

Package ‘tcltk’

October 12, 2016

Version 3.3.1

Priority base

Title Tcl/Tk Interface

Author R Core Team

Maintainer R Core Team <R-core@r-project.org>

Description Interface and language bindings to Tcl/Tk GUI elements.

License Part of R 3.3.1

Imports utils

R topics documented:

tcltk-package	1
TclInterface	2
tclServiceMode	6
TkCommands	7
tkpager	11
tkProgressBar	11
tkStartGUI	13
TkWidgetcmds	13
TkWidgets	16
tk_choose.dir	18
tk_choose.files	18
tk_messageBox	19
tk_select.list	20
Index	22

tcltk-package *Tcl/Tk* インタフェイス

Description

Tcl/Tk の GUI 要素へのインタフェイスと言語連携.

Details

このパッケージはプラットフォーム独立な Tcl スクリプト言語と Tk GUI 要素へのアクセスを提供する。サポートされるウィジェットのリストは [TkWidgets](#) を、それらと共に動作する命令に対しては [TkWidgetcmds](#)、それ以上はこれらのファイルの参考文献を見よ。

Tcl/Tk のドキュメントは Windows ではディレクトリ '[R_HOME](#)/Tcl/doc' に、Unix ではシステムのマンページにある。

関数の完全なリストは `ls("package:tcltk")` を使う。

(Unix の場合) Tk は DISPLAY 変数が設定されていなければ初期化されないが、Tcl は依然として使うことが出来る。これは実際にはそれを使わないセッション中で **tcltk** に依存するパッケージのロードを許すのに最も役に立つ(例えばインストールの過程で)。

Author(s)

R Core Team

管理者 : R Core Team <R-core@r-project.org>

TclInterface

低水準の *Tcl/Tk* インタフェイス

Description

これらの関数と変数は R と Tcl インタプリタと Tk GUI ツールキットの間の基本的な連携機能を提供する。Tk ウィンドウは R オブジェクトにより表現可能である。Tcl 変数はクラス `tclVar` のオブジェクトによりアクセス可能で、Tcl への C レベルのインタフェイスはクラス `tclObj` のオブジェクト経由でアクセスできる。

Usage

```
.Tcl(...)
.Tcl.objv(objv)
.Tcl.args(...)
.Tcl.args.objv(...)
.Tcl.callback(...)
.Tk.ID(win)
.Tk.newwin(ID)
.Tk.subwin(parent)
.TkRoot

tkdestroy(win)
is.tkwin(x)

tclvalue(x)
tclvalue(x) <- value

tclVar(init = "")
## S3 method for class 'tclVar'
as.character(x, ...)
## S3 method for class 'tclVar'
tclvalue(x)
```

```
## S3 replacement method for class 'tclVar'
tclvalue(x) <- value

tclArray()
## S3 method for class 'tclArray'
x[...]
## S3 replacement method for class 'tclArray'
x[...] <- value
## S3 method for class 'tclArray'
x$i
## S3 replacement method for class 'tclArray'
x$i <- value

## S3 method for class 'tclArray'
names(x)
## S3 method for class 'tclArray'
length(x)

tclObj(x)
tclObj(x) <- value
## S3 method for class 'tclVar'
tclObj(x)
## S3 replacement method for class 'tclVar'
tclObj(x) <- value

as.tclObj(x, drop = FALSE)
is.tclObj(x)

## S3 method for class 'tclObj'
as.character(x, ...)
## S3 method for class 'tclObj'
as.integer(x, ...)
## S3 method for class 'tclObj'
as.double(x, ...)
## S3 method for class 'tclObj'
as.logical(x, ...)
## S3 method for class 'tclObj'
as.raw(x, ...)
## S3 method for class 'tclObj'
tclvalue(x)

## Default S3 method:
tclvalue(x)
## Default S3 replacement method:
tclvalue(x) <- value

addTclPath(path = ".")
tclRequire(package, warn = TRUE)
tclVersion()
```

Arguments

objv	Tcl オブジェクトの名前付きベクトル
win	ウィンドウ構造
x	オブジェクト
i	文字列または(引用符なしの)名前
drop	論理値. 単一要素のベクトルが単一の Tcl オブジェクトにされるべきか, または長さ1のリストにすべきかを指示する
value	tclvalue 付値に対しては文字列. tclObj 付値に対してはクラス tclObj のオブジェクト
ID	ウィンドウの ID
parent	結果のウィンドウの親になるウィンドウ
path	Tcl パッケージを含むディレクトリへのパス
package	Tcl パッケージ名
warn	論理値. もし見つからなければ警告するか?
...	追加引数. 下を見よ.
init	初期化値

Details

これらの関数の多くは一般的な使用は意図されておらず, Tk ウィジェットと Tcl オブジェクトを作り操作する命令により内部的に使われる. 最も低いレベルでは `.Tcl` は命令を文字列として Tcl インタプリタに送り結果をクラス `tclObj` のオブジェクトとして返す(下を見よ). より新しい変種の `.Tcl.objv` は `tclObj` の名前付きリストの型式での引数を受け付ける.

`.Tcl.args` は `tag = value` 対の `R` の引数リストを Tcl の `-option value` スタイルに変換するので, 二つの言語間の単純な翻訳を可能にする. 先立つオプションフラグの無い値を送るには, 単にタグ付きでない引数を使う. 先立つ値のないオプションが必要な稀なケースでは `tag = NULL` を使うことが出来る. 殆どの値は単に文字列モードに変換され命令文字列に挿入されるが, ウィンドウオブジェクトはそれらの ID 文字列を使い渡され, コールバックは `.Tcl.callback` の結果を使い渡される. タグは単純に `-` を先頭につけてオプションフラグに変換される.

`.Tcl.args.objv` は `.Tcl.args` と同じような役割を果たすが, `.Tcl.objv` に渡すのに適当な `tclObj` オブジェクトのリストを生成する. リストの名前は内部的に `.Tcl.objv` を使い Tcl オプションに変換される.

