Package ‘RSurvey’

April 11, 2018

Title Geographic Information System Application

Version 0.9.3

Description A geographic information system (GIS) graphical user interface (GUI) that provides data viewing, management, and analysis tools.

Depends R (>= 3.1.0)

Imports colorspace, graphics, grDevices, inlmisc, MBA, methods, raster, rgdal, rgeos, sp, stats, tcltk, utils

Suggests dichromat, leaflet, rgl, XML

SystemRequirements Tcl/Tk (>= 8.5), Tktable (>= 2.9, optional)

License CC0

Copyright This software is in the public domain because it contains materials that originally came from the United States Geological Survey (USGS), an agency of the United States Department of Interior. For more information, see the official USGS copyright policy at https://www2.usgs.gov/visual-id/credit_usgs.html

URL https://github.com/USGS-R/RSurvey

Encoding UTF-8

ByteCompile true

RoxygenNote 6.0.1

NeedsCompilation no

Author Jason C. Fisher [aut, cre] (<https://orcid.org/0000-0001-9032-8912>)

Maintainer Jason C. Fisher <jfisher@usgs.gov>

Repository CRAN

Date/Publication 2018-04-11 16:02:23 UTC

R topics documented:

BuildHistogram *GUI: Histogram Input Parameters*

Description

A graphical user interface (GUI) for specifying input parameters for the `hist` function.

Usage

```
BuildHistogram(d, var.names = NULL, var.default = 1L,
  processed.rec = NULL, parent = NULL)
```

Arguments

`d` list, data.frame, matrix, or numeric. Vector(s) of values for which the histogram is desired.

`var.names` character. Names corresponding to each vector (column) in argument `d`.

`var.default` character or integer. Vector name or index in argument `d`.

`processed.rec` integer. Vector of record indexes for processed data.

`parent` tkwin. GUI parent window

Value

`NULL`

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`plot.histogram`

Examples

```
## Not run:
  BuildHistogram(iris)

## End(Not run)
```

`CheckEntry` *Control Content in Entry Widget*

Description

This function enforces content control on entry widgets.

Usage

```
CheckEntry(obj.class, ent.str = "")
```

Arguments

`obj.class` character. Name of object class, either *real*, *integer*, or *logical*

`ent.str` character. Value from entry widget

Value

Returns a character string that can be easily converted to the desired object class.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
CheckEntry("numeric", "3.14ab")
CheckEntry("integer", "3.")
```

ChooseColor *GUI: Color Picker*

Description

A graphical user interface (GUI) for selecting a color.

Usage

```
ChooseColor(col, parent = NULL)
```

Arguments

col	character. Initial color, see 'Value' section
parent	tkwin. GUI parent window

Value

Returns a selected color in terms of its RGB components, a string of the form "#RRGGBB" where each of the pairs RR, GG, BB consist of two hexadecimal digits giving a value in the range 00 to FF.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

col2rgb

Examples

```
## Not run:
  ChooseColor(col = "#669933")

## End(Not run)
```

ChoosePch *GUI: Plotting Symbol Picker*

Description

A graphical user interface (GUI) for selecting a plotting symbol to use.

Usage

```
ChoosePch(pch = NA, parent = NULL)
```

Arguments

`pch`	numeric or character. Initial plotting symbol
`parent`	tkwin. GUI parent window

Value

Returns an object of class numeric or integer, specifying the selected plotting symbol.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`points`

Examples

```
## Not run:
  ChoosePch(pch = "+")

## End(Not run)
```

Data *Set or Query Data and Parameters*

Description

This function is used to set or query parameters and their attributes.

Usage

```
Data(option, value, which.attr = NULL, clear.proj = FALSE,
  clear.data = FALSE, replace.all = NULL)
```

Arguments

`option` character. Parameter name, see ‘Parameters’ section.

`value` Parameter value specified for `option` (optional)

`which.attr` character. A non-empty character string specifying which attribute is to be accessed.

`clear.proj` logical. If true, basic graphical user interface (GUI) preferences will be saved and all other data removed.

`clear.data` logical. If true, only datasets will be removed.

`replace.all` list. A replacement list of parameter values.

Value

If `value` is given, the object specified by `option` is returned. A `NULL` value is returned for objects not yet assigned a value and where no default value is available. Default values are specified internally within this function.

Data

Imported unprocessed data is saved to the data frame `data.raw`, see `ImportText`. Processed point data is saved to the data frame `data.pts`, and interpolated surface data is saved to the list `data.grd`.

Parameters

Parameters undefined elsewhere in the help documentation include:

`ver` package version number

`win.loc` default horizontal and vertical location for GUI placement in pixels.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
# set a parameter
Data("test1", 3.14159265)
Data("test2", list(id = "PI", val = 3.14159265))

# retrieve a parameter value
Data("test1")
Data("test2")
Data(c("test2", "id"))
Data(c("test2", "val"))

# get all parameter values
d <- Data()

# remove all saved parameter values
Data(replace.all = list())
```

```
# recover saved parameter values
Data(replace.all = d)
```

DefineGrid *GUI: Define Interpolation Grid*

Description

A graphical user interface (GUI) for defining the interpolation grid.

