TCL: Tool Command Language
Tk: (GUI) Toolkit

Language for doers

Shimon Panfil, Ph.D

What is Tcl?

Tcl (pronounced *tickle*) is

- extensible
- embeddable
- multi-platform

scripting language developed by John Ousterhout and others. For history, comparison etc. visit links:

- http://www.tcl.tk
- http://icemcfd.com/tcl/comparison.html

Tcl-Python-Perl- ....

1. Tcl/Lisp/Python: A "User" point of view , by Jeffrey Templon
2. Q: perl vs Tcl: complementary, orthogonal , by Tim Bunce
3. Why you should not use Tcl , by Richard Stallman
4. Re: Why you should not use Tcl , by John Ousterhout
5. Re: Why you should not use Tcl , by Wayne Throop
6. Perl Peeves , by Tom Christiansen
7. GNU Extension Language Plans , by Richard Stallman

Tcl: Hello world

```
#!/usr/bin/tclsh

#hello.tcl
set s "hello world!"
puts "* $s *"
exit 0
#end of hello.tcl
$ ./hello.tcl
* hello world! *
```

Tcl command

command: command arg1 arg2 arg3 . . .
command is the name of built-in command or Tcl procedure. Space and tabs separate command and arguments, eol and ; terminate commands.

grouping: strings inside {. . . } or ". . . " are combined into single argument

substitution: value of variable or nested command are taken

Tcl does grouping than substitution and finally call command.

Mathematical expressions

Tcl does not evaluate mathematical expressions, expr command does the job. It concatenates all arguments into string and parses it as mathematical expression according to C-syntax.

```
#!/usr/bin/tclsh
set a 2.0
set b 2.0
set c [expr $a*$b]
set d $a*$b
puts "$d=$c"
exit 0
```

Mathematical expressions Cont'd

```
$ ./2x2.tcl
2.0*2.0=4.0
```

Substitution and grouping

Command line is separated into "words". {. . . }, [. . .], ". . . " are also "words". Double quotes make grouping with substitutions Braces make grouping without substitutions they may nest. If first symbol of the word is

dollar: variable substitution

bracket: command substitution, recursive invocation of Tcl, [. . .] is replaced by its result.

brace: Tcl simply strips {}.

backslash: backslash substitution, disables special meaning of the subsequent symbol

Substitution and grouping cont'd

Tcl always performs a single round of substitutions. If the result contains variables or nested commands these are not expanded.

Control structures

```
if {expr} {
    body1
} else {
    body2
}
```

Control structures cont'd

```
foreach arg $argv {
  switch -glob -- $arg {
    -v  {set Verbose true}
    -f* {set FileToRead
         [string range $arg 2 end]}
    -h {puts stderr $Usage
        exit 1}
    default {
        error "bad argument: $arg"
        exit 1}
  }
}
```

Control structures cont'd

```
set n 5
set result 1
for {set i $n} {$i} {incr i -1} {
    set result [expr $result * $i]
}
set fd [open "foo" "r"]
while {[gets $fd line] != -1} {
    puts "$line"
}
```

Procedures

```
proc fact {n} {
    set result 1
    while {$n>1} {
        set result [expr $result * $n]
        set [expr $n -1]
    }
    return $result
}
```

Procedures cont'd

upvar old new creates variable new in procedure which is actually variable old which exist outside the procedure. uplevel command execute command in the calling context of procedure. Actually both upvar and uplevel take additional argument pointing the exact level in the execution stack we want to reference.

Tcl features

Tcl supports a lot of functions for

- string processing
- list processing
- arrays (key–value pairs) processing

It also supports error trapping, sockets, namespaces, packages and all that.

Tcl Extensions

It is quite easy to incorporate Tcl into your own programs, you need to link your own functions with Tcl library to create your own Tcl interpreter that is tailored to your specific needs.
You can also extend the functionality of Tcl. A number of such extensions are well known e. g.:

expect: automatic control of interactive programs; popular dejaGnu test software is written on top of expect.

incr Tcl: OO extension of Tcl

TcX: extended Tcl for UNIX,contains many functions mirroring system calls

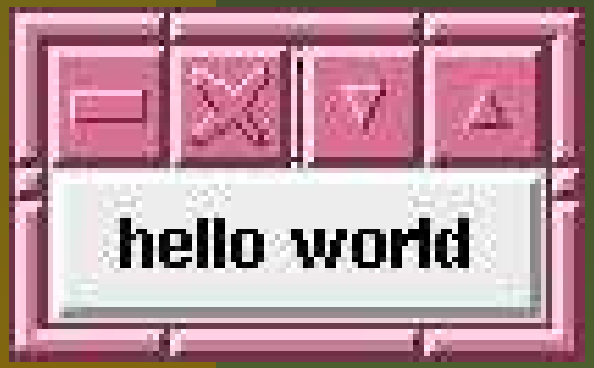
Tcl extensions cont'd

Img: image processing

Tk: Tk bestows on Tcl the ability to create and manipulate GUI objects. Initially developed for X Window system it has been successfully ported to other platforms.

Tk Hello world

```
#!/usr/bin/wish -f
set b [button .b \
        -text "hello world" \
   -command exit]
pack $b
```



Tcl/Tk applications

TkDVI is a DVI previewer built with the Tcl/Tk toolkit.
http://tkdvi.sourceforge.net/

TkVNC is a VNC viewer written in pure Tcl/Tk. It is designed to be embedded inside other applications, but can be used as a standalone VNC viewer.
http://www.ifost.org.au/Software/tkvnc/index.html

Xpvm is a TCL/TK based tool that allows full manageability of the PVM cluster as well as the ability to monitor cluster performance.
http://www.csm.ornl.gov/pvm/

My Tcl/Tk applications

I use Tcl/Tk mainly as

glue language: to combine a number of independent computational programs (which I write in C) e. g. Monte Carlo photon simulations and thermodynamic calculations;

GUI: to set a lot of parameters in a less error prone way;

Data Viewer: to represent 3d numerical data in human understandable form.