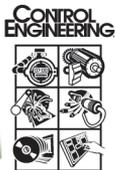




# InduSoft

# Web Studio™ v7.1

## Quick Start Guide



2012  
ENGINEERS'  
CHOICE  
AWARDS



2010  
ENGINEERS'  
CHOICE  
AWARDS





# InduSoft Web Studio™ v7.1

## Quick Start Guide



---

InduSoft® is a registered trademark of InduSoft, Inc.

InduSoft Web Studio™, EmbeddedView™, and CEView™ are trademarks of InduSoft, Inc.

Windows, Windows XP, Windows XP Embedded, Windows Embedded Standard 7, Windows Embedded Compact, Windows 2003 Server, Windows 2008 Server, Windows Vista, Windows 7, Windows CE, and Internet Explorer are registered trademarks of Microsoft Corp. in the United States and other countries.

Other brand or product names are trademarks or registered trademarks of their respective owners.

Copyright © 2012 InduSoft, Inc. All rights reserved worldwide.

This document shall not be reproduced or copied in any manner without expressed written authorization from InduSoft.

The information contained within this document is subject to change without notice. InduSoft, Inc. does not guarantee the accuracy of the information.

PN: 032312-QSG-IWS-A-EN-PT

---

---

## Contents

<b>INTRODUCTION.....</b>	<b>3</b>
About this software.....	4
Conventions used in this documentation.....	10
Comparison of InduSoft Web Studio software components.....	12
Install the full InduSoft Web Studio software.....	14
Install EmbeddedView or CEView.....	21
Execution modes.....	26
<b>THE DEVELOPMENT ENVIRONMENT.....</b>	<b>28</b>
Title bar.....	29
Status bar.....	30
Application menu.....	31
Quick Access Toolbar.....	32
Ribbon.....	34
Home tab.....	34
View tab.....	35
Insert tab.....	35
Project tab.....	36
Graphics tab.....	36
Format tab.....	37
Help tab.....	38
Project Explorer.....	39
Global tab.....	39
Graphics tab.....	41
Tasks tab.....	42
Comm tab.....	45
Screen/Worksheet Editor.....	47
<b>ABOUT TAGS AND THE PROJECT DATABASE.....</b>	<b>48</b>
Tag Name Syntax.....	50
Tag Data Type.....	51
Using Array Tags.....	53
Using Indirect Tags.....	57
<b>TUTORIAL: BUILDING A SIMPLE PROJECT.....</b>	<b>59</b>
Creating a new project.....	60
Specifying the startup screen.....	62
Creating tags.....	64

- Creating the startup screen..... 66
  - Drawing the startup screen's title..... 68
  - Drawing a button to open another screen..... 70
  - Saving and closing the startup screen..... 71
- Creating the synoptic screen..... 72
  - Drawing the synoptic screen's title..... 72
  - Drawing "Date" and "Time" displays..... 73
  - Placing an "Exit" icon..... 74
  - Testing the project..... 76
  - Placing an animated tank..... 76
  - Placing a level slider..... 79
  - Drawing a tank selector..... 80
  - Testing the project..... 81
- Configuring the communication driver..... 83
  - Monitoring device I/O during runtime..... 86
- Downloading your project to a Windows Embedded device..... 88
- Deploying your project as a web application..... 91

## Introduction

---

InduSoft Web Studio (or IWS, for short) is a powerful, integrated tool that exploits key features of Microsoft operating systems and enables you to build full-featured SCADA (Supervisory Control and Data Acquisition) or HMI (Human-Machine Interface) programs for your industrial automation business.

This *InduSoft Web Studio Quick Start Guide* is intended for individuals using IWS for the first time. This publication will help you quickly familiarize yourself with the basic functions of IWS.

## About this software

---

InduSoft Web Studio is powerful software for developing HMI, SCADA, and OEE/Dashboard projects that can be deployed anywhere.

Each IWS project consists of:

- A project tags database to manage all runtime data, including both internal variables and scanned I/O;
- Configurable drivers to communicate in real-time with programmable logic controllers (PLCs), remote I/O devices, and other data-acquisition equipment;
- Animated HMI screens and OEE dashboards; and
- Optional modules such as alarms, events, trends, recipes, reports, scriptable logic, schedulers, a security system, and a complete database interface.

After you develop your project, you can either run it locally on your development workstation or download it to a remote workstation and run it there using InduSoft Web Studio or EmbeddedView/CEView runtime software. The workstation processes I/O data from connected devices according to your project parameters and then reacts to, displays, and/or saves the data.

### Product features

#### Alarms

In addition to all the alarm functions you'd expect, IWS v7.1 also sends alarms using multi-media formats like PDF. Use remote notification to have alarms sent right to your inbox, a printer, or a smartphone! Alarms are real-time and historical, log data in binary format or to any database.

#### Animation

IWS gives you great command over graphics. Paste images, and even rotate them dynamically. Fill bar graphs with color, or adjust the scale of objects with easy-to-use configuration. Other animations include "command" (for touch, keyboard and mouse interaction), hyperlink, text data link, color, resize (independent height and width), position, and rotation (with custom rotation point).

**Database**

Connect to any SQL database (MS SQL, MySQL, Sybase, Oracle), or MS Access or Excel, and ERP/MES systems (including SAP), even from Windows Embedded CE. Flexible enough to have a built in interface without the need for knowing SQL (for trends, alarms/events, grid and other objects), or use any SQL statement you need anywhere you need it.

**Drivers**

IWS v7.1 contains over 240 built-in drivers for most PLCs, temperature controllers, motion controllers, and bar code/2D/RFID readers. InduSoft driver toolkits allow you the flexibility to build your own drivers. Use these built in drivers without the need for OPC servers (but are an optional connection method).

**Email**

Send email using SMTP to desktop, email enabled phone, or any enabled device. Get real-time information on alarms, process values, and other events. InduSoft Web Studio v7.1 supports SSL encryption allowing the use of third-party providers such as Gmail.

**Events**

IWS v7.1 offers traceability for operator initiated actions or internal system activity. Log events such as security system changes (user logon or off), screen open/close, recipe/report operations, custom messages and system warnings. Also any tag value changes including custom messages.

**FDA Traceability**

Take advantage of built-in functionality to create 21 CFR part 11 compliant projects with traceability and e-signatures. These features are often used for pharmaceutical and food applications, but also for any application where traceability is a must.

**FTP**

Automatically upload or download files during runtime to/from remote storage locations using FTP protocol and flexible

scripting functions. Configure FTP via scripting or the included configuration interface.

### **Graphics and Design Tools**

Create powerful screens to meet any application need using the improved tools in our graphic interface. Combine built-in objects to create any functionality required. Store graphics in the library for future use, or easily make project across a product line share a consistent "look and feel".

### **Historical Performance**

We have optimized the trend history module, and designed it to load millions of values from SQL Relational Databases with high performance, with built-in data decimation in the Trend Control. Easy to use tools provide quick access to Statistical Process Control (SPC) values without any need for programming.

### **Intellectual Property Protection**

Screens, documents, scripts and even math worksheets can be individually password protected. This prevents unauthorized viewing or editing of your corporate custom functionality. Protect the entire project with just a few mouse clicks.

### **Multi-Language**

Develop your application in one of many development languages, including English, Portuguese, German, and French.

### **.NET and ActiveX**

Use third-party controls to enhance your project. IWS is a container for ActiveX and .NET controls. Add functionality such as browsers, media players, charting, and other tools that support the ActiveX or .NET interface standards.

### **OPC**

Drivers for all major brands of PLCs are built in, but any OPC server may optionally be used. IWS supports OPC DA (Server/Client), OPC HDA(Server), UA (Client) and OPC .NET 3.0 (Client). InduSoft Web Studio v7.1 also supports OPC XML as an additional add-on.

**PDF Export**

Send Alarms, Reports, or any file (including .doc or .txt) to a production supervisor, quality manager, or maintenance staff using the included PDF writer.

**Recipes**

Save time and maintain consistency by automating part parameters or productions quantities with any triggering event.

**Redundancy**

For critical applications where data is vital, IWS v7.1 supports web server, database and overall system redundancy.

**Reports**

Create clear, concise reports in text format, graphical RTF, XML, PDF, HTML, and CSV or integrate with Microsoft Office. Get the data you need, in the format you need it, to make informed decisions, fast.

**Scalable**

Develop once and deploy everywhere. Take an application created for Windows CE and deploy it on any supported Microsoft operating system, including Embedded, Windows 7 (and soon Windows 8), and Server editions.

**Scheduler**

Schedule custom tag changes on date/time, frequency, or any trigger. Use this for simulation, to trigger reports or other functionality at a particular time of day, or even to trigger driver worksheets to read/write at a scan rate you choose.

**Scripting**

Two powerful scripting languages are supported. Use built-in InduSoft functions or use standard VBScript to take advantage of widely available resources. Both can be used simultaneously to give you the functionality you need.

**Security**

IWS includes support for group and user accounts, e-signatures, and traceability, as well as support for the ADAM

Server, in addition to standard LDAP Servers. Integrate your project to the Active Directory (Users and Groups).

### **SSL Support for Emails**

Native support for Secure Socket Layer (SSL), makes it easy and secure to send emails from InduSoft Web Studio using third-party tools such as Gmail!

### **Standards**

Take advantage of common industry standards to develop applications that are compatible with any format. TCP/IP, .Net, ActiveX, OPC (client and server), ADO/ODBC, COM/DCOM, OLE, DDE, XML, SOAP, and HTML are supported.

### **SNMP**

Easily configure managed networked devices on IP networks (such as switches and routers) using incorporated SNMP configuration commands and an easy-to-use configuration interface.

### **Symbols**

Included library features push buttons, pilot lights, tanks, sliders, meters, motors, pipes, valves and other common objects. Use the included symbols in your project, modify existing symbols to suit your needs, or create your own from scratch. Plus support for third-party symbol libraries and graphic tools.

### **Tag Database**

IWS features an object oriented database with boolean, integer, real, strings, arrays, classes (structures), indirect tags and included system tags.

### **Thin Clients**

Remotely view screens as web pages using Internet Explorer web browser, or InduSoft Secure Viewer. Use SMA (Studio Mobile Access) to monitor or access process values and alarms with remote devices such as tablets and mobile phones. Enhanced SMA offers data in easy-to-read widgets that can be viewed on any WebKit (HTML5) based web browser found on iPads, and Android phones and tablets.

**Trends**

Real-time and Historical trends are supported. Log data in binary format or to any database locally and remotely. Color or fill trends with graphic elements to enhance clarity of data. Date/Time based or numeric (X/Y plot) trends give you the flexibility to display information that best suits your application.

**Troubleshooting**

Quickly debug and verify a project using local and remote tools for troubleshooting, including status fields, DatabaseSpy and LogWin. Capture screen open and close times, see communications in real-time, and messages related to OPC, recipes/reports, security, database errors and even custom messages. Quickly get your project finished using these powerful tools.

## Conventions used in this documentation

---

This documentation uses standardized formatting and terminology to make it easier for all users to understand.

### Text conventions

This documentation uses special text formatting to help you quickly identify certain items:

- Titles, labels, new terms, and messages are indicated using *italic* text (for example, *Object Properties*).
- File names, screen text, and text you must enter are indicated using **monospace** text (for example, **D:\Setup.exe** ).
- Buttons, menu options, and keyboard keys are indicated using a **bold** typeface (for example, **File** menu).

In addition, this documentation segregates some text into **Tip**, **Note**, and **Caution** boxes:

- **Tips** provide useful information to save development time or to improve the project performance.
- **Notes** provide extra information that may make it easier to understand the nearby text, usually the text just before the note.
- **Cautions** provide information necessary to prevent errors that can cause problems when running the project, and may result in damage.

### Mouse and selection conventions

Because most PCs used for project development run a version of Microsoft Windows with a mouse, this documentation assumes you are using a mouse. Generally, a PC mouse is configured for right-handed use, so that the left mouse button is the primary button and the right mouse button is the secondary button.

This documentation uses the following mouse and selection conventions:

- **Click** and **Select** both mean to click once on an item with the left mouse button. In general, you click buttons and you select from menus and lists.
- **Double-click** means to quickly click twice on an item with the left mouse button.
- **Right-click** means to click once on an item with the right mouse button.

- **Select** also means you should use your pointing device to highlight or specify an item on the computer screen. Selecting an item with a touchscreen is usually the same as selecting with a mouse, except that you use your finger to touch (select) a screen object or section. To select items with your keyboard, you typically use the Tab key to move around options, the Enter key to open menus, and the Alt key with a letter key to select an object that has an underlined letter.
- **Drag** means to press down the appropriate mouse button and move the mouse before releasing the button. Usually an outline of the item will move with the mouse cursor.

## Windows conventions

This documentation uses the following Windows conventions:

- **Dialogs** are windows that allow you to configure settings and enter information.
- **Text boxes** are areas in dialogs where you can type text.
- **Radio buttons** are white circles in which a black dot appears or disappears when you click on the button. Typically, the dot indicates the option is selected or enabled. No dot indicates the option is cleared or disabled.
- **Check boxes** are white squares in which a check  (Titlebar) appears or disappears when you click on it with the cursor. Typically, a check  indicates the option is selected or enabled. No check  indicates the option is cleared or disabled.
- **Buttons** are icons in boxes appear "pressed" when you click on them.
- **Lists** are panes (white boxes) in windows or dialogs containing two or more selectable options.
- **Combo boxes** have arrows that, when clicked, show part or all of an otherwise concealed list.
- **Dockable windows** are windows that you can drag to an edge of the interface and merge with that edge.

## Comparison of InduSoft Web Studio software components

The InduSoft Web Studio software suite actually comprises several individual components that can be installed on different computers to perform different functions.