コールバックは `.Tcl.callback` により処理される原子的なコールバックまたは表現式が可能である. 表現式は原子的なコールバックのリストとして扱われるが以下の例外がある: もし要素が名前ならそれは最初呼び出し側のフレームで評価され, そしてもしそれが明示的な関数定義の時も同様である; `break` 表現式は直接 Tcl の対応物に翻訳される. `.Tcl.callback` は `R` 関数と未評価の呼び出しを Tcl の命令文字列に変換する. 引数は関数クロージャか環境が続くモード `"call"` のオブジェクトでなければならない. 最初のケースでの戻り値は `R_call 0x408b94d4` の型式で16進数は関数のメモリアドレスである. 二番目のケースではそれは `R_call_lang 0x8a95904 0x819bfd0` の形式になる. 表現式に対しては似たような項目の系列が生成され, セミコロンで分離される. `.Tcl.args` は関数や呼び出しが指定されたアドレスに存在し続けているかどうかをコールバックを関連するウィンドウ環境に代入することで確認する特別な用心を行う(下を見よ).

Tk ウィンドウはクラス `tkwin` のオブジェクトとして表現され, これは ID 欄と大局的環境中で囲まれる `R` の環境である `env` 欄を含むリストである. ID 欄の値は Tk ウィンドウ

の名前と同じである。env 環境は parent 変数と num.subwin 変数を含む。もしウィンドウが副ウィンドウとコールバックを受け取ると、それらは変数として環境に付け加えられる。TkRoot は ID "." を持つトップウィンドウである；このウィンドウはそれをウィンドウマネージャの制御で閉じるまじい効果为了避免するため表示されない。parent 変数は TkRoot に対しては定義されていない。

.Tk.ID はウィンドウの ID を取り出し、.Tk.newwin は与えられた ID の新しいウィンドウ環境を作り、そして .Tk.subwin は与えられた親ウィンドウの副ウィンドウである新しいウィンドウを作る。

tkdestroy はウィンドウを破壊し、そしてまたその親からウィンドウへの参照を除く。

is.tkwin は与えられたオブジェクトがウィンドウ環境かどうかをテストするのに使うことが出来る。

tclVar は新しい Tcl 変数を作りそれを init に初期化する。クラス tclVar の R オブジェクトがそれを表現するために作られる。オブジェクトに対して as.character を使うと Tcl 変数名を返す。R から Tcl 変数へのアクセスは tclvalue 関数を使ってなされ、これはまた付値の左辺側でも起きる。もし tclvalue が tclVar オブジェクトで無い引数に渡されると、それは Tcl 変数を明示的に命名する文字列であると仮定される。tclVar で作られた Tcl 変数はユニークに命名され表現するオブジェクトが最早使用されないと自動的にガベージコレクションで未設定にされる。

tclArray は新しい Tcl 配列を作りそれを空の配列に初期化する。クラス tclArray でクラス tclVar を継承する R オブジェクトがそれを表現するために使われる。Tcl 配列の要素には [] や \$ を使った添字操作でアクセスすることが出来、これは又置き換え形式も許す。Tcl 配列は原理的に結合的であり従って順序が付かないことを注意しよう；数値添字 i は名前 as.character(i) の要素を参照する。多重添字はコンマで分離された単一の名前に繋げられる。配列の長さや配列中の名前名のセットはそれぞれ length と names を用いて問い合わせることが出来る；これらは意味があるようには設定出来ず、従って付値形式はエラーメッセージをプリントするためだけに存在する。

Tcl の 'dual-ported' オブジェクトに直接アクセスすることが可能で、それによりそれらの文字列表現を構文解析したり逆構文解析することを避けることが出来る。これはクラス tclObj のオブジェクトを使ってなされる。そうしたオブジェクトの文字列表現は tclvalue を使い取り出すことが出来(しかし設定できない)、そしてモード "character", "double", "integer", "logical" そして "raw" のベクトルへの変換は as.character のような標準の強制変換を使い実行できる。逆にそうしたベクトルは as.tclObj を使い変換できる。長さ 1 のベクトルに対して何が起るかについては曖昧さがあり、drop 引数で制御される；相違が Tcl に対して意味を持つケースがあるが、殆どの場合それはそれらを同等に扱う。tclvalue と as.character はその文字列表現が埋め込みのスペースを持つオブジェクトに関して相違がある。前者は時おり好まれ、特に tclread, tkgetOpenFile そして同様の関数の結果に適用される時が該当する。as.raw メソッドはバイト型ベクトルかバイト型ベクトルのリストを返し、Tcl からバイナリデータを返すのに使うことが出来る。

tclVar オブジェクトの背後にあるオブジェクトは tclObj(x) を使い取り出され、これは又付値型式を許し付値の右辺値は自動的に as.tclObj を使い強制変換される。tclObj オブジェクトに対するプリントメソッドがある；これは後にオブジェクトの文字列表現が続く <Tcl> をプリントする。tclVar オブジェクトに対する as.character は対応する Tcl の名前を返しその値ではない。

Tcl パッケージは tclRequire でロードできる；addTclPath でそれらが見つかるディレクトリを Tcl の検索パスに加えることが必要かもしれない。返り値はもしそれが成功すればクラス "tclObj" のオブジェクトで、失敗すれば FALSE である(警告が発せられる場合)。現在の検索パスを R の文字列ベクトルとして見るためには次を使う

```
strsplit(tclvalue('auto_path'), " ")[1]
```

Tcl バージョン(パッチレベルを含む)は文字列 ("8.6.3" のような)で返される。

Note

釣り合わない括弧を含む文字列は現在多くの状況で上手く処理されない。

See Also

[TkWidgets](#), [TkCommands](#), [TkWidgetcmds](#).

Tcl/Tk のサポートがこの R のビルトにコンパイルされているかどうかを見るには [capabilities\("tcltk"\)](#).

