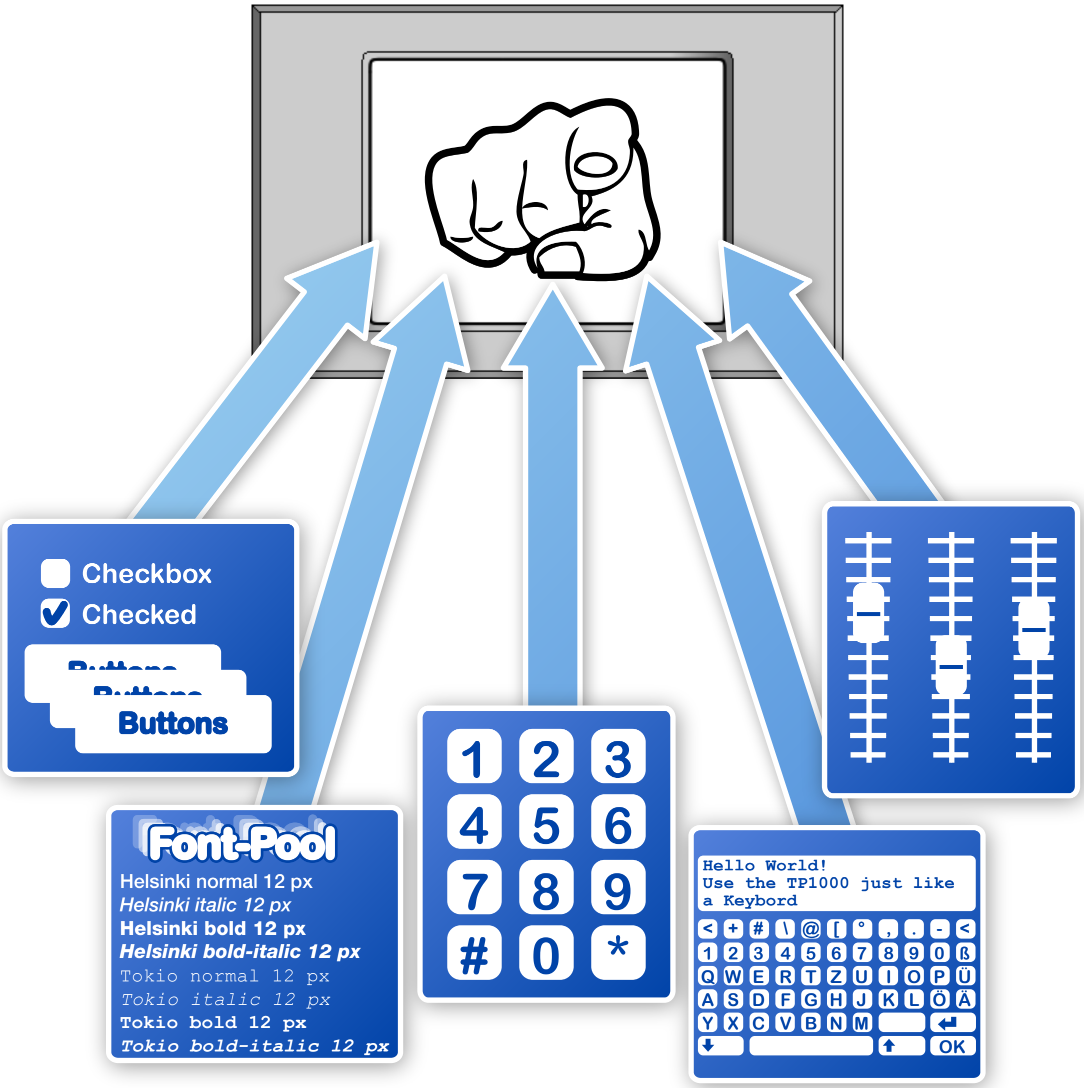
Tiger Graphic Library



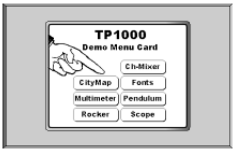
Wilke Techno Logy

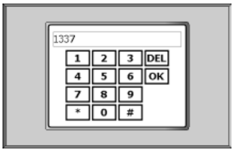
Version of document: v001

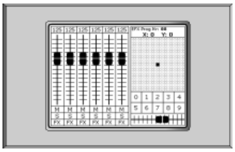
Your 7 steps for a quick start

Welcome!

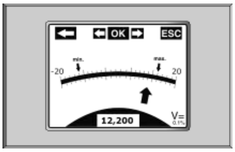
More and more new applications/devices are featuring graphical displays and touch panels as user interface. Applications / devices based on Tiger multitasking computers make no exception.

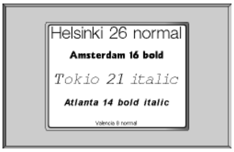




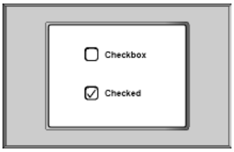


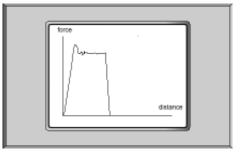
The Tiger Graphic Library simplifies programming the touch panel. The controlling of the devices is already easy to program with the comfortable use of the device drivers for each device with many features. With the Tiger Graphic Library there is a tool to realize a graphical user interface by calling just e few subroutines. In earlier times the part of programming the GUI mostly would take the longest time of the software development.





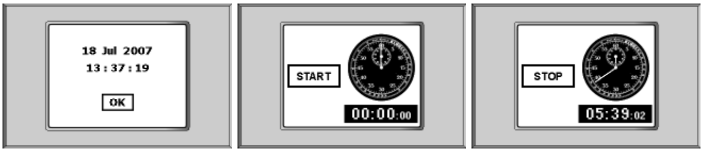
Now with the Tiger Graphic Library you are able to realize your project in a minimum of time and a maximum of user friendliness. Its great features will make it easy for you to give your program an individual nice look without needs for a longer development time. You will find lots of example programs for the use of its subroutines and many applications which can be used as templates for your own programs.





The Tiger Graphic Library provides subroutines for:

- creating buttons, switches and sliders for the touch panel
- graphic fonts with many special chars
- administrating whole pages containing touch panel functionalities and graphical elements for the LCD
- touch panel keyboards in different styles
- setting and displaying the real time clock in different styles
- displaying current measurands
- creating graphs
- showing dynamically created graphics of your own



Additionally the Tiger Graphic Library

- provides many templates for own projects
- will be updated continuously

This short introduction gives you a quick start in working with the Tiger Graphic Library by some simple examples.

Contact

It is our goal that you are all around satiesfied with our products.

Should you have any questions or suggestions about hard- or software, or about the documentation, please let us know. Our technical support will be happy to assist you.

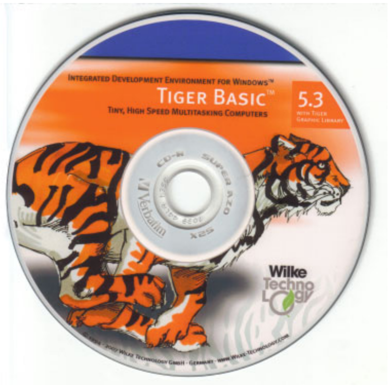
You can contact the Tiger Support Team

by phone:	+49 (241) 918 9032
by fax:	+49 (241) 918 9044
by email:	support@wilke.de
by callback:	http://www.wilke.de/callback.php

Requirements

For using the Tiger Graphic Library two things are required:

- „Tiger BASIC 5.3“ with serial number. This development system contains the Tiger Graphic Library.



