24.1.5.0 Klasse SerialPort

Die Klasse SerialPort (gb.net) wurde entwickelt, um die Kommunikation über eine serielle Schnittstelle (in der Regel eine RS-232-Schnittstelle) zu ermöglichen. Diese Klasse erbt von der Stream-Klasse, so dass Sie Standard-Streams verwenden können, um Daten zu senden und zu empfangen und den Port zu schließen. Die Klasse können Sie erzeugen.

```
Dim hSerialPort As SerialPort
hSerialPort = New SerialPort() As "RS232" ' Event name
```

24.1.5.0.1 Konstanten

Die Klasse besitzt die folgenden Konstanten:

Konstante	Wert	Beschreibung
Bits1	1	Diese Konstante wird verwendet, wenn die serielle Schnittstelle 1 Stoppbit hat.
Bits2	2	Diese Konstante wird verwendet, wenn die serielle Schnittstelle 2 oder 1,5 Stoppbits hat.
Bits5	5	Diese Konstante wird verwendet, wenn die serielle Schnittstelle 5 Datenbits hat.
Bits6	6	Diese Konstante wird verwendet, wenn die serielle Schnittstelle 6 Datenbits hat.
Bits7	7	Diese Konstante wird verwendet, wenn die serielle Schnittstelle 7 Datenbits hat.
Bits8	8	Diese Konstante wird verwendet, wenn die serielle Schnittstelle 8 Datenbits hat.
Both	3	Diese Konstante, wird verwendet, wenn die serielle Schnittstelle sowohl über eine Software- als auch über eine Hardware-Flusskontrolle verwenden soll. Dies bedeutet, dass CRTSCTS plus XON/XOFF für die Durchflussregelung verwendet werden.
Hardware	1	Diese Konstante wird verwendet, wenn die serielle Schnittstelle Hardware-Flusskontrolle verwenden soll. Dies bedeutet, dass RTS/CTS beziehungsweise RFR/CTS für die Flusssteuerung verwendet wird.
Software	2	Diese Konstante wird verwendet, wenn die serielle Schnittstelle Software-Flusskontrolle verwenden soll. Dies bedeutet, dass die speziellen Steuerzeichen Xon (DC1, 11_{hex}, 17_{dec}) und Xoff (DC3, 13_{hex}, 19_{dec}) für die Flusssteuerung im Datenstrom verwendet werden.
None	0	Diese Konstante wird verwendet, wenn die serielle Schnittstelle keine Parität verwenden soll oder wenn die serielle Schnittstelle keine Flusskontrolle einsetzt.
Even	1	Diese Konstante repräsentiert eine gerade Parität für die serielle Schnittstelle. Sie können diesen Wert in der Parity-Eigenschaft setzen oder auslesen.
Odd	2	Diese Konstante steht für eine ungerade Parität der seriellen Schnittstelle. Sie können diesen Wert in der Parity-Eigenschaft setzen oder auslesen.

Tabelle 24.1.5.0.1 : Konstanten der Klasse SerialPort

Es gelten auch die zutreffenden Konstanten aus der Net-Klasse (gb.net). So wird zum Beispiel die Status-Eigenschaft nach dem erfolgreichen Öffnen der seriellen Schnittstelle über SerialPort.Open() auf den Wert *Net.Active* gesetzt.

24.1.5.0.2 Eigenschaften

Die Klasse SerialPort verfügt über diese Eigenschaften:

Eigenschaft	Datentyp	Beschreibung
Blocking	Boolean	Gibt True zurück, wenn der Stream blockiert wird oder setzt den Wert. Wenn diese Eigenschaft auf True gesetzt ist, wird das Lesen aus dem Stream blockiert, wenn nichts zu lesen ist, und das Schreiben in den Stream blockiert wird, wenn beispielsweise der interne Systempuffer voll ist.
ByteOrder	Integer	Liefert oder setzt die Byte-Reihenfolge, die zum Lesen oder Schreiben von Binärdaten in den Stream verwendet wird. Die Eigenschaft kann folgende Werte annehmen: gb.BigEndian = BigEndianByte-Order und gb.LittleEndian = LittleEndianByte-Order.
CTS	Boolean	Gibt den aktuellen Status des Signals "Clear To Send" der seriellen