Usage

```
DefineGrid(grid = NULL, parent = NULL)
```

Arguments

grid	list. Interpolation grid object, see 'Value' section.
parent	tkwin. GUI parent window

Value

Returns an object of class list with the following components:

opt	an integer indicating the option that will be used to define the interpolation grid. Where `opt = 1` indicates grid boundaries based on the extent of point data and a resolution of 100 rows and 100 columns; `opt = 2` indicates grid boundaries based on the extent of point data and a cell resolution defined by the `res` component; `opt = 3` indicates that the grid geometry is explicitly defined by the `geo` component.
res	numeric vector of length 2 with components `x` and `y` giving the grid spacing along the *x*- and *y*-axis, respectively.
geo	numeric vector of length 6 with components `nrows` and `ncols` giving the number of rows and columns, and `xmn`, `xmx`, `ymn`, and `ymx` giving the limits of the grid boundary along the *x*- and *y*-axis.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  DefineGrid()

## End(Not run)
```

EditData *GUI: Data Editor*

Description

A graphical user interface (GUI) for viewing and editing table formatted data.

Usage

```
EditData(d, col.names = names(d), row.names = NULL, col.formats = NULL,
  read.only = FALSE, changelog = NULL, win.title = "Data",
  parent = NULL)
```

Arguments

`d`	list, matrix, or data.frame. Data used to populate the data table.
`col.names`	character. Vector of column names
`row.names`	character. Vector of row names
`col.formats`	character. Vector of format conversion specification strings, see `sprintf` and `strftime`.
`read.only`	logical. Specifies whether the data table is in read only mode.
`changelog`	data.frame. History of all data table edits, see 'Value' section.
`win.title`	character. String to display as the title of the dialog box.
`parent`	tkwin. GUI parent window

Details

Row titles are taken from the row names attribute of argument `d`. Pattern searches are performed using `grep`. Edits are reflected in the `changelog`.

Value

Returns `NULL` if no edits were made; otherwise, new values of `d` and `changelog` are returned as components in a `list`. The `changelog` data table contains the following variables:

`timestamp`	a date-time value that identifies when the edit event occurred.
`record`	row name
`variable`	column name
`old`	value before editing
`new`	value after editing

Note

Requires the Tcl package Tktable.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

BuildHistogram

Examples

```
## Not run:
  tcltk::tclRequire("Tktable", warn = TRUE)

  n <- 1000L
  V1 <- sample(c(1:9, NA), n, replace = TRUE)
  V2 <- sample(LETTERS, n, replace = TRUE)
  V3 <- as.POSIXct(rnorm(n, mean = 0, sd = 1e6), origin = "2010-01-01")
  V4 <- sample(V1 * pi, n)
  d <- data.frame(V1, V2, V3, V4)
  col.names <- c("Integers", "Letters", "DateTime", "Numeric")
  col.formats <- c("%d", "%s", "%m/%d/%Y %H:%M", "")
  obj <- EditData(d, col.names, col.formats)
  str(obj)

  rownames(d) <- paste0(sample(LETTERS, n, replace = TRUE), seq_len(n))
  EditData(d, read.only = TRUE)

  colnames(d) <- NULL
  rownames(d) <- NULL
  EditData(d, read.only = TRUE)

## End(Not run)
```

EditFunction *GUI: Function Editor*

Description

A graphical user interface (GUI) for defining functions in the R language.

Usage

```
EditFunction(cols, index = NULL, fun = NULL, value.length = NULL,
  value.class = NULL, win.title = "Edit Function", parent = NULL)
```

Arguments

`cols`	list. y
`index`	integer. An element index number in `cols`.
`fun`	character. Existing function, only used if `index = NULL`
`value.length`	integer. Required length for the evaluated function.
`value.class`	character. Required class for the evaluated function.
`win.title`	character. String to display as the title of the dialog box.
`parent`	tkwin. GUI parent window

Details

This GUI is appropriate for deriving new variables in a pre-existing data frame or query building.

Value

Returns an object of class list with the following components:

`fun`	user defined function (when evaluated, this string must be parseable).
`class`	object class for the evaluated function.
`summary`	default summary for the evaluated function.
`sample`	first non-missing value for the evaluated function.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`EvalFunction`

Examples