### Comparison of InduSoft Web Studio software components

Component	Functions	Platforms
InduSoft Web Studio	<ul style="list-style-type: none"> <li>Project development</li> <li>Tag integration</li> <li>Remote management</li> <li>Project runtime server</li> <li>Project runtime client</li> </ul>	<ul style="list-style-type: none"> <li>Windows</li> <li>Windows Server</li> <li>Windows Embedded Standard</li> </ul>
EmbeddedView (including Remote Agent)	<ul style="list-style-type: none"> <li>Agent to allow remote management</li> <li>Project runtime server (limited tag count)</li> <li>Project runtime client</li> </ul>	<ul style="list-style-type: none"> <li>Windows Embedded Standard</li> </ul>
CEView (including Remote Agent)	<ul style="list-style-type: none"> <li>Agent to allow remote management</li> <li>Project runtime server (limited tag count)</li> <li>Project runtime client</li> </ul>	<ul style="list-style-type: none"> <li>Windows Embedded Compact</li> </ul>
Web Tunneling Gateway (WTG)	<ul style="list-style-type: none"> <li>Enables a public-facing web server to pass data between your project runtime server and web thin clients, when your project runtime server is located on a secure, internal network</li> </ul>	<ul style="list-style-type: none"> <li>Microsoft IIS for Windows</li> </ul>
Mobile Access	<ul style="list-style-type: none"> <li>Enables your project runtime server to deliver HTML5-enhanced project screens to tablets and smartphones</li> </ul>	<ul style="list-style-type: none"> <li>Microsoft IIS for Windows</li> </ul>
InduSoft Thin Client (a.k.a. Secure Viewer)	<ul style="list-style-type: none"> <li>Web thin client (standalone application)</li> </ul>	<ul style="list-style-type: none"> <li>Windows</li> <li>Windows Server</li> <li>Windows Embedded Standard</li> </ul>

---

Component	Functions	Platforms
		<ul style="list-style-type: none"><li>• Windows Embedded Compact</li></ul>
ISSymbol	<ul style="list-style-type: none"><li>• Web thin client (browser plug-in)</li></ul>	<ul style="list-style-type: none"><li>• Internet Explorer for Windows</li></ul>

The architecture of your finished IWS project depends on which components you install, where you install them, and how you connect them to each other.

In most cases, you should first install the full InduSoft Web Studio software on your primary workstation. Not only does it set up the project development environment on your computer, it also unpacks the rest of the components so that they can be installed on other computers.

There are separate, downloadable installers only for the web thin clients (i.e., Secure Viewer and ISSymbol).

## Install the full InduSoft Web Studio software

---

Install the full InduSoft Web Studio software on your Windows computer in order to develop IWS projects, act as a project runtime server, and/or act as a project runtime client.

To install and run the full InduSoft Web Studio software, you must have:

- A Windows-compatible computer with a standard keyboard, pointer input (e.g., mouse, trackpad, or touchscreen), and SVGA-minimum display;
- A Windows, Windows Server, or Windows Embedded Standard operating system that is currently supported by Microsoft, which at this time includes:
  - Microsoft Windows XP Service Pack 3
  - Microsoft Windows Vista Service Pack 2
  - Microsoft Windows 7 Service Pack 1 or later (see Note 2)
  - Microsoft Windows Server 2003 Service Pack 2
  - Microsoft Windows Server 2008 Service Pack 2
  - Microsoft Windows Server 2008 R2 Service Pack 1 or later (see Note 2)
  - Microsoft Windows XP Embedded Service Pack 3 (see Note 3)
  - Microsoft Windows Embedded Standard 7 (see Note 3)
- Microsoft .NET Framework 3.5.1 (see Note 4);
- Microsoft Internet Explorer 6.0 or later;
- 2 GB free hard drive space or non-volatile memory; and
- An Ethernet or Wi-Fi network adapter, for TCP/IP networking.

We recommend the Home Premium, Professional, Enterprise, and Ultimate editions of Windows, because they include Microsoft Internet Information Services (IIS) as a pre-installed feature that can be turned on. We do not recommend the Starter and Home Basic editions because they do not include Microsoft IIS.

The following items are optional but recommended:

- A DVD-ROM drive, to install the software from disc.  
This is optional because you can also download the installer to your computer.
- A USB or parallel port, to be used with hardkey licensing.

This is optional because softkey licensing is also available.

- Serial COM ports and adapters, to be used for direct communication with PLCs and other devices.

This is optional because many newer device protocols use TCP/IP communication instead of serial communication.

- Microsoft IIS installed and turned on, to make your projects accessible to mobile devices. For more information, see the description of Mobile Access Runtime below.

This is optional because you may choose not to install the Mobile Access Runtime feature.

- Microsoft Visual Studio Team Explorer 2010 installed, to do workgroup collaboration and version control. For more information, see the description of Collaboration below.

This is optional because you may choose not to install the Collaboration feature.

 **Note:**

1. You must have Administrator privileges on the computer in order to install software.
2. Only Windows 7 and Windows Server 2008 R2 are under what Microsoft calls "mainstream support", which means they are actively maintained and additional service packs may be released for them. The rest of the listed operating systems are under what Microsoft calls "extended support", which means they are not actively maintained.
3. Even though you can install the full InduSoft Web Studio software on a Windows XP Embedded or Windows Embedded Standard 7 computer, if you do not plan to do project development on that computer and will use it only as a project runtime server and/or project runtime client, then you should consider installing EmbeddedView instead. EmbeddedView does not support as many project tags as the full software, but it requires fewer system resources and it can be installed and managed remotely.
4. The InduSoft Web Studio installer will also attempt to install Microsoft .NET Framework 3.5.1, especially on older versions of Windows where it is not a pre-installed feature. However, depending on your computer's security settings, this installation may fail without notice. If you experience problems later, while trying to run InduSoft Web Studio, then you should at least

confirm that Microsoft .NET Framework 3.5.1 was successfully installed. To do this in Windows XP or Windows Server 2003, use the *Add/Remove Programs* control panel. To do this in Windows Vista, Windows 7, Windows Server 2008, or Windows Server 2008 R2, use the *Windows Features* control panel. Please note that Microsoft .NET Framework 4 does not include Microsoft .NET Framework 3.5.1.

To install the full InduSoft Web Studio software:

1. Do one of the following:

- Download the zipped installer to your computer, either from our website ([www.indusoft.com](http://www.indusoft.com)) or from another location on your network where you have previously saved it. Extract the zip archive, open the resulting folder, and then locate and double-click `setup.exe`.
- Insert the installation disc into your DVD-ROM drive. If it does not autorun, then manually run `D:\indusoft.htm`. When the page opens in your browser, follow the instructions for Product Installation.

The installation wizard runs and asks you to select a language for the installation.

2. Select a language from the list, and then click **OK**.

This selection determines what language the installation wizard will use, as well as what language the project development application's user interface will be in. You can change the language of the user interface later, after you have installed the software.

The wizard prepares for installation. During this step, it automatically installs SafeNet's Sentinel drivers (a part of the software licensing mechanism) and Microsoft .NET Framework 3.5.1.

3. On the *Welcome* page of the wizard, click **Next** to proceed with the installation.
4. On the *License Agreement* page, click either **Yes** to accept the agreement and proceed or **No** to refuse the agreement and exit the wizard.
5. On the *Customer Information* page, enter your user name and company name, and then click **Next**.
6. On the *Choose Destination Location* page, select the folder where the software should be installed, and then click **Next**.

By default, the software will be installed at: `C:\Program Files\InduSoft Web Studio v7.1\`

7. On the *Select Features* page, select which software features and components you want to install on your computer, and then click **Next**.

---

The following features are available:

**Program Files**

The main program files for the project development application, the project runtime server, and the project runtime client. This feature cannot be deselected.

**Demos**

Premade projects that demonstrate the capabilities of the InduSoft Web Studio software.

**Hardkey Support**

Additional drivers to support the use of hardkey (USB or parallel port) software licenses.

**OPC Components**

Additional components to communicate with other OPC-compatible devices. This includes OPC DA (a.k.a. OPC Classic), OPC UA, OPC .NET (a.k.a. OPC Xi), and OPC XML-DA.

**PDF Printing**

Software for printing runtime reports directly to PDF files.

**Security System Device Driver**

An additional keyboard driver to control user input for project security.

**Symbol Library**

A library of premade but configurable screen objects such as pushbuttons, toggle switches, gauges, dials, indicator lights, and so on.

**Windows CE Runtime**

Also called CEView — project runtime software (server/client, but not development) for Windows Embedded Compact on a variety of processors. Check the documentation for your specific device to see what processor it uses.



**Note:** Selecting this feature will not actually install CEView on your computer. It will simply unpack the installation files and copy them to your program folder, so that you can later install CEView on a Windows Embedded Compact device.

**Windows Embedded Runtime**

Also called EmbeddedView — project runtime software (server/client, but not development) for Windows Embedded Standard computers.

 **Note:** Selecting this feature will not actually install EmbeddedView on your computer. It will simply unpack the installation files and copy them to your program folder, so that you can later install EmbeddedView on a Windows Embedded Standard computer

### Windows Mobile

Additional runtime software for older Windows Mobile devices.

### Mobile Access Runtime

Additional software to automatically configure Microsoft IIS to make your project runtime accessible to mobile devices such as tablets and smartphones.

This feature requires that you have Microsoft IIS installed and turned on, with ASP, ASP.NET, and ISAPI Extensions enabled. The installation wizard will attempt to verify that you do, and if you do not, then it will not install this feature.

For more information about installing and configuring Microsoft IIS, visit: [technet.microsoft.com/library/cc753433.aspx](http://technet.microsoft.com/library/cc753433.aspx)

You do not need to install this feature at this time. You can install it later, after you have installed and turned on Microsoft IIS, or you can install it on another computer that is acting as your project runtime server. There is a separate Mobile Access Runtime installer (`MobileAccess.exe`) that is unpacked with the rest of the software.

 **Note:** To use Mobile Access Runtime, your InduSoft Web Studio software license must have the necessary license add-on. You can still develop projects that include mobile access features, even without the license add-on, but mobile devices will not be able to access them.

### Collaboration

Additional tools for workgroup collaboration and version control within the development environment.

This feature requires that you have Microsoft Visual Studio Team Explorer 2010 installed on the same computer. The

installation wizard will attempt to verify that you do, and if you do not, then it will not install this feature.

You should also have Microsoft Visual Studio Team Foundation Server 2010 installed and running somewhere on your network.

For more information about Microsoft Visual Studio 2010, visit: [msdn.microsoft.com/library/dd831853.aspx](http://msdn.microsoft.com/library/dd831853.aspx)



**Note:** To use Collaboration, your InduSoft Web Studio software license must have the necessary license add-on.

8. On the *Ready To Install* page, click **Install**.  
The installation proceeds.
9. On the final page, click **Finish**.  
The wizard closes.

When you have finished the installation, you should find the InduSoft Web Studio software in your **Start** menu at **Start > All Programs > InduSoft Web Studio v7.1**. It includes the following components:

#### **InduSoft Web Studio v7.1**

The project development application, project runtime server, and project runtime client.

#### **Register**

A utility that manages your InduSoft Web Studio software license.

#### **StartUp**

A shortcut that automatically starts the project runtime server and runs the most recent project.

#### **Remote Agent**

A utility that allows InduSoft Web Studio running on other computers to connect to your computer and send projects to it.

#### **Quick Start Guide**

A brief guide to installing and using the project development application, including a tutorial for building a simple project.

#### **Help Manual**

A complete technical reference and user guide for the InduSoft Web Studio software.

### **Release Notes**

A list of changes in the the InduSoft Web Studio software.

There should also be a shortcut icon on your desktop.

To run the project development application, do one of the following:

- Double-click the shortcut icon on the desktop; or
- Click **Start > All Programs > InduSoft Web Studio v7.1 > InduSoft Web Studio v7.1**.

## Install EmbeddedView or CEView

---

Install EmbeddedView on a Windows Embedded Standard computer, or install CEView on a Windows Embedded Compact device, to use it as a project runtime server and/or client.

You must have already installed the full InduSoft Web Studio software on your computer, either from the installation disc or from the downloadable installer, because the redistributable EmbeddedView and CEView software is included in the InduSoft Web Studio program folder.

To install and run EmbeddedView or CEView, you must have:

- A Windows Embedded-compatible computer or device (a.k.a. the target system);
- A Windows Embedded Standard or Windows Embedded Compact operating system that is currently supported by Microsoft, which at this time includes:
  - Microsoft Windows XP Embedded Service Pack 3
  - Microsoft Windows Embedded Standard 7
  - Microsoft Windows Mobile or Windows CE 5.x/6.x
  - Microsoft Windows Embedded Compact 7
- 1 GB free hard drive space or non-volatile memory;
- At least one USB 2.0 port; and
- An Ethernet or Wi-Fi network adapter, for TCP/IP networking.

The following items are optional but recommended:

- Serial COM ports and adapters, to be used for direct communication with PLCs and other devices.

This is optional because many newer device protocols use TCP/IP communication instead of serial communication.

Installing EmbeddedView or CEView on a target system is actually a two-part procedure. First, you will copy the Remote Agent utility to the target system and then run it. Remote Agent allows you to connect from the project development application to the target system. And second, through this connection, you will install the rest of the EmbeddedView or CEView software.

To install EmbeddedView or CEView:

1. Turn on the target system and make sure it is connected to your TCP/IP network.

If Remote Agent is already installed on the target system, then it will automatically run at start up, and you may skip the next step. Many OEMs preinstall Remote Agent on their devices, as part of a larger InduSoft Web Studio package.

2. Copy the Remote Agent utility to the target system and then run it.
  - a) Locate the correct version of the Remote Agent utility (`CEServer.exe`) for the target system. All versions are stored in your InduSoft Web Studio program folder.

Remote Agent for Windows Embedded Standard is located at:

```
C:\Program Files\InduSoft Web Studio v7.1\Redist  
\WinEmbedded\Bin\CEServer.exe
```

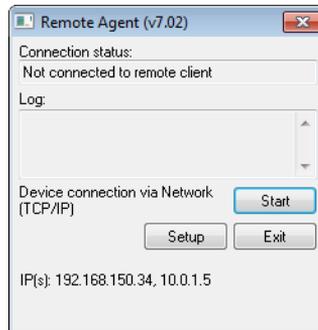
Remote Agent for Windows Embedded Compact is located at:

```
C:\Program Files\InduSoft Web Studio v7.1\Redist\WinCE  
5.0\processor\Bin\CEServer.exe
```

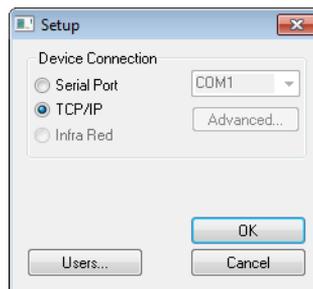
...where *processor* is the specific processor used by the target system. For more information, consult the manufacturer's documentation.

- b) Copy `CEServer.exe` to the target system by either downloading it over the network, transferring it on a USB flash drive, or syncing it with Microsoft ActiveSync. (ActiveSync is also known as Windows Mobile Device Center in Windows Vista or Zune Software in Windows 7.) You may save the file anywhere you want on the target system, as long as it is in permanent (i.e., non-volatile) memory.
  - c) Set Remote Agent to automatically run at start up by creating a link to it in `\Windows\Startup` on the target device.
  - d) Run Remote Agent.

The *Remote Agent* window is displayed.

