



TSerial component

[Properties](#)

[Methods](#)

[Data Types](#)

[Events](#)

[Copyright](#)

[Registration](#)

[Support](#)

[Source Code](#)

Description

The TSerial component provides a way of accessing PC serial ports from a Delphi application without having to directly use the Windows API functions.

The TSerial component is a non-visual component like TTimer : it appears at design time as an icon, but is invisible at run-time.

You need to add one TSerial component to a form for each serial port that you need to access concurrently. If you need to access more than one port, but one at a time, you can change ports just by altering the [Port](#) property.

A comms port is opened by setting the component's [Active](#) property to True, and closed by setting the Active property to False. The Active property is not visible at design-time.

All the port configuration properties can be changed at run-time when a port is active : the TSerial component opens, closes and re-initialises the port(s) as necessary.

TSerial Properties

<u>Active</u>	<u>MessageEndChar</u>	<u>RxQueueSize</u>
<u>Baudrate</u>	<u>MessageStartChar</u>	<u>RxWaiting</u>
<u>CheckParity</u>	<u>Name</u>	<u>StopBits</u>
<u>ComponentIndex</u>	<u>NotifyErrors</u>	<u>Tag</u>
<u>DataBits</u>	<u>Owner</u>	<u>TxQueueSize</u>
<u>ErrorCode</u>	<u>Parity</u>	<u>TxWaiting</u>
<u>FlowMode</u>	<u>Port</u>	
<u>MessageAppendCount</u>	<u>RxEventMode</u>	

TSerial Methods

ReadChar
ReadString

WriteChar
WriteString

ZapRxQueue
ZapTxQueue

TSerial Events

OnError

OnMessage

OnRxData

OnTxEmpty

TSerial Data Types

eDataBits

eFlowControl

eNotifyErr

eOpModes

eParity

eStopBits

sPorts

Active Property

Applies to

TSerial component

Declaration

property Active: Boolean;

Description

Note : Run-time only.

The Active property is the main "on/off" control for the TSerial component. It can only be set at run-time.

Typically, all the port properties such as Baudrate and Parity will be set before setting Active to True, although these can also be changed "on the fly" at run time.

Note that if an error (hardware not available for instance) prevents the port being opened, the Active property will not become True. Thus the success of opening the port can be gauged by setting and then testing Active :

```
Serial1.Baudrate := 4800;  
Serial1.Active := True;  
If Serial1.Active Then  
    { Port open OK actions }  
Else  
    { Port open failed actions }
```

Baudrate Property

Applies to

TSerial component

Declaration

property Baudrate: Word;

Description

The Baudrate property sets the port Baudrate. The Baudrate must be one of the following values :

300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600

CheckParity Property

Applies to

TSerial component

Declaration

property CheckParity: Boolean;

Description

The CheckParity property determines whether parity errors are detected on reception. If errors are detected, and CheckParity is set to True, then the erroneous characters are replaced with a '?'.

ComponentIndex Property

Applies to

TSerial component

Declaration

property ComponentIndex: Integer;

Description

Run-time and read only. The ComponentIndex property indicates the position of the component in its owner's Components property list. The first component in the list has a ComponentIndex value of 0, the second has a value of 1, and so on.

DataBits Property

Applies to

TSerial component

Declaration

property DataBits: eDataBits;

Description

Sets the number of databits per character for transmission and reception.

Value	Meaning
d7bit	7 data bits
d8bit	8 data bits

ErrorCode Property

Applies to

TSerial component

Declaration

property ErrorCode: Integer;

Description

Contains the last error code obtained from the serial port. The following values are significant:

Value	Meaning
IE_BADID	The device identifier is invalid or unsupported.
IE_BAUDRATE	The device's baud rate is unsupported.
IE_BYTESIZE	The specified byte size is invalid.
IE_DEFAULT	The default parameters are in error.
IE_HARDWARE	The hardware is not available (locked by another program).
IE_MEMORY	Cannot allocate the queues.
IE_NOPEN	The device is not open.
IE_OPEN	The device is already open.

FlowMode Property

Applies to

TSerial component

Declaration

property FlowMode: eFlowControl;

Description

Selects the flow-control mode for the comms port. The following are valid flow modes:

Value	Meaning
fcNone	Use no flow control
fcRTS_CTS	Use RTS/CTS flow control. When CTS is inactive, data transmission is suppressed. When data cannot be accepted, RTS is set inactive.
fcDTR_DSR	Use DTR/DSR flow control When DTR is inactive, data transmission is suppressed. When data cannot be accepted, DSR is set inactive.
fcXON_XOF	Use XON/XOF flow control. For both sending and receiving, an XOF character signals that no more data is to be sent until an XON character is received.

MessageStartChar Property

Applies to

TSerial component

Declaration

property MessageStartChar: Char;

Description

When operating in message mode (RxEventMode=rxMessage) this is the start character that identifies the beginning of a message. It defaults to STX (ASCII 2).

