

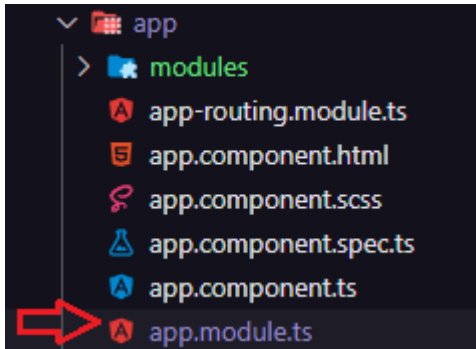
Requisições HTTP

Utilizamos as requisições HTTP para realizar interações com um API.

API utilizada: <https://643700b13e4d2b4a12e1541c.mockapi.io/testeshttp>

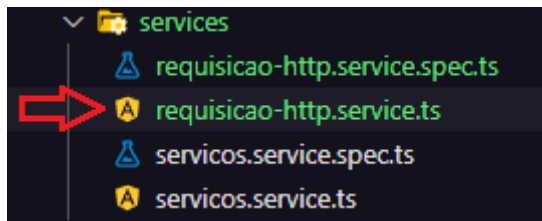
Para utiliza:

- 1) Para iniciar a utilização de requisições HTTP, devemos importar o módulo "HttpClientModule" da biblioteca do angular em nosso arquivo "app.module.ts".



```
src > app > app.module.ts > ...
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3  import { CoreModule } from './modules/core/core.module';
4
5  import { AppRoutingModule } from './app-routing.module';
6  import { AppComponent } from './app.component';
7  import { BrowserAnimationsModule } from '@angular/platform-br
8  import { HttpClientModule } from '@angular/common/http';
9
10 @NgModule({
11   declarations: [
12     AppComponent
13   ],
14   imports: [
15     BrowserModule,
16     AppRoutingModule,
17     CoreModule,
18     BrowserAnimationsModule,
19     HttpClientModule
20   ],
21   providers: [],
22   bootstrap: [AppComponent]
23 })
24 export class AppModule { }
25
```

2) Com o módulo importado podemos criar um serviço onde serão realizadas as requisições. Para este exemplo criamos um serviço chamado de “requisição-http”.



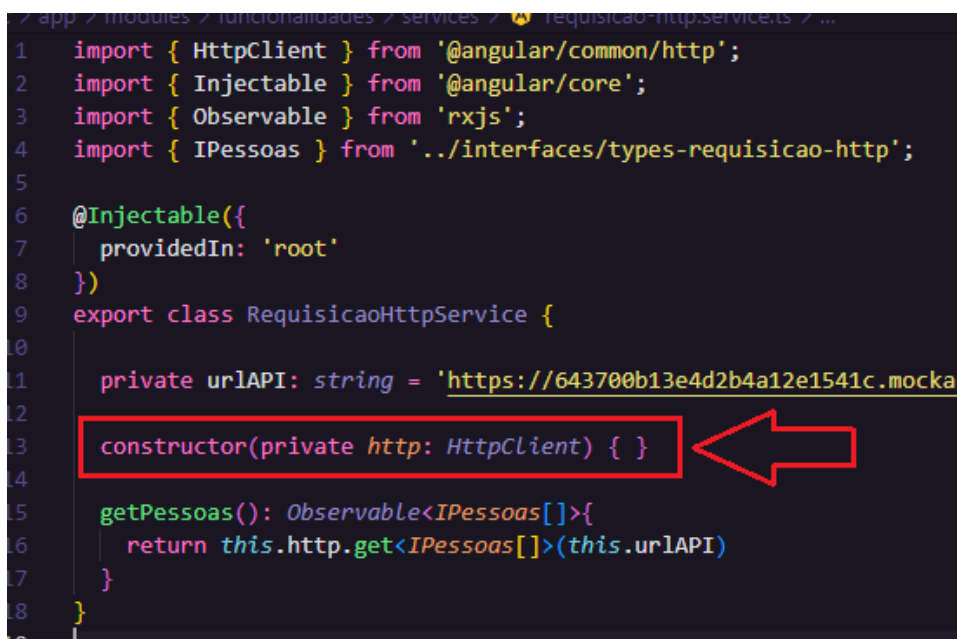
3) Para iniciar as configurações do serviço, primeiro precisamos definir qual a URL da API que será utilizada. Para isso, criamos uma variável privada e atribuímos a URL.

