INTRODUCTION TO THE IMPORTANT PRINCIPAL

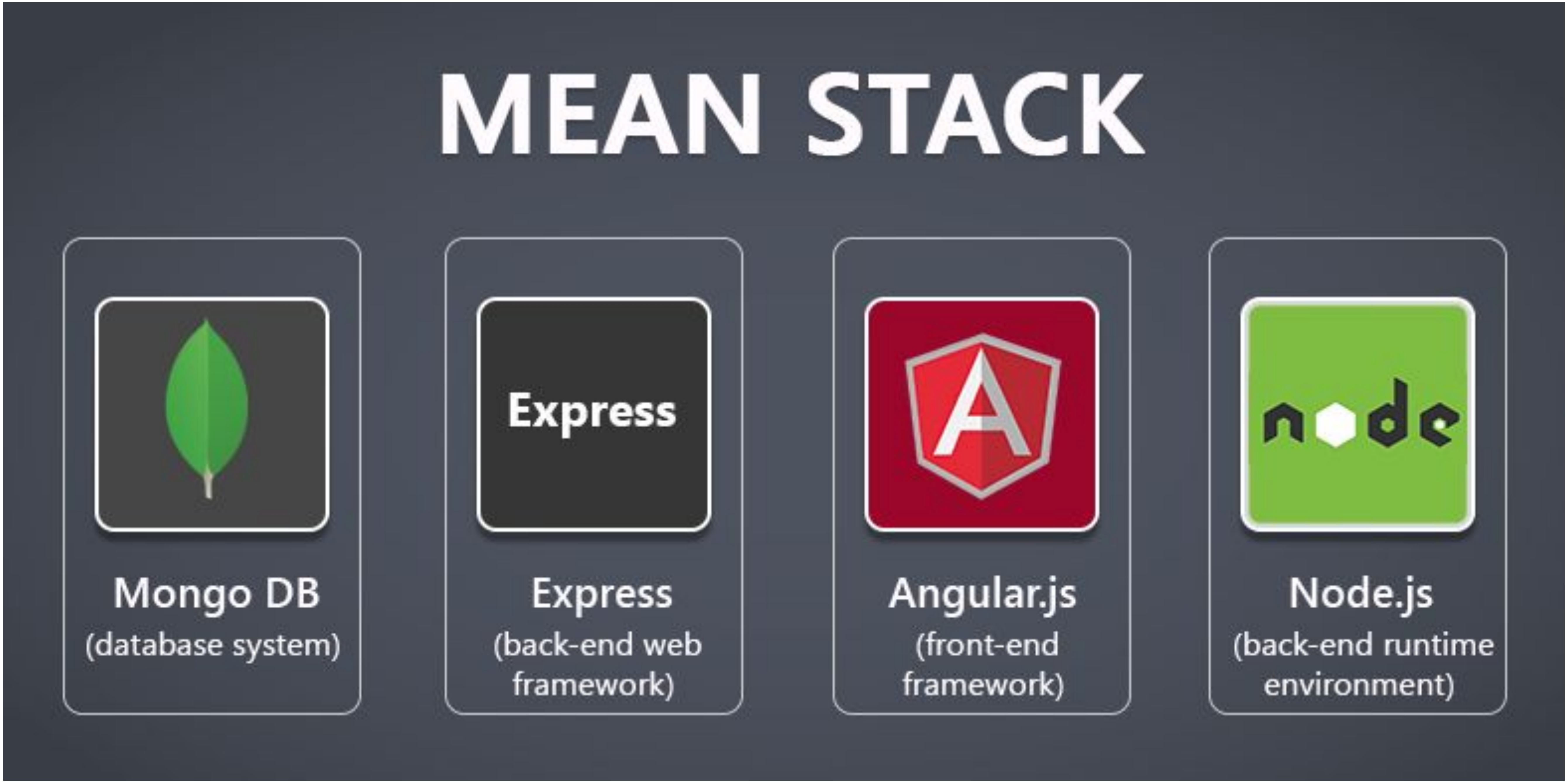
ANGULAR

Let's
Get
Started!

Angular

- Full Framework to build a web application.
- Developed and maintained by Google.
- Based on Typescript.
- Component based development.
- Based for mobile app development, Ionic.
- Part of MEAN Stack

ANGULAR - INTRODUCTION TO ANGULAR PRINCIPAL



MEAN STACK

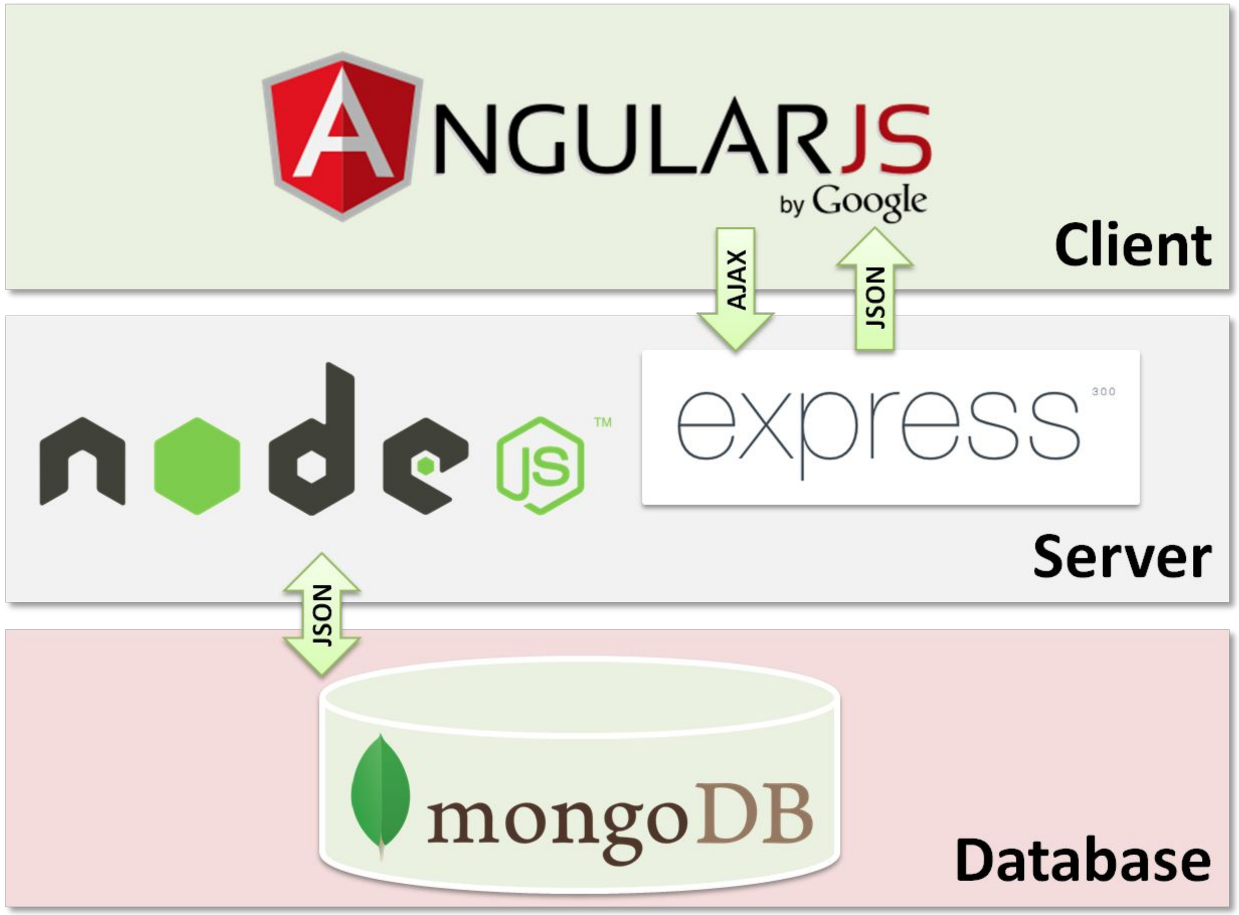
Mongo DB
(database system)

Express
(back-end web framework)

Angular.js
(front-end framework)

Node.js
(back-end runtime environment)

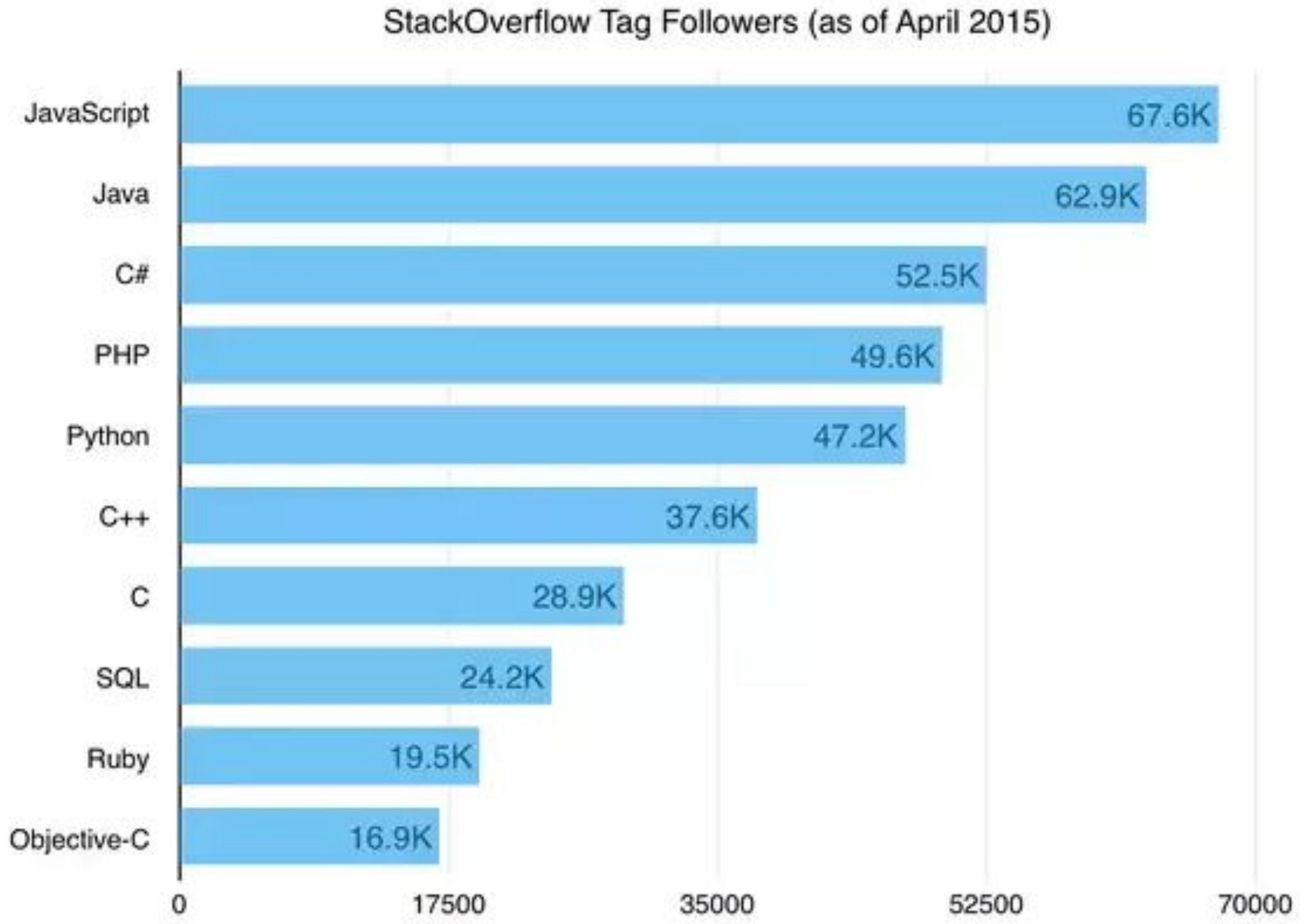
What is MEAN?



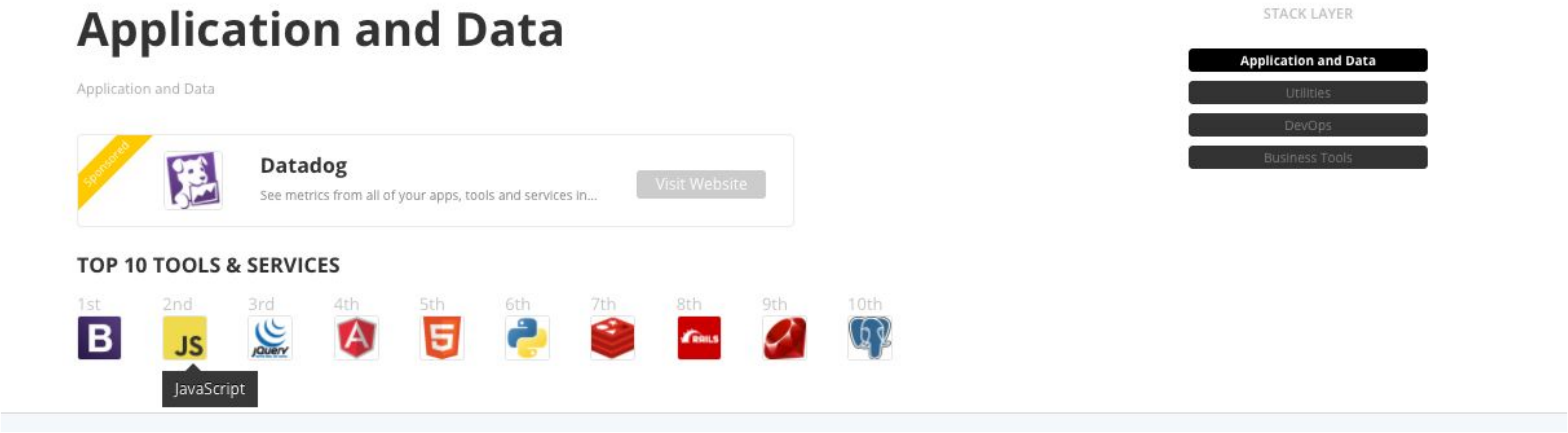
MEAN is an opinionated fullstack javascript framework - which simplifies and accelerates web application development.

Why MEAN?

- 100% Free
- 100% Open Source
- 100% (Javascript and HTML)
- Single language throughout the application.
- Adhere to MVC concept.
- Use of JSON as data structure, compared to before where serialization and deserialization of data structure is needed.



StackOverflow Tag Followers (as of April 2015)



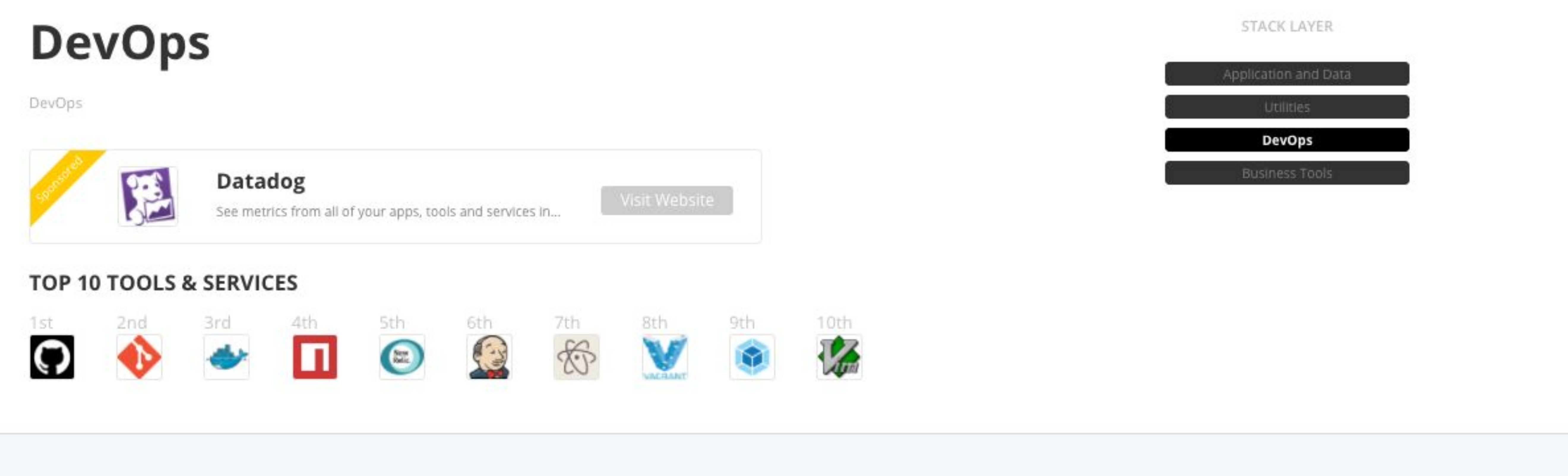
Application and Data

Application and Data

Datadog

See metrics from all of your apps, tools and services in...

