Dokumentace úlohy CSV: csv2xml to Principles of Programming Languages 2015/2016
Name and Surname: Radovan Sroka
Login: xsroka00

Introduction

The main goal of this project was to create a python script which would convert file in 'Comma Separated Values' format into the Extensible Markup Language format (XML) as specified in projects formal requirements.

Design

Projects implementation was fairly straight-forward, command line arguments are parsed via pythons argparse standard library, each optional argument parameter is further checked for format correctness. Conversion of CSV input onto XML output is done on the fly, line by line. Since the conversion follows straight-forward program flow, all of the critical program exceptions cause immediate program execution termination. While python3 supports UTF-8 natively, all of the parsing functions (such as csv.arg parse, open) are setup to use exclusively utf-8 encoding. Source code as well as comments and documentation are written entirely in english.

Arguments Parsing

Upon script execution, argument parsing function is called. This initializes all the variables required for script execution as well as prepares input/output, verifies that input file exists (if –input= option was given) and that output destination is writeable (if –output= option was given). After argparse standard function parses and matches argument options onto program variables, program checks for incompatible argument options. This is followed by argument parameters checks, such as that -s=separator really only contains a single character as defined by project requirements. Option –help is generated automatically via argparse library as well as handling of unsupported arguments.

CSV parsing

CSV file format parsing is implemented via Python's standard 'csv' library. While this library allows for many dialets and options, minimal preset was chosen and more advanced parsing is done during conversion process manually. This includes conversion of characters unsupported by XML format, such as '<','>','&', etc.

XML conversion

XML format comes in many flavors and implementation in this project is based off provided reference output format. I'm using lxml library to generate XML document. If parsing and all checks are successful, XML buffer is then flushed onto output, be it stdout or specified file.

Implementation details

Script is stored within single file, csv.py and this file structured into parts like Parsing, Checking, Generating and Formating output. So in the first part are all the command line arguments parsed and also script reads input into the memory buffer. After argument parsing and initialisation comes Checking part. There are lots of checks there which are checking if some arguments are disjunct with other and so on. Another part is Generating, in that part script split CSV input by CSV library and generate two dimensional array(in python list). There are other checks for CSV validity and error correction. Before the final phase, XML is generated and there are many nested for cycles working on that. Final phase is about formating, substituting and escaping. When script finally finalize XM, it is printed to stdout or some other file or an error occurs and program terminates with appropriate error code.

Ending thoughts

This project also was not my first experience with python3,
For development, I used Fedora Linux, Python 3.5 and Vim.
I was using JExamXML and standard unix diff against my results and reference output.
Testing was done on top of provided reference tests, script was first and foremost tuned to satisfy basic reference tests.