



TEC-IT

WWW.TEC-IT.COM

TWedgeCE

Data Acquisition for Mobile Devices

Version 2.0

User Documentation

January 19, 2010

TEC-IT Datenverarbeitung GmbH
Wagnerstrasse 6
A-4400 Steyr, Austria

t ++43 (0)7252 72720
f ++43 (0)7252 72720 77
office@tec-it.com
www.tec-it.com

1 Content

1	Content	2
1.1	List of Tables	2
1.2	List of Figures	3
2	Introduction	4
2.1	Abstract	4
2.2	Supported Operating Systems	4
3	User Interface	5
3.1	Main Window	5
3.2	Debug Window	6
3.3	User Interface under Windows CE	7
4	Settings	8
4.1	Introduction	8
4.2	Settings in Detail	8
4.2.1	Startup Options	8
4.2.2	General Communication Parameters	8
4.2.3	Parameters for TPC/IP, UPD and Bluetooth	9
4.2.4	Parameters for Serial Communication	10
4.2.5	Rules for Data Packaging	10
4.2.6	String Replacement Table	12
4.2.7	Control Sequence Settings	12
4.2.8	Pre-Evaluation Expression	13
4.2.9	Settings for the Tray Menu	13
4.2.10	Operation Mode	13
4.3	Example Configurations	14
4.3.1	No Newline, No Linefeed	14
4.3.2	MS Excel	14
4.3.3	Replacement Table	14
4.3.4	Extract Sub-String from the acquired Data	14
5	Examples of Use	15
5.1	Example 1	15
6	Available Licenses	16
6.1	Retail License	16
6.2	Demo License	16
6.3	Pricing	16
7	Contact and Support Information	17
Appendix A : Control Commands		18
Appendix B : Function Reference		20
B.1	Functions	20
B.2	Constants	21

1.1 List of Tables

Table 1: Supported Operating Systems	4
Table 2: Status Icons	5
Table 3: Startup Parameters	8
Table 4: General Communication Parameters	9
Table 5: TCP/IP Parameters	9
Table 6: Serial Communication Parameters	10
Table 7: Data packaging parameters	11
Table 8: Standard escape sequences	12
Table 9: Extended escape sequences	12
Table 10: String replacement samples	12
Table 11: Control sequence parameters	13
Table 12: Pre-Evaluation parameters	13
Table 13: Tray menu parameters	13
Table 14: Keyboard emulation parameters	14

Table 15: TWedgeCE control commands	19
Table 16: Functions	21
Table 17: Constants	21

1.2 List of Figures

Figure 1: TWedgeCE Main Window (Windows Mobile 5.0)	5
Figure 2: TWedgeCE Debug Window (Windows Mobile 5.0)	6
Figure 3: TWedgeCE Main Window (Windows CE)	7
Figure 4: TWedgeCE Debug Window (Windows CE)	7

2 Introduction

2.1 Abstract

TWedgeCE is a generic data acquisition application for Microsoft® Windows® CE based handheld devices. It captures data from an arbitrary device interface and simulates keystrokes based on the captured data. These keystrokes are sent subsequently to an application. So it is possible to input data from external devices – such as bar code readers, electronic scales (and many more) – into Microsoft Pocket Excel®, e.g.

TWedgeCE supports the following interfaces:

- RS-232 Interface (serial connection via COM ports)
- Bluetooth® via COM simulator (utilizing the Serial Interface Service in the Bluetooth Stack)
- Native Bluetooth® (client mode)
- TCP/IP (client and server mode)
- UDP (client mode)
- Built-in barcode scanner (only on Symbol® PPT8800)

This document explains the user interface of *TWedgeCE*. Subsequently the program settings are explained.

2.2 Supported Operating Systems

TWedgeCE 2 is available for:




Operating System	Appropriate Download
<ul style="list-style-type: none">▪ Pocket PC 2003 SE (= Second Edition)	 TWedgeCE_PPC2003_SE.exe
<ul style="list-style-type: none">▪ Microsoft Windows Mobile 5▪ Microsoft Windows Mobile 6	 TWedgeCE_WinMobile5.exe
<ul style="list-style-type: none">▪ Microsoft Windows CE 5▪ Microsoft Windows Embedded CE 6	 TWedgeCE_WinCE5_ARMV4I.exe ¹

Table 1: Supported Operating Systems

¹ As indicated by its file name this version is for ARMV4I processors only (e.g. Intel PXA272). Builds for other processors are available on request.

