NAME

vosamp − command-line SAMP utility task

SYNOPSIS

vosamp [-t to] [-p pattern] [-f file] <cmd> [args ...]

OPTIONS

The *vosamp* application accepts the following options:

−h, --help
Print a help summary to the terminal and exit. No processing is done following this flag.

-i, --interact
Interactive mode. If enabled, a command prompt will be printed allowing users to enter commands interactively until an <EOF> is encountered.

-m, --many
Handle multiple messages when in listening mode. If not enabled, *vosamp* can be called to wait for a specific message and will exit when it is received, otherwise the task will continue to run and process multiple messages.

-s *SENDER*, **--sender** *SENDER*
Handle only messages from *sender*.

-q, --quiet
Suppress all output.

-t *TO*, **--to** *TO*
Send to specified app (or all apps if not given). The *TO* argument may be given as either an application public ID or the application name.

-p *PATTERN*, **--pattern** *PATTERN*
Messaging pattern to use when sending messages. The default mode is 'async' to send messages asychrnously, other allowed values are 'sync' for syncronous messages and 'notify' to broadcast without expecting a response.

-f *FILE*, **--file** *FILE*
Send all commands in the *FILE* argument. This mode allows the task to take command input from a text file to process multiple commands with a single invocation.

-k, --nokeepalive
Disable *keep_alive* feature of the task. If the *-k* flag is used, a separate connection to the SAMP Hub will be made for each command processed.

-P *IP*, **--proxy** *IP*
Use specfied IP as the proxy connection. See the discussion below about the *keep_alive* feature for details on how to use this flag.

-T *N*, **--timeout** *N*
Keepalive timeout in seconds. If no new command is received after *N* seconds the application will disconnect from the Hub automatically.

-S *NAME*, **--session** *NAME*
Name of the SAMP session. See below for a description of sessions.

DESCRIPTION

The *vosamp* task may be used to send and receive SAMP (Simple Application Messaging Protocol) messages from the command-line or within a script. It provides a user-friendly command interface that hides the details of the message construction and delivery for common tasks. By default, a message will be broadcast to all other SAMP-enabled applications, the *-t* (or *--to*) flag can be used to name a specific receipient by either the public ID or the application name, the *-p* flag will accept a 'sync' or 'notify' argument to change the default message pattern of asynchronous delivery.

In order to minimize the overhead of connecting with the Hub on each command, *vosamp* will spawn a proxy process that remains connected to the Hub and will process subsequent commands transparently. This proxy process will timeout after some period of inactivity and may be accessed from remote machines (see below for more information).

COMMAND SUMMARY

The *vosamp* task accepts the following commands, specified either on the command-line argument list or in interactive mode:

snoop
Print all received messages.

send <mtype> [<args>...]
Generalized message send. The <mtype> parameter can be either one of the well-known SAMP mtypes or an ad hoc mtype that can be expected to be recognized by apps in the session. The <args> parameter refers to any of the arguments necessary for the specifed mtype.

status
Print Hub availability.

list
List all registered clients.

access <appName>
Print <appName> availability. The *appName* may be specified as either the public ID or application name.

handle <mtype>
Wait for <mtype> message to be received. This command will subscribe the task to the specified <mtype> and then wait until it is received.

load <url>
Load the image or table given by <*url*>. The type of file and the appropriate SAMP *mtype* are determined automatically.

loadImage <url>
Load the named image. The *image.load.fits* mtype will be used for the message.

loadVOTable <url>
Load the named VOTable. The *table.load.votable* mtype will be used for the message.

loadFITS <url>
Load the named FITS bintable. The *table.load.fits* mtype will be used for the

message.

loadSpec <url>
Load the named spectrum. The *spectrum.load.ssa-generic* mtype will be used for the message.

