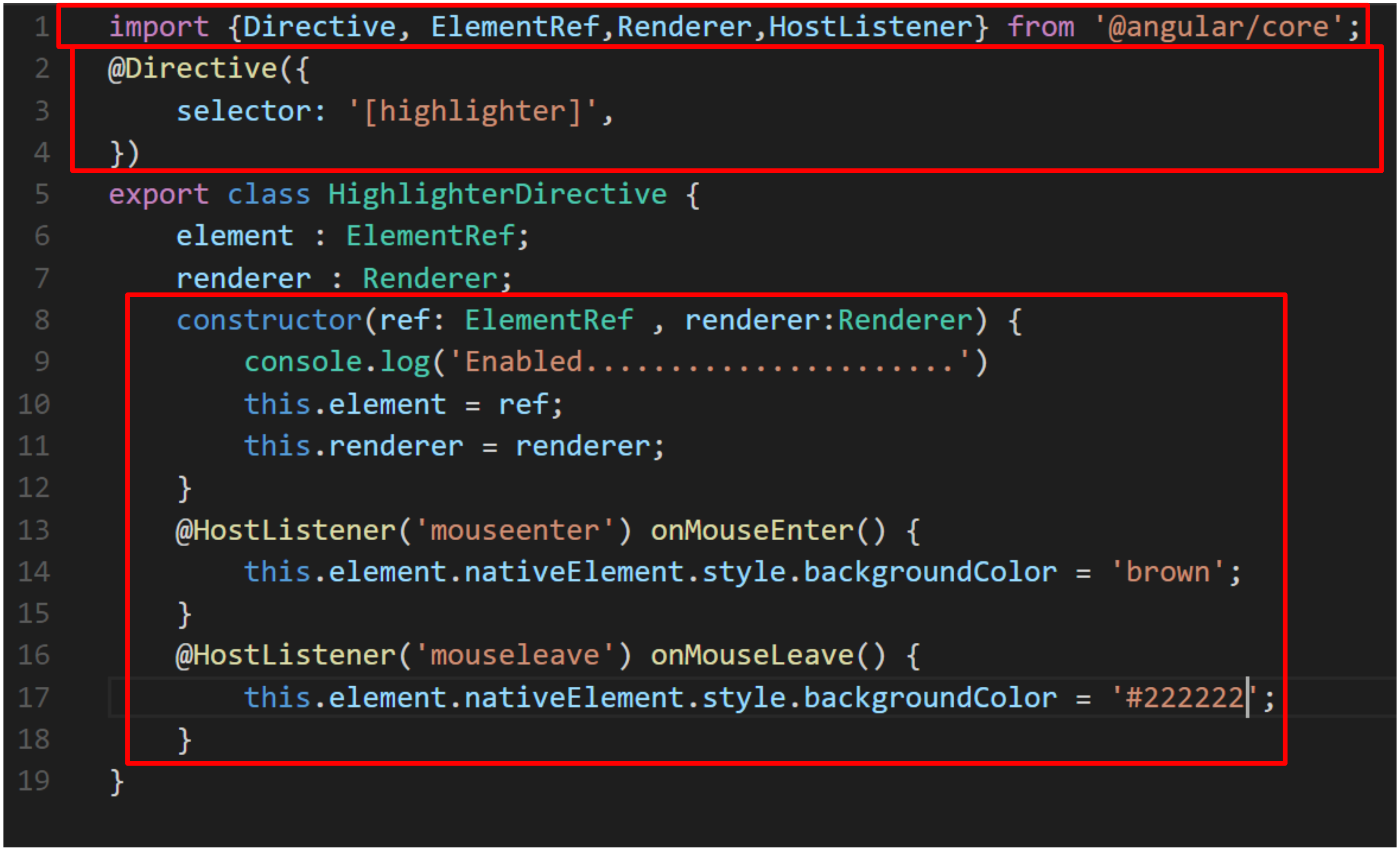
Directives

Directives

- **There are three kinds of directives in Angular:**
- **Components**
  - directives with a template.
- **Structural directives**
  - change the DOM layout by adding and removing DOM elements
    - ngFor [tried with our menu]
    - ngIf[to be tried with login authentication]
    - ngSwitch [to be tried with login authorization]
- **Attribute directive**
  - change the appearance or behavior of an element.