3 User Interface

3.1 Main Window

After starting *TWedgeCE* the following screen appears:

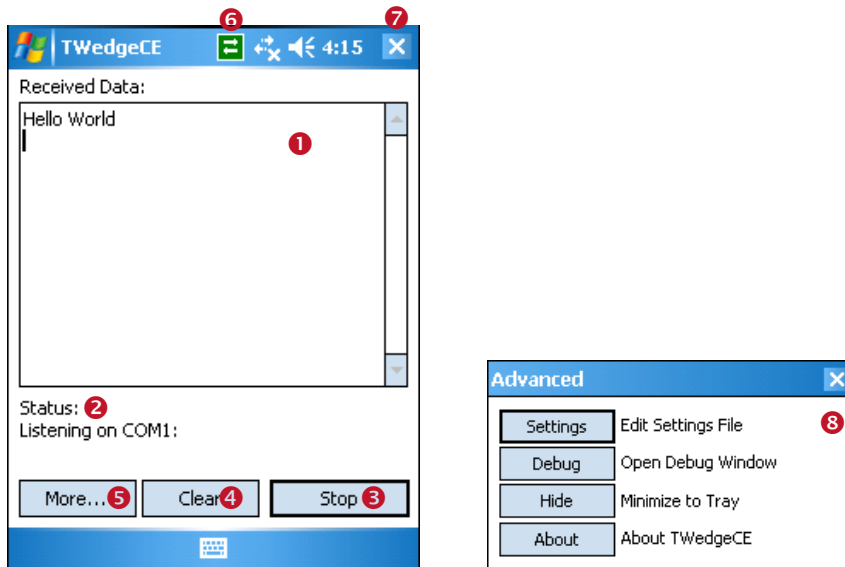


Figure 1: *TWedgeCE* Main Window (Windows Mobile 5.0)

The Start/Stop button **3** allows you to connect/disconnect to the serial device (e.g. to a barcode scanner). If the application is started, the button caption will change to “Stop”. If *TWedgeCE* is started all incoming data is sent to the active application window.

The status area **2** shows the current connection state of *TWedgeCE*. It also shows how many bytes were recently received.

By default the input will be displayed in the text area of *TWedgeCE* **1**. If you want the input to be passed to any other application (e.g. Pocket Word) start the desired application and make sure that it has the input focus. The data will be sent to the actual position of the text cursor. (Prefix, postfix, delimiter settings and the translation table are applied as described under 4.2.5, 4.2.6 and 4.2.7).

Button **4** clears the text input area **1**.

By clicking **5** the menu dialog **8** appears. Using this dialog you can open the program settings, the debug window and the about box. Or you can hide the *TWedgeCE* window.

Icon **6** shows the status of the program:

	<i>TWedgeCE</i> is stopped (not connected).
	<i>TWedgeCE</i> is started (waiting for data).
	<i>TWedgeCE</i> is started, client is not connected (TCP Server only).

Table 2: Status Icons

By clicking on the icon you can bring up a menu which allows you to start, stop, show, hide or exit *TWedgeCE*.

The icon can be turned off on demand (see section 4.2.9, Settings for the Tray Menu).

7 closes *TWedgeCE*.

3.2 Debug Window

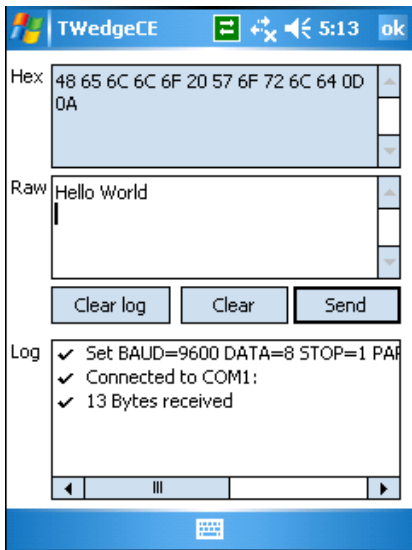


Figure 2: TWedgeCE Debug Window (Windows Mobile 5.0)

TWedgeCE allows you to view some additional information about the received data on the debug screen. You can bring up this screen by clicking the buttons *More...* ► *Debug* in the main window.

- ▶ The Hex box displays the input in hexadecimal codes.
 - ▶ The Raw box displays the received data as plain text.
 - ▶ The Log window monitors some information about the connection.
 - ▶ The Send button sends the content of the Raw box to the open connection.
- Note: For sending data the connection must support writing! (The Serial Port Service provided by the Bluetooth stack, e.g., uses one port exclusively for reading and another port exclusively for writing.) Furthermore the external device must be capable of receiving data.

