

TCL / TK

ΕΠΛ 428 - Προγραμματισμός Συστημάτων
Διδάσκων: Δ.Ζειναλιππουρ, Χ.Χρυσοστόμου

Μαριάννα Ζαννεττή, Μαρία Παπά

16-04-2007

Περιεχόμενα

- Ιστορική Αναδρομή
- Πλεονεκτήματα
- Μειονεκτήματα
- Εγκατάσταση
- Παραδείγματα σε TCL/TK
- Συμπεράσματα
- Βιβλιογραφία

Ιστορική Αναδρομή – TCL (1/3)

- Tool Command Language
- Άνοιξη 1988 από τον John Ousterhout, University of California, Berkeley.
- Χρησιμοποιείται για: rapid prototyping, scripted applications, GUIs, testing και CGI scripting.
- Συνδυασμός Tcl και Tk GUI toolkit ⇒ Tcl/Tk

Ιστορική Αναδρομή – Tk (2/3)

- Το Tk αναπτύχθηκε από τον John Ousterhout – επέκταση για την Tcl scripting language.
- Δουλεύει με Perl, Python, Ruby, και Common Lisp.
- Είναι ανεξάρτητη πλατφόρμας.
- Μπορεί να φορτωθεί από το Tcl shell (tclsh) αλλά συνήθως γίνεται χρήση του wish (Windowing Shell).

Ιστορική Αναδρομή (3/3)

Η TCL/TK τρέχει στα εξής λειτουργικά συστήματα:

- Windows 95, NT
- Solaris and SunOS
- Linux
- HP-UX
- SGI IRIX
- Digital Unix, AIX, SCO Unix
- NetBSD, BSDi, FreeBSD
- Most other Unix-like operating systems
- Macintosh (68K and Power Mac)

Πλεονεκτήματα (1/2)

- Πηγαίος κώδικας ελεύθερα διαθέσιμος στο διαδίκτυο.
- Tcl/Tk αρχικά υλοποιήθηκε για X Window συστήματα. Τρέχει με παρόμοιο τρόπο κάτω από μεγάλη ποικιλία πλατφορμών του UNIX συμπεριλαμβανομένου και του *Linux*
Scripts για WINDOWS NT θα τρέξουν σε HP-UX, sgi-IRIX, SunOS, και Linux ⇒ συμβατότητα
- Ανοικτή ανάπτυξη της συσκευασίας.
Χρήστες καθορίζουν λάθη/παραλήψεις, κάνουν προτάσεις και επεκτάσεις στον υπάρχοντα πυρήνα Tcl.
- Καθαρή, καλά τεκμηριωμένη λειτουργική διεπιφάνεια ⇒ εύκολο να επεκταθεί.

Πλεονεκτήματα (2/2)

- Εύκολο στην εκμάθηση σε σύγκριση με C++, FORTRAN, κ.λπ.
- Interpreted γλώσσα, δε χρειάζεται ο κώδικας να μεταγλωττιστεί.
- Ευρεία ποικιλία εργαλείων.
Προγραμματισμός GUI ευκολότερα, ψηλό επίπεδο αφαιρετικότητας για τον προγραμματιστή, υλοποίηση των διεπιφανειών του χρήστη ευκολότερα και γρηγορότερα.

Μειονεκτήματα (1/1)

- Tcl scripts ερμηνεύονται αντί να μεταγλωττιστούν ⇒ περίπου 1000 φορές πιο αργά από μεταγλωττισμένα C scripts.
- Η μηχανή προσομοιωτών γράφτηκε σε C++ και όχι Tcl. Tcl/Tk γλώσσα για το GUI άρα χρειάζονται τρόποι επικοινωνίας μεταξύ των δύο διαφορετικών γλωσσών.
- Έλλειψη ενός πλούσιου συνόλου δομών δεδομένων. Οι μόνοι τύποι στοιχείων στην Tcl:
 - γραμματοσειρές (strings)
 - πίνακες συσχετίσεων (associative arrays).
- Μερικές φορές η σύνταξη είναι μπερδεμένη.

Εγκατάσταση (1/5)

- Για την TCL χρειάζεται να φορτωθούν:
 - ο πυρήνας της γλώσσας (οι βιβλιοθήκες του TCL και TK)
 - οι εφαρμογές wish και tclsh
 - Documentation
 - βιβλιοθήκες script
 - αποδεικτικές εφαρμογές
- Το Wish συμπεριλαμβάνει κουτί εργαλείων με γραφικό περιβάλλον για την TK.
- Όταν ο πυρήνας είναι διαθέσιμος \Rightarrow φορτώνονται οι διαθέσιμοι κώδικες του TCL και του TK.

