Task documentation SYN: Syntax highlighting in Python for IPP 2016/2017

Name and surname: Róbert Kolcún
Login: xkolcu00

March 15, 2017

1 Problem & Program structure

Main objective in this project is to highlight input text with html tags by given rules in format file. I divided this program into several sections(classes). Main program is in *syn.php* script. Next 3 scripts, first one parse input arguments and read or write to files *(argParser.lib.php)*. Second one, parse and check data from format file *(parser.lib.php)* and the last one highlight input text with html tags *(syntaxHighlight.lib.php)*.

2 Argument parser & I/O

Script *argParser.lib.php* contain one class named *ArgumentParserAndFiles*, which parse input arguments from command line by function *parseIt()*. Program support short and long type of arguments. For example -h and –help. This class work with input, output and format file. Class read input data from input file or stdin if input file is not given, function *getInputFileData()* return whole input data. Output data are written to outputfile or stdout if output file is not given, function *printToOutputFile($inputText)* write data. And function *getFormatLine()* read exactly one line from format file and return it. If format file is not given, input data are written to output with no tags. When error occur, every function return error-code to main script *(syn.php)*.

3 Format file parser & Regular expressions

Class *CommandParser* located in *parser.lib.php* file contain every known rule in $commands and every checked command from format file is stored to $allCommands.

```
private $commands = ['bold', 'italic', 'underline', 'teletype', 'size', 'color'];
private $allCommands = array();
```

Main script call *addFormatString($inputFormatLine)* function from this class, this function parse one input line from format file into 2 sections by regular expression

```
preg_match('(^([\S ]+)\t+([\S\t ]+)$)', $inputFormatLine, $match);
```

First section contain regular expression and second one contain all given rules. Regular expression is passed to *getOriginalRegex($inputRegex)* function. This function convert given regular expression to format of regular expression used in *PHP*, because format of given regular expression is incompatible with regular expressions that are used in *PHP*.

Second section is passed to *checkCommands($inputCommands)* function. Variable $inputCommands is splited to array by regular expression

```
$splitedCommands = preg_split('([\t ]*(,)[\t ]*)', $inputCommands);
```

Every element of this array is splited by ':' delimiter. Every rule is checked with array $commands if exists. Then range of 'size' and value of 'color' is checked.
Now converted regular expression and checked rules are stored to variable $allCommands:

```
$this->allCommands[] = array('regex'=>$regex ,'commands'=>$checkedCommands);
```

4 Code highlighting

The last class *SyntaxHighlight* located in *syntaxHighlight.lib.php* file with constructor *__construct($input, $commands, $isBr)* contain only one function *highlightCode()*. This function iterate variable $commands and find all results which match a regular expression.

```
preg_match_all('('.$this->commands[$i]['regex'].')s', $this->input, $match, PREG_OFFSET_CAPTURE);
```

With flag PREG_OFFSET_CAPTURE set, index of match is stored to variable $match. So then I iterate $match, calculate start and end index of current match:

```
$begin = $element[1];                    // first position of match
$end = strlen($element[0]) + $begin;     // last position of match
```

After that I iterate all commands that correspond to current regular expression and add tags to two arrays, $beginTags array where key is first position of match ($begin), value is tag. And $endTags array where key is last position of match ($end), value is tag. Finally I iterate whole input data backward, if key exist in $beginTags or $endTags array, insert value of this array to $input. Return highlighted text, which main script print to output file.