lockeditemindex	*Index of shape to remove from lock* *Allowed: Integer*

Returns

Description

This functions performs the opposite of `addlockedtocurrent`. *It allows the user to 'unlock' a shape from the currently selected one.*

1.3.11 skycatalog - Tool

Package display
Module viewer

Tool to create and manipulate Viewer `tool` *readable skycatalog tables*
include "skycatalog.g"

Constructors

skycatalog	*Create a skycatalog* `tool`

Functions

addunit	*Add a unit to a column*
done	*Close this* `tool`
fromascii	*Create the* `tool` *from an ascii file*
fromcomponentlist	*Create the* `tool` *from a componentlist*
mergecolumns	*Merge n columns into one*
name	*Return the name of the table*
renamecolumn	*Rename a column*
table	*Return the underling table* `tool`
type	*Return the* `tool` *type*

Description

This `tool` *allows the user to create so called skycatalog tables readable by the Viewer* `tool`. *This can be done from ASCII tables or componentlists. It also provides a front-end to the table* `tool` *to manipulate the structure of the* *skycatalog*.

Example

Here is a small example on how to convert an ascii table "my.ascii" to an AIPS++ table "my.tbl":
The ASCII file looks like this:

Name	*No*	*RA*	*DEC*	*NED-ID*	*Max*	*mom0*
A	*A*	*A*	*A*	*A*	*R*	*R*
NED	*16*	*19:44:47*	*-14:46:51*	*NGC6822ee*	*24.255*	*159.8*
NED	*22*	*01:04:46*	*+02:07:04*	*IC1613e*	*13.002*	*328.1*

```
include 'skycatalog.g';
sca := skycatalog('myscat1.tbl');
sca.fromascii(asciifile='my.ascii',hasheader=T,longcol='RA',latcol='DEC');
# cosmetics
sca.mergecolumns("Name No");
sca.done()
```

Example

An example using a componentlist:

```
im := imagemaketestimage()
cl := im.findsources()
sca := skycatalog('myscat2.tbl')
sca.fromcomponentlist(cl)
sca.done()
# That's it !
```

skycatalog - Constructor

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Create a skycatalog `tool`

Synopsis

skycatalog(filename)

Arguments

filename	*Name of the skycatalog table* *Allowed: String* *Default: unset*

Returns
skycatalog `tool` *or fail*

Description

Construct a skycatalog `tool`*. The specified filename will either be used to associate this* `tool` *wwith skycatalog table on disk or to create a new skycatalog table, which can be filled from a componentlist or ASCII file.*

---

skycatalog.addunit - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Add a unit to a column

Synopsis
addunit(colname, unit)

Arguments

colname	*column name* *Allowed: String*
unit	*Unit to put into column* *Allowed: String*

Returns

Description

---

skycatalog.done - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Close this `tool`

Synopsis

done(delete)

Arguments

delete	*delete the table* *Allowed: Boolean* *Default: F*

Returns

Description

---

skycatalog.fromascii - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Create the `tool` *from an ascii file*

Synopsis

fromascii(asciifile, hasheader, longcol, latcol, dirtype, unit)

Arguments

asciifile	*The source ASCII file* *Allowed: String* *Default: unset*
hasheader	*Does the ASCII file include a header definition?* *Allowed: Boolean* *Default: F*
longcol	*The column name or number containing the Longitude component* *Allowed: String or Integer (see description)* *Default: unset*
latcol	*The column name or number containing the Latitude component* *Allowed: String or Integer (see description)* *Default: unset*
dirtype	*The direction type of the coordinate* *Allowed: String* *Default: J2000*
unit	*The unit of the coordinates* *Allowed: String* *Default: deg*

Returns

T or F

Description

This functions converts an ASCII table into an AIPS++ skycatalog table. These tables can be read by the Viewer `tool` *and displayed as a skycatalog overlay. Whitespace in the ASCII tables is interpreted as a column separator and the number of columns per row must be constant. The ASCII table can start with a description of the columns and the data types in that column. If this header is not specified the* `tool` *tries to determine the data type and will name the columns Column1..ColumnN. In that case the columns containing the coordinate can be given as integer indices.*
The argument dirtype indicates what direction coordinate reference type your coordinate has. At the moment only J2000, B1950, GALACTIC and SUPERGAL are valid types.
The unit is assumed to be the same for long and lat and only has to be specified if the coordinate is given decimals.