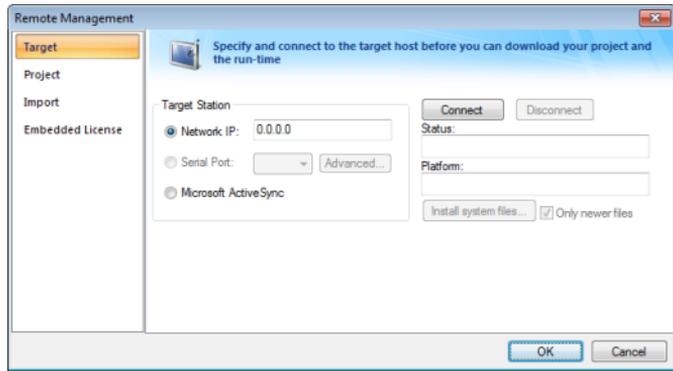


3. Configure the communication settings in Remote Agent.
  - a) Click **Setup**.  
The *Setup* dialog box is displayed.



- b) Select the appropriate connection type: **Serial Port** or **TCP/IP**. If you select **Serial Port**, then also select the COM port and verify the advanced serial communication settings.  
If you are already connected to the target system via ActiveSync, then you do not need to select another connection at this time. However, keep in mind how the target system will actually be used during project run time.
    - c) Click **OK**.
    - d) If you selected TCP/IP for the device connection, then note the IP address.
    - e) Make sure that you leave Remote Agent running on the target system.

- 4. Use the Remote Management tool to connect to the target system.
  - a) Run the project development application on your computer.
  - b) On the **Home** tab of the ribbon, in the **Remote Management** group, click **Connect**.  
The *Remote Management* dialog box is displayed.



- c) Select the appropriate connection type for the target system: **Network IP**, **Serial Port**, or **Microsoft ActiveSync**. If you select **Network IP**, then also type the IP address of the target system. If you select **Serial Port**, then also select the COM port and verify the advanced serial communication settings.
- d) Click **Connect**.  
If you are successfully connected, then the connection status is shown in the **Status** box and the target system's specifications are shown in the **Platform** box. If you are not connected, then check both the connection settings and the physical connections.

 **Note:** In some cases, the Remote Management tool may not be able to connect via Microsoft ActiveSync to a device running Windows CE 6.0 or later. This is because of a problem in the default configuration of Windows CE 6.0. You can fix the problem by using a small utility that is included with InduSoft Web Studio. The utility is located at:

**C:\Program Files\InduSoft Web Studio v7.1\Redist  
\ActiveSyncUnlock.exe**

Copy this file to the device using the stand-alone version of Microsoft ActiveSync and then execute the file on the device. It doesn't matter where on the device the file is located. When

this is done, try again to use the Remote Management tool to connect to the device.

5. Install the rest of the EmbeddedView or CEView software on the target system.
  - a) In *Remote Management*, click **Install system files**.

When the installation is finished, the target system's updated status is displayed in the **Status** box.

With EmbeddedView or CEView installed on the target system, you can now use it as a project runtime server and/or client.

## Execution modes

InduSoft Web Studio, EmbeddedView, and CEView support the following execution modes:

Execution Mode	InduSoft Web Studio	EmbeddedView / CEView
Evaluation Mode	+	-
Demo Mode	+	+
Licensed for Engineering Only	+	-
Licensed for Runtime Only	+	+
Licensed for Engineering + Runtime	+	-

### Evaluation Mode

Enables all of the product's engineering and runtime features.

The first time you install InduSoft Web Studio on a computer, the product runs for forty (40) hours in Evaluation Mode. This evaluation period includes any time you run a product module (engineering or runtime). You can use this evaluation period continuously or not; for example, 10 hours a day for 4 days, or 5 hours a day for 8 days, or 10 hours a day for 3 days plus 5 hours a day for 2 days, and so on.

After running for 40 hours in the Evaluation Mode, the evaluation period ends and the program automatically converts to Demo Mode until you apply a valid license (hardkey or softkey). You cannot reactivate Evaluation Mode, even if you reinstall the software on your computer.



**Note:** Each version of InduSoft Web Studio has an evaluation period that is independent of every other version. For example, if your InduSoft Web Studio v7.0 evaluation period has expired and you are running in Demo Mode because you have not installed a license, when you install InduSoft Web Studio v7.1 on the same computer, the newer version will begin its own 40-hour evaluation period and the older version will continue running in Demo Mode.

### Demo Mode

Allows you to download projects to remote stations and to run projects for testing or demonstration purposes. You can execute runtime tasks and use the debugging tools (*LogWin* and *Database Spy*), but they shut down automatically after running for two hours continuously. You can restart the Demo Mode again and run for another two hours, and so on.

You *cannot* create or modify screens, worksheets, or project settings in Demo Mode.

### Licensed for Engineering Only

Enables all workbench options for an unlimited time.

This mode also allows you to execute the runtime tasks and debugging tools (*Database Spy*, *Output* window, and *LogWin* module) for 24 hours continuously. After the 24-hour period these tasks shut down, but you can restart them again and run for another 24 hours, and so on. You can use this license for development and testing only.

### Licensed for Runtime Only

Enables you to run all runtime and debugging tools (*Database Spy*, *Output* window, and *LogWin* module) for unlimited time, but you cannot create or modify screens and/or worksheets.

The menu options available in Runtime Only mode are the same as the options listed for Demo Mode (see previous table).

### Licensed for Engineering + Runtime

Enables all engineering tools, runtime tasks, and debugging tools (*Database Spy*, *Output* window, and *LogWin* module) for an unlimited period of time.

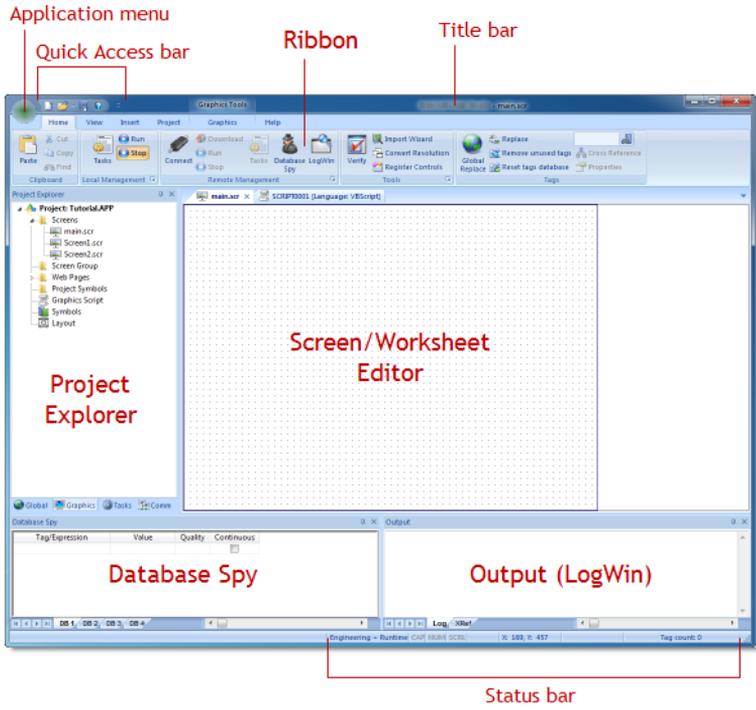


**Note:** The Remote Management tool is always available, regardless of the execution mode, so that you can upload files from or download files to remote stations.

To see which execution mode you are currently running, click **About** on the Help tab of the ribbon; the *About dialog* shows the execution mode, including the time remaining if you are in Evaluation Mode.

# The Development Environment

InduSoft Web Studio incorporates a modern, Ribbon-based Windows interface to provide an integrated and user-friendly development environment.



*The IWS Development Environment*

## Title Bar

The Title Bar located along the top of the development environment displays the application name (e.g., InduSoft Web Studio) followed by the name of the active screen or worksheet (if any).



### *Example of Title Bar*

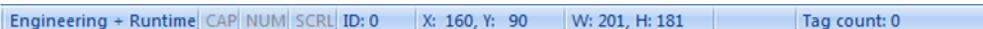
The Title Bar also provides the following buttons (from left to right):

- **Minimize** button : Click to minimize the development environment window to the Taskbar.
- **Restore Down / Maximize**: Click to toggle the development environment window between two sizes:
  - **Restore Down** button  reduces the window to its original (default) size.
  - **Maximize** button  enlarges the window to fill your computer screen.
- **Close** button : Click to save the database and then close the development environment. If you modified any screens or worksheets, the application prompts you to save your work. This button's function is similar to clicking **Exit Application** on the Application menu.

 **Note:** Closing the development environment does *not* close either the project viewer or the runtime system, if they are running.

## Status Bar

The Status Bar located along the bottom of the development environment provides information about the active screen (if any) and the state of the application.



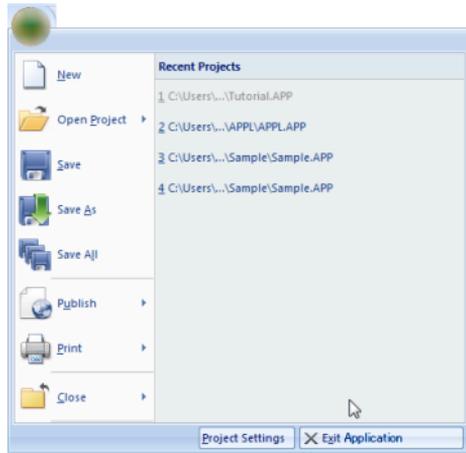
### Example of Status Bar

The Status Bar fields (from left to right) are described in the following table:

Field	Description
Execution Mode	The current <b>execution mode</b> of the application.
CAP	Indicates whether the keyboard <b>Caps Lock</b> is on (black) or off (grey).
NUM	Indicates whether the keyboard <b>Num Lock</b> is on (black) or off (grey).
SCRL	Indicates whether the keyboard <b>Scroll Lock</b> is on (black) or off (grey).
Object ID	The ID number of a selected screen object.
Cursor Position	The location of the cursor on the active screen or worksheet. If it's a screen, then the position of the <i>mouse</i> cursor is given as X,Y coordinates, where X is the number of pixels from the left edge of the screen and Y is the number of pixels from the top edge of the screen. If it's a worksheet, then the position of the <i>text</i> cursor is given as Line and Column.
Object Size	The size (in pixels) of a selected screen object, where W is the width and H is the height.
No DRAG	Indicates whether dragging is disabled ( <b>No DRAG</b> ) or enabled (empty) in the active screen.
Tag Count	The total number of tags used so far in the project.

## Application button

The Application button opens a menu of standard Windows application commands like New, Open, Save, Print, and Close.



*Application button opens menu of commands*

## Quick Access Toolbar

---

The Quick Access Toolbar is a customizable toolbar that contains a set of commands that are independent of the ribbon tab that is currently displayed.

### Move the Quick Access Toolbar

The Quick Access Toolbar can be located in one of two places:

- Upper-left corner next to the Application button (default location); or
- Below the ribbon, where it can run the full length of the application window.

If you don't want the Quick Access Toolbar to be displayed in its current location, you can move it to the other location:

1. Click **Customize Quick Access Toolbar** .
2. In the list, click **Show Below Ribbon** or **Show Above Ribbon**.

### Add a command to the Quick Access Toolbar

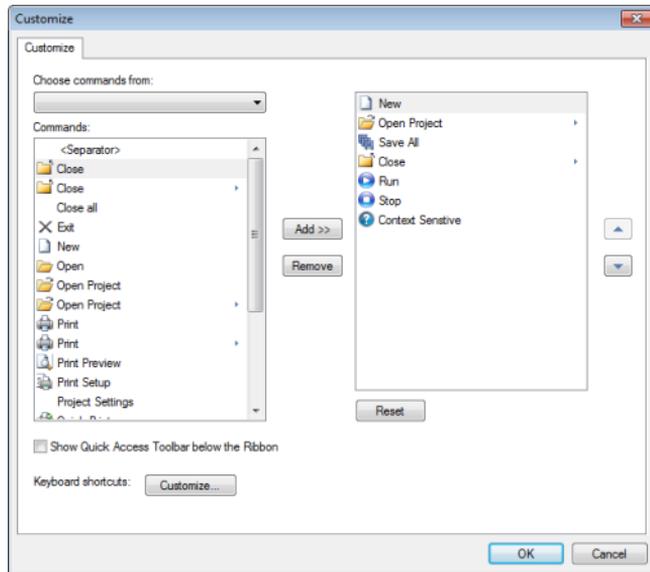
You can add a command to the Quick Access Toolbar directly from commands that are displayed on the ribbon:

1. On the ribbon, click the appropriate tab or group to display the command that you want to add to the Quick Access Toolbar.
2. Right-click the command, and then click **Add to Quick Access Toolbar** on the shortcut menu.

You can also add and remove commands — as well as reset the toolbar to its default — using the *Customize* dialog:

1. Click **Customize Quick Access Toolbar** .

2. In the list, click **More Commands**. The *Customize* dialog is displayed.



***Customize Quick Access Toolbar dialog***

3. In the **Choose commands from** menu, select the appropriate Ribbon tab. The commands from that tab are displayed in the **Commands** list.
4. In the **Commands** list, select the command that you want to add to the Quick Access Toolbar.
5. Click **Add**.

Only commands can be added to the Quick Access Toolbar. The contents of most lists, such as indent and spacing values and individual styles, which also appear on the ribbon, cannot be added to the Quick Access Toolbar.

## Ribbon

---

The new ribbon combines the numerous menus and toolbars from the previous version of IWS into a single, user-friendly interface. Almost all application commands are now on the ribbon, organized into tabs and groups according to general usage.



*The Ribbon interface*

### *Home tab*

The **Home** tab of the ribbon is used to manage your project within the development environment.



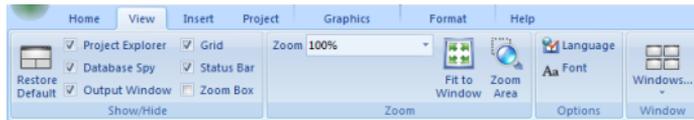
*Home tab of the ribbon*

The tools are organized into the following groups:

- **Clipboard:** [Cut](#), [copy](#), [paste](#), and [find](#) items in project screens and task worksheets.
- **Local Management:** [Run](#) and [stop](#) the project on the local station (i.e., where the development application is installed), as well as manage the [execution tasks](#).
- **Remote Management:** [Connect](#) to a remote station (e.g., a Windows Embedded device) so that you can download the project to it, and then [run](#), [stop](#), and [troubleshoot](#) the project on that station.
- **Tools:** Miscellaneous tools to [verify the project](#), [import tags](#) from other projects, [convert screen resolutions](#), and [register ActiveX and .NET controls](#).
- **Tags:** [Manipulate tags and tag properties](#) in the project database.

## View tab

The **View** tab of the ribbon is used to customize the look of the development environment itself.



### View tab of the ribbon

The tools are organized into the following groups:

- **Show/Hide:** Show and hide the different parts of the development environment, as well as restore the default layout.
- **Zoom:** [Zoom](#) in and out of the screen editor.
- **Options:** Change the [language](#) and [font](#) used in the development environment.
- **Window:** [Arrange the windows](#) in the development environment.