3.3 User Interface under Windows CE

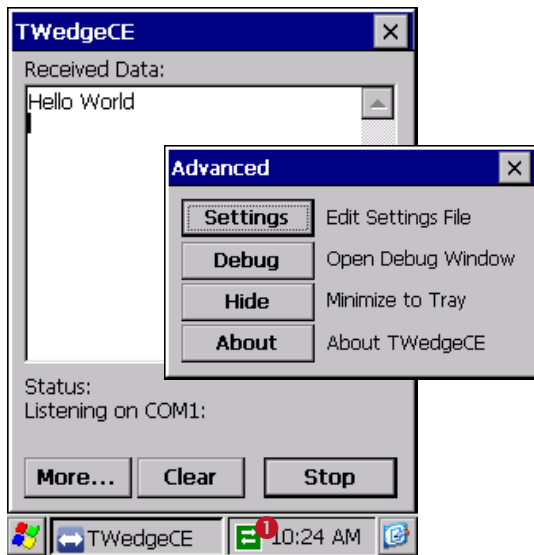


Figure 3: TWedgeCE Main Window (Windows CE)

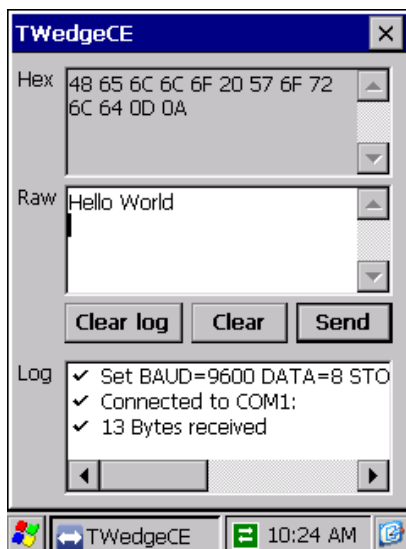



Figure 4: TWedgeCE Debug Window (Windows CE)

Under Windows CE the icon  is located in the system tray in the lower right corner of the screen and not on top of the screen as under Windows Mobile.

4 Settings

4.1 Introduction

TWedgeCE is customized via a single settings file (named "*TWedgeCE.txt*"). This makes it easy to deploy the settings to several mobile devices. You can edit this file by clicking the button *More...* and then *Settings*. A dialog for editing the file appears.

When starting *TWedgeCE* for the first time a standard configuration file will be created. This file contains the following settings:

- Startup Options (see section 4.2.1)
- General Communication Parameters (see section 4.2.2)
- Parameters for TCP/IP, UPD and Bluetooth (see section 4.2.3)
- Parameters for Serial Communication (see section 4.2.4)
- Rules for Data Packaging (see section 4.2.5)
- A String Replacement Table (see section 4.2.6)
- Control Sequence Settings (see section 4.2.7)
- A Pre-Evaluation Expression (see section 4.2.8)
- Settings for the Tray Menu (see section 4.2.9)
- Operation Mode (see section 4.2.10)

4.2 Settings in Detail

4.2.1 Startup Options

Section [Startup] in the ini-file


AutoStart=N	Automatically connect <i>TWedgeCE</i> on startup? When set to "Y" <i>TWedgeCE</i> automatically starts to connect as soon as the program is launched. Possible values are: N, Y
AutoHide=N	Hide <i>TWedgeCE</i> on startup? When set to "Y", the <i>TWedgeCE</i> window will be hidden upon startup. Only the tray icon (see Figure 1, ) will be visible. This option requires the tray icon to be enabled (see section 4.2.9). Possible values are: N, Y

Table 3: Startup Parameters

4.2.2 General Communication Parameters

Section [Com Settings]

IOType=5	IO Type. Possible values are: 3 ...Serial Interface (see also section 4.2.3) 5 ...TCP/IP Client (see also section 4.2.4) 6 ...TCP/IP Server (see also section 4.2.4) 7 ...Bluetooth Client (see also section 4.2.4) 8 ...UDP Client (see also section 4.2.4)
Unicode=N	Receive and transmit Unicode strings? (Unicode uses 2 bytes per character.) If Unicode=N ASCII encoding will be used.
ReadTimerInterval=250	This setting only applies for IOType=3 (Serial Interface)! It specifies the time interval (in milliseconds), in which the COM port is checked for pending data.
RetryCount=1	Retry count.

