Hajek Valentin TP8-Angular info2

Dans ce didacticiel, onallez ajouter les fonctionnalités de persistance des données suivantes avec l'aide du HttpClient d'Angular.

- Le HeroService obtient des données de héros grâce à des requêtes HTTP.

- Les utilisateurs peuvent ajouter, modifier et supprimer des héros et enregistrer ces changements avec HTTP.

- Les utilisateurs peuvent rechercher des héros par leur nom.

[HttpClient](https://angular.io/api/common/http/HttpClient) est le mécanisme utilisé par Angular pour communiquer avec un serveur distant via HTTP.

import { HttpClientModule } from '@angular/common/http';

2)

On installe le package API web :

npm install angular-in-memory-web-api –save

Dans le ng module:

La méthode de configuration [forRoot()](https://angular.io/api/router/RouterModule" \l "forRoot)prend une InMemoryDataServiceclasse qui prépare la base de données en mémoire.

HttpClientModule,

HttpClientInMemoryWebApiModule.forRoot(

InMemoryDataService, { dataEncapsulation: false }

Dans src/app

ng generate service InMemoryData

puis on rempli la classe in-memory-data.service.ts

On peut ensuite supprimer mock-heroes.ts qui est remplacer par la classe in-memory-data.service.ts

On injecte [HttpClient](https://angular.io/api/common/http/HttpClient)dans le constructeur dans une propriété privée appelée [http](https://angular.io/api/common/http).

constructor(

private http: HttpClient,

private messageService: MessageService) {}

Puis on remplace le constructeur de message service par : (on injecte le message service)

private log(message: string) {

this.messageService.add(`HeroService: ${message}`);

}

Url pour accéder à l’api web

private heroesUrl = 'api/heroes';

On convertit la méthode getheroes() pour utiliser http client

/\*\* GET heroes from the server \*/

getHeroes (): Observable<Hero[]> {

return this.http.get<Hero[]>(this.heroesUrl)

}

Permet de renvoyer une réponse unique

-Pour capturer les erreurs :

On importe le catch Erreur :

import { catchError, map, tap } from 'rxjs/operators';

Le catchError() est l’opérateur interceptant un Observable **qui a échoué** . Il transmet à l'erreur un gestionnaire d'erreur qui peut en faire ce qu’il en veut

Le [handleError()](https://angular.io/api/core/ErrorHandler" \l "handleError) signale l'erreur, puis renvoie un résultat inoffensif pour que l'application continue à fonctionner.

On initialise donc une fonction de gestion d’erreur là où handleError va envoyer avec le code fournis :

@param operation - name of the operation that failed

\* @param result - optional value to return as the observable result

\*/

private handleError<T> (operation = 'operation', result?: T) {

return (error: any): Observable<T> => {

// TODO: send the error to remote logging infrastructure

console.error(error); // log to console instead

// TODO: better job of transforming error for user consumption

this.log(`${operation} failed: ${error.message}`);

// Let the app keep running by returning an empty result.

return of(result as T);

};

}

Dans la méthode get heroes on rajoute tap permettant d’enregistrer l’opération :

tap(\_ => this.log('fetched heroes')),