Εγκατάσταση (2/5)

Tclsh (Τυπικός κώδικας για Hello world):

- 1^{ος} Τρόπος :

Γράφουμε: puts "Hello, world!" και αποθηκεύουμε με το όνομα hello.tcl

tclsh85 hello.tcl

και παρουσιάζεται στην οθόνη το μήνυμα "Hello, world!"

- 2^{ος} Τρόπος :

tclsh85

Γράφουμε: puts "Hello, world!" στην γραμμή εντολών και παρουσιάζεται στην οθόνη το μήνυμα "Hello, world!"

Εγκατάσταση (3/5)

Tclsh (Πως μεταφράζω τον πηγαίο κώδικα και το αποτέλεσμα):

- 1^{ος} Τρόπος :

Γράφουμε τις εντολές σε αρχείο κειμένου με επέκταση tcl

Για να τρέξει γράφουμε στην γραμμή εντολών του terminal:

```
tclsh
```

Ακολουθεί το όνομα το αρχείου στο οποίο έχουμε αποθηκεύσει τις εντολές.

```
tclsh <ονομα.tcl>
```

- 2^{ος} Τρόπος:

Γράφουμε την εντολή:

```
tclsh
```

για να μεταφερθούμε στο κέλυφος του tclsh.

Εκτελούμε μία προς μία τις εντολές που θέλουμε.

Τα αποτελέσματα παρουσιάζονται στο τερματικό.

Εγκατάσταση (4/5)

Wish (γραφικό περιβάλλον, window-based εφαρμογών Tcl, τυπικός κώδικας για Hello world):

- 1^{ος} Τρόπος :

Γράφουμε: `puts "Hello, world!"`

σε ένα αρχείο κειμένου και το αποθηκεύουμε με το όνομα `hello.tcl`

`wish85 hello.tcl`

και παρουσιάζεται στην οθόνη το μήνυμα "Hello, world!"

- 2^{ος} Τρόπος :

`wish85`

Γράφουμε την εντολή : `puts "Hello, world!"`

στην γραμμή εντολών και παρουσιάζεται στην οθόνη το μήνυμα "Hello, world!" και ένα νέο παράθυρο με το όνομα `hello`.

Εγκατάσταση (5/5)

Tclsh (γραφικό περιβάλλον, window-based εφαρμογών Tcl, πως μεταφράζω τον πηγαίο κώδικα και το αποτέλεσμα):

- 1^{ος} Τρόπος :

Γράφουμε τις εντολές σε ένα αρχείο κειμένου με επέκταση tcl στο όνομα του και η πρώτη γραμμή πρέπει να είναι `#!/usr/local/bin/wish` και ζητούμε από τον tcl interpreter να τις εκτελέσει . Για να τρέξει γράφουμε στην γραμμή εντολών του Terminal την εντολή **wish** και ακολουθεί το όνομα το αρχείου στο οποίο έχουμε αποθηκεύσει τις εντολές `<ονομα.tcl>`

- **Δεύτερος τρόπος:**

Γράφουμε την εντολή

wish

για να μεταφερθούμε στο κέλυφος του wish και μετά εκτελούμε μία προς μία τις εντολές που θέλουμε.

Τα αποτελέσματα παρουσιάζονται στο τερματικό, και στο παράθυρο που δημιουργούνται.

Παραδείγματα σε TCL (1/8)

- Κάποιες πολύ απλές εντολές:

- `expr 10 * 5`

Το tclsh θα τυπώσει: 50

- `puts "Hi there"`

Το tclsh θα τυπώσει: Hi There

- `exit`

Για έξοδο

Παραδείγματα σε TCL (2/8)

- word count (χωρίς arguments):

```
proc wc {filename} {
    foreach i {l w c} {
        set $i 0
    }
    set f [open $filename]
    while true {
        set txt [gets $f]
        if [eof $f] break
        incr l
        incr w [regexp -all {[^[:space:]]+} $txt]
        incr c [expr {[string length $txt] + 1}]
    }
    close $f
    return [list $l $w $c]
}
```

Παραδείγματα σε TCL (3/8)

