Application Note

8421-15950-07

EAN-07

WEIGH-TRONIX EAN-07

WTComm.OCX

ActiveX Control
NCI-Std Serial Protocol

for

Serial RS-232
or
Ethernet (Client/Server)

TITLE: ActiveX Control: WTComm.OCX Control Documentation and Features				
PART: 8421-15950-07	REV: D	ECO:	APPV'D: BA	28-OCT-04

REFERENCE DOCUMENTS:

- **NCI P/N: 8421-15950-07**
  This document
- **NCI P/N: 1150-16225**
  WTComm.OCX ActiveX Component installation CD

Hardware/Software Requirements

In order to install and use the WTComm.OCX ActiveX control described in this document, it is assumed that you have access to the following:

- Personal computer running Windows Me or newer
- NCI Scale that supports the NCI-Std Serial Protocol with RS-232 Interface or Ethernet (Client or Server) Interface. *(see notes 1 and 2)*
- One available RS-232 serial COM port or alternatively one available Ethernet connection on the host PC.
- One serial RS-232 cable or Ethernet cable *(see note 3)*
- Microsoft Visual Basic® 6.0 Programming System (Professional Edition) or other ActiveX compliant development environment.
- NCI WTComm CD: `1150-16225`

Notes:

1)For proper operation, check the user's guide for instructions and verify that the following configuration is enabled in the setup menu; **Protocol = NCI Std**.
2)Check the user's guide to determine whether the scale will function as a 'client' or 'server' device.
3)An ethernet 'crossover' cable will be required if the scale is connected directly to the PC's ethernet port. A 'straight-thru' ethernet cable is required if the scale connects to an ethernet hub which connects to the PC.

Introduction

This document describes the **WTComm** ActiveX control from Avery Weigh-Tronix/NCI. This software component is a true 32-bit OCX control which can be loaded into any ActiveX compliant development including:

- Microsoft Visual Basic v5.0, v6.0 or .NET (Professional Edition)
- Microsoft Visual C++ v5.0, v6.0, or .NET
- Microsoft Office 97

WTComm is a special purpose control which is used to handle all serial (RS-232 or Ethernet Client/Server) communications between the host computer and any scale configured for the NCI-Standard serial protocol. By setting just a few properties in the control, a developer can have instant access to weight and status information from the scale.

This document will describe how to install the control as well as the properties, methods and events that the control provides to the developer.

What's Provided

One (1) self-installing executable contains the installation files for the WTComm ActiveX control.

- **`WTCommActiveX.EXE`**

- To install the ActiveX control:
  1) Start Windows Me, XP (or newer version)
  2) Copy **`WTCommActiveX.EXE`** into any convenient directory
  3) Run (i.e. double-click) **`WTCommActiveX.EXE`**

  Note: the program will automatically install all necessary files and then register the control.

WTComm

Scale Communications

Description: This is a special purpose serial communications control. It allows you to access weight and status information (from any appropriately configured Avery Weigh-Tronix/NCI scale) from within your application. It does this by handling all of the interface requirements defined in the NCI-Standard serial communications protocol (see SCP-01, p/n: 8408-14788-01). Simply connect the scale to an available serial port on your computer, or Ethernet connection on your network, set a few properties in the control and immediately have acces to weight, status and other scale functions.

IMPORTANT: The scale must be configured to use the NCI-Standard Protocol

This is what the control looks like as an icon in the Visual Basic toolbox

File Name: **WTCOMM.OCX**

Object Type: **WTComm**

Remarks: This control provides an event-driven method of handling serial communications from the scale. Each control you use corresponds to one serial RS-232 port or one TCP connection and is used to access data from one scale. If you need to access more than one scale in your application, you must use one **WTComm** scale communications control for each scale. Each serial port address can be set as a property in the controls 'properties' window.

With the event-driven method, your application will be notified the moment an event takes place, such as when a complete weight message has been received from the scale. In such a case you can use the **OnScaleComm** event to determine which event occured and then handle it in your application program.

