
DEVELOP 10 TIMES FASTER

WINDEV 25
Documentation version 25 - 1 0320

Don't forget to visit our site (www.windev.com) on a regular basis to find out whether upgraded versions are available.

Email address of Free Technical Support: freetechnicalsupport@windev.com

This documentation is not contractually binding. PC SOFT reserves the right to modify or delete any topic dealt with in this document.

Summary

Lesson 2.2. The conditional statements

Lesson 2.3. The loops

Lesson 2.4. The procedures

Lesson 2.5. Questions/Answers

Part 3 - My first database

Lesson 3.1. Overview

Lesson 3.2. WINDEV and the databases

Lesson 3.3. Project and analysis

Lesson 3.4. The full RAD

PART 4 - FULL APPLICATION WITH DATA

Lesson 4.1. Overview

Lesson 4.2. Addition and modification windows

Lesson 4.3. Simple search and record browse

Lesson 4.4. Multicriteria search

Lesson 4.5. Printing the content of a Table control

Lesson 4.6. Printing an order

Lesson 4.7. Printing a list of customers

Lesson 4.8. Statistics: Chart and Pivot Table controls

Lesson 4.9. Sending an email

Lesson 4.10. Identifying the user: the user groupware

Lesson 4.11. Re-use code via the external components

Lesson 4.12. Consuming a Webservice

Lesson 4.13. Monitor the evolution of your applications

Lesson 4.14. Deploying the application

Lesson 4.15. Distributing "Reports and Queries" with your applications

Lesson 4.16. Managing multiple languages

Lesson 4.17. SCM

Part 5 - Managing a HFSQL Client/Server database

Lesson 5.1. Introduction

Lesson 5.2. Implementing a Client/Server database

Lesson 5.3. Managing a Client/Server database

Lesson 5.4. Setup on end-user computers

Part 6 - Optimizing and debuging a project

Lesson 6.1. Overview

Lesson 6.2. Project audits

Lesson 6.3. Performance profiler

Lesson 6.4. Debugging a project

Part 7 - Advanced programming

Lesson 7.1. Overview

Lesson 7.2. Automatic management of errors

Lesson 7.3. Handling external files

Lesson 7.4. Dynamic compilation

Lesson 7.5. Windows event

Lesson 7.6. The threads

Lesson 7.7. The sockets

Lesson 7.8. The FTP

Lesson 7.9. The OOP

Part 8 - Appendices

Appendices 1. Vocabulary

Appendices 2. Using SQL data

Conclusion

Introduction

Preliminary points

Caution: This is a tutorial. We advise you to refer to the online help when using WINDEV.

The purpose of this tutorial is to help you discover WINDEV, get familiar with the editors and learn the WINDEV concepts.

This tutorial does not cover all the features in WINDEV.

Spend at least a few hours to follow this tutorial and learn WINDEV: it is a good investment!

If you try to start developing an application without following this tutorial, you will lose time, much more than the few hours spent on this tutorial.

The tutorial has been designed to be followed in two ways:

- follow all the detailed exercises in each lesson (recommended method).
- or, if you are in a hurry and already have significant experience, you can simply read this tutorial without doing the exercises (all the actions are illustrated). However, in order to quickly assimilate the main concepts, we recommend that you follow the tutorial step by step.

As WINDEV is constantly evolving, the images of the windows in the tutorial may differ from the windows displayed in the product during the various operations.

Tutorial overview

This tutorial has been designed to progressively teach you how to use WINDEV. By following this tutorial:

- you will discover the main concepts explained informally; these are the concepts you must learn and understand.
- you will also be asked to perform operations that illustrate the concepts just explained.

As you progress through the tutorial, if you want to take a closer look at a concept or if you want to get more details about a programming function, see the online help (accessible from the editors).

The size of a lesson is not necessarily proportional to its relevance...

And don't forget to take a look at the examples supplied with WINDEV: they are very instructive!

Legend of the symbols in this tutorial

This symbol indicates the duration of the lesson. Please note that the actual time may vary according to your level of experience.

An example is available to complement the lesson. The examples are available in the WINDEV home page (Ctrl + <).

This symbol introduces a "Tip": reading the associated text is strongly recommended.

This symbol introduces a "Warning": reading the associated text is essential.

This symbol introduces a "Note": reading the associated text is recommended.

Remark

The tutorial may have evolved since this document was printed. Feel free to consult the online version (http://doc.windev.com).

How to access the online help

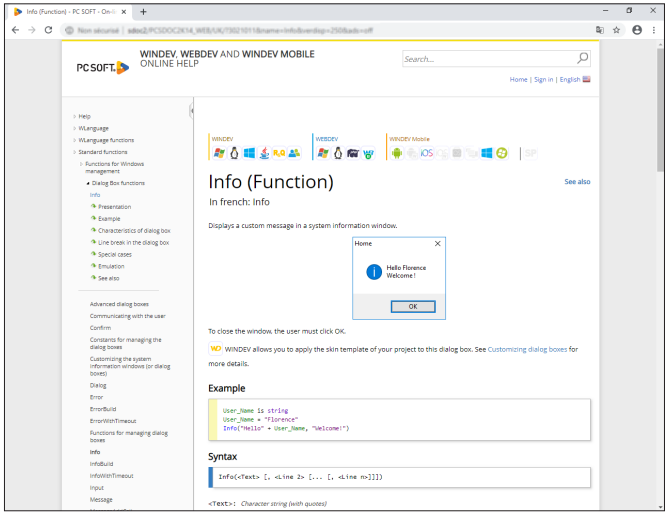
The WINDEV online help allows you to get detailed information about the 3700 WLanguage functions. It also contains the help about the editors and the controls, tips, ...

The online help is available at any time in WINDEV:

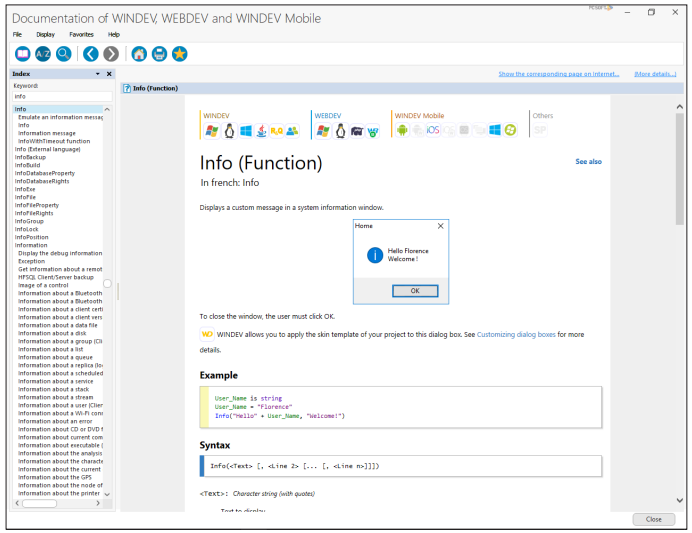
- In the code editor, a specific help is available for each function via the F1 key.
- Each dialog box displayed by WINDEV can propose a ? button allowing you to access the corresponding help page.
- The help menu of editors ("Help" option available on the "Home" pane, in the "Online help" group of WINDEV menu) allows you to start the online help.

▸ The help can be displayed:

- in an Internet browser, if you have access to Internet:



- in a specific "help browser":



Remark

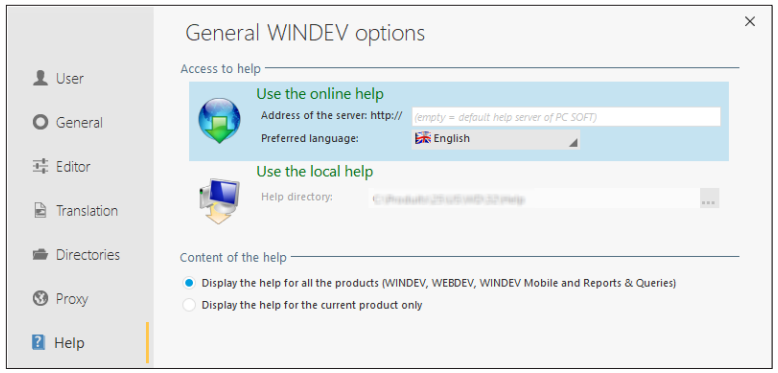
We advise you to check the online help on Internet rather than the local online help. Indeed, the online help on Internet is updated on a regular basis.

The online help of WINDEV, WEBDEV and WINDEV Mobile on Internet is available from any computer equipped with an Internet access, without the product being necessarily installed.

Each Web user can add comments about the documentation pages: personal notes, examples, links, ...

Note: If you have no access to Internet, you have the ability to start the local help from the product:

1. On the "Home" pane, in the "Environment" group, expand "Options" and select "General options of WINDEV".
2. In the "Help" tab, select:
   - the access mode to the help.



   - the help content: WINDEV, WEBDEV and WINDEV Mobile common help or help only for the product currently used.

If you are familiar with WINDEV 24

If you are familiar with WINDEV 24, following this tutorial can only beneficial: it's a good opportunity to "review" the WINDEV features!

What is WINDEV used for?

WINDEV is an IDE (Integrated Development Environment). It allow you to develop applications in many fields:

- Stock management, inventory, product tracking,
- Adjustment and monitoring of machines on an assembly line,
- Taking orders for fast processing in a temporary outlet (fairs, schools, booth, ...),
- Customer forms,
- Help with making snap decisions on a cell phone,
- Checking the identity of visitors at an event: trade fair, presentation of products, ...
- On-call doctors or vets,
- Taking information in a temporary outlet: trade fair, street poll, stadium, ...

WINDEV is an integrated development environment that includes all the tools required to develop an application.

Unlike other programming languages, there is no need to find and add modules to be able to design, check and install an application.

The 5GL (5th Generation Language) of WINDEV, named WLanguage, will surprise you by its simplicity: a few hours are all you need to get the hang of it, a week is usually all it takes to fully master its potential!

No more programming hassle, WLanguage is available in English and in French!

WINDEV 25

Lesson 1.1. Discover WINDEV

This lesson will teach you the following concepts

- Starting WINDEV.

Estimated time: 5 mn

Overview

WINDEV is an IDE (Integrated Development Environment) allowing you to develop Windows applications in several fields: business, industrial, medical, ... The developed applications can give access to information stored in the databases.

As you go through this tutorial, you will learn to create your own applications (with or without database) and to improve them by using the features proposed by WINDEV.

Starting WINDEV

▸ Start WINDEV 25 (if not already done).

▸ A welcome wizard starts **if WINDEV 25 has not been started before**:
- If you worked with an earlier WINDEV version, this wizard allows you to retrieve the existing configurations.
- If you are a new user, this wizard allows you to configure your environment. This allows you to choose the screen configuration used and to configure the Control Centers.

▸ **If WINDEV 25 was already started**, identify yourself if necessary. The development environment starts. The home page is displayed. This home page is used to:
- display the news,
- create a project,
- open an existing project,
- open an example,
- open one of the projects found in the tutorial.

▸ Let's take a closer look at the development environment of WINDEV. To do so, from the home page:
- Click "Tutorial".
- Double-click "Full application (Answer)".
- The corresponding project is opened in the editor. The project dashboard is displayed. The project dashboard allows you to check the progress of a project via several elements (called widgets).

Development environment

The editor

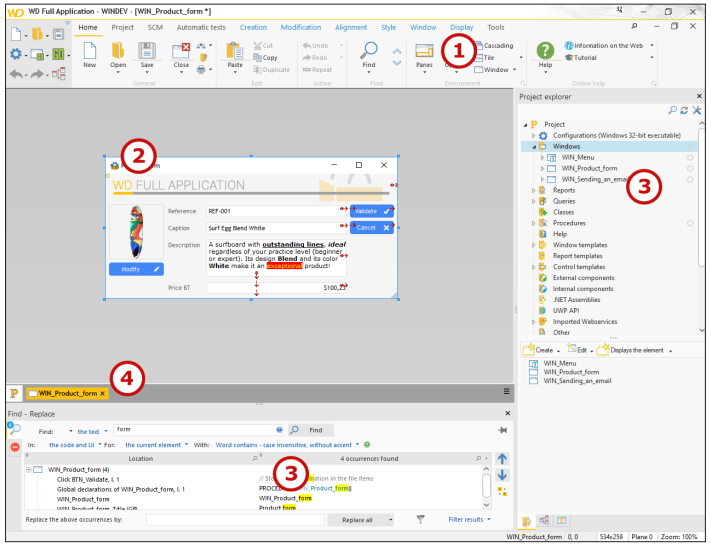
The development environment of WINDEV includes a specific interface and several editors allowing you to create the different elements of your applications.

For example, the window editor is used to create windows, the report editor is used to create reports, ...

▸ To discover WINDEV, we are going to open the "Product form" window:

1. Press Ctrl + E.
2. In the window that is displayed, type the name of the window to open: WIN_Product_form.
3. Validate. The window is displayed in the editor.

All editors are using the same environment:



1. Menu of editors, displayed in ribbon format (we'll see how to use it in the next paragraph).

2. Current editor (window editor in this case). This space allows you to see the element currently created or modified in WYSIWYG (What You See Is What You Get).

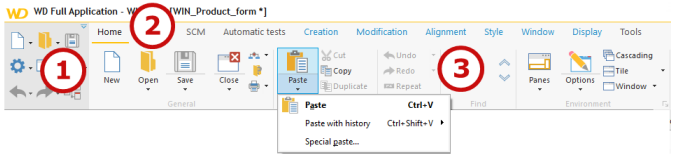
3. Panes. The WINDEV interface includes several horizontal and vertical panes allowing you to quickly access different types of information. For example:
- the "Project explorer" pane (displayed on the right) is used to list all project elements by category.
- the search pane (displayed at the bottom) is used to perform searches in the entire project and in its elements.

These panes can be hidden by pressing Ctrl + W if necessary.

4. Bar of opened documents. This bar is used to quickly see all opened elements. A simple click on the button corresponding to the element displays it in its own editor.

The menu bar (ribbon) in details

The menu bar of WINDEV is presented like a ribbon. This ribbon includes panes in which the different options of editors are grouped.
We are going to take a closer look at the main ribbon elements, as well as how we will be using it in this tutorial.



The different ribbon elements

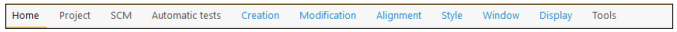
The ribbon includes 3 areas:
- the button area, on the left (1).
- the pane area, at the top (2).
- the option area (3).

Let's take a closer look at these areas.

The button area (1)

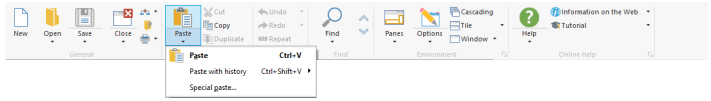
The button area groups the **quick access buttons**. These buttons are used to perform the most usual operations, common to all editors: save, open, create, ...
The product logo is used to display the "About" window, the custom menus and the drop-down menus found in the former interface of editors.
The arrow at the top right of the button area allows you to find the toolbars and drop-down menus of the old editor interface.

The pane area (2)



The different ribbon panes are used to access the options of different editors for the current project. Several types of panes are available:
- the current pane: The name of the pane appears in bold and is underlined by an orange line.
- the popup panes, specific to the current element: The pane name is displayed in blue.
- the available panes: The pane name is displayed in black.

The option area (3)



The options displayed in the ribbon differ according to the selected pane. Several types of options are available:
- Options to check.
- Buttons to click.
- Button with arrow used to expand the options. Two types of buttons with arrow are available:
  - the buttons with arrow used to expand a menu.
  - the buttons with arrow used to expand a menu (click on the arrow) or to perform a default action (click on the button icon).

The options are organized by group. Each group of options has a name and it can also include a group button . This button is used to perform a specific action according to the current group: display the description of current element, display the help, ...

In this tutorial, to identify a menu option, we will be talking about panes and groups. For example: To display the help, on the "Home" pane, in the "Online help" group, click "Help".

The environment colors

The environment is using a light theme by default.
Several other themes are also available:
- Light theme, grey ribbon. In this mode, the menu bar is not colored anymore: it is grayed.
- Gray theme. In this mode, the environment and the interface windows are displayed on a light gray background.
- Dark theme. In this mode, the environment and the interface windows are displayed on a black or dark gray background.

▶ To modify the theme used by the environment:

1. On the "Home" pane, in the "Environment" group, expand "Options" and select "General options of WINDEV".
2. In the "Editor" tab, in the "Themes" area, select the theme to use.
3. Validate. The theme will be taken into account during the next start of WINDEV.

▶ To improve the readability of this manual, the light theme will be used for the different images that illustrate the operations to perform.

▶ To continue this lesson, we are going to close the current project:

1. On the "Home" pane, in the "General" group, expand "Close" and select "Close the project".
2. Save the modified elements if necessary.
3. The WINDEV home page is displayed.

Lesson 1.2. My first window

This lesson will teach you the following concepts

- How to create a window.
- How to type a text and display it.

Estimated time: 30 mn

Overview

To start working with WINDEV, we are going to create a window.

Remark

Windows are used to display or type information on the screen. The user can directly act on the windows via controls, buttons, ...

We will see how to:

- Open a project,
- Create and save a window,
- Type and display data.

These first operations will present the main concepts for developing with WINDEV. Full topics will be presented in the following sections.

Opening project

▸ Start WINDEV 25 (if not already done). Display the WINDEV home page if necessary: press Ctrl + <.

▸ Open the "WD My First Windows" project.
To do so, in the home page, click "Tutorial" and select "My first windows (Exercise)".

Important

In this section, we will focus on the creation of simple windows. The "WD My First Windows" project is an empty project that was already created. The project creation will be presented in another lesson.

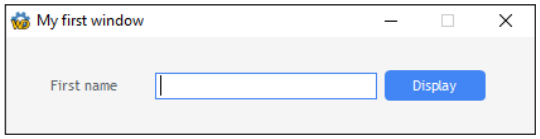
Answer

A corrected project is available. This project contains the different windows created in this part (as well as the ones created in part 2). To open the corrected project, in the home page, click "Tutorial" and select "My first windows (Answer)".

My first window: entering and displaying data

Overview

You are going to create the following window:



This window allows the user to type his first name and to display a welcome message with the "Display" button.

You may think this is too basic but we advise you to create this window. You may well be surprised by how intuitive and easy it is to use the WINDEV editor. Furthermore, this window will teach you principles that are fundamental for the rest of this tutorial.

Creating the window

▸ To create the window:

1. Click among the quick access buttons of the WINDEV menu:

2. The window for creating a new element is displayed. This window is used to create all elements that can be associated with a project.

3. Click "Window" then "Window". The window creation wizard starts.

4. Select "Blank" in the list of "standard" windows displayed on the left. The "Phoenix" skin template is selected by default in the list of skin templates on the right. You can choose another skin template proposed in the list.

Remark

Skin templates allow you to quickly create outstanding interfaces. A skin template defines the window style as well as the style of all controls that will be used in this window. Thus, there is no risk of obtaining an ugly interface.

5. Validate the window creation wizard ("OK" button). The window is automatically created in the editor. The window for saving an element is displayed. This window is used to specify:

- The element title. For a window, this title will be displayed in the title bar of window.
- The element name. This name corresponds to the logical element name. This name will be used to handle the element by programming.
- The element location. This location corresponds to the directory in which the element is saved.

6. Specify the title of "My first window" element (in this case, the element corresponds to the WINDEV window). The element name ("WIN_My_first_window") is automatically proposed.

Remark

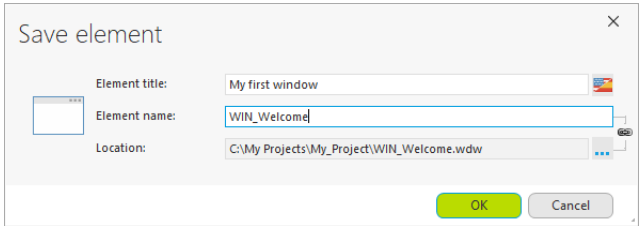
Study the window name proposed by WINDEV: this name starts with the letters "WIN_". This prefix is automatically added because the project is using a code style.

The code style is used to define a prefix for each type of object, allowing you to quickly identify the element:

- a window starts with "WIN_",
- a Button control starts with "BTN_",
- etc.

If you do not want to use this code style, you can simply disable it: on the "Project" pane, in the "Other actions" group, expand "Code style" and uncheck "Use code style".

7. In this example, we want to use a shorter window name: replace "WIN_My_first_window" by "WIN_Welcome".



8. Click "OK" to validate the information shown in the save window.

Entering and displaying the value

To type and display the value, you must create:

- a control where the user will type his first name. Therefore, this type of control is an "edit control".
- a Button control to display the first name entered.

► To create the edit control:

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit" (click the arrow). A list of preset edit controls appears.

Remark

This list offers different Edit controls that can be used directly. These controls are also available among the Business controls. The different controls can be tested directly: simply click the "GO" icon that appears when the control is hovered over.

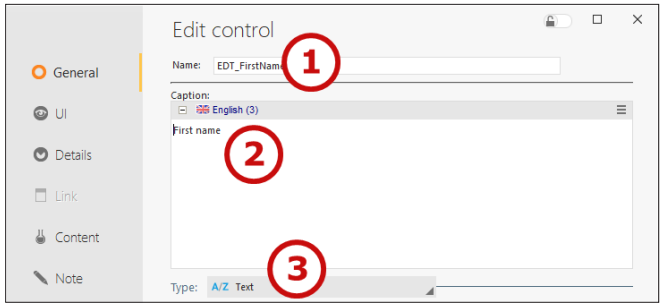
2. In the search box at the top of the list of preset controls, enter "Simple text Edit". The corresponding control appears in the list.

3. Click the "Simple text Edit" control. The control currently created follows the mouse movement.

4. Move the mouse toward the position where the control will be created in the window (at the top of window for example). To drop the control in the window, simply click again.

5. Right-click the control that was just created. The popup menu of control is displayed. Select "Description" from the popup menu. The description window of edit control is displayed.

6. Modify the control characteristics by entering the following information:



- This control is named: "EDT_FirstName" (1).
- The control caption is: "First name" (2).
- This control is a text control (3).

7. Validate the control description window ("OK" button). The control appears in the window editor.

The control caption is truncated in the editor.

► To display the control properly:

1. Select the control (all you have to do is click it with the mouse).

2. In the popup menu (right click), select "Adapt the size".

3. The control is immediately enlarged and the caption is entirely displayed.

Remark: You also have the ability to manually enlarge the input area of control:

1. Select the control.

2. Position the mouse cursor above one of the handles found on the right of control (small square). The mouse cursor turns into arrow.

3. Click with the left mouse button and keep the button down during the move used to enlarge the control.

4. Release the mouse button once the control was resized.

► To create the "Display" Button control:

1. On the "Creation" pane, in the "Usual controls" group, click Ok.

2. Position the control in the window. Create the Button control at the desired location with a click (on the right of edit control for example).

3. Click the control that was just created. The text displayed in the control becomes editable.

Tip

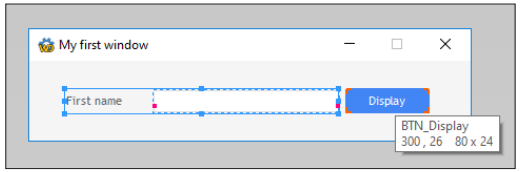
To quickly edit the control caption, you also have the ability to:

- Select the control.
- Press the Enter or Space key.

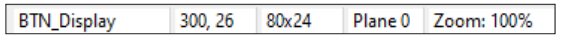
4. Type the caption: "Display". The name of the Button control automatically becomes "BTN_Display".

5. You can see the name of the control:

- in the tooltip displayed when the control is hovered by the mouse cursor:



- in the status bar of editor:



Remark

If you are using a 4K screen, the editor elements are automatically zoomed. The zoom level used depends on the Windows display settings.

► We are going to display the text typed in a dialog box (a mini-window proposed by the system). To do so, we will be using our first WLanguage function: **Info**.

Remark

The programming language supplied with WINDEV is named WLanguage. It is a 5th-generation language (5GL) that includes highly sophisticated commands.

Main WLanguage conventions

- No ending character for end of line is used in WLanguage.
- The '//' characters are used to comment out code lines. These code lines are not interpreted. They provide better code readability.
- In the WLanguage functions, the parameters passed to the function are enclosed in brackets.
- The coma character corresponds to the separator.
- WLanguage is not case sensitive.
- The "space" characters are not interpreted.
- The dot is the decimal separator.
- The name of WLanguage functions is in English. It is available in French.
- The name of WLanguage functions is using a prefix in order to group the functions by family ("Win" for the functions handling windows, "email" for the functions handling emails, ...).

1. Select the "Display" Button control with the mouse.

2. Display the popup menu of control (right mouse click).

3. Select "Code". This option opens the WINDEV code editor. This editor is used to type all WLanguage statements.

Remark

The code editor proposes different events for each type of control. These are the WLanguage events related to the control.

Thus, two events are displayed for the Button control:

- Initializing,
- Click, run when the user clicks the button.

Remark: Additional events can be added if necessary.

4. Write the following WLanguage code in the "Click BTN_Display" event:

```
Info("Hello ", EDT_FirstName)
```

Note about the assisted input: As soon as the first two characters are typed, WINDEV proposes all words of WLanguage vocabulary containing these characters. The assisted development is a very powerful feature. No more mistakes when entering the name of an element: the syntax errors are reduced. All you have to do is select the requested word and press Enter to validate. You can focus on the algorithm.

Remark

This is why the code style is so important. All elements handled in the application code use the same standard so they can be easily identified when writing code.

Remark

When writing this code in the code editor, you may have noticed that the different elements use different colors. This is the syntactic coloring. The code editor allows you to easily identify the different elements handled by the code. Therefore, in the code editor that is using the light theme:

- the WLanguage functions are colored in blue,
- the character strings (between quotes) are colored in purple,
- the names of controls are colored in cyan.

Info displays the message passed in parameter on one or more lines. Our message is built from the text "Hello " and from the value of "EDT_FirstName" control. The "," sign separates the parameters of ***Info***. Each parameter is displayed on a different line.

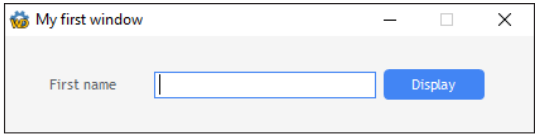
Remark

In this example, the text is displayed in a system window but it can also be displayed it in a window created with WINDEV.

▸ Let's now run the window test:

1. Click among the quick access buttons (or press F9).

2. The window that was just created is automatically saved then it is started in execution.



3. Type your first name.

4. Click the "Display" button.

5. Validate the system window that is displayed (click the "OK" button).

Any developer knows that running a program test can be a long and tiresome job. In WINDEV, a SINGLE CLICK allows you to run the test of window, report or procedure currently created. This is both simple and fast!

▸ Close the test window: click the system closing button of window ("x" button found in the title bar).

▸ The WINDEV editor is redisplayed. Close the code editor: on the "Home" pane, in the "General" group, click "Close".

▸ An error is displayed in the "Compilation errors" pane. This error indicates that the window can be resized while no anchoring is defined. Don't worry: this error will be processed in the next paragraph.

Remark

When developing an application, WINDEV informs you of the possible problems that may occur in your application via the "Compilation errors" pane.
This pane presents different types of errors:

- Compilation errors: The errors signal a major problem (programming error, ...). The project test or execution will not operate.
- Compilation warning: The warnings signal programming cases that may trigger unexpected behaviors.
- Compilation information: The information explains the compiler choices or provides tips for improving the code.
- UI errors: The UI compilation errors inform you of the possible problems detected in your interfaces: images not found, truncated captions, ...
- Programming standard errors: These errors inform you when the programming standard selected in the project description is not respected ("Compilation" tab).
- Automatic test errors: These errors signal a problem when running automatic tests.

Improving the window

During this first test, you have noticed that:

- your window is too big,
- your window can be resized but it only contains two controls,
- the first name is displayed with the case used in the edit control. We are going to force the use of an uppercase letter at the beginning of the word.

We are going to improve the interface of this window.

► To reduce the window size in the editor:

1. Click the window: handles appear around the window.
2. Click the handle found at the bottom right and reduce the window size by keeping the left mouse button down.

► To prevent the window from being resized at run time:

1. Double-click the window. The description window is displayed.
2. In the "UI" tab, uncheck "Resizable" (simply click on the option).
3. Validate.

Remark

Tips for improving the interface and the ergonomics will be presented in a next chapter.

► Run the test of this window to see the result at run time.

► Go back to the editor by closing the test window.

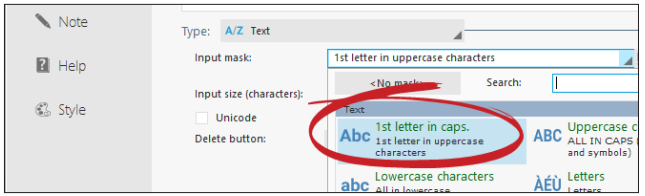
► We are now going to force the use of an uppercase letter when entering and displaying the first name.

Remark

By default, the characters will be typed in uppercase or in lowercase according to the position of [CAPSLOCK] key on the keyboard. This type of input can cause problems when performing a search for example.
WINDEV proposes to manage an input mask for a control. The input mask is used to automatically format the value typed, without programming.

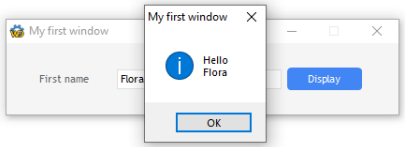
► To modify the input mask:

1. Double-click the "First name" control. The control description window appears.
2. In the "General" tab, expand the "Input mask" list and choose "1st letter in caps.".



3. Validate the control description window.
4. In the editor, the mask name is automatically displayed in the control.

► Run the test of this window to see the result at run time.



Your first window was successfully created. In the rest of this lesson, we are going to discover new WINDEV concepts.

► Validate the message and close the test window (click the cross in the top right corner of window). The editor is redisplayed.

► Close the "WIN_Welcome" window displayed in the editor: on the "Home" pane, in the "General" group, click "Close".

Remark

An example for each type of control

Two types of controls have been used in this lesson: an Edit control and a Button control.

WINDEV proposes more than fifty controls. Several unit examples are available, allowing you to discover the use of each type of control.

▶ To open a unit example:

1. Display the WINDEV home page (press Ctrl + <).
2. Click "Open an example" if necessary.
3. The list of complete examples, training examples and unit examples is displayed.
4. In the "Unit examples" category, select the requested example ("The TreeView control" for example) and double-click its name: the corresponding window is automatically opened in the editor.

Remark: To make things easy, all unit examples corresponding to controls start with the words "The control". These words can be typed in the search control of home page.

▶ Close the project: on the "Home" pane, in the "General" group, expand "Close" and select "Close the project".

WINDEV: main concepts and terminology

After these exercises, let's talk about the main WINDEV concepts and about the terminology specific to WINDEV.

Main concepts

WINDEV allows you to easily create an application. But what is an **Application**?

An **application** is a tool used to automatically perform tasks, actions. An application includes an executable program (or a set of executable programs), libraries, data files, ...

An **executable program** is a file made of elements that can be directly handled by the user (windows, printed reports, ...). It is started by the end user of an application.

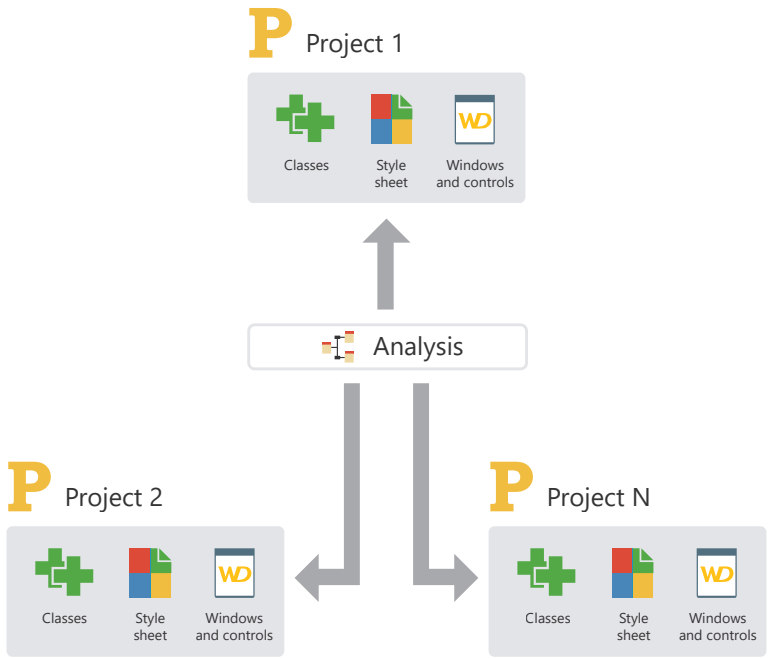
To create an executable, WINDEV proposes to create a **project**. A project links and organizes the different program elements. The executable program will be created from the project.

If your application is using data, WINDEV allows you to define the database structure via **the analysis**. The WINDEV analysis contains the description of the data files (also known as "Tables" in several databases). These data files will contain the application data.

Remark

Describing the data files in the analysis does not mean that they are created. The data files are physically created when running the application.

One or more WINDEV projects can be linked to the same analysis. In this case, we talk of shared analysis. For example, an application for business management can be divided into several executable modules. Each module is using the same analysis (at run time, each executable can also use the same data files).



Terminology

As already seen, a WINDEV project (linked to an analysis if necessary) is used to create an application. Before we actually start working with WINDEV, let's go back to the vocabulary used in WINDEV. Indeed, several terms are specific to WINDEV and they may not have the same significance as the the ones used in the other tools.

The following terms are used **in the analysis**:

- **Data file**: The analysis is used to describe the structure of database files. A "Data file" corresponds to a "table" in some databases.
  In WINDEV, "***Table***" represents a graphic object used to see the content of a data file in a table and/or to enter rows. A table can be used to type the order details for example.

- **Record**: A record is sometimes called row. A data file record corresponds to all items defined for the data file.
- **Item**: In the analysis, an item represents a section of a data file. All items found in a data file are used to define the structure of a record.
- **Key/Index**: With WEBDEV and its HFSQL database, the concept of index is linked to the concept of key. The concept of key is part of the item characteristics. The keys are used to improve the speed for accessing data and to simplify the browse operations performed on the data files. In WINDEV, if a HFSQL data file includes several key items, a single index file will be created at run time.

The following terms are used **in the windows and reports**:

- **Window**: Windows are used to display or type information on the screen. The windows are also called "Screens" or "Dialog boxes". The user can directly act on the windows via controls, buttons, ...
- **Report**: Reports are used to get a custom view of information. This information can come from the database, from text files, from controls found in the windows, ... The reports can be previewed, printed on paper, generated in PDF or in HTML, ...
- **Control**: "Control" is the term used to identify the different graphic objects displayed in a window or report.
- **Skin template**: The skin template is used define the application style: visual appearance of windows, buttons, controls, ...
- **Style**: The style groups the graphic characteristics of an element: background image, border, font, ... The styles of the different elements found in the interface of a WINDEV application are grouped in a style sheet.

Important

In an application, the "CustomerName" entity can correspond to:

- the name of a window control,
- the name of a report control,
- the item of a data file,
- a variable defined by the developer.

Lesson 2.1. The variables

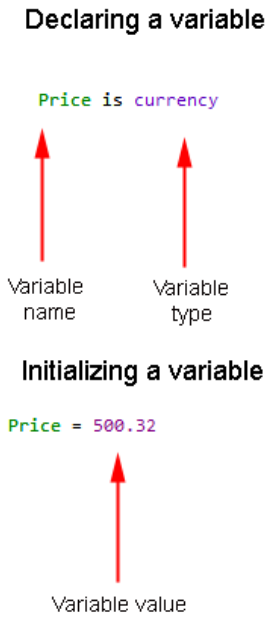
This lesson will teach you the following concepts

- What is a variable?
- The different types of variables.
- The scope of variables.
- The String type in details.
- The Array type in details.

Estimated time: 1 h

What is a variable?

In a programming language, a variable is used to store data. These memory sections contain strings, numbers, etc.
The variables are used to perform calculations, to perform comparisons or to store information that will be used later.



A variable is represented by:
- a name: Name given to the variable so that it can be used by the language.
- a type: Nature of data stored in the variable (see The types of variables).
- a value: Information stored in the variable.
- a scope: Limit for using the variable in the program (see The scope of variables). The scope is mainly defined by the location where the variable is declared.

Declaring a variable

The variable must be declared (which means created) before it can be used.
- Example of simple declaration:

```
Price is currency
```

  - **Price** represents the variable name.
  - **is** is used to declare the variable. The everyday language is used in WLanguage.
  - **currency** corresponds to the variable type.
- Example of multiple declaration:

```
LastName, FirstName are strings
```

  - **LastName**, **FirstName** represent the names of variables.
  - **are** is used to declare a set of variables.
  - **strings** represents the type of variables.

Assignment and use

When the variable is declared, you have the ability to assign it (or to give ti a value).
For example:

```
// Assign a currency variable
Price = 1256.67
// Assign a string variable
LastName = "Doe"
```

The **=** operator is used to perform this assignment.

Remark

In WLanguage, the " character (double quote) is the character used to delimit a character string. In the above example, the doubles quotes are used to assign the Doe value to the LastName variable.

The variable content can be read and handled: all you have to do is use the name given to the variable to access it.
The following example is used to read and display the content of Price variable on the screen:

```
Info(Price)
```

The types of variables

The variable type is used to specify the kind of information that will be stored in the variable. The most common types are:

- boolean (True or False),
- string ("Doe"),
- integer (1234),
- currency (12,32),
- real (7,766666),
- etc.

Important

Use the type corresponding to the information that must be stored. Therefore, you will optimize the memory and you will avoid calculation or process errors when using variables in the WLanguage functions.

Most of these types of variables will be used in this tutorial.
See the online help regarding the relevant type for more details.

Remark

Other types are available, such as arrays, structures, dates, times, ...

Advanced variables are also available. These advanced types group all characteristics of the element currently used in a single variable.

Advanced types can be used to handle XML documents, emails, XLS files, ... This type of variable will be used later in this tutorial.

The scope of variables

The variables can be declared anywhere in the code. However, according to the position of its declaration, the variable cannot be used to perform processes or calculations. We talk of variable scope.

Two types of scope are available:

- Global.
- Local.

Global scope

Global means that the variable has an extended visibility in the code. The variable is visible outside the location where it was declared. Several levels are available:

- Project and Set of procedures,
- Window, Mobile Window, Page, Report.

A variable declared **at project level** has the greatest visibility in the program. The variable is visible anywhere, in all program processes. However, you should not declare too many variables with this scope: indeed, the memory occupied by the variable is always reserved even if the variable is not used. Using a large number of global variables is not recommended in the program architecture. To pass variables to a process, we recommend that you use parameters (see "Parameters and result of procedure" for more details).

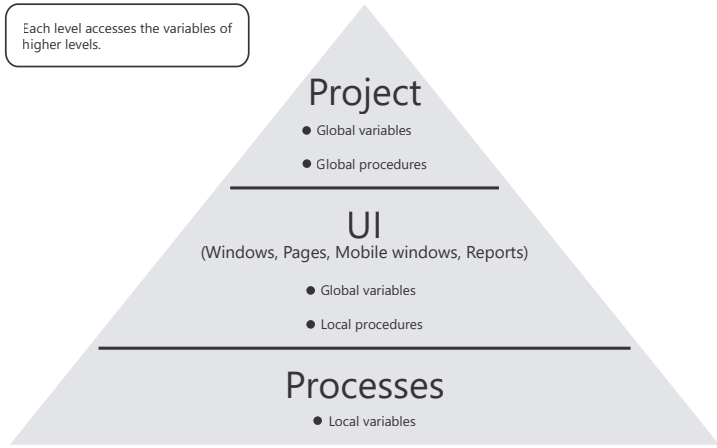
A variable declared **at Set of Procedures level** and a variable declared at project level have the same visibility. The advantage of declaring a variable at the level of a Set is to group (or classify) the variables by theme (set) in order to make the project initialization process more readable.

A variable declared **at Window, Mobile Window, Page or Report level** limits the scope of the variable to all the processes of the element (Window, Mobile Window, Page or Report) and its controls. This makes it possible to encapsulate and limit the uses.

Local scope

Local means that the variable has a limited visibility in the code. The variable is visible in the process where it was declared. This makes it possible to restrict the use of variable to the process.

Summary scope diagram



A variable is global when it is declared:

- in the "Initializing" event of the project (or in the "Declaration" event of the set of procedures). The variable is global to the project.
- in the "Global declarations" event of the window, page or report. The variable is global to the element (window, page or report) where it was declared.

In all other cases, a variable is local to the process or event where it is declared.

Simple operations on the variables

Several mathematical operators can be used to perform calculations on variables:
- + to perform an addition.
- - to perform a subtraction.
- * to perform a multiplication.
- / to perform a division.

Other operators can be used to perform calculations:
- ++ to increment from 1 (add 1 to the variable).
- -- to decrement from 1 (subtract 1 from the variable).
- += to assign by adding a value.
- += to assign by subtracting a value.

Examples:

```
// Declaration of variables
Cnt is int
V1 is int
Res is numeric

// Assignment
Cnt = 10
V1 = 3

// Use of operators
Cnt = Cnt + 3   // Cnt is equal to 13
Cnt ++ // Cnt is equal to 14
Cnt -= 8 // Cnt is equal to 6
Cnt = Cnt * V1 // Cnt is equal to 18
Res = Cnt / 5 // Res is equal to 3.6
```

Comparison operators are also available:
- < less than.
- > greater than.
- <= less than or equal to.
- >= greater than or equal to.
- <> different from.
- = equal to.

Other operators are available. See the online help for more details (keyword: "Operators").

Tips

- It is very convenient to name the variables with long names (and to avoid short names such as i, j, k, ...). When reading the program again, you will be able to easily remember the variable purpose.
- To define the name of variables, all Unicode characters (including the accented characters) are accepted. Meaning improved readability! Caution: some characters are not allowed: space, =, dot, comma, ...
- It is very important to give the proper type to the variable according to its use. For example, to store several digits, you may have to:
  - use a numeric variable if this variable must be used for calculations.
  - use a string variable if this variable must be used to store digits without performing calculations (to store the social security number for example).

Details of variable type: the String variables

The String variables are the most often used types of variables.
Let's present in details some features available for this type of variable.

The String type

The String type is used to store and handle characters and character strings.
We have already seen how to initialize a string variable:

```
LastName is string
// Assign a string variable
LastName = "Doe"
```

There is no need to declare the string length: this length automatically adapts when using the variable.

Tip

To initialize a string variable with a text on several lines, use the following syntax:

```
<Variable name> = [
                <Text of line 1>
                <Text of line 2>
                ]
```

For example:

```
MyString is string
MyString = [
  Example of
  multi-line string
]
```

You also have the ability to assign a string variable with the content of a control handling strings. The following code is used to assign a string variable with the content of an edit control:

```
LastName is string
// Assign a string variable
// with the content of EDT_LastName edit control
LastName = EDT_LastName
```

In addition to the main comparison operators, several powerful operators are used to manage the extractions and concatenations in advanced mode.

Some examples:

- " + ": to concatenate strings.
- " ~= ": to check the flexible equality.

Specific WLanguage functions are used to perform various operations: search, extraction, size, switch to uppercase characters, ...

Examples:

```
str is string
str = "WINDEV is a great tool"
// Extract a sub-string from left
Info(Left(str, 6))  // Displays "WINDEV"
// Extract a sub-string from right
Info(Right(str, 4))  // Displays "tool"
```

Remark

The different WLanguage functions can be nested. A WLanguage function can be used as parameter of another WLanguage function.

For example:

```
Info(Middle(Left(str, 13), 8, 2)) // Displays "is"
```

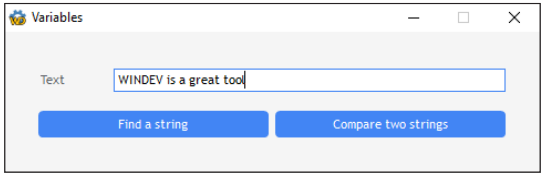
See the help about the character strings and about the functions for handling character strings for more details.

Practical example

To handle the different concepts of this lesson, we are going to create different windows. These windows will be created in the "WD My first windows" project.

▸ Open the "WD My First Windows" project if necessary.
To do so, in the home page (Ctrl + <), click "Tutorial" and select "My first windows (Exercise)".
Caution: WINDEV proposes to open the local copy or overwrite the project with the source version. To continue working on your project started in the previous section, choose "Open the local copy".

To handle the Character String variables, we are going to create the following window:



This window is used to:

- find a string inside another one.
- compare two strings.

▸ Create a new blank window:

1. Click among the quick access buttons. The window for creating a new element is displayed: click "Window" then "Window". The window creation wizard starts.
2. Select "Blank" and the "Phoenix" skin template.
3. Validate. The window is automatically created in the editor. The window for saving an element is displayed.
4. Specify the element title: "Variables". The element name ("WIN_Variables") is automatically proposed.
5. Click on the "OK" to validate the information displayed in the save window.

▸ To create the edit control containing the string:

1. On the "Creation" pane, in the "Usual controls" group, click .
2. Click the location where the edit control will be created (at the top, in the middle of the window for example).
3. Right-click the control and select "Description".
4. In the "General" tab, specify:
   - The control name: "EDT_Text".
   - The control caption: "Text".
5. Select the "Content" tab. This tab is used to define the default value of edit control. Type "WINDEV is a great tool".
6. Validate the description window. The text typed is directly displayed in the control.

▸ The control content is truncated in the editor. To display the control properly:

1. Select the control.
2. Enlarge the control in width with the sizing handles in order for the content to be entirely displayed.

Finding a string

▸ To create a Button control to search for a string:

1. On the "Creation" pane, in the "Usual controls" group, click Ok.
2. Click the location where the control will be created (below the edit control for example).
3. Click the control that was just created. The text displayed in the control becomes editable. Type the caption: "Find a string".
4. Press Enter to validate the input.
5. Adapt the control size.
6. Display the events associated with the control (F2).
7. Write the following code in the "Click" WLanguage event:

```
sSoughtString is string = "WINDEV"
nPos is int
nPos = Position(EDT_Text, sSoughtString)
IF nPos = 0 THEN
  Info(sSoughtString + " not found in the text")
ELSE
  Info(sSoughtString + " found at position " + nPos)
END
```

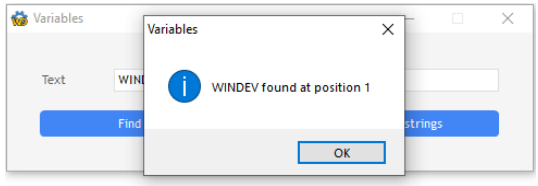
Let's take a look at this WLanguage code:

- Two variables are declared: a String variable corresponding to the sought string and an Integer variable corresponding to the position of sought string.
- **Position** is used to find a string inside another one. In our example, we search for the content of sSoughtString in the content of EDT_Text control.
- If ***Position*** returns 0, it means that the sought string was not found. A message is displayed by ***Info***.

8. Close the code window (click X at the top right corner of code editor).

▸ Let's now run the window test:

1. Click among the quick access buttons (or press F9).
2. The created window is saved and started in execution. Click "Find a string". The WINDEV word is found.



3. Modify the content of edit control (by replacing WINDEV by WD for example) and click "Find a string". The WINDEV word is not found.
4. Close the window.

Comparing two strings

▸ To create a Button control to compare two strings:

1. On the "Creation" pane, in the "Usual controls" group, click Ok.
2. Click the location where the control will be created (for example, to the right of the existing Button control).
3. Click the control that was just created. The text displayed in the button becomes editable. Type the caption: "Compare two strings".
4. Press Enter to validate the input.
5. Adapt the control size.
6. Display the events associated with the control (F2).
7. In the "Click" event of the control, write the following WLanguage code:

```
sStringToCompare is string = "WINDEV"

IF EDT_Text ~= sStringToCompare THEN
  Info("The displayed text corresponds to " + sStringToCompare)
ELSE
  Info("The displayed text does not correspond to " + ...
       sStringToCompare)
END
```

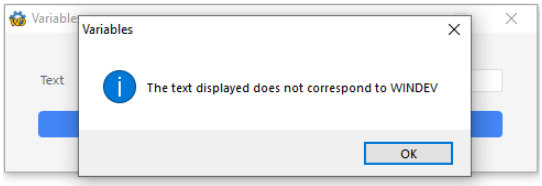
Let's take a look at this WLanguage code:

- A String variable is declared. This variable contains the string to compare.
- The ~= operator corresponds to a flexible equality. This operator is used to perform a comparison while ignoring the case, the spaces found before and after the string and the accented characters.
- If the comparison is true, the edit control corresponds to the word found in the string to compare regardless of the case used.

8. Close the code window (click X at the top right corner of code editor).

▸ Let's now run the window test:

1. Click among the quick access buttons (or press F9).
2. Click "Compare two strings". A message indicates that the comparison is not correct.



3. Validate the message.
4. In the edit control, type "WINDEV" in uppercase characters and click "Compare two strings". The WINDEV word is found. Validate the message.
5. Modify the content of edit control (by replacing WINDEV by WINDEV for example) and click "Compare two strings". The WINDEV word is also found. Validate the message.
6. Close the window.

Details of another variable type: the arrays

The arrays are a common type of variable.

An array is a structured type that is used to group a set of elements of the same type. Each array element can be directly accessed by its subscript.

Specific WLanguage functions are used to handle the arrays and their elements. These functions start with "Array".

Declaration

The declaration of an Array variable is performed as follows:

```
<Array name> is array of <Type of array elements>
```

For example:

```
arrString is array of strings
arrInt is array of int
```

Filling an array and accessing the elements

During its declaration, the array is empty. The elements are added by **Add** via the following syntax:

```
Add(<Array name>, <Element value>)
```

To access the array elements, use the following syntax:

```
<Array name> [<Element subscript>]
```

Important

The subscripts of array elements start from 1.

Example:

```
// Create an array of strings
MyArray is array of strings
// Add elements
Add(MyArray, "WINDEV")
Add(MyArray, "WEBDEV")
Add(MyArray, "WINDEV Mobile")
// Display the content of third element
Trace(MyArray[3])  // Displays "WINDEV Mobile"
```


Tip

Fast array initialization

To initialize an array, you also have the ability to use the following syntax:

```
// Declares an array
arrDay is array of strings
// Initialization with
// the names of the days of week
arrDay = ["Monday", "Tuesday", "Wednesday", ...
"Thursday", "Friday", "Saturday", "Sunday"]
```

See the online help for more details (keyword: "Array").

