Package: svDialogs (via r-universe)

October 24, 2026

Type Package

Version 1.1.2

Title 'SciViews::R' - Standard Dialog Boxes for Windows, MacOS and Linuxes

Description Quickly construct standard dialog boxes for your GUI, including message boxes, input boxes, list, file or directory selection, ... In case R cannot display GUI dialog boxes, a simpler command line version of these interactive elements is also provided as fallback solution.

Maintainer Philippe Grosjean <phgrosjean@sciviews.org>

Depends R (>= 2.6.0)

Imports methods (>= 2.6.0), rstudioapi (>= 0.7), svGUI (>= 1.0.0), utils (>= 2.6.0)

Suggests covr (>= 3.5.0), knitr (>= 1.42), rmarkdown (>= 2.21), spelling (>= 2.2.1), testthat (>= 3.0.0)

SystemRequirements zenity, yad

License GPL-2

URL https://github.com/SciViews/svDialogs, https://www.sciviews.org/svDialogs/, https://sciviews.r-universe.dev/svDialogs

BugReports https://github.com/SciViews/svDialogs/issues

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

VignetteBuilder knitr

Encoding UTF-8

Language en-US

ByteCompile yes

Config/testthat/edition 3

Repository https://sciviews.r-universe.dev

Date/Publication 2025-12-06 12:55:01 UTC

RemoteUrl https://github.com/SciViews/svDialogs

RemoteRef HEAD

RemoteSha a9e37d13d30c4be20301c30e054c1b3683636d56

Contents

svDialogs-package 2
dlg_dir 3
dlg_form 4
dlg_input 7
dlg_list 9
dlg_message 11
dlg_open 13
dlg_save 16
menu 17

Index **20**

`svDialogs-package` *'SciViews::R' - Standard Dialog Boxes for Windows, MacOS and Linuxes*

Description

The dialog box function dispatch to specific dialog boxes depending on the OS and whether a GUI can be displayed or not.

Important functions

- `dlg_message()` display a message box,
- `dlg_input()` prompt for textual input,
- `dlg_list()` select one or more items in a list,
- `dlg_open()` open one or more existing file(s),
- `dlg_save()` prompt for a file to save to (and ask confirmation if the file already exists),
- `dlg_dir()` select a directory,
- `dlg_form()` display a complex dialog box with textual areas, password, checkboxes, comboboxes, font and size, color, etc.

Note

Under Linux, you should install **'yad'** (preferably), or **'zenity'**. If you use 'zenity', you cannot build more complex `dlg_form()` dialog boxes, and some versions trigger a warning like: "Gtk-Message: GtkDialog mapped without a transient parent. This is discouraged." You do not get these inconveniences if you use 'yad' (default choice in case both programs are installed simultaneously).

Author(s)

Maintainer: Philippe Grosjean <phgrosjean@sciviews.org> (ORCID)

Other contributors:

- Paul Hibbins <paulhibbing@gmail.com> [contributor]

See Also

Useful links:

- https://github.com/SciViews/svDialogs
- https://www.sciviews.org/svDialogs/
- https://sciviews.r-universe.dev/svDialogs
- Report bugs at https://github.com/SciViews/svDialogs/issues

---

`dlg_dir` *Modal dialog to select a directory.*

Description

Select an existing directory, or create a new one.

Usage

```
dlg_dir(default = getwd(), title = "Choose a directory", ..., gui = .GUI)

dlgDir(default = getwd(), title = "Choose a directory", ..., gui = .GUI)

## S3 method for class 'gui'
dlg_dir(default = getwd(), title, ..., gui = .GUI)

## S3 method for class 'textCLI'
dlg_dir(default = getwd(), title, ..., gui = .GUI)

## S3 method for class 'nativeGUI'
dlg_dir(
  default = getwd(),
  title,
  rstudio = getOption("svDialogs.rstudio", TRUE),
  ...,
  gui = .GUI
)
```

Arguments

default	The path to the default directory that is proposed (e.g., current working directory).
title	A title to display on top of the dialog box.
...	Pass further arguments to methods.
gui	The 'gui' object concerned by this dialog box.
rstudio	Logical. Should 'RStudio' dialog boxes automatically be used if available? If FALSE, force using OS dialog boxes, but only in 'RStudio Desktop' (ignored in 'RStudio Server'). Can be changed globally with options(svDialogs.rstudio = TRUE|FALSE). TRUE by default.

