NAME

parse − Parse a Tcl script into commands, words, and tokens.

SYNOPSIS

package require **Tcl 8**

package require **parser ?1.4?**

parse command *script* [arg first] [arg length]

parse expr *script* [arg first] [arg length]

parse varname *script* [arg first] [arg length]

parse list *script* [arg first] [arg length]

parse getrange *string* ?index length?

parse getstring *string* [arg first] [arg length]

parse charindex *string* [arg first] [arg length]

parse charlength *string* [arg first] [arg length]

DESCRIPTION

This command parses a Tcl script into *commands*, *words* and *tokens*. Each of the commands below takes a *script* to parse and a range into the script: {*first length*}. The command parses the script from the first index for *length* characters. For convenience *length* can be set to the value "end". The return of each command is a list of tuples indicating the ranges of each sub-element. Use the returned indices as arguments to **parse getstring** to extract the parsed string from the script.

The **parse** command breaks up the script into sequentially smaller elements. A *script* contains one or more *commands*. A *command* is a set of *words* that is terminated by a semicolon, newline or end the of the script and has no unclosed quotes, braces, brackets or array element names. A *word* is a set of characters grouped together by whitespace, quotes, braces or brackets. Each word is composed of one or more *tokens*. A *token* is one of the following types: *text*, *variable*, *backslash*, *command*, *expr*, *operator*, or *expand*. The type of token specifies how to decompose the string further. For example, a *text* token is a literal set of characters that does not need to be broken into smaller pieces. However, the *variable* token needs to be broken into smaller pieces to separate the name of the variable from an array indices, if one is supplied.

The *first* index is treated the same way as the indices in the Tcl **string** command. An index of 0 refers to the first character of the string. An index of end (or any abbreviation of it) refers to the last character of the string. If first is less than zero then it is treated as if it were zero, and if first + length is greater than or equal to the length of the string then it is treated as if it were end.

parse command *script* [arg first] [arg length]

Returns a list of indices that partitions the script into *commands*. This routine returns a list of the following form: *commentRange commandRange restRange parseTree*. The first range refers to any leading comments before the command. The second range refers to the command itself. The third range contains the remainder of the original range that appears after the command range. The *parseTree* is a list representation of the parse tree where each node is a list in the form: *type range subTree*.

parse expr *script* [arg first] [arg length]
Returns a list that partitions an *expression* into subexpressions. The first element of the list is the token type, *subexpr*, followed by the range of the expressions text, and finally by a *subTree* with the words and types of the parse tree.

parse varname *script* [arg first] [arg length]
Returns a list that partitions a *variable* token into words. The first element of the list is the token type, *variable*. The second is the range of the variable's text, and the third is a subTree that lists the words and ranges of the variable's components.

parse list *script* [arg first] [arg length]
Parses a script as a *list*, returning the range of each element. *script* must be a valid list, or an error will be generated.

parse getrange *string* ?index length?
Gets the range in bytes of *string*, optionally beginning at ?index? of length ?length? (both in characters). Equivalent to **string bytelength**.

parse getstring *string* [arg first] [arg length]
Get the section of *string* that corresponds to the specified range (in bytes). Note that this command must be used instead of **string range** with values returned from the parse commands, because the values are in bytes, and **string range** instead uses characters as its units.

parse charindex *string* [arg first] [arg length]
Converts byte oriented index values into character oriented index values, for the string in question.

parse charlength *string* [arg first] [arg length]
Converts the given byte length into a character count, for the string in question.

EXAMPLES

```
set script {
while true {puts [getupdate]}
}
parse command $script {0 end}
```

Returns:

{0 0} {5 30} {35 0} {{simple {5 5} {{text {5 5} {}}}} {simple {11 4} {{text {11 4} {}}}} {simple {16 18} {{text {17 16} {}}}}}

Or in other words, a string with no comments, 30 bytes long, beginning at byte 5. It is composed of a series of subwords, which include while, true, and {puts [getupdate]}.

KEYWORDS

parse, parser