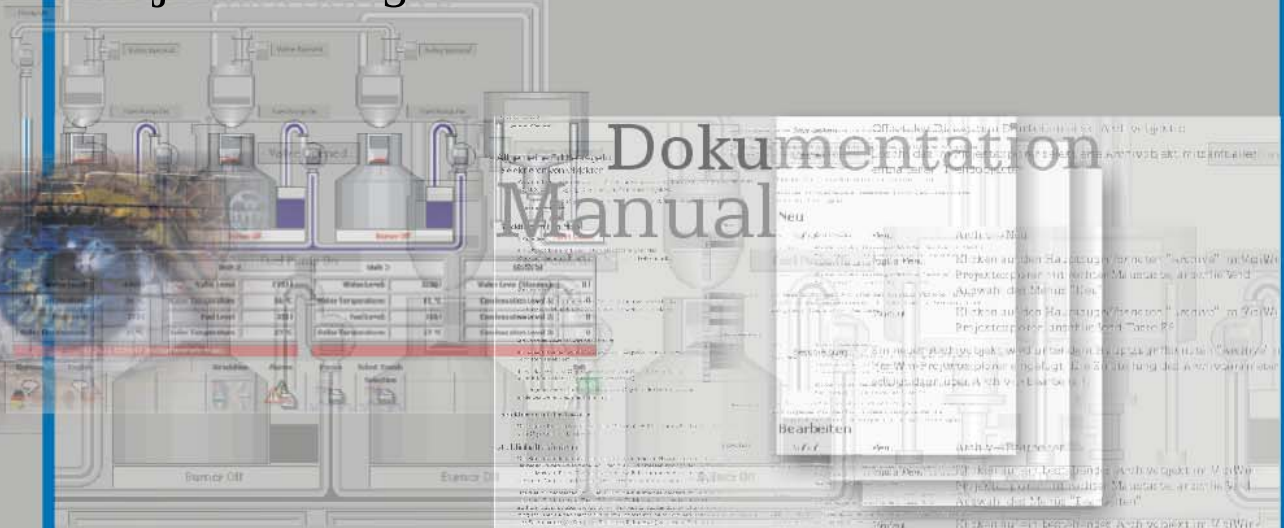


# VisiWinNET 2005

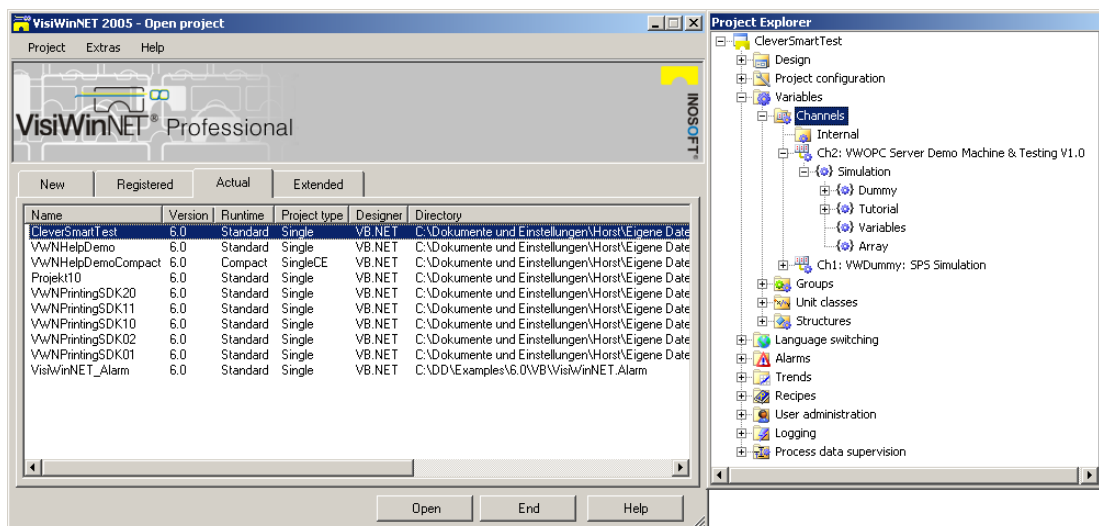
## Project management



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

# VisiWinNET 2005

## Project management



VisiWinNET Project manager and Project explorer







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 Project Management</b> .....	<b>2</b>
2.1 VisiWinNET Professional.....	4
2.2 VisiWinNET Smart.....	6
<b>3 Project manager</b> .....	<b>7</b>
3.1 Start of Project Manager.....	7
3.2 Project Manager Components.....	8
3.2.1 Index card "New".....	9
3.2.2 Index card "Registered".....	10
3.2.3 Index card "Actual".....	11
3.2.4 Index card "Extended".....	11
3.2.5 Menu "Project".....	12
3.2.6 Menu "Extras".....	13
3.2.7 Menu "Help".....	13
<b>4 Project types and settings</b> .....	<b>14</b>
4.1 Project types.....	14
4.1.1 CE-Client.....	15
4.1.2 CE Single.....	15
4.1.3 CE Single (ARM).....	16
4.1.4 Client.....	16
4.1.5 eXP Single.....	17
4.1.6 Group.....	18
4.1.7 Mobile 5.0 Client.....	19
4.1.8 Pocket PC Client.....	19
4.1.9 Server.....	20
4.1.10 Server/Client.....	20
4.1.11 Single.....	21
4.1.12 Web Site.....	21
4.2 Project settings.....	22
<b>5 VisiWinNET Package</b> .....	<b>26</b>
<b>6 Project explorer</b> .....	<b>27</b>
6.1.1 VisiWin Project Explorer operating.....	28
6.1.2 Special context menus.....	30
6.1.3 Template administration.....	31
<b>7 Project configuration</b> .....	<b>39</b>
7.1.1 Client configuration.....	39
7.1.2 Client events.....	40
7.1.3 Common project properties.....	40
7.1.4 Compatibility.....	41
7.1.5 Computer names.....	43
7.1.6 Connection Server project.....	44
7.1.7 Extended Client Configuration.....	45
7.1.8 Extended project properties.....	46
7.1.9 Language settings.....	46
7.1.10 Licence information.....	46
7.1.11 Runtime behaviour.....	47
7.1.12 Runtime configuration.....	48
7.1.13 Server project.....	48
7.1.14 System selection.....	49
<b>8 Project converters</b> .....	<b>50</b>
8.1 Conversion 5.0 to higher versions.....	51
8.2 Conversion 5.x to 6.0.....	51
8.3 Conversion VisiWinStudio to VisiWinNET.....	51

# 1 Preamble

## **About this manual**

This manual contains specific information on the VisiWinNET project management, among others the explanation of the individual components in the development surface.

## **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 Project Management

VisiWinNET as a visualization system solution contains a number of concepts that differ in complexity and objective and with which the projector is in a position to create his individual solutions. With creating these concepts and with the choice of the technologies used great importance was attached to reusability and compatibility. VisiWinNET is a tool with which special individual solutions can be created based on widespread programming knowledge but which also and in particular supports and eases the daily use of these individual solutions as a simple click tool.

The keyword is "Modularization", and does not refer to the created visualization projects alone but also to the availability and adaptability of different development levels that mirror the experience and requirements of the visualization business.

**VisiWinNET Professional**      The high-end tool for the visualization professional. The integration with Microsoft's Visual Studio .NET opens virtually boundless possibilities.

**VisiWinNET Smart**      The simple, slim "Click Tool" for the visualization beginner.

Both development environments create visualization applications that differ in an important detail:

- Screen templates created with VisiWinNET Professional are compiled into the EXE file. Applications created this way based on compiled code run with optimum performance. Changes are, however, only available after re-compilation.
- VisiWinNET Smart is also based on a compiled basic project framework allowing to start the application. Here, however, screen templates are described externally as Smart forms in external XML files. Changing individual template content is, therefore, possible without re-compilation.

It has to be noted that the use of both development environments can absolutely be linked. A project created with "VisiWinNET Professional", for example, can be opened with "VisiWinNET Smart". In the process the screen pages developed in the "Professional" version are provided in the compiled EXE as a static part that can no longer be changed. Application parts added through VisiWinNET Smart can still be changed with the customer without Visual Studio.

Common with both development environments is the access to edit the project databank. The definitions entered here during development time describe at runtime the data exchange with the controls as well as visualization-specific special functions such as alarms, trend recordings etc.

With both development environments the screen template design is done through graphic designers following the "position and parameterize" concept. Objects for different applications (e.g. variable in/output, diagrams, images etc.) are positioned in the templates, and parameterized via a properties window, that way, for example, being linked with process values, curve definitions, texts or image files. In this respect, however, the "Professional" version contains crucial advantages over Smart. Within the Visual Studio .NET development environment the whole extent of the project-oriented VB.NET and C# programming languages is provided, allowing via source code to program events, link databanks, or lodge new control elements (screen objects). All these functions are compiled which means that they run with optimum performance. They can, however, also be re-used. This way, for example, VisiWinNET Smart can integrate the control elements created in "Professional".

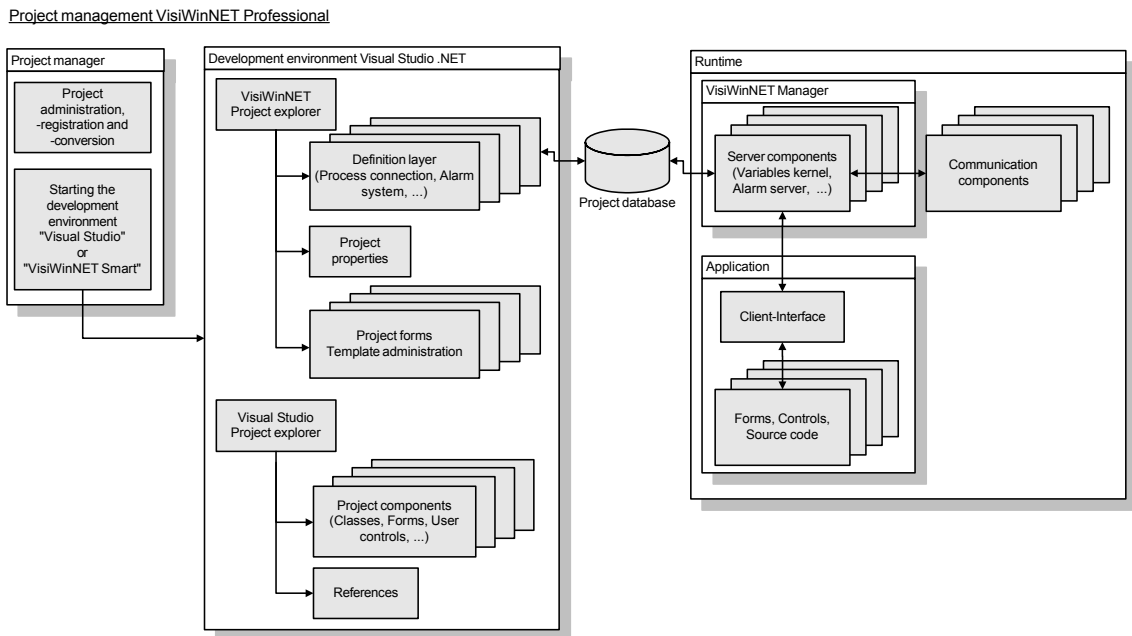
Again, these are used following the "position and parameterize" concept. Screen displays containing e.g. a databank link are, as described above, compiled as static application parts, and are also available in the Smart application.

