NAME

new_svas – creating a new instance of a Shore Value-Added Server

SYNOPSIS

```
// defines class shore_vas:
#include <shore_vas.h>

shore_vas       *new_svas(
    shore_vas             **result = NULL,
    const char            *host = NULL,
    int                   kbytes_shm_small = -1,
    int                   kbytes_shm_large = -1
);

bool
shore_svas::connected() const;
```

DESCRIPTION

The function **new_svas** creates a new instance of the `class shore_vas` in an application's address space. An application program does not invoke this directly; the object cache does so.

The client library method **connected** returns *true* if a connection is established with the server. The server library method attaches the client's transaction to the thread that called the method, if the client is running a transaction. It returns *true* if *there* is *a* client is almost always the case).

ARGUMENTS

Host is a string whose contents are the name of the host on which the server runs. *Host* may be a null pointer, in which case the SVAS looks for a host name in the option value for the option **svas_shost.** See **options(svas)** for a list of applicable options. Once a host name is found, **new_svas** contacts the SVAS running on that host. The server is expected to be listening on the port given by the option **svas_port** or the default port, DEFAULT_PORT. DEFAULT_PORT is a compile-time constant.

The SVAS opens a TCP connection with the server, authenticates the client process to the server, and request that the server allocate shared memory for message-passing according the the values of *kbytes_shm_small* and *kbytes_shm_large. Kbytes_shm_small* indicates how many kilobytes of shared memory to allocate for the transfer of small-object pages (groups of small anonymous objects) between the server and the client. *Kbytes_shm_large* indicates how many kilobytes of shared memory to allocate for the transfer of large-object pages (parts of large objects) between the server and the client. If no value is given for either of these arguments, their values are taken from the options **svas_shm_small_obj** and **svas_shm_large_obj,** respectively.

If the server does not reside on the same machine as the client (application process), shared memory is not used. In this case, all data flow across the TCP connection.

If an error is encountered in any of these steps, and *result* is non-null, **result* will be a null pointer on return from the function.

ENVIRONMENT

New_vas is not available on the server.

The environment variable SHORE_RC, if set, must be the name of a configuration file containing values for the options. If it is not set, the client library uses the name `.svas_rc.` The library searches `$HOME,` then the working directory for a file with that name. If such a file is found, it is read and its commands are interpreted. See **options(svas)** for more information about options.

ERRORS

Should an error occur during the function **new_svas,** information will be logged to the error log for the client. The error log is determined by the value of the option **svas_log.** (The default is to log to the standard error file.) See **options(svas)** for more information about options.

A complete list of errors is in **errors(svas).**

VERSION

This manual page applies to Version 1.1 of the Shore software.

SPONSORSHIP

The Shore project is sponsored by the Advanced Research Project Agency, ARPA order number 018 (formerly 8230), monitored by the U.S. Army Research Laboratory under contract DAAB07-91-C-Q518.

COPYRIGHT

Copyright © 1994, 1995, 1996, 1997, Computer Sciences Department, University of WisconsinMadison. All Rights Reserved.

SEE ALSO

errors(svas), options(svas), and **intro(svas).**

BUGS

We need to describe how to create an instance of class svas_server (on the server side).