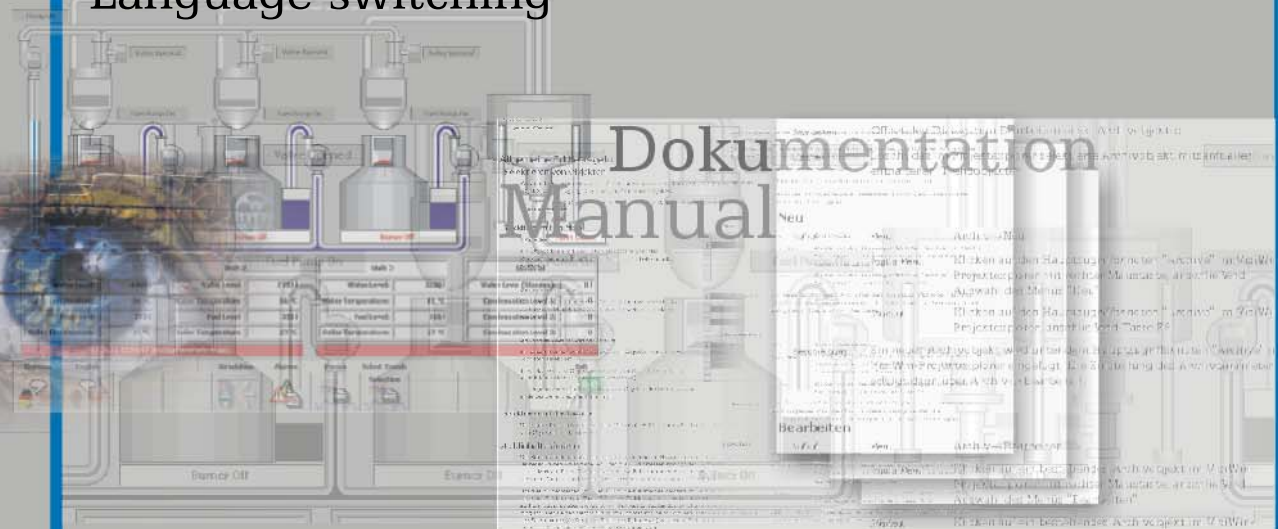


# VisiWinNET 2005

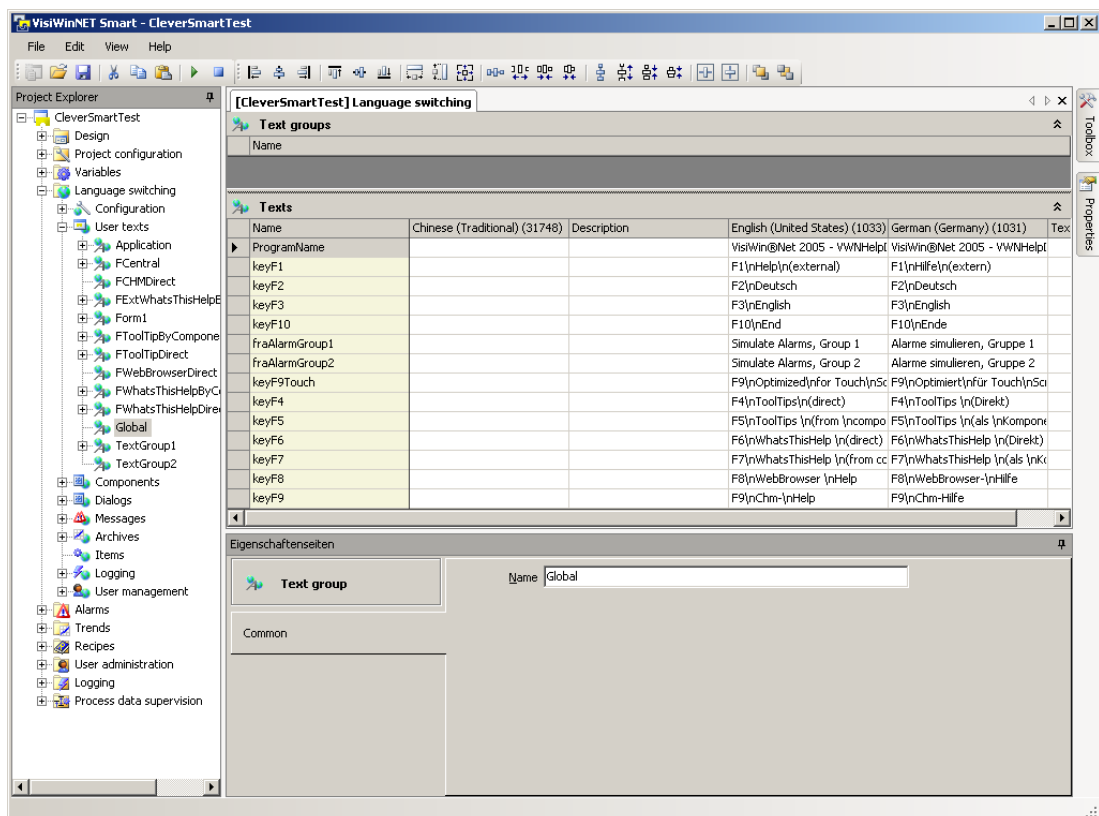
## Language switching



- **VisiWin**
- **VisiWinNET 2005**
- Common
- Class Library
- **Systems**
- Tools
- Technical Informations
- **Inosoft OPCServer**
- Basics and helping tools
- Protocols

# VisiWinNET 2005

## Language switching









The contents of this manual must not otherwise be used without explicit written consent from INOSOFT GmbH.

We have checked the contents of this manual for compliance with the described software. Discrepancies can, however, not be ruled out. For this reason we cannot guarantee full compliance. The contents of the manual are subject to regular checking for necessary updates/amendments. Such amendments will be made in the subsequent edition.

Suggestions for improvement are welcome.

## Legend

In order to point out particular paragraphs the following symbols are used in the INOSOFT documentations:

	<b>Attention</b>	Passages with this sign should be read – and observed – with particular attention.
	<b>Hint</b>	Important paragraph “additional information”
	<b>Tip</b>	Many roads lead to Rome; here a shortcut is to be found.
	<b>In work</b>	Functions that are in preparation or already implemented but not yet prepared for documentation.
	<b>Example execute</b>	Instructions to be carried out in an example
	<b>Observe result</b>	Results to be observed with carrying out the exemplary instructions

© / ™ / ®

Windows®, Windows NT®, Windows 2000®, Windows XP® are registered trademarks of the Microsoft company.

Further product names marked ® are trademarks of the appropriate manufacturers.

INOSOFT GmbH created on

VisiWinNET Version: from 6.04.000

created on 08.06.2010

# Contents

<b>1 Preamble</b> .....	<b>1</b>
<b>2 VisiWinNET language management</b> .....	<b>2</b>
<b>3 Design</b> .....	<b>5</b>
<b>4 Language Management definitions</b> .....	<b>6</b>
4.1 Definitions and product versions in the localization.....	6
4.2 Index text group .....	6
4.2.1 Edit Index text groups .....	7
4.3 Index text .....	8
4.3.1 Edit Index texts.....	8
4.4 Font class.....	13
4.4.1 Edit Font classes .....	13
4.5 Font.....	14
4.5.1 Edit Fonts .....	15
