Steps to Do an External API Call from Angular

1. In the service/program typescript file:

a.
```
import { HttpClient   } from '@angular/common/http';
import { HttpHeaders  } from '@angular/common/http';
import { lastValueFrom } from 'rxjs';
```

b. // Set up a string to hold the root URL/endpoint for the server
```
private theServerURL : string = "your-server-endpoint/root"
```

c. // Have the constructor receive a Dependency Injected **HttpClient** Object
```
constructor(private theServer:HttpClient) {}
```

d. // Make the Http Call from an async method that returns a **Promise<any>**
// and use the server URL variable defined above
// BE SURE YOU ADD WHATEVER IS REQUIRED TO THE BASE SERVER URL YOU DEFINED
```
async getMoviesList()  : Promise<any[]> { }
```

e. // Make the call to the external API and store in our local data source (array)
// `lastValueFrom()` converts the Observable object returned from the http **get()** to a Promise

// To do an HTTP GET:
```
const result : any[] =
      await lastValueFrom(this.theServer.get<any[]>(this.theServerURL))
```

// To do an HTTP DELETE
```
const result : any[] =
      await lastValueFrom(this.theServer.delete(this.theServerURL))
```

// To do an HTTP POST/PUT:

// Use `new HttpHeaders` to create a JSON object with the attributes you want to send
```
const headers = new HttpHeaders ({
                'Content-Type' : 'application/json'
                 });
```

// Use **http.post(API-URL, data-to-send, {header-object})**
```
lastValueFrom(this.theServer.post(this.theServerURL+"/create", newMovie
                                  ,{headers}));
```

// Use **http.put(API-URL, data-to-send, {header-object})**
```
lastValueFrom(this.theServer.put(this.theServerURL+"/update", newMovie
                                  ,{headers}));
```

Consider defining any data used with the API call as `any` *(to avoid possible errors)*

(continued on next page)

Steps to Do an External API Call from Angular

2. In the **app.config.ts** file:

   a. `import { provideHttpClient } from '@angular/common/http';`

   b. // Add "provideHttpClient" as a provider

   ```
   providers: [provideRouter(routes), provideHttpClient()]
   ```

3. In the **component that calls the service** to get the data from the API:

   a. Import the service/code that does the API call:

   ```
   import { MoviesService } from '../movies.service-api';
   ```

   b. Add the service to the component constructor to be Dependency Injected:

   ```
   constructor(private movieService  : MoviesService) {}
   ```

   c. // Add an async **ngOnInit()** method to call the service methods that calls API:

   ```
   async ngOnInit()
   {
     this.moviesList = await this.movieService.getMoviesList();
   }
   ```

   d. // Add **implements OnInit** to export:

   ```
   export class MovieListComponent implements OnInit {
   ```

The following steps are required if you are writing the Server side code as well to avoid CORS errors:

Java / Spring Boot

Add the following before each controller class:

```
@CrossOrigin(origins = "*")
```

C#/ASP .NET

Add the following to the **program.cs**:

1. // With all the other **builder.Service** statements
   // Be sure you have the correct server URL for the **WithOrigins** parameter

```
builder.Services.AddCors(options =>
{
  options.AddDefaultPolicy(policy =>
   {
    policy.WithOrigins("https://localhost:7223").AllowAnyMethod()
                                                .AllowAnyHeader()
                                                .AllowAnyOrigin();

   });
});
```

2. // With all the other **app.xxxxxx** statements

```
app.UseCors();
```