There are many interpreted script GUI frameworks, languages, toolkits. Many are used for Web applications / services and for System management utilities.

TK (or the Tkinter, for TK interface) is a simple toolkit used with Python, Ruby, TCL (tool command language, original) ... Languages that do not have a GUI library. Open source, cross platform: Linux, OSX, Windows, embedded, ...

Examples use tcl/tk ("tickle-tk")

TK has variables

```python
x = StringVar()  # Holds a string; default value ""
x = IntVar()     # Holds an integer; default value 0
x = DoubleVar()  # Holds a float; default value 0.0
x = BooleanVar() # Holds a boolean, returns 0 for False and 1 for True
```
Widgets

Widget names and hierarchy

All widgets in an application exist in a hierarchy from root ".". Dialog widgets are toplevel widgets – but are also in the hierarchy.
Create Widgets

Each widget occupies a window, has a class (use) and options that affect its appearance and behavior—attributes such as fonts, colors, sizes, text labels, and such. e.g., `text='PANIC!'` or `height=20`.

frame (manager) relief, borderwidth, color, foreground, background (all widgets have these options)

grid (manager) treats every window or frame as a table—a gridwork of rows and columns, spanning allows widget to cross cell boundaries

label (display text, bitmaps) text, font, bitmap

button, checkbutton, radiobutton (respond to button events) text, variable or value (radiobutton sets), command (callback procedure)

The Spinbox widget allows the user to select values from a given set. The values may be a range of numbers, or a fixed set of strings.

message (display multiline strings) width, aspect, justify, text
listbox (collection of strings -- entries)
scanning (mouse 2 drag moves entries like scrollbar),
commands (entry insert, delete, get (retrieve))

scrollbar (cont rolls view in other widgets)
yscroll, xscroll, command, orient (default vertical)

scale (edit variable values via slider) horizontal || vertical
length, from, to, tickinterval, command

entry (text edit) commands (insert, delete) index, icursor,
textvariable, width

menubutton (buttons on menubar, menus are added to menubuttons).

menu (menubar selections, entries that can be added) entries are:
command, checkbutton, radiobutton, cascade, separator.
menus also have post. (with dialogs can be pop ups).
underline (keyboard accel), label, text, color, accelerator
canvas (drawing area) create (circles, rectangles, lines, text, bitmaps, widgets), tags and identifiers (ways to manipulate canvas items with events)

text (text field display and edit multiline text) insert, delete ...

Tk widget Demonstration program, part of the Active TCL/TK community distribution. (with code)
**Widget attributes (sample)**

```python
w = WidgetName ( parent, option, ... )
```

- **anchor**: Text position in widget: N, NE, E, SE, S, SW, W, NW, or CENTER. The Default is CENTER.
- **aspect**: Aspect ratio, given as the width/height relation in percent.
- **background or bg**: The background color of the message widget
- **borderwidth or bd**: Border width. Default value is 2.
- **cursor**: Defines the cursor to show when the mouse is over widget.
- **font**: Message font. The default value is system specific.
- **foreground or fg**: Text color. The default value is system specific.
- **highlightbackground**: Together with highlightcolor and highlightthickness, this option controls how to draw the highlight region.
- **justify**: Align multiple lines of text. Use LEFT, RIGHT, or CENTER. Default is LEFT.
- **padx, pady**: Horizontal, Vertical padding. Default is -1 (no padding).
- **text**: Message text. The widget inserts line breaks if necessary to get the requested aspect ratio. (text/Text)
- **width**: Widget width in character units.
Geometry – pack

Tk uses a parent/child terminology (older terms "master/slave") window management.

pack is the packer command (do layout mgmt).

Options are:

  after or before widget
  expand boolean (1, default is false 0) slave absorbs extra space
  fill describes how to expand (x, y, both, default is none)
  in determines who master is (default is widget's parent)
  master must be parent or descendant of parent.
  ipadx, ipady specifies distance for internal padding inside slave (0)
  padx, pady specifies external padding (outside slave inside container)
  side specifies attachment (top (default), bottom, left or right)

Frames are used for hierarchical packing layouts.
Events – bind

Events can be bound to widgets.
   Key, KeyPress, KeyRelease, Button, ButtonPress, ButtonRelease, Enter, Leave, Motion,

Example button event bindings for .b (tcl/tk)

```python
bind .b <Enter> {.b config -state active}
bind .b <Leave> {.b config -state normal}
```

Event Patterns  <modifier of event>

Modifiers:  Control, Shift, Lock, Any, Meta, Button1..5, B1..5, Double, Triple

- `<Any-B1-Motion>` any button1 motion
- `<KeyPress-a>` pressing key a

Actions bound to event
Script substitution enables event specific information to be returned to application. there are 30 substitutions ..

%\(x\), %\(y\)    \(x\) and \(y\) coordinates of event
%\(W\)             path name of event window
%\(K\)             keysym from event (KeyPress...)
%\(A\)             8 bit iso character value of KeyPress
%%                  substitute the \% character

bind .aCanvas <ButtonPress> {
    set xpoint %\(x\)
    set ypoint %\(y\) }

update is a command that enables tcl scripts to escape from a procedure, process all waiting events, and then return to script.

tkwait command can be used to force an application to wait for a response from a window before continuing (modality)

tkwait window .dlg
# Python 3.4 and Tkinter 8.5
# mash-up from http://www.python-course.eu/python_tkinter.php
# Mike Barnes 11/23/2014

# from Tkinter import * # for Python 2*
from tkinter import * # for Python 3*

# Python variable and function definition
# waits 1000 milliseconds to increment and display counter
counter = 0
def counter_label(label):
    counter = 0
def count():
    global counter
    counter += 1
    label.config(text=str(counter))
    label.after(1000, count)

count()

root = Tk()  
root.title("Hello Comp 585")

logo = PhotoImage(file="Hi.gif")
explanation = """At present, only GIF and PPM/PGM formats are supported."""

print(explanation)
Label(root, compound = TOP, # BOTTOM, LEFT, RIGHT, CENTER, 
text = explanation, image = logo).pack(side = "right")

helvetica = Label(root, text = "Helvetica 16", 
fg = "light green", bg = "dark green", 
font = "Helvetica 16 bold italic").pack(fill=X, pady=20)

verdana = Label(root,text="Verdana 10 bold", fg = "blue", 
bg = "yellow", font = "Verdana 10 bold").pack(fill=X, pady = 30)

label = Label(root, fg = "orange", font = "Courier 24 bold")
label.pack()

counter_label(label)    # call the counter_label function

button = Button(root, text='Stop', width = 10, 
command=root.destroy)
button.pack(side = "bottom")

root.mainloop()
References


Tcl Developer Xchange, http://www.tcl.tk/