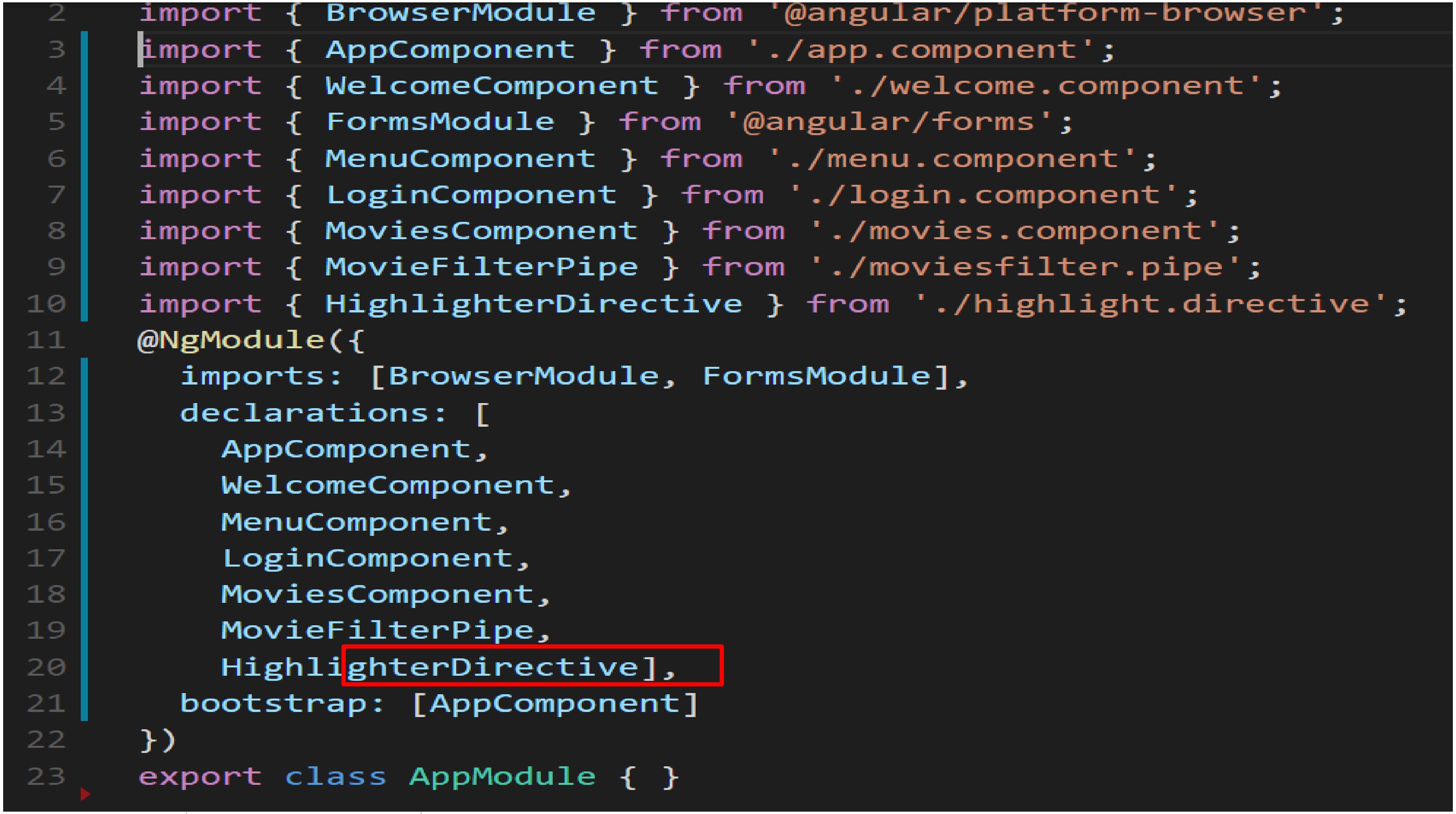
  We will create a style highlighter where onmouseover and mouseout the given component would get highlighted