A screenshot of the 'requisicao-http.service.ts' file in an IDE. The code is as follows:

```
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { Observable } from 'rxjs';
4 import { IPessoas } from '../interfaces/types-requisicao-http';
5
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class RequisicaoHttpService {
10
11   private urlAPI: string = 'https://643700b13e4d2b4a12e1541c.mockapi.io/testeshttp';
12
13   constructor(private http: HttpClient) { }
14
15   getPessoas(): Observable<IPessoas[]>{
16     return this.http.get<IPessoas[]>(this.urlAPI)
17   }
18 }
19
```

A red box highlights the line `private urlAPI: string = 'https://643700b13e4d2b4a12e1541c.mockapi.io/testeshttp';` and a red arrow points to it.

4) Após isso, precisamos definir o “HttpClient” no construtor da classe do serviço para que possamos ter acesso aos métodos de requisição HTTP.

A screenshot of the 'requisicao-http.service.ts' file in an IDE, showing the same code as the previous image. A red box highlights the constructor parameter `http: HttpClient` in the line `constructor(private http: HttpClient) { }`, and a red arrow points to it.

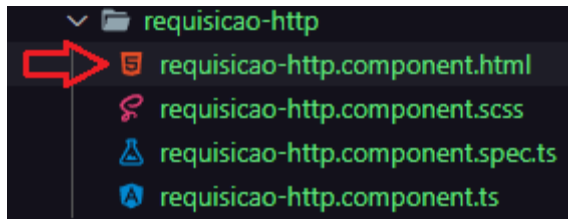
5) Com o HttpClient criado, podemos realizar a configuração do método que irá realizar a requisição para a API. Neste exemplo vamos fazer um “get” nas informações da API. É importante lembrar que este método será um “Observable”.

```
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { Observable } from 'rxjs';
4 import { IPessoas } from '../interfaces/types-requisicao-http';
5
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class RequisicaoHttpService {
10
11   private urlAPI: string = 'https://643700b13e4d2b4a12e1541c.mockapi.
12
13   constructor(private http: HttpClient) { }
14
15   getPessoas(): Observable<IPessoas[]>{
16     return this.http.get<IPessoas[]>(this.urlAPI)
17   }
18 }
19
```

Neste exemplo o método “getPessoas” está retornando uma requisição “get” no “HttpClient” e passando como parâmetro a “urlAPI”.

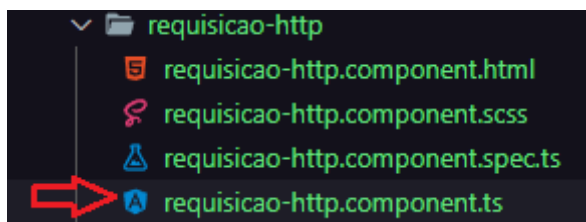
OBS.: IPessoas é referente a interface dos dados.

6) Agora podemos preparar o template no arquivo “.html” do componente para renderizar os dados que serão importados da API. Neste exemplo, iremos iterar no array “pessoas” e para cada elemento do array será apresentado um card.



```
Go to component
1 <div class="div-componente">
2   <h2 class="title-componente">Requisição HTTP</h2>
3   <div class="componente-container">
4     <p class="descricao-componente">
5       Utilizamos as requisições HTTP para realizar interações com um API. No exemplo a seguir es
6       no botão "Buscar". Desta forma será renderizado um card para cada resultado retornado pela
7     </p>
8   </div>
9   <div class="componente-exemplo">
10    <button class="botao-exemplo" (click)="buscarPessoas()">Buscar</button>
11    <div class="exemplo-cards-container">
12      <div class="card-exemplo" *ngFor="let pessoa of pessoas">
13        
14        <p class="pessoa-nome">Nome: {{pessoa.nome}}</p>
15      </div>
16    </div>
17  </div>
18  <div class="componente-buttons-container">
19    <app-arquivo-pdf [endereco]="endereco"></app-arquivo-pdf>
20    <app-repositorio [github]="github"></app-repositorio>
21  </div>
22 </div>
```

7) As informações que estão dentro do array “pessoas” serão as que importaremos da requisição para API. Para isso, vamos configurar o arquivo “.ts” do componente.



8) Primeiramente, vamos criar o array “pessoas” e atribuir um valor vazio ao mesmo.

