which	in	Which test to run Allowed:Integer or vector of integers Default: All tests

Returns
T or Fail

Description

Self-test of the Imagepol `tool`. This function includes forced errors. As long as the function finally returns with a `T`, it has succeeded (regardless of what error messages you might see).
The function invokes many tests. The first one is a general test of everything at a basic level. The succeeding tests work on individual areas of the image module.
Since the number of tests keeps growing, we don't tell you how many there are here or exactly what they do ! If you give too large a value, nothing will happen. `which` can be an integer or a vector of integers.

Example

```
- imagepoltest()
```

Runs all tests.

Example

```
- imagepoltest(2)
```

Runs test 2.

1.2.20 imageprofilesupport - Tool

Package general
Module images

Plot profiles with axis label conversions
include "imageprofilesupport.g"

Constructors

imageprofilesupport	Construct an imageprofilesupport tool

Functions

clearplotter	Clear the plotter
disablespectralrefmenu	Disables spectral reference change menu
done	Destroy this imageprofilesupport tool
getabcissa	Recover the profile abcissa(s)
getabcissaunit	Recover the profile abcissa unit
getabcissaunits	Recovers all the possible abcissa units
getcurrentabcissa	Recover the plotted profile abcissa(s)
getdoppler	Recover the profile doppler type
getisabs	Is the current abcissa unit absolute ?
getmask	Recover the profile mask
getoffsetdoppler	Recovers the offset doppler
getoffsetvalue	Recovers the offset value
getordinate	Recover the profile ordinate
getordinateunit	Recover the profile ordinate unit
getrefframe	Get current reference frame
getx	Get abcissa coordinate of cursor
getxy	Get abcissa and ordinate coordinate of cursor
hasplotter	Has the plotter been made yet ?
hasprofile	Has a profile been set yet ?
insertoffset	Set a new offset
makeabcissa	Make the abcissa
makemenus	Make the abcissa menus
makeordinate	Make an ordinate
makeplotter	Make the plotter
npoints	Return the number of points in the current profile
nprofiles	Return the number of profiles
plot	Plot the profile
plotfile	Saves the plot as a Table
plotter	Recover the plotter
point	Mark the specified coordinate with a symbol
postscript	Generate a postscript file of the plot
setabcissaunit	Set abcissa unit
setcoordinatesystem	Set a new coordinate system
setnoprofile	Reset number of profiles to zero
setordinate	Set an ordinate
setordinateunit	Set ordinate unit
setplotter	Set a new plotter
setprofile	Set a profile
setprofileaxis	Set the profile axis
setspectralref	Set new Spectral reference

settitle	Set plot title with fancy colors
setyrangecallback	Set ordinate data range via a callback function
which	Return indices of current plotted profiles

The imageprofilesupport `tool` is not meant as a general purpose high-level user `tool`. It is meant for use within high-level applications.
This `tool` allows you to plot a 1-D profile with built-in abcissa label coordinate conversions. For example, you can use it to plot a velocity profile and then select the abcissa units to be say km/s or GHz. You can also convert betwreen reference frames (for Spectal coordinates).
You can accumulate many profiles and display them together. You may plot all or selected profiles.
It has two construction modes. By default, `multiabcissa=F` which means there is one abcissa for one or more ordinates. In this mode, you can only use functions `makeabcissa`, `makeordinate` and **setordinate** to generate the profiles.
When `multiabcissa=T`, it is possible to have profiles which have distinct abcissas (including possibly different numbers of points). In this mode, you can only use function **setprofile** to generate the profiles.
You can optionally choose to have an abcissa offset entry box presented. If you do, it enables you to offset the spectra by some value. In particular, for Spectral axes, you can apply a velocity offset so that the spectrum is presented in the rest frame of the line.

