Basic HTTP Calls in Angular

Introduction to HTTP in Angular

- HTTP calls can be used to call out external APIs
- Angular comes with its own HTTP library to call out external APIs
- HTTP requests are asynchronous, because we don't want our page to freeze until the HTTP request returns from the external server
- There are three main approaches to deal with async code:
    i. Callback
    ii. Promises
    iii. Observables

Basic usage of HTTP in Angular

- HTTP has been split into a separate module in Angular.
- This means that to use it you need to import constants from @angualr/common/http

```
import {
    // The NgModule for using @angular/common/http
    HttpClientModule,

    // the class constants
    HttpClient
} from '@angular/common/http';
```

Basic Request

- The first thing we're going to do is make a simple GET request to the jsonplaceholder API.
- What we need is:
    i. Have a button that calls makeRequest
    ii. makeReuqest will call the http library to perform a GET request on our API
    iii. When the request returns, we'll update this.data with the results of the data, which will be rendered in the view

SimpleHttpComponent Component Definition

```
import {Component, OnInit} from '@angular/core';
import {HttpClient} from '@angular/common/http';

@Component({
    selector: 'app-simple-http',
    templateUrl: './simple-http.component.html'
})
export class SimpleHttpComponent implements OnInit {
    data: Object;
    loading: boolean;

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

Building the SimpleHttpComponent template

```
<h2>Basic Request</h2>
<button type="button" (click)="makeRequest()">Make Request</button>
<div *ngIf="loading">loading...</div>
<pre>{{data | json}}</pre>
```

Building the SimpleHttpComponent Controller

```
import {Component, OnInit} from '@angular/core';
import {HttpClient} from  '@angular/common/http';

@Component({
    selector: 'app-simple-http',
    templateUrl: './simple-http.component.html'
})
export class SimpleHttpComponent implements OnInit {
    data: Object;
    loading: boolean;

    constructor(private http: HttpClient){}

    ngOnInit() {}

    makeRequest(): void {
        this.loading = true;
        this.http
            .get('https://jsonplaceholder.typicode.com/posts/1')
            .subscribe(data => {
                this.data = data;
                this.loading = false;
            });
    }
}
```

Making a simple HTTP POST Request

```
makePost(): void {
    this.loading = true;
    this.http.post('http://jsonplaceholder.typicode.com/posts',
    JSON.stringify({
        body: 'bar',
        title: 'foo',
        userId: 1
    }))
    .subscribe(data => {
        this.data = data;
        this.loading = false;
    });
}
```

Making a PUT/PATCH/DELETE/HEAD Request

```
makeDelete(): void {
    this.loading = true;
    this.http.delete('https://jsonplaceholder.typicode.com/posts/1')
    .subscribe(data => {
        this.data = data;
        this.loading = false;
    });
}
```

Custom HTTP Headers

```
makeHeaders(): void {
    const headers: HttpHeaders = new HttpHeaders({
        'X-API-TOKEN': 'ng-book'
    });

    const req = new HttpRequest(
        'GET',
        'https://jsonplaceholder.typicode.com/posts/1',
        {
            headers: headers
        }
    );

    this.http.request(req).subscribe(data => {
        this.data = data['body'];
    })
}
```

Summary

- @angular/common/http is flexible and suitable for a wide variety of APIs
- In general, you can make GET and POST request in an easy and flexible way