## 2.1 VisiWinNET Professional

VisiWinNET Professional is a visualization system based on the "Visual Studio.NET" development environment. The visualization is virtually embedded in the natural development environment of the new .NET technology, however, it is also forced to live with it in a kind of symbiosis. A symbiosis offers mutual advantages:

- "Visual Studio.NET" as the official development tool of the .NET technology is enhanced by the ability to communicate with process levels, and to access visualization-specific extra functions in a comfortable way.
- VisiWinNET not only gains the framework for the visualization surface from Visual Studio.NET. In addition the projector and with this the visualization are provided with all current and future enhancements.

The graph below provides an overview of this symbiosis idea:



### Project manager

The project manager is the starting point of a development with VisiWinNET Professional. It serves to administrate all projects of the VisiWin family on the development computer. Via the project manager new projects are created, existing projects opened, and older projects converted.

### Development environment

In VisualStudio.NET the components are provided by VisiWinNET. Here the content of the VisiWinNET project database is determined via the definition level. Global settings for the project are made through the project properties. These specifications serve to determine the visualization-typical performance of the application: here it is determined which process values are required in the visualization, and which special functions they occupy. Central access to all definitions is possible in the project explorer.

In addition the project explorer offers access to the project forms. Here, new forms from the template administration can be added.

The template administration implemented in VisiWinNET allows lodging finished forms that are already provided with control elements and that are linked with definitions from a project database. If such a form is inserted in another project the definitions linked with the form are also copied into the project.

The graphic design and the functional content of the application are projected via the VisualStudio.NET functions. In this process VisiWinNET provides special control elements that are used for the surface layout, and create a connection with the server components at runtime. The client interface is tied in via the references.

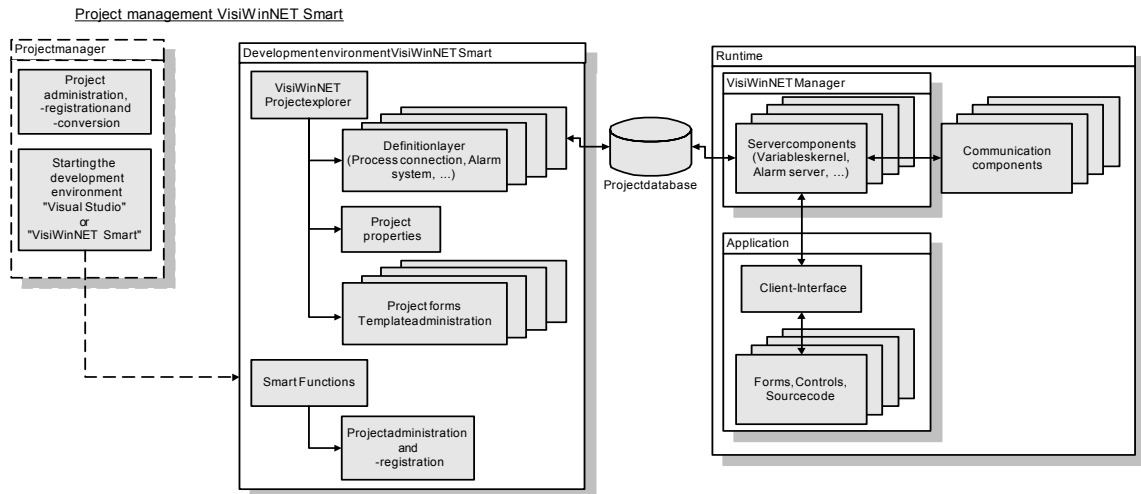
## **Runtime**

The runtime, i.e. the program to be run, is usually started via the application. When loading the first form the VisiWinNET Manager and with it all server components are started through the client interface. The server components read out the definitions from the project database, and start their functions accordingly. During the initialization phase data access is also effected via the communication components.

This mode of integrating VisiWinNET into VisualStudio.NET offers the advantage to the developer of no technological bridges having to be crossed in an application. Development is seamless. There are, however, a few subtleties to be observed with the approach in certain operational procedures: some Visual Studio functions relating to the project administration are replaced or enhanced by equivalent functions in VisiWinNET.

## 2.2 VisiWinNET Smart

VisiWinNET Smart as a parameterizable click tool contains access to the project databank and the screen templates of the visualizations.



### Project Manager

The Project Manager administrates all projects of the VisiWin family on the development computer. Through the Project Manager new Smart projects can be created or existing projects opened. The use of the Project manager is optional. Alternatively, the Smart development environment can be directly opened, and a project created or loaded from there.

### Development Environment

In the Smart development environment the components are provided by VisiWinNET. Here the contents of the VisiWinNET project databank is determined through the definition level. Global settings for the project are made through the project properties. These predefinitions are to determine the visualization-typical performance of the application: here it is determined which process values are required in the visualization, and which special functions they occupy. Central access to all definitions is possible in the Project Explorer.

In addition the Project Explorer provides access to the Smart project forms. Here the graphic design and the functional contents of the application are projected. In the process VisiWinNET provides special control elements that are used to design the surface, and at runtime establish a connection with the server components.

### Runtime

As a rule the runtime, i.e. the executable program, is started through the application. When loading the first form the VisiWinNET Manager and with it all server components are started via the client interface. The server components read the definitions from the project databank, and accordingly start their functions. During the initializing phase the link to the process through the communication components is established, too.

### 3 Project manager

The VisiWin Project Manager provides for the management of existing and new projects.

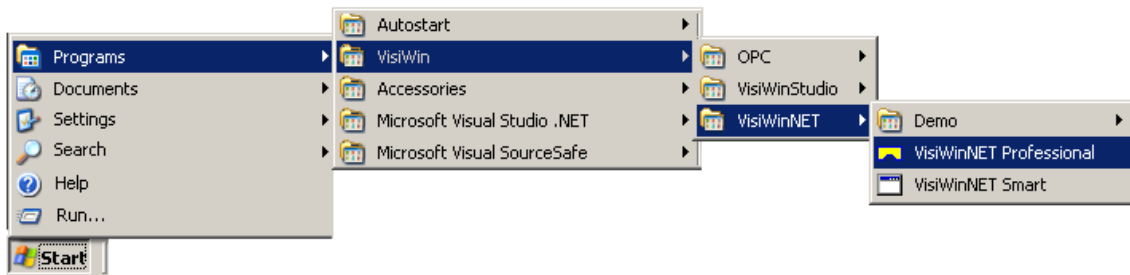
**Manages all existing VisiWin-Projects on the development data processor.** The project manager shows which VisiWin projects are available on the development computer.

**Starts the development environment.** After accessing the desired project the appropriate development environment (depending on either "VisiWinNET Smart" or "Visual Studio .NET" project type) is started and loaded through the Project Manager.

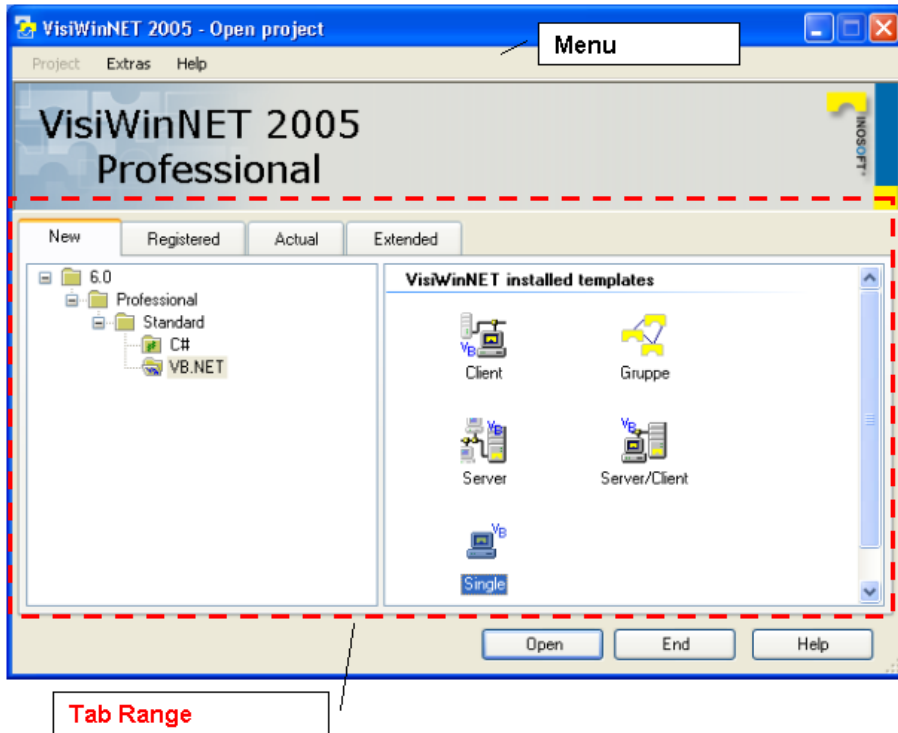
**Creates new projects with all necessary requested components.** The project manager offers the available project types. After accessing a project type the basic properties (project name/path indication) of the new project can be set in the following dialog. After confirmation all project files required for the framework of the new project are created automatically.

#### 3.1 Start of Project Manager

After a successful VisiWinNET-Installation the project manager is started over the Windows-Startmenu (Start→Programs→VisiWin→VisiWinNET→VisiWinNET→VisiWinNET Professional):



### 3.2 Project Manager Components



The project manager includes essentially two operator groups:

- Tab range** Over the tab range the action, that should be processed with the VisiWin Project, is selected:
  - Index card "New"** Provides for creation of a new object. A symbol list is located on the index card, which enables the project type selection.
  - Index card "Registered"** Opens the project. The list indicates all registered projects on the data processor.
  - Index card "Actual"** Provides for the fast open of one of the last adapted projects.
  - Index card "Extended"** Provides for registration of the project, which was imported from another data processor.
- Menu** The menu contains further functions of the Project Manager.
  - Menu "Project"** Contains functions for project administration
  - Menu "Extras"** Settings for used directories
  - Menu "Help"** Introduction to the online help, and access to the information dialog

### 3.2.1 Index card "New"

When creating a new project the developer must make decisions regarding the operating system, the hardware, the required network functionality, and the developing system.