TOP 10 TOOLS & SERVICES



DevOps

DevOps

TOP 10 TOOLS & SERVICES

1st 2nd 3rd 4th 5th 6th 7th 8th 9th 10th

LAMP Stack

LAMP
Linux
Apache
MySQL
PHP, Pearl, Python

ANGULAR KEY FEATURES

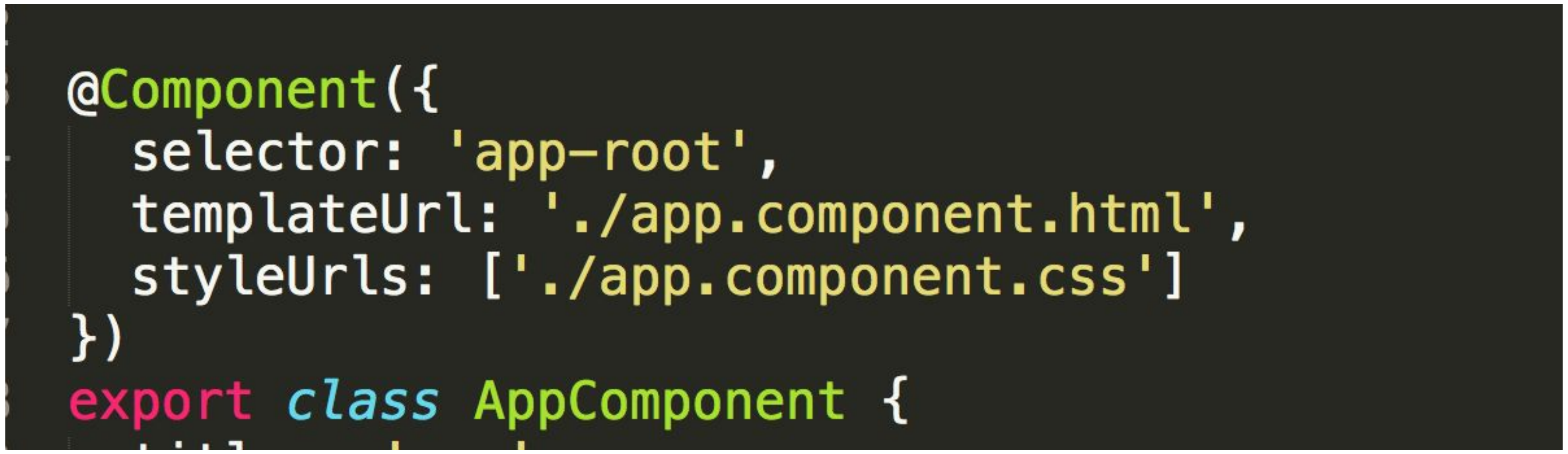
- Full Framework
  - View
  - Model
  - Services
  - Routing
  - Http Requests

- Easy to use
- Scalable
- Fast
- Animations

INSTALLATION AND GET STARTED

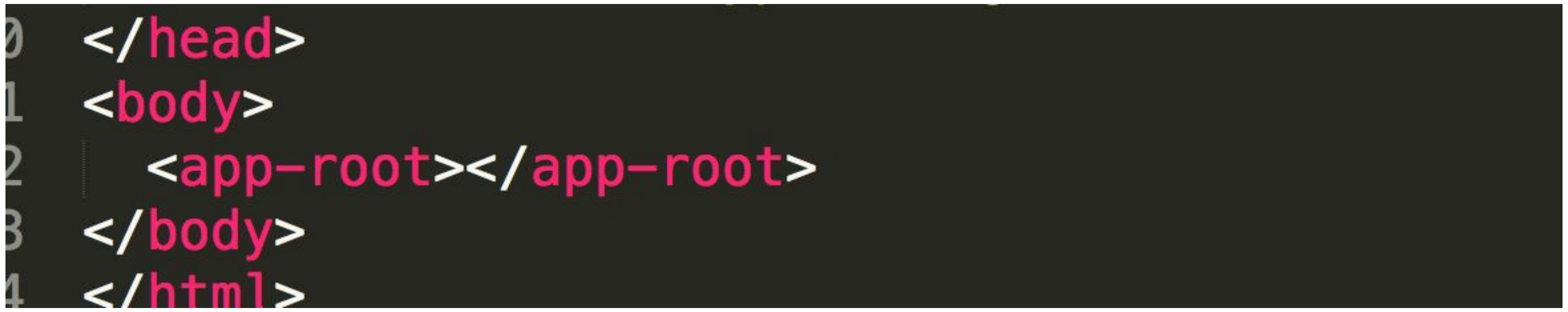
- Installing Angular
  - npm install -g @angular/cli
- Creating new project
  - ng new angular-tutorial
- Open in browser
  - ng serve - - open

COMPONENT METADATA PROPERTIES



```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
```

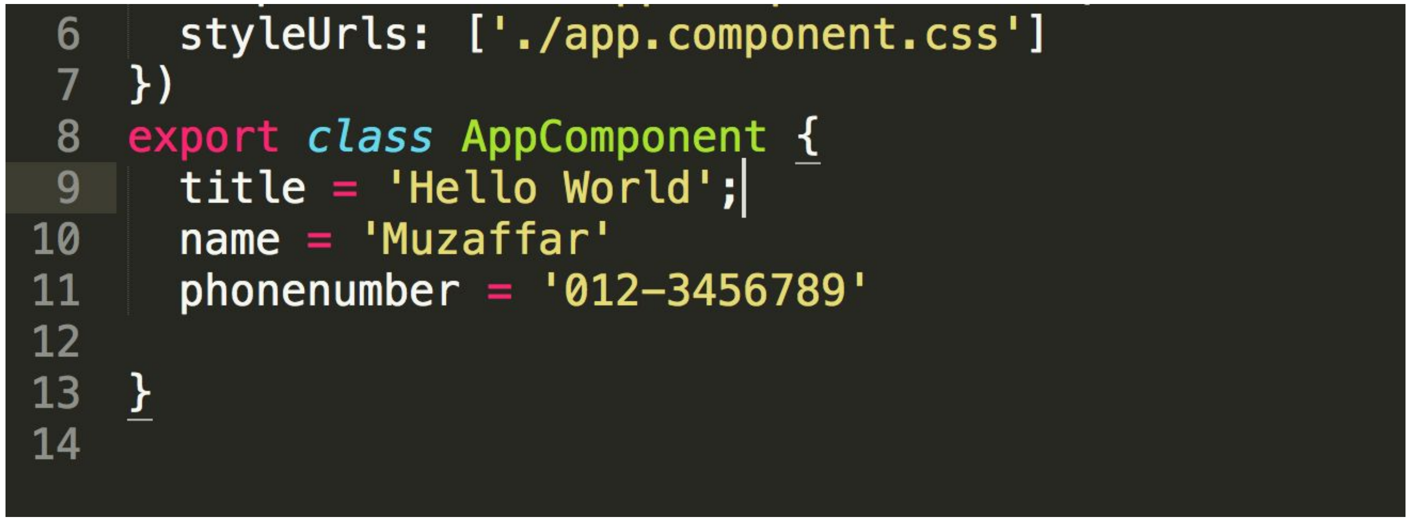
app.component.ts



```
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

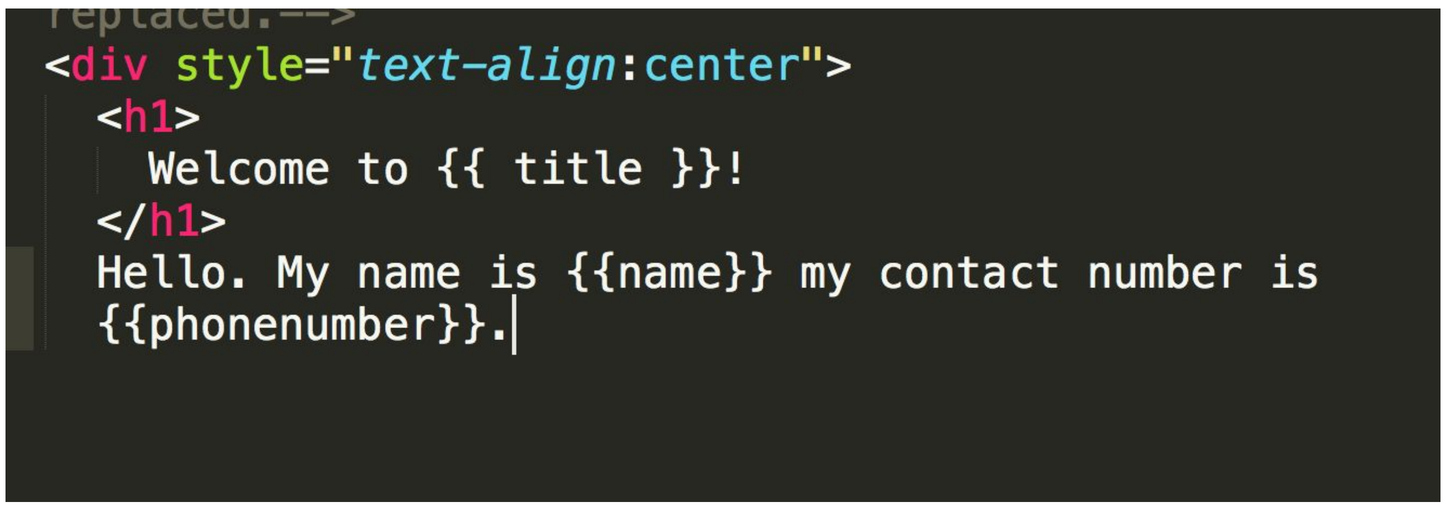
index.html

INTERPOLATION BINDING



```
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'Hello World';
  name = 'Muzaffar'
  phonenumber = '012-3456789'

}

```

app.component.ts



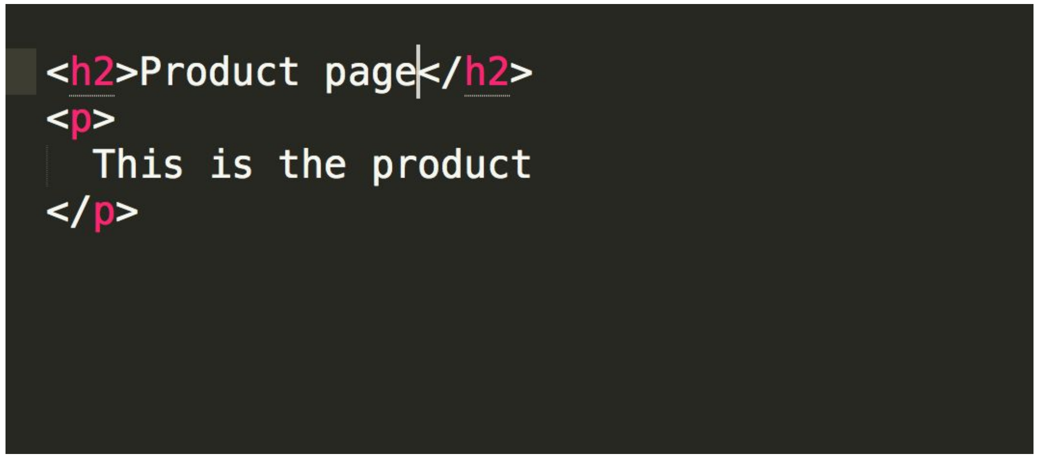
```
replaced.-->
<div style="text-align:center">
  <h1>
    Welcome to {{ title }}!
  </h1>
  Hello. My name is {{name}} my contact number is
  {{phonenumber}}.
```

app.component.html

GENERATING NEW COMPONENT

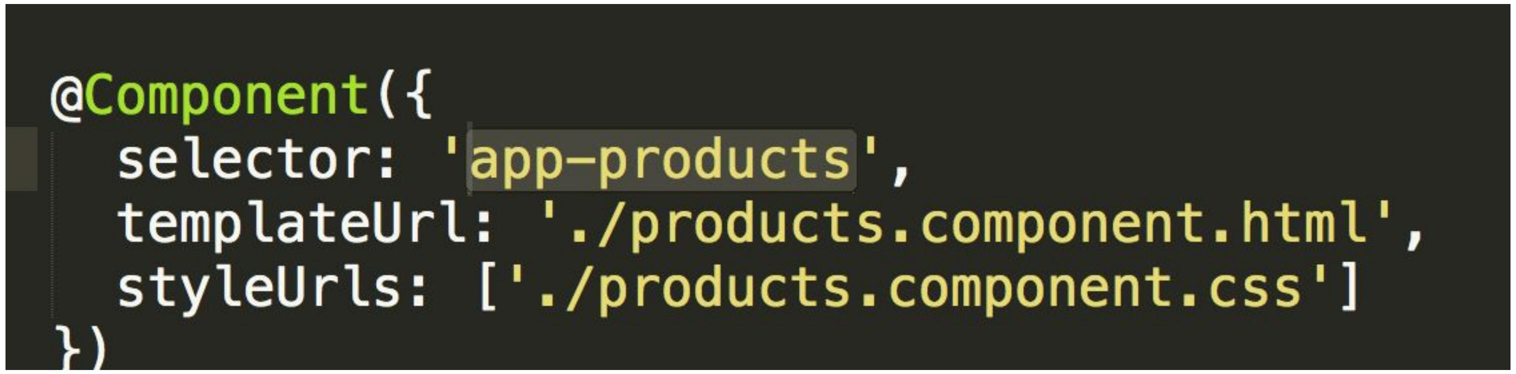
- Generate new component
  - ng g component products
- Call the component from new component to app page using generated selector.

PRODUCT PAGE



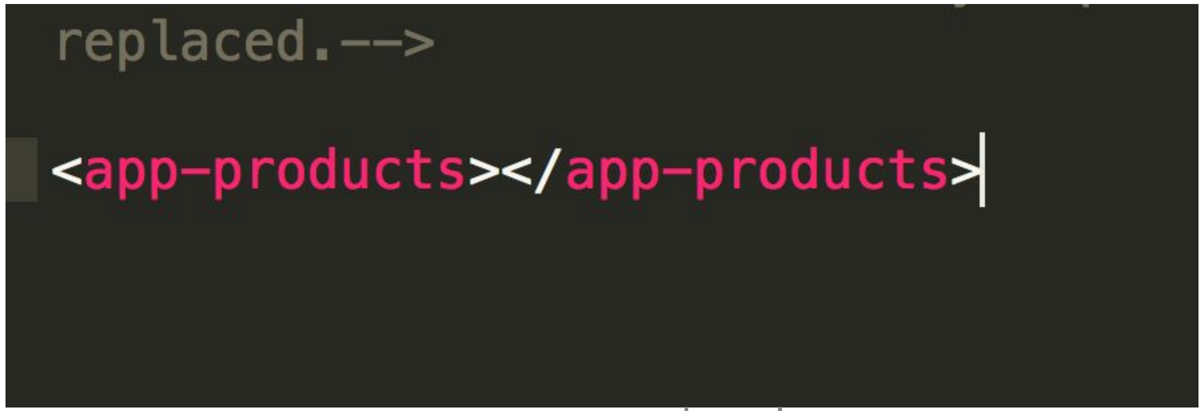
```
<h2>Product page</h2>
<p>
  This is the product
</p>
```

products.component.html