- uniq (υποστηρίζει μόνο -c):

```
proc uniq args {
  ### Parse the arguments
  if {[length $args] && [string equal [lindex $args 0] "-c"]} {
    set count 1
    set args [lrange $args 1 end]
  } else {
    set count 0
  }
  # No args is equivalent to specifying stdin
  if {![length $args]} {set args -}
  set last {}
  set line {}
  set n 0
```


Παραδείγματα σε TCL (4/8)

```
foreach file $args {
    if {[string equal $file "-"]} {
        set f stdin
        set closeme 0
    } else {
        set f [open $file r]
        set closeme 1
    }
    while {[gets $f line] >= 0} {
        if {[string equal $line $last] && $n>0} {
            incr n
        } else {
            if {$count} {
                if {$n>0} {puts [format "%4d %s" $n $last]}
            }
        }
    }
}
```

Παραδείγματα σε TCL (5/8)

```
else {  
    puts $line  
}  
set last $line  
set n 1  
}  
}  
if {$closeme} {close $f}  
}  
if {$count && $n>0} {  
    puts [format "%4d %s" $n $last]  
}  
}
```

Παραδείγματα σε TCL (6/8)

- Απλή επικοινωνία μεταξύ sockets:

```
#!/bin/sh
# \
exec tclsh "$0" "$@"
proc serveConnection {Handle} {
    set LineLength [gets $Handle Line]
    if {$LineLength>=0} {
        #This is where you finally can do something with the data.
        #We simply put it back where it came from.
        puts $Handle "Received: $Line"; flush $Handle
    } elseif {[eof $Handle]} {
        catch {close $Handle}
    }
}
```

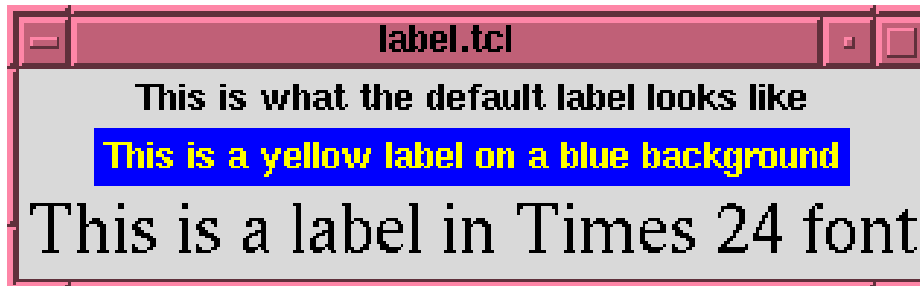
Παραδείγματα σε TCL (7/8)

```
proc acceptConnections {ConnectionFileHandle ClientAddress
    ClientPort} {
    fconfigure $ConnectionFileHandle -blocking 0
    fileevent $ConnectionFileHandle readable [list \
        catch [list serveConnection $ConnectionFileHandle]]
}
socket -server acceptConnections 2000
vwait Dummyvariable
```

Παραδείγματα σε TCL (8/8)

- Μαθηματικές πράξεις (square mean και standard deviation):
 - ```
proc mean2 list { set sum 0 foreach i $list {set sum [expr {$sum+$i*$i}]} expr {double($sum)/[llength $list]} }
```
  - ```
proc stddev list { set m [lavg $list] expr {sqrt([mean2 $list]-$m*$m)} } ;# RS
```

Παραδείγματα σε TK (1/12)



```
#Create three different labels
label .l1 -text "This is what the default label looks like"
label .l2 -text "This is a yellow label on a blue background" \
  -foreground Yellow \
  -background Blue
label .l3 -text "This is a label in Times 24 font" \
  -font {-family times -size 24}
# Put them in the window in row order
grid .l1 -row 0
grid .l2 -row 1
grid .l3 -row 2
```

Παραδείγματα σε TK (2/12)

```
set text Hello
proc dolt {widget} {
    global text
    if {$text == "Hello"} {
        set text "Goodbye"
    } else {
        set text "Hello"
    }
    $widget configure -text $text
}
button .b1 -text "Hello" \
    -command "dolt .b1"
button .b2 -text "Quit" \
    -command "destroy ."
# Put them in the window in row order
grid .b1 -row 0 -column 0
grid .b2 -row 0 -column 1
```

