MiscSerialPort

Inherits From: Object

Declared In: misckit/MiscSerialPort.h

Class Description

The MiscSerialPort class provides a wrapper around the serial ports in NeXTSTEP. It includes the ability to set the port's baud rate, parity, bits per character and to control the modem line DTR. A MiscSerialPort is initialized with the init method and then the various parameters are established for the line. A delegate provides a means to pass received characters to another object to be processed. Sending data is done by calling the ±sendChars:length: method. The line may be paused using the ±suspend method and restarted by the ±continue method in the event that it is desired to allow another mechanism (such as a subprocess) to have access to the port for a time.

Instance Variables

```
char portName[64];          BOOL connected;   BOOL suspended;
                            int fd;     id delegate;        int
currentBaud;                int currentParity;      DPSTimedEntry
*tentry;
```

portName   Full path to the serial device. (ex. "/dev/cufa")

connected   YES if the portName device file   is currently open.

suspended   YES if the port has been paused by ±suspend.

fd   file descriptor for the portName device file.

delegate   points to the delegate object. This object must answer the methods
                        described for the MiscSerialPortDelegate category. The delegate can be
                        nil if you do not wish to receive delegate messages.

currentBaud   current baud rate from the constants in <sgtty.h>

currentParity   parity for the port as described in MiscSerialPort.h

tentry   a timed entry used to poll for received data

Method Types

Initializing the serial port   ± init

Setting and examining parameters   ± setDeviceName:
±setBaud:
± setBaudByName:
± setParity:
± setDelegate:
± (int)filedes
± (BOOL)suspended

Controlling Communications   ± (BOOL)connect
± disconnect
± suspend
± resume
± continue
± dropDTR
± raiseDTR

Sending Data   ± transmitChars:length:

Delegate methods   ± receiveChars:length:
± hangup

Instance Methods

connect
   - (BOOL)**connect**

Opens the port with the current parameters. Be sure to set a valid delegate before doing this.   Returns **YES** if the connect succeded and **NO** if the connection failed.

See also: - **disconnect**

continue
   - **continue**

Synonym for ±**resume**. This method is retained for backward compatability and is considered obsolete due to the fact that **continue** is a C language keyword. Future use is discouraged.

See also: - **suspend, ± resume**

delegate
   - **delegate**

Returns the pointer to the current delegate.

See also: - **setDelegate:**

disconnect

- **disconnect**

Closes the serial port.   Returns *self*.

See also: **-connect**

init

- **init**

Initializes an instance of MiscSerialPort.   Returns *self*.

dropDTR

- **dropDTR**

Lowers the DTR line to the port. This is usefull for hanging up on a modem.   Returns *self*.

See also: **-raiseDTR**

filedes

- (int)**filedes**

Returns the file descriptor for an open port.

raiseDTR
    - raiseDTR

Raises the DTR line for the port.   Returns **self**.

See also: **- dropDTR**

continue
    - continue

Restarts communications after the port has been suspended.   Returns *self*.

See also: **- suspend**

setBaud:
    - setBaud: (int)baud

Sets the baudrate for the port. Baud should be one of the following values as declared in <sgtty.h>.

```
B50             1
B75             2
B110            3
B134            4
B150            5
B200            6
B300            7
B600            8
B1200           9
B1800           10
```

```
B2400          11
B4800          12
B9600          13
B19200         14
B38400         15
B14400         16
B28800         17
B43200         18
B57600         19
```

Note that not all hardware can support the higher baud rates. The 68030 machines in particular have difficulty maintaining baudrates higher than 9600. Returns **self**.

See also: - **setBaudByName:**

setBaudByName:
   - **setBaudByName:** (const char *)speed

Allows the setting of baud rate by a string value such as "9600".   The valid strings are given below, note that not all possible baud rates are supported, only those that are "normal" for terminal use.

valid strings are: "110","300","600","1200","2400","4800","9600","19200","38400","57600".

Returns **self** unless an invalid string was passed in which case the baud rate is not changed and the return value is **nil**.

See also: - **setBaud:**

setDelegate:
    - **setDelegate:** theConsumer

Sets the object that received characters and status changes will be sent to.   Returns **self**.

See also: - **delegate**

setDeviceName:
    - **setDeviceName:** (const char *)name

Sets the path to the port's device file. This should be done before calling connect. If the port is connected and this routine is called with a name that does not match the current name then the port will be disconnected and reconnected (this is not recomended since there is no way to get the connect status).   Returns **self**.

setParity
    - **setParity:** (int)parity

Sets the parity of the port. This is currently stubbed in and not implemented.   Returns **self**.

suspend
    - **suspend**

Disables reception from the port. The port device file remains open and modem signals do not change.

Transmission through the ±**transmitChars:length:** method is still possible. This is usefull if you want to pass the port to a subprocess and then resume reception after the process terminates.   Returns **self**.

See also: ± **resume**

transmitChars:length:
   - **transmitChars:** (const char *)buffer **length:** (int)length

Transmits a buffer of characters through the port.   Returns **self**.

Delegate Methods

receiveChars: length:
   - **receiveChars:** (char *)buffer **length:** (int)length

Called when characters are received on the port to give the delegate a chance to process them. . Returns **self**.

hangup
   - **hangup**

Called on a high to low transition of DCD. Returns **self**. Note: this method does not currently work due to the inability to sense DCD with the NeXT serial drivers.