09

Deixando clara nossa intenção

Transcrição

Quanto tempo levaria para um novo programador compreender esse trecho de código?

```
// app/app.js

// código anterior omitido
.then(notas => notas
    .$flatMap(nota => nota.itens)
    .filter(item => item.codigo == '2143')
    .reduce((total, item) => total + item.valor, 0))
// código posterior omitido
```

Independente do tempo gasto, não ficaria mais claro se o código estivesse organizado dessa forma?

```
.then(sumItems)
```

Conseguimos isso facilmente extraindo o trecho de código para dentro da função `sumItems`:

```
import { handleStatus, log } from './utils/promise-helpers.js';
import './utils/array-helpers.js';

const sumItems = notas => notas
    .$flatMap(nota => nota.itens)
    .filter(item => item.codigo == '2143')
    .reduce((total, item) => total + item.valor, 0)

document
.querySelector('#myButton')
.onclick = () =>
    fetch('http://localhost:3000/notas')
    .then(handleStatus)
    .then(sumItems)
    .then(console.log)
    .catch(console.log);
```

Muito bom, mas se quisermos agora somar todos os itens de outra conta? Nossa função `sumItems` não está preparada para isso, mas podemos adequá-la com pouco esforço.

Nossa função retornará outra que espera receber um array. É essa função resultante que passaremos para o `then`. Através de `closure`, a função retornada lembrará do parâmetro recebido pela função mais externa.

Alterando nosso código:

```
// app/app.js
import { handleStatus, log } from './utils/promise-helpers.js';
```

```
import './utils/array-helpers.js';

const sumItems = code => notas =>
    notas.$flatMap(nota => nota.itens)
    .filter(item => item.codigo == code)
    .reduce((total, item) => total + item.valor, 0)

document
.querySelector('#myButton')
.onclick = () =>
    fetch('http://localhost:3000/notas')
    .then(handleStatus)
    .then(sumItems('2143'))
    .then(console.log)
    .catch(console.log);
```

Uma pequena alteração com grande impacto na legibilidade. Será que podemos fazer mais pelo nosso código? Sim, podemos, e é isso que nós veremos no próximo vídeo.