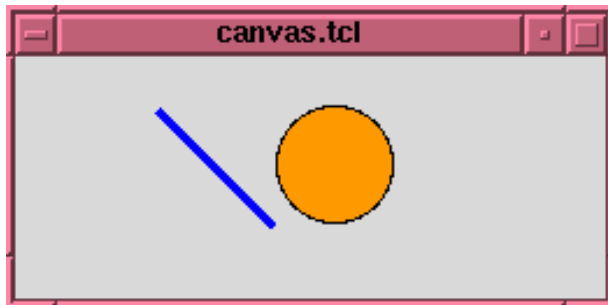


Παραδείγματα σε TK (3/12)



```
# Creates an entry that you can type in.  
# focus puts the cursor in the entry, and the button clears it  
label .l -text "Enter:"  
entry .e -width 40 -relief sunken -bd 2 -textvariable name  
focus .e  
button .b -text Clear -command {set name ""}  
grid .l -row 0 -column 0 -sticky e  
grid .e -row 0 -column 1 -sticky w  
grid .b -row 1 -column 0 -columnspan 2
```

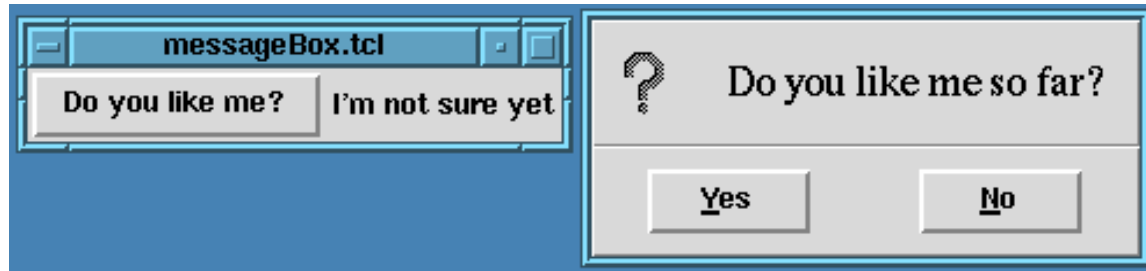

Παραδείγματα σε TK (4/12)



```
# An example of canvases, items, and dragging things around
proc moveit {object x y} {
    .c coords $object [expr $x-25] [expr $y-25] [expr $x+25] [expr $y+25]
}
canvas .c -width 250 -height 100
set myoval [.c create oval 0 0 50 50 -fill orange]
set myline [.c create line 50 50 100 100 -fill blue -width 4]
.c bind $myoval <B1-Motion> {moveit $myoval %x %y}
.c bind $myline <B1-Motion> {moveit $myline %x %y}
grid .c -row 0 -column 0
```

Παραδείγματα σε TK (5/12)