Advanced arrays

Advanced arrays are also available: array with several dimensions, array of arrays, associative array, array of structures, array of objects, ...
See the online help for more details (keyword: "Array").

Lesson 2.2. The conditional statements

This lesson will teach you the following concepts

- The IF statement.
- The SWITCH statement.
- Practical example.

Estimated time: 1 h

Overview

WLanguage is a set of statements used to handle data.

The conditional statements are used to compare variables and/or values between themselves in order to perform different processes. Several conditional statements are available:

- IF... THEN... ELSE... END
- SWITCH...

The IF and SWITCH statements

The IF statement

This statement is used to run an action or another one according to the result of an expression. If the expression is checked, a process is run ; if the expression is not checked, another process can be started.

The IF statement can be used as follows:

```
IF <Expression to check> THEN
   Process to run if the expression is checked
ELSE
   Process to run otherwise
END
```

WLanguage code example: The following code selects a number at random and displays a message according to the value.

```
Tot is Currency
// Selects a number at random between 100 and 4000
Tot = Random(100, 4000)
IF Tot>2000 THEN
  Info("The amount is greater than 2000")
ELSE
  Info("The amount is less than or equal to 2000")
END
```

In this case, the expression to check corresponds to "Tot>2000".

Remark: Several code lines can be run during the process corresponding to a condition. In this case, the following syntax must be used:

```
IF <Expression to check> THEN
  Code line 1
  Code line N
ELSE
  Code line 1
  Code line N
END
```

The SWITCH statement

This statement is used to evaluate an expression and to run a process for each possible expression value.

The SWITCH statement can be used as follows:

```
SWITCH <Expression>
CASE Value 1:
  Process 1...
CASE Value 2:
  Process 2...
...
CASE Value N:
  Process N...

OTHER CASE
  Process ...
END
```

WLanguage code example: The following code retrieves today's date and displays a different message according to its value. A specific message is displayed for the 1st and for the 15th of the month. In the other cases, today's date is displayed.

```
D is Date
D = Today()
SWITCH D..Day      // Checks the day of the date
  CASE 1: Info("We are the first day of the month")
  CASE 15: Info("We are the 15th of the month")
  OTHER CASE: Info("We are the: " + DateToString(D))
END
```

Remark

Several functions and properties are available in WLanguage. The functions can accept parameters and they return results. The properties are directly applied to the controls or variables via the following syntax:

```
<Name of control or variable>..<Property name>
```

In our example, **..Day** is used on the D variable to get the day of the date.

The online help of a function or property can be displayed at any time by pressing F1. See "How to access the online help ?" for more details.

Remarks:

- If the code line "CASE 1:..." is run, the other code lines corresponding to the possible values are not run.
- Several values can be grouped in the same case. The different values are separated by a comma. For example:

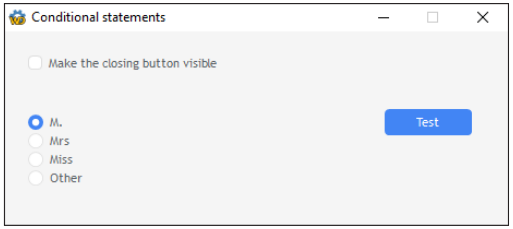
```
Sub is int = 2
SWITCH Sub
 CASE 1,2: Info("Case 1 or 2")
 CASE 3: Info("Case 3")
 OTHER CASE: Info("Other case")
END
```

- Several code lines can be run during the process corresponding to an expression value. In this case, the following syntax must be used:

```
SWITCH <Expression>
CASE Value 1:
 Process 1 - Code line 1...
 Process 1 - Code line 2...
CASE Value N:
 Process N - Code line 1...
 Process N - Code line 2...
OTHER CASE
 Process ...
END
```

Practical example: Using the IF and SWITCH statements

To use the conditional statements, we are going to create the following window:



Two operations can be performed:

- If the user clicks the check box, the "Close" button is displayed.
- If the user clicks the "Test" button, the box checked in the check box is displayed.

Project used

To handle the different concepts of this lesson, we are going to create different windows. These windows will be created in the "WD My first windows" project.

▸ Open the "WD My First Windows" project if necessary.
To do so, in WINDEV's home page (Ctrl + <), click "Tutorial" and select "My first windows (Exercise)".

Creating the window

▸ Create a new blank window:

1. Click among the quick access buttons. The window for creating a new element is displayed: click "Window" then "Window". The window creation wizard starts.
2. Select "Blank" and the "Phoenix" skin template.
3. Validate. The window is automatically created in the editor. The window for saving an element is displayed.
4. Specify the window title: "Conditional statements". The window name ("WIN_Conditional_statements") is automatically proposed.
5. Click "OK" to validate the information shown in the save window.

Creating the window controls for the conditional IF statement

Two controls must be created:

- a Check Box control used to display (or not) the "Close" button.
- a "Close" Button control to close the window.

▸ To create the Check Box control:

1. On the "Creation" pane, in the "Usual controls" group, click "Check box" then click the position where the control will be created in the window (at the top of window for example).
2. Click the control that was just created: the "Option 1" caption becomes editable.
3. Type the option caption: "Make the closing button visible".
4. Press Enter to validate the input.

▸ The option caption is truncated in the editor. To display the control properly:

1. Select the control.
2. In the popup menu (right click), select "Adapt the size".
3. The control is immediately enlarged and the caption is entirely displayed.

▸ To create the "Close" Button control:

1. On the "Creation" pane, in the "Usual controls" group, click Ok.
2. Click the location where the control will be created (for example, on the right of Check Box control).
3. Click the control that was just created. The text displayed in the control becomes editable. Type the caption: "Close".
4. Press Enter to validate the input.

▸ This Button control is used to close the window. We are going to type the corresponding WLanguage code:

1. Select the "Close" Button control.
2. Press F2: the code editor displays the events associated with the control.
3. Write the following WLanguage code in the "Click BTN_Close" event:

```
Close()
```

Close is used to close the current window.

4. Close the code window (click X at the top right corner of code editor).

▸ The "Close" Button control must be invisible when the window is opened. This information corresponds to its "initial status". We are going to modify this status in the description window of control:

1. Select the "Close" Button control.
2. Display the popup menu of control and select "Description".
3. In the "UI" tab, modify the initial state of the control by clicking on "Visible". This option is now unchecked.
4. Validate the control description window.
5. The "Close" Button control is still visible in the editor. Its status was changed in execution only.
6. Save the modifications by clicking among the quick access buttons (on the left of the ribbon) or by pressing Ctrl + S.

Conditional IF statement

In our example, a click on the check box must trigger the display of Close button.

▸ We will enter the WLanguage code associated with the Check Box control:

1. Select the Check Box control.
2. Press F2.
3. Write the following WLanguage code in the event "Whenever modifying CBOX_NoName1":

```
IF CBOX_NoName1 = True THEN
  BTN_Close..Visible = True
ELSE
  BTN_Close..Visible = False
END
```

Let's take a look at this WLanguage code:

- CBOX_NoName1 and BTN_Close respectively correspond to the names of Check Box and Button controls.

Tip

To handle the current control in one of the events associated with it, you can use the keyword MySelf.
In our example, you also have the ability to write:

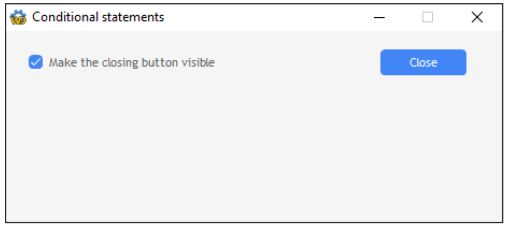
```
IF MySelf = True THEN
```

- The Check Box control is a 2-state control: checked/unchecked. It corresponds in programming to a Boolean variable. If the control is checked, its value is set to True ; if the control is unchecked, its value is set to False.
- This code tests the value of the Check Box control.
  - If Check Box control is checked, Button control BTN_Close becomes visible using **..Visible**.
  - If the Check Box control is unchecked, the Button control BTN_Close becomes invisible.

4. Close the code window (click X at the top right corner of code editor).

▸ Let's now run the window test:

1. Click among the quick access buttons (or press F9). The window is run.



2. In the window in execution, click the check box. The "Close" button becomes visible.
3. Click the "Close" button: the window is closed and the window editor is displayed.

Creating the window controls for the conditional SWITCH statement

Two controls must be created:

- a Radio Button control used to select the value to check.
- a Button control used to run the test.

▸ To create the Radio Button control:

1. On the "Creation" pane, in the "Usual controls" group, expand "Radio button" (click the arrow). The list of default Radio Button controls appears.
2. In the search box at the top of the list of controls, enter "Title". Click the first proposed control (with "M").
3. Click inside the window to create the control (below the Check Box control created beforehand for example).

▸ To create the "Test" Button control:

1. On the "Creation" pane, in the "Usual controls" group, click Ok.
2. Click the location where the control will be created (on the right of Radio Button control, for example).
3. Click the control that was just created. The text displayed in the control becomes editable. Type the caption: "Test".
4. Press Enter to validate the input.

SWITCH statement

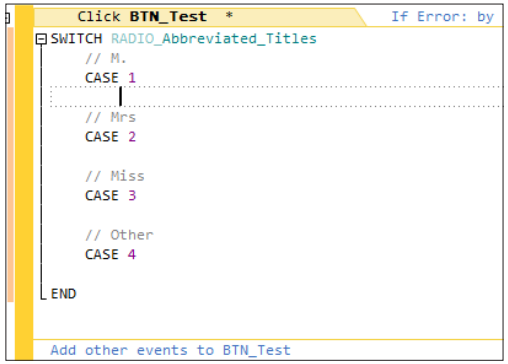
In our example, the "Test" button is used to check the selected value and display a message.

▸ We are going to enter the WLanguage code associated with the "Test" Button control:

1. Select the "Test" Button control.
2. Press F2.
3. Write the following WLanguage code in the "Click BTN_Test" event:

```
SWITCH RADIO_Abbreviated_Titles
```


4. When the Enter key is pressed to go to the next line, the code editor automatically displays the different SWITCH capabilities:



5. Write the following WLanguage code:

```
SWITCH RADIO_Abbreviated_Titles
  // Mr
  CASE 1
      Info("You've selected Mr")
  // Mrs
  CASE 2
      Info("You've selected Mrs")
  // Miss
  CASE 3
      Info("You've selected Miss")
  // Other
  CASE 4
      Info("You've selected Other")
END
```

Let's take a look at this WLanguage code:

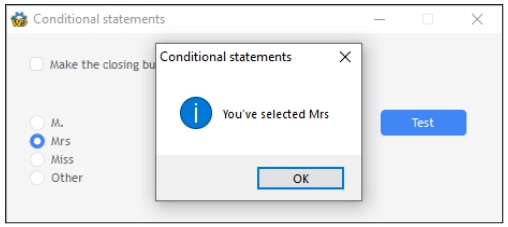
- RADIO_Abbreviated_Titles corresponds to the name of the Radio Button control.
- The Radio Button control is a control proposing several options (4 in our case). A single option can be checked at a time. The Radio Button control corresponds in programming to an Integer variable. Each menu option is associated with a value. If this option is checked, the Radio Button control takes for value the identifier of checked option.
- This code tests the value of the Radio Button control. A message is displayed according to the value of Radio Button control.

6. Close the code window (click X at the top right corner of code editor).

▸ Let's now run the window test:

1. Click among the quick access buttons (or press F9).

2. In the window whose test is run, select an option and click "Test": the message corresponding to the selected option is displayed.



3. Close the window.

Lesson 2.3. The loops

This lesson will teach you the following concepts

- The FOR statement.
- The LOOP statement.
- The WHILE statement.
- Practical example.

Estimated time: 1 h

Overview

The loop statements are used to run a process in a recurring way. A specific loop statement is used according to the number of occurrences. Several statements can be used to perform loops:

- FOR...
- LOOP...
- WHILE...

The FOR statement

The FOR statement is used when the number of occurrences to process is known. This statement is used to manage the number of occurrences via a variable in which the passages performed in the loop will be counted.

The syntax of FOR statement is as follows:

```
FOR Subscript = Start Value TO End Value
  Process to run
END
```

For example, the following code runs the process 2000 times:

```
FOR Cnt = 1 TO 2000
  // Process to run
END
```

Remark: An increment step of subscript can be defined via the STEP keyword. For example, the following code runs the process 2000 times and the Cnt variable decreases by 10:

```
FOR Cnt = 2000 TO 1 STEP -10
  // Process to run
END
```

The LOOP statement

The LOOP statement is used to perform loops when the number of occurrences to process is unknown. In this case, a test must be used to exit from the loop.

The syntax of LOOP statement is as follows:

```
LOOP
  Process to run
  IF <Expression> THEN BREAK
END
```


For example:

```
Counter is int
Counter = 10
LOOP
  // Process to run
  Counter = Counter - 1
  IF Counter = 0 THEN BREAK
END
```

Tip

The LOOP statement and the FOR statement can have the same behavior: all you have to do is use the syntax with exit according to the number of iterations:

```
LOOP (<Number of iterations>)
...
END
```

Example:

```
LOOP(10)
  // Process to run
END
```

The WHILE statement

The WHILE statement and the LOOP statement operate according to the same principle. The difference is that the test of exit condition is performed BEFORE running the loop code. This test is used to compare a variable. This variable starts from a start value and it is modified in the loop until it reaches the value that triggers the exit from the loop.

The syntax of WHILE statement is as follows:

```
<Initialize the variable to its start value>
WHILE <Compare the variable to its end value>
  Process to run
  <Modify the variable>
END
```

For example:

```
Counter is int
Counter = 0
WHILE Counter<10
  // Process to run
  Counter = Counter + 1
END
```

Practical example: Using loops

To check the use of loops, we are going to create a window into which an Image control will be moved.

Project used

To handle the different concepts of this lesson, we are going to create different windows. These windows will be created in the "WD My first windows" project.

▸ Open the "WD My First Windows" project if necessary.
To do so, in WINDEV's home page (Ctrl + <), click "Tutorial" and select "My first windows (Exercise)".

Creating the window

▸ Create a new blank window:

1. Click among the quick access buttons. The window for creating a new element is displayed: click "Window" then "Window". The window creation wizard starts.
2. Select "Blank" and the "Phoenix" skin template.
3. Validate. The window is automatically created in the editor. The window for saving an element is displayed.
4. Specify the window title: "Loops". The window name ("WIN_Loops") is automatically proposed.
5. Click "OK" to validate the information shown in the save window.

Creating controls

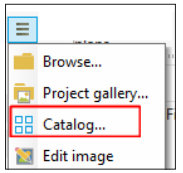
We are going to create the Image control then the buttons used to handle this image.

▸ To create an Image control:

1. On the "Creation" pane, in the "Usual controls" group, click "Image". Position the control in the window.
2. Right-click the control and select "Description".
3. We are going to associate an image to the Image control via the image catalog of WINDEV.

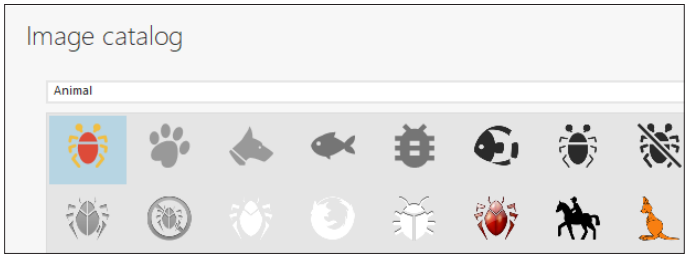
> **Remark**
> As soon as an image can be displayed in a control or window, WINDEV proposes to use the image catalog. This image catalog is started via the "Catalog" option (available by clicking the ≡ button). This catalog contains hundreds of images, cliparts, ...

4. Click the ≡ button on the right of "Image" control. Select "Catalog" from the popup menu that is displayed.



The window of image catalog is displayed.

5. Type "Animal" in the search area. Click the magnifier to start the search.



6. Select and validate ("OK").
7. A window used to configure the image is displayed. Keep the default options and validate.
8. The image is displayed in the Image control. Validate the control description window.
9. Save the window (press Ctrl + S or click among the quick access buttons).

▸ We are now going to create a Button control to allow the image to move by 300 pixels to the right in the window. To do so, we will be using the FOR statement of WLanguage.

1. On the "Creation" pane, in the "Usual controls" group, click Ok.
2. Click the location where the control will be created (below the Image control for example).
3. Click the control that was just created. The text displayed in the control becomes editable. Type the caption: "FOR statement".
4. Press Enter to validate the input.
5. Adapt the control size.
6. Display the events associated with the control (F2).
7. Write the following code in the "Click" WLanguage event:

```
// Moves the image horizontally by 300 pixels
FOR i = 1 TO 300
  IMG_NoName1..X++
END
```

This code is used to modify the X coordinate of Image control (**..X**) in a loop from 1 to 300. At each loop turn, the X coordinate is increased by one pixel.

Remark

When typing conditional statements in the code editor, the different possible syntaxes are proposed in a list by the code editor.

If you select one of these syntaxes, the statement structure is automatically inserted into the code editor. Simply enter the WLanguage code corresponding to each part of the statement.

8. Close the code window (click X at the top right corner of code editor).

▸ Let's now create a Button control to move the image until it reaches the right border of the window. To do so, we will be using the WHILE statement of WLanguage.

1. On the "Creation" pane, in the "Usual controls" group, click Ok.

2. Click the location where the control will be created (for example, to the right of the "FOR statement" control).

3. Click the control that was just created. The text displayed in the control becomes editable. Type the caption: "WHILE statement".

4. Press Enter to validate the input.

5. Adapt the control size.

6. Display the events associated with the control (F2).

7. Write the following code in the "Click" WLanguage event:

```
// Moves the image horizontally until you reach the window side
WHILE IMG_NoName1..X < WinInWidth() - IMG_NoName1..Width
  IMG_NoName1..X++
END
```

This code is used to modify the X coordinate of Image control (**..X**) until a condition is true. In our case, this condition is as follows: the position of the Image control (**..X**) must correspond to the width of the window minus the width of the Image control.

Tip

When a code line is too long to be displayed in the window of code editor, you have the ability to cut it in 2 by using a carriage return.

8. Close the code window (click X at the top right corner of code editor).

▸ Now let's create a last Button control to move the Image control until it reaches the left border of the window. To do so, we will be using the LOOP statement of WLanguage.

1. On the "Creation" pane, in the "Usual controls" group, click Ok.

2. Click the location where the control will be created (for example, to the right of the "WHILE statement" control).

3. Click the control that was just created. The text displayed in the control becomes editable. Type the caption: "LOOP statement".

4. Press Enter to validate the input.

5. Adapt the control size.

6. Display the WLanguage events associated with the control (F2).

7. Write the following code in the "Click" WLanguage event:

```
// Moves the image to the left
LOOP
  IMG_NoName1..X--
  IF IMG_NoName1..X <= 0 THEN BREAK
END
```

This code is used to modify the X coordinate of Image control (**..X**). At each turn, a condition is checked. If this condition is true, the program exits from the loop. In our case, the condition is as follows:

- the position of the Image control is 0,
- the position of the Image control is less than 0 (used to manage the case where the user presses the button several times).

Remark

When typing this code, the LOOP statement may be underlined by a green line and a warning may appear in the pane of compilation errors.

This warning reminds you that an exit statement must be found in the loop code to avoid an infinite loop.

8. Close the code window (click X at the top right corner of code editor).

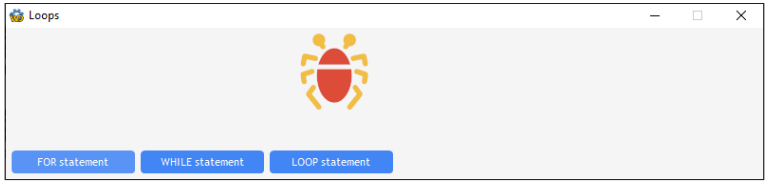
Window test

▸ Let's now run the window test:

1. Click among the quick access buttons (or press F9).

2. The created window is started in execution.

3. Click the different buttons.



4. Close the test window.

Lesson 2.4. The procedures

This lesson will teach you the following concepts

- Types of procedures.
- Creating and calling a procedure.
- Procedure parameters.
- Procedure result.
- Practical example.

Estimated time: 1 h

Overview

A procedure is used to associate an identifier with a code section in order to re-use it.
In this lesson, we are going to present the different types of procedures available in WLanguage, their creation mode, how to call them, pass parameters and retrieve a result.

Types of procedures

Three types of procedures are available:

- **Global procedure**: can be used in all project the events and processes of the project (declared in a set of procedures).
- **Local procedure** in a Window, Page or Mobile Window: can be used in all the events and processes that depend on the object in which this procedure was declared.
- **Internal procedure** in a process or event: can only be used in the process or event where it was declared.

Remark

Scope of procedures
The procedures comply with the scope rules presented for the variables (see "The scope of variables").

Creating and calling a procedure

Global procedure

To create a global procedure, you must:

1. Create (if necessary) a set of procedures (via the "Project explorer" pane, "Procedures" folder). Give a name to the set of procedures.
2. Create a global procedure in the set of procedures (via the "Project explorer" pane, "Procedures, Set name" folder). Give a name to the procedure.
3. Type the code of global procedure. The procedure code has the following format:

```
PROCEDURE <Name of global procedure>()
```

Local procedure

To create a local procedure, you must:

1. Select the element associated with the procedure (window, page, etc).
2. Create a local procedure (via the "Project explorer" pane, expand the element name, "Local procedures" folder).
3. Give a name to the procedure.
4. Type the code of local procedure. The procedure code has the following format:

```
PROCEDURE <Name of local procedure>()
```

Internal procedure

To create an internal procedure, type the following code in the requested process:

```
INTERNAL PROCEDURE <Procedure name>()
<Code of internal procedure>
END
```

Calling a procedure

To call a procedure, use the procedure name (with the possible parameters that will be passed to it).

```
<Procedure name>(<Parameter 1>, ..., <Parameter N>)
```

See the online help for more details (keyword: "Procedure").

Procedure parameters

What is a parameter?

A parameter is a value sent to a procedure during the call to the procedure.

The following example is used to call the **Multiply10** procedure and to pass in parameter the value that will be handled in the procedure:

```
Multiply10(50)
```

You have the ability to pass from 0 to several values in parameter to a procedure. These values can have any type (as with the variables).

The parameter is specified in the procedure declaration in the format of a variable. For example, for the **Multiply10** procedure, the WLanguage procedure code is:

```
PROCEDURE Multiply10(P)
P=P*10
```

P is the parameter expected by the procedure.

Remark

To specify the parameter role in the procedure, you have the ability to typecast the parameter in the procedure declaration.

For example, to use numeric values only, you have the ability to declare:

```
PROCEDURE Multiply10(P is numeric)
```


In the following example, the **Multiplication** procedure expects two Integer parameters and returns the multiplication result.

The WLanguage code of the procedure is:

```
PROCEDURE Multiplication(Nb1 is int, Nb2 is int)
MyResult is int
MyResult = Nb1 * Nb2
RESULT MyResult
```

The WLanguage code used to call the procedure is:

```
res is int
res = Multiplication(10, 50)
// Res is equal to 500
```

How to use the parameters?

By default, **passing parameters in WLanguage is performed by reference** (or by address). The parameter in the procedure represents (references) the variable passed during the call.

Therefore, when a procedure statement modifies the parameter value, the value of the variable corresponding to this parameter is modified.

Example:

- The WLanguage code of the procedure is:

```
PROCEDURE Test_address(P1)
P1 = P1 * 2
```

- The WLanguage code used to call the procedure is:

```
T is int
T = 12 // T is equal to 12 before the call
Test_address(T)
// T is equal to 24 after the call
```

To avoid modifying the value of the variable corresponding to the parameter, the **parameters must be passed by value**. Passing parameters by value allows you to handle a copy of parameter value. If the procedure code modifies the variable value, the value of the variable corresponding to the parameter is not modified.

To force a parameter to be passed by value to a procedure, the LOCAL keyword must be used in front of the parameter name in the procedure declaration. This keyword indicates that the following parameter will not be modified by the procedure.

Example:

- The WLanguage code of the procedure is:

```
PROCEDURE Test_value(LOCAL P1)
// Local indicates that the parameter will be passed by value
P1 = P1 * 2
```

The WLanguage code used to call the procedure is:

```
T is int
T = 12 // T is equal to 12
Test_value(T)
// T does not change
```

Remark

In the same procedure, some parameters can be passed by reference while other parameters can be passed by value. All you have to do is used the LOCAL keyword in front of each parameter passed by value.

Mandatory or optional parameters?

The parameters received in the procedure can be mandatory or optional parameters. A mandatory parameter must be filled during the call to the procedure while an optional parameter can be omitted: in this case, it will take the default value defined when declaring the procedure parameters.

Remark

When declaring a procedure, the optional parameters are the last parameters (they are always specified after all mandatory parameters).

In the following example, the **Multiplication** procedure is using an optional parameter, Nb2. This optional parameter is indicated after the mandatory parameters, by specifying its default value. In this example, the default value of optional parameter is set to 10.

```
PROCEDURE Multiplication(Nb1 is int, Nb2 is int=10)
MyResult is int
MyResult = Nb1 * Nb2
RESULT MyResult
```

The code used to call the procedure is as follows:

```
res is int
res = Multiplication(6)
// Res is equal to 60
```

In this example, the second parameter was not specified. Therefore, its default value will be used.

Procedure result

The procedures can return one or more results. The result can be typecasted. The RESULT keyword must be used to return a value.

See the online help for more details (keyword: Result).

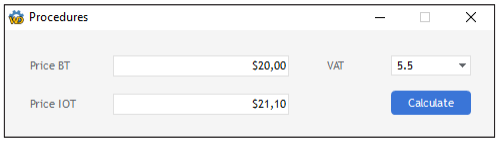
Practical example: Using a procedure

In a new window, we are now going to:

- Create two numeric edit controls containing the value BT and the value IOT.
- Create a Combo Box control used to choose the VAT rate.
- Create a Button control used to calculate and display the value IOT of the amount BT entered.

The calculation result will be displayed in the "Price IOT" control.

This window is as follows:



Project used

To handle the different concepts of this lesson, we are going to create different windows. These windows will be created in the "WD My first windows" project.

▸ Open the "WD My First Windows" project if necessary.
To do so, in WINDEV's home page (Ctrl + <), click "Tutorial" and select "My first windows (Exercise)".

Implementation

▸ Create a new blank window:

1. Click among the quick access buttons. The window for creating a new element is displayed: click "Window" then "Window". The window creation wizard starts.
2. Select "Blank" and the "Phoenix" skin template.
3. Validate. The window is automatically created in the editor. The window for saving an element is displayed.
4. Specify the window title: "Procedures". The window name ("WIN_Procedures") is automatically proposed.
5. Click "OK" to validate the information shown in the save window.

▸ To create the edit control corresponding to the price BT:

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit" (click the arrow). The list of available edit controls is displayed. In the search box at the top of the list, enter "Currency". Select the "Currency Edit" control and position the control in the window.
2. Right-click the control and select "Description".
3. In the description window:
   - Type the control name: "EDT_PriceBT".
   - Type the caption: "Price BT".
4. Validate.

► To create the edit control where the result will be displayed:

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit" (click the arrow). The list of available edit controls is displayed. In the search box at the top of the list, enter "Currency". Select the "Currency Edit" control and position the control in the window.

2. Type the control information: right-click the control and select "Description".

- Specify the control name: "EDT_PriceIOT".
- Modify the caption to "Price IOT".

3. The result displayed in this control must not be modifiable. Click the "UI" tab and choose "Read-only" for the initial status.

4. Validate.

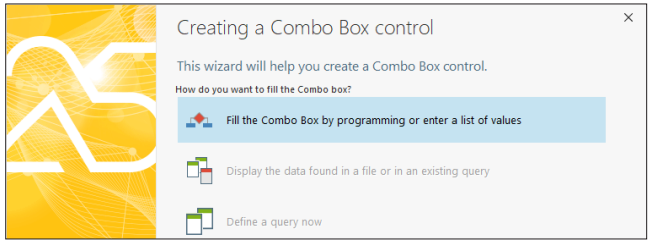
5. Save the window.

► To create the Combo Box control for VAT selection:

1. On the "Creation" pane, in the "Usual controls" group, click "Combo box" then click the position where the control will be created in the window (beside the "Price BT" control).

2. The Combo Box control creation wizard for is displayed. This wizard is used to define the main control characteristics.

3. Select "Fill the combo box by programming or enter a list of values".

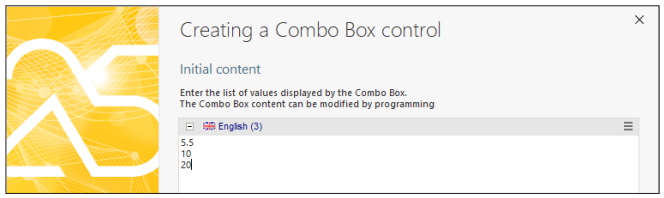


4. Display the next step of the wizard.

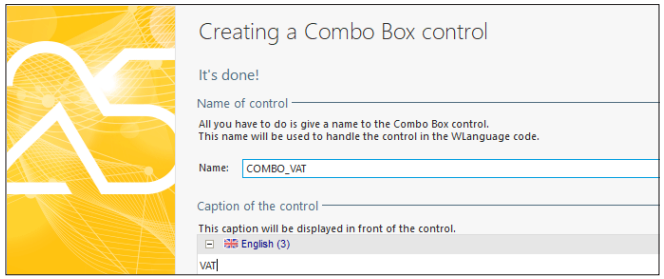
5. Keep the default options. Display the next steps.

6. In the "Initial content" step, type the list of possible VAT values:

- 5.5
- Press the Enter key.
- 10
- Press the Enter key.
- 20.



7. Display the next wizard step: give a name (COMBO_VAT) and a caption (VAT) to the control.



8. Validate.

► To create the "Calculate" Button control:

1. On the "Creation" pane, in the "Usual controls" group, click Ok.

2. Click the location where the control will be created (below the Combo Box control for example).

3. Click the control that was just created. The text displayed in the control becomes editable. Type the caption: "Calculate".

4. Press Enter to validate the input.

► The amount IOT will be calculated via the code of the "Calculate" Button control.

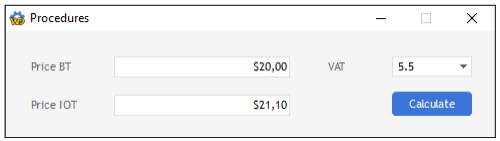
1. Display the code of "Calculate" control ("Code" from the popup menu).

2. Write the following WLanguage code in the "Click BTN_Calculate" event:

```
SWITCH COMBO_VAT..DisplayedValue
  // 5.5 %
  CASE 5.5
      EDT_PriceIOT = EDT_PriceBT * 1.055
  // 10 %
  CASE 10
      EDT_PriceIOT = EDT_PriceBT * 1.1
  // 20 %
  CASE 20
      EDT_PriceIOT = EDT_PriceBT * 1.2
END
```

This code calculates the amount IOT by using the value selected in the Combo Box control (returned by **..DisplayedValue**).

3. Close the code window (click X at the top right corner of code editor).
4. Run the window test (among the quick access buttons). Give a price BT. Select the different values in the Combo Box control and click "Calculate".



Our window operates properly now. **However, instead of using a formula 3 times to calculate the price, a procedure can be used to perform this calculation.**

► Close the test window to go back to the code editor.

Creating and using a procedure

► To create the procedure for calculating the amount IOT:

1. Click the window background.
2. Display the events associated with the window (F2).
3. In the code editor, on the "Code" pane found in the ribbon, in the "Procedures" group, expand "New" and select "New local procedure".
4. In the window that is displayed, type the name of local procedure ("Calc_IOT") and validate ("Add" button).
5. The new procedure local to the window is automatically created and its code is displayed in the code editor.
6. Write the following WLanguage code:

```
PROCEDURE Calc_IOT(PriceBT, VATRate)
cyIOT is currency
cyIOT = PriceBT * (1 + VATRate/100)
RESULT cyIOT
```

Let's take a look at this WLanguage code:

- The Calc_IOT procedure expects 2 parameters: the price before tax and the VAT rate.
- This procedure declares a currency variable. This variable is used to store the calculated price IOT.
- This procedure calculates the price IOT and returns the calculated value.

7. Close the code window (click X at the top right corner of code editor).

Remark

When creating a procedure, comments are automatically generated BEFORE the procedure code. These comments are mainly used to specify the content of parameters and return value.

It is important to fill these comments. Indeed, they will be automatically displayed in a tooltip when typing the call to the procedure in the code editor.

► Let's now call the procedure from the calculation Button control.

1. Select the "Calculate" Button control.
2. Display the events associated with the Button control (press F2, for example).
3. In the "Click" event, replace the existing code with the following code:

```
SWITCH COMBO_VAT..DisplayedValue
  // 5.5 %
  CASE 5.5
      EDT_PriceIOT = Calc_IOT(EDT_PriceBT, 5.5)
  // 10 %
  CASE 10
      EDT_PriceIOT = Calc_IOT(EDT_PriceBT, 10)
  // 20 %
  CASE 20
      EDT_PriceIOT = Calc_IOT(EDT_PriceBT, 20)
END
```

This code calls the Calc_IOT procedure to calculate the amount IOT. Two parameters are passed to the procedure: the price BT and the VAT rate. The result returned by the procedure is assigned to the EDT_PriceIOT control.

4. Close the code window (click X at the top right corner of code editor).

► Let's now run the window test:

1. Run the window test (among the quick access buttons). Give a price BT. Select the different values in the Combo Box control and click "Calculate".
2. The amount IOT is displayed.
3. Close the test window.

Conclusion

This section allowed you to get familiar with the main concepts of WLanguage programming in WINDEV. Several other features are available. Some of them will be presented in this tutorial.

WLanguage is a very powerful language that allows you to develop applications that use:

- the Object-Oriented Programming (OOP),
- the MVP (Model View Presenter),
- a 3-tier programming,
- advanced types (XML, Email, ...), ...

See the online help regarding the relevant topic for more details.

Lesson 2.5. Questions/Answers

This lesson will teach you the following concepts

- Questions/Answers.

Estimated time: 10 mn

Questions/Answers

How to view the element to which the current event belongs?

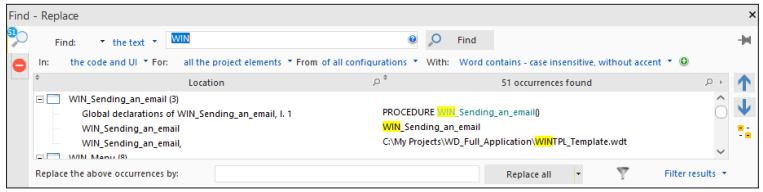
To view the element corresponding to the current event, go to "Code" pane, "Navigation" group, click "Go to element" (Ctrl + Alt + F2). The page containing the requested element is displayed.

How to print the source code?

To print the current source WLanguage, click in the "Home" pane or press Ctrl + P.

How to perform a "find and/or replace"?

The functions for performing searches or replacements (in the code, in the interface, ...) can be accessed from the "Home" pane, in the "Find" group or in the "Find - Replace" pane (Ctrl + F):



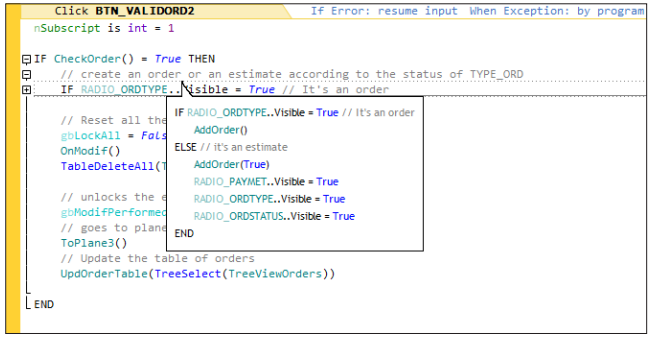
Then, you have the ability to define the different characteristics of the search to perform.

What is the meaning of "+" and "-" signs in the code editor?

The code editor allows you to expand or collapse the WLanguage code. This feature is very useful if your processes use a lot of structured statements (loops, conditions, browses, ...).

To collapse a group of lines of code, on the "Display" pane, expand "Collapse" and select "Collapse all" (or press Ctrl + Shift + * (on the numeric keypad)).

Only the comments remain visible. Hovering over each comment allows you to view the associated WLanguage code in a tooltip:



Press Ctrl + * (on the numeric keypad) to expand the WLanguage entire code. A click performed on the "-" or "+" symbol allows you to collapse or expand the corresponding code section.

Is it possible to identify the person who wrote a code line?

Press F6 to display information (name and creation/modification date) about each code line.

Is it possible to find out the code line number?

In the code editor, to enable (or not) the numbering of code lines, on the "Display" pane, in the "Help for edit" group, click "Display line numbers" (shortcut: Ctrl + Shift + G).

Is there a method to easily display the syntax or the help about a function?

When entering a WLanguage function in the code editor:

- the name of each parameter is displayed in a tooltip for the code line currently in edit. For the parameter currently in edit, hovering the parameter name with the mouse is used to display the parameter details in a tooltip.
  If several syntaxes are available, it is possible to switch from one syntax to another using Ctrl + up arrow or down arrow.
- the syntax of the WLanguage function is displayed in the status bar of the editor.

In the help, the parameters enclosed between [and] are optional parameters.

For WLanguage functions that require names of data files, controls, windows or reports, the assisted input is used to display the list of project elements corresponding to the parameter of the function currently typed.

Examples of assisted input for **HReadFirst**: the <Wizard> option is used to start a code wizard. This wizard asks you questions regarding the function use and it automatically generates the corresponding code.

A help page is associated with each WLanguage function and property. This help page can be directly accessed from the editor by pressing F1 on the name of requested function or property.

What are the useful shortcuts in the code editor?

- To display the events of a control, method, class, procedure or report block, press F2 while on the name of the desired element.
- Ctrl + F2 is used to go back to the initial process or event.
  To move from one process/event to another, press F2 repeatedly. To go back to the initial process or event, press Ctrl + F2 the same number of times.
- Ctrl + L deletes the current line.
- Ctrl + D duplicates the current line (or the selected lines) on the line below.
- Tab (tab key) and Shift + Tab are used to manage the indent for the selected lines.
- Ctrl + / converts the selected lines into comments, Ctrl + Shift + / removes the comments (Caution: / key on the numeric keypad).
- Ctrl + R is used to automatically re-indent the displayed code.

How to communicate with the user?

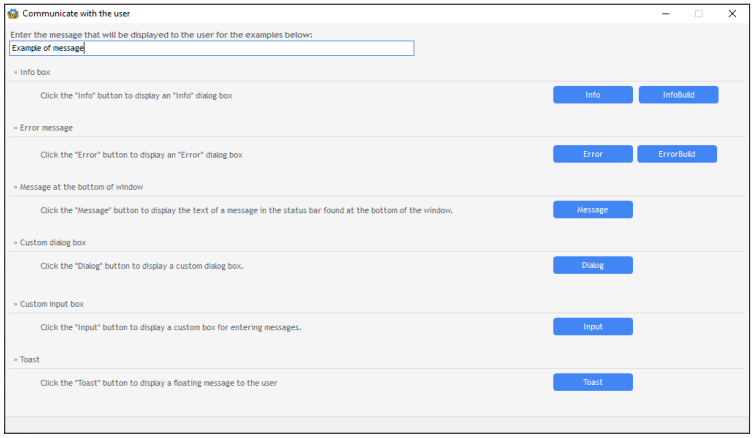
All you have to do is use an advanced dialog box. These dialog boxes are used to manage:

- the directive questioning (**Dialog**): the user answers a question via the buttons containing the text of the action to perform.
- the immediate input (**Input**), by allowing the user to type the requested value in the dialog box.

See the online help for more details (keyword: "Dialog" and "Input").

▶ To check the different modes for communicating with the user:

1. Open the "WD My first windows" project if necessary: on the "Home" pane, in WINDEV's home page (Ctrl + <), click "Tutorial" and select "My first windows (Answer)".
2. Open the "WIN_Dialog" window. This window presents the different dialog modes.
3. Test this window (among the quick access buttons).
4. Click the different buttons to check the different dialog modes.



5. Stop the test of this window and go back to the code editor to study the code of each Button control.

Lesson 3.1. Overview

This lesson will teach you the following concepts

- Overview of the project developed in this part.

Overview of the project developed in this part

We are going to create a project associated with a HFSQL database.

This will allow you to discover the main elements for creating an application:

- Creating a WINDEV project.
- Describing the data files used by the application.

In a next section, we will concentrate on the development of the elements (windows, reports, ...) found in an application with data that will be developed from A to Z, from the interface creation to the final distribution. You will see the main points for developing an application.

In this section, we are going to create a database used to manage orders.

The same type of database will be used in part 4 of this tutorial to develop a full application. The database used is HFSQL Classic, the free database supplied with WINDEV. The HFSQL Client/Server database will be presented later in this tutorial.

Lesson 3.2. WINDEV and the databases

This lesson will teach you the following concepts

- Vocabulary used.
- The different modes for accessing the databases.

Estimated time: 5 mn

Overview

You may have to handle data when creating an application. To store the data, you must create a "database".
In WINDEV, when creating a project that is using data, an "analysis" must be created beforehand.
An analysis contains the description of the data files (or tables) containing the application data.
When running the application, these descriptions will be used to create the database and/or the data files. The information will be stored in this database or in these data files.

Remark

Several tools for performing maintenance operations on the HFSQL databases are supplied with WINDEV. They can be accessed from the HFSQL Control Center.

WINDEV can handle most of the database formats (nearly all of them). The most common formats are:

- HFSQL, the database system supplied with WINDEV. The HFSQL database is available in Classic mode or in Client/Server mode.
- Oracle, SQL Server, MySQL, xBase, SQL Azure, ...
- AS/400, Access, Sybase, Informix, ...
- Any database accessible in SQL language in Windows.
- Text (ASCII files).

Several methods can be used to access data:

- Native Connector (also called Native Access),
- OLE DB access,
- Direct ODBC access,
- ODBC access via OLE DB.

HFSQL

HFSQL is a database that is very powerful, very fast and very reliable.
HFSQL operates in Windows and Linux, on mobile devices (iOS, Android, Windows CE, Windows 10), on networks of any size and type, and it automatically manages hundreds of concurrent accesses.
HFSQL can be freely distributed with your WINDEV applications.

HFSQL proposes all database features, especially:

- the ***log*** process,
- the transactions,
- the replication,
- the triggers.

See the online help for more details about implementing these features.

In the different sections of this tutorial, we will be using a HFSQL Classic database then a HFSQL Client/Server database.

The different modes for accessing the databases

Native Connector (Native Access)

A Native Connector is using a database format directly and exclusively. This optimized type of access is developed specifically for each database format.
In WINDEV, a Native Connector is available for the following databases:

- HFSQL Classic or Client/Server (standard).
- xBase (standard).
- Access (standard).
- XML (standard).
- SQLite (standard).
- Oracle (optional).
- AS/400 (optional).
- SQL Server (optional).
- Sybase (optional).
- Informix (optional).
- DB2 (optional).
- Progress (optional).
- MySQL (optional and free).
- PostgreSQL (optional and free).
- MariaDB (optional and free).
- SQL Azure (optional and free).

Other Native Connectors will be available soon, contact our sales department for more details!
The WLanguage **SQL*** and **HRead*** functions can be used with this type of access. The code is portable and independent of the database.

Direct ODBC access

An access via ODBC is using a multi-database access standard. The ODBC layer must be installed on your computer. In most cases, this layer is already installed in the recent Windows versions. This can be checked in the control panel of Windows by selecting "ODBC data source" (or "ODBC administrator" according to the Windows version used).

Caution: some databases may not be accessible via this method. If you want to use this type of access, check whether an ODBC driver exists and install this driver if necessary.
Only the WLanguage **SQL*** functions can be used with this type of access.

OLE DB access

An access via OLE DB uses a multi-database access standard. This type of access is based on MDAC (Microsoft Data Access Component).

Important

If you are using an OLE DB access, MDAC must necessarily be installed on the user computers (version 2.6 or later).

Some databases may not be accessible via this method. If you want to use this type of access, check whether an OLE DB driver exists.
The WLanguage **SQL*** and **HRead*** functions can be used with this type of access.

ODBC access via OLE DB

In summary, it is a "mix" of OLE DB and ODBC. This is the "heaviest" method and the least efficient one in term of performance. It should not be used on small databases.
The WLanguage **SQL*** and **HRead*** functions can be used with this type of access.

Lesson 3.3. Project and analysis

This lesson will teach you the following concepts

- Creating a project.
- Creating an analysis.

Estimated time: 40 mn

Overview

To create an application with database, you must:

- Create the project linked to the application. This project will group all application elements (windows, codes, queries, reports, ...).
- Create the analysis linked to the project. The analysis is used to describe all data files handled by the application.

Remark

This lesson is used to create an application with HFSQL database. As already seen in the previous lesson, WINDEV allows you to create applications that handle other databases.

The lesson "Using SQL data", page 442 presents the operations that must be performed to create a project and an analysis based on an SQL database (via OLE DB or via Native Connector).

Creating the project

▸ To create the project:

1. Start WINDEV (if not already done).

2. Display the WINDEV home page if necessary (Ctrl + <).

3. In the home page, click "Create a project" and select "Windows or Linux application". The project creation wizard starts. The different steps of the wizard help you create your project. The information specified in this wizard can be modified later.

Tip

To create a project, you can also:

1. Click among the quick access buttons.
2. The window for creating a new element is displayed: click "Project".

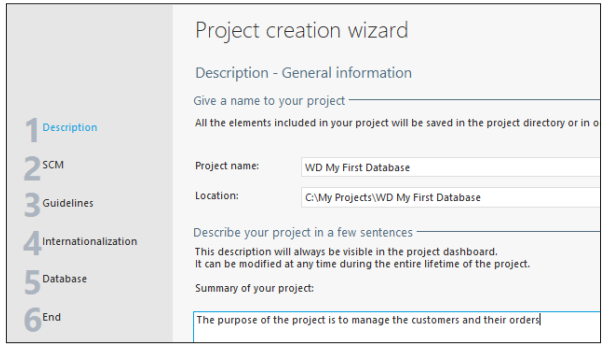
4. The first step of the wizard allows you to select the operating system for which the project is intended. Select "Windows platform" and go to the next step of the wizard ("Next").

5. For this lesson, we are going to create a "blank" project. You can also create a project based on an existing example, provided with WINDEV. Select "Create a blank project" and go to the next step of the wizard.

6. WINDEV then proposes to specify the platform description. Keep the default options ("Windows 32-bit executable" and "Executable with windows"). Go to the next step of the wizard.

7. The wizard proposes to type the name of project, its location and its description. In our case, this project will be named "WD My First Database".

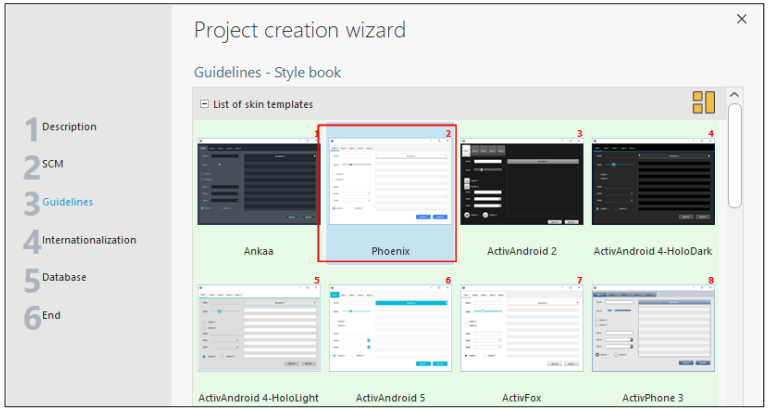
8. WINDEV proposes to create this project in the "\My projects\WD My First Database" directory. You can keep this location or modify it via the [...] button. For the project summary, type "The purpose of project is to manage the customers and their orders".



9. The different steps of the wizard are specified on the left. These steps can be clicked directly. Since the other steps in the "Description" are not strictly necessary, you can click "Guidelines" directly.

10. This step is used to define the code style. Don't modify the suggested options. Go to the next step.

11. This step is used to define the style book. Select "Phoenix".



Go to the next step.

12. Click the "Database" step. We are now going to specify the information regarding the database.

13. Select "Yes, create a new database" and validate. The analysis creation wizard starts.

Remark

To better understand the lessons found in this section and to optimize your training, we advise you to create the "WD My First Database" project.

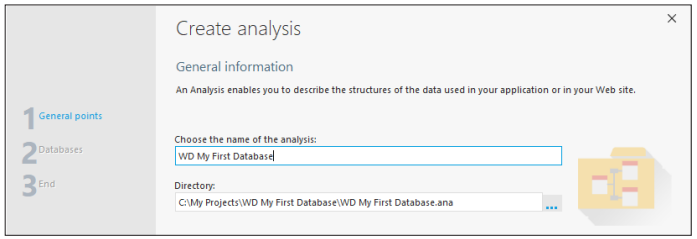
A corrected example can be accessed at any time to check the validity of operations performed.

To open this corrected project, in WINDEV's home page (Ctrl + <), click "Tutorial" and select "My first database (Answer)".

Creating the analysis

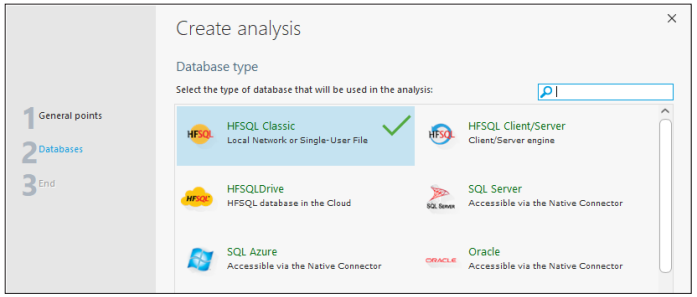
▸ The steps of the analysis creation wizard are as follows:

1. Specify the analysis name and directory. By default, the analysis name corresponds to the project name and the analysis directory is a ".ana" directory in the project directory. We will keep these default parameters.



Go to the next step of the wizard.

