**CHƯƠNG 22: XSLT Results**

* Extensible Stylesheet Language (XSL) là World Wide Web Consortium đặc tả cho định dạng XML. XSL định nghĩa một tài liệu XML về cách hiển thị. XSL của XML giống như CSS của HTML. Có 2 kĩ thuật định nghĩa cho một đặc tả XSL: XSL Formatting Objects và XSL Transformations (XSLT). Trọng tâm chính của chương này là dạng Struts XSLT result.
* Chi tiết kĩ thuật về XSLT có thểm tham khảo ở: <http://www.w3.org/TR/xslt>

1. **Tổng quan**

* Tài liệu XML được sử dụng để trao đổi dữ liệu dễ dàng. Không giống như các cơ sở dữ liệu mà phải chỉ rõ từng định dạng loại dữ liệu làm cho việc trao đổi trở nên khó khăn hơn, tài liệu XML là dạng text bình thường có thể hiểu dễ dàng khi đọc. Ví du, tài liệu XML sau đây có chưa thông tin của một employee (nhân viên).

<employee>

<employeeId>34</employeeId>

<firstName>Jen</firstName>

<lastName>Goodhope</lastName>

<birthDate>2/25/1980</birthDate>

<hiredDate>3/22/2006</hiredDate>

</employee>

* Nếu bạn gửi tập tin XML này, bên nhận sẽ dễ dàng hiểu và có thể thao tác với các công cụ riêng của họ. Tuy nhiên, cũng không hoàn toàn dễ dàng, vì bên nhận có thể có các tập tin XML chứa thông tin nhân viên, mà định dạng khác nhau. Thay vì dùng **employeeId** thì họ dùng **id**, hoặc **employee** thay cho **worker**.

<worker>

<id>50</id>

<firstName>Max</firstName>

<lastName>Ocean</lastName>

<birthDate>12/13/1977</birthDate>

<hiredDate>10/5/2005</hiredDate>

</worker>

* Nếu dữ liệu của tập tin XML đầu tiên được trộn với tập tin XML thứ hai, thì phải có một vài chuyển đổi <employee> thành <worker>, và <employeeId> thành <id>. Đây là nhiệm vụ của XSLT.
* **Hình 22.1** thể hiện cách thức hoạt động của XSLT. XSLT processor có nhiệm vụ chính là đọc XML ban đầu và sử dụng stylesheet để đổi tập tin XML thành một dạng khác.

**Hình 22.1 Cách thức hoạt động của XSLT**



An XSL stylesheet is an XML file with an xsl or xslt extension. The root element of an XSL stylesheet is either **<xsl:stylesheet>** or **<xsl:transform>.** Here is the skeleton of an XSL stylesheet

* Một XSL stylesheet là một file XML với đuôi là xsl hoặc xslt. Element root (gọi là thẻ gốc) của XSL stylesheet là **<xsl:stylesheet>** hoặc **<xsl:transform>.** Sau đây là cuấ trúc của một XSL stylesheet:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

...

</xsl:stylesheet>

* Thẻ **xsl:stylesheet** có 2 thuộc tính. Thuộc tính đầu khai báo version, ở ví dụ trên là 1.0. Thuộc tính thứ 2 khai báo XML namespace. Ở ví dụ trên được chỉ tới W3C XSLT namespace. Tiền tố **xsl** hay được sử dụng cho một XSL stylesheet.
* Danh sách các thẻ được tìm thấy trong đặc tả (**đặc tả ở đâu ko biết nữa**). Sau đây là một vài thẻ quan trọng:
  + **xsl:template**: Định nghĩa một template. Nó có thuộc tính là **match** để liên kết template với một thẻ trong file XML ban đầu. Ví dụ, thẻ **xsl:template** sẽ match với root của XML ban đầu:

<xsl:template match="/">

* + **xsl:value-of**. Đọc một giá trị của một thẻ XML và thêm nó vào ouput stream. Bạn có thể chọn lựa một thẻ XML bằng cách sử dụng thuộc tính **select**. Ví dụ, Xuất giá trị của thẻ **name** thuộc thẻ **result**:

<xsl:value-of select="/result/name"/>

* + **xsl:for-each**. Dùng để lặp hết một node trong XML. Thẻ này cũng có thuộc tính **select** để chỉ rõ thẻ cần lặp trong file XML. Ví dụ, thẻ **xsl:for-each** lặp hết tất cả thẻ **result/supplier** và xuất ra giá trị từng thẻ **supplier** thành định dạng table của HTML.

<table>

<xsl:for-each select="/result/supplier">

<tr>

<td><xsl:value-of select="supplierName"/></td>

<td><xsl:value-of select="address"/></td>

</tr>

</xsl:for-each>

</table>

1. **The XSLT Result Type**

* XML chuyển đổi thành XML khác không phải là dạng biến đổi duy nhất mà XSLT có thể thực hiện. XSLT còn có thể thực hiện XML biến thành XHTML. Thực sự, XSLT có thể biến XML thành bất kì dạng văn bản nào.
* Struts XSLT result type (em chưa biết dich từ này ra sao) kiểm tra trong Value Stack và tạo ra một file XML raw (tạm gọi là file XML chưa được chỉnh sửa), với một thẻ result là thẻ root. Lồng bên trong thẻ này là toàn bộ các thuộc tính của action và các thông tin khác, ví dụ như nơi tạo, ngôn ngữ.. **XSLT result** sẽ sử dụng các **XSLT stylesheet** để chuyển raw XML thành dạng **XML** hoặc **XHTML.**
* XSLT result có các tham số sau:
  + stylesheetLocation: vị trí lưu trữ file stylesheet.
  + excudingPattem. Specifies excluded elements. Note that there's a typo (there is no 1 in excluding) that has not been fixed until Struts version 2.0.9. (Không hiểu rõ)
  + matchingPattern. Đặc tả pattern được matching. Mặc định matches tất cả.
  + parse. Chỉ rõ tham số stylesheetLocation có được parsed thành OGNL expressions hay không. Mặc định là không có (false).

