SAOImageDS9 Interface and Internal Design

Jonathan McDowell, William Joye, and the Chandra X-ray Center Science Data Systems Group

2025 Jul 21

Contents

Preamble

This document provides a detailed technical description of the SAOImageDS9 (”DS9”) interface, its internal design, usage instructions, supported file formats, and customization options. It is intended for developers, astronomers, and advanced users who require in-depth knowledge of DS9’s features and internal workings. The content covers historical development, build instructions, user interface components, command-line operations, data handling, coordinate systems, region definitions, and interoperability with other tools.

While every effort has been made to ensure accuracy and completeness, errors or omissions may remain. If you encounter any mistakes, unclear explanations, or outdated information, please report them to cxchelp@cfa.harvard.edu so we can continue improving this resource.

1 The DS9 main window

1.1 DS9 commands

DS9 can be controlled in three ways. Normally the application is controlled by the GUI menus. Menu items are represented in this document with the notation *Menu - Submenu - Item*.

DS9 also has a command language. The DS9 CL can be used to control DS9 from the unix command line with XPA commands, setting DS9 parameters and returning them:

```
unix> xpaset -p ds9 [command] [value]
unix> xpaget ds9 [command]
```

The same commands can also be used on the initial DS9 startup command line, in which case they are executed in sequence from left to right. A name with no preceding command is interpreted as a file to open.

```
unix> ds9 image.fits
unix> ds9 -[command1] [value1] -[command2] [value2] image.fits  -[command3] [value3] ...
```

In this document commands are denoted with **command** [**value**].

1.2 Normal startup

The core functionality of DS9, and probably the most common way that DS9 is used, is to display and analyse a 2 dimensional astronomical image file in FITS format. We suppose that the path to the file is /mydata/image.fits. Then on the command line the user types

```
ds9 /mydata/image.fits
```

If present, DS9 reads a user-customization startup file $HOME/.ds9.ini contaning TCL commands (see ’startup file’)

ds9 then starts up and opens its main window. This window has a number of panes including the image pane which shows the displayed image. I describe these in the default ’horizontal’ configuration of ds9 but other configurations are possible, from top to bottom:

1. Top row: The main menu bar, at top. (for Mac this may be separate at the top of the screen?)
2. Second row left: The info panel, showing in particular the coordinates of the image cursor
3. The panner panel, to the right of the info panel, showing a broad context view
4. The magnifier panel, to the right of the panner panel, showing a magnified view around the cursor position
5. Third row: Menu button bar array. This region contains a set of convenience buttons that replicate the main menu bar. There are two or more rows of buttons; the top row has the menu names from the main menu bar. Pressing one of these buttons populates the additional rows of buttons with selected menu items from that menu. At startup, 'file' is selected and the second row has entries from the file menu.
6. Fourth row: The image display frame pane, with a color image display showing the image pixels, with color depending on the image pixel value (usually corresponding to intensity). The position of the cursor within the image panel drives the values in the second row panels. In general only a portion of a large image is displayed at any one time (initially the central part).
7. Fifth row: The colorbar, by default below the image frame, showing the colors corresponding to different pixel values.

User actions can generate popup windows of various kinds, and menus can be torn off to make separate windows, but the main window is where most of the action is for the typical DS9 user.

Additional image frames can be created with *Frame - New Frame* (see the section on Frames). Each frame has its own colorbar. *Frame - Tile Frames* causes all currently defined frames to be simultaneously displayed side-by-side with each other. These commands are discussed in more detail in the section on frames.

There are four relevant subsets of the displayed image:

- The full image
- The part of the image DS9 has in memory at the moment ('memory image'). This is displayed in the panner window. This may be the full image, but for larger images it will not be; it can be controlled in the 'bin' menu.
- The part of the image displayed in the currently selected display frame ('display image').
- The part of the image displayed in the magnifier ('magnify image').

DS9 supports a number of operations on the displayed image - binning/blocking, cropping, smoothing, scaling, color-mapping, zooming and panning. The operations are applied in the order shown in the following figure:

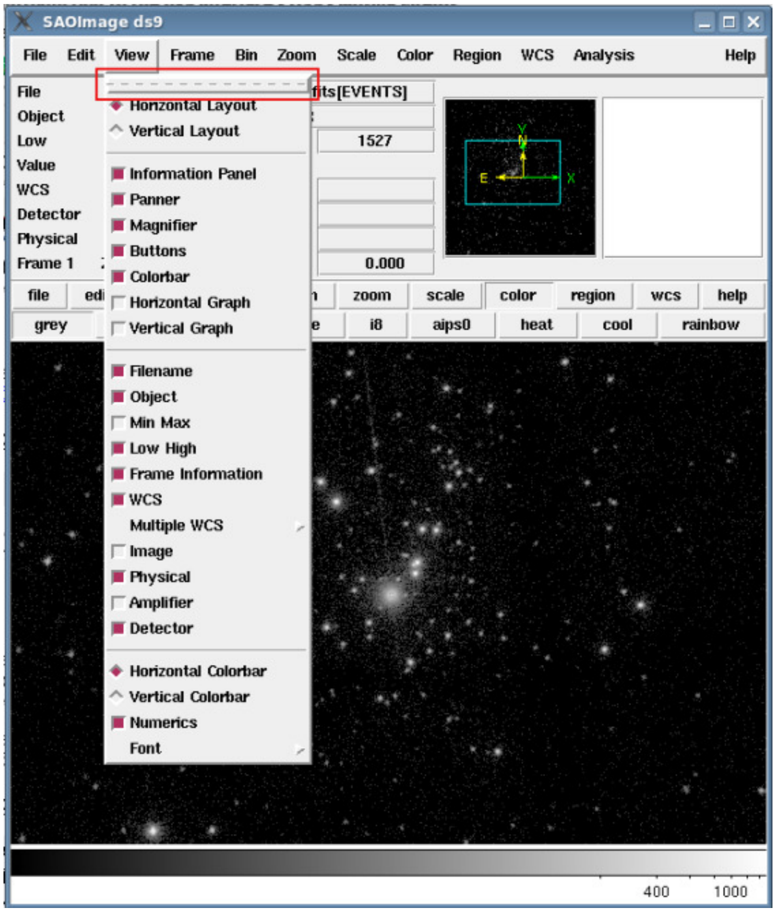


Figure 1: To ‘tear off’ a menu, click on the dashed line before the first menu entry

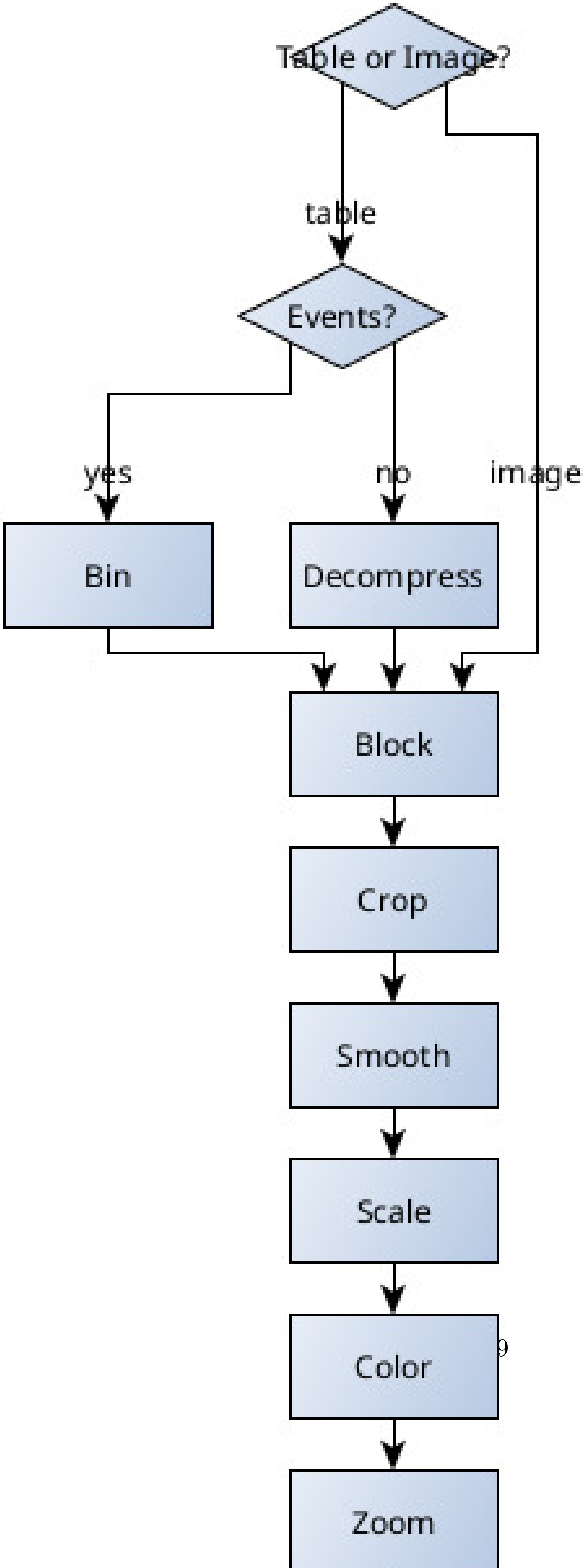


Table or Image?
table
image
Events?
yes
no
Bin
Decompress
Block
Crop
Smooth
Scale
Color
Zoom

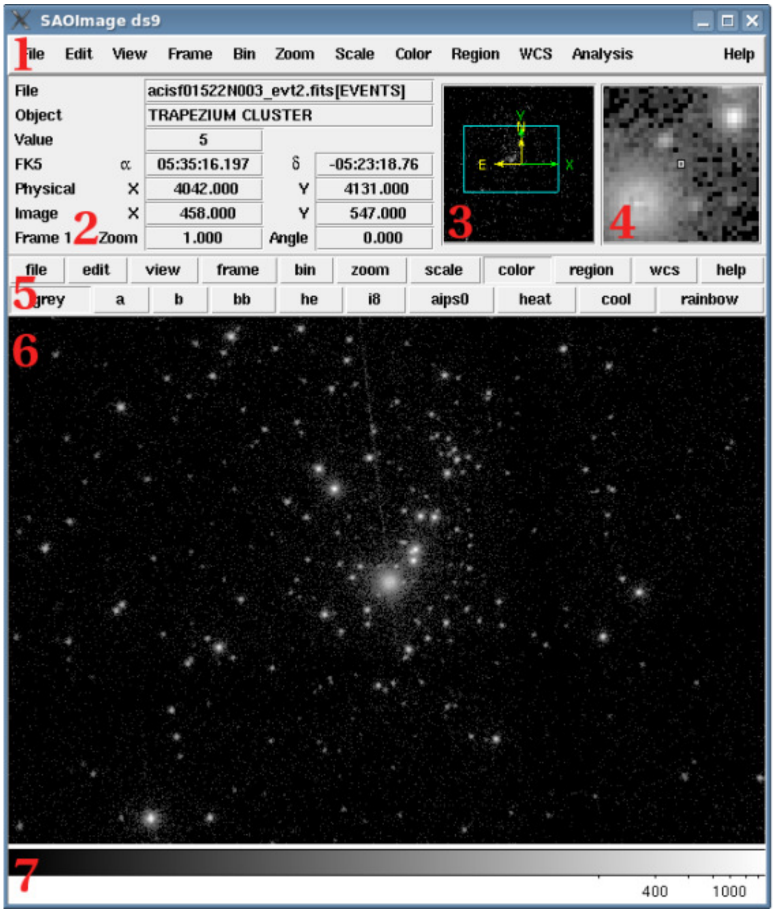
1.3 Layouts

The 'View' menu allows the user to select different DS9 layouts.

Command: **view layout horizontal**

Menu: *View - Horizontal*

Description: Selects the default view described above.



SAOImage ds9
File Edit View Frame Bin Zoom Scale Color Region WCS Analysis Help
1
2
3
4
5
6
7
E
X
Y
file edit view frame bin zoom scale color region wcs help
grey a b bb he i8 aips0 heat cool rainbow
400
1000

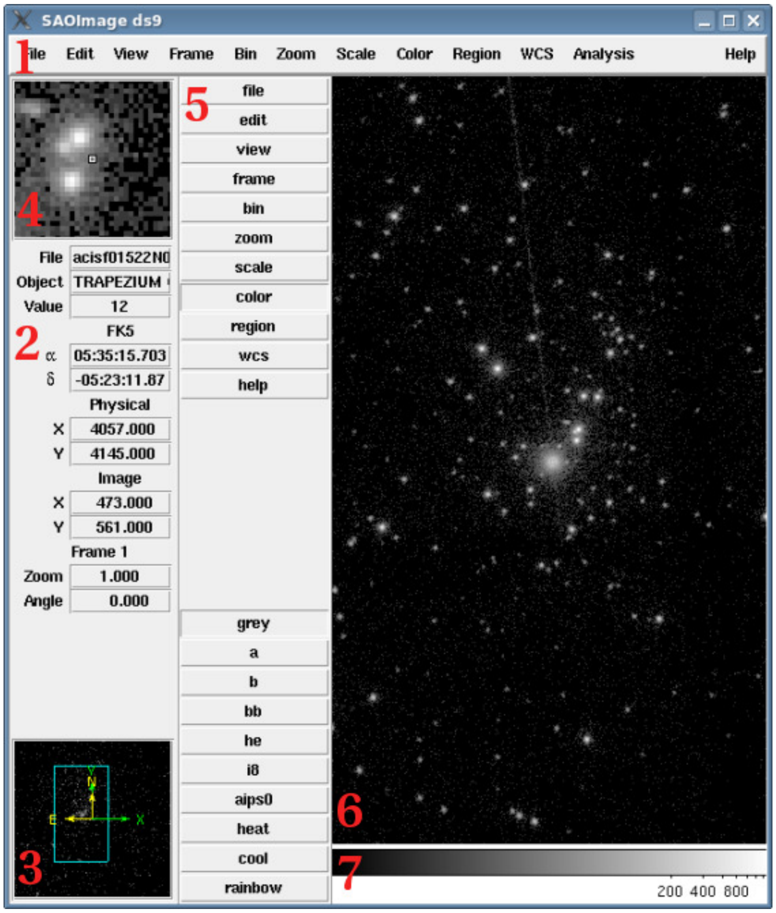
File acisf01522N003_evt2.fits[EVENTS]
Object TRAPEZIUM CLUSTER
Value 5
FK5 α 05:35:16.197 δ -05:23:18.76
Physical X 4042.000 Y 4131.000
Image X 458.000 Y 547.000
Frame 1 Zoom 1.000 Angle 0.000

Figure 3: DS9 in horizontal mode

Command: view layout vertical

Menu: *View - Vertical*

Description: In Vertical mode, a left hand column of panels has magnifier at the top, the info bar below that, and the panner at the bottom. A second column has the menu button array. The main image window is to the right. The main menu bar remains at the top.



File	acisf01522N0
Object	TRAPEZIUM
Value	12
	FK5
α	05:35:15.703
δ	-05:23:11.87
	Physical
X	4057.000
Y	4145.000
	Image
X	473.000
Y	561.000
	Frame 1
Zoom	1.000
Angle	0.000

Figure 4: DS9 in vertical mode

Command: **view layout basic**

Menu: *View - Basic*

Description: In Basic mode, the magnifier, panner, info bar and menu button array are not displayed.

Command: **view layout advanced**

Menu: *View - Advanced*

Description: In Advanced mode, the main image window is at left, a middle column has, from top to bottom, panner, magnifier, and info bar, and the menu button array is at right.

Command: **view icons yes|no**

Menu: *View - Icons*

Description: In Advanced mode only, if *View - Icons* is selected, rows of icon buttons appear at top, left and bottom

Command: **view colorbar yes|no**

Menu: *View - Colorbar*

Description: Controls whether or not the colorbar is displayed.

Command: **colorbar yes|no**

Menu: -

Description: Equivalent to 'view colorbar'.

Command: **colorbar orientation horizontal|vertical**

Menu: *Color - Colorbar - Orientation - Horizontal|Vertical*

Description: The colorbar for each of the frames can be at the bottom or the side of each frame.

Command: **height [value]**

Menu: -

Description: Sets the height of the main image window. Example: xpaset -p height 512

Command: **width [value]**

Menu: -

Description: Sets the width of the main image window.

Command: **iconify yes|no**

Menu: -

Description: Iconfiy the DS9 main window

Command: **lower|raise**

Menu: -

Description: Lower or raise the DS9 main window in the OS window manager stacking hierarchy. This may or may not work depending on the window manager/OS

1.4 Command line

The SAOImageDS9 command line in general is of the form

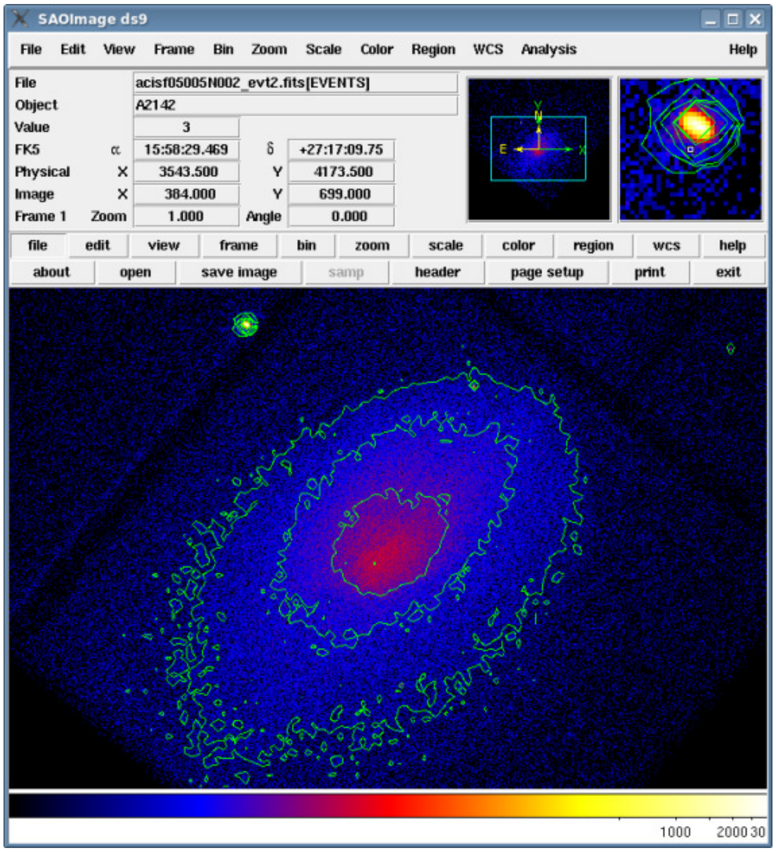
```
ds9 infile -opt1 value1 -opt2 value2 ... -optn valuen
```

The options are executed one at a time in the order they appear on the command line.

For example, to load an X-ray event list, bin and scale the image, change the colormap, draw contours and save them one might use

```
ds9 acisf05005N002_evt2.fits -bin about 3800 3800 -bin factor 2 \
    -scale log -cmap b \
    -contour yes -contour limits 1 100 \
    -contour smooth 5 -contour nlevels 6 -contour save ds9.con &
```

An even more complicated example additionally retrieves an optical image and copies the contours:



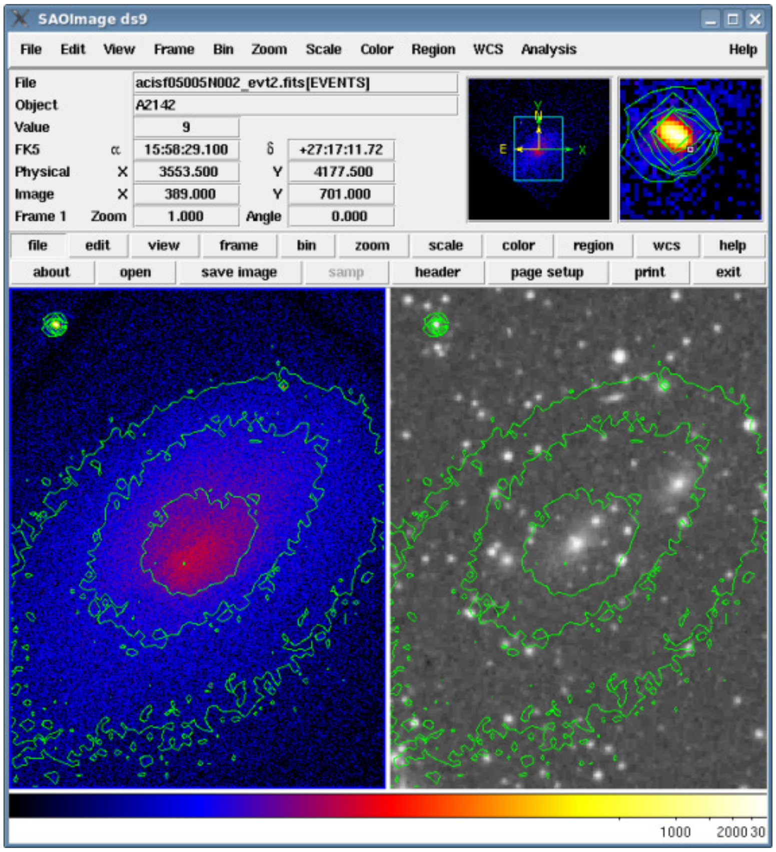
SAOImage ds9
File Edit View Frame Bin Zoom Scale Color Region WCS Analysis Help
file edit view frame bin zoom scale color region wcs help
about open save image samp header page setup print exit
N
E
Y
X
1000
2000 30

File		acisf05005N002_evt2.fits[EVENTS]		
Object		A2142		
Value		3		
FK5	α	15:58:29.469	δ	+27:17:09.75
Physical	X	3543.500	Y	4173.500
Image	X	384.000	Y	699.000
Frame 1	Zoom	1.000	Angle	0.000

Figure 5: Result of the first example contouring command line

```
ds9 acisf05005N002_evt2.fits -bin about 3800 3800 -bin factor 2 \
    -scale log -cmap b \
    -contour yes -contour limits 1 100 \
    -contour smooth 5 -contour nlevels 6 -contour copy \
    -dsssao A2142 -cmap grey -contour paste \
    -frame first -match frames wcs &
```

These options direct DS9 to:


SAOImage ds9
File Edit View Frame Bin Zoom Scale Color Region WCS Analysis Help
file edit view frame bin zoom scale color region wcs help
about open save image samp header page setup print exit
N
E
X
Y
1000
2000 30

File	acisf05005N002_evt2.fits[EVENTS]	
Object	A2142	
Value	9	
FK5	α 15:58:29.100	δ +27:17:11.72
Physical	X 3553.500	Y 4177.500
Image	X 389.000	Y 701.000
Frame 1	Zoom 1.000	Angle 0.000

Figure 6: Result of the second example contouring command line

1	bin about 3800 3800	center the image display at (x,y)=(3800,3800)
2	bin factor 2	bin the data by a factor of 2
3	scale log	set the display to log scale
4	cmap b	use the “b” colormap
5	contour yes	display contours
6	contour limits 1 100	set the minimum and maximum contour limits
7	contour smooth 5	set contour smoothness to 5
8	contour nlevels 6	create six contour levels
9	contour copy	copy the x-ray contours
10	dsssao A2142	retrieve a DSS image of A2142 from the DSS-SAO server (there are also “dsseso” and “dssstsci” options)
11	cmap grey	use the “grey” colormap in the DSS frame
12	contour paste	paste the x-ray contours onto the optical data
13	frame first	select the first DS9 frame
14	match frames wcs	match the WCS of the DSS frame to the current (x-ray) frame

1.5 Main window default behaviour

- As the user moves the cursor in the main window, the info panel updates the coordinates and values shown.

  - "Value" gives the value of the pixel the cursor is currently over.
  - "Image" gives the pixel number the cursor is over. For a 1024 x 1024 image, the pixel number runs from 1 to 1024 on each axis. In addition, there are two derived coordinate values using the WCS (world coordinate system) metadata read from the FITS file header.
  - "Physical" coordinates are usually the original instrument pixel coordinates. They are often the same as image coordinates, but for example the user may have rebinned the image by some factor so that one image pixel corresponds to, say, 4 x 4 original instrument pixels, and the metadata preserves that mapping.
  - "WCS" (or "World") coordinates are usually the celestial coordinates (RA and Dec) of the location on the sky corresponding to that pixel. The transformation between physical coordinates and world coordinates may be one of a number of nonlinear mappings; the tangent point transformation is the most common one in optical astronomy. DS9 uses the WCS metadata to dynamically compute and update the world coordinates as you move the cursor around the image.

  At the same time, the magnify image shown by the magnifier window is updated to follow the cursor position. The panner window is not affected.

- The action of the left mouse button depends on the 'edit mode'. The edit mode is 'none' by default but many users prefer to select 'region' mode (which used to be the default many years ago). In region mode, if you click left button and then drag while holding down the button, a 'region' is defined and displayed on the image pane. The default here is that the region shape is a circle, centered on the position that you intially clicked, and dragging the mouse then changes the radius, which is fixed once the button is released. The region may then be saved for later use using the Region/Save menu.

  Other edit modes are described in later sections.

- Holding down the middle button allows the user to drag the display image around, showing a different section of the memory image as the display image; e.g. dragging to the left causes the rightward portion of the memory image to be displayed.

- Holding down the right button and moving the mouse around dynamically changes the colormap; left-right movements change the bias and up-down movements change the scale.

1.6 Panner window

The panner window displays the memory image, with a rectangle superimposed indicating the subset corresponding to the main-pane display image and a 1-pixel cursor showing the current cursor location.

Dragging the rectangle using the left button will move the display image within the memory image, with the main image pane dynamically updating.

By default a 'compass' showing arrows indicating both the X and Y physical-axis directions in cyan and the N and E celestial directions in yellow is superimposed on the panner image.

Command:	**view panner yes|no**
Menu:	*View - Panner*
Description:	Controls whether the panner window pane appears or not.

Command:	**-**
Menu:	*Edit - Preferences - Panner - Show Compass*
Description:	Controls whether or not the 'compass' is displayed. Takes effect after restart.

1.7 Magnifier window

The magnifier window displays a small region centered on the main-window location of the cursor, at higher magnification. There is no mouse action available in the magnifier window, and the window blanks if you move the cursor out of the main window.

Command:	**view magnifier yes|no**
Menu:	*View - Magnifier*
Description:	Controls whether the magnifier window pane appears or not.

Command:	**magnifier zoom [N]**
Menu:	*Edit - Preferences - Magnifier - Magnification*
Description:	Controls magnification level. Values of 1, 2, 4, 8 and 16 are available.

Command:	**magnifier cursor yes|no**
Menu:	*Edit - Preferences - Magnifier - Show Cursor*
Description:	If this option is selected (on by default), at the center of the magnifier image is shown a small square indicating the cursor position.

Command: **magnifier color [color]**

Menu: *Edit - Preferences - Magnifier - Color - [color]*

Description: Controls the color of the cursor in the magnifier

.

Command: **magnifier region yes|no**

Menu: *Edit - Preferences - Magnifier - Show Graphics*

Description: Selects whether the magnifier image also shows region outlines and other overlay graphics copied from the main window.

1.8 Info panel

The info panel is laid out as a two column (name, value) table; the left hand (name) column contains variable names and the right hand (value) column contains their values. In some rows there are two or three value entries, e.g. (x,y) or (r,g,b). The values are updated dynamically as needed except in infobox freeze mode.

Command: **-**

Menu: *-*

Description: KEYBOARD SHORTCUT: f. Toggles infobox freeze mode. When set, the info panel no longer updates as you move the cursor, until you hit f again.

The variables are:

Command: **view filename yes|no**

Menu: *View - Filename*

Description: Displays the filename in the info box. On by default.

Command: **-**

Menu: *Edit - Preferences - General - Info Box - Filename format*

Description: allows some control over how the info box filename is presented (choices are 'root base', 'full base', 'root' and 'full'.)

Command: **view info yes|no**

Menu: *View - Information Panel*

Description: The ‘view info no’ command allows the user to blank out the entire info pane.

Command: **view object yes|no**

Menu: *View - Object*

Description: The object name, from the OBJECT keyword in the FITS header. On by default.

Command: **view keyword yes|no keyvalue [keywordname]**

Menu: *View - Keyword*

Description: Adds a new row with a text box in the name column with the name of a keyword. The GUI command allows the user to type the name of a keyword; the value is then populated in the corresponding row in the value column. Only one keyword can be added in this way.

Command: **view minmax yes|no**

Menu: *View - Min Max*

Description: If the minimum and maximum pixel value in the currently loaded image segment are MIN and MAX and those pixels are at image coords (XMIN, YMIN) and (XMAX, YMAX), the Minimum row displays XMIN, YMIN, MIN and the Maximum row displays XMAX, YMAX, MAX. Off by default.

Command: **view lowhigh yes|no**

Menu: *View - Low High*

Description: A ’Low High’ row is added showing MIN and MAX (but not their locations). Off by default.

Command: -

Menu: -

Description: Value - the pixel value at the current cursor position. There is no associated command for this, it is always on. For an RGB image, r,g,b pixel values are all given. KEYBOARD SHORTCUT: c. Prints current mouse coordinates and pixel value.

Command: **view units yes|no**

Menu: *View - Units*

Description: The value of the BUNIT keyword is displayed. Off by default

Command: **view wcs yes|no**

Menu: *View - WCS*

Description: WCS - the world coordinate values at the current cursor position for the default WCS for the dataset. When the box is active (the cursor is in the main window) the word 'WCS' in the name column is overridden with the WCS coordinate system name, e.g. 'ICRS', and suitable coordinate names - e.g. α, δ - are displayed next to each value.

Command: **view multi yes|no [coordsys] yes|no**

Menu: *View - Multiple WCS - [coordsys]*

Description: Rows for the additional secondary WCS systems, if present, may be added. See discussion of coordinates. For example, **view multi yes view wcsg yes** or *View - Multiple WCS - WCS g* adds a row for secondary WCS 'g'.

Command: **view physical yes|no**

Menu: *View - Physical*

Description: The x,y physical coordinates at current position, see discussion of coordinates. On by default.

Command: **view image yes|no**

Menu: *View - Image*

Description: The x,y image coordinates at current position, see discussion of coordinates. On by default.

Command: **view detector yes|no**

Menu: *View - Detector*

Description: The x,y detector coordinates at current position, see discussion of coordinates. Off by default.

Command: **view amplifier yes|no**

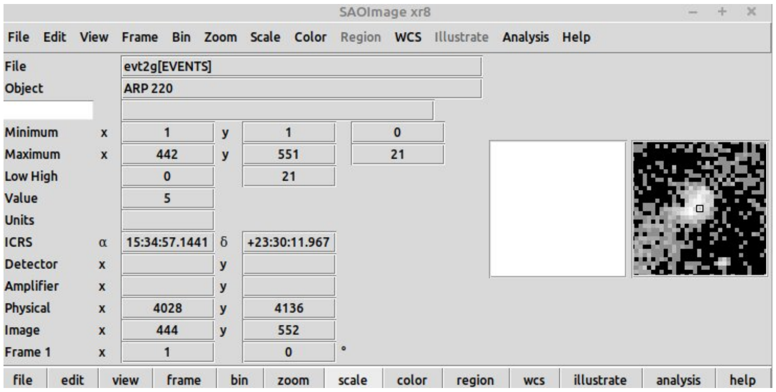
Menu: *View - Amplifier*

Description: The x,y amplifier coordinates at current position, see discussion of coordinates. Off by default.

Command: **view frame yes|no**

Menu: *View - Frame Information*

Description: If multiple frames are displayed, 'Frame 1' or 'Frame 2', etc. is displayed to indicate which frame is selected.


SAOImage xr8
File Edit View Frame Bin Zoom Scale Color Region WCS Illustrate Analysis Help
file edit view frame bin zoom scale color region wcs illustrate analysis help

File		evt2g[EVENTS]			
Object		ARP 220			
Minimum	x	1	y	1	0
Maximum	x	442	y	551	21
Low High		0		21	
Value		5			
Units					
ICRS	α	15:34:57.1441	δ	+23:30:11.967	
Detector	x		y		
Amplifier	x		y		
Physical	x	4028	y	4136	
Image	x	444	y	552	
Frame 1	x	1		0	°

Figure 7: Info box with all 'view' options selected'

1.9 Button bar

Command:	**view buttons yes|no**
Menu:	*View - Buttons*
Description:	Controls whether the button bar area is visible or not.

1.10 Mouse actions

The mouse button bindings depend on the ‘edit mode’. This can be set from the *Edit* menu or with the ’mode’ argument on the DS9 command line, e.g., ‘ds9 -mode region’.

In the default ”None” edit mode,

- Mouse motion will update the coordinates and pixel value info display.
- Left button is not active.
- Middle button pans the image (i.e. moves around the display region within the memory region). Clicking and releasing the middle button will shift the whole image within the frame. See ’Pan’ section.
- Right button changes the colorbar bias (left-right motion) and contrast (up-down motion)

The left button (button 1) modes are descibed in the relevant sections but here is a summary:

Mode	Behaviour
None	No action
Region	Create and edit regions
Crosshair	Hold down to move crosshair
Colorbar	Hold and drag to change colormap contrast and bias
Pan	Click or drag to pan to location or move location
Zoom	Zoom in centered to current location
Rotate	Hold and drag to rotate image
Crop	Click and hold to select one corner, drag and release to select opposite corner for crop rectangle
Catalog	Click on source to select and highlight in catalog table
Examine	Simulate IRAF examine
Illustrate	Create and edit regions for illustration
3D	Click and drg to rotate the face of the cube

Command: **-**

Menu: *-*

Description: KEYBOARD SHORTCUT: k (or up-arrow): move cursor up one pixel, and move selected regions up one pixel. If in crosshair mode, moves crosshair up one pixel. If in pan mode, pans image up one pixel.

Command: **-**

Menu: *-*

Description: KEYBOARD SHORTCUT: l (or right-arrow): move cursor right one pixel, and move selected regions right one pixel. If in crosshair mode, moves crosshair right one pixel. If in pan mode, pans image right one pixel.

Command: **-**

Menu: *-*

Description: KEYBOARD SHORTCUT: h (or left-arrow): move cursor left one pixel, and move selected regions left one pixel. If in crosshair mode, moves crosshair left one pixel. If in pan mode, pans image left one pixel.

Command: **-**

Menu: *-*

Description: KEYBOARD SHORTCUT: j (or down-arrow): move cursor down one pixel, and move selected regions down one pixel. If in crosshair mode, moves crosshair down one pixel. If in pan mode, pans image down one pixel.

1.11 Taking Notes

Command: **notes open**

Menu: *File - Notes*

Description: Displays a window in which the user can type notes.

Command: **notes append [text]**

Menu: -

Description: Append a line of text to the end of the notes. Any directly typed notes will be lost.

Command: **notes insert [text]**

Menu: -

Description: Append a line of text to the top of the notes. Any directly typed notes will be lost.

Command: **notes save [filename]**

Menu: *File - Notes - File - Save*

Description: Save the notes to disk. N.B. Notes are also saved and restored with DS9's backup functionality.

Command: **notes load [filename]**

Menu: *File - Notes - File - Open*

Description: Open a saved notes file

Command: **-**

Menu: *File - Notes - File - Print*

Description: Print the notes

1.12 Startup file

DS9 startup files are not typically used by typical users but are implemented by a few observatories with specialized use of DS9. You are recommended instead to explicitly source a TCL script if you need that sort of customization.

The startup file $HOME/.ds9.ini, if present, contains TCL commands that are executed at the end of DS9 initialization. This capability is used by some observatories to customize DS9 for different instruments.

You can get multiple customized startup files by soft-linking DS9 to different executable names or by using the -title command line option. For example if you do

```
cd /usr/local/bin
ln -s ds9 MegaCamRun
```

to make a soft link called ”MegaCamRun”, then executing /usr/local/bin/MegaCamRun will look for an initialization file $HOME/.MegaCamRun.ini

Alternatively running

```
ds9 -title MegaCamRun
```

will do the same thing.

1.13 Preferences files

Preference files allow the user to customize the appearance and behavior of the GUI . Please note: some preferences take effect immediately, while others require DS9 to be restarted.

User preferences files are stored in the directory $HOME/.ds9. DO NOT EDIT these files, since they may be deleted or overwritten by DS9. They consist of TCL commands to configure DS9.

Users may have several different preference files. DS9 looks for a preference file with its own name. By default, if the application is named ds9, it will look for .ds9. However, if the DS9 application is named foo, or is run with the command line option ’ds9 -title foo’, then DS9 will look for .foo. In this manner, the user can have several predefined preference files that are activated by invoking DS9 with a different application names. The same applies to .ds9.ini initialization files.

Command: **prefs save**

Menu: *Edit - Preferences - Save*

Description: Saves the current preferences to a file .ds9/ds9.8.7.prf in the user’s home directory (for DS9 version 8.7; the version number will depend on the version of DS9 in use). This file will be automatically loaded at next startup. The edit preferences dialog window is closed when a save is performed. The .prf file is an ASCII text file with DS9 TCL commands.

Command: **prefs clear**

Menu: *Edit - Preferences - Clear Preferences*

Description: Resets preferences to their defaults.

1.14 General customization - look and feel

Command: **title [name]**

Menu: -

Description: Give ds9 a title (available on command line only). This title can be used to address the DS9 instance with xpa; useful when multiple ds9 instances are running. Example:
ds9 -title imager1 &; ds9 -title imager2 &; xpaset -p imager1 fits foo.fits

Command: **language locale|da|de|es|en|fr|ja|pt**

Menu: *Edit - Preferences - General - Font - Language*

Description: localizes the dialogs to a given language (default - English; a limited number of languages are available.) Example: ds9 -language fr.

Command: **msg [directory]**

Menu: -

Description: Load a directory of translation tables. Example: ds9 -msg /share/ds9/msgs. The existing language support is in the source tree at SAOImageDS(/ds9/msgs)

Command: **width|height [value]**

Menu: *Frame - Frame Parameters - Display Size*

Description: allows the user to command a change in the DS9 main image pane to the specified number of x and y pixels. The display can also be changed with mouse draggin under the usual window managers.

Command: **prefs open**

Menu: *Edit - Preferences*

Description: brings up a dialog window to control program settings.

Command: **prefs close**

Menu: *Edit - Preferences - Close*

Description: Closes the dialog window.

Command: **-**

Menu: *Edit - Preferences - General - Enable Confirmation Dialogs*

Description: controls whether certain actions result in a confirmation dialog popping up. On by default.

Command: **-**

Menu: *Edit - Preferences - General - Font - GUI*

Description: allows the font used in the GUI to be controlled. Default is Helvetica 9 normal roman.

Command: **-**

Menu: *Edit - Preferences - General - Font - Text*

Description: allows the font used in text dialogs to be controlled. Default is Courier 9 normal roman.

Command: **prefs theme [value]**

Menu: *Edit - Preferences - General - Color - Theme*

Description: Allows selection of different window manager themes. Example - prefs theme radiance.

Command: **prefs nan color [value]**

Menu: *Edit - Preferences - General - Color - Blank/Inf/NaN color*

Description: Controls what color will be used to display pixels with value BLANK, INF or NaN.

Command: **prefs bg color yes|no**

Menu: *Edit - Preferences - General - Color - Use background color instead of theme color*

Description: Controls what color will be used for the areas of the DS9 window that have menus and buttons.

Command: **prefs bg color [value]**

Menu: *Edit - Preferences - General - Color - Background color*

Description: Controls what color will be used for parts of the main window outside the image. Default is white.

Command: **-**

Menu: *Edit - Preferences - General - Dialog Box - Motif/Windows*

Description: allows choice of Motif style or Windows style for dialog boxes.

Command: **-**

Menu: *Edit - Preferences - General - Dialog Box - Center Dialogs*

Description: if set, will ensure dialogs are centered.

Command: **-**

Menu: *Edit - Preferences - General - Dialog Box - Default All Files*

Description: In file open dialogs, all files are shown by default, not just those with the 'right' extensions. Only works for Linux.

Command: **-**

Menu: *Edit - Preferences - Menus and Buttons*

Description: allows the user to control which menu items are selected by default, and which menu items appear on the button bar in the relevant mode.

Command: -

Menu: *Edit - Preferences - General - Autoload FITS regions*

Description: Controls autoloading and display of FITS region extensions present in the same file as the main data being opened.

2 Reading and writing data - simple FITS files

Command: -

Menu: *File - Open*

Description: Open a file. Brings up a dialog box in which the user enters the file name to be opened. The left pane of the box shows subdirectories that you can navigate to, and the right pane shows files available in the current directory. You can click on one of these displayed files or just type the filename in the selection box.

By default only filenames ending in .fits, etc, are shown. However in Unix it's perfectly fine to use files with arbitrary suffix or no suffix at all, so you may perfer to click the "All" checkbox below the selection pane.

Then click OK to load the selected file. If it is read successfully, a subset of its main image (in fact the first FITS HDU containing a non-null image) will be displayed in the main image pane and the info panel will be updated appropriately.

The FITS OBJECT and UNITS keywords will be used to fill the corresponding entries in the info panel. Keywords BSCALE and BZERO will be used to scale the data as it is read in. Keyword BLANK will be used to define a null value and the DATASEC keyword will define a file subsection.

In more detail: FITS files can contain multiple segments ('extensions', 'HDUs'). When given a simple filename, DS9 examines the HDUs in turn. To determine if an HDU contains a valid image:

If the primary HDU is a non-null image, it is selected.

For other HDUs: If an HDU is a non-null image, it is selected. All valid FITS BITPIX keyword values are supported; the nonstandard value -16 for 'unsigned 2 byte integer' is also supported.

If an HDU is a BINTABLE, and if the keyword ZIMAGE is T, a FITS COMPRESSED IMAGE has been found and is selected.

If an HDU is a BINTABLE and the EXTNAME kewyord is either EVENTS, STDEVT or RAYEVENT, and columns with TTYPEn keywords with values X and Y are present, an event file has been found and is selected. See the section on event files.

If an HDU is a BINTABLE and the keyword PIXTYPE is equal to HEALPIX, a HEALPIX image has been found and is selected.

Command: **url [value]**

Menu: *File - Open As - URL*

Description: Open an image on a remote site via a URL

Command: **save fits [filename]**

Menu: *File - Save*

Description: Save the current image as a FITS file. A file selection dialog appears to pick the output filename. Once this is chosen, press OK.

Command: **save fits [filename] image|table**

Menu: *File - Save*

Description: Save the current event list (table) as a FITS image or FITS table. A file selection dialog appears to pick the output filename. Once this is chosen, press OK. If the current data is a table (event list), after this a second small dialog appears allowing you to select Image or Table.

Command: **saveimage fits [filename]**

Menu: *File - Save Image - FITS*

Description: Saves only the part of the image currently displayed, as a FITS file.

Command: **preserve pan yes|no**

Menu: *File - Preserve during Load - Pan*

Description: Preserve the pan attributes while loading a new image

---

Command: **preserve regions yes|no**

Menu: *File - Preserve during Load - Region*

Description: Preserve existing displayed regions while loading a new image

2.1 Qualified filenames

DS9 also accepts a 'qualified filename string' of the form

filename[ext][sect]

where [ext] is an extension name (EXTNAME keyword value) or extension number (1 to n) and [sect] defines a subset of the image. The syntax of the [sect] token is

```
[xspec,yspec,block,zspec]
```

Each specification is either

- Pixel range, colon-separated.
- Full range of pixels on an axis, denoted with an asterisk
- Width w around center c, denoted by wc
- A blocking factor, must be an integer in the third comma-separated field (before the Z axis spec if that is present)

Some examples:

```
foo.fits[10:200,40:100]           # Subset of 2D image
foo.fits[*,40:100]                # 2D image with y subset
foo.fits[*,*,4]                   # Full image blocked by 4
foo.fits[100:200,40:100,5:20]     # Cube subset
foo.fits[100:200,40:100,2,5:20]   # Cube subset, blocked by 2
foo.fits[256@500,256@512]         # 256 x 256 region around 500,512
foo.fits[4][10:200,40:100]        # image subset in HDU extension 4 of the file
```

3 Overall program control

3.1 Backup and restore

DS9 allows the user to back up the application's current state so it can be restored in a later session. When a backup is invoked, DS9 will save in a backup save set all files needed to restore DS9 to that state, including geometry, data files, colormaps, catalogs, contours, and regions.

A backup save set consists of a text file, called a backup script, and an optional directory, which will contain auxiliary data files needed to restore DS9 to a previous state. The backup file and the auxiliary directory maybe moved across file systems, or even platforms, but must remain together in the same directory.

By default, all data image files are saved within the backup save set. However, the user has the option, via the Preferences, to only save only an absolute pathname to the data file, and not the data file itself. This option will dramatically reduce the size of a backup save set, but will restrict the usage to a particular file system and platform.

Currently, there is no support for masks; External Analysis menus will not be saved; Plot Tool windows will not be saved; IIS frames (IRAF) will not be saved; SAMP and XPA sessions will not be saved.

Image files that have been loaded into DS9 via XPA, SAMP, or from URL will always be saved into the save set but the following load operations are not supported in that circumstance: Open Mosaic IRAF Image, Open Mosaic IRAF Segment, Open Mosaic WCS Image, Open Mosaic WCS Segment, Open Mosaic WFPC2, Open RGB Fits Image, Open RGB Fits Cube, Open RGB Array, Open Multi Ext Data Cube, Open Multi Ext Multiple Frames.

Command: **backup [filename]**

Menu: *File - Backup*

Description: Creates a DS9 .bck file with a backup of the complete current state of DS9.

Command: **restore [filename]**

Menu: *File - Restore*

Description: Restores the complete state of DS9 from a .bck backup file.

Command: -

Menu: *Edit - Preferences - General - Always save files during Backup*

Description: Ensures that the full image data file is saved.

Command: -

Menu: *Edit - Preferences - General - Auto Recovery*

Description: Sets an interval for auto backups, default 5 minutes.

3.2 Parallelism and performance

Large Files are files of size more than 4 Gbyte. DS9 is compiled with large file support and recent file systems all support these. DS9 can read files with 'mmap' memory mapping (limited by memory address space) or, slower, by allocating memory.

'ds9 foo.fits' or 'xpaset -p ds9 file foo.fits' uses mmap. 'cat foo.fits — ds9' or 'xpaset -p ds9 fits foo.fits' uses memory allocation.

Command: **prefs threads [value]**

Menu: *Edit - Preferences - General - Number of Threads*

Description: Controls the max (?) number of processing threads used by DS9.

Command: **threads [n]**

Menu: -

Description: Set number of process threads to use

Command: **sleep [n]**

Menu: -

Description: Delay execution for n seconds

Command: **update now**

Menu: -

Description: Update the image frame. Example: xpaset -p ds9 update now. This is not normally needed unless DS9 is under unusually heavy load.

Command: **update [n] [x1 y1 x2 y2]**

Menu: -

Description: Update HDU n, image subset x1:x2 y1:y2

3.3 Stopping DS9

Command: **exit**

Menu: *File - Exit*

Description: Closes all DS9 windows and exits.

Command: **quit**

Menu: *File - Exit*

Description: Quit is a synonym for exit.

4 Scale and Color

The following sections discuss the range and scaling of the displayed pixel values.

4.1 Color visuals

The default color environment for DS9 for modern devices is usually the 'truecolor24' visual, 24-bit RGB colors. In principle you can override the visual with a command line option:

```
ds9                        # default visual, default depth
ds9 -visual true           # true color, default depth
ds9 -visual truecolor      # true color, default depth
ds9 -visual truecolor8     # true color 8-bit
ds9 -visual truecolor16    # true color 16-bit
ds9 -visual truecolor24    # true color 24-bit
```

It is unlikely you will need to do this.

In more detail:

DS9 supports a number of color environments. Not all color environments, or visuals, are available on most machines. In fact, you may be restricted to one or two, based on the color graphics hardware your computer has. A color visual is composed of two parts, the color model and the bit depth. True color uses the value directly as a RGB triplet, to derive the correct color.

You can use the xdpyinfo command to see if one of these visuals are available. NOTE: Linux Users– if your desired visual is not available, use the Xconfigurator command (Red Hat) or similar command under other versions of linux, to configure your X window visuals.

When DS9 is invoked, by default, it will use the default visual. You can find out what the default visual is by using the xdpyinfo command. You can also force DS9 to use another visual by a command line option. If you specify a visual, and it is not available, DS9 will exit with an error message.

4.2 Main window pixel values

The RGB pixel value (r,g,b) displayed in DS9 for a given image pixel is an RGB-valued function F(x) of the pixel value (x) in the data, the color table $C_1(i), (i = 1, N_1)$, the min/max pixel scale limits x_1 to x_2, the color stretch a and bias b, a scaling function s(r), and an intermediate array size N. The value of N_1 is typically of order 200 and N is 16384.

- We define the base color table function as

$$c(y) = \begin{cases} C_1(1) & (y < 1) \\ C_1(\mathrm{int}(y)) & (1 \le y \le N_1) \\ C_1(N_1) & (y > N_1) \end{cases}$$

- We define the dimensionless integer-valued variable $\hat{r}$ as

$$\hat{r} = \mathrm{int}\left(\left(\frac{x - x_1}{x_2 - x_1}\right)\right)$$

  and its real-valued counterpart

$$r = \hat{r}/N$$

- and then index into the color table with

$$F(x) = c(y)$$

  where

$$y = N_1\left(as(r) + t\right)$$

  and the offset t depends on the contrast and bias via

$$t = \frac{1}{2}\left(1 - a(b + 0.5)\right)$$

This gives us the RGB triple F displayed for a given pixel value x.

Note that there are N possible values of r and thus y, and so $c(y)$ is implemented as an array of N color triples.

With the default values of $a = 1.0$, $b = 0.5$ and s linear, this reduces to $t = 0$ and

$$F(x) = c\left((N_1/N)\hat{r}\right)$$

corresponding to a simple rebinning of the color map from N_1 to N bins.