2. Next, you can choose the types of databases used by the project. Select "HFSQL Classic" (WINDEV's default database).



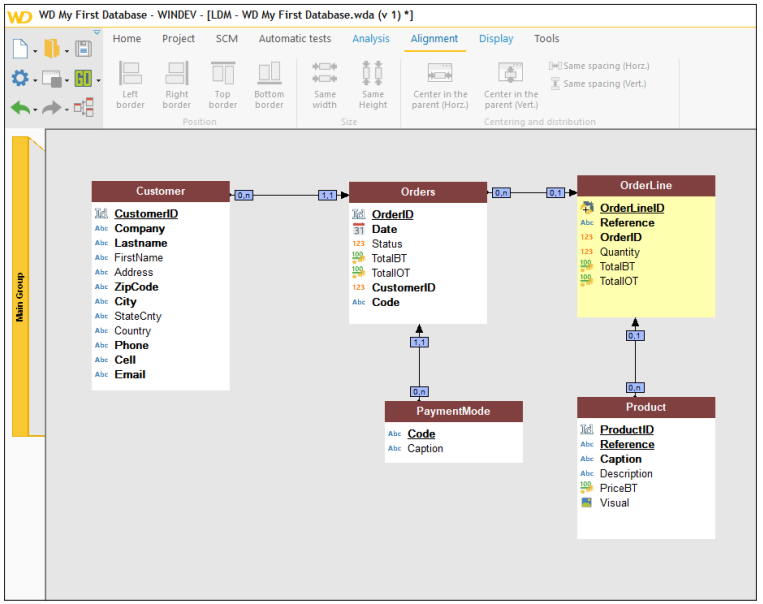
Go to the next wizard step.

3. Validate. The data file creation wizard starts automatically.

Creating the description of data files

Our application for managing customers and orders will be associated with the following analysis. This analysis includes five different data files (tables):

- Customer,
- Orders,
- PaymentMode,
- OrderLine,
- Product.

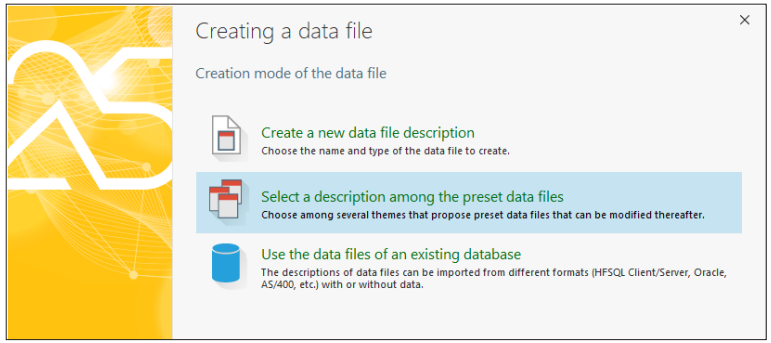


To create the data files of this analysis, we will be using the different methods proposed by WINDEV.

Creating a data file: using a preset file

▸ The steps of the data file creation wizard are:

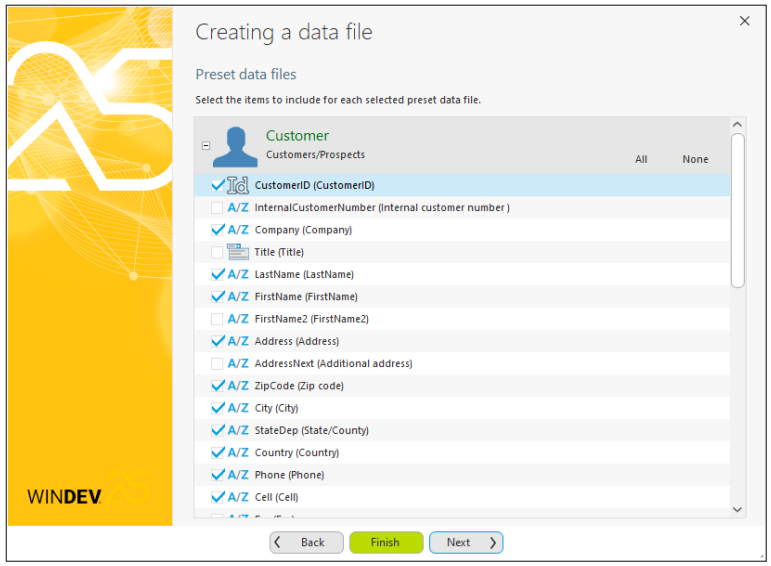
1. In the wizard, select "Select a description among the preset data files".



Go to the next step of the wizard.

2. The list of preset data files is displayed. We are going to create the "Customer" data file. In the list of data files, select "Customer". Go to the next step.

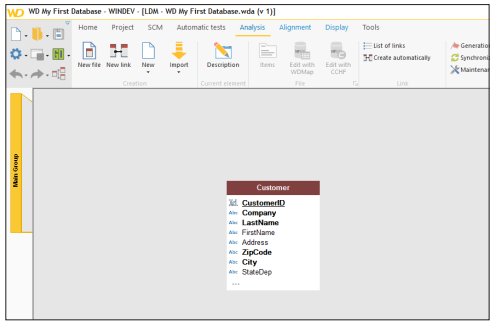
3. The wizard proposes the list of items that will be included in Customer data file. This list is very impressive because it allows you to manage several types of Customer data files.



4. Check the following items: CustomerID, Company, LastName, FirstName, Address, ZipCode, City, StateDep, Country, Phone, Cell, Email.

5. Validate the wizard.

6. The "Customer" data file is automatically created in the data model editor.



7. The window for creating a new element is displayed. We are now going to create the data file containing the orders.

Creating a data file: creating the file and the items

▸ To create a data file from the window for creating a new element:

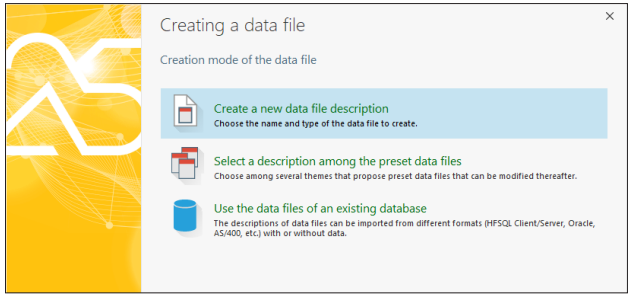
1. Click "Data" then "Data file".

Remark

You also have the ability to create a data file from the data model editor: in the ribbon, on the "Analysis" pane, in the "Creation" group, click "New file".

2. The data file creation wizard starts.

3. In the wizard, select "Create a new description of data file".



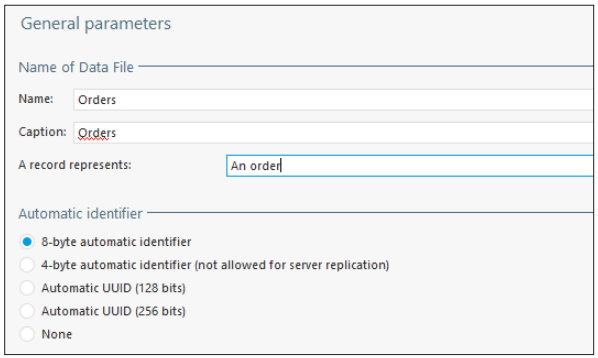
4. Go to the next step of the wizard.

5. We are going to create the "Orders" data file. Type its name ("Orders") in the wizard. This name will be used:

- to handle the data file by programming. The variable associated with the data file will be "Orders".
- to build the name of associated physical data file ("Orders.fic" file). The caption and description of elements represented by the file records are automatically displayed.

Remark

In the wizard, the control "A record represents" is used to get an understandable caption during the description of links between data files. A caption is automatically proposed from the data file name.
In our case, type "An order".



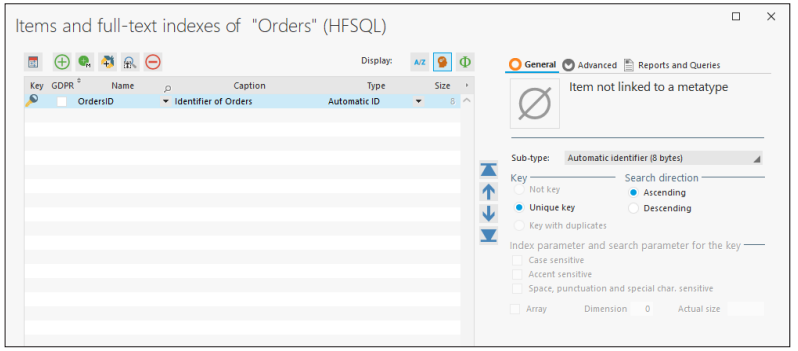
6. In "Automatic identifier", keep "8-byte automatic identifier". If an automatic identifier is defined on the data file, it means that the data file includes a unique key, automatically managed by WINDEV.

Remark

To create the identifier (an identifier is a unique key), you have the ability to create a numeric item whose type is "Automatic identifier".
This identifier is automatically managed by WINDEV. Whenever a record is added to the data file, WINDEV automatically assigns a value to the data file identifier. This value is unique.

7. Go to the next step and select the type of database associated with the data file. We are going to work on HFSQL Classic data files. Go to the next step.

8. Click "Finish" to validate. The data file is automatically created in the analysis. The description window of items and indexes opens up.

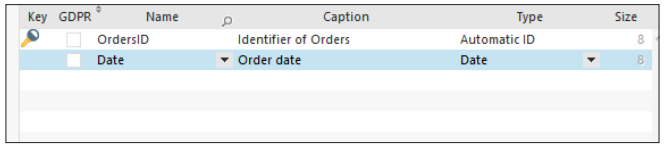


We are going to create the items of the "Orders" data file. In the description window of data file, you will notice that an item was automatically created: "OrdersID". This item corresponds to the automatic file identifier. This item consists of the letters "ID" and the name of the data file.
We are going to create the other items of this data file.

► First, we are going to create the "Date" item. This item will contain the order date.

1. In the description window of items, double-click in the "Name" column of first empty row. This column automatically becomes editable. Type "Date".

2. Click the "Caption" column. The item name is automatically displayed. We are going to modify the item caption by clicking on it: type "Order date". In the "Type" column, the "Text" type is automatically selected. Expand the list and select the "Date" type.



3. This item will be a key item (index) in our data file: the keys are used to accelerate the accesses to data and the sorts.

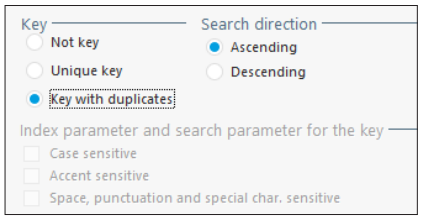
- For a database in SQL format, the engine is using the indexes at the best.
- For a sequential browse of a data file, all you have to do is specify the browse index (which means the key).

Remark

The concept of key is part of the item characteristics. When creating an item, you have the ability to specify whether it is:

- not key,
- unique key: the value of this key will be unique in the entire data file (i.e. in all the data file records).
- key with duplicates: the value of this key can be found several times in the data file.

4. The key definition is performed as follows: re-select the row of the "Date" item to activate the description controls on the right-hand side of the window. Then, simply specify the type of key used. In our case, the date is a key with duplicates.



5. You must also define the search direction of the key. The search direction is used to define the default sort for this item. In our case, when a browse is performed on this key, the default sort order will be "ascending".

▸ We are now going to create the "Status" item that is used to find out the order status.

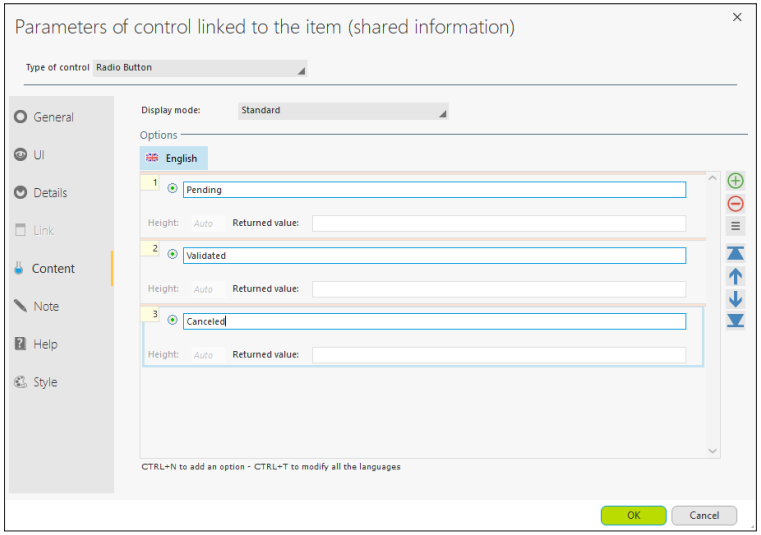
1. Position the cursor on a new table row. Enter:

- the name: Status
- the caption: Order status
- the type: Radio Button, List Box, Combo Box. In the window that appears, you can select the type of control created by default for this item. It will be a radio button in this case. Validate the window.

2. In the lower section of the screen, click the link to display the parameters of the control linked to the selected item.

The information typed in the new window that is displayed will be automatically used when creating windows linked to the data file. You will find here the control type and the caption. We are going to type the following options in the "Content" tab:

- Click the "Content" tab.
- Click the "+" button to add the first option.
- The option 1 corresponds to Pending. Type "Pending" in the edit control found on the right.
- Click the "+" button to add the second option.
- Type "Validated" instead of "Option 2".
- Click the "+" button again.
- Type "Canceled" instead of "Option 3".



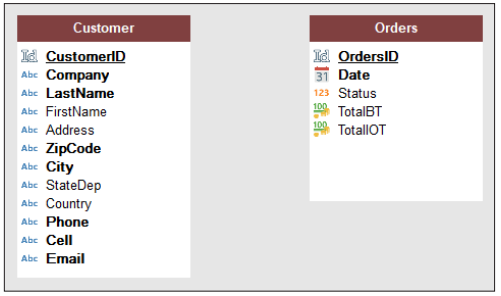
3. Validate the description window of the control linked to the item.

4. Similarly:

- Position the cursor on a new table row and create the item "TotalBT". This item is a "Currency" item.
- Position the cursor on a new table row and create the item "TotalIOT". This item is a "Currency" item.

5. That's it, the description of "Orders" data file is complete. Validate the description window of items.

6. The "Orders" data file appears in the data model editor. It is possible to enlarge the displayed data file. Simply click the data file, select the black handle at the bottom of the data file and move the mouse down.



Importing a CSV file

A different method will be used to create the "PaymentMode" data file, which contains the payment characteristics: importing a CSV file.

Remark

A CSV file is a text file that is using a specific format. This file contains data on each line. The data is separated by a separation character (a comma, a semi-colon or a tabulation).

From the CSV file containing the data, WINDEV will create:

- the description of data file in the analysis,
- the HFSQL data file with the data found in the CSV file.

▶ To import a CSV file into the analysis:

1. In the ribbon, in the "Analysis" pane, in the "Creation" group, expand "Import" and select "Import the descriptions of files/tables".

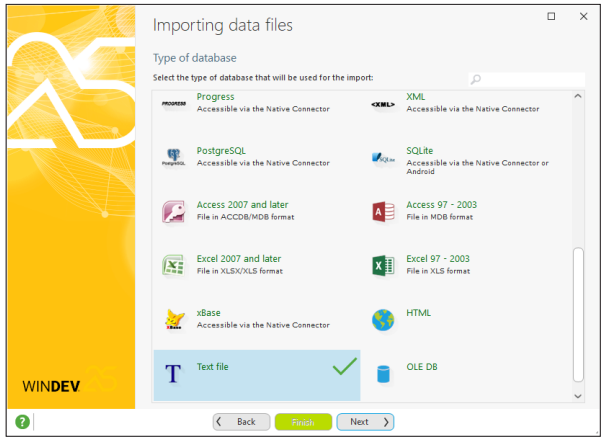
Tip

To import a CSV file (or any other file) into the analysis, you also have the ability to Drag and Drop the CSV file (from the Windows explorer) to the data model editor. This will be presented in the next paragraph.

2. The wizard for importing files starts.

3. Go to the next step.

4. Select the format of files to import. In this case, select "Text file".



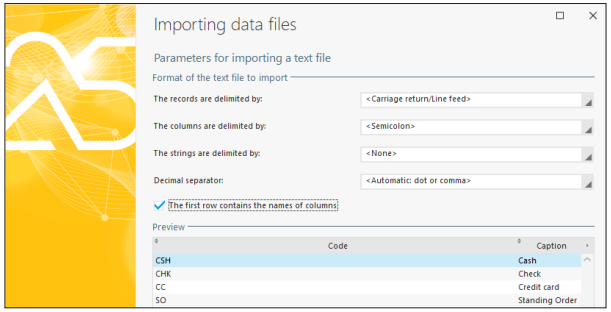
Go to the next step of the wizard.

5. Specify the path of file to import: "\Tutorial\WD\Exercises\WD My First Database\PaymentMode.csv" in the WINDEV setup directory.

6. Go to the next step of the wizard.

7. Specify the following import parameters:

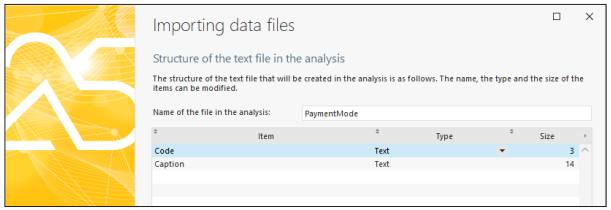
- The records are delimited by: "<Carriage Return/Line Feed>".
- The columns are delimited by: "<Semicolon>".
- The strings are delimited by: "<None>".
- Decimal separator: "<Automatic: dot or comma>"



8. Don't forget to check "The first row contains the names of columns".

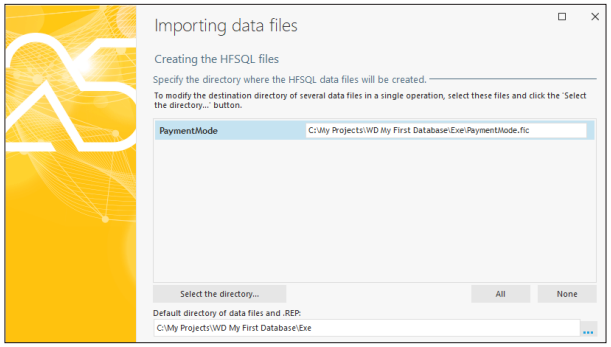
9. Go to the next step.

10. The structure of data file that will be created is displayed.



Keep the default options. Go to the next step of the wizard.

11. The content of CSV file will be automatically converted to HFSQL format. The wizard proposes to create the HFSQL file in the project directory.

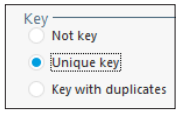


Keep the proposed options and go to the next step.

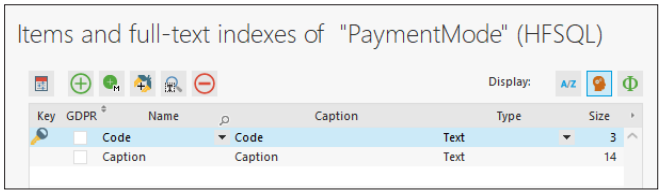
12. Validate the wizard. WINDEV creates the data file.

▸ Let's study the description of imported data file:

1. Select the "PaymentMode" data file.
2. In the popup menu (right mouse click), select "Description of data file".
3. Change the data file caption in the window that opens up: delete "(Imported)".
4. Click the icon to display the description of the data file items.
5. This data file contains no automatic identifier and no unique key. We are going to switch the "Code" item to unique key:
   - Position the selection bar on the code item if necessary.
   - In the right section of screen, click "Unique key".



6. We get the following data:



7. Validate the item description window and the data file description window.

Direct import of existing data files

The last method for creating data files consists in importing the existing HFSQL data files. This method will be used to create the "Product" data file.

▸ To import HFSQL data files:

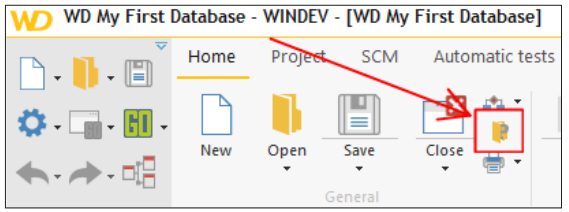
1. In the Windows file explorer, open the following WINDEV subdirectory: "\Tutorial\WD\Exercises\WD My First Database".
2. Select the "Product.fic" file.
3. Drag the file "Product.fic" and drop it in the WINDEV data model editor.
4. The import wizard starts. Validate the different steps. The data file appears in the data model editor.

All necessary data files are now found in the data model editor.

Caution! Only the description of the "Product" data file was imported into the analysis of our project. The data found in the "Product" data file was not imported into the project.

▸ To handle the data found in the file that was just imported:

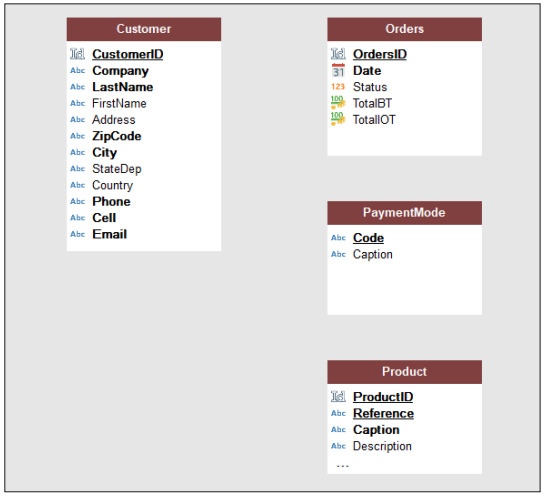
1. Open the file explorer on the directory of your project: on the "Home" pane, in the "General" group, click .



2. Open a new file explorer on the "\Tutorial\WD\Exercises\WD My First Database" directory.
3. Copy the the files "Product.fic", "Product.mmo" and "Product.ndx" from "\Tutorial\WD\Exercises\WD My First Database" to your project's EXE subdirectory via the file explorer.
Remark: Three files must be copied:
   - "Product.fic": file containing the data,
   - "Product.mmo": memo file containing the product images,
   - "Product.ndx": file containing the indexes to optimize the searches performed in the data file.

Creating the links

All data file descriptions required by the account management application have been created.



We are now going to create the links between data files. A link is used to define the integrity constraints (cardinalities) between two data files.

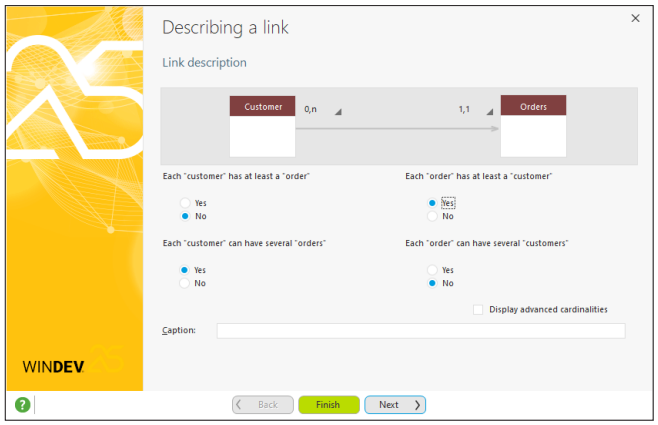
► First, let's create the link between the "Customer" and "Orders" data files: a customer can have one or more orders and each order is linked to a customer.

1. On the "Analysis" pane (found in the ribbon), in the "Creation" group, click "New link". The mouse cursor turns into a pen.

2. Click the "Customer" data file, then the "Orders" data file.

3. The wizard for link creation starts.

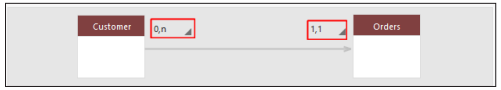
4. Answer the questions asked by the wizard:



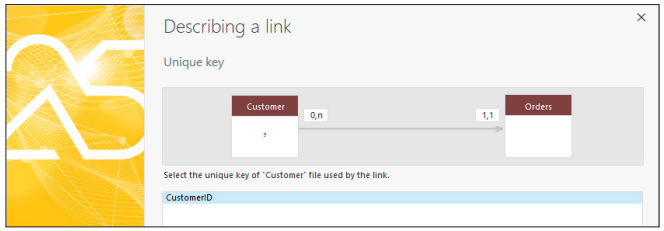
- Each Customer has at least one Order: No
- Each Customer can have several Orders: Yes
- Each Order has at least one Customer: Yes
- Each Order can have several Customers: No

Remark

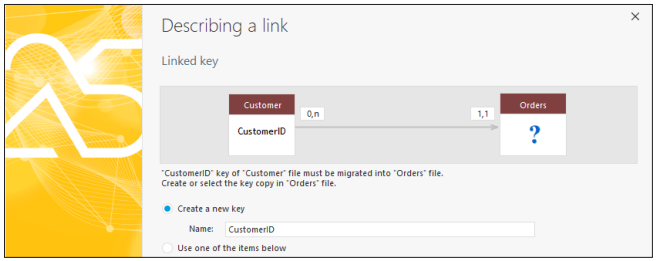
You can also specify the link cardinalities directly in the wizard.



5. Go to the next step. The wizard automatically proposes the key used by the link (CustomerID).



6. Display the next step of the wizard. The wizard offers to create the "CustomerID" key in the Orders data file to store the customer corresponding to the order.



7. Accept this option by going to the next step.

8. This step defines the integrity rules that will be automatically applied. In our case, you have the ability to choose the requested behavior when deleting a customer as well as the requested behavior when modifying the customer identifier.

9. Validate the integrity rules by going to the next step of the wizard.

10. Click "Finish". The link is automatically created in the data model editor.

► Likewise, create a link between the PaymentMode and Orders data files. These two data files are linked as follows:

- An order must have a payment mode.
- A payment mode can be used in several orders.

In the wizard:

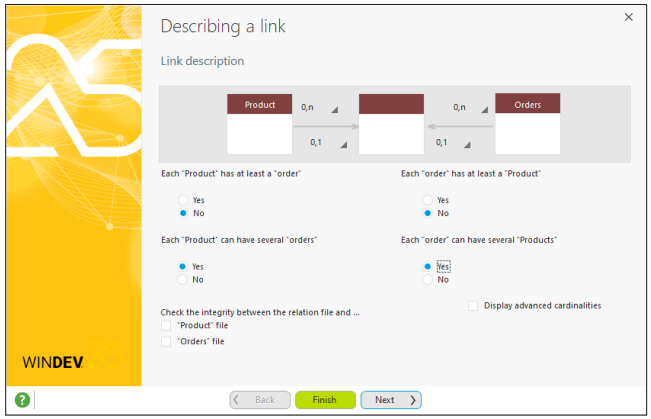
- The cardinalities are as follows: PaymentMode (0,n), Orders (1,1).
- The link key corresponds to the "Code" item.

▸ We are now going to create a link between the "Product" and "Orders" data files. This link will be used to create a link file, the file of order lines.

1. Likewise, create the link between the data files.

2. Answer the questions asked by the wizard:

- Each Product has at least one Order: No
- Each Product can have several Orders: Yes
- Each Order has at least one Product: No
- Each Order can have several Products: Yes



3. Go to the next step. The wizard proposes to create a relation file. Keep "Create the relation file automatically" and go to the next step.

4. The wizard proposes the unique key of the Product data file. Select "Reference". Go to the next step.

5. Validate the key creation by going to the next step.

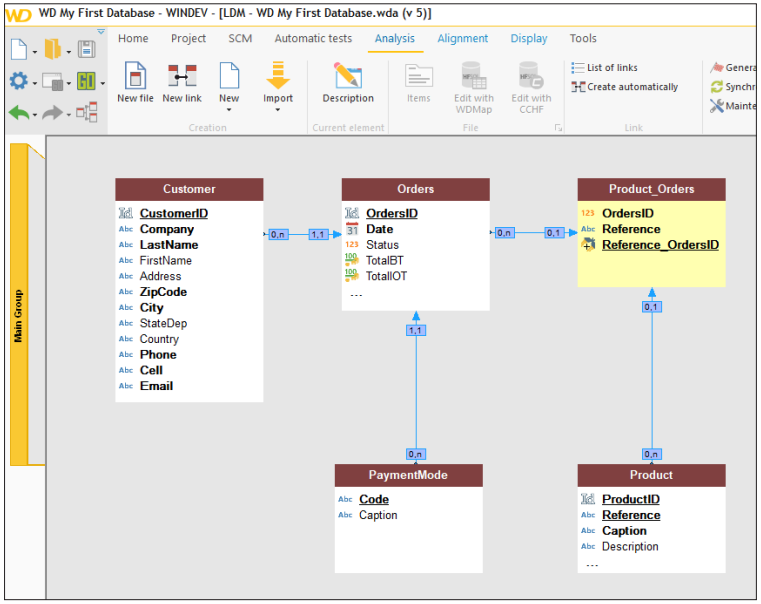
6. Keep the default options regarding the integrity rules and go to the next step.

7. The wizard proposes the unique key of the Orders data file to use: "OrdersID". Go to the next step.

8. Validate the key creation by going to the next step.

9. Keep the default options regarding the integrity rules and go to the next step.

10. Click "Finish". The relation file is automatically created in the data model editor.



We are now going to modify the relation file that was created by WINDEV. Indeed, this relation file will contain the order lines. We are going to:

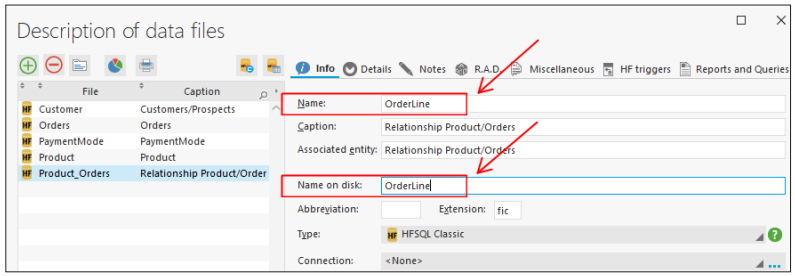
- Modify the name of the data file.
- Modify the name of its items.
- Add items to find out the quantity of products ordered and the total of order line.

▸ First, we are going to rename the data file. We already did something similar when we changed the caption of the imported data file. However, in this case, not only the caption will be modified: we will also rename the physical file linked to the data file description.

1. Select the "Product_Orders" data file.

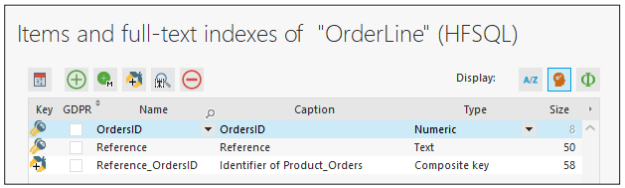
2. In the popup menu (right mouse click), select "Description of data file".

3. In the window that is displayed, modify:
   - the name of data file: "OrderLine".
   - the name on disk: "OrderLine".



► We are now going to modify the items of this relation file.

1. Click the icon to display the description of file items.



2. This relation file has 3 items. Position the selection bar on the item "Reference_OrdersID". This item is a composite key.

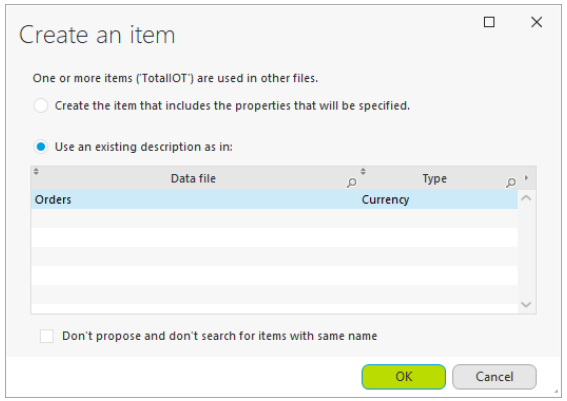
Remark

A composite key is a set of items that constitute an index.
This type of key is used to browse the data file according to complex criteria or to perform specific searches on several items at the same time.

3. To rename this item:
   - Click the "Name" column.
   - Replace "Reference_OrdersID" by "OrderLineID".
   - Click the "Caption" column.
   - Replace the current caption by "Identifier of OrderLine".

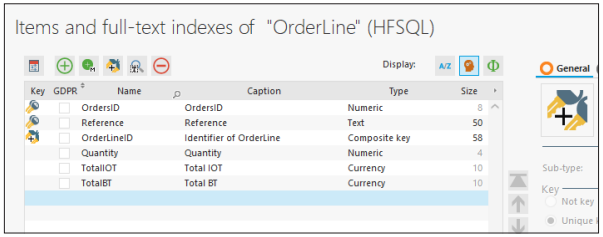
► We are now going to add 3 new items into this relation file: Quantity, TotalIOT and TotalBT.

1. Position the cursor on a new row and create the item "Quantity". This item is a "Numeric" item.
2. Position the cursor on a new row and create the item "Total IOT". This item is a "Currency" item.
3. A window is displayed, indicating that the item already exists in the analysis and proposing to re-use its characteristics:



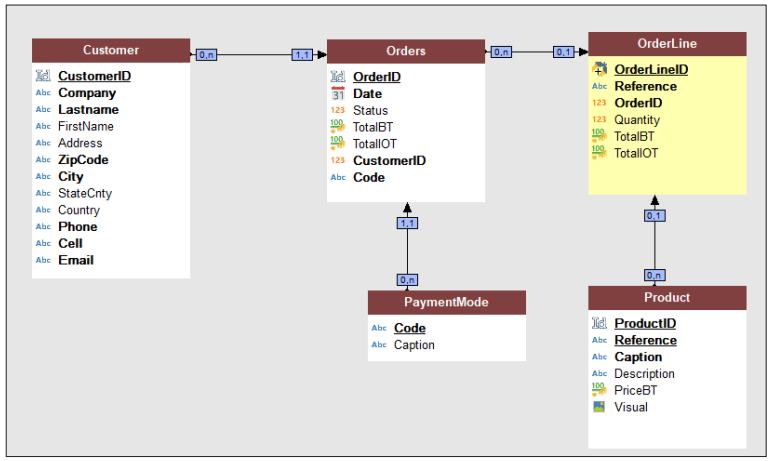
Keep the selected options and validate ("OK").

4. Position the cursor on a new row and create the item "Total BT". This item is a "Currency" item. Once again, use the existing description.
5. The description of the "OrderLine" data file items is as follows.



6. Validate the description of the items ("OK") as well as the data file description.

► The analysis is as follows:



Generating the analysis

Generating the analysis consists in validating the modifications performed in the analysis (creation of data files, addition or deletion of items, etc.) and to apply them to the entire project (pages, linked controls, reports, etc).

The generation is automatically proposed when closing the data model editor while modifications have been performed.

You also have the ability to generate the analysis manually. That's what we are going to do.

► To generate the analysis:

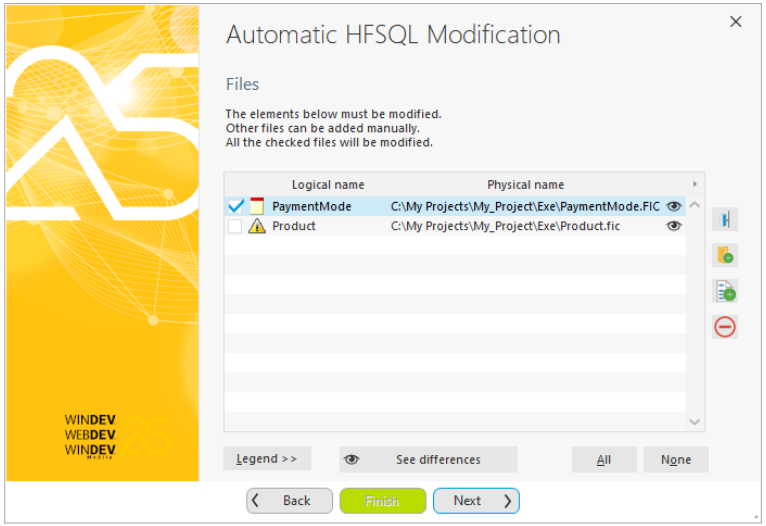
1. In the ribbon of the data model editor, on the "Analysis" pane, in the "Analysis" group, click "Generation".
2. The analysis generation is automatically started.

The descriptions of data files found in the analysis have been modified.

To update the data files of the application, WINDEV automatically starts the procedure for modifying the data files. This operation is used to update the data files (".fic" files) according to their description in the analysis.

► The wizard for automatic modification starts. Validate the different steps until you reach the screen that lists the data files to take into account.

- WINDEV detects that the PaymentMode data file must be updated. Keep this data file selected.
- WINDEV detects that the Product data file is out of sync. Select this data file.



- Go to the next step.
- The wizard proposes to save the existing data files, don't change anything and go to the next step.
- The wizard proposes to type the passwords for protecting the data files modified by the automatic modification. Keep the default options and go to the next step.
- The list of data files to modify is displayed. Validate the wizard.
- The data files are updated.

► Close the data model editor.

The main steps for creating an analysis have been presented.

When your analysis is described and generated, you also have the ability to:

- Create a full application via RAD (Rapid Application Development). We will create the application corresponding to the analysis in this way in the next lesson.

Create a full custom application. This method will be used to develop a full application based on a database in the part 4 of this tutorial. This application will be based on an analysis corresponding to the one that was created in this part.

Lesson 3.4. The full RAD

This lesson will teach you the following concepts

- What is RAD?
- Generating RAD.
- Project test.

Estimated time: 20 mn

What is RAD?

R.A.D. stands for "Rapid Application Development". The RAD is used to automatically build an application, which means all necessary windows, reports and queries.

As already seen, to develop an application in WINDEV, you must create a project and an analysis (if necessary). The analysis contains the definition of the structures of data files used in the processes.

The RAD module of WINDEV is based on this analysis. The RAD module includes a wizard allowing you to choose the application template to generate (the RAD pattern) and the main options regarding the operating mode of your application.

Remark

WINDEV is supplied with several "RAD patterns" allowing you to generate several application templates. You also have the ability to create your own RAD patterns. See the online help for more details (keyword: "RAD pattern").

The windows, the reports, the queries and the code generated by RAD can be customized. You also have the ability to modify the types of controls, the default values, ...

The RAD can also be used to generate several types of windows, it is the window RAD. The Window RAD is available when a new window is created in your application.

Let's see how to use the Project RAD module.

Remark

RID (Rapid graphical Interface Design)

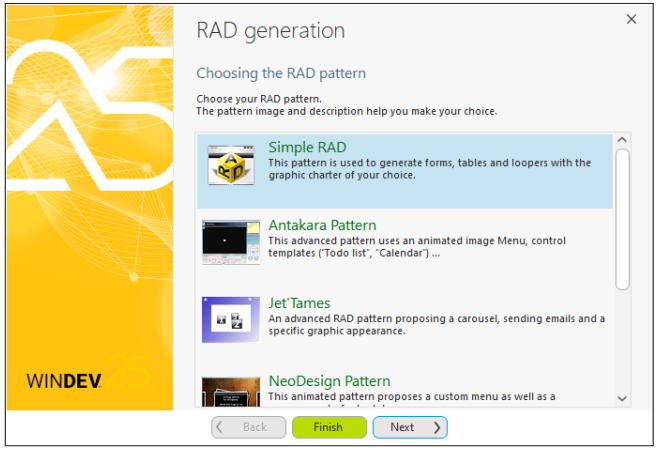
WINDEV also allows you to generate windows containing the controls linked to the analysis items only. The code required for these windows to operate must be written by the developer.

See the online help for more details (keyword: "RID").

Generating RAD

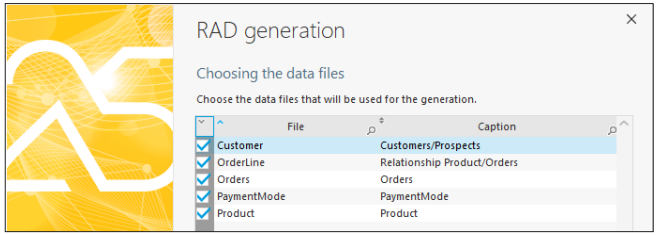
▸ To start generating RAD:

1. On the "Project" pane, in the "Generation" group, click "Full Application RAD". The wizard for generating the RAD application starts.



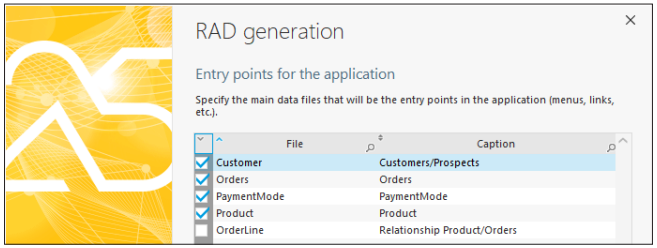
Select the pattern that will be used for RAD generation: "Simple RAD" for example. Go to the next step of the wizard.

2. All data files found in the analysis will be taken into account:



3. Go to the next step.

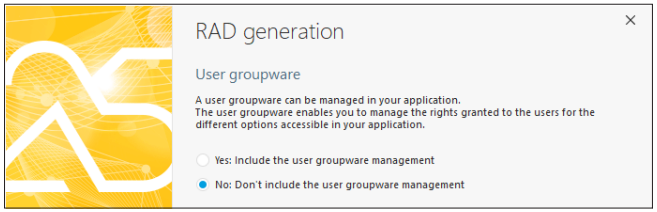
4. The entry points of application correspond to the entries available in the menu. Keep the Customer, Orders, PaymentMode and Product data files only:



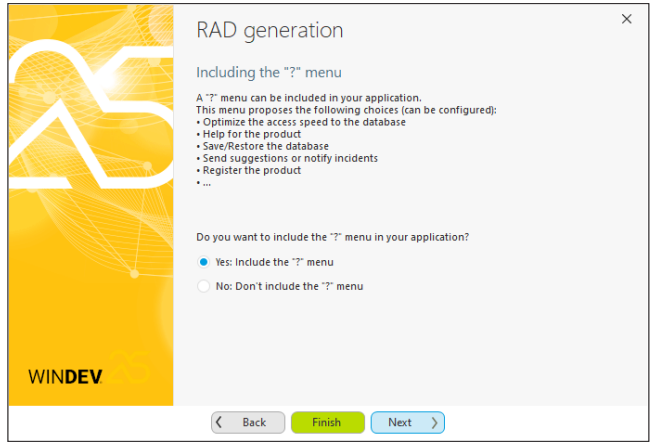
Go to the next step.

5. Specify whether the Table controls generated in the application windows must be allow the input or not. In our example, the Table controls will allow the user to type new information. Select "Yes: Allow the input in the Table controls (editable Table controls)". Go to the next step.

6. Specify whether the user groupware will be used in the application. It will be presented later in this tutorial. Select "No: Don't include the user groupware management".



7. Go to the next step. Specify whether the automatic menu will be included in the application. Answer "Yes: Include the '?' menu".

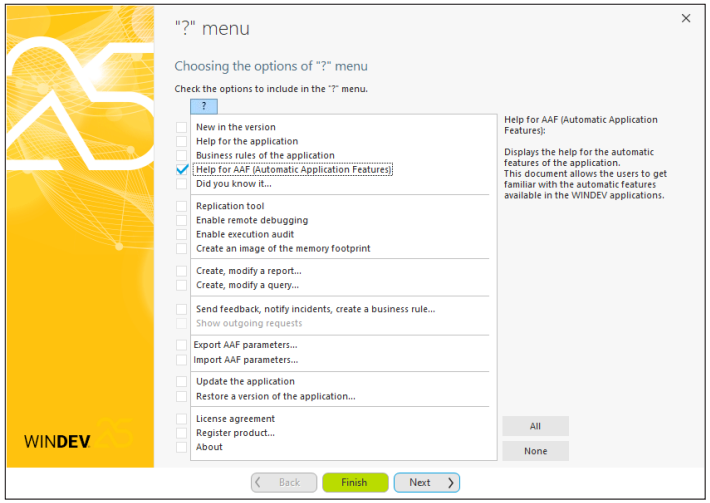


8. Go to the next step. The RAD generation wizard is ended. Validate. The wizard for generating the automatic menu starts.

Remark

The automatic menu is a help menu suited for your applications. This menu allows the users of your applications to easily access several features.

► The different options proposed by the wizard for generating the automatic menu are as follows:



1. Keep "Help for AAF (Automatic Application Features)".
2. End the wizard.

Remark

When creating the '?' menu, the option "? .. Help for AAFs (Automatic Application Features)" automatically added into your application:

- the "CCMenu" component. Indeed, this option is using a component procedure to operate.
- the "WINDEV AAF 25.PDF" file. This file will be automatically opened when using the menu option.

The application is generated and its test can be run.

Remark

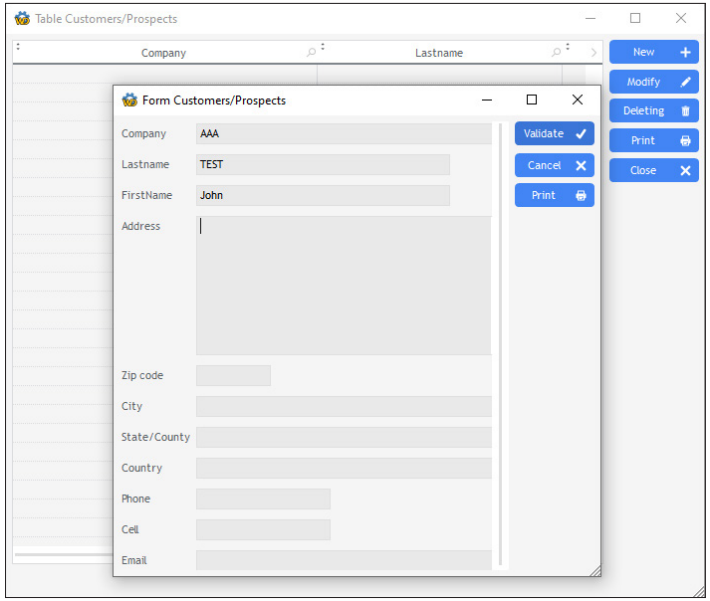
UI errors may appear in the "Compilation errors" pane. These errors signal interface problems, especially in the reports (captions too long for example). To correct these errors, the reports must be modified.

Application test

Let's now run the test of generated application.

► To run the application test:

1. Click GO among the quick access buttons. The application starts.
2. In the menu, select "Customers/Prospects .. List of Customers/Prospects".
3. The list of customers is displayed.
4. To add a new person, click the "New" button. An input form is displayed.



5. Type information about a customer and validate.
6. Close the list of persons.
7. In the menu, select "Product .. List of products". The list of products is displayed.
8. Close the test window. The WINDEV editor is redisplayed.


DEVELOP 10 TIMES FASTER
PART 4
Full application
with data

Lesson 4.1. Overview

This lesson will teach you the following concepts

- Overview of application created in this section.

Estimated time: 5 mn

Overview of application created in this section

In this section, we are going to create an application that handles data files in HFSQL Classic format.

This application for order management will be used to:

- View, enter and modify products,
- Print reports,
- Display statistics,
- Manage use levels,
- Send emails,
- ...

The development of this application will allow us to present the deployment and the distribution of application to the end users.

You will see the main points for developing an application.

Projects supplied

Example project

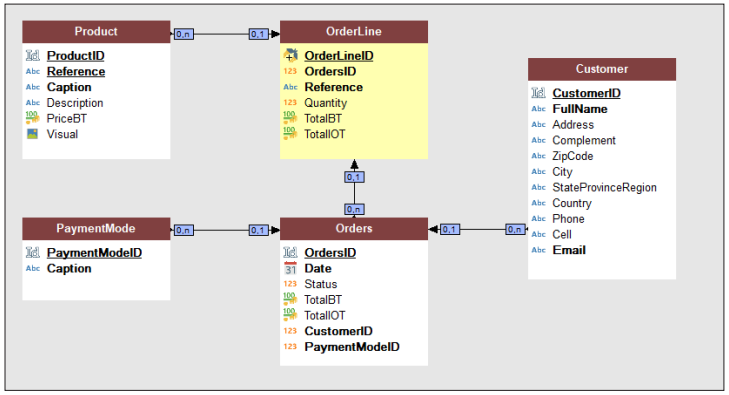
The creation of project and analysis was presented in the previous section. We won't go back to it. We are going to work with a project that already contains an analysis and data files filled beforehand. The application analysis is an improved version of the analysis that was created in the previous section.

► To open this project in WINDEV:

1. Display the WINDEV home page (Ctrl + <).
2. In the home page, click "Tutorial" and select "Full application (Exercise)".

Before we start developing the application, let's take a look at the analysis associated with the "WD Full application" project.

► To display the analysis linked to the project, click  among the quick access buttons. The data model editor is displayed:



This analysis contains the description of 5 data files:

- Customer,
- Orders,
- OrderLine,
- Product,
- PaymentMode.

This analysis is straightforward and it is used to manage orders.

► Close the data model editor (click the cross in the top right corner of menu).

Corrected projects

You don't know which result to get? Or you don't have time to perform all operations? A corrected project is available for the application

The application that we are going to create being quite long (it is a full application that will allow you to discover the main features of WINDEV), two corrected projects are available:

- A corrected project corresponding to lessons 4.2 to 4.4. This project contains all windows developed in these lessons.
  To use this corrected project:
  1. Display the WINDEV home page if necessary (Ctrl + <).
  2. In the home page, click "Tutorial" and select "Full application (With windows)".
- A corrected project corresponding to lessons 4.5 to 4.9. This project contains the entire application.
  To use this corrected project:
  1. Display the WINDEV home page if necessary (Ctrl + <).
  2. In the home page, click "Tutorial" and select "Full application (Answer)".

Lesson 4.2. Addition and modification windows

This lesson will teach you the following concepts

- Creating a menu window used to list the products.
- Creating a product form.
- Managing the product addition and modification.

Estimated time: 50 mn

Overview

We are going to create the different windows used to list, add and modify products. These operations will allow you to discover several topics regarding the management of data files and they will also allow you to use some WINDEV features.

▸ If the "WD Full Application" project was not opened in the previous lesson:

1. Display the WINDEV home page (Ctrl + <).
2. In the home page, click "Tutorial" and select "Full application (Exercise)".

Answer

A corrected project is available. This project contains the different windows created in this lesson. To open the corrected project, in the home page, click "Tutorial" and select "Full application (With windows)".

Creating a window used to list the products

To create the window used to list the products, we are going to create a blank window then add all controls: this will allow us to present the steps required to create this type of window.

Creating the window

▸ To create a window used to list the products:

1. Create a new blank window.
   - Click  among the quick access buttons (or press Ctrl + N).
   - The window for creating a new element is displayed: click "Window" then "Window".
   - In the wizard, click the "Based on a template" tab and choose "Use: WINTPL_Template". Validate.

Remark

The window templates

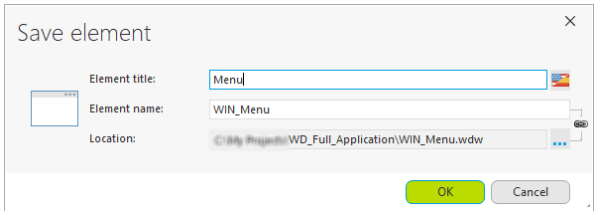
A window template is used to give a standard style to the different application windows.