Further information on the available project types can be found in the equally named "Project types" chapter (below). The settings required when creating a new project are described in the "Project settings" chapter (below).

#### 3.2.1.1 Project types

Below a collection of the VisiWinNET project types and their properties:

##### For VisiWinNET Professional (Microsoft Visual Studio development environment)

Project type	Runtime	Description
<b>CE Single</b>	Compact	Windows CE single-workstation application for x86 processors
<b>CE Single (ARM)</b>	Compact	Windows CE single-workstation application for ARM processors
<b>EXP Single</b>	Compact	Embedded Windows XP single-workstation application
<b>Single</b>	Standard	Windows XP/Vista single-workstation application
<b>Group</b>	Standard	Free-to-assemble group of multiple objects that are loaded together in the development environment. Following selection of this project type an index card to assemble the individual projects is offered (see chap. "Project Settings" (4.2.1.3)).
<b>Server</b>	Standard	Project acting as a data pool for an application distributed in the network. This type contains a VisiWinNET project databank whose contents control the runtime performance. It does, however, not contain a visualization surface. This is implemented through a client project. The data exchange between client and server project can be made locally or through a network.
<b>Client</b>	Standard	Project with visualization surface that at the start connects with a server project.  Following selection of this project type an index card to assemble the individual projects is offered (see chap. "Project Settings" (4.2.1.3)).
<b>Server/Client</b>	Standard	Group with a server and a client project. In the creation process both projects are named after the specified project name.

All project types are available for the VB.NET and C# programming languages.

### For Smart (VisiWinNET Smart development environment)

Project type	Runtime	Description
<b>Single CE</b>	Compact	Windows CE single-workstation application for x86 processors
<b>Single CE (ARM)</b>	Compact	Windows CE single-workstation application for ARM processors
<b>Single</b>	Compact	Embedded Windows XP single-workstation application
<b>Single</b>	Standard	Windows XP/Vista single-workstation application

### 3.2.2 Index card "Registered"

The index card "Registered" serves to open a registered project. All registered projects are listed in alphabetic order. With the index card "Actual" it is possible to open one of the last 10 adapted projects in a faster way. However, not all of the registered projects are listed there. This applies especially, when:

- Since project opening, another ten projects were opened, and likewise the project was dropped out of the current list.
- Since the last project opening a new VisiWin-Installation was processed.

The list includes following information:

<b>Name</b>	Name, under which the new project was created.
<b>Version</b>	Indicates the VisiWinNET version the project is opened with. Multiple VisiWinNET versions can be installed parallelly. The project manager administrates the projects of all versions, selecting the appropriate design tool in the opening process.
<b>Product type</b>	Specifies the VisiWinNET development frame under which the project was created (Professional or Smart)
<b>Runtime</b>	Specifies the runtime component: <ul style="list-style-type: none"> <li>• Standard: runtime for standard PCs</li> <li>• Compact: runtime for Windows CE or Embedded Windows XP computers.</li> </ul>
<b>Project type</b>	Indicates the project type. The project type is determined when a new project is created.
<b>Designer</b>	Indicates the programming language used to create the project.
<b>Directory</b>	Directory, in which the process database and further project files are located.



All newly created VisiWinNET projects are registered.

Projects, created on another developing data processor must be registered first on the personal computer. The index card "Extended" supports the developer with the project registration of an imported project.

Through the "Open"-button the project development environment is started, which name was selected in the list.

The project manager is closed by the "Cancel"-button.

Through the "Help"-button the online help in the project manager is started.



The requested project may also be opened with double click on the project name.

### 3.2.3 Index card "Actual"

The index card "Actual" serves to open one of the ten last adapted projects.

With the "Open"-soft key the development environment of the project, selected in the list, is started.

The "Cancel"-soft key closes the project manager.

With the "Help"-soft key the project manager online help is started.




The requested project may be opened with double click on the list row as well.

### 3.2.4 Index card "Extended"

The index card "Extended" includes two functions:

#### Registration of a VisiWinNET-Project.

Via the  button a dialog is opened allowing selection of a VisiWinNET project file (\*.vwn file ending). Following the selection of a VWN file the project is automatically registered. This function makes sense if a project was copied from another development computer. Here, a project group (\*.sln file ending) can also be searched for, and registered.

### 3.2.5 Menu "Project"

The "Project" menu contains the following entries:

<b>Convert into new VisiWinNET version</b>	Converts older version projects to the currently installed version.
<b>Save as</b>	Saves the project under a different name. Where appropriate the "Select additional files" dialog will ask whether files in the project directory that are not referenced out of the project are to be copied into the newly created project directory.
<b>Save as template</b>	Saves the project as a template. This projects is then available as a template on the "New Project" index card.



The project templates administration allows the user to call off own application shells. Project templates may contain readily projected forms as well as definitions.

Allocation to a category in the tree display on the "New" index card is made by the project to be saved. All information (version, runtime and programming language) used for the allocation to a category is already contained in the project, and cannot be altered any more at this stage.

Where appropriate the question will be asked in the "Select additional files" dialog whether files in the project directory that are not accessed by the project itself are to be copied into the newly created project directory. In particular this refers files such as HTML, XML, RTF or databanks that are used by the application in control elements or via source code but that have nothing to do with the typical buildup of a VisiWinNET application.

The main directory for own project templates is determined in the "Options" dialog of the Project Manager.

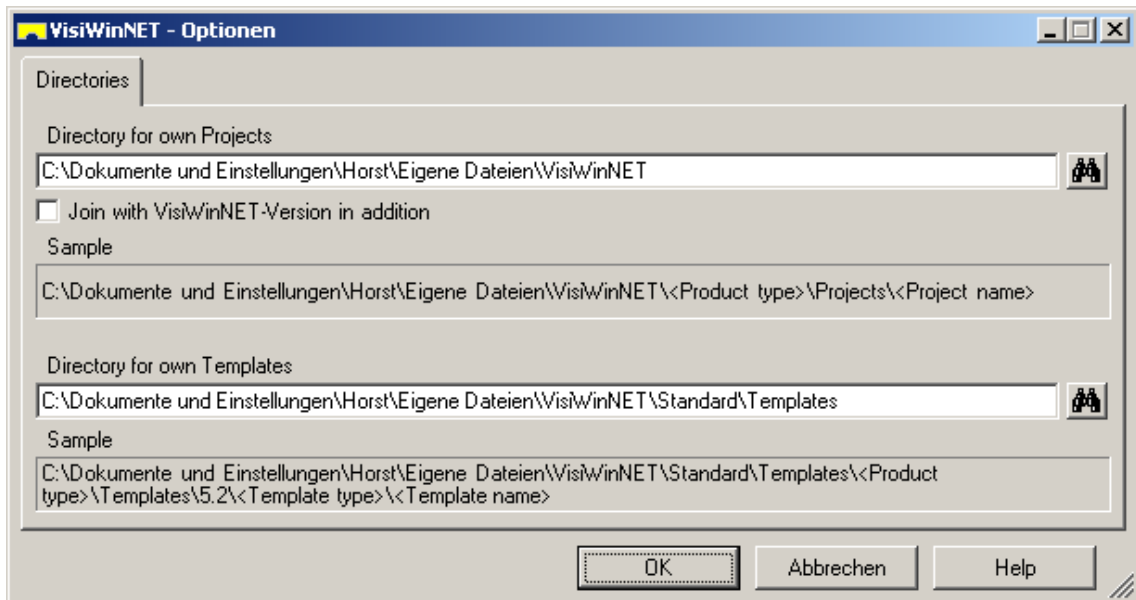
The "Save as a template" dialog contains the following elements:

<b>Name</b>	Under this name the template if offered as a project type on the "New" index card in the Project Manager.
<b>Comment</b>	Here a describing text is expected that is e.g. displayed on the "New" index card as a tool tip on the project type.
<b>Symbol</b>	Determines the symbol representing the template on the "New" index card.

<b>Deregister</b>	Deregisters the project. The project directory remains. The project can, however, no longer be opened through the Project Manager. A re-registration is possible through the "Enhanced" index card.
<b>Remove from hard disc</b>	Carries out the deregistration, and deletes the project directory.

### 3.2.6 Menu "Extras"

The "Options" entry in the "Extras" menu opens a dialog to adjust the most important directories used by VisiWinNET:



#### Directory for own projects

The entry determines the main directory for new VisiWinNET projects.

This path indication is used in the "Project path" field of the "Create new project" dialog (chap. 3.2.1).

#### Add extra VisiWinNET versions

If this option is activated the version is added to the standard path of the new project.

#### Directory for own templates

The VisiWin Project Explorer offers the option of adding further templates to the project. Template files and definition databanks that are saved under the selected directory can be added to the project as components.

### 3.2.7 Menu "Help"

The "Help" menu contains the following entries:

#### Help

The "Help:Contents" button accesses the complete online help for VisiWinNET.

#### About VisiWinNET...

Through this button the information dialog of the Project Manager can be accessed.



With the Hotline Support you are in most cases asked for the version of the installed component. This version number is displayed in the information dialog.

## 4 Project types and settings

### 4.1 Project types

When creating a new project the developer must make decisions regarding the operating system, the hardware, the required network functionality, and the developing system.

The project types at a glance:

Type	Operation system of the visualization surface	Connection to PLC
<b>CE-Client</b>	Windows CE	Server project via Network
<b>CE Single</b>	Windows CE (x86 CPU)	local
<b>CE Single (ARM)</b>	Windows CE (ARM, XScale CPU)	local
<b>Client</b>	Windows XP/Vista	Server project via Network
<b>eXP Single</b>	Embedded XP	local
<b>Gruppe</b>	any combination	any combination
<b>Mobile 5.0 Client</b>	PDA Windows CE 5.0	Server project via Network
<b>Pocket PC Client</b>	PDA Windows CE 4.2 SP2	Server project via Network
<b>Server</b>	-	local
<b>Server/Client</b>	Windows XP/Vista	Server project via Network
<b>Single</b>	Windows XP/Vista	local
<b>Web Site</b>	Windows XP/Vista, CE (in web browser)	Server project via IIS

**4.1.1 CE-Client**

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	Windows CE
<b>Access in Project manager</b>	Professional→Standard→C#→CE Client
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	Connects through a network with a VisiWinNET project of the "Server" type.
<b>Project settings</b>	Common Parameters Server Project Device Settings
<b>Remarks</b>	On the definitions level the project only contains the localization. Here the texts of the applications are projected on the client part. Other definitions [variables, alarms, ...] are defined in the server project.  Client and server projects can at development time within a project group be loaded together in the development environment.