```
## Not run:
  d <- list(x = 1:10, y = 10:1)
  Data("data.raw", d)
  cols <- list()
  cols[[1]] <- list(id = "X", index = 1, fun = "\"X\"")
  cols[[2]] <- list(id = "Y", index = 2, fun = "\"Y\"")
  cols[[3]] <- list(id = "New Variable", fun = "\"X\" + \"Y\"")
  EditFunction(cols, index = 3)

## End(Not run)
```

EditText *GUI: Edit Text*

Description

A graphical user interface (GUI) for viewing and editing text.

Usage

```
EditText(txt, read.only = FALSE, win.title = "View Text",
  is.fixed.width.font = FALSE, parent = NULL)
```

Arguments

txt	character. Text used to populate the window.
read.only	logical. Specifies whether the text is read only.
win.title	character. Title of the dialog box.
is.fixed.width.font	logical. Specifies whether a fixed-width font be used.
parent	tkwin. GUI parent window

Value

Returns an object of class character with edited text.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  txt <- c("\"Hills cherish the ambition",
           "    to turn into partial",
           "   differential equations\"",
           "",
           "        -Donald Hall")
  new.txt <- EditText(txt, is.fixed.width.font = TRUE)

  EditText(txt, read.only = TRUE)

## End(Not run)
```

EvalFunction *Parse and Evaluate an RSurvey Expression*

Description

This function parses and evaluates a character string representation of an **RSurvey** expression.

Usage

```
EvalFunction(txt, cols)
```

Arguments

`txt`	character. A string representation of an R function.
`cols`	list. See `ManageVariables`

Value

Returns the result of evaluating the text expression.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`parse`, `eval`

Examples

```
d <- list(x = 1:10, y = 10:1)
Data("data.raw", d)
cols <- list()
cols[[1]] <- list(id = "X", index = 1, fun = "\"X\"")
cols[[2]] <- list(id = "Y", index = 2, fun = "\"Y\"")
EvalFunction("\"Y\"", cols)
EvalFunction("\"X\" + \"Y\"", cols)
EvalFunction("rnorm(12)", cols)
```

`ExportData` *GUI: Export Data*

Description

A graphical user interface (GUI) for exporting data to text files, shapefiles, or R data files.

Usage

```
ExportData(file.type = "txt", parent = NULL)
```

Arguments

`file.type` character. Output file type: either *txt* for text files, *rda* for R-data files, or *shp* for shapefiles.

`parent` tkwin. GUI parent window

Value

Saves the GUI options in the `export` component of `Data`. List components of `export` include:

`processed` indicates whether exported data are limited to processed records.

`fmts` indicates whether a header line of conversion specification format strings is written (text only).

`cols` indicates whether a header line of column names is written (text only).

`rows` indicates whether the row names are written (text only).

`comment` indicates whether to write comments using the comment character, `com` (text only).

`sep` field separator character (text only).

`dec` string used for decimal points (text only).

`nas` string interpreted as `NA` value (text only).

`com` comment character (text only).

`qmethod` a string specifying how to deal with embedded double quote characters when quoting strings (text only).

`quote` if true, any character or factor columns will be surrounded by double quotes (text only).

`encoding` declares the encoding to be used on the file (text only).

`eol` the character to print at the end of each line (text only).

`zip` indicate whether the file should be compressed using gzip, bzip2, or xz (text only).

`changelog` indicate if a separate text file should be written with the change log (text only).

`ascii` if true, an ASCII representation of the data is written (R data only).

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

write.table, save, writeOGR

Examples

```
## Not run:
  Data(replace.all = obj)
  ExportData(file.type = "txt")

## End(Not run)
```

Format — *GUI: Build C-Style String Formats*

Description

A graphical user interface (GUI) for the system `sprintf` C-library function.

Usage

```
Format(sample = pi, fmt = "", parent = NULL)
```

Arguments

`sample`	logical, integer, numeric, character, or factor. Sample value
`fmt`	character. Conversion specification format, see sprintf.
`parent`	tkwin. GUI parent window

Value

Returns a character string.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

format

Examples

```
## Not run:
  Format(sample = pi, fmt = "%3.8f")
  Format(sample = 3L)
  Format(sample = TRUE)
  Format(sample = "string")

## End(Not run)
```

FormatDateTime *GUI: Build Date-Time String Formats*

Description

A graphical user interface (GUI) for converting between character representations and objects of class `POSIXt` or `Date`.