Example

```
- im := image('ngc1313.cube')             # Open image
- csys := im.coordsys()                   # Coordinate system
- shp := im.shape()                       # Shape of image
-
- p := imageprofilesupport(csys=csys,     # imageprofilesupport tool
                           shp=shp,
                           widgetset=dws)
-
- axis := 3                            # Profile axis
- p.setprofileaxis(axis)
- p.makemenus()                           # Unit menus go into pgplotter
-
- pos := csys.referencepixel()            # Position in image
- p.makeabcissa(pos)                      # Generate abcissa along profile axis
-
- blc := pos; blc[3] := 1                 # Region to extract profile
- trc := pos; trc[3] := shp[3]
- r := drm.box(blc, trc)
```

```
- id1 := p.makeordinate (im, r)           # Generate profile ordinate
-
- p.plot()                                # Plot all profiles
-
- r := drm.box()                          # New region to extract profile from
- id2 := p.makeordinate (im, r)           # Generate second profile ordinate.
   # Non-profile axes are averaged
- p.plot()                                # Plot all profiles
- p.plot(id2)                             # Plot second profile
```

In this example, we plot a profile from an image along the third axis. The axis label menus embedded in the specified frame can be used to redraw the plot with different axis labels.

Events
This `tool` emits six events.

- unitchange: the user has selected a new abcissa unit. The value of the event is the new unit (a string).
- absrelchange: the user has selected a new abs/rel from the abcissa menu. The value of the event is T (for absolute) or F (for relative).
- dopplerchange: the user has selected a new doppler type from the abcissa menu. The value of the event is the new doppler type (a string).
- spectralrefchange: the user has selected a new spectral reference from the menu. The value of the event is the new reference (a string).
- offsetvaluechange: the user has selected a new offset value. The value of the event is the new offset (a quantum).
- offsetdopplerchange: the user has selected a new doppler offset. The value of the event is the new offset doppler (a string).

imageprofilesupport - Constructor

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Construct an imageprofilesupport tool

Synopsis
imageprofilesupport(csys, shp, widgetset, multiabcissa, offset)

Arguments

csys	in	Coordinate System Allowed:Coordsys tool
shp	in	Shape of associated image Allowed:Vector of integer
widgetset	in	The widgetset to use for the GUI Allowed:A widgetserver tool Default: dws
multiabcissa	in	Allow multiple abcissas Allowed:T or F Default: F
offset	in	Show abcissa offset entry ? Allowed:T or F Default: F

Returns
imageprofilesupport tool

Description
This constructs the **imageprofilesupport** `tool`.

Example

```
- im := image('ngc1313.cube')
- csys = im.coordsys()
- p := imageprofilesupport(csys=csys, shp=im.shape())
```

---

imageprofilesupport.clearplotter - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Clear the plotter

Synopsis
clearplotter()

Returns
T

Description

This function clears the plotter. It also resets the internal abcissa and ordinate ranges (to empty) which is a trigger to ensure the axes are redrawn.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(3);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.plot()
- p.clear()
```

---

imageprofilesupport.disablespectralrefmenu - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Disables spectral reference change menu

Synopsis
disablespectralrefmenu()

Returns
T

Description

This function disables the spectral reference conversion menu so that the user can no longer make reference conversions.

---

imageprofilesupport.done - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Destroy this imageprofilesupport tool

Synopsis
done()

Returns
Bool or fail

Description
This function destroys the imageprofilesupport `tool`.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.done()
- p
F
```

---

imageprofilesupport.getabcissa - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the profile abcissa(s)

Synopsis
getabcissa(which)

Arguments

which	in	Abcissa index Allowed:Integer Default: 1

Returns
Record

Description
This function recovers a record holding the specified abcissa. There is only once abcissa unless you are in multi-abcissa mode (set at contruction time). There are a variety of subfields such as 'pixel', 'nativeworld', and 'currentworld' which hold the abcissa in a variety of formats.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
#
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()             # Position in image
- p.makeabcissa(pos)                       # Generate abcissa along profile axis
- abc := p.getabcissa()
- field_names(abc)
pixel nativeworld currentworld name nativeunit currentunit currentveltype currentabs
```

---

imageprofilesupport.getcurrentabcissa - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the plotted profile abcissa(s)

Synopsis
getcurrentabcissa(which)

Arguments

which	in	Abcissa index Allowed:Integer Default: 1

Returns
Record

Description
This function recovers the current plotted abcissa.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
#
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()             # Position in image
- p.makeabcissa(pos)                       # Generate abcissa along profile axis
- abc := p.getcurrentabcissa()
- print shape(abc)
[64]
```

---

imageprofilesupport.getabcissaunit - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the profile abcissa unit

Synopsis
getabcissaunit()

Returns
String

Description
This function recovers the current abcissa unit.