## Insert tab

The **Insert** tab of the ribbon is used to insert new tags, screens, worksheets, and other components into your project.



### Insert tab of the ribbon

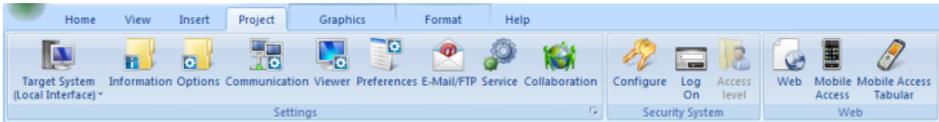
The tools are organized into the following groups:

- **Global:** Insert [tags](#), [classes](#), [translations](#), and [procedures](#) into the [Global tab](#) of the Project Explorer.
- **Graphics:** Insert [screens](#) and [screen groups](#) into the [Graphics tab](#) of the Project Explorer.
- **Task Worksheets:** Insert [task worksheets](#) into the [Tasks tab](#) of the Project Explorer.

- **Communication:** Insert [server configurations and communication worksheets](#) into the [Comm tab](#) of the Project Explorer.

## ***Project tab***

The **Project** tab of the ribbon is used to configure your project settings.



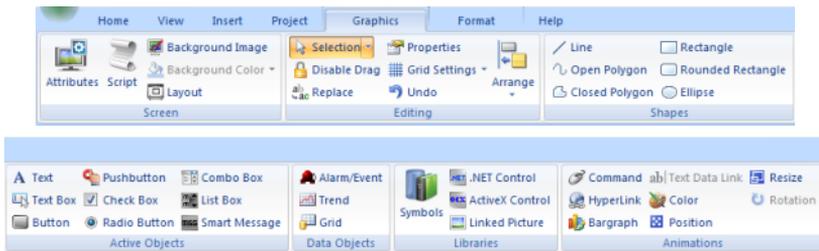
### ***Project tab of the ribbon***

The tools are organized into the following groups:

- **Settings:** Configure the general [project settings](#), set the project to [run as a Windows service](#), or enable workgroup collaboration and version control.
- **Security System:** Enable and configure the [project security system](#).
- **Web:** Configure the project to accept connections from [web thin clients](#) and [mobile devices](#).

## ***Graphics tab***

The **Graphics** tab of the ribbon is used to draw project screens.



### ***Graphics tab of the ribbon***

 **Note:** This tab is available only when you have a project screen open for editing.

The tools are organized into the following groups:

- **Screen:** Configure settings for the project screen itself, such as its [attributes](#), [script](#), and [background color or image](#).
- **Editing:** [Select and edit objects](#) in the project screen.
- **Shapes:** Draw [static lines and shapes](#).
- **Active Objects:** Draw [active objects](#), like buttons and check boxes.
- **Data Objects:** Draw [objects that display historical data](#), like alarms, events, and trends.
- **Libraries:** Select from libraries of premade objects, such as [symbols](#), [.NET](#) and [ActiveX controls](#), and [external picture files](#).
- **Animations:** Apply [animations](#) to other screen objects.

### ***Format tab***

The **Format** tab of the ribbon is used to format and arrange objects in a project screen.



***Format tab of the ribbon***

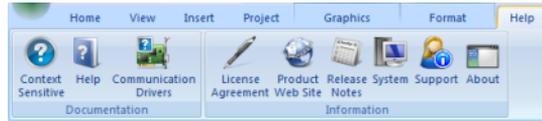
 **Note:** This tab is available only when you've selected one or more objects in a project screen.

The tools are organized into the following groups:

- **Arrange:** Arrange objects in a project screen, including [bring to front and send to back](#), [group](#), [align](#), and [rotate](#).
- **Position:** Precisely adjust the [position](#) of a screen object in a project screen.
- **Size:** Precisely adjust the [size](#) of a screen object.
- **Style:** Change the [fill](#) and [line color](#) of a screen object.
- **Fonts:** Change the [caption font](#) of a screen object.

## ***Help tab***

The **Help** tab of the ribbon provides additional help with using the software.



### ***Help tab of the ribbon***

The tools are organized into the following groups:

- **Documentation:** Access the documentation for the development application, including this [help file / technical reference](#) and notes for the individual [communication drivers](#).
- **Information:** Access other information about InduSoft Web Studio, including the [license agreement](#), [product website](#), and [release notes](#), as well as [system](#) and [support](#) details that make it easier for Customer Support to assist you.

## Project Explorer

The Project Explorer organizes all of the screens, worksheets, and other items that comprise your project and presents them in an expandable tree-view.

To open a folder and view its contents, either click the Expand icon  to the left of the folder or double-click the folder itself.

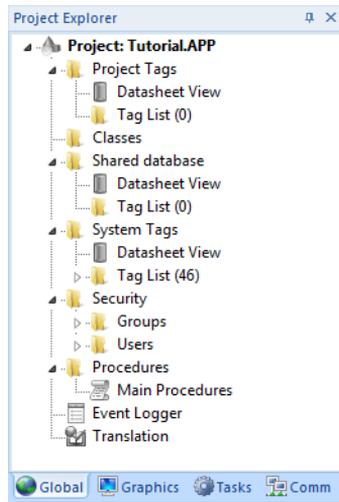
To close a folder, click the Collapse icon  to the left of the folder.

If you right-click any item in the Project Explorer, then a shortcut menu will appear with contextual commands for that item.

There are four main sections, or tabs, in the Project Explorer: Global, Graphics, Tasks, and Comm.

### ***Global tab***

The Global tab of the Project Explorer contains the project tags database, as well as other features that apply to the entire project such as the security system, VBScript procedures, and UI translation.



***Global tab of the Project Explorer***

The folders on the Global tab are described in the following sections:

### **Project Tags**

The project tags database contains all of the data tags that you create during project development, such as screen tags (e.g., `button1_state`) or tags that read from / write to connected devices.

### **Classes**

Classes are compound tags that you can create to associate a set of values, rather than a single value, with an object. For example, where you may normally create separate tags for a tank's pressure, its temperature, and its fill level, you can instead create a "tank" class that includes all three.

### **Shared Database**

The shared database contains tags that were created in another program and then imported into or integrated with your project.

### **System Tags**

System tags are predefined values such as the date, the time, the name of the current user, and so on. You can use these values to develop supervisory functions and housekeeping routines.

All system tags are read-only, which means you cannot add, edit, or remove these tags from the database.

### **Security**

If you choose to enable it, you can use the project security system to control who may log on to your project and what they may do during runtime.

### **Procedures**

Procedures are VBScript functions and sub-routines that can be called by any other script in your project.

### **Event Logger**

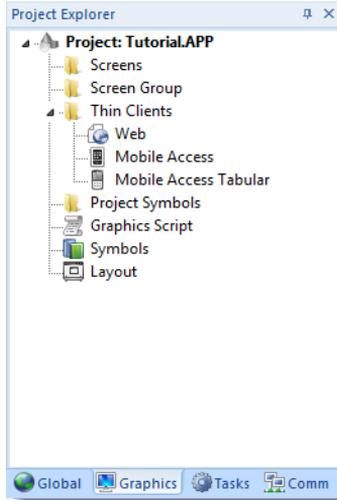
The event logger saves important runtime messages and task results to an external database.

### **Translation**

You can use the translation table to develop a multilingual user interface (MUI) for your project.

## Graphics tab

The Graphics tab of the Project Explorer contains all of the screens, screen groups, and symbols in your project.



*Graphics tab of the Project Explorer*

The folders on the Graphics tab are described in the following sections:

### Screens

You create screens to provide a graphical interface for your project. Each screen can contain many buttons, sliders, dials, indicators, graphs, and so on.

### Screen Groups

You can combine individual screens into screen groups, so that they all open together at the same time.

### Thin Clients

You can deploy your project as a web application to be accessed by thin clients such as desktop web browsers, tablets, and smartphones. You can even deploy different versions of your project with different levels of functionality for each type of client.

### Project Symbols

This folder contains all of the custom symbols that you create for your project. A symbol is a group of interconnected screen objects that work together to perform a single function — for example, lines, rectangles, and text fragments that have been arranged to make a slider control.

### **Graphics Script**

You can use this worksheet to define VBScript sub-routines that are called only when the graphics module starts (i.e., when a client station connects to the server and displays the graphical interface), while it is running, and when it ends.

### **Symbols**

The symbols library contains not only the custom symbols that you create (see Project Symbols above), but also a large selection of premade symbols that are installed with the development application.

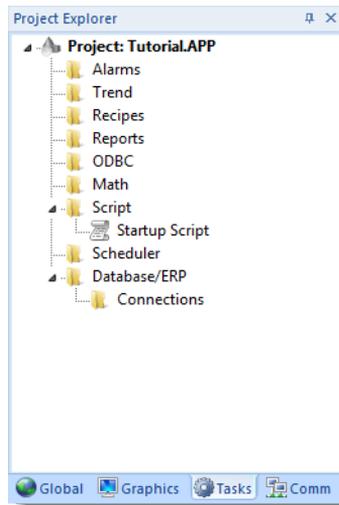
### **Layout**

The layout editor displays all of the screens the are currently open for editing. You can use it to visualize how the screens are arranged together and reuse screens in multiple layouts — for example, to create a common navigation bar across your entire project.

### **Tasks tab**

The Tasks tab of the Project Explorer organizes the worksheets that are processed as background tasks (i.e., server-based maintenance tasks that

are not directly related to screen operations or device I/O) during project runtime.



*Tasks tab of the Project Explorer*

The folders on the Tasks tab are described in the following sections:

### **Alarms**

You can use Alarm worksheets to define when alarms are triggered, how they must be handled, and what messages they generate.

(You can then use the Alarm/Event Control screen object to display your alarms on screen, but that is a separate procedure.)

### **Trends**

You can use Trend worksheets to select project tags that should be displayed as data trends and/or saved as historical data.

(You can then use the Trend Control screen object to actually display your trends on screen, but that is a separate procedure.)

### **Recipes**

You can use Recipe worksheets to select project tags that will load values from and/or save values to an external file. These

worksheets are typically used to execute process recipes, but you can store any type of information such as passwords, operation logs, and so on.

(You can then call the `Recipe` function to actually run a configured Recipe worksheet, but that is a separate procedure.)

### **Reports**

You can use Report worksheets to design runtime reports that are either sent to a printer or saved to disk.

(You can then call the `Report` function to actually run a configured Report worksheet, but that is a separate procedure.)

### **ODBC**

You can use ODBC worksheets to set up connections and exchange data with other ODBC-compliant databases.

### **Math**

You can use Math worksheets to develop complex runtime logic using the built-in scripting language.

### **Script**

You can use Script worksheets to develop complex runtime logic using VBScript.

### **Scheduler**

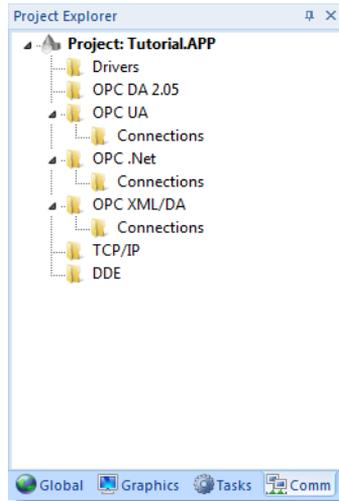
You can use Scheduler worksheets to run commands at specified times, dates, or trigger events.

### **Database**

You can use Database worksheets to set up connections and exchange data with external databases using the standard ADO.NET interface (as an alternative to ODBC).

## **Comm tab**

The Comm tab of the Project Explorer organizes the worksheets that control communication with remote devices, using either direct communication drivers or other common protocols.



***Comm tab of the Project Explorer***

The folders on the Comm tab are described in the following sections:

### **Drivers**

You can use Driver worksheets to communicate with PLCs and other hardware, using any of the hundreds of direct communication drivers that are installed with the development application.

### **OPC DA 2.05**

You can use OPC worksheets to communicate with OPC servers via the OPC Classic protocol.

### **OPC UA**

You can use OPC UA worksheets to communicate with OPC servers via the new OPC Unified Architecture protocol.

### **OPC .Net**

You can use OPC .Net worksheets to communicate with OPC servers via the new OPC .NET 3.0 protocol (formerly OPC Xi).

### **OPC XML/DA**

You can use OPC XML/DA worksheets to communicate with OPC servers via the new OPC XML-DA protocol.

### **TCP/IP**

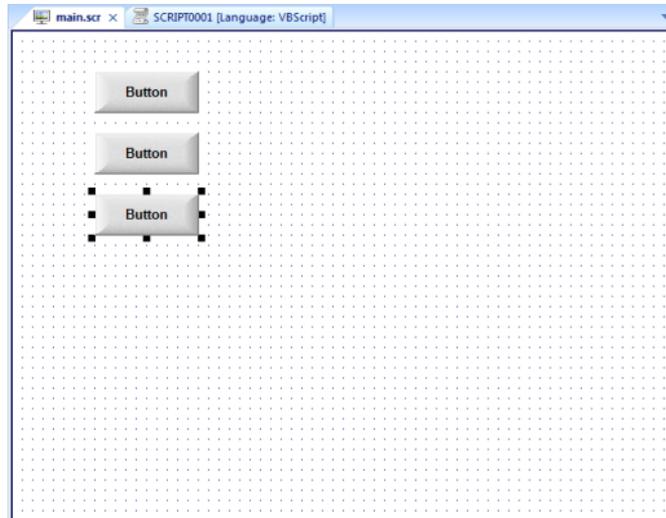
You can use TCP/IP worksheets to configure communication between your own project and other IWS projects. The TCP/IP Client and TCP/IP Server modules enable two or more projects to keep their databases synchronized using the TCP/IP protocol.

### **DDE**

You can use DDE worksheets to communicate with other Microsoft Windows applications, such as Microsoft Excel, that support the Dynamic Data Exchange protocol.

## Screen/Worksheet Editor

Use the powerful, object-oriented screen editor to create and edit a variety of screens and worksheets for your projects. You can input information using your mouse and keyboard, output control data to your processes, and automatically update screens based on data input from your processes.