A window template contains all graphic elements common to all application windows. A window template can also contain code. In this case, this code will also be common to all application windows.

The window template named WINTPL_Template was created for this example. This template contains a logo that will be displayed in all windows:

2. The new window is displayed in the editor. The window for saving an element is displayed.

3. In this window, specify the window title: "Menu".



4. The window name (that will be used in programming) is automatically deduced from the window title. If this name does not suit you, you have the ability to modify it and to specify a title that differs from the window name.

5. Validate.

Remark

This window was named "Menu" because it is the main window of our application. It will be used as menu but it will also display information.

Creating controls

A Table control will be used to create the list of products. This control will be linked to the "Product" data file.

Remark

What is a Table?

Do not confuse "Table" and "Table control".

- We refer to an SQL database as a **Table**.
- A **Table control** is used to view data in a table. In this case, the data can be typed or it can come from a data file (or from a table).

The Table control of WINDEV is used to view or modify data:

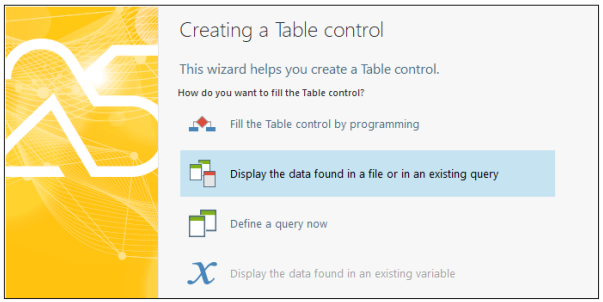
- in memory: it is referred to as a **memory Table** control or Table control filled by programming.
- coming from data files or queries: it is referred to as a **browsing Table** control.
- coming from WLanguage variables: it is referred to as a **Table control on source**.

▶ To create the Table control:

1. On the "Creation" pane, in the "Data" group, expand "Table and list box" and select "Table (Vertical)". The control currently created follows the mouse movement.

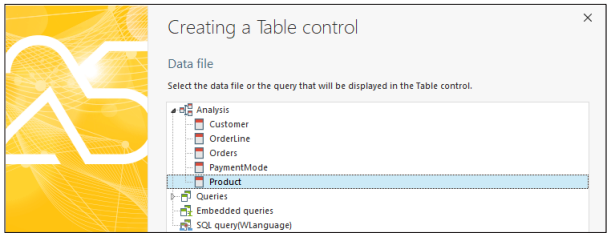
2. Click inside the window: the Table control creation wizard starts.

3. The Table control that is currently created must display the records found in the "Product" data file. In the wizard, select "Display the data found in a file or in an existing query".



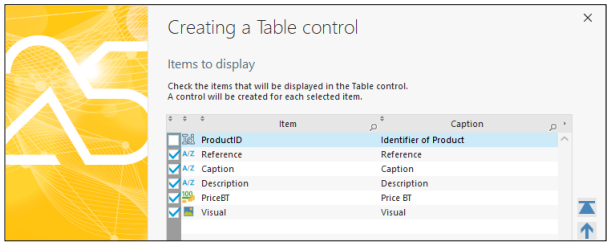
Go to the next step of the wizard ("Next").

4. The next wizard screen proposes the different data files and queries found in the current project. Expand "Analysis" if necessary and select the "Product" data file.



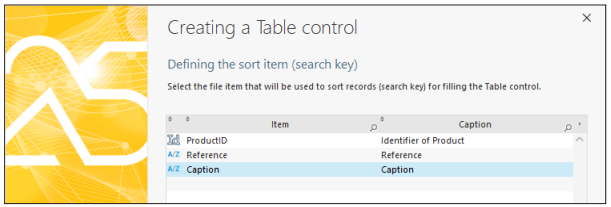
Go to the next step.

5. The wizard proposes the list of items found in the "Product" data file.
By default, all items are checked in order to be displayed in the Table control. In our case, we are going to display all items EXCEPT for the "ProductID" identifier that corresponds to the automatic identifier of data file.
Uncheck "ProductID".



Go to the next step.

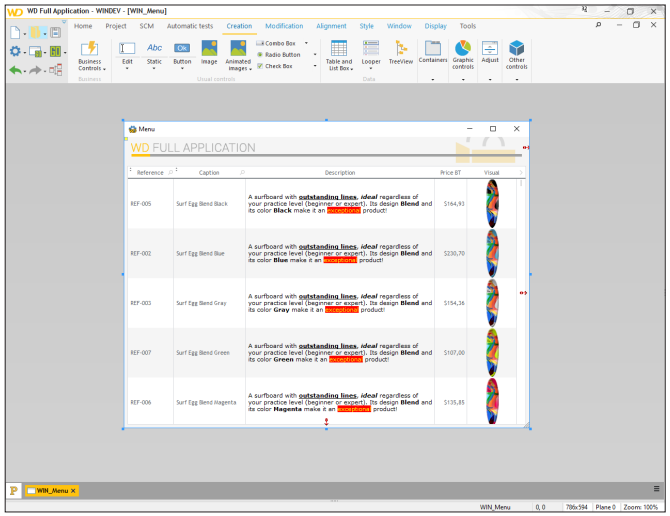
6. Then, the wizard proposes to select the sort item. This item will be used to sort the data displayed in the Table control. The items proposed in the wizard correspond to the items defined as key items in the analysis. The products will be sorted according to their caption. Click on the "Caption" row.



Go to the next step.

7. The following steps correspond to additional parameters. Keep the default options until the end of the wizard and then finish the wizard.

8. The Table control is automatically created in the window. The control is automatically positioned and enlarged in order to occupy the entire available space in the window.



Let's take a look at the control that was just created: the data is already displayed in the control, even in the editor. **This concept is called "Live data"**: the content of data files found on the development computer is used in the windows or reports handled in the editor. This feature is very useful to define the size of controls found in a window for example.

Window test

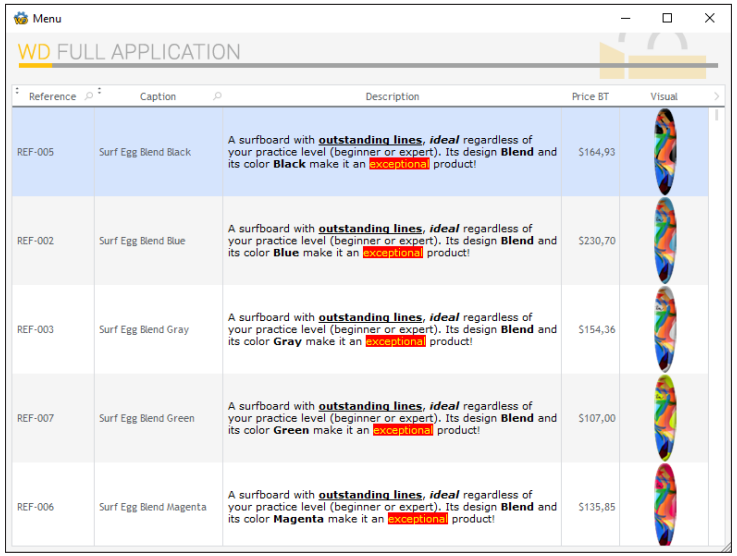
▸ We are now going to run the test of the window that was just created.

1. Click among the quick access buttons.

2. The window is automatically displayed with its data at run time.

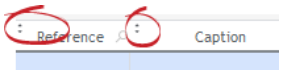
3. You have the ability to scroll the products with the scrollbar (or with the keyboard arrows).

▸ Let's imagine that we are the end user to see the capabilities of Table control at run time.



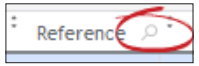
Some remarks:

- First of all, the Table control has the same aspect at run time and in the editor: this is the WYSIWYG concept ("What You See Is What You Get").
- Small logos appear at the top of columns:
  - the double arrow indicates that the column can be sorted ("Reference" or "Caption" column for example).

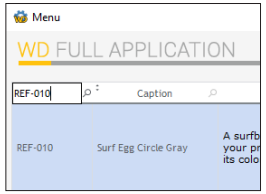


Click the "Caption" column twice for example: the data is displayed according to a different order. An arrow on the left of columns indicates that the column is sorted as well as the sort direction.

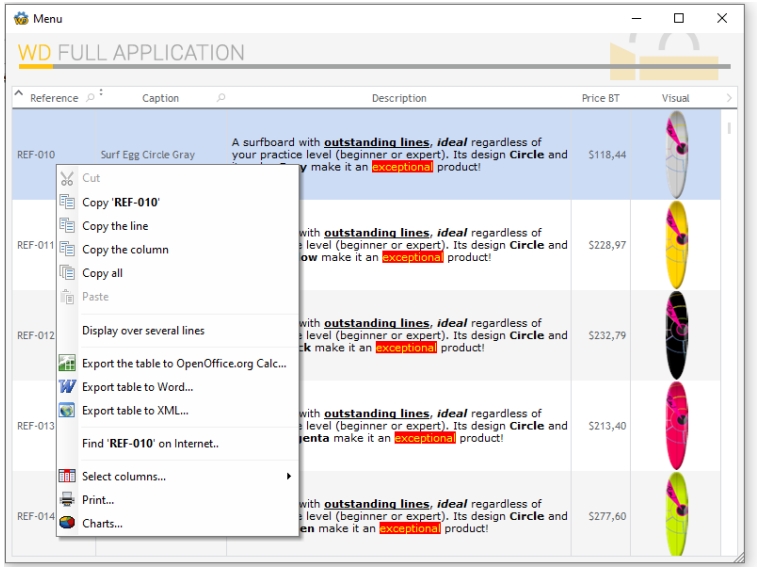
  - the magnifier indicates that a search can be performed in the column.



Click the magnifier of "Reference" column for example. An input area is displayed, allowing you to type the sought reference. Type "REF-010". The product corresponding to this reference is automatically displayed.



- A popup menu is available for the Table control and for each one of its columns. This popup menu can be displayed:
  - via a right mouse click on the column of Table control.
  - via a click on the arrow found at the extremity of column headers.
- This popup menu is used for example to export the content of Table control in Excel format (or OpenOffice.org Calc), in Word format (or OpenOffice.org Writer), to print, ...



In just a few minutes, we have presented some features that are automatically included in the Table control. No programming is required to propose these features to your users. These features are called "AAF" (Automatic Application Features).

Remark

Several controls propose default AAFs. If these features must not be proposed (for security or confidentiality reasons), they can be disabled in the editor or by programming.
See the online help for more details (keyword: "Automatic Application Features").

Now that the list of products is displayed, you may want to modify a product. To do so, we are going to create a window used to display the product details.

▶ Close the window via the cross found at the top of the screen. The WINDEV editor is redisplayed.

Creating a "Product form" window

Creating the window

▶ To create a window used to display the product details:

1. Open the "WD Full Application" project if necessary.
2. Create a new blank window.
   - Click among the quick access buttons.
   - The window for creating a new element is displayed: click "Window" then "Window".
   - The window creation wizard starts.
   - In the "Based on a template" tab, choose "Use: WINTPL_Template" and validate the wizard.
3. The save window is displayed. Type the window title: "Product form". The name ("WIN_Product_form") is automatically proposed.
4. Validate the save window.

Creating edit controls

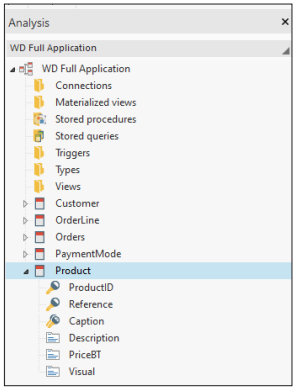
We are going to create a window used to display the characteristics of the product selected in the Table control. This type of window is called a "Form" because it corresponds to a descriptive form of requested element.

In our case, this window will be used to display the content of the different items found in the "Product" data file.

▶ To create the controls in the window:

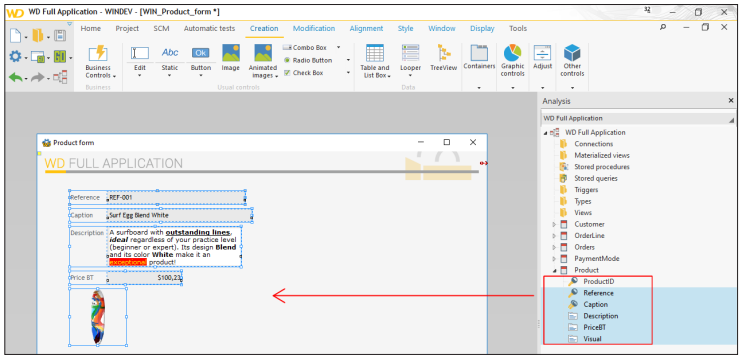
1. Display the "Analysis" pane if necessary: on the "Home" pane, in the "Environment" group, expand "Panes" and select "Analysis". The different data files described in the "WD Full Application" analysis appear in the pane.

2. Click the ▷ icon on the left of "Product" data file: the items found in the data file are listed.



3. With the mouse, select all items displayed in the pane (except for the "ProductID" item). You have the ability to use the multi-selection by holding the Ctrl key down.

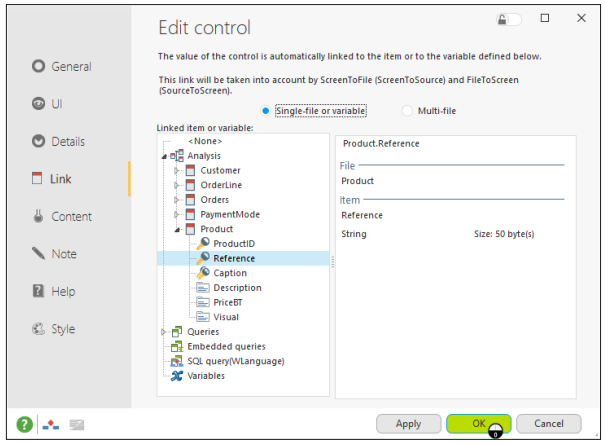
4. Drag and Drop these items to the window that was just created.



5. Different controls are automatically created in the window. These controls are automatically linked to the corresponding item in the data file. To check this:

- Select the "Reference" control for example.
- Display the popup menu (right mouse click) and select "Description".

- Display the "Link" tab of description window:

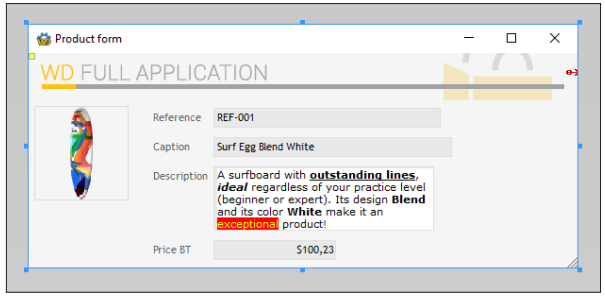


This tab shows that the current edit control is linked to the "Reference" item found in the "Product" data file.

6. Close the description window.

7. Save the window.

▶ Reposition the controls in the window and resize the window in order to get the following interface:



Creating buttons

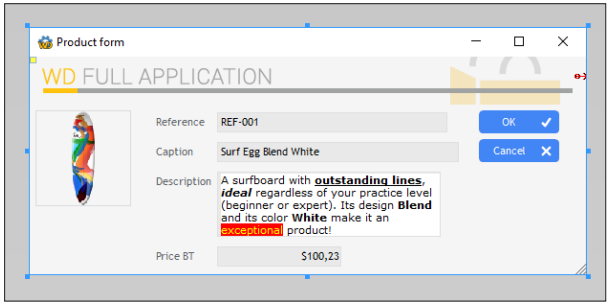
The "WIN_Product_form" window will be used to add a new product or to modify a product found in the list.
We are going to add a validation button and a cancelation button.

▶ To create the Button control to validate the data entered:

1. On the "Creation" pane, in the "Usual controls" group, expand "Button" (click the arrow found below Ok).
2. The list of preset buttons is displayed.
3. Click "Validate": the Button control shape appears under the mouse pointer.
4. Then, click on the right of edit controls to create the Button control.

▶ Add the Button control to cancel the input using the same principle.

1. On the "Creation" pane, in the "Usual controls" group, expand "Button" (click the arrow found below Ok).
2. In the list of preset buttons, click "Cancel": the Button control shape appears under the mouse pointer.
3. Then click below the "Validate" Button control to create the control.



Improving the window interface

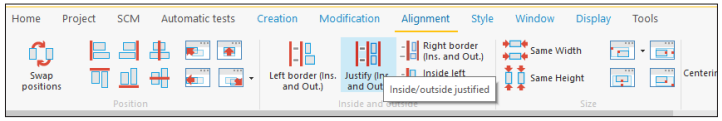
This window being quite simple, let's try to improve its interface. Several details can become very annoying for the user so let's try to sort them out right now.

Aligning controls

▶ One of the first optimizations consists in aligning the controls and in giving them the same size:

1. Select the "Caption" control then all edit controls found in the window (hold the Ctrl key down while clicking the different controls for example). The first selected control will be used as reference for the size of other controls.

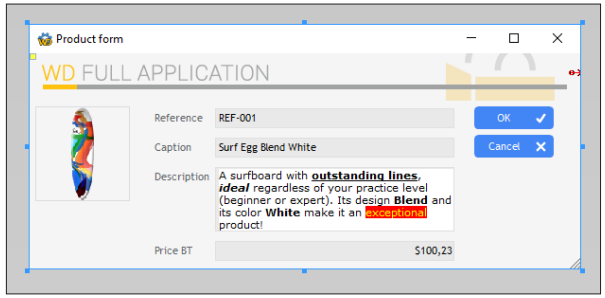
2. In the ribbon, display the "Alignment" pane of WINDEV. This pane contains all alignment options available for the controls.



3. We want the outside and inside borders of controls to be aligned. Click "Justify (Ins. and Out.)".

Remark

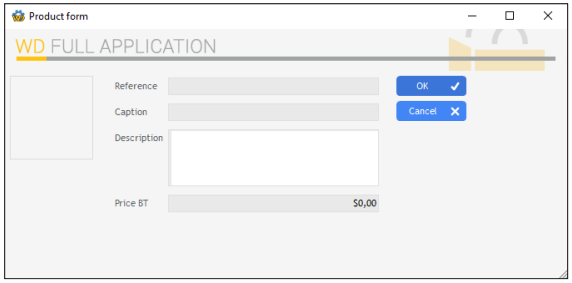
If you don't know which alignment to choose, hover the different options proposed by the "Alignment" pane of WINDEV. When hovering an option with the mouse cursor, the controls selected in the window are automatically positioned and/or resized according to the hovered option.
If the positioning suits you, all you have to do is click the option.
If the positioning does not suit you, all you have to do is hover the alignment options: the controls move back to their initial position.



Managing anchors

▶ Let's see the visual aspect of our window:

1. On the "Modification" pane, in the "Preview" group, click "Preview". This runtime mode is used to display the window interface. The different codes are not run.
2. Enlarge the window via the sizing handle found at the bottom of window. The position of controls is not modified and the window displays a large empty area. Two methods can be used to solve this problem:
   - Forbid the window from being resized.
   - Manage the anchors, which means manage the behavior of controls when resizing the window. That's what we are going to do.



3. Close the preview: click the "Close" button or the cross found in the top right corner of window.

Remark

How to manage anchors in a window?

To manage the anchors:

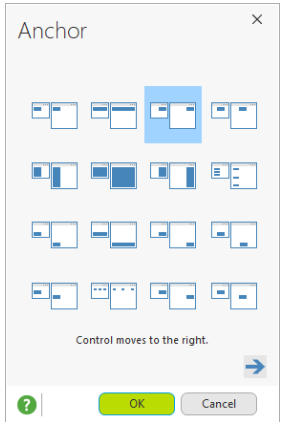
1. In the editor, reduce the window to its minimum size. All outside window borders must be "stuck" to a control.

2. Study the controls one by one and ask yourself the following question : "How will the control react when the window is enlarged?". Then, apply the corresponding anchor.

► For this window, we are going to define the anchors together:

1. The "Validate" and "Cancel" Button control must be anchored to the right.

- Select these two controls.
- Display the popup menu (right mouse click) and select "Anchor".
- In the window for defining anchors, select "Right".



- Validate.
- Small red arrows appear in the editor, indicating that the controls are anchored to right.

2. The edit controls must increase in width:

- Select the 4 edit controls.
- Display the popup menu (right mouse click) and select "Anchor".
- In the window for defining anchors, select "Width" ( ).
- Validate.

3. We are going to test the operating mode of this window in preview:

- On the "Modification" pane, in the "Preview" group, click "Preview".
- Enlarge the window: the controls follow the enlargement in width.
- We must now define the operating mode during the enlargement in height.

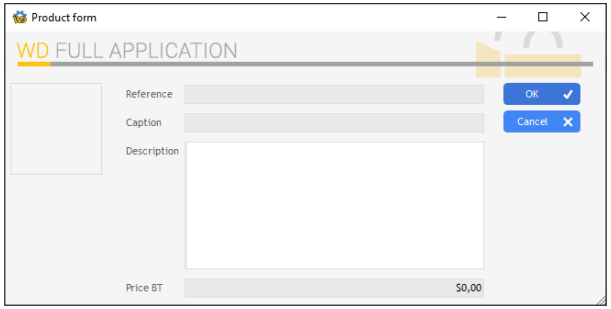
4. The "Description" control can contain a product description that is more or less important. This is the most important window control: it must occupy the maximum available space. Apply an anchor in width and height to it ( ).

5. The "Price BT" control is automatically bordered by a red line: the window editor signals an anchoring problem. Indeed, if the "Description" control stretches to the bottom, the "Price BT" control must move to the bottom accordingly.

6. Open the anchoring window for the "Price BT" control. The best anchor is automatically proposed: "width and bottom" (a green checkmark is displayed). Select the proposed anchor and validate.

7. We are going to test the operating mode of this window in preview:

- On the "Modification" pane, in the "Preview" group, click "Preview".
- Enlarge the window: the controls follow the enlargement in width and in height.



8. Close the window preview to go back to the editor.

Displaying the form from the list of products

Now, let's see how to display the form of the product selected in the list of products. The principle is straightforward: the user will select the product in the Table control and display the product details via a Button control.

▸ We are going to modify the "WIN_Menu" window to create a modification Button control:

1. Position the cursor on the "Menu" window: click "WIN_Menu" in the open documents bar:

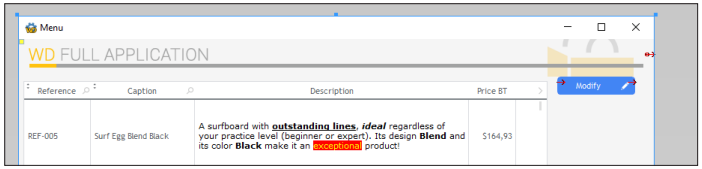
2. At this time, the Table control occupies the entire window. Enlarge the window to the right via the sizing handles.
3. Reduce the size of Table control:
   - Hold the Shift key down.
   - Reduce the Table control via its sizing handles.

> **Remark**
> Pressing the Shift key allows you to ignore the anchoring options of controls during the resize operation.

4. We now have space on the right to create the Button control that will allow us to view the selected product in the Table control.

▸ To create the Button control to view the product:

1. On the "Creation" pane, in the "Usual controls" group, expand "Button" (click the arrow found below Ok).
2. Enter "Modify" in the search box at the top of the list of preset Button controls. Click "Modify button". The shape of the control appears under the mouse pointer. Then, click on the right of the Table control to create the Button control.



3. Right-click the control that was just created. Select "Code" from the menu that is displayed.
4. In the code window that appears, write the following WLanguage code in the "Click" event:

```
Open(WIN_Product_form)
```


> **Remark**
> The assisted code input (also called "completion") is going to help you: as soon as you type the opening bracket "(", a drop-down list proposes the name of all existing windows found in the project. Simply select the name of the window with the keyboard or mouse.
> If the window name is not displayed in the list, it means that this window was not saved beforehand.

5. Let's take a look at this WLanguage code: **Open** is used to open the "WIN_Product_form" window. This function was already presented in the first part of this tutorial.
6. Save the modifications by clicking among the quick access buttons.
7. Close the code window (click X at the top right corner of code editor).

We are now going to modify the "WIN_Product_form" window to display the selected product in the Table control.

▸ Open the "WIN_Product_form" window in the editor: click the corresponding button in the open documents bar.

▸ To display the product data:

1. Display the events associated with the window:
   - Perform a right mouse click in the area found beside the window (called "Home area").
   - Select "Code" from the popup menu that is displayed.
   - The code editor appears. The WLanguage events associated with the window are displayed.
2. Enter the following WLanguage code in the event "End of initialization of WIN_Product_form":

```
// Assigns the content of items to the controls
FileToScreen()
```

FileToScreen is used to display in the controls the data found in the data file, for the current record. In our case, the current record will be the record selected in the Table control of "WIN_Menu" window.

3. Close the code window.
4. Save the window (Ctrl + S).

▸ Display the "WIN_Menu" window in the editor: click the corresponding button in the open documents bar.

▸ Run the window test (among the quick access buttons).
   - In the list of products, click one of the products with the mouse.
   - Click the "Modify" button.
   - The detailed product window is displayed.

▸ Close the different test windows using the "X" at the top right of the windows.

Managing the product modification

We are now going to modify the "WIN_Product_form" window in order to manage the product modification.

We are going to:

- allow the user to modify the product image.
- add Button controls to save or cancel changes made in the product form.

Modifying the product image

The user will be able to modify the product image in the Product form. To do so, we are going to create a "Modify" Button control below the product image. This control will open a file picker to select the desired image.

▸ First of all, display (if necessary) the "WIN_Product_form" window in the editor: click the corresponding button in the open documents bar.

▸ To create the Button control to modify the image:

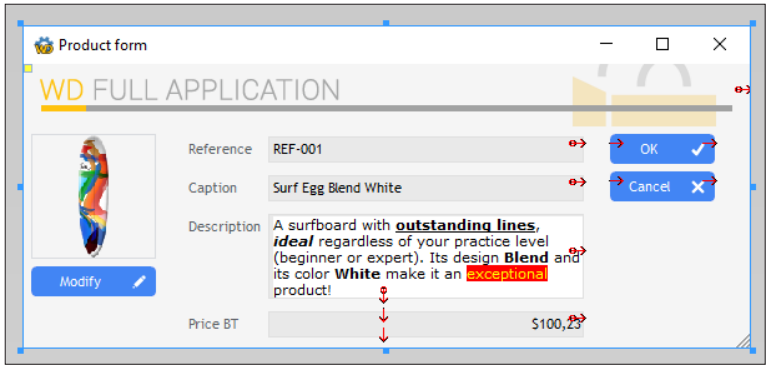
1. On the "Creation" pane, in the "Usual controls" group, expand "Button". Enter "Modify" in the search box at the top of the list of preset Button controls. Click "Modify button".

2. The shape of the control appears under the mouse pointer. Click in the window, below the product image.

3. If necessary, resize the Button control (using the handles) so that the control has the same width as the Image control.

Remark

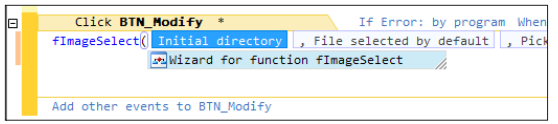
When resizing the control, an automatic magnetism effect simplifies the alignment of Button control with the Image control.



4. Right-click on the Button control. Select "Code" from the menu that is displayed.

5. In the code window that appears, write the following WLanguage code in the "Click" event:

```
fImageSelect(
```



6. The code editor proposes a code wizard.

- Click "Wizard for function fImageSelect".
- The code wizard starts.

Remark

WINDEV proposes several code wizards that will generate the corresponding code lines via simple questions. Don't hesitate to use them, you will avoid syntax errors.

7. Keep all the default options proposed by the wizard and validate. The corresponding WLanguage code is automatically entered.

```
sFile is string

// Opens the image picker
sFile = fImageSelect("", "", "Select an image...")
```

8. Add the following WLanguage code:

```
// If a file was selected
IF sFile <> "" THEN
  // Assigns the image to the Image control
  IMG_Visual = sFile
END
```

This WLanguage code is used to assign the selected file to the Image control in the window.

9. Close the code editor (click the cross in the top right corner of code window).

This feature will be checked when the management of modifications performed in the form is completed.

Validate the product modifications

Two Buttons controls have been created in the "WIN_Product_form" window:

- a "Validate" Button control to manage the validation of modifications,
- a "Cancel" Button control to manage the return to the product list.

We are now going to type the WLanguage code required for them to operate.

▸ First of all, display (if necessary) the "WIN_Product_form" window in the editor: click the corresponding button in the open documents bar.

▶ We will now write the WLanguage code associated with the "Validate" Button control.

1. Right-click the Button control and select "Code" from the popup menu.
2. Write the following code in the "Click" WLanguage event:

```
ScreenToFile()
HModify(Product)
Close()
```

Let's take a look at this WLanguage code:

- ***ScreenToFile*** is used to initialize the items with the values of linked controls, for the current record. This function is equivalent to the following code lines:

```
Product.Reference = EDT_Reference
Product.Caption = EDT_Caption
Product.Description = EDT_Description
...
```

Our window is using less than 10 controls and the benefit is already there; think of the windows that use a lot more controls: a single code line performs all assignments!

- ***HModify*** is used to update the file data for the current record.

3. Save the modifications by clicking among the quick access buttons.
4. Close the code window (click X at the top right corner of code editor).

▶ The modifications performed in the "WIN_Product_form" window must be taken into account in the Table control used to list the products. To do this, we will simply modify the WLanguage code associated with the "Modify" Button control in "WIN_Menu".

1. Display the "WIN_Menu" window in the editor: click the corresponding button in the button bar.
2. Select the "Modify" Button control and open the code editor (F2).
3. Modify the "Click" event as follows:

```
Open(WIN_Product_form)
TableDisplay(TABLE_Product, taCurrentSelection)
```

Let's study this code:

- ***TableDisplay*** is used to update the Table control.
- The *taCurrentSelection* constant is used to specify that the Table control must be updated from the position of selection bar.

4. Save the modifications by clicking among the quick access buttons.
5. Close the code window (click X at the top right corner of code editor).

Test of product modification

▶ To check the modification of a product:

1. Run the project test (among the quick access buttons).

2. WINDEV asks you to select the first project window. In our case, this window corresponds to "WIN_Menu". Select this window and validate.

Remark

The first project window corresponds to the first window that will be opened when starting the application. The first project window can be defined:

- during the project test.
- when creating the executable.
- in the "Project explorer" pane: all you have to do is select the requested window, display the popup menu and select "First project window".

When a first project window is defined, a small red 1 appears in front of the window name in the "Project explorer" pane.

3. The project starts in test mode and the first window is displayed.
4. Select a product from the product list (for example the first one) and click "Edit".
5. The details of the selected product are displayed in the form window.
6. Modify the price of 100.23 Dollars and enter 200.00 Dollars, then click the "Validate" button.
7. When going back to the list of products, you will notice that the price was updated for this product.
8. Display the product details again.
9. Click the "Modify" button below the image. The image picker is displayed. Select an image and validate. The image is displayed in the form.
10. Click the "Cancel" button. The image modification was ignored in the data file.
11. Click the cross to close the application.

Creating a new product

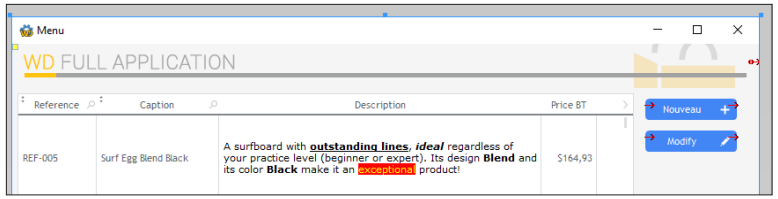
The principle for creating a product is as follows:

- In the product list window, we will add a "New" Button control that will open the "Product form" window.
- Then, we will modify the code of "Product form" window to manage the addition into the Product data file.

Adding a button

▶ To add a "New" Button control into "WIN_Menu":

1. Display the "WIN_Menu" window in the editor: click the corresponding button in the open documents bar.
2. On the "Creation" pane, in the "Usual controls" group, expand "Button" and select "New button".
3. The shape of the control appears under the mouse pointer. Click in the window: the Button control is created.
4. Position the Button control above the "Modify" Button control (if necessary, reposition the "Modify" Button control).



5. Modify the WLanguage code of this control:
   - Right-click on the "New" Button control and select "Code" in the popup menu.
   - Write the following code in the "Click" WLanguage event:

```
HReset(Product)
Open(WIN_Product_form)
TableDisplay(TABLE_Product, taCurrentRecord)
```

Let's study this code:

- ***HReset*** initializes the item variables in the "Product" data file with the default values to manage a new record.
- ***Open*** is used to open the product form to type the new record.
- ***TableDisplay*** is used to re-display the Table control. The ***taCurrentRecord*** constant is used in this code: the Table control is re-displayed from the current record.

6. Save the modifications by clicking among the quick access buttons.
7. Close the code window (click X at the top right corner of code editor).

Addition into the data file

▶ We are now going to modify the window of product form to manage the addition of a new record.

1. Open the "WIN_Product_form" window in the editor.
2. We will modify the WLanguage code associated with the "Validate" Button control:
   - Right-click on the "Validate" Button control and select "Code" in the popup menu.
   - In the "Click" event, replace the existing WLanguage code with the following:

```
ScreenToFile()
IF Product..NewRecord THEN
  HAdd(Product)
ELSE
  HModify(Product)
END
Close()
```

Let's take a look at this WLanguage code:

- **..NewRecord** is used to find out whether the current record must be created.
- If ***HReset*** was called beforehand, the property returns True (click on "New product") and the record must be created by ***HAdd***.
- Otherwise, the current record already exists and it must be modified by ***HModify***.
- ***HAdd*** adds the record into the data file. This function takes the values in memory and writes the content of file items into the data file itself. The indexes are immediately and automatically updated. In this case, it is the "Product" data file that is updated.

Remark

The test code of new record can be replaced by ***HSave***. This function is used to check whether the record is already found in the data file, and it allows you to add it or to modify it. The code becomes:

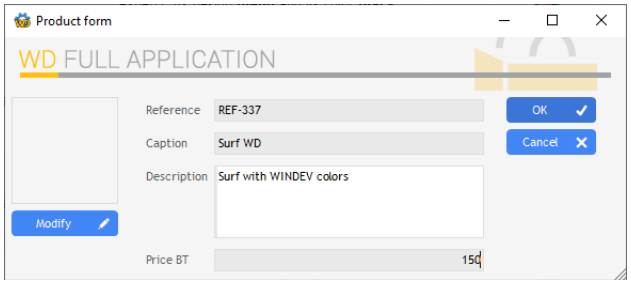
```
ScreenToFile()
HSave(Product)
Close()
```

3. Save the modifications by clicking among the quick access buttons.
4. Close the code window (click X at the top right corner of code editor).

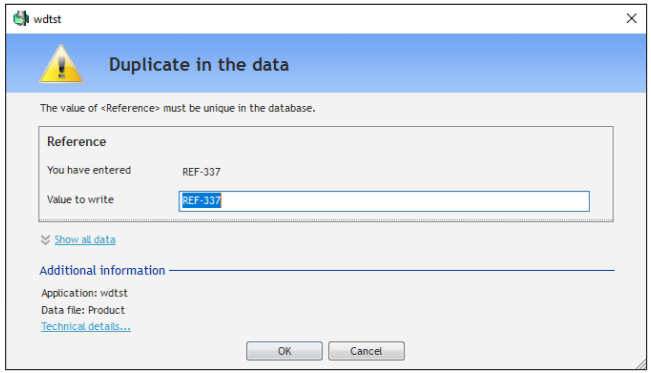
Test of product addition

▶ To check the product addition:

1. Run the project test (GO among the quick access buttons).
2. In the list of products, click the "New" button.
3. Enter a new product with the following characteristics:
   - Reference: REF-337
   - Caption: Surf WD
   - Description: Surf with WINDEV colors
   - Price: 150



4. Validate. The new product is displayed in the list of products.
5. Click the "New" button again.
6. Enter a new product with the following characteristics:
   - Reference: REF-337
   - Caption: Surf WB
   - Description: Surf with WEBDEV colors
   - Price: 150
7. Validate. A specific window is displayed:



This window informs the user that a duplicate was found: indeed, the reference (that is a unique key) is identical for two products. This window is used to modify the reference value: type "REF-338" for example.
This window is one of the windows for automatic management of HFSQL errors.

Remark

Several errors may occur when adding or modifying a record: duplicate error, integrity error, password error, ...

WINDEV proposes several modes for managing these errors:

- **the automatic mode**: for each error that occurred when managing database records, a specific page is displayed to the user. This window allows the user to modify his data directly.
- **the advanced programmed mode**: a procedure or a custom window for error management is called whenever an error occurs when managing the database records.

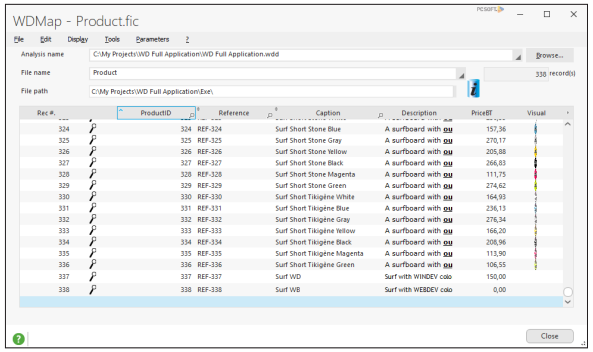
The example "WD Detecting HFSQL Errors", supplied with WINDEV, allows you to check these different modes for error management. This example can be opened via the WINDEV home page (Ctrl + <), "Open an example" option.

Viewing the records

The new added records can be immediately viewed in the Table control of the "WIN_Menu" window.
However, in some cases (when running tests for example), you may want to see the content of the data file directly.
WINDEV proposes a tool used to view the content of data files while the application is being developed (when the viewing windows have not been created yet).
This tool is named WDMAP. We are going to use it to see the content of the Product data file.

▶ To start WDMAP:

1. On the "Tools" pane, in the "Database" group, click "WDMap".
2. Select the "Product" data file. The data file content is displayed.



3. Sort the elements by reference (click the header of the "Reference" column) and the two added records appear.

Lesson 4.3. Simple search and record browse

This lesson will teach you the following concepts

- Managing the search for records.
- Browsing records.

Estimated time: 30 mn

Overview

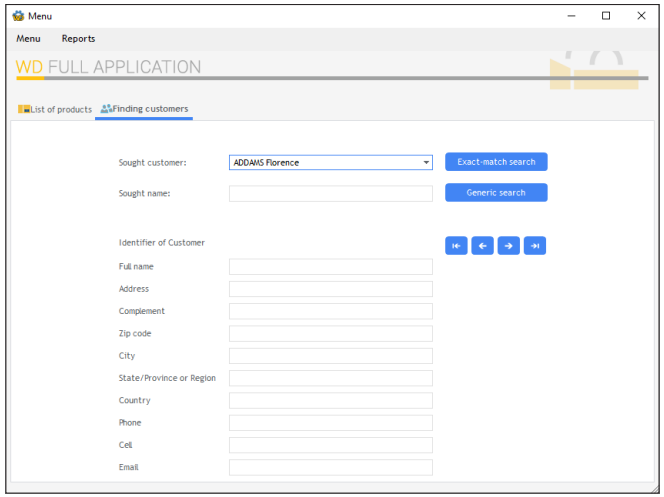
We have managed the addition and modification of records. It's fine. But it would be even better to be able to find a record (to modify it for example). That's what we are going to do now.

We are going to modify the main window of "WD Full Application" application to use tab panes.

- The first tab pane will display the product list and manage the addition and modification of products (these operations have been performed in previous lessons).
- The second tab pane will propose to find a customer by name. The customer form will be displayed in the same tab pane.

Two search modes will be implemented:

- an exact-match search,
- a generic search.



Then, you will have the ability to browse the records found in the data file from the customer displayed.

Answer

A corrected project is available. This project contains the different windows created in this lesson. To open the corrected project, in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (With windows)".

Modifying the window: using a Tab control

In this example, the search will not be performed in a new window: the WIN_Menu window will be used. At this time, this window displays the list of products.

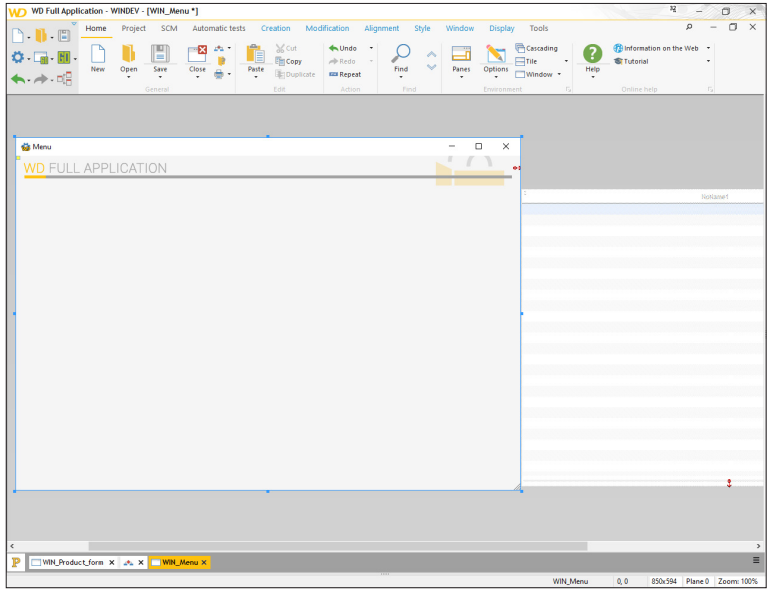
We will use Tab controls to display various data in this window. Tab controls are used to group information by theme. The user can directly access a theme by clicking the requested "pane".

This system is used in all description windows of WINDEV.

Creating the Tab control

▶ To display the data of the "WIN_Menu" window in a Tab control:

1. Display the "WIN_Menu" window if necessary:
   - from the open documents bar.
   - by double-clicking its name the "Project explorer" pane.
2. Select the window controls and move them to the home area of window. This may seem "strange", but it will make it easier to work on the Tab control and then associate the controls with the desired tab pane.

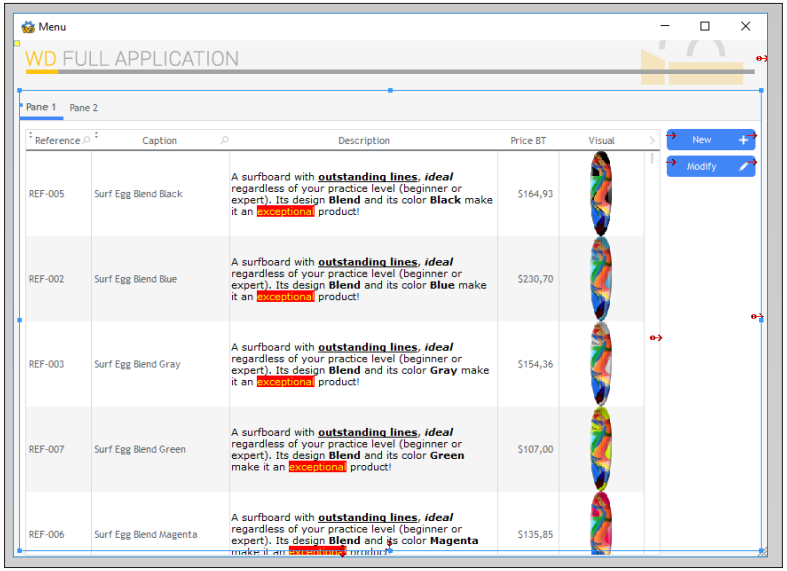


3. Create a Tab control in a window:
   - On the "Creation" pane, in the "Containers" group, expand "Tab and associated" and select "Tab".
   - Click in the window to create the Tab control.
   - Position the Tab control at the top left of window.
   - A Tab control is created with two panes by default. The Tab control occupies the entire available space in the window.
4. Select the controls in the home area of the window and move them to pane 1 of the Tab control.
5. A green border appears when hovering controls on the Tab control. This border indicates that the controls placed on the current pane will be automatically associated with the pane.
6. Enlarge (if necessary) the window and the Tab control to display the Table control and the buttons in the Tab control.

Remark

Handling controls in a Tab control

- To check whether all controls are associated with the current tab pane, click another tab pane: all controls associated with the tab pane disappear.
- To resize the Tab control while ignoring the anchors defined for the controls found in the pane, press the Shift key during the resize operation.



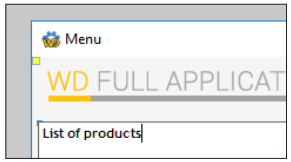
Modifying the Tab control

▸ Modify the name of Tab control:

1. Double-click the Tab control that was just created: the control description window appears.
2. Modify the control name. The control is named "TAB_Viewing".
3. Validate the control description window.

▸ Two panes are created by default. We are going to modify their caption directly:

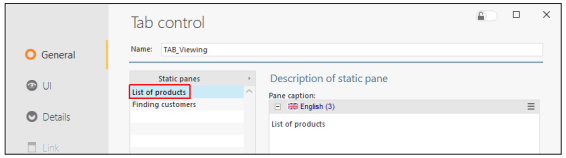
1. Select the Tab control.
2. Click "Pane 1". The pane becomes editable: type the "List of products" caption. Press Enter to validate.



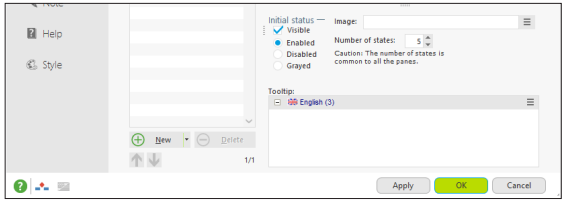
3. Click "Pane 2". The pane is displayed. Click "Pane 2" again. Type the caption: "Finding customers". Press Enter to validate.

▸ We will associate an image with each pane of the Tab control. To do so:

1. Display the description window of Tab control (double-click the control for example). The "General" tab of the description window allows you to configure the characteristics of each pane in the Tab control. The list of panes is displayed on the left. For each selected pane, its characteristics can be modified in the right section of screen.
2. In the description window, select the "List of products" pane.



3. The pane characteristics are displayed.

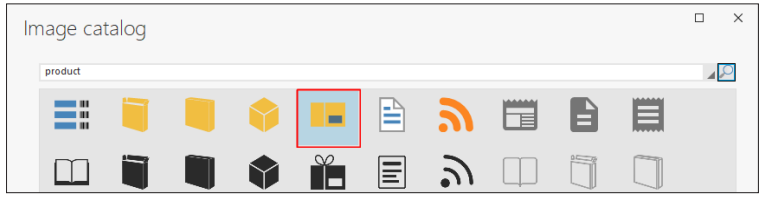


4. We are going to associate an image to the tab pane via the WINDEV image catalog.

Remark

As soon as an image can be displayed in a control or window, WINDEV proposes to use the image catalog. This image catalog is started via the "Catalog" option (available by clicking the ≡ button). This catalog contains hundreds of images, cliparts, ...

5. Click the ≡ button on the right of "Image" control. Select "Catalog" from the popup menu that is displayed. The window of image catalog is displayed.
6. Type "Product" in the search area. Click the magnifier to start the search.



7. Select and validate ("OK").
8. The screen for configuring the generated image is displayed. This screen is used to modify the characteristics of the image to generate: size, lightness, ... Keep the default options and validate.
9. The image is generated in the project directory and the corresponding file is automatically associated with the current element.
10. In the Tab control description window, click "Finding customers" in the list of static panes.
11. Click the ≡ button on the right of "Image" control. Select "Catalog" from the popup menu that is displayed. The window of image catalog is displayed.
12. In the search area, specify "Person". Click the magnifier to start the search.
13. Select, among the proposed images, the icon representing two persons ( ) and validate.
14. Keep the options found in the setting screen of generated image and validate.
15. Validate the description window of Tab control.

▸ A new control was just created: we must now manage its anchoring in the window. When the window is enlarged, the Tab control must also be enlarged in order to occupy the entire available space. Therefore, the control must stretch to the right and to the bottom.

1. Select the Tab control.
2. Display the popup menu (right mouse click) and select "Anchor".
3. In the window for defining the anchor, select and validate.
4. Save the window by clicking among the quick access buttons.

Implementing the search

The "WIN_Menu" window was entirely created. We can now focus on the content of the Tab control pane for finding customers.

This Tab control pane will be divided into several areas:

- An area for typing the sought elements.
- An area for displaying the information found.
- An area containing the browse buttons.

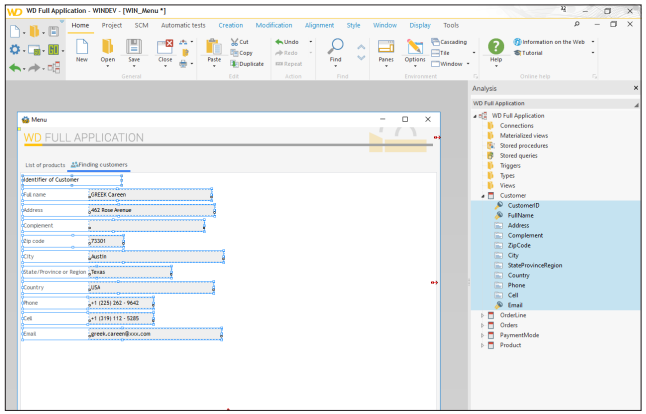
Area for displaying the information found

First of all, we are going to create the different controls used to display the information about the customer.

The method for creating edit controls in the window is the same as the method for creating the form window of product: a simple Drag and Drop from the "Analysis" pane..

▶ To create the different controls used to display information about the customer:

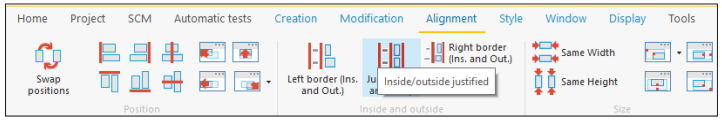
1. Display the WIN_Menu window in the editor and click the "Finding customers" tab pane. The empty tab pane appears.
2. Display the "Analysis" pane in the editor if necessary: on the "Home" pane, in the "Environment" group, expand "Panes" and select "Analysis". The different data files described in the "WD Full Application" analysis appear in the pane.
3. Click the icon on the left of the "Customer" data file: the items found in the data file are listed.
4. With the mouse, select all items displayed in the tab pane. You have the ability to use the multi-selection by holding the Ctrl key down.
5. Drag and Drop these items to the "Finding customers" tab pane.



6. The controls are automatically created in the Tab control pane. These controls are automatically linked to the corresponding item in the data file.
7. Save the window (Ctrl + S).