4.6 Language .....	16
4.6.1 Edit Languages .....	16
<b>5 language management definition parameters</b> .....	<b>17</b>
5.1 Parameter in alphabetical order.....	17
5.1.1 Character set .....	17
5.1.2 Description .....	17
5.1.3 Font name .....	18
5.1.4 Input locale .....	18
5.1.5 Italic.....	18
5.1.6 Language.....	18
5.1.7 Name .....	18
5.1.8 Size .....	18
5.1.9 Speech columns .....	19
5.1.10 Strikethrough .....	19
5.1.11 Underline .....	19
5.1.12 Weight.....	19
<b>6 Access to localizable text</b> .....	<b>20</b>
6.1.1 Text selection dialog.....	20
6.1.2 Functions.....	21

# 1 Preamble

## About this manual

This manual contains specific information on the VisiWinNET localization such as (amongst others) the explanation of the involved components, the operating reference of the editor, and the description of the index text system definitions.

## Questions and Problems

For technical questions and problems please contact your responsible INOSOFT agent or the INOSOFT GmbH Support under +49 (5221) 16 66 02 or email: [Support@INOSOFT.com](mailto:Support@INOSOFT.com)

Frequent questions and problems are dealt with on our homepage under [www.inosoft.com](http://www.inosoft.com)

There you will also find a support area for direct contact with our Main Office.

## 2 VisiWinNET language management

Index texts serve for speech-controlled display of texts in the application. If a project is sold international, all user interface texts (all texts, presented during the application run) must be presented in the respective national language

The easiest technical possibility of realizing a language change would be, to load all texts in dependence to the notified language, routined. I.e.:

```
Private Sub FCentral_Load(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles MyBase.Load
```

```
    Select Case gLanguage
        Case 1031
            btnEnd.Text = "Ende"
        Case 1033
            btnEnd.Text = "End"
    End Select
```

```
End Sub
```

The programmatic effort is comparatively fat there, as each displayed text must be programmed for each language in the source text.

Programm languages like VB.NET, C# etc., support language change by resources.

