NAME

vximtool –- A virtual display server for IRAF IIS protocol clients

SYNOPSIS

vximtool [*options* ...]

DESCRIPTION

VXIMTOOL is a image display server process much like *XIMTOOL*, except that all it normally does is respond to datastream requests to read and write to internal frame buffers maintained as arrays in memory. Multiple frame buffers and frame buffer configurations are supported. It can be used to debug IIS protocol client programs by printing out the protocol packets recieved, or can simply be used as a dummy server in cases where no image display is really needed.

The *-verbose* flag will log all datastream requests to the stderr, otherwise the task runs silently except for error messages. The process is terminated with an EOF on the stdin, if the task is intended to be run as a background job the *-background* flag should be enabled to shut off the check of the stdin and avoid a "waiting for tty input" message that would suspend the background job. Frame buffers are maintained as rasters in memory, up to four frames may currently be defined. If it not necessary to keep the displayed images in memory (i.e. the client will not need to read back any subrasters such as when doing overlay graphics), the *-noraster* flag can be used to disable the memory rasters. In this case any image readback will return an array of zero values.

The default frame buffer size is 512x512, with 2 frames defined initially. The *-config* and *-nframes* flags can be used to change the startup frame buffers used. The frame buffer configuration table is normally taken to be /usr/local/lib/imtoolrc but can be overridden by using an IMTOOLRC environment variable, a *$HOME/.imtoolrc* file, or the *-imtoolrc* command-line flag respectively. The format of the frame buffer configuration file is

configno nframes width height [extra fields]

e.g.

```
1  2  512  512
2  2  800  800
3  1 1024 1024          # comment
```

and so on. At most 128 frame buffer sizes may be defined, each configuration may define up to 4 frames, configuration numbers need not be sequential.

OPTIONS

-background
The task will be run in the background so don't listen to STDIN for an EOF to quit gracefully.

-config *num*
Initial frame buffer configuration number. The default value is 1, indicating a 512x512 frame buffer with 2 frames. See below for information on the frame buffers.

-fifo *pipe*
Specifies the name of the fifo pipe to be used, the *i* and *o* suffixes will be added automatically. The default pipe names will be /dev/imt1i (input pipe) and /dev/imt1o (output pipe).

-fifo_only
If set, only fifo pipes will be used for communication with a client program, sockets will be disabled.

-help Print a summary of command line options to the screen.

-interactive

Allow cursor value strings to be typed into the STDIN in response to cursor read requests from the client.

-imtoolrc *file*
Specifies the frame buffer configuration file to be used. See below for information on frame buffers.

-inet_only
If set, only inet sockets will be used for communication with a client program, fifo pipes and unix sockets will be disabled.

-noraster
If set no client data will be stored in memory, and image readback will return an array of zeroes.

-nframes *num*
Specifies the number of frame buffers to configure at startup. By default there will be 2 frames available, a maximum of 4 frames are allowed.

-port_only
Same as *-inet_only* option. If set, only inet sockets will be used for communication with a client program.

-port *num*
Set the primary port to listen on for client connections. The default is 5137.

-verbose
If set, print information about all IIS packets headers recieved to the STDERR stream.

-unix *name*
Specifies the unix domain socket name to use. A "%d" in the filename will be replaced with the user id.

-unix_only
If set, only unix domain sockets will be used for communication with a client program, inet sockets and fifos will be disabled.

CLIENT CONNECTIONS

VXImtool allows clients to connect in any of the following ways:

fifo pipes
The traditional approach. The default global /dev/imt1[io] pipes may be used, or a private set of fifos can be specified using the *-fifo* command line argument. Values should be specified as the root pathname to a pair of fifo pipes whose last character is ’i’ or ’o’, these characters will be added automatically when opening the pipes. For example, to use the default pipes the path would be specified as simply "/dev/imt1". A value of "none" disables this connection.

tcp/ip sockets
Clients connect via a tcp/ip socket. The default port is *5137*, or a custom port may be specified using the *-port* command line switch. This permits connecting to the server over a remote network connection anywhere on the Internet. A port number of 0 (zero) disables this connection.

unix domain sockets
Like a tcp/ip socket, but limited to a single host system. Usually faster than a tcp/ip socket, and comparable to a fifo. By default each user gets their own unix domain socket, so this option allows multiple users to run ximtools on the same host without having to customize things. The default value is "/tmp/.IMT%d", other sockets may be defined using the *-unix* command line switch. Legal values should be specified as a filename to be used for the socket, up to two "%d" fields are allowed and will be replaced by the userid. An empty string value disables this connection.

By default *vximtool* listens simultaneously for client connections on all three types of ports. Clients may connect simultaneously by different means allowing up to three different displays to be loading at the same time into different frames.

COMMUNICATIONS PROTOCOL

Clients communicate with *vximtool* using a protocol developed originally for IIS (International Imaging Systems) Frame Buffer hardware, the so-called "IIS protocol"; other more modern protocols will likely be supported in the future. The IIS protocol is basically a command packet stream with a header describing the operation to be performed (select frame, load display, read cursor, etc), and an optional data packet containing e.g. pixels. It is beyond the scope of this document to describe fully the details of the protocol; interested users should contact *iraf@noao.edu* for further information.

EXAMPLES

1) Run the vximtool, logging output to the file named "spool":

```
% vximtool -verbose >& spool
```

2) Run the vximtool in the background, connect only on unix sockets, no output:

```
% vximtool -b -unix_only &
```

3) Don't store images in memory, start with initial 1024x1024 frame buffer:

```
% vximtool -noraster -config 3
```

4) Run the vximtool in the background, taking cursor input from a file:

```
% vximtool -i < cursor_file &
```

SEE ALSO

ximtool(1)

COPYRIGHT