Directive Example



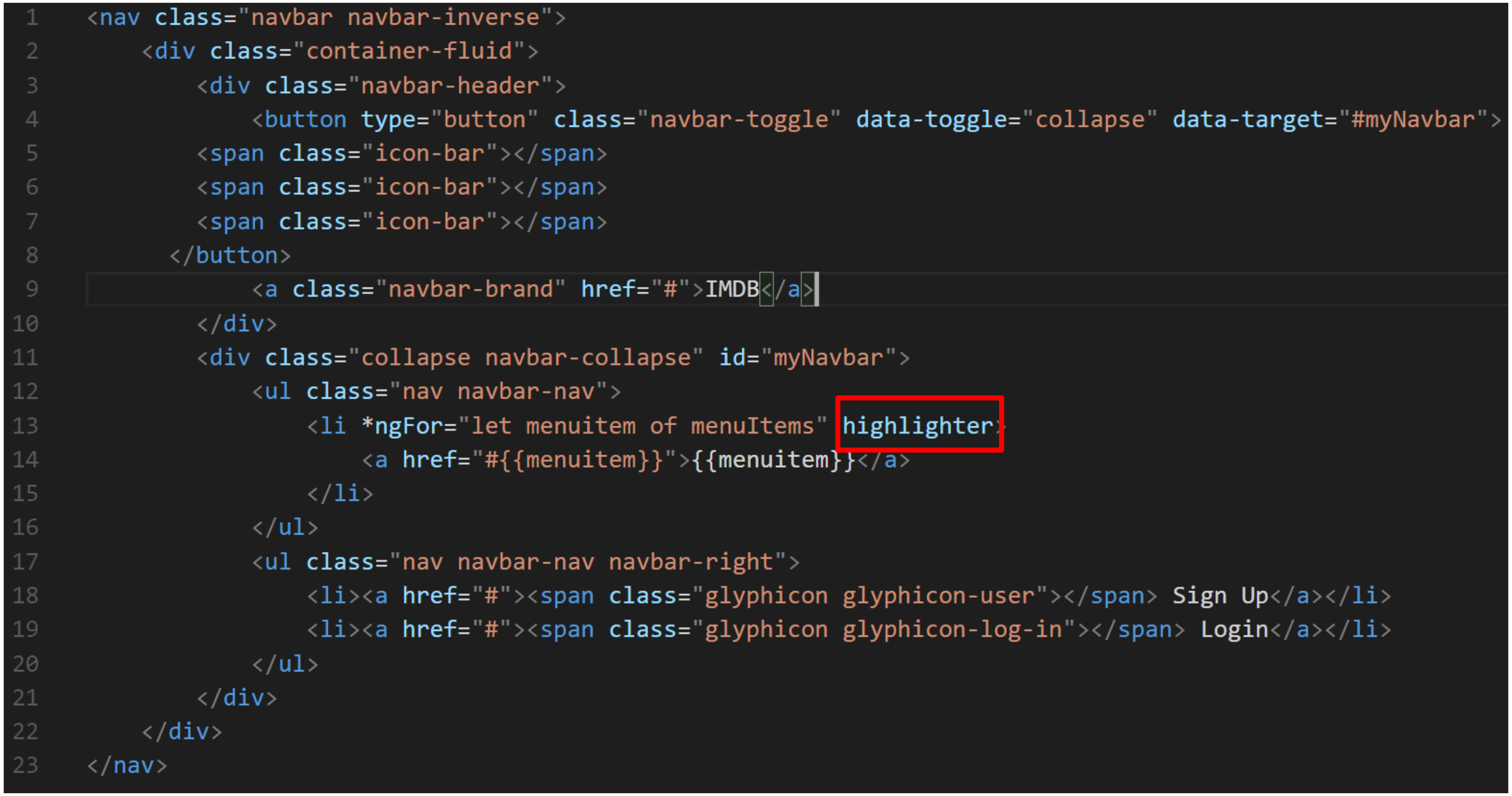
```
import {Directive, ElementRef,Renderer,HostListener} from '@angular/core';
@Directive({
    selector: '[highlighter]',
})
export class HighlighterDirective {
    element : ElementRef;
    renderer : Renderer;
    constructor(ref: ElementRef , renderer:Renderer) {
        console.log('Enabled......................')
        this.element = ref;
        this.renderer = renderer;
    }
    @HostListener('mouseenter') onMouseEnter() {
        this.element.nativeElement.style.backgroundColor = 'brown';
    }
    @HostListener('mouseleave') onMouseLeave() {
        this.element.nativeElement.style.backgroundColor = '#222222';
    }
}
```