4.3 Color tags

However DS9 also supports 'color tags' which modify the F(x) function. Each tag T_j defines a range of pixel values x_j^{min} to x_j^{max} and a replacement RGB color C'_J so that

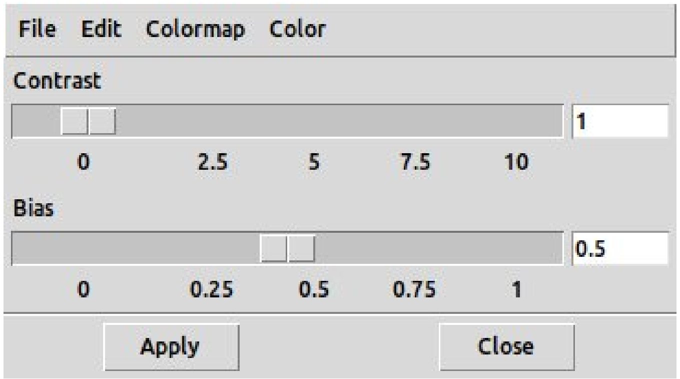
$$F(x) = C'_J$$

for the range $x_j^{\mathrm{min}} \leq x \leq x_j^{\mathrm{max}}$

The purpose of color tags are to highlight (or hide) certain values of data, regardless of the color map selected. The user creates, edits, and deletes color tags via the GUI.

To define a color tag, select *Edit - Colorbar* mode, then double-left-click in the colorbar. This defines a new color tag and pops up a dialog which allows the color C'_J to be selected by name from a list of available colors, and $x_j^{\mathrm{min}}, \quad x_j^{\mathrm{max}}$ to be specified numerically.

Command:	**cmap open**
Menu:	*Color - Colormap Parameters*
Description:	Open the Colormap Parameters dialog.



Contrast 1

Bias 0.5

Figure 8: Colormap Parameters dialog

Command: **cmap close**

Menu: *Color - Colormap Parameters - Close*

Description: Close the Colormap Parameters dialog.

Command: **cmap tag save [filename]**

Menu: *Color - Colormap Parameters - Save Color Tags*

Description: Save color tags defined for current frame.

Command: **cmap tag load [filename]**

Menu: *Color - Colormap Parameters - Load Color Tags*

Description: Loads a tags file saved earlier.

Command: **cmap tag delete**

Menu: *Color - Colormap Parameters - Delete Color Tags*

Description: Deletes tags currently defined in the frame.

Command: -

Menu: *Color - Colormap Parameters - Color*

GUI Default: *Edit - Preferences - Color - Colorbar - Tag color*

Description: Menu allows the user to preselect the color that will be used when a new color tag is selected.

Command: -

Menu: *Color - Colormap Parameters - Download colormap*

Description: takes you to a web page where you can find an index to various extra colormaps. Click through to find one you like, selecting ones with 'sao' or 'lut' filetype. This opens a new frame with a picture of the colormap. You can load an image into this frame.
On reading a colormap:
A .lut file is assumed to be a XImtool/SAOtng format colormap.
A .sao file is assumed to be a DS9/SAOImage format colormap.
Any other filename extension is assumed to be a DS9 format one.

Command: **cmap save [filename]**

Menu: *Color - Colormap Parameters - File - Save*

Description: Save the colormap in .sao format.

Command: **cmap load [filename]**

Menu: *Color - Colormap Parameters - File - Open*

Description: Load a colormap.

4.4 Detailed derivation

In more detail: We have an rgb-valued color scale C(r) that maps an integer r from 1 to N (N =16384) to a color triple.

The color scale is constructed as follows. We read a base colormap file $C_1(i), i = 1$ to N_1, where typically N_1 is of order 200.

The base colormap is stretched to an array C_2 of size $N_2 = 4096$ by

$$C_2(j) = C_1((N_1/N_2)j)$$

We then apply a contrast a to stretch the array values (but in a same size array $N_3 = N_2$)

$$C_3(j) = C_2(N_2/2 + (j - N_2/2) * a)$$

and a bias b (valued 0 to 1) to give array C_4, of size $N_4 = N_3 = N_2$,

$$C_4(j) = C_3(j - (b - 0.5) * N_2/2)$$

We now stretch again to size $N = 16384$

$$C_5(j) = C_4(j * N_4/N)$$

and apply a nonlinear scale function s(x),

$$C(j) = C_5(Ns(j/N)) = C_4(N_4 s(j/N)$$

so that

$$\begin{aligned}
C(j) &= C_5(Ns(j/N)) \\
&= C_4(N_4 s(j/N)) \\
&= C_3(N_4 s(j/N) - (b - 0.5) * N_2/2) \\
&= C_2(N_2/2 + a * (N_4 s(j/N) - (b - 0.5) * N_2/2 - N_2/2)) \\
&= C_1((N_1/N_2) * (N_2/2 + a * (N_4 s(j/N) - (b - 0.5) * N_2/2 - N_2/2)))
\end{aligned}$$

Or

$$C(j) = C_1(k)$$

where

$$k = (N_1/2 + aN_1 s(j/N) - a((b + 0.5) * N_1/2))$$

or

$$k/N1 = (as(j/N) + (1 - a(b + 0.5))/2$$

So for a = 1, b = 0.5, s= 1 we have $k = N_1 j/N$ as expected.

The contrast and bias a and b can be set in the *Color - Colormap Parameters* dialog:

Command: **cmap [a] [b]**

Menu: *Color - Colormap Parameters - Contrast/Bias*

Description: Set contrast and bias values: example: cmap 5 0.5

The two floating point values a and b can be saved in a text file with *Color - Colormap Parameters - Save Contrast/* and restored from that file with *Color - Colormap Parameters - Load Contrast/Bias*.

4.5 Supported scaling functions

The scaling functions are:

linear	Linear	$s(r) = r$
log	Log	$s(r) = \log(pr + 1)/\log(p)$, for a parameter p whose default is 1000.
power	Power	$s(r) = \left(\frac{p^r - 1}{p}\right)$, for parameter p with default 1000.
sqr	Sq Root	$s(r) = \sqrt{r}$
sqr	Squared	$s(r) = r^2$
asinh	asinh	$s(r) = \sinh^{-1}(pr)/3$, where p = 10.
sinh	sinh	$s(r) = \sinh(3r/p)$, where p = 10.
histeq	Histeq	with $s(r)$ depending on the frequency of each r value.
logexp	Log exponent	

Command: **scale [function]**

Menu: *Scale - [function]*

Description: Allows the user to select which scaling function s(r) is used. For example, 'xpaset -p ds9 scale asinh' On the command line, one can use the functions as options directly, so 'ds9 -linear', 'ds9 -log', etc.

.

Command: **scale log exp [value]**

Menu: *Edit - Preferences - Scale - Log Exponent*

Description: Controls the value of p in the log and power functions

.

4.6 Calculating scale range limits

Command: **scale open**

Menu: *Scale - Scale Parameters*

Description: Generates a popup dialog which allows the user to enter the x_1 and x_2 values explicitly. The Scale menu also lets the user specify one of several range limit algorithms to calculate and apply suitable values.

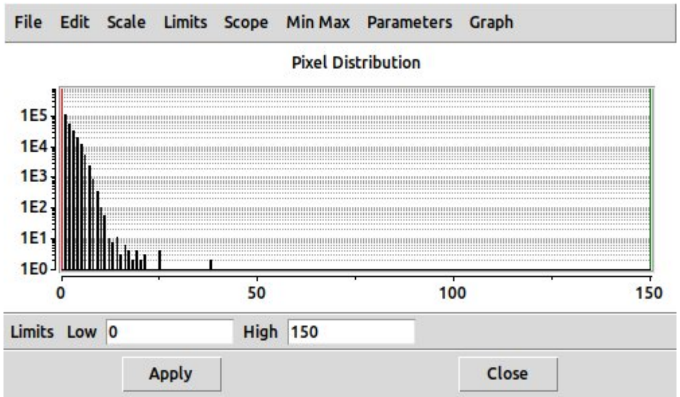


Figure 9: Scale parameters dialog

Command:	**scale close**
Menu:	*Scale - Scale Parameters - Close*
Description:	Closes the dialog

Command:	**scale limits [min] [max]**
Menu:	*Scale - Min Max*
Description:	Min Max: use the minimum and maximum values in the data.

Command:	**-**
Menu:	*Scale - Scale Parameters - Preset Percentiles*
Description:	Set the min and max values using the Xth percentile of the distribution of data values. The user can select X = 99.5, 99, 98, 97, 96, 95, 92.5 or 90.

Command:	**scale mode [X]**
Menu:	*Scale - Other - [X]*
Description:	Set the min and max values using the Xth percentile of the distribution of data values.

Command: **zscale**

Menu: *Scale - Zscale (1)*

Description: The first *Scale - ZScale* menu entry selects this algorithm. ZSCALE is an algorithm inherited from IRAF. The algorithm has three parameters: Contrast (C), Number of Samples (N), and Samples per Line (L).

Command: **scale mode zscale**

Menu: *Scale - Zscale (1)*

Description: Apply Zscale algorithm.
The algorithm defines a sample of N pixels evenly spaced in x and y (so for example if N=36, L = 6 and the image is 50 x 50, then the pixels selected would be spaced 10 apart, starting with (1,1), (11,1), (21,1) ... and ending with (50,50), more or less. The values of those pixels are sorted to provide a monotonic increasing array V(i), i = 1,N, and the midpoint m = N/2 of this array is the median M = V(m). A line of slope S is fit to the V(i) curve, rejecting poorly fitting points near i=1 and i=N using an iterative approach. We then set

$$x_1 = V(m) + (S/C)(1 - m)$$

and

$$x_2 = V(M) + (S/C)(N - m)$$

Command: **-**

Menu: *Scale - Zscale (2)*

Description: The second *Scale - ZScale* menu pops up a *Scale - ZScale - ZScale Parameters* dialog where the ZScale C, N and L parameters can be changed.

These can also be set with

Command: **zscale contrast [value]**

Menu: *Edit - Preferences - Scale - ZScale Parameters - Contrast*

Description: Sets default contrast scale for ZScale

Command: **zscale sample [value]**

Menu: *Edit - Preferences - Scale - ZScale Parameters - Number of Samples*

Description: Sets default number of samples for ZScale

Command: **zscale line [value]**

Menu: *Edit - Preferences - Scale - ZScale Parameters - Samples per Line*

Description: Sets default samples per line for ZScale

Command: **scale mode zmax**

Menu: *Scale - Zmax*

Description: ZMAX: uses the x_1 value calculated by the ZScale algorithm and sets x_2 to the max value of the data.

Command: **-**

Menu: *Scale - User*

Description: This menu entry is checked if the user has entered explicit low, high values in Scale Parameters. If you then change to another scale method (e.g. Scale - Zscale), subsequently selecting Scale - User will remember and restore the most recent user-selected low-high values.

Command: **scale mode minmax**

Menu: *Scale - MinMax*

Description: Specify methods to sample only a subset of the data to determine the min/max. The default, Scan, uses all the data. Sample uses only every nth data point, where n can be set in the *Scale - MinMax - SampleParameters* dialog. If the FITS header keywords DATAMIN/DATAMAX or IRAF-MIN/IRAF-MAX are present in the file header, these may also be chosen to set the min/max values.

Command: **minmax mode scan|sample|datamin|irafmin**

Menu: *Scale - Scale Parameters - Min Max - Scan|Sample|DATAMIN|IRAFMIN*

GUI Default: *Edit - Preferences - Scale - Min Max Parameters - Sample Increment*

Description: Determine min max by scanning all data, sampling every N values, using the DATAMIN keyword if present, using the IRAFMIN keyword if present.

Command: **minmax interval [n]**

Menu: *Scale - Scale Parameters - Min Max - Sample Parameters*

Description: Set the sampling interval for the ’minmax mode sample’ option.

Command: **minmax rescan**

Menu: -

Description: Rescan the data for minmax values

Command: **scale dataset yes|no**

Menu: *Scale - DATASEC*

Description: By default, if the DATASEC keyword is present in the header, its value is used to determine what subset of the data is considered valid data for min/max calculations and for display. The value of this NOAO/IRAF keyword is a string of the form ‘[x1:x2,y1:y2]’, for example ‘[13:628,1:568]’. This command is used to enable or disable DS9’s use of this keyword to filter the data.

4.7 Selecting and overriding the colormap

A wide selection of colormaps are provided by DS9. These are selected by the *Color* menu entries:

Command: **cmap [value]**

Menu: *Color - [value]*

Description: Select colormap.

The color menu also supports the functions

Command: **cmap invert yes|no**

Menu: *Color - Invert colormap*

Description: Inverts the colormap

Command: **-**

Menu: *Color - Reset colormap*

Description: Reset colormap to original state

It is possible to override sections of the colormap by defining 'color tags'.

4.8 Colorbar

Command: **colorbar numerics yes|no**

Menu: *Color - Colorbar - Numerics - Show*

Description: Displays numerical values along each colorbar (on by default).

Command: **colorbar ticks [value]**

Menu: *Color - Colorbar - Number of Ticks*

GUI Default: *Edit - Preferences - Color - Colorbar - Number of Ticks*

Description: The number of values displayed is controlled, but only approximately, by this command

Command: **colorbar space value|distance**

Menu: *Color - Colorbar - Numerics - Space Equal Value|Distance*

Description: If Space Equal Value is set, the values displayed are equally spaced in value x. If Space Equal Distance is set, the values displayed are for points in the colorbar an equal distance apart in scaled value F(x) on the displayed colorbar. This is the default and usually makes more sense when a logarithmic scale is selected.

Command: **colorbar font [value]**

Menu: *Color - Colorbar - Font*

Description: Controls the font of the colobar numeric labels - e.g. 'colorbar font times'

Command: **colorbar fontslant roman|italic**

Menu: *Color - Colorbar - Font - Style*

Description: Controls the font style (roman or italic) of the colobar numeric labels

Command: **colorbar fontsize [value]**

Menu: *Color - Colorbar - Font - Size*

Description: Controls the font size of the colobar numeric labels

Command: **colorbar fontweight normal|bold**

Menu: *Color - Colorbar - Font - Normal|Bold*

Description: Controls the font weight of the colobar numeric labels

Command: **colorbar size [value]**

Menu: *Color - Colorbar - Size*

GUI Default: *Edit - Preferences - Color - Colorbar - Colorbar size*

Description: Allows you to enter a size for the thickness of the colorbars.
Changes applied in the *Color - Colorbar* menu affect all the displayed colorbars for all the frames.

Command: **mode colorbar**

Menu: *Edit - Colorbar*

Description: In Colorbar mode, the left button has the same function as the right button, controlling colorbar bias and contrast.

5 Block, Bin, Zoom and Pan

In DS9, the main image can be zoomed in and out to get a magnified view of a small area or a wide view of a large area of the image. The user can also 'block' the image; 'block by 4' means 'take each 4x4 pixel in the original image and replace it by a single pixel in the new image. The value in the binned pixel can be specified to be either the sum of the original pixel values (appropriate in, e.g., an image where the pixel values are photon counts) or their average (approriate in an image where the pixel values are flux per square arcsecond, etc.).

For event list files, one can also 'bin' the image. This is similar to block but operates on the table colummns used to make the image.

By default, a 1024 x 1024 subset (buffer) of the loaded image is displayed, centered on the image center. The user can change the binning of the image, can change the buffer size, and can zoom in and out.

5.1 Block factor

Command: **block open**

Menu: *Analysis - Block - Block Parameters*

Description: Open the block parameter dialog.

Command: **block close**

Menu: *Analysis - Block - Block Parameters - Close*

Description: Close the block parameter dialog.

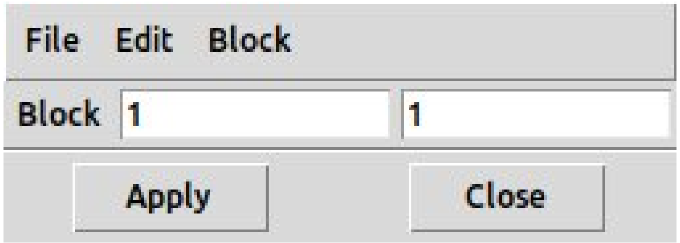


Figure 10: Block Parameters dialog

Command:	**block [N] (or block to [N])**
Menu:	*Analysis - Block - Block N*
Description:	Block by N, where N = 1, 2, 4, 8, 16 or 32, rebins the image so that each pixel in the new displayed image is an NxN group of pixels in the source image.

Command:	**block [N] [M]**
Menu:	*Analysis- Block - Block Parameters - Apply*
Description:	Does the same thing as block [N], but allows different block factors in X and Y.

Command:	**block in|out**
Menu:	*Analysis- Block - Block In/Out*
Description:	Increases or decreases the current blocking level by one factor of 2.

Command:	**block to fit**
Menu:	*Analysis- Block - Block Fit*
Description:	Blocks the image to a level where the entire image fits in the display area.

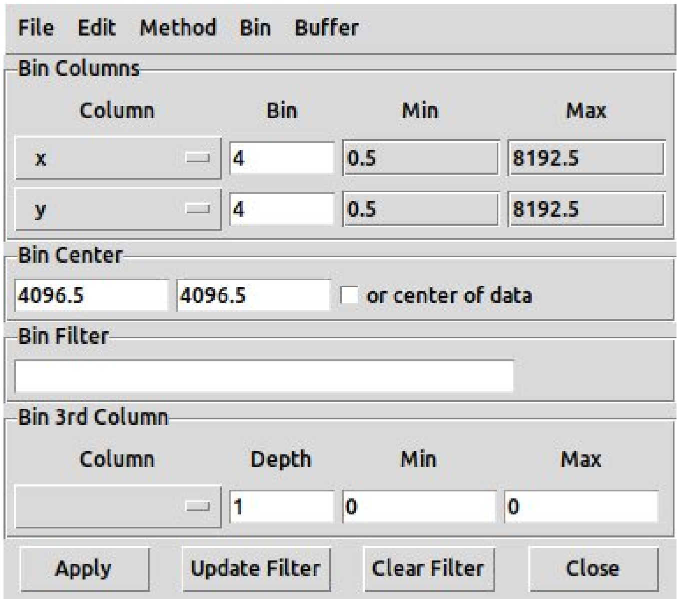
5.2 Bin factor

Binning is only available for table data such as event lists, where the image is made on the fly from a 2D histogram of event coordinate values.

Command: **bin open**

Menu: *Bin - Binning Parameters*

Description: Open the binning parameters menu.



Bin Columns

Column	Bin	Min	Max
x	4	0.5	8192.5
y	4	0.5	8192.5

Bin Center

4096.5 4096.5 □ or center of data

Bin Filter

Bin 3rd Column

Column	Depth	Min	Max
	1	0	0

Figure 11: Binning Parameters dialog

Command: **bin close**

Menu: *Bin - Binning Parameters - Close*

Description: Close the binning parameters menu.

Command: **bin factor [value]**

Menu: *Bin - Bin [value]*

Description: Rebin so the binned pixel size is [value] times the original. Supported steps in the menu are N = 1, 2, 4, 8, 16, 32, 64, 128, 256. Arbitrary floating point values (including values less than 1) are supported in the Bin - Binning Parameters dialog.
Causes the displayed image to be rebinned by that factor compared to the original image. The original resolution is retained in memory, it's just the display that changes.

Command: **bin in|out**

Menu: *Bin - In|Out*

Description: Set the binning to one step less or more than the current binning level. So if the current bin factor is 8, 'Bin - In' sets the bin factor to 4 and 'Bin - Out' sets it to 16.

Command: **bin to fit**

Menu: *Bin - Fit*

Description: Will select a binning factor that ensures the whole image is displayed.

Command: **bin factor [N] [M]**

Menu: *Bin - Binning Parameters - Bin*

Description: Bin the image by N x M with N and M entered as arbitrary values.

Command: **bin about [x] [y]**

Menu: *Bin - Binning Parameters - Bin Center - Apply*

Description: Select the physical coordinates of the bin center. By default, binning uses the center of the image.

Command: **bin about center**

Menu: *Bin - Binning Parameters - Bin Center - center of data*

Description: Bin about the center of the image.

Because the pixels in an image made from an event list may originate from histograms of a real-valued position coordinate, binning can be done by a real-valued factor. It is otherwise the same as blocking of a true image.

The buffer size options are M x M where M = 128, 256, 512, 1024, 2048, 4096, 8192.

Command: **bin buffersize [value]**

Menu: *Bin - M x M*

Description: Set the size of the binning buffer.

Command: **bin function average|sum**

Menu: *Bin - Average|Sum*

Description: Specify whether binned pixel values are the average or the sum of individual pixels. (Also *Bin- Binning Parameters - Method*).

5.3 Zooming and Buffer size

The zoom function displays the value of the pixel closest to the location being rendered. So zoom by 1/4 (zooming out) will only display 1 out of 16 pixels in the original image, while zoom by 4 (zooming in) will display the same pixel 16 times.

Note that if you have opened an event file with X and Y columns that have a large range (e.g 0.5 to 8192.5), by default only a 1024 x 1024 central subset is displayed. Binning by 2 displays a 1024 x 1024 image corresponding to the central 2048 x 2048 subset of the original image. **However,** zooming out by 2 (i.e. zoom = 1/2) still displays only the original 1024 x 1024, with whitespace around it; a change in zoom factor does not change the loaded subset. Selecting *Bin - 2048 x 2048* with zoom set to 1/2 will display the central 2048 x 2048 at full resolution.

To summarize: changing the bin factor will change which subset of the event file is used. Changing the zoom factor will not - you need to change the buffer size as well.

Command: **mode zoom**

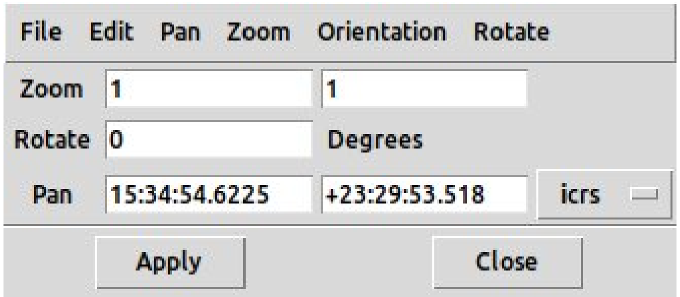
Menu: *Edit - Zoom*

Description: In Zoom mode, clicking the left button will zoom the image in, without moving the current image location. Shift-clicking the left button will zoom back out. Middle button scroll, if available, will also control the zoom.
You can also use the *Zoom* menu and in particular the *Zoom - Pan Zoom Rotate Parameters* dialog.

Command: **pan|rotate|orient|zoom open**

Menu: *Zoom - Pan Zoom Rotate Parameters*

Description: Opens a dialog which allows you to modify and reset the zoom and pan parameters.



Zoom 1 1

Rotate 0 Degrees

Pan 15:34:54.6225 +23:29:53.518 icrs

Figure 12: Pan Zoom Rotate parameter dialog

Command: **pan|rotate|orient|zoom close**

Menu: *Zoom - Pan Zoom Rotate Parameters - Close*

Description: Close the rotate/zoom/pan parameters dialog

Command: **zoom [n]**

Menu: *Zoom - [n]*

Description: (n = 1/32, 1/16, 1/8, 1/4, 1/2, 1, 2, 4, 6, 8, 16, 32). Zoom to given factor.

Command: **zoom [n][m]**

Menu: *Zoom - Pan Zoom Rotate Parameters - Zoom*

Description: Zoom different amoutns in x and y

Command: **zoom in|out**

Menu: *Zoom - In|Out*

Description: changes the zoom factor by one step in either direction. KEYBOARD SHORTCUT: z (in), Z (out)

Command: **zoom to fit**

Menu: *Zoom - Fit*

Description: attempts to pick a zoom factor in which the entire image subset is displayed.

5.4 Orientation and Rotation

Command: **orient none**

Menu: *Zoom - None*

Description: displays the image with the X, Y axes in their usual orientation.

Command: **orient x|y|xy**

Menu: *Zoom - Invert X|Y|XY*

Description: Inverts one or both of the axes.

Command: **-**

Menu: *Edit - Rotate*

Description: In Rotate mode, holding and dragging the left button allows you to rotate the image in the frame.

Command: **rotate to [M]**

Menu: *Zoom - M degrees*

Description: Menu option is M = 0, 90, 180 or 270, or an arbitary value in the dialog, rotates the image by the given amount.

5.5 Pan

Clicking with the middle button changes the displayed center location.

Command: **mode pan**

Menu: *Edit - Pan*

Description: In Pan mode, left button has the same action as middle button. Clicking on a location will move that location in the image to the center of the frame. Dragging has no effect.
For some reason I always expect that clicking at a point in the image/frame would move the image center to that frame location, rather than vice versa. It doesn't.

Command: **-**

Menu: *Zoom - Pan Zoom Rotate parameters*

Description: Allows you to modify and reset the zoom and pan parameters.

Command: **-**

Menu: *Edit - Preferences - Pan Zoom - Click to Center*

Description: In this, the default mode, the new center of the display region will be the location clicked.

Command: **-**

Menu: *Edit - Preferences - Pan Zoom - Drag to Center*

Description: Alternative pan mode

Command: -

Menu: *Edit - Preferences - Pan Zoom - Pan then Zoom*

Description: Alternative pan mode

Command: -

Menu: *Edit - Preferences - Pan Zoom - Mouse Wheel Zoom - Enable*

Description: If this is selected (on by default), scrolling the middle button, if supported, will zoom the image in and out by a factor.

Command: -

Menu: *Edit - Preferences - Pan Zoom - Mouse Wheel Zoom - Factor*

Description: Sets the zoom factor for mouse wheel zoom.

Command: -

Menu: *Edit - Preferences - Bin - Mouse Wheel Bin - Enable*

Description: Enables binning with the mouse wheel

Command: -

Menu: *Edit - Preferences - Bin - Mouse Wheel Bin - Factor*

Description: Sets the mouse wheel binning factor; the default factor is 1.2

Command: -

Menu: *Edit - Pan*

Description: Sets the mode to Pan mode, in which the user can drag the location of the display center around in the image (i.e. shift the location on which the visible subset of the image is centered).

Command: **-**

Menu: *Zoom - Center Image*

Description: Resets the display center to be the image center.

Command: **align yes|no**

Menu: *Zoom - Align*

Description: Puts north at top.

Command: **wcs align yes|no**

Menu: *Zoom - Align*

Description: Same as 'align'

Command: **pan to [x] [y] [coordsys]**

Menu: -

Description: Pan to coordinates x,y. Examples:
pan to 200 200 image
pan to 400 400 physical
pan to 13:29:55 47:11:50 wcs fk5

Command: **pan [x] [y] [coordsys]**

Menu: -

Description: Pan by relative amount x,y. Examples:
pan 200 200 image
pan 400 400 physical

5.6 Crop

Command: **mode crop**

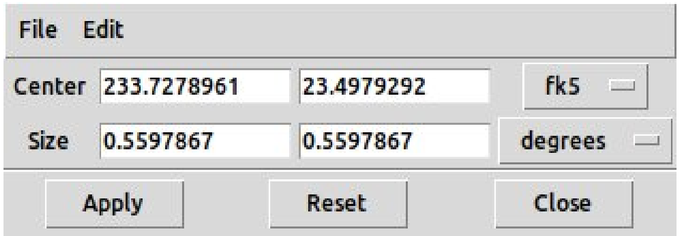
Menu: *Edit - Crop*

Description: In Crop mode, click and drag the left button to define a rectangle with one corner at the starting position and with edges aligned with the image axes. On releasing the left button, the image is cropped to be just the data within that rectangle.

Command: **crop open|close**

Menu: *Zoom - Crop Parameters*

Description: Opens the crop parameters dialog. Allows you to edit the rectangle parameters and has a 'Reset' button to undo the crop.



Center 233.7278961 23.4979292 fk5

Size 0.5597867 0.5597867 degrees

Figure 13: Crop Parameters dialog

Command: **crop [x] [y] [w] [h] [coordsys] [skyframe] degrees|arcmin|arcsec**

Menu: *Zoom - Crop parameters*

Description: Allows you to edit the rectangle parameters and crop the data. Examples:
crop 40 30 10 20 physical
crop +104:51:06.915 +68:33:40.761 8.14 22.00 wcs galactic arcsec

Command: **crop reset**

Menu: *Zoom - Crop Parameters - Reset*

Description: Reset the image, removing the crop

6 Coordinates

The Info panel in DS9 dynamically displays the coordinate values corresponding to the current mouse position in the image, physcial and WCS coordinate system. WCS ('World coordinate system') is a generic name for coordinates defined in the FITS file header as a parameterized mapping from the image physical coordinates.

DS9 supports two kinds of coordinates: (1) WCS coordinates, which are (usually) nonlinear mappings of the pixel plane to the celestial (or other) sphere, and (2) linear rotation/translation mappings which usually represent instrumental coordinates of some kind.

The WCS systems include a primary WCS, and optionally secondary WCS systems labelled WCS A to WCS Z.

The linear systems have four subtypes: Image, physical, amplifier and detector. The image system is the pixel number system of the (possibly) rebinned image being displayed by DS9. The physical system is a linear transformation between the current image coordinates and an 'original' detector coordinate system. Amplifier and Detector coordinates are piecewise linear mappings defined to support mosaic detectors and describe how the chips are arranged relative to one another.

6.1 Image, physical and WCS

The underlying 2D image has pixel numbers than run from 1 to NX, 1 to NY along each axis. Image pixel coordates of pixel 1,1 run from 0.5 to 1.5, with (1.0, 1.0) in the center of the pixel. The physical coordinates represent the rebinned image (for example when created from an event file, see later). If an image is rebinned by a factor m around a reference point with image coordinates X0, Y0 and physical coordinates x0, y0, then the corresponding physical coordinates for an arbitary location X,Y are

$$(x, y) = (x0 + (X - X0)/m, \quad y0 + (Y - Y0)/m)$$

The WCS coordinates usually involve a nonlinear transformation applied to the physical coordinates P = (x,y) using the reference physical position $P_0 = (x_0, y_0)$ (CRPIX keyword family), the corresponding world coordinate position $W_0 = (x_{w0}, y_{w0})$ (CRVAL keyword family), the scale factors $\Delta = (\Delta_X, \Delta_Y)$ (CD keyword family or the older CDELT) and the transformation function T (CTYPE keyword family).

$$W = T(P, P_0, W_0, \Delta)$$

The FITS header will usually define a single coordinate system, typically equatorial coordinates RA and Dec. However DS9 can automatically perform transformations between certain standard astronomical coordinate systems. The user can, for example, select *WCS - Galactic*. In this case the code reads image coordinates from the cursor position, applies the physical transform using header keywords to get physical coordinates, then applies the (RA,Dec) transform from the header keywords to map physical to (RA, Dec), and finally maps (RA, Dec) to galactic (l,b) using standard formulae.

Command: **wcs sky fk4|fk5|icrs|galactic|ecliptic**

Menu: *WCS - FK4/FK5/ICRS/Galactic/Ecliptic*

Description: Set celestial coordinate system displayed in info pane to Equatorial FK4, Equatorial FK5, Equatorial ICRS, Galactic (l_{II}, b_{II}), or Ecliptic.

6.2 FITS coordinate keywords

The standard FITS keywords for FITS images, in use since 1979, are:

CRPIXn	Pixel number coordinate at reference point for axis n
CRVALn	World cordinate value at reference point for axis n
CDELTn	Pixel size, degrees per pixel at reference point along axis n, now deprecated
CDn_m	Modern replacement for CDELT, matrix including skew for axis n,m

For binary tables, similar keywords are used for defining coordinates on table column n:

TCRPXn	Pixel number coordinate at reference point for column n
TCRVLn	World cordinate value at reference point for column n
TCDLTn	Pixel size, degrees per pixel at reference point along column n
TCDn_m	Alternative to TCDLT, matrix including skew for axis n,m

The CRPIX, TCRPX sets of keywords define the primary WCS (world coordinate system).

We can also define additional WCS systems on the same data (for example, J2000 and B1950 and Galactic), by appending a letter A to Z to the standard keywords. We can attach a name to the WCS with the WCSNAME keyword. For example, we can define a WCS P with

CRPIXnP	Pixel number coordinate at reference point for column n
CRVALnP	World cordinate value at reference point for column n
CDELTnP	Pixel size, degrees per pixel at reference point along column n
WCSNAMEP = "value"	Name for WCS system

DS9 treats "WCSNAMEP = 'PHYSICAL'" as special, defining the 'Physical' coordinate system.

In addition to the keywords above from the FITS standard, DS9 also recognizes keywords used by the IRAF system defining a linear 2D transformation with M (n x n rotation matrix) and V (n x 1 translation vector) values:

Keyword	DS9 coordinate system
LTMx_x/LTVx	Physical
DTMx_x/DTVx	Detector
ATMx_x/ATVx	Amplifier

If a WCSNAMEP= PHYSICAL system is present, it supersedes any LTM/LTV keywords.

Command: **wcs system wcs|wcsa...wcsz**

Menu: *WCS - WCS/Multiple WCS*

Description: Select which WCS system to display in the info panel

Command: **wcs reset**

Menu: -

Description: Resets WCS to default system?

6.3 Coordinate format control

Command: **wcs skyformat degrees|sexagesimal**

Menu: *WCS - Degrees/Sexagesimal*

Description: allows user to control which format the coordinates are displayed in.

Command: **prefs precision n1 n2 n3 n4 n5 n6 n7 n8 n9**

Menu: *Edit - Preferences - Precision*

Description: *Edit - Preferences - Precision - Coordinates* allows users to control the precision used for coordinates (in the info panel and in other display contexts). The precision value used is the number of digits to be displayed after the decimal point. Different precision settings are provided for coordinates of linear type (**n1**, default 8), those in decimal degrees (**n2**, default 7), in hours-min-sec format (**n3**, default 4), and in deg-min-sec format (**n4**, default 3). *Edit - Preferences - Precision - Length* allows users to control the precision used for display of length measurements. The precision value used is the number of digits to be displayed after the decimal point. Different precision settings are provided for coordinate lengths of linear type (**n5**, default 8), those in decimal degrees (**n6**, default 7), in arcmin (**n7**, default 5) and in arcsec (**n8**, default 3). *Edit - Preferences - Precision - Angles* allows users to control the precision used for display of angle measurements. The precision value used (**n9**, default 8) is the number of digits to be displayed after the decimal point.

XPA example: xpaset -p ds9 prefs precision 8 7 4 3 8 7 5 3 8

6.4 Support for multiple WCS systems

Most FITS data will have a single WCS but the standard supports the file having multiple 'alternate WCS' definitions, each labelled with a suffix letter. For example, the FITS keyword CDELT2G defines the axis 2 pixel size for WCS g.

Command: **wcs - wcsg**

Menu: *WCS - Multiple WCS - WCS g*

Description: Cause the info box to display the WCS g coordinates instead of the default WCS coordinates. Similar for wcsa through wcsz.

Command: **wcs open**

Menu: *WCS - WCS Parameters*

Description: option brings up a dialog showing the current values of WCS-relevant FITS header keywords and allows you to edit them (with *WCS - WCS Parameters - Apply*). Additional tabs in the dialog provide access to some of the more obscure WCS keywords (B_4_5 etc) which you are unlikely to need.
If multiple WCS are present, or if you want to define one on the fly, *WCS - WCS Parameters - WCS - WCS c* would select 'WCS c' as the WCS whose keywords are shown and editable.

Command: **wcs close**

Menu: *WCS - WCS Parameters - Close*

Description: Closes the WCS Parameters dialog

Command: **wcs save [filename]**

Menu: *WCS - WCS Parameters - Save*

Description: Save current WCS parameters to a file - dump of the relevant FITS header segement

Command: **wcs load [filename]**

Menu: *WCS - WCS Parameters - Open*

Description: Open a saved WCS header

Example of a saved WCS header file:

```
CRPIX1  =                 257.75
```

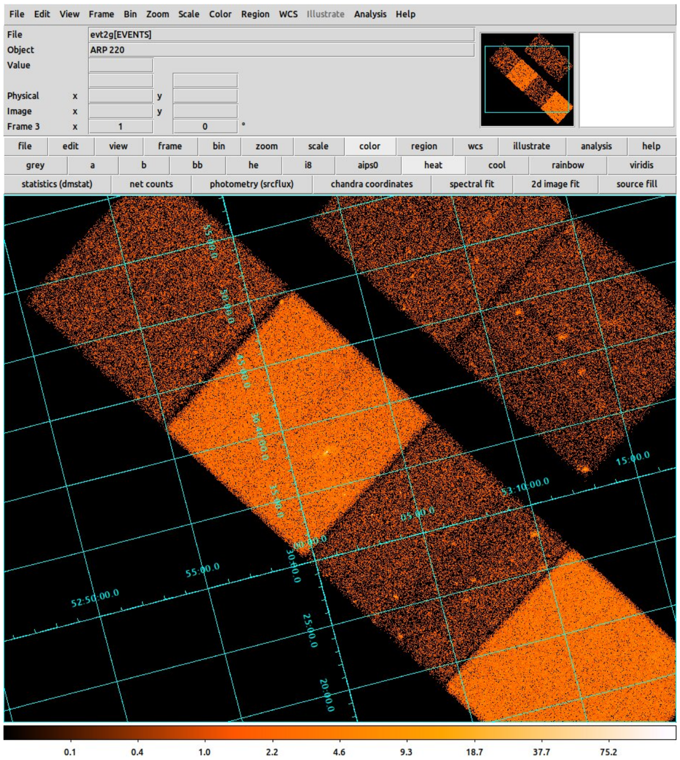
```
CRPIX2  =               258.93
CRVAL1  =      -201.94541667302
CRVAL2  =             -47.45444
CDELT1  =         -2.1277777E-4
CDELT2  =          2.1277777E-4
CTYPE1  = 'RA---TAN'
CTYPE2  = 'DEC--TAN'
```

6.5 Coordinate Grid

Command: **grid yes|no**

Menu: *Analysis - Coordinate Grid*

Description: displays a coordinate grid and axes over the image



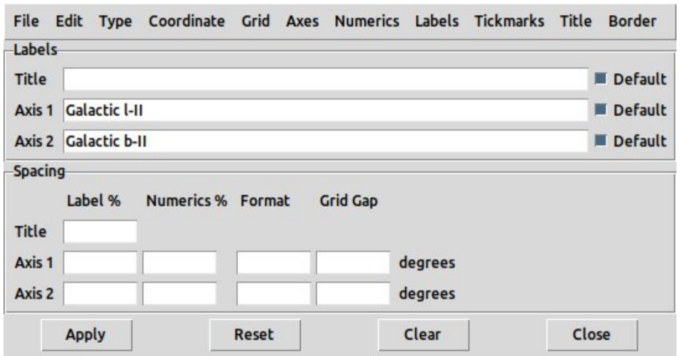
File evt2g[EVENTS]
Object ARP 220
Frame 3 x 1 0 °

Figure 14: Coordinate grid example

Command: **grid open**

Menu: *Analysis - Coordinate Grid Parameters*

Description: Opens a dialog which controls the parameters of the coordinate grid.



Labels

Title ■ Default

Axis 1 Galactic l-II ■ Default

Axis 2 Galactic b-II ■ Default

Figure 15: Coordinate grid parameters dialog

Command: **grid close**

Menu: *Analysis - Coordinate Grid Parameters - Close*

Description: Close the coord grid parameters dialog

Command: **grid grid yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Grid - Show*

Description: Deselecting this suppresses the grid but not the axes.

Command: **-**

Menu: *Analysis - Coordinate Grid Parameters - Apply*

Description: Apply changes to the coordinate grid parameters to regenerate the coordinate grid.

Command: **grid reset**

Menu: *Analysis - Coordinate Grid Parameters - Clear*

Description: Reset the coordinate grid parameters

Command: **grid grid color [value]**

Menu: *Analysis - Coordinate Grid Parameters - Grid - Color*

Description: selects the color used for the grid lines and numerics.

Command: **grid grid width [value]**

Menu: *Analysis - Coordinate Grid Parameters - Grid - Line*

Description: selects the lineweight used for the grid lines.

Command: **grid grid dash yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Grid - Line - Dash*

Description: selects dashed instead of continuous lines.

Command: **grid axes yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Axes - Show*

Description: controls whether the axis lines themselves are drawn.

Command: **grid axes color [value]**

Menu: *Analysis - Coordinate Grid Parameters - Axes - Color*

Description: sets the color of the axis lines.

Command: **grid axes width [value]**

Menu: *Analysis - Coordinate Grid Parameters - Axes - Line*

Description: sets the lineweight of the axis lines.

Command: **grid axes dash yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Axes - Line - Dash*

Description: sets the axes to be dashed lines or not

Command: **grid axes origin [option]**

Menu: *Analysis - Coordinate Grid Parameters - Axes - Origin*

Description: This is greyed out by default except for 3D data. Specifies the origin as lower or upper on the three axes. Options are lll, llu, lul, luu, ull,m ulu, uul, uuu

Command: **grid title yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Title - Show*

Description: to suppress showing the grid title.

Command: **grid title text [value]**

Menu: *Analysis - Coordinate Grid Parameters - Labels - Title*

Description: controls the title; unselect Default and enter a value to override

Command: **grid title def yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Labels - Title*

Description: Controls whether the default title is used

Command: **grid title font [value]**

Menu: *Analysis - Coordinate Grid Parameters - Labels - Title - Font*

Description: Controls grid title font

Command: **grid title fontsize|fontweight|fontslant [value]**

Menu: *Analysis - Coordinate Grid Parameters - Labels - Title - Font*

Description: Controls grid title font size, weight and slant

Command: **grid title color**

Menu: *Analysis - Coordinate Grid Parameters - Title - Color*

Description: control grid title color

Command: **grid labels yes|no**

Menu: -

Description: Show or supress grid axis labels

Command: **grid labels def1|def2 yes|no**

Menu: -

Description: Use defaults for axis1 and axis2 labels

Command: **grid labels text1|text2 [value]**

Menu: *Analysis - Coordinate Grid Parameters - Labels - Axis 1/Axis 2*

Description: overrides the axis label names, if Default is unselected.

Command: **grid type analysis|publication**

Menu: *Analysis - Coordinate Grid Parameters - Type - Analysis/Publication*

Description: Chooses between two styles. For Analysis, the grid extends beyond the active data area and for Publication it is truncated there.

Command: **grid axes type interior|exterior**

Menu: *Analysis - Coordinate Grid Parameters - Type - Interior Axes/Exterior Axes*

Description: controls whether the axes are placed in the interior or at the edge of the image.

Command: **grid numerics yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Numerics - Show*

Description: shows or suppresses the axis numeric labels.

Command: **grid numerics type interior|exterior**

Menu: *Analysis - Coordinate Grid Parameters - Type - Interior Numerics/Exterior Numerics*

Description: In publication mode only (see 'grid type'), controls whether the axis numeric labels are placed in the interior or at the edge of the image.

Command: **grid numerics vertical|horizontal**

Menu: *Analysis - Coordinate Grid Parameters - Type - Vertical Text*

Description: If this is selected, the y axis numbers are printed in a vertical orientation instead of parallel to the Y axis.

Command: **grid numerics font times|helvetica|courier**

Menu: *Analysis - Coordinate Grid Parameters - Numerics - Font*

Description: Controls font for axis numerics

Command: **grid numerics fontslant/fontsize/fontweight [value]**

Menu: *Analysis - Coordinate Grid Parameters - Numerics - Font*

Description: Controls font properties for axis numerics

Command: **grid numerics color [value]**

Menu: *Analysis - Coordinate Grid Parameters - Numerics - Color*

Description: controls the color of the numeric labels.

Command: **grid border yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Border - Show*

Description: draws a border around the coordinate grid.

Command: **grid border color [value]**

Menu: *Analysis - Coordinate Grid Parameters - Border - Color*

Description: controls the color of the border.

Command: **grid border width [value]**

Menu: *Analysis - Coordinate Grid Parameters - Border - Line*

Description: controls the width of the border.

Command: **grid border dash yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Border - Line - Dash*

Description: controls dashed or solid lines for the border.

Command: **grid tickmarks yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Tickmarks - Show*

Description: shows or suppresses the axis tickmarks.

Command: **grid tickmarks color [value]**

Menu: *Analysis - Coordinate Grid Parameters - Tickmarks - Color*

Description: controls the color of the tickmarks.

Command: **grid tickmarks width [value]**

Menu: *Analysis - Coordinate Grid Parameters - Tickmarks - Line*

Description: controls the width of the tickmarks.

Command: **grid tickmarks dash yes|no**

Menu: *Analysis - Coordinate Grid Parameters - Tickmarks - Line - Dash*

Description: controls dashed or solid lines for the tickmarks.

Command: **grid system [coordsys]**

Menu: *Analysis - Coordinate Grid Parameters - Coordinate - WCS*

Description: The coordinates plotted are by default the primary WCS coordinates of the file (usually RA, Dec ICRS). However this can be altered by changing the selection with coordsys = WCS, Image, Physical, Amplifier, Detector.

Command: **grid sky [coordsys]**

Menu: *Analysis - Coordinate Grid Parameters - Coordinate - Galactic/FK4/FK5/ICRS/Ecliptic*

Description: Select coordinate system for grid

Command: **grid skyformat degrees|sexagesimal**

Menu: *Analysis - Coordinate Grid Parameters - Coordinate - Degrees/Sexagesimal*

Description: Set the format style of the coordinates printed

Command: **grid grid gap1|gap2|gap3**

Menu: *Analysis - Coordinate Grid Parameters - Spacing - Grid Gap - Axis 1/Axis 2*

Description: Set the spacing of the grid lines on a given axis.

Command: **grid numerics gap1|gap2|gap3**

Menu: *Analysis - Coordinate Grid Parameters - Spacing - Numerics - Axis 1/Axis 2*

Description: Set the spacing of the numerics relative to the axis lines on a given axis.

Command: **grid format1|format2**

Menu: *Analysis - Coordinate Grid Parameters - Spacing - Format - Axis 1/Axis 2*

Description: Set format of numerics on grid. The format can be a standard C printf format with an extra leading percent, e.g. ”%%1.7G”, or it can be a special DS9 format as defined below.

The format string is a sequence of case-insensitive characters from the list below. If options conflict, the rightmost character takes precedence except that d and h override t. The default formats are ’d.3’, ’d.3’ for degrees and ’hms.1’, ’dms.1’ for sexagesimal.

Format character	Meaning
+	Indicates that a plus sign should be prefixed to positive values. By default, no plus sign is used.
z	Indicates that leading zeros should be prefixed to the value so that the first field is of constant width, as would be required in a fixed-width table (leading zeros are always prefixed to any fields that follow). By default, no leading zeros are added.
i	Use the standard ISO field separator (a colon) between fields. This is the default behaviour.
b	Use a blank to separate fields.
l	Use a letter (”h”/”d”, ”m” or ”s” as appropriate) to separate fields.
g	Use a letter and symbols to separate fields (”h”/”d”, ”m” or ”s”, etc, as appropriate), but include escape sequences in the formatted value so that the Plot class will draw the separators as small super-scripts.
d	Include a degrees field. Expressing the angle purely in degrees is also the default if none of ”h”, ”m”, ”s” or ”t” are given.
h	Express the angle as a time and include an hours field (where 24 hours correspond to 360 degrees). Expressing the angle purely in hours is also the default if ”t” is given without either ”m” or ”s”.
m	Include a minutes field. By default this is not included.
s	Include a seconds field. By default this is not included. This request is ignored if ”d” or ”h” is given, unless a minutes field is also included.
t	Express the angle as a time (where 24 hours correspond to 360 degrees). This option is ignored if either ”d” or ”h” is given and is intended for use where the value is to be expressed purely in minutes and/or seconds of time (with no hours field). If ”t” is given without ”d”, ”h”, ”m” or ”s” being present, then it is equivalent to ”h”.
.	Indicates that decimal places are to be given for the final field in the formatted string (whichever field this is). The ”.” should be followed immediately by an unsigned integer which gives the number of decimal places required, or by an asterisk. If an asterisk is supplied, a default number of decimal places is used which is based on the value of the Digits attribute.

Command: **grid labels gap1|gap2**

Menu: *Analysis - Coordinate Grid Parameters - Spacing - Label|Title % - Axis 1/Axis 2*

Description: Control of the spacing of the axis labels and title

7 Regions

Regions are geometric shapes that can be defined interactively in DS9, or read in from a DS9 ascii region file or from a CIAO FITS region file.

The most common DS9 region is a circle (other region types are described below). It can be created interactively when the DS9 edit mode is *Edit - Region*, or can be specified in a region file which is loaded from the region menu.

Many regions can be defined in a single frame. A subset of them can be 'selected' (by clicking on them) and certain actions apply to selected regions.

Command: **-**

Menu: -

Description: KEYBOARD SHORTCUT: Shift-Drag: define a rectangle, select all regions within that rectangle.

7.1 DS9 Region files

DS9 region files are text files with a particular syntax. The simplest region file is just a single line containing a shape description, for example

```
circle(4096.5,4096.5,8.2)
```

When you save the region to a file from DS9 you get some additional metadata

```
# Region file format: DS9 version 4.1
global color=green dashlist=8 3 width=1 font="helvetica 10 normal roman"
 select=1 highlite=1 dash=0 fixed=0 edit=1 move=1 delete=1 include=1 source=1
physical
circle(4096.5,4096.5,8.2)
```

A region file can contain multiple regions

```
# Region file format: DS9 version 4.1
global color=green dashlist=8 3 width=1 font="helvetica 10 normal roman"
 select=1 highlite=1 dash=0 fixed=0 edit=1 move=1 delete=1 include=1 source=1
```

```
physical
circle(4096.5,4096.5,8.2)
box(1268,1151,100,200)
```

or

```
# Region file format: DS9 version 4.1
global color=green dashlist=8 3 width=1 font="helvetica 10 normal roman"
 select=1 highlite=1 dash=0 fixed=0 edit=1 move=1 delete=1 include=1 source=1
physical;circle(4096.5,4096.5,8.2);box(1268,1151,100,200)
```

or

```
# Region file format: DS9 version 4.1
global color=green dashlist=8 3 width=1 font="helvetica 10 normal roman"
 select=1 highlite=1 dash=0 fixed=0 edit=1 move=1 delete=1 include=1 source=1
physical
circle 4096.5 4096.5 8.2
box 1268 1151 100 200
```

The parentheses forms of the shape specifications are what DS9 will use when saving a region to a file; the forms without parentheses are supported for back compatibility. Semicolons are equivalent to a newline.

The first line of the file, beginning with a hash symbol, describes the version of the file format. This line is optional.