Note there is also a deprecated location parameter that does the same thing as stylesheetLocation.

Note

By default XSLT stylesheets are cached. In development mode it's easier if they are not. You can change this behavior by setting struts.xslt.nocache to true in the struts.properties file.

**(XSLT Result Type chưa rõ các tham số của nó, tham khảo trang 388, sách Struts 2 Design and Programming (2008) )**

Consider the Product action class in Listing 22.1. The supplier property of Product is of type Supplier, shown in Listing 22.2.

* Ví dụ, cho class action Product ở Listing 22.1, trong class này có thuộc tính là supplier kiểu Supplier - chi tiết ở Listing 22.2.

Listing 22.1 Product action class

|  |
| --- |
| package app22a;  import com.opensymphony.xwork2.ActionSupport;  public class Product extends ActionSupport {  private String productId;  private String name;  private String description;  private double price;  private Supplier supplier;  // getters and setters not shown  public String execute() {  productId = "345";  name = "Epson";  description = "Super printer";  price = 12.34;  supplier = new Supplier();  supplier.setSupplierId("20a");  supplier.setName("Online Business Ltd.");  supplier.setAddress("Oakville, Ontario");  return SUCCESS;  }  } |

Listing 22.2 Supplier class

|  |
| --- |
| public class Supplier {  private String supplierId;  private String name;  private String address;  // getters and setters not shown  } |

* + XSLT result sẽ tạo ra một file XML raw từ Product action:

<result>

<actionErrors></actionErrors>

<actionMessages></actionMessages>

<description>

<#text>Super printer</#text>

</description>

<errorMessages></errorMessages>

<errors></errors>

<fieldErrors></fieldErrors>

<locale>

<ISO3Country>

<#text>USA</#text>

</ISO3Country>

<ISO3Language>

<#text>eng</#text>

</ISO3Language>

<country>

<#text>US</#text>

</country>

<displayCountry>

<#text>United States</#text>

</displayCountry>

<displayLanguage>

<#text>English</#text>

</displayLanguage>

<displayName>

<#text>English (United States)</#text>

</displayName>

<displayVariant>

<#text></#text>

</displayVariant>

<language>

<#text>en</#text>

</language>

<variant>

<#text></#text>

</variant>

</locale>

<name>

<#text>Epson</#text>

</name>

<price>

<#text>12.34</#text>

</price>

<productId>

<#text>345</#text>

</productId>

<supplier>

<address>

<#text>Oakville, Ontario</#text>

</address>

<name>

<#text>Online Business Ltd. </#text>

</name>

<supplierId>

<#text>20a</#text>

</supplierId>

</supplier>

<texts>

<#text>null</#text>

</texts>

</result>

1. **Ví dụ:**

* Ví dụ, ứng dụng **app22a** có một action sử dụng XSLT result. Action đó là **Product action** (xem lại Listing 22.1) được chuyển đổi thành XHTML. Chi tiết khai báo action ở Listing 22.3

Listing 22.3. The action declaration

|  |
| --- |
| <package name="app22a" extends="struts-default">  <action name="XSL" class="app22a.Product">  <result name="success" type="xslt">  <param name="stylesheetLocation">  /xsl/Product.xsl </param>  </result>  </action>  </package> |

* Action XSL sử dụng một XSLT result để sử dụng template Product.xsl - ở Listing 22.4

Listing 22.4 Product.xsl template

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <xsl:stylesheet version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">  <xsl:template match="/">  <product>  <productName>  <xsl:value-of select="/result/name"/>  </productName>  <productDescription>  <xsl:value-of select="/result/description"/>  </productDescription>  <price>  <xsl:value-of select="/result/price"/>  </price>  <supplierName>  <xsl:value-of select="/result/supplier/name"/>  </supplierName>  </product>  </xsl:template>  </xsl:stylesheet> |

* Bạn có thể kiểm tra lại ví dụ vừa rồi bằng cách vào đường dẫn sau: <http://localhost:8080/app22a/XSL.action>
* Kết quả ta được:

<?xml version="1.0" encoding="UTF-8"?>

<product>

<productName>Epson</productName>

<productDescription>Super printer</productDescription> <price>12.34</price>

<supplierName>Online Business Ltd.</supplierName>

</product>

Note

A modified org.apache.struts2.views.xslt.XSLTResult class is included in the app22a example. For debugging purpose, I added a method that prints the raw XML to the console or the Catalina.out file. The XSLTResult class is the underlying class of the XSLT result type.

1. **Tổng kết**

XSLT result type chuyển đổi một đối tượng action thành XML . Kiểu kết quả trả ra của XSLT này tuy không thông dụng như Dispatcher, nhưng được dùng trong những ứng dụng mà yêu cầu xuất ra XML, ví dụ web services. Trong chương nay, chúng ta đã biết được cách thức làm việc và sử dụng nó.