	<p>If you cannot connect or if you get disconnected <i>TWedgeCE</i> will try to reconnect for the given number of times before reporting a connection error.</p> <p>This setting can be useful when you lost the connection to a Bluetooth device (out of range) or when you turned off your mobile device without closing (and without stopping) <i>TWedgeCE</i>. When turning the device back on the connection will be re-established, which may take a couple of retries.</p> <p>Possible values range from -1 to 255.</p> <p>-1 ...retry forever 0 ...no retry (= disabled) 1..255 ...one to 255 retries</p>
RetryTimerInterval=5000	<p>Timer interval for connection retries (in milliseconds).</p> <p>When the connection is lost <i>TWedgeCE</i> will wait the given number of milliseconds before trying to reconnect.</p> <p>Possible values range from 1000 to 10000 (1 to 10 seconds).</p>
WriteTimeout=2000	<p>Write timeout in milliseconds</p> <p>This timeout is used for sending data (e.g., in the Debug Window or when using the Check-Alive functionality, see section 4.2.3).</p> <p>Possible values:</p> <p>-1 or 0 ...infinite n ...timeout in milliseconds</p>

Table 4: General Communication Parameters

4.2.3 Parameters for TCP/IP, UDP and Bluetooth

Section [Com Settings] / TCP/IP, UDP and Bluetooth Interface

These settings apply only for IOType=5, 6, 7 and 8.

Host=localhost	<p>The host name.</p> <p>Depending to the selected IO Type this parameter has the following meaning:</p> <ul style="list-style-type: none"> ▪ TCP/IP Client (IOType=5): The name (e.g. www.tec-it.com) or the IP address (e.g. 127.0.0.1) of the server you want to connect to. ▪ TCP/IP Server (IOType=6): The parameter is ignored. ▪ Bluetooth Client (IOType=7): The MAC address of the Bluetooth device (e.g. 00:0C:A7:01:CE:DE). ▪ UDP Client (IOType=8): The name (e.g. www.tec-it.com) or the IP address (e.g. 127.0.0.1) of the server you want to connect to.
Service=80	<p>The port number which specifies the service (e.g. 80 for http).</p> <p>Depending to the selected IO Type this parameter has the following meaning:</p> <ul style="list-style-type: none"> ▪ TCP/IP Client (IOType=5): The server port to which you want to connect to. ▪ TCP/IP Server (IOType=6): The port on which you want to accept connections. ▪ Bluetooth Client (IOType=7): Always use the service number 0! ▪ TCP/IP Client (IOType=8): The server port to which you want to connect to.
CheckAliveTimerInterval=0	<p>Timer interval to check for died connections (in milliseconds).</p> <p>If <i>TWedgeCE</i> is configured as TCP/IP Server you can use the Check-Alive Timer to check whether a connected client is still alive or not. The check is done by sending the CheckAliveString to the client periodically. If the write command fails, <i>TWedgeCE</i> assumes the connection is interrupted and restarts the Server in order to avoid blocking the port.</p> <p>Possible values:</p> <p>0 ...no timeout (= disabled) n ...timeout in milliseconds</p>
CheckAliveString=	<p>The string which is sent to the client as Check-Alive message.</p> <p>You can use an arbitrary character or string (e.g. A)</p>

Table 5: TCP/IP Parameters

4.2.4 Parameters for Serial Communication

Section [Com Settings] / Serial Interface

These settings apply only for IOType=3.

Device=COM8:	Device name. Can be either COM1: to COM9: Symbol PPT 8800: Additionally supported is SCN1: (for the built-in hardware scanner of the device). When using Desvice=SCN1: all other serial connection settings will be ignored.
Baud=9600	Baud rate. Common values are: 110, 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 56000, 57600, 115200, 128000, 256000
Data=8	Number of data bits. Possible values are: 7, 8
Stop=1	Number of stop bits. Possible values are: 1, 2
Parity=N	Parity checking. Possible values are: E,O,N
XOnXOff=N	XonXoff handshake on/off. Possible values are: N, Y
RtsCts=N	RtsCts handshake on/off. Possible values are: N, Y
DtrDsr=N	DtrDsr handshake on/off. Possible values are: N, Y

Table 6: Serial Communication Parameters

- ▶ If the device is capable of Bluetooth communication and if it offers a Bluetooth stack, the Bluetooth communication will be mapped to two different serial communication ports. One for receiving data (e.g. COM8:) and one for transmitting data (e.g. COM9:).
- ▶ When using "Device=SCN1:" on the Symbol PPT 8800 all connection settings in this section will be ignored.