MessageEndChar Property

Applies to

TSerial component

Declaration

property MessageEndChar: Char;

Description

When operating in message mode (RxEventMode=rxMessage) this is the end character that identifies the end of a message. It defaults to ETX (ASCII 3).

MessageAppendCount Property

Applies to

TSerial component

Declaration

property MessageEndCount: Word;

Description

When operating in message mode (RxEventMode=rxMessage) this is the count of characters to be received after the end character defined by MessageEndChar. This allows message extensions such as checksums, that may occur after the message end character, to be captured.

Name Property

Applies to

TSerial component

Declaration

property Name: TComponentName;

Description

The Name property contains the name of the component as referenced by other components. By default, Delphi assigns sequential names based on the type of the component, such as 'Button1', 'Button2', and so on. You may change these to suit your needs.

Note: Change component names only at design time.

NotifyErrors Property

Applies to

TSerial component

Declaration

property NotifyErrors: eNotifyErr;

Description

Selects the action to be taken in the event of a comms error occurring.

Value	Meaning
neDialog	Displays an application-modal dialog box containing a text description of the error.
neEvent	Calls the component's <u>OnError</u> event. The user can supply suitable error-handling code as the event handler.
neNone	Takes no special action. The user should check the component's <u>ErrorCode</u> property regularly to check if any error has occurred.

Owner Property

Applies to

TSerial component

Declaration

property Owner: TComponent;

Description

Run-time and read only. The Owner property indicates which component owns the component.

The form owns all components that are on it. In turn, the form is owned by the application.

When one component is owned by another, the memory for the owned component is freed when its owner's memory is freed. This means that when a form is destroyed, all the components on the form are destroyed also. Finally, when the memory for the application itself is freed, the memory for the form (and all its owned components) is also freed.

Don't confuse ownership of a component with being the parent of a component. A parent is a windowed control that contains a child window. The parent and the owner of a windowed component can be different components.

Port Property

Applies to

TSerial component

Declaration

property Port: sPorts;

Description

Defines the comms port number which the component is currently accessing.

For example :

for COM1: use port := 1

for COM2: use port := 2

Note

The port number can be changed even when the component is active. The component will automatically close and re-open the ports as necessary.

Parity Property

Applies to

TSerial component

Declaration

property Parity: eParity;

Description

Sets the parity bit operation of the port. Note that receive parity checking can be turned off using the CheckParity property, independent of the parity being generated on transmit.

Value	Meaning
paNone	No parity.
paOdd	Odd parity.
paEven	Even parity.
paMark	Parity is stuck at "mark" condition.
paSpace	Parity is stuck at "space" condition.

RxEventMode Property

Applies to

TSerial component

Declaration

property RxEventMode: eOpModes;

Description

Selects the operation mode for processing received data.

Events are generated by the component in response to received data.

OnRxData events signify that a variable amount of data is waiting in the receive queue to be picked up by the user.

OnMessage events signify that a user-defined message has been received in its entirety. The message is passed as a parameter to the event handler.

The events are mutually exclusive : either OnRxData events or OnMessage events will occur but not both.

Value	Meaning
rxNormal	Generate OnRxData events
rxMessage	Generate OnMessage events

RxQueueSize Property

Applies to

TSerial component

Declaration

property RxQueueSize: Word;

Description

Defines the size of the receive queue. This is the maximum number of characters that can be waiting for processing having been received from the port.

RxWaiting Property

Applies to

TSerial component

Declaration

property RxWaiting: Word;

Description

Returns the actual number of characters waiting to be processed in the receive queue.
Run-time only and read-only.

StopBits Property

Applies to

TSerial component

Declaration

property StopBits: eStopBits;

Description

Defines the number of stop bits sent with each character.

Value	Meaning
st1bit	One stop bit
st2bit	Two stop bits

Tag Property

Applies to

TSerial component

Declaration

property Tag: Integer;

Description

The Tag property is available to store an integer value as part of a component. While the Tag property has no meaning to Delphi, your application can use the property to store a value for its special needs.

TxQueueSize Property

Applies to

TSerial component

Declaration

property TxQueueSize: Word;

Description

Defines the size of the transmit queue. This is the maximum number of characters that can be waiting to be sent, having been received from the user application.

TxWaiting Property

Applies to

TSerial component

Declaration

property TxWaiting: Word;

Description

Returns the actual number of characters waiting to be transmitted.
Run-time only and read-only.

eDataBits Type

Declaration

type eDataBits = (d7bit, d8bit) ;

Description

Defines the available options for the TSerial DataBits property.

eFlowControl Type

Declaration

type eFlowControl = (fcNone, fcRTS_CTS, fcDTR_DSR, fcXON_XOF) ;

Description

Defines the available options for the TSerial FlowMode property.

eParity Type

Declaration

type eParity = (paNone, PaOdd, paEven, paMark, paSpace) ;

Description

Defines the available options for the TSerial Parity property.

eStopBits Type

Declaration

type eStopBits = (st1bit, st2bit) ;

Description

Defines the available options for the TSerial StopBits property.

eNotifyErr Type

Declaration

type eNotifyErr = (neDialog, neEvent, neNone) ;

Description

Defines the available options for the TSerial NotifyErrors property.

eOpModes Type

Declaration

type eOpModes = (rxNormal, rxMessage) ;

Description

Defines the available options for the TSerial RxEventMode property.

sPorts Type

Declaration

type sPorts = 1..9 ;

Description

Defines the subrange of allowed values for the TSerial Port property.

