Visual TCL

- vTcl -

Tutorial

- DRAFT -

Last modified: 12 May 1997
Written by Rick Macdonald

Stewart Allen
Larry W. Virden
Convertet to html by Poul-Erik Andreasen 30 August 2000 Your feedback is welcome!

Table of contents

Document scope

This document is about using vTcl.

The scope of this document relates to beginners and people who like to get a feel for vTvl.

For Information on using vTcl in a session to draft a user program see the vTcl-User-Guide.

VTcl Tutorial

Welcome to the vTcl Tutorial!

We'll guide you through some simple examples that demonstrate several basic tasks to get you up and running with vTcl.

1 Before we can start

First we need to make sure we're speaking the same language and working in the same environment:

- software versions.
- terminology and naming conventions used in this tutorial.
- vTcl user preferences.

1.1 Software Versions

If you have older or newer versions of the software, you may find that some of the descriptions don't match what you see. Hopefully, you'll still be able to work through the tutorials.

These tutorials are based on the following versions:

- tcl 8.x
- tk 8.x
- vTcl 1.6 or later

1.2 Terminology and Naming Conventions for this Tutorial

vTcl has several main windows. We'll refer to them by the abbreviations given in the table below. Some of the vTcl windows are transient. Your window manager might not decorate them with Title Bars. The Window menu in the main menu lets you open and close all of the vTcl main windows.

Abbrev	Window Title Bar	Short Description
[Attr]	Attribute Editor	attribute editor for the active widget
[Cons]	Visual Tcl Console	command console
[Func]	Functions	list of functions you've defined

[Geom]	Attribute Editor	geometry parameters for the active widget
[Menu]	Visual Tcl	main menu: File, Edit, Mode, etc
[Tool]	Widget Toolbar	palette of widget icons
[Tops]	Toplevel Windows	list of top level windows you've created
[Tree]	Widget Browser	widget tree diagram of your application
[Vars]	Variables	list of global variables you've defined
[Widg]	Attribute Editor	attribute editor for the active widget

Note that [Widg] and [Geom] are actually sections within the [Attr] window.
Further, we'll refer to a widget within these windows with the convention "[Abbrev]:widget". This reference means to actually click the specified widget with the left mouse button. The widget may be a button, entry field or a sequence of menu items. In the case of the Widget Toolbar which just contains icons, we give the text of the balloon help. For example:

[Menu]:File/New	- select the "New" item in the "File" menu.
[Tool]:button	- select the button widget in the Widget Toolbar.
[Attr]:text	- click in the entry field of the "text" item in the Attribute Editor.
[Attr]:command(...)	- click the "..." button for the "command" item in the Attribute Editor.

Generally, you need to click on an entry field in a vTcl window to make it active for entry, and you need to press ENTER for the change to take effect.
There are several auxiliary windows in vTcl for command and function editing, setting aliases, etc. Here are the windows that we'll be using:

Abbrev	Window Title Bar	Short Description
[Alias]	Widget Alias for ...	entry dialog for widget aliases
[Comm]	Command for ...	button and menu edit window
[Edit]	(toplevel name)	function edit window
[Pref]	Preferences	vTcl preferences

Finally, we'll refer to the windows that _you_ create by names that we'll assign as they are created. For example:

[Hello]	the window that you create in tutorial #1
[Goodbye]	- the window that you create in tutorial #2

A very important window is the Widget Info window, or [Widg], which is the top section of the Attribute Editor. It shows the following:

- Widget - the name of the selected widget. Except for toplevels, the active widget is also highlighted with little square blocks in the corners.
- Class - the type of widget or frame.

- Manager - the geometry manager of the active widget.
- Alias - an alias that you can assign to refer to this widget. Aliases are stored in an array called "widget".
- Insert - widgets selected from the Widget Toolbar, or [Tool], will be inserted into this widget or frame.

