Itcldoc

History

- When starting to implement ATWF I wanted to be able to do inline documentation
- There are some tools like tcldoc, autodoc and zdoc all do not allow to document Itcl an TclOO
- Idea to use itcl-ng parser for splitting up sources
- Found cmdSplit and parsetcl in the wiki
- Decision to use a modified version of cmdSplit
- Decision to use a tree structure for the scanned info influenced by parsetcl

Goal of the Tool

- Collect information of the contents of Tcl scripts
- Be able to also parse Itcl-ng and TclOO
- Be able to parse ensemble commands like namespace and info
- The original source can be generated again including indentation etc.
- Allow Javadoc like output as well as cross reference output

General Structure

- Use of a modfied version of cmdSplit, which preserves a lot more information like indentation, newlines, comments etc.
- Recursive parsing of the parameters of a command
- Recognition of regexp constructs and namespace parts
- Storing the scanned info in dicts
- Further scanning of the input parts using a tree structure and reorganizing tree parts in that structure
-

Scanning Details (1)

- cmdSplit parses a script into commands with parameters
- cmdsplit parses parses whitespace, newlines and comments into „special“ commands
- Further scanning of the scanned commands

Scanning Details (2)

- Further scanning using special characters as tokens second string is token name

{	LC	}	RC
[	LB	]	RB
(	LP	)	RP
“	QUOTE	$	DOLLAR
*	STAR	\	BACKSLASH
+	PLUS	-	MINUS
%	MOD	/	DIV
|	OR	&	AND
<	LT	>	GT
=	EQUAL	@	AT
?	QUEST	:	COLON
.	DOT	;	SEMICOLON
!	BANG		

Scanning Details (3)

- This allows recognition of arrays, rexexp constructs etc.
- Transforming subtrees to different subtrees
  - a DOLLAR node and a STRING node → VARREF node
  - a STRING, a LP, a STRING, a RP mode → ARRAYREF node
  - a COLON, a COLON, a STRING, a COLON, a COLON, a STRING node → NAMESPACE or ITCLCMD or TCLOOCMD node
- Further scanning of the scanned commands

Scanning Details (4)

- Transformation is done using tdom by putting the original scanned info into a tdom tree and the manipulating the tdom tree
- Output for saving on file system is can be xml with the node names as xml tags
- additional info like the indexes as attributes for the tag names
- The STRING nodes are representated as text nodes in dom tree
- The xml output can be converted to dicts and then used as format for saving onto file system

Scanning Details (5)

- The scanning is done on a per script base
- Every source command with the scripts creates its own scanned file
- May be influenced by nagelfar
- Cross reference can be produced by using a tree of parsed files

Status

- This project is a work in progress
- ATM not much activity because of other projects
- Possibly influenced in the future from ATWF
- Will be continued!

Conclusions

- Work in progress
- Priority lower than other project of me like ATWF