Since the **WTComm** control uses Microsoft MSComm and Microsoft Winsock as constituent controls, a lot of the details usually necessary to handle scale communications are hidden and taken care of for you. This includes such items as synchronizing received messages, parsing message strings, extracting and converting weight and status information, handling communication errors and detecting scale disconnect.

Properties, Events and Methods

All properties, methods and events for this control are listed in the following table. Scale related properties and events are listed below and are documented in the following sections.

Standard properties and events that are inherited from the MSComm control or Winsock control are marked with an asterisk (*) and may not be documented here. Please refer to your Visual Basic documentation for details.

Properties listed in **bold** lettering are also accessable on the controls property page(s).

PROPERTIES:

AtZero	CalibError	*CommID
***CommPort**	**CommType**	ConnectStatus
EepromError	*Index	InMotion
*Left	*LocalHostName	*LocalIP
***LocalPort**	*Name	**OpMode**
OverCapacity	RamError	***RemoteHost**
*RemoteHostIP	***RemotePort**	RomError
ScaleCommEvent	***Settings**	StatusChars
StatusNotification	*State	**SyncRate**
*Tag	*Top	UnderCapacity
Wt	WtMode	**WtNotification**
WtRange	WtString	WtUnits
ZeroError		

EVENTS:

OnScaleComm

METHODS:

AbortListen	ChangeUnits	GetHiResWt
GetMetroWt	GetStatus	GetStdWt
ScaleOpen	TareScale	ZeroScale

Note: *Each method call should be allowed to complete and return it's response before any subsequent method call is made.*

PROPERTIES

AtZero Property

Description: Returns the center-zero status.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***AtZero**

Remarks: If the scale is "at-zero", the *center-zero* indicator light on the display (if available) will be on and the scale will be in the *auto-zero-tracking (ie AZT)* mode.

True: At zero (ie within center-zero tolerance, typically +/- 1/4d)
False: Not at zero

Data Type: Integer *(True/False)*
Default: n/a

CalibError Property

Description: Returns status which indicates whether or not the scale is calibrated properly.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***CalibError**

Remarks: True: Scale has not been calibrated or it is a faulty calibration.
False: Scale is calibrated ok

Data Type: Integer (True/False)
Default: n/a

CommPort Property

Description: Sets and returns the RS-232 serial communications port number.

Synopsis: [*form.*]*WTComm.***CommPort**[= *portNumber*]

Remarks: You can set *portNumber* to any number between 1 and 99 at design time. However, the scale communications control generates error 68 (Device Unavailable) if the port does not exist when you attempt to open it with the ScaleOpen method.

Note: This is the property used by the MSCOMM constituent control.

Data Type: Integer
Default: 1

Warning: *You must set the CommPort property before opening the port using the ScaleOpen method.*

CommType Property

Description: Sets and returns the communications type (i.e. RS-232 or TCP/NCI)
This property is available at design-time and is read/write at run time.

Synopsis: [*form.*]*WTComm.***CommType**[= *typeNumber*]

Remarks: This is the setting which determines what hardware interface will be used to communicate with the scale.

Warning: *Do not change the CommType property after successfully opening the port (using the ScaleOpen method).*

Setting	Description
wtRS232	Serial RS-232 communications: PC app is 'master', scale is 'slave'
wtTCPClient	Serial Ethernet communications: PC app is 'client', scale is 'server'
wtTCPServer	Serial Ethernet communications : PC app is 'server', scale is 'client'

Data Type: WT_COMMTYPE_ENUM
Default: wtRS232

ConnectStatus Property

Description: Returns the current scale connection status.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***ConnectStatus**

Remarks: This is the value returned which indicates whether or not the scale is communicating with the computer. The following table lists the ConnectStatus property settings for the scale communications control.

Setting	Description
wtSCALE_OFFLINE	The scale is not communicating (comm port closed)
wtSCALE_ONLINE	The scale is communicating properly.