Eigenschaft	Datentyp	Beschreibung
		Schnittstelle. Der Wert ist False, wenn die Schnittstelle geschlossen ist.
DCD	Boolean	Liefert den aktuellen Status des Signals "Data Carrier Detect" der seriellen Schnittstelle. Der Wert ist False, wenn die Schnittstelle geschlossen ist.
DSR	Boolean	Liefert den aktuellen Status des Signals "Data Set Ready" der seriellen Schnittstelle. Der Wert ist False, wenn die Schnittstelle geschlossen ist.
RNG	Boolean	Gibt den aktuellen Status des Signals "Ring Indicator" der seriellen Schnittstelle zurück. Wenn die serielle Schnittstelle geschlossen ist und Sie diese Eigenschaft lesen, so wird False zurückgegeben.
DTR	Boolean	Liefert oder setzt den aktuellen Status des Signals "Data Terminal Ready" der seriellen Schnittstelle. Dieser Wert kann nicht eingestellt werden, wenn die serielle Schnittstelle geschlossen ist. Wenn die serielle Schnittstelle geschlossen ist und Sie diese Eigenschaft lesen, so wird False zurückgegeben.
RTS	Boolean	Liefert oder setzt den aktuellen Status des Signals "Request To Send" der seriellen Schnittstelle. Dieser Wert kann nicht eingestellt werden, wenn die serielle Schnittstelle geschlossen ist. Wenn die serielle Schnittstelle geschlossen ist und Sie diese Eigenschaft lesen, so wird False zurückgegeben.
DataBits	Integer	Liefert oder setzt die Anzahl der von der seriellen Schnittstelle verwendeten Datenbits. Verwenden Sie eine der folgenden Konstanten: SerialPort.Bits5, SerialPort.Bits6, SerialPort.Bits7 oder SerialPort.Bits8.
StopBits	Integer	Die Stoppbits der seriellen Schnittstelle zurückgeben oder setzen. Verwenden Sie eine der folgenden Konstanten: SerialPort.Bits1 oder SerialPort.Bits2. Achtung: Bei 5-Bit-Datenbreite führt das Setzen von 2 Stopbits zu 1.5 Stopbit!
Parity	Integer	Gibt die aktuelle Parität der seriellen Schnittstelle zurück oder setzt sie. Verwenden Sie eine der folgenden Konstanten: SerialPort.None, SerialPort.Odd oder SerialPort.Even.
Speed	Integer	Setzen Sie die Kommunikationsgeschwindigkeit in Baud. Bis zur stabilen Gambas-Version 3.12.2 musste der Wert ein gültiger Standardwert sein(…, 4800, 9600, …). In späteren stabilen Versionen sind beliebige Baudraten (sogenannte Custom Baudrates) erlaubt. Seien Sie vorsichtig, da einige verbreitete Geschwindigkeitswerte nicht von allen seriellen Schnittstellen (UART) unterstützt werden! Dies gilt besonders für USB/Seriell-Wandler.
FlowControl	Integer	Die aktuelle Datenflusskontrolle der seriellen Schnittstelle wird eingestellt oder deren Wert ausgelesen. Verwenden Sie eine der folgenden Konstanten: SerialPort.None, SerialPort.Hardware, SerialPort.Software oder SerialPort.Both.
EndOfFile	Boolean	Diese Eigenschaft signalisiert, ob die letzte Verwendung von LINE INPUT das Ende der Datei erreicht hat, anstatt eine ganze Zeile mit einem Zeilenendezeichen zu lesen.
EndOfLine	Integer	Gibt das von diesem Stream verwendete Zeilenumbruch-Trennzeichen zurück oder setzt es. Die möglichen Werte sind: gb.Unix für durch Chr$(10) getrennte Zeilen, gb.Windows für durch Chr$(13) und Chr$(10) getrennte Zeilen oder gb.Mac für durch Chr$(13) getrennte Zeilen. Der Wert dieser Eigenschaft wird von LINE INPUT, PRINT und der Eigenschaft Stream.Lines verwendet.
PortName	String	Gibt den aktuellen Pfad der seriellen Schnittstelle zurück oder setzt ihn. Beispiel: mySerialPort.PortName = "/dev/ttyS0". Diese Eigenschaft kann nur geändert werden, wenn die serielle Schnittstelle geschlossen ist.
Handle	Integer	Gibt den mit diesem Stream verknüpften Systemdatei-Deskriptor zurück.