Register the directive



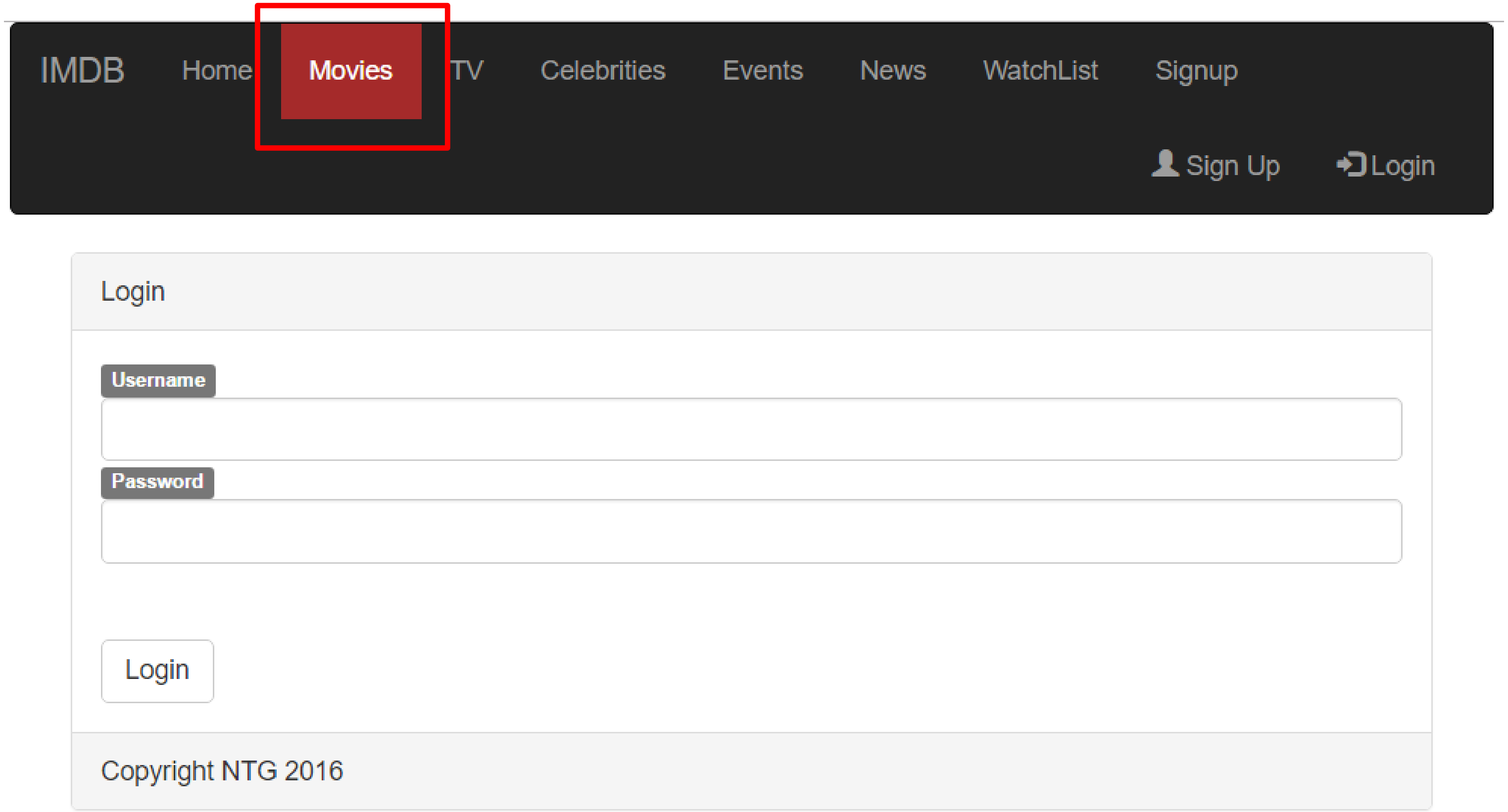
```
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent } from './app.component';
import { WelcomeComponent } from './welcome.component';
import { FormsModule } from '@angular/forms';
import { MenuComponent } from './menu.component';
import { LoginComponent } from './login.component';
import { MoviesComponent } from './movies.component';
import { MovieFilterPipe } from './moviesfilter.pipe';
import { HighlighterDirective } from './highlight.directive';
@NgModule({
  imports: [BrowserModule, FormsModule],
  declarations: [
    AppComponent,
    WelcomeComponent,
    MenuComponent,
    LoginComponent,
    MoviesComponent,
    MovieFilterPipe,
    HighlighterDirective],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Using the Highlight Directive