Data Type: Integer
Default: n/a

EepromError Property

Description: Returns the results of the most recent scale RAM memory test.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***EepromError**

Remarks: True: EEPROM error (ie test **failed**)
False: EEPROM ok (ie test **passed**)

Data Type: Integer *(True/False)*
Default: n/a

InMotion Property

Description: Returns the in-motion status.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***InMotion**

Remarks: True: In-motion (as determined by the particular scale)
False: Stable

Data Type: Integer *(True/False)*
Default: n/a

LocalHostName Property

Description: Returns the local machine name.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***LocalHostName**

Remarks:

Data Type: String
Default: n/a

LocalIP Property

Description: Returns the IP address of the local machine.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***LocalIP**

Remarks: IP address will be in the dotted string format (e.g. `xxx.xxx.xxx.xxx`)

Data Type: String
Default: n/a

LocalPort Property

Description: Sets or returns the local port to use.
This property is available at design-time and is read/write at run time.

Synopsis: [*form.*]*WTComm.***LocalPort**

Remarks: For the server, this is the local port to 'listen' on. If port 0 is specified, a random port is used. After Winsock invokes the Listen method as part of ScaleOpen, the property will contain the actual port that has been used.

Data Type: Long
Default: n/a

OpMode Property

Description: Sets and returns the operational mode (ie method of weight/status access).
This property is available at design-time and may be set and read at run time.

Synopsis: [*form.*]*WTComm.***OpMode**

Remarks: The operational mode determines whether weight/status will be accessed "manually" (ie **async**hronously) by making calls to specific methods, or "automatically" (ie **sync**hronously) by the control itself. In the synchronous mode, the rate at which weight/status information is updated is determined by the **SyncRate** property. The "type" of weight synchronously updated is specified by the **OpMode** property. When any of the synchronous modes are selected the users application will be notified by an event based on the **WtNotification** and/or **StatusNotification** property settings.

See the section entitled ENUMERATED CONSTANTS for a description of the various operational modes that are available.

Data Type: WT_OPMODE_ENUM
Default: wtAsync

OverCapacity Property

Description: Returns status indicating whether or not the current weight exceeds the scales maximum capacity limit. This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***OverCapacity**

Remarks: The criteria for overcapacity may vary for different scales. Please refer to the scale literature for specific overcapacity limit. It is typically +9d above rated capacity.

True: Overcapacity
False: Not Overcapacity

Data Type: Integer *(True/False)*
Default: n/a

RamError Property

Description: Returns the results of the most recent scale RAM memory test.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***RamError**

Remarks: True: RAM error (ie test **failed**)
False: RAM ok (ie test **passed**)

Data Type: Integer *(True/False)*
Default: n/a

RemoteHost Property

Description: Sets and returns the TCP communications remote host IP address or DNS name.
This property is available at design-time and is read/write at run time.

Synopsis: [*form.*]*WTComm.***RemoteHost**[= *hostString*]

Remarks: You can set *hostString* to any valid IP address (e.g. "`192.168.1.0`") or it's 'friendly DNS address (e.g. "`FTP://ftp.microsoft.com`"

Note: This is the property used by the MSWINSOCK constituent control.

Data Type: String
Default: n/a

Warning: *You must set the RemoteHost property before attempting to open the ethernet communications channel with the ScaleOpen method.*

RemoteHostIP Property

Description: Returns the IP address of the remote machine.
This property is not available at design-time and is read-only at run time.

For client applications, after an outgoing connection has been established, this property contains the IP string of the remote (server) machine.

For server applications, after an incoming connection has been established, this property contains the IP string of the remote machine (client) that initiated the connection.

Synopsis: [*form.*]*WTComm.***RemoteHostIP**

Remarks: IP address will be in the dotted string format (e.g. xxx.xxx.xxx.xxx)

Data Type: String
Default: n/a

RemotePort Property

Description: Sets and returns the TCP communications remote port number.
This property is available at design-time and is read/write at run time.

Synopsis: [*form.*]*WTComm.***RemotePort**[= *portNumber*]

Remarks: You can set *portNumber* to any valid IP port number at design time or run-time. The port number must match the setting used by the scale.

Note: This is the property used by the MSWINSOCK constituent control.