```
@Component({
  selector: 'app-products',
  templateUrl: './products.component.html',
  styleUrls: ['./products.component.css']
})
```

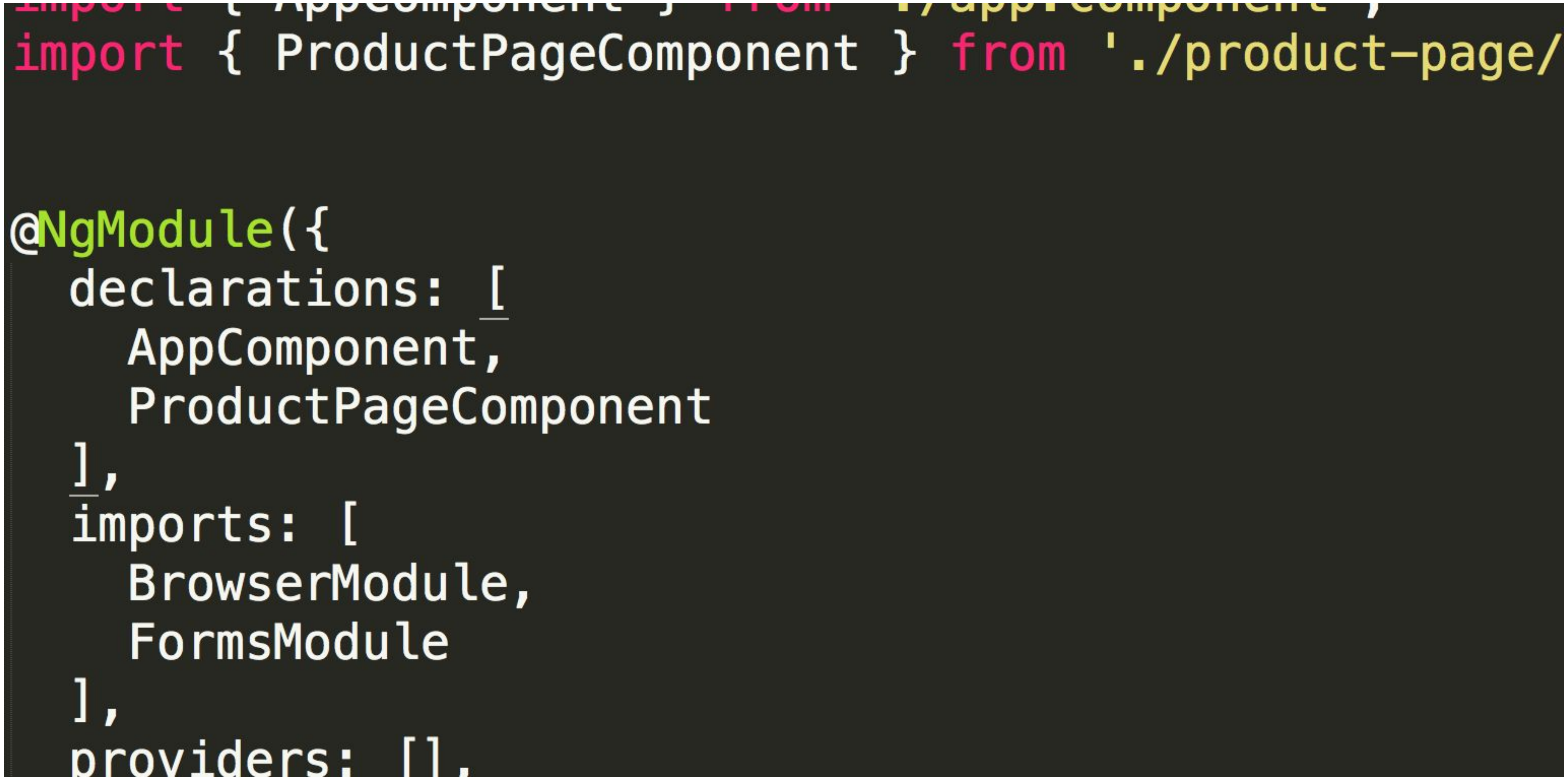
products.component.ts



```
replaced.-->

<app-products></app-products>
```

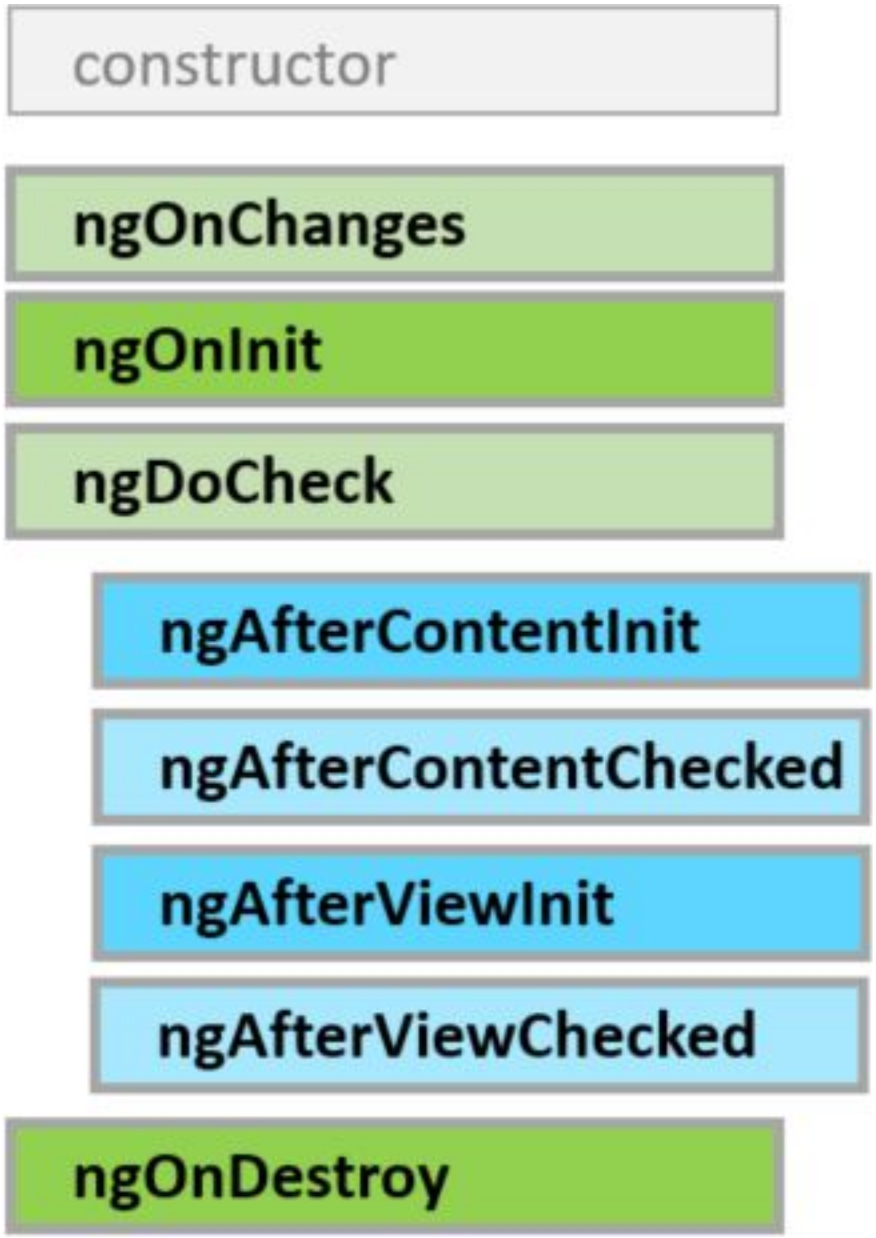
app.component.html

COMPONENT DECLARATION IN APP.MODULE



```
import { AppComponent } from './app.component';
import { ProductPageComponent } from './product-page/


@NgModule({
  declarations: [
    AppComponent,
    ProductPageComponent
  ],
  imports: [
    BrowserModule,
    FormsModule
  ],
  providers: [],
```

ANGULAR LIFECYCLE HOOKS



constructor

ngOnChanges

ngOnInit

ngDoCheck

ngAfterContentInit

ngAfterContentChecked

ngAfterViewInit

ngAfterViewChecked

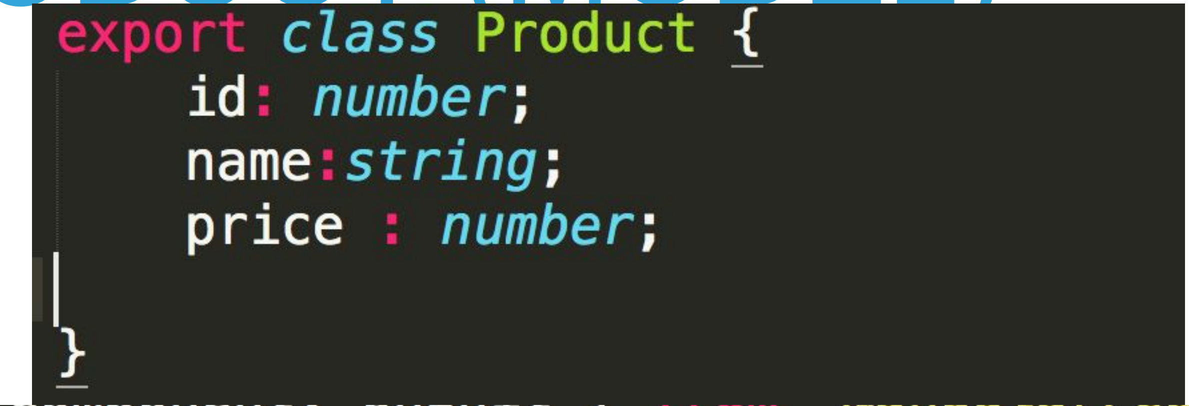
ngOnDestroy

Add a console.log in ngOnInit function

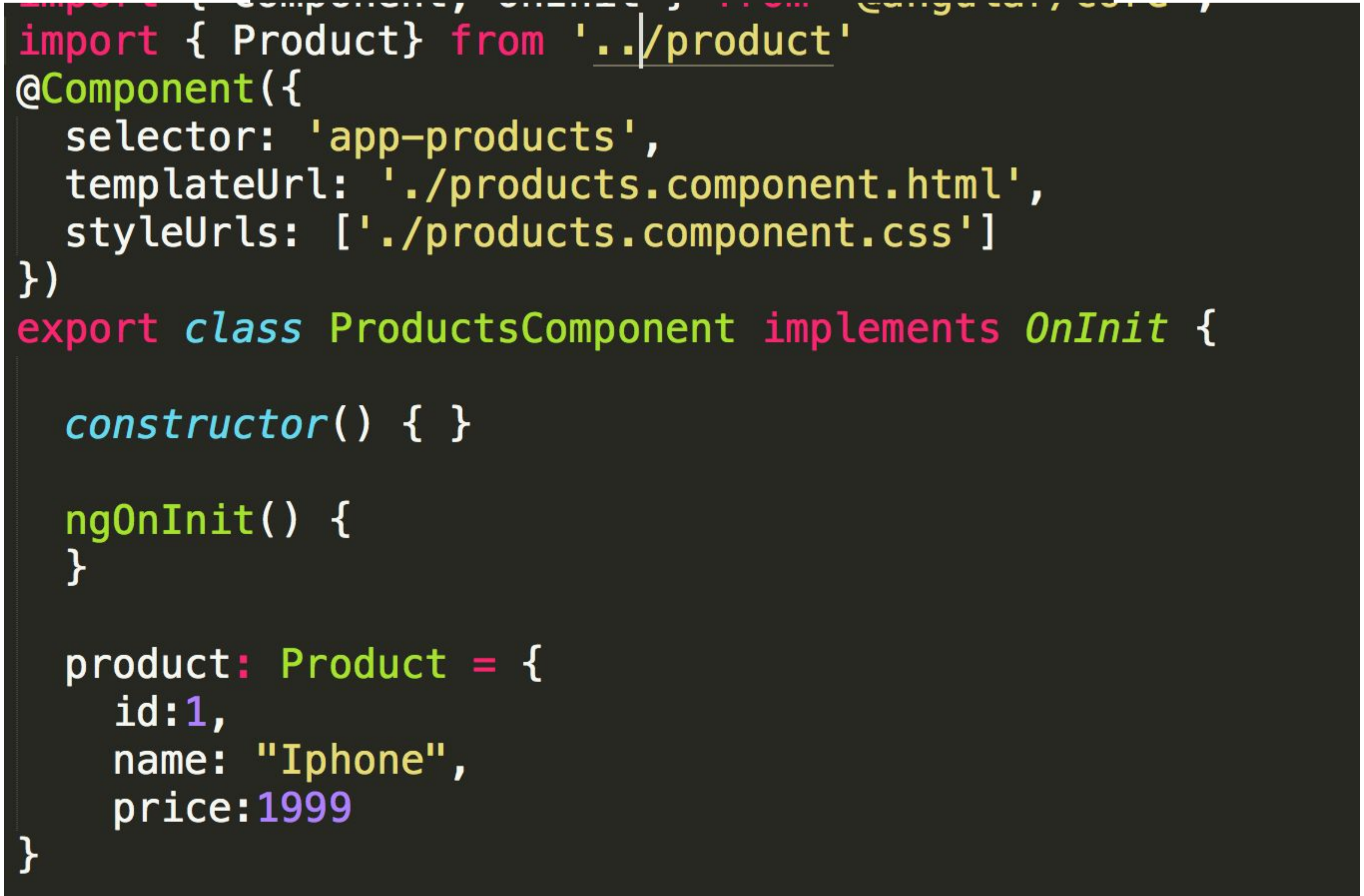
CREATING PRODUCT CLASS

- Add new class
  - ng g class product
- Export the class.

CLASS PRODUCT (MODEL)

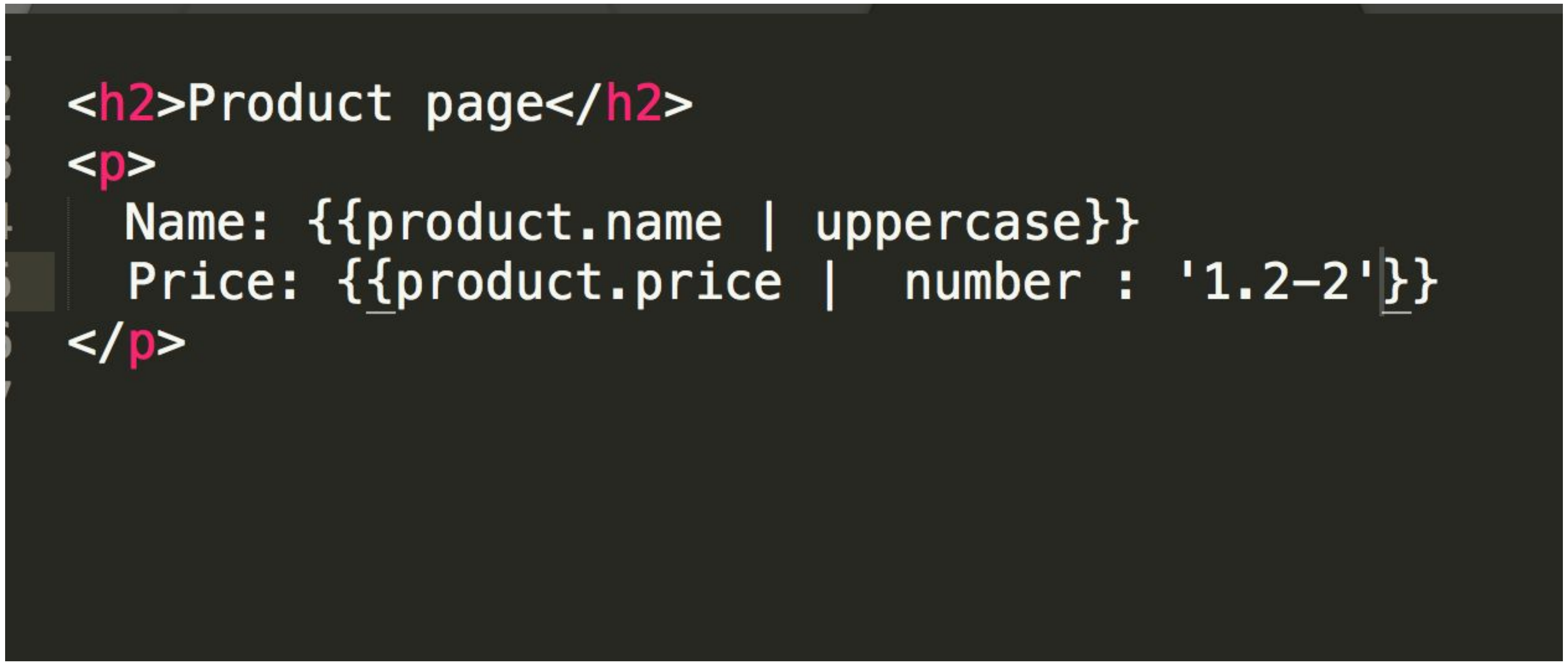


```
export class Product {
    id: number;
    name:string;
    price : number;