▶ We are going to align the controls and give them the same size:

1. Select the "City" control then all edit controls found in the window (keep the Ctrl key down while clicking the different controls for example). The first selected control will be used as reference for the size of other controls.
2. On the "Alignment" pane, in the "Inside and outside" group, click "Justify (Ins. and Out.)".



▶ Run the window test (among the quick access buttons). Click the "Finding customers" tab pane. The window is displayed with empty controls.

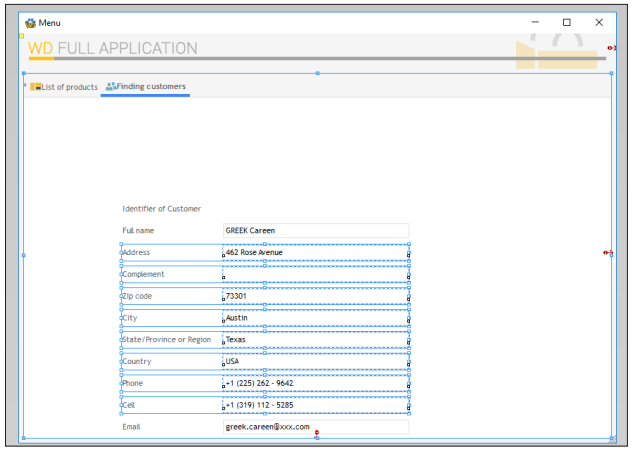
▶ Close the test window to go back to the editor.

Exact-match search

To perform an exact-match search, we are going to select the customer name in a Combo Box control. The "Find" Button control will be used to display the form of matching person. A single person corresponds to the selected name.

▶ The Combo Box control will be positioned above the created controls. If your controls are placed too close to the top of the Tab control pane, it is necessary to move them down. To do so:

1. Select all the controls in the Tab control pane:
   - Press Ctrl + A: all the controls in the window and in the current pane are selected.
   - Press the Ctrl key.
   - Click the Tab control: only the controls in the Tab control pane are selected.
2. With the mouse, move one of the selected controls to the bottom.



3. All controls are moved to the bottom.

► To create the search control:

1. Create a Combo Box control: on the "Creation" pane, in the "Usual controls" group, click "Combo box".

2. Click the "Finding customers" pane, between the tab pane and the control "Identifier of Customer".

3. The Combo Box control creation wizard for is displayed. We will create a Combo Box control based on the "Customer" data file.

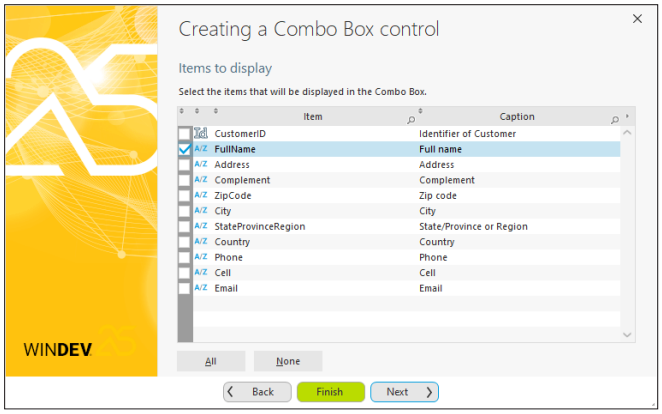
4. Select "Display the data found in a file or in an existing query".

Go to the next wizard step.

5. Select the "Customer" data file. Go to the next step.

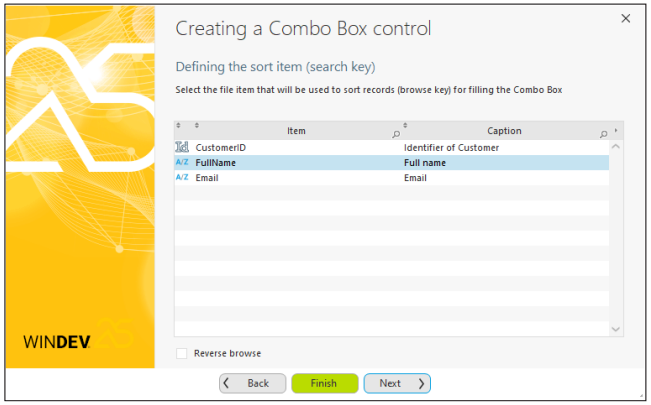
6. We are going to display the last and first names of the person.

- Deselect the "CustomerID" item.
- Select the "FullName" item.



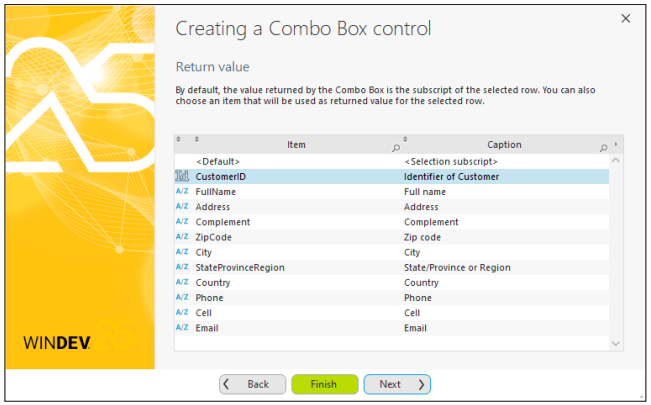
7. Go to the next step.

8. The "FullName" item will also be used to sort the list of customers displayed in the Combo Box control. Select the "FullName" item.



9. Go to the next step.

10. The value returned by the Combo Box control will be the "CustomerID" identifier. This is the value that will be sought in the Customer data file. Select the "CustomerID" item.



Go to the next step.

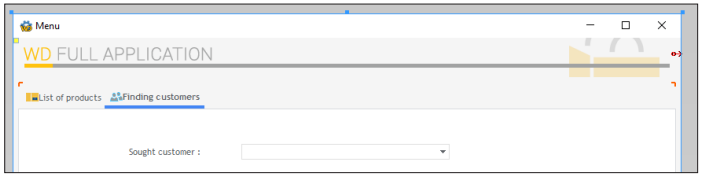
11. The Combo Box control will be linked to no item. Keep "No" and go to the next step.

12. Validate the next step ("Next").

13. Give a name ("COMBO_Customer" for example) and a caption ("Sought customer" for example) to the control.

14. Finish the wizard ("Finish" button). Position the Combo Box control in the window (top left corner for example).

15. Select the Combo Box control and resize it (via the handles) in order for the full customer name to be displayed.



▸ To create the Button control to find customers:

1. Create a Button control: on the "Creation" pane, in the "Usual controls" group, click Ok.
2. Position the Button control beside the Combo Box that was just created.
3. This control is named "BTN_ExactMatchSearch" and its caption is "Exact-match search".
4. If necessary, adjust the size of the control so that the caption is displayed properly in the control.
5. Write the following WLanguage code in the "Click" event of the control:

```
// Finds the customer from his identifier
HReadSeekFirst(Customer, CustomerID, COMBO_Customer)
IF HFound(Customer) THEN
  // Displays the customer data
  FileToScreen()
END
```

6. Let's take a look at this WLanguage code:
   - ***HReadSeekFirst*** is used to perform an exact-match search. In this example, the search is performed on the Customer data file and on the CustomerID item. The sought value corresponds to the last parameter of the function. Here, the sought value corresponds to the value selected in the Combo Box. This value is obtained by using the name of the Combo Box control (COMBO_Customer).
   - ***HFound*** is used to check the search result. If ***HFound*** returns True, a value was found ; if ***HFound*** returns False, no value was found. Any record found is read: it becomes the current record in the data file.
   - In this code, if the record was found, it is displayed by ***FileToScreen***.

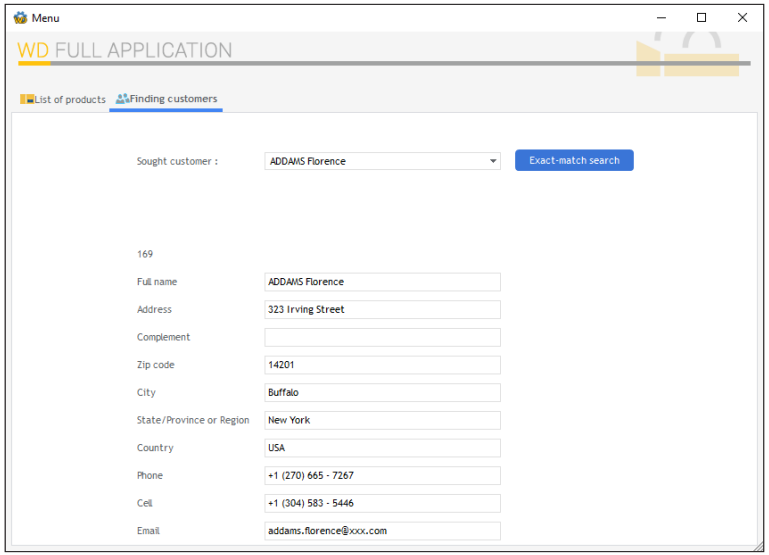
Remark

FileToScreen and ***ScreenToFile*** perform the reverse operation: the data found in the file items is displayed in the corresponding controls.

7. Close the code editor.
8. Save the window (Ctrl + S).

▸ Let's run a window test:

1. Run the window test (among the quick access buttons).
2. Click the "Finding customers" tab pane if necessary.
3. Select a value in the Combo Box control and click on "Exact-match Search".



4. The result is immediate.
5. Close the test window to go back to the editor.

Generic search

We are now going to perform a generic search. Instead of searching for the exact value that was entered, we are going to search for all the elements that start with the value entered.

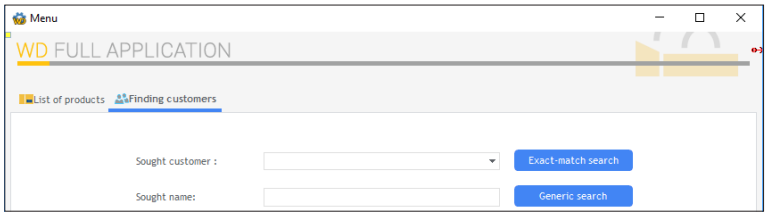
To perform this search, we will create a Edit control to enter the name you are looking for, and a Button control to launch the search.

▸ To create the search control:

1. Display (if necessary) the "Finding customers" pane tab of "WIN_Menu" in the editor.
2. Create an edit control: on the "Creation" pane, in the "Usual controls" group, click .
3. In the window, click below the Combo Box control ("Sought customer"). The edit control is created.
4. This control is named "EDT_Sought_Name" and its caption is "Sought name".

▸ To create the Button control to launch the generic search:

1. Create a Button control: on the "Creation" pane, in the "Usual controls" group, click Ok.
2. Position the control beside the edit control that was just created.



3. This control is named "BTN_GenericSearch" and its caption is "Generic search".
4. If necessary, adjust the size of the control so that the caption is displayed properly in the control.
5. Write the following WLanguage code in the "Click" event of the button:

```
// Finds a customer from his name
HReadSeek(Customer, FullName, EDT_Sought_Name)
IF HFound(Customer) THEN
  // Displays the customer data
  FileToScreen()
ELSE
  Error("No customer matches")
END
```

HReadSeek is used to perform a generic search. In this example, the search is performed on the "Customer" data file and on the "FullName" item. The sought value corresponds to the value typed in the "EDT_Sought_Name" control. This value is obtained by using the control name.

Remark

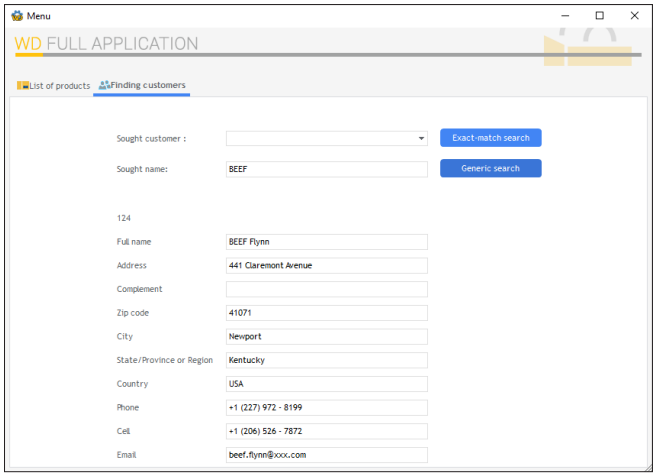
To perform an exact-match search, call ***HReadSeek***: to do so, the ***hIdentical*** constant must be used.

6. Close the code window.
7. Save the window (Ctrl + S).

▸ Let's run a window test:

1. Run the window test (among the quick access buttons).
2. Click the "Finding customers" tab pane if necessary.

3. Enter a value in the edit control (for example "BEEF") and click "Generic search".



4. The result is immediate. However, if several records correspond to the search, only the first one is displayed.
5. Stop the test and go back to the editor.

Browsing forms

We are now going to add four Button controls to browse the different records of the "Customer" data file. These controls are in a "recorder" layout.

They include an icon and they are used to display:

- the first record,
- the previous record,
- the next record,
- the last record.

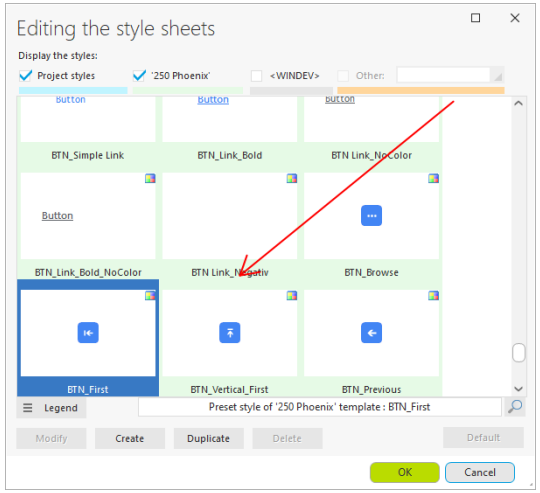
▸ To create the Button control used to display the first file record:

1. If necessary, display the "Finding customers" Tab control pane in the editor.
2. On the "Creation" pane, in the "Usual controls" group, click Ok.
3. The shape of the control appears under the mouse pointer. Then click the Tab control pane, below "Generic search". The Button control is automatically created.

▸ A skin template will be used to standardize the interface of project windows. This skin template defines the style for all controls found in the windows. A specific style is defined for each recorder Button control. To apply this style:

1. Select the Button control if necessary.

2. Display the popup menu (right mouse click) and select "Choose a style".
3. All styles defined for the buttons are displayed in the window.



Select "BTN_First".

4. Validate.

► Modify the name of Button control:

1. Display the description window of control:
   - Select the Button control.
   - Display the popup menu (right mouse click) and select "Description".
2. This control is named "BTN_First".
3. Validate the description window.

► We are now going to type the WLanguage code associated with this control:

1. Right-click the control. Select "Code" from the menu that is displayed.
2. In the code window that appears, write the following WLanguage code in the "Click" event:

```
// Reads the first customer
HReadFirst(Customer)
IF HOut(Customer) = True THEN
  // No customer
  Info("No form to view.")
ELSE
  // Displays the customer data
  FileToScreen()
END
```

HReadFirst is used to read the first file record according to the key used for the last search (FullName in our case).

► Likewise, create 3 Button controls.

- The names of these controls are: "BTN_Previous", "BTN_Next" and "BTN_Last".
- These controls are respectively associated with the styles: "BTN_Previous", "BTN_Next" and "BTN_Last".
- The WLanguage code corresponding to the "Click" event of "BTN_Previous" is:

```
// If no search is in progress
IF HOut(Customer) THEN
    // Reads the last customer
    HReadLast(Customer)
END

// Reads the previous customer
HReadPrevious(Customer)
// If the beginning of file is exceeded
IF HOut(Customer) = True THEN
    Info("Start of file reached.")
ELSE
    // Displays the customer data
    FileToScreen()
END
```

- The WLanguage code corresponding to the "Click" event of "BTN_Next" is:

```
// If no search is in progress
IF HOut(Customer) THEN
    // Reads the first customer
    HReadFirst(Customer)
END

// Reads the next customer
HReadNext(Customer)
// If the end of file is exceeded
IF HOut(Customer) = True THEN
    Info("End of file reached.")
ELSE
    // Displays the customer data
    FileToScreen()
END
```

- The WLanguage code corresponding to the "Click" event of "BTN_Last" is:

```
// Reads the last customer
HReadLast(Customer)
IF HOut(Customer) = True THEN
    // No customer
    Info("No form to view.")
ELSE
    // Displays the customer data
    FileToScreen()
END
```

In these different codes:

- ***HReadLast*** is used to read the last file record according to the key used for the last search.
- ***HReadNext*** reads the record whose key value is immediately greater than the one of current record.
- ***HReadPrevious*** reads the record whose key value is immediately less than the one of current record.

In any case:

- ***HOut*** is used to find out whether the data file is empty.
- ***FileToScreen*** is used to display the record on the screen.

Remark

The code in the "Next" and "Previous" Button controls contains additional lines of code to:

- Check whether a search was already performed in the Customer data file (test of ***HOut*** to find out whether the file reading has started).
- Read the first or last record if necessary. Indeed, the next or previous record cannot be read if no reading was performed in the data file.

► Save the window by clicking among the quick access buttons.

Window test

► Let's run a window test:

1. Run the window test (among the quick access buttons).
2. Click the "Finding customers" tab pane if necessary.
3. Find a customer (perform a generic search on "BEEF" for example).
4. Browse the data file by clicking each one of the browse buttons.
5. Stop the test to go back to the editor.

Lesson 4.4. Multicriteria search

This lesson will teach you the following concepts

- Creating a query with parameters.
- Creating the interface used to select the search criteria.
- Passing parameters to a query.
- Displaying the query result in a Table control.

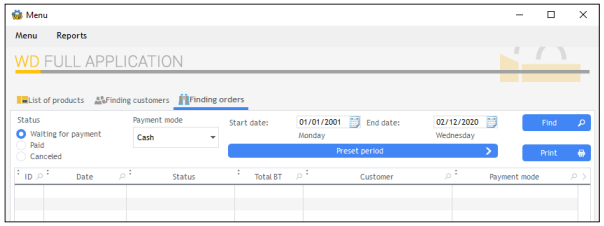
Estimated time: 40 mn

Overview

In the previous lesson, we have explained how to perform a search on a single criterion (the customer name). In this lesson, we will give the user the ability to perform a multicriteria search.

In our example, this search will be done on the "Orders" data file. The user will be able to select:

- the order status,
- the payment mode,
- the order dates taken into account.The interface of "WIN_Menu" window is as follows:



This interface includes:

- controls used to select the search criteria.
- Button controls used to start the search or print the result.
- a Table control used to display the search result. This Table control is based on a query. This query will be used to select the records to display. The Table control will list the search result.

The first step consists in creating the query used to select the records.

Remark

What is a select query?

A select query is a query that will "choose" the records corresponding to the specified criteria.

This type of query is called a select query because the SELECT command is used in SQL language.

Answer

A corrected project is available. This project contains the different windows created in this lesson. To open the corrected project, in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (With windows)".

Creating the query used to find orders

Creating the query

▸ The query editor will be used to create the query.

1. Click among the quick access buttons. The window for creating a new element is displayed: click "Query". The query creation wizard starts.

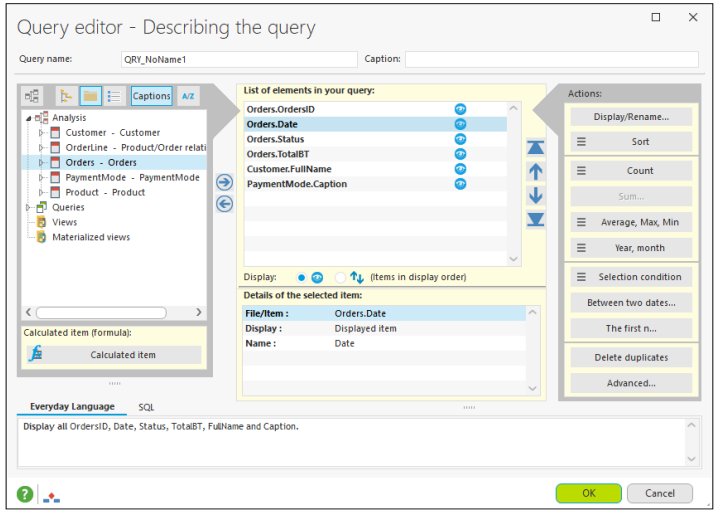
2. Select the "Select" option.
Indeed, this query will be used to select records. Go to the next step.

3. The query description window is displayed. To build the query, we are going to select the elements that will be displayed in the result.

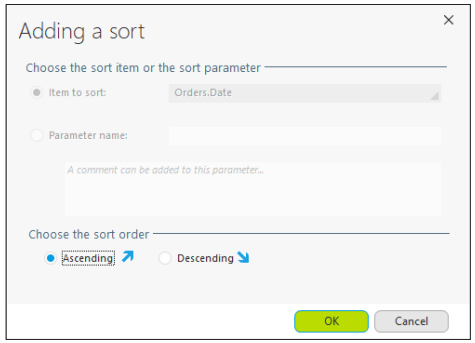
4. Double-click the items found in the analysis on the left of description window. The items taken into account are displayed in the middle of the screen.
Remark: To display the items of different data files, simply click the arrow in front of the data file name.
We want to display:

- information regarding the order. Expand the "Orders" data file (click the arrow) and double-click the items: OrdersID, Date, Status and TotalBT.
- information regarding the customer who placed the order. Expand the "Customer" data file (click the arrow) and double-click the "FullName" item.
- information regarding the payment mode of order. Expand the "PaymentMode" data file (click the arrow) and double-click the "Caption" item.

The description window of query is as follows:

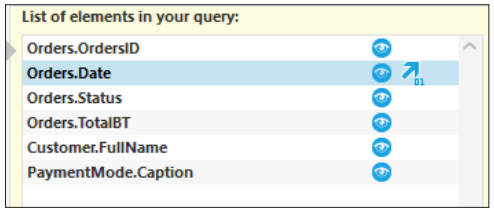


5. The data will be sorted by date.
   - In the list of query elements, select the "Orders.Date" item, then click the "Sort" button and select "Sort on the selected item".
   - In the window that appears, indicate an ascending sort on the item.

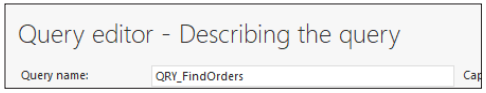


   - Validate.

6. A blue arrow with the number 01 appears on the right of "Orders.Date" item. This arrow indicates that an ascending order has been applied on this item. The number "01" indicates that this sort will be applied first.

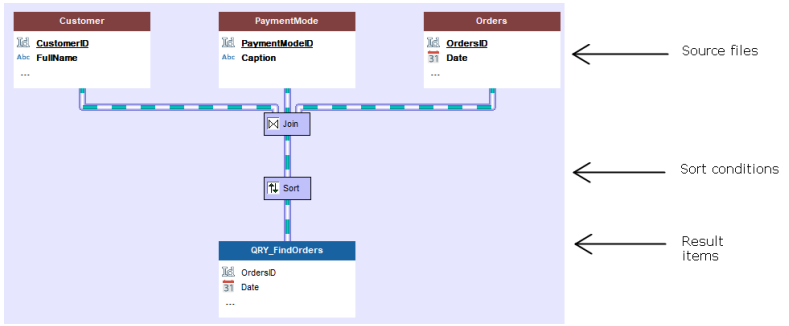


7. Give a name to the query: type "QRY_FindOrders" instead of "QRY_NoName1" in the "Query name" area:



8. Validate the query description window ("OK" button).
9. The save window is displayed. Validate the proposed information.

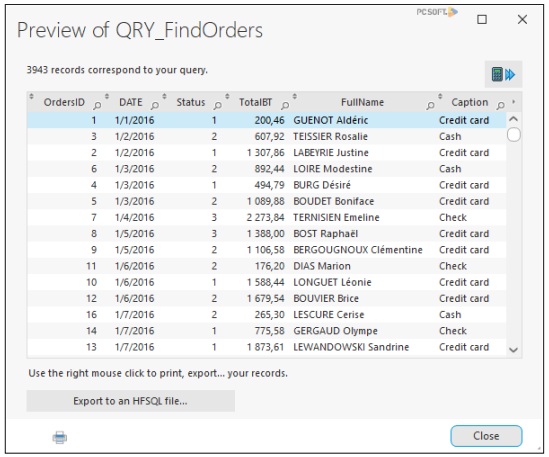
10. The graphic query representation is displayed:



Query test

Like all elements found in a WINDEV project, you have the ability to run the test of query that was just created:

1. Click  .
2. The result is displayed in a window:



The result lists ALL orders.
In our case, we want to display the orders corresponding to the search criteria only. To do so, we must use a query with parameters.

Remark

A popup menu is displayed when a right click is performed on the query result. You have the ability to export the result to:

- an XLSX file (Excel).
- a Word or OpenOffice file.
- an XML file (eXtensible Markup Language).
- a HFSQL file.

3. Close the window.

Using parameters in the query

In our example, the user will be able to select a value for the following search criteria:

- Order status.
- Order payment mode.
- Order date.

We must modify the query in order for these search criteria to correspond to the query parameters.

▸ To define the query parameters, display the description window of query: double-click the background of graphic query representation (or select "Query description" from the popup menu).

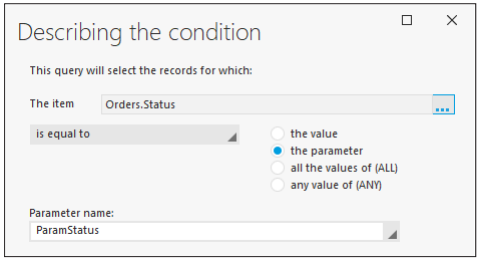
▸ To manage the "order status" parameter:

1. Select the Orders.Status item (in the middle of the screen).

2. Expand "Selection condition" and select "New condition".

3. In the window that is displayed, we are going to specify that the selection condition corresponds to a parameter:

- Select "Is equal to".
- Select "the parameter".
- The name of the parameter is automatically proposed: "ParamStatus".

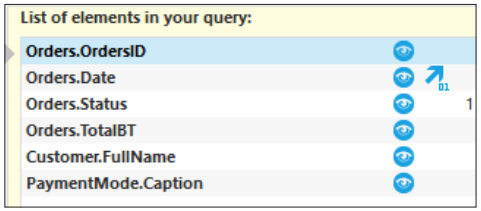


Remark

We advise you to keep "Param" as the prefix of the query parameters. This allows you to easily find them in the code editor.

To find a query parameter, simply type 'Param' and the completion feature of the code editor will propose all the parameters.

4. Validate the condition description window. The number "1" appears on the right of "Orders. Status" item, indicating that a selection condition was defined.



▸ We are now going to define a condition on the payment mode. This item is not found in the query result but a condition will be applied to it. To do so, the item will be included in the query result and it will be made invisible so that it is not visible in the result.

1. On the left part of the query description window, in the "Orders" data file, double-click on the item "PaymentMethodID". The "PaymentModeID" item appears in the list of query elements.

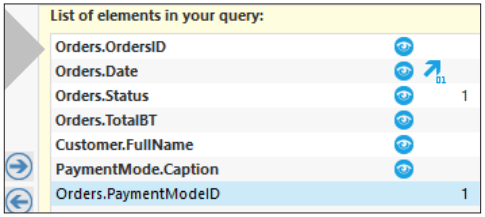
2. To avoid displaying this item in the result:

- Click the icon found on the right of item.
- In the menu that is displayed, select "Don't display".

3. To define a selection condition on the "Orders.PaymentModeID" item:

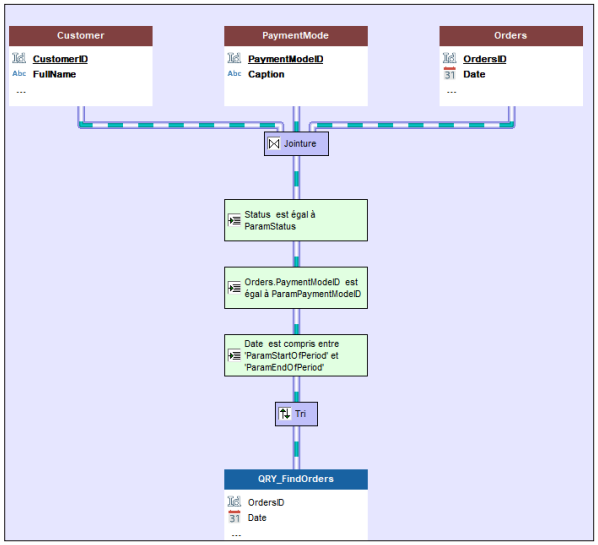
- Select the "Orders.PaymentModeID" item (in the middle of the screen).
- Expand "Selection condition" and select "New condition".
- In the window that is displayed, specify that the selection condition corresponds to a parameter:
  - Select "Is equal to".
  - Select "the parameter".
  - Specify the parameter name: "ParamPaymentModeID".

4. Validate the definition of selection condition.



▶ The last selection condition to define affects the order date. This date must be included between two dates typed by the user.

1. In the list of query elements, select the "Orders.Date" item.
2. Click the "Between two dates ..." button. This button allows you to define a selection condition.
3. In the window that is displayed:
   - The selection condition is "Is included between".
   - Click "the parameter".
   - Specify the parameter name: "ParamStartOfPeriod".
   - Click the second "to parameter".
   - Specify the parameter name: "ParamEndOfPeriod".
4. Validate the definition of selection condition.
5. Validate the description window of query. The query graph is modified to take into account the selection conditions that have been defined.

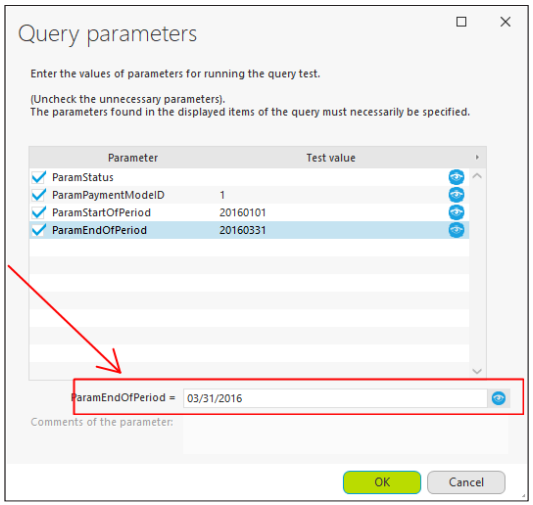


6. Save the query by clicking among the quick access buttons.

Test of query with parameters

▶ To run the test of query with parameters:

1. Click .
2. A window is displayed, allowing you to type the different query parameters.
3. Type the following data:
   - Uncheck the ParamStatus parameter.
   - Select the ParamPaymentModeID parameter. In the lower section of the screen, type "1".
   - Select the ParamStartOfPeriod parameter. In the lower section of the screen, type "01/01/2016".
   - Select the ParamEndOfPeriod parameter. In the lower section of the screen, type "03/31/2016".



4. Validate the window. The query result corresponding to the specified parameters is displayed.
5. Close the window.

We are now going to create the interface used to specify the parameters of this query, to run it and to display the result.

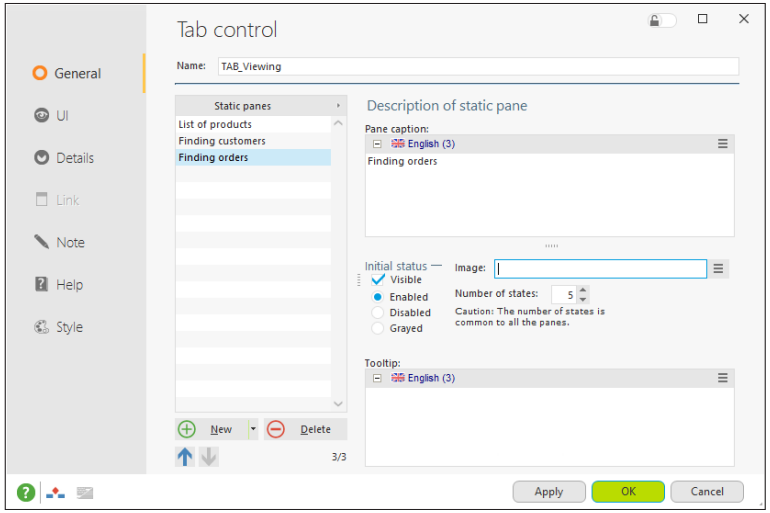
Creating the interface used to perform a multicriteria search

Modifying the Tab control

We are going to add a tab pane into the "WIN_Menu" window to display the result of multi-criteria search.

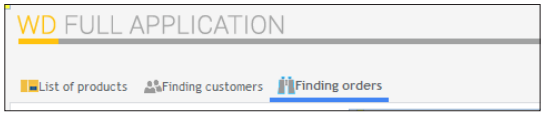
▸ To add a tab pane:

1. Display (if necessary) the "WIN_Menu" window in the editor.
2. Select the Tab control.
3. Display the description window of control ("Description" from the popup menu).
4. In the "General" tab of description window, click the "New" button. A third tab pane appears.
5. Click "Pane 3".
6. In the "Description of static pane" section, type the pane caption: "Finding orders".



7. We are going to associate an image to the tab pane via the WINDEV image catalog. Click the ☰ button on the right of "Image" control. Select "Catalog" from the popup menu that is displayed. The window of image catalog is displayed.

8. In the search area, specify "Search". Click the magnifier to start the search.
9. Among the proposed images, select the icon representing the binoculars ( ) and validate.
10. Keep the options found in the setting screen of generated image and validate.
11. Validate the description window of Tab control.



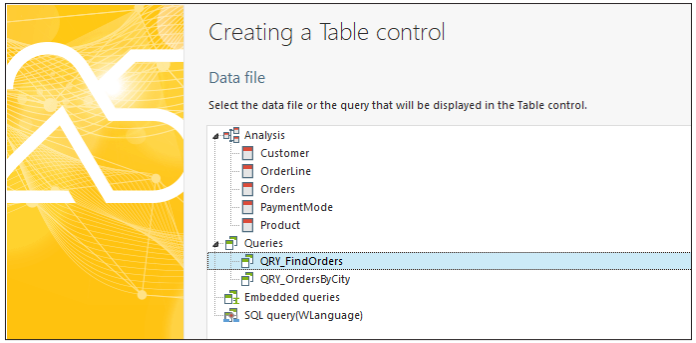
Creating the controls used to configure the criteria and to display the result

We now want to display the result of our multicriteria search.
We are going to create a Table control based on the query then to create the different controls allowing the user to select the search criteria.

Creating the Table control

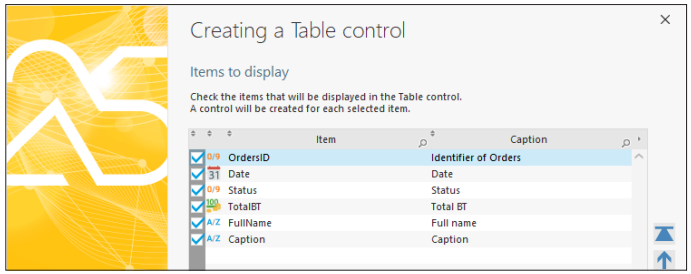
▸ To create the Table control used to display the search result:

1. In the "WIN_Menu" window, click the "Finding orders" pane. The empty tab pane appears.
2. Create a Table control: on the "Creation" pane, in the "Data" group, expand "Table and list box" and select "Table (Vertical)".
3. Click the Tab control pane previously selected: the Table control creation wizard starts.
4. The Table control will be based on the "QRY_FindOrders" query (that was created beforehand). Select "Display the data found in a file or in an existing query". Go to the next step of the wizard.
5. Select the query that will be used as data source for the Table control:
   - Expand the "Queries" group if necessary.
   - Select the "QRY_FindOrders" query.



- Go to the next step of the wizard.

6. Select all suggested items if necessary.

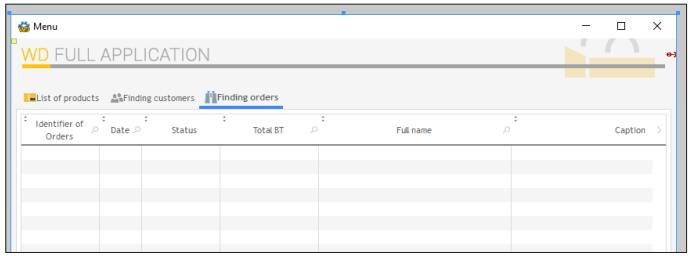


7. Go to the next step of the wizard.

8. Keep the default options in the different wizard steps and validate the creation of Table control.

9. The Table control is automatically created in the "Finding orders" pane of the Tab control.

10. Modify (if necessary) the position of Table control so that it is entirely displayed in the tab pane.



► For better legibility, we are going to rename the captions of columns in the Table control.

1. Open the Table control description (double-click the control).

Remark

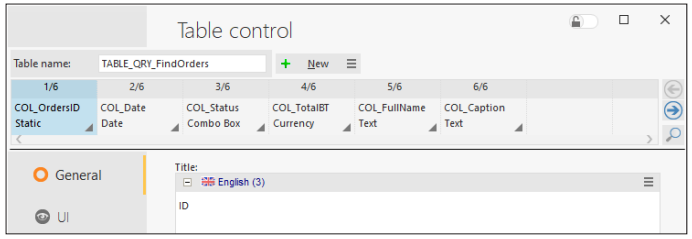
The description window of a Table control includes two sections:

- the upper section, presenting the name of control, the name of columns as well as their type.
- the lower section, containing the different description tabs.

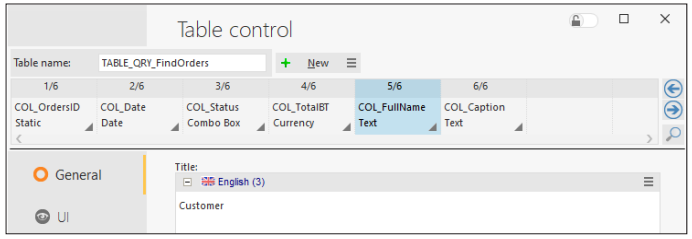
If the name of Table control is selected, the lower section presents the characteristics of Table control.

If a column is selected, the lower section presents the characteristics of columns.

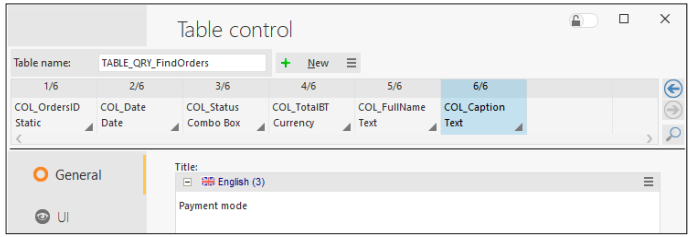
2. Click the "COL_OrdersID" column. The column title is displayed in the lower section of the screen. Replace the "Identifier of Orders" caption by "ID".



3. Click the "COL_FullName" column. Replace the "Full name" caption by "Customer".



4. Click the "COL_Caption" column. Replace the "Caption" caption by "Payment mode".



5. Validate the description window of Table control. The control is automatically updated with the modifications performed.

6. Enlarge the "Date" and "Status" columns in the Table control with the column width handles.

7. Reduce the "ID" and "Payment mode" columns in order for all columns to be displayed in the Table control.

8. Save the window by clicking among the quick access buttons. We are going to check the sizes of columns by running the window.

Remark

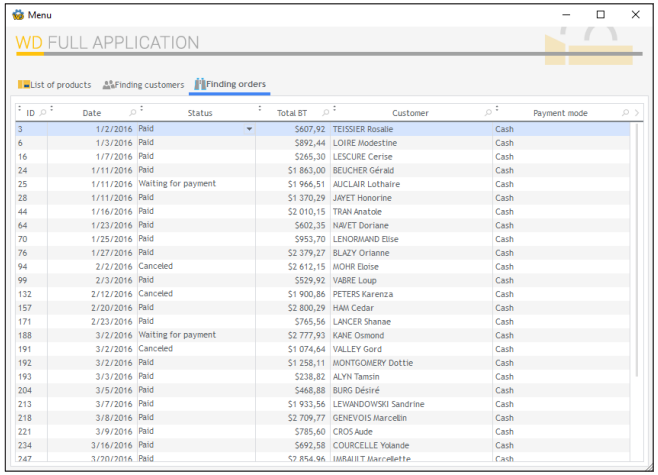
Live Data and controls based on queries

The Live Data is not displayed in the controls that use a query as data source for the following reason: the data displayed in the control depends on the query result therefore it is known at run time only.

► We are going to run a first test of this window:

1. Click among the quick access buttons.

2. Click the "Finding orders" tab pane. Only some orders are displayed, like during the last query test run in the editor, when parameters were specified in the test window of query.



3. Close the test window to go back to the editor.

► Let's take a look at the WLanguage events associated with the Table control:

1. Select the Table control.

2. Display the popup menu (right mouse click) and select "Code".

3. The "Initializing TABLE_QRY_FindOrders" event contains the following code:

```
// Parameters of 'QRY_FindOrders' query
//MySource.ParamStatus = <Value of ParamStatus>
MySource.ParamPaymentModeID = "1"
MySource.ParamStartOfPeriod = "20160101"
MySource.ParamEndOfPeriod = "20160331"
```

The test parameters have been retrieved as default parameters for the execution. We are now going to modify the window in order for the parameters to be typed by the user, via controls.

4. Close the code editor (click the cross in the top right corner of editor).

We are now going to create in our window the different controls allowing the user to select the query parameters. These controls will be positioned above the Table control.

► Move (if necessary) the Table control in the window and reduce its height in order to get available space for creating the different controls.

First parameter: Order status

Three states can be assigned to an order:

- pending,
- paid,
- canceled.

In our analysis, the order status is saved in the "Status" item found in the "Orders" file. This item is a radio button.

To allow the user to select one of these three states, we are going to use the Radio Button control associated with the "Status" item of "Orders" data file.

Remark

The radio buttons are also called "option box". They are used to select a single option among the proposed ones.

How to differentiate between a radio button and a check box?

We will only refer to option boxes as "Radio buttons". An easy way to remember: think of old radios: a single frequency could be selected via the button!

The radio button is used to select a single option.

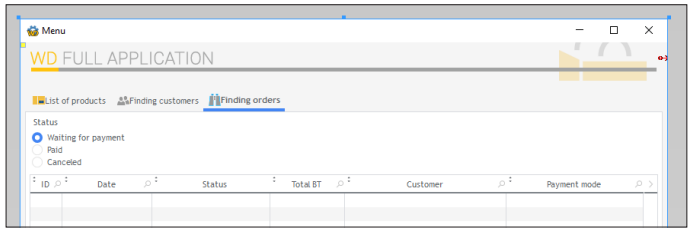
► To create the Radio Button control:

1. Display the "Analysis" pane if necessary: on the "Home" pane, in the "Environment" group, expand "Panes" and select "Analysis". The different data files described in the "WD Full Application" analysis appear in the pane.

2. Click the icon next to the "Orders" data file: the items found in the data file are listed.

3. Select the "Status" item in the Orders data file, then drag and drop this item into the "WIN_Menu" window

4. The Radio Button control is automatically created. Position this control above the Table control.



▶ We are now going to pass the value selected in the Radio Button control in parameter to the query:

1. Display the WLanguage events associated with the Table control:
   - Select the Table control.
   - Display the popup menu (right mouse click) and select "Code".
2. In the "Initializing" event of the Table control, replace the line:

```
//MySource.ParamStatus = <Value of ParamStatus>
```

with the following WLanguage code:

```
MySource.ParamStatus = RADIO_Status
```

In this code, RADIO_Status is the name of the Radio Button control that was just created. The value of this control is associated with the ParamStatus parameter expected by the query.

3. Close the code editor.

▶ Before running the test, we are going to create a Button control to re-display the content of Table control according to the value selected in the Radio Button control:

1. Create a Button control:
   - on the "Creation" pane, in the "Usual controls" group, click Ok.
   - then click on the "Finding orders" pane of the Tab control, at the top right.
2. Select the control and press Enter. The control caption becomes editable. Type "Find" and press Enter on the keyboard.
3. Modify the control style:
   - Display the popup menu of control (right mouse click) and select "Choose a style".
   - In the window that is displayed, press Ctrl + F. In the search control, type "BTN_Search".
   - The style is automatically selected. Validate.
4. Resize the control if necessary.
5. Display the WLanguage events associated with this control: press F2.
6. Write the following WLanguage code in the "Click..." event:

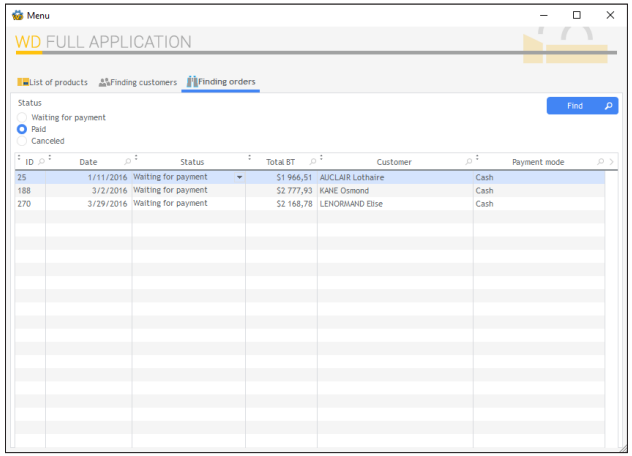
```
// Refreshes the display of Table control
TableDisplay(TABLE_QRY_FindOrders, taInit)
```

In this code, the ***taInit*** constant is used to re-run the "Initializing" event of the Table control (the event in which the parameters are passed to the query).

7. Close the code editor.

▶ We are now going to check how the first parameter is passed:

1. Save the window by clicking among the quick access buttons.
2. Click among the quick access buttons.
3. Select the "Finding orders" tab pane if necessary.
4. Change the status of orders with the radio button and click "Find". The content of Table control is modified.



5. Close the test window.

Second parameter: Payment mode

Several payment modes can be used for an order: cash, checks, ... The available payment modes are stored in the "PaymentMode" data file.

We will be using a Combo Box control based on this data file to allow the user to select the requested payment mode.

Remark

The "Combo Box" control is used to display a list of elements and to select an element from this list.

Unlike a List Box control , a Combo Box control is not expanded: the Combo Box control expands on request or when the cursor is positioned on the control input area.

The elements displayed in a Combo Box control can be defined when creating the control in the editor. These elements:

- are defined by programming.
- come from a data file or from a query.

▸ To create a Combo Box control:

1. On the "Creation" pane, in the "Usual controls" group, click "Combo Box".
2. Click the location where the control must be created in the window (beside the Radio Button control that was just created for example).
3. The wizard for creating the Combo Box control starts.
4. Select "Display the data found in a file or in an existing query" and go to the next step.
5. Select the "PaymentMode" data file and go to the next step.
6. The item that will be displayed in the Combo Box control is "Caption". Uncheck "PaymentModeID" and check "Caption". Go to the next step.
7. Select the sort item: "Caption". Go to the next step.
8. Select the return value: "PaymentModeID". This return value is very important because it will be passed in parameter to the query. Go to the next step.
9. Keep the default options in the different wizard steps and validate the creation of Combo Box control.
10. The Combo Box control is automatically created in the window.

▸ We are going to modify some characteristics of Combo Box control:

1. Select the Combo Box control and display the description window of control ("Description" from the popup menu).
2. In the "General" tab, modify the control caption: replace "PaymentMode combobox" by "Payment mode".
3. In the "Content" tab, specify the initial value displayed by the Combo Box control ("Initial value" at the bottom of description window). In our case, type "1". This value corresponds to a payment in cash.
4. Validate the control description window.

▸ Change the control style: to occupy less space, we are going to select a style that displays the caption above the control.

1. Select the Combo Box control.
2. In the popup menu (right mouse click), select "Choose a style".
3. In the window that is displayed, select the "COMBO_Internal" style and validate.
4. Reduce the control size.

▸ We are now going to use the value selected in the Combo Box control to pass it to the query as parameter:

1. Display the WLanguage events associated with the Table control:
   - Select the Table control.
   - Display the popup menu (right mouse click) and select "Code".
2. In the "Initializing" event of the Table control, replace the line:

```
MySource.ParamPaymentModeID = "1"
```

by the code:

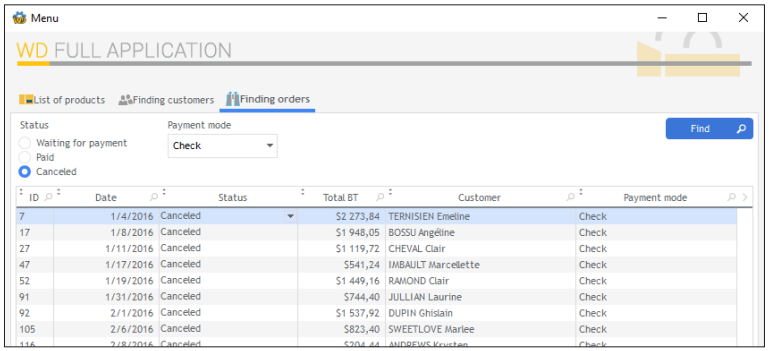
```
MySource.ParamPaymentModeID = COMBO_PaymentMode
```


In this code, COMBO_PaymentMode is the name of the Combo Box control that was just created. The value of this control is associated with the ParamPaymentModeID parameter expected by the query.

3. Close the code editor.
4. Save the window by clicking among the quick access buttons.

▸ We are now going to check how the first two parameters are passed:

1. Click among the quick access buttons.
2. Select the "Finding orders" tab pane if necessary.
3. Change the order status with the Radio Button control and modify the payment mode with the Combo Box control, then click "Find". The content of Table control is modified.



4. Close the test window.

Last parameter: Order date

The last query parameter corresponds to the date of orders taken into account. The user must be able to type a date interval. To do so, we are going to use a control template.

Remark

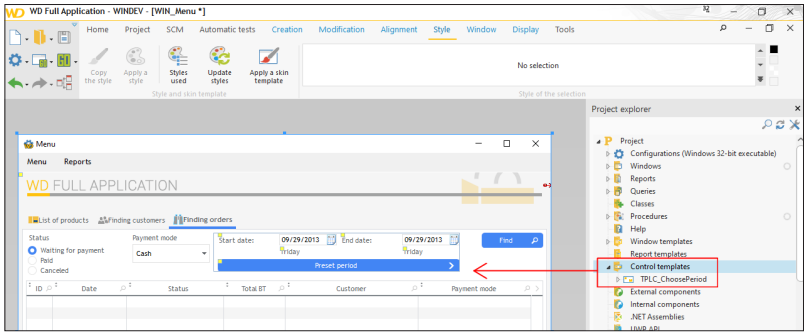
A control template is a specific window containing different controls. All types of controls can be included in this window. A control template is a file whose extension is "WDT".

