NAME

_fini, _info, _init – loadable module configuration entry points

SYNOPSIS

#include <sys/modctl.h>

int _fini(void);

int _info(struct modinfo *∗modinfop***);**

int _init(void);

INTERFACE LEVEL

Solaris DDI specific (Solaris DDI). These entry points are required. You **must** write them.

ARGUMENTS

_info( )

modinfop A pointer to an opaque **modinfo** structure.

DESCRIPTION

_init( ) initializes a loadable module. It is called before any other routine in a loadable module. **_init( )** returns the value returned by **mod_install**(9F). The module may optionally perform some other work before the **mod_install**(9F) call is performed. If the module has done some setup before the **mod_install**(9F) function is called, then it should be prepared to undo that setup if **mod_install**(9F) returns an error.

_info( ) returns information about a loadable module. **_info( )** returns the value returned by **mod_info**(9F).

_fini( ) prepares a loadable module for unloading. It is called when the system wants to unload a module. If the module determines that it can be unloaded, then **_fini( )** returns the value returned by **mod_remove**(9F). Upon successful return from **_fini( )** no other routine in the module will be called before **_init( )** is called.

RETURN VALUES

_init( ) should return the appropriate error number if there is an error, else it should return the return value from **mod_install**(9F).

_info( ) should return the return value from **mod_info**(9F)

_fini( ) should return the return value from **mod_remove**(9F).

EXAMPLES

The following example demonstrates how to initialize and free a **mutex**(9F).

```
#include <sys/modctl.h>
#include <sys/ddi.h>
#include <sys/sunddi.h>

static struct dev_ops  drv_ops;
/*
 * Module linkage information for the kernel.
*/
static struct modldrv modldrv = {
    &mod_driverops,    /* Type of module.  This one is a driver */
   "Sample Driver",
   &drv_ops       /* driver ops */
```

```
};

static struct modlinkage modlinkage = {
        MODREV_1,
        &modldrv,
        NULL
};


/*
 * Global driver mutex
 */
static kmutex_t   xx_global_mutex;


int
_init(void)
{
        int     i;

        /*
          * Initialize global mutex before mod_install'ing driver.
          * If mod_install() fails, must clean up mutex initialization
          */
        mutex_init(&xx_global_mutex, "XXX Global Mutex",
                MUTEX_DRIVER, (void *)NULL);

        if ((i = mod_install(&modlinkage)) != 0) {
                mutex_destroy(&xx_global_mutex);
        }

        return (i);
}

int
_info(struct modinfo *modinfop)
{
        return (mod_info(&modlinkage, modinfop));
}


int
_fini(void)
{
        int       i;
```

```
        /*
         * If mod_remove() is successful, we destroy our global mutex
         */
        if ((i = mod_remove(&modlinkage)) == 0) {
                mutex_destroy(&xx_global_mutex);
        }
        return (i);
}
```

SEE ALSO **add_drv**(1M), **mod_info**(9F), **mod_install**(9F), **mod_remove**(9F), **mutex**(9F), **modldrv**(9S), **modlinkage**(9S), **modlstrmod**(9S)

Writing Device Drivers

WARNINGS Do not change the structures referred to by the **modlinkage** structure after the call to **mod_install( )**, as the system may copy or change them.

NOTES Even though the identifiers **_fini( )**, **_info( )**, and **_init( )** appear to be declared as globals, their scope is restricted by the kernel to the module that they are defined in.

BUGS On some implementations **_info( )** may be called before **_init( )**.