

Para saber mais: A interface Processor

Além das interfaces `Publisher`, `Subscription` e `Subscriber`, temos também a interface `Processor`. Ela serve como um filtro entre as chamadas dos *publishers* para seus *subscribers*. A estrutura da interface é mostrada abaixo:

```
public static interface Processor<T,R> extends Subscriber<T>, Publisher<R> {...}
```

Percebam que ela estende de `Subscriber` e `Publisher`, logo possui os mesmos métodos que estávamos acostumados a trabalhar. Em uma possível validação, pode-se utilizar a interface `Processor` para realizá-la de forma assíncrona. Isso significa que, se precisássemos validar o valor da nota ou data da compra, poderíamos utilizar o método `onNext(...)`:

```
public class NotaFiscalFiltro extends SubmissionPublisher implements Processor {

    // Métodos escondidos, porém utilizando a mesma implementação do NotaFiscalSubscriber

    @Override
    public void onNext(NF nf) {
        if(nf >= 0.0) {
            submit(nf);
        } else {
            System.out.println("Nota fiscal com o valor menor ou igual a zero");
        }
    }
}
```

No código acima, estamos falando que, se o valor da nota fiscal for maior que 0, chamaremos a `NotaFiscalSubscriber` e o mesmo fará o seu processamento, caso contrário, será informado uma mensagem e não será necessário o processamento. É importante ressaltar que haveria uma pequena mudança no `onComplete(...)` também, pois não queremos informar que as notas foram processadas nesse caso. Uma possível solução é chamar o método `close(...)` do publicador. A ideia de estender do `SubmissionPublisher` é que os detalhes do publicador já estão prontos, logo podemos reutilizá-los. Uma vez desenvolvido, podemos utilizar o filtro da seguinte maneira.

```
SubmissionPublisher<NF> publisher = new SubmissionPublisher<>(...);
NotaFiscalSubscriber subscriber = new NotaFiscalSubscriber (...);
NotaFiscalFiltro filtro = new NotaFiscalFiltro (...);
publisher.subscribe(filtro);
filtro.subscribe(subscriber);
```

O restante do código fica igual ao que já conhecemos, com a diferença apenas que agora temos um filtro para as validações as declaradas acima.