4.2.5 Rules for Data Packaging

Section [Com Settings] / DataPackaging

ReadTimeout=100	Read timeout in milliseconds Cut the input stream every ReadTimeout milliseconds. If infinite timeout is set the data packaging will be dependent on NoOfBytes and Delimiter only. Possible values: -1 or 0 ...infinite timeout n ...timeout in milliseconds
NoOfBytes=100	Maximum number of bytes per package Cut the input stream with every NoOfBytes bytes.
Delimiter=	Delimiter-string Cut the input stream with every occurrence of this string (UseDelimiter must be enabled!).
UseDelimiter=N	If enabled, the input stream is cut with every occurrence of the delimiter-string. Possible values are: N, Y
IncludeDelimiter=N	Include the delimiter-string in output stream? Possible values are: N, Y
Prefix=	Prefix (may include control commands) This is a constant string which will be sent before each data package.
Postfix=	Postfix (may include control commands) This is a constant string which will be sent after each data package.
JitterTimeout=0	Jitter timeout (in milliseconds). When receiving multiple packets containing the same data within the given number of milliseconds only the first package will actually be simulated as keystroke sequence. All subsequent packages are suppressed.

	<p>Please consider to use a reasonable value with respect to the ReadTimeout parameter! It is recommended to use this option in conjunction with a delimiter.</p> <p>Possible values:</p> <p>-1 ...infinite timeout</p> <p>0 ...no timeout (= disabled)</p> <p>n ...timeout in milliseconds</p>
--	---

Table 7: Data packaging parameters

► **IMPORTANT!**

Identifying correct data packages is essential for every kind of application! Basically the data received in TWedgeCE is available as continuous input stream. In order to get the data packages which were sent by the connected device back out of this stream (and not to get them merged or split or whatsoever) you have to specify the correct packaging rules! Each of the identified data packages will then be simulated as keystrokes.

For partitioning the input stream into data packages the following rules apply in the given order:

1. **ReadTimeout**
All data which is received within ReadTimeout milliseconds is treated as one data package. This data package may then be split into smaller packages according to the following parameters:
2. **NoOfBytes and Delimiter**
With every occurrence of the specified delimiter string, but at least with every NoOfBytes bytes, the package built in 1. is subdivided.
3. **Optionally**, each of the resulting packages can be modified using the Pre-Evaluation expression (see section 4.2.8), the String Replacement Table (see section 4.2.6) and/or Control Sequences (see section 4.2.7).
4. All packages are simulated as keystrokes.

- The Prefix and the Postfix are sent before and after each data package.
- When using the JitterTimeout all packages with the same content are skipped for the specified time period. Only the first package in a series is actually simulated as keystrokes.

Example

```
NoOfBytes=10
ReadTimeout=100
Delimiter=$
UseDelimiter=Y
IncludeDelimiter=N
```

The scanner reads the following 7 characters: "1234\$56".

Since the Delimiter is set to "\$" and IncludeDelimiter=N the string "1234" will be passed on to the application. The leftover-string "56" is less than 10 bytes long and does not include the delimiter "\$". Thus it is not sent by now.

If some more data is received *within the next ReadTimeout milliseconds* – e.g. the scanner sends "789012345678\$" – the new data will be appended to the leftover "56" from the last scan. Thus the application receives "5678901234" (NoOfBytes=10!). "5678" is stored for another ReadTimeout milliseconds.

Finally, if no more data is received *within the next ReadTimeout*, "5678" will be passed to the application.

Escape Sequences

You can use escape sequences in the Prefix, in the Postfix, in the TranslationTable (left-hand value and right-hand value) and you can use escape sequences in the *Delimiter*-string. The following table shows a list of standard escape sequences:

\\	Backslash
\a	Alert
\b	Backspace
\f	Formfeed
\n	Newline
\r	Carriage return
\t	Horizontal tab

Table 8: Standard escape sequences

Furthermore the following escape sequences are recognized:

\x..	ASCII character as hex code
\0..	ASCII character as octal code
\v \V	Virtual key code (NOT IMPLEMENTED)

Table 9: Extended escape sequences

The hex code identifier “\x” must be followed by two digits (or characters ‘a’ to ‘f’ and ‘A’ to ‘F’ respectively). The octal code identifier “\0” (NOTE: backslash + zero!) must be followed by three octal digits (values 0..7).

E.g. you may specify the space character (decimal value 32) by “\x20” or “\0040”.

4.2.6 String Replacement Table

Section [TranslationTable]

You can add entries to the string replacement table using the syntax

```
searchstring=replacement
```

Thus every occurrence of “searchstring” in the input-stream will be replaced with “replacement” in the output-stream. E.g. you may specify the following table:

\$1={CTRL-ENTER}	Replace “\$1” with ctrl+enter
\$2={TAB}	Replace “\$2” with tabulator key
\$3=Joe	Replace “\$3” with “Joe”

Table 10: String replacement samples

- ▶ Please make sure you have EnableControlSequ=Y for translating control sequences like “{ENTER}” into the corresponding keystrokes and EnableControlSequ=N for pure text replacement.

