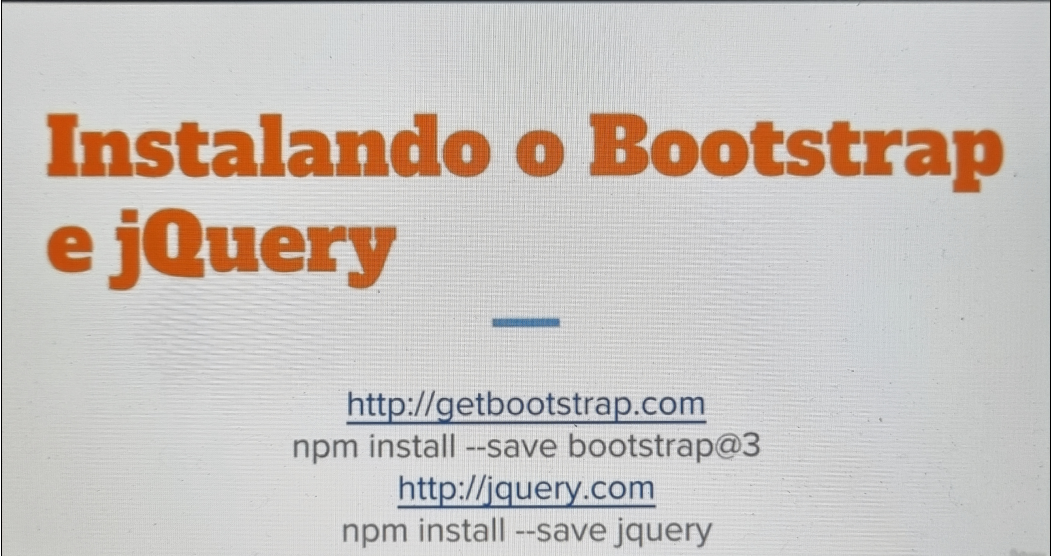
Conversor de moedas

* Aprenderá como criar filtros (pipes).
* Como utilizar o Event Emitter.
* Criar modais com o bootstrap.
* Como fazer requisição HTTP.
* Como criar uma diretiva de validação.
* Entender o que são observers.



**Nota importante!!!**

Nesse projeto utilizamos uma API para a conversão de moedas, a https://fixer.io, que mudou a forma como acessamos os dados, portanto será necessário modificar a URL da API no código do serviço que será criado a seguir para que ela continue funcionando.

A nova API requer uma chave de acesso pessoal. Para facilitar a implementação eu criei uma pessoal e compartilharei com você, assim não haverá a necessidade de você criar a sua.

\* Essa chave tem uma limitação que somente permite converter moeda de Euros (EUR) para outras moedas, então ao executar a aplicação, tenha certeza de selecionar no campo "De" sempre a moeda EUR, pois as demais não funcionarão por limitação da conta gratuita que eles disponibilizam.

\*\* Caso você deseje utilizar essa API em outros projetos, peço que crie a sua própria chave de autenticação no website citado acima, pois essa também possui uma limitação de 1000 requisições mensais, que deverá ser compartilha somente aqui pelos alunos.

\*\*\* Caso a chave que demonstro nos vídeos não funcione por limites de utilização, tente utilizar a seguinte que criei com outro email: 0138904fb6412bdba3aafede67e6c015

A versão 13 do Angular também traz uma nova API para requisições HTTP, que é a HttpClient (introduzida no Angular 4), que veio para substituir a antiga Http.

A seguir segue o código completo do serviço atualizado para a versão 12 do Angular, assim como para a nova API do fixer.io.

Você pode acompanhar normalmente as duas próximas aulas sobre a criação do serviço, substituindo o código antigo pelo novo que segue abaixo:

1. import { Injectable } from '@angular/core';
2. // Importamos o HttpClient ao invés do Http para o Angular 7
3. //import { Http } from '@angular/http';
4. import { HttpClient } from '@angular/common/http';
5. import { Observable } from 'rxjs/Observable';
6. import {
7. Conversao,
8. ConversaoResponse
9. } from '../models';
10. @Injectable()
11. export class ConversorService {
12. // Nova url do fixer.io, que adiciona o parâmetro access\_key, que é a chave de autenticação
13. //private readonly BASE\_URL = "http://api.fixer.io/latest";
14. private readonly BASE\_URL = "http://data.fixer.io/api/latest?access\_key=eba7130a5b2d720ce43eb5fcddd47cc3";
15. constructor(private http: HttpClient) {}
16. /\*\*
17. \* Realiza a chamada para a API de conversão de moedas.
18. \*
19. \* @param Conversao conversao
20. \* @return Observable<ConversaoResponse>
21. \*/
22. converter(conversao: Conversao): Observable<any> {
23. // Na linha abaixo altere a '?' por '&'
24. let params = `&base=${conversao.moedaDe}&symbols=${conversao.moedaPara}`;
25. return this.http
26. .get(this.BASE\_URL + params);
27. // No Angular 6 as duas próximas linha não são mais necessárias
28. //.map(response => response.json() as ConversaoResponse)
29. //.catch(error => Observable.throw(error));
30. }
31. /\*\*
32. \* Retorna a cotação para dado uma response.
33. \*
34. \* @param ConversaoResponse conversaoResponse
35. \* @param Conversao conversao
36. \* @return number
37. \*/
38. cotacaoPara(conversaoResponse: ConversaoResponse,
39. conversao: Conversao): number {
40. if (conversaoResponse === undefined) {
41. return 0;
42. }
43. return conversaoResponse.rates[conversao.moedaPara];
44. }
45. /\*\*
46. \* Retorna a cotação de dado uma response.
47. \*
48. \* @param ConversaoResponse conversaoResponse
49. \* @param Conversao conversao
50. \* @return string
51. \*/
52. cotacaoDe(conversaoResponse: ConversaoResponse,
53. conversao: Conversao): string {
54. if (conversaoResponse === undefined) {
55. return '0';
56. }
57. return (1 / conversaoResponse.rates[conversao.moedaPara])
58. .toFixed(4);
59. }
60. /\*\*
61. \* Retorna a data da cotação dado uma response.
62. \*
63. \* @param ConversaoResponse conversaoResponse
64. \* @return string
65. \*/
66. dataCotacao(conversaoResponse: ConversaoResponse): string {
67. if (conversaoResponse === undefined) {
68. return '';
69. }
70. return conversaoResponse.date;
71. }
72. }

O HttpClient citado acima pertence a outro módulo no Angular 12, que deverá ser modificado também no arquivo 'app.module.ts', conforme código a seguir:

1. import { BrowserModule } from '@angular/platform-browser';
2. import { NgModule } from '@angular/core';
3. import { FormsModule } from '@angular/forms';
4. // Importe o módulo HttpClientModule ao invés de HttpModule
5. //import { HttpModule } from '@angular/http';
6. import { HttpClientModule } from '@angular/common/http';
7. import { AppComponent } from './app.component';
8. import { ConversorModule } from './conversor';
9. @NgModule({
10. declarations: [
11. AppComponent
12. ],
13. imports: [
14. BrowserModule,
15. FormsModule,
16. // Modifique aqui a importação do módulo também
17. //HttpModule,
18. HttpClientModule,
19. ConversorModule
20. ],
21. providers: [],
22. bootstrap: [AppComponent]
23. })
24. export class AppModule { }

Com isso você terá o serviço atualizado e funcionando.