Data Type: Long
Default: 10001

Warning: *You must set the RemotePort property before attempting to open the ethernet communications channel with the ScaleOpen method.*

RomError Property

Description: Returns the results of the most recent scale ROM memory test.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***RomError**

Remarks: True: EEPROM error (ie test **failed**)
False: EEPROM ok (ie test **passed**)

Data Type: Integer *(True/False)*
Default: n/a

ScaleCommEvent Property

Description: Returns the most recent scale event.
This property is not available at design time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***ScaleCommEvent**

Remarks: The ScaleCommEvent property holds the numeric code for the event that caused the OnScaleComm event to occur. Although the MSCOMM constituent control is generating many more serial communications events, the ScaleCommEvent is generated only under certain conditions. The events are described in the following table.

Setting	Description
wtWTCOMM_EV_WEIGHT	weight received (or changed)
wtWTCOMM_EV_STATUS	status received (or changed)
wtWTCOMM_EV_OFFLINE	serial port closed and communications with scale terminated
wtWTCOMM_EV_TIMEOUT	scale not responding (or timed out)
wtWTCOMM_EV_ERROR	scale communications error

Data Type: Integer
Default: n/a

Settings Property

Description: Sets and returns the baud rate, parity, data bit and stop bit parameters used by the RS-232 serial communi cations interface.
This property is not available at design-time and is read/write at run time.

Synopsis: [*form.*]*WTComm.***Settings**[= *paramString*]

Remarks: paramString is composed of four settings and has the following format:

"BBBB,P,D,S" for example: "`9600,E,8,1`"

Where BBBB is the baud rate, P is the first letter of the parity type, D is the number of data bits, and S is the number of stop bits. Valid parameters for each of these four fields is dictated by the particular capabili ties of the scale.

Note: This is the property used by the MSCOMM constituent control.

Data Type: String
Default: n/a

State Property

Description: Returns the state of the Winsock TCP communications control.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***State**

Remarks:

Note: This is the property used by the MSWINSOCK constituent control.

Data Type: Integer
Default: n/a

StatusChars Property

Description: Returns the most recent scale status code string as received from the scale.
This value is read only at run time and is not available at design time.

Synopsis: [*form.*]*WTComm.***StatusChars**

Remarks: Most scales that conform to the NCI Standard Serial Protocol specification SCP-01 (p/n: 8408-14788) will return three scale status characters within the command response string. Some scales may return only two characters of scale status. In either case, this property extracts the two (or three) scale status characters from the most recent scale response and returns it as a string. If the most recent command did not return a scale status in its response or if the scale has not yet been interrogated, then the status string will be null and none of the individual scale status property values should be considered valid.

The most significant bit (ie parity) in each character is masked off. However, all other bits in each status character (seven total), are returned exactly as received and as described in the SCP-01 document.

Data Type: String
Default: null string

StatusNotification Property

Description: Sets and returns the condition under which the OnScaleComm event will be generated for status.

Synopsis: [*form.*]*WTComm.***StatusNotification** [= {**wtNoStatusNotification** | **wtEveryStatus** | **wtChangedStatus**}]

Remarks: You can set this property to cause the OnScaleComm event to occur everytime a status is read from the scale or only when the status value has changed (or not at all).

Note: If the property is set to *wtEveryStatus*, the event will be generated everytime a status is received from the scale (including status received as part of a weight reading). If the property is set to *wtChangedStatus*, then the event will be generated only when the status (ie **StatusChars**) changes from one reading to the next. In either case, an additional event will be generated if a call to one of the following methods also meets the criteria: **GetStatus**, **GetHiResWt**, **GetMetroWt** and **GetStdWt**.

Data Type: Integer
Default: wtNoStatusNotification

SyncRate Property

Description: Sets and returns the rate for synchronous mode operations.
This property is available at design-time and may be set and read at run time.

Synopsis: [*form.*]*WTComm.***SyncRate**

Remarks: The **OpMode** determines whether weight/status will be accessed "manually" (ie **async**hronously) by making calls to specific methods, or "automatically" (ie **sync**hronously) by the control itself. In the synchronous mode, the rate at which weight/status information is updated is determined by the **SyncRate** property. The "type" of weight synchronously updated is specified by the **OpMode** property. When any of the synchronous modes are selected the users application will be notified by an event based on the **WtNo**

tification and/or **StatusNotification** property settings.