Examples

```
tclVersion()

## Not run:
## これらは example() で実行できないがtcltk パッケージをロードした
## 対話的な R セッション中にペーストすれば実行できる
.Tcl("format \"%s\n\" \"Hello, World!\")
f <- function()cat("HI!\n")
.Tcl.callback(f)
.Tcl.args(text = "Push!", command = f) # NB: 異なったアドレス

xyzy <- tclVar(7913)
tclvalue(xyzy)
tclvalue(xyzy) <- "foo"
as.character(xyzy)
tcl("set", as.character(xyzy))

top <- tktoplevel() # Tk ウィジェット, Tk-widgets を見よ
ls(envir = top$env, all = TRUE)
ls(envir = .TkRoot$env, all = TRUE) # .Tcl.args はここで callback ref を置く

## End(Not run)
```

tclServiceMode

Tcl のイベントのサービスをするかどうか

Description

この関数は Tcl のサービスモードを制御したり報告したりする、つまり Tcl がイベントに対応するかどうか。

Usage

```
tclServiceMode(on = NULL)
```

Arguments

on (論理値) イベントサービスをオンにするかどうか。

Details

もし `on == NULL` (既定) で呼び出されると何も変更は怒らない。

これは他のパッケージからのウィジェットを含み全ての Tcl/Tk の行動をブロックすることを注意する。ウィンドウのマッピングを個別に処理するほうがよいかもしれない。

Value

呼び出し前の Tcl のサービスモード。

Examples

```
## 例えば demo(tkcanvas) を見よ
## Not run:
oldmode <- tclServiceMode(FALSE)
# 格好の良い図を作る作業をする。
# 次が実行されるまで何も起きない...
tclServiceMode(oldmode)

## End(Not run)
## 別のアイデアは tkwm.withdraw() ... tkwm.deiconify() を使う
```

TkCommands

ウィジェットでない Tk 命令

Description

これらの関数はウィンドウマネージャインタフェース命令や幾何マネージャのような、ウィジェットでない Tk 命令へのインタフェースである

Usage

```
tcl(...)
tktitle(x)

tktitle(x) <- value

tkbell(...)
tkbind(...)
tkbindtags(...)
tkfocus(...)
tklower(...)
tkraise(...)

tkclipboard.append(...)
tkclipboard.clear(...)

tkevent.add(...)
tkevent.delete(...)
tkevent.generate(...)
tkevent.info(...)
```

```
tkfont.actual(...)
tkfont.configure(...)
tkfont.create(...)
tkfont.delete(...)
tkfont.families(...)
tkfont.measure(...)
tkfont.metrics(...)
tkfont.names(...)
```

```
tkgrab(...)
tkgrab.current(...)
tkgrab.release(...)
tkgrab.set(...)
tkgrab.status(...)
```

```
tkimage.create(...)
tkimage.delete(...)
tkimage.height(...)
tkimage.inuse(...)
tkimage.names(...)
tkimage.type(...)
tkimage.types(...)
tkimage.width(...)
```

```
## 注意：ある種のウィジェットはまた選択とクリア命令を持つ、
## 従って "X".
```

```
tkXselection.clear(...)
tkXselection.get(...)
tkXselection.handle(...)
tkXselection.own(...)
```

```
tkwait.variable(...)
tkwait.visibility(...)
tkwait.window(...)
```

```
## winfo は実際には多数の副命令を持つがめったに使われない、
## 従って代わりに tkwinfo("atom", ...) 等を使う。
```

```
tkwinfo(...)
```

```
# ウィンドウマネージャインタフェイス
```

```
tkwm.aspect(...)
tkwm.client(...)
tkwm.colormapwindows(...)
tkwm.command(...)
tkwm.deiconify(...)
tkwm.focusmodel(...)
tkwm.frame(...)
tkwm.geometry(...)
tkwm.grid(...)
```

```
tkwm.group(...)
tkwm.iconbitmap(...)
tkwm.iconify(...)
tkwm.iconmask(...)
tkwm.iconname(...)
tkwm.iconposition(...)
tkwm.iconwindow(...)
tkwm.maxsize(...)
tkwm.minsize(...)
tkwm.overrideredirect(...)
tkwm.positionfrom(...)
tkwm.protocol(...)
tkwm.resizable(...)
tkwm.sizefrom(...)
tkwm.state(...)
tkwm.title(...)
tkwm.transient(...)
tkwm.withdraw(...)
```

幾何マネージャ

```
tkgrid(...)
tkgrid.bbox(...)
tkgrid.columnconfigure(...)
tkgrid.configure(...)
tkgrid.forget(...)
tkgrid.info(...)
tkgrid.location(...)
tkgrid.propagate(...)
tkgrid.rowconfigure(...)
tkgrid.remove(...)
tkgrid.size(...)
tkgrid.slaves(...)
```

```
tkpack(...)
tkpack.configure(...)
tkpack.forget(...)
tkpack.info(...)
tkpack.propagate(...)
tkpack.slaves(...)
```

```
tkplace(...)
tkplace.configure(...)
tkplace.forget(...)
tkplace.info(...)
tkplace.slaves(...)
```

標準ダイアログ

```
tkgetOpenFile(...)
tkgetSaveFile(...)
tkchooseDirectory(...)
```

```
tkmessageBox(...)
tkdialog(...)
tkpopup(...)
```

```
## ファイル処理関数
tclfile.tail(...)
tclfile.dir(...)
tclopen(...)
tclclose(...)
tclputs(...)
tclread(...)
```

Arguments

x	ウィンドウオブジェクト
value	付値 tktitle に対しては文字列.
...	.Tcl.args により処理

Details

tcl は任意の Tk または Tcl 命令を単に引数リストに対して .Tcl.args.objv を実行し結果を .Tcl.objv に渡すことで呼び出す一般的なインタフェイスを提供する. 他の命令の殆どは単に tcl を特定の第一引数, そしてある場合には副命令を与える第二引数で呼び出す.

tktitle とその付値形式は Tk の wm title への別のインタフェイスを提供する.

