Module "Rechercher et Publier sur le Web"
TP - Web publishing with XML/XSLT

All the XML sample documents mentioned in the sequel can be obtained at the address :
`http://www.lsv.ens-cachan.fr/~sirangel/teaching/EDSP/RechWeb/TPs/`

Any Web browser can be used to experiment XSLT transforms. Just put the following processing instruction in the prologue of the XML document that must be transformed:

```
<?xml-stylesheet href="prog.xsl" type="text/xsl"?>
```

where `prog.xsl` is of course the name of the XSLT stylesheet.

Exercise 1

Take any XML document, write an XSLT program without any template, and apply the transformation to the document. Explain what is going on.

Exercise 2

Write a program with a single template that applies to the document root (set the match attribute to /). The body of the template must instantiate a complete yet simple HTML document showing the main information on a movie: title, year, genre and abstract, as well as director's name. Use only `<xsl:value-of>` XSLT elements, and XPath expressions. Apply the program to an XML document describing a single movie (e.g., Spider-Man.xml).

Exercise 3

Now write a program that uses the `<xsl:apply-templates>` mechanism. The program applies to movies.xml, and creates an HTML representation that shows the list of movies, with the following templates:

- the first template applies to the document root and produces a valid HTML document (tags `<html>`, `<head>`, `<body>`) ; in `<body>`, call for the application of the appropriate templates to the `<movie>` elements;
- a second template that applies to a `<movie>` element and shows the title, genre and country ; also add an XSLT call for the application of the appropriate template to the `<actor>` elements.
- finally the last template applies to the actors, and produces a list with the name of each actor and his/her role in the movie.

Exercise 4

Write an XSLT program which displays the list of name and value of all the attributes of an input document.

Hint: the name() XPath function gives the name of the context node.

Exercise 5

Write an XSLT program that shows the names of the elements of an XML document, and for each element:

- the number of its attributes and
- the number of its descendants of type Element.

Hint: the number of nodes in a nodeset is obtained with the XPath function count().

Exercise 6

Write a program as in Exercise 5 but for each element display also its number in the document order.

Exercise 7

Write an XSLT program that takes any XML document I as input, and produces as output an XHTML document O which shows the structure and content of I. O should be displayed by a Web browser as it appears in an XML editor, using indentation to represent the hierarchical structure, and showing the opening and ending tags for elements. Use nested XHTML lists (`<ul>`) for indentation, and XHTML entities &*lt*; and &*gt*; for displaying element tags.

- Create a first version which takes only account of **Element** and **Text** nodes.
- Then extend the first version by adding attributes (put them in the opening tag, following the XML syntax).

Exercise 8

Write an XSLT program that produces, from the movies.xml document, an XHTML document with:

1. A table of contents, at the top of the page, showing the years, sorted in ascending order, of all the movies.
2. Then the list of movies, sorted by year in ascending order. For each movie, show the year, director, title (in italics) and the summary.

Of course, each year Y in the table of contents is a link that leads to the beginning of the group of movies published in Y. To simplify the task you can assume that each group contains a single movie (i.e. movies.xml does not contain any two movies published in the same year).

Hints :

- Use XHTML internal anchors(`<tag id="tt">...</tag>` can be referred to with `<a href="#tt">...</a>`.)
- A template can create dynamically the value of an attribute by using *interpolation* in attribute values. This is obtained by inserting XPath expressions surrounded by curly brackets inside attributes values. For instance the expression `<li name="{title}"> ...</li>` in the body of a template generates in the output document a `<li>` element whose `name` attribute has the value of the `title` child of the context node.
- The function *genrate-id*(*node set*) returns a unique identifier associated with the first node of the *node-set* in document order (the *node set* being specified by an XPath expression). Thus *generate-id*(.) returns the unique identifier associated with the context node.