}
```



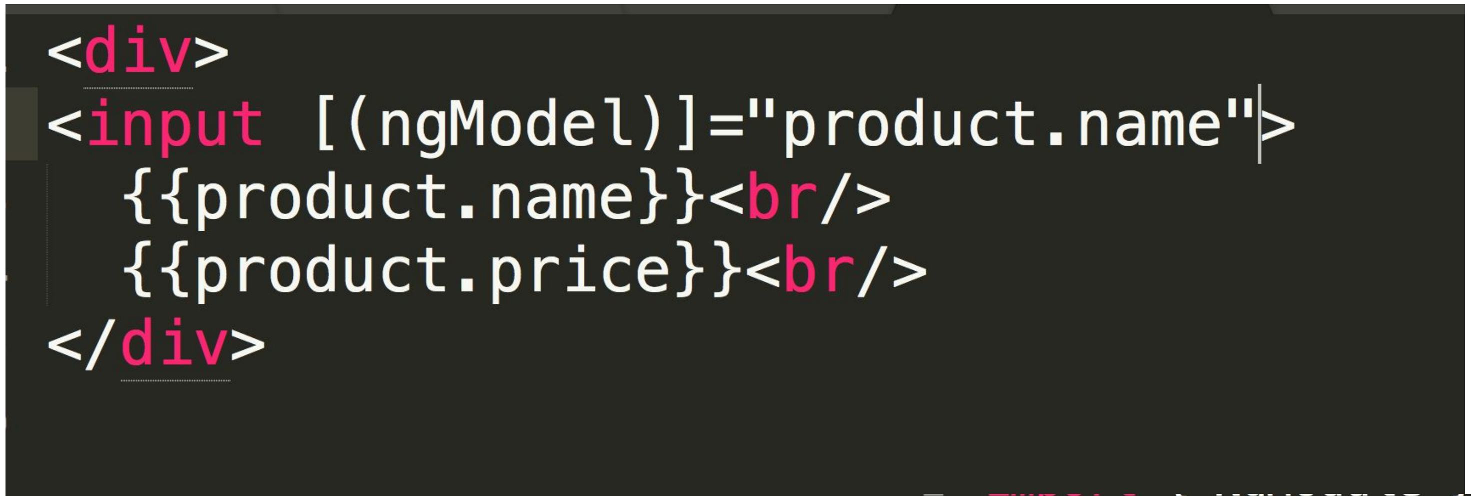
```
import { Component, OnInit } from '@angular/core';
import { Product} from '../product'
@Component({
  selector: 'app-products',
  templateUrl: './products.component.html',
  styleUrls: ['./products.component.css']
})
export class ProductsComponent implements OnInit {

  constructor() { }

  ngOnInit() {
  }

  product: Product = {
    id:1,
    name: "Iphone",
    price:1999
}
```

FORMATTING WITH PIPES

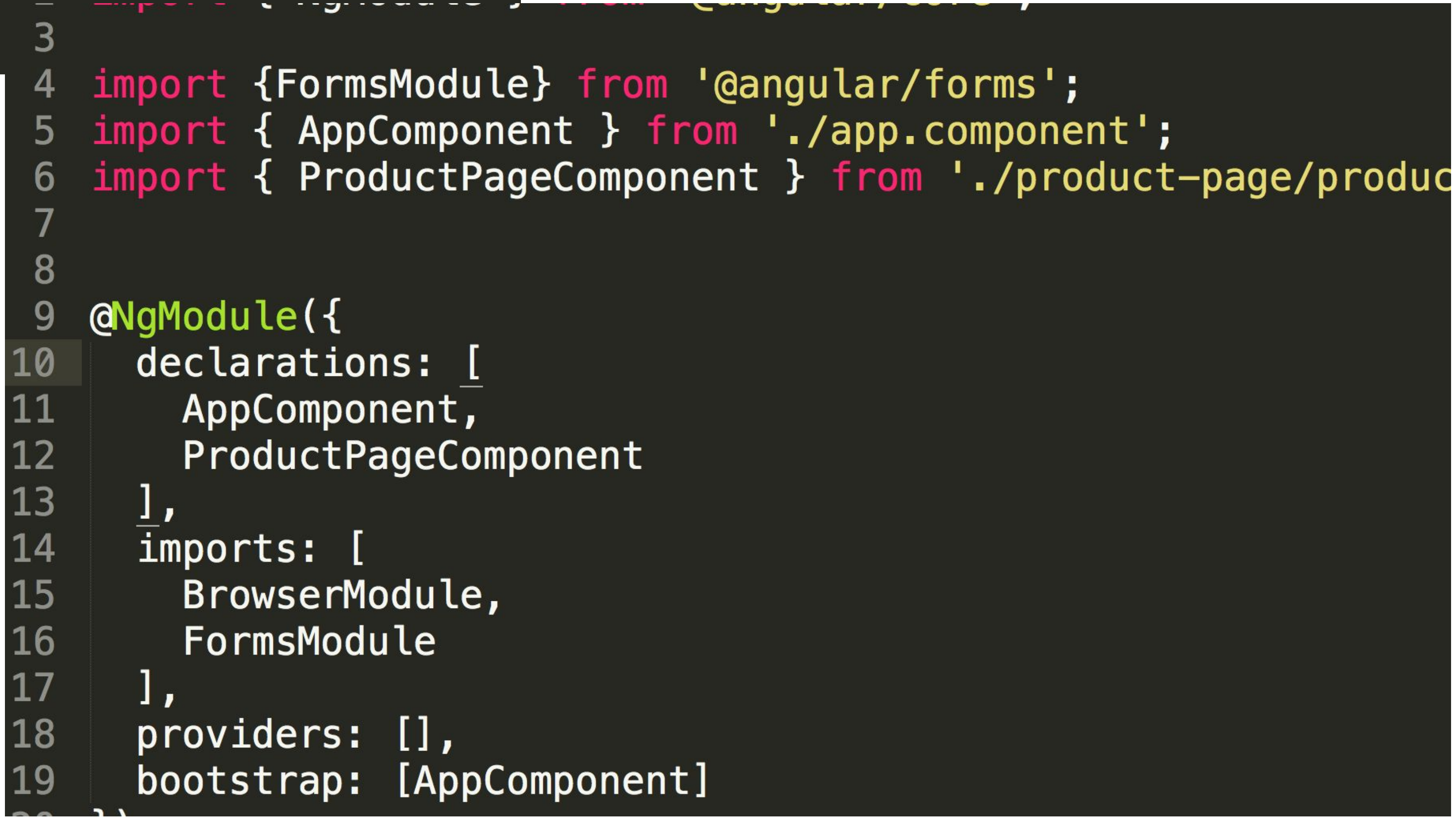


```
<h2>Product page</h2>
<p>
  Name: {{product.name | uppercase}}
  Price: {{product.price |  number : '1.2-2'}}
</p>
```

TWO WAY DATA BINDING - FORM MODULE

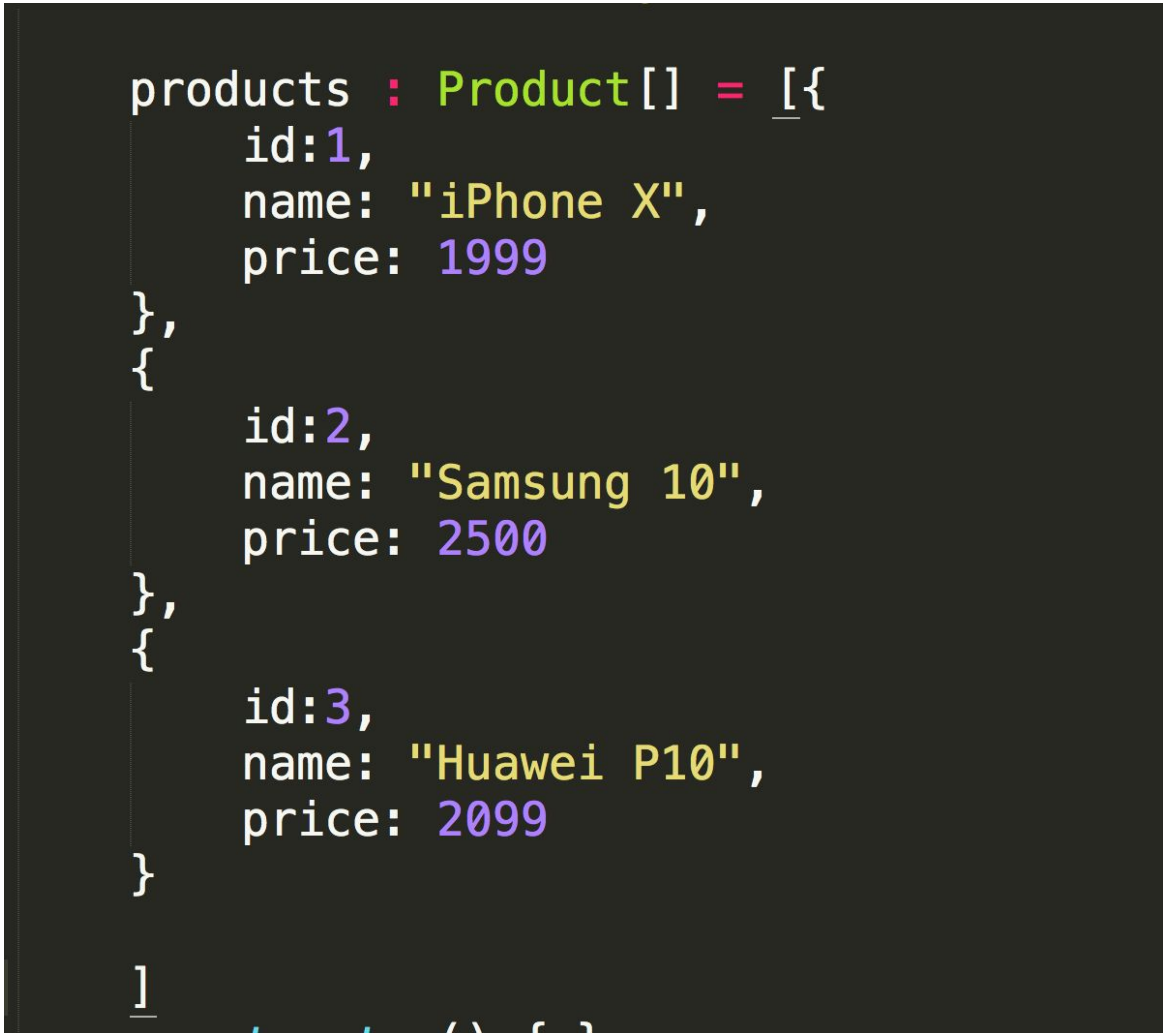


```html
<div>
<input [(ngModel)]="product.name">
  {{product.name}}<br/>
  {{product.price}}<br/>
</div>
```



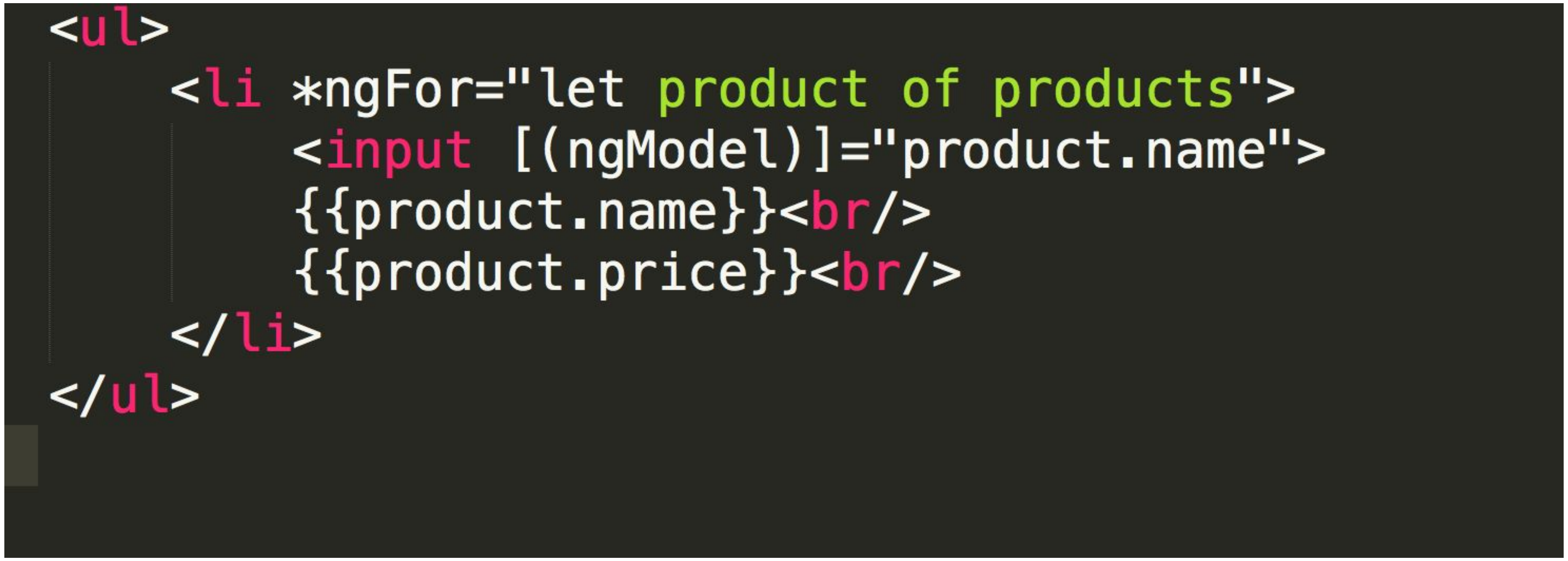
```typescript
import { NgModule } from '@angular/core';

import {FormsModule} from '@angular/forms';
import { AppComponent } from './app.component';
import { ProductPageComponent } from './product-page/produc


@NgModule({
  declarations: [
    AppComponent,
    ProductPageComponent
  ],
  imports: [
    BrowserModule,
    FormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
```

LIST - CREATING ARRAY OF PRODUCTS



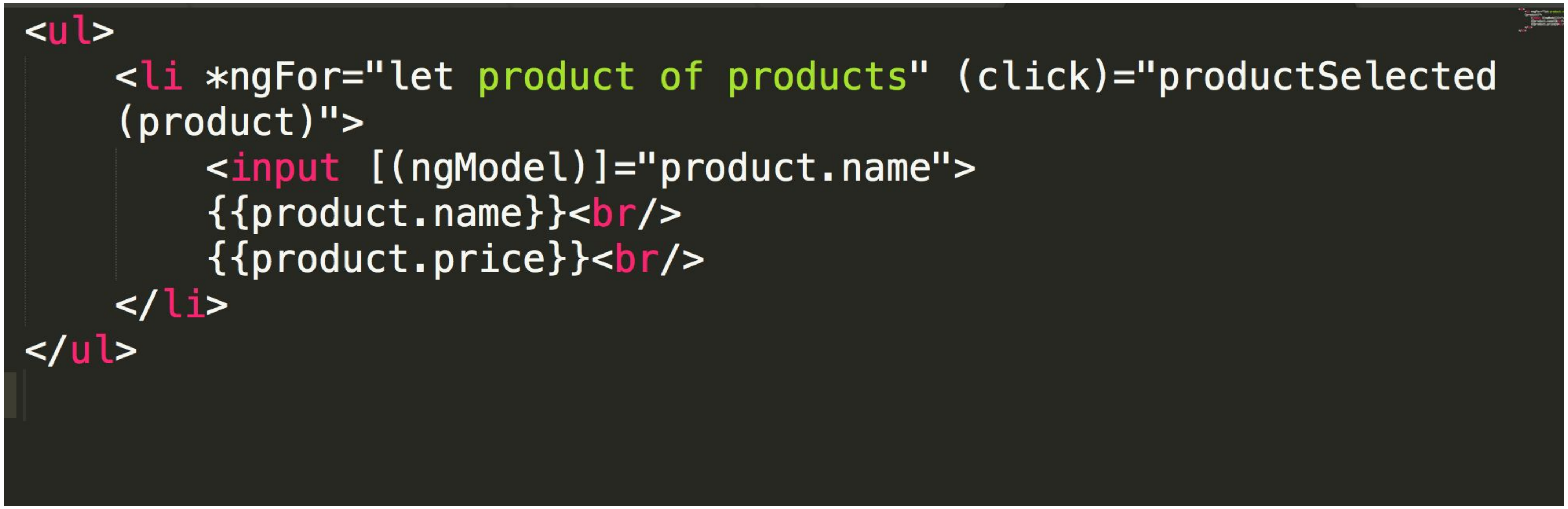
```
products : Product[] = [{
      id:1,
      name: "iPhone X",
      price: 1999
},
{
      id:2,
      name: "Samsung 10",
      price: 2500
},
{
      id:3,
      name: "Huawei P10",
      price: 2099
}