During program run time the resources may be loaded as strings in the control elements. Language change is processed through switch over of the resources in dependence on a national specific setting, respectively the system language.

```
Private Sub FCentral_Load(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles MyBase.Load
    Dim myAssembly As System.Reflection.Assembly
    myAssembly = Me.GetType.Assembly
    Dim myManager As New _
        System.Resources.ResourceManager("MyProject.MyRes", _
        myAssembly)
    Dim myString As System.String
    btnEnd.Text = myManager.GetString("EndButtonText")
```

```
End Sub
```

This saves changeovers in dependence of the language but has the disadvantage that it is still necessary to specify every text to be loaded in the source code.

In dependence on the language, the VisiWin-Language management system enables the optional selection of texts independent of the operating system.

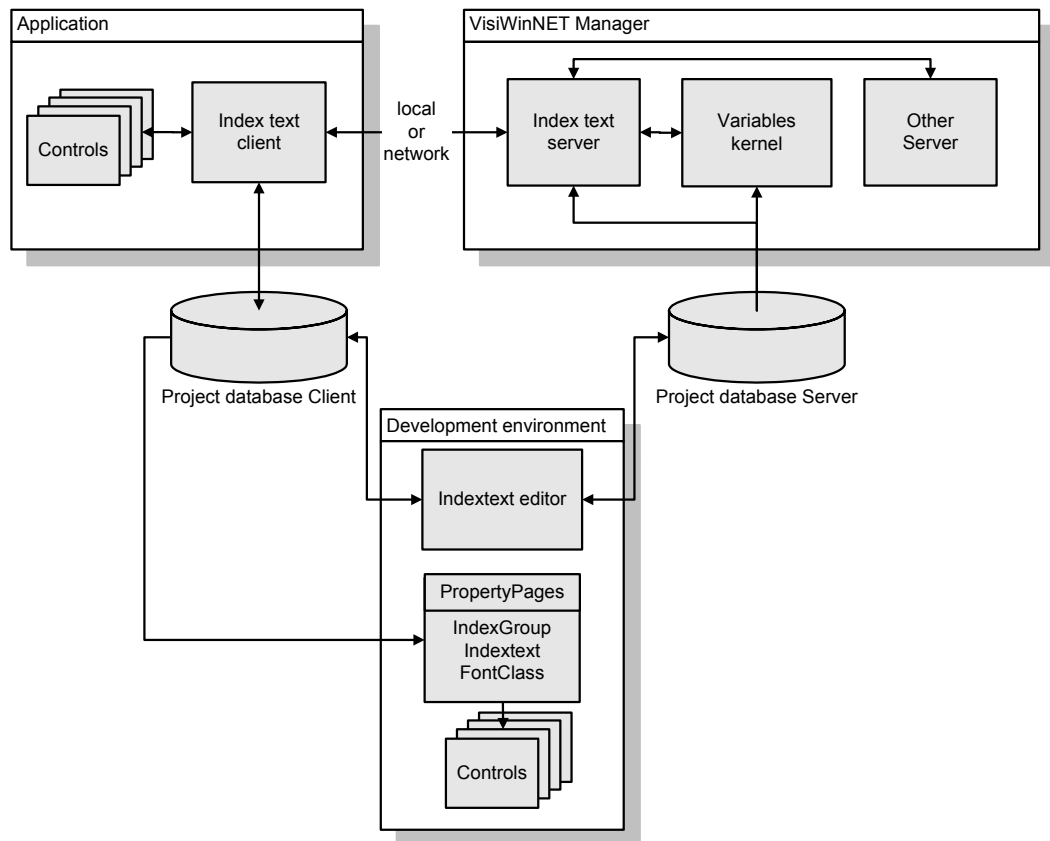
VisiWin-Index texts have following advantages:

- The index text definition is a text group of multilingual texts of the same content. Index texts may be changed during run time. Thus it is possible i.e., to present user texts in the national language of each operator, without restart of application.
- Specific VisiWin-control elements can represent index texts in dependence on the process variable value. Thus in addition to the dependence on the current language the dependence on the variable value is achieved.
- With the definition of font classes, for each language fonts are defined, which are actiated automatically with language change. Thus it is possible, i.e. to switch over between western and cyrillic character set.

- The selection of index texts is processed through the setting of priorities and must not be programmatically.

The block diagram shown below introduces the localization components. Cross references are provided to show further information about the components.

VisiWinNET language management Components



**Variables kernel**

The variables kernel provides data access.

**Index text server**

The index text server accesses the server project database. Its function is the distribution of requested index texts within the system. The index text definitions of the project database contain the information whether dynamic process values are to be tied in with the text. If this is the case the index text server will obtain the appropriate values from the variables kernel.

