Interceptors

Often we want to enforce or apply behavior when receiving or sending HTTP requests within our application. Interceptors are a unique type of Angular Service that we can implement. **Interceptors allow us to intercept incoming or outgoing HTTP requests using the HttpClient. By intercepting the HTTP request, we can modify or change the value of the request.**

**Step1: provide INTERCEPTOR service in app module**

## // app.module.ts

**import { HttpClientModule, HTTP\_INTERCEPTORS } from '@angular/common/http';**

import { MyInterceptor } from './my.interceptor';

import { AppComponent } from './app.component';

@NgModule({

imports: [BrowserModule, HttpClientModule],

declarations: [AppComponent],

bootstrap: [AppComponent],

providers: [

**{ provide: HTTP\_INTERCEPTORS, useClass: MyInterceptor, multi: true }**

]

})

export class AppModule { }

**Step2: Create new interceptor service class file**

## e.g:- // auth.interceptor.ts

import { HttpInterceptor, HttpRequest, HttpHandler } from '@angular/common/http';

export class AuthInterceptorService implements **HttpInterceptor** {

**intercept**(**req: HttpRequest<any>, next: HttpHandler**) {

    console.log('Request is on its way');

    return next.handle(req);

  }

}

**Step3: To modify request in Interceptor, CLONE the request**

* You cannot modify original request, You have to clone the request to do any modifications in request
* You change the request and then you return/forward modified request

export class AuthInterceptorService implements HttpInterceptor {

  intercept(req: HttpRequest<any>, next: HttpHandler) {

**const modifiedRequest = req.clone({**

**headers: req.headers.append('Auth', 'xyz')**

**});**

    return next.handle(modifiedRequest);

  }

}

**Step4: Handling response in INTERCEPTOR**

**Eg:- Logging interceptor**

import {

  HttpInterceptor,

  HttpRequest,

  HttpHandler,

  HttpEventType

} from '@angular/common/http';

import { tap } from 'rxjs/operators';

export class LoggingInterceptorService implements HttpInterceptor {

  intercept(req: HttpRequest<any>, next: HttpHandler) {

    console.log('Outgoing request');

    return next.handle(req**).pipe(**

**tap(event => {**

**if (event.type === HttpEventType.Response) {**

**console.log('Incoming response');**

**console.log(event.body);**

**}**

**})**

    );

  }

}

* you can add multiple interceptors like authentication interceptor, logging interceptor etc..
* Interceptor are executed in a same order as provided in app.module
* E.g:- LoggingInterceptorService interceptor will run before AuthInterceptorService interceptor.

  providers: [

    {

      provide: HTTP\_INTERCEPTORS,

      useClass: LoggingInterceptorService,

      multi: true

    },

    {

      provide: HTTP\_INTERCEPTORS,

      useClass: AuthInterceptorService,

      multi: true

    }

  ],