これらの命令は極端に多数あり, それらとそれらの引数を完全に説明することは困難である. Tcl/Tk のドキュメントを参照して欲しい. 少数の例外を除き pack configure の様な Tk の副命令が tkpack.configure, そして Tcl 副命令は tclfile.dir の様な関数名に強制変換される,

See Also

[TclInterface](#), [TkWidgets](#), [TkWidgetcmds](#)

Examples

```
## Not run:
## これらは examples() では実行できないが
## tcltk パッケージをロードした対話的な R セッション中に
## ペイストすれば実行できる

tt <- tktoplevel()
tkpack(l1 <- tklabel(tt, text = "Heave"), l2< - tklabel(tt, text = "Ho"))
tkpack.configure(l1, side = "left")

## ウィンドウの引き伸ばしを試みそしてそれから

tkdestroy(tt)

## End(Not run)
```

tkpager	<i>Tk</i> のテキストウィジェットを使いファイルを閲覧する
---------	-----------------------------------

Description

これは `file.show` に接続され、ファイルを別ウィンドウで表示する。

Usage

```
tkpager(file, header, title, delete.file)
```

Arguments

<code>file</code>	表示するファイル名をふくむ文字列
<code>header</code>	各ファイルに使われるヘッダー
<code>title</code>	ウィンドウに対して使われる共通タイトル. <code>header</code> と連結されて実際のウィンドウタイトルを作る.
<code>delete.file</code>	論理値. 表示後にファイルを削除するか?

Note

下線に使われる文字列 "`\b_`" は現在黙って取り除かれる. フォントと背景の色は現在 `Courier` と `gray90` に固定されている.

See Also

[file.show](#)

tkProgressBar	<i>Tk</i> による進行度バー
---------------	--------------------

Description

`Tk` の進行度バーウィジェットを表示する.

Usage

```
tkProgressBar(title = "R progress bar", label = "",  
              min = 0, max = 1, initial = 0, width = 300)
```

```
getTkProgressBar(pb)  
setTkProgressBar(pb, value, title = NULL, label = NULL)  
## S3 method for class 'tkProgressBar'  
close(con, ...)
```

Arguments

title, label	文字列で、それぞれウィンドウのタイトルとダイアログボックス上のラベルを与える。
min, max	進行度バーの極値に対する(有限)数値。
initial, value	進行度バーに対する初期値または新しい値。
width	ピクセル単位の進行度バーの幅：ダイアログボックスは40ピクセル広い(フレームがプラスされる)。
pb, con	クラス "tkProgressBar" のオブジェクト。
...	総称的関数との一貫性のため。

Details

tkProgressBar はラベルと進行度バーを含むウィジェットを表示する。

setTkProgressBar は値と、 NULL でない値に対してタイトルとラベルを更新する (ウィジェットが作られた時あれば)。 value の欠損値(NA)と範囲外の値は (黙って)無視される。

進行度バーは終了時には close されなければならない。

これは Tk のバージョン 8.5 またはそれ以降に対する ttk::progressbar ウィジェットを使う、さもなければ BWidget の progressbar の R のコピーを使う。

Value

tkProgressBar に対してはクラス "tkProgressBar" のオブジェクト。

getTkProgressBar と setTkProgressBar に対しては、直近の値を与える長さ1の数値ベクトル (setTkProgressBar に対しては不可視で)。

See Also

[txtProgressBar](#).

(Windows では) Windows のネイティブな制御を使うバージョンについては [winProgressBar](#) (これは又 Tk >= 8.5 で動作する)。

Examples

```
pb <- tkProgressBar("test progress bar", "Some information in %",
                   0, 100, 50)
Sys.sleep(0.5)
u <- c(0, sort(runif(20, 0, 100)), 100)
for(i in u) {
  Sys.sleep(0.1)
  info <- sprintf("%d%% done", round(i))
  setTkProgressBar(pb, i, sprintf("test (%s)", info), info)
}
Sys.sleep(5)
close(pb)
```

tkStartGUI	<i>Tcl/Tk の GUI 開始</i>
------------	------------------------

Description

Tcl/Tk の GUI を開始する

Usage

tkStartGUI()

Details

Tk のテキストウィジェットを使い GUI コンソールの実装を開始する。これは恐らくセッション毎に最大一回呼び出されるべきである。またファイルのページャ (help() が使うような) を Tk のページャに再定義する。

Note

この関数は Windows では使えない。

tkStartGUI() はその評価環境を .GUIenv として保存する。これはインタフェイスを拡張するためにユーザインタフェイスの要素にアクセスすることが出来ることを意味する。三つの主要オブジェクトは名前 Term, Menu そして Toolbar を持ち、そして様々な副メニューとコールバック関数は ls(envir = .GUIenv) で見る事が出来る。

Author(s)