The second line, beginning with 'global', defines region properties that will apply to all regions in the file.

The third line defines the coordinate system in which the coordinate values in the region are expressed.

Fourth and subsequent lines specify the individual region shapes. Shape definitions are of the form

```
+shape(param,param,...param) # prop=value prop=value prop=value
```

so that each shape has a set of parameters, and a set of properties whose values may be set. Parameters are required; properties may be omitted to choose their default values. If no properties are specified the hash mark may be omitted.

The leading + has no effect and is usually omitted. A leading minus sign indicates the region is meant to be excluded rather than included. For example

```
# Region file format: DS9 version 4.1
global color=green dashlist=8 3 width=1 font="helvetica 10 normal roman"
 select=1 highlite=1 dash=0 fixed=0 edit=1 move=1 delete=1 include=1 source=1
physical
circle(4096.5,4096.5,8.2)
-box(4100,4100,10,10)
```

denotes a circle with the part that overlaps with the box excluded.

Some ‘display-only’ shapes instead use the syntax

```
# shape(param,param,...param) prop=value prop=value prop=value
```

with the hash mark before instead of after the shape section.

Lines with a leading hash mark # which are not recognized as valid display-only shapes are ignored and treated as comments.

7.2 Other supported region formats

Command:	-
Menu:	*Edit - Preferences - Region - Default Format*
Description:	allows the default output region file format to be ’ds9’ (default, DS9/Funtools); ’XML’, ’CIAO’, or obsolete legacy formats ’SAOtng’, ’SAOimage’, ’IRAF Pros’, or an unannotated ’X Y’ format.

DS9 can also read CIAO FITS region binary tables.

- CIAO region files

  DS9 also supports Chandra CIAO region files on both input and output. These files are like the DS9 files but don’t have the header lines. So,

  ```
  circle(4096,4096,25.2)
  ```

  is a valid CIAO region file. Instead of BOX, CIAO has ROTBOX and RECTANGLE shapes which DS9 translates into BOX shapes. CIAO does not support display-only shapes, PANDA, or ELLIPSE/BOX ANNULUS shapes.

- SAOimage and IRAF-PROS region format:

  - SAOimage POINT is mapped to DS9 box, circle, point
  - SAOimage ROTBOX is mapped to DS9 box
  - Line, Vector, Projection, Segment, Text, Ruler, Compass, Ellipse Annulus, Box Annulus, Panda, EPanda, BPanda are not supported.

  Example of a PROS region file:

  ```
  /m51.fits
  # Mon Jan 16 16:09:43 1989
  # shape x, y, [x dimension, y dimension], [angle]
  ```

```
 BOX(135.67,213.00,18.00,10.67)
 BOX(504.00,538.00,16.96,106.53,290.714)
 CIRCLE(129.00,306.67,9.89)
-ELLIPSE(486.00,468.00,36.53,44.08,329.036)
-POINT(83.00,349.00)
 POLYGON(131.00,245.00,120.67,280.33,165.00,245.00,140.00,234.67)
 BOX(149.67,205.00,30.00,12.00) & !BOX(149.67,205.00,20.00,8.00)
```

- Funtools

  The Funtools regions are similar to DS9 but do not support Line, Vector, Projection, Segment, Text, Ruler or Compass.

- XY format

  This format consists of a number of coordinate pairs, one per line. The coordinate format for both input and output is specified via the Save Regions Parameters menu or XPA regions point. The first two coordinates are read, the rest of the line is ignored. The comment character '#' may be used at the beginning of line and the line is ignored. This format is very useful for reading in coordinates from other external analysis programs, such as IRAF.

  ```
  Example: # this is a comment
  physical # this overrides the specified coordinate system
  300 300
  400 400 # this is a comment
  ```

7.3 Region mode

Command:	**mode none**
Menu:	*Edit - None*
Description:	Left mouse button has its default behaviour.

Command:	**mode region**
Menu:	*Edit - Region*
Description:	In this mode, which was the default in early versions of DS9, the left button allows the user to create and edit regions. Regions can be either 'selected' or 'unselected'; certain actions apply to currently selected regions only.

Details of button actions in region mode:

- If no region is currently selected: click to start creating a region with reference point at that location; hold down and move, releasing to define the size of the region.

  The type of region is predefined in the Region/Shape menu; the default is a circle.

- Clicking inside an existing region selects the region; clicking and dragging allows you to move it.

- If a region is selected, clicking in an empty area of the image deselects the region (i.e. returns you to the 'no region currently selected' state.)

- Double-clicking on a selected existing region brings up a dialog box allowing you to change the region parameters, including color, line width, etc.

- If one region is selected, left-clicking in a different region selects the new region and deselects the first one. Shift-left-clicking in a different region selects the new region and keeps the old one(s) selected too.

- Shift-left-clicking outside a set of regions, then dragging, specifies a rectangle; at release, all regions within that rectangle are selected.

- Middle and right buttons behave the same as for 'none' mode.

- Note that the list of regions is ordered, and regions are superimposed on each other with the first region at the top and the last region at the bottom; this can easily be seen by making several overlapping regions with different colors and with fill=1.

- Each region has a bounding box and a center. The center of a region is the centroid of the corners of the bounding box. (this also applies to composite regions - the center of a composite region is the centroid of the composite's bounding box). Rotating a region is done around its center.

The Region menu also has entries controlling region selection:

Command:	**region select all**
Menu:	*Region - All*
Description:	selects all the regions currently defined

Command:	**region select none**
Menu:	*Region - None*
Description:	deselects all regions

Command: **region select invert**

Menu: *Region - Invert*

Description: deselects all currently selected regions and selects all currently unselected regions

Command: **region select front|back**

Menu: *Region - Front, Back*

Description: Selects the first or the last region in the list.

Command: **region move front|back**

Menu: *Region - Move To Front/Back*

Description: moves the selected region to the beginning or end of the list.

7.4 Region coordinate systems

The allowed coordinate system names in a region file are

Name	System	Default unit	Meaning
image	image	pix	Pixel coordinates of the current image
physical	physical	scaled pix	Original pixel coords of instrument, as defined by CRPIXnP/CDELTnP or the IRAF keywords LTM/LTV
icrs	ICRS	deg	ICRS celestial coords, using WCS keywords in image
fk4	FK4,B1950	deg	FK4/B1950 coords
fk5	FK5/J2000	deg	FK5/J2000 coords
galactic	IAU galactic	deg	l(II), b(II) in degrees
ecliptic	ICRS ecliptic	deg	Ecliptic lon and lat, degrees
wcs	WCS		Whatever coord system is specified by the file's primary WCS
wcsa	WCSA		Whatever coord system is specified by the file's secondary 'A' WCS
linear	LINEAR		Linear coord system specified by the file's primary WCS

If no coordinate system is specified, 'physical' or 'WCS' is assumed depending on the formatting (decimal or sexagesimal) of the coordinate values.

7.5 Region coordinate value, length and angle formats

7.5.1 Coordinate value formats

Values in these coordinate systems can be expressed in a variety of ways.

Syntax	Example	Interpretation
[num]	5.8	Coordinate value in pixels for IMAGE,PHYSICAL, else in degrees
[num]d	243.3d	Degrees
[num]r	1.745r	Radians
[num]p	1024.3p	Pixels in PHYSICAL system
[num]i	512.0i	Image pixel number
[num]:[num]:[num]	14:07:28.8	hr:min:s for long-like coords, deg:min:s for lat-like coords
[num]h[num]m[num]s 10h20m15.1s	hr:min:s	
[num]d[num]m[num]s -00d20m15.1s	deg:min:s	

7.5.2 Coordinate length formats

Arguments like circle radius or box length can use the following formats:

Syntax	Example	Interpretation
[num]	5.8	Coordinate value in pixels for IMAGE,PHYSICAL, else in degrees
[num]d	243.3d	Degrees
[num]r	1.745r	Radians
[num]p	1024.3p	Pixels in PHYSICAL system
[num]i	512.0i	Image pixel number
[num]”	20.4”	Arcseconds
[num]’	20.4’	Arcmin

Command: -

Menu: *Edit - Preferences - Region - Default Length*

Description: The default length unit is set; it may be Degrees (default), ArcMin or ArcSec.

7.5.3 Angle formats

Angles (e.g. for the orientation of an ellipse) are always expressed as a simple number, interpreted as degrees.

7.6 Region shapes

Here I define the region shape types, their parameters, and the properties that are specific to those shapes; I also discuss how to create them interactively. As noted earlier, region creation is done either by reading a region file, or by selecting *Edit - Region* and then left clicking at the location you want the region to be at.

Command: -

Menu: *Region - Color - Fill*

Description: If this flag is set, the newly created region is displayed as filled, otherwise as an outline.

Command: -

Menu: *Region - Region Parameters - Auto Plot Statistics*

GUI Default: *Edit - Preferences - Region - Auto Plot - Statistics*

Description: If this flag is set, upon region creation a popup window appears showing various statistics of the image pixels within the region.

Command: -

Menu: *Region - Region Parameters - Auto Plot 2D*

Description: If this flag is set, upon region creation a popup window appears showing a line plot of the pixel values vs coordinate.

Command: -

Menu: *Region - Region Parameters*

Description: This menu lets you control some display properties that newly created regions will have. It does not affect previously created regions.

Command: **region epsilon [value]**

Menu: *Edit - Preferences - Region - Mouse Click Epsilon*

Description: Interactive region creation look and feel can be modified by changing epsilon from the default value of 3 pixels to a value between 2 and 10 pixels. This ‘epsilon’ determines how close to the region the pointer position must be to successfully select it by clicking.
Example: xpaset -p ds9 region epsilon 5

7.6.1 Basic shapes

Command: **region shape [value]**

Menu: *Region - Shape*

Description: Select the shape to be created when you click in the image

Command: **region open**

Menu: *Region - Get Information*

Description: Bring up the region shape dialog box. This can also be done by double clicking on a region.

Command: **region close**

Menu: *Region - [shape] - Close*

Description: Close the region shape dialog box.

Command: **region selected**

Menu: -

Description: Used with xpaget; ‘xpaget ds9 region selected’ writes the region details to standard output.

There are four region types that define areas on the image.

- For ‘circle’,

- DS9 region file syntax and example:

  ```
  circle(x,y,radius) # fill=[0|1]
  circle(4096.5,4096.5,8.2) # fill=0
  ```

- Parameters: fill. Is the circle filled in; 0 = no (default), 1 = yes.
- Mouse creation: the reference point is its center, and the release point defines the radius.

Command:	-
Menu:	*Edit - Preferences - Region|Illustrate - Circle - Radius*
Description:	Sets the default circle radius (initially 20 pixels)

- For 'box':
  - DS9 region file syntax and example:

    ```
    box(x,y,width,height,angle)# fill=[0|1]
    box(4080.0,4090.0,20.0,10.0,0.0) # fill=0
    ```

    The angle value is optional and defaults to zero.
  - Mouse creation: the reference point is one corner. Dragging increases the sides of the box, which are parallel to the image axes. The angle can be changed from the region parameter dialog which can be popped up by selecting the region (click inside it) and then double-clicking again inside the selected region (or by using the Region - Get Information menu item).

Command:	-
Menu:	*Edit - Preferences - Region|Illustrate - Box - Size 1/Size 2*
Description:	The default side lengths of the box are 80 and 40 pixels, controlled by this command.

- 'ellipse'
  - DS9 region file syntax and example:

    ```
    ellipse(x,y,a,b,angle) # fill=[0|1]
    ellipse(4096.5,4096.5,48,24,45.0)
    ```

  - Mouse creation: like circle; the ellipticity is controlled by dragging up/down vs left/right. The ellipse axes are always parallel to the image axes.

Command:	-
Menu:	*Edit - Preferences - Region|Illustrate - Ellipse - Radius 1/Radius 2*
Description:	Controls default size of ellipse.

- For ‘polygon’
  - DS9 region file syntax:

    ```
    polygon(x1,y1,x2,y2,x3,y3 ...) # fill=[0|1]
    ```

  - Mouse creation: clicking and dragging creates a square centered on the click location with size determined by the drag release point. After release, clicking again inside the square selects it. Once selected, clicking again on the boundary of the square creates an additional point (node) that may be dragged; subsequent clicks will add more points to the polygon. Note that clicking on the boundary *before* selecting the region will just create an additional polgyon region, probably not what you wanted.

Command:	-
Menu:	*Edit - Preferences - Region|Illustrate - Polygon - Width/Height*
Description:	Controls default size of initial square, initially 20 x 20

7.6.2 Vector shapes

Four additional shapes allow a set (vector) of regions to be defined.

- ‘annulus’:
  - DS9 region file syntax and example:

    ```
    annulus(x,y,r1,r2)
    annulus(4388,4226,15,30)
    annulus(x,y,r1,r2,r3,...)
    annulus(4388,4226,15,30,45,62,90)
    ```

    Additional radii may be defined for a set of concentric annuli, as in the second example. This capability is also present for the elliptical annulus and box annulus shapes described below.

  - Mouse creation: click draws an annulus, with a fixed ratio of inner and outer circle, the outer radius determined by dragging before releasing. Once released and then click-selected, clicking on the individual circles allows you to change their radii individually. Selecting the annulus region by clicking and then performing ’control - drag’, you can create additional annuli. Also, in the double-click dialog box, you can change the number of annuli by typing in the number to the 'Annuli' entry, then click Generate, then Apply.

Command: -

Menu: *Edit - Preferences - Annulus - Annulus - Radius - Inner/Outer*

Description: Controls default radii of annulus, initally 15 and 30 pixels

Command: -

Menu: *Edit - Preferences - Annulus - Annulus - Annuli*

Description: sets the default number of annuli at creation.

Command: **region analysis radial open**

Menu: *Annulus - Analysis - Radial Profile*

Description: Opens a plot window showing surface brightness vs radius. Useful for multiple annuli.

- 'elliptical annulus':
  - DS9 region file syntax and example:

    ```
    ellipse(x,y,a,b,r1,r2,angle)
    ellipse(4124,4132,18.5,63.5,37,127,0)
    ```

  - Mouse creation: Similar to annulus. The double-click dialog box allows the annulus orientation to be selected, and the ellipticity (change the major or minor axis, then click Generate, then Apply).

Command: -

Menu: *Edit - Preferences - Annulus - Elliptical Annulus - Inner - Major/Minor*

Description: Sets defaul elliptical inner annulus shape (defaults 40, 20)

Command:	-
Menu:	*Edit - Preferences - Annulus - Elliptical Annulus - Outer - Major*
Description:	Sets default outer annulus major axis (default 60)

Command:	-
Menu:	*Edit - Preferences - Annulus - Elliptical Annulus - Annuli*
Description:	Sets default number of annuli, default 1

- ‘box annulus’:
  - DS9 region file syntax and example:

    ```
    box(x,y,width1,height1,width2,height2,angle)
    box(4348.501,4214.499,60.499,46.501,120.998,93.002,0)
    ```

  - Mouse creation: Similar to elliptical annulus, but the regions drawn are boxes.

Command:	-
Menu:	*Edit - Preferences - Annulus - Box Annulus - Inner - Width/Height*
Description:	Set Box annulus inner size (defaults 80, 40)

Command:	-
Menu:	*Edit - Preferences - Annulus - Box Annulus - Outer - Width*
Description:	Set Box annulus outer size (default 120)

Command:	-
Menu:	*Edit - Preferences - Annulus - Box Annulus - Annuli*
Description:	Set Box annulus default number of annuli (default 1))

- ‘panda’

  The panda region defines a set of annuli with sectors.

  - DS9 region file syntax and example:

```
panda(x,y,theta1,theta2,ntheta,r1,r2,nr)
panda(4213,4295,0,360,4,67,135,1)
panda(4212.9,4295.8,0,270,4,67.8,135.6,2)
```

– Mouse creation: draws an annulus in which the region between the inner and outer circles is divided in NE, NW, SE, SW sectors. The number of sectors and annuli, and the angular range (default 0 to 360 deg) can be changed in the double-click dialog box: change the values then click Generate, then Apply.

The default parameters are controlled by

Command: -

Menu: *Edit - Preferences - Panda - Panda - Radius - Inner/Outer*

Description: (defaults 15, 30)

Command: -

Menu: *Edit - Preferences - Panda - Panda - Angles - Start/Stop*

Description: (default 0, 360 deg)

Command: -

Menu: *Edit - Preferences - Panda - Panda - Angles - Number*

Description: (default 4)

Command: -

Menu: *Edit - Preferences - Panda - Panda - Annuli*

Description: (default 1)

Command: **region analysis panda open**

Menu: *Panda - Analysis - Radial Profile*

Description: Opens a plot window showing surface brightness vs radius, one line per sector. Useful for multiple annuli.

- ‘epanda’ and ’bpanda’ are the elliptical and box versions of the panda region, with separately specified inner and outer minor and major axis radii and an optional rotation angle phi.
  - DS9 region file syntax and example:

    ```
    epanda(x,y,theta1,theta2,ntheta,a1,b1,a2,b2,nr,phi)
    bpanda(x,y,theta1,theta2,ntheta,a1,b1,a2,b2,nr,phi)
    ```

    The ratio a2/b2 must be the same as a1/b1.
  - Mouse creation: as for panda, but while creating, dragging vertically and horizontally alters the ellipticity.
    The default parameters are controlled by

Command:	-
Menu:	*Edit - Preferences - Panda - Elliptical Panda/Box Panda - Radius - Inner/Outer*
Description:	(defaults 15, 30)

Command:	-
Menu:	*Edit - Preferences - Panda - Elliptical Panda/Box Panda - Angles - Start/Stop*
Description:	(default 0, 360 deg)

Command:	-
Menu:	*Edit - Preferences - Panda - Elliptical Panda/Box Panda - Angles - Number*
Description:	(default 4)

Command:	-
Menu:	*Edit - Preferences - Panda - Elliptical Panda/Box Panda - Annuli*
Description:	(default 1)

7.6.3 Display-only shapes

Other region types are purely for illustrations and analysis. They define a line or curve on the image which can be used to generate a 2D plot, or for 3D data a 3D plot. The *Region Parameters - Auto Plot 2D/Auto Plot 3D* menu options, or alternatively *Edit - Preferences - Region - Auto Plot - 2d/3d*, determine whether or not the plot is shown.

- 'point' draws a single point symbol and has a number of subtypes (circle, box, diamond, etc).

  Parameters describe the symbol type, the size of the symbol, and its color. The symbol type is required.

  - DS9 region file syntax and example:

    ```
    point(x,y) # point=subtype
    point(3883,4387) # point=diamond
    point(x,y) # point=subtype size color=value
    point(3883,4387) # point=diamond 40 color=cyan
    ```

  - Mouse creation: Clicking creates the point; click-dragging does nothing. Additional clicks create new points.

Command:	-
Menu:	*Edit - Preferences - Region - Point - Size*
Description:	The default size of the point at creation is set with this, initially 11 pixels

- 'line' draws a line.

  Parameters L and R can be 0 or 1; if 1, an arrowhead is drawn on the left (L) or right (R) end of the line.

  The 'line' region can be used to make a scale bar on a plot for publication. Double-clicking to get the region-dialog allows you to read off the current length in angular units. If you click-drag one end of the line you can adjust its length, and the displayed value will update, allowing you to choose a suitable lengthy (say 1 arcmin). The region dialog also has a Text box where you can enter a label to be displayed next to the line.

  - DS9 region file syntax and example:

    ```
    line(x1,y1,x2,y2)  # line=L R
    line(3881,4297,3913,4189) # line=0 0
    line(3881,4297,3913,4189) # line=0 1
    line(15:35:01.8309,+23:29:10.479,15:34:59.6494,+23:29:10.976) # line=0 0 text={30"}
    ```

  - Mouse creation: clicking defines one end of the line; dragging and releasing defines the other end. Once the line is defined, selecting it by clicking on it and then clicking again on one end allows you to drag that end and extend or move the line. Clicking on the middle of a selected line allows you to move it around.

- 'vector'

  Vector draws a line with an arrow. However the optional parameter vector=0 can switch off the arrow, just as for 'line' the parameter line=0 1 can swich on an arrow. The real difference between 'line' and 'vector' is that the first specifies two endpoints, while the second specifies one endpoint and a length/angle.

  - DS9 region file syntax and example:

    ```
    # vector(x,y,length,angle) vector=1
    # vector(4099,4012,125.64235,33.310631) vector=1
    ```

    Note the leading # in the syntax here.
  - Mouse creation: vector is like 'line', but an arrowhead is drawn at the release end of the line.

- 'projection'

  - DS9 region file syntax and example:

    ```
    # projection(x1,y2,x2,y2,0)
    # projection(3811,3981,3932,4027,0)
    # projection(x1,y2,x2,y2,width)
    # projection(3904.9992,4213.0005,4289.0002,4255.0008,146.34146)
    ```

    Note the leading # in the syntax here. In the second case with finite width, the xy values define the lower boundary of a box and the width is in the upward (physical +y) direction to define the width of the box.
  - Mouse creation: draws a line, and pops up a line graph showing the intensity profile of image pixels values along that line. By double-click selection you can use the dialog box to set width, as in the second example above.

Command:	-
Menu:	*Edit - Preferences - Region - Projection - Thickness*
Description:	Controls default thickness of the width, initially 0

- 'segment': Segment is an open polygon, a series of line segements.

  - DS9 region file syntax and example:

    ```
    # segment(x1,y1,x2,y2,x3,y3,...)
    # segment(3682.9316,4477.0684,3803,4185,3917,4265,3895,4321,4003,4327)
    ```

  - Mouse creation: 'segment' is like an open polygon. Initially it makes a diagonal line. Click to select and then click again in the line to add a break point. Now you have a broken line with a node in the middle. Clicking on the node allows you to move that node around; clicking elsewhere in the line adds a new node.

- 'text' - allow user to annotate the image with text

- DS9 region file syntax and example:

```
# text(x,y) text={blah blah}
# text(3991,4381) text={Region}
# text(x,y) textangle=theta color=name width=w font="font desc" text={blah blah}
# text(3990,4380) textangle=19 color=red width=2
      font="times 14 normal roman" text={All the data}
```

- Mouse creation: clicking adds the word 'Region' to the image. Double-clicking brings up a control box allowing you to change the text and properties such as the font and color.

- ‘ruler’

  'ruler' draws a two-sided arrow and text showing the length of the arrow. If the angle is nonzero, dashed lines projecting the arrow in x and y directions are also shown. The optional 'format' parameter lets you specify a C printf style format to control the way the number will be displayed.

  The 'ruler' region is mainly used for interactive measurements; for a distance indicator on printed output the 'line' region is recommended.

  - DS9 region file syntax and examples:

```
# ruler(x1,y2,x1,y2) ruler=coordsys unit
# ruler(x1,y2,x1,y2) ruler=coordsys unit format={c-format}
# ruler(4161,4177,4309,4337) ruler=fk5 degrees
# ruler(4161,4177,4309,4337) ruler=fk5 physical
# ruler(4116.0002,4071.0003,4297,4174) ruler=fk5 degrees format={%8.3f}
```

  - Mouse creation: draws a two-sided arrow and text showing the length of the arrow in world coordinate units (degrees). The double-click dialog box allows you some control over what quantity (physical or world) is measured and in what units.

- ‘compass’

  This draws a pair of arrows, one north and one east, to indicate the cardinal directions on the image. Changing the wcs to galactic makes the arrows be in the direction of galactic north and east. The labels used for the arrows can be edited. The NA and EA parameters can be 1 or 0 to control whether or not the north and east axis lines terminate in an arrowhead.

  - DS9 region file syntax and example:

```
# compass(x,y,length) compass=wcs {North-label} {East-label} NA EA
# compass(4091,4239,86.838931) compass=fk5 {N} {E} 1 1
# compass(36.6126736,53.0266578,65.355") compass=galactic {b} {l} 1 1
# compass(4091,4239,86.838931) compass=fk5 {Severniy} {Vostochniy} 1 1
```

– Mouse creation: draws a northward and eastward arrow (dragging selects the lengths of the arrows), labelled ’N’ and ’E’, useful for showing the axis directions on an illustration. The double-click dialog box allows you to change the labels.

Command:	**region analysis plot2d open**
Menu:	*[shap] - Analysis - Plot2D*
Description:	If a currently selected region is a display-only region, this will pop up a plot with the pixel values vs distance along the region

.

7.7 Composite regions

One may also define a ‘composite region’ by selecting multiple regions (e.g. using the shift-click to select second and subsequent regions) and then the Region/ Composite Region / Create menu entry. The composite region can then be moved around as a unit. It can be saved to a DS9 region file; an example is:

```
# Region file format: DS9 version 4.1
global color=green dashlist=8 3 width=1 font="helvetica 10 normal roman" select=
1 highlite=1 dash=0 fixed=0 edit=1 move=1 delete=1 include=1 source=1
physical
# composite(3988.5,4200,0) || composite=1
circle(4105,4153,46.6119) ||
circle(3872,4247,20)
circle(4253,4207,20)
```

Here the ’# composite(x,y,angle)” directive and the —— after the first circle region specify that the first two circle entries form a composite region; the third circle in this example is an independent region not part of the composite.

Region File syntax:	**composite** # composite (3988.5,4200.0) —— composite=1
Command:	**region composite**
Menu:	*Region- Composite Region - Create*
Description:	Creates a composite region.

Command: **region dissolve**

Menu: *Region - Composite Region - Dissolve*

Description: Dissolve a composite region

7.8 Template (Footprint) regions

In the GUI region menu, the *Region - Instrument FOV* entry allows the user to specify one of a set of template regions defined relative to the WCS pointing direction. Supported examples include the Chandra ACIS instrument footprint.

These regions are defined in terms of component regions (polygons, boxes etc). Saving such a region to a file saves the template as a composite region made up of these component regions.

Users may define their own templates and load them via the Region - Template menu.

Command: **region template [filename]**

Menu: *Region - Template*

Description: Open a template region file.

Command: **region template [filename] at [x y coordsys]**

Menu: *Region - Template*

Description: Open a template region and relocate it. Example: region template foo.tpl at 13:29:55.92 +47:12:48.02 fk5

Command: **region savetemplate [filename]**

Menu: *Region - Template*

Description: Save a template region to a file.

7.9 Generic region properties

These properties may be associated with a region in the properties section of a region file entry; most of the following properties may also be specified in the 'global' line of a region file, and they may be set in the interactive GUI in the Region menu (submenus Color, Properties, Font, etc).

First some properties that change the display of the region:

Region File syntax: **color**

# color=green
# color=#48f

Command: **region color [value]**

Menu: *Region - Color*

Description: Changes the region color. Values may be color names or hex values with 3, 6 or 9 digits. The default region color is green.

Region File syntax: **fill**

# fill = 1

Command: **region fill yes|no**

Menu: *Region - Fill*

Description: Determines if the region is displayed as an outline shape (fill=0) or a shape filled with the current region color (fill=1). In the GUI the fill value may be checked (1) or unchecked (0) in the Region - Color menu.

Region File syntax: **width**

# width=2

Command: **region width [value]**

Menu: *Region - Width*

Description: Determines the width of the line used to render the region outline. Default 1, values allowed are 1 to 4.

Region File syntax: **dash**

# dash=1

Command: **region dash yes|no**

Menu: *Region - Width - Dash*

Description: Lines will be dashed, using the current dashlist value. (Default 0).

Region File syntax: **dashlist**

#dashlist=a b
#dashlist=4 2

Command: -

Menu: -

Description: Dashlist. Sets the dash sequence to use if the dash property is set. The first parameter is the length of the dash, the second is the length of the inter-dash gap. Can only be set in the region file itself.

Region File syntax: **text**

# text={This message has both a ” and ’ in it}

Command: -

Menu: -

Description: Text. Draws the given text next to the region as a label.

Region File syntax: **font**

# font=”times 12 bold italic”

Command: **region font [value]**

Menu: *Region - Font*

Description: Determines the font family of any text used. Example: region font times. The default is ’helvetica 10 normal roman’. Options are times, helvetica, courier.

Region File syntax: **font**

# font=”times 12 bold italic”

Command: **region fontsize [value]**

Menu: *Region - Font - Size*

Description: Determines the font size of any text used. Example: xpaset -p ds9 region fontsize 12. Predefined menu options are 9, 10, 12, 14, 16, 20, 20, 30, 36, 72; there is an option for ‘other font size’ where a numerical integer value can be entered.

Region File syntax: **font**

# font=”times 12 bold italic”

Command: **region fontweight [value]**

Menu: *Region - Font - Weight*

Description: Determines the font weight for any text used. Example: xpaset -p ds9 region fontweight bold Options are: normal, bold.

Region File syntax:	**font**
	# font=”times 12 bold italic”
Command:	**region fontslant [value]**
Menu:	*Region - Font - Slant*
Description:	Determines the font slant for any text used. Example: xpaset -p ds9 region fontslant italic. Options are: roman, italic

Region File syntax:	**source|background**
	# source # background
Command:	**region source|background**
Menu:	*Region - Properties - Source/Background*
Description:	Properties that associate a region as being either source or background, or have more specific ’tags’ associated. These are not used by DS9 directly, but other applications may use them and the xpaget command can test them: ’xpaget ds9 region background’ tests if the currently selected region is a background one. KEYBOARD SHORTCUT: s (set source property); b (set background property for selected region).

Region File syntax:	**include**
	global include = 0
Command:	**region include|exclude**
Menu:	*Region - Properties - Include/Exclude*
Description:	This may only be used in the ’global’ line; include=1 or include=0 are set in the individual region lines using the leading + or -. Default is include=1. KEYBOARD SHORTCUT: i (set include =1); e (set include = 0).

Another set of properties control the interactive behaviour:

Region File syntax: **fixed**

# fixed = 1

Command: **region fixed yes|no**

Menu: *Region - Properties - Fixed In Size*

Description: fixed: if this is set to 1, changes to the image magnification do not scale the region size accordingly. Default is 0.

Region File syntax: **select**

# select = 0

Command: **-**

Menu: -

Description: select: if this is set to 0, mouse actions can neither select, edit, nor move the region. Default is 1.

Region File syntax: **highlite**

# highlite = 0

Command: **-**

Menu: -

Description: highlite: if this is set to 0, selecting the region does not make the ‘edit handles’ visible. Default is 1.

Region File syntax:	**edit**
	# edit = 0
Command:	**region edit yes|no**
Menu:	*Region - Properties - Can Edit*
Description:	edit: if this is set to 0, mouse actions do not allow the region to be edited even when selected. Default is 1.

Region File syntax:	**delete**
	# delete = 0
Command:	**region delete yes|no**
Menu:	*Region - Properties - Can Delete*
Description:	delete: if this is set to 0, mouse actions do not allow the region to be deleted. Default is 1.

Region File syntax:	**move**
	# move = 0
Command:	-
Menu:	*Region - Properties - Can Move*
Description:	move: if this is set to 0, mouse actions do not allow the region to be moved even when selected. Default is 1.

Region File syntax:	**rotate** # rotate = 0
Command:	**region rotate yes|no**
Menu:	*Region - Properties - Can Rotate*
Description:	rotate: if this is set to 0, mouse actions do not allow the region to be rotated even when selected. Default is 1.

Regions can be copied and pasted using xpaset:

Command:	**region copy|cut|paste**
Menu:	-
Description:	Copy, cut or paste seleted region. The paste command takes the coordsys as the argument, e.g. xpaset -p ds9 region paste wcs

7.10 Region tags

Region File syntax:	**tag**
	# tag ={Group 1}
	# tag={Quasars} tag={PointSrcs}

Command:	**region group [tag] new**
Menu:	*Region - New Group*
Description:	A way to group related regions. A single region line may have multiple tags associated. To select multiple regions, shift-click on the second and subsequent regions, or click-drag to select all regions within a rectangle. KEYBOARD SHORTCUT: g (create a new group). Shift-g (create a new group with default name).

Command: **region group [tag] select**

Menu: -

Description: Select all regions in a tag group

Command: **region group [tag] update**

Menu: -

Description: Update a tag group. For example : select the group. Now deselect some regions or select additional ones. Then 'update' will update the group membership list.

Command: **region group [tag] color [value]**

Menu: -

Description: Set color for regions in a tag group

Command: **region group [tag] font [string]**

Menu: -

Description: Set font for regions in a tag group. Example: region group foo font {times 14 bold}

Command: **region group [tag] cut|copy|delete**

Menu: -

Description: Copy, cut, delete regions in a tag group

Command: **region group [tag] move [dx] [dy]**

Menu: -

Description: Adjust position of all regions in a tag group

Command: **region group [tag] movefront|moveback**

Menu: -

Description: Move all regions in a group to the front or back

Command: **region group [tag] property [name] yes|no**

Menu: -

Description: Set the named property for all regions in a group. Example: region group foo property fill yes

Command: **region groups**

Menu: *Region - Groups*

Description: Brings up a dialog which lists the named groups. Clicking on the name of a group selects all the regions in that group. The same region can belong to more than one group.

7.11 Special region directives for complicated data

- Tile: For a mosaic detector with multiple segments (tiles), some coordinate systems (image, physical) apply on a per-tile basis. We can specify we are referring to tile 2 by

  ```
  tile 2;point(100,100)
  ```

- : Multiple WCS; FITS supports alternate WCS to the primary one, distinguished by a trailing letter (e.g. CTYPE2B keyword for WCS B). The primary WCS uses no trailing letter. We can specifiy this a wcsx keyword:

  ```
  wcsa;point(100,100)
  wcsf;point(50,50)
  wcs;point(1,10)
  ```

7.12 Region I/O

Command: **region load [filename]**

Menu: *Region - Open*

Description: This menu item puts up a file manager dialog to allow loading of region files from disk. By default only files with names ending in '.reg' are displayed in the dialog; you can click 'All' for all files to be displayed. The filename to be loaded is shown in a selection text box, and you can edit this name.
On selecting a filename with 'ok', a new dialog pops up allow you to control the file format to be read and whether it goes in the current frame or all frames. You can also override the region colors here. Usually you'll just want to click OK again.

Command: **region save [filename]**

Menu: -

Description: Allows you to save all currently defined regions to a file. The first dialog is similar to that from Region - Open and allows you to edit the filename. It will warn you if you are overwriting an existing file. The second dialog allows you to select the output format and coordinate system.

Command: **region format ds9|xml|ciao|saotng|saoimage|pros|xy**

Menu: *Region - Save - Save Regions - Format*

Description: Specify region file format flavour to save the regions in

Command: **region system image|physical|wcs|wcsa-z**

Menu: *Region - Save - Save Regions - Coordinate System*

Description: Specify coordinates for output region file

Command: **region sky fk4|fk5|icrs|galactic|ecliptic**

Menu: *Region - Save - Save Regions - Coordinate System*

Description: Specify celestial coordinates for output region file

Command: **region skyformat degrees|sexagesimal**

Menu: *Region - Save - Save Regions - Coordinate System*

Description: Specify celestial coordinate representation for output region file

Command: **region save select [filename]**

Menu: *Region - Save Selection*

Description: This is like Region - Save but only writes out the currently selected regions.

Command: **region delete**

Menu: *Region - Delete All*

Description: deletes all currently defined regions.

Command: **region delete select**

Menu: *Region - Delete Selection*

Description: deletes all currently selected regions. KEYBOARD SHORCUT: Del

Command: **region delete load [filename]**

Menu: *Region - Delete All and Open*

Description: is simply a combination of *Region - Delete All* followed by *Region - Open*.

Command: **region command [string]**

Menu: *-*

Description: Add a region. Example: 'region command {circle 100 100 20}

7.13 Region information

Command: **region strip yes|no**

Menu: -

Description: Display only selected regions (yes) or all regions (no, default)

Command: **region show yes|no**

Menu: -

Description: Regions are or are not visible on the display

Command: **region showtext yes|no**

Menu: -

Description: Region text labels are or are not visible on the display

Command: **region list**

Menu: *Region - List*

Description: lists all currently defined regions. It pops up the dialog to choose format and coordinate system to list the regions in. On pressing OK, it then raises a window showing what would be written to the region file if you did 'Region - Save'.

Command: **region list select**

Menu: *Region - List Selection*

Description: is like *Region - List* but only lists the selected regions.

Command: -

Menu: *Region - Get Information*

Description: pops up dialogs, one for each currently selected region, allowing access to the region edit properies (position, color, etc). This menu entry is really 'Region - Allow Me To Edit It'.

Command: **region [-options]**

Menu: -

Description: Access region information via xpaget. Options are of the form "-opt value" and include:
-format ds9, ciao, saotng, saoimg, pros, xy
-system image, physical, wcs, wcsa...z
-sky fk4, fk5, icrs, galactic ecliptic
-skyformat degrees, sexagesimal
-delim nl
-delim [character]
-prop select, edit, move, rotate, delete, fixed, include, source 1, source 0
-group [tag]
-strip yes|no
-wcs yes|no
Example: xpaget ds9 region -format ds9 -system wcs -sky fk5 -skyformat sexagesimal -prop edit 1 -group foo

Command: **region centroid**

Menu: *Region - Centroid*

Description: Calculates the centroid of the data near the middle of the selected region and adjusts the region to be centered at that point, redisplaying it.
A circle of radius r (default 10) pixels is used around the region center, and n (default 30) iterations are done.

Command: **region centroid radius|iteration [value]**

Menu: *Region - Region Parameters - Centroid Parameters - Radius/Iteration*

GUI Default: *Edit - Preferences - Region - Centroid - Radius/Iteration*

Description: Control r or n parameter for centroiding

Command: **region centroid auto yes|no**

Menu: *Region - Region Parameters - Auto Centroid*

Description: If set, a newly created region executes this procedure automatically.

Command: **region analysis stats open|close**

Menu: *Region - Get Information - Analysis - Statistics*

Description: open a dialog showing various statistics on the pixel values within the region.

Command: **region analysis stats save**

Menu: *Region - Get Information - Analysis - Statistics - File - Save Data*

Description: write the stats info to a file

Command: **region analysis histogram open|close**

Menu: *Region - Get Information - Analysis - Histogram*

Description: open a plot showing a histogram of the pixel values within the region.

Command: **region analysis histogram save**

Menu: *Region - Get Information - Analysis - Histogram - File - Save Data*

Description: write the histogram to a file

Command: **region analysis plot3d open|close**

Menu: *Region - Get Information - Analysis - Plot3D*

Description: For a data cube, open a plot showing a histogram of the pixel values within the region as a function of the z slice number

Command: **region analysis plot3d open|close**

Menu: *Region - Get Information - Analysis - Plot3D - File - Save Data*

Description: For a data cube, save a histogram of the pixel values within the region a a function of slice number

8 Data cubes and event files

8.1 Slices and cubes

A FITS Cube is a FITS image which contains more than 2 axes (NAXES>2). DS9 will automatically detect if a cube is present and will load all additional images. In addition, individual images can be loaded one at a time into a cube. DS9 will display the Cube dialog box which allows the user to select which 2 image to be displayed.

Command: **fits slice [filename]**

Menu: *File - Open As - Slice*

Description: Used to open an image and add it as a new slice at the end of the current data cube.

Command: **save fits [filename] slice**

Menu: *File - Save As - Slice*

Description: Save current slice as a FITS image.

8.2 Movies

Once a cube is created, *File - Create Movie - Slice Movie* followed by *File - Create Movie - MPEG* or *File - Create Movie - Animated GIF* will make an MPEG or GIF movie file. *File - Create Movie - Display* can be used to set the frame rate.

Command: **movie slice|frame mpeg|gif [delay] blink|fade [filename]**

Menu: *File - Create Movie*

Description: Saves the current slice set, or frames set, as an animated GIF or an MPEG. The delay and blink/fade arguments are optional.

Command: **blink yes|no**

Menu: *Frame - Blink Frames*

Description: This shows a single frame at any one instant, but every few seconds cycles through the set of frames.

Command: **blink interval [N]**

Menu: *Frame - Frame Parameters - Blink Interval - [N]*

Description: Sets the time interval in seconds for the Blink Frames mode

.

Command: **fade yes|no**

Menu: *Frame - Fade Frames*

Description: Cycles through a time sequence where the frames fade into one another.

Command: **fade interval [N]**

Menu: *Frame - Frame Parameters - Fade Interval - [N]*

Description: Sets the time interval in seconds for the Fade Frames mode

.

Command: **movie 3d slice|frame mpeg|gif [delay] [filename] [opts]**

Menu: *File - Create Movie - 3D movie*

Description: Saves the current 3D data, as an animated GIF or an MPEG. Options are 'number' (of frames), 'az from', 'az to', 'slice from ', 'slice to', 'zoom from n', 'zoom to n', 'oscillate n', 'repeat n'. Example: movie 3d mpeg ds9.mpg number 10 az from -60 az to 60 oscillate 1

8.3 Mosaic WCS files

Command: **mosaicimage wcs [filename]**

Menu: *File - Open As - Mosaic WCS*

Description: Mosaic images consist of multiple chips each with their own coordinate system.
Note: A FITS Mosaic Data Cube is a FITS mosaic image which contains more than 2 axes (NAXES>2). DS9 will automatically detect if a mosaic data cube is present and will load all additional images. At the same time, DS9 will display the data cube dialog box which allows the user to select which 2D image is to be displayed.

Command: **mosaic wcs [filename]**

Menu: *Open As - Mosaic WCS Segment*

Description: Open mosaic segment in current frame

Command: **mosaicimage wcs new [filename]**

Menu: -

Description: Open mosaic image in new frame

Command: **mosaic wcs new [filename]**

Menu: -

Description: Open mosaic segment in new frame

Command: **save mosaic [filename]**

Menu: *File - Save As - Mosaic WCS Segment*

Description: Save mosaic

Command: **save mosaicimage [filename]**

Menu: *File - Save As - Mosaic WCS*

Description: Save mosaic

Command: **scale scope local|global**

Menu: *Scale - Local|Global*

Description: Determines whether the range limit algorithm is applied separately for each mosaic segment or once for the whole dataset.

Command: **-**

Menu: *File - Open As - Mosaic WFPC2*

Description: Special support for WFPC images

8.4 Mosaic IRAF files

Command: **-**

Menu: *File - Open As - Mosaic IRAF*

Description: Loads a Mosaic IRAF file.

Command: **prefs irafalign yes|no**

Menu: *Edit - Preferences - General - Mosaic IRAF DETSEC Align*

Description: If this is set, the DETSEC metadata will be used to align the components of the mosaic.

Command: **-**

Menu: *File - Open As - Mosaic IRAF Segment*

Description:

8.5 Multiple Extension Fits Images

Some telescopes provide data with multiple image extensions.

A FITS Multiple Extension Data Cube file is a FITS file with one or more extensions, that is to be displayed as a data cube. Each image does not have to be the same size. However, only the coordinate systems from the first extension will be used for contours and grids.

DS9 provides a way to load these as a data cube;

Command: **mecube [filename]**

Menu: *File - Open As - Multiple Extension Cube*

Description: Load ME FITS file as data cube into current frame.
Example: ds9 -mecube foo.fits # load multiple extension fits file as data cube

Command: **multiframe [filename]**

Menu: *File - Open As - Multiple Extension Frames*

Description: Load ME FITS file as separate images in separate frames.
Example: ds9 -multiframe foo.fits # load multiple extension fits file as multiple frames

Command: **mecube new [filename]**

Menu: -

Description: Load ME FITS file as data cube into new frame

Command: **save mecube [filename]**

Menu: *File - Save As - Multiple Extension Cube*

Description: Save current data as an ME FITS file.

8.6 Table files other than event files

On opening a FITS file, DS9 makes an image from the first non-null HDU which is either a FITS IMAGE or a binary table with a compatible dataset (ZIMAGE compressed image, HEALPIX image, or event file0.

If binary table extension is found for which the ZIMAGE keyword is T, a compressed FITS image is read.

If binary table extension is found for which the PIXTYPE keyword is HEALPIX, a HEALPIX image is read. Only full-field HEALPIX images are supported.

If the file doesn't have any valid images or event files, an error is reported.

8.7 Event List files

An X-ray event file is stored as a FITS table with a set of columns, usually including ones called X and Y. DS9 recognizes an event file if it finds a binary table for which the EXTNAME keyword is EVENTS, STDEVENT or RAYEVENT; and for which column names X and Y are present.

When the file is loaded, DS9 makes an image on the fly, binning the data on the X and Y columns with a pixel size of 1 in the X and Y column units.

Columns with other names can be used as the columns to bin on instead; I will call these the axis columns. The center of the image is found from the image bounds determined using any of the following keywords, checked in this order: TDMIN/TDMAX, TLMIN/TLMAX, TALEN, and AXLEN.

In other words, if you are binning on columns named ENERGY and TIME with FITS keywords TTYPE5='ENERGY',TTYPE13='TIME', then it looks for the keywords TDMIN5 and TDMAX5 to define the range of the x (i.e. ENERGY) axis, and failing that TLMIN5 and TLMAX5, and then TALEN5, and finally AXLEN5. If none of these are present it uses the possible range of the data type of the axis columns (e.g 1 to 32767 for a 2 byte integer column).

By default only a 1024 x 1024 subset of the image at a binning value of 1 is loaded.

Instead of using the GUI to define the columns to bin on, you can use a qualified filename in the 'open file' dialog (or on the command line) of the form

```
filename[ext][binspec][filterspec][sect]
```

The ext and sect specifiers are the same as for the FITS image case. The binspec is of the form "bin=colx,coly" or "bin=colx,coly,colz" and the filterspec is of the form of entries with 'colname=value' or 'colname logical operator value' separated by '&&'.

Examples:

```
ds9 foo.fits # default load
ds9 foo.fits[1] # load first extension
ds9 foo.fits[BCKGRD] # load extension named 'BCKGRD'
ds9 foo.fits[bin=detx,dety] # bin on detx,dety
ds9 foo.fits[2][bin=rawx,rawy] # load ext 2, cols rawx,rawy
ds9 foo.fits[bg_events][bin=rawx,rawy] # load ext bg_events, cols rawx,rawy
ds9 foo.fits[bin=x,y,pha] # bin on x,y,pi
ds9 foo.fits[bin=pi] # bin on x,y,pi
ds9 'foo.fits[ccd_id==3&&energy>4000]' # quoted filter
ds9 '"foo.fits[ccd_id==3 && energy>4000]"' # double quoted filter
ds9 'foo.fits[events][pha>5,pi<2]' # load extension 'events' and filter
```

The shell environment variable DS9_BINKEY may also be used to specify default bin cols for FITS bin tables. Example:

```
$ export DS9_BINKEY='[bin=rawx,rawy]'
$ ds9 foo.fits # load FITS bin table, bin on rawx, rawy
```

Command: **bin cols [xname] [yname**

Menu: *Bin - Binning Parameters - Bin Columns*

Description: Select the columns to bin on.

Command: **bin colsz [xname] [yname] [zname]**

Menu: *Bin - Binning Parameters - Bin 3rd Column*

Description: Select one of the other columns as a Z axis and create a 3D image.

Command: **bin depth [value]**

Menu: *Bin - Binning Parameters - Bin 3rd Column - Depth*

Description: Selects the number of bins to use on the third axis.

Command: **-**

Menu: *Bin - Binning Parameters - Apply*

Description: Causes the data cube to be created and the data cube dialog box to be opened. The first slice of the data cube is displayed; the data cube dialog box commands such as *Data cube - Next* can be used to navigate the slices of the cube.

Data cube - Interval selects a time interval to control how *Data cube - Play* sequentially cycles through the slices of the cube.

Command: **bin filter [string]**

Menu: *Bin- Binning Parameters - Bin Filter*

Description: The data may be filtered; Enter a comma separated list of CIAO type filter strings ”colname=min:max” entered in the dialog box (e.g. ”energy =2500:8000, ccd_id=7”) and then execute *Bin- Binning Parameters - Apply*. > and < may also be used to define ranges.

Note that if your /tmp directory is mounted -noexec, binary table filtering does not work. Before starting DS9, set the environment variable FILTER_TMPDIR to a directory that is both writable and can execute.

Command: **save fits [filename]**

Menu: *File - Save*

Description: The filtered data may be saved as a FITS image. In the GUI, after you press ’ok’ on the file selection dialog, a small dialog appears allowing you to select Image or Table.

8.8 HEALPIX files

HEALPIX is an encoding designed for all-sky data. FITS supports the HEALPIX paradigm with binary or ascii tables.

Only full-sky HEALPIX images are supported in DS9.

At load time, the user may provide just a file name or a file name along with FITS extension name/number, image section specification, and Healpix parameters. DS9 will automatically convert a FITS HEALPIX binary or ascii table into a 2D image for display. FITS extension names and parameters are case insensitive. The users may specify a number of parameters on how to construct the image. Any table with keyword PIXTYPE=HEALPIX or NSIDE=x will be processed as an HEALPIX image. The following FITS keywords will be used if present and not overwritten by a command line option: NSIDE, COORDSYS, ORDER.

Syntax:

```
filename
filename[ext]
filename[ext][sect]
filename[sect]
filename[ext,sect]

filename[ext][hpx]
```

```
    filename[ext][hpx][sect]
    filename[ext][sect][hpx]
    filename[hpx]
    filename[hpx][sect]
    filename[sect][hpx]
    filename[ext,hpx]
```

where [ext] and [sect] are extension and subset specifiers as described earlier for standard FITS images and the [hpx] specifier is special to HEALPIX with the following forms:

```
    [order=ring|nested] # default ring
    [layout=equatorial|north|south] # default equatorial
    [col=<column number>] # defaut 1
    [quad=<quadurant number>] # (1-4) default 1
    [system=equatorial|galactic|ecliptic|unknown] # default unknown
```

Example:

```
ds9 foo.fits # default load
ds9 foo.fits[1] # load first extension
ds9 foo.fits[order=ring,layout=equatorial,col=1,quad=1,system=unknown]
ds9 foo.fits[1,order=nested] # first extension, nested order
```

9 Special file formats

9.1 Raw binary array

Raw data arrays are supported. To load an array, the user must provide the dimensions, pixel depth, and optional header size and architecture type. This may be done by specifying a 'qualified filename' of the form

```
filename[xdim=n1,ydim=n2,zdim=n3,bitpix=m,endian=native$|$little$|$big,skip=N].
```

where

- xdim, ydim, and optionally zdim define the dimensions of the array.
- bitpix defines the data type of the array using the FITS BITPIX conventoin
- endian specifies whether the data is native, little (Intel) or big endian.
- skip gives N, an even integer instructs the reading process to skip a given number of bytes. N should usually be a multiple of 4.