See the section entitled ENUMERATED CONSTANTS for a description of the various synchronous rates that are available.

Data Type: WT_SYNCRATE_ENUM
Default: wtSyncFastest

UnderCapacity Property

Description: Returns status indicating whether or not the current weight exceeds the scales minimum capacity limit. This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***UnderCapacity**

Remarks: The criteria for undercapacity may vary for different scales. Please refer to the scale literature for specific undercapacity limits.

True: Undercapacity
False: Not Undercapacity

Data Type: Integer *(True/False)*
Default: n/a

Wt Property

Description: Returns the current net weight value as received from the scale.
This value is read only at run time and is not available at design time.

Synopsis: [*form.*]*WTComm.***Wt**

Remarks: This value will be weight (in the current units-of-measure) in single precision suitable for direct use in calculations.

Note: If the current scale units-of-measure is *pounds-ounces*, then the net weight will be converted and presented in the decimal ounces equivalent.

Data Type: Single
Default: n/a

WtMode Property

Description: Indicates whether the standard weight value returned is the **net** weight or **gross** weight.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***WtMode**

Remarks: Normally, all weights are *gross* weights. If the scale supports the Tare feature and a weight (of a container for example) has been "tared", then the weight returned will be *net* weight.

wtNetMode *(net weight, a tare is in effect)*
wtGrossMode *(gross weight, no tare in effect)*

Data Type: WT_MODE_ENUM
Default: n/a

WtNotification Property

Description: Sets and returns the condition under which the OnScaleComm event will be generated.

Synopsis: [*form.*]*WTComm.***WtNotification** [= {**wtNoWtNotification | wtEveryWeight** | **wtChangedWeight**}]

Remarks: You can set this property to cause the OnScaleComm event to occur everytime a weight is read from the scale, only when the weight value has changed, or never.

Note: If the property is set to *wtEveryWeight*, the event will be generated at the rate determined by the **UpdateRate** property. If the property is set to *wtChangedWeight*, then the event will be generated only when the weight (ie **WtString**) changes from one reading to the next. In either case, an additional event will be generated if a call to one of the following methods also meets the criteria: **GetHiResWt**, **GetMetroWt** and **GetStdWt**.

Data Type: Integer
Default: wtNoWtNotification

WtRange Property

Description: Returns the current range of operation for a multi-ranging scale.
This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***WtRange**

Remarks: Multi-ranging scales have two ranges of operation which are indicated in this property using the following enumerations:

wtLowRange *(the low capacity range of operation)*
wtHighRange *(the high capacity range of operation)*

Data Type: WT_RANGE_ENUM
Default: n/a

WtString Property

Description: Returns the original net weight and units of measure only string value as received from the scale. This value is read only at run time and is not available at design time.

Synopsis: [*form.*]*WTComm.***WtString**

Remarks: The <LF> preamble is stripped off. Also, all characters from <CR><LF> through <CR><ETX> are stripped off before the string is returned.

The following examples show three typical weight responses that may be received from the scale and the actual string value that would be returned in the **WtString** property.

```
<LF> X X X X . X X u u <CR> <LF> h h h <CR> <ETX>          <- typical lb, kg, oz etc weight
     X X X X . X X u u                                     <- WtString returns only

<LF> X X l b _ X X . X X o z <CR> <LF> h h h <CR> <ETX>    <- typical lb-oz weight
     X X l b _ X X . X X o z                               <- WtString returns only

<LF> X X X X X X M M <CR> <LF> h h h <CR> <ETX>           <- typical metro counts
     X X X X X X M M                                       <- WtString returns only
```

Data Type: String
Default: n/a

WtUnits Property

Description: Returns the current units-of-measure for the **Wt** property value. This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***WtUnits** = *enumUnits*

Remarks: The units-of-measure may be one from the following enumerated list.