Integrated Development Environment for Windows™
Tiger Basic™
5.3
with Tiger Graphic Library
Tiny, High Speed Multitasking Computers
Wilke Techno Logy

- A compiler with following minimum configuration:
  - Pentium Processor 500 MHz or faster
  - Harddisk with minimum 250 MByte free disk space
  - SVGA graphics (800 x 600) or higher resolution
  - Mouse
  - Windows 2000 / Me or newer version
  - 1 free COM port (serial interface)
  - CD-ROM or DVD drive

Preparing Soft- and Hardware

Installation of the Tiger-BASIC IDE

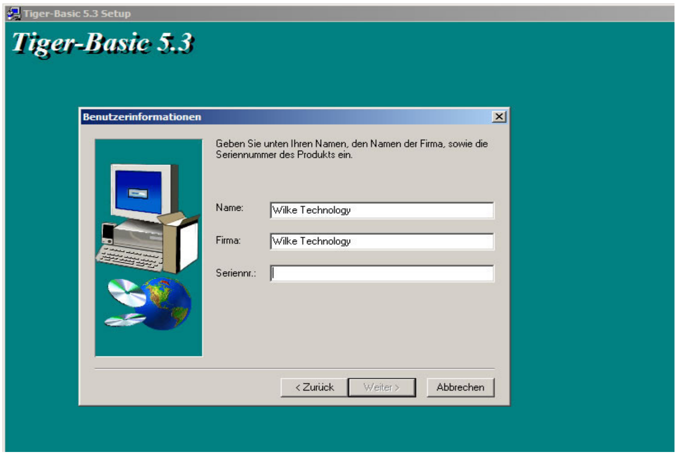
The Tiger Graphic Library is included in version 5.3 of the Tiger-BASIC® development environment.

So this version has to be installed on your computer to use the Tiger Graphic Library and all of its functions. You should install this version even when you have already an older version Tiger-BASIC® installed.

The installation is by default done into an own directory, so that this version can work parallel to your older version.

Start with the installation of the development environment to your computer:

- Close all currently running Windows applications.
- Place the CD „Tiger BASIC 5.3 with Tiger 2-support“ into the CD-ROM / DVD drive.
- Normally the setup program is automatically started. If this is not the case, please run the program „SETUP.EXE“ from the root directory of the CD.
- The setup program first shows a window with a welcome message. The an input box for entering the registration information is displayed:



Geben Sie unten Ihren Namen, den Namen der Firma, sowie die Seriennummer des Produkts ein.

Name: Wilke Technology

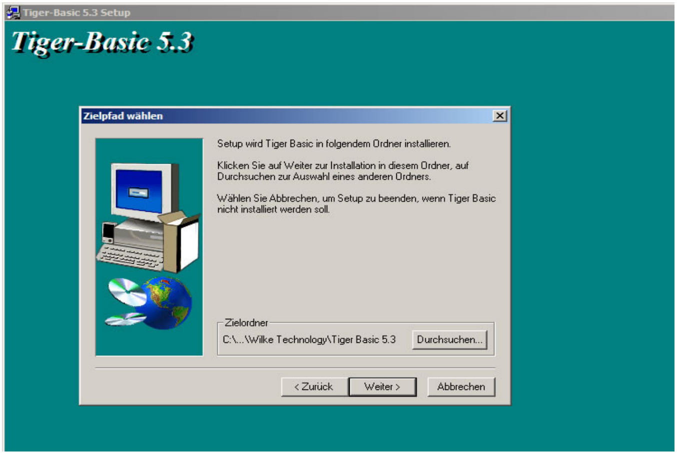
Firma: Wilke Technology

Seriennr.:

Please enter the serial number exactly as it is printed on the sticker on the CD cover, so including the hyphens.

If you don’t have purchased a license yet, you can enter a „0“ at this point. The software will then run in a „Lite“ mode with full functionality, but restricted to compiling up to 2,000 codelines. This is not sufficient to work with the Tiger Graphic Library, so a license is needed.

- Next you choose the directory into which the development environment should be installed. Normally the default setting can adopted as it stands:



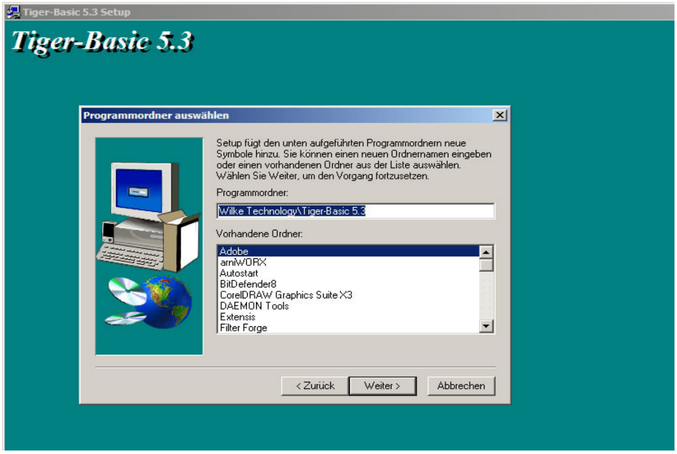
Setup wird Tiger Basic in folgendem Ordner installieren.

Klicken Sie auf Weiter zur Installation in diesem Ordner, auf Durchsuchen zur Auswahl eines anderen Ordners.

Wählen Sie Abbrechen, um Setup zu beenden, wenn Tiger Basic nicht installiert werden soll.

Zielordner
C:\...\Wilke Technology\Tiger Basic 5.3

- Now you determine the program group for the Windows start menu. Here again the default setting should be retained:

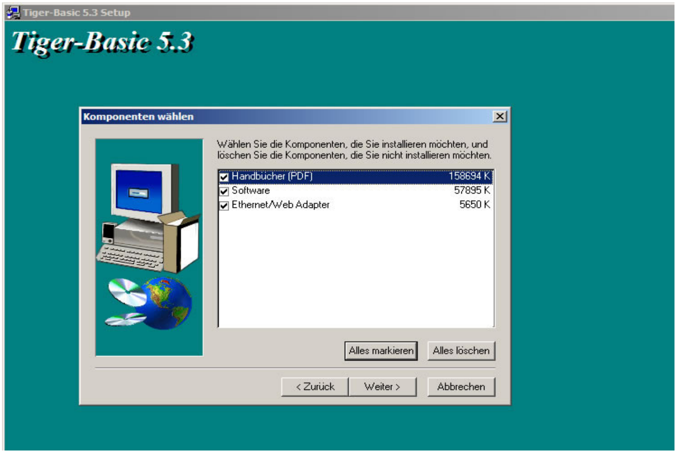


Setup fügt den unten aufgeführten Programmordnern neue Symbole hinzu. Sie können einen neuen Ordnernamen eingeben oder einen vorhandenen Ordner aus der Liste auswählen. Wählen Sie Weiter, um den Vorgang fortzusetzen.

Programmordner:
Wilke Technology\Tiger-Basic 5.3

Vorhandene Ordner:
Adobe
amiWORX
Autostart
BitDefender8
CorelDRAW Graphics Suite X3
DAEMON Tools
Extensis
Filter Forge

- Finally you decide what components should be installed. All components marked with a checkmark will be installed.

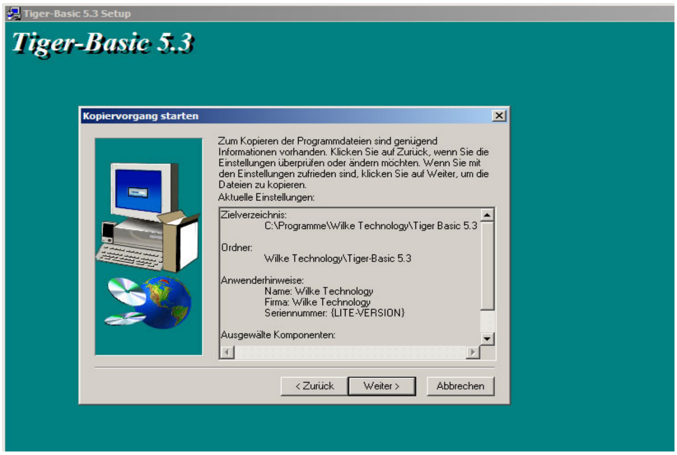


Wählen Sie die Komponenten, die Sie installieren möchten, und löschen Sie die Komponenten, die Sie nicht installieren möchten.

✓	Handbücher (PDF)	158694 K
✓	Software	57895 K
✓	Ethernet/Web Adapter	5650 K

So you can e.g. decide to no install the PDF manuals if you disk space is limited.