**Other Servers**

Specialized components such as the alarm server display text information. They request the texts from the index text server.

**Index text client**

The index text client supplies the index texts to the control elements of an application. Application-related index texts are obtained directly from the client project database. Texts that are used system-wide by multiple clients have to be obtained via the index text server. A localization function in the application causes the index text client to request the texts directly in the new language.

**Controls**

Fundamentally, all VisiWinNET control elements that allow text display in any shape or form are localizable, and therefore directly or indirectly connected to the index text system. The output allows display of firm texts, texts that are changeable dependent on a process value, and dynamic texts than can contain current process values.

**Project database Server**

Texts that are used system-wide, i.e. with all client applications (e.g. alarm texts in alarm controls) are saved in the project database. They are passed on via the index text server to other servers (e.g. alarm server, archive server, ...). Fundamentally, in the server project database such texts are saved that are to be used system-wide (i.e. with all client applications).

**Project database Client**

Application-dependent texts such as the inscription of a control area are saved in the database of the client project. Fundamentally, the client project database contains all texts that are to be selected via control element properties.



The distinction between client and server project database only takes place within a client-server project. A single-workstation application ("Single" project type) works with only one project database that contains the system-wide as well as the application-specific texts.

**Index text editor**


The index text editor allows the projection of the localization definitions. This encompasses the texts as well as fonts and font classes.

**Property pages**

The property pages allow user-friendly selection of definitions from the project database at development time. They are an integral part of the VisiWinNET control elements, can, however, also be tied in as components with the development of own user control elements. As a projection support for the localization VisiWinNET provides property pages to select texts, text groups and font classes.

### 3 Design

VisiWin provides an editor for the projection of the localization. This is integrated with the development environment when VisiWinNET is installed. The definitions of the localization are stored in the project database.

The text editor is represented in the VisiWinNET Project Explorer by the  symbol. This is the main access node for the definitions of the localization.

Following the first click on the "Localization" node the editor initializes itself in the Project Explorer. The following access nodes are added:

<b>User texts</b>	After opening a new project the "User Texts" group is empty. Under this node new index text groups can be created. All application-specific text can be projected in the "User Text" branch.
<b>Alarms, Archives, Variables, Logging, User administration</b>	These branches contain language-dependent text that is used by the appropriate systems (editors). The contents (sub-groups and text) depend on the individually projected definitions in the appropriate editors. Text in these text groups can be edited but neither deleted nor can text be added. The generation of these text is controlled by the other system editors.
<b>Components, Dialogs</b>	These groups contain all text that is required by the VisiWin control elements or other VisiWin components. A further hierarchic classification provides further information about the utilization. The text in these sub-groups is available for the selected project languages directly after a new project has been created. It can be edited but not deleted. Also, new text cannot be added. The contents of this area depend on the VisiWinNET installation version. When a new language is added this text must also be translated in order for the system to run error-free.
<b>Configuration language switching</b>	Here special definitions of the localization are listed.

#### Definitions in the configuration node

<b>Fonts</b>	List of fonts that can be extended by the developer. Fonts that are defined here are used by the font classes of the project.
<b>Font classes</b>	Allows access to the font classes defined in the project. Font classes contain a font for every project language. In the controls reference can be established to a font class so that with a change of the language the font is changed, too.
<b>Languages</b>	Lists the languages of the project. Languages projected here appear as columns in the text and font classes tables.


## 4 Language Management definitions

### 4.1 Definitions and product versions in the localization

Below a collection of the definitions of the VisiWinNET localization with the additional information as to which product version contains the definition.

Definition	VisiWinNET Enterprise	VisiWinNET Standard	VisiWinNET Compact	VisiWinNET Embedded
<b>Index text group</b>	✓	✓	✓	✓
<b>Index text</b>	✓	✓	✓	✓
<b>Font class</b>	✓	✓	✓	✓
<b>Font</b>	✓	✓	✓	✓
<b>Language</b>	✓	✓	✓	✓

### 4.2 Index text group

Index text groups provide for index text structuring. They appear in the VisiWin-Project Explorer under the main node "Index texts" (symbol ) and provide a.o.t. for following functionality:

- Structuring by function groups (operator texts/alarms/archives...).
- Structuring by functionality (Access of a Label control to index texts with the definition of an index text group).
- Possibility of structuring by project devices (i.e. forms) for better export of programm divisions.
- Path specification for the access on index texts from the source code.

Index text groups can be nested hierarchically like a file structure. This enhances the clarity of the project layout.

#### Parameter of the index text group definitions

Name	Description
<b>Name</b>	Unique index text group designator

### 4.2.1 Edit Index text groups

Text groups are displayed in the Project Explorer under the "Localization" node.