Examples:

```
ds9 -array foo.data[xdim=512,ydim=512,zdim=1,bitpix=16] # load 512x512 short
ds9 -array foo.data[dim=256,bitpix=-32,skip=4] # load 256x256 float with 4 byte head
ds9 -array foo.data[dim=512,bitpix=32,arch=little] # load 512x512 long, intel
```

An alternate supported syntax for the qualified filename is

```
filename[array([type][dim]:[skip][endian]]
```

where

- type is a single character defining the data type:

```
      'b' 8 -bit unsigned char
      's' 16-bit short int
      'u' 16-bit unsigned short int
      'i' 32-bit int
      'l' 64-bit int
      'r' 32-bit float
      'f' 32-bit float
      'd' 64-bit float
```

- dim can be one of:
  - a single integer for the xy dims, e.g: 1024 - implies the array is 1024 x 1024
  - a pair of integers specifying the x and y dims separately, separated by a period: e.g. 1024.2048
  - a triple of integers specifying x,y,z dims, separated by periods, e.g 512.512.512
- skip: an integer, the number of bytes to skip.
- endian: letter 'l' or 'b' for little or big endian.

Examples:

```
ds9 -array bar.arr[array(s512)]   # load 512x512 short
ds9 -array bar.arr[array(r256:4)] # load 256x256 float with 4 byte head
ds9 -array bar.arr[array(i512l)]  # load 512x512 long, intel
```

You can also use the environment variable DS9_ARRAY to specify the qualification string, example:

```
export DS9_ARRAY='[dim=256,bitpix=-32]'
ds9 -array foo.arr # load 256x256 float
```

Command: **array [filename][opts]**

Menu: *File - Import - Array*

Description: Directly load in a raw binary data array. The options (in the Import Array dialog) describe the array dimensions and the data type: [xdim=n1,ydim=n2,zdim=n3,bitpix=m,endian=native|little|big,skip=N]. Example: ds9 -array foo.arr[dim=512,bitpix=-32,endian=little]. The skip option allows you to skip the first N bytes. The option 'array new [filename]' loads into a new frame.

Command: **export array [filename] big|little|native**

Menu: *File - Export - Array*

Description: Exports the image to a raw binary data array, specitying the endianness.

Command: **array mask [filename]**

Menu: *Analysis - Mask Parameters - File - Import - Array*

Description: Loads a raw binary data array as a mask.

9.2 Shared memory data

DS9 has the ability to load a shared memory segment into the current frame.

Command: **shm key [n]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm key 102.

Command: **shm shmid [n] [name]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm shmid 102 foo.

Command: **shm fits key [n] [name]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm fits key 100 foo.

Command: **shm mosaicimage iraf key [n] [name]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm iraf key 100 foo.

Command: **shm mosaicimage wcs key [n] [name]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm mosaicimage wcs key 100 foo.

Command: **shm mosaic wcs key [n] [name]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm mosaic wcs key 100 foo.

Command: **shm rgbcube|rgbimage|rgbarray key [n] [name]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm rgbcube key 100 foo.

Command: **shm array key|shmid [n] [xdim=x,ydim=y,bitpix=b,skip=s]**

Menu: -

Description: Load shared memory segment. Example: xpaset -p ds9 shm array shimid 102 [dim=32,bitpix=-32]

9.3 JPEG

DS9 can read JPEG images. For a normal frame, the average of the luminosity (the average of R, G, B) is used. For an RGB frame, the file is loaded directly to the separate channels.

Command: **export jpeg [filename] [value]**

Menu: *File - Export - JPEG*

Description: Save image data to a JPEG file, choosing the JPEG quality factor (0 to 100). Export saves the image with whatever scaling/limits/colorbar/contrast/bias/etc applied. It does not include any regions, illustrations. The output is the same raw/original dimensions as the image read in, ignoring any zoom performed.

Command: **jpeg [filename]**

Menu: *File - Import - JPEG*

Description: Load a JPEG image into DS9. Only the IMAGE coordinate ssytem is available.

Command: **saveimage [filename].jpeg [value]**

Menu: *File - Save Image- JPEG*

Description: Create a JPEG of a screen capture of the current main window, as zoomed etc,, including colorbar and overlays such as regions.

Command: **jpeg new [filename]**

Menu: -

Description: Load a JPEG image into a new frame. Only the IMAGE coordinate ssytem is available.

Command: **jpeg slice [filename]**

Menu: *File - Import - Slice - JPEG*

Description: Load a JPEG image into the current slice.

9.4 GIF

DS9 can read GIF images. For a normal frame, the average of the luminosity (the average of R, G, B) is used. For an RGB frame, the file is loaded directly to the separate channels.

Command: **export gif [filename]**

Menu: *File - Export - GIF*

Description: Save current image as a GIF file. Export saves the image with whatever scaling/limits/colorbar/contrast/bias/etc applied. It does not include any regions, illustrations. The output is the same raw/original dimensions as the image read in, ignoring any zoom performed.

Command: **gif [filename]**

Menu: *File - Import - GIF*

Description: Load a GIF image into DS9. Only the IMAGE coordinate ssytem is available.

Command: **saveimage gif [filename]**

Menu: *File - Save Image - GIF*

Description: Create a GIF of a screen grab of the current main window, including colorbar and overlays such as regions.

Command: **gif new [filename]**

Menu: -

Description: Load a GIF image to a new frame

Command: **gif slice [filename]**

Menu: *File - Import - Slice - GIF*

Description: Load a GIF image into the current slice.

9.5 PNG

DS9 can read PNG images. For a normal frame, the average of the luminosity (the average of R, G, B) is used. For an RGB frame, the file is loaded directly to the separate channels.

Command: **export png [filename]**

Menu: *File - Export - PNG*

Description: Save current image as a PNG file. Export saves the image with whatever scaling/limits/colorbar/contrast/bias/etc applied. It does not include any regions, illustrations. The output is the same raw/original dimensions as the image read in, ignoring any zoom performed.

Command: **png [filename]**

Menu: *File - Import - PNG*

Description: Load a PNG image into DS9. Only the IMAGE coordinate ssytem is available.

Command: **saveimage png [filename]**

Menu: *File - Save Image - PNG*

Description: Create a PNG of a screen grab of the current main window, including colorbar and overlays such as regions.

Command: **png new [filename]**

Menu: -

Description: Load a PNG image to a new frame

Command: **png slice [filename]**

Menu: *File - Import - Slice - PNG*

Description: Load a PNG image to the current slice.

9.6 EPS

Command: **saveimage eps [filename]**

Menu: *File - Save Image - EPS*

Description: Create an EPS file of a screen grab of the current main window, including colorbar and overlays such as regions.

9.7 TIFF

DS9 can read TIFF images. For a normal frame, the average of the luminosity (the average of R, G, B) is used. For an RGB frame, the file is loaded directly to the separate channels.

Command: **export tiff [filename] [compression]**

Menu: *File - Export - TIFF*

Description: Save current image as a TIFF file. Specify the compression option. Options are none, jpeg, packbits, and deflate. Export saves the image with whatever scaling/limits/colorbar/contrast/bias/etc applied. It does not include any regions, illustrations. The output is the same raw/original dimensions as the image read in, ignoring any zoom performed.

Command: **tiff [filename]**

Menu: *File - Import - TIFF*

Description: Load a TIFF image into DS9. Only the IMAGE coordinate ssytem is available.

Command: **saveimage [filename].tiff [compression]**

Menu: *File - Save Image - TIFF*

Description: Create a TIFF of a screen grab of the current main window, including colorbar and overlays such as regions. Specify compression to use.

Command: **tiff new [filename]**

Menu: -

Description: Load a TIFF image to a new frame

Command: **tiff slice [filename]**

Menu: *File - Import - Slice - TIFF*

Description: Load a TIFF image to the current slice.

9.8 ENVI

ENVI is the NV5 Environment for Visualizing Images, used for geospatial applications. BIL, BIP and BSQ encodings are supported.

Command: **envi [new] [header][filename]**

Menu: *File - Import - ENVI*

Description: Load an ENVI header and file. Example:
xpaset -p ds9 envi foo.hdr foo.bsq
xpaset -p ds9 envi new bar.hdr

Command: **export envi e.hdr e.bsq [big—little—native]**

Menu: *File - Export- ENVI*

Description: Export the current image to an ENVI format header and file. Use the endianness specified.

9.9 NRRD

NRRD (Nearly Raw Raster Data) is a file format for N-dimensional images. DS9 supports the raw and gzip encodings.

Command:	**export nrrd [filename] [big|little|native]**
Menu:	*File - Export- NRRD*
Description:	Export the current image to an NRRD format file. Use the endianness specified.

Command:	**nrrd [filename]**
Menu:	*File - Import- NRRD*
Description:	Load an NRRD file. The filename may have a '[sect]' qualification

Command:	**nrrd new [filename]**
Menu:	-
Description:	Load an NRRD file in a new frame.

Command:	**nrrd mask [filename]**
Menu:	*Analysis - Mask Parameters - File - Import - NRRD*
Description:	Loads a mask from an NRRD format file.

9.10 Used-defined file formats

DS9 supports external file formats via an ASCII description file. The format description file (FDF) is called .ds9.filin and is looked for in the the local directory and then the user home directory.

The FDF consists of a sequence of file format descriptions separated by blank lines and optional comment lines starting with #. Each format description has four lines:

- Help description
- A space-separated list of templates
- A space-separated list of file types (not currently used)
- The command line for the loading this file type (with '$filename' representing the filename)

An example FDF defining an IRAF IMH file:

```
For Example:
# File access descriptions:
#       help explanation
#       file template
#       file type
#       access command
IRAF IMH files
*.imh
IMH
i2f -s $filename
```

The user would then read a file foo.imh in this format into DS9 with

```
ds9 -i2f -s foo.imh
```

DS9 assumes that the output of the user command (i2f in this case) is a streamed FITS file with the image data, sent to standard output. DS9 then sucks in that data from standard input, equivalently to

```
cat foo.fits | xpaset ds9 fits
```

10 Frames

Simple use of DS9 has a single image in the main display area. However, DS9 can load multiple images into memory; each image is a 'Frame'. When multiple frames are loaded, you can display one frame at a time or all of them side by side.

10.1 Frames - basic use

Command:	frame new
Menu:	*Frames - New Frame*
Description:	Creates a new frame. If only the single default frame was present, whatever was previously displayed in the main window is replaced by a blank canvas. Then using the usual *File - Open* process a new image can be loaded in the frame and displayed.

There are several modes to view frames.

Command: **tile yes|no**

Menu: *Frame - Tile Frames*

Description: Displays all the frames at once in a grid in the main image pane. A 'current frame' is still defined, and the Go To Frame commands will select which one.

Command: **tile no**

Menu: *Frame - Single Frame*

Description: This displays a single frame at a time - the 'current frame'; this is the default.

Command: **single**

Menu: *Frame - Single Frame*

Description: This displays a single frame at a time - the 'current frame'; this is the default.

Command: **frame frameno [n]**

Menu: *Frame - Go To Frame - Frame [n]*

Description: This will select the nth frame to be the current one. In Single Frame mode, it will redisplay this frame.

Command: **frame first**

Menu: *Frame - First Frame*

Description: This will make the first frame in the list be the current one. In Single Frame mode, it will display this frame.

Command: **frame previous**

Menu: *Frame - Previous Frame*

Description: This will make the previous frame in the list be the current one. In Single Frame mode, it will display this frame. KEYBOARD SHORTCUT: Shift-Tab

Command: **frame next**

Menu: *Frame - Next Frame*

Description: This will make the next frame in the list be the current one. In Single Frame mode, it will display this frame. KEYBOARD SHORTCUT: Tab

Command: **frame last**

Menu: *Frame - Last Frame*

Description: This will make the last frame in the list be the current one. In Single Frame mode, it will display this frame.

Command: **tile grid|column|row**

Menu: *Frame - Frame Parameters - Tile - Grids|Columns|Rows*

Description: This controls the layout of the tiled frames, either as a 2D grid, a single vertical column, or a single horizontal row.

Command: **tile grid mode automatic|manual**

Menu: *Frame - Frame Parameters - Tile - Tile Parameters - Grid - Automatic|Manual*

Description: Automatic layour of the frames is the default. In manual mode, the number of grid columns and rows and the gap in pixels between the frames can be controlled.

Command: **tile grid direction x|y**

Menu: *Frame - Frame Parameters - Tile - Tile Parameters - Direction - X|Y*

Description: Specifies whether the increment for each new frame is horizontal or vertical (?). The menu variant updates the screen following *Frame - Frame Parameters - Tile - Tile Parameters - Apply*.

Command: **tile grid layout [N M]**

Menu: *Frame - Frame Parameters - Tile - Tile Parameters - Layout - Column, Rows - [N] [M]*

Description: Specifies the number of columns and rows in the grid. The menu variant updates the screen following *Frame - Frame Parameters - Tile - Tile Parameters - Apply*.

Command: **tile grid gap [N]**

Menu: *Frame - Frame Parameters - Tile - Tile Parameters - Layout - Columns|Rows - [N]*

Description: Specifies the gap in pixels between grid frames. The menu variant updates the screen following *Frame - Frame Parameters - Tile - Tile Parameters - Apply*.

Command: **frame move first|back|forward|last**

Menu: *Frame - Move Frame - First/Backward/Forward/Last*

Description: Shuffles the order of the frames in the list

Command: **frame show|hide [N]**

Menu: *Frame - Show/Hide Frames - Frame [N]*

Description: Frames can be temporarily hidden with selecting or deselecting frame number n.

Command: **frame show|hide all**

Menu: *Frame - Show/Hide Frames - All*

Description: Show or hide all frames at once

.

Command: **frame center**

Menu: *Zoom - Center Image*

Description: Center the current frame on the image center

Command: **frame center [n]|all**

Menu: -

Description: Center frame n, or center all frames

Command: **frame clear**

Menu: *Frame - Clear Frame*

Description: Clears the current frame, replacing the image with a blank canvas.

Command: **frame clear [n]|all**

Menu: -

Description: Clear numbered frame or all frames

Command: **frame refresh**

Menu: *Frame - Refresh Frame*

Description: Updates certain changes to the current frame.

Command: **frame reset**

Menu: *Frame - Reset Frame*

Description: Removes certain changes to the current frame.

Command: **frame delete**

Menu: *Frame - Delete Frame*

Description: Deletes the current frame, so there is one fewer frame in the list of frames.

Command: **frame delete [N]**

Menu: -

Description: Deletes the Nth frame, so there is one fewer frame in the list of frames.

Command: **frame delete all**

Menu: *Frame - Delete Frame*

Description: Deletes all the frames

Command: **movie frame mpeg|gif [DT] blink|fade [filename]**

Menu: *File - Create Movie - Format - Frames Movie*

Description: Creates a movie where multiple DS9 frames are displayed in sequence. Either MPEG or animated GIF formats are available. The GUI version has options *File - Create Movie - Type- MPEG|GIF*, *File - Create Movie - Transition - Blink|Fade* and *File - Create Movie - Delay - [DT]*. The delay is in units of 0.01s and the default is 0.50s.

Command: **match|lock frame wcs**

Menu: *Frame - Match|Lock - Frame - WCS*

Description: Aligns the frames to match the frames to the WCS of the current frame. If lock is used, the frames stay matched as you change (zoom, pan etc) the current frame. Most useful in Tile Frames mode.

Command: **match|lock frame [coordsys]**

Menu: *Frame - Match|Lock - Frame - [coordsys]*

Description: Aligns the frames to match the frames to the given coordinate system of the current frame (WCS, physical, image, WCS a, etc. If lock is used, the frames stay matched as the current one is changed.

There are an additonal set of commands used with xpaget to query the state of the frame:

Command: **frame has contour**

Menu: *-*

Description: Does the frame have contours.

Command: **frame has contour aux**

Menu: -

Description: Does the frame have auxiliary contour information.

Command: **frame has fits**

Menu: -

Description: Does the frame have a FITS dataset loaded

Command: **frame has fits bin|cube|mosaic**

Menu: -

Description: Does the frame have a FITS binary table, cube or mosaic loaded

Command: **frame has marker highlite|paste|select|undo**

Menu: -

Description: Does frame have certain states set

Command: **frame has [coordsys]**

Menu: -

Description: Example: 'xpaget ds9 frame has physical' - does the frame have a physical coord system.

Command: **frame has system [coordsys]**

Menu: -

Description: Example: 'xpaget ds9 frame has system physical' - does the frame have a physical coord system.

Command: **frame has wcs [coordsys]**

Menu: -

Description: Example: 'xpaget ds9 frame has wcs wcsp' - does the frame have a wcs 'p' coord system.

10.2 Crosshair

Command: **mode crosshair**

Menu: *Edit - Crosshair*

Description: Displays a crosshair in each frame. The user can use the mouse (left click and drag) to move the crosshair in the current frame.

Command: **match|lock crosshair [coordsys]**

Menu: *Frame - Match|Lock - Crosshair - WCS*

Description: If in crosshair mode, will let the user move the crosshair and then click, in the current frame and the crosshairs in the other frames jump to the corresponding WCS-matched position. If lock is used, the frames stay matched as you move the crosshair. That is, the crosshairs in the other frames dynamically follow the WCS position of the crosshair in the current frame.

Command: **mode crosshair**

Menu: *Edit - Crosshair*

Description: In Crosshair mode, a crosshair will be superimposed on the image and the left mouse button will move the location of the crosshair. Clicking will make the crosshair jump to the mouse position.

Command: **crosshair [x] [y] [coordsys] [skyframe]**

Menu: -

Description: Command the crosshair to a given position. Examples:
crosshair 100 100 physical
crosshair 23:01:00 +58:52:51 wcs fk5

10.3 Simple frames - advanced operations

Tile Frames is usually most useful for these. While a set of different coordinate systems are supported, WCS (the default WCS of the data in the current frame, usually RA/Dec) is usually what you want.

Command: **mode crop**

Menu: *Edit - Crop*

Description: Selects crop mode. A left mouse click followed by dragging and release will define a rectangle in the current frame. On release, the image will be cropped to this rectangle.

Command: **match|lock crop [coordsys]**

Menu: *Frame - Match/Lock - Crop - [coordsys]*

Description: If crop mode is selected, this will make the cropped area in the other frames match the one in the current frame.

Command: **crop match|lock [coordsys]**

Menu: -

Description: Same as the above

Command: **bin match|lock**

Menu: *Frame - Match/Lock - Bin*

Description: makes the binning of other frames match the binning of the current frame.

Command: **block match|lock**

Menu: *Frame - Match/Lock - Block*

Description: makes the blocking of other frames match the binning of the current frame.

Command: **colorbar lock**

Menu: *Frame - Match/Lock - Colorbars*

Description: Locks the frame colorbars to that of the current frame.

Command: **match|lock scale**

Menu: *Frame - Match/Lock - Scale*

Description: Locks the frame scales.

Command: **match|lock scalelimits**

Menu: *Frame - Match/Lock - Scale and Limits*

Description: Locks the frame scales and min/max values.

Command: **match|lock smooth**

Menu: *Frame - Match/Lock - Smooth*

Description: Locks the frame smoothing functions

Command: **match|lock axes**

Menu: *Frame - Match/Lock - Axes Order*

Description: Locks the axes order for 3D frames.

Command: **match|lock 3d**

Menu: *Frame - Match/Lock - 3d*

Description: In the case where *Edit - 3d* mode is selected, allows user rotation of the current frame to drive the rotation of other frames.

10.4 Color (RGB/HSV/HLS) frames

A color frame displays a 3D color image with three separately addressable color channels. DS9 supports three color models using different triples of color coordinaes to describe the color: RGB (red, green, blue); HSV (hue, saturation, value); HLS (hue, lightness, saturation). I will descibe the RGB cases; HSV and HLS cases are parallel.

A FITS RGB image on disk may be three separate FITS image files, one FITS file with three extensions, or as a FITS 3D n x n x 3 data cube, with three slices, each representing the red, green, and blue channel. A FITS RGB image may be loaded all a one time, or one channel at a time. Once loaded, the multiple FITS images are treated as one FITS image.

Example:

```
ds9 -rgb -red r.fits -green g.fits -blue b.fits # rgb image from 3 fits images
ds9 -rgbimage rgb.fits # load rgb image consisting of one fits file with 3 image exts
ds9 -rgbcube cube.fits # load rgb image consisting of one fits data cube
```

You can open an RGB frame and populate it with data from red, green and blue files from the DS9 command line by

```
ds9 -rgb -red red.fits -green green.fits -blue blue.fits
```

For HSV and HLS the equivalent commands are

```
ds9 -hsv -hue  h.fits  -saturation s.fits -value v.fits
ds9 -hls -hue  h.fits  -lightness l.fits -saturation s.fits
```

Command: **rgb|hsv|hls**

Menu: *Frame - New Frame RGB/HSV/HLS*

Description: Creates a new RGB/HSV/HLS frame. The frame has three image planes, one for each of the color coordinates.

Command: **frame new rgb**

Menu: *Frame - RGB*

Description: Pops up an RGB control window which allows one of the three coordinates to be selected as 'current', and to view or hide each of the three coordinates. By default all three coordinates red, green, blue are visible, making a color image, but it can be convenient to switch two of them off to see just one of the color planes. (HSV, HLS similar)

Command: **rgb red|green|blue**

Menu: *Frame - RGB - Current - Red|Green|Blue*

Description: Select red, green or blue as the current RGB channel

Command: **rgb red|green|blue [filename]**

Menu: *File - Open*

Description: Load a file into the current RGB channel. To load an RGB image in the GUI: Select 'red' as current and execute *File - Open* to load a file in the red channel. Then select 'green' as current, and again *File - Open* to load the green file; then select 'blue' as current, and again *File - Open* to load the blue file.

On the command line, you can do

ds9 -rgb -rgb channel red -fits e.red -rgb green -fits e.green -rgb blue -fits e.blue

or more simply

ds9 -rgb -red foo.red -green -foo.green -blue foo.blue

Note that ds9 -rgb red foo.red -rgb green foo.green -rgb blue foo.blue loads into three separate frames because you haven't created a new RGB frame.

Command: **rgb channel red|green|blue**

Menu: *Frame - RGB - Current - Red|Green|Blue*

Description: 'rgb channel red' is the same as 'rgb red'. You can then load a file in the channel

Command: **rgb view red|green|blue yes|no**

Menu: *Frame - RGB - View - Red|Green|Blue*

Description: Select or deselect the given RGB channel to make the relevant data visible or not.

Command: **rgbarray filename[opts]**

Menu: *File - Import - RGBArray*

Description: Load a raw data array cube into an RGB frame. Example: xpaset -p ds9 rgbarray foo.arr[dim=512,zdim=3,bitpix=-32,endian=little]

Command: **rgbarray new filename[opts]**

Menu: *File - Import - RGBArray*

Description: Load a raw data array cube into a new RGB frame. Example: xpaset -p ds9 rgbarray foo.arr[dim=512,zdim=3,bitpix=-32,endian=little]

Command: **rgbcube [new] [filename]**

Menu: *File - Open As - RGB Cube*

Description: Load a FITS RGB cube into an existing or new RGB frame

Command: **rgbimage [new] [filename]**

Menu: *File - Open As - RGB Image*

Description: Load a FITS RGB image into an existing or new RGB frame.

Command: **save rgbcube [filename]**

Menu: *File - Save As - RGB Cube*

Description: Write out RGB frame as a FITS file with an N x M x 3 cube in the primary array.

Command: **save rgbimage [filename]**

Menu: *File - Save As - RGB Image*

Description: Write out RGB frame as a FITS file with a null primary and 3 image extensions (r,g,b)

Command: **view rgb red|green|blue yes|no**

Menu: -

Description: Display the RGB values of the current pixel in the DS9 info pane

Command: **-**

Menu: *Frame - RGB - Align - (coordsys)*

Description: Aligns the frame based on the given coordinate system; Image is the default, but WCS may be useful.

Command: **rgb lock [option]**

Menu: *Frame - RGB - Lock*

Description: Lets you lock the R, G, B frames on any of the following: wcs, crop, slice, bin, axes, scale, scalelimits, block, smooth. See the single-frame case for more explanation.
By default each of the frames has indepdendent bin, scale, smoothing and colorbar but Lock allows them to be locked together so changes affect all the frames at once.

For an event list, you can load the same event list into each of the R, G, B frames and then on each of the frames use *Bin - Binning Parameters - Bin Filter* followed by *Bin - Binning Parameters - Apply* to apply different energy filters (e.g. ’energy=500:1000’, ’energy=1000:2500’, ’energy=2500:7000’) to each of the frames, to generate an RGB ’true X ray color’ image.

10.5 3D mode

There are two ways of viewing a data cube: you can load a 3D image into a regular 2D frame and step through ’slices’ in the z direction one at a time, or you can use a 3D frame which implements a simple ray-tracing method to view a cube from different angles.

The 3D module implements a simple ray-trace algorithm. For each pixel on the screen, a ray is projected back into the view volume, based on the current viewing parameters, returning a data value if the ray intersects the FITS data cube. To determine the value returned, there are 2 methods available, Maximum Intensity Projection (MIP) and Average Intensity Projection (AIP). MIP returns the maximum value encountered, AIP returns an average of all values encountered. At this point, normal DS9 operations are applied, such as scaling, clipping and applying a color map.

Rendering time is independent of the actual data cube size. Instead, the time it takes to render is based on how many rays are needed to project the data cube upon the screen in the view volume and the current zoom factor. The new module requires no special hardware or graphical processor unit (GPU) and the rendering time is adequate for interactive GUI manipulation on most computers. The rendering engine is developed using the POSIX thread library, allowing multiple light weight processes to be spawned to complete an image in parallel. The number of threads actually generated is a user specified parameter. Since all modern hardware contain multiple CPU cores, the default value is 8 threads. For larger work stations, this number can be increased. For every doubling in the number of CPU cores available, rendering times decrease approximately 75%.

Example:

```
# create 3d frame, load fits file
# set view angle to az 45 el 30 deg
# set rendering method to Average Intensity Projection

% ds9 -3d mycube.fits-3d vp 45 30-3d method aip

% xpaset -p ds9 3d
% xpaset -p ds9 file mycube.fits
% xpaset -p ds9 3d vp 45 30
% xpaset -p ds9 3d method aip
```

All 2-D graphics, regions, cross hairs, contours, and coordinate grids, are applied to the current slice, which is selected by the user. When the user wishes to match or lock a 2-D image and to a 3-D data cube, the current slice is used to determine the rendering solution. The user can crop the data cube for all 3 axes via the command line or the GUI. The new module also supports FITS event files binned into a data cube.

All printing support has been extended to the new 3-D module. The user may generate 3-D images in Postscript, JPEG, TIFF, and other formats, just as in the 2-D case. Furthermore, native printing is supported for the Windows version.

DS9 analysis macros have been enhanced to fully support the new 3-D module, allowing the user to invoke external analysis tasks based on the current view parameters and to return results back into DS9 in the form of text, plot, 2-D image, or 3-D image.

Command:	**3d**
Menu:	*Frame - New Frame 3D*
Description:	Loads a 3D image in a new frame. This displays a 2D projection of the data cube and pops up the *Frame - Cube* dialog, which allows you to select which slice in the 3rd dimension is being displayed.

Command:	**frame new 3d**
Menu:	-
Description:	Equivalent to the 3d command.

You can choose a slice statically with First, Previous, Next, Last, or an explicit numeric value. You can also choose Play (and Stop) to make DS9 cycle through the cube repeatedly.

Command:	**cube open|close**
Menu:	*Frame - Cube*
Description:	Open the Cube dialog window.

Command:	**cube play|stop**
Menu:	*Frame - Cube - Play|Stop*
Description:	Cause DS9 to start or stop cycling through the slices

Command:	**cube first|last**
Menu:	*Frame - Cube - First|Last*
Description:	Cause DS9 to go to the first or last slice

Command:	**cube prev|next**
Menu:	*Frame - Cube - Previous|Next*
Description:	Cause DS9 to go to the previous or next slice. KEYBOARD SHORTCUT: + (next slice); - (previous slice).

Command: **cube [value]**

Menu: *Frame - Cube - [value]*

Description: Cause DS9 to go to the selected slice

Command: **cube interval [value]**

Menu: *Frame - Cube - Interval*

Description: Allows you to choose the time interval between slices in Play mode (default is 1 s).

Command: **cube order [abc]**

Menu: *Frame - Cube - Axes Order*

Description: Allows you to select which 2 axes are used as x,y for the projection and which axis is used for the third dimension z (slice direction). Example: cube order 132

Command: **cube axis [number]**

Menu:

Description: Specify the 3rd axis to use in the case of a 4 or higher dimension axis dataset. Example: xpaset -p ds9 cube axis 3

Command: **cube match|lock wcs**

Menu: -

Description: Match or lock cube frames

The *Frame - Cube - Coordinate* menu allows control over whether image coordinates or WCS coordinates are displayed.

If you select *Edit - 3D* mode, when a 3D image is loaded, an additional dialog pops up, the 3d dialog. You can now grab the image with the left button and rotate it interactively in 3D space.

Command: **3d open**

Menu: *Frame - New Frame 3D*

Description: Opens a new 3D frame. *Frame - 3D* opens a control dialog.

Command: **3d close**

Menu: *Frame - 3D - Close*

Description: Closes the 3D frame

Command: **3d az [value]**

Menu: *3d - Azimuth*

Description: Control azimuth of the cube's orientation, deg.

Command: **3d el [value]**

Menu: *3d - Elevation*

Description: Control elevation of the cube's orientation, deg.

Command: **3d scale [value]**

Menu: *3d - Z Axis Scale*

Description: Control the Z axis scale - default 1. Allows you to stretch or shrink the scale on which the z axis is rendered.

Command: **3d method mip|aip**

Menu: *3d - Render - MIP|AIP*

Description: Optimize the 3D rendering. The MIP mode causes the maximum value along a projected ray to be displayed, while AIP mode takes the average value along the ray.

Command: **3d background none|azimuth|elevation**

Menu: *3d - Render - None|Azimuth|Elevation*

Description: The Azimuth and Elevation options cause the renderer to precalculate and cache the image seen at different orientations.

Command: **3d border yes|no**

Menu: *3d - Border - Show*

Description: Controls visibility of the cube border, outlined in blue by default.

Command: **3d highlite yes|no**

Menu: *3d - Highlight - Show*

Description: Controls visibility of the highlighting of the current slice, outlined in cyan by default.

Command: **3d border|highlite color [value]**

Menu: *3d - Border|Highlite - Color*

Description: Allows you to set the color used to outline the border or the highlighting of the current slice.

Command: **3d compass yes|no**

Menu: *3d - Compass - Show*

Description: Controls the display of an axis triad drawn at the origin, which can help the user understand the current orientation as they move the image around.

Command: **3d compass color [value]**

Menu: *3d - Compass - Color*

Description: Sets the color of the 3d compass axis triad.

Command: **3d match**

Menu: *Frame - Match - 3d*

Description: Matches the orientation of multiple 3d frames

Command: **3d lock yes|no**

Menu: *Frame - Lock - 3d*

Description: Matches and locks together the orientation of multiple 3d frames, so you can rotate one and the others follow in sync

Command: **match|lock slice [coordsys]**

Menu: *Frame - Match/Lock - Slice - WCS/Image*

Description: Locks the selected slice for multiple 3D frames.

Command: **crop 3d zmin zmax [coordsys]**

Menu: *-*

Description: Crop a 3D image to a given range in the 3rd axis. The zmin and zmax values are given in the given coordsys, e.g. 'crop 3d 0.25 0.50 wcs'

Command: **mode crop**

Menu: *Edit - Crop*

Description: For a 3D frame, there are additional mouse functions in crop mode. Shift-drag will move the front crop plane forward; control-drag will move the back crop plane backward.

Command: **-**

Menu: *[shape] - Analysis - Plot2D*

Description: Double clicking on a region brings up the region shape dialog. This region shape menu option generates a popup plot of the data within the region allowing you to scan through slices.

Command:	-
Menu:	*Region - Region Parameters - AutoPlot3D*
Description:	If this is set, the data plot is popped up automatically at region creation.

Command:	-
Menu:	*Edit - Preferences - 3d - Render - Background*
Description:	Allows you to select ’none’, ’Azimuth’, or ’Elevation’ as being in the background for the render, for improved performance

11 Profile Graphs

Command:	**view graph horizontal|vertical yes|no**
Menu:	*View - Horizontal|Vertical Graph*
Description:	If either or both of these options are set, a graph is added at the side of the main window which shows, and dynamically updates, 1D sections in x or y passing through the current x,y position.

Command:	**graph open|close**
Menu:	*Analysis - Graph Parameters*
Description:	Opens the Graph Parameters dialog.

The graph parameters menu controls aspects of the graph, and the Preferences - Graphs menu controls the defaults at startup.

Command:	**graph grid yes|no**
Menu:	*Analysis - Graph Parameters - Grid*
GUI Default:	*Edit - Preferences - Graphs - Grid*
Description:	Displays a grid

Command: **graph log yes|no**

Menu: *Analysis - Graph Parameters - Axis - Linear/Log*

GUI Default: *Edit - Preferences - Graphs - Axis - Linear/Log*

Description: Controls whether or not the y axis is logarithmic.

Command: **graph method average|sum**

Menu: *Analysis - Graph Parameters - Method - Average/Sum/Median*

GUI Default: *Edit - Preferences - Graphs - Method - Average/Sum/Median*

Description: Controls how the y value is calculated in the extraction region.

Command: **graph font [value]**

Menu: *Analysis - Graph Parameters - Font*

GUI Default: *Edit - Preferences - Graphs - Font*

Description: Controls the font for the axis numerics.

Command: **graph fontsize [value]**

Menu: *Analysis - Graph Parameters - Font - Size*

Description: Controls the font for the axis numerics.

Command: **graph fontweight [value]**

Menu: *Analysis - Graph Parameters - Font - Weight*

Description: Controls the font for the axis numerics.

Command: **graph fontslant [value]**

Menu: *Analysis - Graph Parameters - Font - Slant*

Description: Controls the font for the axis numerics.

Command: **graph size [value]**

Menu: *Analysis - Graph Parameters - Size*

GUI Default: *Edit - Preferences - Graphs - Size*

Description: controls the height (for horizontal) or width (for vertical) of the space used by the graph.

Command: **graph thickness [value]**

Menu: *Analysis - Graph Parameters - Thickness*

GUI Default: *Edit - Preferences - Graphs - Thickness*

Description: Controls the thickness in pixels of the projection section used to calculate the graph values. Default is 1.

Command: **-**

Menu: *Analysis - Graph Parameters - Apply*

Description: Applies any GUI changes made to the graph parameters menu.

12 Inspecting Data

12.1 View header

Command: **header [n]**

Menu: *File - Header*

Description: Displays a selected FITS extension header in a popup window.

Command:	**header close**
Menu:	*File - Header - File - Close*
Description:	Close the popup header window

Command:	**header save [n] [filemame]**
Menu:	*File - Header - File - Save*
Description:	Save header of FITS extension n to the given text file.

Command:	**fits header**
Menu:	-
Description:	Dumps the current FITS header to standard output.

Command:	**fits header [n]**
Menu:	-
Description:	Dumps the header for HDU n.

Command:	**fits header [n] keyword [name]**
Menu:	-
Description:	Dumps the value of the header keyword for HDU n. Example: xpaget ds9 fits header keyword OBSERVER

Command:	**fits width|height|depth|bitpix**
Menu:	-
Description:	Dump the current FITS image x,y,z dimensions and BITPIX values using xpaget. Example: xpaget ds9 fits height

.

Command: **fits size**

Menu: -

Description: Returns the x,y dimensions in image pixels. Example:
xpaget ds9 fits size
1024 1024

Command: **fits size [coordsys] [skyframe] [units]**

Menu: -

Description: Returns the image dimensions in world coordinates. Example:
xpaget ds9 fits size galactic arcmin
20.04862 19.97157
Options: wcs, wcsa...wcsz, fk4, fk5, icrs, galactic, ecliptic; degrees, arcmin, arcsec.

Command: **fits count**

Menu: -

Description: This appears to be a synonym for 'fits depth'.

Command: **iis filename**

Menu: -

Description: Return IIS (Internet Information Services) Filename. Can support an optional mosaic number, e.g. xpaget ds9 iis filename 4; xpaset -p ds9 iis filename foo.fits

12.2 View filename or version

Command: **version**

Menu: *Help - About DS9*

Description: View the current version of DS9. Example: xpaget ds9 version

Command: **help**

Menu: -

Description: Issue command line help. 'ds9 -help'. Only available on command line.

Command: **file**

Menu: -

Description: The command 'xpaget ds9 file' (or 'xpaget foo file' if DS9 was started as 'ds9 -title foo') echoes the full path of the currently loaded file to the terminal. Not available on command line. Can be used in SAMP.

12.3 Dump data

Command: **fits image|table|slice**

Menu: -

Description: Dump the current DS9 FITS data to standard output. Example: ds9 fits image   foo.fits.
Note that this command can also load data:
cat foo.fits — xpaset ds9 fits

Command: **data [coordsys] [skyframe] [x] [y] [w] [h] yes|no**

Menu: -

Description: Retrun array of data values from the image given a lower left corner and a width and height. [coordsys] and [skyframe] specify the coordinate system for x and y and implies the units for w and h (pixels or deg). The last argument says whether to return the bare values (yes) or also return the coordinates of each value (no). Used with the xpaget command; not available to command line or to xpaset.

```
xpaget ds9 data image 342 613 3 3 yes
```

Echoes 9 lines to standard output, each line gives the values of a 3 x 3 box starting at image coords (342,613).

```
xpaget ds9 data image 342 613 3 3 no
```

As above, but each line is of the form '(x,y) = value' instead of just 'value'.

```
xpaget ds9 data physical 4028 4137 3 3 no
4030,4139 = 6
4029,4139 = 9
4028,4137 = 15
4030,4137 = 9
4029,4137 = 7
4028,4138 = 17
4030,4138 = 5
4029,4138 = 12
4028,4139 = 17
xpaget ds9 data wcs fk5 15:34:57.1 +23:30:12.5 0.0002 0.0002 no
233.7379167,23.5036089 = 12
233.7379167,23.5034722 = 7
233.7377676,23.5036089 = 5
233.7377676,23.5034722 = 9
```

12.4 Print image

DS9 provides strong Postscript printing support. This is not a screen capture method, but a full level 1/2/3 postscript driver. The postscript images generated are detailed and accurate as possible, given the resolution of the data, and the printing resolution.

Postscript Level

- Level 1– The postscript generated consist of a color lookup table and image data, encoded in ASCIIHEX. All line graphics and text are postscript elements.

- Level 2– The postscript generated consist of a color lookup table and image data, compressed with RLE, and encoded in ASCIIHEX85. All line graphics and text are postscript elements.
- Level 3– The postscript generated consist of a color lookup table and image data, compressed with GZIP. All line graphics and text are postscript elements.

DS9 supports three color models for level 2/3 postscript: RGB, CMYK, Grayscale. All three color models generate approximately the same size files.

Unlike graphic elements, image data must be sampled when encoded into postscript. The specified resolution parameter is in "pixels per inch" and is independent of the current screen resolution. A value of 96 maps to the current screen resolution. Any value over 96 will over sample the image, while any value under 96 will under sample. For high resolution printers and posters, the higher resolution sampling, the better. However, keep in mind the size of the postscript file grows with the square of the increase in resolution.

Command: **print**

Menu: *File - Print*

Description: Prints the main image pane including colorbar and horizontal and vertical graphs if present. The rest of the window, including info and menu button panes, are not included. The properties of the print output can be controlled by *File - Page Setup* and *Edit - Preferences*.

Some options are:

Command: **print destination printer|file**

Menu: -

Description: Specify whether destination is to printer or a file

Command: **print command [printer]**

Menu: *Edit - Preferences - Postscript - Print To - Printer*

Description: allows the target printer to be selected. Example: print command lp -Plpm

Command: **print filename [filename]**

Menu: *Edit - Preferences - Postscript - Print To - File*

Description: allows the output filename to be selected.

Command:	**print color [value]**
Menu:	*Edit - Preferences - Postscript - Postscript - Color*
Description:	allows the user to specify rgb, cmyk, or gray.

Command:	**print level [value]**
Menu:	*Edit - Preferences - Postscript - Postscript - Level*
Description:	allows the user to specify the Postscript level 1,2,3.

Command:	**print resolution [value]**
Menu:	*Edit - Preferences - Postscript - Postscript - DPI*
Description:	allows the user to specify the resolution: ”screen”, or 72, 96, 144, 150, 225, 300, 600 or 1200 dpi

Command:	**-**
Menu:	*Edit - Preferences - Print Coordinates*
Description:	Hitting the ’c’ key brings up a dialog which gives the current cursor coordinates. This option controls which coordinate system is used.

Command:	**pagesetup orient portrait|landscape**
Menu:	*Edit - Preferences - Page Seup - Layout - Orientation - Portrait/Landscape*
Description:	allows the user to specify the page orientation.

Command:	**pagesetup scale [value]**
Menu:	*Edit - Preferences - Page Setup - Layout - Scale*
Description:	allows the user to specify the scaling.

Command: **pagesetup size [value]**

Menu: *Edit - Preferences - Page Setup - Page Size*

Description: allows the user to specify the page size (letter, legal, tabloid, poster, a4). The GUI also allows setting an explicit size in mm or in.

Command: **psprint**

Menu: -

Description: Invokes postscript printing. Same arguments as print.

Sometimes images that are saved as PS or JPG files look great on the screen, but are poor when printed out as hardcopy. This is because the files are created by default using the RGB color model, but color printers use the CMYK (cyan, yellow, magenta, black) color model. The mapping between RGB and CMYK is not bijective and information can be lost; some RGB colors are not present in the CMYK color gamut. A color of #0000FF (RGB blue) is usually mapped to an excessively purple shade in CMYK. For this reason, you may wish to use colormaps without deep blues and purples, such as BB or Heat. The DS9 authors recommend that if the exact hardcopy color is important you should use CMYK definitions and that CMYK (100,65,0,0) gives a nice clean blue.

Also, in practice, different printers and monitors may display colors differently. In principle tuning a gamma correction for a given monitor or printer could improve things, but it's a bit hit and miss. Note that printers tend to wash out blues and purples, so avoid or oversaturate these colors. Our experience is that although ApJ requestes CMYK images, submitting RGB images to them results in printed results closer to the original.

12.5 Illustrate menu and mode

Command: **illustrate**

Menu: *Edit - Illustrate*

Description: This mode makes the top level Illustrate menu available (it is normally greyed out) and disables the Region menu. The Illustrate menu is similar to the region menu, and allows you to select a region shape, then left-click in the image to create an instance of that shape; it also has the selection choices All, None, etc., as well as Save/List/Delete selection and the ordering choices (Front/Back).

The difference between the Illustrate menu and the Region menu is that the regions and text created by Illustrate do not move or resize when the image is panned or zoomed; unlike Region mode, they are not tied to the image and its coordinates.

Saving Illustrate entries creates a '.seg' file which is a valid region file; however the format is somewhat different from that saved in '.reg' (e.g. 'circle x y r' rather than 'circle(x,y,r)', and there is no option to use different coordinate systems.

Command: **illustrate save [filename]**

Menu: *Illustrate - Save*

Description: Save current illustration to a file

Command: **illustrate load [filename]**

Menu: *Illustrate - Open*

Description: Load region file to current illustration

Command: **illustrate select all|none|invert|front|back**

Menu: *Illustrate - All|None|Invert|Front|Back*

Description: Control selection of illustrate regions

Command: **illustrate move front|back**

Menu: *Illustrate - Move To Front/Move To Back*

Description: Shuffle layering of illustrate regions

Command: **illustrate delete select|all**

Menu: *Illustrate - Delete Selection/Delete All*

Description: Delete selected, or all, illustrate regions

Command: **illustrate shape [value]**

Menu: *Illustrate - Shape*

Description: Specify current illustrate region shape

Command: **illustrate color [value]**

Menu: *Illustrate - Color*

Description: Specify current illustrate region color

Command: **illustrate fill yes|no**

Menu: *Illustrate - Color - Fill*

Description: Turn on/off illustrate region fill state

Command: **illustrate width [value]**

Menu: *Illustrate - Width*

Description: Specify current illustrate region line width

Command:	**illustrate dash yes|no**
Menu:	*Illustrate - Width - Dash*
Description:	Turn on/off illustrate region line dash

Command:	**illustrate copy|cut|paste**
Menu:	-
Description:	Copy, cut, paste selected region to/from buffer

Command:	**illustrate undo**
Menu:	-
Description:	Undo last illustrate operation

Command:	**illustrate open|close**
Menu:	-
Description:	Open or close illustate save file

Command:	**illustrate command [string]**
Menu:	-
Description:	Add region to illustration. Example: illustrate command {circle 100 100 20}

12.6 Examine mode

Command:	**-**
Menu:	*Edit - Examine*
Description:	Examine mode emulates an old IRAF examine functionality. Clicking in the image generates a new frame showing a zoomed in region around the clicked location. Clicking again generates a further new frame of a further-magnified region.

Command: **-**

Menu: *Edit - Preferences - Examine - Magnification*

Description: allows you to control the magnification step

Command: **-**

Menu: *Edit - Preferences - Examine - Mode*

Description: selects whether or not a new frame is generated each time you click.

Command: **iexam**

Menu: -

Description: Examine coordinates at current location. Example: xpaget ds9 iexam; the command waits until you click the mouse button and then returns the current cursor coordinates. iexam not available for xpaset or command line.

Command: **iexam cooordinate wcs [coordsys] [frame] [units]**

Menu: -

Description: Examine coordinates at current location. Example: xpaget ds9 iexam coordinate wcs fk5 sexagesimal; xpaget ds9 iexam coordinate wcs galactic degrees

Command: **iexam data**

Menu: -

Description: Examine pixel value at current location. Example: xpaget ds9 iexam data

Command: **iexam data [n] [m]**

Menu: -

Description: Examine pixel values in n x m box around current location. Example: xpaget ds9 iexam data 3 3

Command: **iexam [string]**

Menu: -

Description: Click at location, fill in variables accordingly. Variables are $width, $height, $depth, $bitpix, $filename, $regions, $filename[$regions], $env, $pan, $value, $x, $y, $z. Example: xpaget ds9 iexam 'Location $x $y in file $filename'. JCM Note: This command appears to be no longer working.

12.7 Crosshair

When the crosshair mode is on with *Edit - Crosshair*, The crosshair position can be read off from *Analysis - Crosshair Parameters - Crosshair*. Conversely, entering coordinates in the *Analysis - Crosshair Parameters* coordinate boxes and using *Analysis - Crosshair Parameters - Apply* will move the crosshair to those coordinates.

Command: **cursor [x] [y]**

Menu: -

Description: Move crosshair and mouse cursor by given number of pixels. Selected regions will also move by the given amount.

12.8 Pixel Table

Command: **pixeltable yes|open**

Menu: *Analysis - Pixel Table*

Description: displays a window with a table showing the pixel values in the vicinity of the cursor, as a function of the image pixel number coordinates.

Command: **pixeltable no|close**

Menu: *Analysis - Pixel Table - Close*

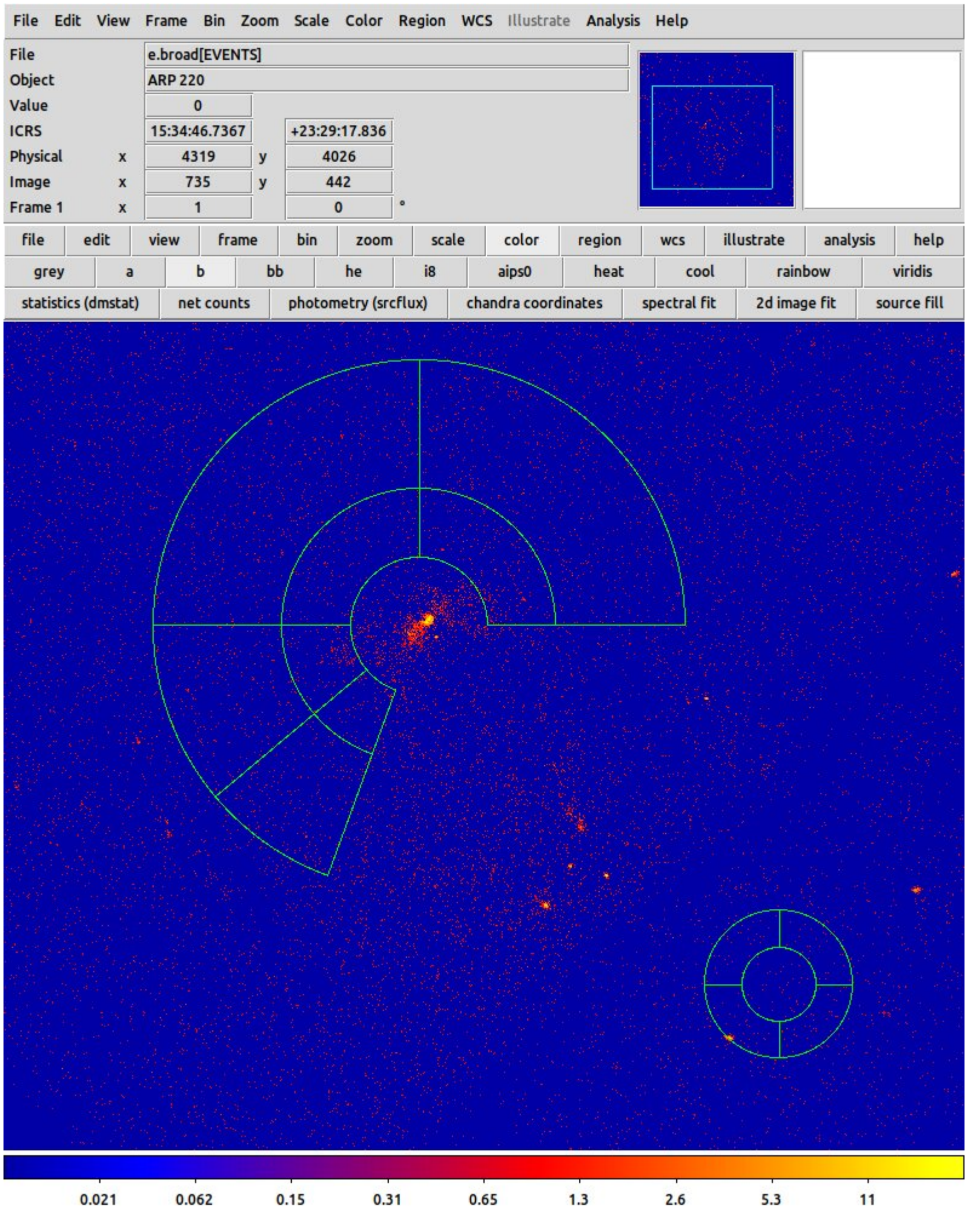
Description: Closes the pixel table window

Command:	-
Menu:	*Analysis - Pixel Table - Size*
GUI Default:	*Edit - Preferences - Pixel Table - Size*
Description:	controls the size of the vicinity shown, in pixels. The maximum available is a 13 x 13 region around the cursor, default is 5x5.

