

Εισαγωγή στην Tcl

Τι είναι η Tcl;

Το αρκτικόλεξο Tcl προέρχεται από τις λέξεις «Tool Control Language». Η Tcl είναι μια γλώσσα προγραμματισμού για scripts γενικής χρήσεως, τα οποία επίσης μπορούν να ενσωματωθούν ως βιβλιοθήκες σε προγράμματα άλλων γλωσσών. Δεν μπορεί όμως να υποκαταστήσει άλλες υψηλού επιπέδου γλώσσες. Η Tcl διαθέτει την επέκταση **tk** που επιτρέπει την κατασκευή GUI.

Το συντακτικό της Tcl

Ένα Tcl script αποτελείται από μία ή περισσότερες εντολές. Οι εντολές γράφονται είτε σε ξεχωριστές γραμμές είτε διαχωρίζονται με ερωτηματικά (semicolon) « ; ».

Οι εντολές στην Tcl σχηματίζονται από λέξεις που διαχωρίζονται με κενά. Η πρώτη λέξη είναι το όνομα της εντολής και οι άλλες αποτελούν ορίσματα (arguments) για την εντολή.

```
command arg1 arg2 arg3
```

Ο χαρακτήρας «#» δηλώνει ότι ακολουθεί μη εκτελούμενο σχόλιο.

```
#!/usr/bin/tclsh
# My first Tcl script
puts "This is a test"
>This is a test
```

Η εντολή «**puts**» εμφανίζει στην οθόνη (εάν δεν δηλώνεται κάποιο αρχείο) οτιδήποτε περικλείεται με εισαγωγικά.

```
#!/usr/bin/tclsh
set a Hello
set b world
puts "$a $b"
>Hello world
```

Η εντολή «**set**» δίνει στην μεταβλητή «**a**» το περιεχόμενο «**Hello**» και στην μεταβλητή «**b**» το περιεχόμενο «**world**». Η εντολή «**puts**» εμφανίζει στην οθόνη ομαδοποιημένα ό,τι αντιπροσωπεύουν οι μεταβλητές «**a**» και «**b**». Για να μεταφερθεί το περιεχόμενο των μεταβλητών πρέπει να προηγηθεί το «**\$**». Αυτό

λέγεται υποκατάσταση (substitution). Τα εισαγωγικά `` ` ` επιτρέπουν υποκαταστάσεις στο εσωτερικό τους.

Η εντολή χειρίζεται τα ορίσματα ανάλογα με τον τρόπο οργάνωσής τους. Τα ορίσματα μπορεί να διαχωρίζονται απλώς με κενά ή να ομαδοποιούνται με `` ``, [], () και { }.

Ο interpreter της Tcl επεξεργάζεται τις εσώκλειστες εντολές (nested) με σειρά από μέσα προς τα έξω.

```
#!/usr/bin/tclsh
set b 42; set a [expr $b+2]; puts $a
>44
```

Ο interpreter εκλαμβάνει την πρώτη λέξη μέσα σε ορθογώνιες παρενθέσεις [] ως μια εντολή της οποίας το αποτέλεσμα θα αποτελέσει όρισμα για την εξωτερική. Εδώ η εντολή «**expr**» υπολογίζει το άθροισμα «**\$b+2**» και η «**set**» αποδίδει το αποτέλεσμα στην μεταβλητή «**a**». Η εντολή «**expr**» δέχεται ως όρισμα μια έκφραση (μαθηματική ή σύγκρισης).

Οι αγκύλες { } δεν επιτρέπουν υποκατάσταση στο εσωτερικό τους όταν είναι εξωτερικές:

```
#!/usr/bin/tclsh
set b 42; set a {expr $b+2}; puts $a
> expr $b+2
```

Όταν όμως είναι εσωτερικές:

```
#!/usr/bin/tclsh
set b 42; set a [expr {$b+2}]; puts $a
>44
```

for και while loops στην Tcl

Το «**for**» συντάσσεται με 4 ορίσματα τα οποία δηλώνονται με αγκύλες { }:

```
for arg1 arg2 arg3 arg4
```

Εδώ οι αγκύλες επιτρέπουν υποκατάσταση. Η πρώτη αγκύλη εκτελείται ως εντολή Tcl. Η δεύτερη αγκύλη ως μια συνθήκη που είτε αληθεύει (1) είτε όχι (0). Για όσο αληθεύει η συνθήκη εκτελείται επανειλημμένα η τέταρτη αγκύλη ως εντολή Tcl.