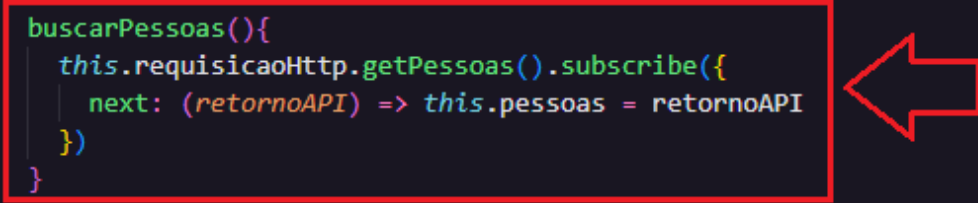
```
1  import { Component } from '@angular/core';
2  import { RequisicaoHttpService } from '../../services/requisicao-http.service';
3  import { IPessoas } from '../../interfaces/types-requisicao-http';
4
5  @Component({
6    selector: 'app-requisicao-http',
7    templateUrl: './requisicao-http.component.html',
8    styleUrls: ['./requisicao-http.component.scss']
9  })
10 export class RequisicaoHttpComponent {
11
12     endereco: string = '../../assets/RequisiçãoHttp.pdf'
13     github: string = 'https://github.com/LuanSOliveira/Testes-Angular/tree/main'
14
15     pessoas: IPessoas[] = []
16
17     constructor (private requisicaoHttp: RequisicaoHttpService){}
18
19     buscarPessoas(){
20         this.requisicaoHttp.getPessoas().subscribe({
21             next: (retornoAPI) => this.pessoas = retornoAPI
22         })
23     }
24 }
```

9) Agora precisamos importar no construtor o serviço que foi criado onde realizamos as configurações da requisição HTTP.

```
1  import { Component } from '@angular/core';
2  import { RequisicaoHttpService } from '../../services/requisicao-http.service';
3  import { IPessoas } from '../../interfaces/types-requisicao-http';
4
5  @Component({
6    selector: 'app-requisicao-http',
7    templateUrl: './requisicao-http.component.html',
8    styleUrls: ['./requisicao-http.component.scss']
9  })
10 export class RequisicaoHttpComponent {
11
12     endereco: string = '../../assets/RequisiçãoHttp.pdf'
13     github: string = 'https://github.com/LuanSOliveira/Testes-Angular/tree/main/src/app'
14
15     pessoas: IPessoas[] = []
16
17     constructor (private requisicaoHttp: RequisicaoHttpService){}
18
19     buscarPessoas(){
20         this.requisicaoHttp.getPessoas().subscribe({
21             next: (retornoAPI) => this.pessoas = retornoAPI
22         })
23     }
24 }
```

10) O próximo passo é criar um método que será executado ao clicar um botão no arquivo “.html” do componente para que os dados da API sejam carregados no array “pessoas”. Para este exemplo vamos criar o método “buscarPessoas()”.

```
1  import { Component } from '@angular/core';
2  import { RequisicaoHttpService } from '../../services/requisicao-http';
3  import { IPessoas } from '../../interfaces/types-requisicao-http';
4
5  @Component({
6    selector: 'app-requisicao-http',
7    templateUrl: './requisicao-http.component.html',
8    styleUrls: ['./requisicao-http.component.scss']
9  })
10 export class RequisicaoHttpComponent {
11
12     endereco: string = '../../../assets/RequisiçãoHttp.pdf'
13     github: string = 'https://github.com/LuanSOliveira/Testes-Angular/t
14
15     pessoas: IPessoas[] = []
16
17     constructor (private requisicaoHttp: RequisicaoHttpService){}
18
19     buscarPessoas(){
20         this.requisicaoHttp.getPessoas().subscribe({
21             next: (retornoAPI) => this.pessoas = retornoAPI
22         })
23     }
24 }
25
```



Na configuração deste método vamos executar o seguinte processo:

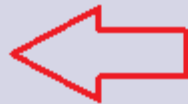
- Realizar a chamada do método “getPessoas” criado no serviço.
- Realizar a inscrição no “Observable” do serviço criado utilizando o método “subscribe” e passar, como parâmetro, um objeto com a propriedade “next”, onde teremos uma função que irá atribuir os valores de retorno da API ao array “pessoas”.

11) Desta forma, ao clicar no botão “Buscar”, serão renderizados no navegador os cards com as informações importadas da API.

Requisição HTTP

Utilizamos as requisições HTTP para re
Desta forma será renderizado um card p

Buscar



Como Utilizar

Requisição HTTP

Utilizamos as requisições HTTP para realizar int
Desta forma será renderizado um card para cad

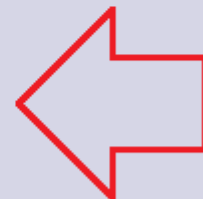
Buscar



Nome: João



Nome: Maria



Como Utilizar