Value

The modified 'gui' object is returned invisibly. Use its gui$res component to get the returned directory (the string is empty when the user cancelled the dialog box, see example).

See Also

dlg_open(), dlg_save()

Examples

```
## Not run:
# A quick default directory changer
setwd(dlg_dir(default = getwd())$res)

## End(Not run)
```

---

dlg_form *Modal dialog to fill a series of different fields.*

Description

A customizable form dialog box with checkboxes, entries, lists, etc.

Usage

```
dlg_form(
  form,
  title = "Fill the form",
  message = NULL,
  columns = 1,
  strip.type = TRUE,
  ...,
  gui = .GUI
)
```

```
dlgForm(
  form,
  title = "Fill the form",
  message = NULL,
  columns = 1,
  strip.type = TRUE,
  ...,
  gui = .GUI
)

## S3 method for class 'gui'
dlg_form(
  form,
  title = "Fill the form",
  message = NULL,
  columns = 1,
  strip.type = TRUE,
  ...,
  gui = .GUI
)

## S3 method for class 'textCLI'
dlg_form(
  form,
  title = "Fill the form",
  message = NULL,
  columns = 1,
  strip.type = TRUE,
  ...,
  gui = .GUI
)

## S3 method for class 'nativeGUI'
dlg_form(
  form,
  title = "Fill the form",
  message = NULL,
  columns = 1,
  strip.type = TRUE,
  ...,
  gui = .GUI
)
```

Arguments

form	Named list of default values, or list of possible items. Names are the labels of corresponding entries in the dialog box, followed by an indicator of the type of entry to place in the dialog box (see details).

`title`	The title of the form.
`message`	An optional message to display in the dialog box. Use `\\n` for line break, or provide a vector of character strings, one for each line.
`columns`	Arrange the entries on this number of columns (by row).
`strip.type`	Do we strip the type from the names in results?
`...`	Pass further arguments to methods.
`gui`	The 'gui' object concerned by this dialog box.

Details

The form content is defined by a named list. Items are default values, or a list of possible items, e.g., for the combobox. Names are labels displayed in front of each field in the form. Follow them by a code that represents the type of entry you want to use:

- `:TXT` for simple (default) textual box,
- `:H` for hidden text (password),
- `:RO` for read-only text,
- `:NUM` for null of positive integers with up/down arrows,
- `:CHK` for checkbox: `TRUE` or `FALSE`,
- `:CB` for read-only combobox,
- `:CBE` for editable combobox,
- `:FL` to select one existing file,
- `:MFL` to select multiple existing files,
- `:SFL` to select or create one file,
- `:DIR` to select a directory,
- `:CDIR` to select or create a directory,
- `:FN` to select font and font size,
- `:DT` to enter a date,
- :CLR‘ to enter a RGB color,
- :BTN‘ to create a button that execute some code,
- `:LBL` to add a label.

For the moment, the form dialog box is only supported on Linux. You have to install **yad** to get access to it. On Ubuntu, you do so by `sudo apt-get install yad`. For other system, look at the documentation.

Value

The modified 'gui' object is returned invisibly. Use its `gui$res` component to get the list of returned fields.

Note

`dlg_form()` can use strings with embedded quotes, but you have to escape them (and need to double the backslash escape sign). Currently, this is not done automatically, on the contrary to the other `dlg_xxx()` functions!

See Also

dlg_input(), dlg_list()

Examples

```
## Not run:
# Ask a series of items at once in a dialog box
form <- list(
  "Name:TXT" = "John Smith",
  "Age:NUM" = 25,
  "Sex:CB" = c("male", "female"),
  "Married:CHK" = FALSE
)
dlg_form(form, "My data")$res

## End(Not run)
```

---

dlg_input *Modal dialog to input a string or a value.*

Description

Prompt for some data in a modal dialog box.

Usage

```
dlg_input(message = "Enter a value", default = "", ..., gui = .GUI)

dlgInput(message = "Enter a value", default = "", ..., gui = .GUI)

## S3 method for class 'gui'
dlg_input(message = "Enter a value", default = "", ..., gui = .GUI)

## S3 method for class 'textCLI'
dlg_input(message = "Enter a value", default = "", ..., gui = .GUI)

## S3 method for class 'nativeGUI'
dlg_input(
  message = "Enter a value",
  default = "",
  rstudio = getOption("svDialogs.rstudio", TRUE),
  ...,
  gui = .GUI
)
```

Arguments

message	The message to display in the dialog box. Use \\n for line break, or provide a vector of character strings, one for each line.
default	The default value in the text box. Single string or NULL.
...	Pass further arguments to methods.
gui	The 'gui' object concerned by this dialog box.
rstudio	Logical. Should 'RStudio' dialog boxes automatically be used if available? If FALSE, force using OS dialog boxes, but only in 'RStudio Desktop' (ignored in 'RStudio Server'). Can be changed globally with options(svDialogs.rstudio = TRUE|FALSE). TRUE by default.