TMessageEvent Type

Declaration

type TMessageEvent = procedure (Sender : TObject ; RxMessage : String) of Object;

Description

Defines the type for the event-handler procedure for the OnMessage event.
The RxMessage parameter contains the received message.

OnError Event

Applies to

TSerial component

Declaration

property OnError : TNotifyEvent;

Description

The OnError event occurs when a serial-port error is detected, if the error-handling mode set by the NotifyErrors property is set to neEvent.

When handling this event the user's code should examine the ErrorCode property to determine the nature of the error.

OnRxData Event

Applies to

TSerial component

Declaration

property OnRxData : TNotifyEvent;

Description

The OnRxData event occurs when there is data in the receive queue to be handled, if the received data handling mode set by the RxEventMode property is set to rxNormal.

NOTE : Windows does not guarantee to send a message for every character received, especially at higher baud rates. Therefore, the best plan is to use the OnRxData handler to process **all** the data that is currently in the receive queue, using successive calls to ReadChar or ReadString until the queue is empty.

Example :

```
procedure TForm1.Serial1RxData(Sender: TObject);
var
    in_char : Char;
begin
    while Serial1.ReadChar(in_char) > 0 do
        { process in_char }
    end;
```

OnTxEmpty Event

Applies to

TSerial component

Declaration

property OnTxEmpty : TNotifyEvent;

Description

The OnTxEmpty event occurs when the transmit queue becomes empty. The user event handler can use the event to add more data to the transmit queue using the [WriteChar](#) or [WriteString](#) methods.

Note that a OnTxEmpty event occurs when the TSerial component first becomes active and when the port number is changed.

OnMessage Event

Applies to

TSerial component

Declaration

property OnMessage : TMessageEvent;

Description

The OnMessage event occurs when a message is received, if the received data handling mode set by the RxEventMode property is set to rxMessage.

Message handling works as follows :

A message starts with a character defined by the MessageStartChar property.

A message ends with a character defined by the MessageEndChar property.

The number of characters defined by the MessageAppendCount property will be included **after** the end character.

The maximum length of a message is 255 bytes, since the message is stored as a String.

The RxMessage parameter of the user event handler is a String that contains the received message.

Example :

A message consists of STX (ASCII 2), 10 data characters, ETX(ASCII 3) and a one-byte checksum.

```
Set    MessageStartChar := Chr(2)
        MessageEndChar := Chr(3)
        MessageAppendCount := 1
        RxEventMode := rxMessage
```

The OnMessage event will occur when the entire message has been received.

ReadChar Method

Applies to

TSerial component

Declaration

function ReadChar (var c : Char) : Integer;

Description

Reads one character from the input queue, if available.
The character is returned via the c parameter.

The function returns the number of characters read, which will be either 0 or 1.

ReadString Method

Applies to

TSerial component

Declaration

```
function ReadString (var s : String) : Integer;
```

Description

Reads up to 255 characters from the input queue, as many as are available.
The character is returned via the s parameter.

The function returns the number of characters read, which will be between 0 and 255.

WriteChar Method

Applies to

TSerial component

Declaration

```
procedure WriteChar (c : Char);
```

Description

Writes the character parameter c to the output queue.

WriteString Method

Applies to

TSerial component

Declaration

```
procedure WriteString (s : String);
```

Description

Writes the string parameter s to the output queue.

ZapRxQueue Method

Applies to

TSerial component

Declaration

```
procedure ZapRxQueue;
```

Description

Deletes any data that is currently in the receive queue.

ZapTxQueue Method

Applies to

TSerial component

Declaration

```
procedure ZapTxQueue;
```

Description

Deletes any data that is currently in the transmit queue.

Copyright Notice

This product and documentation is Copyright © R J Crowther, 1995

Versions :

SERIAL	1.04	20/1/96
HELP FILE	1.04	20/1/96

Registration

Registration now brings you the source code for T SERIAL.

Contact the author at :

Mr R J Crowther
20, Howe Road
Haverhill
Suffolk
CB9 9NJ
England

Email :

Compuserve	100255,660
Internet	100255.660@compuserve.com

Compuserve :

The product will be placed on Compuserve's SWREG registration forum. This is by far the easiest method of registration for Compuserve users. The registration code is 9364.

Registration costs **£ 10** or **\$ 15**

I much prefer UK currency for direct orders, as bank conversion costs are high.

Support

Contact the author, preferably by email :

Email : Compuserve 100255,660
 Internet 100255.660@compuserve.com

I cannot guarantee a specific response to unregistered shareware customers. Registered copies will obtain a quicker response !