Example

```
- im := image('ngc1313.cube')             # Open image
- csys := im.coordsys()                   # Coordinate system
- shp := im.shape()                       # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.getabcissaunit()
GHz
```

---

imageprofilesupport.getabcissaunits - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recovers all the possible abcissa units

Synopsis
getabcissaunits()

Returns
Vector of strings

Description
This function recovers all of the possible abcissa units that the user might select from.

Example

```
- im := image('ngc1313.cube')            # Open image
- csys := im.coordsys()                  # Coordinate system
- shp := im.shape()                      # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.getabcissaunits()
GHz MHz kHz Hz km/s m/s pix
```

---

imageprofilesupport.getdoppler - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the profile doppler type

Synopsis
getdoppler()

Returns
String

Description
This function recovers the current doppler type. If the profile axis is not spectral, the velocity type menu will be disabled. Regardless, this function will still return the current menu choice.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.getdoppler()
radio
```

---

imageprofilesupport.getoffsetvalue - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recovers the offset value

Synopsis
getoffsetvalue()

Returns
Quantum

Description
This function recovers the current abcissa offset.

---

imageprofilesupport.getoffsetdoppler - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recovers the offset doppler

Synopsis
getoffsetdoppler()

Returns
String

Description
This function recovers the current abcissa offset doppler.

---

imageprofilesupport.getisabs - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Is the current abcissa unit absolute ?

Synopsis
getisabs()

Returns
T or F

Description
This function returns T if the abcissa is currently showing an absolute coordinate. Else it returns F.

Example

```
- im := image('ngc1313.cube')             # Open image
- csys := im.coordsys()                   # Coordinate system
- shp := im.shape()                       # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.getisabs()
T
```

---

imageprofilesupport.getmask - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the profile mask

Synopsis

getmask(which)

Arguments

which	in	ordinate index Allowed:Integer Default: 1

Returns

Vector of Bool

Description

This function recovers the mask for the specified profile. The mask (T for good pixels, F for bad pixels) is created for you by function makeordinate and created by you explicitly with function setordinate.
Masked pixels are not plotted.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
-
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- idx := p.makeordinate (im, r)
```

```
- p.getmask(idx)
[T T T T T T T T T T T T T T T T T T T T T T T T T T T T T T T T T
T T T T T T T T T T T T T T T T T T T T T T T T T T T T T T T]
```

---

imageprofilesupport.getordinate - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the profile ordinate

Synopsis
getordinate(which)

Arguments

which	in	ordinate index Allowed:Integer Default: 1

Returns
Record

Description
This function recovers a record holding the ordinate for the specified profile. There are two subfields: 'unit' (string) and 'data' (vector float) in the record holding the ordinate.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
-
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
```

```
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- idx := p.makeordinate (im, r)
- o := p.getordinate(idx)
- print o.unit
Jy/beam
- print o.data
[0.0692182779 0.313889951 0.221464232 -0.339388996 0.196267739 0.00308039412 0.145770207
0.083701618 -0.0170051232 -0.0674641132 -0.147130117 -0.0748230442 -0.0087018162
0.042476207 0.0843885541 0.0214372594 0.0204261355 -0.0142198177 -0.150894299 -
0.292182982 -0.287655115 -0.220203012 -0.311543584 -0.696057141 -0.594372749
-0.591924787 -0.640231609 -0.751704574 -0.889986157 -0.973421454 -0.9192608
-0.853011549 -1.01908207 -1.15568936 -1.04332018 -0.943786085 -0.671383262
-0.398980319 -0.274277657 -0.602319181 -0.813411593 -0.665026307 -0.283517033
-0.198741049 -0.649372399 -0.961182058 -0.79399693 -0.628294706 -0.550167799 -0.513776541
-0.453341246 -0.401189208 -0.362751096 -0.153813794 -0.0688151047 -0.11182557 -0.106346995
-0.0373723656 -0.0238029882 0.014433424 -0.106619254 -0.173805982 -0.253587902 -0.561424077]
```

---

imageprofilesupport.getordinateunit - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the profile ordinate unit

Synopsis
getordinateunit()

Returns
String

Description
This function recovers the ordinate unit.

Example