Value

The modified 'gui' object is returned invisibly. The text entered by the user at the input box, or an empty string if the dialog box was cancelled can be obtained from gui$res (see example).

Note

The 'RStudio' version of this dialog box does not allow for an empty string. So, for better portability, try never to expect an empty string from the user (the 'Cancel' button is there to dismiss the dialog box). On MacOS, single and double quotes are temporarily replaced by their slanted versions (unicode characters u3032 and u2033, respectively) because the command that triggers the dialog box does not allow quotes inside strings. Regular quotes are reset on the output. This is the only hack we found that was working. Better solutions are welcome!

See Also

dlg_list(), dlg_form(), dlg_message()

Examples

```
## Not run:
# Ask something...
user <- dlg_input("Who are you?", Sys.info()["user"])$res
if (!length(user)) {# The user clicked the 'cancel' button
  cat("OK, you prefer to stay anonymous!\n")
} else {
  cat("Hello", user, "\n")
}

## End(Not run)
```

dlg_list *Modal dialog to select one or more items in a list.*

Description

Display a list and allow user to select either one, or multiple items in that list.

Usage

```
dlg_list(
  choices,
  preselect = NULL,
  multiple = FALSE,
  title = NULL,
  ...,
  gui = .GUI
)

dlgList(
  choices,
  preselect = NULL,
  multiple = FALSE,
  title = NULL,
  ...,
  gui = .GUI
)

## S3 method for class 'gui'
dlg_list(
  choices,
  preselect = NULL,
  multiple = FALSE,
  title = NULL,
  ...,
  gui = .GUI
)

## S3 method for class 'textCLI'
dlg_list(
  choices,
  preselect = NULL,
  multiple = FALSE,
  title = NULL,
  ...,
  gui = .GUI
)
```

```
## S3 method for class 'nativeGUI'
dlg_list(
  choices,
  preselect = NULL,
  multiple = FALSE,
  title = NULL,
  rstudio = getOption("svDialogs.rstudio", TRUE),
  ...,
  gui = .GUI
)
```

Arguments

choices	The list of items. It is coerced to character strings.
preselect	A list of preselections, or NULL (then, the first element is selected in the list). Preselections not in choices are tolerated (but they are ignored without warning or error).
multiple	Is it a multiple selection dialog box?
title	The title of the dialog box, or NULL to use a default title instead.
...	Pass further arguments to methods.
gui	The 'gui' object concerned by this dialog box.
rstudio	Logical. Should 'RStudio' dialog boxes automatically be used if available? If FALSE, force using OS dialog boxes, but only in 'RStudio Desktop' (ignored in 'RStudio Server'). Can be changed globally with options(svDialogs.rstudio = TRUE|FALSE). TRUE by default.

Value

The modified 'gui' object is returned invisibly. A list with selected items, or a character vector of length 0 if the dialog box was cancelled is available from gui$res (see examples).

Note

RStudio does not provide (yet) a graphical list selector (as of version 1.1.447). Consequently, a Tk version is used (if 'tcltk' is available) for 'RStudio Desktop' and a textual version at the R Console is used in the other cases, for 'nativeGUI' as a temporary workaround (should be implemented in Shiny later on). Also note that the textual version only reports preselection when multiple == TRUE, and they are not used automatically if you do not respecify them in your feedback (limitation of utils::select.list(graphics = FALSE)). On MacOS, and outside of R(64).app, which has his own list selection dialog box, single and double quotes are temporarily replaced by their slanted versions (unicode characters u3032 and u2033, respectively) because the command that triggers the dialog box does not allow quotes inside strings. Regular quotes are reset on the output. This is the only hack we found that was working. Better solutions are welcome, of course!

See Also

dlg_form(), dlg_input()

Examples

```
## Not run:
# Select one or several months
res <- dlg_list(month.name, multiple = TRUE)$res
if (!length(res)) {
  cat("You cancelled the choice\n")
} else {
  cat("You selected:\n")
  print(res)
}

## End(Not run)
```

---

dlg_message *Display a modal message box.*

Description

A message box with icon, text, and one to three buttons.

