

initFromDeviceDescription:

Indirect device drivers also need to implement the requiredProtocols class method.

registerDevice
setDeviceKind:
setLocation:
setName:

The registerDevice method should be invoked at the end of initialization. Generally, indirect and direct drivers also invoke setUnit:.

init
initFromDeviceDescription:
free

Registering the class+ deviceStyle

+ registerClass:

setName:
name
setUnit:
unit

Converting an IOReturn value+ stringFromReturn:

stringFromReturn:
errnoFromReturn:

Adding and removing the driver from UNIX device switch tables

+ addToBdevswFromDescription:open:close:strategy: dump:p
+ addToCdevswFromDescription:open:close:read:write: ioctl:
 getc:putc:
+ blockMajor
+ characterMajor
+ removeFromBdevsw
+ removeFromCdevsw
+ setBlockMajor:
+ setCharacterMajor:

Getting the Driver Kit version of the IODevice

+ driverKitVersion
+ driverKitVersionForDriverNamed:

Getting and setting parameter values

setCharValues:forParameter:count:
getCharValues:forParameter:count:
setIntValues:forParameter:count:
getIntValues:forParameter:count:

1 if this driver has no block major number. The block major number is set using `setBlockMajor`;, w
`addToBdevswFromDescription....`

1 if this driver has no character major number. The character major number is set using `setCharacter`
invoked by `addToCdevswFromDescription....`

`initFromDeviceDescription:`

1, this method does nothing and returns NO. Otherwise, this method sets the character major number (setCharacterMajor:) and returns YES.

stringFromReturn:

(const char *)deviceKind

Returns a string that identifies the object in general terms. For example, IOCSIDisk objects return the description of setDeviceKind: for more information.

setDeviceKind:

(int)errnoFromReturn:(IOReturn)returnValue

Returns a UNIX error number that corresponds to the specified IOReturn value. Subclasses that add

(IOReturn)getCharValues:(unsigned char *)array forParameter:(IOParameterName)parameter
count:(unsigned int *)count

Gets the array of character values associated with parameter. IODevice accepts the following characters:
IO_CLASS_NAME (which returns [[self class] name]), IO_DEVICE_NAME (which returns [self name]),
IO_DEVICE_KIND (which returns [self deviceKind]).

Subclasses should override this method if they support parameters not understood by the superclass.
overriding this method:

getIntValues:forParameter:count:, setCharValues:forParameter:count:

(IOReturn)getIntValues:(unsigned int *)array forParameter:(IOParameterName)parameter
count:(unsigned int *)count

Returns IO_R_UNSUPPORTED. Subclasses should implement this method if necessary to return integer values associated with parameter. See getCharValues:forParameter:count: for an example of this kind of method. This method should return IO_R_SUCCESS if parameter is a valid parameter with which data can be read otherwise, it should return IO_R_UNSUPPORTED.

getCharValues:forParameter:count:, setIntValues:forParameter:count:

init

Initializes and returns a newly allocated IODevice. Returns self if successful otherwise, returns nil.

(const char *)location

Returns the device-specific location of the IODevice. For example, "0xf7f04000". See the description of the IODevice protocol for information on how this location is used.

setLocation:

(const char *)name

Returns the device-specific name of the IODevice. For example, "sd0a". See the description of the IODevice protocol for information on how the name is used.

setName:

registerDevice

Registers the IODevice in the current name space and adds a string to the system log that announces the registration. The IODevice must be ready to perform I/O, its name must have been set already using setName:, and its location (set with setLocation:) must be either valid or NULL.

This method also probes all indirect IODevices that require this object's protocols, giving them a chance to initialize themselves.

Each IODevice should invoke this method at the end of its initialization. Returns self.

unregisterDevice

(IOReturn)setCharValues:(unsigned char *)array forParameter:(IOParameterName)parameter
count:(unsigned int)count

Returns IO_R_UNSUPPORTED. Subclasses should implement this method if necessary to set (from the array) character values associated with parameter. See getCharValues:forParameter:count: for an example of the kind of method. This method should return IO_R_SUCCESS if parameter is a valid parameter with which data can be written otherwise, it should return IO_R_UNSUPPORTED.

setIntValues:forParameter:count:, getCharValues:forParameter:count:

(void)setDeviceKind:(const char *)type

Sets a string that identifies the object in general terms. For example, IOFrameBufferDisplay object sets its deviceKind to "Linear Framebuffer". The string should be no longer than IO_STRING_LENGTH - 1 characters. The name IO_DEVICE_KIND refers to this string.

deviceKind

(IOReturn)setIntValues:(unsigned int *)array forParameter:(IOParameterName)parameter
count:(unsigned int)count

(void)setLocation:(const char *)location

Sets the device-specific location of the IODevice. For example, "0xf7f04000". If the location is invalid, should be set to NULL. The location is used in the system log when this object is registered and unregistered.
location

(void)setName:(const char *)name

Sets the device-specific name of the IODevice. For example, "sd0a". The name should be no longer than IO_STRING_LENGTH - 1 characters.

The specified name is used to identify this instance. For example, it's used in the system log when the device is registered and unregistered, and it's used by the UNIX command iostat. The name is also used by user-level programs to identify the IODevice object, using the IODeviceMaster method lookUpByDeviceName:objectNumber:deviceKind:. The name IO_DEVICE_NAME refers to this string.

name

(void)setUnit:(unsigned int)unit

Sets the IODevice's unit number, a device-specific number that can be used like a UNIX minor number.
unit

(const char *)stringFromReturn:(IOReturn)returnValue

Returns the text string that corresponds to the specified IOReturn value. Subclasses that add additional IOReturn values should override this method and invoke stringFromReturn: against the superclass for IOReturn values that the subclass doesn't handle.

(unsigned int)unit

Returns the IODevice's unit number, a device-specific number that can be used like a UNIX minor number.
setUnit:

(void)unregisterDevice

Removes the IODevice from the current name space.

registerDevice