NAME

snobol4tcl − SNOBOL4 Tcl/Tk interface

SYNOPSIS

-INCLUDE 'stcl.sno'

tclhandle = STCL_CREATEINTERP()

STCL_DELETEINTERP(tclhandle)

STCL_EVALFILE(tclhandle,tclfilename)

value = STCL_GETVAR(tclhandle,varname)

STCL_SETVAR(tclhandle,varname,value)

STCL_EVAL(tclhandle,tclstmt)

DESCRIPTION

Tcl is an embedable scripting language developed by John Osterhout, while at the University of Clifornia, Berkeley. Tk is a graphical user interface toolkit developed for Tcl.

This page describes STCL, an experimental facility for invoking Tcl and Tk from SNOBOL4 programs, inspired by Arjen Markus' "ftcl" FORTRAN/Tcl interface

STCL_CREATEINTERP creates a Tcl interpreter and returns a handle which can be passed to the remaining functions.

STCL_DELETEINTERP destroys a Tcl interpreter.

STCL_EVALFILE reads a Tcl script file into the referenced Tcl interpreter.

STCL_GETVAR retrieves the string value of named variable from a Tcl interpreter. **STCL_GETVAR** stores a string value of named variable in a Tcl interpreter.

STCL_EVAL evaluates a string containing Tcl code in a Tcl interpreter.

FILES

NDBM, GDBM, and SDBM create two files: *filename*.dir, *filename*.pag. Berkeley DB creates a single *filename*.db file.

EXAMPLE

```
-INCLUDE 'stcl.sno'
      INTERP = STCL_CREATEINTERP()
      TCL_VERSION = STCL_GETVAR(INTERP, "tcl_version")
      OUTPUT = IDENT(TCL_VERSION) "Could not get tcl_version" :S(END)
      OUTPUT = "Tcl Version: " TCL_VERSION

* check Tcl version
      NUM = SPAN('0123456789')
      VPAT = NUM '.' NUM
      TCL_VERSION VPAT . VER                  :S(CHECKV)
      OUTPUT = "could not parse tcl_version"  :(END)

CHECKV  LT(VER, 8.4)                          :S(CHECKTK)

* Tcl 8.4 and later can dynamicly load Tk!
      STCL_EVAL(INTERP, "package require Tk")  :F(END)

* Check for Tk
CHECKTK TK_VERSION = STCL_GETVAR(INTERP, "tk_version")  :F(NOTK)
      DIFFER(TK_VERSION)                      :S(HAVETK)

NOTK    OUTPUT = "Could not find tk_version"  :(END)

      STCL_EVAL(INTERP, "package require Tk")  :F(END)

* Check for Tk

CHECKTK TK_VERSION = STCL_GETVAR(INTERP, "tk_version")  :F(NOTK)
      DIFFER(TK_VERSION)                      :S(HAVETK)

NOTK    OUTPUT = "Could not find tk_version"  :(END)

HAVETK  OUTPUT = "Tk version: " TK_VERSION
      SEP = ';'

      STCL_EVAL(INTERP,
+           'button .hello -text "Hello, world" -command {set foo 1}' SEP
+           "pack .hello" SEP
+           'button .other -text "Other Choice" -command {set foo 2}' SEP
+           "pack .other" SEP
+           "global foo" SEP
+           "vwait foo")

      OUTPUT = STCL_GETVAR(INTERP, "foo")
END
```

SEE ALSO

tclsh(n), **Tcl**(n).

AUTHOR

Philip L. Budne

BUGS

NOTE! By default the STCL extension is not built into **snobol4**(1), it must be explicitly included at build time. In Tcl 8.4 and later, Tk can be dynamicly loaded by Tcl at runtime, but in earlier releases, it has to be included at compile time. When dynamicly linked libraries are not available, this can cause the SNOBOL4 interpreter executable to expand by up to four fold! STCL should be a dynamicly loaded SNOBOL4 extension.