*Screen/Worksheet Editor*

Other screen editor features include:

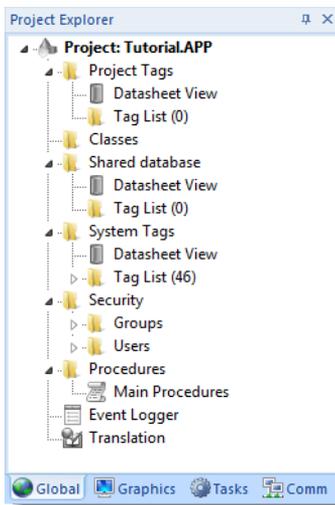
- Simple point-and-click, drag-and-drop interface
- Grouping objects to preserve the construction steps of individual objects
- Editing objects without having to ungroup internal object components or groups
- Handling bitmap objects and background bitmaps
- Status line support in project windows and dialogs

## About Tags and the Project Database

---

Tags are a core component of any IWS project. Simply put, tags are variables used by IWS to receive and store data obtained from communication with plant floor devices, from the results of calculations and functions, and from user input. In turn, tags can be used to display information on screens (and Web pages), to manipulate screen objects, and to control [runtime tasks](#).

But tags are more than simple variables. IWS includes a real-time database manager that provides a number of sophisticated functions such as time-stamping of any value change, checking tag values against runtime minimum and maximum values, comparing tag values to alarming limits, and so on. A IWS tag has both a value and various properties that can be accessed, some at development and others only at runtime.



All tags are organized into one of the following categories, which are represented by folders on the [Global tab](#) of the *Project Explorer*.

- **Project Tags** are tags that you create during project development. Places where project tags are used include:
  - Screen tags
  - Tags that read from/write to field equipment

- Control tags
- Auxiliary tags used to perform mathematical calculations
- **Shared Database** tags are created in a PC-based control program and then imported into IWS's tags database.

For example you might create tags in SteepleChase and import them into IWS so IWS can read/write data from a SteepleChase PC-based control product.

You cannot modify shared tags within IWS — you must modify the tags in the original PC-based control program, and then re-import them into the Tags database.

- **System Tags** are predefined tags with predetermined functions that are used for IWS supervisory tasks. For example,
  - Date tags hold the current date in string format
  - Time tags hold the current time in string format

Most system tags are *read-only*, which means you cannot add, edit, or remove these tags from the database.

To see a list of the system tags, select the **Global** tab in the *Project Explorer*, open the **System Tags** folder, and open the **Tag List** subfolder. The above figure shows a partial list of system tags.

After creating a tag, you can use it anywhere within the project, and you can use the same tag for more than one object or attribute.

## Understanding the Tag Name Syntax

---

Observe the following guidelines when naming a tag:

- Your tag names **must be unique** — you cannot specify the same name for two different tags (or functions). If you type an existing tag name, IWS recognizes that the name exists and will not create the new tag.
- You must begin each tag name with a *letter*. Otherwise, you can use letters, numbers, and the underscore character (`_`) in your tag name.
- You *cannot* use the following symbols in a tag name:

`` ~ ! @ # $ % ^ & * ( ) - = \ + \ [ ] { } < > ?`

- You can use a maximum of 255 characters for a tag name or a class member name. You can use uppercase and lowercase characters. Tag names are *not* case sensitive. Because IWS does not differentiate between uppercase and lowercase characters, you can use both to make tag names more readable. (For example: **TankLevel** instead of **tanklevel**.)
- Tag names must be different from system tag names and math functions.



**Note:** Use the @ character at the beginning of a tag name to indicate that the tag will be used as an [indirect tag](#) in the project.

Some valid tag examples include:

- **Temperature**
- **pressure1**
- **count**
- **x**

---

## Choosing the Tag Data Type

---

Another consideration when designing a tag is what type of data the tag will receive. IWS recognizes the following, standard tag *data types*:

- **Boolean** (*one bit*): Simple boolean with the possible values of 0 (false) and 1 (true). Equivalent to the "bool" data type in C++. Typically used for turning objects off and on or for closing and opening objects.
- **Integer** (*four bytes*): Integer number (positive, negative, or zero) internally stored as a signed 32-bit. Equivalent to the "signed long int" data type in C++. Typically used for counting whole numbers or setting whole number values. Examples: 0, 5, -200.
- **Real** (*floating point, eight bytes*): Real number that is stored internally as a signed 64-bit. Equivalent to the "double" data type in C++. Typically used for measurements or for decimal or fractional values.
- **String** (*alphanumeric data, up to 1024 characters*): Character string up to 1024 characters that holds letters, numbers, or special characters. Supports both ASCII and UNICODE characters. Examples: **Recipe product X123, 01/01/90, \*\*\* On \*\*\***.

You can also make a tag into a compound tag by assigning it a [Class](#). A Class is a template consisting of two or more tag definitions, each with its own data type. You can use Classes in projects that have items (e.g., tanks of liquid) with multiple attributes (e.g., fill level, temperature, pressure) to be monitored or controlled.

You can find these tag types (and their respective icons) in the [Global tab](#) of the *Project Explorer*.

**See also:** [Understanding Tag Properties and Parameters](#)

### Changing How Boolean Tags Receive Numeric Values

By default, if any numeric value other than 0 (i.e., ≠0) is written to a Boolean tag, then the tag automatically assumes a value of 1. You can change this behavior, if necessary, by editing the `project_name.app` file to change the following setting:

```
[Options]
BooleanTrueAboveZero=value
```

If `BooleanTrueAboveZero` is set to the default 0, then the project will behave as described above. If `BooleanTrueAboveZero` is set to 1, then the project will behave as follows:

- When you write any numeric value less than or equal to 0 (i.e.,  $\leq 0$ ) to a Boolean tag, the tag assumes a value of 0 (false).
- When you write any numeric value greater than 0 (i.e.,  $> 0$ ) to a Boolean tag, the tag assumes a value of 1 (true).



**Caution:** This is a global runtime setting. If you only want to change how certain tags are handled, then you should not change this setting.

## Using Array Tags

IWS tags can consist of a single value or an array of values.

 **Note:** The maximum array size is 16384 as long as it does not exceed the maximum number of tags supported by the license (Product Type) selected for the project. Each array position (including the position 0) counts as one tag for licensing restrictions, because each position has an independent value.

An array tag is a set of tags with the same name, which is identified by indexes (a matrix of  $n$  lines and 1 column). The maximum array size depends on the product specification. You can use the following syntax to access an array tag:

***ArrayTagName***[***ArrayIndex***]

For example: **tank**[0], **tank**[1], **tank**[2], and **tank**[500].

 **Caution:** You must specify a maximum index for each array tag in the **size** column of any datasheet. You can specify  $n$  to indicate the array tag has positions from 0 to  $n$ . For example, if the size of TagA is 3, the tag elements could be **TagA**[0], **TagA**[1], **TagA**[2], and **TagA**[3].

Use the array tag whenever possible because it optimizes memory use and simplifies the configuration task. For example, if you want a display to monitor each tank, you could use array tags to configure a single display containing tags linked to any tank. For example (using the **tk** tag as an index containing the number of the tank): **pressure**[**tk**], **temperature**[**tk**], and **temperature**[**tk+1**].

An array index can be a tag, a numeric value, or an expression with the arithmetic operator "+".

 **Note:** When you refer to an array with an index using the + arithmetic operation, you must use the following syntax:

***ArrayTagName***[***NumValue1+NumValue2***]

Where *NumValue1* and *NumValue2* can be an integer tag or a numerical constant. For example: `temperature[tk+2]` or `temperature[tk+6]`.

Using array tags in any IWS task can save a significant amount of project development time. For example, if you needed tag points related to the temperature of four tanks. The conventional configuration method is the following:

- **temperature1**: high temperature on tank 1
- **temperature2**: high temperature on tank 2
- **temperature3**: high temperature on tank 3
- **temperature4**: high temperature on tank 4

Using array tags simplifies this task, as follows:

- **temperature[j]**: high temperature on tank {j}

 **Note:** When you create a four-position array tag, the system creates five positions (from 0 to 4). For example:

```
tag_example[15] //start position=0, end position=15
```

Therefore, the `tag_example[15]` array has 16 elements.

When using another tag to reference the index of an array, if the value of the tag is outside the size of the array, then the following results are given:

- If *IndexTag* is greater than the size of the array, then `MyArray[IndexTag]` will point to the end position of the array; and
- If *IndexTag* is less than 0, then `MyArray[IndexTag]` will point to the start position of the array.

## Array Tags

An *array* tag consists of a set of tags that all have the same name, but use unique array indexes (a matrix of *n* lines and one column) to differentiate between each tag. An *array index* can be a fixed value, another tag or an expression. Maximum array sizes are determined by product specifications.

You can use array tags to:

- Simplify configurations
- Enable multiplexing in screens, recipes, and communication interfaces

- Save development time during tag declaration

You specify array tags in one of two formats:

- For a simple array tag, type:

***ArrayTagName*** [***ArrayIndex***]

- For a complex array tag (where the array index is an expression consisting of a tag and an arithmetic operation), type:

***ArrayTagName*** [***ArrayIndex+c***]

Where:

- ***ArrayTagName*** is the tag name;
- [***ArrayIndex***] is the unique index (fixed value or another tag);
- + is an arithmetic operation; and
- ***c*** is a numerical constant.



#### Note:

- You must specify a maximum index for each array tag by typing a value (*n*) in the Array Size column of an *Project Tags* datasheet or in the Array Size field on a *New Tag* dialog. (See "[Creating project database Tags](#)").

When you create an *n*-position array tag, IWS actually creates ***n+1*** positions (from 0 to *n*). For example, if you specify ***ArrayTag***[15], the array will have 16 elements, where 0 is the start position and 15 is the end position.

- You must not use spaces in an array tag.

When IWS reads a tag it begins with the first character and continues until it finds the first space or null character. Consequently, the system does not recognize any characters following the space as part of the array tag.

For example, if you type ***a***[***second + 1***], IWS regards ***a***[***second*** as the tag and considers it invalid because IWS does not find (recognize) the closing bracket. However, if you type ***a***[***second +1***], this is a valid array tag.

You can specify an array tag wherever you would use a variable name. Also, because array tags greatly simplify configuration tasks and can save development time, we suggest using them whenever possible.

For example, suppose you want to monitor the temperature of four tanks. The conventional configuration method is:

- **temperature1** — high temperature on tank 1
- **temperature2** — high temperature on tank 2
- **temperature3** — high temperature on tank 3
- **temperature4** — high temperature on tank 4

You can use array tags to simplify this task as follows (where [n] represents the tank number):

- **temperature[n]** — high temperature on tank [n]

The following table contains some additional examples of an array tag:

### Array Tag Examples

Array Tag Example	Description
<b>Tank [1], Tank [2], Tank [500]</b>	Simple arrays, where the array indexes (1, 2, and 500) are numerical constants. For example, tank numbers.
<b>Tank [tk]</b>	A simple array, where the array index (tk) is a tag. For example, a tag representing the tank number.
<b>Tank [tk+1]</b>	A complex array, where the array index (tk+1) is an expression. For example, the value of tk (tank number) plus 1.

 **Note:** When using another tag to reference the index of an array, if the value of the tag is outside the size of the array, then the following results are given:

- If **IndexTag** is greater than the size of the array, then **MyArray[IndexTag]** will point to the end position of the array; and
- If **IndexTag** is less than 0, then **MyArray[IndexTag]** will point to the start position of the array (i.e., **MyArray[0]**).

## About indirect tags

IWS supports indirect access to tags in the database. For example, consider a tag **X** of the String type. This tag can hold the name of any other tag in the [database](#) (that is, it can provide a pointer to any other type of tag, including a class type). The syntax for an indirect tag is straightforward: **@IndirectTagName**. For example, assume that a tag named **X** holds a "TEMP" string. Reading and/or writing to **@X** provides access to the value of the **TEMP** variable.



**Note:** Any tag created as a string-type tag is potentially an indirect tag (pointer).

To refer to a class-type tag, you can declare a string-type tag that points to a class tag. For example:

Class	<b>TANK</b> with members <b>Level</b>
Tag	<b>TK</b> of the class <b>TANK</b>
Tag	<b>XCLASS</b> of the String type

To access the **TK.Level** value, you must store the "**TK.Level**" value within the **XCLASS** tag and use the syntax, **@XCLASS**. You can also refer to a member of a class-type tag directly; identifying a class-type that points to a class member.

For example:

Class	<b>TANK</b> with members <b>Level</b>
Tag	<b>TK</b> of the class <b>TANK</b>
Tag	<b>XCLASS</b> of the class <b>TANK</b>

To access the **TK.Level** value, you must store the "**TK**" value within the **XCLASS** tag and use the syntax, **@XCLASS.Level**.

When creating tags for indirect use, place an **X** in the tag column rather than creating them as strings. For the type, write the type of tag for which you are creating a reference. Follow the **XCLASS** example: **@Z Integer**, **@X Class:TANK**.

## Indirect Tags

Indirect tags "point" to other database tags (including class-type tags). Using indirect tags can save development time because they keep you from having to create duplicate tags (and the logic built into them).

You create an indirect tag from any string-type tag simply by typing the @ symbol in front of the tag name **@TagName**.

- To reference a simple tag, assume the **strX** tag (a string tag) holds the value "**Tank**", which is the name of another tag, then reading from or writing to **@strX** provides access to the value of the **Tank** tag.
- To reference a class-type tag and member, you simply create a string tag that points to the class tag and the member. For example, if a tag **strX** (a string tag) holds the value "**Tank.Level**", which is the name of the class tag, then reading from or writing to **@strX** provides access to the value of the **Tank.Level** member.
- You can also point directly to a class-type tag member; by identifying a class-type that points to a class member. For example: to access the **Tank.Level** member of the class, you must store the "**Tank**" value within the **strX** tag and use the syntax, **@strX.Level**.

## **Tutorial: Building a Simple Project**

---

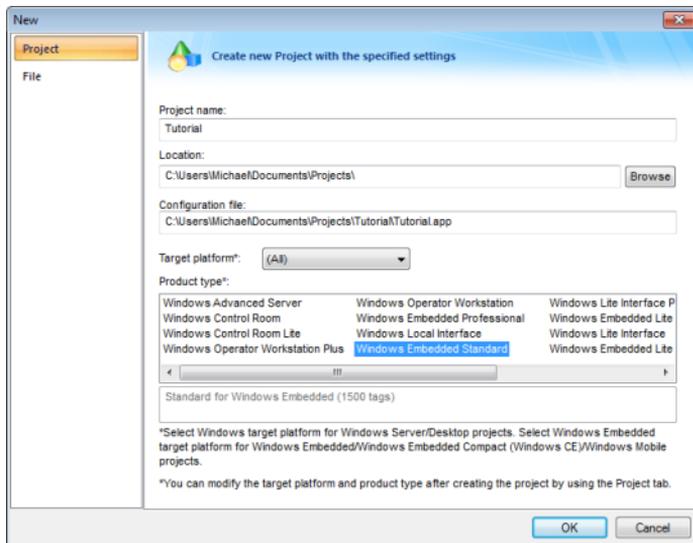
This section explains, using a step-by-step tutorial, how to build a simple project, as well as how to select and configure an I/O driver.