Usage

```
FormatDateTime(sample = as.POSIXct("1991-08-25 20:57:08"), fmt = "",
  parent = NULL)
```

Arguments

`sample`	POSIXt or Date. Sample date-time
`fmt`	character. Conversion specification format
`parent`	tkwin. GUI parent window

Value

Returns a character string representing the formatted date-time value.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`strptime`, `format`

Examples

```
## Not run:
  new.fmt <- FormatDateTime(fmt = "%A %B %d %I:%M %p")
  FormatDateTime(Sys.Date())

## End(Not run)
```

GetBitmapImage *Create Icon Bitmap Image*

Description

Create a small TK bitmap image.

Usage

```
GetBitmapImage(type)
```

Arguments

type	character. Icon image type, see 'Details'

Details

Icon image types include: left, right, up, down, top, bottom, upleft, upright, downleft, downright, next, previous, copy, paste, find, delete, view, info, plus, minus, print, and histogram. A recommended editor for bitmap design is Paul Obermeier's poBitmap tool; specify a square icon 11 pixels on each side.

Value

An image of class tclObj.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

tkimage.create

Examples

```
## Not run:
  types <- c("left", "right", "up", "down", "top", "bottom", "upleft", "upright",
             "downleft", "downright", "next", "previous", "copy", "paste", "find",
             "delete", "view", "info", "plus", "minus", "print", "histogram")
  Fun <- function(k) print(types[k])
  tt <- tcltk::tktoplevel(padx = 50, pady = 50)
  i <- 0
  j <- 0
  d <- 5
  for (k in seq_along(types)) {
    img <- paste("img", k, sep = ".")
    but <- paste("but", k, sep = ".")
    assign(img, GetBitmapImage(types[k]))
    assign(but, tcltk::ttkbutton(tt, width = 2, image = get(img),
                                 command = local({k <- k; function() Fun(k)})))
    tcltk::tkgrid(get(but), row = i, column = j, padx = 5, pady = 5)
    i <- k %/% d
    j <- ifelse(j < d - 1, j + 1, 0)
  }

## End(Not run)
```

`GetFile` *GUI: Select File to Open or Save As*

Description

A graphical user interface (GUI) for selecting files to open or save.

Usage

```
GetFile(cmd = c("Open", "Save As"), file = NULL, exts = NULL,
  initialdir = NULL, initialfile = NULL, defaultextension = NULL,
  win.title = cmd, multi = FALSE, parent = NULL)
```

Arguments

`cmd` character. Specifies whether an `"Open"` or `"Save As"` file management pop up dialog box is implemented.

`file` character. File name that the data are to be read from. Alternatively, `file` can be a readable text-mode `connection`.

`exts` character. Vector of default file extensions.

`initialdir` character. Files in this directory will be displayed in the dialog box.

`initialfile` character. File name to display in the dialog box.

`defaultextension` character. String appended to the file name if the user enters a file name without an extension.

`win.title` character. String to display as the title of the dialog box.

`multi` logical. If true, multiple files may be selected.

`parent` tkwin. GUI parent window

Value

If `multi` is false, returns the file path as a character object with the following attributes:

`directory`	directory containing the file
`name`	file name
`extension`	file extension
`type`	file type

Otherwise, a list is returned containing a object of class character for each file.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  GetFile()

## End(Not run)
```

ImportDataset *GUI: Import Data from Package Dataset*

Description

A graphical user interface (GUI) for importing data from selected R package datasets.

Usage

```
ImportDataset(classes = NULL, parent = NULL)
```

Arguments

`classes`	character. The object classes of data sets that can be loaded. Set to `NULL` to enable loading for all object classes.
`parent`	tkwin. GUI parent window

Value

Returns an object of list class with the following components:

`d`	table data
`src`	vector of length 3 that includes the dataset name, package name, and access date.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`data`

Examples

```
## Not run:
  obj <- ImportDataset(c("data.frame", "matrix"))

## End(Not run)
```

ImportSpreadsheet *GUI: Import Data from XML Spreadsheet File*

Description

A graphical user interface (GUI) for loading selected data sets from an Open XML Spreadsheet file (‘.xlsx’).

Usage

```
ImportSpreadsheet(parent = NULL)
```

Arguments

parent	tkwin. GUI parent window

Value

Returns an object of list class with the following components:

d	table data
src	vector of length 2 that includes the pathname of the spreadsheet file and access date.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

References

The code in this function was derived with permission from Schaun Wheeler’s xlsxToR function, accessed on 2014-01-01.

Examples