]
```

LIST - RETRIEVING LIST USING *NGFOR

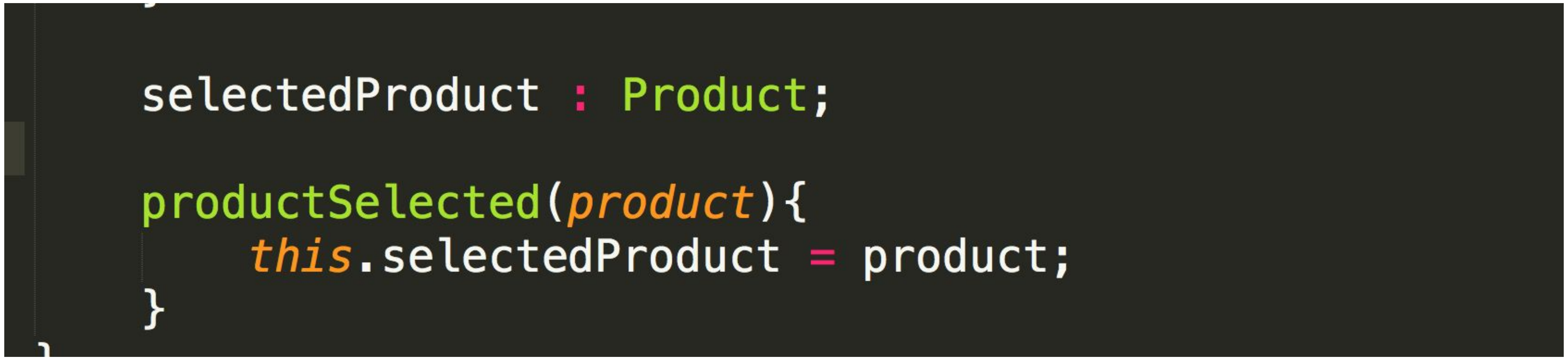


```
<ul>
    <li *ngFor="let product of products">
        <input [(ngModel)]="product.name">
        {{product.name}}<br/>
        {{product.price}}<br/>
    </li>
</ul>
```

EVENT HANDLER



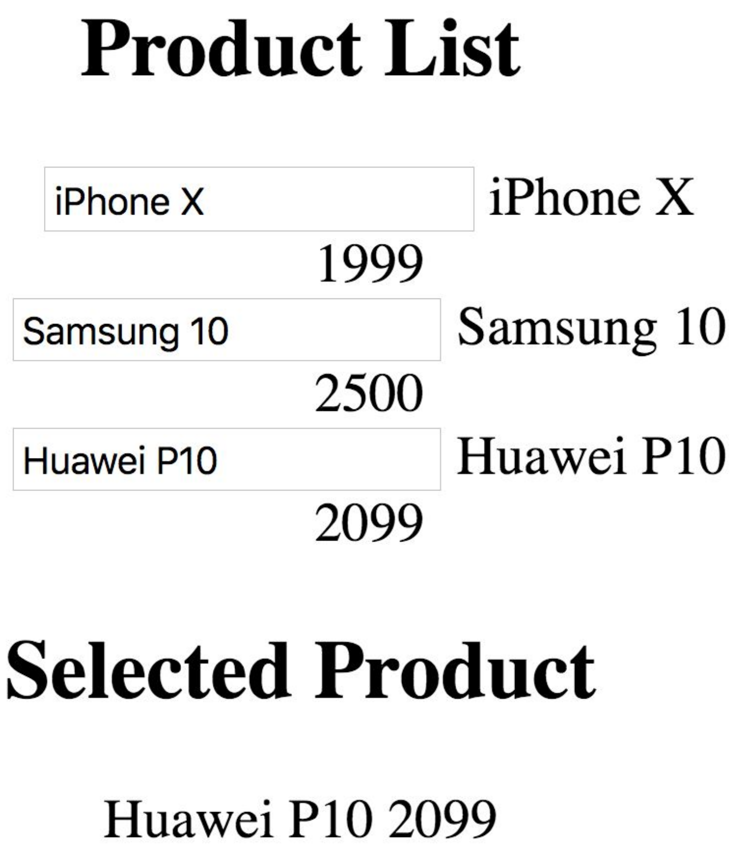
```
<ul>
    <li *ngFor="let product of products" (click)="productSelected
    (product)">
        <input [(ngModel)]="product.name">
        {{product.name}}<br/>
        {{product.price}}<br/>
    </li>
</ul>
```



```
    selectedProduct : Product;

    productSelected(product){
        this.selectedProduct = product;
    }
```

EXERCISE

- Show the selected product in the HTML.



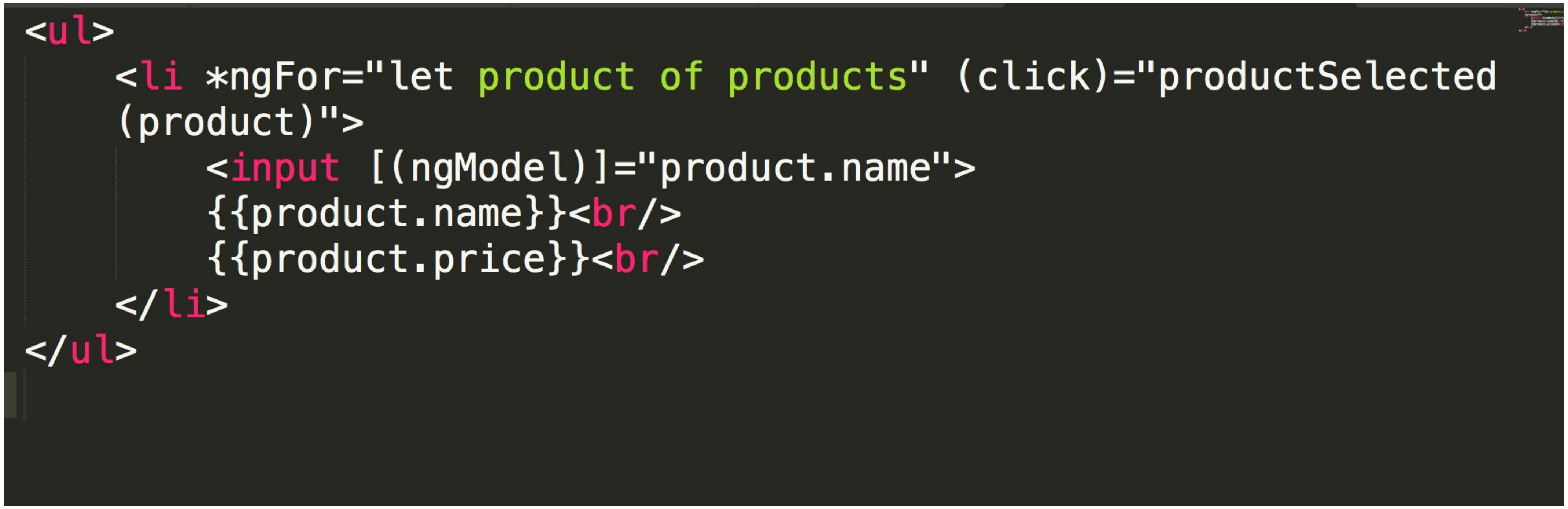
Product List

iPhone X iPhone X
1999
Samsung 10 Samsung 10
2500
Huawei P10 Huawei P10
2099

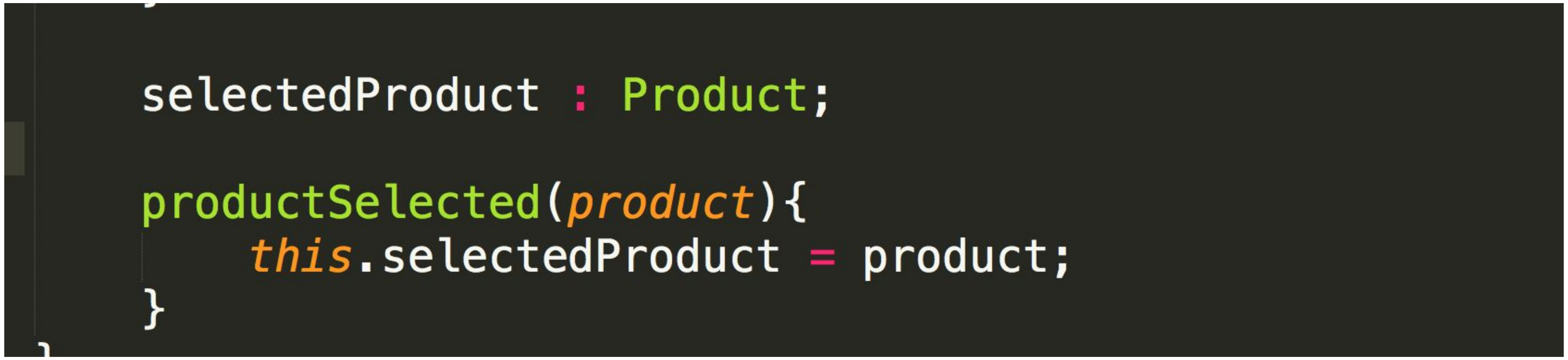
Selected Product

Huawei P10 2099

EVENT HANDLER



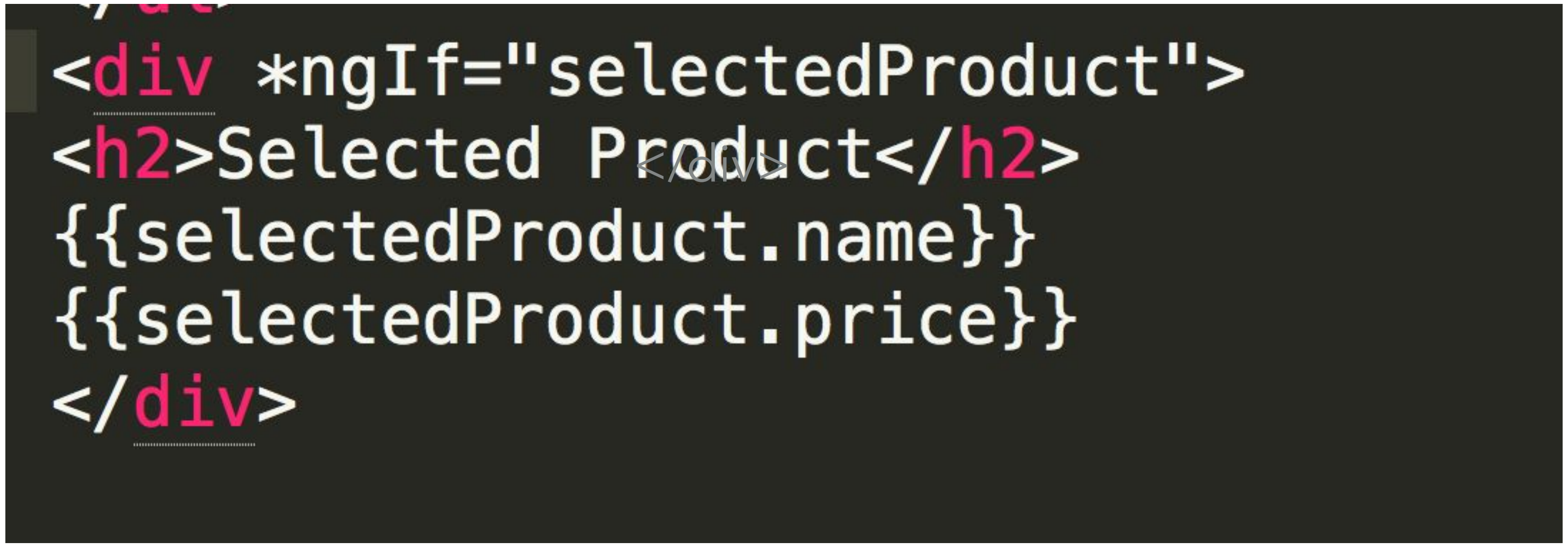
```
<ul>
    <li *ngFor="let product of products" (click)="productSelected
    (product)">
        <input [(ngModel)]="product.name">
        {{product.name}}<br/>
        {{product.price}}<br/>
    </li>
</ul>
```



```
    selectedProduct : Product;

    productSelected(product){
        this.selectedProduct = product;
    }
```

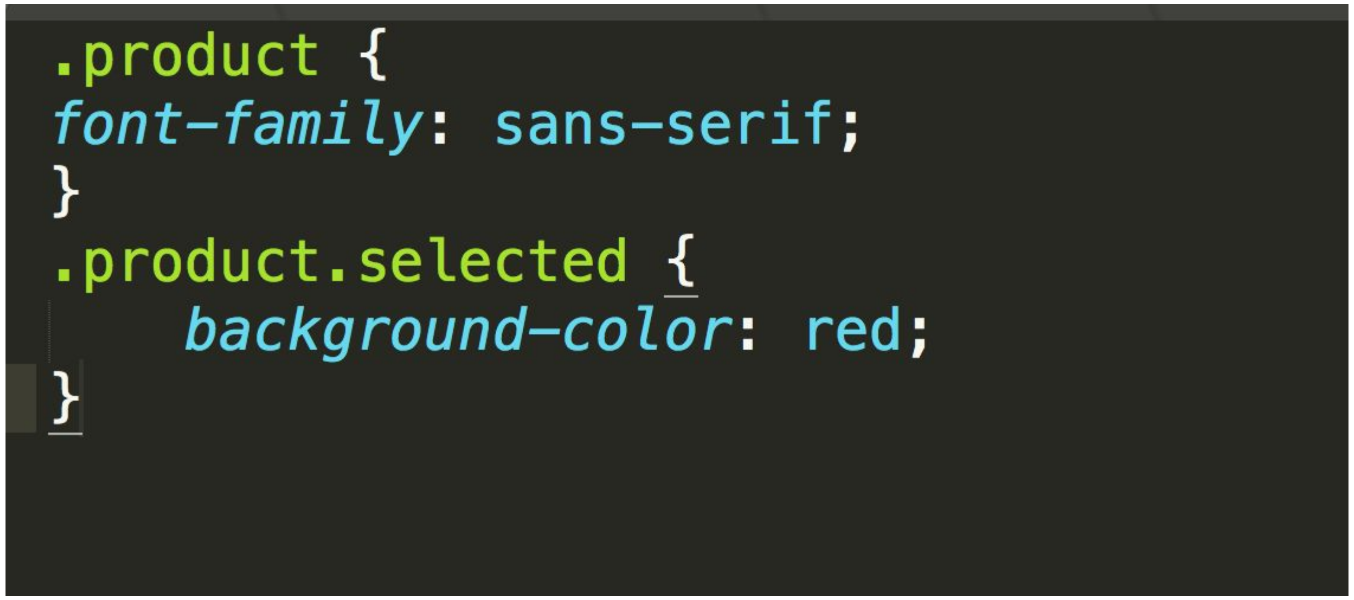
CONDITIONAL RENDERING - *NGIF

- Show the selected product in the HTML.

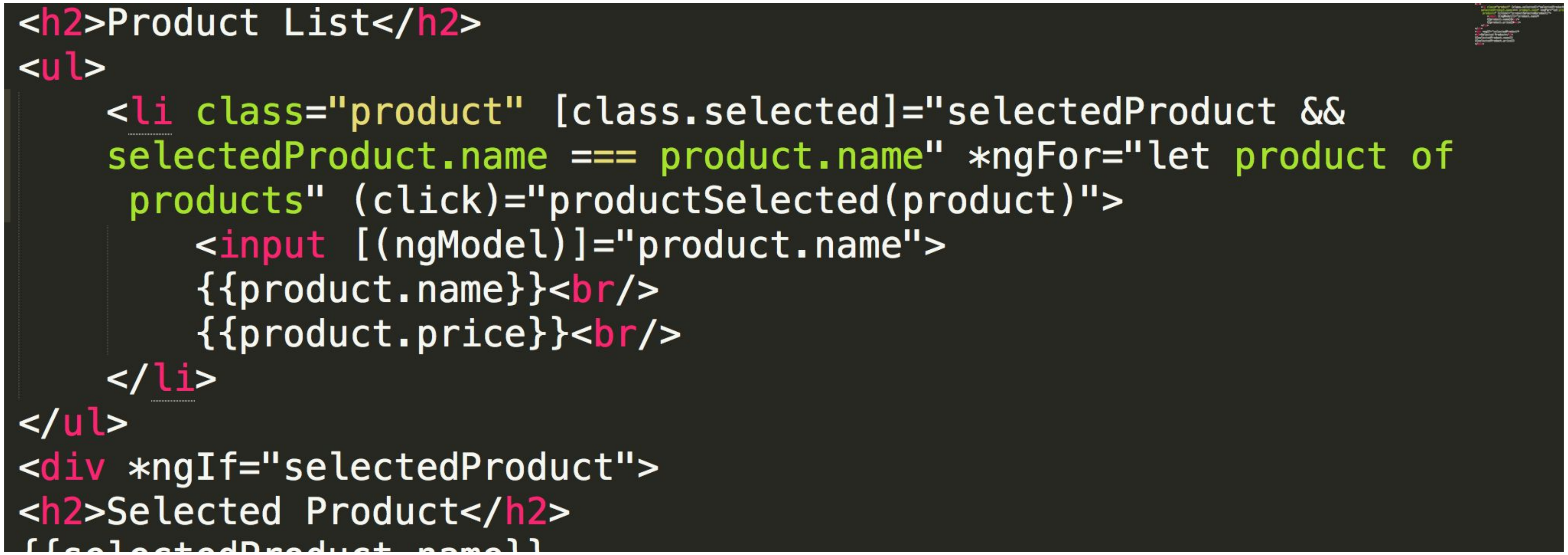


