04

Refatorando a lógica SAX

Transcrição

Nosso código está funcionando, mas ainda podemos melhorá-lo, a começar pela visibilidade dos atributos que criamos em `LeitorXml.java`. Afinal, nada nos impede de, no método `main()`, executarmos `logica.produtos = null`, fazendo com que nosso retorno seja nulo. Pensando nisso, seria lógico impedirmos alterações da variável `produtos`.

Temos então um problema de encapsulamento. Relembrando conceitos de Orientação a Objetos, tornaremos nossos atributos privados (`private`), de modo que somente a classe `LeitorXml.java` os enxergue. Fazendo isso, precisaremos gerar o getter da lista `produtos`, que é a única variável que deverá ser acessada.

```
public class LeitorXml extends DefaultHandler{

    private List<Produto> produtos = new ArrayList<>();
    private StringBuilder conteudo;
    public List<Produto> getProdutos() {
        return produtos;
    }
//...
```

De volta ao método `main()`, usaremos o `getProdutos()` para recuperarmos as informações dessa variável.

```
public static void main(String[] args) throws Exception {
    XMLReader leitor = XMLReaderFactory.createXMLReader();
    LeitorXml logica = new LeitorXml();
    leitor.setContentHandler(logica);
    InputStream ips = new FileInputStream("src/vendas.xml");
    InputSource is = new InputSource(ips);
    leitor.parse(is);

    System.out.println(logica.getProdutos());
}
```

Outro ponto a analisar são os nomes das nossas classes. O `LeitorXml`, que trata o conteúdo e aplica a lógica de leitura do XML, é o que chamamos de `Handler`, uma classe que trabalha com os eventos. Pensando assim, vamos renomeá-lo para `ProdutosHandler` utilizando o "Refactor" do Eclipse, fazendo com que todo o projeto seja atualizado com o novo nome.

A classe `LeArquivoXmlDeOutraForma` também possui um nome pouco adequado. A técnica utilizada nessa classe é chamada de "SAX", acrônimo de "Simple API for XML". Sendo assim, vamos renomear essa classe para `LeXmlComSax.java`. Já primeira técnica, que utilizamos em `Sistema.java`, é chamada de "DOM" (Document Object Model).

```
public class LeXmlComSax {
    public static void main(String[] args) throws Exception {
        XMLReader leitor = XMLReaderFactory.createXMLReader();
        ProdutosHandler logica = new ProdutosHandler();
        leitor.setContentHandler(logica);
```

```
            InputStream ips = new FileInputStream("src/vendas.xml");
            InputSource is = new InputSource(ips);
            leitor.parse(is);

            System.out.println(logica.getProdutos());
        }
    }
```

Já em relação à organização das nossas classes, criaremos um novo pacote `br.com.alura.Handlers`, que armazenará nossos leitores de conteúdo. Moveremos `ProdutosHandler` para esse novo pacote. Com isso, terminamos a nossa refatoração.

A grande vantagem desse método de leitura de um XML é que ela simplesmente dispara eventos, sem manter informações na memória. Assim, temos a oportunidade de escolher quais dados deverão ser tratados, trazendo um processamento mais otimizado, por exemplo quando queremos extrair as vendas de abril de um relatório de vendas anual. A desvantagem é que esse código é relativamente mais complicado.

No próximo capítulo aprenderemos ainda outra forma de ler um arquivo XML.