The main benefit of a control template is the re-usability. A control template found in a project can be re-used in any project window.

Furthermore, the control templates can be overloaded: code can be added, the controls can be moved in the window that is using the control template. The controls can also be modified.

► To manage the order date:

1. In the "Project explorer" pane, expand the "Control templates" folder.
2. Drag the control template named "TPLC_ChoosePeriod" from the "Project explorer" pane and drop it in the "WIN_Menu" window (beside the "Payment mode" control).



3. Select the created control template and display its description ("Description" from the popup menu).
4. In the description window, rename the control template. The new name is "CTPL_ChoosePeriod".
5. Validate the description window.
6. Reposition and align the controls if necessary.

► We are now going to use the selected dates to pass them to the query as parameter:

1. Display the WLanguage events associated with the Table control:
   - Select the Table control.
   - Display the popup menu (right mouse click) and select "Code".
2. In the "Initializing" event of the Table control, replace the lines:

```
MySource.ParamStartOfPeriod = "20160101"
MySource.ParamEndOfPeriod = "20160331"
```

by:

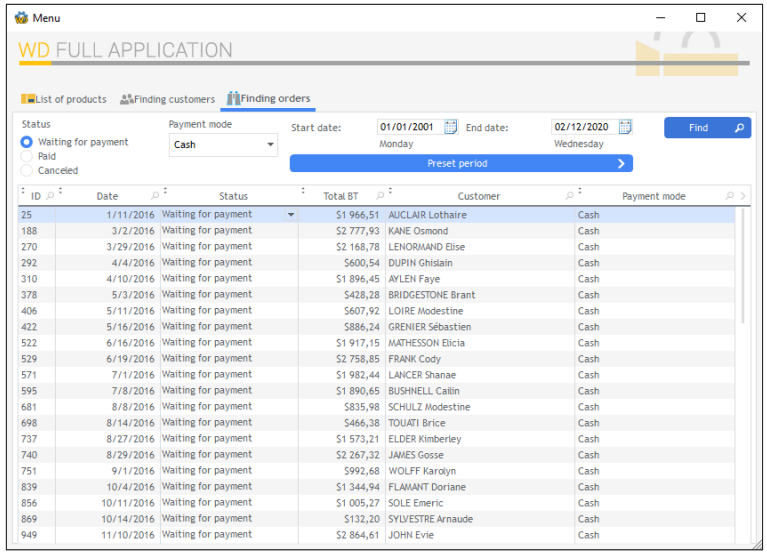
```
MySource.ParamStartOfPeriod = EDT_StartDate
MySource.ParamEndOfPeriod = EDT_EndDate
```

In this WLanguage code, EDT_StartDate and EDT_EndDate are the names of two edit controls found in the control template. Their values are associated with the parameters ParamStartOfPeriod and ParamEndOfPeriod expected by the query.

3. Close the code editor.
4. Save the window by clicking among the quick access buttons.

► We are now going to check how the parameters are passed:

1. Click among the quick access buttons.
2. Select the "Finding orders" tab if necessary.
3. Define the different search parameters:
   - Status of orders,
   - Payment mode,
   - Range of dates to take into account.
4. Click on the "Find" control. The content of Table control is modified.



ID	Date	Status	Total BT	Customer	Payment mode
25	1/11/2016	Waiting for payment	$1 966,51	AUCLAIR Lothaire	Cash
188	3/2/2016	Waiting for payment	$2 777,93	KANE Osmond	Cash
270	3/29/2016	Waiting for payment	$2 168,78	LENORMAND Elise	Cash
292	4/4/2016	Waiting for payment	$600,54	DUPIN Ghislain	Cash
310	4/10/2016	Waiting for payment	$1 896,45	AYLEN Faye	Cash
378	5/3/2016	Waiting for payment	$428,28	BRIDGESTONE Brant	Cash
406	5/11/2016	Waiting for payment	$607,92	LOIRE Modestine	Cash
422	5/16/2016	Waiting for payment	$886,24	GRENIER Sébastien	Cash
522	6/16/2016	Waiting for payment	$1 917,15	MATHESSON Elicia	Cash
529	6/19/2016	Waiting for payment	$2 758,85	FRANK Cody	Cash
571	7/1/2016	Waiting for payment	$1 982,44	LANCER Shanae	Cash
595	7/8/2016	Waiting for payment	$1 890,65	BUSHNELL Caitlin	Cash
681	8/8/2016	Waiting for payment	$835,98	SCHULZ Modestine	Cash
698	8/14/2016	Waiting for payment	$466,38	TOUATI Brice	Cash
737	8/27/2016	Waiting for payment	$1 573,21	ELDER Kimberley	Cash
740	8/29/2016	Waiting for payment	$2 267,32	JAMES Gosse	Cash
751	9/1/2016	Waiting for payment	$992,68	WOLFF Karolyn	Cash
839	10/4/2016	Waiting for payment	$1 344,94	FLAMANT Doriane	Cash
856	10/11/2016	Waiting for payment	$1 005,27	SOLE Emeric	Cash
869	10/14/2016	Waiting for payment	$132,20	SYLVESTRE Arnaude	Cash
949	11/10/2016	Waiting for payment	$2 864,61	JOHN Evie	Cash

5. Close the test window.

Lesson 4.5. Printing the content of a Table control

This lesson will teach you the following concepts

- Printing the content of a Table control.
- Printing an order form.

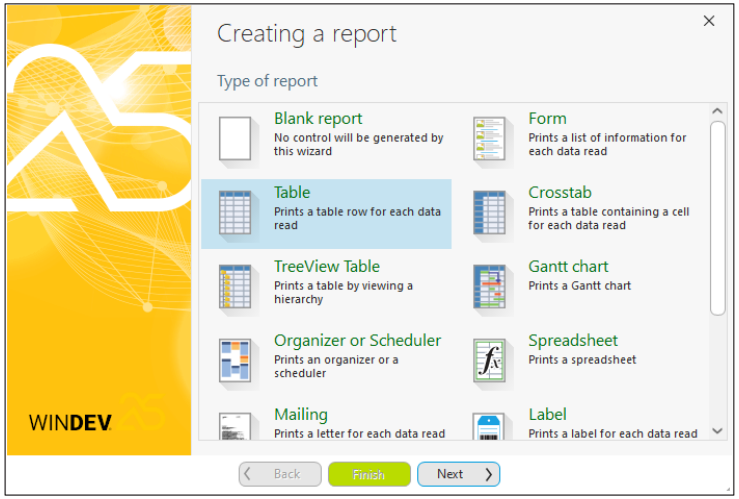
Estimated time: 20 mn

Overview

WINDEV allows you to print your data:

- by programming.
- via the reports. A report is the name given to the graphic representation of a print. A report is created via the report editor.

WINDEV proposes several types of reports:



- Blank,
- Form,
- Table,
- Crosstab,
- TreeView table,
- Gantt chart.
- Organizer or scheduler,
- Spreadsheet,
- Mailing,
- Label,
- Report on form,
- Multicolumn report,
- Composite.

We are now going to explain how to create the different types of reports in the "WD Full Application" application.

Answer

If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (With windows)".

A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)". In this case, to follow the rest of the lesson, select the "Windows 32-bit executable" configuration in the "Project explorer" pane.

Printing the content of a Table control

In the previous lesson, we have created a Table control used to display the result of a multi-criteria search. Let's take a look at the different methods used to print the content of this Table control.

Direct print via the AAF (Automatic Application Feature)

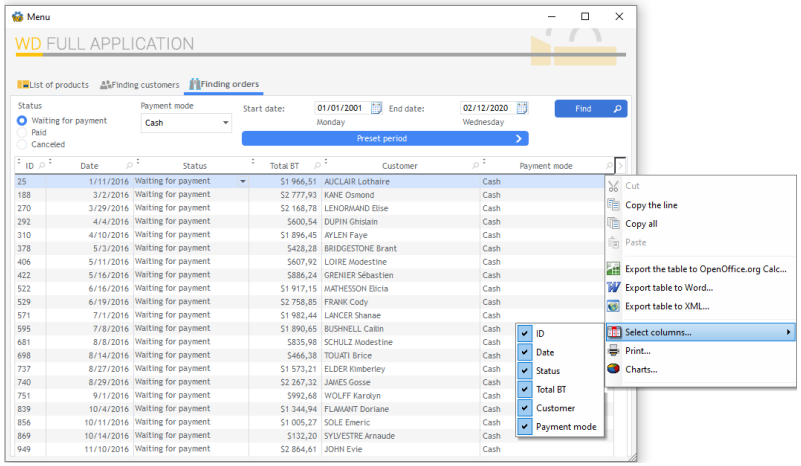
In the previous lessons, you have noticed that several automatic features (called AAF) were proposed by WINDEV. An AAF allows you to print the content of a Table control.

Remark

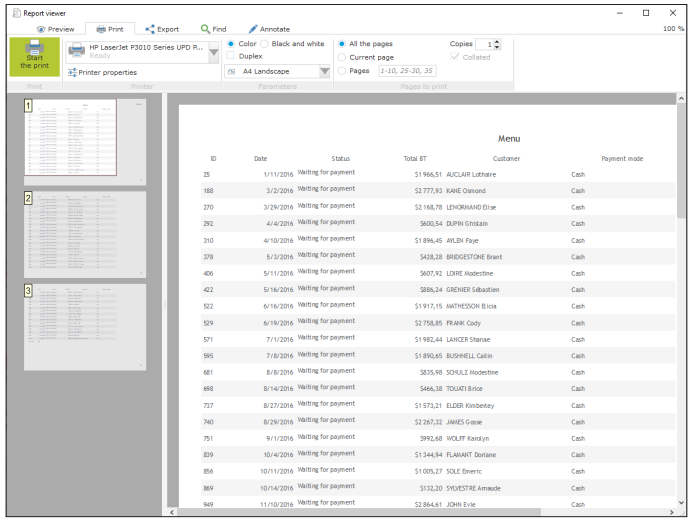
The entire list of AAFs (Automatic Application Features) is available in the WINDEV AAF 25.PDF file. This file is automatically included in your application if you choose the "Help for AAF" option when creating the automatic menu.

▶ Let's run a quick test:

1. Display "WIN_Menu" in the window editor (if necessary).
2. Click GO among the quick access buttons.
3. Select the "Finding orders" tab pane if necessary.
4. Define the different search parameters:
   - Status of orders,
   - Payment mode,
   - Range of dates to take into account.
5. Click "Find".
6. When the data is displayed in the Table control, display the popup menu of control (icon at the top right or right click on control).



7. Click "Print".
8. Choose (if necessary) to print the content of Table control in "Landscape" mode. The report corresponding to the control is displayed in the report viewer.



Remark

In test mode, the content of the Table control is printed directly.

At run time, the end user will be able to print directly or to start "Reports and Queries" in order to create the corresponding report. See "Distributing "Reports & Queries" with your applications" for more details about Reports and Queries.

9. Close the report viewer and stop the application test.

You want to customize the report proposed by default? All you have to do is create an automatic report on Table control. This report (if it exists) will be automatically used by the option for printing the Table control.

Creating an automatic report on Table control

To create a report used to print the data found in a Table control, all you have to do is create a "Report on Table control": the report corresponding to the Table control is automatically created.

▸ To create a "Report on Table control":

1. If necessary, display the "Finding orders" tab pane (click it) in the window editor.

2. On the "Creation" pane, in the "Data" group, expand "Table and List Box" and select "Report on Table control".

3. Our window containing 2 Table controls on different tab panes, WINDEV asks you to select the requested Table control. In our case, select "TABLE_QRY_FindOrders" and validate.

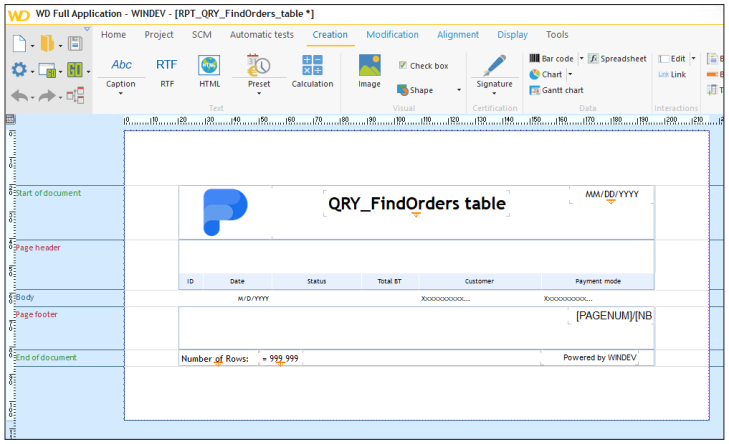
4. The shape of a Button control appears under the mouse cursor. This control is automatically created and it contains the WLanguage code used to print the content of the Table control.

5. Click the position where the control will be created in the window (below the "Find" Button control for example in the "Finding orders" tab pane).

6. The report is automatically created.

7. A window is displayed, allowing you to save the report. Validate.

8. The report editor appears with the report that was automatically created.



9. In this report, you will find the titles of the Table control columns in the window. We are going to perform a modification in this report: we want to modify the title.

- Select the control corresponding to the report title ("QRY_FindOrders Table" is displayed in the control) and double-click this control.
- In the description window, modify the caption and enter "Sought orders".
- Validate.

10. Go back to the "WIN_Menu" window (click the "WIN_Menu" button in the bar of open elements).

11. Modify (if necessary) the size and position of the "Print" Button control (to match the size of the "Find" Button control).

12. Save the window by clicking among the quick access buttons.

13. Run the window test: click among the quick access buttons.

- Select the "Finding orders" tab pane if necessary.
- Define the different search parameters:
  - Status of orders,
  - Payment mode,
  - Range of dates to take into account.
- Click "Find".

14. When the data is displayed in the Table control, click the the "Print" button or select "Print" from the popup menu of the Table control: the report that was just created is used in both cases.

Lesson 4.6. Printing an order

This lesson will teach you the following concepts

- Creating a report based on a query.
- Printing a report based on a query with parameters.

Estimated time: 30 mn

Overview

We want to give the user the ability to print the details of requested order. The report can be directly printed via the popup menu of Table control.

Answer

If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (With windows)".

A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)". In this case, to follow the rest of the lesson, select the "Windows 32-bit executable" configuration in the "Project explorer" pane.

Creating the "Order form" report

First of all, we are going to list the information that will be displayed in the report:

- The order characteristics: date and order number.
- The personal customer details: name, address, zip code, city and country.
- The characteristics of order lines:
  - Ordered quantity,
  - Product reference,
  - Product caption,
  - Total BT,
  - Total IOT.

To easily create this report, the data to print will be grouped in a query. This query can be used by the report or by any other element of WINDEV project (Table control, Looper control, ...).

Remark

WINDEV proposes to create reports from several data sources: data files, queries, controls, text files, ...

In most cases, we advise you to group the data to print via a query and to create a report based on this query. To add an information into the report, all you have to do is add the corresponding item into the query.

The reports based on data files must be simple reports, which means reports used to display data coming from a single data file.

Creating the query

▸ The query editor will be used to create the base query of report.

1. Click among the quick access buttons. The window for creating a new element is displayed: click "Query". The query creation wizard starts.

2. Select the "Select" option.
Indeed, this query will be used to select the records that will be printed in the report. Go to the next step.

3. The query description window is displayed. To build the query, we are going to select the elements that will be displayed in the result.

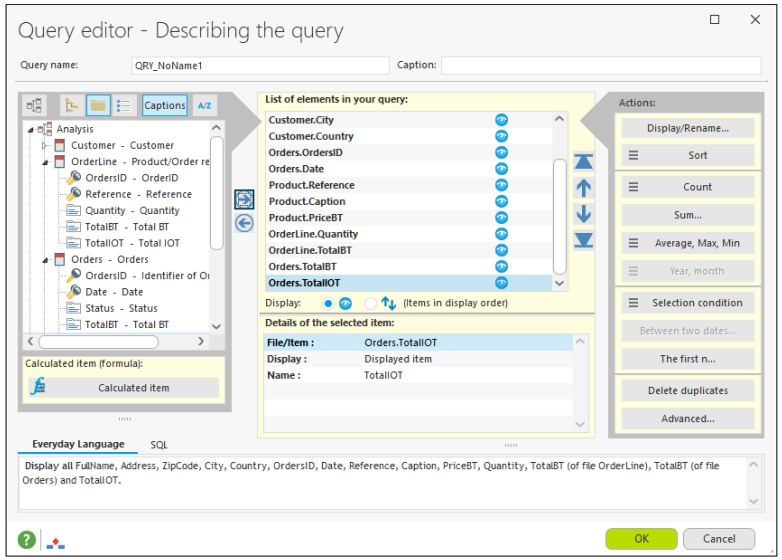
4. Double-click the items found in the analysis on the left of description window. The items taken into account are displayed in the middle of the screen.
We want to print in the report:

- information regarding the customer. In the "Customer" data file, double-click the FullName, Address, ZipCode, City, and Country items.
- information regarding the order. In the "Orders" data file, double-click the OrderID and Date items.
- information regarding the product. In the "Product" data file, double-click the Reference, Caption, and PriceBT items.
- information regarding the order line. In the "OrderLine" data file, double-click the Quantity and TotalBT items.
- information regarding the grand total of order. In the "Orders" data file, double-click the TotalBT and TotalIOT items.

Remark

The order in which the items are inserted into the query is important. Indeed, this order corresponds to the order used to display the data in the report. The creation of the corresponding report will be simplified if this order is properly defined.

The description window of query is as follows:



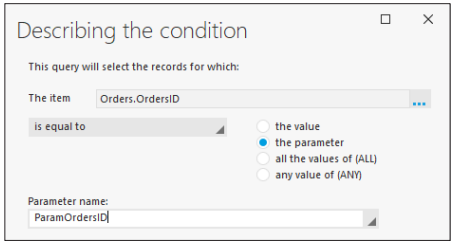
At this time, this query is used to select all orders and the corresponding order lines.
We want to select the data corresponding to a single order whose identifier is known. Therefore, we are going to define the order number in parameter.

▸ To manage the "Order identifier" parameter:

1. Select the Orders.OrdersID item (in the middle of the screen).

2. Expand "Selection condition" and select "New condition".

3. In the window that is displayed, we are going to specify that the selection condition corresponds to a parameter:

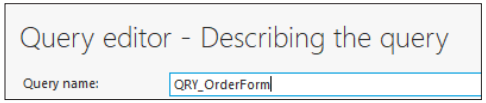


Perform the following operations:

- Select "Is equal to".
- Check "the parameter".
- The name of the parameter is automatically proposed: "ParamOrdersID".

4. Validate the condition description window. The number "1" is displayed on the right of Orders. OrdersID item, indicating that a selection condition was defined.

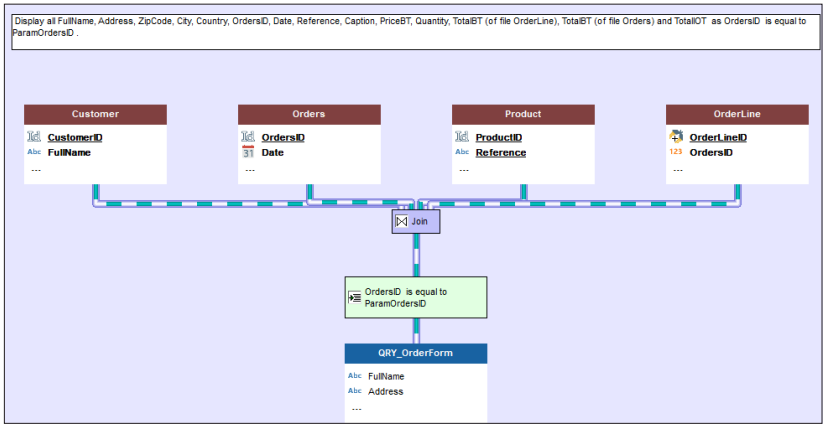
5. Give a name to the query: type "QRY_OrderForm" instead of "QRY_NoName1" in the "Query name" area:



6. Validate the query description window ("OK" button).

7. The save window is displayed. Validate the proposed information.

8. The graphic query representation is displayed:



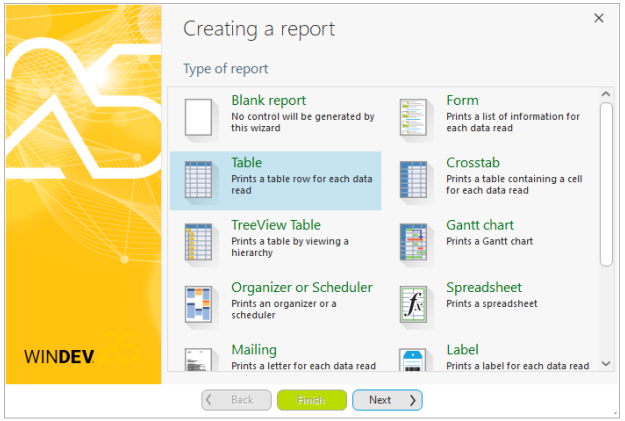
Creating the report based on a query

▸ To create a report:

1. Click  among the quick access buttons.

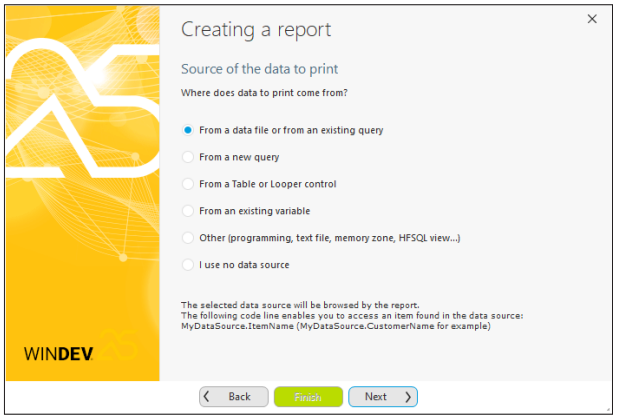
2. The window for creating a new element is displayed: click "Report" then "Report". The report creation wizard starts.

3. The report creation wizard proposes several types of reports:



4. Select "Table". Go to the next step.

5. Select the data source of report. The report will be based on the query that was just created. Select "From a data file or from an existing query".



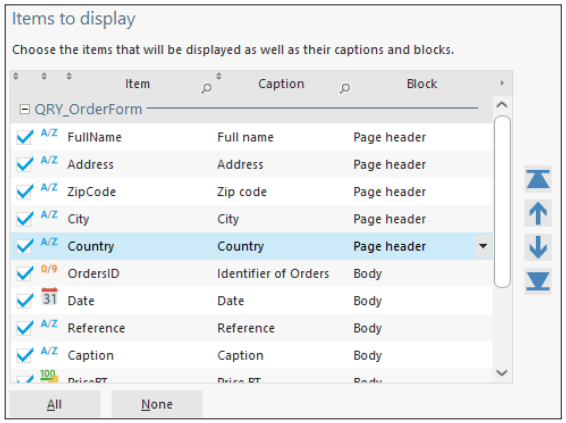
Go to the next step.

6. In the list of data files and queries, select the "QRY_OrderForm" query. Go to the next step.

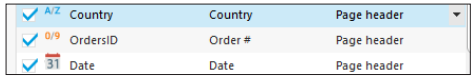
7. The wizard asks you to specify whether a break is required. No break will be used in this report. This concept will be presented later in this tutorial. Answer "No". Go to the next step.

8. You are going to specify the order in which the items will be printed and how they will be distributed in the different blocks. In the wizard, the items are listed according to the order defined in the query:

- The items regarding the customer will be displayed in the "Page header" block. Indeed, this information must not be repeated on each order line.
  For the FullName, Address, ZipCode, City and Country items, click the line corresponding to the item. In the "Block" column, expand the combo box and select "Page header".

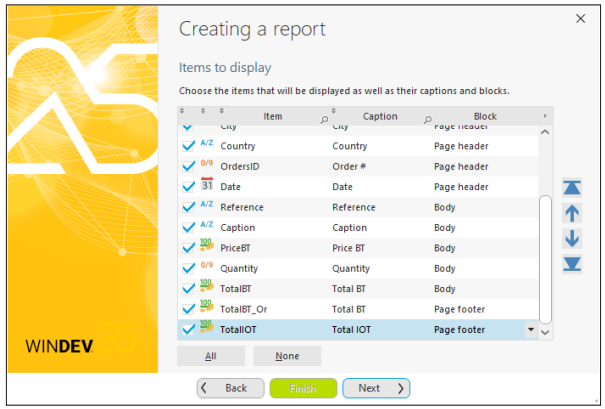


- The items regarding the order will also be displayed in the "Page header" block. Indeed, this information must not be repeated on each order line. For the OrdersID and Date items, click the line corresponding to the item. In the "Block" column, expand the combo box and select "Page header".
- Furthermore, we are going to modify the caption of "OrdersID" item.
  - Select the line containing the OrdersID item.
  - Click the Caption column and type "Order #".



  - The items regarding the order lines will be displayed in the report body. These items will be displayed for all the order lines.

- The items regarding the order totals will be displayed in the page footer. Indeed, this information must not be repeated on each order line. For the TotalBT_Or and TotalIOT items, click the line corresponding to the item. In the "Block" column, expand the combo box and select "Page footer".



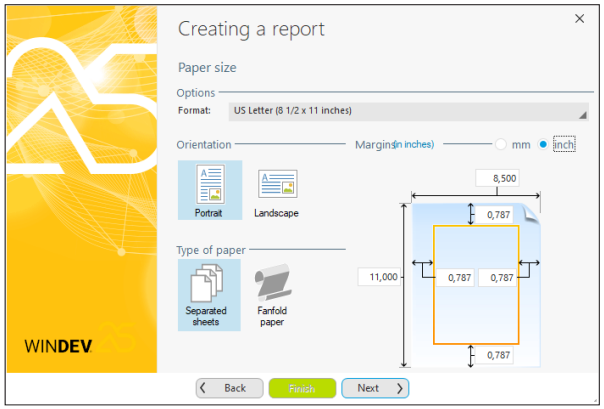
The following table presents the different assignments of items in the order presented in the wizard:

Item	Block
FullName	Page header
Address	Page header
ZipCode	Page header
City	Page header
Country	Page header
OrdersID	Page header
Date	Page header
Reference	Body
Caption	Body
PriceBT	Body
Quantity	Body
TotalBT	Body
TotalBT_Or	Page footer
TotalIOT	Page footer

9. Go to the next step.

10. The wizard proposes to create a counter, a sum or an average on the numeric items found in the report. In this report, the calculations are performed by the query. Click the "No calculation" button. Go to the next step.

11. This step is used to define the report layout.



We will keep the default values with the "Portrait" orientation.

Remark

Print margins

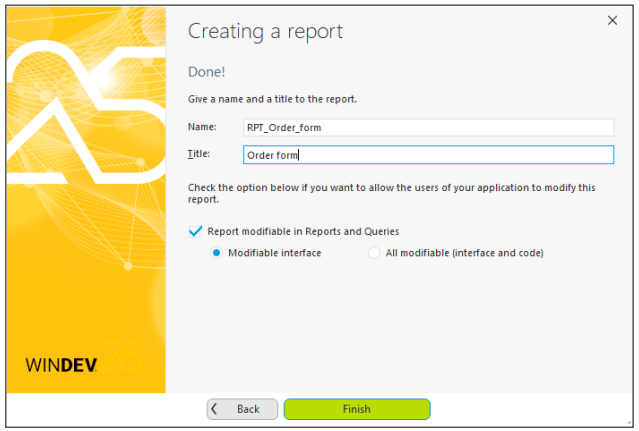
When choosing the print margins, don't forget to take into account the physical margins of printers. The physical margins of printers are margins where no print is allowed. Furthermore, the physical margins differ according to the type of printer.

12. Go to the next step.

13. This step allows you to select the skin template used for the report. We recommend that you use the same skin template as the one used for the windows. In our case, select the "Phoenix" skin template for example and go to the next step.

14. All we have to do now is give a name and a title to the report.

- Type the title : "Order form".
- The "RPT_Order_form" name is automatically proposed.



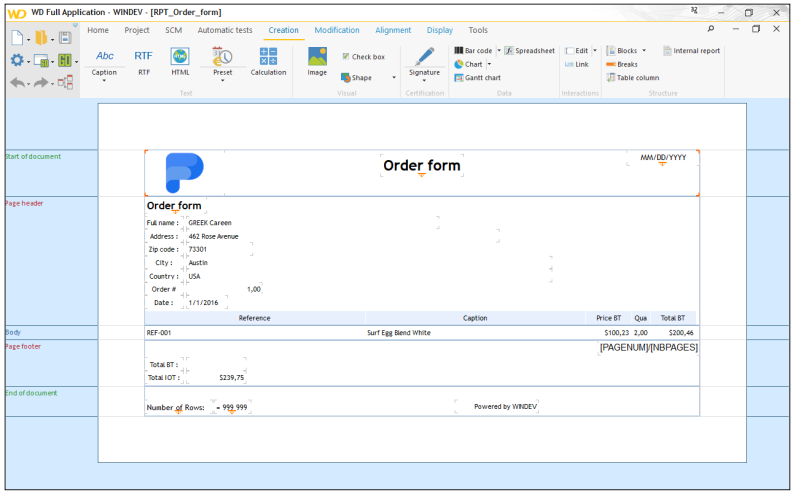
15. Validate ("Finish").

16. The report being too large to be printed in Portrait mode, the report editor proposes to use the landscape mode, to reduce the table or to print the table on several pages. In our case, accept the landscape mode.

17. Accept to reduce the table if necessary.

18. The window for saving the report is displayed. Validate the save information.

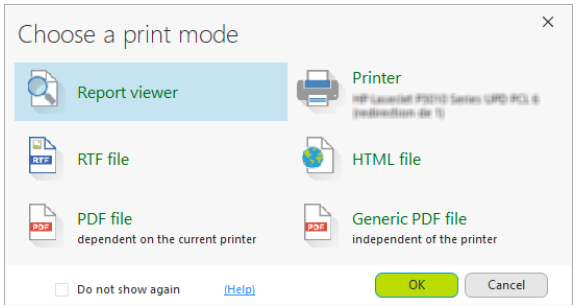
19. The report is displayed in edit in the report editor:



20. The different order lines are grouped in a table.

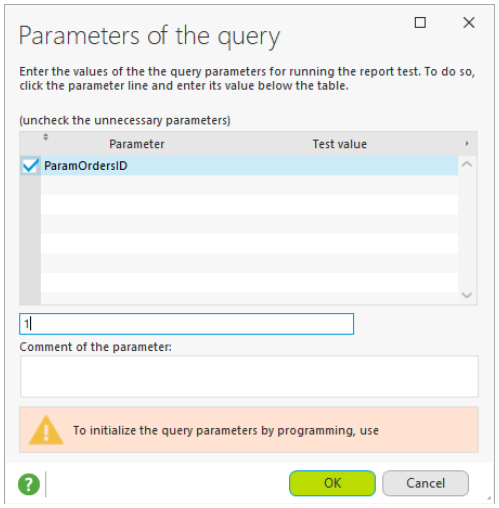
► Run this report by clicking among the quick access buttons.

1. The report editor asks for the print destination. The print destination can be:



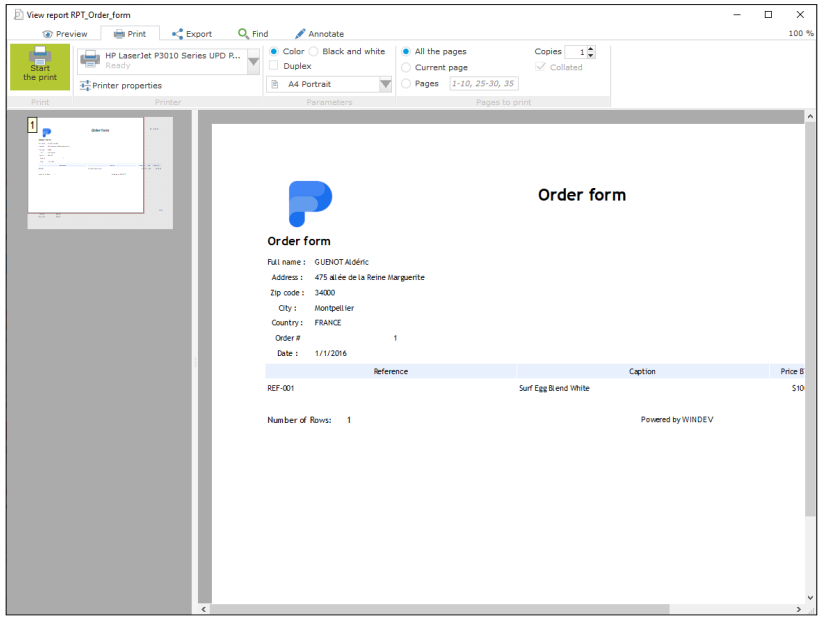
Select "Report viewer" and validate.

2. The report editor asks for the parameters of query used by the report. Don't forget that a parameter was used to specify the number of the order to print. For the example, type the test value "1".



Validate.

3. The report is displayed as requested in the report viewer.



You can:

- Print the current page or the entire document by clicking the printer.
- Create a duplicate copy.
- Select a different zoom value.
- Save the report as a Word document (in RTF format).
- Save the report in HTML format.
- Save the report in PDF format.
- Save the report in XML format.
- Create an email with the report in HTML format in message body.
- Create an email with the report in PDF format in attachment.
- Annotate in the document.
- Perform a search in a document.
- Add watermarks.

► Close the report viewer.

Modifying the "Order form" report

We are going to perform modifications regarding the layout in the report that was just created.

▸ Delete the number of pages displayed in the report:

1. Select the [PAGENUM]/[NBPAGES] control.
2. Press the Del key on the keyboard.

▸ We want to position the information regarding the customer and the order in the page header:

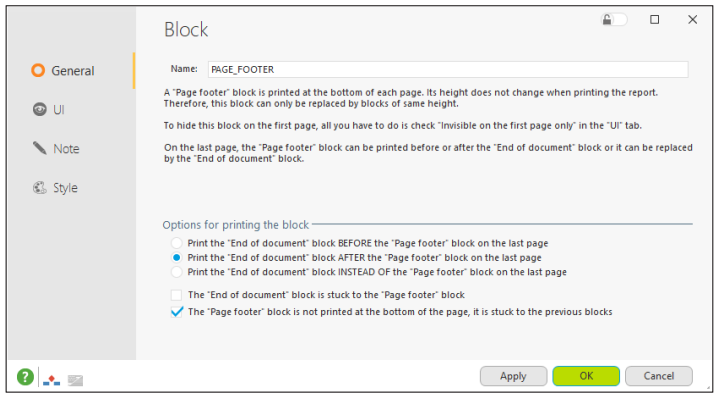
1. Delete the "Order form" caption found in page header.
2. Delete the captions found in front of the customer information (Full name, ...).
3. Position the item containing the city beside the zip code.
4. Select the customer details and move them (with the mouse) to the right of report.
5. Move the order number and the order date up (to the top of "Page header" block).

▸ We are going to position the totals properly in the page footer:

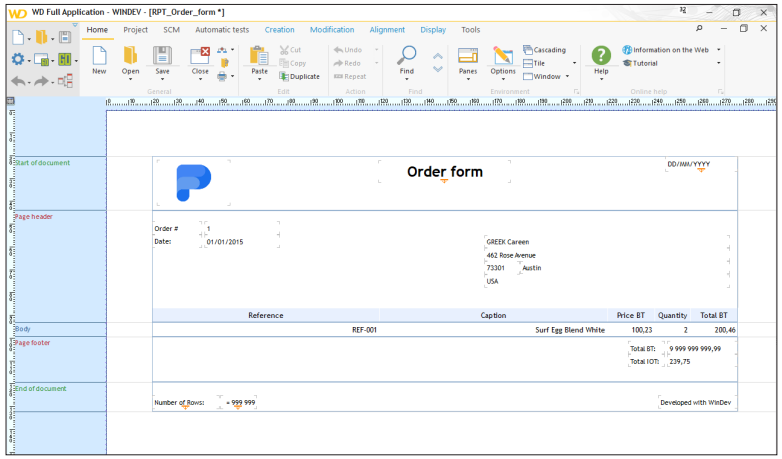
1. Select the controls (captions and items) corresponding to the totals found in the page footer.
2. Position these controls with the mouse in the bottom right corner of table.

▸ Modify the print options of "Page footer" block:

1. Display the description window of "Page footer" block:
   - Click the "Page footer" block.
   - Display the popup menu (right mouse click) and select "Block description".
2. In the "General" tab, check the following options:
   - Print the "End of document" block AFTER the "Page footer" block on the last page.
   - The "Page footer" block is not printed at the bottom of the page, it is stuck to the previous blocks.



3. Validate the description window.
4. The report is displayed in the report editor:



5. Save the report by clicking among the quick access buttons.

Our report is created.

Displaying the printed report from a menu option

In our application, the "RPT_Order_form" report will be printed from an option found in the popup menu of the Table control used to list the requested orders.

As already seen, the Table control proposes an automatic menu. We want to keep the options of this automatic menu and add an option that will be used to print the selected order form.

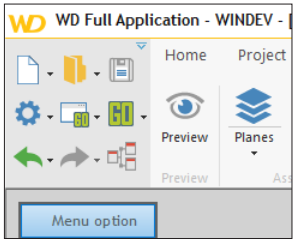
The principle is as follows:

1. We are going to create a new popup menu. This menu will contain the WLanguage code used to print the report.
2. We are going to link this popup menu to the Table control and specify that the default popup menu will be displayed just after the popup menu defined for the Table control.

Creating the popup menu

▸ To create a popup menu in the "WIN_Menu" window:

1. Open the "WIN_Menu" window in the editor if necessary (double-click its name in the "Project explorer" pane for example).
2. On the "Window" pane, in the "Bars and menus" group, expand "Popup menus" and select "New popup menu".
3. A new popup menu appears in the editor. This menu option includes a single option named "Menu option".



4. Display the description of popup menu:
   - Select "Menu option".
   - Display the popup menu (right mouse click).
   - Select "Description of popup menu".
5. Give a name to the popup menu ("MENU_Order") and validate.

Remark

Popup menus and windows

A popup menu is associated with a window. If a window includes several popup menus (popup menu of window, popup menus of controls, ...), all these menus are saved with the window.

▸ We are now going to define the option caption and write its WLanguage code.

1. Select "Menu option".
2. Press the Space key on the keyboard: the caption becomes editable. Type the new caption ("Print the order form") and validate.
3. Display the WLanguage events associated with the option:
   - Select the option.
   - Display the popup menu (right mouse click) and select "Code".
4. Write the following WLanguage code:

```
// Report viewer
iDestination(iViewer)
// Initializes the report query
iInitReportQuery(RPT_Order_form, ...
  TABLE_QRY_FindOrders.COL_OrdersID)
// Starts printing the report
iPrintReport(RPT_Order_form)
```

In this code:

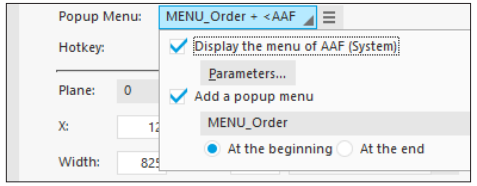
- ***iDestination*** is used to specify that the report will be printed in the report viewer.
- ***iInitReportQuery*** is used to specify the parameters expected by the query associated with the report. In our case, the query expects the order number in parameter. This order number is found in the COL_OrdersID column of the TABLE_QRY_FindOrders Table control for the current row.
- ***iPrintReport*** is used to print the specified report (the RPT_Order_form report in our case).

5. Close the code window and the window containing the popup menu.
6. Save the "WIN_Menu" window.

Associating the popup menu with the Table control

▸ Last step: we are going to link the popup menu to the Table control.

1. In the "WIN_Menu" window, display the "Finding orders" tab.
2. Select the Table control and display its description ("Table description" from the popup menu of control).
3. In the "UI" tab, expand the "Popup menu" combo box.
4. Check "Add a popup menu", select the "MENU_Order" menu and select "At the beginning" to specify that the menu is placed before the system menu.

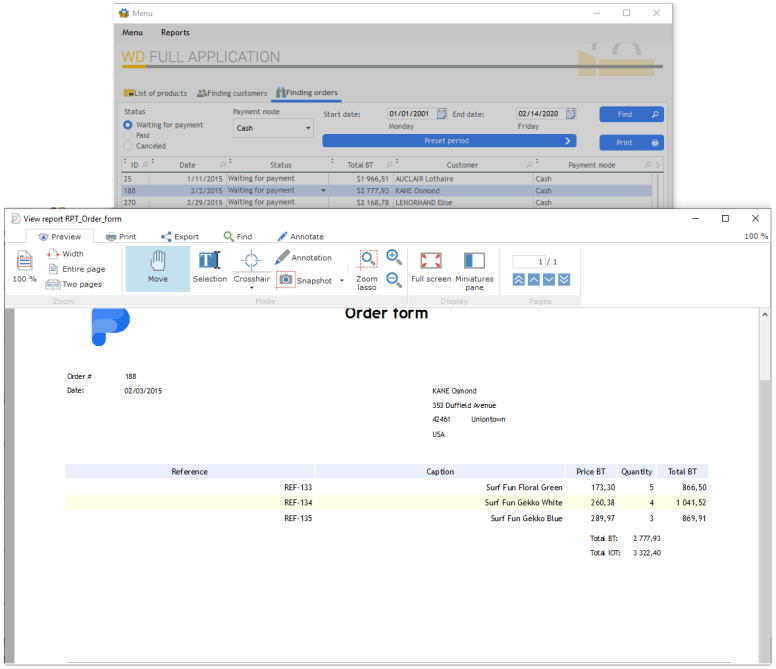


5. Click inside the window to validate the popup menu.
6. Validate.

Print test

▸ Now, all we have to do is run a real test:

1. Run the test of "WIN_Menu" window.
2. Select the "Finding orders" tab pane.
3. Specify the criteria and start a search.
4. Select one of the orders displayed in the Table control.
5. Print the order via the popup menu.



6. Close the report viewer and the test window.

Lesson 4.7. Printing a list of customers

This lesson will teach you the following concepts

- Creating a report with breaks.
- Starting the report print.

Estimated time: 20 mn

Overview

This lesson is used to print a list of customers, grouped by country and by state or province.
To do so, we are going to use a table report in order to clearly represent a list of data.

Answer

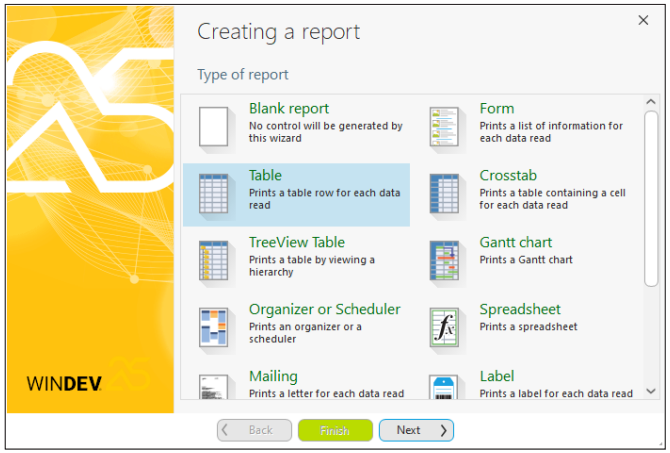
If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (With windows)".
A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

Creating the report

▸ To create a report:

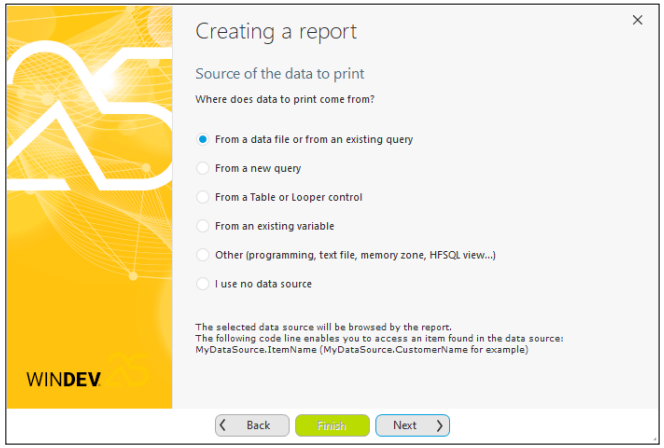
1. Click among the quick access buttons.

2. The window for creating a new element is displayed: click "Report" then "Report". The report creation wizard starts. The report creation wizard proposes several types of reports:



3. Select "Table". Go to the next step.

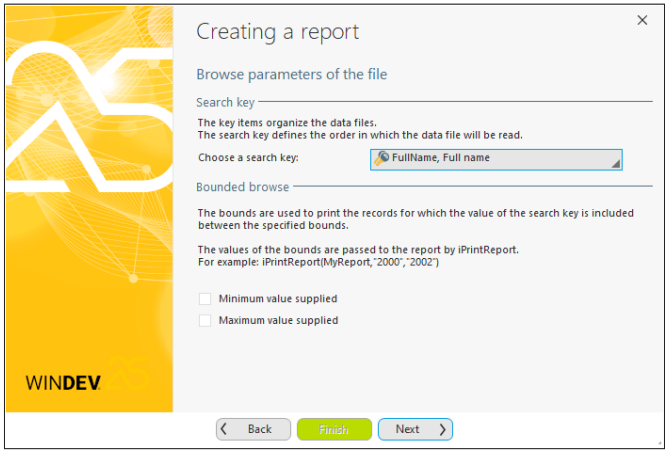
4. Select the data source of report. The report will be based on the Customer data file. Select "From a data file or from an existing query".



Go to the next step.

5. In the list of data files and queries, select the "Customer" data file. Go to the next step.

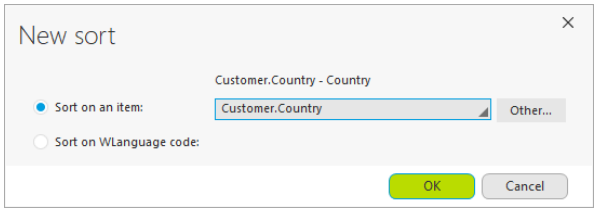
6. Define the data file search key. In our case, the Customer data file will be browsed using the customer's full name. Select the "FullName" search key.



Go to the next step.

7. We are now going to define the sort option of data in the report. By default, the wizard proposes the item used as search key in the data file. As we want to create a report used to list the customers by country and by state, we are going to add a sort on these two items.

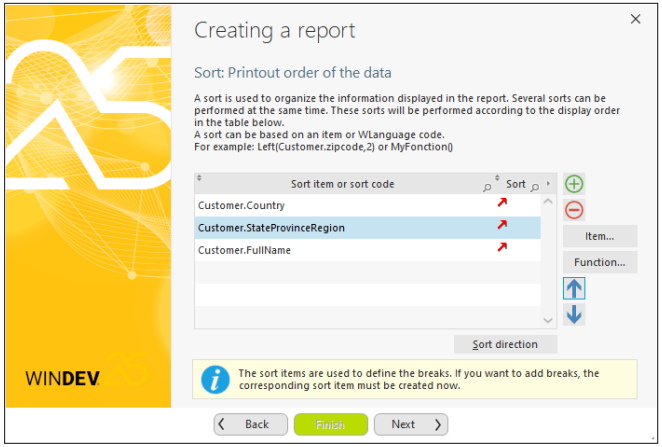
8. To add a sort on the country:
   - Click on "+" to add a sort item.
   - In the window that is displayed, select the Customer.Country item.



   - Validate.

9. To add a sort on the state:
   - Click on "+" to add a sort item.
   - In the window that is displayed, select the Customer.StateProvinceRegion item.
   - Validate.

10. Modify the order of sort items. The customers must be sorted by country, by region, then in alphabetical order.
   - The "Customer.Country" item appears in second sort item. Select this item and move it in first sort item via the arrow buttons.
   - The "Customer.StateProvinceRegion" item appears in third sort item. Select this item and move it in second sort item via the arrow buttons.



Go to the next step.

11. The wizard asks you to specify whether a break is required.

Remark

What is a break?

A **Break** is an operation used to group the records (or rows) according to one or more criteria. Caution, the records (or rows) will be printed.

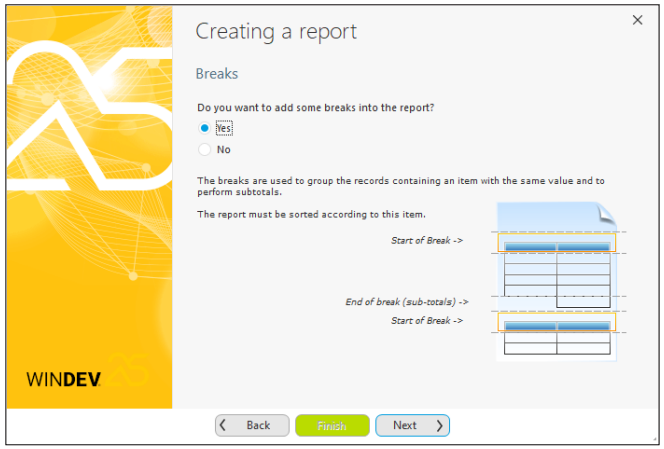
A break is NECESSARILY linked to a sort.

You will notice the presence of a break header and footer.

The information common to the different rows is found in the break header.

The totals, counters, ... are found in the break footer.

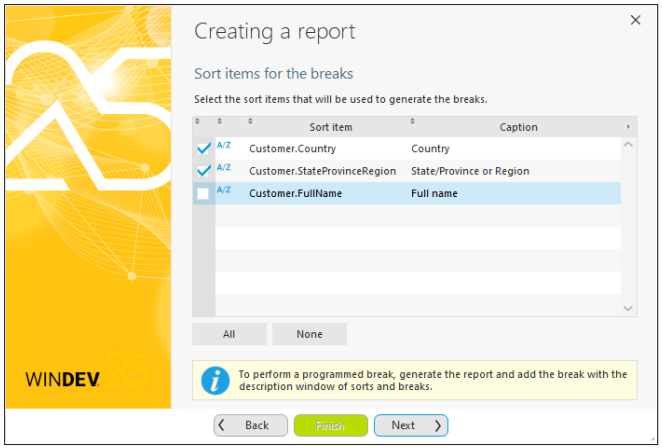
This break is used to group a set of records according to the same criterion. In our example, the break is performed on the country and it is used to group all the customers living in the same country.