## Creating a new project

---

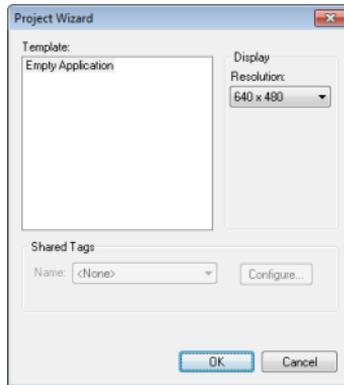
This part of the tutorial shows how to create a new project, including how to give it a name and select the target platform.

1. Click the **Application** button in the top-left corner of the development environment, and then click *New* on the **Application** menu.. The *New* dialog is displayed.
2. Click the **Project** tab.
3. In the **Project name** box, type the name of your project.  
For this tutorial, type `Tutorial`.  
The development application automatically creates a new directory of the same name and assigns your project file to that directory. (Notice the **Configuration file** text box in the figure.) To put your project file somewhere other than in the default projects folder, click **Browse** and navigate to the preferred location.
4. In the **Product type** list, select the type of project that you want to build.  
For this example, select **Windows Embedded Standard**. This is a tag and feature-limited product type that can be safely deployed on Windows Embedded devices.



**Selecting the target platform and product type**

5. Click **OK**.  
The *New* dialog is closed and the *Project Wizard* dialog is displayed.
6. In the **Template** list, select **Empty Application**.
7. In the **Resolution** list, select **640 x 480**.



***Specifying an empty Application with 640x480 resolution***

8. Click **OK**.  
The *Project Wizard* dialog is closed and the new project is created in the development environment.

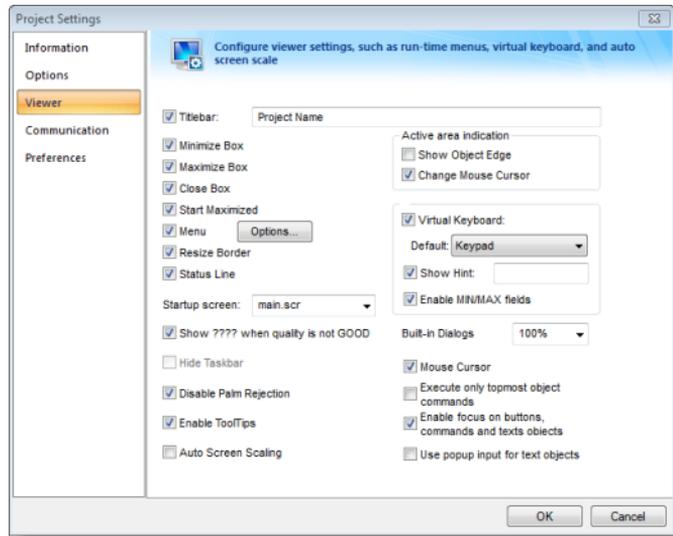
## Specifying the startup screen

---

This part of the tutorial shows how to open the project settings and then specify which screen should be displayed on startup.

- Use the **Information** tab to provide information that identifies the project (such as project description, revision number, Company name, Author's name, field equipment, and general notes).
  - Use the **Options** tab to specify generic settings for the project, such as the Target System, Automatic Translation, Alarm history and Events, Default Database and Shared Tags.
  - Use the **Viewer** tab to enable/disable the runtime desktop parameters.
  - Use the **Communication** tab to specify communication parameters relating to the project in general.
  - Use the **Web** tab to specify the Web Solution settings, such as the Data Server IP address.
  - Use the **Preferences** tab to enable/disable warning messages when using the development application.
1. On the **Project** tab of the ribbon, in the **Settings** group, click **Viewer**. The *Project Settings* dialog is displayed with the Viewer tab selected.

2. In the **Startup screen** box, type `main.scr`.



### *Specifying the startup screen*

When you run the project, it will automatically display the main screen (or whichever screen you specify) first. You can specify a screen *before* you create it.

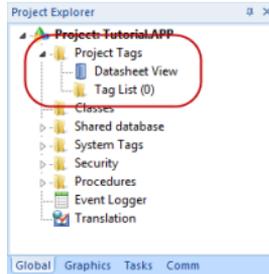
3. Click **OK**.

## Creating tags

---

This part of the tutorial shows how to create new tags by adding them to the Project Tags datasheet.

A tag is any variable that holds a value. All tags created in an project are stored in the Project Tags folder, on the Global tab of the Project Explorer.



*Project Tags folder*

1. In the Project Explorer, click the **Global** tab.
2. Double-click **Project Tags** to expand the folder.
3. Double-click **Datasheet View** to open the *Project Tags* datasheet.
4. Use the following parameters to create a tag for the sample project.
  - a) **Name**: Specify a unique tag name. For this tutorial, type `Level1`.
  - b) **Array**: Specify the top array index of the tag. (Simple tags have an Array of 0.) For this tutorial, type 3.

Each array index relates to one of the three tanks:

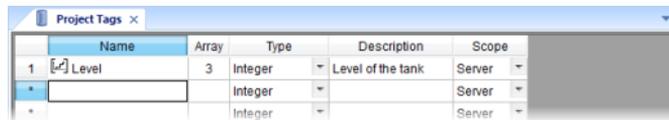
- `Level1[1]` is the level of Tank #1
- `Level1[2]` is the level of Tank #2
- `Level1[3]` is the level of Tank #3

You will not use `Level1[0]` in this tutorial, even though it is a valid tag.

- c) **Type**: Specify the data type of the tag: Boolean, Integer, Real, String, or Class. For this tutorial, select **Integer**.
- d) **Description** (optional): Type a description of the tag for documentation purposes only.

- e) **Scope:** Specify how the tag is managed between the Server and the Thin Client stations.
- Select **Local** if you want the tag to have independent values on the Server and Client stations.
  - Select **Server** if you want the tag to share the same value on the Server and Client stations.

For this tutorial, select **Server**.



	Name	Array	Type	Description	Scope
1	Level	3	Integer	Level of the tank	Server
*			Integer		Server
*			Integer		Server

### ***Creating the Level tag***

5. Save and close the *Project Tags* datasheet.

You will create additional tags as you build the project.

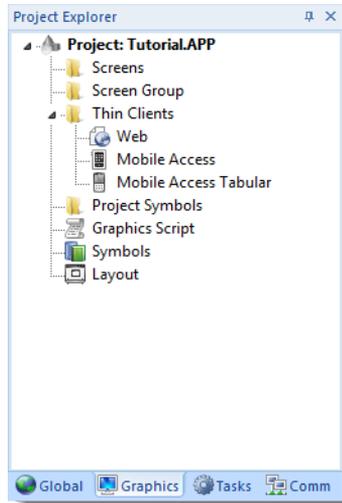
 **Tip:** You can sort the data in the *Project Tags* datasheet or insert/remove additional columns by right-clicking on it and then choosing the applicable option from the pop-up menu.

## Creating the startup screen

---

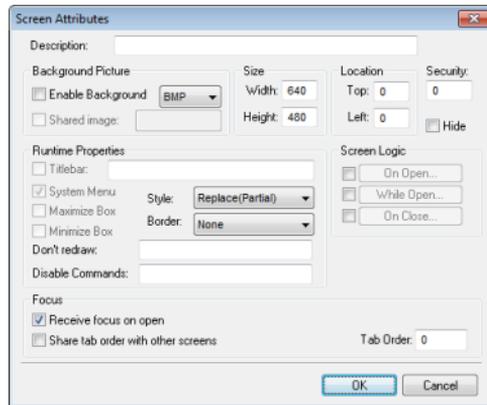
This part of the tutorial shows how to create your first screen, which will contain a single button that opens another screen.

1. In the *Project Explorer*, click the **Graphics** tab.



2. Right-click **Screens**, and then click **Insert** on the shortcut menu.  
The development application stores all screens created for an project in this Screens folder.

The *Screen Attributes* dialog is displayed.



**Screen Attributes dialog**

3. Use this dialog to set screen properties such as size and type. For this tutorial, click **OK** to accept the default settings. The *Screen Attributes* dialog is closed, and the new screen is opened in the workspace for editing.
4. On the **Graphics** tab of the ribbon, in the **Screen** group, click **Background Color**. A standard color picker is displayed.
5. In the color picker, select a light gray color.



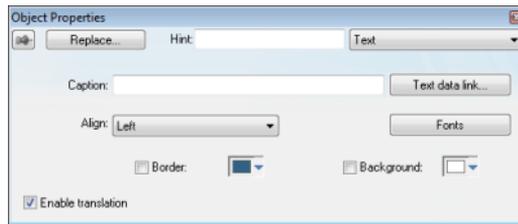
**Color picker**

That color is applied to the screen.

## Drawing the startup screen's title

This part of the tutorial shows how to draw the startup screen's title using a Text object.

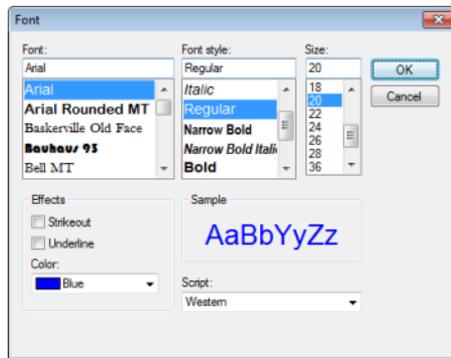
1. On the **Graphics** tab of the ribbon, in the **Active Objects** group, click **Text**. Your mouse cursor changes from an arrow to a crosshair.
2. Click on the screen, type `Welcome to the Tutorial Application`, and then press `Return`. This creates a new Text object with the specified text.
3. Double-click the object to open its *Object Properties* dialog.



**Object Properties: Text dialog**

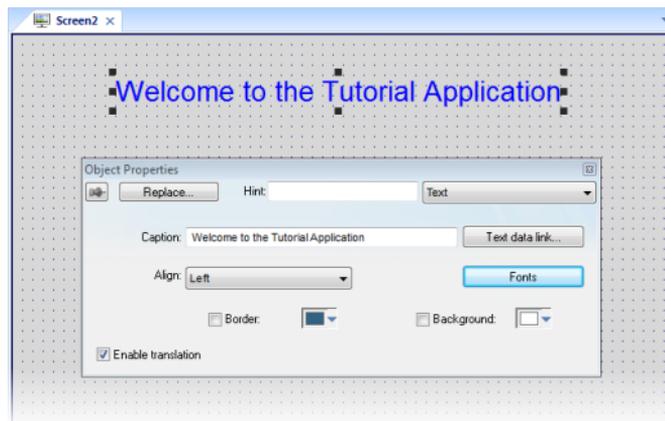
- Double-clicking on any screen object opens an *Object Properties* dialog containing the properties for that object. The properties shown in the dialog change depending on the type of object.
- The *Object Properties* dialog also contains a pin button that controls whether this dialog remains open. The button changes state (and function) each time you click on it, as follows:
  - When the pin button is released, the focus is passed to the object on the screen as soon as it is selected. It is recommended that this button is kept released when you want to manipulate the objects (Copy, Paste, Cut, or Delete). Although the *Object Properties* dialog is on the top, the keyboard commands (**Ctrl+C**, **Ctrl+V**, **Ctrl+X**, or **Del**) are sent directly to the objects.
  - When the pin button is pressed, the focus is kept on the *Object Properties* dialog, even when you click the objects on the screen. We recommend you keep this button pressed when you want to modify the settings of the objects. You can click an object and type the new property value directly in the *Object Properties* dialog (it is not necessary to click on the window to bring focus to it). Also, when the pin button is pressed, the *Object Properties* dialog does not automatically close when you click on the screen.

- Click **Fonts** to open *Font* dialog, and then specify the font settings. For this tutorial...
  - Font is **Arial**
  - Font style is **Regular**
  - Size is **20**
  - Color is **Blue**



**Specifying the font settings**

- Click **OK** to close the *Font* dialog. The font settings are applied to the Text object.



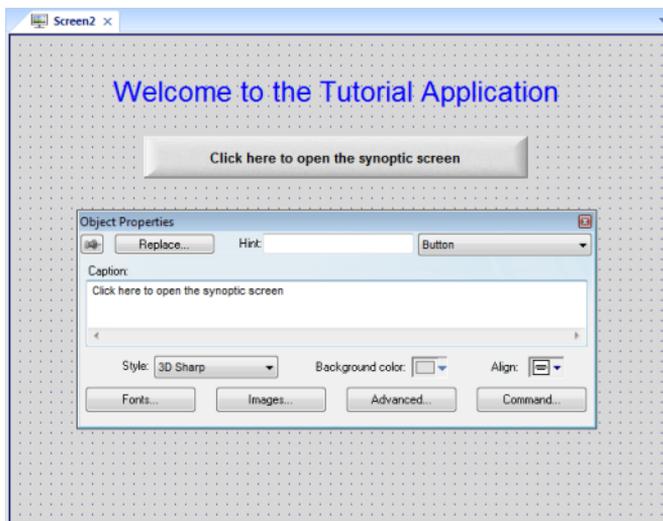
**Font settings applied to Text object**

6. Close the *Object Properties* dialog (i.e., click the Close button in the dialog's top-right corner).

### ***Drawing a button to open another screen***

This part of the tutorial shows how to draw and configure a button that will open another screen.

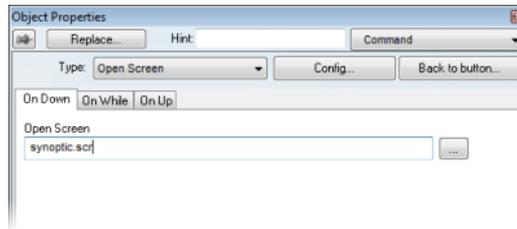
1. On the **Graphics** tab of the ribbon, in the **Active Objects** group, click **Button**. Your mouse cursor changes from an arrow to a crosshair.
2. Click and hold on the screen, and then drag the cursor to draw the Button object.
3. Double-click the object to open its *Object Properties* dialog.
4. In the **Caption** box, type the following text: Click here to open the synoptic screen.



### ***Adding a caption to the button***

5. Click **Command**.  
The *Object Properties* dialog changes to show the properties for the Command animation.
6. In the **Type** list, select **Open Screen**.

7. In the **Open Screen** box, type `synoptic.scr`.



### **Configuring an Open Screen command on the button**

You can specify a screen that you have not yet created.

8. Close the *Object Properties* dialog.

## **Saving and closing the startup screen**

This part of the tutorial shows how to properly save and close a screen.

1. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu. A standard Windows *Save* dialog is displayed.
2. In the **File name** box, type `main.scr`.
3. Click **Save**.  
The file is saved in your project folder (`\project_name\Screen\main.scr`), and the *Save* dialog is closed.
4. Click the Application button at the top-left of the development application, and then click **Close** on the Application menu.