- If you have made all settings, they are once again shown as summary in one window:

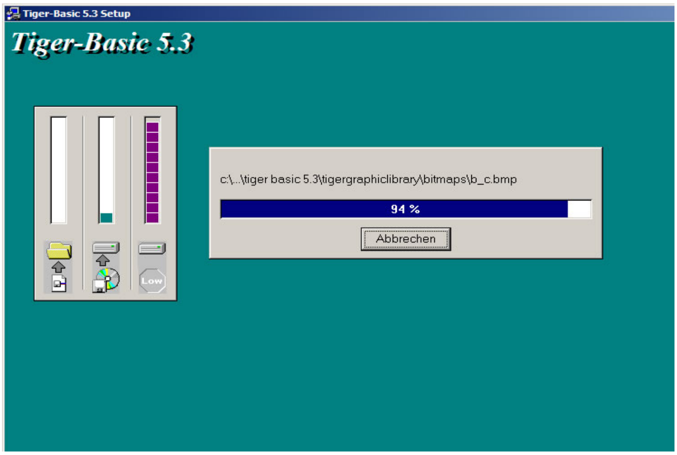


Zum Kopieren der Programmdateien sind genügend Informationen vorhanden. Klicken Sie auf Zurück, wenn Sie die Einstellungen überprüfen oder ändern möchten. Wenn Sie mit den Einstellungen zufrieden sind, klicken Sie auf Weiter, um die Dateien zu kopieren.
Aktuelle Einstellungen:

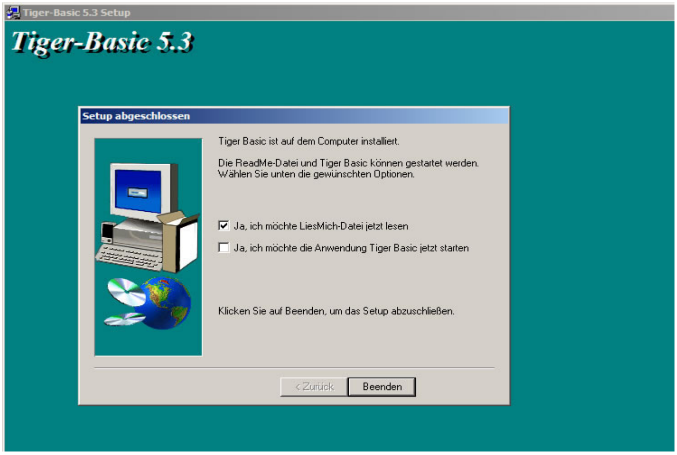
```
Zielverzeichnis:
          C:\Programme\Wilke Technology\Tiger Basic 5.3

Ordner:
          Wilke Technology\Tiger-Basic 5.3

Anwenderhinweise:
          Name: Wilke Technology
          Firma: Wilke Technology
          Seriennummer: (LITE-VERSION)

Ausgewälte Komponenten:
```

Check again if all settings are correct. If this is the case, please start the installation process by clicking on „Next >“.

- The copying process of all files required for the installation is now started. The current progress of the installation is constantly shown:



- When the installation process is completed please click on „Finish“ to quit the installation program:



Tiger Basic ist auf dem Computer installiert.

Die ReadMe-Datei und Tiger Basic können gestartet werden. Wählen Sie unten die gewünschten Optionen.

☑ Ja, ich möchte LiesMich-Datei jetzt lesen

☐ Ja, ich möchte die Anwendung Tiger Basic jetzt starten

Klicken Sie auf Beenden, um das Setup abzuschließen.

With that Tiger-BASIC® 5.3 is completely installed on your computer and with it everything that you need to start programming your device using the Tiger Graphic Library.

Connecting the TP1000 for programming

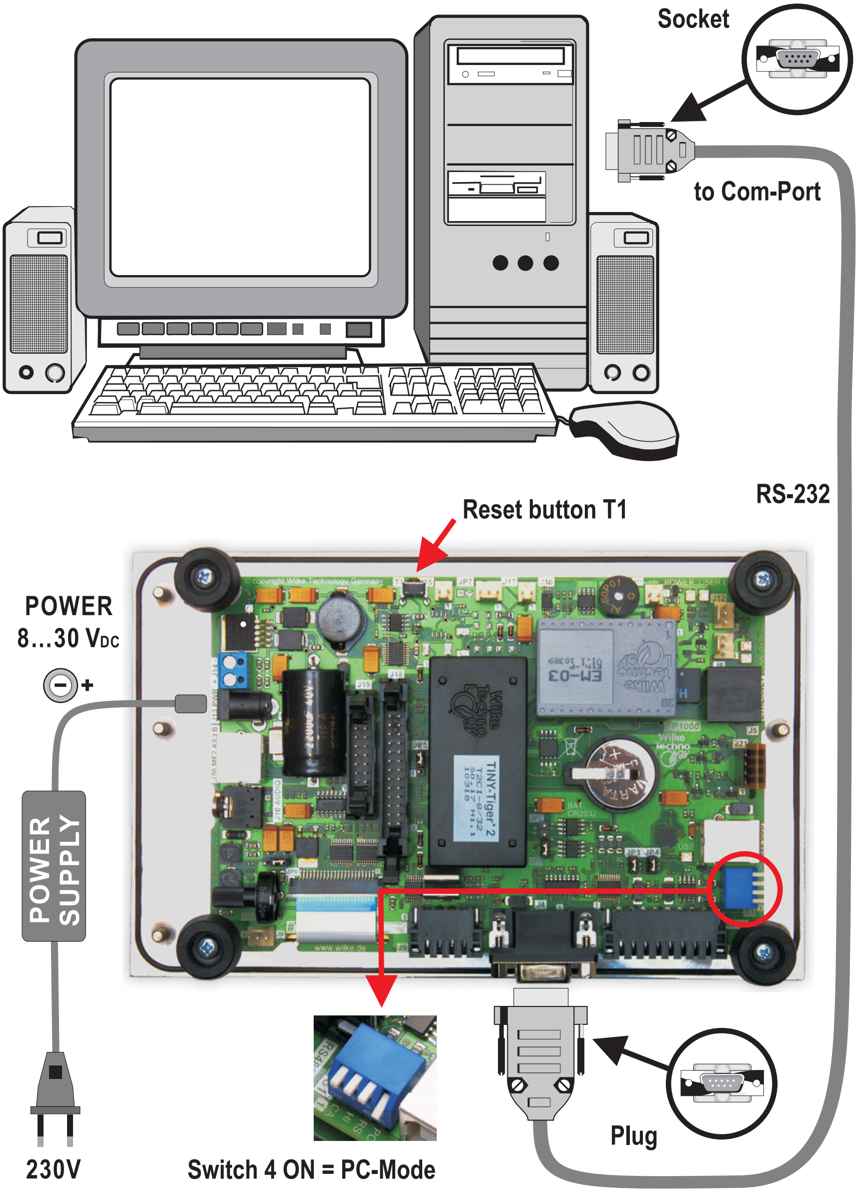
The sample programs of the Tiger Graphic Library are developped and tested on a TP1000 touchpanel computer, so you should have one connected to follow the sample programs in the next chapter.

The TP1000 gets ready for programming quite easily:

- Connect the TP1000 with a serial port of your computer with a serial cable (1:1, included in TP1000 startup pack).
- Connect the TP1000 with the power supply using the power supply unit delivered with the TP1000 startup pack. The Power-LED (PWR) should light up now. **IMPORTANT: take care of the correct polarity!**
- Starten the Tiger-BASIC® development environment on your computer.
- Select from the menu **Optionen [Options]** the command **Übertragung [Communication]** and select in the dialog box the COM port, to which the TP1000 is connected to. For the models with TINY-Tiger® 2 (...-T2B) the baud rate is 115,200 Bd, parity is „None“. For the models with TINY-Tiger® 1 (...-T1) the baud rate is 38,400 Bd, the parity is „Even“.
- At the switch S1 of the TP1000 select the operation mode „PC-Mode“ (switch 4 to ON) and press the RESET button T1.
- Select from the menu **Anzeigen [View]** the command **Tiger-Status [Tiger status]** and you receive a status message an the screen about the type and state of the connected Tiger computer.

Now there is a working connection between the units.

