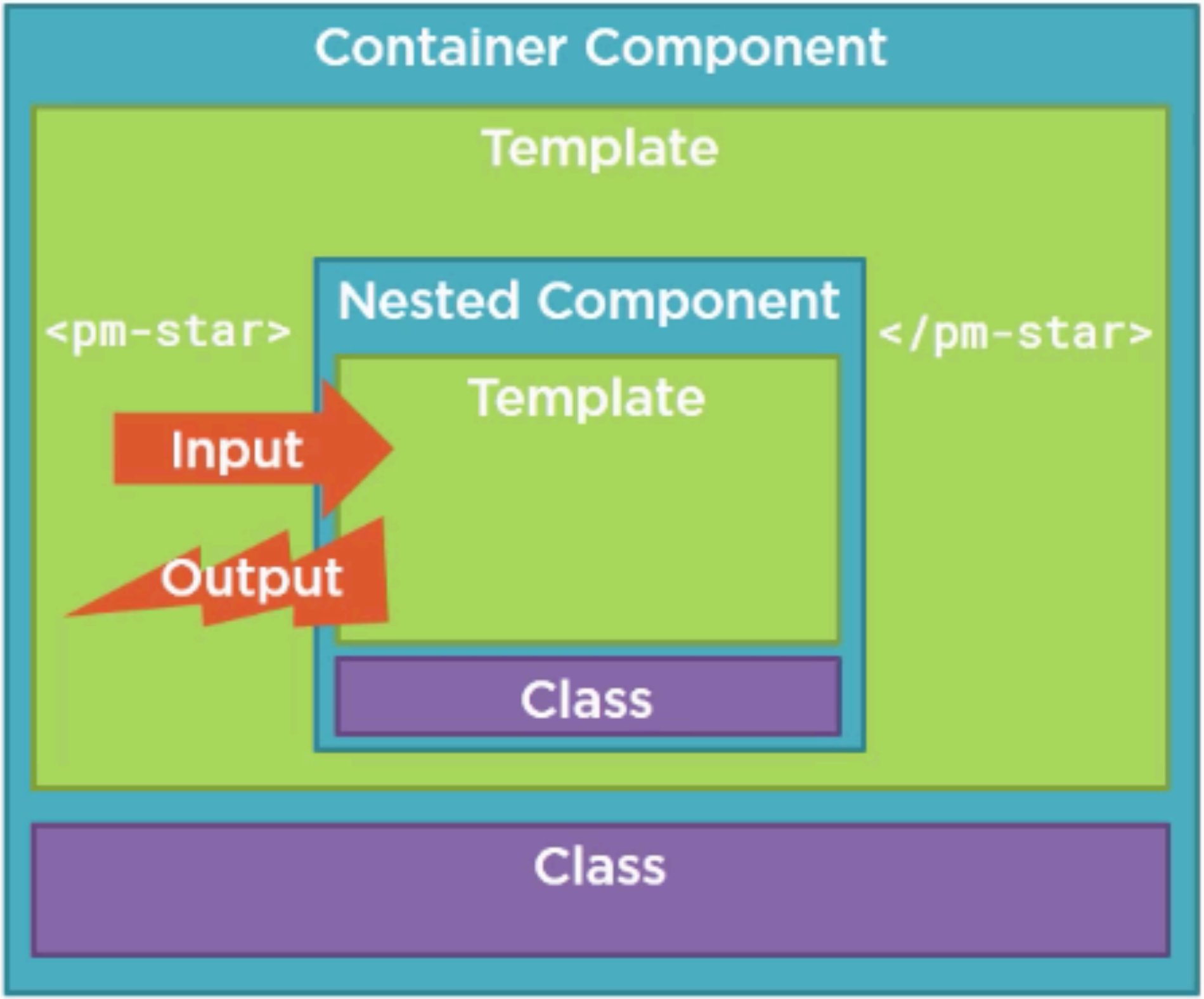
Component

```
constructor(private _myService) {}
```

