Advanced Angular

Lecture 2

Agenda

- Recap last lecture points
- Adding DevTools
- Interceptors
- Environment variables
- Angular application deployment
- Firebase with Angular

DevTools

For Angular App

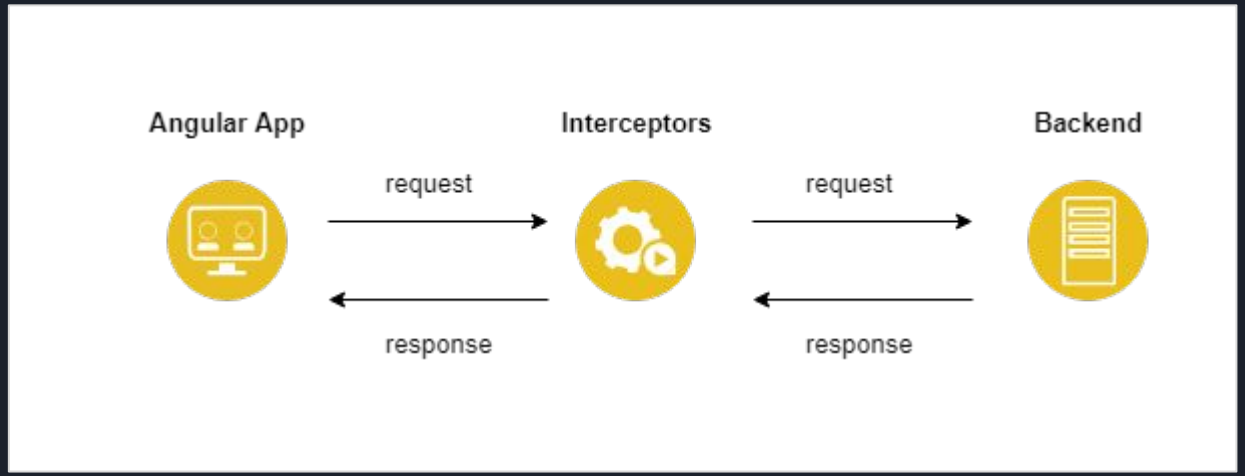
Setup one of the following extensions:

- Augury
- Angular DevTools

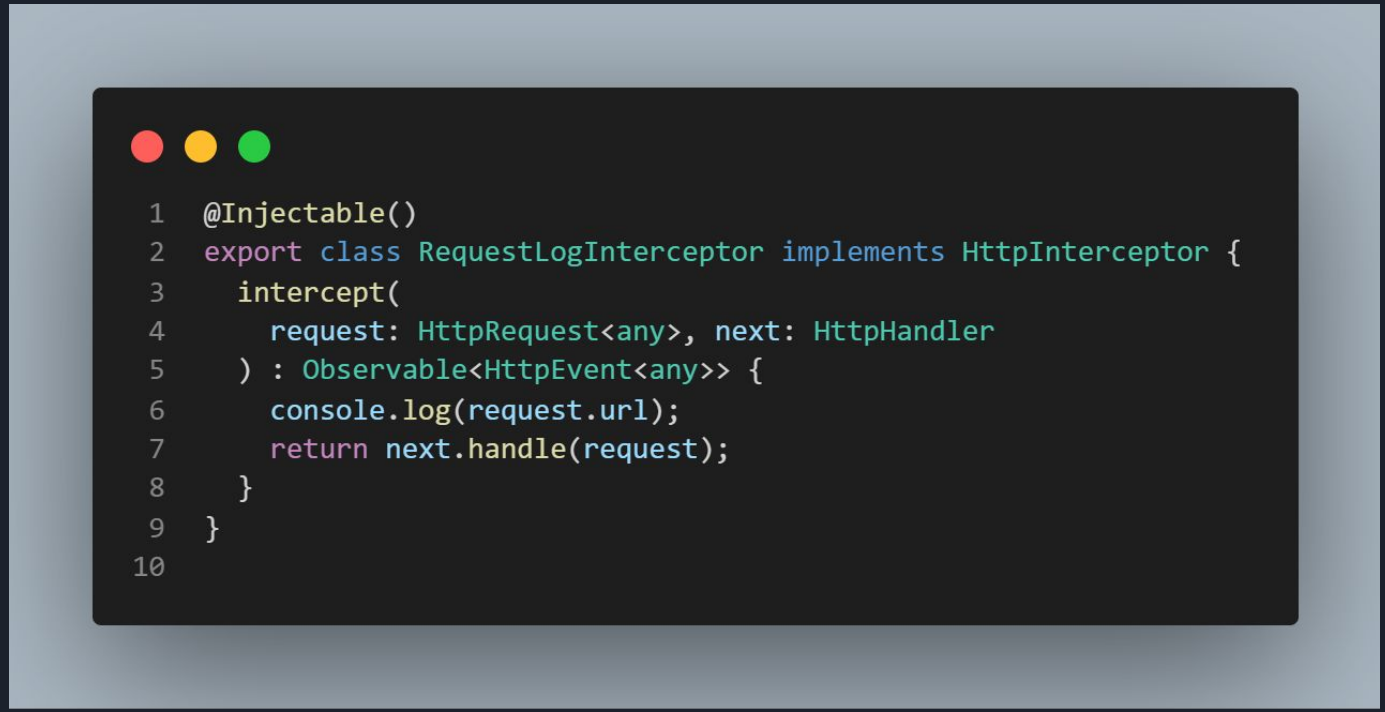
Interceptors

- Interceptors provide a mechanism to intercept and/or mutate outgoing requests or incoming responses.
- It’s a simple way provided by the framework to intercept and modify the application’s http requests globally before they are sent to the server.
- In order to implement an Interceptor, you need to create a class that implements the intercept method of the HttpInterceptor interface.

Interceptors



Interceptors



```
@Injectable()
export class RequestLogInterceptor implements HttpInterceptor {
  intercept(
    request: HttpRequest<any>, next: HttpHandler
  ) : Observable<HttpEvent<any>> {
    console.log(request.url);
    return next.handle(request);
  }
}
```

Interceptors

In app.module :

```
{ provide: HTTP_INTERCEPTORS, useClass: LoadingInterceptor, multi: true },
```

In interceptor :

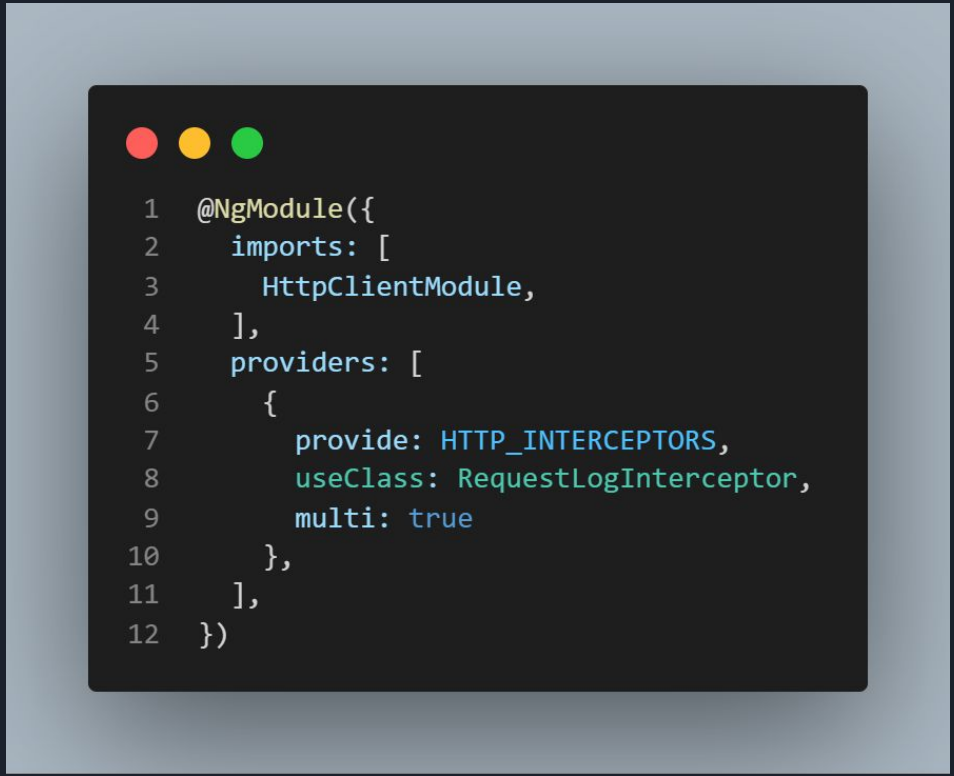
```
import { finalize } from 'rxjs/operators';

intercept(
  request: HttpRequest<unknown>,
  next: HttpHandler
): Observable<HttpEvent<unknown>> {
  const authReq = request.clone({
    headers: new HttpHeaders({
      'Content-Type': 'application/json',
      Authorization: 'my-auth-token',
    }),
  });
  return next.handle(authReq).pipe(finalize(() => alert('done')));
}
```

Interceptors

Providing the Interceptor :

