Highlights:

* Understanding communication with server
* Exploring HttpClient class
* Handling response through observables

**Demo Steps:**

**Problem Statement:**Fetching books data using HttpClient class. Output is as shown below:



1. Create **BookComponent**by using the following CLI command

1. ng generate component book

2. Create a file with name**book.ts** under book folder and add the following code

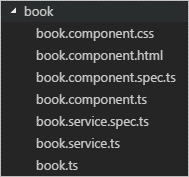
1. export class Book {
2. id: number=0;
3. name: string="";
4. }

**Line 1-4:** Create a Book class with two properties id and name to store book id and book name

3. Create a service called **BookService** using the following CLI command

1. ng generate service book

This will create two files called book.service.ts and book.service.spec.ts as shown below



**Note:** Add **BookService**to providers in app.module.ts

4. In the example used for custom services concept, add HttpClientModule to the **app.module.ts** to make use of Http class.

1. import { BrowserModule } from '@angular/platform-browser';
2. import { NgModule } from '@angular/core';
3. import { HttpClientModule } from '@angular/common/http';
4. import { AppComponent } from './app.component';
5. import { BookComponent } from './book/book.component';
6. import { BookService } from './book/book.service';
7. @NgModule({
8. declarations: [
9. AppComponent,
10. BookComponent
11. ],
12. imports: [
13. BrowserModule,
14. HttpClientModule
15. ],
16. providers: [BookService],
17. bootstrap: [AppComponent]
18. })
19. export class AppModule { }

**Line 6:** Imports BookService class into root module class

**Line 16:** Adds HttpClientModule to the imports property.

**Line 18:** Adds BookService to the providers property to make it injectable into the components of the root module

5. Create **books.json**under assets folder and move books data to it

1. [
2. { "id": 1, "name": "HTML 5" },
3. { "id": 2, "name": "CSS 3" },
4. { "id": 3, "name": "Java Script" },
5. { "id": 4, "name": "Ajax Programming" },
6. { "id": 5, "name": "jQuery" },
7. { "id": 6, "name": "Mastering Node.js" },
8. { "id": 7, "name": "Angular JS 1.x" },
9. { "id": 8, "name": "ng-book 2" },
10. { "id": 9, "name": "Backbone JS" },
11. { "id": 10, "name": "Yeoman" }
12. ]

6. Add the following code in **book.service.ts** file

1. import { Injectable } from '@angular/core';
2. import { HttpClient } from '@angular/common/http';
3. import { Observable } from 'rxjs';
4. import { Book } from './book';
5. @Injectable()
6. export class BookService {
7. private booksUrl = './assets/books.json';
8. constructor(private http: HttpClient) { }
9. getBooks(): Observable<Book[]> {
10. return this.http.get<Book[]>(
11. this.booksUrl
12. );
13. }
15. }

**Line 2:** Imports HttpClient class from @angular/common/http module.

**Line 3:** Imports Observable class from the rxjs module

**Line 6:** @Injectable() decorator makes the class as service which can be injected into components of an application

**Line 9:** Stores the JSON file path in a variable called booksUrl

**Line 11:** Injects HttpClient class into a service class

**Line 13-16:** getBooks() method returns Books data as a observable

**Line 14-15:** Makes an asynchronous call (ajax call) by using get() method of HttpClient class to the json file and fetches the data. HttpClient receives the JSON response as of **Book[]**object.

The http.get<Book[]>() method will automatically convert the received JSON data from the file to an array of Book type.

**Note**: The JSON object array data in the file should have the same key names of object literals as mentioned in the Book class imported from './book'.

7. Write the code given below in**book.component.ts**

1. import { Component, OnInit } from '@angular/core';
2. import { Book } from './book';
3. import { BookService } from './book.service';
4. @Component({
5. selector: 'app-book',
6. templateUrl: './book.component.html',
7. styleUrls: ['./book.component.css']
8. })
9. export class BookComponent implements OnInit {
10. books!: Book[];
11. errorMessage: string="";
12. constructor(private bookService: BookService) { }
13. getBooks() {
14. this.bookService.getBooks().subscribe(
15. books => this.books = books,
16. error => this.errorMessage = <any>error);
17. }
18. ngOnInit() {
19. this.getBooks();
20. }
21. }

**Line 3:** Imports BookService class into a component

**Line 15:** Inject the  BookService class into the component class through the constructor

**Line 18-20:** Invokes the service class method getBooks() which makes an http call to books.json file and the response is returned. Once the response is returned, observable has a subscribe method, which has a success callback and failure callback.

8. Write the code given below in **book.component.html**

1. <h2>My Books</h2>
2. <ul class="books">
3. <li \*ngFor="let book of books">
4. <span class="badge">{{book.id}}</span> {{book.name}}
5. </li>
6. </ul>
7. <div class="error" \*ngIf="errorMessage">{{errorMessage}}</div>

**Line 2-6:** Display book details

**Line 8:** Displays error messages when http get operation fails.

9. Add the following code in **book.component.css** which has styles for books

1. .books {
2. margin: 0 0 2em 0;
3. list-style-type: none;
4. padding: 0;
5. width: 15em;
6. }
7. .books li {
8. cursor: pointer;
9. position: relative;
10. left: 0;
11. background-color: *#EEE;*
12. margin: .5em;
13. padding: .3em 0;
14. height: 1.6em;
15. border-radius: 4px;
16. }
17. .books li:hover {
18. color: *#607D8B;*
19. background-color: *#DDD;*
20. left: .1em;
21. }
22. .books .badge {
23. display: inline-block;
24. font-size: small;
25. color: white;
26. padding: 0.8em 0.7em 0 0.7em;
27. background-color: *#607D8B;*
28. line-height: 1em;
29. position: relative;
30. left: -1px;
31. top: -4px;
32. height: 1.8em;
33. margin-right: .8em;
34. border-radius: 4px 0 0 4px;
35. }

10. Save the files and run the application to check the output in the browser.