Peter Dalgaard

TkWidgetcmds	<i>Tk のウィジェット命令</i>
--------------	---------------------

Description

これらの関数は Tk のウィジェット命令へのインタフェイスである。

Usage

```
tkactivate(widget, ...)  
tkadd(widget, ...)  
tkaddtag(widget, ...)  
tkbbox(widget, ...)  
tkcanvasx(widget, ...)  
tkcanvasy(widget, ...)  
tkcget(widget, ...)  
tkcompare(widget, ...)  
tkconfigure(widget, ...)  
tkcoords(widget, ...)  
tkcreate(widget, ...)
```

```
tkcurselection(widget, ...)
tkdchars(widget, ...)
tkdebug(widget, ...)
tkdelete(widget, ...)
tkdelta(widget, ...)
tkdeselect(widget, ...)
tkdlineinfo(widget, ...)
tkdtag(widget, ...)
tkdump(widget, ...)
tkentrycget(widget, ...)
tkentryconfigure(widget, ...)
tkfind(widget, ...)
tkflash(widget, ...)
tkfraction(widget, ...)
tkget(widget, ...)
tkgettags(widget, ...)
tkicursor(widget, ...)
tkidentify(widget, ...)
tkindex(widget, ...)
tkinsert(widget, ...)
tkinvoke(widget, ...)
tkitembind(widget, ...)
tkitemcget(widget, ...)
tkitemconfigure(widget, ...)
tkitemfocus(widget, ...)
tkitemlower(widget, ...)
tkitemraise(widget, ...)
tkitemscale(widget, ...)
tkmark.gravity(widget, ...)
tkmark.names(widget, ...)
tkmark.next(widget, ...)
tkmark.previous(widget, ...)
tkmark.set(widget, ...)
tkmark.unset(widget, ...)
tkmove(widget, ...)
tknearest(widget, ...)
tkpost(widget, ...)
tkpostcascade(widget, ...)
tkpostscript(widget, ...)
tkscan.mark(widget, ...)
tkscan.dragto(widget, ...)
tksearch(widget, ...)
tksee(widget, ...)
tkselect(widget, ...)
tkselection.adjust(widget, ...)
tkselection.anchor(widget, ...)
tkselection.clear(widget, ...)
tkselection.from(widget, ...)
tkselection.includes(widget, ...)
tkselection.present(widget, ...)
tkselection.range(widget, ...)
tkselection.set(widget, ...)
```

```

tkselection.to(widget, ...)
tkset(widget, ...)
tksize(widget, ...)
tktoggle(widget, ...)
tktag.add(widget, ...)
tktag.bind(widget, ...)
tktag.cget(widget, ...)
tktag.configure(widget, ...)
tktag.delete(widget, ...)
tktag.lower(widget, ...)
tktag.names(widget, ...)
tktag.nextrange(widget, ...)
tktag.prevrangle(widget, ...)
tktag.raise(widget, ...)
tktag.ranges(widget, ...)
tktag.remove(widget, ...)
tktype(widget, ...)
tkunpost(widget, ...)
tkwindow.cget(widget, ...)
tkwindow.configure(widget, ...)
tkwindow.create(widget, ...)
tkwindow.names(widget, ...)
tkxview(widget, ...)
tkxview.moveto(widget, ...)
tkxview.scroll(widget, ...)
tkyposition(widget, ...)
tkyview(widget, ...)
tkyview.moveto(widget, ...)
tkyview.scroll(widget, ...)

```

Arguments

```

widget      これが適用されるウィジェット
...         .Tcl.args により処理される

```

Details

これらの命令は極端に多すぎてそれらと引数を完全に説明することは困難である。詳細については Tcl/Tk のドキュメントを見て欲しい。少数の例外を除けば可能性として `.a.b selection clear` のような副命令を持つ Tcl ウィジェット命令は `tkselection.clear` のような関数名に変換されウィジェットがその第一引数とされるというパターンになる。

See Also

[TclInterface](#), [TkWidgets](#), [TkCommands](#)

Examples

```

## Not run:
## これらの関数は examples() では実行できないが
## tcltk パッケージがロードされた対話的な R セッション中に
## ペーストすれば実行できるであろう

```

```
tt <- tktoplevel()
tkpack(txt.w <- tktext(tt))
tkinsert(txt.w, "0.0", "plot(1:10)")

# コールバック関数
eval.txt <- function()
  eval(parse(text = tclvalue(tkget(txt.w, "0.0", "end"))))
tkpack(but.w <- tkbutton(tt, text = "Submit", command = eval.txt))

## テキストを編集し終了した時はボタンを押すことを試みよ :

tkdestroy(tt)

## End(Not run)
```

TkWidgets

Tk ウィジェット

Description

Create Tk widgets and associated R objects.

Usage

```
tkwidget(parent, type, ...)

tkbutton(parent, ...)
tkcanvas(parent, ...)
tkcheckboxbutton(parent, ...)
tkentry(parent, ...)
ttkentry(parent, ...)
tkframe(parent, ...)
tklabel(parent, ...)
tklistbox(parent, ...)
tkmenu(parent, ...)
tkmenubutton(parent, ...)
tkmessage(parent, ...)
tkradiobutton(parent, ...)
tkscale(parent, ...)
tkscrollbar(parent, ...)
tktext(parent, ...)
tktoplevel(parent = .TkRoot, ...)

ttkbutton(parent, ...)
ttkcheckboxbutton(parent, ...)
ttkcombobox(parent, ...)
ttkframe(parent, ...)
ttklabel(parent, ...)
ttklabelframe(parent, ...)
ttkmenubutton(parent, ...)
```

```
ttknotebook(parent, ...)
ttkpanedwindow(parent, ...)
ttkprogressbar(parent, ...)
ttkradiobutton(parent, ...)
ttkscale(parent, ...)
ttkscrollbar(parent, ...)
ttkseparator(parent, ...)
ttksizegrip(parent, ...)
ttkspinbox(parent, ...)
ttktreeview(parent, ...)
```

Arguments

parent	ウィジェットウィンドウの親.
type	必要なウィジェットのタイプを記述する文字列.
...	.Tcl.args により処理される.

Details

これらの関数は Tk ウィジェットを作る. `tkwidget` は与えられたタイプのウィジェットを作り, 他は単に `tkwidget` を対応する `type` 引数で呼び出す.

`ttk` で始まる関数は Tk 8.5 またはそれ以降用のテーマ付きウィジェット用である. チュートリアルが <http://www.tkdocs.com> にある.

ウィジェットとその引数を完全に説明するのは不可能である. Tcl/Tk のドキュメントを参照して欲しい.