Usage

```
dlg_message(
  message,
  type = c("ok", "okcancel", "yesno", "yesnocancel"),
  ...,
  gui = .GUI
)

dlgMessage(
  message,
  type = c("ok", "okcancel", "yesno", "yesnocancel"),
  ...,
  gui = .GUI
)

msg_box(message)

msgBox(message)

ok_cancel_box(message)

okCancelBox(message)

## S3 method for class 'gui'
dlg_message(
  message,
```

```
  type = c("ok", "okcancel", "yesno", "yesnocancel"),
  ...,
  gui = .GUI
)

## S3 method for class 'textCLI'
dlg_message(
  message,
  type = c("ok", "okcancel", "yesno", "yesnocancel"),
  ...,
  gui = .GUI
)

## S3 method for class 'nativeGUI'
dlg_message(
  message,
  type = c("ok", "okcancel", "yesno", "yesnocancel"),
  rstudio = getOption("svDialogs.rstudio", TRUE),
  ...,
  gui = .GUI
)
```

Arguments

message	The message to display in the dialog box. Use \\n for line break, or provide a vector of character strings, one for each line (except under RStudio where it is not possible to force line break inside the message string).
type	The type of dialog box: 'ok', 'okcancel', 'yesno' or 'yesnocancel'.
...	Pass further arguments to methods.
gui	The 'gui' object concerned by this dialog box.
rstudio	Logical. Should 'RStudio' dialog boxes automatically be used if available? If FALSE, force using OS dialog boxes, but only in 'RStudio Desktop' (ignored in 'RStudio Server'). Can be changed globally with options(svDialogs.rstudio = TRUE|FALSE). TRUE by default.

Value

The modified 'gui' object is returned invisibly. A string with the name of the button ("ok", "cancel", "yes" or "no") that the user pressed can be obtained from gui$res (see example). msg_box() just returns the name of the button ("ok"), while ok_cancel_box() returns TRUE if "ok" was clicked or FALSE if "cancel" was clicked.

Note

On 'RStudio' or with 'zenity' under Linux, only two buttons are available. So, when using type = "yesnocancel", two successive dialog boxes are displayed: one with the message and 'yes'/'no' buttons, and a second one asking to continue, and if the user clicks 'no', the function returns "cancel". This is clearly sub-optimal. So, for a clean experience on all supported platforms, try to avoid 'yesnocancel' as much as possible.

See Also

dlg_list(), dlg_input()

Examples

```
## Not run:
# A simple information box
dlg_message("Hello world!")$res

# Ask to continue
dlg_message(c("This is a long task!", "Continue?"), "okcancel")$res

# Ask a question
dlg_message("Do you like apples?", "yesno")$res

# Idem, but one can interrupt too
res <- dlg_message("Do you like oranges?", "yesnocancel")$res
if (res == "cancel")
  cat("Ah, ah! You refuse to answer!\n")

# Simpler version with msgBox and okCancelBox
msg_box("Information message") # Use this to interrupt script and inform user
if (ok_cancel_box("Continue?")) cat("we continue\n") else cat("stop it!\n")

## End(Not run)
```

---

dlg_open *Modal dialog to select a file.*

Description

Select an existing file, or create a new one.

Usage

```
dlg_open(
  default = "",
  title = if (multiple) "Select files" else "Select file",
  multiple = FALSE,
  filters = dlg_filters["All", ],
  ...,
  gui = .GUI
)

dlgOpen(
  default = "",
  title = if (multiple) "Select files" else "Select file",
  multiple = FALSE,
```

```
  filters = dlg_filters["All", ],
  ...,
  gui = .GUI
)

dlg_filters

dlgFilters

## S3 method for class 'gui'
dlg_open(
  default,
  title,
  multiple = FALSE,
  filters = dlg_filters["All", ],
  ...,
  gui = .GUI
)

## S3 method for class 'textCLI'
dlg_open(
  default,
  title,
  multiple = FALSE,
  filters = dlg_filters["All", ],
  ...,
  gui = .GUI
)

## S3 method for class 'nativeGUI'
dlg_open(
  default,
  title,
  multiple = FALSE,
  filters = dlg_filters["All", ],
  rstudio = getOption("svDialogs.rstudio", TRUE),
  ...,
  gui = .GUI
)
```