**4.1.2 CE Single**

<b>Development environment</b>	Professional (Visual Studio), Smart
<b>Operating system</b>	Windows CE
<b>Access in Project manager</b>	Professional→Compact→C#→CE Single Professional→Compact→VB.NET→CE Single Smart→Compact→CE Single Smart→Compact→CE Single
<b>Developer Language</b>	C#, VB.NET (für Professional) No programming in Smart
<b>Network integration</b>	Single workstation application
<b>Project settings</b>	Common Parameters Device Settings
<b>Remarks</b>	CE-Projekt for x86 CPU

**4.1.3 CE Single (ARM)**

<b>Development environment</b>	Professional (Visual Studio), Smart
<b>Operating system</b>	Windows CE
<b>Access in Project manager</b>	Professional→Compact→C#→CE Single (ARM) Professional→Compact→VB.NET→CE Single (ARM) Smart→Compact→CE Single (ARM) Smart→Compact→CE Single (ARM)
<b>Developer Language</b>	C#, VB.NET (für Professional) No programming in Smart
<b>Network integration</b>	Single workstation application
<b>Project settings</b>	Common Parameters Device Settings
<b>Remarks</b>	CE-Project for ARM, XScale CPU

**4.1.4 Client**

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	Windows XP/Vista
<b>Access in Project manager</b>	Professional→Standard→C#→Client Professional→Standard→VB.NET→Client
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	Connects through a network with a VisiWinNET project of the "Server" type.
<b>Project settings</b>	Common Parameters Server Project Device Settings
<b>Remarks</b>	On the definitions level the project only contains the localization. Here the texts of the applications are projected on the client part. Other definitions [variables, alarms, ...] are defined in the server project.  Client and server projects can at development time within a project group be loaded together in the development environment.

#### 4.1.5 eXP Single

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	Windows Embedded XP
<b>Access in Project manager</b>	Professional→Compact→C#→eXP Single Professional→Compact→VB.NET→eXP Single Smart→Compact→eXP Single
<b>Developer Language</b>	C#, VB.NET (for Professional) No programming in Smart
<b>Network integration</b>	Single workstation system
<b>Project settings</b>	Common Parameters Device Settings
<b>Remarks</b>	-

#### 4.1.6 Group

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	XP/Vista, CE, Embedded XP
<b>Access in Project manager</b>	Professional→Standard→C#→Group Professional→Standard→VB.NET→Group
<b>Developer Language</b>	All
<b>Network integration</b>	No specifications
<b>Project settings</b>	Common Parameters Projects
<b>Remarks</b>	<p>A project group loads several projects into the development environment. Creating a project group makes sense if the developer wants to edit or synchronize data from multiple projects within a development environment instance.</p> <p>When debugging the project that within the project group was determined as the start project is started.</p> <p>To determine the start project within a project group the following steps are to be carried out:</p> <ul style="list-style-type: none"> <li>• Open the Visual Studio project folder explorer in the development environment.</li> <li>• Mark the selected start project node.</li> <li>• Select the "Set as start project" context menu entry.</li> </ul> <p>The node of the start project is then displayed bold in the project folder explorer.</p> <p>Projects that are created as type libraries (in VisiWinNET these are in particular the Server projects) cannot be determined as start projects.</p>



**4.1.7 Mobile 5.0 Client**

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	CE 5.0 für PDA-Geräte
<b>Access in Project manager</b>	Professional→Standard→C#→Mobile 5.0 Client Professional→Standard→VB.NET→Mobile 5.0 Client
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	Connects through a network with a VisiWinNET project of the "Server" type.
<b>Project settings</b>	Common Parameters Server Project Device Settings
<b>Remarks</b>	To develop this project type the "Windows Mobile 5.0 SDK for Pocket PC" is required.

**4.1.8 Pocket PC Client**

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	CE 4.2 SP1 for PDA devices
<b>Access in Project manager</b>	Professional→Standard→C#→Pocket PC Client Professional→Standard→VB.NET→Pocket PC Client
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	Connects through a network with a VisiWinNET project of the "Server" type.
<b>Project settings</b>	Common Parameters Server Project Device Settings
<b>Remarks</b>	-

**4.1.9 Server**

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	Windows XP/Vista
<b>Access in Project manager</b>	Professional→Standard→C#→Server Professional→Standard→VB.NET→Server
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	For data exchange client applications connect with a project of this type.
<b>Project settings</b>	Common Parameters
<b>Remarks</b>	In the development environment this project type contains the complete definitions level [variables, alarms, texts, ...] but no own design components.

**4.1.10 Server/Client**

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	Windows XP/Vista
<b>Access in Project manager</b>	Professional→Standard→C#→Server/Client Professional→Standard→VB.NET→Server/Client
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	At runtime the client part project connects with the server part project.
<b>Project settings</b>	Common Parameters
<b>Remarks</b>	A project of the "Server/Client" type is a project group in which a server project serves as a data interface with the controls. The second part project, i.e. the client project is configured so that it contacts the server when started.

**4.1.11 Single**

<b>Development environment</b>	Professional (Visual Studio), Smart
<b>Operating system</b>	Windows XP/Vista
<b>Access in Project manager</b>	Professional→Standard→C#→Single Professional→Standard→VB.NET→Single Smart→Compact→Single Smart→Standard→Single
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	Single workstation application
<b>Project settings</b>	Common Parameters Device Settings
<b>Remarks</b>	-

**4.1.12 Web Site**

<b>Development environment</b>	Professional (Visual Studio)
<b>Operating system</b>	Windows XP/Vista
<b>Access in Project manager</b>	Web→Standard→C#→Web Site
<b>Developer Language</b>	C#, VB.NET
<b>Network integration</b>	Fetching of visualization surfaces in the web browser through an internet information server
<b>Project settings</b>	Common Parameters
<b>Remarks</b>	



Please request our sales department


## 4.2 Project settings

Depending on the selected project type the following index cards are displayed in the "Create new project" dialog:

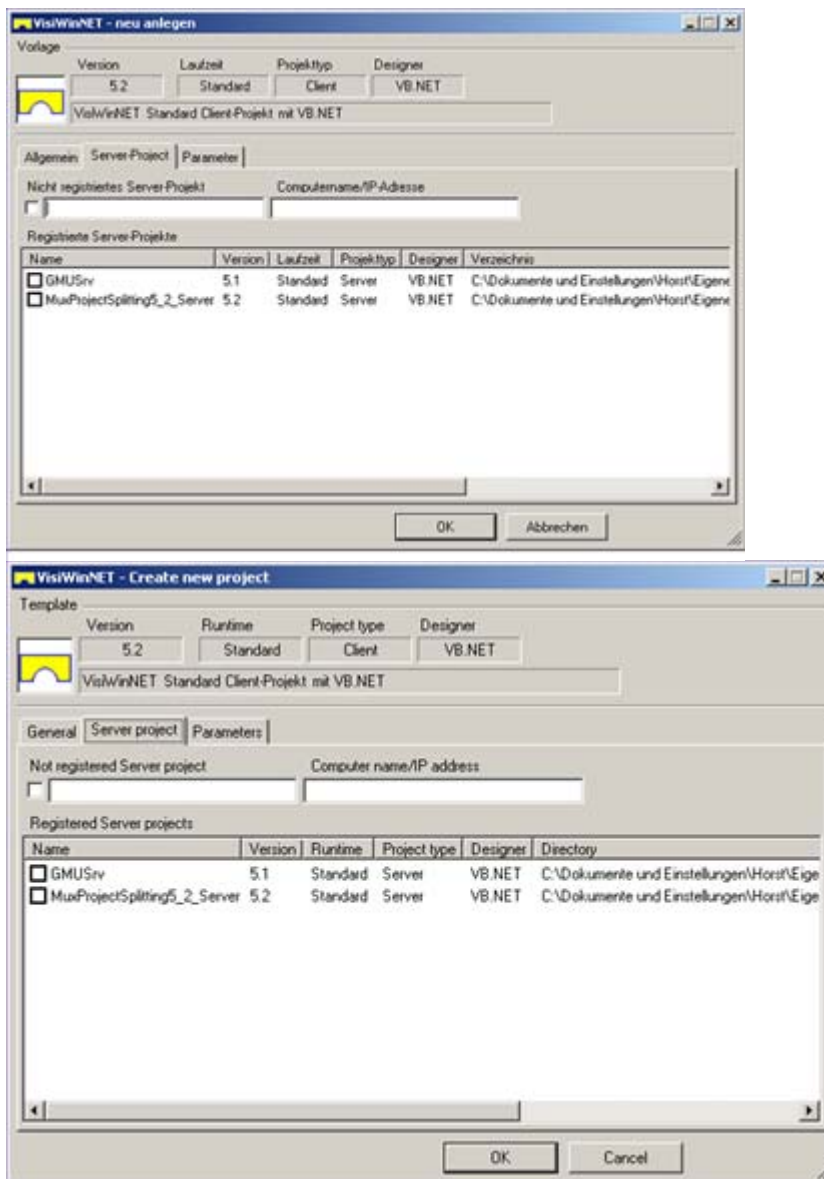
<b>Common</b>	Setting of project name and path
<b>Parameters</b>	Optional extra information for project administration
<b>Server Project</b>	Selection of a server project for the client project types
<b>Projects</b>	Selection of the part projects of a project group
<b>Device Settings</b>	Selection of the screen resolution

### 4.2.1.1 Common Settings

The following properties are common to all project types:

Tab	Property	Description
<b>General</b>	<b>Project name</b>	Determines the name of the new project.  <div style="border-left: 1px solid black; padding-left: 5px;">                     On a development computer two projects of the same product version can never be registered with the same name. Therefore, entering an already existing project name is reported by the dialog. It is, however, possible to choose a project name that is already being used, specifying a new directory for the project. Then two projects under the same name exist on the computer. However, only the registered (i.e. the new) project can be edited and started. Through the "Extended" index card the old, no longer registered project can be re-registered.                 </div>
		
	<b>Project directory</b>	Determines the index into which the project framework is to be saved. Later in the development all further project files should also be saved in this index.
	<b>Existing projects</b>	This list shows all project existing on the computer. It is designed as a help in finding a new project name resp. a new project index..
	<b>Parameters</b>	<b>project version</b>
<b>Customer</b>		This parameter, too, is a setting to be freely allocated by the developer. If required the name of the customer can be specified here.
<b>Developer</b>		Also optional is the input of the name of the developer.
<b>Comment</b>		The comment, too, is not compulsory. It can, however, contain instructions for the developer in case of an extended version maintenance.