See Also

[TclInterface](#), [TkCommands](#), [TkWidgetcmds](#)

Examples

```
## Not run:
## これらは examples() では実行できないがtcltk パッケージがロードされた
## 対話的な R セッションにペーストすれば実行できる

tt <- tktoplevel()
label.widget <- tklabel(tt, text = "Hello, World!")
button.widget <- tkbutton(tt, text = "Push",
                          command = function()cat("OW!\n"))
tkpack(label.widget, button.widget) # geometry manager
                                   # see Tk-commands

## ボタンを押しそれから...

tkdestroy(tt)

## テーマ付きウィジェットに対するテスト
if(as.character(tcl("info", "tclversion")) >= "8.5") {
  # テーマ付きウィジェットを使う
  # テーマのリスト
  as.character(tcl("ttk::style", "theme", "names"))
  # テーマを選ぶ -- ここでは XP 以前の windows
  tcl("ttk::style", "theme", "use", "winnative")
}
```

```
} else {  
    #Tk 8.0 ウィジェット  
}  
  
## End(Not run)
```

tk_choose.dir	フォルダーを対話的に選択する
---------------	----------------

Description

フォルダーを対話的に選択するために Tk ウィジェットを使う。

Usage

```
tk_choose.dir(default = "", caption = "Select directory")
```

Arguments

default	最初にどのディレクトリを示すか.
caption	選択ダイアログのキャプション.

Value

長さ1の文字列ベクトル. もし 'Cancel' が選択されれば文字列 NA.

See Also

[tk_choose.files](#)

Examples

```
## Not run:  
tk_choose.dir(getwd(), "Choose a suitable folder")  
  
## End(Not run)
```

tk_choose.files	ファイルのリストを対話的に選択する
-----------------	-------------------

Description

ゼロもしくはそれ以上のファイルを対話的に選択するために Tk のファイルダイアログを使う。

Usage

```
tk_choose.files(default = "", caption = "Select files",  
                multi = TRUE, filters = NULL, index = 1)
```

Arguments

default	どのファイル名を最初に表示するか.
caption	ファイル選択ダイアログ上のキャプション.
multi	複数のファイルの選択を認めるか.
filters	ファイル名フィルターの二列の文字列行列.
index	使用されない.

Details

`file.choose` と異なり, `tk_choose.files` は常にファイルのリストを与える文字列ベクトルを返そうと試みる. もしユーザがダイアログをキャンセルするとゼロ個のファイルが返されるが, `file.choose` はエラーを引き起こす.

`filters` の書式は例で見ることが出来る. ファイルのパターンは拡張子で指定され, "*" は任意のファイルで "" は拡張子を持たない任意のファイルである (ピリオドを含まないファイル名). (特定のプラットフォームでは他の書式が使えるかもしれない.) 一つのファイルタイプに対して複数の拡張子を持つ方法は第一列に同じ名前の複数の列を持つことで, 拡張子がファイル選択ウィジェット中で名前が付くかどうかはプラットフォーム次第である. この書式はリリース前に変更されるかもしれない.

Value

ゼロもしくはそれ以上のファイルパスの文字列ベクトル.

Note

Tk 8.5.0-8.5.4 中のバグは複数ファイルの選択の使用を妨げる.

See Also

[file.choose](#), [tk_choose.dir](#)

Examples

```
Filters <- matrix(c("R code", ".R", "R code", ".s",
                  "Text", ".txt", "All files", "*"),
                4, 2, byrow = TRUE)
Filters
if(interactive()) tk_choose.files(filter = Filters)
```

tk_messageBox

Tk のメッセージボックス

Description

Tk を使った総称的なメッセージボックスの実装

Usage

```
tk_messageBox(type = c("ok", "okcancel", "yesno", "yesnocancel",
                      "retrycancel", "aburtretrycancel"),
              message, caption = "", default = "", ...)
```

Arguments

type	文字列. ダイアログボックスのタイプ. その名前が示唆するボタンを持つ. 省略可能.
message	文字列. ダイアログボックスの情報欄.
caption	表示されるウィジェット上のキャプション.
default	文字列. 既定で使われるボタンの名前.
...	この名前の Tk 関数に渡される追加の名前付き引数. 一例は icon = "warning".

Value

押されたボタンの名前を与える文字列.

See Also

'raw' インタフェイスに対しては [tkmessageBox](#).

tk_select.list	リストから項目を選択する
----------------	--------------

Description

Tk のリストボックスを使い文字列ベクトルから項目を選択する.

Usage

```
tk_select.list(choices, preselect = NULL, multiple = FALSE,
               title = NULL)
```

Arguments

choices	項目の文字列ベクトル.
preselect	文字列ベクトルまたは NULL. もしヌルでなく文字列がリスト上に登場すればこの項目が最初に選択される.
multiple	論理値: 一つ以上の項目が選択可能か?
title	ウィンドウタイトルに対するオプションの文字列, またはタイトルがなければ NULL.

Details

これは OK と Cancel ボタンを備えた Tk のリストボックスとして実装された [select.list](#) のバージョンである. もしリストが長すぎてスクリーンに都合よく収まらなければスクロールバーが表示される.

ダイアログボックスはモーダルであり, R セッションが進行可能になる前に選択されるかキャンセルされなければならない. ある項目上でダブルクリックすることはそれを選択しそれから OK をクリックすることに同値である.

Tk のバージョンが 8.5 かそれ以降ならテーマ付きウィジェットが使われる.

Value

選択された項目の文字列ベクトル. もし `multiple` が偽で項目が選択されなければ (または `Cancel` が使われると), "" が返される. もし `multiple` が真で項目が選択されないと (または `Cancel` が使用されると)長さ 0 の文字列ベクトルが返される.

See Also

[select.list](#) (Windows と OS X GUI 上を除くテキストバージョン), [menu](#) (この `graphics = TRUE` モードはほとんどの Unix 風システムでこれを使う).