Text groups can be hierarchically nested similar to an index structure.

The top hierarchy of the text groups provides information about the origin, and with this about the possible projection steps:

- |  |  |
|--|--|
| <b>User text</b>   | The developer can project his own hierarchy of text groups underneath the "User Text" node.  |
| <b>Alarms, archives, variables, logging, user administration</b> | These groups are automatically generated by definitions in the other system editors. Editing these text groups is not possible. The localizable text within the text groups can be edited but not deleted and no text can be added.                      |
| <b>Components, dialogs</b>                                       | These groups contain text used by the system. This is already in existence when a new project is created. Editing these text groups is not possible. The localizable text within the text groups can be edited but not deleted and no text can be added. |

This means that new text groups and text can only be created under the "User Text" node.

The only setting of a text group is the name. The editor provides the following functions for the projection of text groups:

- |                                       |   |
|---------------------------------------|---|
| <b>Create new group</b>               | Through the "New" entry in the context menu of the "User Text" node or the existing text group node a sub-group is created.   |
| <b>Edit the parameters of a group</b> | A click on a text group node loads the name to the VisiWinNET properties page. Here the parameter values of the text group can be edited.   |
| <b>Delete group</b>                   | Through the "Delete" entry in the context menu of a text group node a group is deleted. In the process all secondary definitions (sub-groups and text) are also deleted from the project. |

## 4.3 Index text

The index text definition includes a text for each design language. Index texts provide a.o.t. following functionality:

- Index texts may include format statements for dynamic process variable value presentation.
- The index text name may be used as offset, to indicate text in dependence on a process variable value.
- Index text definitions may be enhanced in the speech size. (Also see Languages)

### Parameter of the index text definition

Name	Description
<b>Name</b>	Unique index text designator
<b>Description</b>	Describing short text for the translator.
<b>Speech columns</b>	Include the texts, that should be displayed.


### 4.3.1 Edit Index texts

Index text is projected in text groups, and displayed in the table editor. Every text contains an editable set of parameters: Here the name and the localizable text are set.

The table editor of the localization is opened through clicking on an appropriate node in the Project Explorer.

The editor provides the following functions for text projection:

**Create new text** Through the "New" entry in the context menu of the table editor a new text is added to the project.

**Edit the parameters of a text** Editing a text directly in the fields of the table editor is allowed. A button  is built into the language columns that through the "Edit Text" dialog allows to add special functions to text:

- dynamic process variable values
- date/time formattings
- dynamically linked index text.

**Delete text** One or multiple text can be deleted by:





- first highlighting the text that is to be deleted (click on the selector column at the l.h. margin of the table, where applicable with the Ctrl of Shift key held down for multiple selection)
- then selecting the "Delete" entry in the context menu of the table editor.



## “Edit Text” dialog



The dialog for editing an index text contains the following components:

<b>Index text</b>	The index text text field contains the text as well as the formatting instructions.
<b>Preview</b>	The preview text field displays the contents of the index text field with interpreted formatting instructions.
<b>Parameters</b>	Through these function blocks process variables can be dynamically tied in with index text.

The function blocks contain the following operating elements:


Element	Description
	Opens the dialog for the selection of a process variable. The variable selected here is added as an element to the “Variables” list.
	Deletes the element selected in the “Variables” list.
	Moves the element selected in the “Variables” list upwards by one entry.
	Moves the element selected in the “Variables” list downwards by one entry.
“Variables” list	Contains the names of all variables whose values are to be dynamically tied in with the text at runtime.
Positions before decimal point	Determines the number of positions before the decimal point with values that are to be tied in numerically.
Positions after decimal point	Determines the number of positions after the decimal point with values that are to be tied in numerically.
Leading zeroes	Determines with values that are to be tied in numerically that the number of positions before the decimal point is to be made up with leading zeroes.

Index text / date-time format	<p>Depending on the selected formatting the input box provides the facility of tying in either further index text or date/time formatting.</p> <p>If the formatting is set to "Index text" (firm or variable) existing text can be added to the index text. In the process firm text is to be directly adopted. Variable text can be added by selecting an index text group and a process variable. The value of the process variable determines at runtime the index text to be displayed in the selected index text group.</p> <p>The "Date/time format" (system or variable) formatting allows the display of formatted date/time details. The "System" format shows at runtime the current system time with the index text. Through the "Variable" format the value of the process variable selected in the "Parameter" field is converted into a date/time format.</p>
"Add" control box	<p>Adds a format instruction to the index text that equals the settings of the Parameters/Formatting function blocks.</p>
Decimal	<p>Sets the output format of a dynamically tied in value to whole number decimal.</p>
Floating point	<p>Sets the output format of a dynamically tied in value to afflicted with positions after decimal point.</p>
Hexadecimal	<p>Sets the output format of a dynamically tied in value to the hexadecimal notation.</p>
Octal	<p>Sets the output format of a dynamically tied in value to the octal notation.</p>
Binary	<p>Sets the output format of a dynamically tied in value to the binary notation.</p>
Text	<p>Allows the output of a process variable of the VT_BSTR data type.</p>
Index text (firm)	<p>Allows to statically tie in another index text definition. If this setting is selected for formatting the "Index text" input box is activated. Through the  button the dialog to select an index text is opened.</p>
Index text (variable)	<p>The selection of an index text group and a process variable allows to dynamically tie in index text. Through the  button the dialog to select an index text group is opened.</p>