1.3 vTcl user preferences

vTcl supports several user preference settings in the Preferences window, [Pref], which you get from [Menu]:File/Preferences. You'll have to set yours to match what we used here or you'll have difficulties following the tutorials. If you find your version of vTcl has more preferences than are defined here, try leaving them in the default state.

The preferences in these examples were the defaults at the time that this tutorial was written. Feel free to customize vTcl preferences as you like after you've finished these tutorials.

The Basics

[X] Use Balloon Help
[ ] Ask for Widget name on insert
[X] Short automatic widget names
[ ] Save verbose widget configuration
[ ] Save global variable values
[ ] Window focus selects window

Default Geometry Manager

( ) Grid ( ) Pack (X) Place

Widget Option Encapsulation

(X) List ( ) Braces ( ) Quotes

2 Tutorial #1

The first time a task is presented, there are extra comments and instructions. As these basics are repeated, the instructions become more terse.

1) Start vtcl.

Four windows open: [Menu], [Widg], [Func], [Attr].

You'll probably need to move them around so that you can see them all at once. I like to place the [Attr] window on the right side of the screen and stretch it to be full height. There's a lot of info in this one window. Also, depending on the font size that you use, you may need to stretch the width of this window to see the little controls on the right edge. For example, [Attr]:command has a button to the right of the entry field with three dots on it.

2) Check that [Menu]:File/Preferences match the list above.

3) Make sure that you also have the [Tops] window showing.

Turn it on with [Menu]:Window/Toplevels.
Note that [Menu]:Window lets you open and close all of the vTcl main windows.

4) Create a window.

a - click [Tool]:toplevel
It's the top left square with the blue strip across the top.

Note the balloon help when you hold the mouse still over an icon for a couple of seconds! Have a look at the balloon help for each icon now to get familiar with what they are.
Note also that [Widg], [Attr] and [Geom], which are actually all in the Attribute Editor window, filled up with info and options for the toplevel window just created. The information in these windows changes as you select different widgets by clicking on them with the left mouse button to make them active.

5) Change the title of the window.

a - click [Geom]:title
b - erase the default title: New Toplevel 1
c - type our new title: Hello
d - press ENTER.

Note that the title bar of your new window changes, as does the name in the [Tops] window.

6) Create a button in the window.

a - click [Tool]:button
Note the button created in the Hello window!

7) Change the text on the button.

a - click [Attr]:Text
b - erase: button
c - type: Say Hello
d - press ENTER
Note that the text of the button changes to "Say Hello".

8) Give the button something to do when it's clicked.

a - click [Attr]:Command(...)
Note the [Comm] editor window that pops up
b - type in the [Comm] editor window: puts "Hello, World!"
c - click [Comm]:OK

9) Move the button around to make it look nice.

The "Place" geometry manager allows you to simply drag and drop widgets in the window to reposition them. Don't drag the little black squares in the corners: they resize the button itself. Click and drag the center of the button.

In your [Hello] window,

a - Try moving your "Say Hello" button to the middle of the window. Also try resizing the whole window itself using the normal window manager methods (dragging window corners and edges) to make it all look nice.

10) Save your program to a disk file.
a - click [Menu]:File/SaveAs
b - type the name hello1.tcl in the Selection fiel
c - click Save or press Enter.

It's now ready to run by executing "wish hello1.tcl", or "chmod 755 hello1.tcl" and execute it directly as "hello1.tcl". On UNIX systems, the message is written to the xterm where you execute the program. If you're on some other platform and you don't see the message anywhere, just go on to the next tutorial where we add a pop-up window for the message.

3 Tutorial #2

Tutorial #1 can be improved by popping up a hello window rather than just writing to stdout.

If you are continuing from the same vTcl session as Tutorial #1, you're ready to go. Otherwise, [Menu]:File/Close any existing application then [Menu]:File/Open the file "hello1.tcl" from Tutorial #1.

Besides creating and hiding a second window, this tutorial introduces the very useful "alias" feature. You name widgets and toplevels and then refer to them by name using the "widget" array. This avoids having to use the long complicated names that are generated.