12.9 Fonts: special symbols

Entering special characters into an entry dialog used to be done by selecting a special symbol font. Such symbol fonts are no longer implemented with the latest OS font and scripting support, especially with scalable anti-alias fonts such as Xft for Linux. Most newer fonts (if not all) now have greek characters as part of the font. The greek chars start at unicode
u0391 for 'A' and
u03b1 for 'a'. Each OS has a tool used to build and copy a string of characters. Then use the Edit:Paste menu of DS9 to insert the character string.

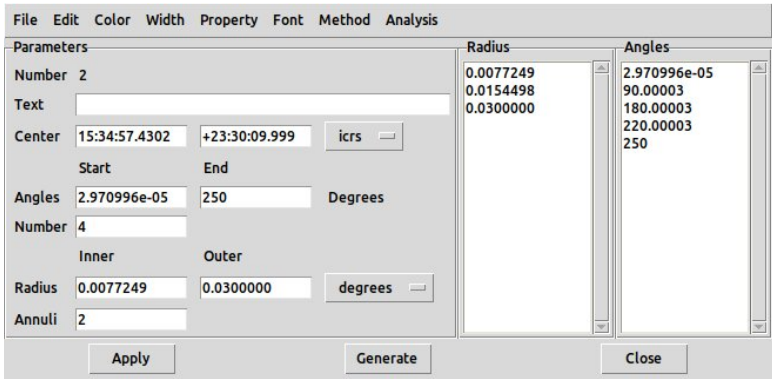
- Linux- Gnome: gnucharmap
- Linux- KDE: kcharselect
- MacOSX: Character Viewer (Select Edit:Special Characters) Now click and drag the characters to a terminal window. Then select the string and select Edit:Copy.
- Windows: Character Map (from Start button, select All Programs, Accessories, System Tools and then Character Map)



File Edit View Frame Bin Zoom Scale Color Region WCS Illustrate Analysis Help
file edit view frame bin zoom scale color region wcs illustrate analysis help
grey a b bb he i8 aips0 heat cool rainbow viridis
statistics (dmstat) net counts photometry (srcflux) chandra coordinates spectral fit 2d image fit source fill
0.021 0.062 0.15 0.31 0.65 1.3 2.6 5.3 11

File e.broad[EVENTS]
Object ARP 220
Value 0
ICRS 15:34:46.7367 +23:29:17.836
Physical x 4319 y 4026
Image x 735 y 442
Frame 1 x 1 0 °

Figure 16: Example of panda regions



Parameters

Number 2

Text

Center 15:34:57.4302 +23:30:09.999 icrs

	Start	End	
Angles	2.970996e-05	250	Degrees
Number	4		

	Inner	Outer	
Radius	0.0077249	0.0300000	degrees
Annuli	2		

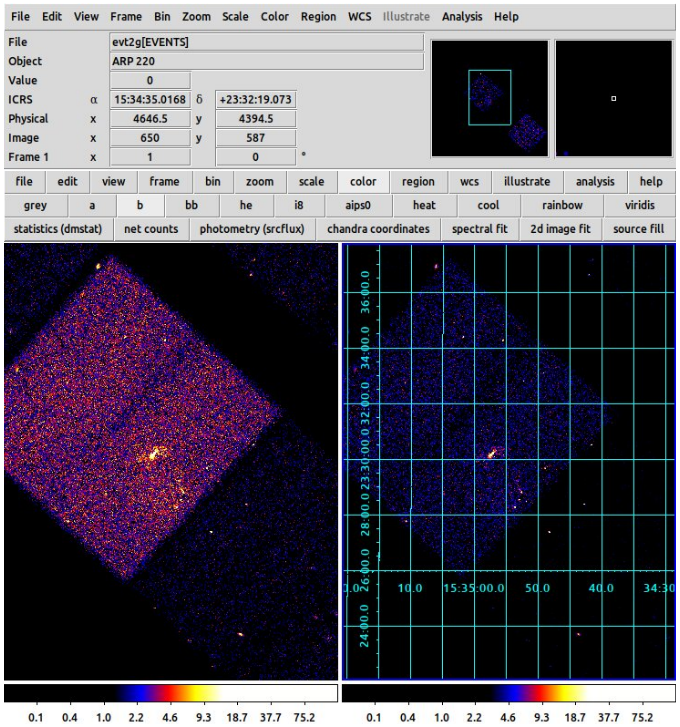
Radius

0.0077249
0.0154498
0.0300000

Angles

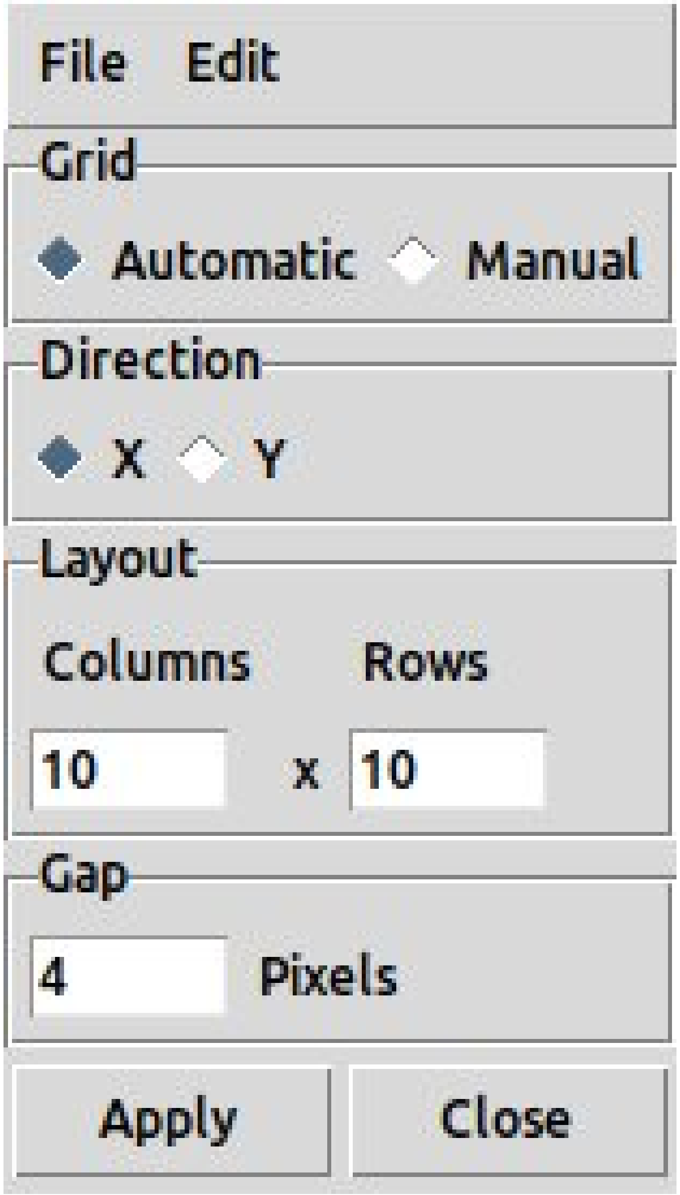
2.970996e-05
90.00003
180.00003
220.00003
250

Figure 17: Panda shape dialog



File		evt2g[EVENTS]		
Object		ARP 220		
Value		0		
ICRS	α	15:34:35.0168	δ	+23:32:19.073
Physical	x	4646.5	y	4394.5
Image	x	650	y	587
Frame 1	x	1		0 °

Figure 18: Example of two tiled frames



Grid

● Automatic ○ Manual

Direction

● X ○ Y

Layout

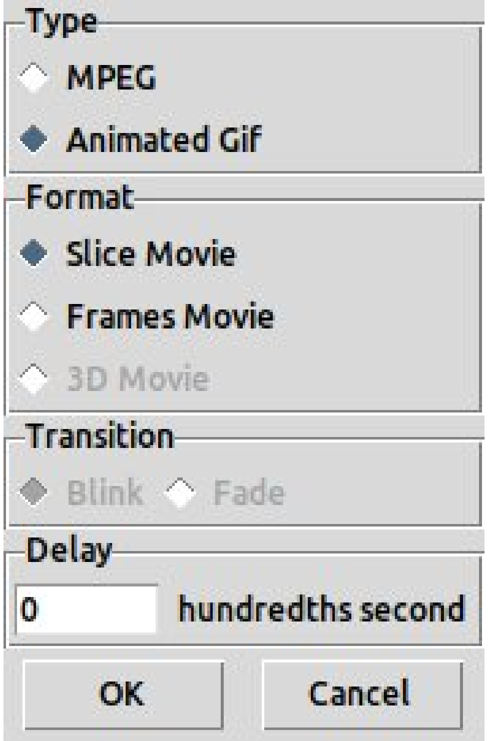
Columns Rows

10 x 10

Gap

4 Pixels

Figure 19: Tile Parameters dialog



Type

○ MPEG

● Animated Gif

Format

● Slice Movie

○ Frames Movie

○ 3D Movie

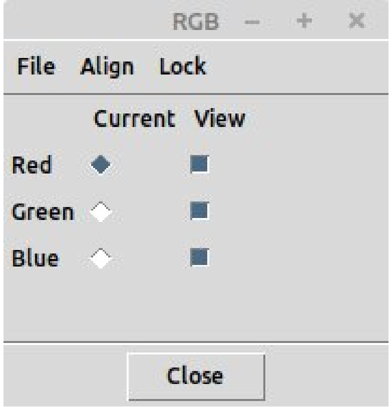
Transition

● Blink ○ Fade

Delay

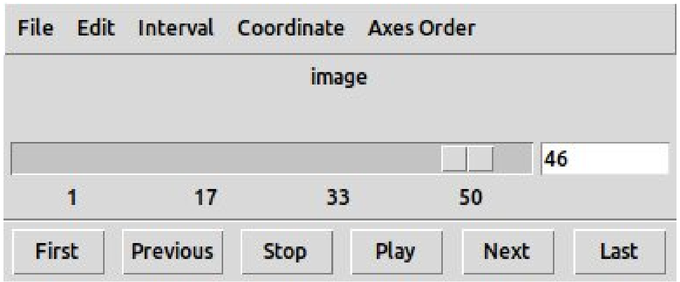
0 hundredths second

Figure 20: Create Movie dialog



	Current	View
Red	●	■
Green	○	■
Blue	○	■

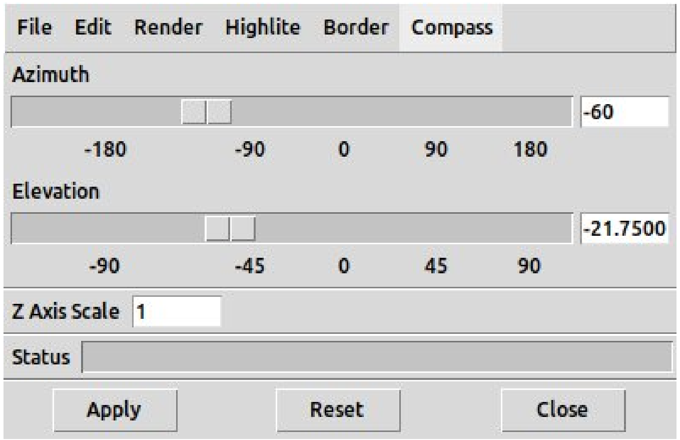
Figure 21: Popup dialog to control the red, green, blue frames



image

46

Figure 22: Cube dialog



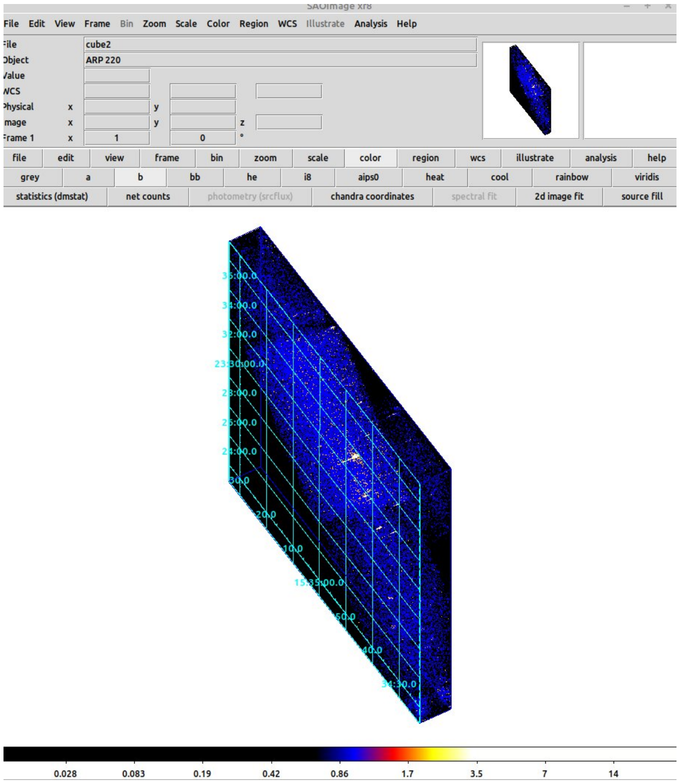
Azimuth -60

Elevation -21.7500

Z Axis Scale 1

Status

Figure 23: 3D control dialog

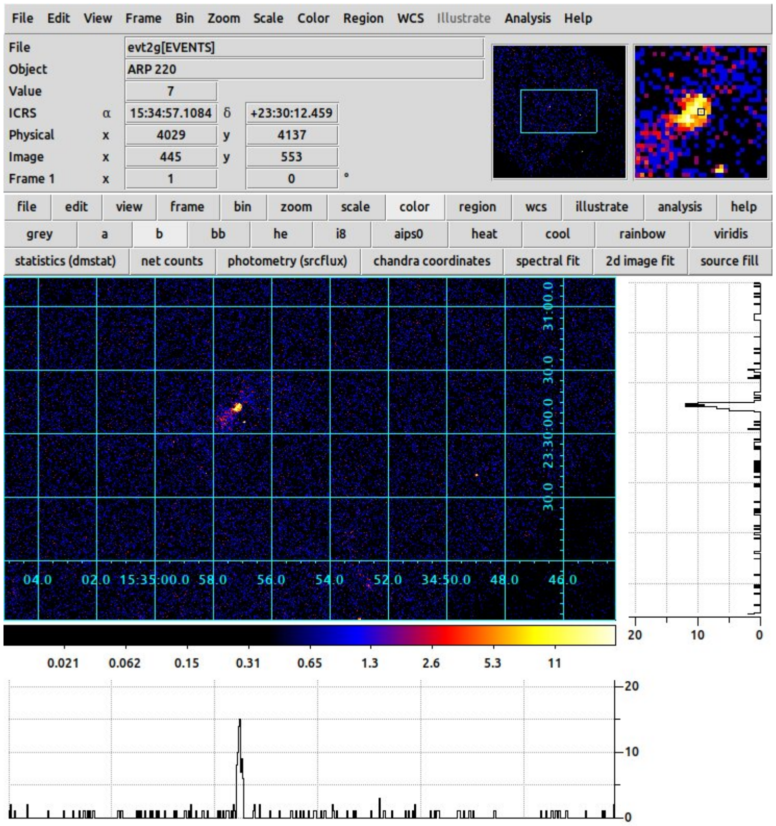

SAOImage xr8
File Edit View Frame Bin Zoom Scale Color Region WCS Illustrate Analysis Help
Value
WCS
Physical x y
Image x y z
file edit view frame bin zoom scale color region wcs illustrate analysis help
grey a b bb he i8 aips0 heat cool rainbow viridis
statistics (dmstat) net counts photometry (srcflux) chandra coordinates spectral fit 2d image fit source fill
36:00.0
34:00.0
32:00.0
23:30:00.0
28:00.0
26:00.0
24:00.0
30.0
20.0
10.0
15:35:00.0
50.0
40.0
34:30.0
0.028 0.083 0.19 0.42 0.86 1.7 3.5 7 14

File cube2

Object ARP 220

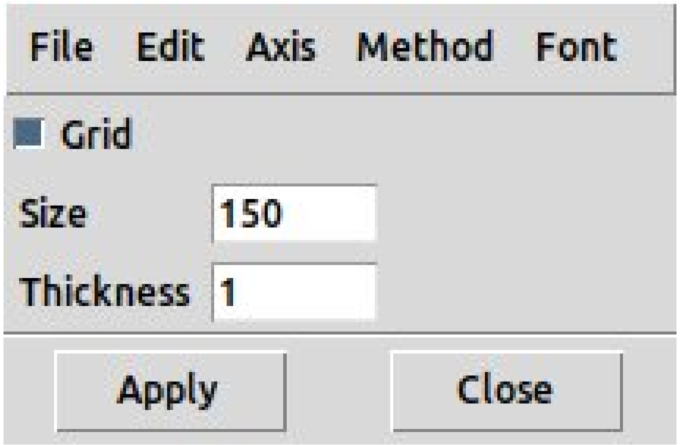
Frame 1 x 1 0 °

Figure 24: 3D frame example


File Edit View Frame Bin Zoom Scale Color Region WCS Illustrate Analysis Help
file edit view frame bin zoom scale color region wcs illustrate analysis help
grey a b bb he i8 aips0 heat cool rainbow viridis
statistics (dmstat) net counts photometry (srcflux) chandra coordinates spectral fit 2d image fit source fill
31:00.0 30.0 23:30:00.0 30.0
04.0 02.0 15:35:00.0 58.0 56.0 54.0 52.0 34:50.0 48.0 46.0
20 10 0
0.021 0.062 0.15 0.31 0.65 1.3 2.6 5.3 11
20
10
0

File	evt2g[EVENTS]			
Object	ARP 220			
Value	7			
ICRS	α	15:34:57.1084	δ	+23:30:12.459
Physical	x	4029	y	4137
Image	x	445	y	553
Frame 1	x	1		0 °

Figure 25: Horizontal and vertical graphs

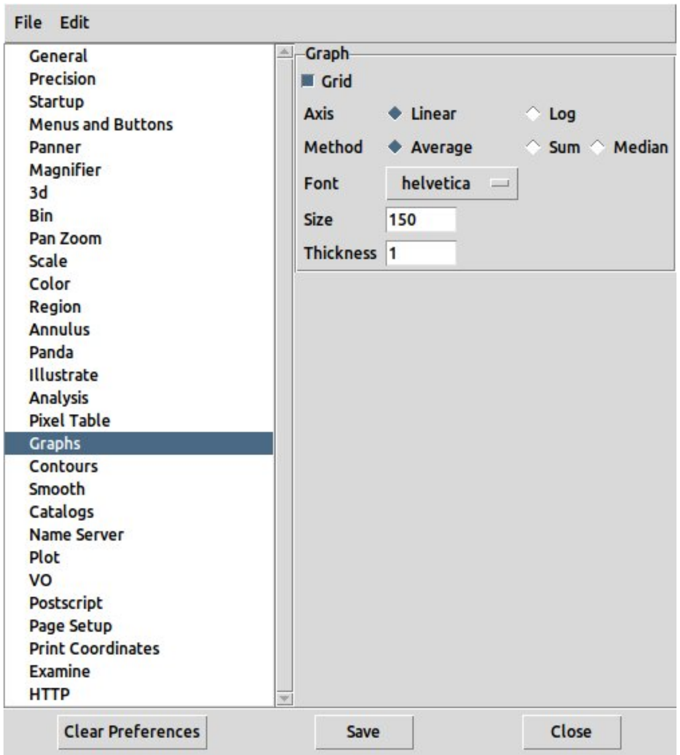


■ Grid

Size 150

Thickness 1

Figure 26: Graph Parameters dialog



Graph

■ Grid

Axis ◆ Linear ◇ Log

Method ◆ Average ◇ Sum ◇ Median

Font helvetica

Size 150

Thickness 1

Figure 27: Edit Preferences - Graphs dialog

13 Prism

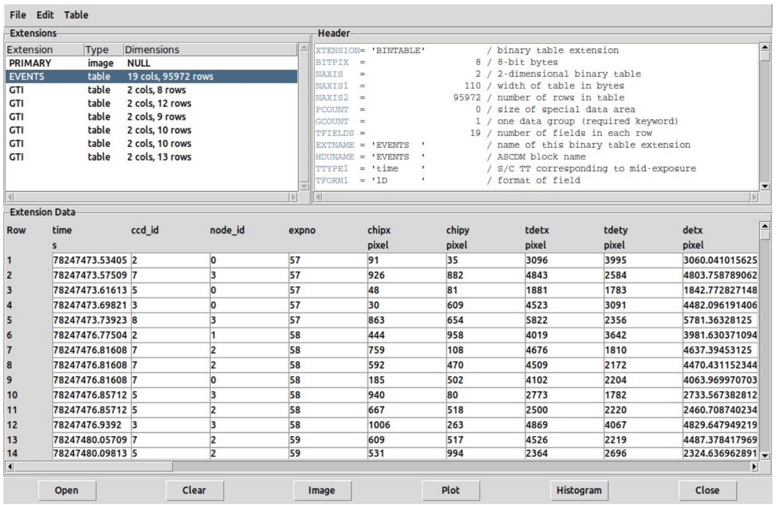
Prism is a file viewer within DS9. It was inspired by and replaces an earlier standalone program in the CIAO package.

13.1 Main Prism window

Command: **prism open**

Menu: *File - Prism*

Description: opens a Prism browser and populates it with data from the current frame. On opening, the browser has four panes - a top left 'Extensions' pane showing the structure of the FITS file; a top right pane showing the header for the currently selected extension, and a main central pane (the data pane). If the extension is an image this pane is empty, but if it is a table the pane shows the data as rows and columns. Finally a menu pane at the bottom has a set of controls.



Extension	Type	Dimensions
PRIMARY	image	NULL
EVENTS	table	19 cols, 95972 rows
GTI	table	2 cols, 8 rows
GTI	table	2 cols, 12 rows
GTI	table	2 cols, 9 rows
GTI	table	2 cols, 10 rows
GTI	table	2 cols, 10 rows
GTI	table	2 cols, 13 rows

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                  110 / width of table in bytes
NAXIS2  =                95972 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                   19 / number of fields in each row
EXTNAME = 'EVENTS  '           / name of this binary table extension
HDUNAME = 'EVENTS  '           / ASCDM block name
TTYPE1  = 'time    '           / S/C TT corresponding to mid-exposure
TFORM1  = '1D      '           / format of field
```

Row	time	ccd_id	node_id	expno	chipx	chipy	tdetx	tdety	detx
	s				pixel	pixel	pixel	pixel	pixel
1	78247473.53405	2	0	57	91	35	3096	3995	3060.041015625
2	78247473.57509	7	3	57	926	882	4843	2584	4803.758789062
3	78247473.61613	5	0	57	48	81	1881	1783	1842.772827148
4	78247473.69821	3	0	57	30	609	4523	3091	4482.096191406
5	78247473.73923	8	3	57	863	654	5822	2356	5781.36328125
6	78247476.77504	2	1	58	444	958	4019	3642	3981.630371094
7	78247476.81608	7	2	58	759	108	4676	1810	4637.39453125
8	78247476.81608	7	2	58	592	470	4509	2172	4470.431152344
9	78247476.81608	7	0	58	185	502	4102	2204	4063.969970703
10	78247476.85712	5	3	58	940	80	2773	1782	2733.567382812
11	78247476.85712	5	2	58	667	518	2500	2220	2460.708740234
12	78247476.9392	3	3	58	1006	263	4869	4067	4829.647949219
13	78247480.05709	7	2	59	609	517	4526	2219	4487.378417969
14	78247480.09813	5	2	59	531	994	2364	2696	2324.636962891

Figure 28: Prism window for an event file

Command: **prism open [filename]**

Menu: *File - Prism - Open*

Description: opens a new dataset from disk.

Command: **prism clear**

Menu: *File - Prism - Clear*

Description: clears all the Prism panes; a new dataset needs to be loaded.

Command: **prism close**

Menu: *File - Prism - Close*

Description: closes the Prism window, but not the subsequently opened associated plot windows and control panels.

Command: **prism image**

Menu: *File - Prism - Image*

Description: If you load a file into Prism (with File - Prism - Open), and it is an image (as opposed to a bintable), that does not automatically render the image into the current display fram; you have to execute this command to display the image.

Command: **prism import xml|rdb|tsv [filename]**

Menu: *File - Prism - Import - VOTable, Starbase, Tab-Separated-Value*

Description: imports tables in VOTable, Starbase, or TSV formats.

Command: **prism export xml|rdb|tsv [filename]**

Menu: *File - Prism - Export - VOTable, Starbase, Tab-Separated-Value*

Description: saves the current Prism dataset in VOTable, Starbase, or TSV formats.

Command: **prism mode newplot|newgraph|overplot**

Menu: -

Description: Controls whether the next plot is a new window or overplots the previous plot.

Command: **prism plot [xcol] [ycol] [xerr] [yerr] [opt]**

Menu: *File - Prism - Plot*

Description: Creates a scatter plot from the data, as described below. The x and y columns are specified by name, and optionally x and y error colums. The opt string specifies which quantities x, y, ex (error on x), ey (error on y) are used; possible values are xy, xyex, xyey, xyexey.

Command: **prism histogram [xcol] [nbins] [min] [max]**

Menu: *File - Prism - Histogram*

Description: Creates a histogram of the data, as described below.

The top menu options *File - Prism - Table - Plot* and *File - Prism - Table - Histogram* do the same thing as the above two commands.

Command: **prism first|last|next|previous**

Menu: *File - Prism - Table - First|Last|Next|Previous Block*

Description: For a large table, only a subset or 'block' of the data - by default, 1000 rows - is loaded into the data pane and is accessible via the X and Y scrollbars in the pane. These commands allow the user to navigate to different sections of the extension data.

Command: **prism goto [n]**

Menu: *File - Prism - Table - Goto Row*

Description: Navigate to given row of the data

Command: **prism current [text]**

Menu: -

Description: Sets the name of the current Prism set to be [text]

Command: **prism ext [n]**

Menu: *File - Prism - Extensions*

Description: Go to the specified HDU number. In the GUI, click on the desired HDU in the Extensions pane.

Command: **prism ext [extname]**

Menu: *File - Prism - Extensions*

Description: Go to the HDU with the given value of EXTNAME. In the GUI, click on the desired HDU in the Extensions pane.

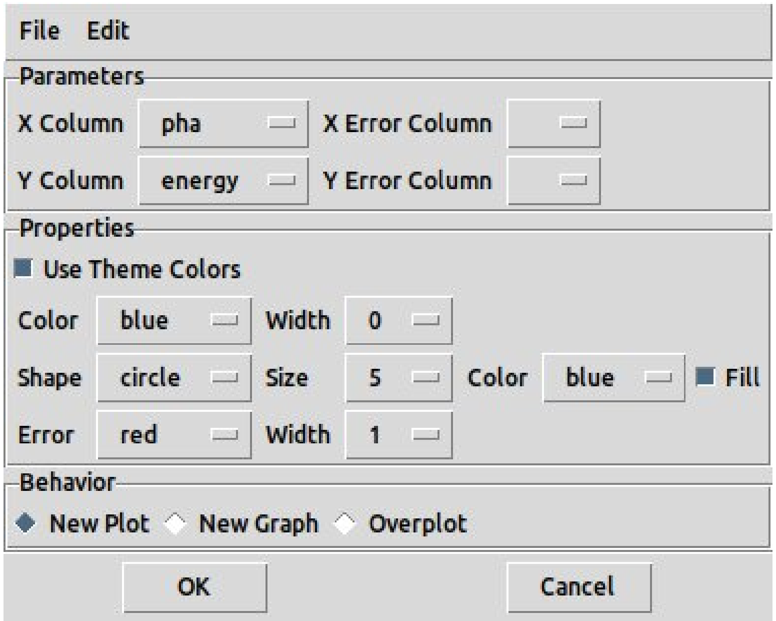
13.2 Plot Tool

The Plot Tool has two variants, Line Plot and Bar Plot. They may be launched directly from DS9 by *Analysis - Plot Tool - Line/Bar* or from Prism by *File - Prism - Plot*. In the latter case the plot tool is connected to the table columns in the current Prism data table.

Command: **plot line**

Menu: *File - Prism - Plot*

Description: launches a dialog which allows the user to make a scatter plot of one column against another.
The dialog provides the choice of X and Y columns, and optionally X and Y error columns (in the command language, described by the tokens 'xy', 'xyex', 'xyey', 'xyexey'). as well as allowing control of plot point and error bar colors and width, and plot point shape, size and color. However, the 'Use Theme Colors' box in the plot dialog must be unchecked if colors are to be changed.
The dialog also allows one to select New Plot (default, creates a new window), New Graph (adds a new plot within the existing plot window), or Overplot (on an existing plot).



Parameters

X Column: pha | X Error Column:

Y Column: energy | Y Error Column:

Properties

■ Use Theme Colors

Color: blue | Width: 0

Shape: circle | Size: 5 | Color: blue | ■ Fill

Error: red | Width: 1

Behavior

● New Plot ○ New Graph ○ Overplot

Figure 29: Prism plot dialog

Command: **plot bar**

Menu: *File - Prism - Histogram*

Description: brings up a dialog to create a histogram by binning a selected column. The dialog also permits specification of the min and max data values and the number of bins, as well as the border and fill color of the histogram bars. The user may select New Plot, New Graph or Overplot.

Once *File - Prism - Plot/Histogram - OK* is selected, the dialog disappears and the plot window *Prism Plot* (or just *Plot Tool* if started from outside Prism) is created.

Other plot commands:

Command: **plot line|bar [filename] [ref] [title] [xlabel] [ylabel] [xyexey]**

Menu: -

Description: Create a new plot from a data file, optionally with given title, xlabel, ylabel. Specify what columns are in the file with 'xy', 'xyex', 'xyey', 'xyexey'. The resulting plot has a top label of 'title' and a window title of 'ref'. The 'ref' is a title assigned by the user which can be used later to distinguish that specific plot when multiple ones are up. (see 'plot current')

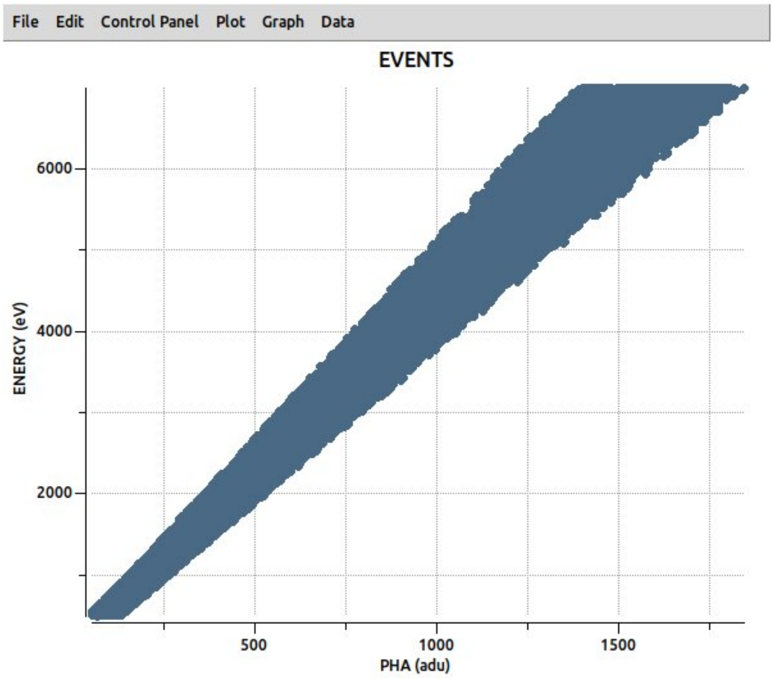


Figure 30: Prism plot of two event file columns against each other

Command:	**plot line|bar [ref] stdin**
Menu:	-
Description:	Create plot from stdin. The file streamed to stdin includes a one line header with space-separated fields of Title, X axis label, Y axis label, and the number N of data columns. Supported values of N are 2 (x and y), 3 (x, y and x error) and 4 (x, y and x and y errors). Example: 'cat stdin.2.dat — xpaset ds9 plot data xy'

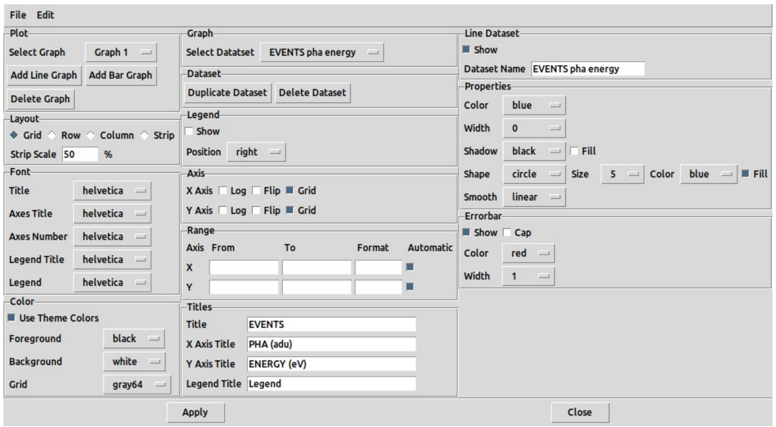
A suitable test plot file stdin.2.dat is

```
Title XAxis YAxis 2
0.00 0.153
80.00 0.021
160.00 0.008
240.00 0.007
320.00 0.007
400.00 0.006
480.00 0.006
560.00 0.006
```

