

**NAME**

format\_dev, mount\_dev, dismount\_dev, dismount\_all, list\_devices, list\_volumes, get\_device\_quota – Class ss\_m Methods for Device Management

**SYNOPSIS**

```
#include <sm_vas.h> // which includes sm.h

static rc_t      format_dev(
    const char*      device,
    uint4            quota_in_KB,
    bool             force);

static rc_t      mount_dev(
    const char*      device,
    u_int&          vol_cnt,
    devid_t&        devid,
    vid_t           local_vid = vid_t::null);

static rc_t      dismount_dev(const char* device);
static rc_t      dismount_all();
static rc_t      list_devices(
    const char**&    dev_list,
    devid_t*&        devid_list,
    u_int&           dev_cnt);

static rc_t      list_volumes(
    const char*      device,
    lvid_t*&         lvid_list,
    u_int&           lvid_cnt);

static rc_t      get_device_quota(
    const char*      device,
    smksize_t&       quota_KB,
    smksize_t&       quota_used_KB);
```

**DESCRIPTION**

The above class **ss\_m** methods all deal with managing the devices that hold volumes.

A device is either an operating system file or operating system device and is identified by a path name (absolute or relative). A device has a quota. A device may have multiple volumes on it (in the current implementation the maximum number of volumes is 1).

A volume is where data are stored. A volume is identified uniquely and persistently by a logical volume ID (lvid\_t). Volumes can be used whenever the device they are located on is mounted by the SM. Volumes have a quota. The sum of the quotas of all the volumes on a device cannot exceed the device quota. Methods for volume management are described in **volume(ssm)**.

**format\_dev(device, quota\_in\_KB, force)**

The **format\_dev** method is used to format a storage device for use in holding volumes. The *device* parameter the path-name of the device to format. The *quota\_in\_KB* is the quota (maximum usable space) for the device, specified in kilobytes. Setting the *force* parameter to **true** indicates that the device should be formatted even if *device* already exists. Setting it to **false** will generate the POSIX error, **EEXIST** indicating that *device* already exists. Since raw-devices always "exist", **true** must always be used. The SM will now allow a device to be formatted if it is already mounted.

**mount\_dev(device, vol\_cnt, devid, local\_vid)**

The **mount\_dev** method makes all volumes on a device available for access. The number of volumes on the device is returned in *vol\_cnt*. The **list\_devices** method described below can be used to determine the IDs of the volumes on the device. The ID of the device (unique only to the server) is returned in *devid*. *Device IDs are not used* but some VASs may find them useful for their own purposes. The *local\_vid* parameter is only meant to be a temporary hack for those VASs using the physical ID version of the SM interface. *Local\_vid* is used to specify the local handle that should be when a volume is mounted. The default value, **vid\_t::null** indicates that the SM can use any number it wants to use.

It is OK to mount a device multiple times, as long as *device* is always the same (ie. you cannot specify another path that is a hard/soft link to the path given in previous mount requests). Device mounts are not reference counted, so only a single **dismount\_dev** call is necessary to dismount a device.

**dismount\_dev(device)**

The **dismount\_dev** method flushes all cached pages on the device and makes all volumes on the device unavailable. **Note:** Currently, there is no check made to make sure no transactions are using a device when it is dismounted. If a transaction operation accesses a volume on a device that is no longer mounted, an error is returned from the operation. If a transaction previously accessed a volume on a device that is no longer mounted, and the transaction aborts, a fatal error will occur, shutting down the server.

**dismount\_all()**

The **dismount\_all** method dismount all mounted devices.

**list\_devices(dev\_list, devid\_list, dev\_cnt)**

The **list\_devices** method returns, in *dev\_list*, an array of char\* pointers to the names of all mounted devices. Note that the use of a char\*'s is a temporary hack until a standard string class is available. The char\* pointers are pointing directly into the device mount table. The *devid\_list* is changed to point to an array of device IDs. **Note:** *dev\_list* and *devid\_list*

must be deleted with `delete []` by the caller if they are not null (0). They will be null if an error is returned or if there are no devices. The *dev\_cnt* parameter is the number of elements in the returned lists.

**list\_volumes(device, lvid\_list, lvid\_cnt)**

The **list\_volumes** method returns, via *lvid\_list*, an array containing the IDs of all volumes on *device*. The *lvid\_cnt* parameter is set to the length of the list returned. *Lvid\_list must be deleted with delete* *lvid\_list* is not null (0). *lvid\_list* will be null if an error is returned or if there are no volumes on the device.

**get\_device\_quota(device, quota\_KB, quota\_used\_KB)**

The **get\_device\_quota** method returns the quota (in K-bytes) in *quota\_KB* and the amount of the quota, allocated to volumes on the device, in *quota\_used\_KB*.

**ERRORS**

All of the above methods return a **w\_rc\_t** error code.

See **errors(ssm)** for more information on error handling.

**TRANSACTION ISSUES**

Many of the above methods cannot be run in the scope of a transaction. The reason for this restriction is to avoid the implication that rolling back (aborting) the transaction would rollback the effect of the method.

**EXAMPLES**

TODO

**VERSION**

This manual page applies to Version 2.0 of the Shore Storage Manager.

**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. Further funding for this work was provided by DARPA through Rome Research Laboratory Contract No. F30602-97-2-0247.

**COPYRIGHT**

Copyright (c) 1994-1999, Computer Sciences Department, University of Wisconsin -- Madison. All Rights Reserved.

**SEE ALSO**

**intro(ssm)**, **volume(ssm)**.