PreProccessFilter 1

Introduction

This preprocess filter normalizes the syntax of an xml file by rearranging the structure of it in order to fix syntactic issues and to be compliant with a semantic model's concepts. Its main functionality is to rearrange the order of the children of one node, and create a new parent node for each (semantically related) group of the children. It is applied mostly in cases that the xml elements under a parent node are ordered by the name of the elements and not by the semantic-relations that exist.

The filter is implemented as a maven java project. It can be executed either by the command line, providing a number of arguments or by filling a configuration file and running the produced jar.

User Manual

The user can use the filter either by running it by the command line providing the required parameters or by filling a configuration file and running the jar. These approaches are described in the two following sections.

Running by the config.properties file and the jar

In order to use the filter by running the jar the user has to fill a config.properties file. Specifically the user has to fille the following:

- inputFilePath : the path of the xml input file
- outputFilePath: the path of the ouput xml file that will be produced
- parentNode: the parent node that contains the elements that will be rearranged.
- newParentNode: the new parent node that will be created
- intermediateNode: the main intermediate node that will be the parent of the semantically-grouped elements
- intermediateNodes: the names of the elements that will be rearranged separated by ","
- delimeter: the delimeter that will be used for splitting the initial values of the parent Node

After the filling of the config.properties file the user can run the jar (***preProcessFilter1-1.0.jar***) either by double-clicking on it or by running it from the command file with the command:

java –jar preProcessFilter1-1.0.jar

IMPORTANT: the config.properties file and the preProcessFilter1-1.0.jar must be in the same folder

Running by using command line arguments

To run the filter by the command line the user has to provide the following arguments:

-inputFile : the path of the xml input file
-outputFile: the path of the output xml file that will be produced
-parentNode: the parent node that contains the values that will be splitted and re-assigned
-newParentNode: the new parent node that will be created
-intermediateNode: the main intermediate node that will be the parent of the semantically-grouped elements
-intermediateNodes: the names of the elements that will be rearranged separated by ","
-delimeter: the delimeter that will be used for splitting the initial values of the parent Node

The command to run the filter is the following:

java -jar preProcessFilter1-1.0.one-jar.jar -inputFile "inputFilePath" -outputFile "outputFilePath" -parentNode "parentNodeName" -newParentNode "newParentNodeName" -intermediateNode "intemediateNode" -intermediateNodes "intNode1,intNode2,..." -delimeter "delimeter"

If the command runs successfully the following message will be returned:

"Successful Preprocessing!!!"

Example

An example of using the filter is described in this section.

The input file is the events.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
    <events>
        <date>2012</date>
        <date>2013</date>
        <date>2014</date>
        <actor>actor</actor>
        <actor>actor3</actor>
        <other>someothervalues</other>
        <place>place1</place>
        <place>place2</place>
        <place>place3</place>
    </events>
</root>
```

The elements that will be rearranged are highlighted with red color. The node events contain information about the date, the actor and the place of a number of events. They are ordered

by the name of the element and not by the fact that they refer to the same event. The first date is the date of the first event, the first actor is the actor of the first event and the first place is the place of the first event. The goal is to create intermediate parent nodes for each event and to add as children to each node the related date, actor and place. The new xml file will be called eventsOutput.xml.

So, the configuration file will be the following:

```
#  Configuration file
#   If you want to use '\' make sure to change it '\\'
#   Separate the tags by using ','

inputFilePath= events.xml

outputFilePath= eventsOutput.xml

parentNode=events

intermediateNode=event

tags=date,actor,place
```

By running the "***preProcessFilter1-1.0.jar"*** the eventsOutput.xml file will be produced (the new values are in green fonts):

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <events>
    <event>
      <date>2012</date>
      <actor>actor</actor>
     <place>place1</place>
    </event>
    <event>
      <date>2013</date>
      <actor>actor3</actor>
      <place>place2</place>
    </event>
    <event>
      <date>2014</date>
      <place>place3</place>
      </event>
     <other>someothervalues</other>
  </events>
</root>
```