```
<div *ngIf="selectedProduct">
<h2>Selected Product</h2>
{{selectedProduct.name}}
{{selectedProduct.price}}
</div>
```

DYNAMIC CLASS



```
.product {
font-family: sans-serif;
}
.product.selected {
    background-color: red;
}
```

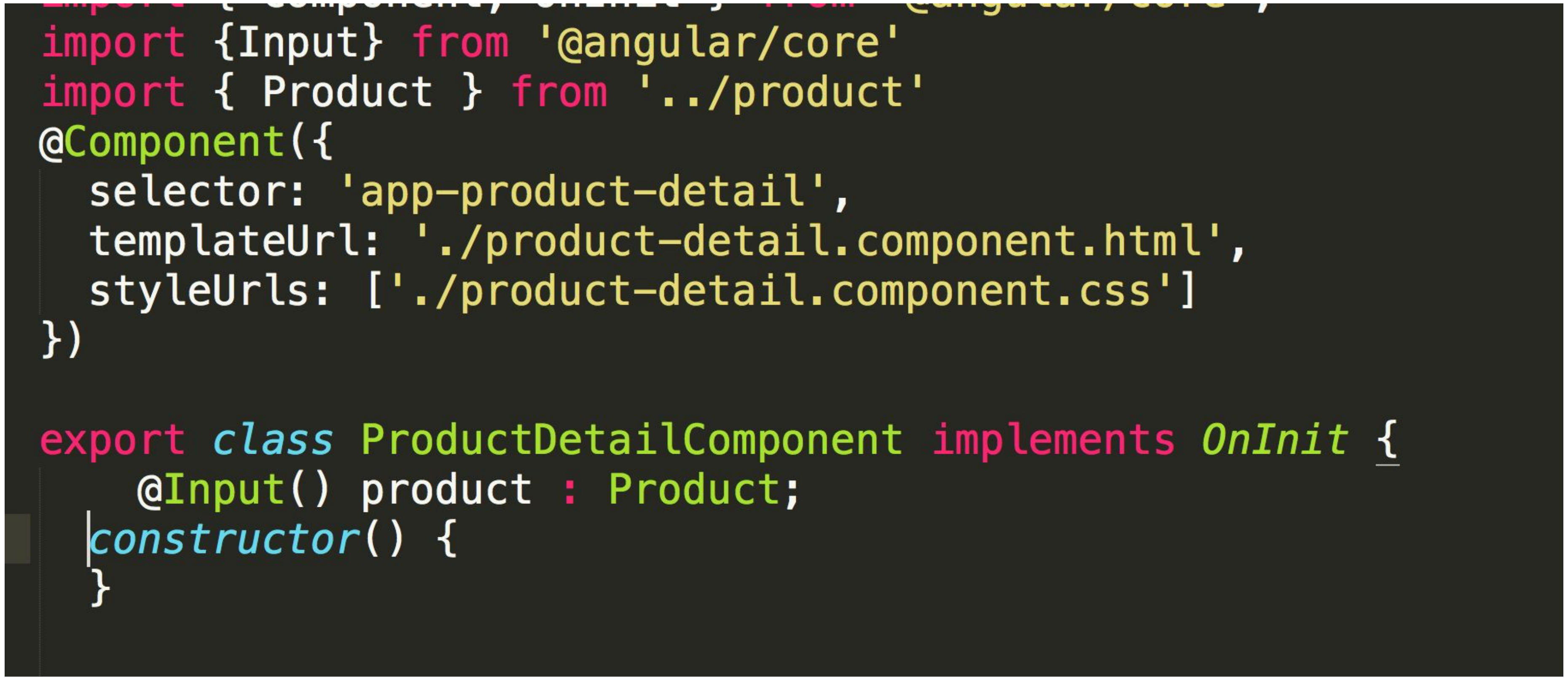


```
<h2>Product List</h2>
<ul>
    <li class="product" [class.selected]="selectedProduct &&
    selectedProduct.name === product.name" *ngFor="let product of
     products" (click)="productSelected(product)">
        <input [(ngModel)]="product.name">
        {{product.name}}<br/>
        {{product.price}}<br/>
    </li>
</ul>
<div *ngIf="selectedProduct">
<h2>Selected Product</h2>
{{selectedProduct.name}}
```

EXERCISE

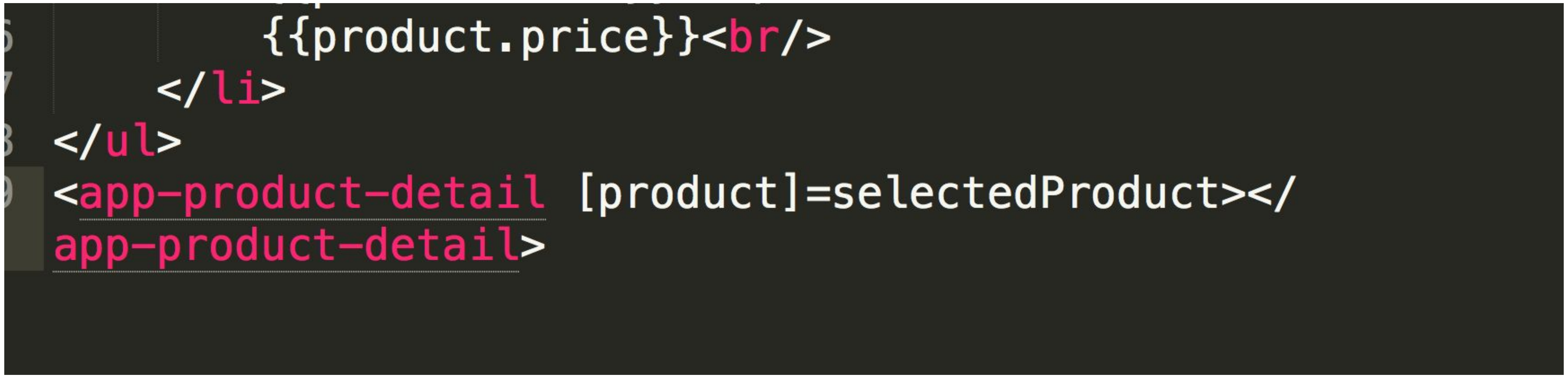
- Create a new component, call it product-detail.
- Move the “Product detail part” from the list page into the newly created component.
- Call the product-detail component from product-list.

PASSING PROPERTIES WITH @INPUT



```
import { Component, OnInit } from '@angular/core';
import {Input} from '@angular/core'
import { Product } from '../product'
@Component({
  selector: 'app-product-detail',
  templateUrl: './product-detail.component.html',
  styleUrls: ['./product-detail.component.css']
})

export class ProductDetailComponent implements OnInit {
    @Input() product : Product;
  constructor() {
  }
```

product-detail.component.ts

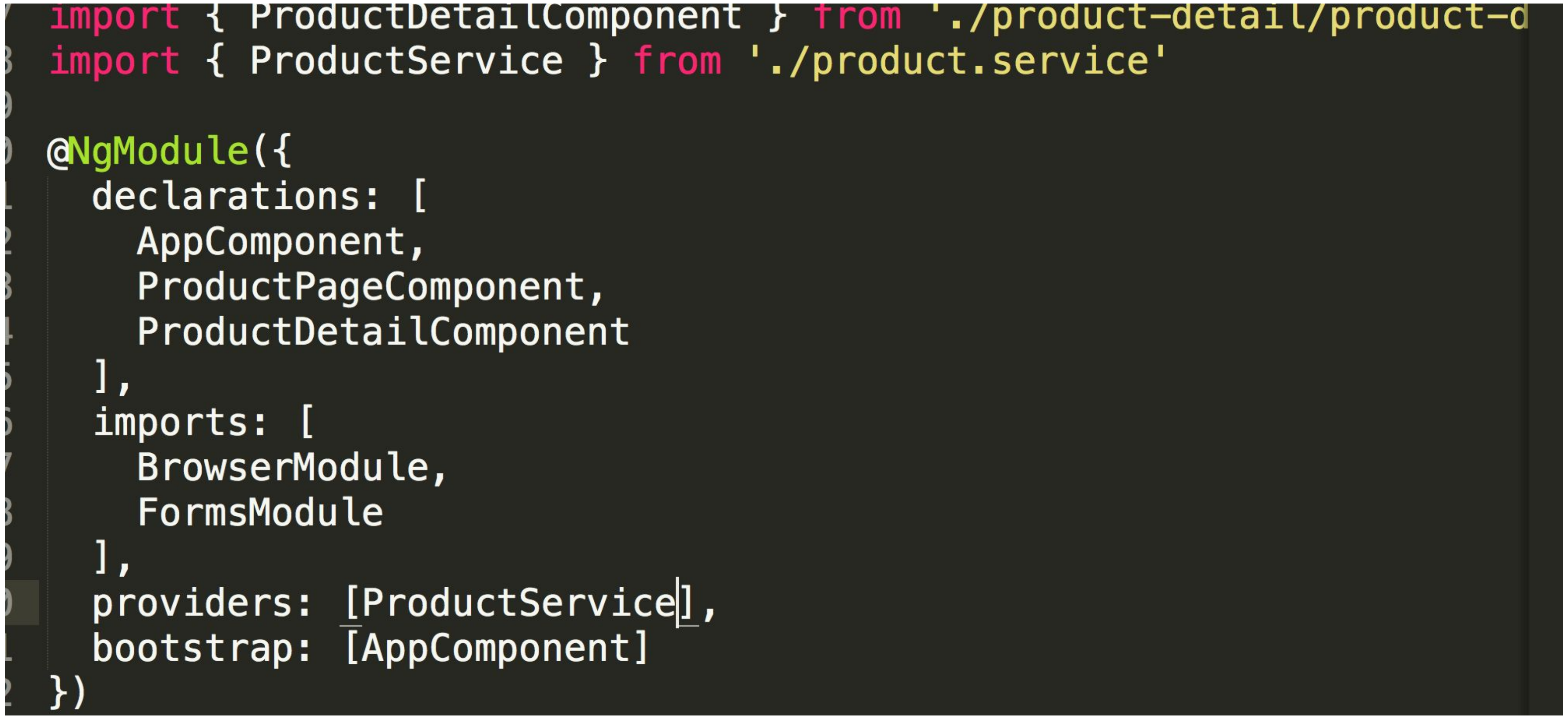


```
        {{product.price}}<br/>
    </li>
</ul>
<app-product-detail [product]=selectedProduct></
app-product-detail>
```

product-page.component.html

ORGANISING CODE WITH SERVICES

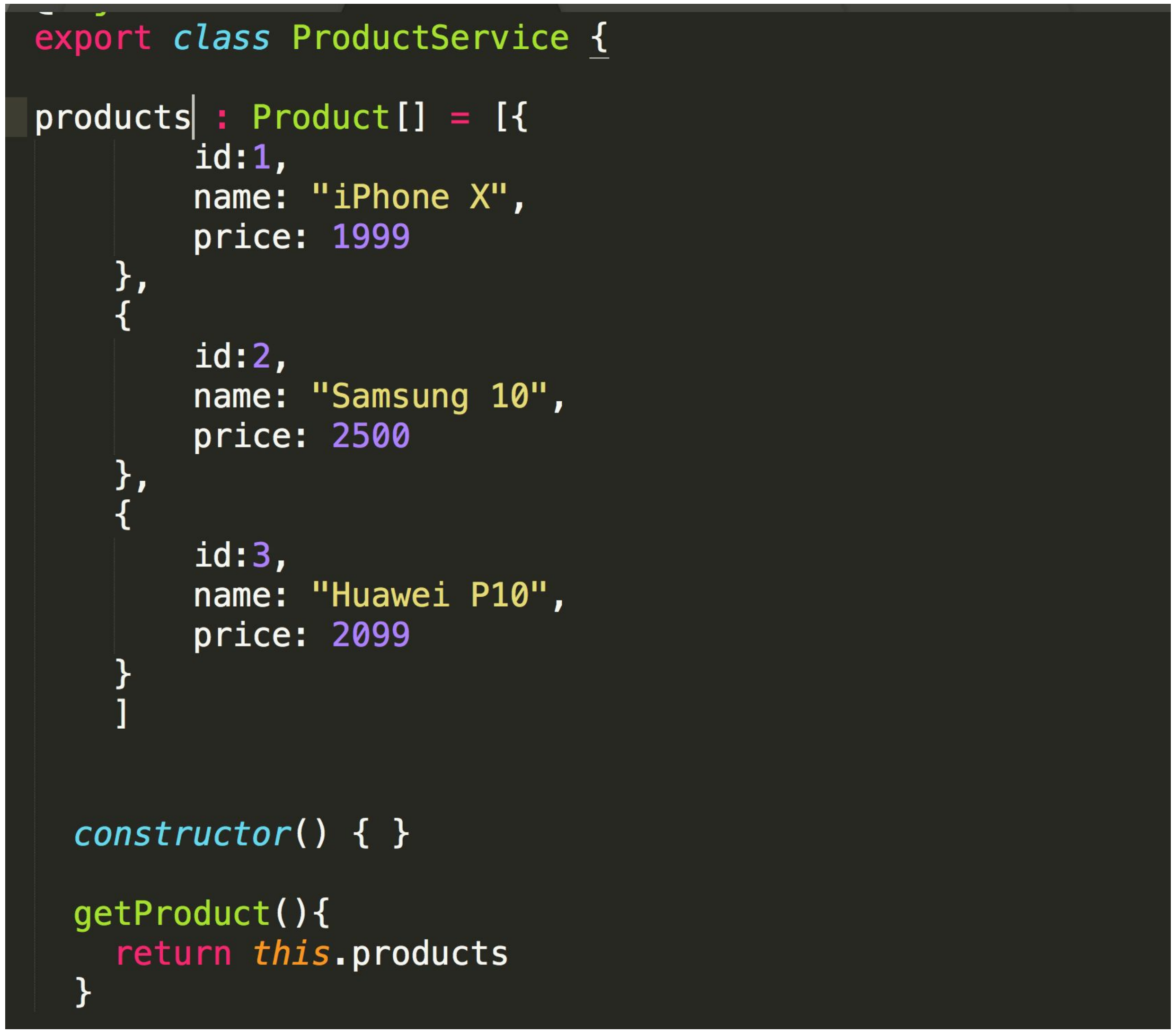
- Generate new service
  - ng g service products
- Add the product service into app.module.ts



```
import { ProductDetailComponent } from './product-detail/product-d
import { ProductService } from './product.service'