## Creating the synoptic screen

---

This part of the tutorial show how to create your second screen, which will include an animated tank of liquid and some basic controls for that tank.

1. In the *Project Explorer*, click the **Graphics** tab.
2. Right-click the **Screens** folder, and then click **Insert** on the shortcut menu. The *Screen Attributes* dialog is displayed.
3. Use this dialog to set attributes such as size and type. For this tutorial, click **OK** to accept the default settings.
4. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu. A standard Windows *Save* dialog is displayed.
5. In the **File name** box, type `synoptic.scr`.
6. Click **Save**.  
The file is saved in your project folder (`\project_name\Screen\synoptic.scr`), and the *Save* dialog is closed.

### ***Drawing the synoptic screen's title***

As in a previous part, this part of the tutorial shows how to draw the synoptic screen's title using a Text object.

1. On the **Graphics** tab of the ribbon, in the **Active Objects** group, click **Text**.
2. Click on the screen, type `Synoptic Screen`, and then press Return.
3. Double-click the object to open its *Object Properties* dialog.
4. Click **Fonts** to open *Font* dialog, and then specify the font settings. For this tutorial...
  - Font is **Arial**
  - Font style is **Bold**
  - Size is **20**
  - Color is **Blue**
5. Close the *Object Properties* dialog.
6. Move the Text object to the top left corner of the screen.
7. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu.

This figure shows how your screen should look after you've created the date and time objects.

### Synoptic Screen

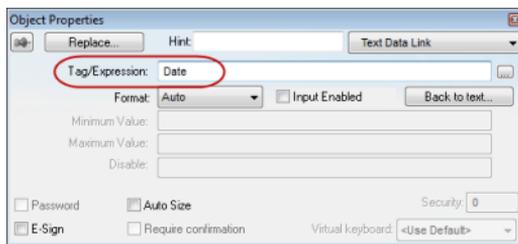
*Finished screen title*

## Drawing "Date" and "Time" displays

This part of the tutorial shows how to draw "Date" and "Time" displays by linking Text objects to system tags.

**Date** and **Time** are system tags that hold the current date and time of the local station. These tags are available to any project.

1. On the **Graphics** tab of the ribbon, in the **Active Objects** group, click **Text**.
2. Click on the screen, type Date: #####, and then press Return.
3. Double-click the object to open its *Object Properties* dialog.
4. Click **Text Data Link**.  
The *Object Properties* dialog changes to show the properties for the Text Data Link animation.
5. In the **Tag/Expression** box, type Date.



### Specifying the Date system tag

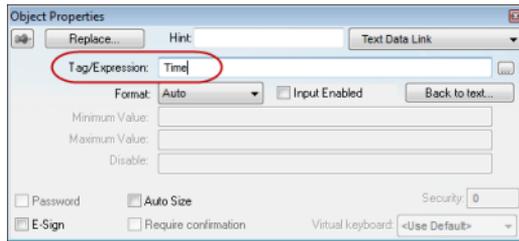
During runtime, the project replaces the ##### characters of the Text object with the value of the system tag **Date**.

6. Close the *Object Properties* dialog.
7. On the **Graphics** tab of the ribbon, in the **Active Objects** group, click **Text**.
8. Click on the screen, type Time: #####, and then press Return.
9. Double-click the object to open its *Object Properties* dialog.

10. Click **Text Data Link**.

The *Object Properties* dialog changes to show the properties for the Text Data Link animation.

11. In the **Tag/Expression** box, type `Time`.



### ***Specifying the Time system tag***

During runtime, the project replaces the ##### characters of the Text object with the value of the system tag `Time`.

12. Close the *Object Properties* dialog.

13. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu.

This figure shows how your screen should look after you've created the date and time objects.



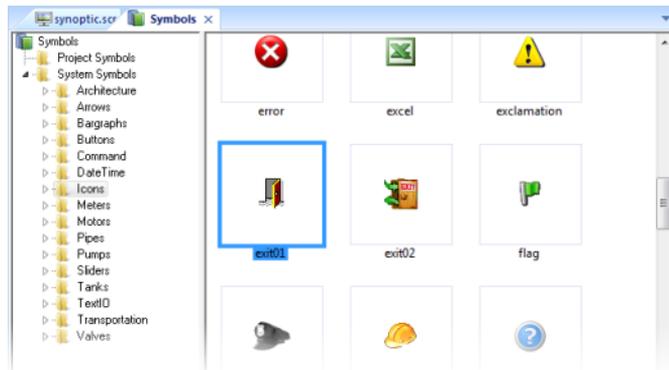
### ***Finished date and time objects***

## ***Placing an "Exit" icon***

This part of the tutorial shows how to place an icon (by selecting and configuring a Linked Symbol) that allows the user to exit the project, .

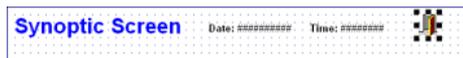
1. On the **Graphics** tab of the ribbon, in the **Libraries** group, click **Symbols**. The symbols library is displayed.
2. In the Symbols menu tree, open the **System Symbols** folder and then open the **Icons** sub-folder.
3. In the Icons sub-folder, select **exit01**.

The symbol will be displayed in the symbol viewer to the right of the menu tree.



**Selecting the "exit01" symbol**

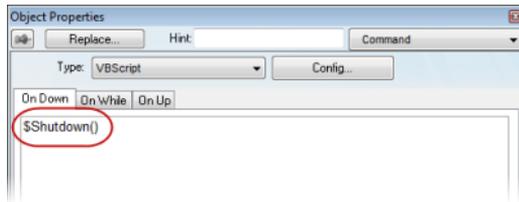
4. Click on the symbol.  
The mouse cursor will change to show that the symbol is ready to be placed in a screen.
5. Switch back to the screen where you want to place the symbol and then click in it.  
The symbol is placed as a Linked Symbol object.



**Placing the Linked Symbol object**

6. With the object still selected, click **Command** (on the **Graphics** tab of the ribbon, in the **Animations** group) to apply this animation to the object.
7. Double-click the object to open its *Object Properties* dialog.
8. In the **Type** list, select **VBScript**.
9. In the **On Down** box, type `$Shutdown()`.

Shutdown is one of InduSoft Web Studio's built-in scripting functions, but it can be used within VBScript by prefacing it with a dollar sign (\$).



**Specifying the Shutdown command on the symbol**

10. Close the *Object Properties* dialog.
11. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu.

Now, when a user clicks this icon during runtime, the project will stop and exit to the station's desktop.

### **Testing the project**

This part of the tutorial show how to test the project so far.

1. On the **Home** tab of the ribbon, in the **Local Management** group, click **Run**. The project runs and the startup screen is displayed.
2. Click the button to open the synoptic screen. The synoptic screen is displayed.
3. Click the exit icon to shut down the project.

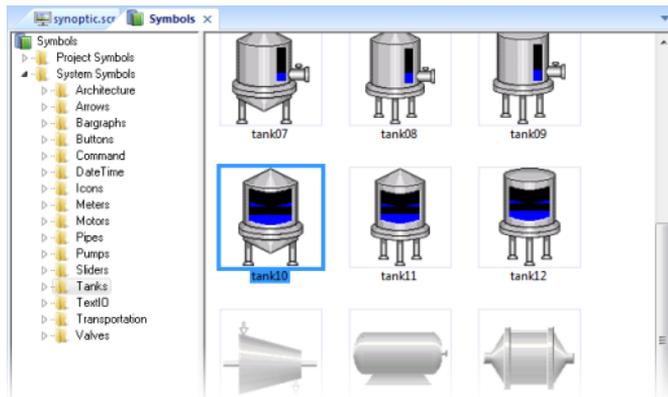
If any part of the project doesn't work as expected, switch back to the development application (**ALT+TAB**) and then click **Stop** on the Home tab of the ribbon.

### **Placing an animated tank**

This part of the tutorial shows how to select an animated tank from the Symbol Library and place it on the screen (similar to how you selected and placed the "Exit" icon), then associate some project tags with the tank's properties.

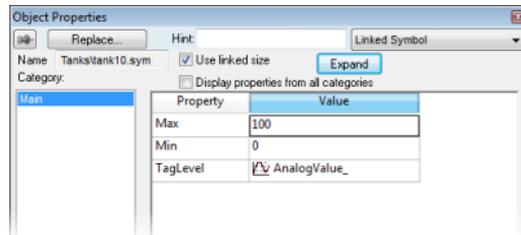
1. On the **Graphics** tab of the ribbon, in the **Libraries** group, click **Symbols**.
2. In the Symbols menu tree, open the **System Symbols** folder and then open the **Tanks** sub-folder.

- In the Tanks sub-folder, select a tank symbol.  
You may select any tank you like; they all function basically the same way.



**Selecting a tank symbol**

- Click on the symbol.  
The mouse cursor will change to show that the symbol is ready to be placed in a screen.
- Switch back to the screen where you want to place the symbol and click in it.  
The symbol is placed as a Linked Symbol object.
- Double-click the object to open its *Object Properties* dialog.



**The tank symbol's properties**

A tank is an arrangement of different objects and animations (for example a rectangle, a bar graph, etc.), all combined together as a Linked Symbol.

You can modify the properties of this symbol by editing the properties list. For this tutorial, you will modify the tag associated with the tank level.

7. For the property **TagLevel1**, delete the existing value and then type `Level[Index]`.  
Note that you do not need to reopen the Project Tags datasheet to create tags as you develop the project.  
Because you have not previously created the tag **Index** in the Project Tags database, an alert message asks you if you would like to create it.
8. Click **Yes**.  
A *New Tag* dialog is displayed.
9. Configure the new tag with **Array** as 0, **Type** as `Integer`, and **Scope** as `Local`.



**Configuring a new tag**

10. Click **OK** to close the *New Tag* dialog.

You can use the tag **Index** to set the array position of the tag **Level1**, and show the level for any of the three tanks in the same object:

- When **Index** equals 1, the tank object shows the level of Tank #1 (i.e., `Level[1]`);
- When **Index** equals 2, the tank object shows the level of Tank #2 (i.e., `Level[2]`); and
- When **Index** equals 3, the tank object shows the level of Tank #3 (i.e., `Level[3]`).

Also, because the tag scope is local, the tag can have different values for the Server and Client stations at the same time. Consequently, the local user (i.e., the Server station) can be monitoring the level of Tank #1 while the remote user (i.e., the Client station) is monitoring the level of Tank #2.

11. Close the *Object Properties* dialog.
12. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu.

This figure shows how your screen should look after you've created the tank object.

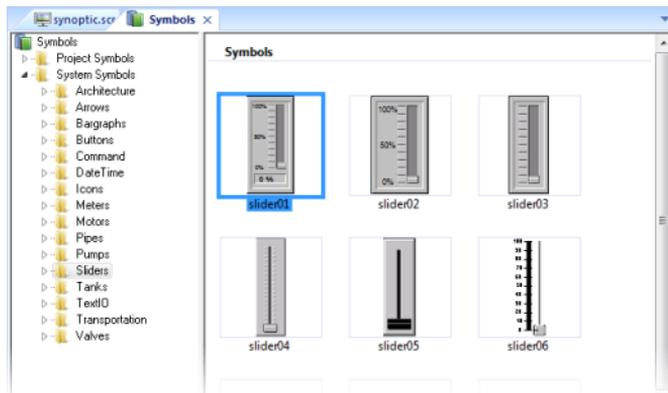


*Finished tank object*

### ***Placing a level slider***

This part of the tutorial shows how to select a slider control from the Symbol Library and then connect it to the animated tank.

1. On the **Graphics** tab of the ribbon, in the **Libraries** group, click **Symbols**.
2. In the Symbols menu tree, open the **System Symbols** folder and then open the **Sliders** sub-folder.

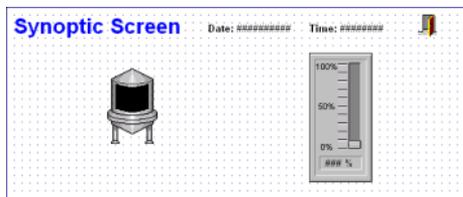


*Selecting a slider symbol*

3. In the Sliders sub-folder, select a slider control.  
You may select any slider you like; they all function basically the same way.
4. Click on the symbol.  
The mouse cursor will change to show that the symbol is ready to be placed in a screen.

5. Switch back to the screen where you want to place the symbol and click in it.  
The symbol is placed as a Linked Symbol object.
6. Double-click the object to open its *Object Properties* dialog.
7. For the property **TagName**, delete the existing value and then type `Level[Index]`.  
Just as with the tank, you need to modify the symbol property associated with the slider level.
8. Close the *Object Properties* dialog.
9. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu.

This figure shows how your screen should look after you've created the level slider object.



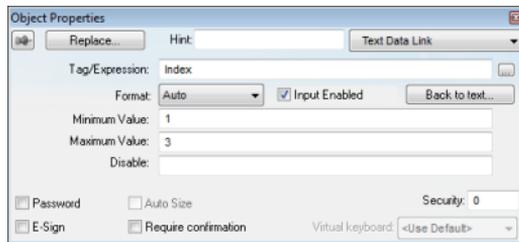
*Finished level slider object*

### ***Drawing a tank selector***

This part of the tutorial shows how to draw a text input box that can be used to change which real-world tank is represented by the animated tank on the screen.

1. On the **Graphics** tab of the ribbon, in the **Active Objects** group, click **Text**.
2. Click on the screen, type **Tank: #**, and then press Return.
3. Double-click the object to open its *Object Properties* dialog.
4. Click **Text Data Link**.  
The *Object Properties* dialog changes to show the properties for the Text Data Link animation.
5. In the **Tag/Expression** box, type `Index`.
6. Select the **Input Enabled** option.  
This allows the operator to enter a new value for the tag during runtime.
7. In the **Minimum Value** box, type `1`.

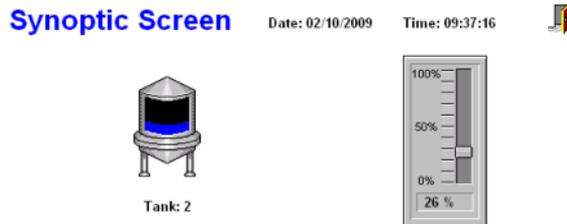
- In the **Maximum Value** box, type 3.



### Configuring the "Tank" text input

- Close the *Object Properties* dialog.
- Click the **Application** button at the top-left of the development application, and then click **Save** on the Application menu.

This figure shows how your screen should look after you've created the tank selector object.



### Finished tank selector object during runtime

### Testing the project

This part of the tutorial show how to test the project again with the animated tank, the level slider, and the tank selector.

- On the **Home** tab of the ribbon, in the **Local Management** group, click **Run**. The project runs and the startup screen is displayed.
- Click the button to open the synoptic screen. The synoptic screen is displayed.
- Type the tank number (1, 2, or 3) in the Tank label, and then use the slider to adjust the tank level. Note that you can view/adjust the level of each tank independently.