```
- im := image('ngc1313.cube')
```

```
- csys := im.coordsys()
- shp := im.shape()
-
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- p.makeordinate (im, r)
- p.getordinateunit()
Jy/beam
```

---

imageprofilesupport.getrefframe - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Get current reference frame

Synopsis
getrefframe()

Returns
String

Description

Get the current reference frame of the profile. If you are yet to set the profile axis, this returns an empty string. Otherwise it is the selected refererence frame (for Spectral coordinates), or the native reference frame (for Direction coordinates), or blank (other coordinates).

---

imageprofilesupport.getx - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Get abcissa coordinate of cursor

Synopsis
getx(color)

Arguments

color	in	Color of location marker Allowed:PGPlot color (expressed as integer index) Default: 7

Returns
Float

Description
Get the abcissa coordinate of the cursor and mark the x-location with a vertical line.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- p.makeordinate (im, r)
- x := p.getx()
```

---

imageprofilesupport.getxy - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Get abcissa and ordinate coordinate of cursor

Synopsis
getxy(symbol, color)

Arguments

symbol	in	PGPlot symbol to mark location Allowed:Integer Default: 2 (cross)
color	in	PGPlot color of symbol Allowed:Integer Default: 1 (foreground)

Returns
Record

Description
Get the abcissa and ordinate coordinate of the cursor and mark the location. Returns a record with fields 'x' and 'y' (each holding a float).

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- p.makeordinate (im, r)
- xy := p.getxy()
```

---

imageprofilesupport.hasplotter - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Has the plotter been made yet ?

Synopsis
hasplotter()

Returns
T or F

Description
This function returns T if plotter has been created and F if it is yet to be made.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.hasplotter()
F
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.plot()
- p.hasplotter()
T
```

---

imageprofilesupport.hasprofile - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Has a profile been set yet ?

Synopsis
hasprofile()

Returns
T or F

Description
This function returns T if a profile has been created and F otherwise. The profile is considered to have been created after makeabcissa has been called. The profile is considered to have been destroyed when function setnoprofile has been called.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.hasprofile()
F
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.hasprofile()
T
- p.setnoprofile()
- p.hasprofile()
F
```

---

imageprofilesupport.insertoffset - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set a new offset

Synopsis
insertoffset(offset, doppler)

Arguments

offset	in	Offset Allowed:Quantum or quantum string Default: No change
doppler	in	Doppler Allowed:String Default: No change

Returns
T, F or fail

Description
This function sets a new offset. The plot is redrawn after.

---

imageprofilesupport.makeplotter - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Make the plotter

Synopsis
makeplotter(parent, size)

Arguments

parent	in	Parent frame for plotter Allowed:Tk agent Default: Standalone
size	in	Size of plotter Allowed:Vector of int Default: [295,215]

Returns
T or fail

Description
This function creates the plotter. If you supply a parent frame, a pgplotter

`tool` will be embedded in it. If you don't supply a parent frame, a standalone pgplotter `tool` will be created.
Calling this function is optional. If you don't call this function, then a call to plot will create a stand alone plotter (a pgplotter `tool`.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.makeplotter()
```

---

imageprofilesupport.makemenus - Function

Package general
Module images
Tool imageprofilesupport

Make the abcissa menus

Synopsis

makemenus(parent)

Arguments

parent	in	Parent frame for plotter Allowed:Tk agent Default: Use PGPlotter

Returns

T or fail

Description

This function creates three menus from which the user controls what form the abcissa labelling will take. These menus are to control whether the abcissa coordinate is absolute or relative, what the units are, and, if applicable (for

spectral profile axes) what the doppler type is. If you don't give the parent frame, the menus are embedded in the Pgplotter `tool` that is generated for plotting.
If you haven't yet specified the profile axis via function setprofileaxis. (maybe you don't know it when you have to create these menus), empty menus will be made.