```
<nav class="navbar navbar-inverse">
    <div class="container-fluid">
        <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
            <a class="navbar-brand" href="#">IMDB</a>
        </div>
        <div class="collapse navbar-collapse" id="myNavbar">
            <ul class="nav navbar-nav">
                <li *ngFor="let menuitem of menuItems" highlighter>
                    <a href="#{{menuitem}}">{{menuitem}}</a>
                </li>
            </ul>
            <ul class="nav navbar-nav navbar-right">
                <li><a href="#"><span class="glyphicon glyphicon-user"></span> Sign Up</a></li>
                <li><a href="#"><span class="glyphicon glyphicon-log-in"></span> Login</a></li>
            </ul>
        </div>
    </div>
</nav>
```

The Output



Login

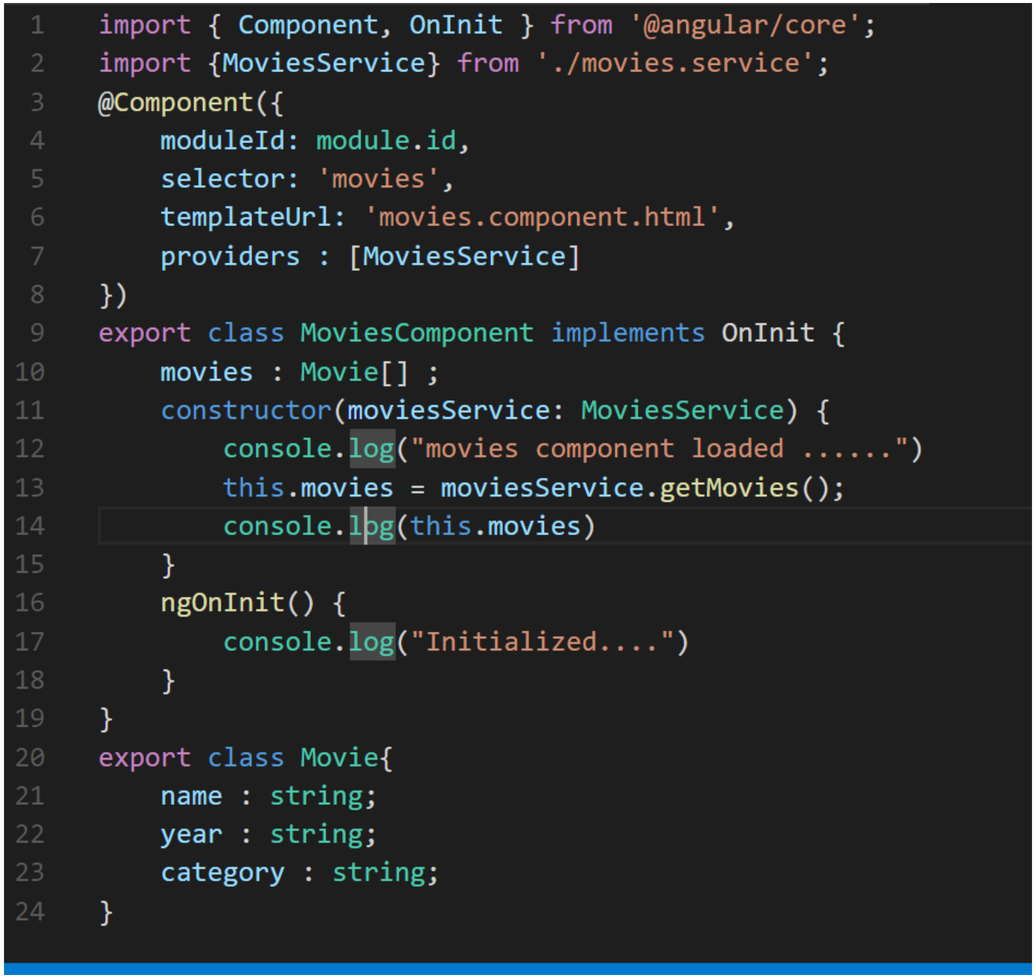
Username

Password

Copyright NTG 2016

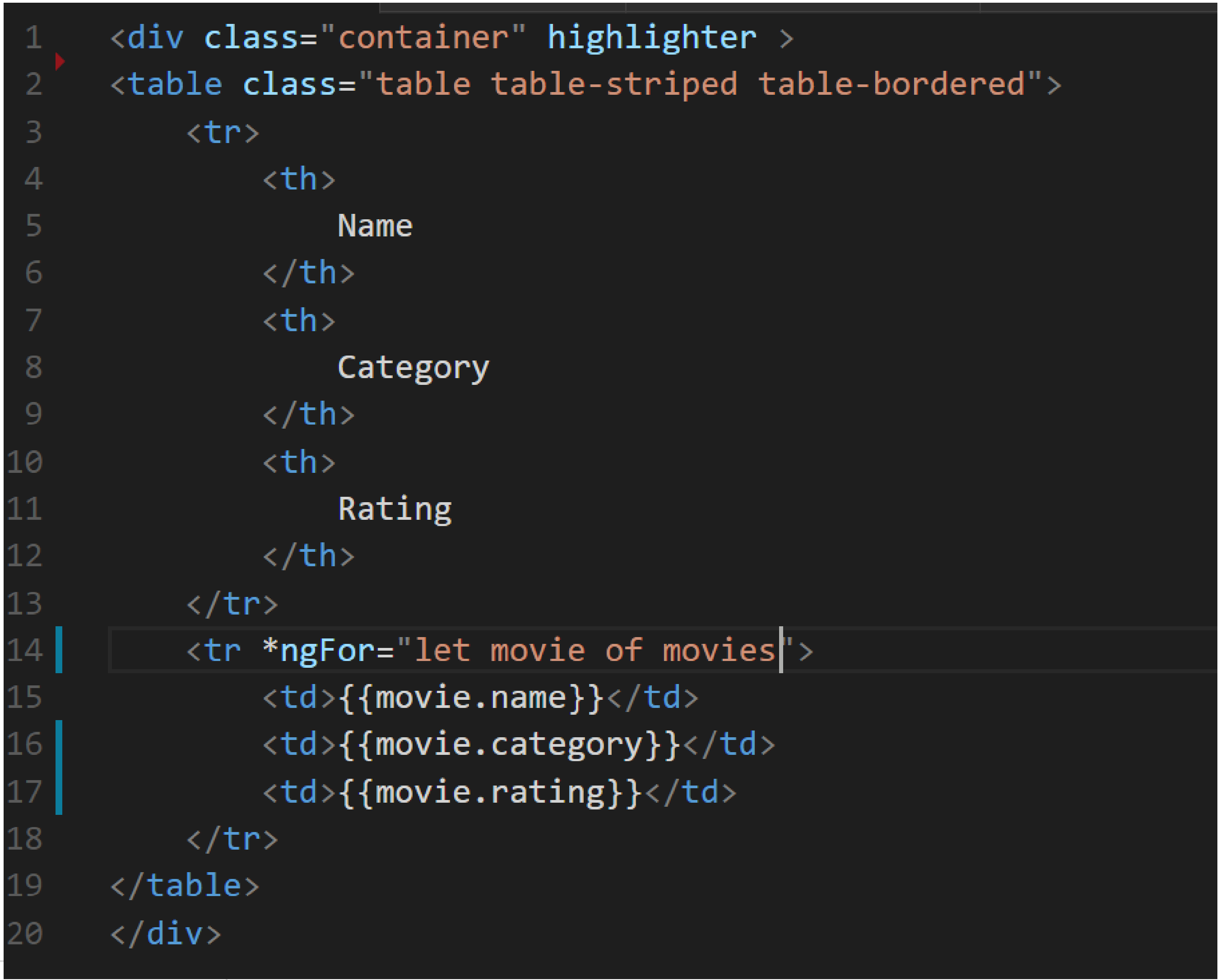
Developing the Movies Component

The movies Component



