


Chapter 6

Tcl/Tk - 1


Summary

- ✔ Tcl
  - ✔ commands
  - ✔ quoting and substitutions
  - ✔ assignment (set)
  - ✔ Lists and arrays


Tcl/Tk list - recap

Just a sequence of words:

```
% set dow {Mon Tue Wed Thu Fri Party Sun}
Mon Tue Wed Thu Fri Party Sun
% lindex $dow 3
Thu
%
lindex $dow 1
Tue
```



Lists within lists - recap

```
% set a {0 1 {2 x} {3 x y} 4}
0 1 {2 x} {3 x y} 4
% lindex $a 3
3 x y
```

Iteration over list - recap

```
% foreach day $dow {
     puts Day\ of\ week\ is\ $day
  }
Day of week is Mon
Day of week is Tue
Day of week is Wed
Day of week is Thu
Day of week is Fri
Day of week is Party
Day of week is Sun
```


List operators

```
sort    ?options? <list>
split   <string> ?splitcharacters?
concat  <list> <list>
lindex  <list> <index>
... + lots more
```


Sorting a list

```
% lsort $dow
Fri Mon Party Sun Thu Tue Wed
```


Split (string to list)

```
% split "/usr/local/bin/tclsh" /
{} usr local bin tclsh

% join {{} usr local bin} /
/usr/local/bin
```


Arrays - recap

```
% set work(Mon) 8
8
% set work(Tue) 10
10
% foreach day $dow {
     puts $day\ I\ worked\ $work($day)hrs
  }
Mon I worked 8hrs
Tue I worked 10hrs
can't read "work(Wed)": no such element in array
```



Arrays - recap

```
% array size work
2
% array names work
Tue Mon
```



Multidimensional arrays

```
% set lect(Mon,8-10) cs2106
cs2106
% set lect(Mon, 8-10) cs2106
wrong # args: should be "set varName ?new-
Value?"
```



if then else

```
% if {3==3} {puts hi}
hi
% if {3==4} {puts hi}
% if {3 == 3} {puts hi}
hi
% if {"help" == "help"} {puts hi}
hi
```

while

```
% set c 3
3
% while {$c>=0} {
       puts "c is $c"
       set c [expr $c-1]
   }
c is 3
c is 2
c is 1
c is 0
```


foreach

```
% foreach day $dow {
     puts $day\ I\ worked\ $work($day)hrs
  }
Mon I worked 8hrs
Tue I worked 10hrs
...
```


Control - summary

```
if {test} {thenpart} {elsepart}
while {test} {body}
for {init} {test} {incr} {body}
continue
switch $x {a {a-part} b {b-part}... }
```


Procedures

```
% proc fac x {
     if {$x<=1} {return 1}
     expr $x * [fac [expr $x-1]]
  }
% fac 4
24
```

Globals

```
% proc printerrorno {
    global errno
    puts “The error is $errno”
  }
```


Parameter passing

- ✔ By value
- ✔ i.e. not arrays

Parameter passing

```
% set arr(1) 10
% set arr(2) 20
% proc prnt ar {
    puts $ar(1)
    puts $ar(2)
  }
% prnt arr
can’t read "ar(1)": no such variable
```


Parameter passing

```
% proc prnt ar {
    upvar $ar arra
    puts $arra(1)
    puts $arra(2)
  }
% prnt arr
10
100
```

Parameter passing

```
% proc prnt ar {
      upvar $ar arra
      set i 1
      while {$i<=[array size arra]} {
         puts $arra($i)
         incr i
      }
   }
% prnt arr
10
20
```


End of Tcl, Start of Tk

Widget creation commands

First parameter is a 'dotted' name.

```
% label  <name>   -
optional parameter pairs ...
% canvas <name>   -
optional parameter pairs ...
% button <name>   -
optional parameter pairs ...
% frame  <name>   -
optional parameter pairs ...
% ... and so on
```

The dot heirarchy indicates the relationships between the widgets.