Example

```
- im := image('ngc1313.cube')              # Open image
- csys := im.coordsys()                    # Coordinate system
- shp := im.shape()                        # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(3);
- p.makemenus()
```

---

imageprofilesupport.makeabcissa - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Make the abcissa

Synopsis
makeabcissa(pixel)

Arguments

pixel	in	Pixel coordinate for abcissa Allowed:Vector of integer

Returns
T or fail

Description
This function creates the abcissa. You specify the absolute pixel coordinate for which the abcissa should be generated. It will take the full range of the shape (specified at construction) for the profile axis.

You must call function setprofileaxis before calling this function so that it knows which axis is the profile axis.

Example

```
- im := image('ngc1313.cube')             # Open image
- csys := im.coordsys()                   # Coordinate system
- shp := im.shape()                       # Shape of image
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
```

---

imageprofilesupport.makeordinate - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Make an ordinate

Synopsis
makeordinate(im, region, ci, ls, which)

Arguments

im	in	The image Allowed:Image tool
region	in	The region of interest Allowed:Region tool or naked record
ci	in	PGPlot color index of this ordinate when plotted Allowed:Integer Default: 1
ls	in	PGPlot line style of this ordinate when plotted Allowed:Integer Default: 1
which	in	Which ordinate index to set Allowed:Integer or unset Default: Next slot

Returns
Integer or fail

Description
This function creates an ordinate. You specify an Image `tool` and a Region `tool`. The image data in the region are averaged over all axes except the profile axis. The region may also be specified as a naked record with fields 'blc' and 'trc' holding vectors of absolute pixel coordinates.
This function returns an index identifying the ordinate number that you are displaying. You can display many profiles simultaneously this way. To reset the number of profiles to zero use function setnoprofile.
You must call function setprofileaxis before calling this function so that it knows which axis is the profile axis.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
-
- pos := csys.referencepixel()
- p.makeabcissa(pos)
-
- blc := pos; blc[3] := 1
- trc := pos; trc[3] := shp[3]
- r := drm.box(blc, trc)
- idx := p.makeordinate (im, r)
-
- p.plot(which=idx)
```

---

imageprofilesupport.npoints - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Return the number of points in the current profile

Synopsis
npoints()

Returns
Integer

Description
This function returns the number of points in the current profile. Will be zero if none yet made. It is function makeabcissa that really determines the number of points (the ordinate must conform).

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
-
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- p.npoints()
64
```

---

imageprofilesupport.nprofiles - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Return the number of profiles

Synopsis
nprofiles()

Returns
Integer

Description
This function returns the number of profiles stored.

---

imageprofilesupport.plot - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Plot the profile

Synopsis
plot(xautoscale, yautoscale, which)

Arguments

xautoscale	in	Autoscale abcissa ? Allowed:T or F Default: T
yautoscale	in	Autoscale ordinate ? Allowed:T or F Default: T
which	in	Ordinate index Allowed:Integer or unset Default: All

Returns
T or fail

Description
This function plots all or the specified ordinate profile.
Every time you call functions makeordinate or setordinate a new ordinate profile is added to the display. Each of these functions returns an index which you can use here to specify via argument `which` which profile you want to see. By default, all profiles available are displayed.
By default the abcissa and ordinates are autoscaled - the ranges are worked out from all of the current profiles available.
If you turn off x-autoscaling, then the new plot is drawn with the current abcissa range (e.g. from a zoom).
If you turn off y-autoscaling, then the new plot is drawn with the y-range provided via the call back function setyrangecallback is used. If there is no callback, then the current range will be used.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
-
- p := imageprofilesupport(csys=csys, shp=shp,
                           widgetset=dws)
-
- f1 := dws.frame()
- axis := 3
- p.setprofileaxis(axis);
- p.makemenus()
-
- pos := csys.referencepixel()
- p.makeabcissa(pos)
-
- blc := pos; blc[3] := 1
- trc := pos; trc[3] := shp[3]
- r := drm.box(blc, trc)
- id1 := p.makeordinate (im, r)                   # Profile 1
-
- r := drm.box()
- id2 := p.makeordinate (im, r)                   # Profile 2
-
- p.plot(which=id1)                               # Plot profile 1
- p.plot(which=id2)                               # Plot profile 2
- p.plot()                                        # Plot all profiles
```

---

imageprofilesupport.plotter - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Recover the plotter

Synopsis
plotter()

Returns
pgplotter or pgplotwidget tool or fail

Description
This function recovers the underlying plotter `tool`. This will be either a PGPlotter (stand alone) or a PGPlotWidget (embedded) `tool`.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
-
- p := imageprofilesupport(csys=csys, shp=shp,
                           widgetset=dws)
-
- f1 := dws.frame()
- axis := 3
- p.setprofileaxis(axis);
- p.makemenus()
-
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- p.makeordinate (im, r)
- p.plot()
- pl := p.plotter()
```