```
640.00 0.005
720.00 0.005
```

Command:	**plot data [xyexey]**
Menu:	-
Description:	Load new data into existing plot from stdin. Example: 'cat data.dat — xpaset ds9 plot data xy'

Command:	**plot gui**
Menu:	*Plot Tool - Control Panel - Open*
Description:	Open the Plot Control Panel gui. Allows the user to modify the plot or graph once created. It brings up a window that allows control of plot and graph properties and dataset properties.



Plot
Select Graph: Graph 1

Layout
● Grid ○ Row ○ Column ○ Strip
Strip Scale 50 %

Font
Title: helvetica
Axes Title: helvetica
Axes Number: helvetica
Legend Title: helvetica
Legend: helvetica

Color
☑ Use Theme Colors
Foreground: black
Background: white
Grid: gray64

Graph
Select Datatset: EVENTS pha energy

Legend
☐ Show
Position: right

Axis
X Axis ☐ Log ☐ Flip ☑ Grid
Y Axis ☐ Log ☐ Flip ☑ Grid

Range

Axis	From	To	Format	Automatic
X				☑
Y				☑

Titles
Title: EVENTS
X Axis Title: PHA (adu)
Y Axis Title: ENERGY (eV)
Legend Title: Legend

Line Dataset
☑ Show
Dataset Name: EVENTS pha energy

Properties
Color: blue
Width: 0
Shadow: black ☐ Fill
Shape: circle Size: 5 Color: blue ☑ Fill
Smooth: linear

Errorbar
☑ Show ☐ Cap
Color: red
Width: 1

Figure 31: Plot control panel dialog

Command:	**plot close**
Menu:	*Plot Tool - close*
Description:	Close the Plot Tool

The Plot Tool window has additional menus as follows:

Command: **plot backup|restore [filename]**

Menu: *Plot Tool - Backup/Restore*

Description: saves the plot to a .plb script file, or restores the plot from such a file. The save procedure also creates a .plb.dir subdirectory containing the data and .plt format plot commands.

Command: **plot save [filename]**

Menu: *Plot Tool - Save Data*

Description: dumps the x,y values to a text file with one line per x,y pair.

Command: **plot list yes|no**

Menu: *Plot Tool - List Data*

Description: lists the x,y values in a new window with one line per x,y pair. The window has its own menu allowing you to print it, save it to a file, or change the font of the output values.

Command: **plot load [filename] [xyexey]**

Menu: *Plot Tool - Load Data*

Description: allows you to load new data to be plotted from a text file with one line per x,y pair.Example: 'xpaset ds9 plot load foo.dat xy'

Command: **plot export eps|gif|tiff|jpeg|png [filename]**

Menu: *Plot Tool - Export - EPS/GIF/TIFF/JPEG/PNG*

Description: saves the plot to a file in the selected image format. The jpeg and tiff cases have options: export jpeg [filename] [qualvalue], export tiff [filename] none/jpeg/packbits/deflate

Command: **plot mode pointer|zoom**

Menu: *Plot Tool - Edit - Pointer/Zoom*

Description: In zoom mode, left button click-and-drag defines a rectangle to which the plot zooms in on when the button is released. Right button zooms out.
In pointer mode, if the plot was created from a catalog (see Catalog Tool etc.) clicking on a data point in the plot will highlight the corresponding row in the catalog table.

Command: **plot show yes|no**

Menu: *Plot Tool - Data - Show*

Description: controls whether the data points are made visible or not.

Command: **plot name [string]**

Menu: *Plot Tool - Data - Name*

Description: brings up a dialog box where the user can edit the plot dataset name, which is used in the plot legend.

Command: **plot current [ref]**

Menu: -

Description: Make the named plot be the current plot. If multiple plots are up, you can refer to specific ones with a tag. Examples: xpaget ds9 plot current; xpaset -p ds9 plot current plot1prism. One can also use the commands 'plot current graph' and 'plot current dataset'.

Command: **plot line color|width [value]**

Menu: *Plot Tool - Data - Color/Width*

GUI Default: *Edit - Preferences - Plot - Dataset - Line - Color/Width*

Description: Allow the color and the width of the plotted line to be altered.
In order to change the colors, *Plot Tool - Plot - Use Theme Colors* must be deselected first.

Command: **plot line dash yes|no**

Menu: *Plot Tool - Data - Width - Dash*

Description: Control whether plot lines are dashed

Command: **-**

Menu: *Plot Tool - Plot - Use Theme Colors*

GUI Default: *Edit - Preferences - Plot - Use Theme Colors*

Description: Enforces use of theme colors. Deselect to allow controlling the color explicitly.

Command: **plot line shape symbol [value]**

Menu: *Plot Tool - Data - Shape - Shape*

GUI Default: *Edit - Preferences - Plot - Dataset - Shape*

Description: provides a submenu where different point shapes may be selected: none, circle, square, diamond, plus, cross, splus ('simple plus'), scross ('simple cross'), triangle or arrow. Make sure the shape is not 'none' if you want to see the points and not just the line.

Command: **plot line shape size [value]**

Menu: *Plot Tool - Data - Shape - Size*

GUI Default: *Edit - Preferences - Plot - Dataset - Shape - Size*

Description: allows the size of the data points to be altered.

Command: **plot line shape color [value]**

Menu: *Plot Tool - Data - Shape - Color*

GUI Default: *Edit - Preferences - Plot - Dataset - Shape - Color*

Description: allows the color of the data points to be altered.

Command: **plot line shape fill yes|no**

Menu: *Plot Tool - Data - Shape - Fill*

GUI Default: *Edit - Preferences - Plot - Dataset - Shape - Fill*

Description: controls whether the data points are drawn filled or open.

Command: **plot line fill yes|no**

Menu: *Plot Tool - Data - Shadow - Fill*

GUI Default: *Edit - Preferences - Plot - Dataset - Shadow - Fill*

Description: Fill a solid color (shadow) axsbetween the line and the X axis

Command: **plot line fill color [value]**

Menu: *Plot Tool - Data - Shadow - Color*

GUI Default: *Edit - Preferences - Plot - Dataset - Shadow*

Description: provides a submenu where shadow color can be selected.

Command: **plot line smooth [value]**

Menu: *Plot Tool - Data - Smooth*

Description: has options step, linear, cubic, quadratic, catrom (Catmull-Rom spline) to control data smoothing. This determines how you draw the line going through the data points. Default is linear, but step may be more appropriate in some cases.

Command: **plot error yes|no**

Menu: *Plot Tool - Data - Errorbar - Show*

GUI Default: *Edit - Preferences - Plot - Errorbar - Show*

Description: controls whether errorbars are drawn.

Command: **plot error cap yes|no**

Menu: *Plot Tool - Data - Errorbar - Cap*

GUI Default: *Edit - Preferences - Plot - Errorbar - Cap*

Description: controls whether errorbars are drawn with caps (serifs) or not.

Command: **plot error color [value]**

Menu: *Plot Tool - Data - Errorbar - Color*

GUI Default: *Edit - Preferences - Plot - Errorbar - Color*

Description: allows you to select errobar color.

Command: **plot error width [value]**

Menu: *Plot Tool - Data - Errorbar - Width*

GUI Default: *Edit - Preferences - Plot - Errorbar - Width*

Description: allows you to select errobar width.

Command: **plot add graph line|bar**

Menu: *Plot Tool - Add Graph - Line/Bar*

Description: adds a new graph to the plot, of either type Line or Bar.

Command: **plot current graph [label]**

Menu: *Plot Tool - Select Graph*

Description: changes the currently selected graph, when multiple ones have been created.

Command: **plot delete graph**

Menu: *Plot Tool - Delete Graph*

Description: deletes the current graph.

Command: **plot axis x|y log yes|no**

Menu: *Plot Tool - Graph - X Axis/Y Axis - Log*

GUI Default: *Edit - Preferences - Plot - Graph - X Axis/Y Axis - Log*

Description: controls whether the axis is or is not logarthmic, for the current graph.

Command: **plot axis x|y auto yes|no**

Menu: *Plot Tool - Graph - Axes Range - X/Y - Automatic*

Description: Whether the X and Y axis ranges are autoscaled

Command: **plot axis x|y min|max [value]**

Menu: *Plot Tool - Graph - Axes Range - X/Y - From|To*

Description: Explicitly set the X and Y axis ranges

Command: **plot axis x|y format [value]**

Menu: *Plot Tool - Graph - Axes Range - X/Y - Format*

Description: Set the C format for displaying the axis numerics

Command: **plot axis x|y flip yes|no**

Menu: *Plot Tool - Graph - X Axis/Y Axis - Flip*

GUI Default: *Edit - Preferences - Plot - Graph - X Axis/Y Axis - Flip*

Description: controls whether the axis is flipped or not.

Command: **plot axis x|y grid yes|no**

Menu: *Plot Tool - Graph - X Axis/Y Axis - Grid*

GUI Default: *Edit - Preferences - Plot - Graph - X Axis/Y Axis - Grid*

Description: controls whether grid lines are plotted along the X or Y axes.

Command: **-**

Menu: *Plot Tool - Plot - Grid*

GUI Default: *Edit - Preferences - Plot - Grid*

Description: selects the color of the grid superimposed on the plot.

Command: **plot layout grid|row|column|strip**

Menu: *Plot Tool - Plot - Layout - Grid, Row, Column, Strip*

GUI Default: *Edit - Preferences - Plot - Layout*

Description: defines how the multiple graphs are laid out in the plot window.

Command: **plot layout strip scale [value]**

Menu: *Plot Tool - Plot - Layout - Strip Parameters*

Description: allows a percentage value to control spacing of how the strips are plotted.

Command: **plot legend yes|no**

Menu: *Plot Tool - Graph - Legend - Show*

GUI Default: *Edit - Preferences - Plot - Graph - Show Legend*

Description: adds a legend to the current graph showing the different datasets and the colors used for them.

Command: **plot legend position right|left|top|bottom|plotarea**

Menu: *Plot Tool - Graph - Legend - Right, Left, Top, Bottom, Plot Area*

GUI Default: *Edit - Preferences - Plot - Graph - Legend Position*

Description: specifies where the legend is placed relative to the graph.

Command: **plot font title font|size|weight|slant [value]**

Menu: *Plot Tool - Plot - Title*

GUI Default: *Edit - Preferences - Plot - Title*

Description: controls the font, size, normal/bold state, and italic state of the plot title.

Command: **plot font labels font|size|weight|slant [value]**

Menu: *Plot Tool - Plot - Axes Title*

GUI Default: *Edit - Preferences - Plot - Axis Title*

Description: Controls axis title font info

Command: **plot font axes font|size|weight|slant [values]**

Menu: *Plot Tool - Plot - Axes Number*

GUI Default: *Edit - Preferences - Plot - Axis Numbers*

Description: Controls axis numbers font info

Command: **-**

Menu: *Plot Tool - Plot - Legend Title*

GUI Default: *Edit - Preferences - Plot - Legend Title*

Description: Controls legend title font info

Command: **-**

Menu: *Plot Tool - Plot - Legend*

GUI Default: *Edit - Preferences - Plot - Legend*

Description: Controls legend font info

Command: **plot title [string]**

Menu: *Plot Tool - Graph - Titles - Title*

Description: allows the user to override the default values used for the text in the plot title

---

Command: **plot title x|y [string]**

Menu: *Plot Tool - Graph - Titles - X|Y Axis Title*

Description: allows the user to override the default values used for the text in the X axis and Y axis labels

---

Command: **-**

Menu: *Plot Tool - Graph - Titles - Legend Title*

Description: allows the user to override the default values used for the text in the legend title (default: ”Legend”).

Command: **-**

Menu: *Plot Tool - Plot - Foreground*

GUI Default: *Edit - Preferences - Plot - Foreground*

Description: selects the color of all of Title, Axes Title, Axes Number.

Command: **plot background [color]**

Menu: *Plot Tool - Plot - Background*

GUI Default: *Edit - Preferences - Plot - Background*

Description: selects the color of the background of the plot window.

Command: **plot current dataset [tag]**

Menu: *Plot Tool - Graph - Dataset*

Description: selects the current dataset from those already loaded.

Command: **plot duplicate**

Menu: *Plot Tool - Graph - Duplicate Dataset*

Description: makes a copy of the current dataset and adds it to the dataset list.

Command: **plot delete dataset**

Menu: *Plot Tool - Graph - Delete Dataset*

Description: deletes the current dataset.

Command: **-**

Menu: *Plot Tool - Data - Show*

Description: may be unselected to make the data points invisible.

Command: **plot stats yes|no**

Menu: *Plot Tool - Statistics*

Description: brings up a window listing the min, max, mean, median, variance and standard deviation of the Y values from the plot. The window has its own menu allowing you to print it, or change the font of the output values.

Command: **plot bar color [value]**

Menu: *Edit - Preferences - Plot - Bar Dataset - Color*

Description: sets the bar color for a bar plot

Command: **plot bar width [value]**

Menu: *Edit - Preferences - Plot - Bar Dataset - Width*

Description: sets the bar width for a bar plot

Command: **plot bar fill yes|no**

Menu: *Edit - Preferences - Plot - Bar Dataset - Fill*

Description: sets/unsets the bar fill flag.

Command: **plot bar border color [value]**

Menu: *Edit - Preferences - Plot - Bar Dataset - Border*

Description: sets the border color for a bar plot

Command: **plot bar border width[color]**

Menu: *Edit - Preferences - Plot - Bar Dataset - Border - Width*

Description: sets the border width for a bar plot

Command: **plot print**

Menu: *Plot Tool - File - Print*

Description: Print the plot

Command: **plot print destination printer|file**

Menu: *Plot Tool - File - Printer|File*

Description: Select destination for plot print

Command: **plot print [filename]**

Menu: *Plot Tool - File - Print - Name*

Description: Select output filename for plot

Command: **plot print command [text]**

Menu: *Plot Tool - File - Print - Command*

Description: Print command (e.g. 'lp -Pfoo'

Command: **plot print color rgb|gray**

Menu: *Plot Tool - File - Print - Postscript*

Description: Select color or gray for print

Command: **plot pagesetup orient|scale|size**

Menu: *Plot Tool - File - Page Setup*

Description: The pagesetup commands are also available within in plot

14 Masks

A mask is an image used as a 1-bit overlay on the main image.

Note that there are two different dialogs with the same title "Mask Parameters" - *Analysis - Mask Parameters* and *Analysis - Mask Parameters - File - Open - OK - Mask Parameters*.

Command: **mask open**

Menu: *Analysis - Mask Parameters*

Description: Opens the Analysis - Mask Parameters dialog

Command: **mask load [filename]**

Menu: *Analysis - Mask Parameters - File - Open*

Description: Allows the user to open an image as a 'mask'. The image is displayed in a separate graphics layer overlay on the main image with a chosen degree of transparency.

Analysis - Mask Parameters - File - Open As - Mosaic (IRAF, WCS, WFPC2) are also supported for different input types.

The mask is displayed superimposed on the data.

Command: **mask transparency [value]**

Menu: *Analysis - Mask Parameters - Transparency*

Description: Value is 0 to 100. The GUI has an adjustable slider bar to adjust the transparency.

Command: **-**

Menu: *Analysis - Mask Parameters - File - Apply*

Description: Applies the changes to the mask parameters

Command: **mask system [coordsys]**

Menu: *Analysis - Mask Parameters - Align*

Description: Provides options to align the mask image with the main image based on WCS or other coordinates. Example:
mask system physical

Command: **mask color [value]**

Menu: *Analysis - Mask Parameters - File - Open - OK - Mask Parameters - Color*

Description: Set the mask color. Example: mask color red.

Command: **mask mark [option]**

Menu: *Analysis - Mask Parameters - File - Open - OK - Mask Parameters - Block Pixel*

Description: Display mask pixels if their values are consistent with the selected option. Options are nonzero, zero, nan, nonnan, range [x] [y]. For the latter the command syntax is 'mask range [x] [y]' (no 'mark')

Command: **mask blend source|screen|darken|lighten**

Menu: *Analysis - Mask Paremeters - Blend*

Description: Provides options (Source, Screen, Darken, Lighten) to blend the main and mask images.

Command: **mask clear**

Menu: *Analysis - Mask Parameters - File - Clear*

Description: Clears the current mask.

Command: **mask close**

Menu: *Analysis - Mask Parameters - File - Close*

Description: Closes the current mask.

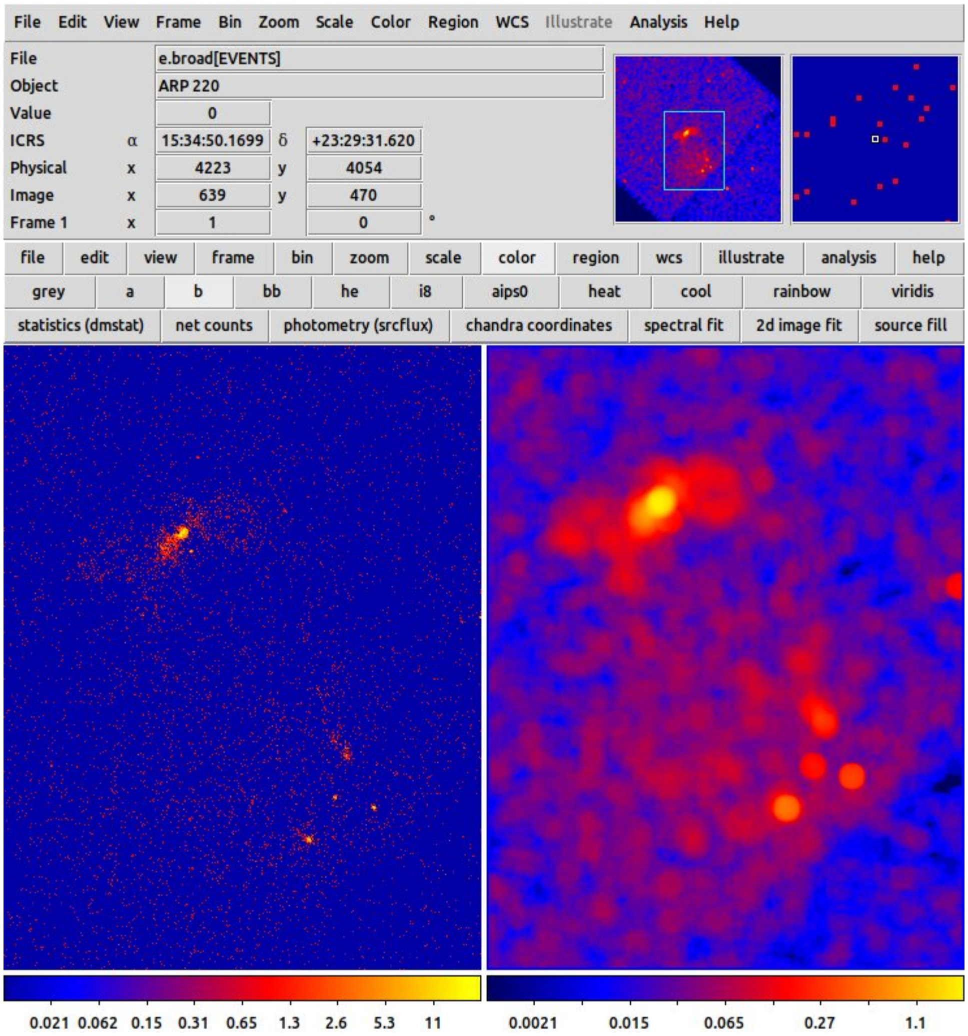
15 Smoothing and contouring

15.1 Smoothing

DS9's smoothing algorithms use a kernel with two parameters, radius r and kernel size k. The kernel width is 2k+1, and r must be less than or equal to k.

Kernels supported are the boxcar (width 2r+1), tophat (width 2r+1) and gaussian with $\sigma = r/2$,

$$f(x,y) = \frac{1}{2\pi sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$


File Edit View Frame Bin Zoom Scale Color Region WCS Illustrate Analysis Help
file edit view frame bin zoom scale color region wcs illustrate analysis help
grey a b bb he i8 aips0 heat cool rainbow viridis
statistics (dmstat) net counts photometry (srcflux) chandra coordinates spectral fit 2d image fit source fill
0.021 0.062 0.15 0.31 0.65 1.3 2.6 5.3 11
0.0021 0.015 0.065 0.27 1.1

File e.broad[EVENTS]
Object ARP 220
Value 0
ICRS α 15:34:50.1699 δ +23:29:31.620
Physical x 4223 y 4054
Image x 639 y 470
Frame 1 x 1 0 °

Figure 32: Tiled frames showing unsmoothed and smoothed data

Command: **smooth yes|no**

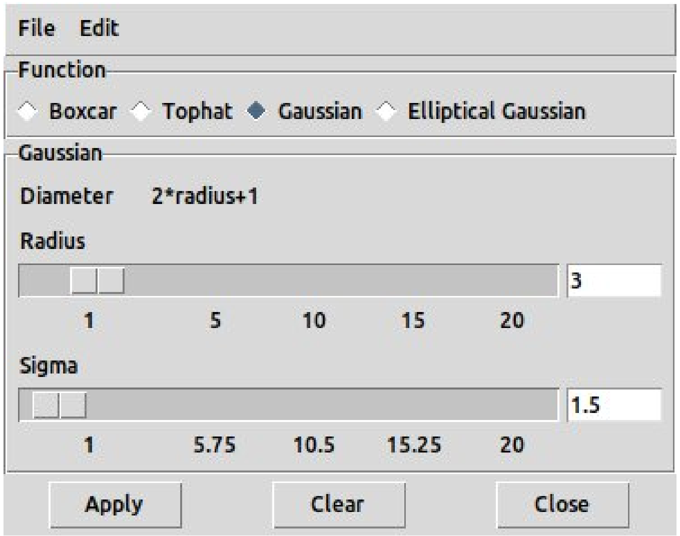
Menu: *Analysis - Smooth*

Description: Smooths the data. Deselecting this removes the smoothing.

Command: **smooth open**

Menu: *Analysis - Smooth Parameters*

Description: Open Smooth Parameters dialog


Function

◇ Boxcar ◇ Tophat ◆ Gaussian ◇ Elliptical Gaussian

Gaussian

Diameter 2*radius+1

Radius 3

Sigma 1.5

Figure 33: Smooth Parameters dialog

Command: **smooth close**

Menu: *Analysis - Smooth Parameters - Close*

Description: Close the dialog

Command: **-**

Menu: *Analysis - Smooth Parameters - Apply*

Description: smooths the data, according to parameters which depend on the smoothing function type.

Command: **smooth function boxcar|tophat|gaussian**

Menu: *Analysis - Smooth Parameters - Function*

GUI Default: *Edit - Preferences - Smooth - Function*

Description: can be Boxcar, Tophat, Gaussian or Elliptical Gaussian.

Command: **smooth radius [value]**

Menu: *Analysis - Smooth Parameters - Boxcar - Radius*

Description: define the diameter (2r+1) of the square boxcar smoothing kernel.

Command: **smooth radius [value]**

Menu: *Analysis - Smooth Parameters - Tophat - Radius*

Description: define the diameter (2r+1) of the circular tophat smoothing kernel.

Command: **-**

Menu: *Analysis - Smooth Parameters - Gaussian - Radius, Sigma*

Description: define the diameter (2r+1) of the region smoothed, and the sigma of the gaussian to use.

Command: **smooth angle|radius|radiusminor|sigma|sigmaminor [value]**

Menu: *Analysis - Smooth Parameters - Elliptical Gaussian - Angle, Major Radius, Minor Radios, Ma*

Description: define the equivalent parameters for an elliptical gaussian, including the angle of the major axis to the X axis.

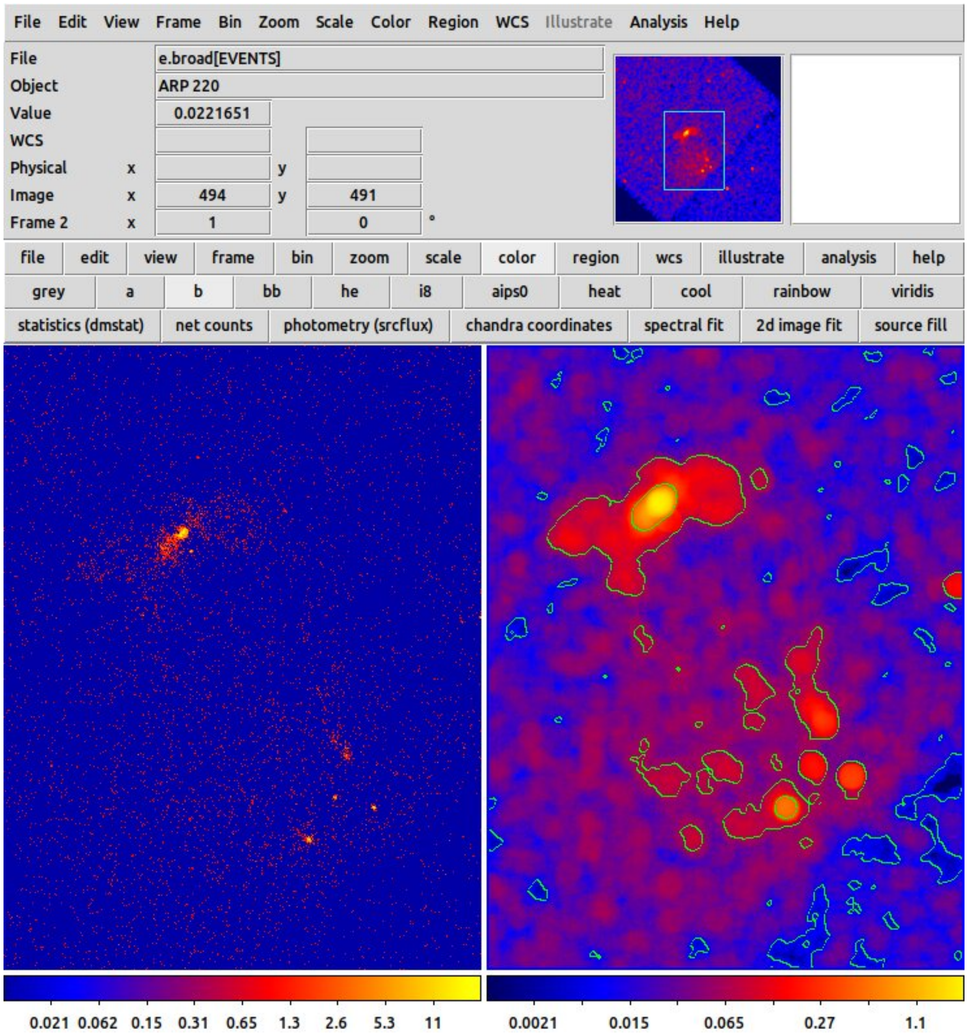
Command: **smooth no**

Menu: *Analysis - Smooth Parameters - Clear*

Description: removes the smoothing

15.2 Contours

DS9 can create and display contours as an overlay on an image. The Analysis Contours menu is used to display contours. To create, copy, paste, and configure contours, use the Analysis Contour Parameters menu.



File e.broad[EVENTS]

Object ARP 220

Value 0.0221651

WCS

Physical x y

Image x 494 y 491

Frame 2 x 1 0 °

File Edit View Frame Bin Zoom Scale Color Region WCS Illustrate Analysis Help
file edit view frame bin zoom scale color region wcs illustrate analysis help
grey a b bb he i8 aips0 heat cool rainbow viridis
statistics (dmstat) net counts photometry (srcflux) chandra coordinates spectral fit 2d image fit source fill
0.021 0.062 0.15 0.31 0.65 1.3 2.6 5.3 11
0.0021 0.015 0.065 0.27 1.1

Figure 34: Contouring applied to smoothed data from earlier example

DS9's contour algorithm was taken from HEASOFT FV; the author is unknown.

Command: **contour yes|no**

Menu: *Analysis - Contours*

Description: Draws contours on the main image.

Command: **contour open|close**

Menu: *Analysis - Contour Parameters*

Description: Control the contour generation from the contour parameters dialog. When creating a new contour, a dialog box appears, in which the user selects the number of contour levels, smoothness, and the distribution of the contours.

Appearance of the contours is controlled with

Command: **contour color [value]**

Menu: *Analysis - Contour Parameters - Color*

GUI Default: *Edit - Preferences - Contours - Color*

Description: Controls the color of the contour lines. The menu version requires Apply.

Command: **contour width [value]**

Menu: *Analysis - Contour Parameters - Width*

GUI Default: *Edit - Preferences - Contours - Width*

Description: Controls the width of the contour lines. The menu version requires Apply

Command: **contour dash yes|no**

Menu: *Analysis - Contour Parameters -Dash*

Description: Controls whether the contour lines are dashed. The menu version requires Apply.

Command: -

Menu: *Analysis - Contour Parameters - Apply*

Description: Used to actually apply the changes. specified in *Analysis - Contour Parameters*.

Generation of the contours is affected by

Command: **contour smooth [value]**

Menu: *Analysis - Contour Parameters - Contour - Smoothness*

Description: Determines the density of sampling (each pixel, each 2nd pixel, etc) for contour evaluation.

Command: **contour nlevels [value]**

Menu: *Analysis - Contour Parameters - Contour - Levels*

Description: Defines the number of contour levels. A number from 1 to 10 is recommended; larger numbers will take longer to generate.

Command: **contour scale [function]**

Menu: *Analysis - Contour Parameters - Scale*

Description: Controls the spacing of the contour levels with the same options (linear, log, pow, squared, srt, asinh, sinh, histequ) as for the pixel intensity scaling.

Command: **contour log exp [value]**

Menu: *Analysis - Contour Parameters - Scale - Log Exponent*

Description: Controls the log exponent parameter in the contour scaling functions.

Command: **contour limits [min] [max]**

Menu: *Analysis - Contour Parameters - Limits*

Description: Controls the range of the contour levels with the same options (Min Max, ZScale, 99%, etc.) as for the pixel intensity scaling. By default the range of the data is used.

Command: **contour mode minmax|zscale|zmax|[value]**

Menu: *Analysis - Contour Parameters - Limits -ZScale*

Description: Controls the range of the contour levels with the same options (ZScale, 99%, etc.) as for the pixel intensity scaling.

Command: **contour method smooth|block**

Menu: *Analysis - Contour Parameters - Method - Smooth/Block*

GUI Default: *Edit - Preferences - Contours - Method*

Description: Chooses whether the smoothing (default) or blocking method is used to generate the contours. Block blocks down the image by the smoothness factor before calculating the contours; Smooth smooths the image with a gaussian before calculating the contours. 'block' mode is faster as the smoothing parameter increases. Conversely, 'smooth' mode is much slower as the smoothing parameter increases.

Command: **contour scope global|local**

Menu: *Analysis - Contour Parameters - Scope - Global/Local*

Description: Determines whether the range limit algorithm is applied to all the data in memory or just the displayed region, as for the pixel intensity scaling.

Command: **contour generate**

Menu: *Analysis - Contour Parameters - Generate*

Description: Recalculate the contour levels with the current parameters

Command:	**contour levels [x1 x2 x3 ...]**
Menu:	*Analysis - Contour Parameters - Levels*
Description:	Directly enter or edit the numerical contour levels. In the GUI, follow this with Apply (not Generate).

In the GUI, these generation options must be followed by

Command:	**-**
Menu:	*Analysis - Contour Parameters - Apply*
Description:	Calculate the contours and actually apply the changes.

The following two options are used in conjunction with *Frame - Go to Frame* and *Frame - Match Frames - WCS* to copy contours from one frame to another - for example, to put X-ray contours on top of an optical image:

Command:	**-**
Menu:	*Analysis - Contour Parameters - File - Copy Contours*
Description:	Puts a copy of the contour data from the current frame in a buffer.

Command:	**-**
Menu:	*Analysis - Contour Parameters - File - Paste Contours*
Description:	Takes the buffered contour data and applies it to the current frame.

An example case:

- *File - Open* loads an event file in Frame 1; adjust the scale and color etc.
- *Analysis - Contour Parameters - Apply* to generate the contours, after adjusting Contour Parameters appropriately
- *Analysis - Imager Servers - SAO-DSS - Retrieve* to get a matching optical image, which opens in Frame 2. Adjust scale and color as desired.
- *Frame - Go to Frame 1* to select the X-ray frame.
- *Frame - Match Frames - WCS* to align the optical frame to the X-ray one.

- *Analysis - Contour Parameters - File - Copy Contours* to select the contours from the X-ray image
- *Frame - Go to Frame 2* to select the optical frame.
- *Analysis - Contour Parameters - File - Paste Contours* to overlay the X-ray contours on the optical image.

Command: **contour clear**

Menu: *Analysis - Contour Parameters - Clear*

Description: Will clear and reset any pasted contours in the current frame.

Command: **contour save levels [filename]**

Menu: *Analysis - Contour Parameters - File - Save Contour Levels*

Description: Writes a .lev file contaning the contour levels (not the contours themselves).

Command: **contour load levels [filename]**

Menu: *Analysis - Contour Parameters - File - Load Contour Levels*

Description: Reads a .lev file in and sets the contour levels accordingly (you still need a Generate command to remake the contours).

15.2.1 Contour files

Command: **contour save [filename] [coordsys] [skyframe]**

Menu: *Analysis - Contour Parameters - File - Save*

Description: Writes a .ctr file contaning the contours (the contours themselves, not just the levels).
A contour file is an ASCII file containing a header, global properties, coordinate system, contour levels, and contour points.
The Save dialog has an option to change the coordinate system of the saved contours. Example: 'contour save ds9.ctr wcs fk5'.

Command:	**contour load [filename]**
Menu:	*Analysis - Contour Parameters - File - Load*
Description:	Reads a previously saved .ctr file and displays it in the current frame. The Load GUI dialog has options to change the color, and line weight used to draw the contours.

Command:	**contour convert**
Menu:	*Analysis - Contour Parameters - File - Convert to Polygons*
Description:	Creates a polygon region set from the current contours. There are now no DS9 contours defined (so *Contour Parameters - File - Save* will not do anything) but instead *Regions - List* will list a set of polygon regions and *Regions - Save* will save them as a DS9 region file.

The contour file format has a header:

```
# Contour file format: DS9 version 7.5
```

followed by one or more global properties lines, as in region files:

```
global color=green width=1 dash=1 dashlist=8 3
```

Global properties affect all contour levels unless a line level attribute is specified. The global keyword is first, followed by a list of keyword = value pairs. Multiple global property lines may be used.

For each contour file, it is important to specify the coordinate system used to interpret the contour points, i.e., to set the context in which the positions are interpreted. For this purpose, the following keywords are recognized:

```
        PHYSICAL                # pixel coords of original file using LTM/LTV
        IMAGE                   # pixel coords of current file
        FK4,B1950               # sky coordinate systems
        FK5,J2000               # sky coordinate systems
        ICRS                    # sky coordinate systems
        GALACTIC                # sky coordinate systems
        ECLIPTIC                # sky coordinate systems
        WCS                     # primary WCS
        WCSA                    # secondary WCS
        LINEAR                  # linear primary WCS
```

Each contour level is composed of a line value, line attributes and one or more contours. Line attributes are specified at the start of a new level and only affect contours within that level .

```
level=15.78775 color=pink width=2 dash=yes dashlist=2 2
```

Each contour is composed of one or more X,Y coordinates, with a deliminator of space or comma, starting with "(" and ending with ")". Contours are not closed.

```
(202.4836468 47.22380226
 202.4833538 47.2239185
 202.4831634 47.22409874
 202.4829883 47.22428858)
```

16 Catalogs and Archives

16.1 Name Resolver

DS9 includes an astronomical name resolver connected to the display, using external name servers. Supported nameservers are ned-sao (NED@SAO), ned-cds (NED@CDS), simbad-sao (SIMBAD@SAO), simbad-cds (SIMBAD@CDS), vizier-sao (VIZIER@SAO) and vizier-cds (VIZIER@CDS).

Command: **-**

Menu: *Edit - Preferences - Name Server - Default*

Description: Sets the default nameserver prior to startup.

Command: **nameserver open**

Menu: *Analysis - Name Resolution*

Description: Puts up the Name Resolution dialog window.

Command: **nameserver close**

Menu: *Analysis - Name Resolution - Close*

Description: Closes the Name Resolution dialog window.

Command: **nameserver server [nameserver]**

Menu: *Analysis - Name Resolution - Name Server - [nameserver]*

Description: Allows the name server to be selected. NED, SIMBAD and Vizier servers at both CDS and SAO are supported; SIMBAD at CDS is the default.

Command: **nameserver name [name]**

Menu: *Analysis - Name Resolution - Object - [name]*

Description: Supplies a name for nameserver search. For the GUI, must be followed by a Retrieve

Command: **-**

Menu: *Analysis - Name Resolution - Retrieve*

Description: Sends the current name to the name resolver and fills in the FK5 RA,Dec coordinates in the Name Resolution dialog.

Command: **nameserver skyformat sexagesimal|degrees**

Menu: *Analysis - Name Resolution - fk5 - sexagesimal|degrees*

Description: Selects the format of the returned coordinates in the dialog.

Command: **nameserver pan**

Menu: *Analysis - Name Resolution - File - Pan To*

Description: Centers the image on the coordinates returned by the nameserver.

Command: **nameserver crosshair**

Menu: *Analysis - Name Resolution - File - Crosshair To*

Description: Puts up a crosshair initially centered on the coordinates; the crosshair can then be moved.

16.2 Image Servers

The *Analysis - Image Servers* dialog allows the user to retrieve an image centered on a given location from the selected image server. By default the location is the current image center but a dialog allows the user to change the values of the object location and the size of the retrieved image. By default the retrieved image is displayed in a new frame.

The supported servers in DS9 8.7 are DSS (hosted at SAO, ESO or STScI), 2MASS (at IPAC), VLA, NVSS and VLSS (all hosted at NRAO) and SkyView (at HEASARC).

Image Servers and surveys supported by them are as follows:

Image server	**Token**	**Surveys**
IPAC 2MASS	2mass	j, h, k
ESO-DSS	dsseso	DSS1, DSS2-red, DSS2-blue, DSS2-infrared
SAO-DSS	dsssao, dss	-
STSCI-DSS	dssstsci	all, poss2ukstu_red, poss2ukstu_ir, poss2ukstu_blue, poss1_blue, poss1_red, quickv, phase2_gsc2, phase2_gsc1
NVSS	nvss	-
VLA	vla	first, stripe82, gps
VLSS	vlss	-
SkyView	skyview	(many)

Each image server may support multiple surveys. SkyView supports a very large number of surveys and the XPA tokens for them are not documented. SAO-DSS, NVSS and VLSS have only one survey each so no selection is needed. The STSCI-DSS 'all' option selects the best image from the combined list.

Command: **2mass**

Menu: *Analysis - Image Servers - 2MASS (NASA/IPAC)*

Description: Sets the current image server to 2MASS.

More generally:

Command: **[imserver]**

Menu: *Analysis - Image Servers - [imserver]*

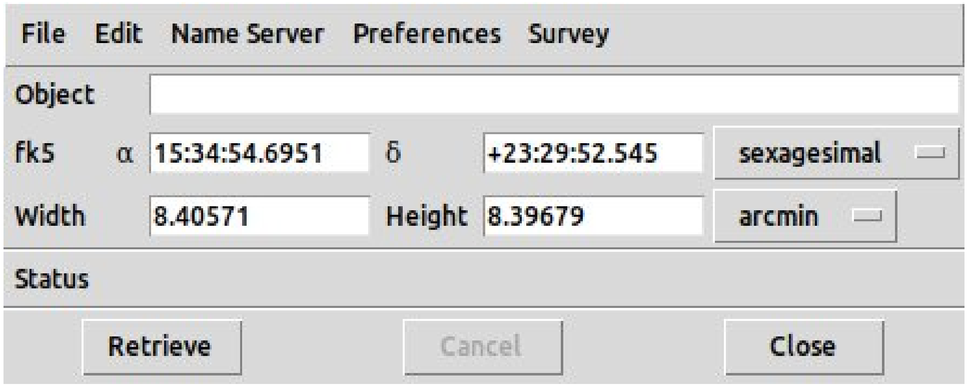
Description: Sets the current image server.

Once an image server is selected, we may need to specify a particular survey supported by the server, if it supports more than one. We then need to specify coordinates and a field size to retrieve an image to load into DS9.

Command: **[imserver] open**

Menu: *Analysis - Image Servers - [imserver]*

Description: Opens the image server dialog window for the given server.



Object

fk5 α 15:34:54.6951 δ +23:29:52.545 sexagesimal

Width 8.40571 Height 8.39679 arcmin

Status

Figure 35: Example of image server dialog for 2MASS

Command: **[imserver] update frame**

Menu: *Analysis - Image Servers - [imserver] - Update from Current Frame*

Description: Changes the target coordinates to the center of the current frame.

Command: **[imserver] update crosshair**

Menu: *Analysis - Image Servers - [imserver] - Update from Current Crosshair*

Description: Changes the target coordinates to the current crosshair position, if DS9 is in crosshair mode.

Command: **nameserver server [nameserver]**

Menu: *Analysis - Image Servers - [imserver] - Name Server*

Description: Specifies the name server to use. Supported nameservers are ned-sao (NED@SAO), ned-cds (NED@CDS), simbad-sao (SIMBAD@SAO), simbad-cds (SIMBAD@CDS), vizier-sao (VIZIER@SAO) and vizier-cds (VIZIER@CDS).

Command: -

Menu: *Analysis - Image Servers - [imserver] - Retrieve*

Description: Retrieves the image based on the current target coordinate and size parameters. Can only be done from the GUI.

Command: -

Menu: *Analysis - Image Servers - [imserver] - Acknowledgement*

Description: brings up a text box with a suitable acknowledgement for the server. Can only be done from the GUI.

Command: **[imserver] name [object]**

Menu: *Analysis - Image Servers - [imserver] - Object*

Description: Allows the user to enter an object name and uses the name server selected. (for GUI, user must click Retrieve)

.

Command: **[imserver] name clear**

Menu: *Analysis - Imager Servers - [imserver] - Edit - Clear*

Description: Clear the selected object name in the imserver dialog.

Command: **[imserver] [position]**

Menu: *Analysis - Imager Servers - [imserver] - fk5*

Description: Specify the coordinates of the source directly. Either sexagesimal or decimal degree format, as previously selected. Example: **2mass 00:40:00 +42:13:02**.

Command: **[imserver] size [width] [height] degrees|arcmin|arcsec**

Menu: *Analysis - Imager Servers - [imserver] - Width|Height*

Description: Sets the field size to retrieve in arcmin.

Command: **[imserver] save yes|no**

Menu: *Analysis - Image Servers - [imserver] - Preferences - Save Image on Download*

Description: Saves the retrieved image locally on disk.

Command: **[imserver] frame current|new**

Menu: *Analysis - Image Servers - [imserver] - Preferences - New Frame/Current Frame*

Description: Selects whether the image ends up in the current or new frame.

Command: **[imserver] survey [name]**

Menu: *Analysis - Image Servers - [imserver] - Survey*

Description: Sets which of several surveys supported by the server is used.

16.3 Archives

The DS9 archives command retrieves data from external archives.

There are two special cases:

- *Archives - Chandra public archive by obsid* takes an OBSID (e.g. 869) and displays a dialog with links to the primary data products for that obsid. On downloading the EVT2 file, the evt2 image is displayed in a new frame. Downloading another data product such as the ASOL file creates a new blank frame and pops up an error message.
- *Archives - Chandra public archive by cone search* provides a dialog which allows the user to enter an object name and a search radius; *... File - Retrieve* executes the name server and populates the dialog table with details on the selected observations. A simple filter language is provided to select a subset of the observations. Double clicking on a row in the table then pops up a retrieval window with links to the data products.

The remaining archives share an interface, including the Chandra CXC archive option.

- *Archives - Chandra CXC*, or *Archives - (survey)* for other values of (survey), provides a dialog with a list of observations matching the current frame location.
- *Archives - (survey) - Object* provides a dialog where a name can be entered (and resolved using the selected name server at *Archives - (survey) - Name Server*), or coordinate values can be entered. The coordinate values are initialized to the center of the current frame, and can be updated using **2mass update frame—crosshair**, *Archives - (survey) - File - Update from Current Frame* or *Archives - (survey) - File - Update from Current Crosshair*.

- Once the object is resolved, *Archives - (survey) - Retrieve* queries the archive and returns the updated list of observations.

- The list of observations may be printed using *Archives - (survey) - File - Print* or exported to a TSV file using *Archives - (survey) - Export - Starbase* or *Archives - (survey) - Export - Tab-Separated-Value*.

- Click on one entry in the list to select, and use *Archives - (survey) - Load* to load the observation into DS9.

- *Archives - (survey) - Preferences - New Frame/Current Frame* controls whether the observation is loaded in a new frame or in the current frame.

- *Archives - (survey) - Preferences - Save Image on Download* saves the resulting data to local disk.

- *Archives - (survey) - Clear* clears the current list of observations.

16.4 Catalog server search

DS9 provides support for loading, displaying, filtering, and saving catalogs. DS9 allows you to overlay symbols from multiple catalogs on the current image.

Local and on-line catalog access is supported. Most major catalogs can be retrieved from online servers. Both the CDS and SDSS catalog servers are supported. Local catalog files in starbase (rdb) or CSV (with or without header) are supported (as well as FITS and VOTable).

On-line catalogs are available via services provided by the VizieR catalog access tool, CDS, Strasbourg, France (VizieR is a joint effort of the Centre de Donnees Astronomiques de Strasbourg and ESA-ESRIN Information Systems Division) and by the Sloan Digital Sky Survey.

A selection of popular catalogs is provided in the Analysis menu. In addition, you can search for other catalogs based on title, keywords, mission, wavelength, and object type.

When a catalog is overlayed on an image, each displayed catalog symbol consists of a shape, color, and text. An advanced symbol editor is available that allows you to specify the shape, size, color, and text of each symbol, based on catalog column values. These symbol expressions can be saved for future use.

Along with the overlay display, a catalog list is provided in a separate window. It displays the column values for each catalog object. The catalog list can be sorted and filtered, and the catalog display will be automatically updated. Advanced filtering options are available. Catalogs can be loaded and saved as local files in ASCII Starbase format. Each catalog contains header information which can be displayed. The list can be printed separately from the image.

An interactive connection between the displayed catalog symbols and the catalog list is provided. When you select one or more rows within the catalog list, the corresponding symbols are highlighted on the image display. Conversely, selecting multiple symbols on the image display will highlight the corresponding rows within the catalog list. Catalog symbols can be converted to regions for use with analysis tasks.

Command: -

Menu: *Analysis - Catalogs - Search for Catalogs*

Description: provides a general catalog search tool.

Command: **catalog server [value]**

Menu: *Analysis - Catalogs - Search for Catalogs - Catalog Server*

GUI Default: *Edit - Preferences - Catalogs - Server*

Description: Select which catalog server to use. Options are cds (default), adac, sao, inasan, iucaa, bejing, saao.

Command: -

Menu: *- Search for Catalogs - Name or Designation*

Description: allows you to enter an object name.

Command: -

Menu: *... - Search for Catalogs - Retrieve*

Description: then generates a list of catalogs relevant to that object.

Alternatively, the Search for Catalogs dialog provides the ability to search for catalogs by wavelength band, mission, or astronomical topic. Again, the Retrieve option will then generate the list of relevant catalogs.

Command: -

Menu: *... Search For Catalogs - Load*

Description: Once the list has been generated, click on one catalog in the list and select *Search for Catalogs - Load*. This brings up a new window, the Catalog Source List window, with information on the relevant entries from that catalog.

Command: **catalog location [code]**

Menu: *Edit - Preferences - Catalogs - IAU Location Code*

Description: The code is set to the default value of 500 (Geocentric). The code is used for tasks which calculate which near field objects are in the field of an image. Currently the only such case is Skybot, which is no longer working.

The user may create several Catalog Source List windows with entries from different catalogs. A new window may be created with

Command: **catalog new**

Menu: *Analysis - Catalog Tool*

Description: Brings up a new Catalog Tool window.

This allows you to load a previously saved catalog xml file. Otherwise you won't normally bring up this window directly, it will come up from a catalog search in the Analysis - Catalogs menus.

For example, you can directly load a source list from a specific catalog with sources that match the current image location, e.g.

Command: **catalog 2mass**

Menu: *Analysis - Catalogs - Infrared - 2MASS Point Sources*

Description: Open a source catalog list window (also called the Catalog Tool window), and displaysource circles on the image, for sources from the 2MASS catalog in the current field.

More generally:

Command: **catalog [name]**

Menu: *Analyis - Catalogs -*

Description: Open the named catalog.

Supported catalog tokens for the CL are: ned, simbad, denis, skybot, aavos, ac, ascss, cmc, gaia, gsc1, gsc2, gsc3, nomad, ppmx, sao, sdss5, sdss6, sdss7, sdss8, sdss9, tycho, ua2, ub1, ucac2, ucac2sup, ucac3, ucac4, urat1, 2mass, iras, csc, xmm, rosat, first, nvss, chandralog, cfhtlog, esolog, stlog, xmmlog.

Also supported are

Command: **catalog cds [name]**

Menu: -

Description: Open the named CDS catalog. Example: catalog cds 2mass.

and

Command: **catalog cds [id]**

Menu: -

Description: Open the CDS catalog with the given Vizier catalog ID. Example: catalog cds {I/252}.

The Catalog Tool window has (see image):

- A catalog description pane, giving the name of the catalog and identifying information from the catalog server.
- An object pane, giving the coordinates and optionally a user-entered object name, and a search radius.
- A table header pane, giving a filter text box, the option to sort on a table column, a box to specify the max number of rows retrieved, an indication of the actual number of rows found, and an interface to specify which columns in the table are the RA, Dec columns - usually prefilled with the correct values using IVOA UCD metadata.
- A source list table with the actual source data, where each row is a source from the catalog, and the columns depend on the catalog but usually include at least a name and an RA,Dec pair. The RA/Dec columns are identified using VO UCDs. If the wrong ones are picked, the user can change the selection with dropdown menus in the table header pane.

Now the following actions are available:

Command: **catalog retrieve**

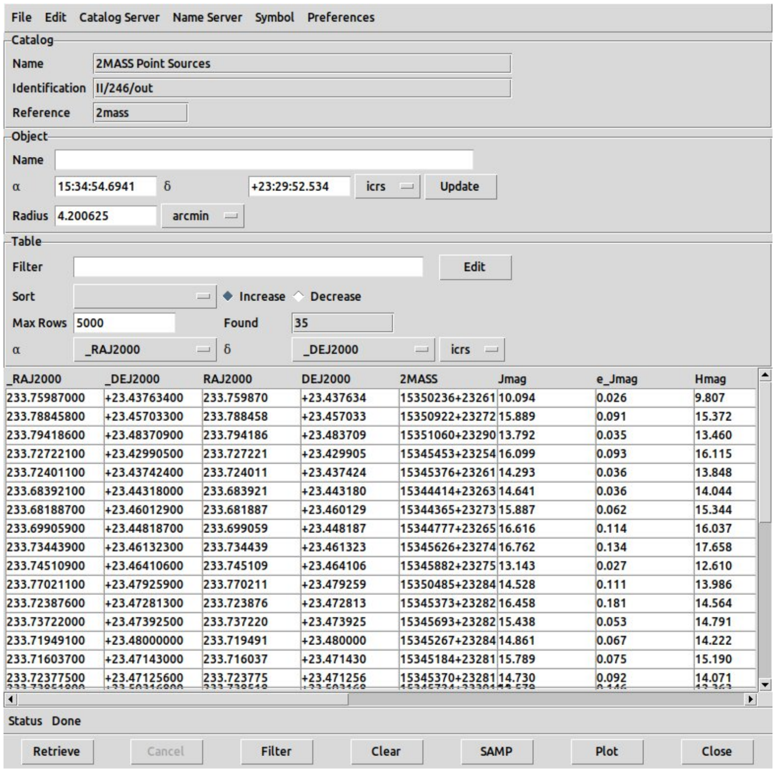
Menu: *Catalog Tool - Retrieve*

Description: Generates a new source list table given the current retrieval parameters.

Command: **catalog cancel**

Menu: *Catalog Tool - Cancel*

Description: cancels a too-slow retrieval in process.



Catalog

Name: 2MASS Point Sources

Identification: II/246/out

Reference: 2mass

Object

Name:

α: 15:34:54.6941 δ: +23:29:52.534 icrs

Radius: 4.200625 arcmin

Table

Filter:

Sort: ● Increase ○ Decrease

Max Rows: 5000 Found: 35

α: _RAJ2000 δ: _DEJ2000 icrs

_RAJ2000	_DEJ2000	RAJ2000	DEJ2000	2MASS	Jmag	e_Jmag	Hmag
233.75987000	+23.43763400	233.759870	+23.437634	15350236+23261	10.094	0.026	9.807
233.78845800	+23.45703300	233.788458	+23.457033	15350922+23272	15.889	0.091	15.372
233.79418600	+23.48370900	233.794186	+23.483709	15351060+23290	13.792	0.035	13.460
233.72722100	+23.42990500	233.727221	+23.429905	15345453+23254	16.099	0.093	16.115
233.72401100	+23.43742400	233.724011	+23.437424	15345376+23261	14.293	0.036	13.848
233.68392100	+23.44318000	233.683921	+23.443180	15344414+23263	14.641	0.036	14.044
233.68188700	+23.46012900	233.681887	+23.460129	15344365+23273	15.887	0.062	15.344
233.69905900	+23.44818700	233.699059	+23.448187	15344777+23265	16.616	0.114	16.037
233.73443900	+23.46132300	233.734439	+23.461323	15345626+23274	16.762	0.134	17.658
233.74510900	+23.46410600	233.745109	+23.464106	15345882+23275	13.143	0.027	12.610
233.77021100	+23.47925900	233.770211	+23.479259	15350485+23284	14.528	0.111	13.986
233.72387600	+23.47281300	233.723876	+23.472813	15345373+23282	16.458	0.181	14.564
233.73722000	+23.47392500	233.737220	+23.473925	15345693+23282	15.438	0.053	14.791
233.71949100	+23.48000000	233.719491	+23.480000	15345267+23284	14.861	0.067	14.222
233.71603700	+23.47143000	233.716037	+23.471430	15345184+23281	15.789	0.075	15.190
233.72377500	+23.47125600	233.723775	+23.471256	15345370+23281	14.730	0.092	14.071

Status Done

Figure 36: Example catalog tool with 2MASS source list

Command: **catalog clear**

Menu: *Catalog Tool - Clear*

Description: Clears the table. (Note that closing the catalog tool window is not enough to cause the source circles in the display image to disappear).

Command: **catalog update**

Menu: *Catalog Tool - File - Update from Current Frame*

Description: Changes the retrieve coordinates in the object pane to the current frame center

Command: -

Menu: *Catalog Tool - File - Update from Current Cursor*

Description: Changes the retrieve coordinates in the object pane to the current cursor pos.

Command: **catalog sort [col] incr|decr**

Menu: *Catalog Tool - Table - Sort*

Description: Sort the table. The GUI version has a dropdown menu to select a table column to sort on.

Command: **catalog filter [string]**

Menu: *Catalog Tool - Table - Filter*

Description: Filter the table. The GUI provides a text box to enter the filter. The table may be filtered using TCL expression syntax, with column names prefixed by a dollar sign, e.g. $Jmag<14.2

Command: **catalog ra [col]**

Menu: *Catalog Tool - Table - alpha*

Description: Define which column in the table is the RA column.

Command: **catalog dec [col]**

Menu: *Catalog Tool - Table - dec*

Description: Define which column in the table is the Dec column.

Command: **catalog print**

Menu: *Catalog Tool - File - Print*

Description: Prints the table to a printer or to a text file.

Command: **catalog plot [xcol][ycol] [xerr][yerr]**

Menu: -

Description: Make scatter plot of one column against another, optionally with error columns.

Command: **-**

Menu: *Catalog Tool - Plot*

Description: This menu item will bring up a new window with a scatter plot of one table column against another. The Plot dialog allows you to define which columns to plot on the X and Y axis; the default is RA and Dec. The plot generated is a Prism Plot as described earlier.

Command: **catalog import rdb|tsv|fits [filename]**

Menu: *Catalog Tool - Import - Starbase/TSV/FITS*

Description: Loads a source list from a Starbase, tab-sep-value or FITS-table catalog.

Command: **catalog export rdb|tsv [filename]**

Menu: *Catalog Tool - Export - Starbase/TSV*

Description: Exports the current source list to a Starbase, or tab-sep-value file.

Command: **catalog save [filename]**

Menu: *Catalog Tool - Save*

Description: Saves the current source list in an XML file.

Command: **catalog load [filename]**

Menu: *Catalog Tool - Open*

Description: Opens the catalog tool and loads a previously saved XML source list.

Command: **catalog close**

Menu: *Catalog Tool - Close*

Description: Closes the catalog tool

Command: **catalog current [ref]**

Menu: -

Description: Current catalog. Used to specify which version of the Chandra source catalog to use.

Command: **catalog regions**

Menu: *Catalog Tool - File - Copy to Regions*

Description: Copy source locations to region format. They are added to the region list in the current frame. These may then be saved to a region file with *Region - Save Regions*.

Command: **catalog samp**

Menu: *Catalog Tool - SAMP*

Description: Will draw the source locations in the current frame. Once the SAMP plot has been made, if *Edit - Catalog* is selected, clicking on a row in the source table will highlight a source symbol (default: green circle point) in the display frame. Clicking on a source circle in the display frame will highlight the corresponding row in the source table. Mutiple rows may be selected by holding down the SHIFT or CTRL key while selecting them.

The source list window allows controlling of various options:

Command: **catalog coordinate [ra] [dec] [coordsys]**

Menu: *Catalog Tool - Object - alpha,delta - Update*

Description: Specify the coordinates for the search center.

Command: **catalog name [value]**

Menu: *Catalog Tool - Object - Name*

Description: Specify the name of the object whose coordinates are to be used in the catalog search. In the GUI, use Retrieve to update the table

Command: **catalog radius [value] degrees|arcmin|arcsec**

Menu: *Catalog Tool - Object - Radius*

Description: Specify the seach radius for the catalog search.

Command: **catalog edit yes|no**

Menu: *Catalog Tool - File - Edit*

Description: Allows you to directly edit numerical values in the cells of the source table.

Command: **catalog panto yes|no**

Menu: *Catalog Tool - Preferences - Pan To*

Description: Selected by default. When a source is selected in the list, the image window will pan to be centered on its location.

Command: **catalog allrows**

Menu: *Catalog Tool - Preferences - All Rows*

Description: If set, all catalog rows are available in the table.

Command: **catalog allcols**

Menu: *Catalog Tool - Preferences - All Columns*

Description: If set, all catalog columns are available in the table, otherwise a preferred subset is shown.

Command: **catalog maxrows [N]**

Menu: *Catalog Tool - Table - Max Rows*

Description: Set the maximum number of rows to be retrieved.

Command: **catalog header**

Menu: *Catalog Tool - File - Header*

Description: Brings up a window with metadata describing the table columns.

Command: **catalog sky [coordsys]**

Menu: *Catalog Tool - Object - [coordsys]*

Description: Specify coord system for retrieve. Valid tokens: fk4, b1950, j2000, icrs, galactic, ecliptic.

Command: **catalog psystem [wcs]**

Menu: *Catalog Tool - Object - [coordsys]*

Description: Specify coord system for retrieve. Valid tokens: wcs, wcs[letter].

Command: **catalog show yes|no**

Menu: *Catalog Tool - File - Show*

Description: (on by default) can be used to show or hide the displayed source symbols.

Command: **catalog psky [coordsys]**

Menu: *Catalog Tool - Table - [coordsys]*

Description: Specify coord sys for table rows. (fk4, fk5, icrs, galactic, ecliptic).

Command: **catalog system [wcs]**

Menu: *Catalog Tool - Table - [coordsys]*

Description: Specify coord system for table. Valid tokens: wcs, wcs[letter].

Command: **mode catalog**

Menu: *Edit - Catalog*

Description: In catalog mode, when a catalog is loaded in the image, left clicking on a region displayed on the image that corresponds to a catalog source will highlight that entry in the catalog dialog window.
Clicking on a row in the catalog dialog window will highlight that source region (in red, by default) whatever edit mode is selected.

Command: **-**

Menu: *Catalog Tool - SAMP - Connect/Disconnect*

Description: connects and disconnects the SAMP connection.

Command: **-**

Menu: *Catalog Tool - SAMP - Send*

Description: allows you to send the data over SAMP, either broadcasting or to a specific named DS9.

Command: **-**

Menu: *Catalog Tool - File - Acknowledgment*

Description: brings up a window with text containing a suitable acknowledgement for using the catalog (however, this text may be empty).

16.4.1 Catalog symbol control

Command: -

Menu: *Catalog Tool - Symbol*

Description: provides options to control the symbol properties plotted for the sources in DS9. If multiple catalogs are used at once for the same frame, it may be useful to use different symbols for each catalog.

Command: -

Menu: *Catalog Tool - Symbol - Shape*

GUI Default: *Edit - Preferences - Catalogs - Shape*

Description:

Command: -

Menu: *Catalog Tool - Symbol - Color*

GUI Default: *Edit - Preferences - Catalogs - Color*

Description:

Command: -

Menu: *Catalog Tool - Symbol - Width*

GUI Default: *Edit - Preferences - Catalogs - Width*

Description:

Command: -

Menu: *Catalog Tool - Symbol - Font*

GUI Default: *Edit - Preferences - Catalogs - Font*

Description:

An advanced symbol editor is available that allows you to specify the shape, size, color, and text of each symbol, based on catalog column values. For each row of the catalog, one or more conditional expressions are evaluated. For the first expression to evaluate true, a given symbol is displayed, with the specified shape, color, size and text properties. As with the filter, the value of a particular column can be indicated as

```
$columnname
```

For the condition entry, the expression you type in is automatically evaluated via TCL expr after macro expansion.

```
1                             # always
0                             # never
true                          # always
false                         # never
$Jmag>2                       # conditional
sin($Jmag)>.5                 # conditional
[string equal $Class SNR]     # conditional
[regexp {*SNR*} $Class]       # conditional
```

For the size, size2, and angle entries, the expression you type in is also automatically evaluated via TCL expr after macro expansion.

```
2               # value of '2' is used
$Jmag           # value of column Jmag is used
$Jmag/2.        # value of column Jmag div 2 is used
(4+2)/3         # value of '2' is used
```

For the text portion, this is not true. It is assumed to be text, unless you explicitly use an expr operator.