Index

- *Topic **file**
 - tk_choose.dir, 18
 - tk_choose.files, 18
- *Topic **misc**
 - TclInterface, 2
 - tclServiceMode, 6
 - TkCommands, 7
 - tkpager, 11
 - tkStartGUI, 13
 - TkWidgetcmds, 13
 - TkWidgets, 16
- *Topic **package**
 - tcltk-package, 1
- *Topic **utilities**
 - tk_messageBox, 19
 - tk_select.list, 20
 - tkProgressBar, 11
- .Tcl (TclInterface), 2
- .Tcl.args, 17
- .Tk.ID (TclInterface), 2
- .Tk.newwin (TclInterface), 2
- .Tk.subwin (TclInterface), 2
- .TkRoot (TclInterface), 2
- [[. tclArray (TclInterface), 2
- [[<- . tclArray (TclInterface), 2
- \$. tclArray (TclInterface), 2
- \$<- . tclArray (TclInterface), 2
- addTclPath (TclInterface), 2
- as.character.tclObj (TclInterface), 2
- as.character.tclVar (TclInterface), 2
- as.double.tclObj (TclInterface), 2
- as.integer.tclObj (TclInterface), 2
- as.logical.tclObj (TclInterface), 2
- as.raw.tclObj (TclInterface), 2
- as.tclObj (TclInterface), 2
- capabilities, 6
- close.tkProgressBar (tkProgressBar), 11
- file.choose, 19
- file.show, 11
- getTkProgressBar (tkProgressBar), 11
- is.tclObj (TclInterface), 2
- is.tkwin (TclInterface), 2
- length.tclArray (TclInterface), 2
- length<- . tclArray (TclInterface), 2
- menu, 21
- NA, 12
- names.tclArray (TclInterface), 2
- names<- . tclArray (TclInterface), 2
- print.tclObj (TclInterface), 2
- R_HOME, 2
- select.list, 20, 21
- setTkProgressBar (tkProgressBar), 11
- tcl (TkCommands), 7
- tclArray (TclInterface), 2
- tclclose (TkCommands), 7
- tclfile.dir (TkCommands), 7
- tclfile.tail (TkCommands), 7
- TclInterface, 2, 10, 15, 17
- tclObj (TclInterface), 2
- tclObj<- (TclInterface), 2
- tclopen (TkCommands), 7
- tclputs (TkCommands), 7
- tclread (TkCommands), 7
- tclRequire (TclInterface), 2
- tclServiceMode, 6
- tcltk (tcltk-package), 1
- tcltk-package, 1
- tclvalue (TclInterface), 2
- tclvalue<- (TclInterface), 2
- tclVar (TclInterface), 2
- tclvar (TclInterface), 2
- tclVersion (TclInterface), 2
- tk_choose.dir, 18, 19
- tk_choose.files, 18, 18
- tk_messageBox, 19
- tk_select.list, 20
- tkactivate (TkWidgetcmds), 13
- tkadd (TkWidgetcmds), 13

- tkaddtag (TkWidgetcmds), 13
- tkbbox (TkWidgetcmds), 13
- tkbell (TkCommands), 7
- tkbind (TkCommands), 7
- tkbindtags (TkCommands), 7
- tkbutton (TkWidgets), 16
- tkcanvas (TkWidgets), 16
- tkcanvasx (TkWidgetcmds), 13
- tkcanvasy (TkWidgetcmds), 13
- tkcget (TkWidgetcmds), 13
- tkcheckbutton (TkWidgets), 16
- tkchooseDirectory (TkCommands), 7
- tkclipboard.append (TkCommands), 7
- tkclipboard.clear (TkCommands), 7
- TkCommands, 6, 7, 15, 17
- tkcompare (TkWidgetcmds), 13
- tkconfigure (TkWidgetcmds), 13
- tkcoords (TkWidgetcmds), 13
- tkcreate (TkWidgetcmds), 13
- tkcurseselection (TkWidgetcmds), 13
- tkdchars (TkWidgetcmds), 13
- tkdebug (TkWidgetcmds), 13
- tkdelete (TkWidgetcmds), 13
- tkdelta (TkWidgetcmds), 13
- tkdeselect (TkWidgetcmds), 13
- tkdestroy (TclInterface), 2
- tkdialog (TkCommands), 7
- tkdlineinfo (TkWidgetcmds), 13
- tkdtag (TkWidgetcmds), 13
- tkdump (TkWidgetcmds), 13
- tkentry (TkWidgets), 16
- tkentrycget (TkWidgetcmds), 13
- tkentryconfigure (TkWidgetcmds), 13
- tkevent.add (TkCommands), 7
- tkevent.delete (TkCommands), 7
- tkevent.generate (TkCommands), 7
- tkevent.info (TkCommands), 7
- tkfind (TkWidgetcmds), 13
- tkflash (TkWidgetcmds), 13
- tkfocus (TkCommands), 7
- tkfont.actual (TkCommands), 7
- tkfont.configure (TkCommands), 7
- tkfont.create (TkCommands), 7
- tkfont.delete (TkCommands), 7
- tkfont.families (TkCommands), 7
- tkfont.measure (TkCommands), 7
- tkfont.metrics (TkCommands), 7
- tkfont.names (TkCommands), 7
- tkfraction (TkWidgetcmds), 13
- tkframe (TkWidgets), 16
- tkget (TkWidgetcmds), 13
- tkgetOpenFile (TkCommands), 7
- tkgetSaveFile (TkCommands), 7
- tkgettags (TkWidgetcmds), 13
- tkgrab (TkCommands), 7
- tkgrid (TkCommands), 7
- tkicursor (TkWidgetcmds), 13
- tkidentify (TkWidgetcmds), 13
- tkimage.create (TkCommands), 7
- tkimage.delete (TkCommands), 7
- tkimage.height (TkCommands), 7
- tkimage.inuse (TkCommands), 7
- tkimage.names (TkCommands), 7
- tkimage.type (TkCommands), 7
- tkimage.types (TkCommands), 7
- tkimage.width (TkCommands), 7
- tkindex (TkWidgetcmds), 13
- tkinsert (TkWidgetcmds), 13
- tkinvoke (TkWidgetcmds), 13
- tkitembind (TkWidgetcmds), 13
- tkitemcget (TkWidgetcmds), 13
- tkitemconfigure (TkWidgetcmds), 13
- tkitemfocus (TkWidgetcmds), 13
- tkitemlower (TkWidgetcmds), 13
- tkitemraise (TkWidgetcmds), 13
- tkitemscale (TkWidgetcmds), 13
- tklabel (TkWidgets), 16
- tklistbox (TkWidgets), 16
- tklower (TkCommands), 7
- tkmark.gravity (TkWidgetcmds), 13
- tkmark.names (TkWidgetcmds), 13
- tkmark.next (TkWidgetcmds), 13
- tkmark.previous (TkWidgetcmds), 13
- tkmark.set (TkWidgetcmds), 13
- tkmark.unset (TkWidgetcmds), 13
- tkmenu (TkWidgets), 16
- tkmenubutton (TkWidgets), 16
- tkmessage (TkWidgets), 16
- tkmessageBox, 20
- tkmessageBox (TkCommands), 7
- tkmove (TkWidgetcmds), 13
- tknearest (TkWidgetcmds), 13
- tkpack (TkCommands), 7
- tkpager, 11
- tkplace (TkCommands), 7
- tkpopup (TkCommands), 7
- tkpost (TkWidgetcmds), 13
- tkpostcascade (TkWidgetcmds), 13
- tkpostscript (TkWidgetcmds), 13
- tkProgressBar, 11
- tkradiobutton (TkWidgets), 16
- tkraise (TkCommands), 7
- tkscale (TkWidgets), 16
- tkscan.dragto (TkWidgetcmds), 13