```
## Not run:
  obj <- ImportSpreadsheet()

## End(Not run)
```

`ImportText` *GUI: Import Data from Text File*

Description

A graphical user interface (GUI) for reading table formatted data from a text file.

Usage

```
ImportText(parent = NULL)
```

Arguments

`parent`	tkwin. GUI parent window

Details

This GUI is a wrapper for the `read.table` function. Data connections are defined as the path to the file to be opened, a complete URL (e.g. http://, https://, ftp:// or file://), or windows clipboard. Files are limited to text format (e.g., '`.tsv`' '`.csv`', or '`.txt`'); however, they can be compressed by gzip, bzip2, or xz with additional extension '`.gz`', '`.bz2`', or '`.xz`', respectively.

Conversion specification formats are the character representation of object types used to: identify column classes prior to reading in data, and format values for printing. Conversion specifications are based on C-style string formatting commands for `numeric`, `integer`, and `character` object classes, see `sprintf`; for example, a format string of " Calendar date and time objects of class `POSIXct` are defined by the ISO C99 / POSIX standard, see `strftime`; for example, "02/26/2010 02:05:39 PM" is represented using "

Comments located above data records and header lines are preserved; all other comments are ignored. Requires the specification of a comment character.

Performance issues associated with reading in large files can be alleviated by specifying formats in a header line, and giving the maximum number of rows to read in.

Value

Sets the following components in `Data`:

`data.raw`	imported data table.
`cols`	a list with length equal to the current number of data variables. Each component in `cols` is linked to a specific variable, see `ManageVariables`.
`comment`	vector of comment strings
`import`	a list of saved GUI options

Components of the `import` list include:

`source`	a vector of length 2 that includes the pathname of the text file and access date.

`fmts` indicates whether the file contains the conversion specification format strings of the variables.

`cols` indicates whether the file contains the names of the variables.

`skip` Number of lines skipped before data is read.

`sep` Field separator string

`dec` Used in the file for decimal points.

`na` String interpreted as `NA` values.

`quote` Set of quoting characters

`comment` Comment character

`encoding` Encoding that was assumed for input strings, see `Encoding`.

`str.as.fact` If true, character variables are converted to factors.

Note

Requires the Tcl package Tktable.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`read.table`

Examples

```
## Not run:
  ImportText()

## End(Not run)
```

LaunchGui *GUI: Main Graphical User Interface*

Description

Launches the main graphical user interface (GUI) for the **RSurvey** package. May be used to specify coordinate variables, render plots, and access all other package functionality.

Usage

```
LaunchGui()
```

Value

Quaries and sets the `vars` list component of `Data`. The components of `vars` include:

`x, y, z` index number for the corresponding coordinate-dimension variable in `cols`, see `ManageVariables` function for details.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  LaunchGui()

## End(Not run)
```

`ManagePackages` *GUI: Package Manager*

Description

This function installs R packages suggested by **RSurvey**. If a suggested package is unavailable on the local computer, an attempt is made to acquire the package from CRAN using an existing network connection.

Usage

```
ManagePackages()
```

Value

`NULL`

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`install.packages`, `requireNamespace`

Examples

```
## Not run:
  ManagePackages()

## End(Not run)
```

ManagePolygons *GUI: Polygon Manager*

Description

A graphical user interface (GUI) for managing and manipulating polygons that is based on the **rgeos** package.

Usage

```
ManagePolygons(polys = NULL, poly.data = NULL, poly.crop = NULL,
  crs = sp::CRS(as.character(NA)), parent = NULL)
```

Arguments

`polys`	list. A list of polygons, components are objects of class `gpc.poly`.
`poly.data`	character. Name of the polygon that defines the data boundary limits.
`poly.crop`	character. Name of the polygon that defines the crop region for interpolated data.
`crs`	CRS. Default coordinate reference system
`parent`	tkwin. GUI parent window

Details

The text file representation of a polygon is of the following format:

<number of contours>
<number of points in first contour>
<hole flag>
x1 y1
x2 y2
...
<number of points in second contour>
<hole flag>
x1 y1
x2 y2
...

The hole flag is either 1 to indicate a hole, or 0 for a regular contour. See the `read.polyfile` function for details.

Value

Returns an object of class list with components `polys`, `poly.data`, `poly.crop`, and `crs` (see 'Arguments' section).

