Dokumentace úlohy XQR: XML Query v Python 3 do IPP 2016/2017
Jméno a příjmení: Martin Bažík
Login: xbazik00

1 Task

The task was to create a script in Python 3 that would carry out query upon a XML file. Output of this script is a XML file containing all the elements from the source XML file that meet the conditions defined in the provided query.

2 Parameters

The scanning of parameters is done using the library `getopt`. In case the incorrect combination of the parameters is given, the error is printed out on the standard error output.

3 Opening input file

The input file is opened by the method `parse()` from library `xml.etree.ElementTree`. The input file can be either a file or stdin depending on the parameters. If the file does not exist or user does not have the required permissions to open the file, the error of the output file is handled by the exception.

Class XML is created for the input file. When the object of this file is constructed the method `parse()` parses the input file into the tree of XML elements. This tree is afterwards processed in the method `process_xml()`. The whole processing of input file is described in the section 4.

4 Processing input file

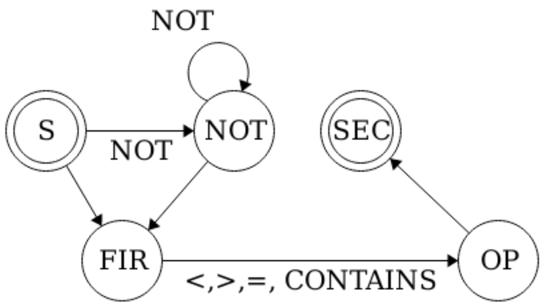
The element tree is processed according to the XML query defined in the section 5. The process itself is implemented in method `process_xml()` and is carried out by iterating the element tree using method `iter()` that iterates all sub-elements recursively so it is possible to execute the depth search. Firstly all the elements with a corresponding name, defined in query, are selected. After that this selection is filtered, so only relevant elements remain. The relevancy of the element is decided by the attributes of the `element`: `text` and `attrib`. Attribute `text` represents value of the element and attribute `attrib` is a dictionary of element's attributes. These two values are compared to the value of a condition that was defined in query to decide should the given element be selected. The condition is evaluated in the method `cond_choose()`, where conditions for numbers and strings are evaluated individually. If the condition is true, the element is returned, else `None` is returned. Number of selected elements is decreased afterwards depending on the limit given by query. The selected elements are stored in an array as attribute of the XML object. The result is generated and redirected to the output by the method `generate_output()` described in the section 7.

5 XML query

XML query is defined in parameter `--query` or can be found in a file defined in parameter `--qf`. XML query is parsed in the object `Query` into a dictionary containing all the necessary values separately identified by the key. The query itself is parsed as following. Firstly of query is split into separated by whitespaces into tokens and then, by searching the keywords, their values are stored in the prepared dictionary. These parts consist of SELECT, FROM, WHERE, LIMIT. At the places where token consists of two parts representing element and attribute, these parts are stored separately. The condition, defined in WHERE parsed by Finite State Machine shown in the section 6.

6 Finite State Machine

Firstly it looks for the token NOT. If no token NOT is found, it proceeds to parse the first operand. When an operator is found, it parses the operator and proceeds to the second operand.



7 Output

The output file is either given by ”–output” parameter or is the standard output. The output is generated by the method `generate_output()` of the object `XML`, where the selected elements are turned into text by the method `tostring()`. Depending on the parameter `-n` XML head is generated and depending on the paramater `--root` the enclosing element is generated. Lastly the output file is created, containing the string created within the method `generate_output()`.