The picture below shows the connection of the TP1000 to the computer and the power supply:



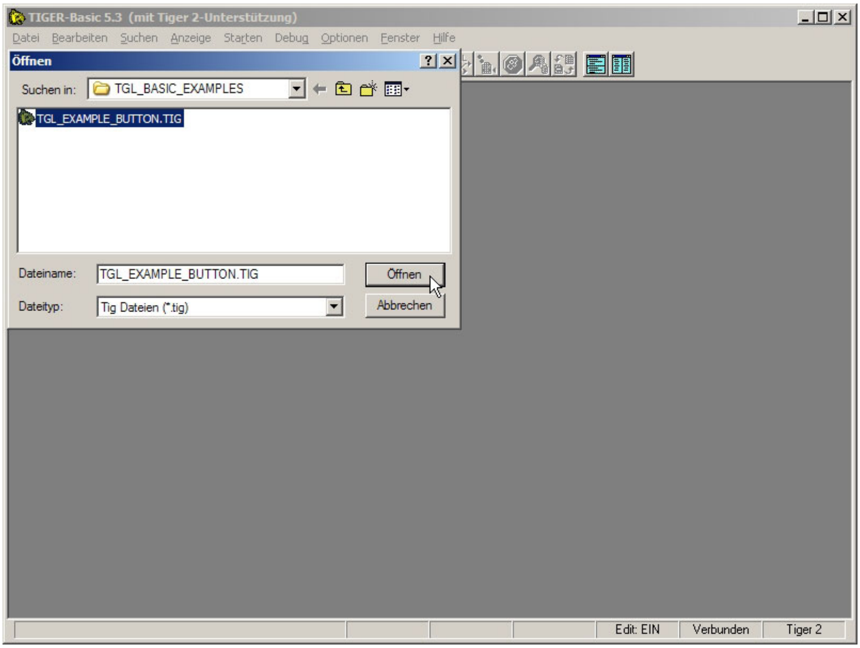
Connection of TP1000 for programming

Example for buttons

Directly after installation of hardware und software you can start checking out the applications and sample programs of the Tiger Graphics Library.

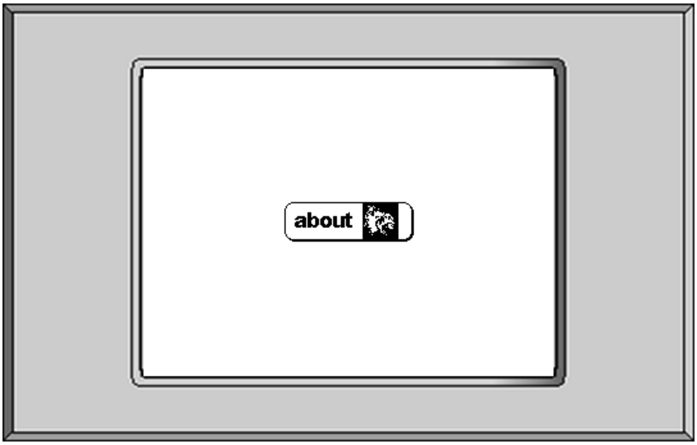
You can find the sample programs for the Tiger Graphic Library in the subdirectory „..\Examples\TGL_Examples“ of your Tiger-BASIC® 5.3 installation and its further subdirectories.

We start with a simple example of creating a button on the LCD that is also forming a sensitive area of the touchpanel. Start the Tiger-BASIC® development environment and load the source code of this first example into the editor with the command **Öffnen [Open]** from the menu **Datei [File]**. The name of the program is **TGL_EXAMPLE_BUTTON.TIG** and it is located in the subdirectory **..\Examples\TGL_Examples\TGL_BASIC_EXAMPLES**.


TIGER-Basic 5.3 (mit Tiger 2-Unterstützung)
Datei Bearbeiten Suchen Anzeige Starten Debug Optionen Fenster Hilfe
Öffnen
Suchen in: TGL_BASIC_EXAMPLES
TGL_EXAMPLE_BUTTON.TIG
Dateiname: TGL_EXAMPLE_BUTTON.TIG
Dateityp: Tig Dateien (*.tig)
Öffnen
Abbrechen
Edit: EIN
Verbunden
Tiger 2

After loading the program code you can immediately transfer the program to the TP1000 and execute it there. Use the command **Ausführen [Run]** from the menu **Starten [Start]** or press the key **F5**. The program is first compiled and then transferred to the TP1000, where it is automatically started.

On the TP1000 now back display backlight should be switched on and a button is displayed in the centre of the screen:



When you press the button, you will hear a beep as acknoledge. Pressing the touchpanel outside the button has no function and gives no feedback.

So what happens in the program? First some settings are made and the Tiger Graphic Library is included. In the section „IDENTIFIERS“ one element and one window are defined - we don't need more at this point. We define some required variables and include the file for device driver installation. Now in the section „TGL ELEMENTS AND WINDOWS“ it gets interesting:
With the library function *bTglCreateButton* we create a button containing a bitmap graphic. With the function *bTglPlaceButtonInWindow* we place this element (button) into a window. The parameters of the functions determine look, position and attributes of the button. Finally the window containing the button is shown on the LCD with the function *bTglShowWindow*.

Now we hear that a button is pressed, but we don't actually see it. So we now modify the program by including a second graphic that is displayed when the button is pressed. We need to make the following changes to the source code:

In the section „IDENTIFIERS“ we define a second element ID for the „pressed“ button:

```
' elements
#define BUTTON_ID                0
#define GRAPHIC_ID               1
```

For the second bitmap we need to declare asecond datalabel:

```
datalabel BUTTON, GRAPHIC
```

In the section „TGL ELEMENTS AND WINDOWS“, we have to create the graphic element for the pressed button and link it to the button element by inserting the following lines at the end of the section:

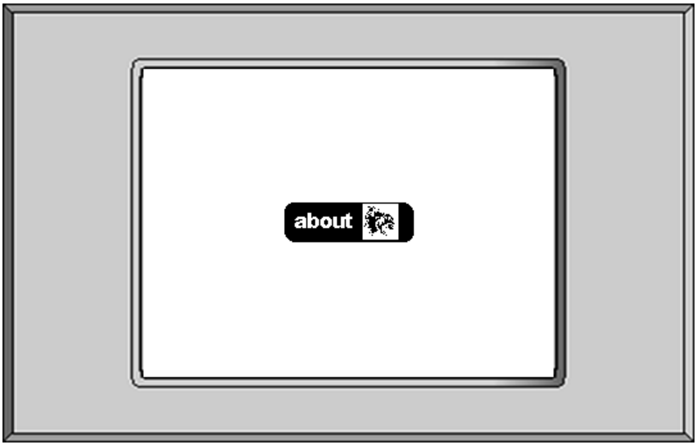
```
call bTglCreateGraphic( &
101, 31, &                  ' width, height of element
GRAPHIC, 104, &             ' address, format width of bitmap
GRAPHIC_ID, &               ' identifier of element
blReturn )                  ' return code (0: OK; >0: error)

call bTglLink( &
BUTTON_ID, GRAPHIC_ID,&     ' ID of placed element, background
blReturn )                  ' return code (0: OK; >0: error)
```

4

Finally, in the section „FLASH“, we have to include the bitmap to be used for the pressed button graphic:

```
BUTTON::
data filter "btn_abouttiger.bmp", "GRAPHFLT", 0
GRAPHIC::
data filter "btn_abouttiger_invert.bmp", "GRAPHFLT", 0
```

Now recompile and start the program again with the command **Ausführen [Run]** from the menu **Starten [Start]** or press the key **F5**. After the new version is started, pressing the button will - besides the acoustic signal - change its look like this:



When you use a different bitmap for the pressed button, you will get a 3D effect. Try changing the line for including the bitmap to:

```
GRAPHIC::
data filter "btn_abouttiger_active.bmp", "GRAPHFLT", 0
```

Run this version, and whenever you press the button it looks like the button is „sinking in“ a little bit.

What you can also easily change are the attributes of the button. If you don’t want the TP1000 to beep when the button is pressed, you just have to modify one parameter of the bTglCreateButton function:

```
call bTglCreateButton( &
101, 31, &                   ' width, height of element
BUTTON, 104, &               ' address, format width of bitmap
TGL_KEY_ATTR_BEEP_OFF, &     ' key attr. auto repeat, beep, type
BUTTON_ID, &                 ' identifier of element
blReturn )                   ' return code (0: OK; >0: error)
```

Recompile this and you won’t hear anything when touching the button. You can also make the button a „switch button“, meaning with every touch of the button you toggle between normal and pressed graphic:

```
call bTglCreateButton( &
101, 31, &                      ' width, height of element
BUTTON, 104, &                  ' address, format width of bitmap
TGL_KEY_ATTR_SWITCH_BUTTON, &   ' key attr. autorepeat, beep, type
BUTTON_ID, &                    ' identifier of element
blReturn )                      ' return code (0: OK; >0: error)
```

Download this version and on the first touch of the button the alternative graphic for the button gets shown. Only a second touch of the button switches back to the standard button graphic.

