NAME

lookup, _lookup– locating Shore objects by path name

SYNOPSIS

```
VASResult       shore_vas::lookup(
    const Path          pathname,
    lrid_t              *result,
    PermOp              targetperm = Permissions::op_read,
    bool          errorIfNotFound = FALSE,
    bool          follow = FALSE
);

// server only:
VASResult       shore_vas::_lookup(
    const Path          pathname,
    bool          *found,
    lrid_t              *result = 0,
    bool          errorIfNotFound = TRUE,
    PermOp              targetperm = Permissions::op_read,
    bool          follow = TRUE,
    serial_t            *file = 0
);

// server only:
VASResult       shore_vas::_lookup(
    lrid_t              &dir,
    const    Path       pathname,
    PermOp              pathperm,
    PermOp              targetperm,
    bool          *found,
    lrid_t              *result,
    serial_t            *file,
    bool          errorIfNotFound = TRUE,
    bool          follow = TRUE
);
```

DESCRIPTION

The lookup methods translate from path names to object identifiers. (See **directory(svas)** for ways to translate object identifiers to path names.) They do not affect the working directory. They all check permissions along the path. They can be used to follow or not to follow symbolic links and cross references.

ARGUMENTS

Pathname is the Shore pathname to follow. It may be absolute or relative. In the first two methods, if it is relative, it is relative to the current working directory. In the third method, it is relative to the directory identified by *dir*.

If an object with such a pathname exists, its object identifier is returned in ∗*result.* If no such object exists, ∗*result* is unchanged.

If *follow* is TRUE, and the pathname encounters symbolic links or cross-references along the way, they will be followed, otherwise the search will terminate at the point at which a symbolic link or cross-reference is found. If the path name does not identify an object, and *errIfNotFound* is TRUE, all forms of lookup issue an error message and return ST_FAILURE. If *errIfNotFound* if FALSE, they are silent, but indicate that no such object was found as follows: **lookup** returns ST_FAILURE, and both forms of **_lookup** return ST_OK. ∗**Found,** where present, is set to TRUE if such an object exists, and to FALSE if not.

At the end of the search, if an object is found, it is checked for the permissions given in *targetperm.* Along the path, **lookup** and the first form of **_lookup** require search (execute) permissions; the second form of **_lookup** checks for the permissions indicated in *pathperm.*

Both forms of **_lookup** return the logical file identifier for the Shore Storage Manager *file* in which the object resides. In the first form, the file identifier is returned only if *file* is a non-null pointer.

ENVIRONMENT

Lookup is available in the client library and in the server; both forms of **_lookup** are available only on the server.

ERRORS

Deadlocks can occur while locks are being acquired. See **transaction(svas)** for information about deadlocks.

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.

SEE ALSO

errors(svas), directory(svas), file_system(svas), registered(svas), and **transaction(svas)**