---

skycatalog.fromcomponentlist - Function

Package display

Module	**viewer**
Tool	**skycatalog**

Create the `tool` *from a componentlist*

Synopsis
fromcomponentlist(code)

Arguments

code	*The componentlist* `tool` *Allowed: componentlist* `tool` *Default: unset*

Returns
T or F

Description
Invoke this function to fill the skycatalog table with components of a componentlist.

---

skycatalog.mergecolumns - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Merge n columns into one

Synopsis
mergecolumns(columns, name)

Arguments

columns	*Names of the columns to merge* *Allowed: String*
name	*name of the output column* *Allowed: String* *Default: unset*

Returns

Description
If the name argument is unset, the name of the output column is taken from the first column in column array. The output column will always be of type string.

---

skycatalog.name - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Return the name of the table

Synopsis
name()

Returns
String

Description

This function returns the name of the table.

---

skycatalog.renamecolumn - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Rename a column

Synopsis
renamecolumn(oldname, newname)

Arguments

oldname	*column name* *Allowed: String*
newname	*new column name* *Allowed: String*

Returns

Description

---

skycatalog.table - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Return the underling table `tool`

Synopsis
table()

Returns
table `tool`

Description

---

skycatalog.type - Function

Package	**display**
Module	**viewer**
Tool	**skycatalog**

Return the `tool` *type*

Synopsis
type()

Returns
skycatalog

Description

1.3.12 is_viewer - Function

Package display
Module viewer

Is the argument a viewer tool?

Synopsis
is_viewer(tool)

Description
This global function can be used to determine if the supplied argument is a viewer tool. If so it returns True, otherwise it returns False.

Arguments

tool	*The variable that you wish to test* *Allowed: Any glish variable* *Default: no default*

Returns
T or F, indicating if the argument is a viewer tool or not.

Example

```
include 'viewer.g';
if (is_viewer(dv)) {
  myviewer := dv;
} else {
  myviewer := viewer();
}
```

This example shows how you can use this function to determine if the default viewer (dv) is a viewer tool.

1.4 widgets - Module

Generic compound widgets based on Tk
include "widgetserver.g"

Description

The `widgets` *module contains GUI widgets built from basic Tk widgets. These widgets should be of general use to constructors of GUIs.*

Tools

actionoptionmenu	*Tk widget where the action menu label changes with selection*
autocli	*Tool to build a CLI from a record*
autogui	*Tool to build and operate a graphical form interface*
blinklabel	*Glish/Tk label which can be made to blink*
checkmenu	*Tk widget to provide a flat check button menu*
combobox	*A widget for selecting one or more items from a list.*
deleteregions	*Tk widget to delete image regions from a Table*
dialogbox	*Tk widget to get input from the user*
extendoptionmenu	*Tk widget where the (interactively extendable) menu label changes with selection*
fontchooserwidget	*AIPS++widget to help users select a font*
gchooser	*Tool for generating widgets for choosing items interactively*
gopher	*Tool for getting useful information*
guientry	*Tool for generating widgets for type-in of a GUI*
helpmenu	*Tk widget for a Help menu with the standard AIPS++ help items.*
histogramgui	*Window to display a histogram and allow selection of data min and max.*
listboxcontainer	*A scrolllistbox with an underlying itemcontainer.*
menuframes	*Tk widget to hide and expose a number of embedded frames on demand*
messageline	*Tk widget to display and log one-line messages.*
multiautogui	*Tool to manage a number of autoguis*
multiscale	*Glish/Tk widget to specify multiple values in a single domain*
optionmenu	*Tk widget where the menu label changes with selection*
pixelrange	*Widget to handle a pixel range specification*
popupselectmenu	*Tk widget to add a popup select menu to another widget*
profilewidthentry	*Widget to handle a 1-D profile width specification*
radiobuttons	*Widget handling multiple selectable radio buttons*
recordbrowserwidget	*AIPS++ widget to view Glish records*
restoreregions	*Tk widget to restore image regions of interest from a table*
rollup	*Tk widget to hide and expose an embedded frame on demand*
saveregions	*Tk widget to save image regions to a Table*
scrolllistbox	*A Tk listbox with optional scrollbars.*
scrolltext	*A Tk text widget with optional scrollbars.*
selectablelist	*Tk widget for selection from a menu or a listbox depending on list length*
synclistboxes	*A collection of synchronized scrolllistboxes.*
tabdialog	*Tk widget where the menu label changes with selection*