NAME

kstat_create – create and initialize a new kstat

SYNOPSIS

#include <sys/types.h>
#include <sys/kstat.h>

kstat_t ∗**kstat_create(char** ∗*module*, **int** *instance*, **char** ∗*name*, **char** ∗*class* , **uchar_t** *type*, **ulong_t** *ndata*, **uchar_t** *ks_flag***);**

INTERFACE LEVEL

Solaris DDI specific (Solaris DDI)

ARGUMENTS

module The name of the provider's module (such as "sd", "esp", ...). The "core" kernel uses the name "unix".

instance The provider's instance number, as from **ddi_get_instance**(9F). Modules which don't have a meaningful instance number should use **0**.

name A pointer to a string that uniquely identifies this structure. Only **KSTAT_STRLEN - 1** characters are significant.

class The general class that this kstat belongs to. The following classes are currently in use: **disk**, **tape**, **net**, **controller**, **vm**, **kvm**, **hat**, **streams**, **kstat**, and **misc**.

type The type of kstat to allocate. Valid types are:

KSTAT_TYPE_NAMED named - allows more than one data record per kstat

KSTAT_TYPE_INTR interrupt - only one data record per kstat

KSTAT_TYPE_IO I/O - only one data record per kstat

ndata The number of type-specific data records to allocate.

flag A bit-field of various flags for this kstat. **flag** is some combination of:

KSTAT_FLAG_VIRTUAL Tells **kstat_create( )** not to allocate memory for the kstat data section; instead, the driver will set the **ks_data** field to point to the data it wishes to export. This provides a convenient way to export existing data structures.

KSTAT_FLAG_WRITABLE Makes the kstat's data section writable by root.

KSTAT_FLAG_PERSISTENT

Indicates that this kstat is to be persistent over time. For persistent kstats, **kstat_delete**(9F) simply marks the kstat as dormant; a subsequent **kstat_create( )** reactivates the kstat. This feature is provided so that statistics are not lost across driver close/open (such as raw disk I/O on a disk with no mounted partitions.)

Note: Persistent kstats cannot be virtual, since ks_data points to garbage as soon as the driver goes away.

DESCRIPTION

kstat_create( ) is used in conjunction with **kstat_install**(9F) to allocate and initialize a **kstat**(9S) structure. The method is generally as follows:

```
kstat_t  *ksp;

ksp = kstat_create(module, instance, name, class, type, ndata, flags);
if (ksp) {
        /* ... provider initialization, if necessary */
        kstat_install(ksp);
}
```

kstat_create( ) allocates and performs necessary system initialization of a **kstat**(9S) structure. **kstat_create( )** allocates memory for the entire kstat (header plus data), initializes all header fields, initializes the data section to all zeroes, assigns a unique kstat ID (KID), and puts the kstat onto the system's kstat chain. The returned kstat is marked invalid because the provider (caller) has not yet had a chance to initialize the data section.

After a successful call to **kstat_create( )** the driver must perform any necessary initialization of the data section (such as setting the name fields in a kstat of type **KSTAT_TYPE_NAMED**). Virtual kstats must have the **ks_data** field set at this time. The provider may also set the **ks_update**, **ks_private**, and **ks_lock** fields if necessary.

Once the kstat is completely initialized, **kstat_install**(9F) is used to make the kstat accessible to the outside world.

RETURN VALUES

If successful, **kstat_create( )** returns a pointer to the allocated kstat. **NULL** is returned on failure.

CONTEXT

kstat_create( ) can be called from user or kernel context.

SEE ALSO

kstat(3K), **kstat_delete**(9F), **kstat_install**(9F), **kstat_named_init**(9F), **kstat**(9S), **kstat_named**(9S)

Writing Device Drivers