Gura モジュールリファレンス – tcl, tk

Updated: June 25, 2012

copyright © 2011-2012 Yutaka SAITO

目次

1. このリファレンスについて 3

2. オーバービュー 4

2.1. モジュール構成 4

2.2. 命名規則 4

2.3. イベントの扱い 4

3. tclモジュール 6

3.1. モジュール関数 6

3.2. tcl.interpクラス 6

3.2.1. インスタンスの生成 6

3.2.2. インスタンスメソッド 6

3.3. tcl.variableクラス 6

3.3.1. インスタンスプロパティ 6

3.4. tcl.timerクラス 6

3.4.1. インスタンスメソッド 6

4. tkモジュール 7

4.1. モジュール関数 7

4.2. imageクラスの拡張 7

4.2.1. インスタンスメソッド 7

4.3. tk.Widgetクラス 8

4.3.1. インスタンスの生成 8

4.3.2. インスタンスメソッド 9

4.4. tk.Menuクラス 12

4.4.1. インスタンスメソッド 12

4.5. tk.Canvasクラス 13

4.6. tk.CanvasItemクラス 13

4.7. tk.CanvasTagクラス 13

4.8. tk.Textクラス 13

4.9. tk.TextTagクラス 13

4.10. tk.Notebookクラス 13

4.11. tk.Treeviewクラス 13

4.12. tk.TreeviewItemクラス 13

4.13. tk.TreeviewTagクラス 13

4.14. tk.FontTクラス 13

4.15. tk.Imageクラス 13

4.16. 注意事項 13

このリファレンスについて

本リファレンスはGura のtkモジュールで定義されている関数やクラスの仕様について説明します。

Tcl/Tkのオフィシャルサイトはhttp://www.tcl.tk/ です。

オーバービュー

モジュール構成

モジュールは、Tclインタープリタとのインターフェースを提供するモジュールtcl、そのモジュールを使ってTkの機能を実現するモジュールtkからなります。

Tcl/Tkには多くの関数がありますが、tclやtkの中でそれらの関数を個別に実装しているわけではありません。ユーザが呼び出したメソッド名と引数をもとにしてtclのコマンドを生成し、tclインタープリタに渡して実行しています。

命名規則

ウィジェットを生成する手続きは、親ウィジェットを表す変数のメソッドとして実装されます。

ウィジェットを生成する手続きは、最初の文字を大文字にした名前のメソッドになります。例えば、canvasはCanvasになります。

名前空間ttkに属する手続きは、"ttk$" につづけて、最初の文字を大文字にした名前をつけたメソッドになります。例えば、ttk::button はttk$Buttonになります。

イベントの扱い

Tcl/Tk は、イベントが発生すると関連付けられたプロシージャを評価します。この際、プロシージャ中にパーセント記号 "%" と一文字のアルファベットまたは記号が記述されていると、評価の前にその部分をイベントに付随するパラメータ値に置換します。例えば、マウスをクリックしたときに発生するButtonPressイベントが発生すると、関連付けたプロシージャ中の %x や %y という部分をそれぞれマウスのx, y 座標数値に置換してから評価が行われます。置換文字列の一覧は<http://www.tcl.tk/man/tcl8.5/TkCmd/bind.htm> にまとめられています。

Guraはブロックでイベントハンドラを指定できますが、イベントのパラメータ値はブロック引数として受け取ります。このとき、ブロック引数の名前が、Tcl/Tk の置換文字列の名前に対応します。上の例を使えば、ブロック引数にx, y という名前の引数を指定すると、これらにx, y 座標値が入ります。一般の関数呼び出しの慣例では引数の位置によって受け渡しする値を指定しますが、Tcl/Tkのイベントハンドラでは名前で受け取る値が決まることに注意してください。

また、Tcl/Tkが置換する結果は文字列になるので、数値など他の方で受け取りたい場合はブロック引数に明示的に型指定をしてください。

ブロック引数では、Tcl/Tkが定義する一文字の識別子のほかに、可読性を高めるため以下の別名も受け付けられるようにしています。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 引数名 | Tcl/Tk |  | 引数名 | Tcl/Tk |  | 引数名 | Tcl/Tk |
| serial | # |  | place | p |  | root | R |
| above | a |  | state | s |  | subwidget | S |
| numbuttons | b |  | time | t |  | type | T |
| count | c |  | width | w |  | widget | W |
| detail | d |  | x | x |  | x\_root | X |
| user\_data | d |  | y | y |  | y\_root | Y |
| focus | f |  | char | A |  | action | d |
| height | h |  | border\_width | B |  | index | i |
| widget\_num | i |  | delta | D |  | wouldbe | P |
| keycode | k |  | send\_event | E |  | current | s |
| mode | m |  | keysym | K |  | edited | S |
| override\_redirect | o |  | keysym\_num | N |  | validate | V |