- Since interceptors are dependencies of the HttpClient, you must add them to providers of app module.
- The multi: true option provided tells Angular that you are providing multiple interceptors if found.



```
@NgModule({
  imports: [
    HttpClientModule,
  ],
  providers: [
    {
      provide: HTTP_INTERCEPTORS,
      useClass: RequestLogInterceptor,
      multi: true
    },
  ],
})
```

Fontawesome

You have 2 ways to use font awesome :

- **First install fontawesome free icons :**

npm install --save @fortawesome/fontawesome-free

- **Second add fontawesome css path to angular.json file after css in styles array:**

"node_modules/@fortawesome/fontawesome-free/css/all.css",

- Then use fontawesome directly

Fontawesome

Another way by installing all icons packages using all the following command lines :

npm install @fortawesome/fontawesome-svg-core

npm install @fortawesome/free-brands-svg-icons

npm install @fortawesome/free-regular-svg-icons

npm install @fortawesome/free-solid-svg-icons

npm install @fortawesome/angular-fontawesome

Website : https://www.npmjs.com/package/@fortawesome/angular-fontawesome

Fontawesome

Add fontawesome Module to appModule :

- import { FontAwesomeModule } from '@fortawesome/angular-fontawesome';

Add icon to ts file :

```
import { faCoffee } from '@fortawesome/free-solid-svg-icons';

export class AppComponent {

  faCoffee = faCoffee;

}
```

In HTML : <fa-icon [icon]="faCoffee"></fa-icon>

Environments Variables

What is Angular Environment Variable

The Environment Variable are those variables, whose value changes as per the environment we are in. This will help us to change some behavior of the App based on the environment.

First, import the default environment in the component. Note that you should not import any other environment files like environment.prod, but only the default environment file.

```
import { environment } from '../environments/environment';

this.key = environment.key;
```

Angular Deployment

Use command line :

ng build

to generate a build that used for deployment

ng build --prod

To generate an optimized build version for production

You can use vercel to deploy your application to live server and get url for your application

https://vercel.com/solutions/angular

Firebase

We are using angularFire to integrate firebase with angular as it’s the official Angular library for Firebase.

```
ng add @angular/fire
```

AngularFire smooths over the rough edges an Angular developer might encounter when implementing the framework-agnostic Firebase JS SDK & aims to provide a more natural developer experience by conforming to Angular conventions.

Firebase

After running the previous command the angularfire will create all the configuration needed to integrate with firebase.

Tha main and important changes are :

UPDATE src/app/app.module.ts

UPDATE src/environments/environment.ts

UPDATE src/environments/environment.prod.ts

Firebase

To read collection from firebase:

```
import { Firestore, collection , collectionData } from
'@angular/fire/firestore';

constructor(private firestore: Firestore) {
    // TO get collection
    const data = collection(firestore, 'movies');

    collectionData(data).subscribe(data => console.log(data)))

}
```

Firebase

To read document data:

```
import { Firestore,  doc , docData } from
'@angular/fire/firestore';

constructor(private firestore: Firestore) {

 // note : P0N8iMrfOUMgyHUzMS0L is the id of specific movie

 const dataItem = doc(firestore, 'movies/P0N8iMrfOUMgyHUzMS0L');
 docData(dataItem).subscribe(data => console.log(data));
}
```

Firebase

To add new document to collection:

```
import { Firestore, collection , doc , setDoc } from
'@angular/fire/firestore';

constructor(private firestore: Firestore) {}

addData() {
setDoc(doc(collection(this.firestore, "users")), {
      name: "marina",
      age: "21",
      email: "marina@gmail.com"
})}
```

Firebase

Update already exist document in collection:

```
import { Firestore, collection , doc , setDoc } from
'@angular/fire/firestore';

constructor(private firestore: Firestore) {}

addData() {
setDoc(doc(this.firestore, "users" ,"xbxecL5cqCbn4RSjSguL"), {
      name: "Ahmed",
}, { merge: true });
}
```

Firebase

To Delete document to collection:

```
import { Firestore, doc , deleteDoc } from
'@angular/fire/firestore';

constructor(private firestore: Firestore) {}

deleteData() {
 // note : users is the collection name and 4q5gCHqJ8EMfd52CMGoq
is the id of the document you want to delete.
 deleteDoc(doc(this.firestore, "users", "4q5gCHqJ8EMfd52CMGoq"));
}
```

Thank you

Lab

Task 1 : For the products app

- Create a new interceptor to show loader when request fires and hide loader when request resolve

Task 2 : Create a new application and link it with firebase using the same steps mentioned in the lecture to :

- Create a new collection called products
- In each collection should contain a product with : name, quantity,price, category and description
- Get products list from firestore, get single product , update existing product , add new product and delete. [even get single collection]