Setting *(enumUnits)*	**Description**
wtNotAvailable	A weight reading has not yet been performed. Therefore the current units-of-measure has not been determined.
wtMetro	**Wt** is will be in “Metrology Counts” (single precision float)

wtPounds	Weight is in pounds (single precision float)
wtKilograms	Weight is will be in kilograms (single precision float)
wtGrams	Weight is will be in grams (single precision float)
wtOunces	Weight is will be in ounces (single precision float)
wtPoundsOunces	Weight is will be converted to ounces (single precision float)

Data Type: Integer (enumerated constants)
Default: 1

ZeroError Property

Description: Returns the current scale status which indicates whether or not the zero error condition exists. This property is not available at design-time and is read-only at run time.

Synopsis: [*form.*]*WTComm.***ZeroError**

Remarks:
True: The scale is unable to be zeroed due to some error.
False: The scale is able to be zeroed.

Data Type: Integer *(True/False)*
Default: n/a

EVENTS

OnScaleComm Event

Description: The OnScaleComm event is generated whenever the value of the ScaleCommEvent property changes.

Synopsis: **Sub** [*form.*]*WTComm*.**OnScaleComm()**

Remarks: The ScaleCommEvent property contains the numeric code of the actual cause of the event that generated the OnScaleComm event. See **ScaleCommEvent** in the section entitled WTCONST.BAS for a list of event codes.

When an event occurs, your application should determine which specific event it was that triggered the event and then provide additional handling code as necessary.

Data Type: n/a
Default: n/a

METHODS

AbortListen Method

Description: Commands the Winsock control to abort its 'listening' mode.

Synopsis: [*form.*]*WTComm.***AbortListen** ()

Remarks: When the ScaleOpen method is called and the CommType is configured for Ethernet (server) mode, the Winsock constituant control of WTComm will be placed in the 'Listen' state. It will remain in that state until it either receives a connection request from a client (and completes the connection) or is aborted by calling this **AbortListen** method.

Return Value: none
Default: n/a

ChangeUnits Method

Description: Commands the scale to change to the next available units-of-measure selection (if available).

Synopsis: [*form.*]*WTComm.***ChangeUnits** () as Boolean

Remarks: You can use this method with scales that support more than one units-of-measure and you want to change to the next available selection. If successful, the new units-of-measure will be indicated in the *NetWtUnits* property setting.

Note: All status related properties are also updated as a result of calling this method.

Return Value: Boolean - **True** if successful, **False** otherwise.
Default: n/a

GetHiResWt Method

Description: Returns the current (high-resolution) scale weight as a single precision type value which can be used directly in calculations by the application program.

Synopsis: [*form.*]*WTComm.***GetHiResWt** (ByRef psngWtVar as Single) as Boolean

Remarks: You can use this method when you want the actual weight value (ie not a formatted string) for use in calculations that are based on weight. The weight value will be in the units-of-measure currently active in the scale and as described by the *NetWtUnits* property setting. The single precision value will overwrite the current value in the variable referenced by the psngWtVar parameter. The *NetWt* property is also updated as a result of calling this method.

Parameter	Type/Description		
psngWtVar	Single	-	Name of the users variable which will be used to receive the high-resolution weight value.

Note: All status related properties are also updated as a result of calling this method.

Return Value: Boolean - **True** if successful, **False** otherwise (variable value set to 0.0).
Default: n/a

GetMetroWt Method

Description: Returns the current scale weight (in metrology counts) as a single precision type value which can be used directly in calculations by the application program.

Synopsis: [*form.*]*WTComm.***GetMetroWt** (ByRef psngWtVar as Single) as Boolean

Remarks: You can use this method when you want the actual normalized weight counts value (ie not a formatted string) for use in calculations. The weight value (in counts) is "unit-less" and is independent of the scales current units-of-measure setting. The single precision value will overwrite the current value in the variable referenced by the psngWtVar parameter. The *NetWt* property is **NOT** updated as a result of calling this method. Counts typically range 0-100,000 or 0-1 million and have no fractional parts.

Parameter	Type/Description		
psngWtVar	Single	-	Name of the users variable which will be used to receive the metrology counts value.

Note: All status related properties are updated as a result of calling this method.

Return Value: Boolean - **True** if successful, **False** otherwise (variable value set to 0.0).
Default: n/a

GetStatus Method

Description: Commands the scale to return the current scale status string.