```
foo             # will put 'foo' above the symbol
$Jmag           # will put the value of column Jmag above the symbol
(4+2)/3         # will put the text '(4+2)/3' above the symbol
[expr (4+2)/3]  # will put the text '2' above the symbol
[expr $Jmag/2.] # will take the value of Jmag and div by 2
```

And finally, one special case for shape = text and text = empty. In this case, the row number is displayed.

Command: -

Menu: *Catalog Tool - Symbol - Advanced*

Description: pops up a symbol editor window which can define the use of different symbols for table entries satisfying different logical conditions. An expression editor is provided to construct the conditions. For example, if the table has a redshift column, different symbol colors can be associated with different value ranges of the redshift.

Command: -

Menu: *Catalog Tool - Symbol - Advanced - Symbol Editor - Text - Edit*

Description: To add source names to the display, open the symbol editor with *Catalog Tool - Symbol - Advanced*, and use *Symbol Editor - Text - Edit* to add a column name in the Text field, then *Symbol Editor - Apply*.

Command: -

Menu: *Symbol Editor - File - Save*

Description: saves the current symbol definitions in a .sym file.

Command: -

Menu: *Symbol Editor - File - Open*

Description: opens a previously saved file.

16.4.2 Catalog Filtering

The catalog list can be sorted and filtered, and the catalog display will be automatically updated. A filter is conditional expression, when evaluated for each row of the catalog, if true, the row is displayed, and if false, the row is not displayed. The conditional expression can be any valid TCL expression. The value of a column may be indicated with `$colname`, for example

```
$_RAJ2000>180. && $_RAJ2000<270.
$Jmag>11
log($Kmag*10)<.3
[string equal $OTYPE_S SNR]
```

Command: -

Menu: *Catalog Tool - Table - Filter - Edit*

Description: pops up a dialog which helps you construct the filter, with Column and Operator and Math Function options. Compound expressions such as ”$Jmag<14——$Jmag>16.5” are supported.
Supported math functions are acos, asin, atan, atan2, ceil, cos, cosh, exp, floor, fmod, hypot, log, log10, pow, sin, sinh, sqrt, tan, tanh, abs, double, int and round.
Supported operators are -, !, (,), *, /, %, +, -, <, >, <=, >=, ==, != and &&.

Command: -

Menu: *Catalog Tool - Filter - Edit - File - Save*

Description: Saves the current filter to an .flt file.

Command: **catalog filter load [filename]**

Menu: *Catalog Tool - Filter - Edit - File - Open*

Description: Opens a .flt file containing a previously saved filter.

16.5 Footprint Servers

Command: -

Menu: *Analysis - Footprint Servers*

Description: This menu allows you to select from a choice of astronomical telescope footprint servers. For example, one may select the Chandra server. This pops up a footprint table dialog, which shows a table of observations whose footprints overlap the current image. Each row in the table gives the properties of the footprint of one observation, and the corresponding outline of the telescope detectors is superimposed on the image (by default, in green). A pointing direction and search radius is shown at the top of the table.
Clicking on a row in the table will change the color of the corresponding footprint outline in the image (by default, from green to red).
The footprint server dialog has Object and Table panes similar to those in the catalog tool, allowing the target position to be modified and the table entries to be refreshed accordingly.
In addition, a list of instruments is provided; one may deselect some of these and execute Retrieve to refresh the table with only those entries corresponding to selected instruments.
As with the catalogs, a Copy to Regions function is available.
Currently supported servers are CXC (Chandra) and HLA (Hubble Legacy Archive).

Command: **footprint cxc|hla**

Menu: *Analysis - Footprint Servers - [server]*

Description: Calling the footprint server results in opened a window with a table of relevant observations, and corresponding observation sky coordinate footprints overplotted on the current frame. Clicking on a row in the table highlights the corresponding footprint region in the frame.

Command: -

Menu: *Edit - Footprint Mode*

Description: If this mode is selected, clicking on a footprint outline in the image will highlight the corresponding entry in the footprint table dialog.

Command: **footprint retrieve**

Menu: *Analysis - Footprint Servers - [server] - Retrieve*

Description: The Retrieve button updates the footprint table if changes have been made to the position and radius.

Command: **footprint cancel**

Menu: *Analysis - Footprint Servers - [server] - File - Cancel*

Description: Cancel the (slow) retrieval currently underway.

Command: **footprint clear**

Menu: *Analysis - Footprint Servers - [server] - Clear*

Description: Clear the footprint table

Command: **footprint close**

Menu: *Analysis - Footprint Servers - [server] - Close*

Description: Clost the footprint server window

Command: **footprint name** [**value**]

Menu: *Analysis - Footprint Servers - [server] - Object - name*

Description: Sets the name of the object which is name-resolved to a position for the next Retrieve action

Command: **footprint coordinate** [**x**] [**y**] **fk5**

Menu: *Analysis - Footprint Servers - [server] - Object - alpha/delta*

Description: Sets the position for the next Retrieve action

Command: **footprint sky icrs|fk5|galactic...**

Menu: *Analysis - Footprint Servers - [server] - Object - [coordsys]*

Description: Sets the coordinate system for the object coordinates

Command: **footprint skyformat [degrees|sexagesimal]**

Menu: *Analysis - Footprint Servers - [server] - Object - [coordsys] - Degrees/Sexagesimal*

Description: Sets the format for the object coordinates

Command: **footprint radius [value]**

Menu: *Analysis - Footprint Servers - [server] - Object - radius*

Description: Sets the search radius for the next Retrieve action

Command: **footprint crosshair**

Menu: *Analysis - Footprint Servers - [server] - File - Update from Current Crosshair*

Description: Sets the position for the next Retrieve action via the current crosshair position

Command: **footprint update**

Menu: *Analysis - Footprint Servers - [server] - Object - Update*

Description: Update coordinates for current field

Command: **footprint current cxc|hla**

Menu: *-*

Description: Set current footprint server

Command: **footprint filter [string]**

Menu: *Analysis - Footprint Servers - [server] - Table - Filter*

Description: Filter the rows in the footprint table

Command: **footprint sort [colname] incr|decr**

Menu: *Analysis - Footprint Servers - [server] - Table - Sort [colname] - Increase/Decrease*

Description: Sort the footprint table on specified column

Command: **footprint filter load [filename]**

Menu: *Analysis - Footprint Servers - [server] - Table - Filter Edit - File - Open*

Description: Load a table filter string saved in a .flt file

Command: **footprint panto yes|no**

Menu: *Analysis - Footprint Servers - [server] - Preferences - Panto*

Description: Pan to footprint location upon load

Command: **footprint save [filename]**

Menu: *Analysis - Footprint Servers - [server] - File - Save*

Description: Saves the current footprint regions as an XML file

Command: **footprint export rdb [filename]**

Menu: *Analysis - Footprint Servers - [server] - File - Export Starbase*

Description: Saves the current footprint regions as a Starbase RDB file

Command: **footprint export tsv [filename]**

Menu: *Analysis - Footprint Servers - [server] - File - Export Tab-Separated-Value*

Description: Saves the current footprint regions as a TSV file

Command: **footprint print**

Menu: *Analysis - Footprint Servers - [server] - File - Print*

Description: Print footprint list

Command: **footprint regions**

Menu: *Analysis - Footprint Servers - [server] - File - Copy to Regions*

Description: Convert footprints to DS9 regions

Command: **footprint show yes|no**

Menu: *Analysis - Footprint Servers - [server] - File - Show*

Description: Display footprint overlays or not

16.6 VO SIA

The sia command provides support for the Virtual Observatory (VO) Simple Image Access (SIA) protocol.

Command: **sia [archive]**

Menu: *Analysis - Archives - [archive]*

Description: Select a catalog for an SIA search. Pops up a dialog box which will be filled with the images returned by the search.

Command: **sia coordinate [x] [y] fk5**

Menu: *Analyis - [archive] - alpha/delta*

Description: Define SIA search position on the current catalog. Example: xpaset -p ds9 sia coordinate 202.48 47.21 fk5.

Command: **sia crosshair**

Menu: *Analysis - Archives - [archive] - File - Update from Current Crosshair*

Description: Update the search position from the crosshair position

Command: **sia sky [system]**

Menu: *Analysis - Archives - [archive] - Object - alpha/delta*

Description: Set coordinate system. Example: xpaset -p ds9 sia sky fk5

Command: **sia skyformat [degrees|sexagesimal]**

Menu: *Analysis - Archives - [archive] - Object - [coordsys]*

Description: Set coordinate system format

Command: **sia radius [value] [units]**

Menu: *Analysis - Archives - [archive] - Object - Radius*

Description: Set SIA search radius. Example: xpaset -p ds9 sia radius 1 degrees

Command: **sia current [catalog]**

Menu: *Analysis - Archives - [archive] - File - Update from Current Frame*

Description: Update the search position from the current position

Command: **sia name [value]**

Menu: *Analysis - Archives - [archive] - Object - Name*

Description: Update the search position based on an object name

Command: **sia update**

Menu: -

Description: Update the SIA search following changes to the search position

Command: **sia save [filename.xml]**

Menu: *Analysis - Archives - [archive] - File - Save*

Description: Save table results of SIA search in a file

Command: **sia retrieve**

Menu: *Analysis - Archives - [archive] - Retrieve*

Description: Execute the SIA search

Command: **sia cancel**

Menu: *Analysis - Archives - [archive] - Cancel*

Description: Cancel SIA search in progress

Command: **sia clear**

Menu: *Analysis - Archives - [archive] - Clear*

Description: Clear SIA search

Command:	**sia close**
Menu:	*Analysis - Archives - [archive] - Close*
Description:	Close SIA dialog window

17 Control and Interprocess communication - TCL, XPA and SAMP

17.1 Batch mode

It is possible, but cumbersome, to use DS9 in batch mode without a physical display. DS9 is written as an interactive, window client program, and as a result, does require a window server to be available for rendering (X11, Windows, or MacOSX). Under X11, we recommend using xvfb for batch mode. You set up a virtual display buffer, reset your DISPLAY variable, then invoke DS9 with a number of command line options or use xpa from a shell script as a batch processor. Example:

```
% export DISPLAY=:1
% Xvfb :1 -screen 0 1024x768x16 &
% ds9 -file cmap.fits -zoom to fit -cmap b -grid skyformat degrees -grid yes -regions ../EMS-names
```

17.2 HTTP access

Edit - Preferences - HTTP supports definition of a proxy host and its settings for those functions that perform HTTP requests.

17.3 TCL scripting

Command:	**console**
Menu:	*File - Open TCL console*
Description:	Displays a window in which the user can enter low level TCL commands that will be passed to the DS9 intepreter and executed. The console window has a lot of menu options for fine control, none of which are currently documented.

Command: source [filename]

Menu: *File - Source TCL*

Description: Allows the user to open a .tcl file whose contents will be passed to the DS9 interpreter and executed.

Command: tcl [string]

Menu: -

Description: Execute a single TCL command. Only works if XPA_METHOD is 'local'. Example: 'xpaset -p ds9 tcl {puts {Hello, World}}'

17.4 XPA

At startup, by default, the XPA and SAMP systems are initialized.

DS9 uses XPA for interprocess communication. When DS9 starts, XPA initializes itself. XPA uses either IP sockets or UNIX sockets, based on whether your machine is configured to connect to the internet or not. In the case where your machine is configured for the internet, but you are not currently connected, XPA gets very confused. So, you can define a shell variable, XPA_METHOD, that tells XPA which method to use.

The default XPA_METHOD = "inet" will use internet based sockets and allow access from other machines. XPA_METHOD = "local" uses local UNIX sockets, most useful when you are not connected to the internet.

Command: xpa info

Menu: *File - XPA - Information*

Description: Pops up a window that gives information about the XPA message bus.

Command: xpa connect|disconnect

Menu: *File - XPA - Connect / Disconnect*

GUI Default: *Edit - Preferences - Startup - Initialize XPA*

Description: Connect or disconnect to XPA xpans server

17.5 XPA Scripting

XPA (X Public Access) is a messaging system which provides communication between Unix programs through a set of access points. The two most common actions are retrieving information (xpaget) and issuing commands (xpaset). For more information, see the XPA Messaging System page and the XPA Access Points section of the DS9 manual.

XPA commands may be issued one at a time from the terminal or collected in a script to run in batch mode. Unlike the command line syntax, there is no predetermined stopping point - commands may be sent to DS9 as long as the GUI is open.

First, open DS9 with the data file:

```
unix% ds9 acisf05005N002_evt2.fits &
```

The xpans name server is used to manage the names and ports of XPA access points. Use ”xpaget xpans” to see the list of available access points:

```
unix% xpaget xpans
DS9 ds9 gs /tmp/.xpa/DS9_ds9.22972 username
```

Now that DS9 is running and linked to an XPA server, we can use xpaset to modify the display and add contours. The `xpaset -p` command is used to set DS9 parameters. (xpaset on its own passes data such as an image; the p option is for passing parameters with no data).

```
unix% xpaset -p ds9 bin about 3800 3800
unix% xpaset -p ds9 bin factor 2
unix% xpaset -p ds9 scale log
unix% xpaset -p ds9 cmap b
unix% xpaset -p ds9 contour yes
unix% xpaset -p ds9 contour limits 1 100
unix% xpaset -p ds9 contour smooth 5
unix% xpaset -p ds9 contour nlevels 6
unix% xpaset -p ds9 contour save xpa.con
unix% xpaset -p ds9 contour copy
unix% xpaset -p ds9 dssao A2142
unix% xpaset -p ds9 cmap grey
unix% xpaset -p ds9 contour paste
unix% xpaset -p ds9 frame first
unix% xpaset -p ds9 match frames wcs
```

This command line produces the figure shown.

The commands are:

1	bin about 3800 3800	center the image displ
2	bin factor 2	bin the data by a fact
3	scale log	set the display to log
4	cmap b	use the ”b” colormap
5	contour yes	display contours
6	contour limits 1 100	set the minimum and
7	contour smooth 5	set contour smoothne
7	contour nlevels 6	create six contour leve
9	contour save xpa.con	save the contours to t
10	contour copy	copy the x-ray contou
11	dsssao A2142	retrieve a DSS image
12	cmap grey	use the ”grey” colorm
13	contour paste	paste the x-ray contou
14	frame first	select the first ds9 fra
15	match frames wcs 7 match the WCS of the DSS frame to the current (x-ray) frame	

Command:	**cd [filename]**
Menu:	-
Description:	Set the current working directory used for file open/save dialogs

17.6 XPA Remote commands

You can command DS9 on one machine with XPA commands sent from another machine. XPA uses port 14285 by default.

Command:	**acl [client] +**
Menu:	-
Description:	Grant client access to server

Suppose that DS9 is running on a machine called kirk.harvard.edu and you want to control it from another machine called spock.harvard.edu.

Example: on kirk.harvard.edu, type:

```
xpaset -p ds9 -acl spock.harvard.edu +
```

This can be set up in advance by clearing a /home/[user]/acls.xpa file on kirk.harvard.edu containing the line

```
DS9:ds9 spock.harvard.edu +
```

On the client side (spock), the client needs to communicate with the xpansname server program on the server machine (kirk) to find the XPA server communication info. This can be done in two ways:

On spock.harvard.edu, type

```
setenv XPA_NSINET 'kirk.harvard.edu:14285'
```

or

```
xpaget -i 'kirk.harvard.edu:14285' ds9
```

You should now be able to type xpaset commands on spock that will change the state of the DS9 running on kirk.

17.7 SAMP

SAMP is a messaging protocol that enables astronomy software tools to interoperate and communicate. SAMP-enabled tools include TOPCAT, SAOImageDS9, Aladin, and Astropy. SAOImageDS9 SAMP support is available for all ports of DS9, including linux, MacOS, and Windows.

Broadly speaking, SAMP is an abstract framework for loosely-coupled, asynchronous, RPC-like and/or event-based communication, based on a central service providing multi-directional publish/subscribe message brokering. The message semantics are extensible and use structured but weakly-typed data. For more information on SAMP, please see https://www.ivoa.net/documents/SAMP/

SAMP defines a number of message types, or MTYPEs, which are common among SAMP tools, such as image.load.fits, table.load.votable, coord.pointAt.sky. These messages allow images and tables to be exchanged between tools.

The samp implementation for DS9 is based on the XPA model with 2 additional SAMP MTYPEs, ds9.get and ds9.set, each with a required 'cmd' string argument and an optional 'url' string argument. They return either OK or ERROR and a value or error text.

Any SAOImageDS9 function that can be invoked via the command line can also be invoked via SAMP.

ds9.set may be called via notification, call and call/wait. ds9.get can only be called via call and call/wait. Most of the ds9.get calls return a value string, but a few will return a url instead.

DS9 SAMP setup commands are:

Command:	**samp broadcast image|table**
Menu:	*File - SAMP - Image/Table - Broadcast - (name)*
Description:	Broadcast image or table to the target DS9 process with the given process name (started with ds9 -title). The current image/table in the source DS9 will then be set to the target DS9 and displayed there.

Command: **samp connect|disconnect**

Menu: *File - SAMP - Connect/Disconnect*

GUI Default: *Edit - Preferences - Startup - SAMP Connect*

Description: SAMP disconnect and connect

Command: **-**

Menu: *Edit - Preferences - Startup - SAMP WebHub Start*

Description: on by default; initializes SAMP at startup.

Command: **samp send image|table [application]**

Menu: *-*

Description: SAMP send data to application. Example: samp send table topcat

Command: **samp hub info**

Menu: *File - SAMP Hub - Information*

Description: Send info to SAMP

Command: **samp hub start|stop**

Menu: *File - SAMP Hub - Start/Stop*

GUI Default: *Edit - Preferences - Startup - SAMP Hub Start*

Description: Start/stop samp hub

SAMP's calls to DS9 make use of the DS9 command syntax. For example:

```
ds9.set("contour clear")
ds9.set("contour generate")
ds9.set("contour load ds9.ctr")
```

See the Astropy section for usage that is more likely to be immediately useful to the user.

17.8 Astropy

You can control DS9 from Astropy by using the SAMP interface.

We assume you have not already started a SAMP hub in python and that we will be using the SAMP hub that DS9 initializes when it starts up. In this case, the SAMP hub internal ID of the DS9 client is always ”c1”. The ecall_and_wait method we will use has as its third argument a timeout argument in seconds; we will use a 10 second timeout. (In other words, if you are not a SAMP expert and you are only using it in DS9, then always use ‘ds9.ecall_and_wait(”c1”,”ds9.set”,”10”,cmd=”whatever”)’ and just change the cmd= bit.)

- Import the SAMP client in astropy, create an instance of the client, and connect to SAOImageDS9's SAMP hub:

```
py> from astropy.samp import SAMPIntegratedClient
py> ds9 = SAMPIntegratedClient()
py> ds9.connect()
```

- Issue either ds9.set or ds9.get to invoke DS9 API functions using the ecall_and_wait command. Note that all arguments are strings and must be in quotes. This example shows how to set up an RGB image.

```
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="rgb")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="rgb red")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="url http://ds9.si.edu/download/data/673nmos.f
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="zscale")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="rgb green")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="url http://ds9.si.edu/download/data/656nmos.f
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="zscale")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="rgb blue")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="url http://ds9.si.edu/download/data/502nmos.f
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="zscale")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="rotate 270")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="zoom to fit")
```

- Disconnect the DS9 client

```
py> ds9.disconnect()
```

Some more examples:
A 3D data cube GIF movie:

```
py> from astropy.samp import SAMPIntegratedClient
py> ds9 = SAMPIntegratedClient()
py> ds9.connect()
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="3d")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="url http://ds9.si.edu/download/data/image3d.fits")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="cmap viridis")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="movie 3d gif 3d.gif number 10 az from -90 az to 9
py> ds9.disconnect()
```

Use of the imexam task. This example illustrates the use of imexam, which queries the user to click the mouse and returns the coordinates at that location of the image in astropy:

```
py> from astropy.samp import SAMPIntegratedClient
py> ds9 = SAMPIntegratedClient()
py> ds9.connect()
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="url http://ds9.si.edu/download/data/img.fits")
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="zscale")
py> print('Click anywhere in image:')
py> coord = ds9.ecall_and_wait("c1","ds9.get","0",cmd="imexam wcs icrs")
py> print('Coordinate is ', coord['samp.result']['value'])
py> ds9.disconnect()
```

Displaying a 2D numpy array in DS9:

This example shows how to display the contents of a numpy array as an image within SAOImageDS9. First, create your numpy array and fill it with values. Next, create a numpy memmap array and copy the contents of the original array. Now flush the content to a file on disk. Next load the filename into DS9, it will display the initial contents. Now update the numpy memmap array with new values, and flush to disk. Use the command update to inform SAOImageDS9 to update the screen with the new data values. Continue this flush / update cycle. Be sure to clear the SAOImageDS9 frame before deleting the memmap file.

```
py> from tempfile import mkdtemp
py> import os.path as path
py> filename = path.join(mkdtemp(), 'newfile.dat')
py> import numpy as np
py> aa = np.random.rand(1024,2048)
py> fp = np.memmap(filename, dtype='float32', mode='w+', shape=(1024,2048))
py> fp[:] = aa[:]
py> fp.flush()
py> from astropy.samp import SAMPIntegratedClient
py> ds9 = SAMPIntegratedClient()
py> ds9.connect()
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="array "+ filename +"[xdim=1024,ydim=2048,bitpix=-
py> aa = np.random.rand(1024,2048)
py> fp[:] = aa[:]
py> fp.flush()
py> print("pause 1")
py> import time
py> time.sleep(2)
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="update")
py> print("pause 2")
py> time.sleep(2)
py> ds9.ecall_and_wait("c1","ds9.set","10",cmd="frame clear")
py> ds9.disconnect()
py> import os
py> os.remove(filename)
```

17.9 Web browser and VO

DS9 contains a simple web browser used to access documentation and other internally needed network documents.

Command:	**web**
Menu:	*Analysis - Web browser*
Description:	Starts a copy of the browser for use by the user, although it seems unlikely that this capability is useful on its own.

Command:	**web [url]**
Menu:	-
Description:	Opens web browser at given URL

Command:	**web new [tag] [url]**
Menu:	-
Description:	Opens new instance of web browser with given name

Command:	**web [tag] click [n]**
Menu:	-
Description:	Click n times

Command:	**web [tag] click back|forward|stop|reload**
Menu:	-
Description:	Web page navigation

Command:	**web [tag] clear**
Menu:	-
Description:	Clear given web browser

Command:	**web [tag] close**
Menu:	-
Description:	Close given web browser

DS9 capabilities provides access to some VO analysis. DS9 supports access to the Rutgers X-ray analysis server in this way.

Command:	**vo open**
Menu:	*Analysis - Virtual Observatory*
Description:	Brings up a VO dialog. After waiting some time the dialog is populated with VO options, currently the useful one is the Rutgers X-ray analysis. Select primary MOOC server; this brings up an internal web browser window with the Rutgers Chandra-Ed data. Clicking on one of the observation titles causes that data to be loaded into ds9.

Command:	**vo connect|disconnect chandra-ed**
Menu:	-
Description:	Connect or disconnect from the Rutgers Chandra-Ed server

Command:	**vo internal**
Menu:	*Edit - Preferences - VO - Browser - Use Internal Web Browser*
Description:	sets a default value;

Command:	**vo server [url]**
Menu:	*Edit - Preferences - VO - VO Server*
Description:	sets a default value giving the list of known servers.

Command:	**vo delay [value]**
Menu:	*Edit - Preferences - VO - Keep-Alive*
Description:	sets a default value of 15 minutes.

If you can't connect to VO services, it may be a problem with network firewalls. In this case you can use DS9's internal browser and a web proxy connection.

When you click on one of the Virtual Observatory servers in the VO list, DS9 will attempt to connect to that server and (if the internal Web display is enabled) display its Web page. The square box to the left of the server name turns yellow while the connection is being established and then green to signal success.

A direct connection is fast and flexible. Among other things, it allows you to perform analysis on your own local data (the VO server will retrieve the image from DS9) and also allows you to use an external browser to load images.

Some system managers configure their firewall explicitly to prevent computers in their care from making a direct connection to an external host. Instead, they only allow external access through a Web proxy server (such as SOCKS). If you are using a computer behind a restricted firewall of this sort, then DS9 will not be able to connect directly to a VO server. The yellow box will not turn green and eventually DS9 will display an error message.

In this case, you can choose to have DS9 communicate with the VO servers through your Web proxy server. DS9 will use your proxy to send its commands and retrieve its data and analysis results, rather than doing this directly. Note that the following restrictions apply:

- The transfer of data is slower.
- You must use the internal Web browser for loading images, etc.
- You cannot perform analysis on local data.

- There is a (large but finite) restriction on the number of annuli, and number of polygon points you can specify in a region, as well as the total number of regions allowed.

If your computer and firewall have been configured to require use of a Web proxy server, you will have to tell DS9 about this server. Click the Configure Web Proxy button and type the relevant information into the boxes. (Your systems administrator will be able to tell you the details.) At this point, you should be able to connect to a VO server successfully. Please let us know if you have problems!

A final note: you may, of course, choose to use the Web proxy even if your computer and firewall are configured to allow direct connections. In this case, there is no need to configure the proxy server.

17.10 IRAF support

DS9 retains some support for use within the classic NOAO image processing package IRAF. DS9 for Windows and MacOSX is also a fully functional IRAF display server. To direct image output from IRAF to DS9 running under Windows or MacOSX, use the IMTDEV environment variable, which must be set before entering IRAF. For example, if the machine is named 'foo.bar.edu', type:

```
$ setenv IMTDEV inet:5137:foo.bar.edu
$ cl
cl> display dev$pix
```

When DS9 starts, it will look for IRAF configuration files in the following order:

```
$HOME/.imtoolrc
/usr/local/lib/imtoolrc
$IMTOOLRC
$imtoolrc
```

If none of these are found it will use the following default configuration:

```
 1  2  512  512  # imt1|imt512
 2  2  800  800  # imt2|imt800
 3  2 1024 1024  # imt3|imt1024
 4  1 1600 1600  # imt4|imt1600
 5  1 2048 2048  # imt5|imt2048
 6  1 4096 4096  # imt6|imt4096
 7  1 8192 8192  # imt7|imt8192
 8  1 1024 4096  # imt8|imt1x4
 9  2 1144  880  # imt9|imtfs full screen (1152x900 minus frame)
10  2 1144  764  # imt10|imtfs35 full screen at 35mm film aspect ratio
11  2  128  128  # imt11|imt128
12  2  256  256  # imt12|imt256
```

```
13  2  128 1056  # imt13|imttall128 tall & narrow for spectro.
14  2  256 1056  # imt14|imttall256 tall & wider for spectro.
15  2 1056  128  # imt15|imtwide128 wide & thin for spectro.
16  2 1056  256  # imt16|imtwide256 wide & fatter for spectro.
17  2 1008  648  # imt17|imtssy Solitaire fmt w/ imtool border
18  2 1024  680  # imt18|imtssn Solitaire fmt w/out imtool border
19  1 4096 1024  # imt19|imt4x1
```

If the actual configuration of IRAF assumes a different buffer size, images displayed from IRAF will appear corrupted and trigger a series of error messages.

The IRAF graphcap file dev$graphcap must also be consistent with the imtoolrc file used; changes made to dev$graphcap must also be implemented in imtoolrc.

NOAO issued this note, hopefully now obsolete:

```
In a smooth installation the imtoolrc file is installed as a
/usr/local/lib/imtoolrc symlink pointing to the dev$imtoolrc file in the
iraf system.  This is normally what's used but XImtool (and DS9) also
allow a $HOME/.imtoolrc and IMTOOLRC environment variable defining the
path as fallbacks.  There are several practical problems with  this:
for some reason the imtoolrc link won't be created
if the /usr/local/lib directory doesn't exist when the install script is
run on the machine, even though it's run as root and the file can be
directory easily.  On PC-IRAF systems there is also a typo in the
install script (extra logical or at line 515) which causes it to exit
before the display setup is run (i.e. no /dev fifos or imtoolrc). If
users don't catch this or see it in the README file they'll think
everything went fine.
```

Note that DS9 uses a different approach to displaying images than the older saoimage and ximtool programs. An additional data buffer is used, which means that while with the older programs you see incremental progress in displaying an image, with DS9 you see nothing for a while and then the image is rendered all at once. The overall time to finish rendering is almost the same. (Nowadays with modern computers it shouldn't usually be an issue anyway).

On very old/slow machines, interactive control of the color gamma/bias is frustratingly slow in 24 bit mode. Instead of changing a color look up table, as in 8 bit mode, DS9 has to update every pixel on the screen. If your cpu speed is slow, you can select the Edit:Preferences:True Colorbar to tell DS9 not to update the entire screen, only a part of the screen.

Some relevant IRAF commands:

- **DISPLAY**:

  The IRAF DISPLAY task can be used to send an image to DS9. IRAF uses one of three communications protocols to do so: fifo, port (socket), and unix domain name. The defaults in DS9 are:

```
fifo  /dev/imt1
port   5137
unix   /tmp/.IMT%d
```

If your IRAF installation uses a different configuration, you can accommodate this on the DS9 command line with the following commands (these are only available on the command line and can't be used with XPA or SAMP):

Command: fifo [device]

Menu: -

Description: Start DS9 with a specified FIFO, e.g., ds9 -fifo /dev/imt2

Command: port [device]

Menu: -

Description: Start DS9 with a specified port, e.g., ds9 -port 5138

Command: unix [device]

Menu: -

Description: Start DS9 with a specified IRAF unix name, e.g., ds9 -unix /img/.IMT3

Command: fifo_only

Menu: -

Description: Starting with ds9 -fifo_only ignores the port and unix options.

Command: port_only

Menu: -

Description: Starting with ds9 -port_only ignores the fifo and unix options.

Command: unix_only

Menu: -

Description: Starting with ds9 -unix_only ignores the port and fifo options.

Note that when IRAF sends an image to DS9, it is IRAF that does the color scale distribution, and so in DS9 the *Scale* menu is not active.

In IRAF's Display, use the ztrans and z1,z2 parameters to set the upper/lower bounds and distribution. You can also use the zscale parameter to auto determine z1,z2.

```
ztrans=[linear|log|none|user]
z1=min
z2=max
zscale=[yes|no]
```

What actually is sent from IRAF to DS9 is one byte per pixel, values 0-200, which already has applied both the upper and lower clipping bounds and the distribution. So this is why the SCALE menu is disabled in DS9 when it receives a image from IRAF.

- **IMEXAMINE**:

  Note that if you have many multiple frames in DS9, IMEXAMINE can only address frames with frame numbers 1 to 9.

- **MSCZERO**:

  The mscred package task msczero uses the IRAF IIS image display protocol. But there is a minor issue with it. Before using msczero, type the following in the IRAF cl:

```
cl> set disable_wcs_maps=""
cl> flpr
```

18 Analysis menu files

18.1 Analysis menu files - use

Command: analysis load [filename]

Menu: *Analysis - Load Analysis Commands*

Description: Loads an external analysis menu in .ans format can be appended to *Analysis -*

Command: **analysis clear**

Menu: *Analysis - Clear Analysis Commands*

Description: Deletes the loaded menu. (Caution: this includes removing CIAO's DAX from the menu if in the CIAO environment).

Analysis files can be autoloaded with *Edit - Preferences - Analysis - Analysis File* entries, if *Edit - Preferences - Analysis - Analysis File - Autoload* is enabled.

If *Analysis - Analysis Command Log* is selected, or *Edit - Preferences - Analysis - Analysis Log - Show Command* is selected, the commands are echoed to the analysis log.

The DS9 CL allows additional capabilities not present in the GUI:

Command: **analysis task [n]**

Menu: -

Description: Run task [n] in th current analysis menu. The numbers are in order of task loading, starting with 0

Command: **analysis task [name]**

Menu: -

Description: Run task in the current analysis menu with the given name.

Command: **analysis message [text]**

Menu: -

Description: Append a message to the output

Command: **analysis text [text]**

Menu: -

Description: Append text to the output

18.2 Analysis menu files - syntax

Each file type known to DS9 can have user-defined analysis commands associated with it. These analysis commands are defined at start-up time , or loaded by the user, by means of an ASCII analysis description file. The analysis commands are available for execution, either via the Analysis Menu or the XPA point Analysis. In addition, commands may be bound to events, such as keystrokes or mouse clicks. This type of command is called a bind command.

DS9 searches for an analysis file named ds9.ans or ds9.analysis in the current directory and $HOME to be loaded at startup. In addition, DS9 will search the following directories for any analysis files to be loaded at startup in the form of *.ds9: the current directory, $HOME/bin, /usr/local/bin, /opt/local/bin, and /soft/saord/bin. Finally, the user may specify analysis files to be loaded at startup in the preferences analysis panel. The user may also load or clear current analysis commands via command line options or the Analysis menu.

When activated, either from the menu, XPA, or bound event, an analysis command first is macro-expanded to fill in user-defined arguments and then is executed externally. Results may be displayed in a separate text window, plot window, or in a image frame.

Task names may contain space characters. All lines may be indented. Also, the '#' character is a comment character. A separator can be inserted in the menu by specifying the following sequence '- - -'.

Example:

```
# this will insert a menu separator
---
```

The analysis file that defines the known analysis commands consists of comment and separator lines and one or more task descriptors, each of which has the following format:

- Line 1 (not including comment lines): Menu label (task name) to be used
- Line 2: A space separated list of file templates
- Line 3: Command type [menu | bind <event> | button]
- Line 4: The command line for the analysis program

The allowed command types are:

- **menu**

  A menu command creates an menu option under the Analysis menu option, and can be invoked by the user via the GUI or XPA.

  Example:

  ```
  # Menu command example
  My Analysis Task
  *.fits
  menu
  $data | doit | $text
  ```

- **button**

  A button command creates an button option in the current buttonbar.

  Example:

  ```
  # Button command example
  My Analysis Task
  *.fits
  button
  $data | doit | $text
  ```

- **web**

  A web command allows the user to invoke the internal web browser from the analysis menu.

  Example:

  ```
  # web command example
  HTTP based
  *
  web
  http://hea-www.harvard.edu/RD/ds9/ref/index.html

  File based
  *
  web
  file:/home/joye/index.html
  ```

18.3 Analysis files: Macros

The following macros are macro-expanded to fill in user-defined arguments before the command is executed. Strings that contain `$<macroname>` that the user does not want to be expanded may be escaped by using

```
$$<macroname>
```

All strings that contain

```
$<string>
```

that are not a macro name will not be affected.

For example:

```
 echo "$$data $foo" | $text
```

will display a text dialog that contains ”$data $foo”

The macros are:

- width, height, depth

  Substitute the width, height, or depth of the data file in the command line.

  Syntax:

  ```
  $width
  $height
  $depth
  ```

  Example:

  ```
  echo "$width $height $depth" | $text
  ```

- bitpix

  Substitute the bitpix of the data file in the command line.

  Syntax:

  $bitpix

  Example:

  ```
  echo "$bitpix" | $text
  ```

- data

  Data from the current frame becomes the input data to the command string. This data is in the form of a FITS image. This macro can only used at the beginning of the command string.

  Syntax:

  $data

  Example:

  ```
  $data | dosomething | $text
  ```

- entry

  Display an entry modal dialog. The returned string is substituted. If cancel is selected, the command line is not executed.

  Syntax:

  ```
  $entry(<message>)
  ```

  Example:

```
echo "$entry(Enter something here)" | $text
```

- dir

  Substitute the full path name of the directory where analysis file found.

  Syntax:

  $dir

  Example:

  ```
  echo "$dir" | $text
  ```

- env

  Substitute the value of a shell environment variable.

  Syntax:

  ```
  $env(<shell variable>)
  ```

  Example:

  ```
  echo "$env(PATH)" | $text
  ```

- filedialog

  Display the standard file dialog. Substitutes the returned pathname. Argument specifies if an open file or save file dialog is invoked.

  Syntax:

  ```
  $filedialog([open|save])
  ```

  Example:

  ```
  echo "$filedialog(open)" | $text
  ```

- filename

  Substitute the filename of the data file in the command line. A full filename includes any absolute or relative path. A root filename contains no path. A (2D) subsection defines a subimage from xmin,ymin to xmax,ymax. For Frame3D, a 2D subsection will also include the current slice (PLANE=) parameter if not 1. A 3D subsection defines a subimage from xmin,ymin,zmin to xmax,ymax,zmax and no PLANE parameter.

  Syntax:

```
$filename # filename with extname, (2d) subsections, filters
$filename(root|root,base) # root filename withwith extname,no subsections, no filters)
$filename(full|full,base) # full filenamewith extname,no subsections, no filters)
```

Example:

```
dosomething $filename | $text
dosomething $filename(root) | $text
```

- filename[regions]

  Combination of $filename and $regions macros. Generates a series of filenames, each with a region.

  Syntax:

  ```
  $filename[$regions]
  $filename[$regions(<options>)]
  ```

  Example:

  ```
  dosomething $filename[$regions] | $text
  ```

- geturl

  This macro differs from all other macros, including $url, in that no subprocess pipe is created. Only HTTP is supported. The contents of the url are retrieved and sent to $text, $plot, or $image. No other processing is allowed. The primary purpose of this macro is to support external analysis for the Windows platform, which has no subprocess support.

  Syntax:

  ```
  $geturl(http://<hostname>:<port>/<query>)
  ```

  Example:

  ```
  $geturl(http://foo.bar.edu/foo.html) | $text
  ```

- image

  The resulting image data is displayed in a DS9 frame. This macro should be the last macro of a command line. The optional parameter indicates whether a new frame - and what type of frame - is created for the new data. The macro is removed from the command line before execution.

  Syntax:

```
$image
$image([new|rgb|3d|current])
```

Example:

```
doit | $image(new)
```

- message

  Display a message dialog box, with option buttons. After displaying the message, the macro is removed from the command line before execution. If cancel or no is selected, the command line is not executed.

  Syntax:

  ```
  $message(<message>)
  $message([ok|okcancel|yesno],<message>)
  ```

  Example:

  ```
  $message(okcancel,This is a Message)| doit | $text
  ```

- messageok

  Display a message dialog box, with option buttons. After displaying the message, the button selected is substituted in the command line.

  Syntax:

  ```
  $messageok(<message>)
  $messageok([ok|okcancel|yesno],<message>)
  ```

  Example:

  ```
  echo "$messageok(okcancel,This is a Message)" | $text
  ```

- null

  Expect no output or results from analysis task. Note: no error message will be returned if the analysis task fails to execute correctly.

  Syntax:

  ```
  $null
  ```

Example:

```
echo "Hello, world" > foo | $null
```

- pan

  Substitute current pan location of the particular data file are returned. The default coordinate system is physical.

  Syntax:

  ```
  $pan
  $pan(<coordinate system>,<format>)
  ```

  where:

  ```
  coordinate system = [image|physical|detector|amplifier|wcs|wcsa...wcsz]
  sky frame         = [fk4|fk5|icrs|galactic|ecliptic]
  sky format        = [hms|sexagesimal|degrees]
  ```

  Example:

  ```
  echo $pan(fk5,sexagesimal) | $text
  ```

- plot

  Display data in plot window. This macro should be the last macro of a command line. The data is read via STDIN and consist of a pair of coordinates, with option error values. (xy, xyex, xyey, xyexey) Default dimension is xy. The macro is removed from the command line before execution.

  For $plot(stdin) only:

  The title, x axis label, and y axis label are assumed to be on the first line of input, delimited with a new-line. However, if the data starts with $BEGINTEXT, alltextbetween$BEGINTEXT and $ENDTEXTwillberemovedfromthedataanddisplayedinaseparatetextdialogwindow, withtheremainingdat an error is assumed to have occurred and a text dialog window is displayed only.

  Syntax:

  ```
  $plot
  $plot(,,,)
  $plot(<title>,<x axis label>,<y axis label>,[xy|xyex|xyey|xyexey])
  $plot(stdin)
  ```

Example:

```
doit | $plot(This is aTitle,X Axis,Y Axis)
doit | $plot(stdin)
```

- regions

  Substitute region definition in specified region format, coordinate system, and coordinate format. The default coordinate system is physical, default coordinate format degrees, and default region format DS9. Arguments may appear in any order, as long as they are separated by ',' and no spaces. If one or more properties are specified, only regions with all of the specified properties will be substituted.

  Syntax:

  ```
  $regions
  $regions(<options>)
  ```

  where options are one of the following:

  ```
  regions format    = [ds9|ciao|saotng|saoimage|pros|xy]
  property          = [include|exclude|source|background]
  coordinate system = [image|physical|detector|amplifier|wcs]
  sky frame         = [fk4|fk5|icrs|galactic|ecliptic]
  sky format        = [sexagesimal|degrees]
  ```

  also, the old SAOTNG formats are also supported:

  ```
  $regions_pixels
  $regions_degrees
  $regions_hms
  $include_regions
  $include_regions_pixels
  $include_regions_degrees
  $include_regions_hms
  $exclude_regions
  $exclude_regions_pixels
  $exclude_regions_degrees
  $exclude_regions_hms
  ```

  Example:

```
dosomething $regions | $text
dosomething $regions(pros) | $text
dosomething $regions(source,wcs,fk5) | $text
dosomething $regions(saotng,background,exclude,ecliptic,sexagesimal) | $text
```

- text

  Display text in a text dialog window. This macro should be the last macro of a command line. To display text from only STDOUT use '|' as the pipe command. To display text from both STDOUT and STDERR, use '|&' as the pipe command. No parameters are required. The macro is removed from the command line before execution.

  Syntax:

  $text

  Example:

  ```
  doit | $text # stdout
  doit |& $text # stdout and stderr
  ```

- url

  URLs are processed and stored in a temporary file. Only HTTP and anonymous FTP are supported.

  Syntax:

  ```
  $url(http://<hostname>:<port>/<query>)
  $url(ftp://<hostname>/<filename>)
  ```

  Example:

  ```
  $url(http://legacy.gsfc.nasa.gov/rosat/data/p000s26b.img.Z) | uncompress | $image
  $url(ftp://legacy.gsfc.nasa.gov/rosat/data/hri/images/rh100193_img.fits) | $image
  ```

- vo_method

  Returns the vo method.

  Syntax:

  ```
  $vo_method
  ```

  Example:

  ```
  echo '$vo_method' | $text
  ```

- value

  Substitute the value at the location of the cursor of an bind event.

  Syntax:

  ```
  $value
  ```

  Example:

  ```
  echo "$value" | $text
  ```

- x, y, z

  Substitute coordinates of an bind event. When a bind event is triggered, the x,y coordinates of the mouse of the particular data file are returned. The default coordinate system is physical. This macro is only available for bind commands. For datacubes, the z coordinate is returned based on the current slice selected.

  Syntax:

  ```
  $x
  $x(<coordinate system>,<format>)
  $y
  $y(<coordinate system>,<format>)
  $z
  $z(<coordinate system>)
  ```

  where:

  ```
  coordinate system = [image|physical|detector|amplifier|wcs|wcsa...wcsz]
  sky frame         = [fk4|fk5|icrs|galactic|ecliptic]
  sky format        = [hms|sexagesimal|degrees]
  ```

  Example:

  ```
  echo "$x $y" | $text
  echo "$x $y $z" | $text
  echo "$x(fk5,sexagesimal) $y(fk5,sexagesimal)" | $text
  echo "$x(wcs) $y(wcs) $z(wcs)" | $text
  ```

- xpa

  Returns the xpa access point name.

  Syntax:

  $xpa

  Example:

```
echo '$xpa' | $text
```

- xpa_method

  Returns the xpa method.

  Syntax:

  $xpa_method

  Example:

  ```
  echo '$xpa_method' | $text
  ```

18.4 Analysis files: Buttonbar

Creates an analysis buttonbar, in which button commands may be created. After all buttons commands have been defined, use 'endbuttonbar' to complete the definition.

Example:

```
buttonbar
Hello
*.fits
button
echo "hello" | $text
World
*.fits
button
echo "world" | $text
endbuttonbar
```

18.5 Analysis files: Help

The user may define their own HELP message. This message will be available to the user as a menu item. An optional label maybe specified. The default label is Help. When invoked, an text dialog window will appear, containing the message. Multiple HELP items may be defined within a menu, or across hierarchical menus.

Example:

```
help Main Help
A help message may contain
multiple lines of description of the tasks
in the menu or menus
endhelp
```

18.6 Analysis files: Parameters

The user may define his own macros or parameters to be evaluated before the command line is executed. To do this, the user defines a parameter segment that is referenced in the command line. Parameters maybe displayed in a notebook style via use of the 'tab' 'endtab' keywords. The parameter definition has the following format:

```
param <name>
  <variable> <entry|text|checkbox|menu|combobox|open|save> <title> <default> <{comment}>
  ...
endparam

param <name>
tab <text>
  <variable> <entry|text|checkbox|menu|comboxbox|open|save> <title> <default> <{comment}>
  ...
endtab
tab <text>
  <variable> <entry|text|checkbox|menu|combobox|open|save> <title> <default> <{comment}>
 ...
endtab
endparam
```

It can also use references to external parameter files:

```
param <name>
@<iraf param filename>
end
```

The definition either consists of a number of variables, one per row, or the name of a IRAF style parameter file. DS9 will look for the IRAF parameter file in:

```
./<filename>
$UPARM/<filename>
$HOME/iraf/<filename>
```

The following macros are supported if encounted in the default field:

```
$width
$height
$depth
$bitpix
$filename
$filename[region]
$env
```

Example:

```
param foo
  var1 entry {Variable 1} {default} {this is a entry}
  var2 text {Variable 2} {static} {this is text}
  var3 checkbox {Variable 3} 1 {this is a checkbox}
  var4 menu {Variable 4} {AAA|BBB|CCC} {this is a menu}
  var5 combobox {Variable 5} {XXX|YYY|ZZZ} {this is a combobox}
  var6 open {Variable 6} {default} {this is a open file dialog}
  var7 save {Variable 7} {default} {this is a save file dialog}
endparam

param bar
  tab {First Params}
    var1 entry {Variable 1} {default} {this is a entry}
    var2 text {Variable 2} {static} {this is text}
    var3 checkbox {Variable 3} 1 {this is a checkbox}
    var4 menu {Variable 4} {AAA|BBB|CCC} {this is a menu}
    var5 combobox {Variable 5} {XXX|YYY|ZZZ} {this is a combobox}
  endtab
  tab {Second Params}
    var6 open {Variable 6} {default} {this is a open file dialog}
    var7 save {Variable 7} {default} {this is a save file dialog}
  endtab
endparam

param foobar
  var1 entry {filename} {$filename} {image filename}
  var2 text {width} {$width} {image width}
  var3 text {height} {$height} {image height}
  var4 text {depth} {$depth} {image depth}
  var5 entry {bitpix} {$bitpix} {image bitpix}
endparam
```

To use parameters, specify the param name at the beginning of your command line:

```
Parameter Test
*
menu
$param(foobar); echo "$var1 $var2 $var3 $var4 $var5" | $text
```

When the menu item is selected, the user will be presented with a dialog box that contains entry, text, open, save, checkbox, or menu choices for each variable specified. If the user clicks ok, the values are substituted in the command line before execution.

18.7 Analysis files: Hierarchical Menus

The user may define hierarchical menus. Use this to organized crowded menus. To do this, frame menu entries with 'hmenu <label>' and 'endhmenu'. Hierarchical menu labels may contain spaces. Multiple levels may be implemented.

Example:

```
hmenu Stuff
  hello
  *
  menu
  echo "Hello" | $text

  world
  *
  menu
  echo "World" | $text

  hmenu More Stuff
    hello world
    *
    menu
    echo "Hello World" | $text
  endhmenu
endhmenu
```

The above create an hierarchical menu with two members, hello and world.

18.8 Sample analysis file

```
# Analysis command descriptions:
#  menu label
#  file templates
#  menu/bind
#  analysis command line

param foo
  var1 entry {Variable 1} {default} {this is a entry}
  var2 text {Variable 2} {static} {this is text}
  var3 checkbox {Variable 3} 1 {this is a checkbox}
  var4 menu {Variable 4} {AAA|BBB|CCC} {this is a menu}
  var5 combobox {Variable 5} {XXX|YYY|ZZZ} {this is a combobox}
  var6 open {Variable 6} {default} {this is a open file dialog}
  var7 save {Variable 7} {default} {this is a save file dialog}
endparam

param bar
  tab {First Params}
    var1 entry {Variable 1} {default} {this is a entry}
    var2 text {Variable 2} {static} {this is text}
    var3 checkbox {Variable 3} 1 {this is a checkbox}
    var4 menu {Variable 4} {AAA|BBB|CCC} {this is a menu}
    var5 combobox {Variable 5} {XXX|YYY|ZZZ} {this is a combobox}
  endtab
  tab {Second Params}
    var6 open {Variable 6} {default} {this is a open file dialog}
    var7 save {Variable 7} {default} {this is a save file dialog}
  endtab
endparam

param foobar
  var1 entry {filename} {$filename} {image filename}
  var2 text {width} {$width} {image width}
  var3 text {height} {$height} {image height}
  var4 text {depth} {$depth} {image depth}
  var5 entry {bitpix} {$bitpix} {image bitpix}
endparam

param barfoo
    @analysis.par
endparam

# Help Main Help

help Main Help
These menus contain a test for each possible feature

supported by the ds9 (blank line above)
endhelp
---

hmenu Test Web
```

```
  help Web Help
  Help for web features
  endhelp

  Web Test url
  *
  web
  http://hea-www.harvard.edu/RD/ds9/

  Web Test file
  *
  web
  file:/home/joye/saods9/ds9/tests/hv.html
endhmenu

hmenu Test Basics
  help Basic Help
  Help for basic features
  endhelp
  ---
  Test escape char # this is a comment
  *
  menu
  echo "this is not a macro $$xpa" | $text

  Test pass thru # this is a comment
  *
  menu
  echo "this is not a macro $foo" | $text

  Test $xpa # this is a comment
  *
  menu
  echo $xpa | $text

  Test $xpa_method
  *
  menu
  echo $xpa_method | $text

  Test $vo_method
  *
  menu
  echo $vo_method | $text

  Test $filename
  *.fits
  menu
  echo $filename | $text

  Test $filename(root)
  *.fits
  menu
  echo $filename(root) | $text

  Test $xdim $ydim $bitpix
  *.fits
  menu
  echo "$xdim $ydim $bitpix" | $text

  Test $xcen $ycen
  *.fits
  menu
  echo "$xcen $ycen" | $text

  Test $dir
  *
  menu
  echo $dir | $text

  Test $env
  *
  menu
  echo $env(PATH) | $text
endhmenu

hmenu Test Regions
  help Regions Help
  Help for regions features
  endhelp
  ---
  Test $regions
  *.fits
  menu
  echo "$regions ds9_s:$regions(ds9,source,image) ciao_b:$regions(ciao,background) saotng_i:$regio

  Test $regions wcs
  *.fits
  menu
  echo "$regions(ds9,wcs) $regions(ds9,wcs,fk5,sexagesimal) $regions(ds9,wcsa) " | $text

  Test $include_regions_pixels
  *.fits
  menu
  echo "ds9_s: $source_regions ds9_b: $background_regions_pixels ds9_i: $include_regions_degrees d

  Test $filename $regions
  *.fits
  menu
  echo "$filename[$regions]" | $text

  Test $filename $regions()
  *.fits
  menu
  echo "$filename[$regions()]" | $text
endhmenu

hmenu Test Output
  help Output Help
  Help for output features
  endhelp
  ---
  Test $null
  *
  menu
  echo "This is Text" >/dev/null | $null

  Test $text
  *
  menu
  echo "This is Text" | $text

  Test $text stderr
  *
  menu
  ls foofoofoo | $text
```

```
Test $plot
*
menu
cat xy.dat | $plot

Test $plot(title,x,y,xyey)
*
menu
cat xye.dat | $plot(Title,X Axis,Y Axis,xyey)

Test $plot(title,x,y,xyexey)
*
menu
cat xyee.dat | $plot(Title,X Axis,Y Axis,xyexey)

Test $plot(title,x,y,4)
*
menu
cat xyey.dat | $plot(Title,X Axis,Y Axis,4)

Test $plot(title,x,y,5)
*
menu
cat xyeye.dat | $plot(Title,X Axis,Y Axis,5)

Test $plot(stdin)
*
menu
cat xye.stdin.dat | $plot(stdin)

Test $plot(stdin) text
*
menu
cat xye.stdin.text.dat | $plot(stdin)

Test $plot(stdin) error
*
menu
cat xy.stdin.error.dat | $plot(stdin)

Test $data
```

```
  *.fits
  menu
  $data | $image(new)

  Test $image
  *
  menu
  cat img16.fits | $image
endhmenu

hmenu Test Dialogs
  help Dialogs Help
  Help for dialog features
  endhelp
  ---
  Test $message(message)
  *
  menu
  $message(ok,This is a Message) | echo "hello" | $text

  Test $message(ok,message)
  *
  menu
  $message(ok,This is a Message) | echo "World" | $text

  Test $entry(message)
  *
  menu
  echo "$entry(Enter Something)" | $text
endhmenu

hmenu Test Params
  help Param Help
  Help for param features
  endhelp
  ---
  Test $param
  *
  menu
  $param(foo);echo "$var1 $var2 $var3 $var4 $var5 $var6" | $text

  Test $param tab
  *
  menu
  $param(bar);echo "$var1 $var2 $var3 $var4 $var5 $var6" | $text

  Test $param macro
  *
  menu
  $param(foobar);echo "$var1 $var2 $var3 $var4 $var5" | $text

  Test $param @file
  *
  menu
  $param(barfoo);echo "$var1 $var2 $var3" | $text
endhmenu

hmenu Test Network
  help Network Help
  Help for network features
  endhelp
  ---
  Test $url(http://)
  *
  menu
  $url(http://legacy.gsfc.nasa.gov/FTP/rosat/data/cdrom/vol1/IMAGES/00h/p000s26b.img.Z) | gunzip

  Test $url(ftp://)
  *
  menu
  $url(ftp://legacy.gsfc.nasa.gov/rosat/data/hri/images/fits/rh100193_img.fits) | $image