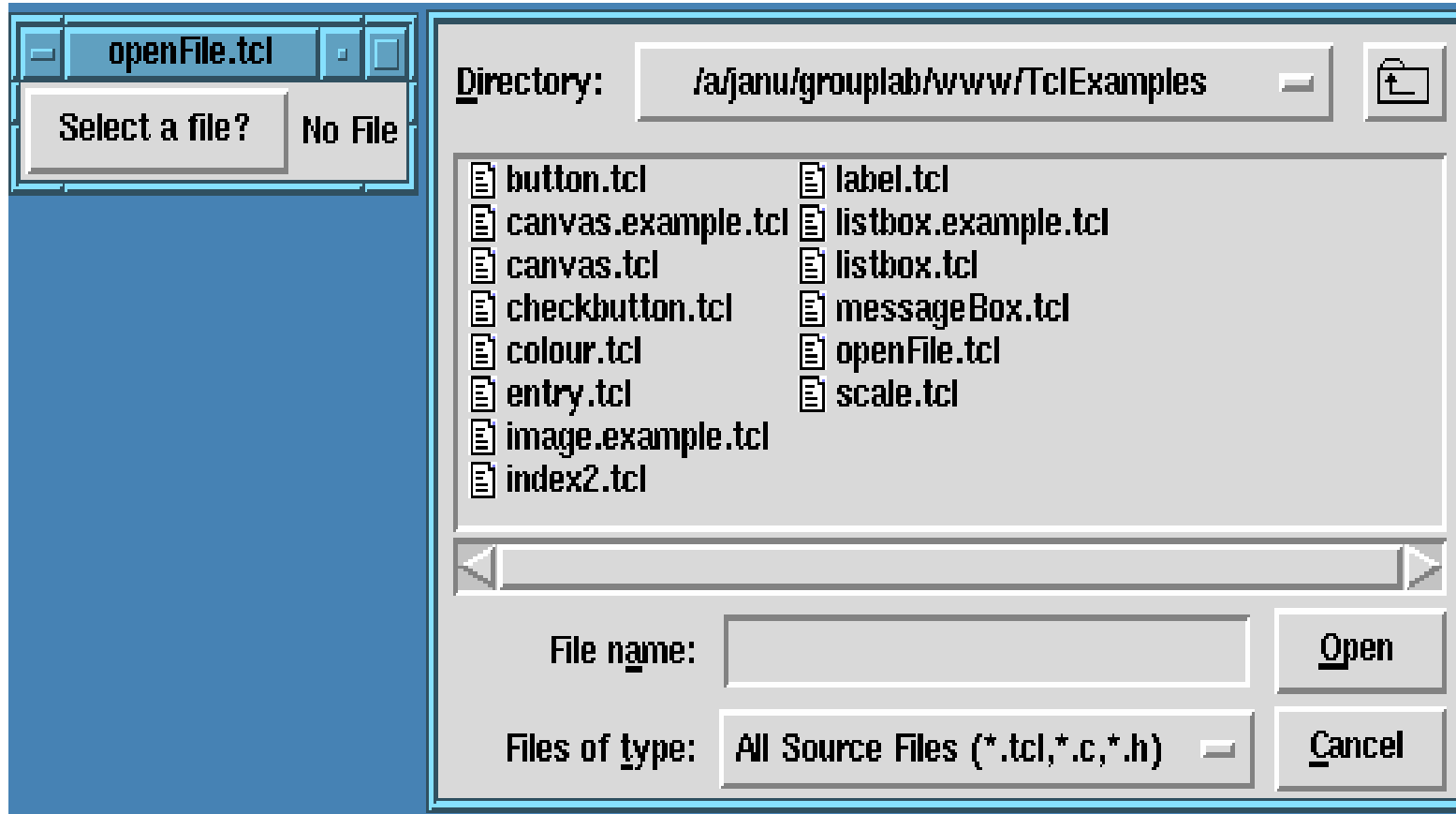
```
proc dolt {label} {  
  set button \  
    [tk_messageBox \  
      -icon question \  
      -type yesno \  
      -title Message \  
      -parent . \  
      -message "Do you like me so far?"]  
  $label configure -text $button  
}  
label .l -text "I'm not sure yet"  
button .b -text "Do you like me?" \  
  -command "dolt .l"  
grid .b -row 0 -column 0  
grid .l -row 0 -column 1
```



Παραδείγματα σε TK (6/12)

```
set types {
    {"All Source Files"  {.tcl .c .h} }
    {"Image Files"      {.gif}      }
    {"All files"        *}
}
proc dolt {label} {
    global types
    set file [tk_getOpenFile -filetypes $types -parent .]
    $label configure -text $file
}
label .l -text "No File"
button .b -text "Select a file?" \
    -command "dolt .l"
grid .b -row 0 -column 0
grid .l -row 0 -column 1
```

Παραδείγματα σε TK (7/12)



Παραδείγματα σε TK (8/12)

```
#!/usr/local/bin/wish
# execlog - run a program with exec and log the output
# Set window title
wm title . ExecLog

# Create a frame for buttons and entry.

frame .top -borderwidth 10
pack .top -side top -fill x

# Create the command buttons.

button .top.quit -text Quit -command exit
set but [button .top.run -text "Run it" -command Run]
pack .top.quit .top.run -side right

# Create a labeled entry for the command

label .top.l -text Command: -padx 0
```

Παραδείγματα σε TK (9/12)

```
entry .top.cmd -width 20 -relief sunken \  
    -textvariable command  
pack .top.l -side left  
pack .top.cmd -side left -fill x -expand true  
  
# Set up key binding equivalents to the buttons  
  
bind .top.cmd <Return> Run  
bind .top.cmd <Control-c> Stop  
focus .top.cmd  
  
# Create a text widget to log the output  
  
frame .t  
set log [text .t.log -width 80 -height 10 \  
    -borderwidth 2 -relief raised -setgrid true \  
    -yscrollcommand {.t.scroll set}]
```