この対応づけは、tk.bindSubstDict変数に辞書として登録されており、必要に応じて追加することができます。例えば、%hで置換されるイベントパラメータをhogeという引数名で受け取りたい場合は、以下のようなコードを追加します。

|  |
| --- |
| tk.bindSubstDict[`hoge] = "h" |

# tclモジュール

## モジュール関数

tcl.Tk\_MainLoop()

tcl.interpクラス

### インスタンスの生成

tcl.interp()

### インスタンスメソッド

tcl.interp#command(func:function)

tcl.interp#eval(objs+)

tcl.interp#evalscript(script:string)

tcl.interp#timer()  
tcl.timerインスタンスを生成します。

tcl.interp#variable(value?, varName?:string)  
tcl.variableインスタンスを生成します。

tcl.variableクラス

### インスタンスプロパティ

tcl.timerクラス

### インスタンスメソッド

tcl.timer#cancel()

tcl.timer#start(msec:number, msecCont?:number, count?:number):reduce {block}

# tkモジュール

モジュール関数

tk.bell(args\*, opts%):map

tk.bind(args\*, opts%):map

tk.bindtags(args\*, opts%):map

tk.tkerror(args\*, opts%):map

tk.update()

tk.winfo$atom(args\*, opts%):map

tk.winfo$atomname(args\*, opts%):map

tk.winfo$containing(args\*, opts%):map

tk.winfo$interps(args\*, opts%):map

tk.winfo$pathname(args\*, opts%):map

tk.tk$FocusFollowsMouse(args\*, opts%):map

tk.tk$SetPalette(args\*, opts%):map

tk.tk$bisque(args\*, opts%):map

tk.tkwait$variable(args\*, opts%):map

tk.tkwait$visibility(args\*, opts%):map

tk.tkwait$window(args\*, opts%):map

tk.mainloop()

tk.tclDump(flag:boolean => true)

imageクラスの拡張

### インスタンスメソッド

tkモジュールをインポートすると、imageクラスに以下のメソッドが追加されます。

image#to\_tk()

tk.Widgetクラス

### インスタンスの生成

例えば、buttonウィジェットを生成するメソッドの一般式は以下のようになります。

tk.Widget#Button(options%)

このメソッドは、Tclコマンドとして"button pathName %options%" を実行します。pathNameは新しく生成するbuttonウィジェットのパス名、optionsはtk.Widget#Buttonメソッドに辞書形式で渡した引数の内容が渡されます。

ウィジェット生成メソッドの戻り値は、新しく生成したウィジェットのパス名を持ったtk.Widgetインスタンスです。menuウィジェットやtextウィジェットのように、ウィジェット特有のメソッドを持つものについては、tk.Widgetクラスから派生したクラスのインスタンスを返します。

