ANGULAR 2 FUNDAMENTALS

December 13, 2017

PART 11

USING AGULAR HTTP SERVICE

- HTTP SERVICE TO FETCH DATA
- There is a @angular/http service
- Open app.module.ts and import HttpClientModule

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

- Add in imports portion:

```
imports: [
  routes,
  BrowserModule,
  HttpClientModule,
```

MAKE A SIMPLE GET REQUEST

- Open image.service.ts and define constructor:

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

  - Remember to import our HttpClient in our image service

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

- We don’t need the image array anymore in ImageService so remove it:

```
images: Image[] = [
new Image('First image', 'First image description', 'https://angularbooks.com/img/angu
lar4/img1.jpg', 'https://angularbooks.com/img/angular4/img1-l.jpg'),
new Image('Second image', 'Second image description', 'https://angularbooks.com/img/an
gular4/img2.jpg', 'https://angularbooks.com/img/angular4/img2-l.jpg'),
new Image('Third image', 'Third image description', 'https://angularbooks.com/img/angu
lar4/img3.jpg', 'https://angularbooks.com/img/angular4/img3-l.jpg'),
new Image('Fourth image', 'Fourth image description', 'https://angularbooks.com/img/an
gular4/img4.jpg', 'https://angularbooks.com/img/angular4/img4-l.jpg'),
```

MODIFY GETIMAGES() FUNCTION

- Make our first http request

```
getImages(): Observable<any> {
  return this.http.get('http://angularbook.app/api/v1/images')
    .map((response) => response);
}
```

- Don’t forget to import

```
import 'rxjs/add/operator/map';
import {Observable} from 'rxjs/Observable';
```

- Open image-list.component.ts and find:

```
images: Image[] = [];
```

To

```
images: Observable<Image[]>;
```

MORE ON OBSERVABLES

- Import observable component also:

```
import {Observable} from 'rxjs/Observable';
```

- Open image-list.component.html and find:

```
<a2g-image *ngFor="let image of images" [image]="image" (click)="onSelect(image)"></a2
g-image>
```

- Modify to:

```
<ng-image *ngFor="let image of images | async" [image]="image" (click)="onSelect(image)
"></ng-image>
```

- Visit app it should be working now:

IS IT WORKING?

- Congrats data has been communicated:
- But what is observable and rxjs?

OBSERVABLES

- This is an era of asynchronous programming (async)
- When sending request, new techniques are used
- We don’t have to wait for our request to finish to move to another task
- Observable is cancellable.
- To use observable we need to use an external library because es6 doesn’t support it by default
- RxJS provides some observable operators that we can use to transform, filter, compose and query streams of data.

HOW OUR HTTP REQUEST WORKS

- To use observable we have to import rxjs first

```
import 'rxjs/Rx';
```

- OR Import map operator

```
import 'rxjs/add/operator/map';
```

- When we use http.get it will return an observable

```
getImages(): Observable<any> {
  return this.http.get('http://angularbook.app/api/v1/images')
}
```

- To render an observable we need to use a specific async pipe in template

```
<ng-image *ngFor="let image of images | async" [image]="image" (click)="onSelect(image)
"></ng-image>
```

JUST A NOTE

- (response) => response) is called a fat arrow function in es6 thats equivalent to:

```
function(response) {
    return response;
}
```

MORE ON TYPES

- Optionally if we know that our data is of type image we can use

```
getImages(): Observable<any> {
  return this.http.get('http://angularbook.app/api/v1/images')
    .map((response: HttpResponse<any>) => response as Image[]);
}
```

- If you don’t want to import observables

```
getImages(): any {
  return this.http.get('http://angularbook.app/api/v1/images')
    .map((response: HttpResponse<any>) => response as Image[]);
}
```

END OF PART 11