@NgModule({
  declarations: [
    AppComponent,
    ProductPageComponent,
    ProductDetailComponent
  ],
  imports: [
    BrowserModule,
    FormsModule
  ],
  providers: [ProductService],
  bootstrap: [AppComponent]
})
```

ORGANISING CODE WITH SERVICES (2)

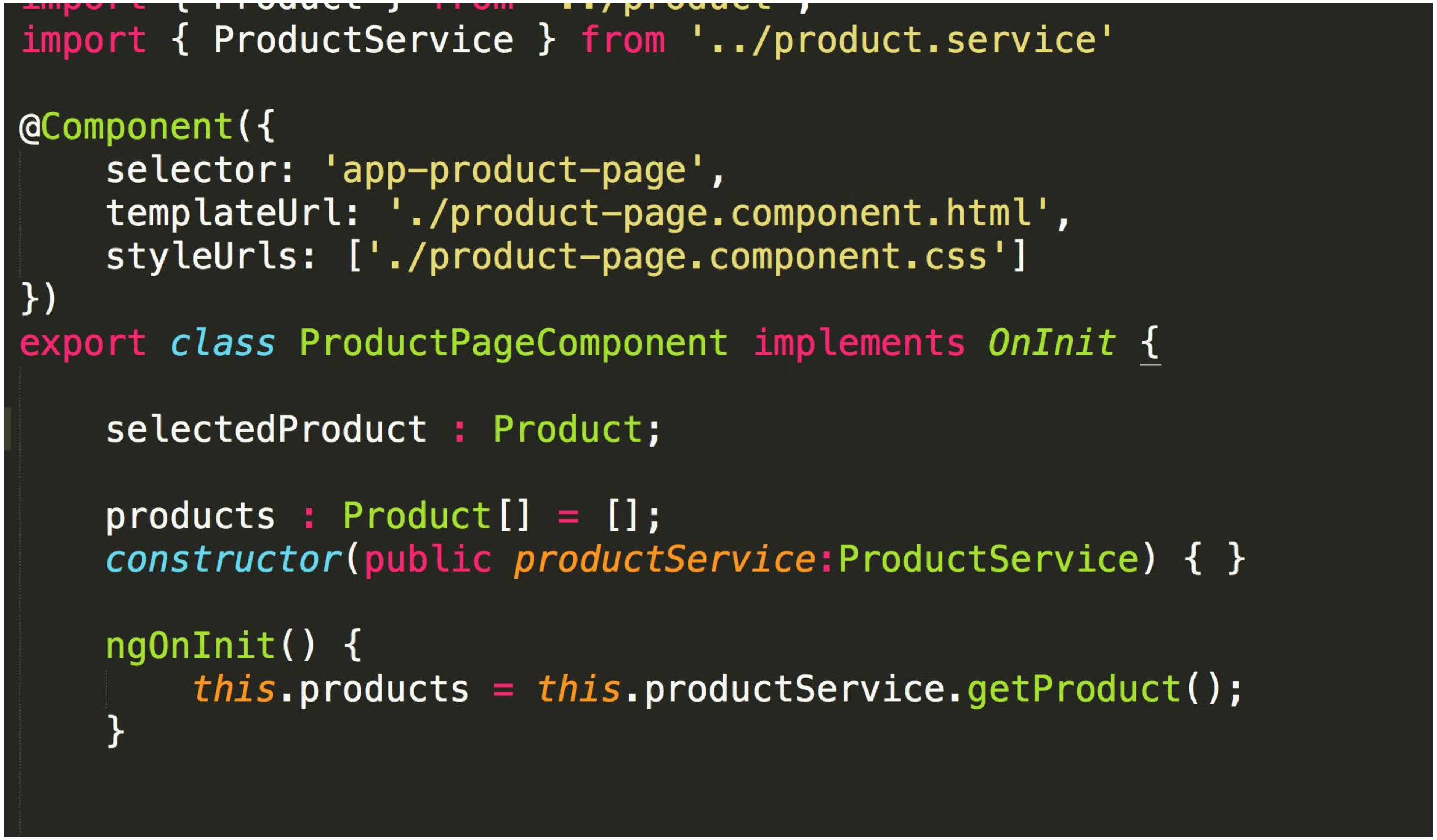
- Move the data into service.
- Create a get function to retrieve data from service



```
export class ProductService {

products : Product[] = [{
        id:1,
        name: "iPhone X",
        price: 1999
    },
    {
        id:2,
        name: "Samsung 10",
        price: 2500
    },
    {
        id:3,
        name: "Huawei P10",
        price: 2099
    }
    ]


  constructor() { }

  getProduct(){
    return this.products
  }
```

ORGANISING CODE WITH SERVICES (3)

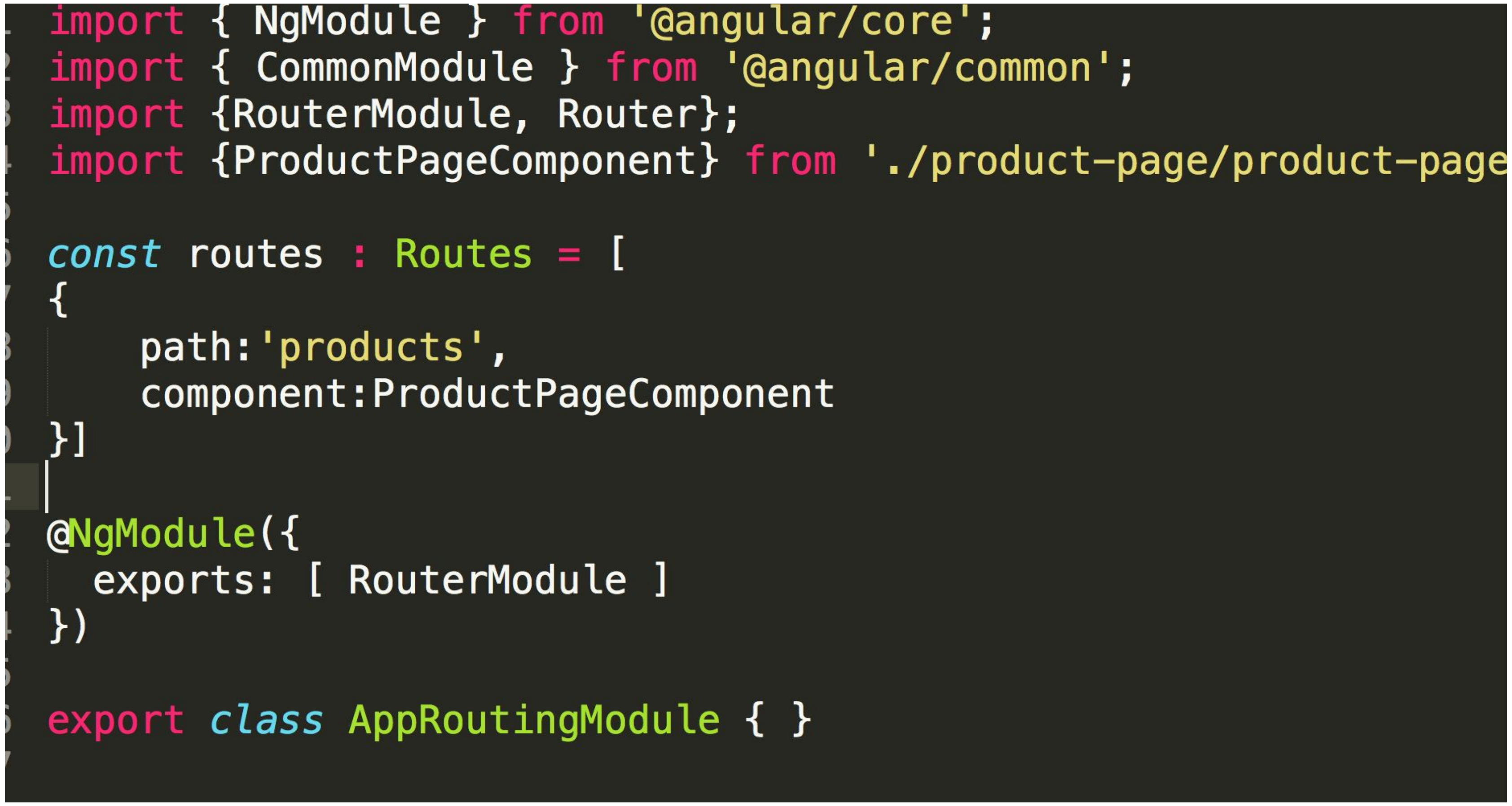
- Inject and call the getProduct function from product-page



```
import { Product } from '../product';
import { ProductService } from '../product.service'

@Component({
    selector: 'app-product-page',
    templateUrl: './product-page.component.html',
    styleUrls: ['./product-page.component.css']
})
export class ProductPageComponent implements OnInit {

    selectedProduct : Product;

    products : Product[] = [];
    constructor(public productService:ProductService) { }

    ngOnInit() {
        this.products = this.productService.getProduct();
    }
```

ROUTING WITH ANGULAR

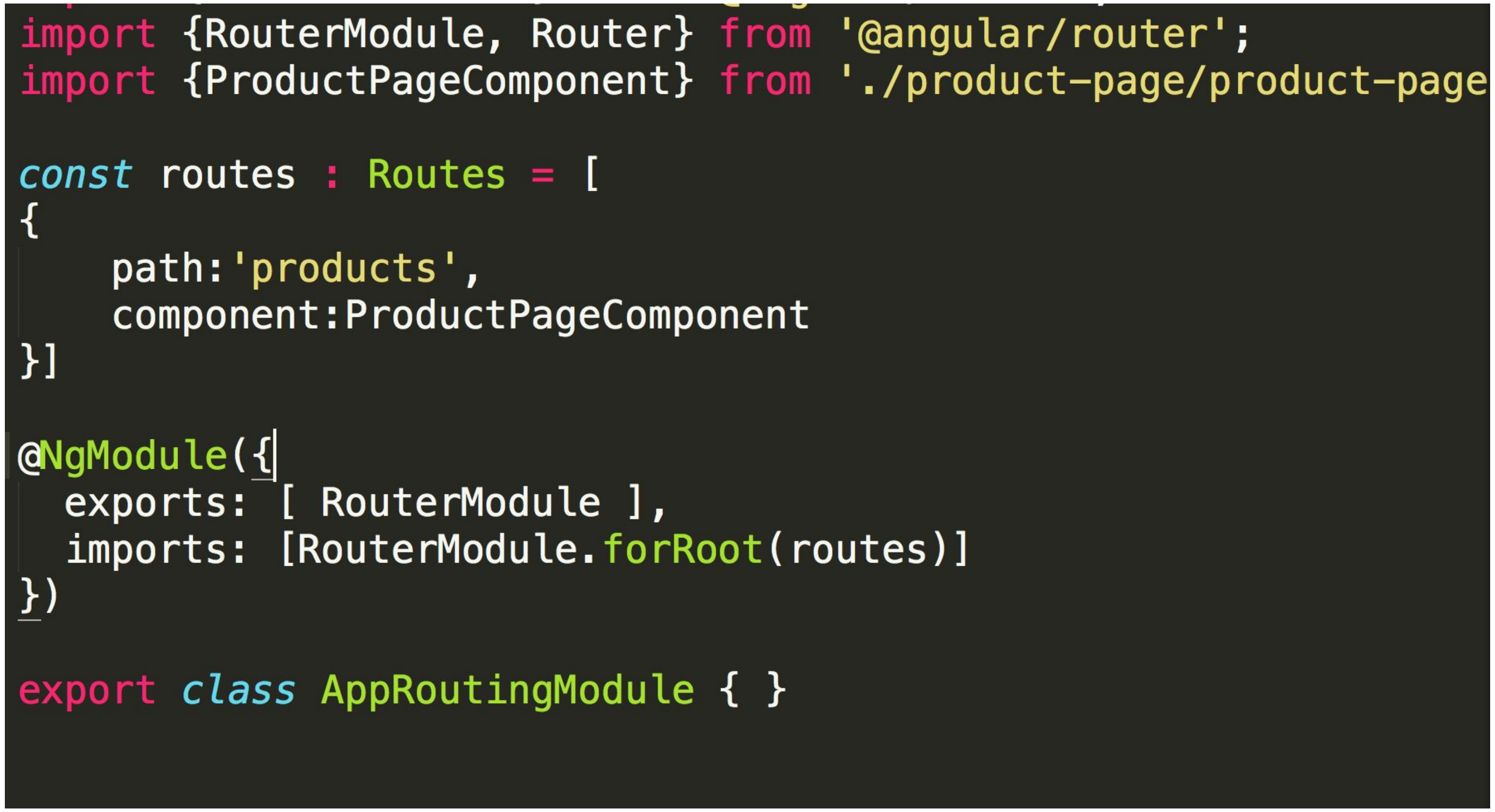
- Generate new module:
  - ng generate module app-routing --flat --module=app
- Define the route, export the module.



```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import {RouterModule, Router};
import {ProductPageComponent} from './product-page/product-page

const routes : Routes = [
{
    path:'products',
    component:ProductPageComponent
}]

@NgModule({
  exports: [ RouterModule ]
})

export class AppRoutingModule { }
```

ROUTING WITH ANGULAR

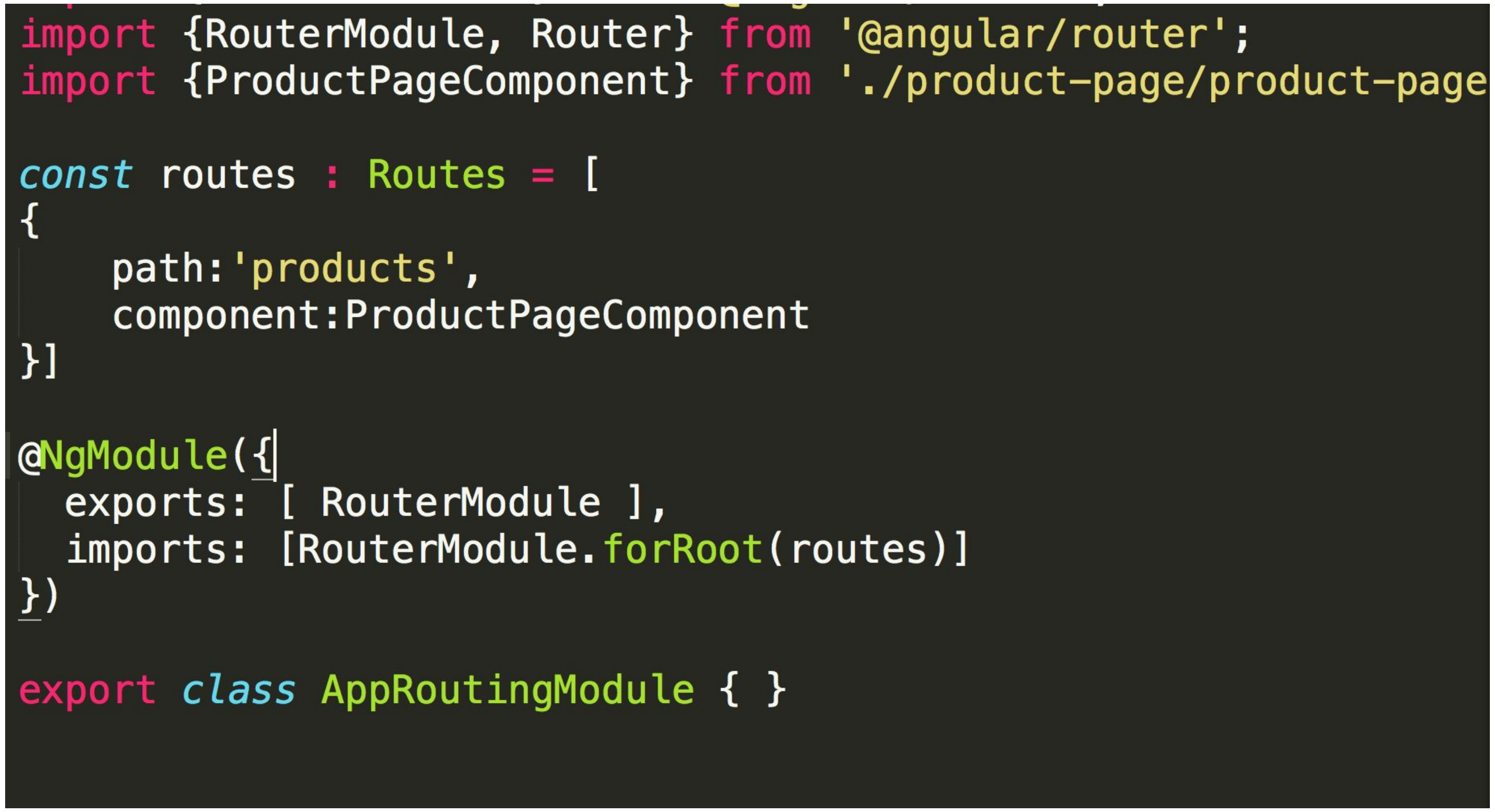
- Generate new module:
    - ng generate module app-routing --flat --module=app
- Define the route, export the module.