ウィジェット生成メソッドと、生成するTkウィジェットの対応表を以下にまとめます。

|  |  |  |
| --- | --- | --- |
| メソッド | 生成するTkウィジェット | メソッドの戻り値 |
| tk.Widget#Button | button | tk.Widget |
| tk.Widget#Canvas | canvas | tk.Widget |
| tk.Widget#Checkbutton | checkbutton | tk.Widget |
| tk.Widget#Entry | entry | tk.Widget |
| tk.Widget#Frame | frame | tk.Widget |
| tk.Widget#Label | label | tk.Widget |
| tk.Widget#Labelframe | labelframe | tk.Widget |
| tk.Widget#Listbox | listbox | tk.Widget |
| tk.Widget#Menu | menu | tk.Menu |
| tk.Widget#Menubutton | menubutton | tk.Widget |
| tk.Widget#Message | message | tk.Widget |
| tk.Widget#Panedwindow | panedwindow | tk.Widget |
| tk.Widget#Radiobutton | radiobutton | tk.Widget |
| tk.Widget#Scrollbar | scrollbar | tk.Widget |
| tk.Widget#Spinbox | spinbox | tk.Widget |
| tk.Widget#Text | text | tk.Text |
| tk.Widget#Toplevel | toplevel | tk.Widget |
| tk.Widget#ttk$Button | ttk::button | tk.Widget |
| tk.Widget#ttk$Checkbutton | ttk::checkbutton | tk.Widget |
| tk.Widget#ttk$Combobox | ttk::combobox | tk.Widget |
| tk.Widget#ttk$Entry | ttk::entry | tk.Widget |
| tk.Widget#ttk$Frame | ttk::frame | tk.Widget |
| tk.Widget#ttk$Intro | ttk::intro | tk.Widget |
| tk.Widget#ttk$Label | ttk::label | tk.Widget |
| tk.Widget#ttk$Labelframe | ttk::labelframe | tk.Widget |
| tk.Widget#ttk$Menubutton | ttk::menubutton | tk.Widget |
| tk.Widget#ttk$Notebook | ttk::notebook | tk.Notebook |
| tk.Widget#ttk$Panedwindow | ttk::panedwindow | tk.Widget |
| tk.Widget#ttk$Progressbar | ttk::progressbar | tk.Widget |
| tk.Widget#ttk$Radiobutton | ttk::radiobutton | tk.Widget |
| tk.Widget#ttk$Scale | ttk::scale | tk.Widget |
| tk.Widget#ttk$Scrollbar | ttk::scrollbar | tk.Widget |
| tk.Widget#ttk$Separator | ttk::separator | tk.Widget |
| tk.Widget#ttk$Sizegrip | ttk::sizegrip | tk.Widget |
| tk.Widget#ttk$Spinbox | ttk::spinbox | tk.Widget |
| tk.Widget#ttk$Style | ttk::style | tk.Widget |
| tk.Widget#ttk$Treeview | ttk::treeview | tk.Treeview |
| tk.Widget#ttk$Widget | ttk::widget | tk.Widget |
| tk.Widget#ttk$Image | ttk::image | tk.Widget |
| tk.Widget#ttk$Vsapi | ttk::vsapi | tk.Widget |

### インスタンスメソッド

tk.Widget#wm$aspect(minNumber?, minDenom?, maxNumber?, maxDenom?):map  
Tclコマンド: wm aspect window ?minNumber mindenom maxNumber maxDenom

tk.Widget#wm$attributes(opts%):map  
Tclコマンド: wm attributes window

tk.Widget#wm$client(name?):map  
Tclコマンド: wm client window ?name?

tk.Widget#wm$colormapwindows(windowList?):map  
Tclコマンド:

tk.Widget#wm$command(value?):map  
Tclコマンド:

tk.Widget#wm$deiconify():map  
Tclコマンド:

tk.Widget#wm$focusmodel(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$forget():map  
Tclコマンド:

tk.Widget#wm$frame():map  
Tclコマンド:

tk.Widget#wm$geometry(newGeometry):map  
Tclコマンド:

tk.Widget#wm$grid(baseWidth?, baseHeight?, widthInc?, heightInc?):map  
Tclコマンド:

tk.Widget#wm$group(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$iconbitmap(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$iconify(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$iconmask(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$iconname(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$iconphoto(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$iconposition(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$iconwindow(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$manage(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$maxsize(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$minsize(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$overrideredirect(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$positionfrom(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$protocol(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$resizable(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$sizefrom(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$stackorder(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$state(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$title(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$transient(args\*, opts%):map  
Tclコマンド:

tk.Widget#wm$withdraw():map  
Tclコマンド:

tk.Widget#winfo$(args\*, opts%):map  
Tclコマンド:

tk.Widget#grid$(args\*, opts%):map  
Tclコマンド:

tk.Widget#place$(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$bisque(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$chooseColor(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$chooseDirectory(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$dialog(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$focusFollowsMouse(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$focusNext(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$focusPrev(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$getOpenFile(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$getSaveFile(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$menuSetFocus(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$messageBox(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$optionMenu(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$popup(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$setPalette(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$textCopy(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$textCut(args\*, opts%):map  
Tclコマンド:

tk.Widget#tk$textPaste(args\*, opts%):map  
Tclコマンド:

tk.Menuクラス

### インスタンスメソッド

tk.Menu#Separator(opts%)

tk.Menu#Cascade(opts%) {block?}

tk.Menu#Command(opts%) {block?}

tk.Menu#Checkbutton(opts%) {block?}

tk.Menu#Radiobutton(opts%) {block?}

tk.Canvasクラス

tk.Canvas#Tag() {block?}

tk.CanvasItemクラス

tk.CanvasTagクラス

tk.Textクラス

tk.TextTagクラス

tk.Notebookクラス

tk.Treeviewクラス

tk.TreeviewItemクラス

tk.TreeviewTagクラス

tk.FontTクラス

tk.Imageクラス

注意事項

Tkイメージオブジェクトは、明示的にメソッドdestroy() を実行しないとメモリから削除されません。