There are numerous possibilities for the functionality and look of graphic buttons. In the subdirectory **..\Examples\TGL_Examples** of your Tiger-BASIC installation there are several examples programs with a name starting with **TGL_BUTTON_**, showing different possibilities for creating graphical buttons.

Instead of bitmap graphics, you can also create text buttons. The advantage of text buttons is that the text can be altered while the program is running, e.g. for buttons in different languages.

We have to make some adjustments to our program to switch from graphic to text button.

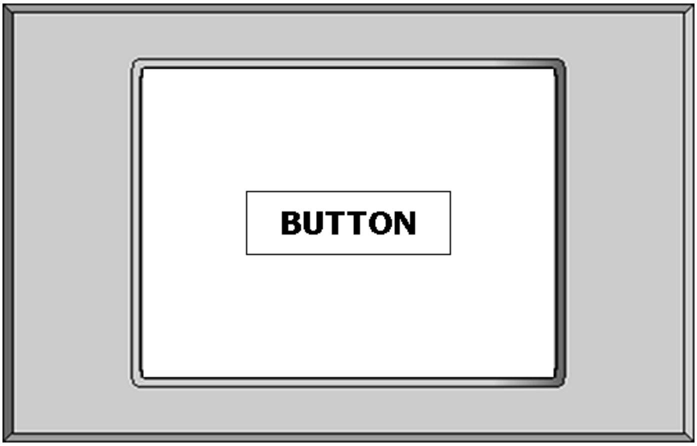
We start with the section „IDENTIFIERS“, where we have to define a font ID:

```
' elements
#define BUTTON_ID              0
' windows
#define WINDOW_ID              0
' fonts
#define FONT_ID                0
```

Rewrite the complete section „TGL ELEMENTS AND WINDOWS“, we now have to create a font to be used for the button text and then the textbutton itself:

```
  call bTglCreateFontParams( &
  FONT_ID, &                  ' identifier of font
  "Valencia", 18, "bold", &   ' name, size, type of font
  "center", "center", &       ' alignment horizontal, vertical
  "prop", 0, &                ' spacing type, blank
  SPACING_CHAR_DEFAULT, 0, &  ' spacing char, vertical
  "imm", "char", &            ' overlay, wrap mode
  blReturn )                  ' return code (0: OK; >0: error)

  call bTglCreateTextbuttonWnd( &
  160, 50, &                  ' width, height of element
  "BUTTON", &                 ' text
  FONT_ID, 1, &               ' font identifier, frame width
  TGL_KEY_ATTR_DEFAULT, &     ' key attr.: auto repeat, beep, type
  BUTTON_ID, WINDOW_ID, &     ' identifier of element, window
  80, 95, &                   ' x, y coordinate on lcd
  0h, &                       ' keycode
  blReturn )                  ' return code (0: OK; >0: error)
```

The section „FLASH“ and the declaration of the datalabels can be deleted, it is not needed for the text button. After compilation and download, the text button looks like this:



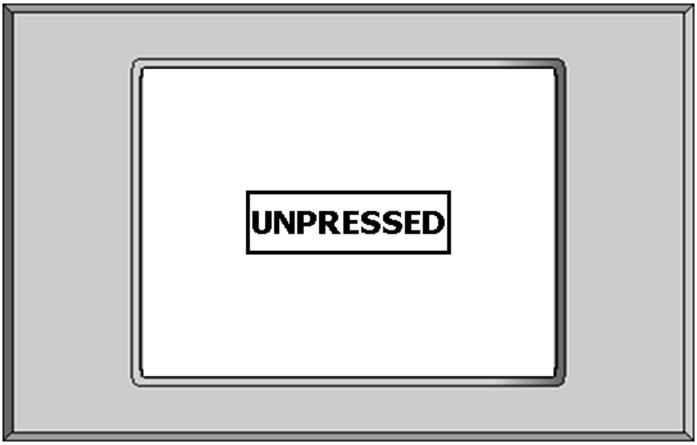
As with graphic buttons, for text buttons you also have the possibility to create an „alternative“ button text (label) to be displayed when the button is pressed. The font parameters can be left unchanged, but we need to modify the rest of the section „TGL ELEMENTS AND WINDOWS“ so that the button is displayed with different text in unpressed and pressed state:

```
call bTglCreateTextbuttonWnd( &
160, 50, &                     ' width, height of element
"UNPRESSED", &                 ' text
FONT_ID, 3, &                  ' font identifier, frame width
TGL_KEY_ATTR_DEFAULT, &        ' key attr.: auto repeat, beep, type
BUTTON_ID, WINDOW_ID, &        ' identifier of element, window
80, 95, &                      ' x, y coordinate on lcd
0h, &                          ' keycode
blReturn )                     ' return code (0: OK; >0: error)

call bTglCreateLabel( &
160, 50, &                     ' width, height of element
"PRESSED", &                   ' text
FONT_ID, 5, &                  ' font identifier, frame width
LABEL_ID, &                    ' identifier of element
blReturn )                     ' return code (0: OK; >0: error)

call bTglLink( BUTTON_ID, LABEL_ID, blReturn)
```

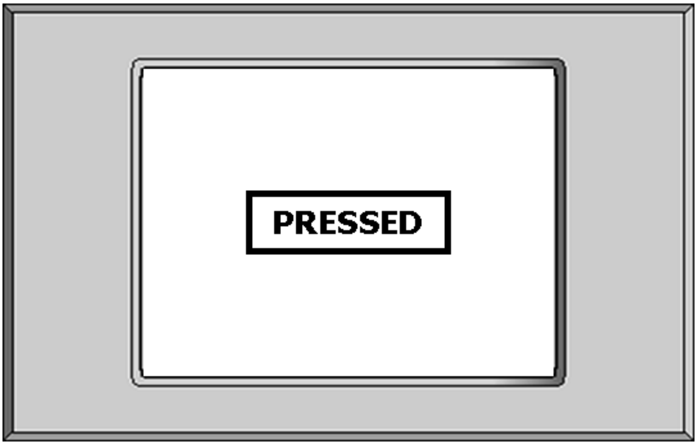
With this we create a text label and link this label to the button. The label is a new element, so we need to define its ID in the section „IDENTIFIERS“:

```
' elements
#define BUTTON_ID                 0
#define LABEL_ID                  1
```

That’s all. After downloading the revised program the button is displayed:



Whenever the button is touched, the text label with the alternative text and the wider frame is displayed:



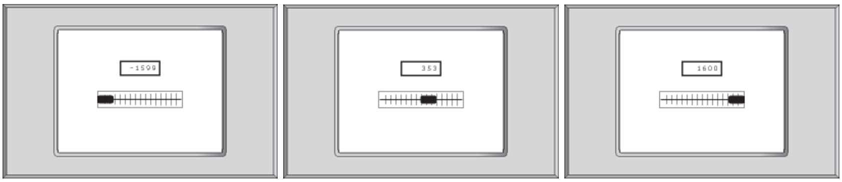
As already mentioned, you have almost the same options for text buttons as you have for graphic buttons. You can enable / disable the beep or make the button a switch button. There are more possibilities for the functionality and look of graphic buttons. In the subdirectory **..\Examples\TGL_Examples** of your Tiger-BASIC installation there are several examples programs with a name starting with **TGL_TEXTBUTTON_**, showing different possibilities for creating text buttons.

Example for sliders

For our second example we take a look at the slider element. A slider is a control element for easily setting values in a certain predefined value range. Open the program **TGL_EXAMPLE_SLIDER.TIG** with the command **Öffnen [Open]** from the menu **Datei [File]**. The file is located in the subdirectory **..\Examples\TGL_Examples\TGL_BASIC_EXAMPLES**.

After loading the program code you can immediately transfer the program to the TP1000 and execute it there. Use the command **Ausführen [Run]** from the menu **Starten [Start]** or press the key **F5**. The program is first compiled and then transferred to the TP1000, where it is automatically started.

On the TP1000 now back display backlight should be switched on and a button is displayed in the centre of the screen:



Minimum *any value* *Maximum*

You can now touch at any position inside the slider area, and directly this will become the current position of the slider and with it set a new value. You can also touch on the current slider position and „drag“ the slider to the desired position/value.

So how does the program work? Many elements we already know from the button example: Some initial settings are done and the Tiger Graphic Library is included. In the section „IDENTIFIERS“ we define three elements (for the slider bar, the slider button and the text label displaying the current value, one window in which the elements are placed and one font (for the text label). We again define some required variables and include the file for device driver installation.

