01

Convertendo XML para HTML

Transcrição

O XML é uma linguagem genérica que organiza os nossos dados de forma hierárquica, e é natural desejarmos exibi-los de diversas formas, como em uma página Web. Como essas páginas utilizam a linguagem HTML, teríamos que converter nosso XML para tal linguagem. Em outra situação, desejaríamos exportar esses dados em um serviço, transformando-os em um JSON.

Ou seja, é muito comum precisarmos converter um XML para outro formato. Pensando nisso, foi criado O XSL (Extensible Stylesheet Language), uma linguagem que faz essa conversão. Para trabalharmos com esse conceito, criaremos um aquivo `xmlParaHtml.xls` .

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

</xsl:stylesheet>
```

Essa é a estrutura básica de um XSL. Um XLS pode trabalhar com diversos arquivos XML diferentes, da mesma forma que um XML pode ser utilizado por vários XLS diferentes. Sendo assim, a maneira mais interessante de associarmos este arquivo com o nosso XML é por meio de uma nova classe. No caso, criaremos a `ConversarParaHtml.java` no pacote `Teste` e, como ela deverá ser executada no console, criaremos também um método `main()` .

Para trabalharmos com os nossos arquivos, precisaremos de uma referência deles na memória, e conseguiremos isso com o já conhecido `FileInputStream` .

```
public class ConversorParaHtml {

    public static void main(String[] args) throws Exception {
        InputStream xsl = new FileInputStream("src/xmlParaHtml.xsl");

        InputStream dados  = new FileInputStream("src/vendas.xml");
    }

}
```

A classe que fará a "transformação" dos formatos é chamada justamente de `Transformer` . Conseguiremos uma nova instância dessa classe por meio de uma `TransformerFactory` , passando nossa variável `xsl` como argumento de `newTransformer()` .

```
public static void main(String[] args) throws Exception {
    InputStream xsl = new FileInputStream("src/xmlParaHtml.xsl");

    InputStream dados  = new FileInputStream("src/vendas.xml");

    Transformer transformer = TransformerFactory.newInstance().newTransformer(xsl);
}
```

O Eclipse nos indicará um erro, pois o método `newTransformer()` não está encontrando um argumento do tipo `Source`. Para corrigirmos isso, criaremos uma nova variável `xslSource` que será uma instância de `StreamSource` recebendo o nosso `xsl`.

```
public static void main(String[] args) throws Exception {
    InputStream xsl = new FileInputStream("src/xmlParaHtml.xsl");
    StreamSource xslSource = new StreamSource(xsl);

    InputStream dados  = new FileInputStream("src/vendas.xml");

    Transformer transformer = TransformerFactory.newInstance().newTransformer(xslSource);
}
```

Com a nossa classe instanciada, podemos invocar o método `transform()`, que nos pedirá uma `xmlSource`. Criaremos essa variável, novamente recebendo uma instÇanica de `StreamSource`, dessa vez utilizando nossos `dados` como base. Também precisaremos de um `StreamResult`, que nada mais é do que o formato que gostaríamos de obter como saída. Nesse caso, definiremos uma variável `saida` que receberá uma instância de `StreamResult` cujo diretório será `src/vendas.html`.

```
public static void main(String[] args) throws Exception {
    InputStream xsl = new FileInputStream("src/xmlParaHtml.xsl");
    StreamSource xslSource = new StreamSource(xsl);

    InputStream dados  = new FileInputStream("src/vendas.xml");
    StreamSource xmlSource = new StreamSource(dados);

    StreamResult saida = new StreamResult("src/vendas.html");

    Transformer transformer = TransformerFactory.newInstance().newTransformer(xslSource);
    transformer.transform(xmlSource, saida);
}
```

Executando nosso código e atualizando o projeto, teremos um novo arquivo `vendas.html` que consiste somente nos valores do nosso arquivo XML. Isso acontece pois, por padrão, o nosso transformador exibe somente o conteúdo das tags, e não definimos nenhuma instrução diferente dessa no nosso `xmlParaHtml.xsl`. No próximo vídeo aprenderemos a formatar o HTML resultante utilizando o arquivo XSL.