```
import { Component, OnInit } from '@angular/core';
import {MoviesService} from './movies.service';
@Component({
    moduleId: module.id,
    selector: 'movies',
    templateUrl: 'movies.component.html',
    providers : [MoviesService]
})
export class MoviesComponent implements OnInit {
    movies : Movie[] ;
    constructor(moviesService: MoviesService) {
        console.log("movies component loaded ......")
        this.movies = moviesService.getMovies();
        console.log(this.movies)
    }
    ngOnInit() {
        console.log("Initialized....")
    }
}
export class Movie{
    name : string;
    year : string;
    category : string;
}
```

Movies Template



```html
<div class="container" highlighter >
<table class="table table-striped table-bordered">
    <tr>
        <th>
            Name
        </th>
        <th>
            Category
        </th>
        <th>
            Rating
        </th>
    </tr>
    <tr *ngFor="let movie of movies">
        <td>{{movie.name}}</td>
        <td>{{movie.category}}</td>
        <td>{{movie.rating}}</td>
    </tr>
</table>
</div>
```

Component Communication

```
import { Component } from '@angular/core';

 @Component({
  selector: 'app-root',
  template: `<h1 [ngClass]={myclass:one,myclass2:two}>hello</h1>
  {{name}} {{sal}} {{3+6}} {{getName()}}
  From Child Component {{message.name}}

  <app-person [parentData]="pData" (myevent)="message=$event"></app-person>
  <input type="text"  [(ngModel)]="cdata"/>
  <button type="button" (click)="changePdata()">pass data to child</button>
  <div *ngIf="test">this is test for ngif</div>
  `,
  styleUrls: ['./app.component.css']

})
export class AppComponent {
```

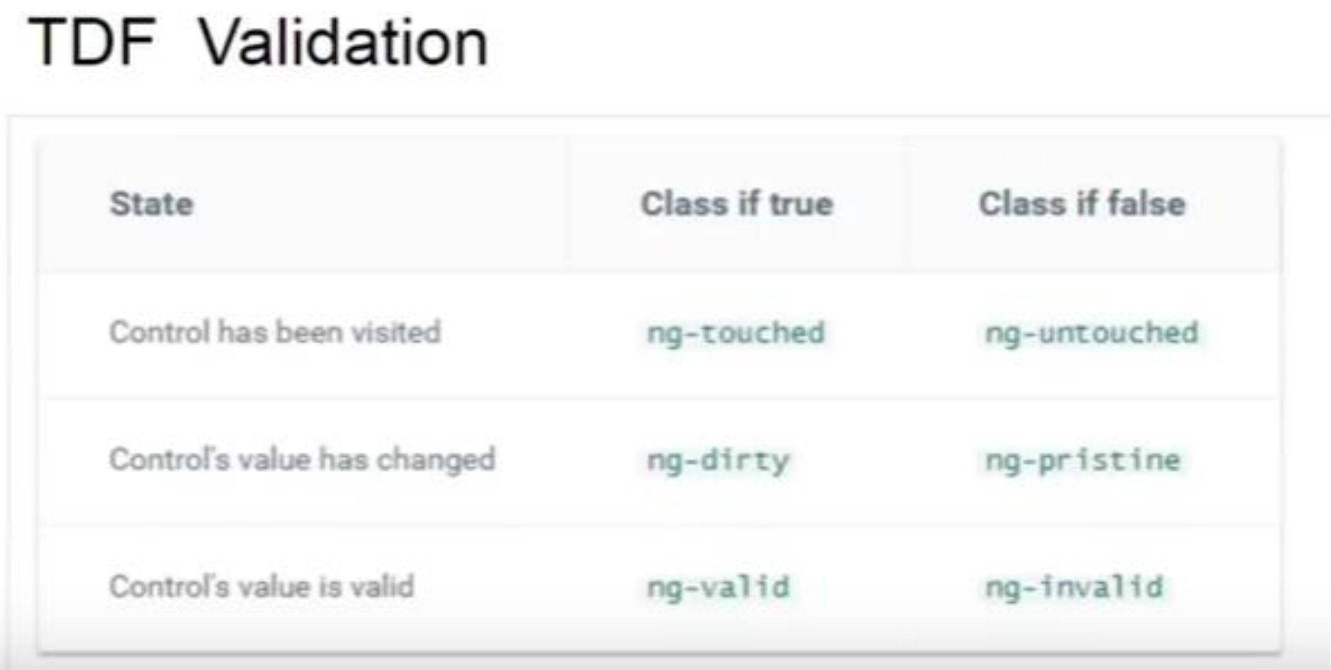
```
export class AppComponent {
 cdata:string;
 pData="Testing data from parent";
 public message;
 name="kishori";
 sal=1234.45;
 test:boolean=false;
 one=false;
 two=true;
 getName(){
  return 'abcd';
 }
 changePdata(){
  this.pData="abc change from parent";
 }
```

Component Communication- Child Component (person.component.ts)