Παραδείγματα σε TK (10/12)

```
scrollbar .t.scroll -command {.t.log yview}
pack .t.scroll -side right -fill y
pack .t.log -side left -fill both -expand true
pack .t -side top -fill both -expand true

# Run the program and arrange to read its input

proc Run {} {
    global command input log but
    if [catch {open "$command & cat"} input] {
        $log insert end $input\n
    } else {
        fileevent $input readable Log
        $log insert end $command\n
        $but config -text Stop -command Stop
    }
}

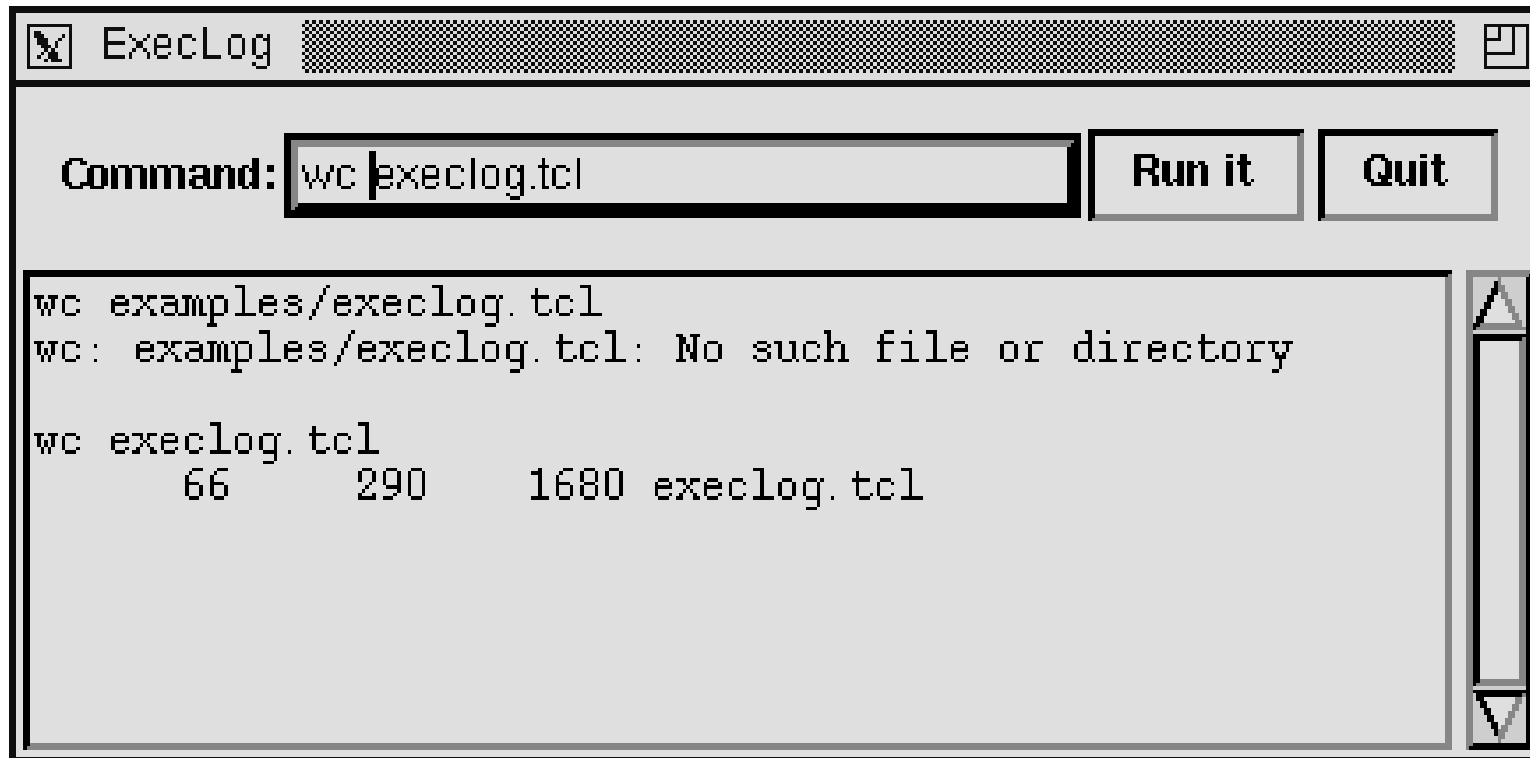
# Read and log output from the program
```

Παραδείγματα σε TK (11/12)

```
# Read and log output from the program
proc Log {} {
    global input log
    if [eof $input] {
        Stop
    } else {
        gets $input line
        $log insert end $line\n
        $log see end
    }
}

# Stop the program and fix up the button
proc Stop {} {
    global input but
    catch {close $input}
    $but config -text "Run it" -command Run
}
```