4.2.7 Control Sequence Settings

Section [ControlSequences]

EnableControlSequ=N	Use control sequences for the prefix, for the postfix and for the replacement table (right-hand value).
ActivationDelay=100	NOT IMPLEMENTED

Table 11: Control sequence parameters

TWedgeCE recognizes a certain amount of control commands. They can be used to send key-strokes (e.g. CTRL+C for copy, CTRL+V for paste, TAB for setting the focus to the next input area etc.), to activate other program windows or to pop up message boxes. Control sequences can be used in the prefix, in the postfix or in the right hand values of the string replacement table.

Control commands are enclosed in braces. E.g. you can send the key press sequence <Enter>, <Cursor Right> with the following control sequence:

```
{ENTER}{RIGHT}
```

- ▶ If EnableControlSequ=N the text "{ENTER}{RIGHT}" will be posted to your application and not the key press sequence.

For a list of valid control commands please refer to Appendix A.

4.2.8 Pre-Evaluation Expression

Section [PreEvaluation]

Expression=	Pre-evaluation expression (used to modify the acquired data).
Enable=N	Use pre-evaluation expression (y/n)?

Table 12: Pre-Evaluation parameters

The pre-evaluation expression is an expression, that can be used to modify the acquired data, before it is processed any further. The pre-evaluation is executed before the replacement of strings (see section 4.2.6) and before the execution of control commands (see section 4.2.7). The prefix and the postfix are added to the modified input data afterwards.

Within the expression use the system variable "DATA" to reference to the acquired data string.

For a list of available functions, please refer to Appendix B.

Examples

Use the first five characters of the input only:

```
Expression=Left (DATA, 5)
```

Cut the first two characters of the input string:

```
Expression=Mid (DATA, 2, Len (DATA) - 2)
```

Prepend the prefix "pfx" and append the suffix "sfx" to the input data:

```
Expression="pfx" + DATA + "sfx"
```

4.2.9 Settings for the Tray Menu

Section [TrayMenu]


Enable=Y	Show the small program icon inside the notification area (y/n)? (see Figure 1, )
----------	---

Table 13: Tray menu parameters

4.2.10 Operation Mode

Section [Operation Mode] in the ini-file

KeyboardEmulation=Y	Emulate keystrokes so that other applications will receive data from TWedgeCE.
EmulationMode=0	Sets the emulation mode:

	<p>0... Default mode. 1... Alternative mode.</p> <p>If the default mode does not work for your application (e.g. for terminal server) you may try this alternative keystroke simulation mode.</p> <p>IMPORTANT NOTE: In mode 1 the following characters are not (yet) supported: # < > ä ö ü ß. Besides that you might eventually experience problems with different language specific keyboard layouts.</p>
--	---

Table 14: Keyboard emulation parameters

► If “KeyboardEmulation” is disabled *TWedgeCE* will NOT send ANY data!

4.3 Example Configurations

4.3.1 No Newline, No Linefeed

Some barcode scanners append a “newline” character and a “linefeed” character at the end of the scanned text (which is equivalent to the escape sequence “\r\n”). If you do not want these characters to be sent to your application you can define the sequence “\r\n” as input stream delimiter. Thus every data package will be cut with the occurrence of “\r\n”. If IncludeDelimiter is set to N the delimiter “\r\n” will not be included in the output.

```
Delimiter=\r\n
UseDelimiter=Y
IncludeDelimiter=N
```

4.3.2 MS Excel

To change to the next data cell in MS Pocket Excel after receiving data you can use the following line in the settings file:

```
Postfix={CTRL-ENTER}{DOWN}
```

The keystroke {CTRL-ENTER} fills the current cell with the received data. The keystroke {DOWN} selects the cell below for the next input.

4.3.3 Replacement Table

If you want to replace every occurrence of “\$1” in the input stream with the content of your clipboard you can do this by the following statement. It sends the paste command Ctrl+V every time “\$1” occurs in the input string (EnableControlSequ=Y must be set!).

```
[TranslationTable]
$1={CTRL-V}
```

4.3.4 Extract Sub-String from the acquired Data

You can configure *TWedgeCE* to use a sub-string from the acquired input data only. To do this use the pre-evaluation expression. The following configuration will remove the first character and use the following 8 characters from the acquired data packet only:

```
[PreEvaluation]
Expression=MID(DATA,1,8)
Enable=Y
```

5 Examples of Use

5.1 Example 1

The input device provides the following data. Note: “\x0a\x0d” is the hex code for the newline linefeed combination (which could also be written as “\r\n”).

```
BC280229C00000000=00001T9\x0a\x0d
```

Task

Remove the first character "B", transmit the next 16 characters "C280229C00000000" as keystrokes, remove the end of the string "=00001T9\x0a\x0d". And finally press "enter".

