Tcl Variable Utilities

Version 0.1.1

Alex Baker

https://github.com/ambaker1/vutil

April 26, 2023

Abstract

When running a parametric study on an Tcl file, the parameters must be configured so that they can be modified from a top-level. The simplest way to achieve this is by commenting-out lines which specify the parameter within the main file. This package provides two additional methods:

1. **The default value method:** This method requires modification to the main file. It simply assigns default values to variables if the variables do not exist.
2. **The lock method:** This method requires no modification to the main file, it simply allows for variables to be set and locked. Any attempts to set the variable will be reversed.

Additionally, the vutil package provides garbage collection for TclOO objects, so that when a variable tied to an object goes out of scope or is modified, the corresponding object is destroyed.

Default Values

A common technique to handle default values is to use the *info exists* command in Tcl. This method is formalized with the vutil command *default*, which simply assigns default values to variables if the variables do not exist. Variables set using the *default* command will be available to override from a top-level.

```
default $varname $value
```

$varname	Name of variable to set
$value	Default value for variable

The example below shows how default values are only applied if the variable does not exist.

Example 1: Variable defaults

Code:

```
set a 5
default a 7
puts $a
unset a
default a 7
puts $a
```

Output:

```
5
7
```

Variable Locks

The lock method uses Tcl variable traces to prevent any further modification to variables while locked. This method does not require any modification of the main file. The main command for this method, *lock*, locks variables for editing. Any attempts to modify a locked variable will be reversed.

```
lock $varName $value
```

`$varName`	Variable name to lock.
`$value`	Value to lock variable at.

The command *unlock* unlocks previously locked variables so that they can be modified again.

```
unlock $var1 $var2 ...
```

`$var1 $var2 ...`	Variables to unlock.

Example 2: Variable locks

Code:

```
lock a 5
set a 7
puts $a
unlock a
set a 7
puts $a
```

Output:

```
5
7
```

Variable-Object Ties

As of Tcl version 8.6, there is no garbage collection for Tcl objects, they have to be removed manually with the "destroy" method. The command *tie* is a solution for this problem, using variable traces to destroy the corresponding object when the variable is unset or modified. Tie is separate from lock; a tie will override a lock, and a lock will override a tie.

```
tie $varName $object
```

`$varName`	Variable name to tie to object.
`$object`	Object to tie variable to.

In similar fashion to *unlock*, tied variables can be untied with the command *untie*.

```
untie $var1 $var2 ...
```

`$var1 $var2 ...`	Variables to untie.

Example 3: Variable-object ties

Code:

```
oo::class create foo {
    method hi {} {
        puts hi
    }
}
tie a [foo create bar]
set b $a; # alias variable
unset a; # triggers ``destroy''
$b hi; # throws error
```

Output:

```
invalid command name "::bar"
```