Ticol Quickstart Guide

The recommended script editor is Notepad++

Command Line Help

```
ticol.exe /?
```

Running a Script File

```
ticol.exe <scriptname>
```

Running a Script File With Arguments

```
ticol.exe <scriptname> arg1 arg2 ...
```

Running a Script File Without autoexec.tcl

```
ticol.exe <scriptname> /na
```

Entering the Command Line Interface (CLI)

```
ticol.exe
```

Run a Script from the CLI

```
run <scriptname> ?<args>?
```

```
load <scriptname>
run
```

View a Tcl Script from the CLI

```
load <scriptname>
dump
```

Get Help from the CLI

```
help <topic>
```

```
find <topic>
```

Run a Script with Single Step Debugging

```
ticol.exe <scriptname> /bp
```

(and enter at least one: **halt** command into the script)

Run Tcl Commands from the Windows Console

```
ticol.exe ; "<quoted>;<tcl>;<commands>;<separated>;<by>;<semicolons>"
```

Protect a Script

```
ticol.exe <scriptname> /c
```

View Preprocessed Source Code (Unprotected)

```
ticol.exe <scriptname> /echo
```

Examples

```
ticol.exe
puts "Hello world"
exit
```

```
ticol.exe
puts [expr 22/7.0]
exit
```

```
ticol.exe
load hanoi
run
exit
```

```
ticol.exe
load hanoi
dump
exit
```

```
ticol.exe
help call by name
exit
```

```
ticol.exe
find print
exit
```

```
ticol.exe
run hanoi 17
exit
```

```
ticol.exe hanoi 17 /na
```

```
ticol.exe hanoi 17 /g /na
ticol.exe hanoi /c

ticol.exe hanoi /echo

ticol.exe ; "set a 4; puts [expr 4*atan(1)]"

ticol.exe ; "set a 4; puts \"Pi is [expr 4*atan(1)]\""
```

Important Points to Note

i. Tcl 'functions' are actually [commands] and may take -arguments

ii. Tcl looks a little like C/C++ but the Tcl syntax requires brace openings to be on the same line as the opening command. This is one of the few rules of Tcl syntax

```
if {1} {
        # Do something
} else {
        # Do nothing
}
```

iii. Tcl comments are defined by hash characters. Ticol also offers /* ... */

iv. Commands are wrapped in square brackets inside statements or unwrapped if standalone

v. Square brackets are evaluated first, even inside strings

vi. Braces delay or prevent evaluation of commands. Each command (function) call removes one layer of braces

vii. Tcl understands nothing whatsoever about "C-like" expressions. The [expr] command processes these either standalone or in flow control commands depending on the setting for [option expression]

viii. You can have flow control commands evaluate in Tcl [expr] mode or in Tcl command mode using [option expression]. Get this wrong and loops will hang!

```
option expression on
while {$i<10} { ...}

option expression off
while {[< $i 10]} { ...}
```

ix. Tcl has functions such as abs() similar to C/C++ but these are available from the expression handler '[expr]'. Ticol also allows you to call these via [funct]

```
puts [expr "round(4*atan(1),3)"]
```

x. Ticol has an [option] command which may be used to change behaviour on the fly

```
option
option expression
option expression on
option expression off
```

-O-