```
import { Component, Input, Output,EventEmitter } from '@angular/core';


@Component({
   selector:'app-person',
   template:`
        <div>This is person Compnent {{strmsg}}</div>
        <button (click)=onclick()>click me</button>
   `
})
export class PersonComponent{
per={name:"kishori",address:"Aundh"}
@Input('parentData') public strmsg;
@Output() public myevent=new EventEmitter();
onclick(){
   this.myevent.emit(this.per);
}

}
```



TDF Validation

State	Class if true	Class if false
Control has been visited	`ng-touched`	`ng-untouched`
Control's value has changed	`ng-dirty`	`ng-pristine`
Control's value is valid	`ng-valid`	`ng-invalid`

Form Handling(Template driven Forms)

```

{{title|uppercase}}
<div class="form-group">
<form #pform="ngForm"  (ngSubmit)="onsubmit(pform.value)" >
 Person Id : <input #myidRef="ngModel" class="form-control" type="text" name="pid"
required minlength="3" ngModel><br/>
 <div *ngIf="myidRef.errors && (myidRef.dirty || myidRef.touched)">

 <div [hidden]="!myidRef.errors.required">
   Pls enter valid data
</div>
<div [hidden]="!myidRef.errors.minlength">
    Pls enter min length to 3
 </div>
</div>


 <br><br> Person Name :<input type="text" name="name" ngModel>
 Person Designation <input type="text" class="form-control" name="designation"
ngModel>
  <input type="submit" name="btn1" value="Submit">

</form>
</div>
```

WebServiceDemo – app.module.ts

```
import { HttpClientModule, HttpClient } from '@angular/common/http’;
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { PtabComponent } from './person/person.component';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { HttpModule } from '@angular/http';
import {PerService} from './person.service';
import { Person } from './Person';
import {FormsModule } from '@angular/forms';
@NgModule({
  declarations: [
    AppComponent,PtabComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,HttpClientModule,FormsModule
  ],
  providers: [PerService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Ptab.component.ts

```

import { PerService } from "../person.service";
import { Component } from "@angular/core";
import { Person } from "../Person"
@Component({
   selector:'ptab',
   templateUrl:'./person.component.html',
   styleUrls:['./person.component.css']
})
export class PtabComponent{
perarr:Person[];
showform=false;
p:Person={pid:0,pname:"",mobile:""};
choice:string="add";
constructor(private pservice:PerService){}
ngOnInit(){
   this.pservice.getPersons()
   .subscribe((r)=>this.perarr=<Person[]>r);

}

getPersonById(){
   this.pservice.getPersonById(this.p.pid)
   .subscribe((r)=>this.p=<Person>r);

}

```

Ptab.component.ts

```
updatePerson(p1:Person,ch:string){
    //to display form
    this.showform=true;
    //no need to call getPersonById we can pass the object from HTML
   //assign u to choice to call update function in onsubmit
   this.choice="ch";
   //display person in form for updation
   this.p={pid:p1.pid,pname:p1.pname,mobile:p1.mobile};
}

onsubmit(){
  //hide the form
  this.showform=false;
  if(this.choice=="add"){
    this.pservice.addPerson(this.p)
    .subscribe((r)=>this.perarr=<Person[]>r);
}else{
   //reset choice to add
   this.choice="add";
   this.pservice.updatePerson(this.p)
    .subscribe((r)=>this.perarr=<Person[]>r);
 } } }
```

WebService Communication-person.service.ts

- import { Observable } from 'rxjs';
- import { HttpClient } from '@angular/common/http';
- import { Response, Headers, RequestOptions, } from '@angular/http';
- import { Person } from './Person';
- import { Injectable } from '@angular/core';
- @Injectable()
- export class PerService{
-
-         personUrl="http://localhost:9090/PersonWebService123/persons";
-
-   constructor(private http: HttpClient){
-
- }

Person.ts

```
export class Person{
    pid;
    pname;
    mobile;

}
```