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

polyfile, gUnion, SetPolygons

Examples

```
## Not run:
  ManagePolygons()

## End(Not run)
```

ManageVariables *GUI: Variable Manager*

Description

A graphical user interface (GUI) for managing variables in the data table.

Usage

```
ManageVariables(cols, vars, query, changelog, parent = NULL)
```

Arguments

`cols`	list. See ‘Value’ section
`vars`	list. See ‘Value’ section
`query`	character. See ‘Value’ section
`changelog`	data.frame. See ‘Value’ section
`parent`	tkwin. GUI parent window

Details

This GUI lets you: (1) specify the names and format of variables; (2) add new variables based on user defined functions, see `EditFunction`; (3) display data in a spreadsheet, see `EditData`; and (4) remove and (or) reorder variables in the data table.

Value

Returns an object of class list with components `cols` and `vars`. The `cols` object is a list whose length is equal to the current number of data variables. Each component in `cols` is linked to a specific variable, and contains the following components:

`name` variable name

`format` conversion specification format (optional)

`id` unique identifier that is created from `name`.

`fun` expression evaluated when computing the variables vector of values.

`index` variable's component index number in the `data.raw` data table, see `ImportText`. Only required for variables directly linked to data columns in `data.raw`.

`class` data class of the vector object.

`summary` summary of the variable's descriptive statistics (see `summary`).

`comments` user comments

The `vars` object is a list with components:

`x, y, z, sort.on`
the index number of the corresponding state variable in `cols`. These indexes are updated to reflect the removal and (or) reordering of variables in `cols`.

`query` if required, variable names are updated.

`changelog` if required, names in the `variable` component are updated.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  Data(replace.all = obj)
  ManageVariables(obj$cols, obj$vars, obj$query, obj$changelog)

## End(Not run)
```

`Plot3d` *Plot Points and Surface in 3D*

Description

This function renders raster and point data in three-dimensional (3D) space.

Usage

```
Plot3d(r = NULL, p = NULL, xlim = NULL, ylim = NULL, zlim = NULL,
  vasp = NULL, hasp = NULL, cex.pts = 1, n = NULL,
  color.palette = grDevices::terrain.colors, maxpixels = 5e+05)
```

Arguments

`r`	RasterLayer. Gridded surface data
`p`	SpatialPointsDataFrame. Spatial point data
`xlim`	numeric. Vector of length 2 giving the minimum and maximum values for the *x*-axis.
`ylim`	numeric. Vector of length 2 giving the minimum and maximum values for the *y*-axis.
`zlim`	numeric. Vector of length 2 giving the minimum and maximum values for the *z*-axis.
`vasp`	numeric. The *z/x* aspect ratio for spatial axes.
`hasp`	numeric. The *y/x* aspect ratio for spatial axes. Defaults to 1 (one unit on the *x*-axis equals one unit on the *y*-axis) when `r` is projected,
`cex.pts`	numeric. Amount by which point symbols should be magnified relative to the default.
`n`	integer. Number of contour levels desired.
`color.palette`	function. Color palette to be used to assign colors in the plot.
`maxpixels`	integer. Maximum number of cells to use for the plot.

Details

The interpolated surface is rendered using **rgl**, a 3D visualization device system for R based on OpenGL. The mouse is used for interactive viewpoint navigation where the left, right, and center mouse buttons rotate the scene, rotate the scene around the x-axis, and zooms the display, respectively.

Value

Used for the side-effect of a new plot generated.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`matplot`, `boxplot`

Examples

```
## Not run:
  Plot3d()
  rgl::rgl.quit()

## End(Not run)
```

ProgressBar *GUI: Progress Bar*

Description

A progress bar that shows the status of long-running operations.

Usage

```
ProgressBar(win.title = "Progress Bar", label = "", maximum = 100,
  nsteps = NULL, min.nsteps = 10L, parent = NULL)

SetProgressBar(pb, value, label = NULL, step = NULL)
```

Arguments

`win.title`	character. String to display as the title of the dialog box.
`label`	character. String to display in the dialog box.
`maximum`	numeric. Maximum value for the progress bar. The minimum value is zero.
`nsteps`	numeric. Total number of increments the progress bar will make.
`min.nsteps`	numeric. Minimum number of increments. If greater than `nsteps`, the dialog box is not opened.
`parent`	tkwin. graphical user interface parent window
`pb`	ProgressBar. Object returned from `ProgressBar`, see 'Value' section.
`value`	numeric. Value for the progress bar, between zero and `maximum`.
`step`	numeric. Number of progress bar increments. If equal to `nsteps`, the dialog box will close.

Value

For `ProgressBar` an object of class `"ProgressBar"` and mode `list` is returned. Components of the list object include:

`GetValue`	function that returns the value of the progress bar.
`MoveProgressBar`	function that moves progress bar, passes a numeric argument.
`SetLabel`	function that sets label in the dialog box, passes a character argument.

`DestroyWindow` function that closes the dialog box.

`GetWindowState` function that returns false if the dialog box has been closed, otherwise true.

`nsteps` see ‘Arguments’ section

For `SetProgressBar`, the previous value of the progress bar. An error is returned if the progress has terminated prematurely.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

References

The code in this function was derived from the `tkProgressBar` function, version v3.0.2.

Examples