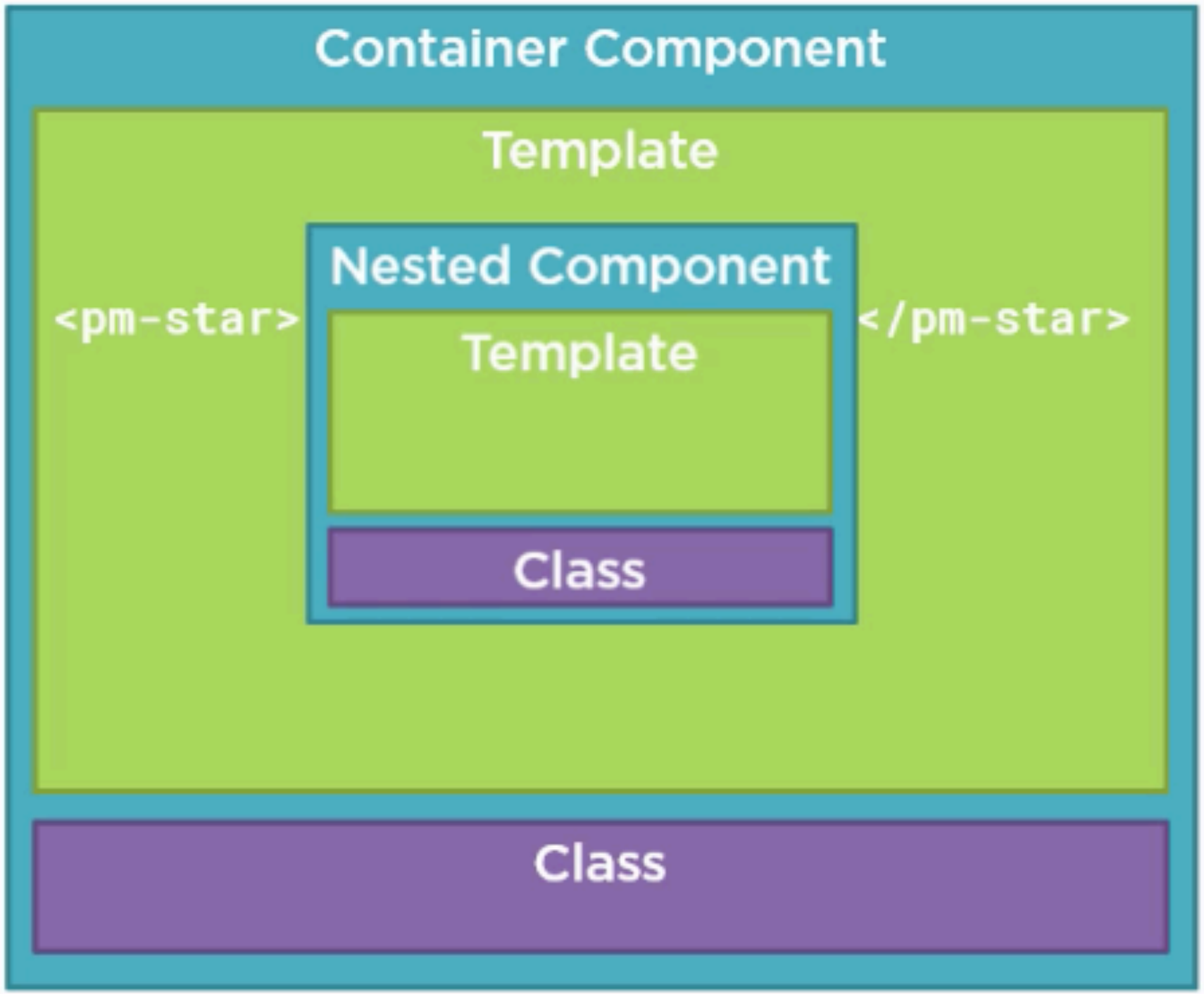
Nest-able Component's Public API



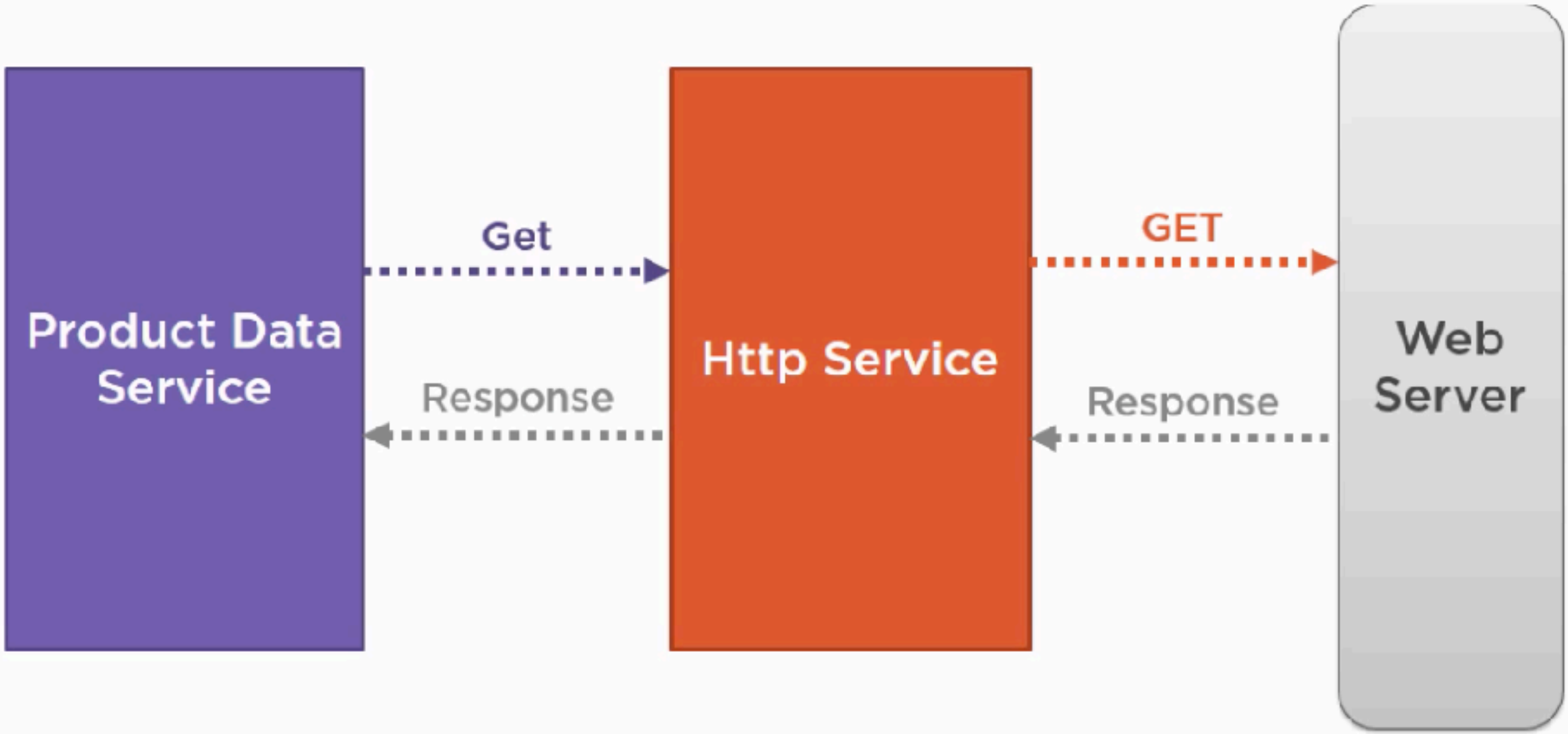
Building a Nested Component



g a Nested Component



Sending an Http Request



Sending an Http Request

product.service.ts

```
...
import { HttpClient } from '@angular/common/http';



@Injectable()
export class ProductService {
  private _productUrl = 'www.myWebService.com/api/products';

  constructor(private _http: HttpClient) { }

  getProducts() {
   return this._http.get(this._productUrl);

  }
}
```

Sending an Http Request

product.service.ts

```
import { HttpClient } from '@angular/common/http';


@Injectable()
export class ProductService {
  private _productUrl = 'www.myWebService.com/api/products';

  constructor(private _http: HttpClient) { }

  getProducts() {
   return this._http.get(this._productUrl);

  }
}
```

Observables

Interactive diagrams of Rx Observables


```
map(x => 10 * x)
```

Subscribing to an Observable

```
x.then(valueFn, errorFn)                          //Promise
x.subscribe(valueFn, errorFn)                     //Observable
x.subscribe(valueFn, errorFn, completeFn)         //Observable
let sub = x.subscribe(valueFn, errorFn, completeFn)
```

product-list.component.ts

```
ngOnInit(): void {
    this._productService.getProducts()
            .subscribe(
              products => this.products = products,
              error => this.errorMessage = <any>error);
}
```