Therefore you could use the following configuration:

```
[ComSettings]
;DataPackaging
Delimiter=\x0a\x0d
UseDelimiter=Y
IncludeDelimiter=N
Postfix={ENTER}

[PreEvaluation]
Expression=MID(DATA,1,16)
Enable=Y
```

The delimiter is used to identify the end of the input data and to crop the newline linefeed. The pre-evaluation extracts the relevant data from the input string. Finally, the postfix is used to send an “enter”.

6 Available Licenses

6.1 Retail License

Retail Licenses of *TWedgeCE* are available for

- **Single User**
Installation on exactly one device
- **Workgroup**
1-10 installations within one company on exactly one site (or subsidiary).
- **Office**
1-100 installations within one company on exactly one site (or subsidiary).
- **Site**
1-250 installations within one company on exactly one site (or subsidiary).
- **Enterprise**
Unlimited number of installations worldwide (within one enterprise).

6.2 Demo License

A demo installation (for evaluation purposes) is available on <http://www.tec-it.com>.

6.3 Pricing

Please contact us for actual prices.

7 Contact and Support Information

TEC-IT Datenverarbeitung GmbH

Address: Wagnerstr. 6
AT-4400 Steyr
Austria/Europe

Phone: +43 / (0)7252 / 72 72 0
Fax: +43 / (0)7252 / 72 72 0 – 77

Email: <mailto:support@tec-it.com>

Web: <http://www.tec-it.com>

AIX is a registered trademark of IBM Corporation.

HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C, World Wide Web Consortium, Laboratory for Computer Science NE43-358, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139.

JAVA® is a registered trademark of Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA.

JAVASCRIPT® is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

Microsoft®, Windows®, Microsoft Word®, Microsoft Excel® are registered trademarks of Microsoft Corporation.

Navision is a registered trademark of Microsoft Business Solutions ApS in the United States and/or other countries.

Oracle® is a registered trademark of Oracle Corporation.

PCL® is a registered trademark of the Hewlett-Packard Company.

PostScript is a registered trademark of Adobe Systems Inc.

SAP, SAP Logo, R/2, R/3, ABAP, SAPscript are trademarks or registered trademarks of SAP AG in Germany (and in several other countries).

All other products mentioned are trademarks or registered trademarks of their respective companies. If any trademark on our web site or in this document is not marked as trademark (or registered trademark), we ask you to send us a short message (<mailto:office@tec-it.com>)

Appendix A: Control Commands

The following control commands are recognized by *TWedgeCE*. Please note that you have to put control commands in braces (e.g. {RETURN} for the return key)!

CANCEL	
BACKSPACE	Backspace key
TAB	Tabulator key
CLEAR	
ENTER	Enter key (keypad)
RETURN	Return key
SHIFT	Shift key
ESC	Escape key
SPACE	Space key
PAGE-UP	Page up key
PAGE-DOWN	Page down key
END	End key
HOME	Home key
LEFT	Cursor left key
UP	Cursor up key
RIGHT	Cursor right key
DOWN	Cursor down key
SELECT	
PRINT	Print key
EXECUTE	
SNAPSHOT	Print screen key
INS	Insert key
DEL	Delete key
HELP	
MUL	Multiply key (keypad)
ADD	Add key (keypad)
SEP	Separator key (keypad)
SUB	Subtract key (keypad)
DEC	Decimal key (keypad)
DIV	Divide key (keypad)
F1	F1 key
F2	F2 key
F3	F3 key
F4	F4 key
F5	F5 key
F6	F6 key
F7	F7 key
F8	F8 key
F9	F9 key

F10	F10 key
F11	F11 key
F12	F12 key
LMENU	Left alt key
RMENU	Right alt key (Alt Gr)
LWIN	Left windows key
RWIN	Right windows key
APPS	Application key (Microsoft Natural Keyboard)
ACT:window	Activate window named "window" and bring it to front
MSG:text	Displays "text" as message box

Table 15: TWedgeCE control commands

Toggle Keys

If you want to emulate the keystroke SHFIT+CTRL+ALT+S you have to use the control string {SHIFT-CTRL-ALT-S}

- ▶ **NOTE: ALWAYS USE THIS ORDER FOR TOGGLE KEYS!**
Otherwise the control sequence will not be recognized correctly!
- ▶ 1. SHIFT
- ▶ 2. CTRL
- ▶ 3. ALT