```
import {RouterModule, Router} from '@angular/router';
import {ProductPageComponent} from './product-page/product-page

const routes : Routes = [
{
    path:'products',
    component:ProductPageComponent
}]

@NgModule({
  exports: [ RouterModule ],
  imports: [RouterModule.forRoot(routes)]
})

export class AppRoutingModule { }
```

ROUTING WITH ANGULAR

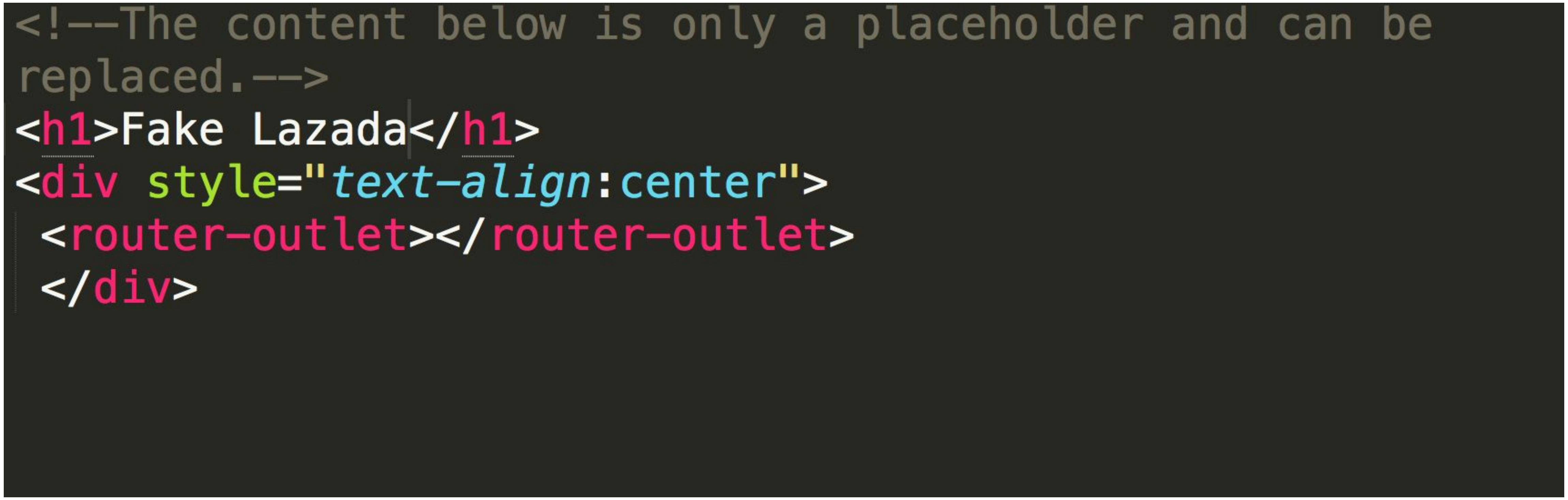
- Generate new module:
  - ng generate module app-routing --flat --module=app
- Define the route, export the module.



```
import {RouterModule, Router} from '@angular/router';
import {ProductPageComponent} from './product-page/product-page

const routes : Routes = [
{
    path:'products',
    component:ProductPageComponent
}]

@NgModule({
  exports: [ RouterModule ],
  imports: [RouterModule.forRoot(routes)]
})

export class AppRoutingModule { }
```

ROUTING WITH ANGULAR (2)

- In app.component.html, change the code and add router-outlet element as follows:



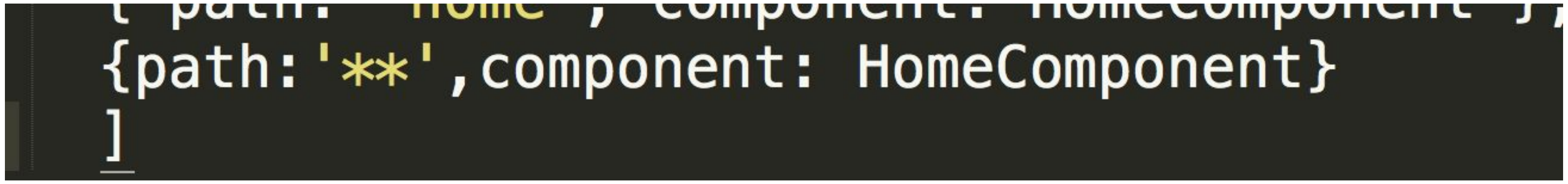
```
<!--The content below is only a placeholder and can be
replaced.-->
<h1>Fake Lazada</h1>
<div style="text-align:center">
 <router-outlet></router-outlet>
 </div>
```

ROUTING WITH ANGULAR (EXERCISE)

- Add a new component, call it home.
- Inside home html, add a simple welcome message.
- Create route to home.

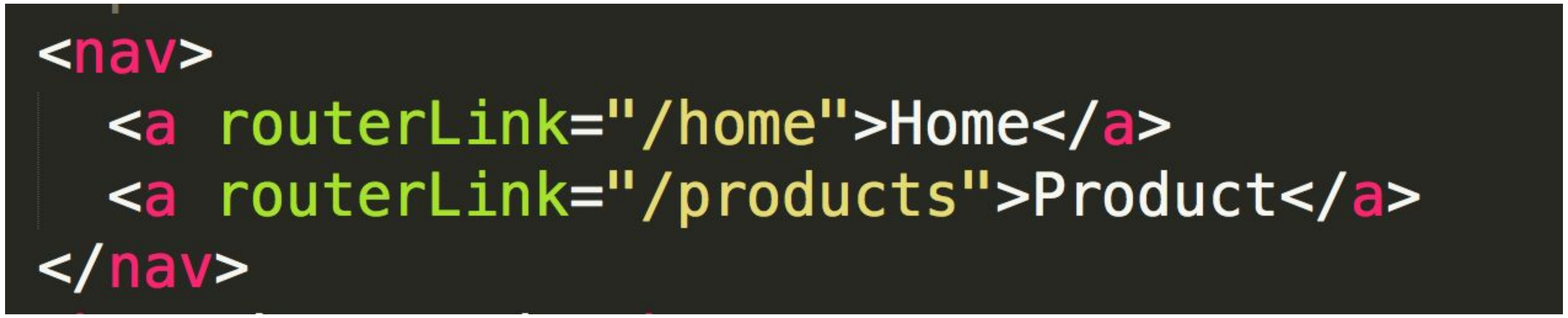
ROUTING WITH ANGULAR (3)

- Create an ‘otherwise’ routing as below



```
{ path: 'home', component: HomeComponent },
{path:'**',component: HomeComponent}
]
```

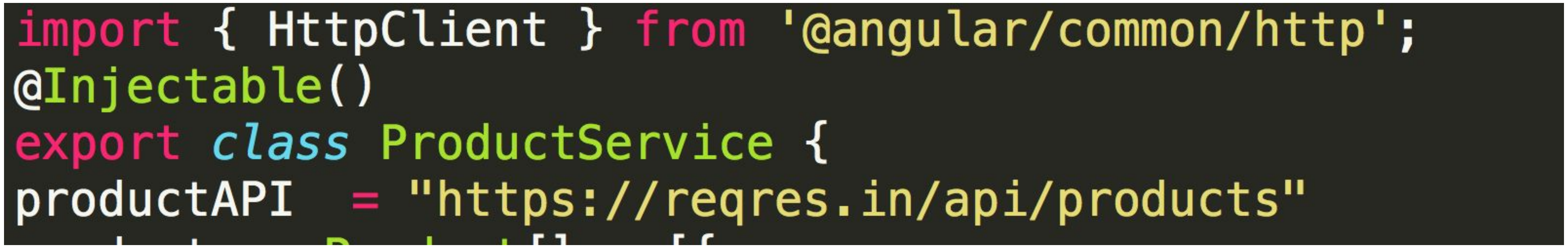
- Create link from one page to another using router-link



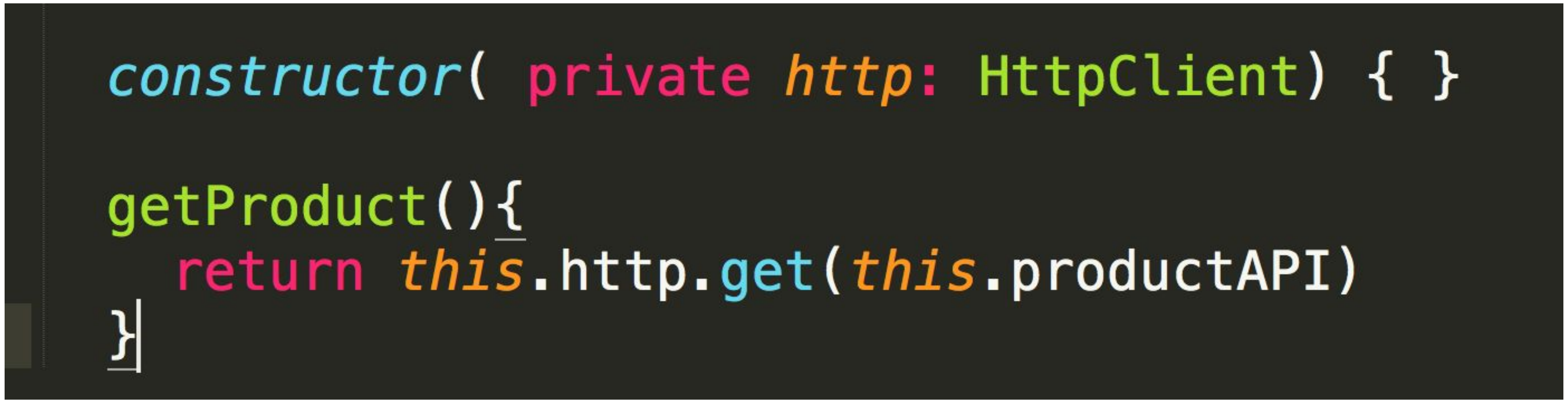
```
<nav>
  <a routerLink="/home">Home</a>
  <a routerLink="/products">Product</a>
</nav>
```

HTTP REQUEST USING ANGULAR (1)

- Import HTTPClient in product.service
- Inject HTTP Client in the service.
- Call http.get API



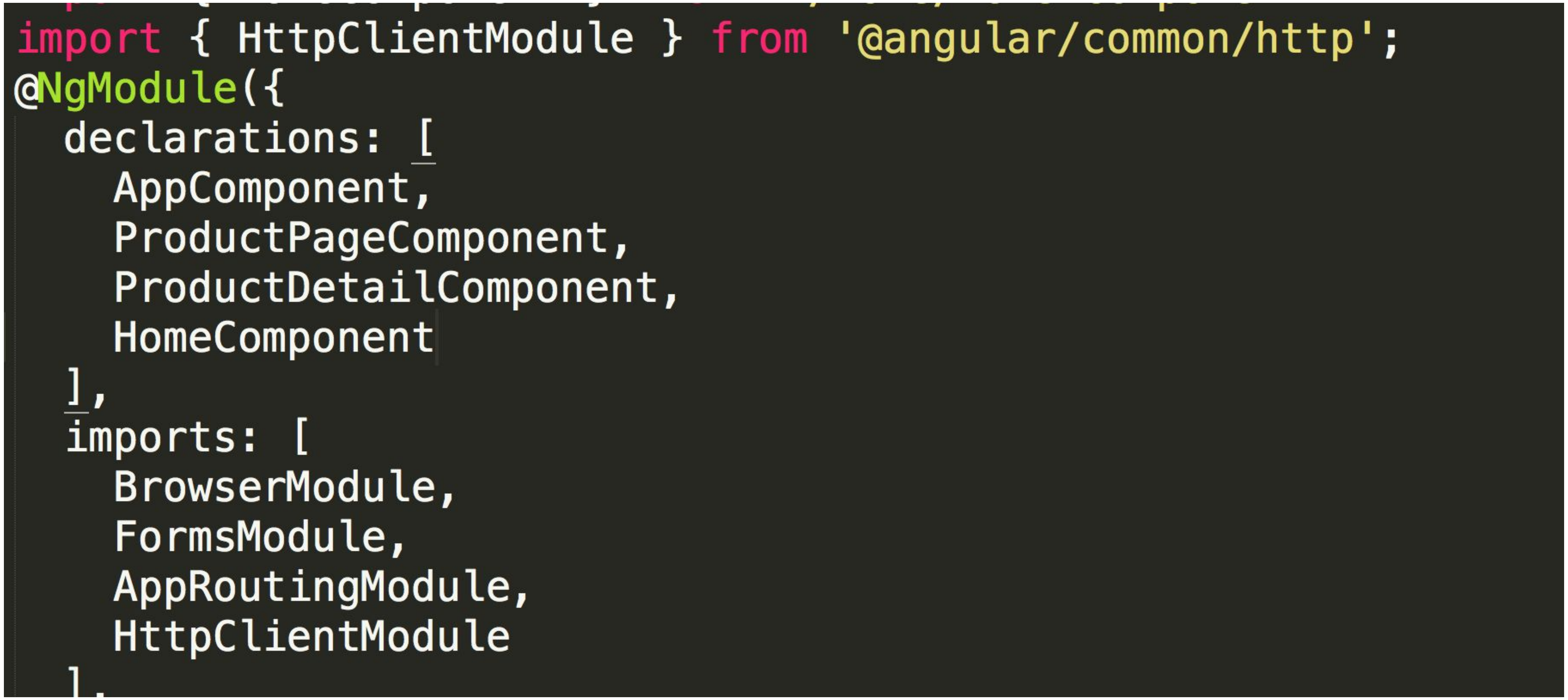
```
import { HttpClient } from '@angular/common/http';
@Injectable()
export class ProductService {
productAPI  = "https://reqres.in/api/products"
```



```
  constructor( private http: HttpClient) { }

  getProduct(){
    return this.http.get(this.productAPI)
  }
```

HTTP REQUEST USING ANGULAR (2)

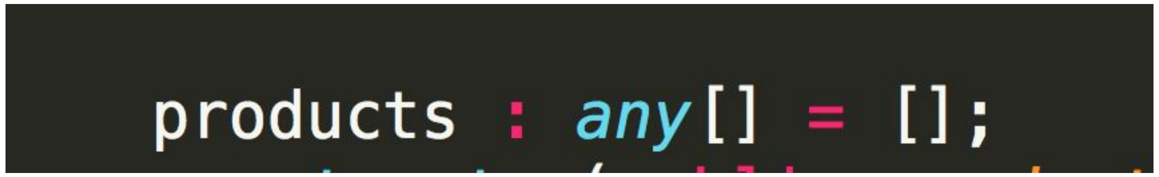
- Import HTTPClientModule in App.module.
- Add the module under imports



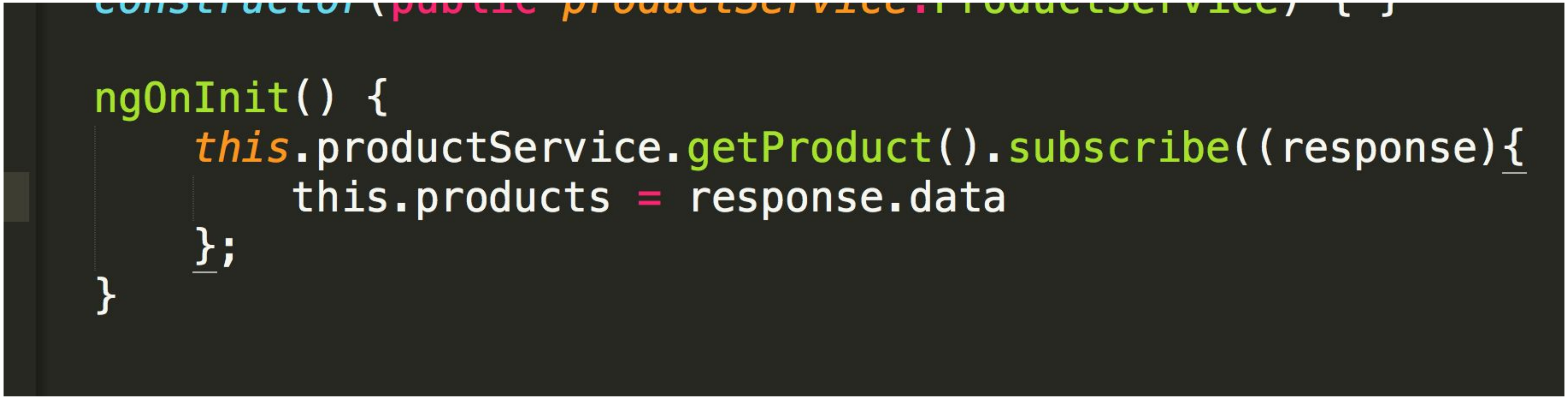
```
import { HttpClientModule } from '@angular/common/http';
@NgModule({
  declarations: [
    AppComponent,
    ProductPageComponent,
    ProductDetailComponent,
    HomeComponent
  ],
  imports: [
    BrowserModule,
    FormsModule,
    AppRoutingModule,
    HttpClientModule
  ],
```

HTTP REQUEST USING ANGULAR (3)

- In product-page component change the code to retrieve the data from HTTP.
- We will subscribe to the result from Service. This is call Observable in Javascript



```
  products : any[] = [];
```



```
constructor(public productService:ProductService) { }

ngOnInit() {
    this.productService.getProduct().subscribe((response){
        this.products = response.data
    };
}
```