---

imageprofilesupport.point - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Mark the specified coordinate with a symbol

Synopsis
point(x, y, x, errx, erry, symbol, color)

Arguments

x	in	Abcissa coordinate Allowed:Float
y	in	Ordinate coordinate Allowed:Float
x	in	Abcissa coordinate Allowed:Float
errx	in	Half width of x error bar Allowed:Float Default: None
erry	in	Half width of y error bar Allowed:Float Default: None
symbol	in	PGPlot marker symbol Allowed:Integer Default: 2 (cross)
color	in	PGPlot color index of symbol Allowed:Integer Default: 1 (foreground)

Returns
T

Description
This function marks the specified location with a symbol and optional error bars. The specified point is not retained in any draw lists so this is a one off draw.

---

imageprofilesupport.postscript - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Generate a postscript file of the plot

Synopsis
postscript(file, color, landscape)

Arguments

file	in	postscript file name Allowed:String
color	in	Use color ? Allowed:T or F Default: T
landscape	in	Use landscape or portrait Allowed:T or F Default: T

Returns
T or fail

Description
This function saves the current plot in a postscript file.

---

imageprofilesupport.plotfile - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Saves the plot as a Table

Synopsis
plotfile(file)

Arguments

file	in	file name Allowed:String

Returns
T or fail

Description
This function saves the current plot in an AIPS++ Table. The plot can then be reopened and new commands added or old commands changed.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys,
                           shp=shp,
                           widgetset=dws)
- f1 := dws.frame()
- axis := 3
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- p.makeordinate (im, r)
- p.plot()
- p.plotfile('save.plot')        # save
- p2 := pgplotter('save.plot')    # reopen
```

---

imageprofilesupport.setabcissaunit - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set abcissa unit

Synopsis
setabcissaunit(unit, doppler)

Arguments

unit	in	Abcissa unit Allowed:String
doppler	in	Abcissa Doppler type Allowed:String Default: unset

Returns
T

Description

This function sets the plot abcissa unit. You can only call this function after a call to makemenus. The specified unit must exist in the menu list. Any extant plot will be redrawn after this call. In general spectra will be plotted in the default RADIO doppler type when the abcissa is a velocity. This behavior can be changed by the doppler argument.

---

imageprofilesupport.setcoordinatesystem - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set a new coordinate system

Synopsis
setcoordinatesystem(csys, shp, resetref)

Arguments

csys	in	New coordinate system Allowed:Coordsys tool
shp	in	Shape of associated image Allowed:Vector of integer
resetref	in	Reset Reference Frame to Native of new Coordinate System Allowed:T or F Default: F

Returns
Integer or fail

Description

This function sets a new coordinate system and associated image shape. This function will update the x-axis menus appropriately.
After calling this function, you must remake the abcissa and ordinate profiles (functions makeabcissa and makeordinate or setordinate).
The argument `resetref` is used to reset any reference frame selection (in the GUI) back to the native one contained in the new coordinate system (or any

extra frame conversion currently set in the supplied coordinate system). Otherwise, the GUI selection is left as it is. For example, the user might have asked to convert a Spectral coordinate to TOPO. If `resetref=F` TOPO is left selected and conversions are made to it. If `resetref=T` then, if the new coordinate system has LSRK say as the native frame, then LSRK is now selected in the GUI and no extra conversions are made.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- axis := 3;
- p := imageprofilesupport(csys=csys, shp=shp)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- p.makeordinate(im, r)
- p.plot()
-
- im2 := image('ngc1399.cube')
- csys2 := im2.coordsys();
- shp2 :=  im2.shape();
- axis2 := 1;
- p.setprofileaxis(axis2)
- p.setcoordinatesystem(csys2, shp2)
- pos := csys.referencepixel()
- p.makeabcissa(pos);
- p.makeordinate(im2, r)
- p.plot()
```

---

imageprofilesupport.setnoprofile - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Reset number of profiles to zero

Synopsis
setnoprofile()

Returns
T

Description
This function resets the number of accumulated profiles to zero.