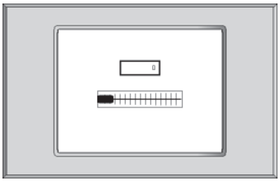
In the section „TGL ELEMENTS AND WINDOWS“ we have 4 library function calls: With the function *bTglCreateSliderWnd* the slider control is created and placed in the window. The function *bTglCreateGraphic* creates the graphic element for the slider button, which is linked to the slider element with the function *bTglLink*. Finally with function *bTglCreateLabelVarWnd* a text label for variable content is created, for showing the curent slider value.

In the section „GET AND SHOW CURRENT SLIDER VALUE“ three library functions are used: First the windows containing slider and text label is shown on the LCD with *bTglShowWindow*. Then, continuously in a loop, the function *lTglGetSliderValue* is used to get the current slider value (calculated from minimum, maximum value and current position). Also the current value is passed to the text label with function *bTglShowLong* and thus displayed.

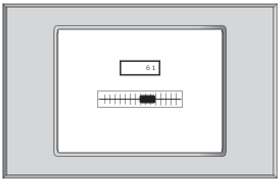
Now we will start making some changes to the slider. First we will adjust the value range of the slider from the initial -1,599 to 1,600 to a percentage range of 0 to 100. We just need to change two values for that in section „TGL ELEMENTS AND WINDOWS“:

```
call bTglCreateSliderWnd( &
168, 32, &                     ' width, height of element
0, 100, &                      ' min, max value primary direction
0, 0, &                        ' min, max value secondary direction
"X", &                         ' type of slider (direction)
dlSlideBar, 168, &             ' address, format width of bitmap
SLIDEBAR_ID, WINDOW_ID, &      ' identifier of element, window
76, 120, &                     ' x, y coordinate on LCD
blReturn )                     ' return code (0: OK; >0: error)
```

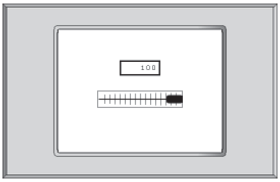
Recompile and start the program again with the command **Ausführen [Run]** from the menu **Starten [Start]** or press the key **F5**. After the new version is started, the same slider positions as before will give you different values:



Minimum



any value



Maximum

As one might want to use a variety of different value ranges, also some optical variations of the sliders’ appearance might be requested. This also can be easily done: Just create a graphic for the slider and a graphic for the slider button as you need it. In the program you only have to change the name of the graphics included with DATA FILTER and the dimensions of the graphics in the function calls for the slider elements.

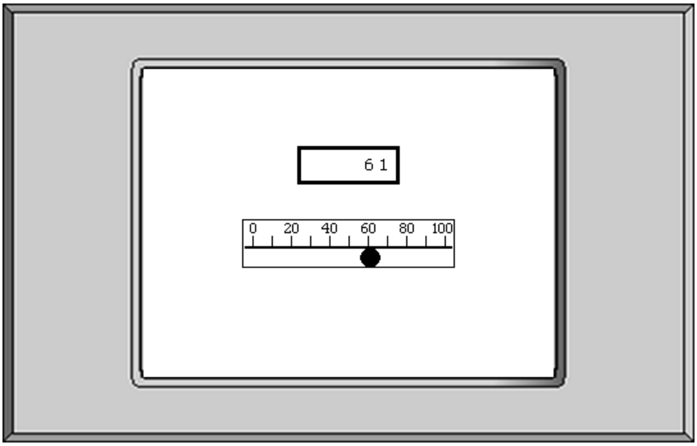
To give an example we will now use a slider graphic of 166 x 38 pixels (instead of 168 x 32) with a fitting slider button of 16 x 38 pixels (instead of 32 x 16). Make the following changes to the section „TGL ELEMENTS AND WINDOWS“:

```
call bTglCreateSliderWnd( &
166, 38, &                    ' width, height of element
0, 100, &                     ' min, max value primary direction
0, 0, &                       ' min, max value secondary direction
"X", &                        ' type of slider (direction)
dlSlideBar, 168, &            ' address, format width of bitmap
SLIDEBAR_ID, WINDOW_ID, &     ' identifier of element, window
77, 117, &                    ' x, y coordinate on LCD
blReturn )                    ' return code (0: OK; >0: error)

call bTglCreateGraphic( &
16, 38, &               ' width, height of element
dlSliderButton, 16, &         ' address, format width of bitmap
SLIDERBUTTON_ID, &            ' identifier of element
blReturn )                    ' return code (0: OK; >0: error)
```

In the section „FLASH" you have to change the graphics that are invoked:

```
dlSlideBar::
data filter "chnl_x_slidebar_alt.bmp", "GRAPHFLT", 0
dlSliderButton::
data filter "chnl_x_slider_black_alt.bmp", "GRAPHFLT", 0
```

Recompiling the program will give you an alternative look of the slider, while the value range (0...100) remains unchanged:



Until now, we only created horizontal sliders. Of course, sliders can also work vertically or even in both directions simultaneously. So our next step would be creating a vertical slider.

Of course, we again need new graphics for the vertical layout of the sliders (we have prepared some for this demo). And we have to change the positions of slider and text label. But most important, we have to change the parameter of the slider element for the vertical functionality.

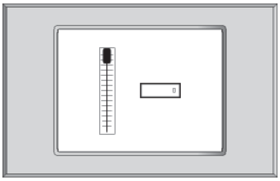
So let's get to the modifications we have to do in our source code. In the section „FLASH" we have to invoke the graphics for a vertical slider:

```
dlSlideBar::
dlSlideBar::
data filter "chnl_slidebar.bmp", "GRAPHFLT", 0
dlSliderButton::
data filter "chnl_slider_black.bmp", "GRAPHFLT", 0
```

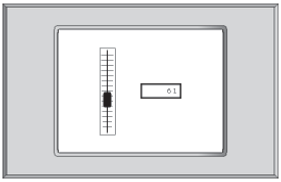
The slider graphic has a size of 32 x 170 pixels (where only 31 are used horizontally), the slider button has a size of 16 x 30 pixels. Now make the following changes to the section „TGL ELEMENTS AND WINDOWS":

```
call bTglCreateSliderWnd( &
31, 170, &                    ' width, height of element
0, 100, &                     ' min, max value primary direction
0, 0, &                       ' min, max value secondary direction
"Y", &                        ' type of slider (direction)
dlSlideBar, 32, &             ' address, format width of bitmap
SLIDEBAR_ID, WINDOW_ID, &     ' identifier of element, window
80, 36, &                     ' x, y coordinate on LCD
blReturn )                    ' return code (0: OK; >0: error)

call bTglCreateGraphic( &
16, 30, &                     ' width, height of element
dlSliderButton, 16, &         ' address, format width of bitmap
SLIDERBUTTON_ID, &            ' identifier of element
blReturn )                    ' return code (0: OK; >0: error)

call bTglLink( &
SLIDEBAR_ID, &                ' identifier of slide bar
SLIDERBUTTON_ID,&             ' identifier of slider button
blReturn )                    ' return code (0: OK; >0: error)

call bTglCreateLabelVarWnd( &
80, 30, &                     ' width, height of element
FONT_ID, 3, &                 ' font identifier, frame width
LABEL_ID, WINDOW_ID, &        ' identifier of element, window
160, 105, &                   ' x, y coordinate on lcd
blReturn )                    ' return code (0: OK; >0: error)
```

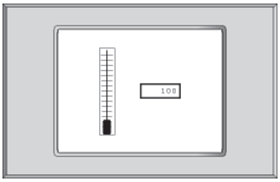
Compiling and running the program will now give you a slider like the initial one, just that you are now have a vertical slider and repositioned elements:



Minimum



any value



Maximum

There are of course more options for sliders. You can create two-dimensional sliders that give back two values in x- and y-direction or you can reverse the direction of a slider by setting the first value parameter of one direction higher than the second value.

In the subdirectory **..\Examples\TGL_Examples** of your Tiger-BASIC installation there are several examples programs with a name starting with **TGL_SLIDER_**, showing different possibilities for creating sliders.

Components of the Graphic Library

The Tiger Graphic Library consists of various components that are handled differently, but combine to one specific result: An output on the display that might include a suitable handling of the touchpanel. Following is an overview of the available components of the Tiger Graphics Library, including a short description of each component.