tkscan.mark (TkWidgetcmds), 13
 tkscrollbar (TkWidgets), 16
 tksearch (TkWidgetcmds), 13
 tksee (TkWidgetcmds), 13
 tkselect (TkWidgetcmds), 13
 tkselection.adjust (TkWidgetcmds), 13
 tkselection.anchor (TkWidgetcmds), 13
 tkselection.clear (TkWidgetcmds), 13
 tkselection.from (TkWidgetcmds), 13
 tkselection.includes (TkWidgetcmds), 13
 tkselection.present (TkWidgetcmds), 13
 tkselection.range (TkWidgetcmds), 13
 tkselection.set (TkWidgetcmds), 13
 tkselection.to (TkWidgetcmds), 13
 tkset (TkWidgetcmds), 13
 tksize (TkWidgetcmds), 13
 tkStartGUI, 13
 tktag.add (TkWidgetcmds), 13
 tktag.bind (TkWidgetcmds), 13
 tktag.cget (TkWidgetcmds), 13
 tktag.configure (TkWidgetcmds), 13
 tktag.delete (TkWidgetcmds), 13
 tktag.lower (TkWidgetcmds), 13
 tktag.names (TkWidgetcmds), 13
 tktag.nextrange (TkWidgetcmds), 13
 tktag.prevrange (TkWidgetcmds), 13
 tktag.raise (TkWidgetcmds), 13
 tktag.ranges (TkWidgetcmds), 13
 tktag.remove (TkWidgetcmds), 13
 tktext (TkWidgets), 16
 tktitle (TkCommands), 7
 tktitle<- (TkCommands), 7
 tktoggle (TkWidgetcmds), 13
 tktoplevel (TkWidgets), 16
 tktype (TkWidgetcmds), 13
 tkunpost (TkWidgetcmds), 13
 tkwait.variable (TkCommands), 7
 tkwait.visibility (TkCommands), 7
 tkwait.window (TkCommands), 7
 tkwidget (TkWidgets), 16
 TkWidgetcmds, 2, 6, 10, 13, 17
 TkWidgets, 2, 6, 10, 15, 16
 tkwindow.cget (TkWidgetcmds), 13
 tkwindow.configure (TkWidgetcmds), 13
 tkwindow.create (TkWidgetcmds), 13
 tkwindow.names (TkWidgetcmds), 13
 tkwininfo (TkCommands), 7
 tkwm.aspect (TkCommands), 7
 tkwm.client (TkCommands), 7
 tkwm.colormapwindows (TkCommands), 7
 tkwm.command (TkCommands), 7
 tkwm.deiconify (TkCommands), 7
 tkwm.focusmodel (TkCommands), 7
 tkwm.frame (TkCommands), 7
 tkwm.geometry (TkCommands), 7
 tkwm.grid (TkCommands), 7
 tkwm.group (TkCommands), 7
 tkwm.iconbitmap (TkCommands), 7
 tkwm.iconify (TkCommands), 7
 tkwm.iconmask (TkCommands), 7
 tkwm.iconname (TkCommands), 7
 tkwm.iconposition (TkCommands), 7
 tkwm.iconwindow (TkCommands), 7
 tkwm.maxsize (TkCommands), 7
 tkwm.minsize (TkCommands), 7
 tkwm.overrideredirect (TkCommands), 7
 tkwm.positionfrom (TkCommands), 7
 tkwm.protocol (TkCommands), 7
 tkwm.resizable (TkCommands), 7
 tkwm.sizefrom (TkCommands), 7
 tkwm.state (TkCommands), 7
 tkwm.title (TkCommands), 7
 tkwm.transient (TkCommands), 7
 tkwm.withdraw (TkCommands), 7
 tkXselection.clear (TkCommands), 7
 tkXselection.get (TkCommands), 7
 tkXselection.handle (TkCommands), 7
 tkXselection.own (TkCommands), 7
 tkxview (TkWidgetcmds), 13
 tkyposition (TkWidgetcmds), 13
 tkyview (TkWidgetcmds), 13
 ttkbutton (TkWidgets), 16
 ttkcheckboxbutton (TkWidgets), 16
 ttkcombobox (TkWidgets), 16
 ttkentry (TkWidgets), 16
 ttkframe (TkWidgets), 16
 ttklabel (TkWidgets), 16
 ttklabelframe (TkWidgets), 16
 ttkmenubutton (TkWidgets), 16
 ttknotebook (TkWidgets), 16
 ttkpanedwindow (TkWidgets), 16
 ttkprogressbar (TkWidgets), 16
 ttkradiobutton (TkWidgets), 16
 ttkscale (TkWidgets), 16
 ttkscrollbar (TkWidgets), 16
 ttkseparator (TkWidgets), 16
 ttksizegrip (TkWidgets), 16
 ttkspinbox (TkWidgets), 16
 ttktreeview (TkWidgets), 16
 txtProgressBar, 12
 winProgressBar, 12