---

imageprofilesupport.setordinate - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set an ordinate

Synopsis
setordinate(data, mask, err, ci, ls, which)

Arguments

data	in	The ordinate data Allowed:Vector of float
mask	in	The ordinate mask Allowed:Vector of Bool Default: All good
err	in	The ordinate error Allowed:Vector of float Default: None
ci	in	PGPlot color index of this ordinate when plotted Allowed:Integer Default: 1
ls	in	PGPlot line style (pos) or graph marker (neg) of this ordinate when plotted Allowed:Integer Default: 1
which	in	Which ordinate index to set Allowed:Integer Default: Next slot

Returns
Integer or fail

Description

This function sets an ordinate directly, whereas function makeordinate generates it for you from an image.
If you don't set the mask it is assumed to be all T (good).
This function returns an index identifying the ordinate index that you are setting. You can display many profiles simultaneously this way. To reset the number of profiles to zero use function setnoprofile. You can ovewrite an ordinate by specifying `which`.
If argument `ls` is positive, then the points are drawn as a line with PGPLOT linestyle `ls`. If it is negative, then the points are drawn as graph markers with PGPLOT marker value `abs(ls)`.
You must call function setprofileaxis before calling this function so that it knows which axis is the profile axis.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
-
- pos := csys.referencepixel()
- p.makeabcissa(pos)
-
- blc := pos; blc[3] := 1
- trc := pos; trc[3] := shp[3]
- r := drm.box(blc, trc)
- local data, mask
- im.getregion(data, mask, region=r, dropdeg=T)
- idx := p.setordinate (data, mask, 1)
- p.plot()
```

---

imageprofilesupport.setprofile - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set a profile

Synopsis
setprofile(abcissa, ordinate, orderr, mask, unit, doppler, ci, ls, which)

Arguments

abcissa	in	The abcissa Allowed:Vector of float
ordinate	in	The ordinate Allowed:Vector of float
orderr	in	The error in the ordinate Allowed:Vector of float Default: None
mask	in	The mask Allowed:Vector of Bool Default: All good
unit	in	The abcissa unit Allowed:'pix' or valid unit Default: 'pix'
doppler	in	The doppler type Allowed:'radio', 'optical', 'true' Default: 'radio'
ci	in	PGPlot color index of this ordinate when plotted Allowed:Integer Default: 1
ls	in	PGPlot line style of this ordinate when plotted Allowed:Integer Default: 1
which	in	Which profile index to set Allowed:Integer Default: Next slot

Returns
Integer or fail

Description

This function sets a profile, both abcissa and ordinate directly. It allows spectra with distinct abcissas to be plotted together. Such simultaneous profiles can have different numbers of points as well.

This function is only available if you construct with the argument `multiabcissa=T`. Otherwise, you are restricted to one abcissa with one or more ordinates.
If you don't set the mask it is assumed to be all T (good).
This function returns an index identifying the profile index that you are setting. You can display many profiles simultaneously this way. To reset the number of profiles to zero use function setnoprofile. You can ovewrite a profile by specifying `which`.
You must specify the unit of the abcissa vector which must always be in absolute coordinates. You may use absolute pixels (i.e. those of the construction coordinate system) by setting `unit='pix'`. Otherwise `unit` can be any valid world unit consistent with the coordinate type of that axis. If the profile axis pertains to a spectral coordinate, you may use a velocity (e.g. 'm/s') or frequency (e.g. 'MHz') unit. Also for spectrla abcissas, if you specify a velocity unit, you shouyld specify the doppler type.
You can mix abcissa specification units with subsequent calls.
You must call function setprofileaxis before calling this function so that it knows which axis is the profile axis.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, multiabcissa=T)
- p.setprofileaxis(axis);
- p.makemenus()
-
- id := [];
- x := 1:100
- y := (1:100)/100.0;
- id[1] := p.setprofile (abcissa=x, ordinate=y, unit='pix');
--
- x := 120:200
- n := length(x);
- y := (1:n) / n;
- id[2] := p.setprofile (abcissa=x, ordinate=y, unit='pix');
- p.plot()
```