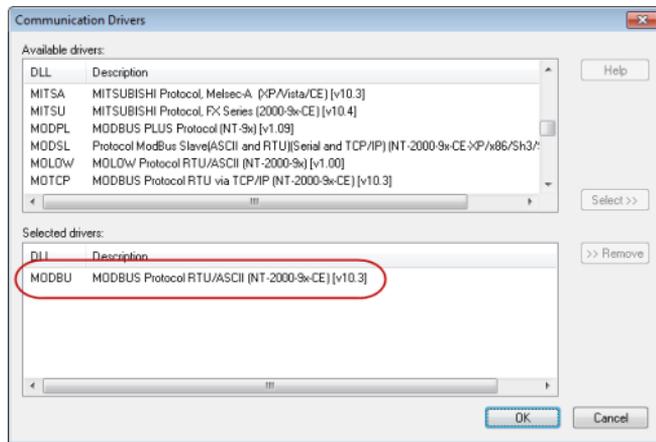
4. Click the exit icon to shut down the project.

If any part of the project doesn't work as expected, switch back to the development application (**ALT+TAB**) and then click **Stop** on the Home tab of the ribbon.

## Configuring the communication driver

This part of the tutorial shows how to select and configure a driver to communicate with an external I/O device.

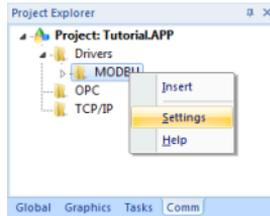
1. In the *Project Explorer*, click the **Comm** tab.
2. Right-click the **Drivers** folder, and then click **Add/Remove Drivers** on the shortcut menu.  
The *Communication Drivers* dialog is displayed.
3. Select a driver from the **Available drivers** list, and then click **Select**.  
For this tutorial, select MODBU.  
The driver is moved to the **Selected drivers** list.



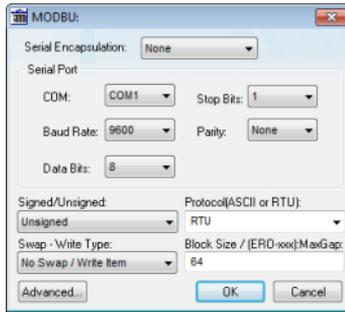
*MODBU driver selected*

4. Click **OK**.  
The *Communication Drivers* dialog is closed, and the driver is added to the Drivers folder in the *Project Explorer*.

5. In the Project Explorer, right-click the **MODBU** folder, and then click **Settings** on the shortcut menu.



The *Communication Settings* dialog is displayed.



**Communication Settings dialog for MODBU driver**

6. Configure the communication settings as needed for the target device. For this tutorial, accept the default settings.

 **Note:** For more information about a specific driver, click **Communication Drivers** on the **Help** tab of the ribbon.

7. Click **OK** to close the dialog.
8. In the Project Explorer, right-click the **MODBU** folder and then click **Insert** on the shortcut menu.  
A new driver worksheet named MODBU001.drv is created and opened for editing.
9. Configure the worksheet header:
  - a) In the **Description** box, type Tutorial Modbus.  
This setting is for documentation only; it does not affect the runtime project in any way.
  - b) In the **Enable Read When Idle** box, type 1.

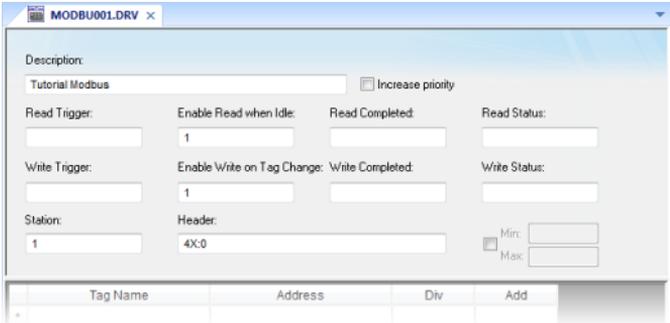
This setting is a trigger that takes a Boolean value. A value of 1 — either entered manually as above or evaluated from a tag/expression — forces your project to continue reading tag values from the target device even when there are no changes in value.

- c) In the **Enable Write On Tag Change** box, type 1.  
This setting is also a trigger. A value of 1 forces your project to write tag values to the target device only when those values change, rather than continuously. This saves system resources and improves performance during runtime.
- d) In the **Station** box, type 1.  
This indicates the I/O device number to be accessed by this driver. Typically, the PLC is specified as Device #1.
- e) In the **Header** box, type 4X:0.

You must use a driver-specific format. The format for the MODBU driver is:

```
register_type:initial_offset
```

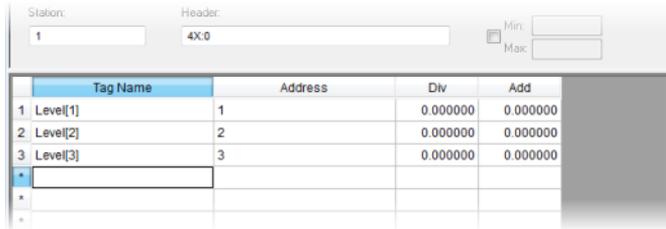
Register Type	Description
0X	Coil Status
1X	Input Status
3X	Input Register
4X	Holding Register
ID	Slave ID Number



**Completed worksheet header**

10. In the worksheet body, enter the tags and their associated device addresses — for each tag:
  - a) In the **Tag Name** field, type the name of the project tag.
  - b) In the **Address** field, type the value to be added to the header to form the complete device address.

Tag Name	Address	Complete Device Address
Level [1]	1	4X:1 (Holding Register 1)
Level [2]	2	4X:2 (Holding Register 2)
Level [3]	3	4X:3 (Holding Register 3)



**Completed worksheet body**

11. Click the Application button at the top-left of the development application, and then click **Save** on the Application menu.
12. When prompted to choose the driver sheet number, type 1 and then click **OK**.

### ***Monitoring device I/O during runtime***

This part of the tutorial shows how to monitor device I/O during runtime by using the *Log* window.

1. On the **Home** tab of the ribbon, in the **Local Management** group, click **Run**. The project runs and the startup screen is displayed.
2. Press **ALT+TAB** to switch back to the development application.
3. Right-click in the *Output* window, and then click **Settings**. The *Log Settings* dialog is displayed.
4. Select the **Field Read Commands**, **Field Write Commands**, and **Protocol Analyzer** options.
5. Click **OK** to close the *Log Settings* dialog.

You can now monitor the device I/O during runtime.

## Downloading your project to a Windows Embedded device

---

This part of the tutorial shows how to download your project to a Windows Embedded device, such as a plant-floor HMI panel.

After configuring a project and testing it locally (on the development station), you can download it to a remote station — either a Windows PC that is running IWS or a Windows Embedded device that is running CEView.

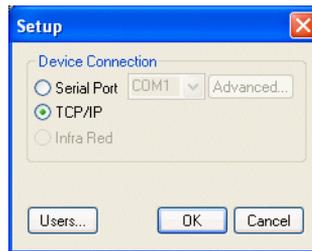
1. On the desktop of the remote station, click **Start > All Programs > InduSoft Web Studio v7.1 > Remote Agent** .  
The Remote Agent utility runs.



**Remote Agent utility**

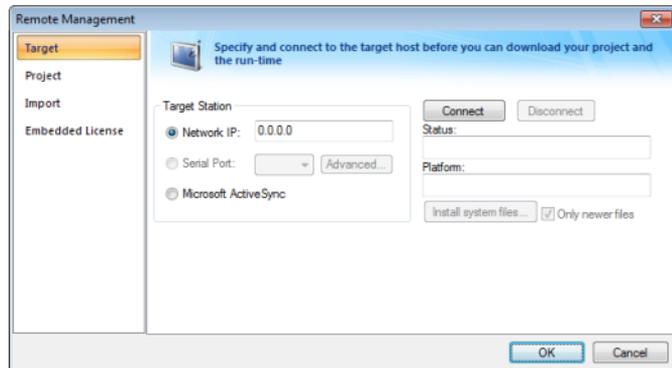
2. Click **Setup**.  
The *Setup* dialog is displayed.
3. Select the type of connection — **Serial**, **TCP/IP**, or **Infrared** — between the remote station and the development station.

 **Note:** For better performance, we recommend that you use TCP/IP whenever possible.



#### **Selecting TCP/IP on the remote station**

4. Click **OK** to close the *Setup* dialog, but leave the Remote Agent utility running on the remote station.
5. In the development application, click **Connect** on the **Home** tab of the ribbon. The *Remote Management* dialog is displayed.



#### **Remote Management dialog**

6. Select the type of connection to the target (remote) station.  
This selection should match the selection you previously made in the Remote Agent utility on the remote station.
7. If you selected **Network IP**, type the IP address of the remote station.
8. Click **Connect**.

If you successfully connect to the remote station, then information about that station is displayed in the **Status** and **Platform** boxes.

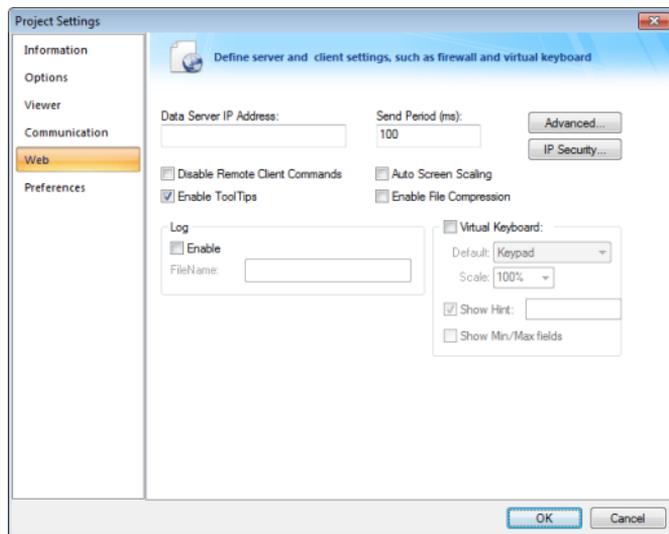
9. If the remote station is a Windows Embedded device, click **Install system files**.  
The system files are installed on the remote station.
10. Click the **Project** tab.
11. Click **Download**.  
The project files are downloaded to the remote station.
12. Click **Run**.  
Your IWS project is run on the remote station.

## Deploying your project as a web application

This part of the tutorial shows how to deploy your project as a web application, to which remote users can connect with Internet Explorer.

For Internet Explorer to work as a web thin client, it must install an ActiveX control that "plays" IWS project screens. If your computer is connected to the Internet, then IE will automatically download the control from InduSoft's public server when you access a runtime project for the first time.

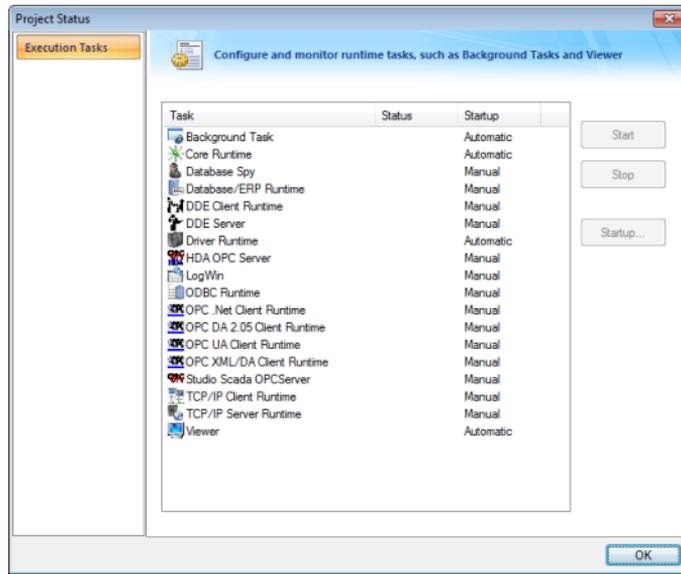
1. Configure the IP address of the data server.
  - a) On the **Project** tab of the ribbon, in the **Web** group, click **Thin Client**. The *Project Settings* dialog is displayed with the **Web** tab selected.



**Web tab of Project Settings dialog**

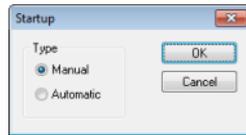
- b) In the **Data Server IP Address** box, type the IP address of the web server. For this tutorial, type 127.0.0.1, which is the standard loopback address (a.k.a. "localhost").
  - c) Click **OK** to close the dialog.
2. Make sure the data server is set to start up when you run your project.
  - a) On the **Home** tab of the ribbon, in the **Local Management** group, click **Tasks**.

The *Execution Tasks* dialog is displayed.



**Execution Tasks dialog**

- b) In the list of tasks, select **TCP/IP Server Runtime**, and then click **Startup**. The *Startup* dialog is displayed.



**Startup dialog**

- c) Select **Automatic**, and then click **OK**.
- d) Click **OK** to close the *Execution Tasks* dialog.
3. Save and close all open screens and worksheets.
4. Click the Application button at the top-left of the development application, and then click **Publish > Save All As HTML** on the Application menu. Your project screens are saved as HTML files in the Web sub-folder of your project folder (i.e., `\project_name\Web`).

5. Configure a web server to make the Web sub-folder available to the network.

For this tutorial, use NT Web Server, the free, lightweight web server program that is included with IWS. Simply copy the program file from [...] \InduSoft Web Studio v7.1\Bin\NTWebServer.exe to the Web sub-folder, and then double-click the file to run it. It automatically serves the contents of whatever folder it's in, without any further configuration.



**Caution:** NT Web Server is provided for testing purposes only. It should **never** be used in a real production environment or on a secure network.

You can also configure the "root directory" or "home directory" setting of some other web server program (e.g., Microsoft IIS) to point to the Web sub-folder, or you can copy the Web sub-folder to an existing web server on your network. The web server (which makes the HTML files available to clients) and the data server (which actually runs your IWS project and exchanges data with the clients) do not need to be the same computer.

6. On the **Home** tab of the ribbon, in the **Local Management** group, click **Run**.
7. Open a web browser (e.g., Microsoft Internet Explorer), and then enter the URL address of the synoptic screen on the web server.  
For this tutorial, type `http://127.0.0.1/synoptic.html` and then press **Return**. (127.0.0.1 is the standard IP address for the loopback network interface, a.k.a. "localhost.")  
After a few moments, during which the browser downloads and installs the ActiveX control, the synoptic screen is displayed in the browser.

Notice that you can modify the level of any tank either locally using the project viewer or remotely using the web browser, and changes on one client appear immediately on the other. They work equally well.