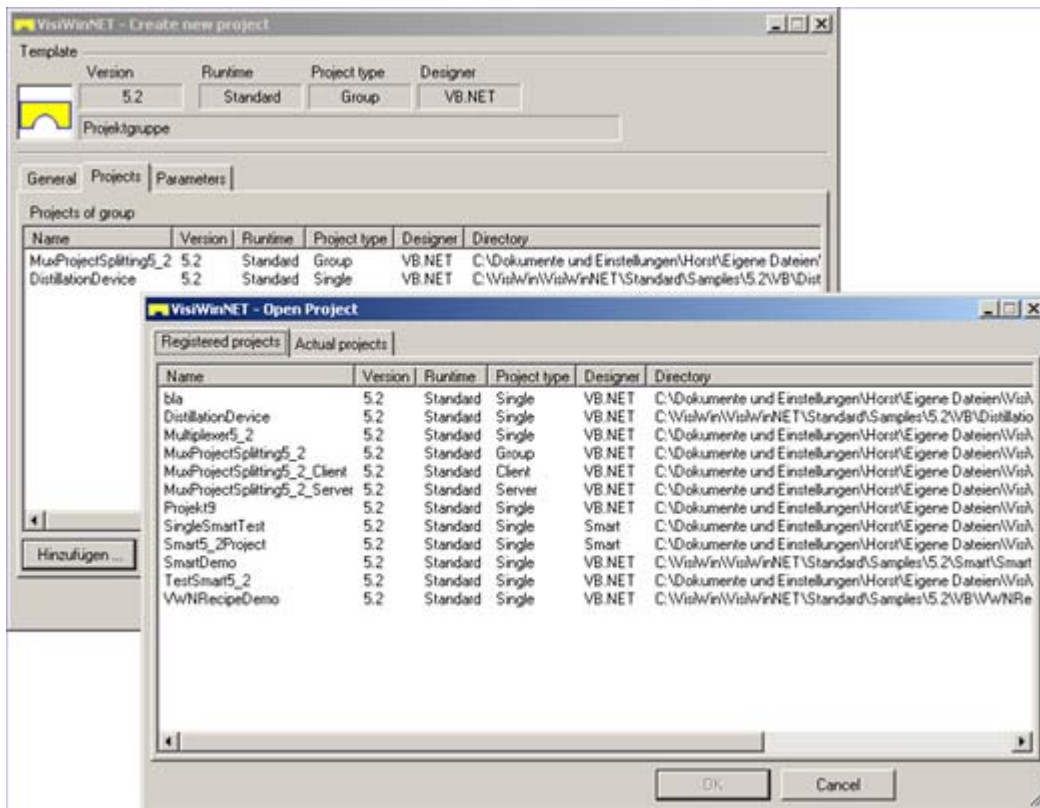
#### 4.2.1.2 Settings "Server Project"



The settings mentioned here determine with which WisiWinNET project of the "Server" type data are to be exchanged.

- The name of the server project can as a project registered on the development computer directly selected from the lower list. If the server project is not installed on the development computer the project name can also be directly entered in the appropriate input field. In that case the control box forward of the input field is to be activated.
- The Computer Name/IP Address input field determines the computer on which the server project runs or is started. The IP address or the network computer name is expected as input.

### 4.2.1.3 Settings "Projects"

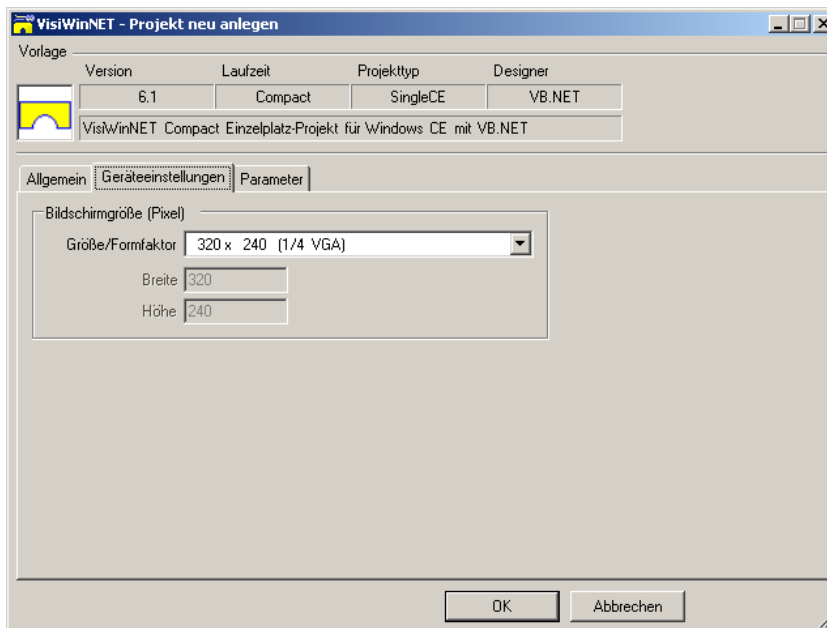


The settings mentioned here are only relevant for a "Group" type project.

They determine which VisiWinNET projects are to be commonly loaded in the development environment as a project group.

The "Add" button opens a dialog for selection of a registered project.

#### 4.2.1.4 Device Settings



The hardware settings mainly refer to the screen resolution used with the target system. The "Size/Form factor" selection list contains the most used screen resolutions. Through selecting the "other resolution" entry the "width" and "height" fields for manual input of the resolution are displayed.

## 5 VisiWinNET Package



This chapter describes the function of the VisiWinNET package in Visual Studio (VisiWinNET Professional). The component described here is not used in VisiWinNET Smart.

The integration of VisiWinNET into Visual Studio is done through the VisiWinNET installation. In the process a so-called "Package" is registered in Visual Studio. When loading a VisiWinNET project the components of the package carry out multiple tasks for the developer.

- Integration of the VisiWinNET control elements into the toolbox
- Loading of the VisiWinNET Project Explorer for access to the definitions and configuration entries of the project databank
- Loading of the VisiWinNET tool bar

The VisiWinNET tool bar is displayed in the tool bar area (underneath the main menu) of Visual Studio.

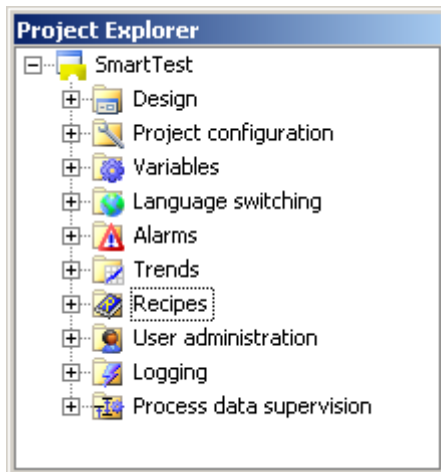


It contains the following buttons (from left to right):


<b>show VisiWinNET Project explorer</b>	The project explorer displays the project components as a tree. Further information about the project explorer functionality can be found in the equally named "Project explorer" chapter in this manual.
<b>Show About-Dialog</b>	Displays the info dialog of VisiWinNET Professional. Here information on the package and installation versions is to be found.
<b>Show Options</b>	Opens the dialog for editing the options to the editors of VisiWinNET.
<b>Start NetViewer</b>	Starts the NetViewer Client as a diagnosis program in case support is required. Further information is available in the Internet on the INOSOFT homepage ( <a href="http://inosoft.com/Support/Online-Support.asp">http://inosoft.com/Support/Online-Support.asp</a> ).

## 6 Project explorer

In the development environment the VisiWinNET project explorer administrates all project components.



If after the start of the development environment the Project Explorer is not displayed the window can be activated as follows:

- In VisiWinNET Professional (Visual Studio) through the  button in the AddIn button bar.
- In VisiWinNET Smart through the Display→Project explorer menu.

The VisiWinNET project explorer provides access to the definition level of a VisiWinNET project. Subsequent to the start of the development environment every node in the tree of the project explorer represents a component or a component group.

In addition the VisiWinNET Project Explorer lists the forms of the project under the "Design" node.

The "Project configuration" area contains project-wide settings.

### VisiWinNET Systems

VisiWinNET works modularly which means that individual systems can be selected as required, and combined to an application. Selection of the active systems is made in the project configuration (highlight the "Project Configuration→Systems" Project Explorer node, then make the settings on the VisiWinNET properties page.

Below the list of all systems with references to further information.

System	Description/Manual
<b>Kernel</b>	Data access
<b>Alarm</b>	Alarm system
<b>Language</b>	VisiWinNET Localization
<b>Recipe</b>	Recipe system

<b>Trend</b>	Archive system
<b>UserManager</b>	User administration
<b>Logbook</b>	Logging
<b>Redundancy</b>	Data access
<b>Supervision</b>	Process data supervision

### 6.1.1 VisiWin Project Explorer operating



The VisiWin-Project Explorer functions may be divided in two groups:


- general functions for navigation im Projektextplorer
- Component functions in the project explorer


#### 6.1.1.1 Navigation


The access to the component functions is generally controlled by the selection of the appropriate node in the tree view. A selected node is presented with blue background in the node text. Navigation is processed...



**...with the mouse** One mouse-click selects a node in the tree view

**...with the keyboard** With the buttons  and  the cursor goes from the selected node to the node underneath or above.


A node with a -Sign includes further subnodes. For access to the subnodes, the node may be expanded. Expanding is processed ...


**...with the mouse** Mouse-click on the -Sign expands the appropriate node in the tree view.



**...with the keyboard** With key  the current selected node is expanded.

After this process, the -sign changes to following -sign.

If many nodes are expanded already, to get a better view, shut nodes again. This is processed ...

**...with the mouse** Mouse-click on the -sign shuts the appropriate node in the tree view.

**...with the keyboard** With key  the current selected node is shut.

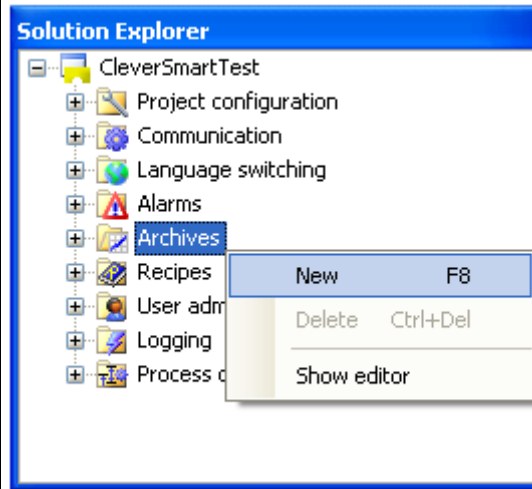
After this process the -sign changes to following: -sign.