Date/time format (system)	Displays at runtime the system time with the selected formatting with the index text. Through the  the dialog to select a date/time format is opened.
Date/time format (variable)	Displays at runtime the value of the selected process variable as a date/time format with the selected formatting with the index text. Through the  button the dialog to select a date/time format is opened.


## Operation

The following steps are required to tie in a dynamic process variable value:

- Adding a process variable to the list through the  button:  
In the subsequent variable selection dialog a process variable can be selected.
- Selecting the display mode in the "Formatting" and "Parameters" operating groups.
- Putting the cursor into the "Index text" text field to the position in the text where the process variable value is to appear at runtime.
- Operate "Add" button.

→A format block is added to the text.

The following steps are required to tie in a dynamic index text:

- Selection of the "Index text (variable)" formatting.
- Selection of an index text group in the "Index" text field.
- Adding a process variable to the variables list through the  button:  
The value of the process variable set here is at runtime used for the selection of the index text definition in the selected index text group. This does, however, also mean that all index text definitions must have plain numeric names.
- Putting the cursor into the "Index text" text field to the position in the text where the index text that is to be dynamically changed is to appear.
- Operate "Add" button.

→A format block is added to the text.

## Format block breakup

The format block consists of the following components: @ABC.D@

Example: @1v4.5@

Format symbol	Example	Meaning
<b>@</b>	<b>@</b>	Format start symbol
<b>A</b>	<b>1</b>	Indenture number of the variable in the parameters list
<b>B</b>	<b>v</b>	Format symbol, resulting from the "Variable depiction" and "Leading zeroes" settings
<b>C</b>	<b>4</b>	Positions before decimal point
<b>.</b>	<b>.</b>	Identifier to indicate that settings have been made that are different from the standard settings for positions before and after decimal point
<b>D</b>	<b>5</b>	Positions after decimal point
<b>@</b>	<b>@</b>	Format end symbol

## 4.4 Font class

The font class definition includes a reference to a font definition for each design language.

Font classes enhance the property "Font" in the VisiWin-Control elements. If the language is changed during run time, the control elements, in which a font class was defined as property, changes to the appropriate speech font of the font class. Thus the change between different character sets (i.e. western / greek/ cyrillic) is possible, if the appropriate language was selected.

### Font class definition parameters

Name	Description
<b>Name</b>	Unique font class designator.
<b>Speech columns</b>	Include the fonts, which are to be presented within a language change.

#### 4.4.1 Edit Font classes

Font classes are displayed in the table editor through a click on the equally named node in the Project Explorer (Language Change→Configuration→Font Classes). Every font class contains an editable set of parameters that is dependent on the project languages: Here the name and fonts for every project language are set.

The table editor of the language change is opened through a click on an appropriate node in the Project Explorer.

The editor provides the following functions for the projection of font classes:

<b>Create new font class</b>	Through the "New" entry in the context menu of the table editor a new font class is added to the project.
<b>Edit the parameters of a font class</b>	It is allowed to edit a font class directly in the fields of the table editor. The language columns contain a selection list to select a font definition contained in the project.
<b>Delete font class</b>	One or multiple font classes can be deleted by: <ul style="list-style-type: none"> <li>• first highlighting the font class that is to be deleted (click on the selector column at the l.h. margin of the table, where applicable with the Ctrl or Shift key held down for multiple selection</li> <li>• then selecting the "Delete" entry in the context menu of the table editor.</li> </ul>

## 4.5 Font

Font definitions are components of font classes. They provide a.o.t. for following functionality:

- Inclusion in the run time generation.
- Each font definition describes an entire Font. With language change in addition to the text, automatically the font is changed. Thus the automatic change of character sets (i.e. cyrillic, greek) is possible.

### Font definition parameters

Name	Description
<b>Character Set</b>	Character set identification
<b>Fontname</b>	Name of Font
<b>Italic</b>	Italic
<b>Name</b>	Definition name
<b>Size</b>	Font size
<b>Strikethrough</b>	canceled
<b>Underline</b>	Underscored
<b>Weight</b>	Font intensity

### 4.5.1 Edit Fonts

Fonts are displayed in the table editor through a click on the equally named node in the Project Explorer (Language Change→Configuration→Fonts). Every font contains an editable set of parameters: Here the name and further font characteristics are set.