Eigenschaft	Datentyp	Beschreibung
IsTerm	Boolean	Gibt True zurück, wenn ein Stream einem Terminal zugeordnet ist.
Term	.Stream.Term	Gibt ein virtuelles Objekt zurück, mit dem das dem Stream zugeordnete Terminal verwaltet werden kann.
Lines	.Stream.Lines	Gibt ein virtuelles Objekt zurück, mit dem Sie einen Stream zeilenweise durchlaufen können. (For Each aString In hStream.Lines ... Next)
InputBufferSize	Integer	Gibt die Anzahl der Bytes des internen Eingangspuffers zurück.
OutputBufferSize	Integer	Gibt die Anzahl der Bytes des internen Ausgangspuffers zurück.
Tag	Variant	Gibt den Tag zurück oder setzt ihn. Diese Eigenschaft ist für den Programmierer bestimmt und wird nie von der Komponente verwendet. Tag kann einen beliebigen Variant-Wert enthalten.
Status	Integer	Gibt den aktuellen Status eines SerialPort-Objekts wieder. Net.Active - Der Port ist geöffnet. Net.Inaktiv - Der Port ist geschlossen.

Tabelle 24.1.5.0.2 : Eigenschaften der Klasse SerialPort

24.1.5.0.3 Methoden

Die Klasse besitzt diese sieben Methoden:

Methode	Rückgabetyp	Beschreibung
Open([Polling As Integer])	-	Öffnet die serielle Schnittstelle. Die Eigenschaft Status wird auf den Wert Net.Active gesetzt. Das optionale Argument *Polling* bestimmt die Anzahl der Versuche.
Close( )	-	Schließt den Stream. Diese Anweisung entspricht exakt der Close()-Anweisung.
Begin( )	-	Beginnt damit, die in den Stream geschriebenen Daten zu puffern, so dass beim Aufruf der Send-Methode alles gesendet wird.
Send( )	-	Sendet alle Daten auf einmal, die seit dem letzten Anruf von "Begin" in den Puffer geschrieben worden sind.
Drop( )	-	Löscht die Daten, die seit dem letzten Aufruf der Begin()-Methode gepuffert wurden.
ReadLine([Escape As String])	String	Liest eine Textzeile aus dem Stream, wie bei der Anweisung LINE INPUT. Wenn Escape angegeben ist, dann werden Zeilenumbrüche zwischen zwei Escape-Zeichen ignoriert.
Watch(Mode As Integer, Watch As Boolean)	-	Starten oder stoppen Sie die Beobachtung des Stream-Datei-Deskriptors zum Lesen oder Schreiben, nachdem er geöffnet wurde. Modus ist der Watch-Typ: gb.Read zum Lesen und gb.Write zum Schreiben. Ist das Argument Watch True, dann wird die Beobachtung aktiviert; bei False wird die Beobachtung de-aktiviert.

Tabelle 24.1.5.0.3 : Methoden der Klasse SerialPort

24.1.5.0.4 Ereignisse

Die Klasse SerialPort verfügt über diese Ereignisse:

Ereignis	Beschreibung
CTSChange(CurrentValue As Boolean)	Das Ereignis wird ausgelöst, wenn sich der CTS-Signalwert ändert. CurrentValue speichert den aktuellen Wert dieser Eigenschaft.
DSRChange(CurrentValue As Boolean)	Das Ereignis wird ausgelöst, wenn sich der DSR-Signalwert ändert. CurrentValue speichert den aktuellen Wert dieser Eigenschaft.
DTRChange(CurrentValue As Boolean)	Das Ereignis wird ausgelöst, wenn sich der DTR-Signalwert ändert. CurrentValue speichert den aktuellen Wert dieser Eigenschaft.