1) Make sure that you have these windows showing:

    [Menu], [Widg], [Func], [Attr], [Tops].
    Turn on any missing windows with [Menu]:Window.

2) Create a second toplevel window.

    a - click [Tool]:toplevel

3) Change the title of the window.

    a - click [Geom]:title
    b - erase the default title: New Toplevel 2
    c - type our new title: Goodbye
    d - press ENTER

4) Give the button an alias for easy reference.
    a - click [Menu]:Options/SetAlias
        Note the little alias entry window pops up.
    b - type: msg
    c - click [Alias]:OK

5) Put a message in the window.
    a - click [Tool]:message
        Note that a message widget is added to the active window.
6) Change the message text to the familiar friendly greeting.

    a - click [Attr]:Text
    b - erase: message
    c - type: Hello, World!
    d - press ENTER

7) Change the aspect ratio attribute to put the message all on one line.

a - click [Attr]:Aspect
a - change 150 to some bigger number (1500 will do)
b - press ENTER
Note how the message all fits on one line now.

8) Now add a button to the window.

a - click [Tool]:button
Note that the button is placed on top of the message, but we'll move it in a minute.

9) Change the text on the button.

a - click [Attr]:Text b - erase: button
c - type: Goodbye
d - press ENTER

10) Move the button and message around to make it all look nice.

In the [Goodbye] window,

a - Drag the "Goodbye" button down so it's not covering the "Hello" message.
b - Arrange the placement of the two widgets and resize the window itself using the normal window manager controls to make it all look nice.

11) Make sure the Goodbye button is the active widget.

a - click [Goodbye]:Goodbye By clicking the "Goodbye" button, you "select" it for editing.
Note the little black squares in the corners to indicate this. If you just finished dragging the "Hello" message, the "Goodbye" button wouldn't be the active (selected) widget.

12) Give the Goodbye button a command to hide the [Goodbye] window.

a - click [Attr]:Command(...)
b - type in the [Comm] window: Window hide $widget(msg)
c - click [Comm]:OK

13) Position the goodbye window.

Drag the [Goodbye] window (by the window manager title bar) to place it where you want it to pop up when you execute the program. vTcl enters this geometry placement into your application, but not all window managers will honor this placement.

14) Hide the [Goodbye] window.

In the [Tops] window,

a - select Goodbye in the listbox
b - click [Tops]:Hide

When you click [Menu]:File/Save to save your applications, make sure that you have your toplevel windows showing or hidden as you would want them to be when a user runs your application. In our program, if you save it with the "Goodbye" window showing, the user will get both windows at startup.

15) Give the "Hello" button the command to show the second window.

In the [Hello] window,

a - click the "Hello" button to make it active.
b - click [Attr]:Command(...)
c - erase: puts "Hello, World!"
d - type: Window show $widget(msg)
e - click [Comm]:OK

16) Save your program to a disk file.

a - click [Menu]:File/SaveAs
b - type the name hello2.tcl in the Selection field.
c - click Save or press Enter.

It's now ready to run by executing "wish hello2.tcl", or "chmod 755 hello2.tcl" and execute it directly as "hello2.tcl".
Click "Say Hello", and the Goodbye window pops up. Click "Goodbye" and the window goes away again.

4 Tutorial #3

If you are continuing from the same vTcl session as Tutorial #2, you're ready to go. Otherwise, [Menu]:File/Close any existing application then [Menu]:File/Open the file "hello2.tcl" from Tutorial #2.

In [Menu]:Mode are items that toggle between EDIT and TEST mode. We've been in EDIT mode up to this point. TEST mode allows you to actually run your application from within vTcl! The current mode is shown on the bottom line of the [Menu] window. This indicator is in fact a convenience button that you can click to toggle between the two modes.

1) Change to TEST mode to test the program from within vTcl.
    a - click [Menu]:Mode/TestMode
        Note that the indicator in the [Menu] window changes to show TEST. Now you can click the "Hello" and "Goodbye" buttons and see that the application works without leaving vTcl!