In the above commands, the <*url*> may be an explicit URI containing an 'http' or 'file' prefix, if a filename or directory path is specified the URL will be constructed internally when sending the message.

pointAt <ra> <dec>
Point at given coords. The <ra> and <dec> parameters are assumed to be ICRS coordinates in decimal degrees, the *coord.pointAt.sky* mtype is used.

showRow [<url>] [<id>] <row>
Highlight specified <row> (zero-indexed). The table may be specified using either a <url> or a table <id> if one was specified at the time the table was loaded, the *table.highlight.row* mtype is used.

selectRows [<url>] [<id>] <rows>
Select specified rows. (zero-indexed) The table may be specified using either a <url> or a table <id> if one was specified at the time the table was loaded, the *table.select.rowList* mtype is used. The <rows> argument is specified as a comma-delimited list of row numbers or ranges, where *ranges* are hyphen-delimited strings (e.g. "1,3,5-9,11-15").

bibcode <bibcode>
Load the specified bibcode. The *bibcode.load* mtype is used.

exec <cmd>
Execute a client command. The <cmd> string is sent to the client unchanged, it is up to the client to interpret the command properly. The *client.cmd.exec* mtype is used.

setenv <name> <value>
Set an environment value. The *client.env.set* mtype is used.

getenv <name>
Get an environment value. The value of the requested variable is printed. The *client.env.get* mtype is used.

setparam <name> <value>
Set a parameter value. The *client.param.set* mtype is used.

getparam <name>
Get a parameter value. The value of the requested variable is printed. The *client.param.get* mtype is used.

session list
List nodes in current session.

session leave|exit|logout

Leave the current session.

session <name>
Join the named session.

KEEP-ALIVE CONNECTIONS AND SESSIONS

In the standard SAMP interaction, and application is required to first register with the *Hub* before sending or receiving messages. This registration can add significant overhead to an application that may only send a single message, significantly slowing it's use within a scripting environment. Unless the *-k* (or *--nokeepalive*) flag is set, the first time VOSAMP is started it will execute the specified command and then fork a child process that stays connected to the Hub. Subsequent VOSAMP calls will simply forward the command to this child proxy process, thereby avoiding a new Hub registration.

The proxy process by default will listen on inet port 3999 (as of this writing there is no option to change it) for new commands, however there is no restriction that the only application that can connect to it must be running on the same host. The *-P* (or *--proxy*) flag can be used to specify an alternate proxy to be used; the argument is of the form

node [*':'* *port*]

where *node* can be a simple host name, a fully-qualified domain name or and IP address, and *port* number number is optional. The proxy will run for up to an hour if no new commands are received before disconnecting from the Hub, this timeout value may be changed by using the *-T* flag to specify the timeout in seconds.

SAMP *Sessions* are an experimental concept in which *vosamp* tasks on separate machines register independently with their local Hub, but also register with the *vosession* task running elsewhere. Commands which are sent to the local child proxy are also forwarded to the *vosession* instance and then on to other vosamp clients registered in the same session. The *session* commands allow a vosamp client to register with a new session, list the other machines involved in the session and later leave the sessions. Commands which require a URL will upload local data to the *vosession* manager and rewrite the URL appropiately so it may be served to the other vosamp clients. Since SAMP is designed only for local desktop messaging, this approach provides a means to share messages and data between desktops (e.g. when working collaboratively with colleagues). See the *vosession* man page or contact the author for further information on using this feature.

RETURN STATUS

On exit the **vosamp** task will return a zero indicating success, or a one indicating an error.

EXAMPLES

1) Load a VOTable to Topcat:

```
% vosamp load /path/example.xml
% vosamp load http://foo.edu/example.xml
% vosamp load http://foo.edu/query?RA=0.0&DEC=0.0&SR=0.1
```

2) Send a command string to IRAF:

```
% vosamp -t iraf exec "display dev$pix 1"
```

3) List all clients in a SAMP session:

```
% vosamp list
```

4) Check whether a Hub is available from a script:

```
vosamp access Hub
if ($status == 1) then
  echo "No Hub available, quitting ....."
  exit $status
endif
```

BUGS

No known bugs with this release.

KNOWN SHORTCOMINGS

No known bugs with this release.

- The 'handle' command should allow a command to be executed with message argument substitution.
- A flag is needed to change the child proxy port being used

Revision History

Feb 2013 - First public release

Author

Michael Fitzpatrick (fitz@noao.edu), Feb 2013

SEE ALSO

vosession

The description of commonly used SAMP mtypes is gen at

http://wiki.ivoa.net/twiki/bin/view/IVOA/SampMTypes