### 6.1.1.2 Definition functions

Each component contains specific functions. These functions are accessible through the object menu of the appropriate node in the project explorer. The object menu is opened with right click on the appropriate node.

Many specific component functions are accessible through a shortcut as well. Shortcut ability for functions is indicated in the object menu on the right of the function name

Example:



Find the shortcut for creation of a new archive object in key **F8**.  
 With actuation of this key a new archive object is inserted.

### 6.1.2 Special context menus

Certain nodes in the project explorer contain special context menus allowing access to administration functions that are not specific for the individual systems:

Node in VisiWinNET Project explorer	Menu	Function
<b>Node "Forms" / Folder node</b>	Update display	In certain situations an adjustment of the collection of the forms existent in the project in the VisiWinNET project explorer is necessary. If for example a form has been added via the Visual Studio functions, and that form is not displayed in the VisiWinNET project explorer, a rebuild of the form collection can be forced through the "Update form collection" menu item.
	Add folder	Creates a new folder. The indicated folder is created as an index on the hard disk under the project index.  For larger projects with many forms the developer thus gains an additional structuring level.
	Add form	Creates a new form via the VisiWinNET form selection dialog. The form selection dialog supports different form types in dependence on the content of the template indices. Some forms can be added through the Template administration.
<b>Form node</b>	Display designer	Opens the design view of the selected form.
	Display code	Opens the code view of the selected form.
	Rename	Opens a dialog for entering a new form name. After a new form name has been entered the form is saved under the name specified here.
	Delete	Deletes the reference to the selected form from the project. The form file is not deleted.

### 6.1.3 Template administration

This chapter describes the buildup of the VisiWinNET Template administration.

#### Why templates?

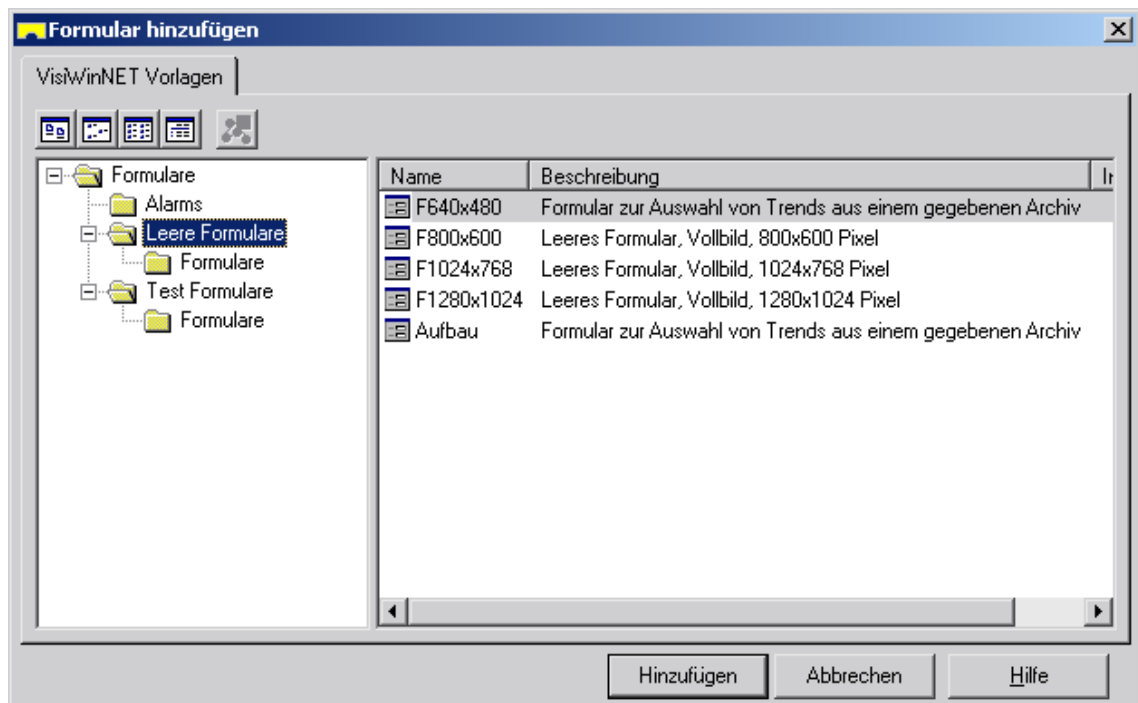
There frequently is the requirement to create multiple projects with equal application parts, matching the different machine versions to be visualized. In practice, the individual functional parts of a machine are often divided into screen templates. This makes the visualization modular.

The visualization projects are now assembled from individual forms without the necessity of intervening with design or source code. Creating a project by clicking individual forms together is basically possible in Visual Studio through the functions in the "Solution Explorer".

VisiWinNET projects, however, not only require the form files but also the definitions from the project database. A solution for the common administration of form files and definitions is offered by the template administration.

#### Use of templates

The use of templates is positively simple: Via the "Add form" entry of the context menu on the "Forms" node in the VisiWinNET project explorer a dialog for selecting a form is opened.



Here a selection can be made between existing templates. A form that has been added this way can be linked with definition data or further files in the template administration.

## Creating templates

The good news	When creating own templates the usual projection method can be maintained.
The bad news	A little more effort is, however, required. One step is the creation of an Ascii file containing the necessary linking information used to tie in the template with a project.

The collection below shows the steps for creating a template:

<b>Creation of a form that can be added to other projects as a template</b>	Creation is as usual made in the form designer. Control elements accessing definitions from the project database (such as the VarIn referring to an process variable designator) can also be linked via the appropriate property pages. This means that there are no restrictions compared to the standard project development.
<b>Creation of a project database with the appropriate definitions</b>	<p>With form development finished, and tested, the project database definitions used by the form are to be extracted.</p> <p>For this, first a new project is to be created whose project database serves as a memory for the definitions to be used. It is recommended to use the same name for the project and the form.</p> <p>Subsequently the definitions used by the form can be copied into the new project from the individual system via "Global copying/importing".</p> <p>The project database thus created is copied into the template index, and linked with the form as explained in the steps below.</p>
<b>Copying of the relevant data into the template index</b>	<p>In the project manager settings the index for own templates is determined. See chapter "Menu Extras" (3.2.6). In this index further folders are to be created subdividing the own templates into functional groups. These subfolders are later displayed as a classification level when a new template is added.</p> <p>Into the new subindex (now identifying a subject area) the files belonging to a template are to be copied:</p> <ul style="list-style-type: none"> <li>• Form files (&lt;Form name&gt;.vb resp.. &lt;Form name&gt;.cs, &lt;Form name&gt;.resx)</li> <li>• Definition project database(&lt;Project name&gt;.vwn)</li> </ul>
<b>Creation/adaption of the template information file</b>	<p>Beside the template files VisiWinNET expects a file with the name "Info.ini". Here further properties of the template files in the index are to be specified as an Ini file.</p> <p>The sections enclosed in pointed brackets indicate the information of a template from the index. The values of the section define the template properties. The following entries are permitted:</p>

Entry	Explanation
<b>Description</b>	Supplies a description for the form.
<b>AddAsTemplate</b>	0: The name of the form cannot be changed when the template is added.  1: The name of the form can be changed when the template is added. In the adding process the name is provided with an index.
<b>VWN</b>	Name of the VWN file in the same template subindex. From this project database all definitions are read out, and added to the same project the template form is added to.
<b>File&lt;n&gt;</b>	List of files that are added in addition to the selected form. Classes, modules or, again, forms can be chosen from the index as additional files.
<b>Reference&lt;s&gt;</b>	List of references that are added to the project when the template is added.
<b>Type</b>	Specification of template type:  Form: The template is a form  UserControl: The template is a user control element  Class: The template is a class or a module without designer.

**Example:**

Two forms "FAlarmOverview" and "FAlarmReport" are to be lodged as templates. Both forms use text and alarm definitions that are saved in a VisiWinNET-project named "FAlarm\_". Whereas "FAlarmReport" can also be used as a separate form, the "FAlarmOverview" form refers to "FAlarmReport" in the source code. This way, a project containing, "FAlarmOverview" but not "FAlarmReport" cannot be compiled.



In the project manager the index for own templates is to be found via the "Options" button.



A new folder named "Alarm" is to be added to this index.



This new index could (and should) contain all own templates that thematically relate to the VisiWinNET alarm administration.

The data of the two forms are to be copied into this project. Subsequently, a new VisiWinNET project named "Alarms" is to be created. Into this now all definitions must be copied via the "Copy/import" operation.

Finally, the "Alarms.vwn" project database of the new project is to be copied into the template index.



The starting point with the administration of definitions in the template administration is always a project database.

The copied definitions can be:

The texts from the localization that are used with both forms.

Definitions from the alarm system: here, however, it is to be considered that every alarm definition refers to an variable from the data access. To make sure the template functions completely the process variables must also be copied. It also questionable whether the template content will really be universally valid for all future projects. If the alarm definitions are not universally valid at least the name of the template index should provide information about the content of the definitions.



In the new template index a new file with the name "Info.ini" is to be created. With a text editor the following content is to be filled in:

```
[FAlarmOverview.vb]
Description=Overview alarm system
AddAsTemplate=0
VWN=Alarm.vwn
Type="Form"
File1= FAlarmReport
Reference1=System
Reference2=System.IO
Reference3=System.Xml
Reference4=System.Data
Reference5=System.Drawing
Reference6=System.Windows.Forms
Reference7=VisiWinNET.Standard.Client
Reference9=VisiWinNET.Standard.Forms
[FAlarmReport.vb]
Description=Alarm system Report
AddAsTemplate=0
VWN=Alarm.vwn
Type="Form"
Reference1=System
Reference2=System.IO
Reference3=System.Xml
Reference4=System.Data
Reference5=System.Drawing
```

Reference6=System.Windows.Forms  
 Reference7=VisiWinNET.Standard.Client  
 Reference9=VisiWinNET.Standard.Forms



The configuration thus described gives the following instructions:

- "FAlarmOverview" is displayed in the selection dialog with the comment "Overview alarm system" (Description=...)
- When adding "FAlarmOverview" a name cannot be chosen (AddAsTemplate=0). This is usually recommended when a template is involved that always exists only once within a project.
- In addition to "FAlarmOverview" "FAlarmReport" is automatically inserted (File1=...)
- In addition to "FAlarmOverview" the definitions from the "Alarm.vwn" vwn file are automatically inserted (VWN=...)
- Here, the type provides the information required for Visual Studio that the inserted file is a source code file that is supported by a designer (Type=...)
- The reference list indicates the assemblies used by the template (Reference<n>=...). Usually the references shown in this example do not have to be specified as they become automatically existent when a new project is created. If assemblies are used in a template that not already exist in the reference list of a new project they can here be accommodated as references.