The table editor of the language change is opened through a click on an appropriate node in the Project Explorer.

The editor provides the following functions for the projection of fonts:

- |                                      |   |
|--------------------------------------|---|
| <b>Create new font</b>               | Through the "New" entry in the context menu of the table editor a new font definition is added to the project   |
| <b>Edit the parameters of a font</b> | The VisiWinNET properties page displays the parameters of the font highlighted in the table editor. Editing a font is, however, also allowed directly in the fields of the table editor.  |
| <b>Delete font</b>                   | One or multiple fonts can be deleted by: <ul style="list-style-type: none"><li>• first highlighting the font that is to be deleted (click on the selector column at the l.h. margin of the table, where applicable with the Ctrl of Shift key held down for multiple selection</li><li>• then selecting the "Delete" entry in the context menu of the table editor.</li></ul> |

## 4.6 Language

A definition set as a project language allows a change to the selected language at runtime. In the development environment adding a language has the effect of a column being added to the text and font classes tables. The column head contains amongst others the language identifier (LCID). The fields of the column allow the allocation of a text to the index definitions of the project. In the font class definition a font is to be specified in the individual columns.

Language definitions contain an adjustable key layout for the text column in the text editor. At runtime, however, the information is not used.

### Language definition parameters

<b>Input locale</b>	Defines the key layout for the text column.
<b>Language</b>	Sets the languageidentifier for the project language.

### 4.6.1 Edit Languages

Languages are displayed in the table editor through a click on the equally named node in the Project Explorer (Language Change→Configuration→Languages). Every language contains an editable set of parameters: Here the name and further characteristics are set.

The table editor of the language change is opened through a click on an appropriate node in the Project Explorer.

The editor provides the following functions for the projection of languages:

**Create new language** Through the "New" entry in the context menu of the table editor a new language is added to the project. Adding a language has the effect of a column being added to the text and font classes tables

**Edit the parameters of a language** Editing a language is allowed directly in the fields of the table editor.

**Delete languages** One or multiple languages can be deleted by:

- first highlighting the language that is to be deleted (click on the selector column at the l.h. margin of the table, where applicable with the Ctrl of Shift key held down for multiple selection
- then selecting the "Delete" entry in the context menu of the table editor.

The deletion has the effect of all appropriate text and fonts being irrevocably deleted from the project.

## 5 language management definition parameters

Below an alphabetic collection of the parameters with the additional information as to which VisiWinNET version supports the parameter.

Parametername	VisiWinNET Standard	VisiWinNET Compact	VisiWinNET Embedded
<b>Character set</b>	✓	✓	✓
<b>Description</b>	✓	✓	✓
<b>Italic</b>	✓	✓	✓
<b>Fontname</b>	✓	✓	✓
<b>Input locale</b>	✓	✓	✓
<b>Language</b>	✓	✓	✓
<b>Name</b>	✓	✓	✓
<b>Size</b>	✓	✓	✓
<b>Speech columns</b>	✓	✓	✓
<b>Strikethrough</b>	✓	✓	✓
<b>Underline</b>	✓	✓	✓
<b>Weight</b>			

### 5.1 Parameter in alphabetical order

#### 5.1.1 Character set

Parameter for

Font

Description

Indicates the font character set. The provided character sets depend on the selected font. A character set includes the fonts, which are used in one region or country.

Example:

Cyrillic characters are used in Russia. Western fonts are not common there, as the entire literature is based on Cyrillic characters. To export an application to Russia, the user interface must use the Cyrillic character set.

#### 5.1.2 Description

Parameter for

Index text

Description

The "Description" parameter serves to explain to a translator the utilization of a text. Here the developer can enter a describing text supplying additional information about the utilization of the index text definition. At runtime the parameter has no function.

### 5.1.3 Font name

Parameter for

Font

Description

Indicates the font name. .

### 5.1.4 Input locale

Parameter for

Language

Description

Determines the key layout in the appropriate language text column of the editor. A key layout sets the allocation of the keys on the keyboard to the characters of a character set. This setting is only valid for the editor, and has no function at runtime.

### 5.1.5 Italic

Parameter for

Font

Description

Indicates, if the font *Italic* shall be presented.

### 5.1.6 Language

Parameter for

Language

Description

Determines the language identifier (LCID) for the project language. The LCID is a numeric constant. An individual LCID is allocated to every language known worldwide.

### 5.1.7 Name

Parameter for

Index text, Index text group, Font class, Font

Description

Unique definition designation by the name.