12. Answer "Yes". Go to the next step.

13. Specify the sort item on which the break must be performed. In our case, we will be using several breaks:
   - the first break is performed on the country.
   - the second break is performed on the province.

Uncheck the "Customer.FullName" item.

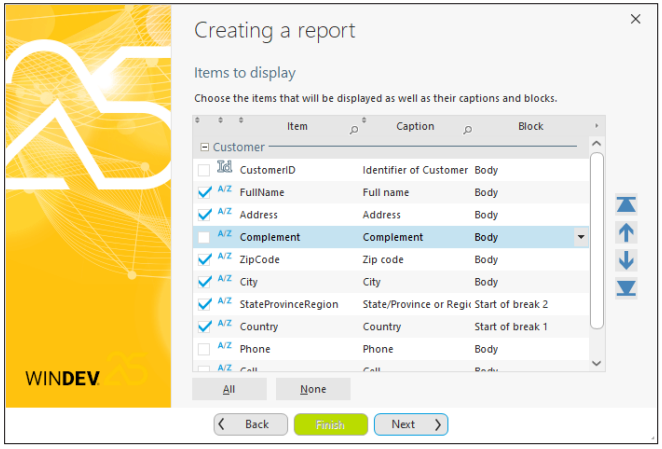


14. Go to the next step.

15. You are going to specify the order in which the items will be printed and how they will be distributed in the different blocks. In our example:

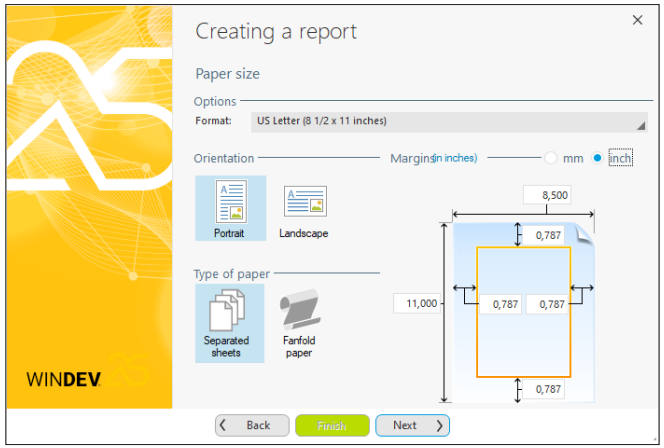
- only the country will be found in the "Start of break 1" block. The "Country" item is automatically associated with the "Start of break 1".
- only the state will be found in the "Start of break 2" block. The "StateProvinceRegion" item is automatically associated with the "Start of break 2".
- only the FullName, Address, ZipCode, City, StateProvinceRegion and Country items will be displayed in the report.

Uncheck the CustomerID, Complement, Phone, Cell and Email items.



Go to the next step.

16. This step is used to define the report layout.



We will keep the default values with the "Portrait" orientation.

Remark

Print margins

When choosing the print margins, don't forget to take into account the physical margins of printers. The physical margins of printers are margins where no print is allowed. Furthermore, the physical margins differ according to the type of printer.

17. Go to the next step.

18. This step allows you to select the skin template used for the report. We recommend that you use the same skin template as the one used for the windows. In our case, select the "Phoenix" skin template for example and go to the next step.

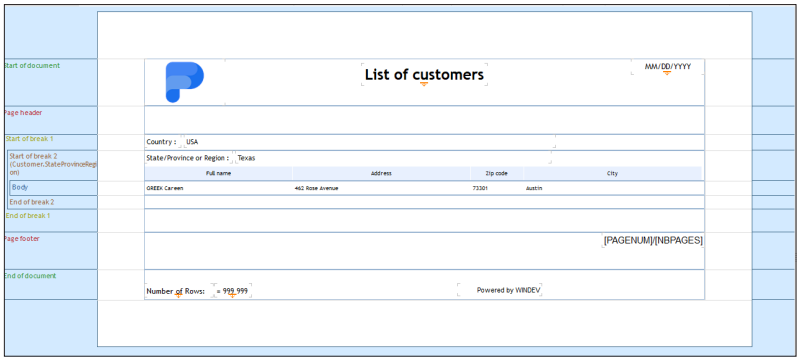
19. All we have to do now is give a name and caption to the report and save it.

- Type the title: "List of customers".
- Keep the name proposed by default: "RPT_List_of_customers".

20. Validate.

21. Validate in order to switch to landscape mode.

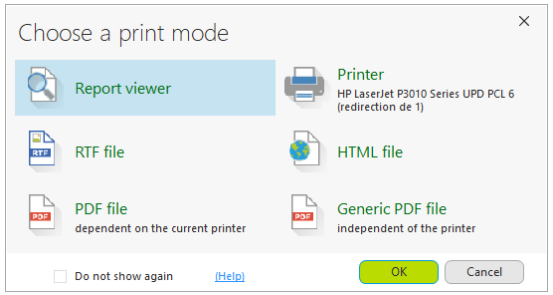
22. Accept to reduce the report by validating.



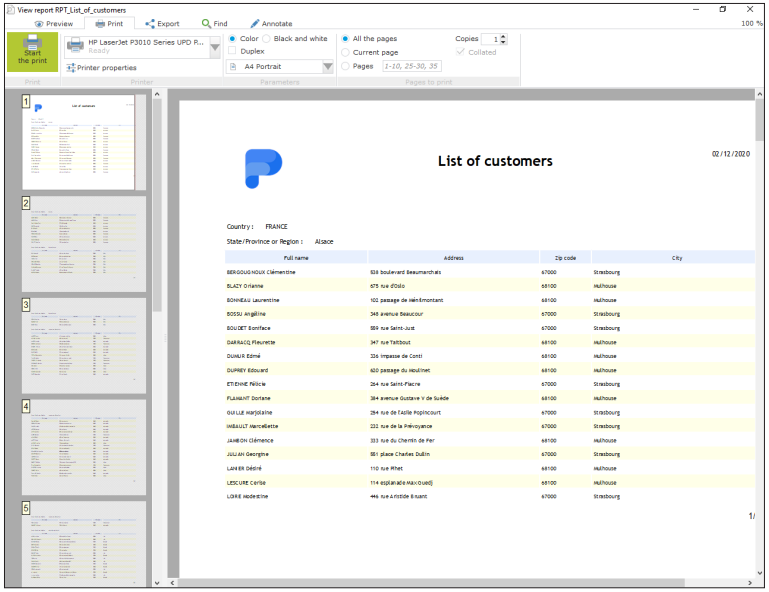
23. The report is displayed in the report editor. Save the report.

The report is completed. Let's now run the report test to see the result.

► Run this report by clicking among the quick access buttons. The print destination can be:



► Choose "Report viewer" and validate. The report is run and displayed in the report viewer.



You have the ability to print the current page or the entire document by clicking the printer ("Print" pane).

Starting the report print by programming

Our report being completed, we are going to create a menu in our "WIN_Menu" window to print this report directly.

► To insert a menu into the "WIN_Menu" window:

1. Display the "WIN_Menu" window in the editor (double-click its name in the "Explorer" pane for example).

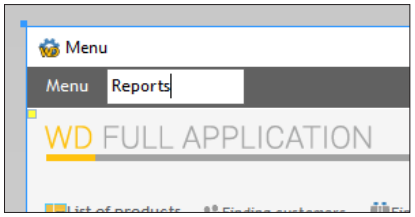
2. On the "Window" pane, in the "Bars and menus" group, expand "Main menu" and select "Add main menu".

3. A menu is inserted into the window, below the title bar. This menu contains an option named "Menu".

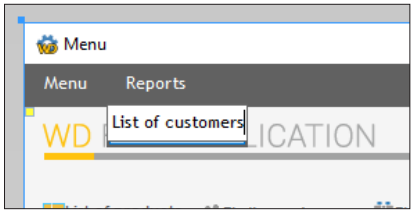
4. Select the "Menu" option:

- Display the popup menu (right mouse click).
- Select "Add after".

5. In the input area, type "Reports".



6. Select the "Reports" option:
   - Display the popup menu (right mouse click).
   - Select "Transform to expand a sub-menu".
7. In the input area that is displayed, type "List of customers". Press Enter to validate.



► To type the WLanguage code of the option "List of customers":

1. Select the "List of customers" option in the editor.
2. Display the popup menu (right mouse click).
3. Select "Code". The code editor appears.
4. Write the following code in the event "Selecting the menu":

```
// The report is printed in the report viewer
iDestination(iViewer)

// Prints the report
iPrintReport(RPT_List_of_customers)
```

Let's study this code:

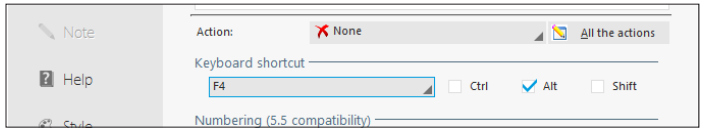
- ***iDestination*** allows you to configure the print destination. You have the ability to print:
  - in a text file,
  - in HTML format,
  - in PCL format,
  - in PDF, RTF, XLS or XML format,
  - on a fax.

  In our case, the report will be printed in the report viewer window.
- ***iPrintReport*** is used to print a report.

► Test the window and its menu options by clicking among the quick access buttons.

► We are now going to modify this menu by adding an option used to exit from the application.

1. In the editor, select the "Menu" option.
2. In the popup menu (right mouse click), select "Transform to expand a sub-menu".
3. In the input area that is displayed, type "Exit".
4. We are going to associate this menu option with the "Alt + F4" shortcut:
   - Select the "Exit" option.
   - In the popup menu (right mouse click), select "Option description".
   - In the "General" tab, in the "Keyboard shortcut" area, select "F4" and check "Alt".



   - Validate.
5. Display the WLanguage code of the option and write the following WLanguage code in the event "Selecting the menu":

```
// Asks the user whether he wants to exit from the application
IF YesNo(No, "Exit from the application?") = Yes THEN
  // End of application
  EndProgram()
END
```

Let's take a look at this WLanguage code:

- ***YesNo*** is used to establish a dialog with the user by asking him a question. The user can give an answer to the question via 2 buttons: yes or no.
- ***EndProgram*** (called if the user clicks "Yes") is used to end the application.

► Run the window test by clicking among the quick access buttons.

Lesson 4.8. Statistics: Chart and Pivot Table controls

This lesson will teach you the following concepts

- Displaying data in a chart.
- Creating summary tables via the Pivot Table control.

Estimated time: 30 mn

Overview

The presentation of statistics or summary tables is often required in a management application. This type of presentation can be used for example to follow:

- the evolution of orders in time,
- the evolution of turnover,
- the evolution of stocks,
- ...

Any executive manager wants to get this information.

WINDEV proposes several controls allowing you to easily include this information in your applications. Two specific controls will be used in this lesson:

- the Chart control.
- the Pivot Table control.

Answer

If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (with windows)".

A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

Displaying data in a Chart control

To handle the Chart control in real conditions, we are going to create a Chart control displaying the status of sales in the "WD Full Application" application.

First of all, we are going to create a query used to select the data that will be displayed in the Chart control.

Selecting the data that will be displayed in the Chart control

To create our chart, we want to get the sum of orders by date.

We are now going to create a query used to perform a sum. Indeed, we are going to calculate the total amount of orders (TotalIOT item in the Orders data file).

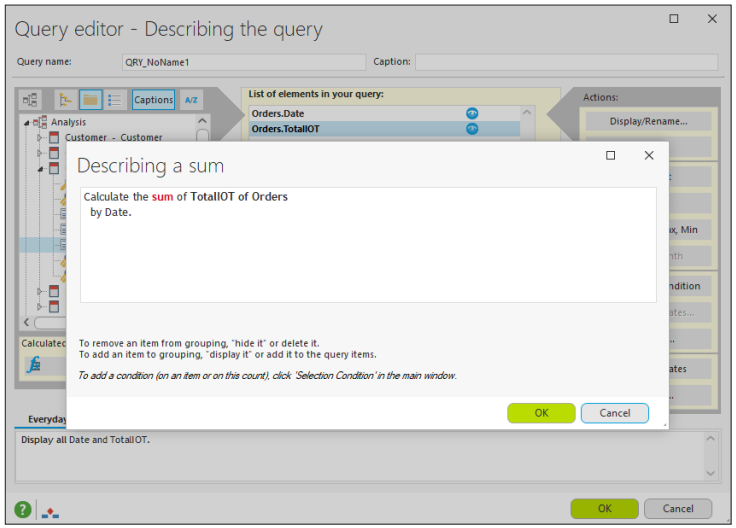
We are going to calculate the total amount of all orders per date (the Turnover per date).

► Create a new query:

1. Click  among the quick access buttons.
2. The window for creating a new element is displayed: click "Query".
3. We are going to create a select query. Select the "Select" option. Go to the next step.
4. The description window of query is displayed.
5. Add the items Orders.Date and Orders.TotalIOT to the query:
   - On the left, expand the "Orders" data file.
   - Double-click the Date item then the TotalIOT item.
   - The two items appear in the middle of the screen (in the "List of elements in your query" area).

► To calculate the sum of values of "Orders.TotalIOT":

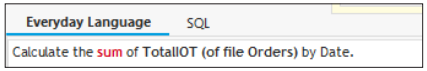
1. Select the "Orders.TotalIOT" item in the middle.
2. In the "Actions", on the right, select "Sum". The sum description window appears.



3. Validate the sum description. The sum of "TotalIOT" was added into the list of query result.

Remark

You will notice that the query editor of WINDEV creates the queries in everyday's language (and also in SQL language).

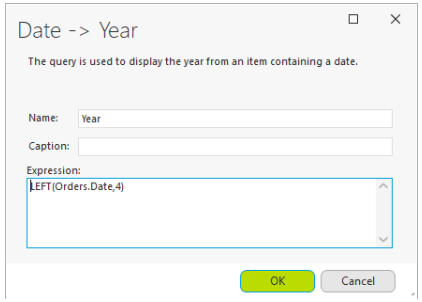


This allows you to check the purpose of your query.

► We are now going to group the data by year and to sort it:

1. Select the "Orders.Date" item in the middle.

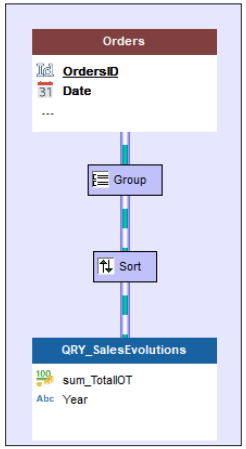
2. In the "Actions", on the right, click "Year, month" and select "Year". A window appears allowing you to create the Year item:



3. Validate this window ("OK").
4. The "Year" item is displayed in the middle of query description.
5. Select the "Year" item and define the sort:
   - Display the popup menu of "Year" item.
   - Select "Sort item .. Sort in ascending order".
   - An arrow indicating the sort appears in the query description.

► The query is created. We are going to give it a name and to save it.

1. At the top of the query description window, enter:
   - the name "QRY_SalesEvolutions".
   - the caption "Sum of orders by date".
2. Validate the description window of query.
3. The save window is displayed. Validate the proposed information.
4. The graphic query representation is as follows:



5. Click to run the query test.

Creating the Chart control

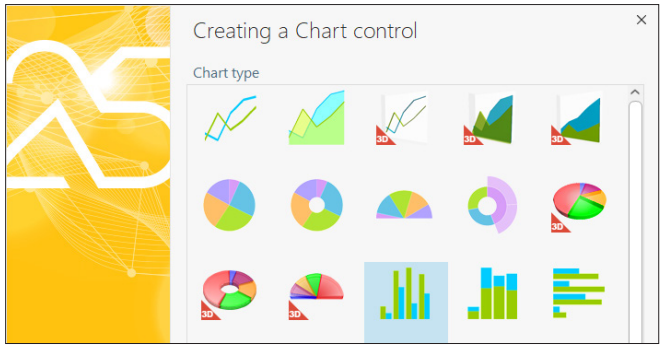
We are going to include the Chart control in a new tab pane of WIN_Menu.

▸ To create a new tab pane in "WIN_Menu":

1. Display "WIN_Menu" in the window editor (if necessary).
2. Double-click the Tab control: the control description window appears.
3. In the "General" tab, select the "Finding orders" tab pane and click the "New" button. A new tab pane appears.
4. Select the new tab pane (named "Pane 4").
5. In the right section of the screen, type the caption of tab pane: "Chart".
6. In the right section of the screen, select an image in the image catalog:
   - Click the ≡ button on the right of "Image" control. Select "Catalog" from the popup menu that is displayed.
   - The window of image catalog is displayed.
   - Type "Chart" in the search control.
   - Start the search by clicking the button with the magnifier.
   - Select for example, and validate the different screens that appear.
7. Validate the description window of Tab control.
8. A new tab pane appears in the window.

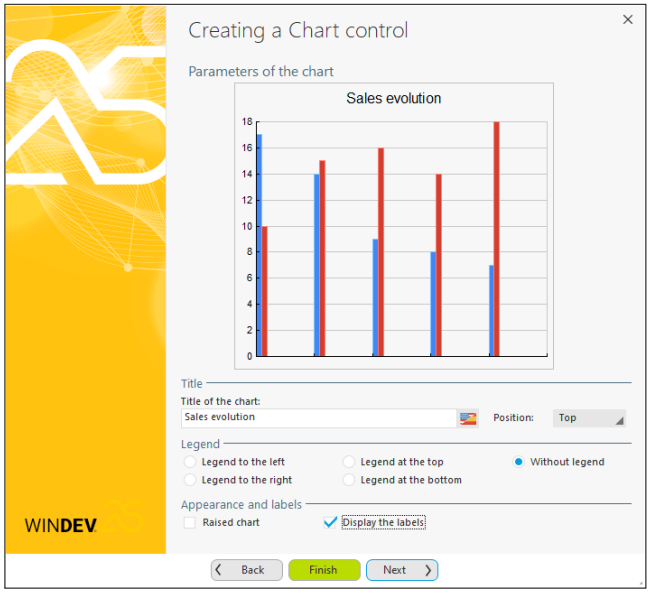
▸ To create the Chart control:

1. In the "WIN_Menu" window, select the "Chart" tab pane if necessary.
2. On the "Creation" pane, in the "Graphic controls" group, click "Chart". The control appears under the mouse cursor.
3. Click the "Chart" tab pane. The Chart control creation wizard starts.
4. In the wizard, select a "Column" chart.



Go to the next wizard step.

5. Type the chart parameters:
   - the title: "Sales evolution".
   - the legend: the chart has no legend.
   - the labels must be displayed.

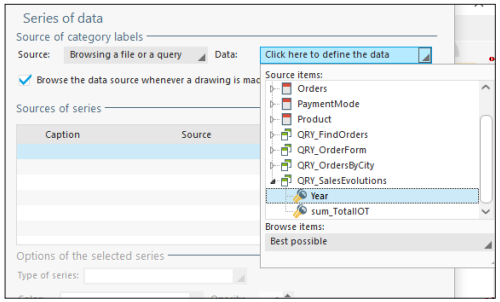


6. Go to the next step.
7. Type the parameters of axes:
   - X-axis title: Year
   - Y-axis title: TO

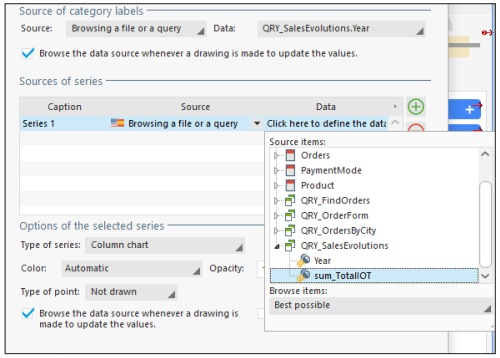
   Keep the default options and go to the next step.

8. We are now going to define the data source.

- For the labels (values displayed at the bottom of chart, the years in our example):
  - the source corresponds to: "Browsing a file or a query",
  - the data corresponds to the Year item in the QRY_SalesEvolutions query.



- For the series (values displayed in the Chart control): the Chart control will display a single series corresponding to the total IOT calculated by the QRY_SalesEvolutions query.
  - Double-click the "Sources of series" table.
  - Type the caption: "Series 1".
  - The source corresponds to: "Browsing a file or a query",
  - The data corresponds to the "sum_TotalIOT" item in the QRY_SalesEvolutions query.



Go to the next step.

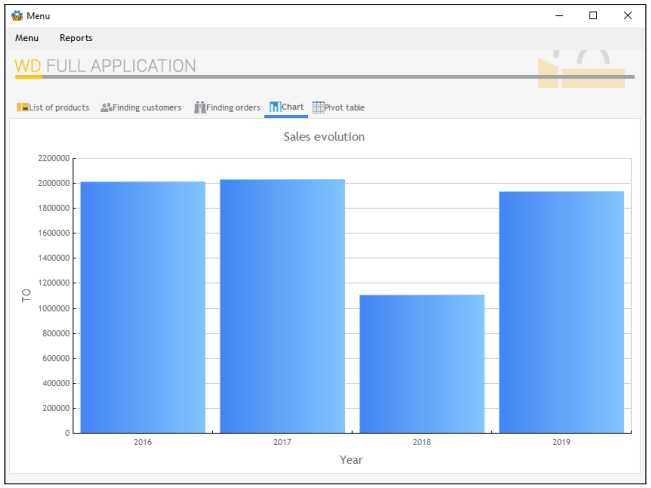
9. No background image will be associated with the Chart control. Go to the next step.

10. Give a name to the Chart control: "CHART_SalesEvolutions" and validate the wizard.

11. The Chart control is automatically created in the editor.

12. Click to run the window test.

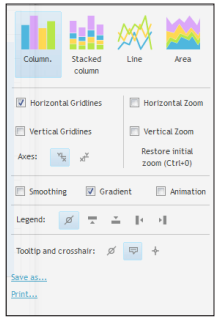
13. Click the "Chart" tab pane to see the chart.



Remark

Automatic features of Chart control (AAF)

Like the Table control, the Chart control proposes several automatic features accessible via its popup menu.



You can for example:

- change the type of chart,
- save the chart,
- print the chart, ...

14. End the test and go back to the editor.

Example

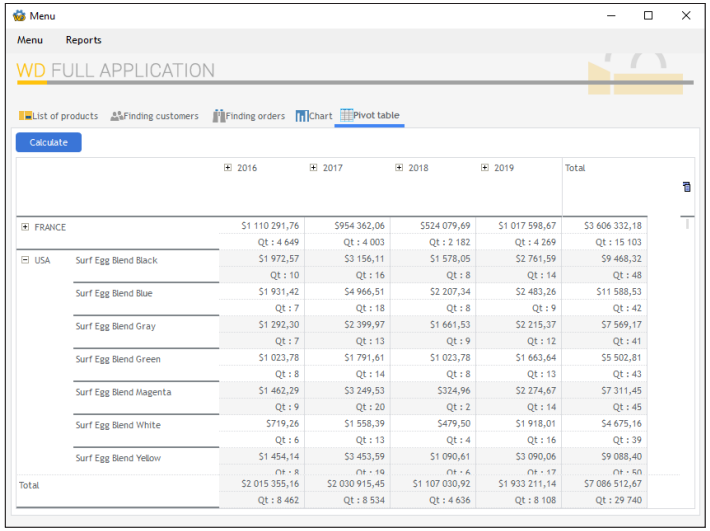
To master the use of a Chart control, don't forget to take a look at the examples supplied with WINDEV:

- Unit example: The Chart control
- Training example: WD Chart

These examples are available from the WINDEV home page (Ctrl + <).

Creating summary tables with the Pivot Table control

To handle the Pivot Table control in real conditions, we are going to create a Pivot Table control used to see the sales of products per country and per year in quantity and in turnover.



Like for the Chart control, we are going to create the Pivot Table control in a new tab pane of WIN_Menu.

▸ To create a new tab pane in "WIN_Menu":

1. Display "WIN_Menu" in the window editor (if necessary).
2. Double-click the Tab control: the control description window appears.
3. In the "General" tab, select the "Chart" tab pane and click the "New" button. A new tab pane appears.
4. Select the new tab pane (named "Pane 5").
5. In the right section of the screen, type the caption of tab pane: "Pivot Table".
6. In the right section of the screen, select an image in the image catalog:
   - Click the button on the right of "Image" control. Select "Catalog" from the popup menu that is displayed.
   - The window of image catalog is displayed.
   - In the search control, enter "Array".
   - Start the search by clicking the button with the magnifier.
   - Select for example, and validate the different screens that appear.
7. Validate the description window of Tab control.
8. The new tab pane appears in the window.

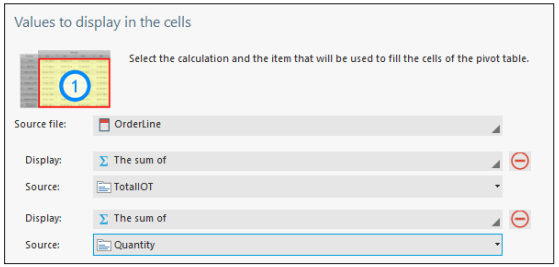
Creating the Pivot Table control

▸ To create a Pivot table control:

1. In "WIN_Menu", select the "Pivot Table" tab pane if necessary.
2. On the "Creation" pane, in the "Data" group, expand "Table and List Box" and select "Pivot Table (PVT)". The control appears under the mouse cursor.
3. Click the "Pivot Table" tab pane. The wizard for creating a Pivot Table control starts.
4. Go to the next step.
5. Various information must be displayed **in the cells**:
   - the total sales amount.
   - the quantity sold.

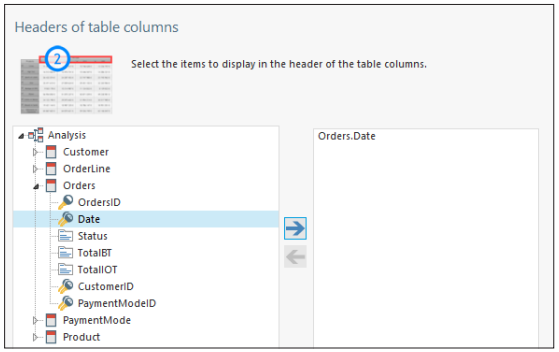
   We are going to select the source data file in the wizard: OrderLine.
   For the first information, select:
   - Display: "The sum of".
   - Source: "TotalIOT".

   Click the "Add an additional value" button.
   For the second information, select:
   - Display: "The sum of".
   - Source: "Quantity".



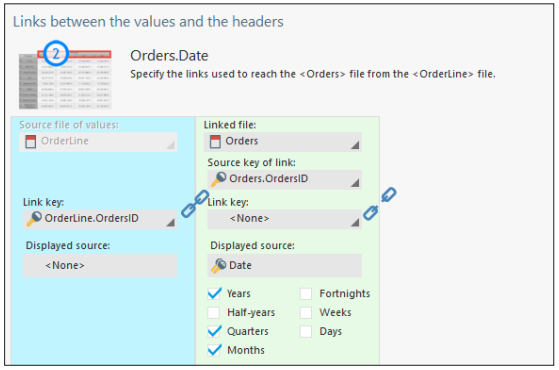
Go to the next step.

6. The years must be displayed **in the column headers**. On the left, expand the Orders data file and double-click the Date item.



Go to the next step.

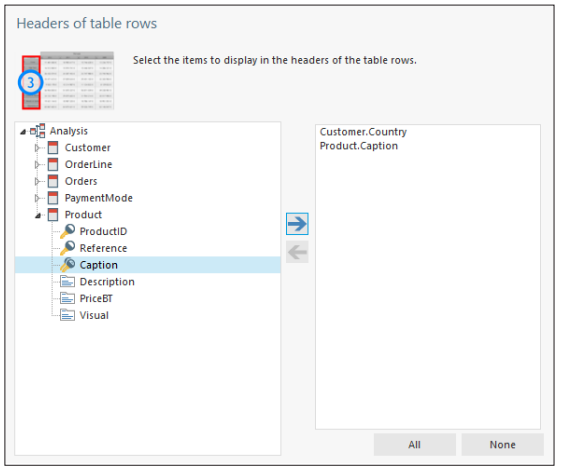
7. The wizard proposes a link to reach the Orders data file and it proposes to display three levels of information in header: the years, the quarters and the months.



8. Go to the next step.

9. The products grouped by country must be displayed **in the row headers**. In the left section:

- expand the Customer data file and double-click the Country item.
- expand the Product data file and double-click Caption.



Go to the next step.

10. The wizard proposes a link for each row header (Customer.Country and Product.Caption). Validate each link and go to the next step.

11. Give a name to the Pivot Table control: PVT_Sales.

12. Validate the wizard.

13. The Pivot Table control is automatically created in the window as well as a "Calculate" Button control. This Button control will allow the user to calculate the data to display in the Pivot Table control. This Button control can be positioned anywhere in the window.

14. Save the window (among the quick access buttons).

Testing the Pivot Table control

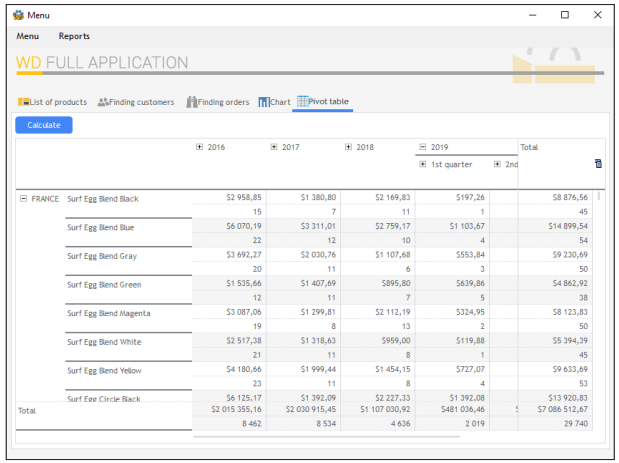
▶ Run the window test (among the quick access buttons).

1. Click the "Pivot Table" tab pane, then the "Calculate" control.

Remark

CAUTION: The calculation time depends on the size of the database and on the number of row and column headers. The result of the pivot table can be saved to avoid re-calculating at each interrogation.

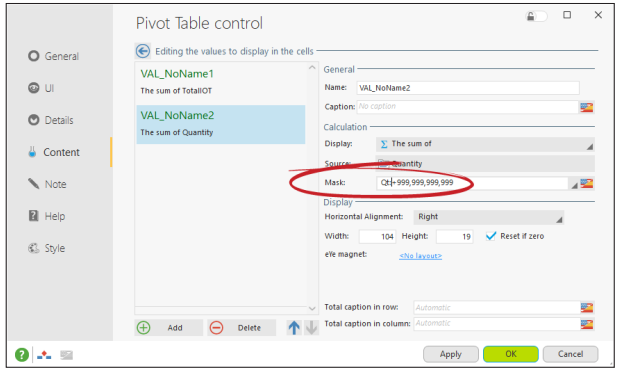
2. Click the "+" signs to expand the different columns and rows.



▸ Stop the test. We will make a small improvement in this Pivot Table control. Indeed, nothing indicates that one of the numbers in the cells corresponds to a quantity. We are going to use a specific display mask for this value.

▸ To use a display mask in a cell:

1. Display the description of Pivot Table control (double-click the control).
2. In the "Content" tab, click "VAL_NoName2". The description of values displayed in the cells appears.
3. In the "Mask" area, add the prefix "Qt: ".



4. Validate the description window.
5. Run the window test again.

Lesson 4.9. Sending an email

This lesson will teach you the following concepts

- How to send an email from a WINDEV application.
- How to include a supercontrol in a window.
- How to define the tab order in a window.
- How to open a non-modal window.

Estimated time: 20 mn

Overview

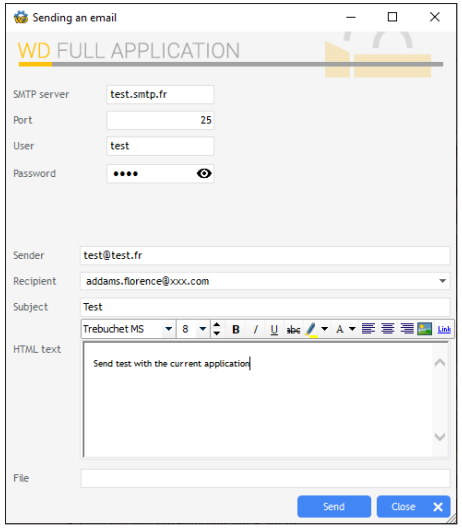
Several WLanguage functions allow you to manage the incoming and outgoing emails. You also have the ability to access the email characteristics:

- sender, recipients,
- outgoing date, subject, message,
- attachments...

WINDEV proposes several methods for managing emails:

- Management via Lotus Notes, Outloook or MS Exchange:
  - **The Lotus Notes or Outlook messaging software**: these programs allow you to send and receive emails.
  - The "**Simple Mail API**" (also called SMAPI or Simple MAPI): this management mode of emails is used by most of the Microsoft applications, especially by Microsoft Exchange.
- Management via the POP3, IMAP and SMTP protocols:
  - The **POP3** protocol: this protocol for receiving emails is recognized by all service providers. It is used to communicate with the server directly, available at your ISP. This protocol is used to list the incoming messages and to read them.
  - The **IMAP** protocol: this protocol for receiving emails is used to leave the emails the server so that they can be consulted from different messaging clients or webmails.
  - The **SMTP** protocol: this protocol for sending emails is recognized by all service providers.

In this lesson, we are going to create a window allowing the user to send an email from the "WD Full Application" application. This window is as follows:



We are going to use the SMTP protocol. Indeed, this mode is commonly used all over the world.
See the online help for more details about the other methods.

Answer

To follow this lesson, you must have followed the lessons of this part until "Printing a list of customers", page 215.
A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

Example

See the "WD Mail" example (full example supplied with WINDEV) for more details. This example is accessible from the WINDEV home page (Ctrl + <).

A window for sending emails

This window will contain all controls allowing the user to type the different email elements. A "Send" Button control will group all processes used to send the email.

Creating the window

▶ Create a new window:

1. Open the "WD Full Application" project if necessary.
2. Create a new blank window.
   - Click  among the quick access buttons.
   - The window for creating a new element is displayed: click "Window" then "Window".
   - The window creation wizard starts.
   - In the "Based on a template" tab pane, select "Use: WINTPL_Template" and validate the wizard.
3. The window for saving an element is displayed. The title of this window being "Sending an email", its name is automatically proposed: "WIN_Sending_an_email". Validate the proposed information.

Creating the controls used to configure the sending

In a first time, we are going to create all controls for configuring the SMTP server used to send messages. Four edit controls are required:

- SMTP server,
- Port of SMTP server,
- User name,
- User password.

► To create the edit control corresponding to the name of SMTP server:

1. On the "Creation" pane, in the "Usual controls" group, click .
2. The control shape appears under the mouse cursor.
3. Click the top left corner of window: the edit control is automatically created.
4. Select the control and press Enter. The caption becomes editable. Type "SMTP server" and validate. The control name automatically corresponds to EDT_SMTP_Server.

► To create the edit control corresponding to the port:

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit".
2. Select a preset Integer edit control.
3. The control shape appears under the mouse cursor.
4. Click below the "SMTP server" control: the edit control is automatically created.
5. Select the control and press Enter. The caption becomes editable. Type "Port" and validate. The control name automatically corresponds to EDT_Port.

► To create the edit control corresponding to the user name:

1. On the "Creation" pane, in the "Usual controls" group, click .
2. The control shape appears under the mouse cursor.
3. Click below the "Port" control: the edit control is automatically created.
4. Select the control and press Enter. The caption becomes editable. Type "User" and validate. The control name automatically corresponds to EDT_User.

► To create the edit control corresponding to the user password:

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit".
2. Select a preset "Password" edit control.
3. The control shape appears under the mouse cursor.
4. Click below the "User" control: the edit control is automatically created.

Tip

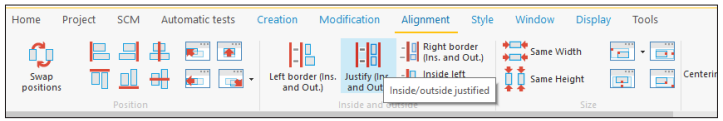
The "Password" edit control appears in the editor with a bullet and an eye.

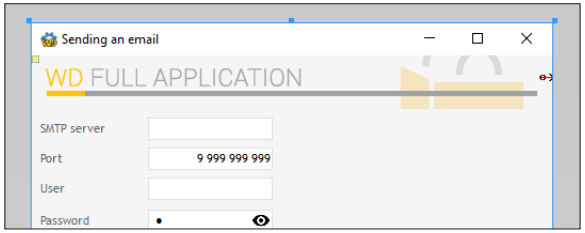
At run time, when the user types his password, the characters will be replaced by bullets. A click on the eye will allow the user to see his password in clear: this allows the user to check his password.

This feature can be disabled by programming if necessary. See the online help for more details.

► Align the created controls:

1. Select the four controls.
2. On the "Alignment" pane, in the "Inside and outside" group, click "Justify (Ins. and Out.)".





3. Save the window (or Ctrl + S).

Creating the controls used to type the email characteristics

To write an email, the user must have:

- A control used to type the sender address.
- A control used to type or select the recipient address.
- A control used to type the email subject.
- A control used to type the email text. The user must have the ability to format the email text.
- A control used to add attachments.

We are now going to create these controls in our window.

► To create the edit control corresponding to the sender address:

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit".
2. Enter "Email" in the search box at the top of the list of preset controls. Select "eMail Edit".
3. The control shape appears under the mouse cursor.
4. Click below the "Password" control: the edit control is automatically created.
5. Select the control and press Enter. The caption becomes editable. Type "Sender" and validate.

► The control used to type the recipient address must list the addresses of the customers found in the database but it must also allow the user to type another address. To do so, we will be using an "Editable combo box" control linked to the Customer data file.

1. On the "Creation" pane, in the "Usual controls" group, click "Combo Box".
2. The control shape appears under the mouse cursor.
3. Click below the "Sender" control: the wizard for creating the Combo Box control is automatically started.
4. Select "Display the data found in a file or in an existing query". Go to the next step.

5. Select the Customer data file. Go to the next step.
6. The email addresses of customers must be displayed in the control:
   - Uncheck the "CustomerID" item.
   - Check the "Email" item.

   Go to the next step.
7. The sort item is the "Email" item. Go to the next step.
8. The return value is the "Email" item. Go to the next step.
9. Keep the default options. Go to the next step.
10. In the "Additional parameters" screen, check "Allow the input". Indeed, the user must have the ability to type a new email address. Go to the next step.
11. Modify the name and caption of Combo Box control:
    - The control name is "COMBO_Recipient".
    - The control caption becomes "Recipient".
12. Validate the wizard. The control is automatically created in the window.

▶ To create the edit control corresponding to the email subject:

1. On the "Creation" pane, in the "Usual controls" group, click .
2. The control shape appears under the mouse cursor.
3. Click below the "Recipient" control: the edit control is automatically created.
4. Select the control and press Enter. The caption becomes editable. Type "Subject" and validate.

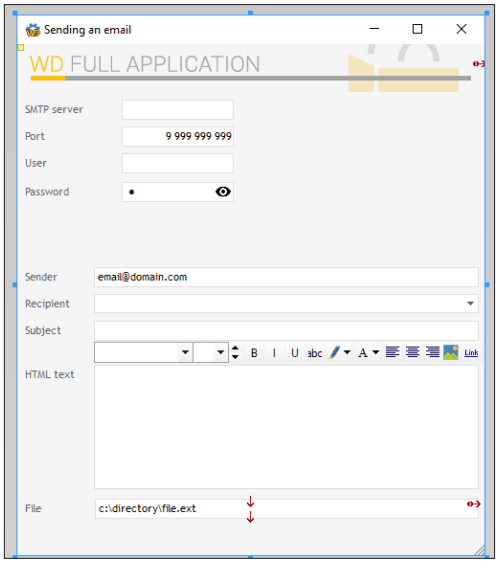
▶ For the message body, we will be using an HTML edit control: the user will have the ability to format the email text via a specific toolbar.

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit".
2. Enter "HTML" in the search box at the top of the list of preset controls. Select "HTML Edit".
3. The control shape appears under the mouse cursor.
4. Click below the "Subject" control: the edit control is automatically created.
5. Enlarge the control in order for several lines to be visible.
6. Open the control description window (double-click the control).
   - On the "General" tab, modify the mode for displaying the formatting toolbar. This toolbar must always be visible.
   - Validate the control description window.
7. Reposition the control if necessary in order for the toolbar to be displayed properly.

▶ We are going to create a file picker allowing the user to add an attachment. Instead of creating it from scratch, we will be using a preset edit control of file type. Via the popup menu of control ("Browse" option), the user will be able to select the file to attach.

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit". The list of preset controls proposed by default is displayed.
2. Select the "File" control and perform a Drag and Drop toward the "WIN_Sending_an_email" window: drop the control below the "HTML text" control. The file picker is immediately created.

▶ Align the different controls found in the window.



We will now create the Button control to send emails.

Sending the email

▶ To create the Button control to send emails:

1. On the "Creation" pane, in the "Usual controls" group, click Ok.
2. Click the location where the control will be created (at the bottom of the window for example).
3. Select the control and modify its caption (press Enter for example). The new caption is "Send".
4. Edit the WLanguage code associated with this Button control: select "Code" in the popup menu (right click).
5. Write the following lines of code in the "Click" event:
   - WLanguage code for opening the connection and starting the SMTP session:

```
MySession is EmailSMTPSession
MySession..Name = EDT_User
MySession..Password = EDT_Password
MySession..ServerAddress = EDT_SMTP_Server
MySession..Port = EDT_Port
// Starts the SMTP session
IF NOT EmailStartSession(MySession) THEN
  Error("Unable to connect to the SMTP server.", ...
    ErrorInfo())
  RETURN
END
```

This code is using an advanced ***emailSMTPSession*** variable. The different properties of this variable are used to define the characteristics of SMTP session. Then, ***EmailStartSession*** associated with this variable is used to start the session.

- WLanguage code for preparing the email:

```
MyMessage is Email

MyMessage..Sender = EDT_Sender
MyMessage..Subject = EDT_Subject
MyMessage..HTML = EDT_HTML_Text
MyMessage..Message = HTMLToText(EDT_HTML_Text)

// Adds a recipient
Add(MyMessage..Recipient,...
 COMBO_Recipient..DisplayedValue)

// Adds the attachment if necessary
IF EDT_File <> "" THEN
  EmailLoadAttachment(MyMessage, EDT_File)
END
```

This code is using an **Email** variable. The different properties of this variable are used to define the characteristics of email to send. This code associates the content of different window controls to the properties of Email variable.

- WLanguage code for sending the email:

```
// Send the email
IF EmailSendMessage(MySession, MyMessage) = False THEN
  Error("Message not sent.", ErrorInfo())
ELSE
  // Message sent
  ToastDisplay("Message sent", toastShort, ...
    vaMiddle, haCenter)
END
```

The email is sent by ***EmailSendMessage***. All you have to do is pass in parameter the variable containing the characteristics of SMTP session and the variable containing the characteristics of email to send.
If the email is sent, a Toast message is displayed, indicating that the email was sent. A Toast message corresponds to a furtive message.

- WLanguage code for closing the SMTP session:

```
// Closes the SMTP session
EmailCloseSession(MySession)
```

This code closes the session with ***EmailCloseSession***.

▸ Save the window and its code (or Ctrl + S).

Improving the window

We are going to improve our window:

- Add a Button control to close the window.
- Format the window via the management of anchors and the tab order.
- Start the window from the "WIN_Menu" window.

Closing the window

▸ To add a Button control to close the window:

1. On the "Creation" pane, in the "Usual controls" group, expand "Button": the list of preset buttons is displayed.
2. Click the "Close" button.
3. Click the position in the window where the control must be created (for example at the bottom, to the right of the "Send" Button control).

Formatting

▸ To define the anchors:

- Select the "Sender", "Recipient" and "Subject" controls: these controls must be anchored in width.
- Select the "HTML text" control: this control must be anchored in width and in height.
- Select the Button controls: these controls must be anchored to the right and at the bottom.
- Select the control for file selection: this control must be anchored in width and at the bottom.

▸ To define the tab order:

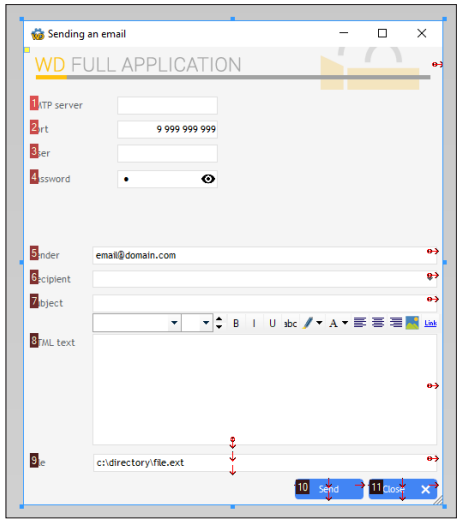
Remark

The tab order of controls is the order in which the user can type the values in the different window controls. Going from a control to another one is performed when pressing the Tab key at run time.
The default tab order corresponds to the creation order of controls. It can be modified:

- by specifying an automatic tab order: the first control in edit will be the control found in the top left corner of window, the second one will be the one found immediately to its right or below...
- by specifying a tab order by selection.

1. Display the tab order via the F5 key.
2. Define the automatic tab order: on the "Window" pane, in the "Order" group, expand "Navigation" and select "Define automatically".

3. The numbers are modified and they now appear in order.



4. Press the F5 key again to hide the numbers.
5. Save the window (or Ctrl + S).

Non-modal opening of window

The window for email management will be opened from the "WIN_Menu" window. Its opening mode is specific because this window must not prevent the information displayed in the "WIN_Menu" window from being viewed.

▸ To open the "WIN_Sending_an_email" window from the "WIN_MENU" window:

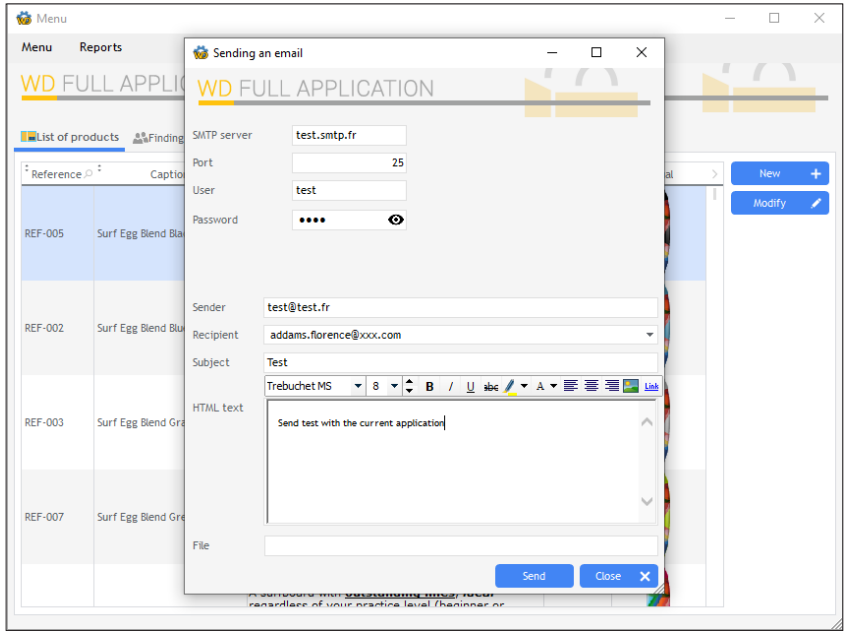
1. Open "WIN_Menu" in the editor (double-click its name in the "Project explorer" pane, for example).
2. In the editor, expand the "Menu" option and select "Exit".
3. Display the popup menu of "Exit" option (right mouse click) and select "Add before".
4. Type the option caption ("Send an email") and validate.
5. Select the "Exit" option again.
6. Display the popup menu of "Exit" option (right mouse click) and select "Insert a separator".
7. Display the popup menu of "Send an email" option (right mouse click) and select "Code".
8. Write the following WLanguage code:

```
// Opens the window for sending emails
OpenChild(WIN_Sending_an_email)
```

In this WLanguage code, ***OpenChild*** is used to open the window in "non-modal" mode: the user will be able to write an email and to see the information displayed in the main window at the same time.

▸ Save the window and its code (or Ctrl + S).

▸ Run the project test (among the quick access buttons) and open the window for sending emails via "Menu .. Send an email".



Lesson 4.10. Identifying the user: the user groupware

This lesson will teach you the following concepts

- What is the user groupware?
- Integrating the user groupware.
- Configuring the user groupware.
- Checking the user groupware.

Estimated time: 20 mn

Overview

An application can be used by several contributors with different profiles. It is often necessary to define several access levels according to the user.

Let's take a simple example: an application for sales management proposes the following features:

- Viewing the price list.
- Modifying the price list.
- Entering orders.
- Entering customers.

The accesses differ according to the user. Some examples:

- the administrative assistants can see the price list and create orders.
- the sales people can see the price list, place orders and create new customers.
- the sales directors have access to all options.

WINDEV allows you to easily manage these access levels in your applications via the user groupware.

We are going to include the user groupware in our "WD Full Application" application and to configure it.

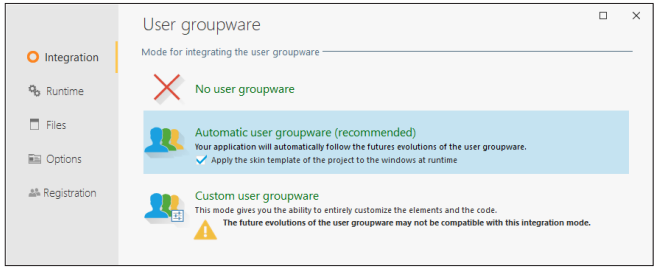
Answer

If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

Integrating the user groupware

▸ To include the user groupware in the "WD Full Application" project:

1. On the "Project" pane, in the "Project" group, click "User groupware". The window for configuring the user groupware is displayed.



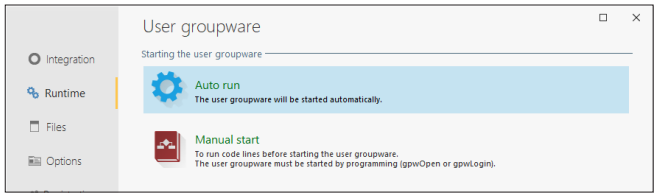
2. In the "Integration" tab, two integration modes are available:

- **Automatic user groupware**: all elements of user groupware are included in your application. This allows you to benefit from all groupware evolutions. Furthermore, the skin template of project can be applied to the groupware windows.
- **Custom user groupware**: all groupware elements are included in your application via an internal component. You can customize the different elements of the user groupware. However, the evolutions will not be taken into account.

3. Select "Automatic user groupware".

4. Select the "Runtime" tab. Two start modes are available:

- Auto run: the groupware is started as soon as the application is started.
- Manual start: the groupware will be started by programming.