New widget commands

When you create a widget ".b", a new command ".b" is created, which you can use to further communicate with it.

The geometry managers in Tk assemble the widgets:

```
% pack <name> .... where ....
```

Widget becomes command

```
canvas .c
pack .c
button .q -text Quit -command {exit}
pack .q
.q configure -foreground blue
.q configure -background red
```


Example 2

```
foreach r {raised sunken flat groove ridge} {
    frame .$r -width 15m -height 10m -relief $r -
borderwidth 8
    pack .$r -side left -padx 2m -pady 2m
}
.flat configure -background blue
. configure -background gray
```


Example 2

Example 3

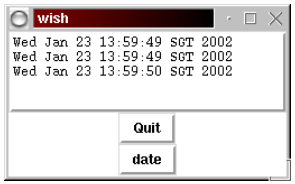
```
radiobutton .cs2105 -text CS2105 -variable course -
value CS2105 -anchor w
radiobutton .cs2106 -text CS2106 -variable course -
value CS2106 -anchor w
pack .cs2105 .cs2106
```

Example 3

Example 3

```
set course CS2106
```


Tcl/Tk example software



Code is

CODE LISTING **SimpleProg.tcl**

```
#!/usr/local/bin/wish8.1 -f

text .log -width 60 -height 5 -bd 2 -relief raised
pack .log
button .buttonquit -text "Quit" -command exit
pack .buttonquit
button .buttondate -text "date" -command getdate
pack .buttondate
proc getdate {} {
   set result [exec date]
   .log insert end $result
   .log insert end \n
}
```

Listbox

Tutorial...

Scrollbars

Tutorial...

Scales

Tutorial...

Tcl/Tk menus

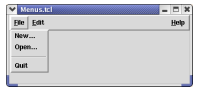
- ✔ Make up a frame for the menu
- ✔ Add in the top level menu items
- ✔ For each top level item, add in the drop-menu items
- ✔ For each nested item, add in any cascaded menus.
- ✔ Remember to pack it...

Tcl/Tk menus

CODE LISTING **Menus.tcl**

```
#!/usr/bin/wish

frame .mbar -relief raised -bd 2
pack .mbar -side top -fill x

frame .dummy -width 10c -height 100
pack .dummy

menubutton .mbar.file -text File -underline 0 -menu .mbar.file.menu
menu .mbar.file.menu -tearoff 0
.mbar.file.menu add command -label "New..." -command "newcommand"
.mbar.file.menu add command -label "Open..." -command "opencommand"
.mbar.file.menu add separator
.mbar.file.menu add command -label Quit -command exit
pack .mbar.file  -side left

menubutton .mbar.edit -text Edit -underline 0 -menu .mbar.edit.menu
menu .mbar.edit.menu -tearoff 1
.mbar.edit.menu add command -label "Undo..." -command "undocommand"
.mbar.edit.menu add separator
.mbar.edit.menu add cascade -label Preferences -menu .mbar.edit.menu.prefs
menu .mbar.edit.menu.prefs -tearoff 0
.mbar.edit.menu.prefs add command -label "Load default" -command "defaultprefs"
.mbar.edit.menu.prefs add command -label "Revert" -command "revertprefs"
pack .mbar.edit  -side left

menubutton .mbar.help -text Help -underline 0 -menu .mbar.help.menu
menu .mbar.help.menu -tearoff 0
.mbar.help.menu add command -label "About ThisApp..." -command "aboutcommand"
pack .mbar.help  -side right

proc aboutcommand {} {
    tk_dialog .win {About this program} "Hugh wrote it!"  {} 0 OK
}
```


Tcl/Tk menus



Tk canvas

- ✔ Draw items
- ✔ Items may be tagged
- ✔ Described in Robert Biddle's "Using the Tk Canvas Facility", a copy of which is found at **~cs3283/ftp/CS-TR-94-5.pdf**.
- ✔ Dynamically created variable names (**node$nodes**).