```
## Not run:
  maximum <- 10
  label <- "Estimated time to completion is being calculated\u2026"
  pb <- ProgressBar(label = label, maximum = maximum, nsteps = maximum)

  for (i in seq_len(maximum)) {
    est.time <- system.time(Sys.sleep(1))["elapsed"] * (maximum - i)
    label <- paste("Estimated time to completion is", round(est.time), "secs")
    ans <- try(SetProgressBar(pb, value = i, label = label, step = i))
    if (inherits(ans, "try-error")) break
  }

## End(Not run)
```

`Rename` *GUI: Rename Values in Character Vector*

Description

A graphical user interface (GUI) for renaming values in a vector of character strings.

Usage

```
Rename(names = NULL, cur.name = NULL, win.title = NULL, parent = NULL)
```

Arguments

`names` character. Vector of character strings

`cur.name` character. Sets the combo box value, name must be included in `names`.

`win.title` character. String to display as the title of the dialog box.

`parent` tkwin. GUI parent window

Value

Returns a character vector with updated values of `names`.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run: 
  Rename(names = c("Name1", "Name2", "Name3"), cur.name = "Name2")

## End(Not run)
```

`Search` *GUI: Search Data Table*

Description

A graphical user interface (GUI) for establishing find and replace arguments in a data table.

Usage

```
Search(is.replace = FALSE, defaults = NULL, parent = NULL)
```

Arguments

`is.replace` logical. If true, the replace component is included.

`defaults` list. See 'Value' section

`parent` tkwin. GUI parent window

Value

Returns an object of list class with the following components:

`find.what` string to search for

`replace.with` replacement string

`is.match.word` indicates whether matches be restricted to whole words only.

`is.match.case` indicates whether the search is case sensitive.

`is.reg.exps` if true, the search is made using regular expression; that is, a pattern that describes a set of strings.

`is.search.col` indicates whether the search is limited to a single column.

`is.perl` indicates whether Perl style regular expressions should be used.

`is.replace.first`
indicates whether to replace for only the first instance.

`is.search.sel` indicates whether the search limited to selected cells.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  Search()

## End(Not run)
```

SetAxesLimits *GUI: Axes Limits*

Description

A graphical user interface (GUI) for specifying axes limits.

Usage

```
SetAxesLimits(lim = NULL, parent = NULL)
```

Arguments

`lim` list. Contains the current plotting limits, see ‘Value’ section.

`parent` tkwin. GUI parent window

Value

Returns an object of class list containing the following components:

`x1, x2` minimum and maximum `x` value.

`y1, y2` minimum and maximum `y` value.

`z1, z2` minimum and maximum `z` value.

`x1.chk, x2.chk` if true, a default value is used for the minimum and maximum `x` value.

`y1.chk, y2.chk` if true, a default value is used for the minimum and maximum `y` value.

`z1.chk, z2.chk` if true, a default value is used for the minimum and maximum `z` value.

`x` vector of `x` limits `(x1,x2)`, default is `(NA,NA)`.

`y` vector of `y` limits `(y1,y2)`, default is `(NA,NA)`.

`z` vector of `z` limits `(z1,z2)`, default is `(NA,NA)`.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  SetAxesLimits()

## End(Not run)
```

`SetConfiguration` *GUI: Window and Plotting Parameters*

Description

A graphical user interface (GUI) for specifying universal plotting parameters.