Arguments

default	The default file to start with (use /dir/* or /dir/*.* to start in a given directory).
title	A title to display on top of the dialog box.
multiple	Is a multiple selection of files allowed?
filters	A specification of file filters as a nx2 matrix, or a character string with even number of items. First items is the label, second one is the filter. See dlg_filters

	for examples. This is currently ignored on MacOS and RStudio, since such kind of filter is defined differently there.
...	Pass further arguments to methods.
gui	The 'gui' object concerned by this dialog box.
rstudio	Logical. Should 'RStudio' dialog boxes automatically be used if available? If FALSE, force using OS dialog boxes, but only in 'RStudio Desktop' (ignored in 'RStudio Server'). Can be changed globally with options(svDialogs.rstudio = TRUE|FALSE). TRUE by default.

Format

An object of class matrix (inherits from array) with 22 rows and 2 columns.

An object of class matrix (inherits from array) with 22 rows and 2 columns.

Value

The modified 'gui' object is returned invisibly. The chosen file(s), or an empty string if the "cancel" button was clicked is found in gui$res (see example).

Note

On 'RStudio Server', multiple = TRUE cannot be honored for now. So, you can only select one file there, and a warning is issued to remind you that. On 'RStudio Desktop', the OS-native dialog box is used instead in case of multiple = TRUE. Also, the textual version is painful to indicate the full path of several files. So, it should use globbing, and/or indication of a path followed by a selection in a list (to be done in further versions).

See Also

dlg_save(), dlg_dir()

Examples

```
## Not run:
# Choose one R file
dlg_open(title = "Select one R file", filters = dlg_filters[c("R", "All"), ])$res
# Choose several files
dlg_open(multiple = TRUE)$res

## End(Not run)
```

dlg_save *Modal dialog to select a file to save to.*

Description

Select an existing file, or create a new one to save data.

Usage

```
dlg_save(
  default = "untitled",
  title = "Save file as",
  filters = dlg_filters["All", ],
  ...,
  gui = .GUI
)

dlgSave(
  default = "untitled",
  title = "Save file as",
  filters = dlg_filters["All", ],
  ...,
  gui = .GUI
)

## S3 method for class 'gui'
dlg_save(default, title, filters = dlg_filters["All", ], ..., gui = .GUI)

## S3 method for class 'textCLI'
dlg_save(default, title, filters = dlg_filters["All", ], ..., gui = .GUI)

## S3 method for class 'nativeGUI'
dlg_save(
  default,
  title,
  filters = dlg_filters["All", ],
  rstudio = getOption("svDialogs.rstudio", TRUE),
  ...,
  gui = .GUI
)
```

Arguments

default	The default file to start with (use /dir/* or /dir/*.* to start in a given directory, but without predefined name).
title	A title to display on top of the dialog box.

filters	A specification of file filters as a nx2 matrix, or a character string with even number of items. First items is the label, second one is the filter. See dlg_filters for examples. This is currently ignored on MacOS, since such kind of filter is defined differently there.
...	Pass further arguments to methods.
gui	The modified 'gui' object is returned invisibly. The chosen file, or an empty string (if the "cancel" button was clicked or confirmation was cancelled) is placed in gui$res (see example). For existing files, confirmation is always asked!
rstudio	Logical. Should 'RStudio' dialog boxes automatically be used if available? If FALSE, force using OS dialog boxes, but only in 'RStudio Desktop' (ignored in 'RStudio Server'). Can be changed globally with options(svDialogs.rstudio = TRUE|FALSE). TRUE by default.

Note

In case the file already exists, the user is prompted to confirm he wants to overwrite the existing file. If he clicks 'Cancel', then the return is an empty string. The 'RStudio' version of this dialog box currently ignores the filters = argument.

See Also

dlg_open(), dlg_dir()

Examples

```
## Not run:
# Choose one R filename to save some R script into it
dlg_save(title = "Save R script to", filters = dlg_filters[c("R", "All"), ])$res

## End(Not run)
```

---

menu *Manage custom R menus.*

Description

Create, populate and rework custom R menus.

Usage

```
menu_names()

menuNames()

menu_items(menuname)
```

```
menuItems(menuname)

menu_add(menuname)

menuAdd(menuname)

menu_add_item(menuname, itemname, action)

menuAddItem(menuname, itemname, action)

menu_del(menuname)

menuDel(menuname)

menu_del_item(menuname, itemname)

menuDelItem(menuname, itemname)
```

Arguments

menuname	A character string naming a menu.
itemname	A character string naming a menu item on an existing menu.
action	A character string with the action to perform when the menu item is selected, or "none" for no action. Use "enable" or "disable" to activate or deactivate an existing menu item.