---

imageprofilesupport.setplotter - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set a new plotter

Synopsis
setplotter(plotter)

Arguments

plotter	in	New plotter Allowed:Pgplotter tool

Returns
T or fail

Description

This function allows you to set a plotter with one of your own creation. It must be a pgplotter `tool`.
After calling this function, you must remake the abcissa and ordinate profiles (functions makeabcissa and makeordinate or setordinate).

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- axis := 3;
- p := imageprofilesupport(csys=csys, shp=shp)
- p.setprofileaxis(axis);
- p.makemenus()
- pos := csys.referencepixel()
- p.makeabcissa(pos)
- r := drm.box()
- p.makeordinate(im, r)
- p.plot()
-
- pl := pgplotter()
- p.setplotter(pl)
- p.makemenus()
- p.makeabcissa([1,1])
- p.setordinate(data2)
- p.plot()
```

imageprofilesupport.setprofileaxis - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set the profile axis

Synopsis
setprofileaxis(axis)

Arguments

axis	in	The profile axis Allowed:Integer Default: -1

Returns
T

Description

This function sets the profile axis. The default, -1, means that no axis is effectively set. The x-unit menus will be updated appropriately, if they have already been made.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
```

---

imageprofilesupport.setordinateunit - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set ordinate unit

Synopsis
setordinateunit(unit)

Arguments

unit	in	Ordinate unit Allowed:String

Returns
T

Description

This function sets the ordinate unit. You need to do this if you are using setordinate rather than makeordinate to set the ordinate.

---

imageprofilesupport.setspectralref - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set new Spectral reference

Synopsis
setspectralref(value)

Arguments

value	in	The new spectral reference label Allowed:String

Returns
T, F or fail

Description

This function lets you set the spectral reference directly (rather than selecting the menu item interactively). When you do so, a **spectralrefchange** event is emitted. Returns F if string is invalid.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- p.setprofileaxis(axis);
- p.makemenus()
- p.setspectralref ('TOPO')
```

---

imageprofilesupport.setyrangecallback - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set ordinate data range via a callback function

Synopsis

setyrangecallback(callback)

Arguments

callback	in	Callback function Allowed:Glish function

Returns

T

Description

In function plot you may turn off profile ordinate autoscaling. If you do so, then the new display is drawn with the current ordinate range if there is one.

If there is no current ordinate range available, the ordinate range is provided via this call back function. The callback function must return a vector of two floats, the ordinate minimum and maximum.

---

imageprofilesupport.settitle - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Set plot title with fancy colors

Synopsis
settitle(text, ci)

Arguments

text	in	Title text Allowed:Vector of strings
ci	in	PGPLot colour index for each string Allowed:Vector of integers

Returns
T

Description

This function allows you to write a title with different words in different colors.

Example

```
- im := image('ngc1313.cube')
- csys := im.coordsys()
- shp := im.shape()
- p := imageprofilesupport(csys=csys, shp=shp, widgetset=dws)
- t := "A silly title";
- ci := [1,2,7];       # Reg Green Yellow
- p.settitle(t, ci)
```

---

imageprofilesupport.which - Function

Package	**general**
Module	**images**
Tool	**imageprofilesupport**

Return indices of current plotted profiles

Synopsis
which()

Returns
unset or vector of int

Description

This function allows returns the profile indices for the profiles currently plotted.

1.2.21 imageprofilesupporttest - Function

Package	**general**
Module	**images**

Run test suite

Synopsis
imageprofilesupporttest(which, destroy)

Arguments

which	in	Which test to run Allowed:Integer or vector of integer Default: All tests
destroy	in	Destroy plotters Allowed:T or F Default: T

Returns
T or fail

Description
Run the test suite.