Appendix B: Function Reference

B.1 Functions

Return	Function	Description
long	Abs («Number»)	Returns the absolute value of a number.
long	Asc («Text»)	Returns the ASCII value of a given character or of the first character of "text".
char	CDate («Text»)	Converts the string "text" to a date. Provides the current date (Now ()) if no conversion is possible.
double	CDbl («Expr»)	Converts any value to a double value (floating-point notation). The result is 0.00 when a conversion is not possible.
char	Chr («Number»)	Returns the corresponding character for the specified ASCII value "Number".
long	CLng («Expr»)	Converts any value into a whole number. If a conversion is not possible, the result is 0.
string	CStr («Expr»)	Converts a value into a text.
long	Day («Date»)	Determines the day of the month [1..31].
long	DayOfWeek («Date»)	Returns the day of the week of a specified date [1..7]. 1=Sunday, 2=Monday, ...
double	DayOfYear («Date»)	Returns the day of the year of a specified date [1..366].
double	Exp («Number»)	Returns the value e^{Number} , where e is the base of the natural logarithms.
double	Exp10 («Number»)	Returns the 10^{Number} .
long	Find («Text», «SearchText», «nStart»)	Searches the string "Text" for "SearchText" starting from Position "nStart". Returns the position of the string or -1. The first character of a string is located at position 0.
long	FindReverse («Text», «SearchText», «nExclude»)	Searches the string "Text" for "SearchText" in reverse order excluding "nExclude" characters at the end. Returns the position of the string or -1. The first character of a string is found at position 0.
string	Format («Number», «Pattern»)	Formats "Number" according to the specified pattern string "Pattern". Format placeholders: # digit or no value, 0 '0' or digit . decimal point , comma + - sign
double	Fract («Number»)	Returns the fractional unit
long	Hour («Date»)	The hour of a specified date [00..23].
string	IIf («Condition», «TrueExpr», «FalseExpr»)	Returns the value of «TrueExpr» if «Condition» is evaluated as (TRUE or not equal to 0). Returns the value of «FalseExpr» if «Condition» is evaluated as (FALSE or equals 0).
long	IsEmpty («Text»)	Test whether the string "Text" is empty or not.
long	IsEven («Number»)	Returns TRUE if "Number" is even.
long	IsLastPage ()	Returns TRUE if the page being printed is the last page of the document.
bool	IsLeapYear («Date»)	Returns TRUE if the specified date occurs within a leap year.
long	IsOdd («Number»)	Returns TRUE if "Number" is uneven.
string	LastValue («DatafieldName»)	Returns the last value of a given data-field (the value of the data-field from the previous record).
string	Left («Text», «nLength»)	Returns the first characters "nLength" of a string.
long	Len («Text»)	Returns the length of the given string.
double	Log («Number»)	Returns the natural logarithm of "Number".
double	Log10 («Number»)	Returns the logarithm of "Number".
string	Mid («Text»,	Returns the substring of string "Text" starting at position "nStart" with length

	«nStart», «nLenght»)	"nLength". The first character of a string is located at position 0.
long	Minute («Date»)	The minutes of a specified date/time [00..59].
long	«a» % «b»	Modulo operator: Remainder of the integer division a / b
long	Month («Date»)	The month of the specified date [1..12].
date	Now ()	Actual date and time.
double	Pow («Number», «Power»)	Returns the result of "Number" raised to the power of "Power".
string	Replace («Text», «SearchText», «ReplaceText»)	Replaces each occurrence of "SearchText" in string "Text" with "ReplaceText".
string	Right («Text», «nLength»)	Returns the last characters "nLength" of a string.
double	Round («Number», «Precision»)	Returns "Number" rounded using precision digits. If 0 is "Precision" the result will be rounded to a whole number.
long	Second («Date»)	The second of a specified date/time [00..59].
double	Sqrt («Number»)	Returns the square root of "Number".
long	SumOfDigits («Number»)	The sum of all digits of "Number".
long	SumOfDigits1 («Number»)	Returns the one digit sum of all digits of "Number".
string	ToLower («Text»)	Converts all character in the string "Text" to lower case.
string	ToUpper («Text»)	Converts all character in the string "Text" to upper case.
string	Trim («Text»)	Removes leading and trailing spaces.
string	TrimLeft («Text»)	Removes leading spaces.
string	TrimRight («Text»)	Removes trailing spaces.
double	Value («Text»)	Converts "Text" to a double value.
long	WeekOfYear («Date»)	The calendar week of a specified date/time [1..52].
long	Year («Date»)	The year of the specified date/time.

Table 16: Functions

B.2 Constants

Function	Description
False	Logical value FALSE. This value is usually the result of a condition. If the condition is not fulfilled the resulting value is FALSE.
True	Logical value TRUE. This value is usually the result of a condition. If the condition is fulfilled the resulting value is TRUE.
"\n"	Linefeed.

Table 17: Constants