Elements

The Tiger Graphic Library is based on elements. Everything you can see on the LCD or you can use in your program are elements. The simplest element is a graphic. It is nothing else but a bitmap of a certain size placed on the LCD. More complex elements are e.g. sliders or buttons. These elements additionally contain touch panel functions. In order to display an element on your LCD you have to create it first.

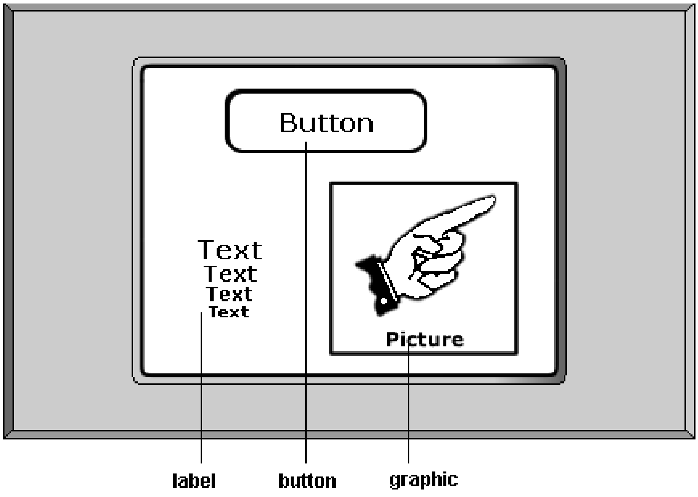


fig.: Elements of the Tiger Graphic Library

Every element has a unique identifier. This identifier is given by the user when he creates the element. You can use or change this element by its identifier. Ensure that you use every identifier for an element only once, even if the elements differ in type. Two graphic elements need two different identifiers. Possible values for the identifiers are 0...65534. In the default configuration the maximum value for an element is **1999**. You can change this limit by creating your own copy of the configuration file *TigerGraphicLibraryConf.INC*. Please read chapter *Configuration* before doing this.

Available elements in the Tiger Graphic Library are:

- Graphic
- Label
- Button
- Text button
- Slider

Windows

A window is a container of many elements which can be shown on LCD together. A created element is saved in the memory but it is not used yet. To show an element on the LCD, it has to be placed into a window. It is possible to create many windows, which can be switched at any time. E.g. you could generate a keyboard and wait for input. After completing the input, a second window with status information can be shown.

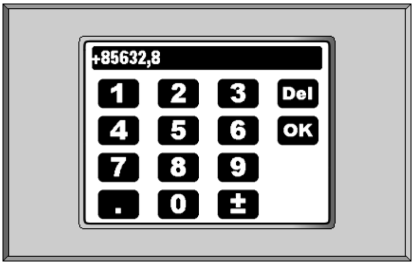


fig.: Input number

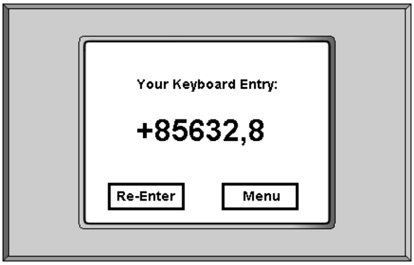


fig.: Display Input

Each window has its own identifier. You can use or change a specific window by its identifier. Possible values for the identifiers are 0...65534. In the default configuration the maximum value for a window is *99*. You can change this limit by creating your own copy of the configuration file *TigerGraphicLibraryConf.INC*. Please read chapter *Configuration* before doing this.

The identifiers of the windows differ from the identifiers of the elements. You can use elements and windows with identical identifiers, e.g. an element with identifier 5 and a window with identifier 5 at the same time are allowed.

There is no problem to use one element in different windows, but never use one element several times in one window.

After placing an element in a window you can show or hide each element as needed in your project.

Creating and Placing Elements

You create an element by defining its width, height and occasionally its specific attributes. After that you can place your element in one or more windows at the position with the coordinates x, y. For the same element this position can be different in each window. Some elements have specific attributes e.g. key codes for buttons which can be different in each window, too. For details see the special chapters for each element.

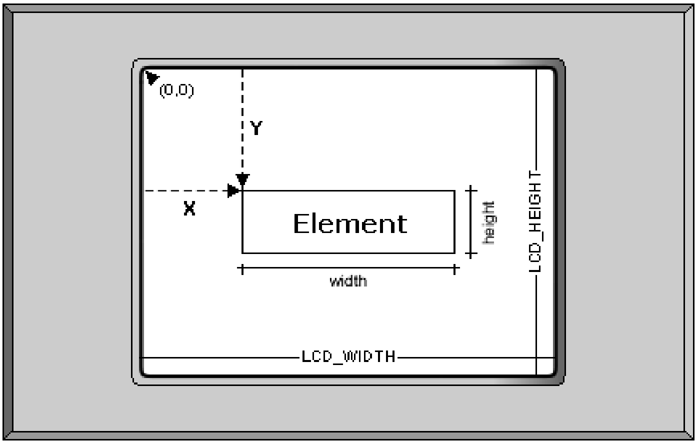
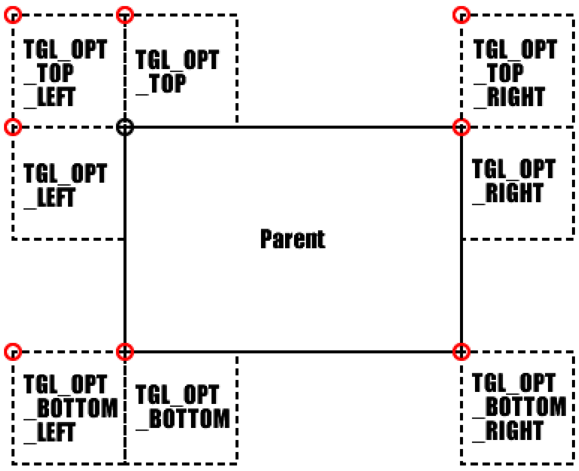


fig.: Creating and placing elements

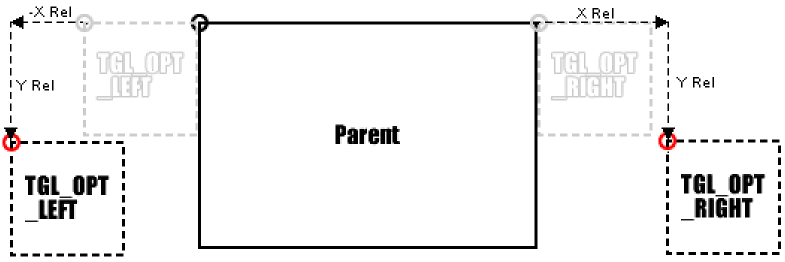
With the placing of elements in a window you will define an assembly of elements which will be shown together on LCD with all its touch panel functionalities. The Tiger Graphic Library will administrate these windows for you. After placing your elements in windows you just need to call a single subroutine for switching from one window to another.

Docking

The standard way to place an element into a window is to set the absolute x-/y-coordinates. The docking function is an easy way to place elements relative to other elements. You can move the docking point with the x-/y-offset.



○ Source
○ Docking Point



○ Source
○ Docking Point

fig.: Docking elements

Integrated Applications

Integrated applications are no elements, but consist of one or more predefines complete windows, which can be easily shown in your projects. Normally they are ready for use after calling an initialization subroutine and calling one single subroutine.

Available integrated applications in the Tiger Graphic Library:

- Keyboards
- RTC Applications

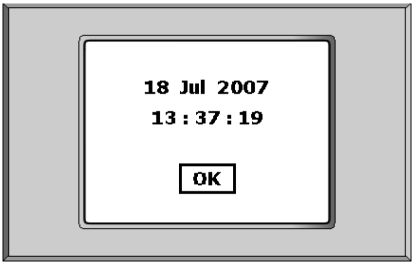


fig.: RTC application as integrated application

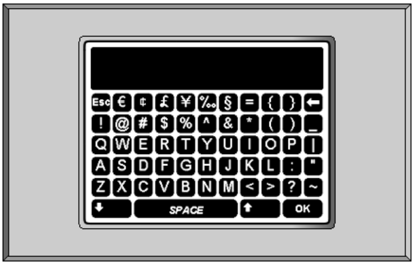


fig.: Keyboard as integrated application

Besides the integrated applications you will find templates in this manual. Have a look on these applications. Surely you will find some useful solutions for your own applications.

