07

Isolando o acesso à API em um serviço

Transcrição

A lógica de acesso à nossa API esta fixa no método `importaDados` de `NegociacaoController`. O problema dessa abordagem é que se quisermos acessar a API em outro controller, teremos que repetir código. Podemos enviar isso isolando a lógica de importação na classe `NegociacaoService`.

```
// app/ts/services/NegociacaoService.ts

import { NegociacaoParcial, Negociacao } from '../models/index';

export class NegociacaoService {

    obterNegociacoes(handler: Function): Promise<Negociacao[]> {

        return fetch('http://localhost:8080/dados')
            .then(res => handler(res))
            .then(res => res.json())
            .then((dados: NegociacaoParcial[]) =>
                dados.map(dado => new Negociacao(new Date(), dado.vezes, dado.montante))
            )
            .catch(err => console.log(err));

    }
}
```

Um ponto a destacar é que nosso método recebe uma `handler`, isto é, a função que considerará ou não a operação válida. Não podemos deixar simplesmente fixo esse código no serviço, queremos dar flexibilidade para o programador pode aplicar outros critérios que não sejam `res.ok`.

Por fim, o retorno do método será a Promise, resultado de `fetch`. No entanto, `Promise` é um tipo genérico e precisamos indicar qual tipo estará disponível ao acessarmos seu retornado através de `then`. É por isso que usamos `Promise<Negociacao[]>`, pois através de `then` teremos acesso ao array de negociações.

Como de costumo, vamos criar um barrel para facilitar a importação do serviço:

```
// app/ts/services/index.ts
export * from './NegociacaoService';
```

Agora, vamos importar nosso serviço, inclusive adicionar como propriedade de `NegociacaoController` uma instância dessa classe para que possamos utilizá-la:

```
import { NegociacoesView, MensagemView } from '../views/index';
import { Negociacao, Negociacoes } from '../models/index';
import { domInject, throttle } from '../helpers/decorators/index';
import { NegociacaoParcial } from '../models/index';
import { NegociacaoService } from '../services/index';
```

```
export class NegociacaoController {

    // código anterior omitido

   // mais uma propriedade da classe!
    private _service = new NegociacaoService();

// código anterior omitido

    @throttle()
    importaDados() {

        function isOk(res: Response) {

            if(res.ok) {
                return res;
            } else {
                throw new Error(res.statusText);
            }
        }

        this._service
            .obterNegociacoes(isOk)
            .then(negociacoes => {
                negociacoes.forEach(negociacao =>
                    this._negociacoes.adiciona(negociacao));
                this._negociacoesView.update(this._negociacoes);
            });
    }
```

Excelente! Agora, se o acesso a API mudar, mudaremos em um único lugar que ele se refletirá em todos os lugares que nossa classe de serviço for utilizada. No entanto, nosso código deixa a desejar. O motivo? Veremos no próximo vídeo.