### 6.1.3.1 Integrated templates

Below a description of all templates already provided by VisiWinNET Professional:

File	Name	Description	Contained in "Compact"
<b>Empty forms</b>	<b>Form</b>	Empty form with title line	x
	<b>F640x480</b>	Empty form without title line and without margin size 640x480	x
	<b>F800x600</b>	Empty form without title line and without margin size 800x600	x
	<b>F1024x768</b>	Empty form without title line and without margin size 1024x768	x
	<b>F1280x1024</b>	Empty form without title line and without margin size 1280x1024	x

<b>Notification</b>	<b>FMessageBox</b>	Message window: shows a message in a dialog field, and waits for the user to click on a button. Subsequently a whole number is returned indicating which button was clicked on. The click-sensitive operating elements are enlarged for use with a touchscreen input device.	
	<b>FMessageBoxTouch</b>	Message window: shows a message in a dialog field, and waits for the user to click on a button. Subsequently a whole number is returned indicating which button was clicked on. The click-sensitive operating elements are enlarged for use with a touchscreen input device.	x
<b>UserManagement</b>	<b>FChangePassword</b>	Dialog allowing password change by the currently logged on user	
	<b>FChangePasswordTouch</b>	Dialog allowing password change by the currently logged on user. The click-sensitive operating elements are enlarged for use with a touchscreen input device.	x
	<b>FLogOn</b>	Dialog to log on or verify a user (VisiWinNET user administration) through user name and password	
	<b>FLogOnTouch</b>	Dialog to log on or verify a user (VisiWinNET user administration) through user name and password. The click-sensitive operating elements are enlarged for use with a touchscreen input device.	x
	<b>FRight</b>	Dialog to add a new right, or change an existing right	
	<b>FRights</b>	Dialog to administer (add, change or delete) all rights definitions in the project	

<b>FRightsTouch</b>	Dialog to administer (add, change or delete) all rights definitions in the project	x
	The click-sensitive operating elements are enlarged for use with a touchscreen input input device.	
<b>FRightTouch</b>	Dialog to add a new right, or change an existing right	x
	The click-sensitive operating elements are enlarged for use with a touchscreen input input device.	
<b>FUser</b>	Dialog to add a new user or change the parameters of an already existing user	
<b>FUserGroup</b>	Dialog to add a new user group or change the parameters of an already existing user group	
<b>FUserGroups</b>	Dialog to administer all user groups of the project	
<b>FUserGroupsTouch</b>	Dialog to administer all user groups of the project	x
	The click-sensitive operating elements are enlarged for use with a touchscreen input input device.	
<b>FUserGroupTouch</b>	Dialog to add a new user group or change the parameters of an already existing user group	x
	The click-sensitive operating elements are enlarged for use with a touchscreen input input device.	
<b>FUsers</b>	Dialog to administer all users of the project	
<b>FUsersCombo</b>	Dialog to administer all users of the project. The process includes filtering for individual user groups according to a selection list.	

**FUsersComboTouch**

Dialog to administer all users of the project. The process includes filtering for individual user groups according to a selection list.

The click-sensitive operating elements are enlarged for use with a touchscreen input device.

**FUsersTouch**

Dialog to administer all users of the project

The click-sensitive operating elements are enlarged for use with a touchscreen input device.

**FUserTouch**

Dialog to add a new user or change the parameters of an already existing user

The click-sensitive operating elements are enlarged for use with a touchscreen input device.



When adding a template please observe that in the source code call-up examples are shown as comments.

It is imperative that these comments are taken into consideration with the implementation.

## 7 Project configuration

VisiWinNET projects contain settings that influence the runtime performance of the project. Partly these settings are already offered with the creation of a project. Other settings can be changed subsequently in the project configuration.

The settings of the project configuration can be accessed through the equally named node in the Project Explorer. The Project Explorer arranges the properties in different areas. After a click on an area the appropriate settings are displayed on the VisiWinNET properties page.

The available settings depend on the type of the project.

### 7.1.1 Client configuration

Access

Project Explorer node

Project Configuration→Client

Index card on VisiWinNET properties page

Configuration



This configuration dialog is only implemented in the "Standard" project types.

Settings

The client configuration incorporates the settings that are used by the client DLL. This DLL is referred to by every VisiWinNET project. With the start of an application the client DLL is automatically loaded in the application process.

Setting

Description

#### **Timeout watchdog**

The time for the timeout of the connection dialup specifies for how long the client DLL waits until the server component confirms the initialization of the project.

The "calls with established connection" setting specifies the timeout value for the answer to a data request.

If the "Timeout" control box is deactivated the client waits without time restriction.

If a timeout occurs this is noted in the events display.

#### **Reconnect on error**

"Re-connect following failure" allows the application to re-establish a connection with the system servers after a connection failure.

The "infinite" and "...times" settings specify how many re-connecting attempts are to be carried out before the system answers with a failure message.

If the "Reconnect following error" option is deactivated no further reconnecting attempts are started in the event of a connection error.

#### **Try to connect on startup**

Determines how often with the start the system client attempts to connect with the server.

The "infinite" and "...times" settings determine the number of connection attempts before the system reacts with an error message.

**Disabling of standard functions**

Disables the standard functions of the "F1" (access online help) and "F10" (activate main menu) keys.

**Client type**

Setting for the application logon with the system servers. The setting that is to be made here depends on the active license. A (cheaper) viewer license only allows reading access to process variable values.

**Times setting**

The "Blink main cycle" determines the basic interval used by the blink functions of the control elements.

The "AlarmLine main cycle" determines the change interval of the alarm line if the display changes between multiple alarms ("ToggleAlarms" property).

The "Longer display time for top alarm" option leads to the upcoming alarm with the highest priority being displayed for double as long as determined in the "MainCycleAlarmLine" parameter.

**7.1.2 Client events**

Access

Project Explorer node      Project configuration→Client

Index card on VisiWinNET properties page      Events



This configuration dialog is only implemented in the "Standard" project types.

Settings

A list of events is available in the dialog. For every event settings can be made whether it is to be written to the events display (window that is blended over the application if an error or an event occurs) or into the log file of the application (as a standard in the path "<VisiWinNET 2005\Log>".

**7.1.3 Common project properties**

Access

Project Explorer node      Project configuration→Global

Index card on VisiWinNET properties page      General

Settings

The general project properties return the settings made in the project manager when the project was created.



A change of the settings displayed here is not possible. The following standard tasks are fulfilled by other parts of the project administration:

- Re-naming the project: This function is provided in the project manager.
- Changing the project type: Through project converter (please contact support).

### 7.1.4 Compatibility

Access	Project Explorer Node	Standard: Project Configuration→Client Compact: Project Configuration→Runtime
	Index Card on the VisiWinNET properties page	Compatibility
Settings	Contains settings that describe a different behavior towards the 5.x and 6.0 versions.	
Setting	Description	
<b>"Limit" event in the "VarIN" control element enhanced</b>	<p>This setting refers to the occurrence of the "Limit" event in the "VarIn" control element. The event occurs during input to the control element if limit value monitoring is activated through the "LimitCheck" or "UseItemLimits" properties. In particular with checking multi-digit input values a selection can be made between two behaviors:</p> <ul style="list-style-type: none"> <li>• deactivated (compatible with versions 6.0, 5.x): The "Limit" event occurs immediately during input if the specified value exceeds the upper limit value. If the value undercuts the lower limit value the "Limit" event is held back until the input has been ended by moving the cursor or pressing the Enter key. If for example the lower limit had been set to "20" the user can set the value "23" by entering "2" and "3" without triggering the "Limit" event.</li> <li>• activated (standard behavior from version 6.1): If the lower limit is for example set to "20" the user can set the value "23" by entering "2" and "3" only while triggering the "Limit" event. The event will occur after "2" has been entered as that value undercuts the lower limit value. In practice this setting is used for example by entering digits through the a touchscreen number pad (NumPad). Here an incomplete or limit-breaking input is signaled by a color change in the limit value fields.</li> </ul>	
<b>"DecPoint" property in the VisiWinNET item can be written to</b>	<p>This setting refers to all VisiWinNET control elements that link with an analog value as "Process value" in the VWItem property.</p> <p>When linking with analog values the "DecPoint" property within the "VWItem" property is available. The function of this property is influenced by the "'DecPoint' property in the VisiWinNET item can be written to" setting:</p> <ul style="list-style-type: none"> <li>• deactivated (compatible with versions 6.0, 5.x): If the "UnitConversion" property in the control element is set to 'true' then 'DecPoint' has no function at runtime.</li> <li>• activated (standard behavior from version 6.1): The point shifting as set in 'DecPoint' is where applicable also used for the unit conversion activated through 'UnitConversion'. This effects, in addition to the conversion through a unit class, positions after decimal point to be shown with floating point figures.</li> </ul>	

**Optimized activation of variables with form changes**

Variable values required in forms are commonly logged in with the variable kernel once the form is completely loaded. This effects a higher speed with form changes as activation (deactivation not necessary) of required variables with the appropriate communication channel is now made commonly and, thus, faster. Activation of this option, however, also means that the first 'Change' event of an item (cause: ChangedByConfig) does not always contain the current value from the communication channel but possibly an older or the start value.

Deactivation of this setting restricts the functions of the following components:

VisiWinNET.DataAccess.MultiItemActivator (Implemented in VisiWinNET.Standard.Client/ VisiWinNET.Compact.Systems)

VisiWinNET.Forms.BaseForm (Implemented in VisiWinNET.Standard.Forms/ VisiWinNET.Compact.Forms)

In VisiWinNET SMART the forms directly inherit from "VisiWinNET.Forms.BaseForm". Here, too, deactivation of the setting causes functional restrictions.

### 7.1.5 Computer names

Access

Project Explorer Node

Project configuration → Computer names

Access

Following a double click on the above node a table editor opens.



This configuration dialog is contained in the „Standard“ project types only.

Settings

Contains a table listing synonyms for computer names.