User Graphics

It is certainly possible to draw and show graphics by your own with the graphic functions of the Tiger-BASIC™ programming language at the same time as showing a window assembled by the subroutines of the Tiger Graphic Library.

For easy displaying your own graphic the Tiger Graphic Library provides you some special subroutines for LCD output. To be sure not to be disturbed by the outputs of the Tiger Graphic Library please use these subroutines.

Available special subroutines for LCD output in the Tiger Graphic Library:

- *vTglShowUserGraphic*
- *vTglShowUserGraphicParams*
- *vTglHideUserGraphic*
- *sTglGetWindowGraphic*
- *vTglPutWindowGraphic*
- *vTglClearWindowGraphic*

For LCD outputs mind using the one of the two LCD layers the Tiger Graphic Library does not use. The LCD device driver will „or“ these two layers. In the default configuration the Tiger Graphic Library uses the layer 1. You can determine this layer by the define *SHOW_WINDOW_LAYER* in the file *TigerGraphicLibraryDefs.INC*. If you use the special subroutines for LCD output you need not care about this!

Graphic Fonts

The Tiger Graphic Library works with individual graphic fonts. You do not need the LCD fonts anymore. Now you have the possibility to design the fonts by your own style as simple as the LCD fonts do, without any need of more program code.

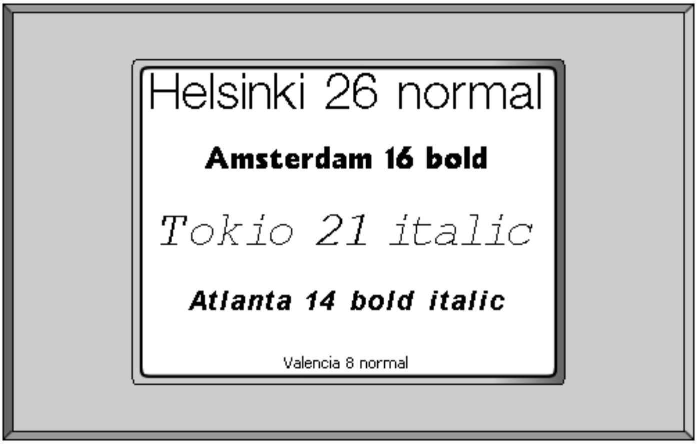


fig.: Various graphic fonts

Usually the text graphics are placed on the LCD using labels. If required, the labels can have touch panel functionalities.

It is also possible to create text graphics without creating any element. For details see the subroutines in chapter *Text Graphics*

Available special subroutines for Text Graphics in the Tiger Graphic Library:

- *bTglCreateFontParams*
- *bTglCreateFont*
- *bTglSetFontParams*
- *bTglGetFontParams*
- *sTglBuildTextGraphic*
- *lTglCalcTextToWindow*
- *sTglCalcTextToPages*
- *lTglGetLineHeight*
- *lTglCalcTextGraphicWidth*

Further Information

For working with the Tiger Graphic Library, but also for working with the Tiger computers in general, there is a lot of additional informationen like e.g. data sheets or manuals available. Following is a list of where to find these.

Data sheets

- Data sheet for TP1000 Touchpanel Computer
  Name: **DATA_Sheet_TP1000_Vxxx_EN.pdf**
  Available at:
  • on the „Info-CD" in directory „\English\TP1000"
  • on our website **www.wilke.de** in the section „Downloads", there under „Datenblätter [Data Sheets]"
- Data sheet for TINY-Tiger Module
  Name: **DATA_Sheet_TINY_Tiger_Module_Vxxx_en.pdf**
  Available at:
  • on the „Info-CD" in directory „\English\Tiger_Modules"
  • on our website **www.wilke.de** in the section „Downloads", there under „Datenblätter [Data Sheets]"
- Data sheet for TINY-Tiger 2 Module
  Name: **DATA_Sheet_T2CI_V1_0_Vxxx_EN.pdf**
  Available at:
  • on the „Info-CD" in directory „\English\Tiger_Modules"
  • on our website **www.wilke.de** in the section „Downloads", there under „Datenblätter [Data Sheets]"
- Data sheet for Ethernet-Adapter EM03
  Name: **DATA_Sheet_EM03_Eth_P_V1_1_Vxxx_EN.pdf**
  Available at:
  • on the „Info-CD" in directory „\English\Ethernet_Web_Adapter\Datasheet_Ethernet"
  • on our website **www.wilke.de** in the section „Downloads", there under „Datenblätter [Data Sheets]"

Manuals

● Tiger-BASIC Manuals (German)
Names:
• Hardware Manual: **Hardware_v5.pdf**
• Programming Manual: **Programmierung_v5.pdf**
• Device Driver Manual: **Device-Treiber_Applikationen_v5.pdf**
• Manual for software version 5.2: **Neue_Funktionen_v5_2.pdf**
Available at:
• on the „Info-CD" in directory „\Deutsch\Tiger_Basic_Handbücher"
• in your Tiger-BASIC® installation in subdirectory „..\Manual"
• on our german website **www.wilke.de** in the section „Downloads", there under „BASIC-Tiger™ Handbücher"
• in printed form as hardback manual set in our Online-Shop

● Tiger-BASIC Manuals (English)
Names:
• Hardware Manual: **Hardware_v5.pdf**
• Programming Manual: **Programming_v5.pdf**
• Device Driver Manual: **Device-Driver_Applications_v5.pdf**
• Manual for software version 5.2:
Manual_Tiger-BASIC_Update_v5_2_englisch.pdf
Available at:
• on the „Info-CD" in directory „\English\Tiger_Basic_Manuals"
• in your Tiger-BASIC® installation in subdirectory „..\Manual"
• on our english website **www.wilke.de** in the section „Downloads", there under „BASIC-Tiger™ Manuals"
• in printed form as hardback manual set in our Online-Shop

● Manual for Tiger Graphic Library
Name: **TigerGraphicLibrary_Vx.xx.pdf**
Available at:
• on the „Info-CD" in directory
„\English\TigerGraphicLibrary\TigerGraphicLibraryV1.00\Manual\TigerGraphicLibrary"
• in your Tiger-BASIC® installation in subdirectory
„..\Manual\TigerGraphicLibrary"
• on our english website **www.wilke.de** in the section „Downloads", there under „BASIC-Tiger™ Handbücher [BASIC-Tiger™ Manuals]"

- Manual for Ethernet Programming Library
  Name: **Ethernet_Web_Adapter__Programming_Guide_V_x_xx.pdf**
  Available at:
  • on the „Info-CD“ in directory „\English\Ethernet_Web_Adapter\Manuals“
  • in your Tiger-BASIC® installation in subdirectory „..\Manual\Ethernet_Web_Manual“
  • on our english website **www.wilke.de** in the section „Downloads“, there under „BASIC-Tiger™ Handbücher [BASIC-Tiger™ Manuals]“

- Beschreibungen neuer Treiber und/oder Funktionen
  Namen: unterschiedlich, je nach Treiber und Funktion
  Zu finden:
  • on the „Info-CD“ in directory „\Deutsch\Tiger_Basic_Handbücher\Neue_Treiber_und_Funktionen“ or „\English\Tiger_Basic_Manuals\new_driver_and_functions“
  • in your Tiger-BASIC® installation in subdirectory „..\Manual\Extra_Manuals“
  • on our english website **www.wilke.de** in the section „Downloads“, there under „BASIC-Tiger™ Handbücher [BASIC-Tiger™ Manuals]“

Current Versions

Drivers, examples, applications, libraries, data sheets and manual addendums in their newest versions are available for download on our website under **http://www.wilke.de/downloads.php**.

Additionally you can have new versions and updates, when available, be automatically sent to you by an email newsletter (ca. 2 - 4 times per year). Registration for the newsletter under **http://www.wilke.de/newsletter.php**.

Thank you for your purchase at Wilke Technology - and please use our free of charge technical support when you need technical help or anything else we can assist you at. Have fun working with the Tiger Graphic Library!

Your Tiger Support Team

support@wilke.de
Tel.: +49 (241) 918 9032
Fax: +49 (241) 918 9044

TINY, HIGH SPEED MULTITASKING COMPUTERS

Wilke Technology

Wilke Technology GmbH
52018 Aachen · Germany · P.O.Box 1727
52070 Aachen · Krefelder Str. 147
Tel: +49.241.918 900· Fax: +49.241.918 9044
e-mail: support@wilke.de· http://www.wilke-technology.com