Ereignis	Beschreibung
RTSChange(CurrentValue As Boolean)	Das Ereignis wird ausgelöst, wenn sich der RTS-Signalwert ändert. CurrentValue speichert den aktuellen Wert dieser Eigenschaft.
DCDChange(CurrentValue As Boolean)	Das Ereignis wird ausgelöst, wenn sich der DCD-Signalwert ändert. CurrentValue speichert den aktuellen Wert dieser Eigenschaft.
RNGChange (CurrentValue As Boolean)	Das Ereignis wird ausgelöst, wenn sich der RNG-Signalwert ändert. CurrentValue speichert den aktuellen Wert dieser Eigenschaft.
Read( )	Das Ereignis wird ausgelöst, wenn Daten von der seriellen Schnittstelle gelesen werden können.

Tabelle 24.1.5.0.4 : Ereignisse der Klasse SerialPort

Mit diesem Quelltext wird ein Temperaturwert von einem Temperatursensor (DAC) über ein serielles Interface USB-RS232-Adapter alle 500ms ausgelesen, interpoliert und angezeigt:

```
' Gambas class file

Public hRS232 As SerialPort
Public sTemperatureDigit As String
Public TTimer As Timer

Public Sub Form_Open()

  FMain.Center
  FMain.Resizable = False

  hRS232 = New SerialPort As "hRS232"
  TTimer = New Timer As "TTimer"
  TTimer.Delay = 500 ' The temperature is read out every 500ms
  SetLEDColor(pboxStatus, "red")

  StartMeasurement()

End

Public Sub StartMeasurement()

  hRS232.PortName = "/dev/ttyUSB0"

  hRS232.Speed = 4800

  hRS232.DataBits = SerialPort.Bits8
  hRS232.StopBits = SerialPort.Bits1
  hRS232.Parity = SerialPort.None
  hRS232.FlowControl = SerialPort.None

  Try hRS232.Open(3)
  If Error Then
     Message.Error(("Error when opening the V24-RS232-USB adapter interface!"))
     SetLEDColor(pboxStatus, "red")
  Else
     If hRS232.Status = Net.Active Then
        TTimer.Start()
        Print hRS232.DTR
        Print hRS232.DSR
        Print hRS232.Blocking
        Print hRS232.CTS
        Print hRS232.RTS
     Endif
  Endif

End

Public Sub hRS232_Read()
  Read #hRS232, sTemperatureDigit, Lof(hRS232)
End

Public Sub TTimer_Timer()

  If hRS232.Status <> Net.Active Or If Asc(sTemperatureDigit) = 0 Then
     lblTemperaturAnzeige.Text = "--- °C"
  Else
     lblTemperaturAnzeige.Text = Str(Interpolation(Asc(sTemperatureDigit))) & " °C"
     TextBox1.Text = Asc(sTemperatureDigit)
     pboxStatus.SetFocus()
     SetLEDColor(pboxStatus, "green")
  Endif
```

```
End

Private Function Interpolation(iArgument As Integer) As Float

  Dim fX0, fX1, fY0, fY1, f As Float

' Experimentally determined data points
  fX0 = 22
  fY0 = 22.3
  fX1 = 30
  fY1 = 34.5

' Interpolation
  f = ((fY1 - fY0) / (fX1 - fX0)) * (iArgument - fX0) + fY0

  Return Round(f, -1)

End

Private Sub SetLEDColor(picBox As PictureBox, sLEDColor As String)
  picBox.Picture = Picture["LED/led_" & sLEDColor & ".svg"]
End

Public Sub Form_Close()

  If hRS232.Status = Net.Active Then
     hRS232.Close()
     FMain.Close()
  Endif

End
```

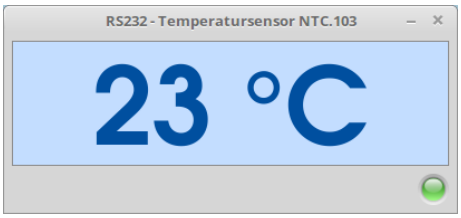

Abbildung 24.1.5.0.1: Temperatur-Anzeige