Synopsis: [*form.*]*WTComm.***GetStatus** (ByRef pstrSclStat as String) as Boolean

Remarks: You can use this method read the most current scale status string into a string variable that you specify. Only the actual status characters are returned (typically two or three characters). The preamble and postamble special communications characters are removed. The most significant bit (ie the parity bit), of each character, is masked to zero before it is returned. All seven other bits in each character are retained as described in the serial protocol specification.

Parameter	Type/Description		
pstrSclStat	String	-	Name of the users string variable which will be used to receive the scale status string. The string must be large enough to accomodate the largest scale status string likely to be read.

Note: All status related properties are also updated as a result of calling this method.

Return Value: Boolean - **True** if successful, **False** otherwise (and user specified status string set to null).
Default: null string

GetStdWt Method

Description: Returns the current (standard resolution) scale weight as a single precision type value which can be used directly in calculations by the application program.

Synopsis: [*form.*]*WTComm.***GetStdWt** (ByRef psngWtVar as Single) as Boolean

Remarks: You can use this method when you want the actual weight value (ie not a formatted string) for use in calculations that are based on weight. The weight value will be in the units-of-measure currently active in the scale and as described by the *NetWtUnits* property setting. The single precision value will overwrite the current value in the variable referenced by the psngWtVar parameter. The *NetWt* property is also updated as a result of calling this method.

Parameter	Type/Description
psngWtVar	Single - Name of the users variable which will be used to receive the standard resolution weight value.

Note: All status related properties are also updated as a result of calling this method.

Return Value: Boolean - **True** if successful, **False** otherwise (and user variable set to 0.0).
Default: n/a

ScaleOpen Method

Description: Sets and returns the state of the scale communications port (open or closed).
This property is not available at design time.

Synopsis: **Function** [*form.*]*WTComm*.**ScaleOpen (** {**True** | **False**} **)** as Integer

Remarks: The following table lists the ScaleOpen parameter settings for the scale communications control.

Setting	Description
True	Open the port and establish communications with scale.
False	Close the port.

Calling the ScaleOpen method with the parameter set to True opens the serial port and establishes the communications link with the scale. Setting it to False closes the port. The communications control automatically closes the serial port when your application terminates.

Note: This is similar to the PortOpen property used by the MSCOMM constituent control with the additional requirement that a communications link with the scale also be established.

Data Type: Integer
Return Value: Integer - **wtSCALE_ONLINE** or **wtSCALE_OFFLINE**
Default: n/a

TareScale Method

Description: Commands the scale to tare (ie subtract out) the current weight on the scale.

Synopsis: [*form.*]*WTComm.***TareScale** () as Boolean

Remarks: You can use this method subtract the weight of an empty container on the scale so that its' weight will not be included in the total product weight. When a tare value is *in effect*, the status will indicate that weight readings are **net**. Otherwise, weight readings are **gross**. Not all scales support the tare function.

Note: All status related properties are also updated as a result of calling this method.

Return Value: Boolean - **True** if successful, **False** otherwise.
Default: n/a

ZeroScale Method

Description: Commands the scale to zero the current weight on the scale.

Synopsis: [*form.*]*WTComm.***ZeroScale** () as Boolean

Remarks: You can use this method to zero the scale of any/all weight that might be on the scale. The scale will be at *center-zero* (as indicated by the **AtZero** status property) and a*utomatic-zero-tracking* will be in effect. Some scales have restrictions on the maximum value that can be zeroed. Please refer to the scales literature or details.

Note: All status related properties are also updated as a result of calling this method.

Return Value: Boolean - **True** if successful, **False** otherwise.
Default: n/a

Enumerated Constants

Enumerated Constant Definitions

Description: This is the public constant declarations list for enumerated constants defined in the **WtComm** ActiveX control (NCI p/n: 1150-16225) developed by Weigh-Tronix/NCI. The developer should use these constants when accessing various properties, events and methods in the scale communications control.

Note: Since these constants are defined and made public in the WTComm control, no additional file needs to be added to the project in order to use these constants.