### 5.1.8 Size

Parameter for

Font

Description

Indicates the Size (in points) of the Font.

Example:

8 pt

10 pt

12 pt

### 5.1.9 Speech columns

Parameter for

Index text, Font class

Description

Font classes and index texts include a column in the editor tables for each design language.

Index text definition: The texts are included in the speech columns. With the integrated button the editor for editing format characters (dynamic process variable value inclusion) is opened.

Font class definitions: Numerical references to font definitions are included in the speech columns. The integrated button provides for a representative list of all designed font definitions.

Also see:

→Languages

### 5.1.10 Strikethrough

Parameter for

Font

Description

Indicates, if the font shall be presented strikethrough.

Example: ~~durchgestrichener Text~~

### 5.1.11 Underline

Parameter for

Font

Description

Indicates, if the font should be presented underscored.

Example:

Underscored text

### 5.1.12 Weight

Parameter for

Font

Description

Indicates the font weight (thickness of letters). The Windows-Standard-FontDialog allows the settings *Normal* and **bold** only. In the font table, i.e. an extra bold value may be set, if necessary.

## 6 Access to localizable text


Following the projection of localizable text VisiWinNET supports different access functions.

The most widely used way is linking control elements through the text selection dialog. This dialog supports the projector with the selection of the text that is to be displayed in the control element in the currently set language.

For plain programmatic access to variables VisiWinNET contains functions that allow the programmer reading individual texts, and monitoring the localization.

Changing the language to be currently displayed is again made through the



### 6.1.1 Text selection dialog

All control elements displaying text from the localization contain the "LocalizedText" property. This property contains a  button in the property window to open the text selection dialog.



The text selection dialog lists the texts from the localization hierarchically. Within the dialog new texts and text groups can be added.

If for example a Label is to be linked with a localizable text there are two different procedures:

**The text was already defined in the localization editor** The text group display on the left allows navigating to the desired text group. Subsequently, the appropriate text is to be selected from the text list on the right. Through OK the dialog is closed, and the link with the text established.

**The text was not yet defined** The desired group is to be marked in the text group display. Subsequently a new text can be added through the  (new text) button. A new text group, too, can be added through the  (new group) button. The link with the new text is, again, established through marking the text, and then closing the dialog through the OK button.

#### Notes:

- Within the text selection dialog the texts are only displayed in the developer language. A language change is made through the project explorer in the "Project configuration→Global" node on the "Languages" index card.
- The text selection dialog contains two search functions. In the left input field at the top margin of the dialog a part of a complete text designator (i.e. the full text group path together with the text name itself) can be specified. During the input the dialog already navigates to the first appropriate hit. Further navigation to the next hit is triggered through the  button ("search further within the full names").
- Beside searching for text names searching for text contents is also possible. For this a search item is to be specified in the "search text in column..." input field. Further navigation to the next hit is triggered through the  button ("search further in column...").

## 6.1.2 Functions



The following description contains examples for VisiTinNET Professional. In VisiWinNET SMART these examples cannot be carried out because of the missing programming environment.

### Reading texts at any time

Reading a text is carried out with 'GetText'

```
string sMsg = VisiWinNET.LanguageSwitching.Localization.GetText("Messages.AppClose");
```

In the example shown the text with the name "AppClose" from the "Ustertexts.Messages" text group is read to the sMsg variable. In VisiWinNET Compact an "@" character must be put in front of the text name. In this case the above example would look like this:

```
string sMsg = VisiWinNET.LanguageSwitching.Localization.GetText("@Messages.AppClose");
```

### Reading texts in the event of changes

Changing a text at runtime can basically have two reasons:

- Change of the language currently active in the project
- Change of a process variable value that was included in the text.

The 'LocalizedText' object contains the 'TextChanged' event in which the above mentioned reasons report a text change.

```
VisiWinNET.LanguageSwitching.LocalizedText defaultText= null ;
private void InitLocalizedTexts ()
{
    defaultText = new VisiWinNET.LanguageSwitching.LocalizedText ();
    defaultText.TextGroup = "WindowsControls";
    defaultText.TextChange +=
        new VisiWinNET.LanguageSwitching.TextChangeHandler(defaultText_TextChange);
    defaultText.Text = "TextBoxDefaultName";
}
void defaultText_TextChange(object sender,
    VisiWinNET.LanguageSwitching.TextChangeEventArgs e)
{
    textBoxName.Text = defaultText.DisplayText;
}
```

In the example shown above the "defaultText" object is created and initialized so that it accesses the "TextBoxDefaultName" text from the "WindowsControls" user text group.

If for example the language changes the appropriate text in the "defaultTextChange" event is written to the TextBox control element with the name "textBoxName".