These synonyms are used in the following systems:

- User administration: "User group allocation with other computers" expects computer names with which users can be allocated to other user groups.
- Data access: In the "Computer name" field of an OPC Server communication channel a computer name is expected.
- Archive system: A trend definition set as "Remote Trend" contains a computer name.

At least at development time it is not yet known with these settings through which name (which IP address) the remote computer can be addressed. The developer can instead of the real computer name also use one of the synonyms determined here (with leading @ character) in the above mentioned definitions. Only the real computer names must then be added to the list of synonyms on the target system.

Setting

Description

**Alias**

The name that is in the above mentioned definitions used as a synonym.

**Computer name**

Network name or IP address of the computer

### 7.1.6 Connection Server project

Access

Project Explorer Node

Project configuration → Server project

Index card on VisiWinNET properties page

Connection



This configuration dialog is contained in the "Client" project type only.

If Client and Server projects are running on the same computer these settings are without a function.

Setting


Determines parameters for the connection with the Server project. For the connection two ports are to be determined.

Setting

Description

#### **ServerPort**

Port number for the connection from Client to Server. The port number put in here must be identical with Client and Server. To make the setting with the Server project the following steps are required:

- Start "VisiWinNET.Server.exe".
- Following the appearance of the  symbol next to the system time in the task bar the exe configuration dialog is to be opened through the "Configuration" entry of the context menu.
- Here the Server port is put in.

#### **ClientPort**

Port number for the connection from Server to Client. For the callback functions of the Server a second connection is established through the port set here.

### 7.1.7 Extended Client Configuration

Access

Project Explorer Node

Project configuration→Client

Index card on VisiWinNET property page

Extended



This configuration dialog is contained in the „Standard" project types only.

Settings

The settings to be made here concern the event display and the log file created by the client.

Errors and messages occurring within the application client are displayed as event display within a window and as a log file.

The following settings influence the event display:

Setting

Description

**Window position**

Determines where on the screen the event display is positioned.

**Always in the foreground**

Determines whether the event display can be covered by other windows. If the option is deactivated it is possible that the event display window is covered in the event of a form change.

**Deactivate**

With this option activated no event window is displayed.

The following settings influence the log file:

Setting

Description

**Size**

The log file is a circular buffer. The size determines how large the file can become before the oldest entries are overwritten. The specification is in kilobyte.

**Name**

Determines the name of the log file. As standard the file is created in the path "<VisiWinNET 2005\Log>".

**Delete on startup**

If this option is activated old entries are deleted from the log file when the application is restarted.

### 7.1.8 Extended project properties

Access	Project Explorer node	Project configuration→Global
	Index card on VisiWinNET properties page	Enhanced
Settings	The enhanced properties allow the projector to store general information in the project databank. The settings made here can be read programmatically at runtime through the properties of the "ProjectInfo" class ("VisiWinNET.Project" namespace).	

### 7.1.9 Language settings

Access	Project Explorer node	Project configuration→Global
	Index card on VisiWinNET properties page	Languages
Settings	<p>The dialog lists the languages contained in the project. One of these languages can be marked as the developer language.</p> <p>The language marked as the developer language has the following function:</p> <ul style="list-style-type: none"> <li>• When selecting text, e.g. through the 'LocalizedText' properties of the VisiWinNET control elements, the text in the selection dialog is displayed in the developer language.</li> </ul> <p>With input in the system editors (e.g. alarm definition, "Text" parameter) the specified text is allocated to the developer language.</p>	

### 7.1.10 Licence information

Access	Project Explorer node	Projektkonfiguration→Licence information
	Index card on VisiWinNET properties page	Licence information
Settings	<p>Provides information on the number of communication channels and variables required to determine the runtime license. To find these values click the "Update" button.</p> <p>The VisiWin runtime licenses are graded by the number of process variables. For further information see the current INOSOFT price list.</p> <p>The rule is: Power Tags are variables allocated to a communication channel. Internal variables only exist in the variable kernel, and are not exchanged with communication channels.</p> <p>The required runtime license depends on the respective larger number of the two mentioned variable types.</p>	

### 7.1.11 Runtime behaviour

Access

Project Explorer node

Project configuration → Runtime

Index card on VisiWinNET properties page

Standard



This configuration dialog is only implemented in the "Standard" project types.

Settings

Contains runtime and VWN Manager settings.

Setting

Description

#### Time Configuration

Cycle time settings for the VWEngine. The cycle time itself (if the cycle data arrived faster than required the system cycle will pause until the cycle time set was reached; If data collecting takes longer than specified the system clock will automatically prolong, which means that no data request will get lost) as well as the break between two cycles (provides for computing time indication to the VW-Application or other process) can be set.

ms cycle time

The system cycle time for data collecting; A basic pulse rate of 1 sec. (100 ms) is preset.

ms idle time

Break between two system cycles (preset: 1 ms)

#### VisiWinNET Manager

The main menu of the VisiWinNET Manager can be configured.

Configuration disabled

The dialog Configuration cannot be selected during runtime

Variable dialogbox disabled

The dialog VWEngine cannot be selected during runtime

Start/Stop/Exit disabled

The menu levels Start, Stop and Exit cannot be selected during runtime

#### Runtime data

Encode data

Determines whether the data created at runtime from the systems "Logging", "Recipe", "Alarm" and "User administration" are to be coded. This setting offers enhanced security against manipulation of the created files.

Password for writing access

Determines the password for the Access databanks created at runtime from the "Logging", "Recipe", "Alarm" and "User administration" systems. If this password is not entered when the databank is opened (e.g. with Access) the file can only be opened in write-protected mode..

## Communication

Timespan waiting for VisiWin driver	Sets the time (in seconds) that the variable kernel waits for a VisiWin driver in the initializing phase. If the driver does not start within the timespan set here an error message is generated.
Read all variables once with start	Determines whether the variable kernel is to read all process values once in the initializing phase. This serves to initialize the internal variable cache.

### 7.1.12 Runtime configuration

Access

Project Explorer node      Project configuration → Runtime  
 Index card on VisiWinNET properties page      Configuration



This configuration dialog is contained in the „Compact“ project types only.

Settings

Description

#### Times setting

The "Blink main cycle" determines the basic interval used by the blink functions of the control elements.

The "AlarmLine main cycle" determines the change interval of the alarm line if the display changes between multiple alarms ("ToggleAlarms" property).

The "Longer display time for top alarm" option leads to the upcoming alarm with the highest priority is displayed in the alarm line for double as long as determined in the "MainCycleAlarmLine" parameter.

### 7.1.13 Server project

Access

Project Explorer Node      Project configuration → Server project  
 Index card on VisiWinNET properties page      General



This configuration dialog is contained in the "Client" project type only.

Settings

Determines the Server project. The settings are identical with the settings made when a new Client project is created in the Project Manager.

### 7.1.14 System selection

Access

Project Explorer node

Project configuration → Server components

Index card on VisiWinNET properties page

General

Settings

The systems highlighted here are in the development environment displayed as editors. Through these editors the system-specific definitions are written into the project databank.

At runtime only the system servers highlighted here are loaded which means for example that with the de-selection of the trend system the recording functions of the trend server do not become active.

## 8 Project converters

VisiWinNET as a product is available in different versions that can be installed parallel. The installed version as indicated (see menu entry Help→\*About VisiWinNET...) contains the following information:

**<Main Version>.<Release> <SPx> <TPx>**

<b>Main Version</b>	5:For Visual Studio 2003/ .NET Framework 1.1 6:For Visual Studio 2005/ .NET Framework 2.0
<b>Release</b>	Release versions embrace new features of a "Technical Preview" and bugfixes from previous "Service Packs".
<b>SP&lt;x&gt;</b>	Optional indication of an installed Service Pack (with consecutive numbering)
<b>TP&lt;x&gt;</b>	Optional for Beta testers: Indication of a Technical Preview (with consecutive numbering) in anticipation of future features

More information about the installation:

- Service Packs may change an already existing Release installation. By this the runtime components of all customer projects based on the Release are changed.
- Technical Previews may change already installed Technical Previews. As a Release the TP version designators carry the forthcoming release number. This means that TP versions are overwritten when the next Release is installed.
- New Release versions leave earlier Releases untouched. This allows, for example, the projector to keep projects that are already finished on a development computer in their version for further maintenance. He can start new projects with a new version.

Because of changes in the informations of project databank and application the projects are not 100 percent compatible between the Releases. To lift a project up to a new version a converter must be used.

The converting function is started through the "Project☐Convert into new VisiWinNET version" menu entry.

The appropriate project converters do the bulk of the job. Yet further adaptations may have to be made manually. The following chapters refer to these manual adaptations.

## 8.1 Conversion 5.0 to higher versions

### Notes on conversion

The conversion has only made changes to the project file (vwn), and changed the VisiWinNET version number in the Visual Studio project file (proj).

The source text in the vb or cs files was not changed. As, however, changes were made in the VisiWinNET interfaces (VisiWinNET.Forms, VisiWinNET.Client / VisiWinNET.Systems) the Visual Studio Compiler will highlight some program lines as faulty. Consequently, these have to be adapted as follows:

Object	Property/Method		replace by
<b>Control Element ComboBox</b>	ItemHeighth	>>	ItemHeight
<b>Control Element ListBox</b>	ItemHeighth	>>	ItemHeight
<b>Class RecipeFiles</b>	DeleteRecipeFile	>>	Delete
<b>Class Trend</b>	LocalizedText	>>	Text

## 8.2 Conversion 5.x to 6.0

### Notes on conversion

The appropriate entry for conversion lies under the "Extras" menu in the Project Manager of VisiWinNET 2005.

Subsequent to the conversion of a Professional project Visual Studio provides a conversion message that the project wants to convert, too. Here, NO new directory should be specified but let the project be overwritten as otherwise it would not be registered with VisiWin.

## 8.3 Conversion VisiWinStudio to VisiWinNET

VisiWinStudio projects are based on VB6 projects. These are technologically and in content not particularly suitable for conversion to .NET. From VisiWinNET's perspective there is no appropriate converter for the following reasons:

- The interfaces and control elements of both VisiWinNET and the framework have been completely redesigned.
- The code implemented by the developer is only restrictedly upwardly compatible.
- Due to the linguistic enhancements of VB.NET over VB6 an adoption of existing sources does not seem appropriate.

The contents of the project databank can, however, widely be saved. The latest VisiWinStudio versions did already implement the global Copy/Add functions. Definitions copied in VisiWinStudio can be added again into VisiWinNET projects of version 5x.

For the above mentioned reasons, however, the surface must be completely re-projected.