NetWtUnits	**(WT_UOM_ENUM)**
wtNotAvailable	NetWt value units-of-measure not yet determined
wtMetro	NetWt value is in normalized metrology counts
wtPounds	NetWt value is in pounds
wtKilograms	NetWt value is in kilograms
wtGrams	NetWt value is in grams
wtOunces	NetWt value is in ounces
wtPoundsOunces	NetWt value is in ounces (converted from pounds-ounces)

WtNotification	**(WT_NOTIFY_ENUM)**
wtNoWtNotification	(no event fired)
wtEveryWeight	(everytime a weight is received)
wtChangedWeight	(only when the weight value has changed)

StatusNotification	**(WT_STATUSNOTIFY_ENUM)**
wtNoStatusNotification	(no event fired)
wtEveryStatus	(everytime a status is received)
wtChangedStatus	(only when the status value has changed)

WtRange	**(WT_RANGE_ENUM)**
wtLowRange	(in low-range of dual range scale)
wtHighRange	(in high-range of dual range scale)

WtMode	**(WT_MODE_ENUM)**
wtNetMode	(net weight)
wtGrossMode	(gross weight)

OpMode	**(WT_OPMODE_ENUM)**
wtAsync	(asynchronous, update only when method called)
wtSyncStdReso	(synchronous, standard-resolution weight)
wtSyncHighReso	(synchronous, high-resolution weight)
wtSyncMetro	(synchronous, metrology counts)

*Note: synchronous updates at rate specified in **SyncRate** property.*

SyncRate	**(WT_SYNCRATE_ENUM)**
wtSyncFastest	(fastest rate, limited by scale master/slave response)
wtSync1PerSec	(update 1/sec)
wtSync2PerSec	(update 2/sec)
wtSync3PerSec	(update 3/sec)
wtSync4PerSec	(update 4/sec)
wtSync5PerSec	(update 5/sec)
wtSync6PerSec	(update 6/sec)
wtSync7PerSec	(update 7/sec)
wtSync8PerSec	(update 8/sec)
wtSync9PerSec	(update 9/sec)
wtSync10PerSec	(update 10/sec)

State (Winsock)	**Value**	**Description**
sckClosed	0	Default. Closed
sckOpen	1	Open
sckListening	2	Listening
sckConnectionPending	3	Connection pending
sckResolvingHost	4	Resolving host
sckHostResolved	5	Host resolved
sckConnecting	6	Connecting
sckConnected	7	Connected
sckClosing	8	Peer is closing the connection
sckError	9	Error

WTCONST.BAS

Constant Definitions

Description: This is the public constant declarations file for use with the **WTComm** ActiveX control (NCI p/n: 1150-15950-07) developed by Weigh-Tronix/NCI.

The developer must add this file to the application project to use these constants when accessing various properties, events and methods in the scale communications control.

ConnectStatus

wtSCALE_OFFLINE = 0
wtSCALE_ONLINE = 1

ScaleStatus

ScaleCommEvent

wtWTCOMM_EV_WEIGHT = 1	(weight updated or changed)
wtWTCOMM_EV_STATUS = 2	(status changed)
wtWTCOMM_EV_OFFLINE = 3	(serial port has been closed and comm with scale terminated)
wtWTCOMM_EV_TIMEOUT = 4	(scale not responding, or timed out)
wtWTCOMM_EV_ERROR = 5	(scale communications error, see note)

Note: scale communications errors include framing, data lost, transmit buffer full, receive buffer full, parity,and port DCB error.

Serial Communications Notes:

A TIMEOUT event will be generated if the scale does not respond to a command within approximately three (3) seconds.

Any combination of three consecutive TIMEOUT or ERROR events will result in an automatic call to the ScaleOpen(False) method (to close the serial port and halt communication attempts) and then the OFFLINE event will be generated.

The scale must be configured for NCI Standard protocol.

Weighing Products & Systems

Postal Scales

POS Scales

Dot Matrix
Impact Printers

Thermal Graphic
Label Printers

U-Mail® Desktop
Mailing System

www.wt-nci.com

Avery Weigh-Tronix
3990 Brickway Blvd.
Santa Rosa, CA. 95403-1070

Tel: (707) 527-5555
Fax: (707) 579-0180