Details

On Windows, the function manages custom menus in RGui the same way as winMenuAdd() and similar function do. Menus are added to the right and new menu entries are added to the bottom of the menu. It is currently not possible to add menus for **'Rterm.exe'** under Windows.

On Unix/Linux, under Gnome, you must install a little Gtk2 program called ctxmenu, as well as a few other utilities to manage the menu actions. You can download corresponding files (GPL-2 license) and get further instructions at the bottom of http://www.sciviews.org/SciViews-R/. The R code in **'svDialogs'** only creates menu configuration files in ~/.ctxmenu/tmp/ and only in interactive R session and after the user agrees to do so (unless options(svDialogs.tmpfiles = TRUE)). Once you installed these files, you can access the menus by setting up keyboard shortcuts to activate main and context menus. The respective commands are ctxmenu-main and ctxmenu-context and you can use the preference panel to assign, e.g., <shift-menu> and <ctrl-menu>, or other keyboard shortcuts to these commands. Once everything is set up, you should see your menus appearing when a console where R + **'svDialogs'** runs is the active window and you hit these shortcuts (after you have defined at least one custom menu). Note also that you can define custom context menus for other applications too, see the README file in the ctxmenu download.

On MacOS, these functions are not implemented yet (but see source of the package for experimental code commented out and try the JGR version for a first implementation there).

Action is treated as R input (echoed at the command line, parsed and executed), except if it is "none". In this case, no action is run when the menu item is selected (merely as a placeholder for

future menu actions). You can change the action of an existing menu by reissuing the command with a different action argument.

If the menuname= parameter of menu_add_item() does not exists, it is automatically created. For creating submenus, separate successive menu names with slashes. Use "-" as name for separation menus under Windows or Unix/Linux.

Value

These function return NULL invisibly. They are used for their side-effect of creating, changing, or deleting custom R menus.

See Also

dlg_open(), dlg_save()

Examples

```
## Not run:
# A quick default directory changer
setwd(dlg_dir(default = getwd())$res)

## End(Not run)
```

Index

∗ **Modal dialog box**
  dlg_dir, 3
  dlg_form, 4
  dlg_input, 7
  dlg_list, 9
  dlg_message, 11
  dlg_open, 13
  dlg_save, 16
  menu, 17
∗ **datasets**
  dlg_open, 13
∗ **misc**
  dlg_dir, 3
  dlg_form, 4
  dlg_input, 7
  dlg_list, 9
  dlg_message, 11
  dlg_open, 13
  dlg_save, 16
  menu, 17

dlg_dir, 3
dlg_dir(), *2*, *15*, *17*
dlg_filters *(*dlg_open*)*, 13
dlg_form, 4
dlg_form(), *2*, *8*, *10*
dlg_input, 7
dlg_input(), *2*, *7*, *10*, *13*
dlg_list, 9
dlg_list(), *2*, *7*, *8*, *13*
dlg_message, 11
dlg_message(), *2*, *8*
dlg_open, 13
dlg_open(), *2*, *4*, *17*, *19*
dlg_save, 16
dlg_save(), *2*, *4*, *15*, *19*
dlgDir *(*dlg_dir*)*, 3
dlgFilters *(*dlg_open*)*, 13
dlgForm *(*dlg_form*)*, 4
dlgInput *(*dlg_input*)*, 7
dlgList *(*dlg_list*)*, 9
dlgMessage *(*dlg_message*)*, 11
dlgOpen *(*dlg_open*)*, 13
dlgSave *(*dlg_save*)*, 16

menu, 17
menu_add *(*menu*)*, 17
menu_add_item *(*menu*)*, 17
menu_del *(*menu*)*, 17
menu_del_item *(*menu*)*, 17
menu_items *(*menu*)*, 17
menu_names *(*menu*)*, 17
menuAdd *(*menu*)*, 17
menuAddItem *(*menu*)*, 17
menuDel *(*menu*)*, 17
menuDelItem *(*menu*)*, 17
menuItems *(*menu*)*, 17
menuNames *(*menu*)*, 17
msg_box *(*dlg_message*)*, 11
msgBox *(*dlg_message*)*, 11

ok_cancel_box *(*dlg_message*)*, 11
okCancelBox *(*dlg_message*)*, 11

svDialogs *(*svDialogs-package*)*, 2
svDialogs-package, 2