Παραδείγματα σε TK (12/12)



```
ExecLog  
Command: wc execlog.tcl [Run it] [Quit]  
wc examples/execlog.tcl  
wc: examples/execlog.tcl: No such file or directory  
wc execlog.tcl  
66 290 1680 execlog.tcl
```

Συμπεράσματα (1/1)

- Η TCL/TK είναι ελεύθερα διαθέσιμη στο Internet και είναι εύκολο να μπορεί να την αποκτήσει κάποιος.
- Λόγω της ελεύθερης διάθεσης και μεγάλου αριθμού εγχειριδίων καθίσταται μια από τις πιο εύκολες γλώσσες λογισμικού.
- Η TCL/TK δίνει την δυνατότητα στους χρήστες να δημιουργήσουν γραφικό περιβάλλον πιο εύκολα και με λιγότερο κώδικα.
- Οι χρήστες είναι ελεύθεροι να καθορίσουν λάθη και παραλήψεις και να κάνουν προτάσεις και επεκτάσεις στον πυρήνα Tcl ⇒ δυνατότητα στο χρήστη να την προσαρμόζει στον δικό του τρόπο σκέψης, διαδικασίες του συστήματος μπορούν να τις υλοποιήσουν με δικό τους τρόπο βάσει του τι χρειάζονται αυτοί.
- Το Tcl/Tk είναι συμβατό σύστημα.
- Τα πλεονεκτήματα της γλώσσας είναι περισσότερα από τα μειονεκτήματα της.
- Τα βασικά μειονεκτήματα της είναι ο χρόνος και το μπέρδεμα που καμιά φορά παρουσιάζεται.

Βιβλιογραφία (1/2)

- Extensible Hierarchical Object-Oriented Logic Simulation with an Adaptable Graphical User Interface. Date updated: 8 July 1996. *Advantages of Tcl/Tk*. Available at: <http://www.cs.mun.ca/~donald/msc/node22.html>. Last accessed: Apr. 15, 2007
- Extensible Hierarchical Object-Oriented Logic Simulation with an Adaptable Graphical User Interface. Date updated: 8 July 1996. *Disadvantages of Tcl/Tk*. Available at: <http://www.cs.mun.ca/~donald/msc/node23.html>. Last accessed: Apr. 15, 2007
- TCL. *TCL | Advantages and disadvantages*. Available at: <http://cs1.mcm.edu/~agirrej/CSC4310/advantages.htm>. Last accessed: Apr. 15, 2007
- Hydroinformatics: modelling and information systems for integrated water resources management. *Evaluation of Tcl and Tk Toolkit by development of a graphical user interface for a genetic algorithm*. Available at: http://www.unesco-ihe.org/hi/MSc_abstracts/1997/Fernandez.htm
- WIKIT. Date updated: 12 Oct 2006. *BOOK Tcl and the Tk Toolkit*. Available at: <http://wiki.tcl.tk/103>

Βιβλιογραφία (2/2)

- WIKIT. Date updated: 14 Oct 2004. *Sample Math Programs*. Available at: <http://wiki.tcl.tk/951>
- WIKIT. Date updated: 21 Aug 2006. *Example Scripts Everybody Should Have*. Available at: <http://wiki.tcl.tk/460>
- SICStus Prolog. Date updated: 2007. *What Is Tcl/Tk?* Available at: http://www.sics.se/sicstus/docs/4.0.0/html/sicstus/What-Is-Tcl_002fTk_003f.html
- Tcl/Tk Cookbook. *Getting Started*. Available at: <http://www.bitd.clrc.ac.uk/Publications/Cookbook/gs.html>
- Tips, Hints, and Notes on Data Explorer. *Constructing Tcl/Tk Custom Interactors for Data Explorer*. Available at: <http://www.personal.psu.edu/dept/cac/outreach/dx/dxlink.html>
- Windowing Systems and Toolkits. Date updated: Fall 1998. *Examples of Tk Widgets*. Available at: http://pages.cpsc.ucalgary.ca/~saul/personal/archives/Tcl-Tk_stuff/tcl_examples/
- Tcl Basics. Date updated: 1997. *Tk by Example*. Available at: <http://www.beedub.com/book/2nd/TKEXAMPL.doc.html>

Τέλος Παρουσίασης

ΕΡΩΤΗΣΕΙΣ;