Usage

```
SetConfiguration(parent = NULL)
```

Arguments

`parent`	tkwin. GUI parent window

Value

Queries and sets the following components of `Data`:

`cex.pts`	amount by which point symbols should be magnified relative to the default value, `1.0`. For example, `cex.pts = 0.5` reduces the point symbol to half of its default size.
`nlevels`	approximate number of contour levels desired.
`asp.yx, asp.zx`	the `y/x` and `z/x` aspect ratios, respectively.
`legend.loc`	position of the points legend in the main plot region: *bottomleft*, *topleft*, *topright*, or *bottomright* to denote legend location.
`scale.loc`	position of the scale bar in the main plot region: *bottomleft*, *topleft*, *topright*, or *bottomright* to denote scale location.
`arrow.loc`	Position of the north arrow in the main plot region: *bottomleft*, *topleft*, *topright*, or *bottomright* to denote arrow location.
`useRaster`	if true, a bitmap raster is used to plot the gridded data instead of using polygons.
`draw.key`	if true, a color key should be drawn for the gridded data.
`dms.tick`	if true and the gridded data is projected, the axes tickmarks are specified in degrees, minutes, and decimal seconds (DMS).
`contour.lines`	if true, contour lines will be plotted on the 2D interpolated surface.
`make.intervals`	if true, represent point values within intervals. See `findInterval` function for details. Unused if `quantile.breaks` is true.

`proportional` indicates whether proportional circle symbols should be used to represent the point data.

`quantile.breaks` if true, breaks in the point data are set to the sample quantiles.

`bg.lines` if true, grids and graticules are drawn.

Note

Re-importing data does not affect values specified in this GUI.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  SetConfiguration()

## End(Not run)
```

SetCrs *GUI: Coordinate Reference System*

Description

A graphical user interface (GUI) for specifying PROJ.4 arguments associated with a coordinate reference system (CRS). The arguments must be entered exactly as in the PROJ.4 documentation, in particular there cannot be any white space in +<arg>=<value> strings, and successive such strings can only be separated by blanks.

Usage

```
SetCrs(crs = sp::CRS(as.character(NA)), parent = NULL)
```

Arguments

`crs` CRS. Coordinate reference system described using PROJ.4 arguments.

`parent` tkwin. GUI parent window

Value

Returns an updated value of the `crs` argument.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`CRS`, `checkCRSArgs`

Examples

```
## Not run:
  SetCrs("+init=epsg:4326")

## End(Not run)
```

`SetPlotAnnotation` *GUI: Plot Annotation*

Description

A graphical user interface (GUI) for specifying labels to add to a plot.

Usage

```
SetPlotAnnotation(parent = NULL)
```

Arguments

`parent`	tkwin. GUI parent window

Value

Queries and sets the following components of `Data`:

`credit`	mapping credit note
`explanation`	explanation of gridded-data values.
`legend.title`	title to be placed at the top of the points legend.
`legend.subtitle`	subtitle to be placed at the top of the points legend.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  SetPlotAnnotation()

## End(Not run)
```

SetPolygonLimits *GUI: Polygon Limits*

Description

A graphical user interface (GUI) for specifying polygon limits.

Usage

```
SetPolygonLimits(poly.names = NULL, poly.data = NULL, poly.crop = NULL,
  parent = NULL)
```

Arguments

`poly.names`	character. Vector of polygon names
`poly.data`	character. Name of the polygon that defines the data limits boundary.
`poly.crop`	character. Name of the polygon that defines the crop region for interpolated data.
`parent`	tkwin. GUI parent window

Value

Queries and sets the following components of `Data`:

`credit`	mapping credit note
`explanation`	explanation of gridded-data values.
`legend.title`	title to be placed at the top of the points legend.
`legend.subtitle`	subtitle to be placed at the top of the points legend.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

Examples

```
## Not run:
  SetPolygonLimits(c("Polygon1", "Polygon2", "Polygon3"))

## End(Not run)
```

SetSortOrder *GUI: Sort Order*

Description

A graphical user interface (GUI) for specifying the variable used to sort the data set.

Usage

```
SetSortOrder(col.ids, sort.on = NULL, parent = NULL)
```

Arguments

`col.ids`	character. Vector of variable names
`sort.on`	integer. Index for the variable used to sort the data set.
`parent`	tkwin. GUI parent window

Value

Returns an object of integer class that specifies the index of the variable used to sort the data set. Attributes for this object include: `decreasing`, a logical value indicating if the sort order is increasing or decreasing; and `na.last`, a logical value for controlling the treatment of `NA`s during sorting. If true, missing values in the data are put last; otherwise, they are put first; if `NA`, they are removed.

Author(s)

J.C. Fisher, U.S. Geological Survey, Idaho Water Science Center

See Also

`order`

Examples

```
## Not run:
  col.ids <- c("Variable1", "Variable2", "Variable3")
  sort.on <- 2
  attr(sort.on, "decreasing") <- TRUE
  attr(sort.on, "na.last") <- FALSE
  SetSortOrder(col.ids, sort.on)

## End(Not run)
```

Index