Ακολουθως εκτελείται η τρίτη αγκύλη ως εντολή Tcl και επαναλαμβάνεται η εκτέλεση της τέταρτης αγκύλης.

```
#!/usr/bin/tclsh
for {set i 0} {$i < 10} {incr i} {
puts "$i [expr sqrt($i)]"
}
```

Ομοίως το «while» συντάσσεται με δύο ορίσματα τα οποία δηλώνονται με αγκύλες { }:

```
while arg1 arg2
```

Το πρώτο όρισμα είναι μια συνθήκη, η οποία είτε αληθεύει (1) είτε όχι (0). Το δεύτερο όρισμα είναι μια εντολή Tcl η οποία εκτελείται επανειλημμένα για όσο αληθεύει η συνθήκη του πρώτου ορίσματος.

```
#!/usr/bin/tclsh
set i 0
while {$i < 5} {
    puts "tetragono tou $i = [expr pow($i,2)]"
    incr i
}
```

Εάν το αποτέλεσμα της εκτέλεσης του script θέλουμε να γραφεί σε ένα αρχείο με το όνομα «test_out.txt», τότε πρέπει να το «ανοίξουμε» προηγουμένως με την εντολή:

```
set outfile [open test_out.txt w];
```

Στο τέλος του script πρέπει να «κλείσουμε» το αρχείο με την εντολή:

```
close $outfile
```

```
#!/usr/bin/tclsh
# Script open-close file
set outfile [open test_out.txt w];
set i 0
while {$i < 5} {
    puts $outfile "tetragono tou $i = [expr pow($i,2)]"
    incr i
}
close $outfile
```

Έλεγχος ροής με «if»

Το «if» συντάσσεται στην Tcl με δύο ορίσματα. Το πρώτο αποτελεί μια συνθήκη που δηλώνεται με αγκύλες { } ή με ορθογώνιες παρενθέσεις []. Το δεύτερο όρισμα δηλώνεται με αγκύλες και είναι μια εντολή Tcl η οποία εκτελείται εφόσον αληθεύει η συνθήκη του πρώτου ορίσματος.

```
#!/usr/bin/tclsh
if {3+5<10} {
puts sigatoprama
}
```

Το «if» μπορεί να συμπληρώνεται με το «else» προκειμένου να εκτελεσθεί ένα εναλλακτικό Tcl script εάν δεν αληθεύει η πρώτη συνθήκη.

```
#!/usr/bin/tclsh
set price 200
if [expr $price > 100] {
    puts "Too expensive"
}
else {
    puts "Price is acceptable"
}
```

Αριθμητικοί και λογικοί (Boolean) τελεστές (operators)

**	Υψώνει σε δύναμη
+, -, *, /, %	Αριθμητικές πράξεις
==, !=, <, >, <=, >=	Συγκρίσεις
&&	Λογικό AND (καί το ένα και το άλλο)
	Λογικό OR (είτε το ένα είτε το άλλο)
!	Μετατρέπει ό,τι ακολουθεί στο αντίθετό του

```
#!/usr/bin/tclsh
set x 3
set y 8

puts [expr $x == $y]
puts [expr $y > $x]

if {$y > $x} {
    puts "y is greater than x"
}

> y is greater than x
```

```
#!/usr/bin/tclsh
set a 2
if {
    $a == 1
    || $a == 2
    || $a == 3
} then {
    puts "a is one, two or three"
}
> a is one, two or three
```

Μερικές ενσωματωμένες μαθηματικές συναρτήσεις

abs (x)	Απόλυτη τιμή
acos (x)	Αντίστροφο συνημίτονο (σε ακτίνια)
asin (x)	Αντίστροφο ημίτονο (σε ακτίνια)
atan (x)	Αντίστροφη εφαπτομένη (σε ακτίνια)
cos (x)	Συνημίτονο (το όρισμα σε ακτίνια)
exp (x)	Εκθετική συνάρτηση
int (x)	Μετατροπή αριθμού σε ακέραιο
log (x)	Φυσικός (νεπέριος) λογάριθμος
log10 (x)	Δεκαδικός λογάριθμος
pow (x, y)	Ύψωση του αριθμού x στην δύναμη y
rand ()	Δίνει έναν τυχαίο αριθμό στο διάστημα [0,1)
round (x)	Στρογγυλοποίηση στον πλησιέστερο ακέραιο
sin (x)	Ημίτονο (το όρισμα σε ακτίνια)
sqrt (x)	Τετραγωνική ρίζα
tan (x)	Εφαπτομένη (το όρισμα σε ακτίνια)

Παράδειγμα χρήσης της Tcl για χειρισμό βιολογικών μορίων μέσω του VMD

```
#!/usr/bin/tclsh
mol load pdb test_pdb.pdb
set sar [atomselect top "chain A and within 3 of water"]
$sar writepdb sar.pdb
mol load pdb sar.pdb
```

Ένα πολύ χρήσιμο κομμάτι της Tcl περιλαμβάνει την χρήση των «lists». Δυστυχώς στο πλαίσιο αυτού του σύντομου εισαγωγικού μαθήματος δεν θα ασχοληθούμε με αυτό το αντικείμενο.