  Test $geturl $text
  *
  menu
  $geturl(http://hea-www.harvard.edu/RD/saord-cgi/funtools?funcnts+$filename+$regions(source,,)+$r

  Test $geturl $plotstd
  *
  menu
  $param(ltc);$geturl(http://hea-www.harvard.edu/RD/saord-cgi/funtools?funhist_plot+$filename[$reg
endhmenu

hmenu Test Other
```

```
  help Other Help
  Help for other features
  endhelp
  ---
  hmenu Test MultiLevel
    test
    *
    menu
    echo "Hello World" | $text
  endhmenu
endhmenu

$x $y
*.fits
bind x
echo "$x $y" | $text

$x(fk5,hms) $y(fk5,hms)
*.fits
bind y
echo "$x(fk5,hms) $y(fk5,hms)" | $text

$x(wcs,fk5,hms) $y(wcs,fk5,hms)
*.fits
bind z
echo "$x(wcs,fk5,hms) $y(wcs,fk5,hms)" | $text
```

19 Troubleshooting errors

19.1 The temp directory

For a number of the Analysis functions, DS9 requires temporary disk space to download and store data. By default, this 'temp' directory is defined by the TMP or TEMP environment variable. This is usually defined as /tmp for Linux and MacOSX users. For Windows users, this will vary, depending on which version of Windows you have. In any case, if the temp directory is not writable, or you have specified an invalid directory in the preferences, these functions will fail with a variety of error messages.

Note that if your /tmp directory is mounted -noexec, binary table filtering does not work. Before starting DS9, set the environment variable FILTER_TMPDIR to a directory that is both writable and can execute.

19.2 Failure due to stripped binaries

If DS9 fails to start with the following error message:

```
Application initialization failed: Can't find a usable tk.tcl in the following directories...
```

It is possible that your system admin stripped the DS9 binary in an attempt to save disk space - but this breaks it (on Unix). DS9 is based on tcl/tk which is a scripting language which requires many support files. To create a stand alone application, we fool tcl/tk into thinking that it has a valid installation. To do this, DS9 is really a chimera of an application and an attached zip archive.. The first thing DS9 does on startup is to create a virtual file system in memory and unpack that archive into memory. The application DS9 is already stripped of debugging symbols when built. The system strip command can remove part of the zip archive, and hence DS9 is unable to un-compress it. In summary, don't strip the DS9 binary and everything works fine.

19.3 Failure due to invalid DATASEC keyword

The DATASEC keyword used in some FITS files specifies what portion of the image is valid data, for calculating min / max and for displaying. This is very important for images created from CCDs with over scan and bias strips. By default, this support is enabled. However, a number of fits images have invalid values for this keyword. Therefor, when DS9 opens the image, it finds no valid data to display - all the user sees is a white image. To correct this problem, either disable DATASEC support, via the Scale menu, or correct the the value of DATASEC in the fits header. You can also change the default behavior by disabling DATASEC from the preferences menu.

19.4 Startup file permission

The DS9 startup file ds9.ini must not be group/world writeable.

20 MacOSX and Windows compatibility

20.1 X11 and XQuartz

Up until MacOSX 10.8 (Mountain Lion), Apple provided their own version of a X11 server. At first, it was based on XFree86 (X11R6.6) and available with versions up to MacOSX 10.4. Later with MacOSX versions 10.5 to 10.7, the Apple's X11 server was based upon X.org (X11R7.2).

The Apple version of X11 server for MacOSX 10.5 to 10.7 contains a bug which fails if you invoke certain X11 calls on a window if its location is not at 0,0 on the screen. Hence, within DS9, if you 'Save Image' and your window is not exactly in the upper left corner, it will fail.

Starting with MacOSX 10.8, Apple no longer provides a X11 window server. The user must go to the XQuartz site and download/install directly. The current version is 2.7.3.

20.2 Command line startup in Aqua

If you start DS9 in MacOSX Aqua directly on the command line

```
% /Applications/SAOImage\ DS9.app/Contents/MacOS/ds9 foo.fits
```

you may get errors such as

```
The document "foo.fits" could not be opened. SAOImageDS9 cannot open files in the "Flexible Image
```

Instead, use the MacOSX open command, which sets up the user environment in the same was as when you double-click on an icon:

```
% open /Applications/SAOImage\ DS9.app foo.fits
```

20.3 Opening files in Aqua

By default, DS9 MacOSX Aqua uses the MacOSX standard dialog box to open and save files. This can be a problem in that the native MacOSX dialog will not allow arbitrary extensions to the file name, such as foo.fits[2]. You must use the Unix like standard dialogs to be able to specify an extension. Select `Edit->Preferences->General` to change the default standard dialog.

20.4 Mac: Setting up environment to use external analysis programs

In Unix to use external analysis programs, such as funtools, you set a PATH environment variable. But when you double click on a MacOSX application, it does not parse any shell startup files, such as /.profile. In older versions of MacOSX, the environment is defined using a special environment file, .MacOSX/environment.plist. This file could be created with the MacOSX utility /Developer/Applications/PropertyListEditor.app.

However, the environment.plist file is now deprecated and creating a property list file using the launchctl command now appears to be the recommended approach.

20.5 Windows compatibility notes

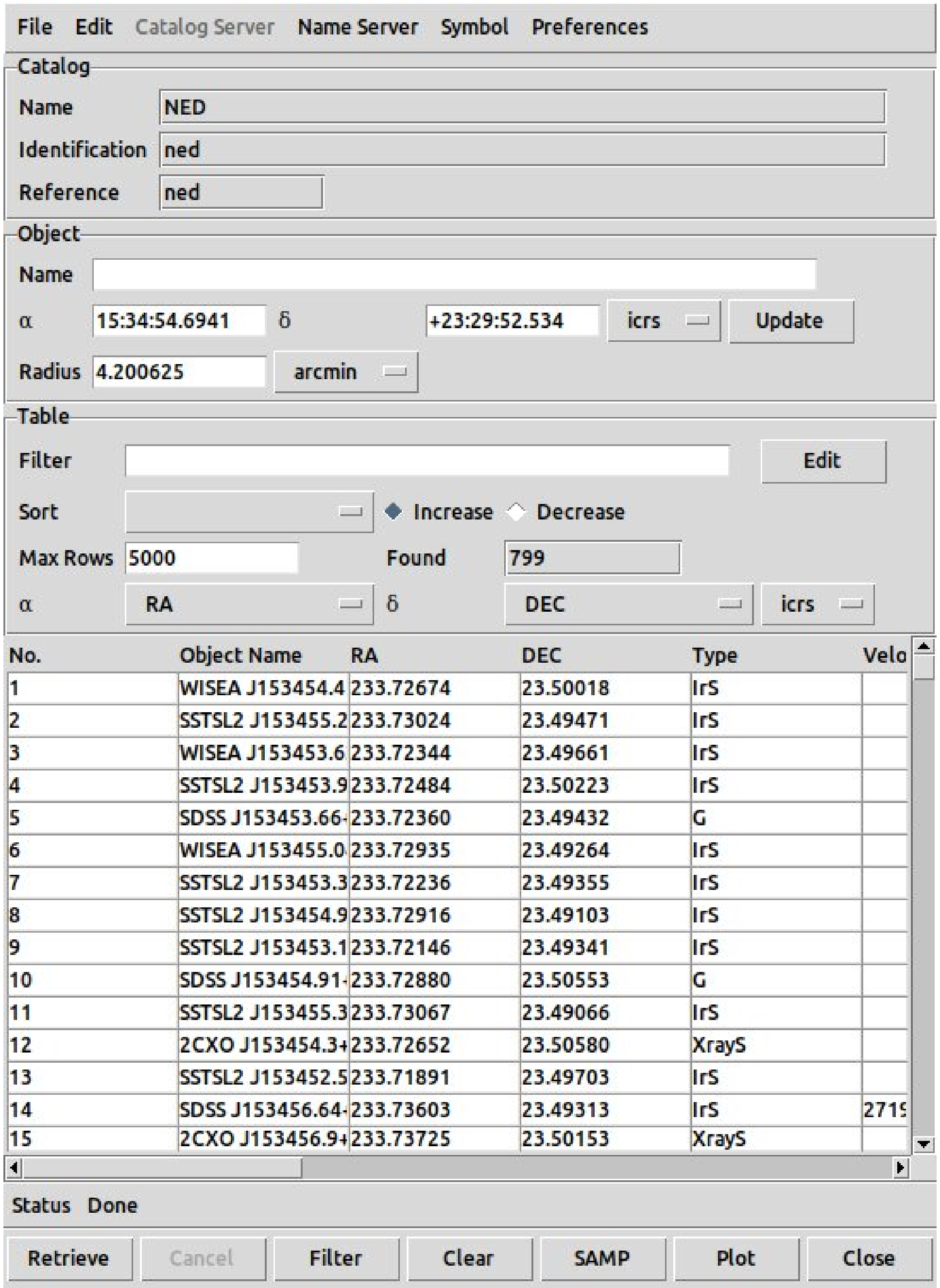
Some Windows issues:

- Note that running DS9 in Windows XP compatibility mode breaks 'Save Image' (only a small strip of the image is saved to the output jpg or png, etc.). To fix this, un-check the compatibility option in the properties dialog.
- By default, the windows port of DS9 uses the Windows standard dialog box to open and save files. This can be a problem in that the native Windows dialog will not allow extensions to the file name, such as foo.fits[2]. You must use the Unix like standard dialogs to be able to specify an extension. Select Edit-¿Preferences-¿General:Dialogbox to change the default standard dialog.

- In certain configurations, when you create an auxiliary window in DS9, such as a Pixel Table, or Analysis Plot, it will retreat behind the main DS9 window. Then, when you bring the auxiliary window to the front and move the mouse out of it, it automatically goes behind the main DS9 window again.

  To fix things so that the auxiliary window stays on top of the DS9 window, do the following:

  Go to the icon task bar at the bottom of the screen. Bring the auxiliary window to the front by clicking on its icon in the icon task bar. While the mouse still is on the aux window icon, press the mouse button, and keeping it pressed, move the mouse off the task bar. Release the mouse while off the task bar. The auxiliary window will now stay on top of the main DS9 window.



Catalog

Name: NED

Identification: ned

Reference: ned

Object

Name:

α: 15:34:54.6941 δ: +23:29:52.534 icrs

Radius: 4.200625 arcmin

Table

Filter:

Sort: ● Increase ○ Decrease

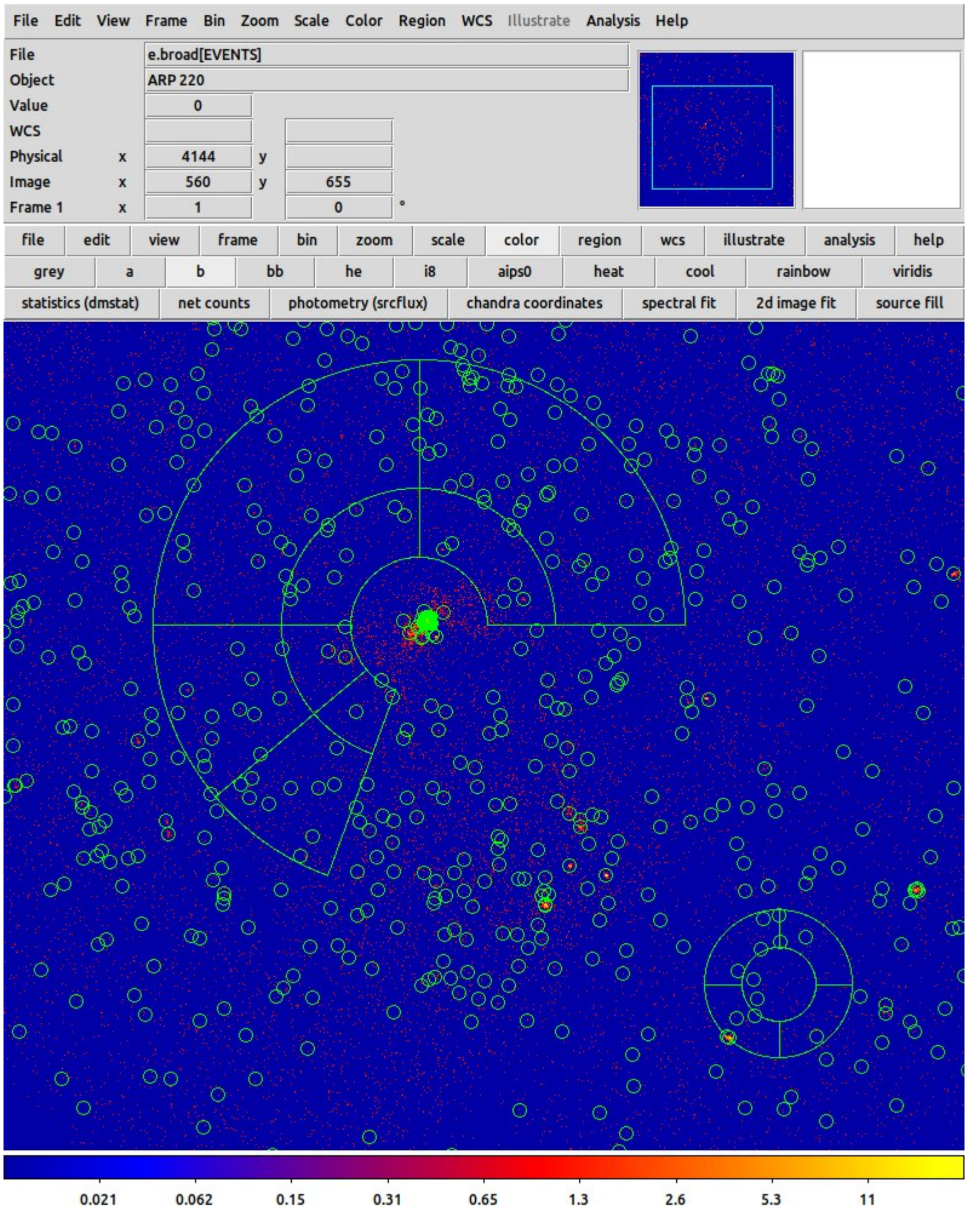
Max Rows: 5000 Found: 799

α: RA δ: DEC icrs

No.	Object Name	RA	DEC	Type	Velo
1	WISEA J153454.4	233.72674	23.50018	IrS	
2	SSTSL2 J153455.2	233.73024	23.49471	IrS	
3	WISEA J153453.6	233.72344	23.49661	IrS	
4	SSTSL2 J153453.9	233.72484	23.50223	IrS	
5	SDSS J153453.66-	233.72360	23.49432	G	
6	WISEA J153455.0	233.72935	23.49264	IrS	
7	SSTSL2 J153453.3	233.72236	23.49355	IrS	
8	SSTSL2 J153454.9	233.72916	23.49103	IrS	
9	SSTSL2 J153453.1	233.72146	23.49341	IrS	
10	SDSS J153454.91-	233.72880	23.50553	G	
11	SSTSL2 J153455.3	233.73067	23.49066	IrS	
12	2CXO J153454.3+	233.72652	23.50580	XrayS	
13	SSTSL2 J153452.5	233.71891	23.49703	IrS	
14	SDSS J153456.64-	233.73603	23.49313	IrS	271
15	2CXO J153456.9+	233.73725	23.50153	XrayS	

Status Done

Figure 37: Dialog resulting from selecting NED



File e.broad[EVENTS]

Object ARP 220

Value 0

WCS

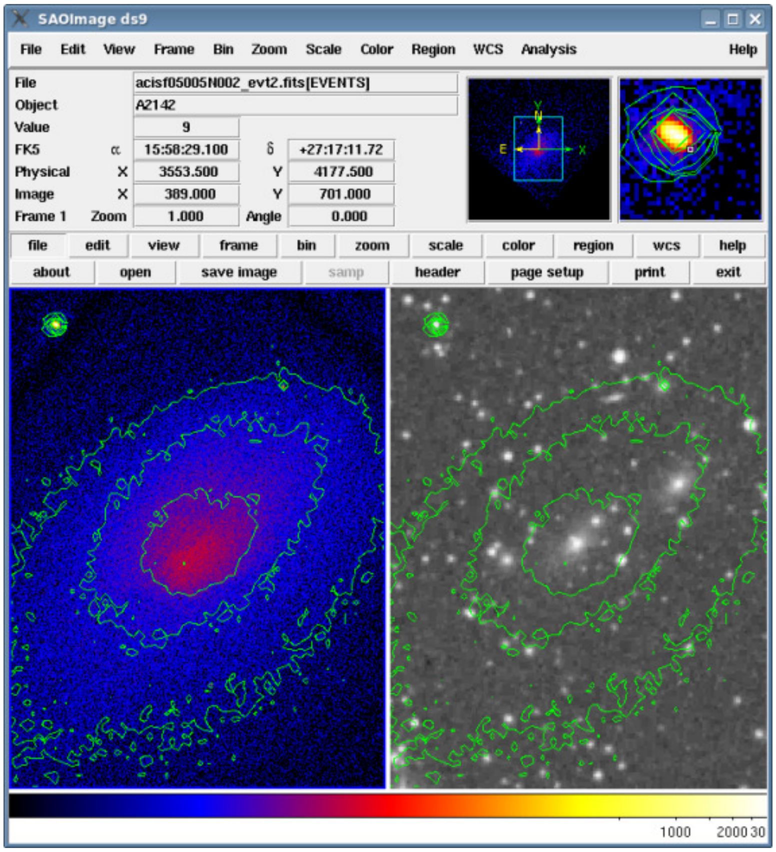
Physical x 4144 y

Image x 560 y 655

Frame 1 x 1 0 °

File Edit View Frame Bin Zoom Scale Color Region WCS Illustrate Analysis Help
file edit view frame bin zoom scale color region wcs illustrate analysis help
grey a b bb he i8 aips0 heat cool rainbow viridis
statistics (dmstat) net counts photometry (srcflux) chandra coordinates spectral fit 2d image fit source fill
0.021 0.062 0.15 0.31 0.65 1.3 2.6 5.3 11

Figure 38: Corresponding NED sources displayed on image


SAOImage ds9
File Edit View Frame Bin Zoom Scale Color Region WCS Analysis Help
file edit view frame bin zoom scale color region wcs help
about open save image samp header page setup print exit
N
E
X
Y
1000
2000 30

File acisf05005N002_evt2.fits[EVENTS]
Object A2142
Value 9
FK5 α 15:58:29.100 δ +27:17:11.72
Physical X 3553.500 Y 4177.500
Image X 389.000 Y 701.000
Frame 1 Zoom 1.000 Angle 0.000

Figure 39: Result of the second example contouring command line

Introduction to DS9 internals

This section of the document will describe the DS9 code for the benefit of future DS9 developers.

DS9 is a Tcl/Tk application. The GUI is implemented as a very thin layer of Tk. A number of Tk Canvas widgets in C++ were created to support all the functionality needed while computational work is done in C++. DS9 inherited SAOImageTNG's support of regions, XPA, external analysis support, and the general GUI. However, the visualization techniques come directly from the original SAOImage.

The current version of DS9 is composed of the Tk widgets created along with support from about 20 other open source products. The distributed binaries consist of a self-contained self-extracting archive and application, which provides an independent Tcl/Tk environment without installation.

The bulk of DS9 is written in a mixture of C/C++ and Tcl/Tk, with additional components using HTML, Roff, M4, yacc, and TeX. DS9 consists of about 728K lines of dedicated code, split about evenly between C/C+ and Tcl/Tk An additional 3 million LOC of OTS code are distributed with the build package (including the Tcl/Tk base libraries themselves).

The SAO-developed parts of DS9 are released under a GPL3 licence.

The DS9 code base supports three windowing systems: X11 (for Linux and MacOS); Aqua (Mac), and Windows. There are some minor packaging differences for the three platforms. Due to differences in available system libraries, precompiled binaries are created for 20 different variations of these platforms.

The DS9 code makes direct X library calls for its graphics. However on the Aqua platform, TK itself provides Aqua wrappers for the X calls, so our DS9 code can just pretend it is still using X11. Windows is supported similarly.

21 History of DS9 development

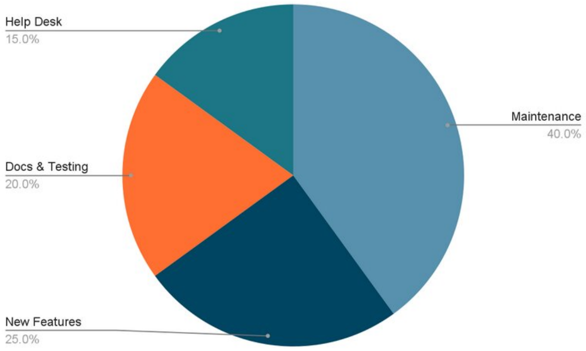
The first versions of DS9 were made available in 1999.

DS9 development was initially funded by a NASA Applied Information System Research Program grant: NAG5-3996, "Future Directions for Astronomical Image Display" with Eric Mandel as the Principal Investigator . The scope of work for this grant included, among other things, a set of portable and reusable widgets (frames, panner, magnifier, etc) that could be used to create custom astronomical imaging displays. DS9 then was the original product demonstration/reference implementation for this set of widgets [Joye & Mandel, 1999ASPC..172..429J] . However, DS9 was quickly adopted by the community and major observatories like NASA's Chandra X-ray Observatory and ESA's XMM Newton mission as the imaging software of choice. In the Final grant report, Mandel makes the following prophetic statement:

> With support for its further development being provided by SAO, we are confident that DS9 will soon become the de facto image display standard in the astronomical community.

Over the next decade or so, DS9 development continued with Bill Joye as the developer and funding coming from several sources including SAO's High Energy Department funds and NASA/GSFC HEASARC grants. Funding for 3D rendering was provided by the JWST Mission Office at the Space Telescope Science Institute [Beck, Joye, and Conti, 2012AAS...22013517B]. In 2015, the Chandra X-ray Center at SAO took over full funding of the DS9 project which had become a critical component in multiple Chandra subsystems: standard data processing Verification and Validation, production of the Chandra source catalog, observation planning, and end-user data analysis. Chandra continues to fund DS9 development; however, with major anticipated reductions in the Chandra budget, support for DS9 is expected to be terminated.

DS9 typically has 1 release per year in late summer/early fall, and there are typically two or three beta releases throughout the year. As the only developer, Joye handles all aspects of the project from coding bug fixes and enhancements, testing, providing user support, generating documentation and project status reports. A small group of scientists and engineers provides input on project priorities and assists with some testing. The current WBS for DS9 looks like the figure below.



Work on DS9 has been presented at the Astronomical Data Analysis Software & Systems (ADASS) conference series. A collection of papers is available in NASA's ADS:

https://ui.adsabs.harvard.edu/public-libraries/47Zk2IAvRoOI7VFXs4R3RQ

21.1 Code evolution

Some examples of past significant changes to the code that addressed evolving community needs include:

- FITS format. The interesting thing about 'standards' is that they change. While the original FITS standard was published in 1981, it has continued to evolve with the most recent major version, 4.0, being adopted in 2016/2018. Changes to the format include new data-types such as the 8-byte long, and new extension types such as compressed image formats.
- World Coordinate Systems (WCS). Not only has the format of FITS files changed, but the community established WCS conventions (meta-data) have also changed. New libraries needed to be integrated into the code to support the latest conventions.

- New detectors/missions means new requirements. The development of new hardware often precipitates a need for software changes. Trivial examples include needing to increase the numerical precision displayed to accommodate new milliarcsecond resolution instruments. More complex examples include support for missions such as JWST whose WCS cannot be expressed within the existing standards and requires non-standard processing. DS9 currently relies on the approximate WCS provided by JWST.
- Evolution of the Internet. From secure HTTP protocol to the demise of FTP these changes directly affect how DS9 interacts with online resources and how users obtain the software.
- Interprocess communication. A key element of the DS9 design was the ability for the application to communicate with other applications. Originally this has been through the "XPA" protocol which was designed and implemented under the same NASA grant that DS9 was developed. The astronomical community has now largely migrated to the "SAMP" protocol and much of the work on DS9 over the past few years has been directed towards this effort.
- User Hardware. Over the history of its development, DS9 has supported many different, some now defunct, operating systems: SunOS/Solaris, SGI/IRIX, IBM/AIX, HP/UX, DEC/Alpha, Linux, Windows, and Apple . It is also worth noting that it also has evolved from the days of 8-bit displays (limited to 256 colors and having to manage color tables) to today's 24/32-bit RGB/A displays.

21.2 Overview of Tcl/Tk

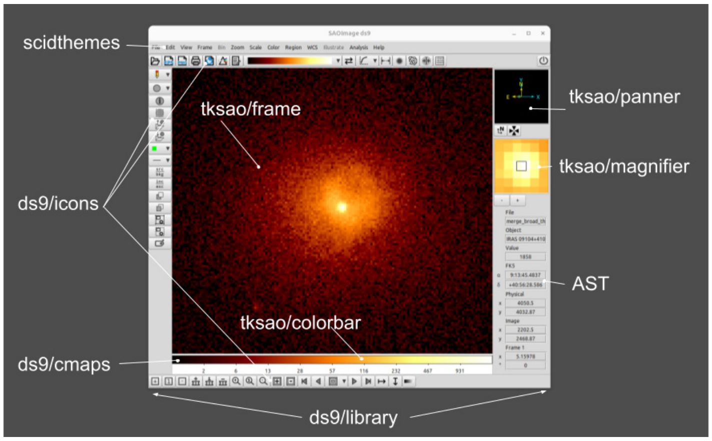
Tcl is a high level interpreted dynamic programming language; it is coupled with Tk, a graphical widget toolkit. Both packages are open source under a BSD-like licence and are written in C/C++ and in Tcl itself. Tcl/Tk was designed by John Ousterhout at Berkeley with a first full release in 1991. Tcl/Tk is currently developed by a distributed core team. SAO does not develop Tcl/Tk itself; we ingest the public release and distribute it with DS9 with only minor modifications.

21.3 DS9 code layout

DS9 is a Tcl/Tk application, with the GUI defined via Tcl/Tk code, but all of the real work is performed in C++. There are a number of parsers defined, written in YACC and LEX, for processing external data files such as regions files and colormaps, and all documentation is written in HTML.

All DS9 functionality is available via the GUI, or from the command line, SAMP, or XPA. Command line options, SAMP and XPA all support the same command syntax. SAMP is a xmlrpc based communication protocol, as defined by the International Virtual Observatory Alliance while XPA is an SAO-developed socket based communication protocol. Both are used extensively within the astronomical community.

The DS9 executable runs the main Tcl/Tk event loop which renders the DS9 user interface. Below is an example of the DS9 user interface with just some of the display elements labeled with their corresponding source code locations:



File
merge_broad_th
Object
IRAS 09104+410
Value
1858
FK5
α 9:13:45.4837
δ +40:56:28.586
Physical
x 4050.5
y 4032.87
Image
x 2202.5
y 2468.87
Frame 1
x 5.15978
° 0

Figure 40: The Tcl/Tk DS9/library module provides the top level window in which all the other elements are displayed. In this example the "Advanced" view is being used. This module also handles all the events such as clicking to create new regions or panning the frame. The image itself is a custom frame widget written in C/C++. Similarly the magnifier, panner, and colorbar are all custom widgets written in C/C++. The AST library is used to compute the celestial coordinates used in the information panel. The overall look and feel of the user interface (font color, size, style, spacing, etc) is provided by the base "ttk" widget classes; the scidthemes provides a collection of ttk styles.

21.4 DS9's zipfs internal file system

On linux, the DS9 program is downloaded from ds9.si.edu as a single file. This file actually contains two concatenated sections: the DS9 executable proper, and a compressed file system containing the scripts and libraries. When DS9 starts up, it mounts this compressed 'zipfs' file system in memory and accesses it.

The advantage of this approach is robustness for the user. The installation process is very simple: download a single file and put it somewhere in your path. It avoids complexities with unpacking distributions and with defining paths to libraries and shared files that might get moved somewhere the executable can't find them.

Paths in the internal file system are referred to in the code with a path of the form 'zipfs:/mntpt/foo'.

22 Building and Linking the DS9 executable

22.1 Downloading and building from source

The DS9 code is available on github. The source code assumes standard system libraries.

22.2 Source tree

The root of the source tree is a directory SAOImageDS9/. Underneath this are a series of directories that include both OTS and DS9 source, and some directories that handle operating system portability.

DS9 is built from a collection of 17 packages developed and/or now solely maintained by SAO. The DS9 project has taken over development of some projects such as 'funtools', 'xpa', and 'tkblt' since (a) they have been abandoned by their creators, (b) they provide essential functionality and (c) actively developed replacement packages do not exist. DS9 also relies on an additional 16 third-party packages which are still being actively maintained by the author/community.

The following table summarizes the different top level components of the DS9 code base as of DS9 v8.6.

We assign each component to one of three categories: SAO (the code was written at SAO); OTS (the code is off-the-shelf software, externally maintained, and external updates may require maintenance work on our end); and AOTS ('adopted OTS', the code used to be OTS but is no longer maintained and we now treat it as internal even though we didn't write it originally). Most AOTS code has undergone only minor maintenance since we took it over, but the BLT code has been signficantly updated. LOC for Makefiles and build macros are not called out in the table.

Table 1: Major components of the DS9 code

Component	C/C++	Tcl/Tk	Parser	Other	Category	Description
	LOC	LOC				
Core code						
ds9	689	97986	14929	0	SAO	Core DS9 GUI definition code, see next section
tksao	76880	0	9765	0	SAO	Defines the DS9 Frame, Colorbar, etc
Astronomy standards						
fitsy	14794	0	1526	0	SAO	FITS format I/O routines
tclfitsy	1044	0	0	0	SAO	TCL interface to fitsy
funtools	142772	25	0	0	SAO	FITS table filtering
ast	584510	0	0	0	OTS	Coordinate systems
Image, Data I/O						
tkimg	294027	8204	0	0	OTS	Image formats support
tkblt	30222	2762	0	0	SAO	Line graphics plots
tkmpeg	1341	0	0	0	AOTS/SAO	Write MPEG movies
tkagif	833	0	0	0	SAO	Write animated gifs
tktable	12809	1655	0	0	OTS	Table display
tksvg	5535	0	0	0	OTS	SVG graphics
Portability (mostly configure/make)						
unix	0	0	0	0	AOTS	Unix support
macos	0	0	0	0	AOTS	Mac support
tkmacosx	122	0	0	0	AOTS?	Macs support
win	0	0	0	0	AOTS	Windows support
tkwin	1285	0	0	0	SAO	Windows support
Look and feel						
awthemes	0	0	0	6852	OTS	Themes in TK
scidthemes	0	0	0	342	OTS	Themes in TK
ttkthemes	0	0	0	8234	OTS	Themes in TK
Communications						
xpa	24434	272	0	0	SAO	Interprocess control
tclxmlrpc	400	0	398	0	SAO	RPC support in TCL
openssl	440935	0	0	0	OTS	Access to remote data
tls	4422	746	0	0	OTS	TCL interface to openssl
Parser support						
fickle	0	1007	0	0	AOTS	TCL lexer
taccle	0	1726	0	0	AOTS	TCL parser
Infrastructure						
vector	1943	0	0	0	SAO	2D/3D vector library
tclzipfs	5181	0	0	0	AOTS	In-memory file system
tclsignal	635	0	0	0	AOTS	Tcl/Tk signal handler
tclxml	18121	12277	0	0	OTS	XML I/O for Tcl
tkhtml1	22324	433	0	0	OTS	HTML rendering
TCL/TK base						
tcl8.6	729908	45778	0	0	OTS	TCL language
tk8.6	306535	33079	0	0	OTS	TK language
tcllib	27083	375471	0	0	OTS	TCL utilities

tklib	1134	130316	0	0	OTS	TK utilities
tkcon	0	9910	0	0	OTS	TK console tool
tclconfig	0	0	0	0	OTS	Autoconf macros

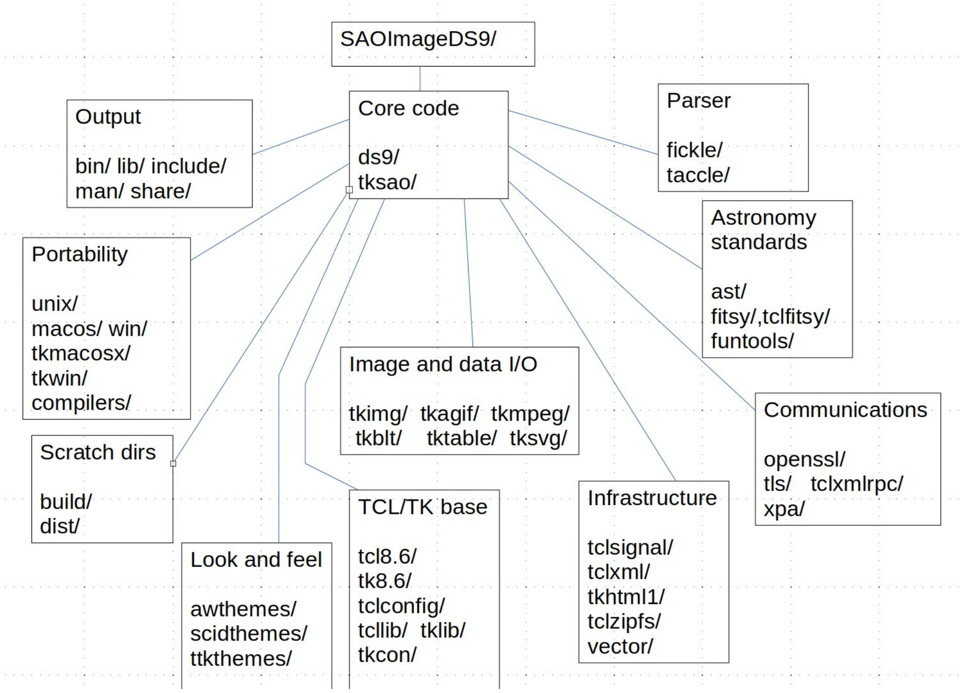


Figure 41: DS9 major software components

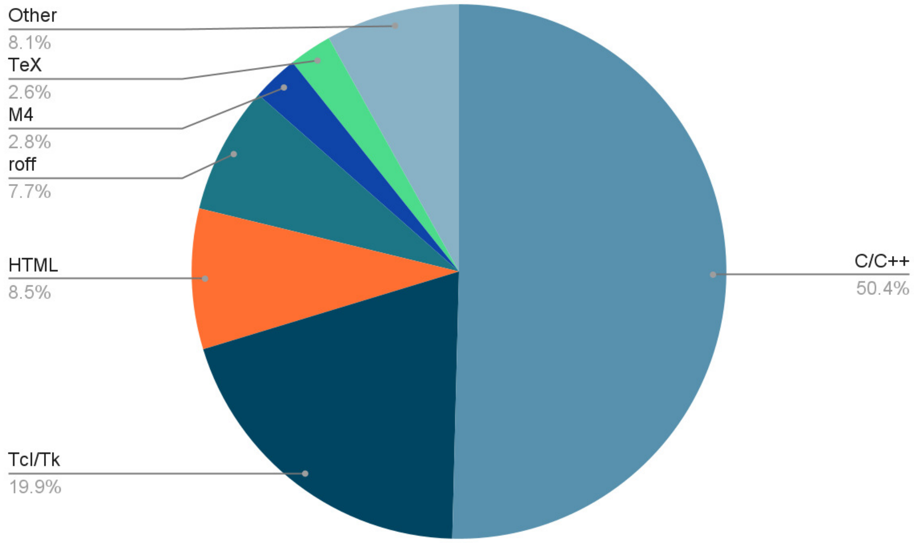


Figure 42: Distribution of languages used in the total code base. This includes the C/C++ code that makes up Tcl/Tk itself as well as supporting third party packages and the SAO developed code. TeX and HTML are text markup languages. Roff is the language used to define man pages. M4 is the language used for the configure (autoconf) file.

22.3 Structure of the DS9 core components

Here we unpack the DS9 and tksao components in a little more detail.

Table 2: Major sub-components of the DS9 core components

Component	C/C++ LOC	Tcl/Tk LOC	Parser LOC	Other LOC	Description
Core code					
ds9/library	0	93594	0	0	DS9 functions, including regions
ds9/parsers	0	3997	0	0	DS9 parsers - fcl source
ds9/parsers	0	221366	0	0	DS9 parsers - autogenerated tcl
tksao/saotk.C	105	0	0	0	Initialization code
tksao/colorbar	18098	0	0	0	Color bars
tksao/frame	101440	0	0	0	Image frames
tksao/magnifier	3809	0	0	0	Magnifier
tksao/panner	4651	0	0	0	Panner
Data					
ds9/cmaps	0	0	0	13153	Colormap tables
ds9/icons	0	0	0	(100 files)	Icons (png)
ds9/template	0	0	0	703	Instrument outlines
Portability					
ds9/macos	223	0	0	0	Portability
ds9/win	223	0	0	134	Portability
ds9/unix	320	0	0	0	Portability
ds9/msgs	0	0	0	8464	Localization
Infrastructure					
ds9/util	0	68	0	0	Message support
ds9/tclconfig	0	0	0	4071	Build wrappers
tksao/list	450	0	0	0	List class
tksao/tclconfig	0	0	0	4072	Build wrappers
tksao/tkutil	3582	0	0	0	Utilities (convolve etc)
tksao/widget	1730	0	0	0	Graphics widget classes
tksao/iis	1152	0	0	0	IRAF IIS RPC

22.4 Building and Linking the executable

22.4.1 Building and Linking the executable - Unix

- Note that on linux distributions, you may need to install libxml2-dev and libxslt1-dev in particular.
- The build process uses an environment variable EXE; you may wish to ensure this is not already set.

Then the following simple procedure will build DS9:

```
git clone https://github.com/SAOImageDS9/SAOImageDS9
cd SAOImageDS9
unix/configure
make
```

At the final stage of the Make process the makefile links the DS9 executable.

- First the ds9.zip Makefile target makes a zip file, ds9.zip, from the contents of ds9/library.
- Then, in the ds9/unix directory, the ds9 Makefile target creates 'ds9base' by

  ```
  g++ -pipe -g -Wall  -o ds9base ds9.o tkAppInit.o {lots of libraries}
  ```

- Now it combines these outputs by

  ```
  strip ds9base
  cat ds9base ds9.zip > ds9
  zip -A ds9
  ```

  This creates a final self-decompressing DS9 executable that is the concatenation of the ds9base linked executable with the ds9.zip directory tree of Tcl scripts, html files, libraries, etc. This is the file that you download and run.

22.4.2 Building and Linking the executable - MacOS/X11

Instead of concatenating the binary and zip file, they are left as two separate files.

22.4.3 Building and Linking the executable - MacOS/Aqua

A DMG installation package is created with both executable and support files included.

22.4.4 Building and Linking the executable - Windows

Cygwin is used to create a self-extracting zip executable that unpacks into a Windows application directory.

23 Running DS9

23.1 Starting DS9

The user runs the DS9 executable from the command line either as simply

```
bash> ds9
```

and then using the menus to select files to display.

They can alternatively specify an image file to display on the command line

```
bash> ds9 image.fits
```

Of course there are a plethora of command line options and configuration file choices that can do more sophisticated things, but usually users do one of the above two things.

What happens now in the code?

The main() Tcl/TK program is in ds9/unix/tkAppInit.c. It is compiled with symbols TK_LOCAL_APPINIT=SAOA and TK_LOCAL_MAIN_HOOK=SAOLocalMainHook which override the default TK interface with our home grown ones, so that the first subroutine to run is *SAOLocalMainHook* and this is followed by a call to Tk:

```
Tk_Main( argv, argc, SAOAppInit);
```

The first routine, *SAOLocalMainHook*, defines the startup script as libary/ds9.tcl (the version in the ds9.zip section of the executable, accessed via the zipfs virtual file system). It also tells TCL/TK to look for TCL and TK in zipfs:/mntpt/tcl8.6 and zipfs:/mntpt/tk8.6.

Next, *SAOAppInit* initializes zipfs using TclZipfs_Init. It then starts Tcl_Init and Tk_init, and initializes all the various subsystems (TKblt, Tktable, Tls, Tksao, Tkhtml1, Tclxpa, Tclfitsy, Tkmpeg, Tksvg, Tkagif, Tclxml, Tclxmlrpc, Tkimg, Zlibtcl, Jpegtcl, Tkimgjpeg, Tifftcl, Tkimgtiff, Tkimgwindow, and Signal_Ext.).

(Note that one of the routines called here is *Tksao_Init*, found in tksao/saotk.C, which creates DS9 frames, colorbars, panner, magnifier, etc.)

Then, *Tk_main* begins its event loop, taking commands from the ds9.tcl startup script.

ds9.tcl defines frams and menus, starts up the XPA and SAMP interprocess communication, loads preferences, and creates the main image frame. Finally it calls ‘wm deiconify $ds9(top)’ to raise the main window, and you are up and running with interactive DS9 driven by the TK event loop.

23.2 Startup problems

Note that if your machine is not connected to the internet, XPA startup and thus DS9 startup will hang, unless you have set the environment variable XPA_METHOD to ”local”.

On an X11 machine that is disconnected from the internet you may get the following error message

```
_X11TransSocketINETConnect: Can’t get address for foo.bar.edu
couldn’t connect to display "foo.bar.edu:0.0"
```

This means that DS9 is unable to determine a valid X11 Display server. You can fix this by doing the following:

```
$ xhost +
$ set DISPLAY=:0.0
$ export DISPLAY
$ ds9
```

23.3 Process flow: rendering a pixel value

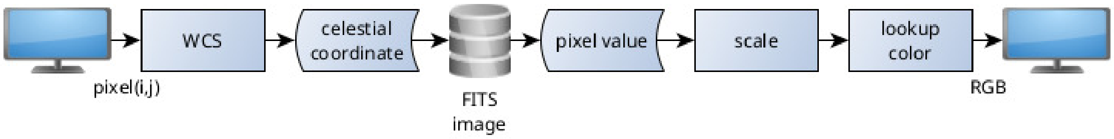


Figure 43: Process flow in pixel rendering

This figure illustrates the logical data flow necessary to render pixel values from a FITS image.

For each pixel (i,j) in the frame, DS9 will use the WCS to compute the celestial coordinates for that location. DS9 will then convert that celestial location to the closest pixel location in the FITS image. The pixel value at that location then has to be scaled to the range of colors in the user selected color map. The scaling depends on the type of data being imaged. X-ray images are usually logarithmically scaled whereas Optical images may use linear scaling. The user chooses the scaling. After scaling, the closest index in the color map is chosen. The color map provides a lookup table from scalar pixel value to a red, green, blue color tuple. This value is stored in the display frame buffer. Once all pixel values in the frame have been mapped, the display is updated from the frame buffer.

23.4 Process flow: drawing a circle region

I want to illustrate the flow of the code when a user takes a mouse action, and how that ends up causing astronomy related routines to be called, and then passing that back to cause the image to be updated. This is just one example.

We assume we are in edit region mode.

- **Event loop**

  In ds9/library/frame.tcl, routine BindEventsFrame, there is a line

  ```
  $ds9(canvas) bind $which <Button-1> \
      [list Button1Frame $which %x %y]
  ```

This tells DS9 to associate a Button1 press with the command Button1Frame on the current frame ($which) at the current pixel location (%x %y).

- **Parser**

  Further down in the same file we find ‘proc Button1Frame’ which implements this command. It looks at which mode you are in with a switch statement. We find the entry ”pointer - region” (in this language, the - is an ”or”: if the mode is ’pointer’ OR if the mode is ’region’, do this..)

  This then calls MarkerButton which is defined in ds9/library/marker.tcl. This looks to see if we are already in a marker, which we aren’t yet, so it falls through to the line

  ```
    MarkerCreate $which $x $y
  ```

  This looks at the current selected shape type which is ’circle’ and calls

  ```
   MarkerCreateShape $which $x $y
  ```

  MarkerCreateShape then defines a string called ’cmd’, starting with

  ```
  set cmd "$which marker create $marker(shape) $x $y"
  ```

  and appending parameters to the string depending on the shape type, as well as other properties. Later it does

  ```
  set imarker(id) [eval $cmd]
  ```

  which saves this command string for execution.

  When the command is executed, the parser in tksao/frame/lex.L sees the strings ’marker’ and ’create’ and returns the tokens MARKER_ and CREATE_.

  Then the parser tksao/frame/parser.Y maps MARKER_ and CREATE_. to ’marker :’ in the same file and then ’markerCreate :’. This tests the shape type somehow and selects CIRCLE_. The CIRCLE_ option in the markerCreate parser entry calls the C++ routine createCircleCmd:

- **Marker code:**

  ```
  {fr->createCircleCmd(fr->mapToRef(Vector($2,$3),Coord::CANVAS),
              $4,
              currentColor,currentDash,currentWidth,currentFont,
              currentText,currentProps,NULL,taglist,cblist);}
  ```

  This routine is found in tksao/frame/frmarker.C; it instantiates a Circle object with ’new Circle’ and passes this to the createMarker routine, found in the same file at the line

```
Marker* Base::createMarker(Marker* m)
```

this calls ‘update’ to update the pixmap and return a pixel value of some kind.

- **Making the calls to X**

  Somewhere in ’update’, a call is made to draw the circle which reaches tksao/frame/circle.C and the routine Circle::Circle, which calls BaseEllipse, defined in tksao/frame/baseellipse.c. This in turn calls marker.C:BaseMarker and thus marker.C, which returns the marker object with its values initialized.

  circle.C contans the routine Circle:renderXArcDraw which, like some other routines in this file, makes direct X calls, in this case XDrawArc.

  Base::updateBase, Base::updatePixmap draws the pixmaps etc.

  tksao/frame/base.C Base:updatePixmap is an important routine; Note the fall throughs in the switch statement. DS9 sets various flags as it goes of things that need to be updated.

  If you only need to update the pixmap that’s easier, (updateMagnifier and updatePM) but you might need to call BASE as well which also calls updateBase, updatePanner; and in the worse case of case MATRIX you first need to update all the coordinate matrices (computationally expensive) and then do the BASE and PIXMAP updates as well.

  The updatePM routine will make an XCopyArea call for the pixmap update. The actual flushing of the X buffer to the screen is done by the Tcl/TK event loop itself.

24 Historical and descriptive material

24.1 The Story of SAOImageDS9: How DS9 got its name

In 1990, Mike Van Hilst, at the Smithsonian Astrophysical Observatory, Center for Astrophysics, Harvard University, developed SAOImage. SAOImage was first implemented in X10, then reimplemented in X11. In fact, it was one of the first X11 based applications publicly made available. SAOImage was a brilliant program, implementing techniques in scientific visualization 20 years ago that are still being used by today’s applications. Since Mike’s departure from SAO, SAOImage has been maintained by Jessica Mink.

In the mid 1990s, with the administrative support of Steve Murray, Eric Mandel developed SAOtng, or (SAOImage, The Next Generation), named after the Star Trek series. TNG was based on IRAF’s XIMTOOL graphics libraries and Tcl. It explored new GUI interfaces and supported a new external analysis interface. In particular, it utilized XPA, (X11 Public Access, also written by Eric) which allowed TNG to be scripted via a shell, or from other application.

In 1998, while working with Eric, William Joye began a complete rewrite of TNG, based on the experience developed while supporting TNG. This project was funded by the NASA Applied Information Systems Research Program, under the title ”Future Directions for Astronomical Image Display”. For lack of a name, the new project was referred to as DS9, the logical extension of the

Star Trek series. The name continues to be in use. Current funding is provided by the NASA High Energy Astrophysics Science Archive Center and the Chandra X-ray Science Center.

DS9 is a Tcl/Tk application. The GUI is implemented as a very thin layer of Tk. A number of Tk Canvas widgets in C++ were created to support all the functionality needed. Basically, all the real work is done in C++. DS9 inherited TNG's support of regions, XPA, external analysis support, and the general GUI. However, all the visualization techniques come directly from SAOImage.

The current version of DS9 is composed of the Tk widgets created along with support from about 20 other open source products (including Tcl/Tk, AST, BLT, HCompress, HTMLWidgget, plio, rics, tcllib, tclxml, tkcon, tkimg, tktable, wcssubs, xmlrpc, XPA, zip, zlib, and zvfs). The distributed binaries consist of a self-contained self-extracting archive and application, which provides an independent Tcl/Tk environment without installation.

The first versions of DS9 were made available in 1999. Since then, the popularity of DS9 has grown far beyond expectations.

Development funding was provided by NASA's Applied Information System Research Program (NASA/ETSO), Chandra X-ray Science Center (CXC) and the High Energy Astrophyisics Science Archive Center (NASA/HEASARC).

24.2 SAOImageDS9 Acknowledgement

The authors of DS9 are William Joye, Eric Mandel and Steve Murray (deceased), all at the Smithsonian Astrophysical Observatory.

DS9 development has been made possible by funding from the Chandra X-ray Science Center (CXC) (NAS8-03060) and the High Energy Astrophysics Science Archive Center (HEASARC) (NCC5-568). Additional funding was provided by the JWST Mission office at Space Telescope Science Institute (NAS-03127) to improve capabilities for 3-D data visualization.

If you have found DS9 to be helpful in your research, we requiest that you acknowledge its use in your publications. The recommended text is:

”This research has made use of SAOImageDS9, developed by Smithsonian Astrophysical Observatory”

When possible, also cite the following paper:

Joye, W.A., & Mandel, E., 2003, *Astronomical Data Analysis Software and Systems XII*, ASP Conference Series, Vol. 295, p. 489 (2003adass..295..489J).

As of 2025, DS9 continues to be actively suppported. For questions, bug reports, or feedback, please contact the helpdesk at cxchelp@cfa.harvard.edu.

Command:	about
Menu:	*Help - About SAOImageDS9*
Description:	Displays some of the above information.

24.3 SAOImageDS9 Copyright

DS9 is composed of approximately 20 open source packages, all of which are distributed under their own open source license agreements, usually GPL, LGPL, or BSD. In addition, several open source packages have been developed at the Smithsonian Astrophysical Observatory, Cambridge, MA, USA and are distributed under the terms of the GNU General Public License as published by the Free Software Foundation. As long as you continue to adhere to the provisions of the licenses, you are free to distribute DS9 along with your software.