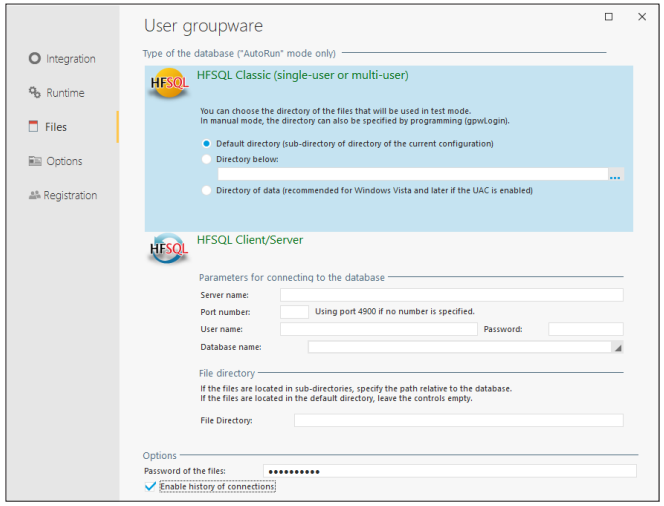
5. Keep the "Auto run" option.

6. Select the "Files" tab. This tab is used to define the format and location of the data files found in the user groupware. In our case, we will be using HFSQL Classic data files, in the location specified by default.

Remark

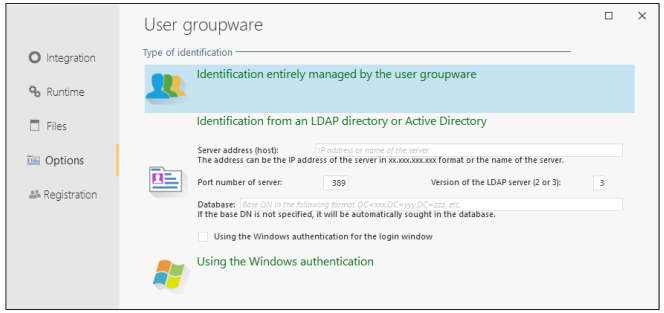
If you (or the end user) is using Windows Vista (or a more recent operating system), we advise you to use the "Data directory" option.

7. In the "Files" tab, select "Enable history of connections". This option allows the supervisor to get information about the connected users.

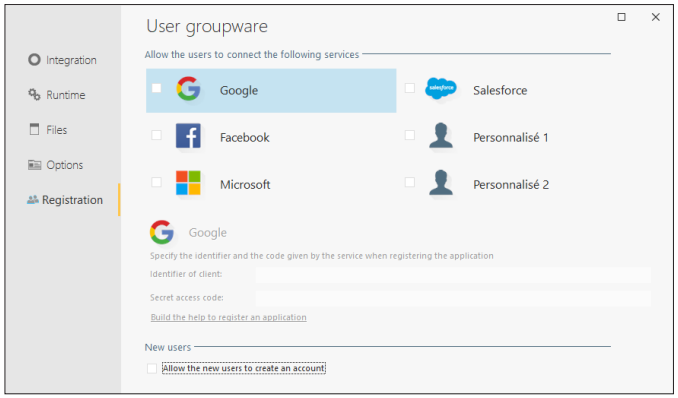


8. Select the "Options" tab. This tab is used to define the identification mode of user:

- management by the user groupware.
- using an LDAP directory or Active Directory. In this case, when installing the application, the user will be able to enter the parameters of his LDAP directory or Active Directory.
- using the Windows authentication.



9. Select the "Registration" tab.

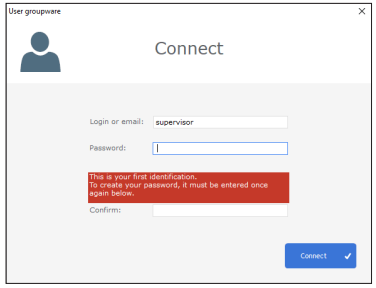


This tab allows authorizing the use of a specific connection service. For the specified service, it is necessary to specify the corresponding identifiers (provided when registering the application with the selected service).

10. Validate. A message is displayed, indicating that a Supervisor user is created.

Remark

A single user exists by default, the supervisor. During the first application start, connect yourself by using the name: "supervisor". You will have the ability to define the password used by the supervisor.

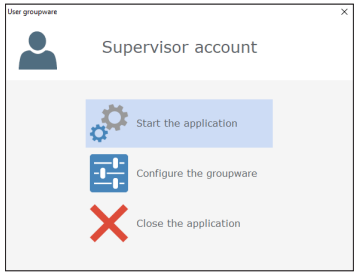


11. Validate this message. The user groupware is included in the application.

▸ Let's now run the test of our application:

1. Run the project test (GO among the quick access buttons). A login window is displayed.
2. Connect yourself as supervisor.
3. Define the password and confirm it. Validate.

4. A new menu is displayed, allowing you to run the application test or to configure the application.



5. Select "Configure the groupware".

Configuring the user groupware

Configuring the groupware consists in defining the different application users as well as their rights on the different windows and controls.

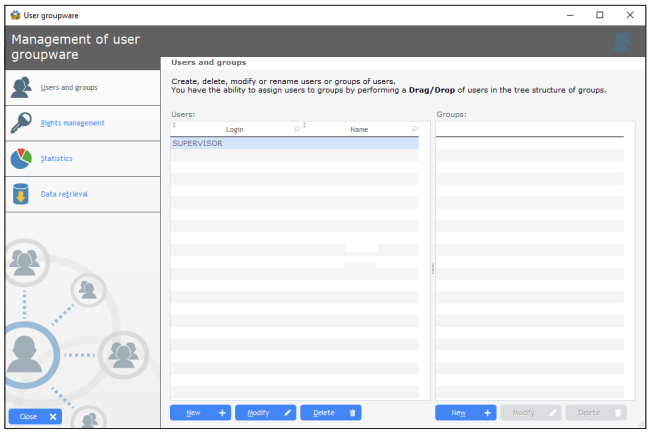
Remark

The configuration of users can be performed:

- when developing the application. The necessary data files (regarding the users and their rights) can be installed along with the application.
- when the application is installed, by the administrator of the application.

Creating users and groups

▸ To configure the user groupware, we are going to create a "Test_1" user and associate this user with the "Tests" group. Then, we are going to configure the management of rights for the group.



▶ To create a user:

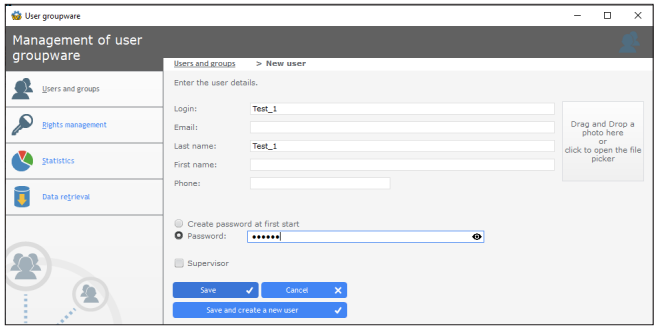
1. Click the "New" button found below the "Users" area. The screen for entering a new user is displayed.

2. Type the following information:

- Login: Test_1
- Last name: Test_1
- Password: Test_1

Remark

By default, the password is hidden during the input. To check the password, click on the eye icon: the password appears in clear as long as the mouse button is pressed.

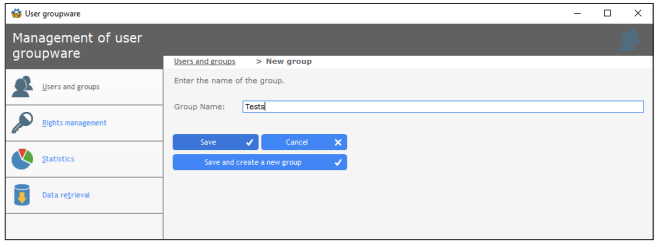


3. Click on "Save". The "Test_1" user appears in the list of users defined for the user groupware.

▶ To create a new group of users:

1. Click the "New" button found below the "Groups" area. The screen for entering a new group is displayed.

2. Enter the name of the group: "Tests".

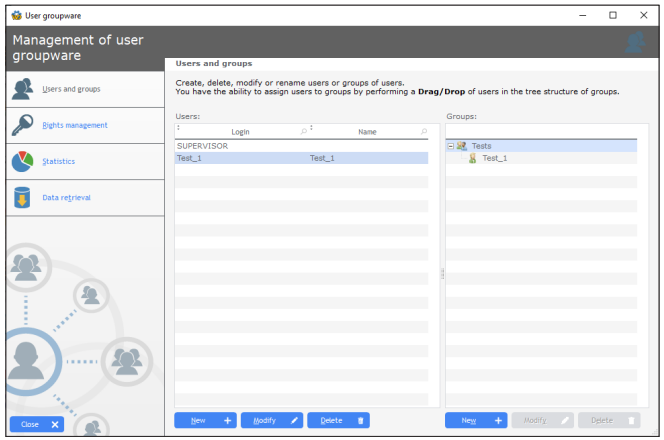


3. Click on "Save". The "Tests" group appears in the list of groups defined for the user groupware.

▶ To associate the user with the group:

1. Select the "Test_1" user in the window.

2. Drag and Drop the "Test_1" user to the "Tests" group.



3. The association is performed.

Defining the rights

We are now going to define rights for the "Tests" group. These rights will be granted to all users found in the group. In our example, the users found in the "Tests" group will not be allowed to:

- Display the window for sending emails.
- Create or modify a product.

▶ To define the rights:

1. Click "Rights management" on the left of the window.

2. Select the "Tests" group.

3. Click "Next".

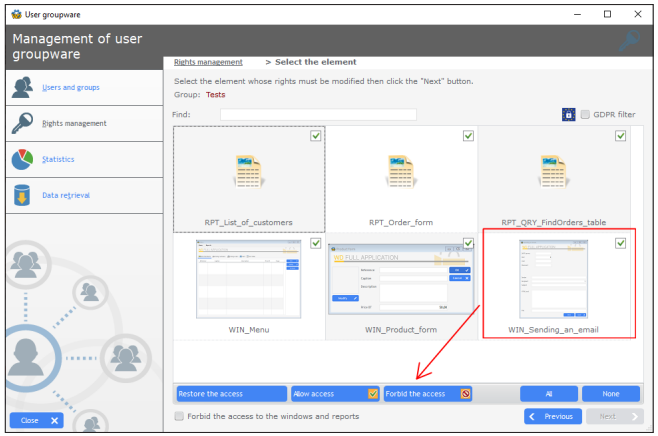
4. The window that is displayed allows you to select each window or report of the application.

- For each window or report, it is possible to specify whether or not the element will be accessible by the group.
- For each window, you can define whether the window controls will behave like the application (default mode) or whether they will be disabled, invisible or grayed.

▶ To forbid the access to the "WIN_Sending_an_email" window:

1. Select the window in the list.

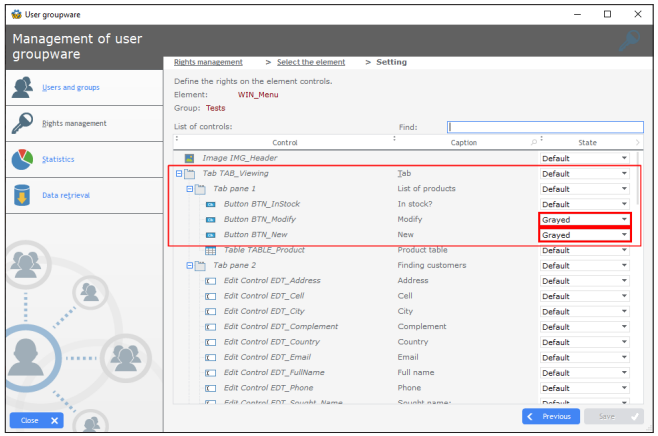
2. Click the "Forbid the access" button.



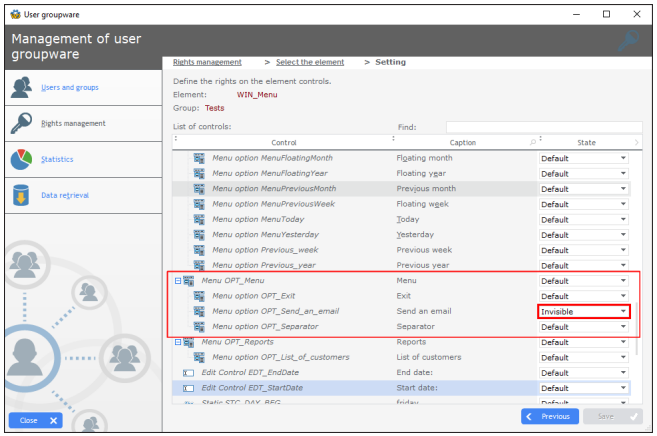
3. The window is forbidden.

▸ Define the rights on the "WIN_Menu" window. This window contains a menu option used to display the window for email management as well as the "New" and "Modify" buttons.

1. Select the "WIN_Menu" window.
2. Click the "Next" button.
3. The window for configuring the rights on the window controls is displayed.
4. Switch the "BTN_Modify" and "BTN_New" buttons to "Grayed":



5. Switch the "Send an email" option to invisible:



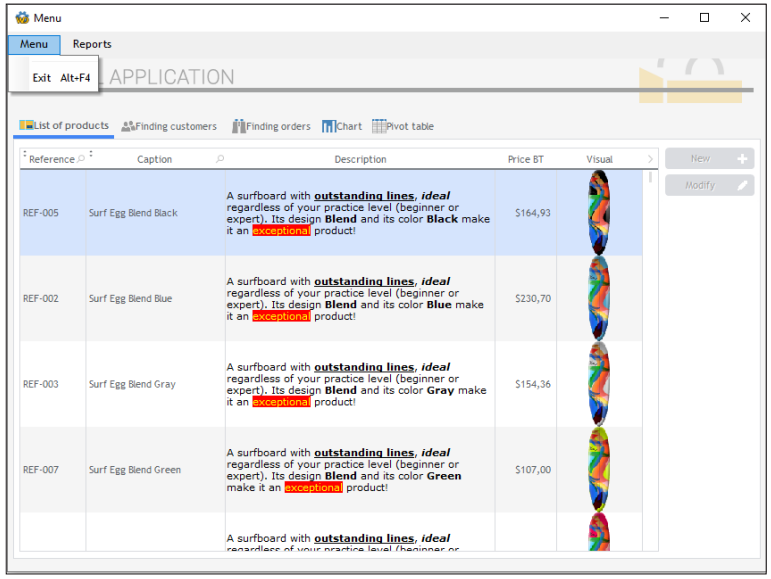
6. Click the "Save" button.
7. Close the configuration window.
8. The WINDEV editor is displayed.

Application test

▸ We are now going to run the application test by using the "Test_1" login.

1. Run the project test (GO among the quick access buttons). A login window is displayed.
2. Connect yourself as "Test_1" with the "Test_1" password. Validate.

3. The application starts. If necessary, select the tab "List of products". You will notice that the "New" and "Modify" buttons are grayed and that the "Send an email" option is invisible.



4. Exit from the application and go back to the editor.

Disabling the management of user groupware

The user groupware will no longer be used in the rest of this tutorial. Therefore, it can be disabled.

1. On the "Project" pane, in the "Project" group, click "User groupware".
2. In the window that is displayed, in the "Integration" tab, select "No user groupware".
3. Validate.

Remark

If automatic tests are performed in your application, the user groupware must be configured in order not to use the login window. To do so, fill the "Automatic login in test mode" option in the "Runtime" tab of the window for configuring the groupware.

Lesson 4.11. Re-use code via the external components

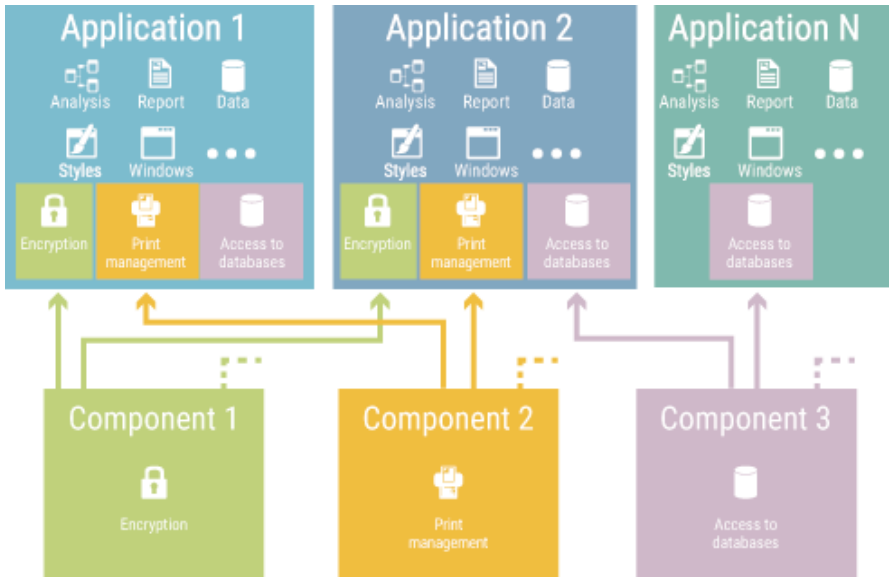
This lesson will teach you the following concepts

- What is an external component?
- Creating an external component, step by step.
- Distributing an external component.

Estimated time: 40 mn

Overview

An external component is a building block that can be re-used in an unlimited number of projects (and therefore executables).



An external component allows you to have an identical process with a single source code to perform a given operation even if this process must be performed by several projects.

The projects that use an external component have access in the WINDEV editor to the name of objects, procedures or methods made visible by the component creator. The projects cannot see or modify the source code. This ensures confidentiality and stability of source code.

Creating an external component is **child's play**.

How to proceed? Don't change anything, create your windows, procedures, classes. Then, when you're done, create a Component configuration with all the elements needed for your component, and generate. It's that simple!

A component can contain code, windows, an analysis, data files, etc!

Remark

Five methods can be used to share "code" in WINDEV:

1. The sets of procedures.
2. The classes.
3. The supercontrols (via a dictionary).
4. The external components.
5. The internal components.

Let's present some cases in which the external components can be useful.

Teamwork

A developer creates and maintains an external component that is made available to other developers. No risk of modifications "made by mistake"!

The huge projects

The external components allow you to have several small projects as well as a central project in which the elements found in the other projects are visible. The external components are a lot more convenient to use than the libraries (WDL files).

The databases accessed by several projects

When the same database is accessed by several projects, you often find inconsistencies in the database caused by modified or old source code. By grouping the operations used to access the database (at least in write mode) in an external component, a single source code must be checked and maintained ; therefore, the risks of database inconsistency are reduced.

Furthermore, using a component avoids recompiling the application when the analysis is modified.

The processes used in several projects

Complex processes are often used in several projects. These processes can be re-used via "sets of procedures" or "classes". In this case, the modifications may not be applied to the different projects, making these elements no longer compatible between themselves.

Using external components prevents such out-of-sync modifications, preserves the compatibility between projects and simplifies the common modifications.

Furthermore, the PRIVATE keyword allows you to ensure the confidentiality of your code at all levels of external component. When re-using your external component in another WINDEV project, the call to a PRIVATE procedure will not be allowed but the documentation regarding the use of procedure will be displayed!

The ability to distribute a feature or set of features

The external components allow you to develop a feature or a set of features. Other WINDEV developers will be able to include these features in their own projects. The developers who are using an external component can see the component elements that are made visible. However, the source code cannot be viewed or modified.

Your external components can be distributed (free of charge or not)!

Multi-product external component

An external component can be intended to operate in:

- a WINDEV application,
- a WEBDEV application,
- a WINDEV Mobile application,
- the three types of applications.

In this last case, WINDEV allows you to:

- include the elements coming from different products (WEBDEV and/or WINDEV Mobile) in the same external component.
- specify the corresponding WLanguage code for each runtime platform (for example, a window is displayed by ***Open*** in WINDEV and a page is displayed by ***PageDisplay*** in WEBDEV).

Step by step

Step 1 : Creating an external component

We are going to create an external component from the "WD Full Application" project. This project is used to manage orders, products and customers. This external component will be used to immediately find out in another application the customers corresponding to a given city.

During the call to the external component, you will have the ability to:

- Pass a city in parameter.
- Retrieve a string containing the customer name and the total amount of his orders.

To avoid having to develop the code required for the component to operate, the "WD Full Application" project contains all necessary elements.

Answer

If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (with windows)".

A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

► For the component needs, the project contains:

- a "QRY_OrdersByCity" query. This query is a query with parameters used to find out the customers corresponding to a given city as well the total amount of their orders.
- a set of procedures "SET_Component". This set of procedures contains:
  - the "GiveOrdersByCity" procedure that returns, for the city passed in parameter, a string containing the customer name and the total amount of his orders.
  - the "DataLocation" procedure that is used to locate the data required by the component.
- an analysis used to described the data files for storing information.

► We are now going to create our component. To do so, a project configuration must be created.

Remark

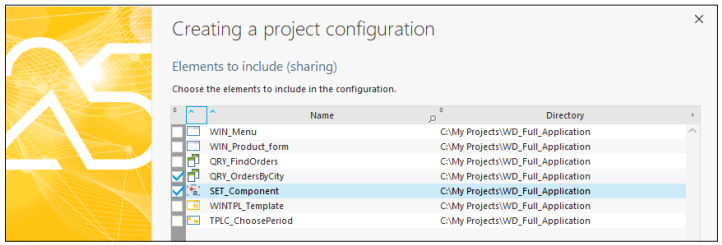
The project configurations are used to create several "targets" from the same project:

- A 32-bit application,
- A 64-bit application,
- A component,
- ...

You have the ability to choose the requested configuration at any time and to generate in a single operation all elements for all project configurations.

► To create a project configuration whose type is "Component":

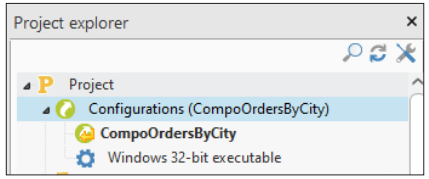
1. On the "Project" pane, in the "Project configuration" group, expand "New configuration" and select "External component (WDK)".
2. The wizard for creating a project configuration starts. WINDEV proposes to create a project configuration whose type is "Component". Go to the next step.
3. Give a name to your project configuration: "CompoOrdersByCity" for example. Go to the next step.
4. Keep the default options. Go to the next step.
5. Select the elements that will be included in the project configuration. In this example, these elements will also be found in the component.
   - Click the "None" button.
   - Select the "QRY_OrdersByCity" and "SET_Component" elements.



6. Go to the next step and validate the configuration creation.
7. The new configuration automatically becomes the current configuration.

Remark

To find out and modify the current configuration, use the "Project explorer" pane.



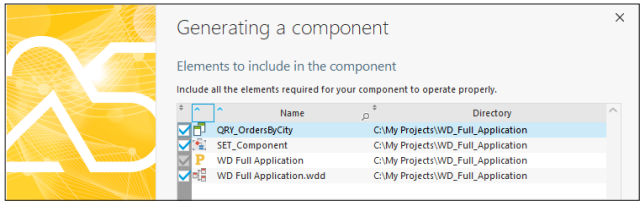
The configuration in bold corresponds to the current configuration.

To change the current configuration, double-click the name of the configuration to enable.

▶ We are now going to generate our component.

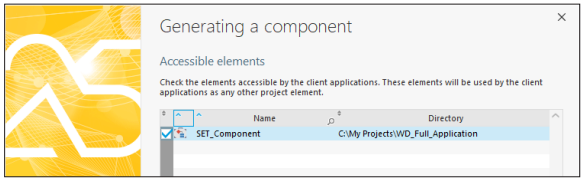
1. On the "Project" pane, in the "Generation" group, click "Generate".

2. The wizard for generating the component starts. Go to the next step of the wizard.
WINDEV lists the elements that will be included in the component (elements specified in the configuration and analysis).



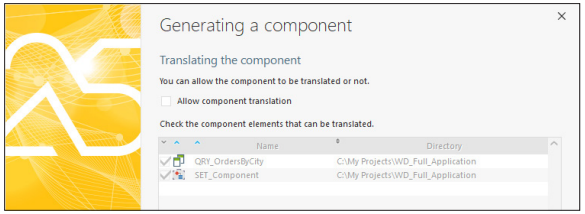
3. Select all elements ("All" button) and go to the next step.

4. The wizard asks you to select the component elements that will be accessible from the client application. In our example, only the set of procedures "SET_Component" will be used:



5. Go to the next step.

6. WINDEV proposes to allow the translation of component. This feature is interesting if your component must be used by multilingual applications.



If the option "Allow component translation" is checked, the specified component elements can be translated from the application that is using the component.
This translation will be performed via WDMSG, independent module used to check out and check in the project messages to translate.
In this example, don't check this option. Go to the next step.

7. Choose the component languages. Our example will contain the English language only. Go to the next step.

8. The wizard proposes to manage the different component versions. In our example, the component was just created.
Keep the default options. Go to the next step.

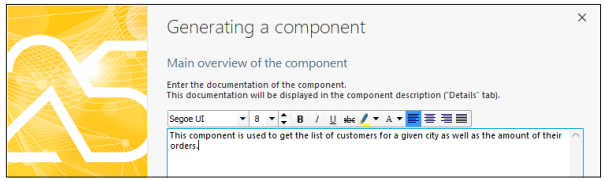
9. Type the information regarding the component:

- Owner,
- Caption, Copyright, ...

10. Go to the next step.

11. You have the ability to associate an image with your component. Go to the next step.

12. Specify the main component overview. This overview will allow the user to find out the component purpose. Type for example:



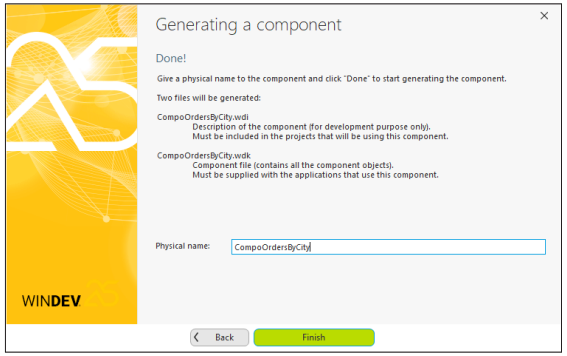
13. Go to the next step. The wizard will automatically generate the documentation about the component. This documentation can be based on the comments inserted into your source code.

14. Go to the next step. You have the ability to modify the generated documentation. Don't do anything. Go to the next step.

15. You have the ability to create a help file associated with your component (CHM format). This help file will be supplied along with the component. The users will be able to access it by pressing F1 from the component code.
We will generate no help file. Go to the next step.

16. The component will not be saved in the SCM. Go to the next step.

17. All we have to do now is give a physical name to our component.



▶ Validate ("Finish"). A message indicates that the component was generated. Click "End" to validate the message.

▶ Well done, you've just created your first ready-to-use component!

Step 2 : Using the external component

Once created, your component can be used in any other WINDEV project. Let's now see how this component can be re-used.

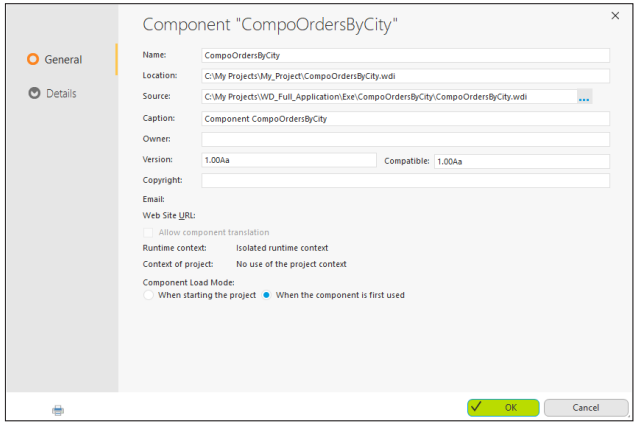
▸ To do so, we are going to create a new project and import our component into this project.

1. Close the current project: on the "Home" pane, in the "General" group, expand "Close" and select "Close the project".
2. Validate the closing of project and save the modifications if necessary.
3. The WINDEV home page is displayed.
4. Create a new project: click "Create a project" in the home page.
   - This project is a Windows application.
   - This project is named "CompUse" and it has no analysis.
5. On the "Project" pane, in the "Project" group, expand "Import" and select "An external component .. From a file".
6. In the directory of "WD Full Application" project, select the "EXE\CompoOrdersByCity" sub-directory, then the "CompoOrdersByCity.wdi" file.

Remark

If the component was created from:
- the "WD Full Application (Exercise)" example, select the "My Projects\My Examples\WD Full Application (Exercise)\EXE\CompoOrdersByCity" sub-directory, then the "CompoOrdersByCity.wdi" file.
- the "WD Full Application (with windows)" example, select the "My Projects\My Examples\WD Full Application (With windows) (Exercise)\EXE\CompoOrdersByCity" sub-directory, then the "CompoOrdersByCity.wdi" file.

7. Click "Open", the description of our component is displayed. This description contains:
   - On the "General" tab, the elements typed when creating the external component as well as its location.



- On the "Details" tab, the component description as well as the help that was automatically generated. This allows you to identify the re-usable component elements.

Remark

The component description can be accessed at any time. Simply select the component in the "Project explorer" pane and select the "Description" option from the popup menu. In this case, you also have the ability to define the load mode of component.

8. Validate ("Close" button). The component is included in your project.

▸ We are now going to create a blank window to use the component.

1. Create a blank window.
   - The window title is "Customers by city".
   - The window name is "WIN_Customers_by_city".
   - Save.
2. Display the WLanguage events associated with the window ("Code" from the popup menu).
3. We are going to call the ***DataLocation*** function of component in the "End of initialization" event. This function expects in parameter the path used to access the data files handled by the component. Type the access path to the data found in the "WD Full Application" example. For example:

```
DataLocation(...// Specify the path of YOUR data
  "C:\WINDEV\Tutorial\Exercises\" + ...
  "WD Full Application\Exe")
```

Remark

If your project is using another procedure named "DataLocation", the name of component procedure must be prefixed by the name of the set of procedures used. The code becomes:

```
SET_Component.DataLocation(...
```

4. Close the code editor.
5. Add the following controls into the window:
   - A text Edit control whose caption is "City" and whose name is "EDT_City".
   - A Table control named "TABLE_Result", filled by programming and that includes 2 columns:
     - a "Name" column of Text type.
     - a "Total sales" column of Currency type.
   - A Button control whose caption is "Search" and whose name is "BTN_Search".

▶ You can now edit the WLanguage events associated with "BTN_Search". When this control is clicked on, we will run the search procedure found in the component. This procedure:

- expects the city name in parameter
- returns a string in the following format:

```
Name of customer 1 + TAB + Total sales 1 + CR +
Name of customer 2 + TAB + Total sales 2 + ...
```

So, the code of the "Click" event from "BTN_Search" should:

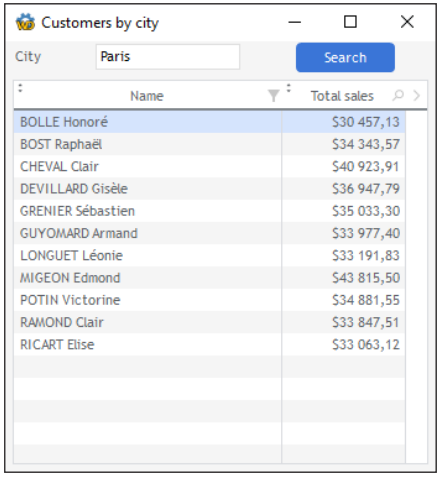
- call the ***GiveOrdersByCity*** procedure of component while passing the content of EDT_City control in parameter.
- process the returned string to add it into the Table control.

Write the following WLanguage code:

```
sResultList is string
// Gets the list of customers and their orders
// for the specified city
sResultList = GiveOrdersByCity(EDT_City)
// If the list is not empty
IF sResultList <> "" THEN
 // Clears the Table control
 TableDeleteAll(TABLE_Result)
 // Browses the results
 FOR EACH STRING sACustomer OF sResultList SEPARATED BY CR
     // Adds this client to the Table control
     TableAdd(TABLE_Result, sACustomer)
 END
ELSE // If the list is empty
 InfoBuild("No customer found for %1", EDT_City)
END
```

▶ Close the code editor and save your window.

▶ Run the window test: click among the quick access buttons. In the edit control, type "Paris" (while respecting the case) and click the "Search" button. The list of customers is displayed.



▶ That's it! Child's play isn't it?

You know how to create a component and how to re-use it in your applications. You also have the ability to manage the setup procedures of your components, in order to distribute them separately from your applications for example.

Distributing an external component

Two methods can be used to distribute a component:

1. Provide the necessary files "manually", this is a "standard" distribution.
2. Create a more "professional" distribution, via the setup editor of WINDEV (WDInst).

Standard distribution

In this case, you must supply all the files required for your component to operate. These files are created when generating the component (WDI, WDK and WDO files, images, other documents, ...). These files will be manually copied from their source directory to the destination directory. The WINDEV projects that use this component will find the dependent files in this destination directory.

List of files that must be supplied for a standard distribution:

- the files automatically generated by WINDEV (WDK, WDI, ...).
- the dependency files.
- the WDO file must be supplied if the component is using dependency files. This file contains the references to the external files used in the component.

Professional distribution

The distribution of components via a setup procedure consists in providing a setup program to the users of WINDEV component. This program installs all the files required for using the component in the directory specified by the user.

This setup mode is used to automatically manage:

- the WDO file and the setup of additional files used by the component.
- the automatic setup of necessary tools (MDAC, ODBC driver for HFSQL, ...).
- the automatic update of data files used by the component (if necessary).
- the uninstall program of component.

► Close the "CompoUse" project: on the "Home" pane, in the "General" group, expand "Close" and select "Close the project".

► Open the "WD Full Application" project that was used beforehand (for example, display the home page and select the "WD Full Application" project found in the recent projects). If necessary, select the project configuration corresponding to the component in the "Project explorer" pane.

► To create the setup, go to the "Project" pane, "Generation" group, and click on "Setup procedure".
The wizard for creating the component setup starts.

We won't go into details about the different setup modes of a component. Follow the instructions given by the wizard.
See the online help for more details (keyword: "External component, Distributing a component").

Lesson 4.12. Consuming a Webservice

This lesson will teach you the following concepts

- Overview.
- Importing and consuming a Webservice.

Estimated time: 10 mn

Overview

In most cases, an XML Web service is defined as an application accessible via the standard Internet protocols. More specifically, Web services allow several computers connected via Internet to interact.

Web services allow you to run procedures and processes on a remote Web server (.NET, SOAP or J2EE) from a client computer.

With WINDEV, these Web services can be used as client, via the SOAP protocol on HTTP (the standard Internet protocol for transferring HTML pages), with the SOAPxx, DotNetxx and J2EExx functions.

Regardless of the platform of the Web server (.NET, J2EE, ...), a Web service is accessible via the SOAP protocol.

Remark

With WINDEV, you don't even have to be an expert in this field. A wizard takes care of ("almost") everything!

Practical example

A Webservice specific to the Tutorial allows you to check the different operations that can be performed on a Webservice.

Integrated to the "WD Full Application" project, this Webservice is used to interrogate a supplier database to check whether a product is available (stock) from its reference.

In a first time, the Webservice will be imported into the "WD Full Application" project then it will be used in the application to check the product availability from a Product form.

Importing a Webservice

▸ Close the current project if necessary. The WINDEV home page is displayed.

▸ In the home page, click "Tutorial" and select "Full application (Exercise)". The project is loaded.

Answers

If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (With windows)".

A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)"

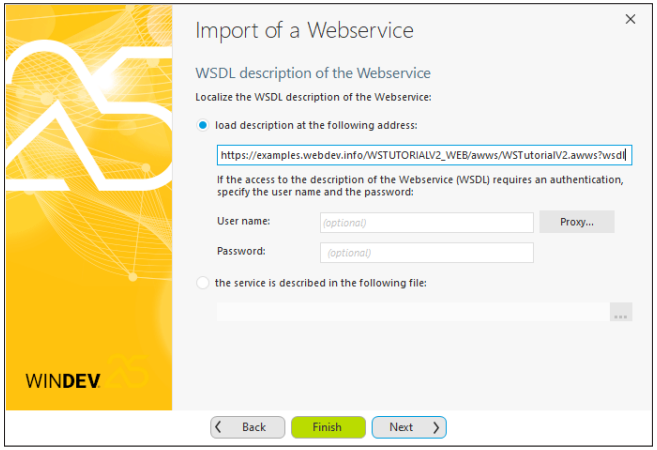
▸ Position (if necessary) on the "32-bit Windows executable" configuration: in the "Project explorer" pane, in the "Configurations" folder, double-click "Windows 32-bit executable".

▸ To import a Webservice into the project:

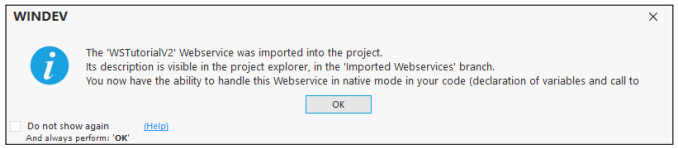
1. On the "Project" pane, in the "Project" group, expand "Import" and select "A Webservice".
2. The import wizard starts. Go to the next step.
3. Specify the address into which the WSDL description of Webservice must be imported:

```
https://examples.webdev.info/WSTUTORIALV2_WEB/awws/WSTutorialV2.awws?wsdl
```

Reminder: This Webservice is used to interrogate a supplier database to check the availability (stock) of a product from its reference.

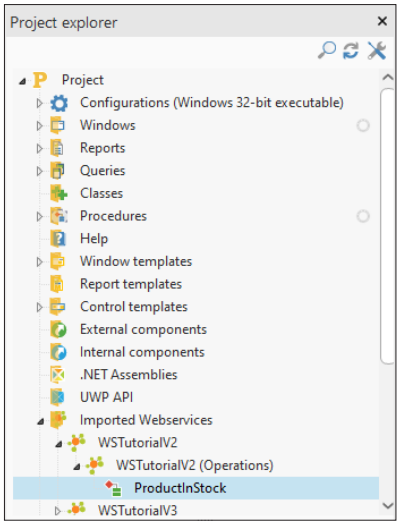


4. Go to the next step. The Webservice is imported.



5. Validate the information window. The imported Webservice is in the "Imported Webservices" folder of the "Project explorer" tab pane.
6. In the "Project explorer" pane, expand the "Imported Webservices" folder.

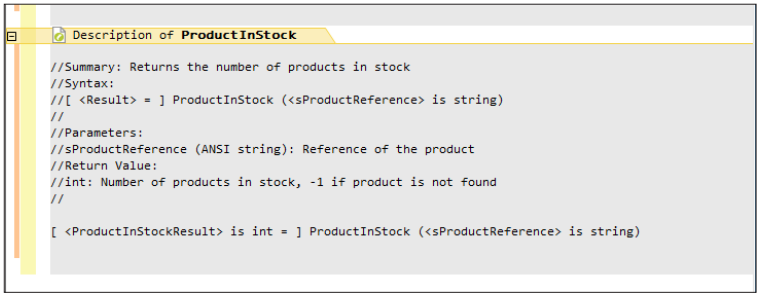
Let's take a closer look at the information displayed in the "Project Explorer" pane:



The structure includes:

- the Webservice name (WSTutorialV2 in this example),
- the name of each function (ProductInStock in this example).

To find out how to call the Webservice, simply double-click the name of the function in the "Project explorer" pane. The code editor displays the function description, with the prototype for calling the function:

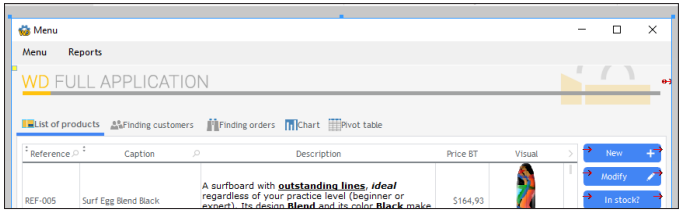


Consuming a Webservice

In our "WD Full Application" example, the call to the Webservice will be included in the tab used to see the list of products. A button "In stock?" is used to check whether the product displayed via the Webservice is available.

▶ To use the Webservice:

1. Open the "WIN_Menu" window in the editor (double-click its name in the "Project explorer" pane, for example).
2. Add a Button control in the "List of products" tab pane:
   - On the "Creation" pane, in the "Usual controls" group, click Ok.
   - Click below the "Modify " Button control in the window.
   - The control is automatically created.
3. Modify the characteristics of the control ("Description" from the popup menu). This control is named "BTN_InStock" and its caption is "In stock?".



4. Display the events associated with the control ("Code" from the popup menu).
5. Write the following WLanguage code in the "Click BTN_InStock" event:

```
// Displays the Webservice response
InfoBuild(...
    "Number of ""%1"" products in stock: %2", ...
    COL_Reference, ProductInStock(COL_Reference))
```

Let's study this code:

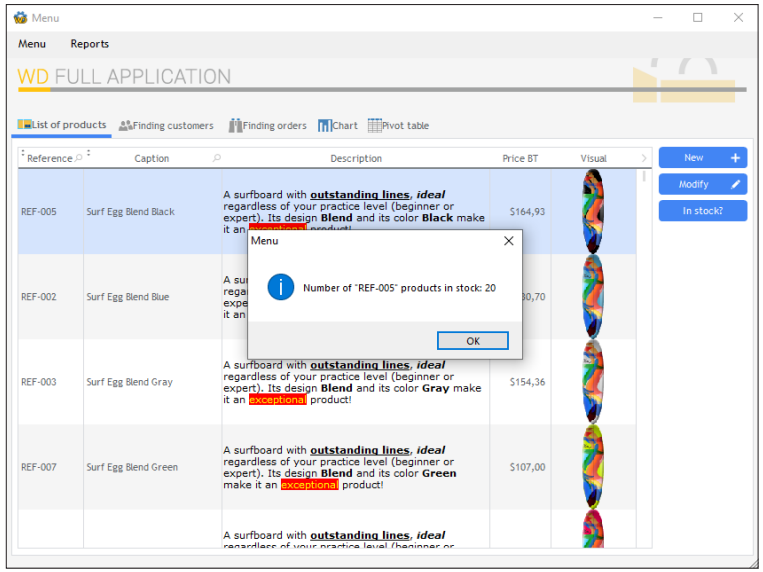
- The ProductInStock function of Webservice is called. This code is using the function prototype that was displayed beforehand in the code editor.
- The response is formatted and displayed.

6. Close the code editor and save the window (or Ctrl + S).

▶ We are going to check the operating mode of Webservice:

1. Run the project test (among the quick access buttons).
2. Click the "List of products" tab if necessary.

3. Select any product in the Table control and click "In stock?".



4. Validate the information window and close the application.

Lesson 4.13. Monitor the evolution of your applications

This lesson will teach you the following concepts

- What is the dashboard?
- Automatic tests.

Estimated time: 20 mn

Overview

The project dashboard is an essential element for managing the WINDEV projects. The project dashboard gives an overall view of the progress of a project.

The dashboard includes several elements (widgets) that give an overall view of the project status.

In this section, we will present the operations that can be performed on the dashboard elements as well as the management of automatic tests and the optimization of queries.

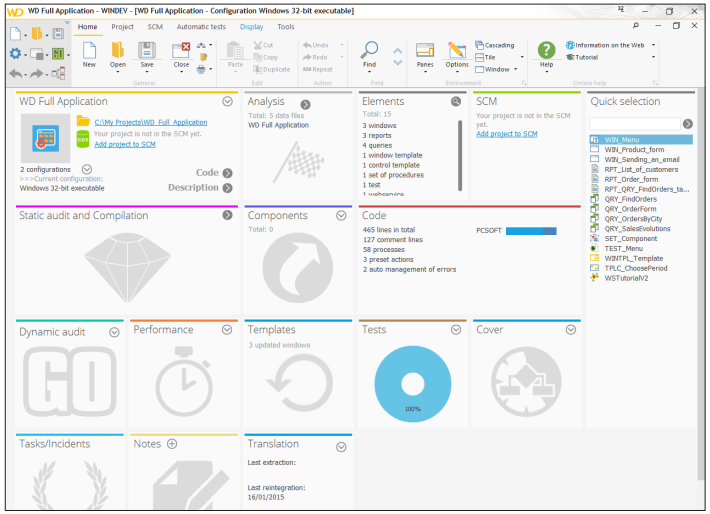
Answer

If you did not create the windows in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (with windows)".

A full corrected application project is also available: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)"

▸ To display the project dashboard (if not already done), on the "Project" pane, in the "Project" group, click  .

The dashboard



The project dashboard includes several Widgets. Each Widget displays various information about the project.

For example, the "Performances" Widget is used to start the performance profiler or to open the last performance report.

The "Static audit and Compilation" Widget is used to quickly see whether the project contains compilation errors.

You have the ability to enable (or not) a Widget, to add one or to delete one at any time.

Automatic tests

One of the most interesting features of the dashboard is to give information about the tests that have been run on the application.

Several tests have already been run while developing our application.

The automatic tests are a category of specific tests. The automatic tests are used to automatically perform some operations on your windows. These tests are recorded as WLanguage scenarios and they can be easily modified in the code editor. Once recorded, the automatic test can be re-run as many times as necessary, to test for example the impact of a modification made to a window, a procedure, ...

Let's give it a try! We are going to create an automatic test on the edit window that was created at the beginning of this lesson.

Remark

The automatic tests can be run on the windows, procedures, classes.

▸ To create an automatic test on a window, all you have to do is run the test of the window:

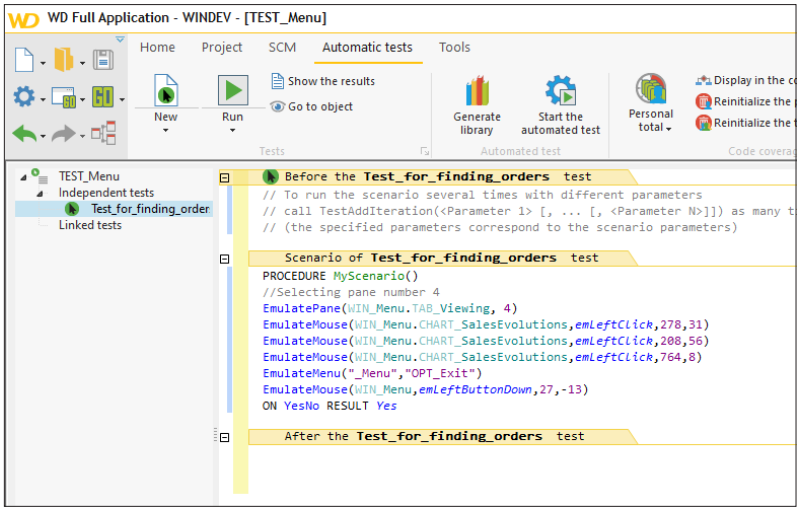
1. Open the "WIN_Menu" window in the editor.
2. On the "Automatic tests" pane, in the "Tests" group, expand "New" and select "Record a new scenario".
3. In the window that is displayed, click "Start recording".
4. The test of the window is run.
5. Click the "Finding orders" tab pane.
6. Choose the "Credit card" payment mode.
7. Click "Find".
8. Close the window via "Menu .. Exit" and confirm the end of application.

Remark

This option may not be available if you did not follow the entire tutorial. In this case, click the "x" to close the window.

9. The recording of the automatic test has ended.
10. The test editor proposes to save a description for the automatic test we just created. Enter the name of the automatic test: "Test for finding orders". Validate.

11. The test editor displays the WLanguage code of the test that was just saved:



The test is available and it is not passed (icon in front of the test name in the editor).

12. The test can be run at any time:

- Select the test name in the test editor.
- In the popup menu (right mouse click), select "Run".

13. The test is automatically run and the test editor displays the result in the "Test result" pane (the test was successfully run in our case).

14. Save the test if necessary.

Remark

The WLanguage functions used to run automatic tests are the EmulateXXX functions.

► We are now going to perform a modification in our window and to run the test again. The modification affects the "Find" Button control. We are going to gray it.

1. Display the "WIN_Menu" window if necessary.

2. In the editor, click the "Finding orders" tab pane and select the "Find" Button control.

3. Display the control description ("Description" from the popup menu).

4. In the "UI" tab, select "Grayed".

5. Validate the control description window.

6. Save the window.

7. Go back to the test editor (click the corresponding button in the open documents bar).

8. Run the test again.

9. The test appears again in the test editor. The "Compilation errors" tab pane reports several test errors.

10. Redisplay the "WIN_Menu" window in the editor.

11. Display the description of the "Find" Button control ("Description" from the popup menu).

12. In the "UI" tab, select "Enabled".

13. Validate the control description window.

14. Go back to the test editor (click the corresponding button in the open documents bar).

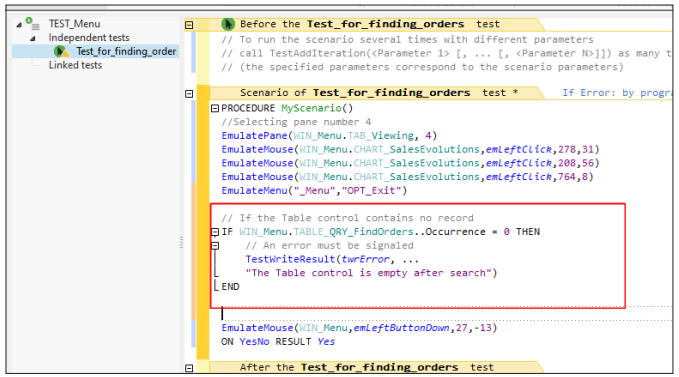
15. Run the test again. The test is now successfully run.

► Another feature of the test editor is the ability to modify or add WLanguage code in the test scenarios. We are going to add code lines to check whether the Table control contains at least one record.

1. Display the test editor if necessary (for example, double-click "TEST_Menu" in the "Tests" folder of the "Project explorer" pane).

2. Add the following code before the line "EmulateMenu("_Menu","OPT_Exit")":

```
// If the Table control contains no record
IF WIN_Menu.TABLE_QRY_FindOrders..Count = 0 THEN
  // An error must be signaled
  TestWriteResult(twrError, ...
       "The Table control is empty after search")
END
```



3. Save the test.

4. Click the icon. The test is successfully run. Indeed, the Table control contains at least one record.

The test editor proposes several features that will not be presented in this tutorial:

- the ability to use a set of test values.
- the definition of input and output parameters for the test.
- the ability to create a test library to run the test of an executable on a computer other than the development computer for example.

See the online help for more details (keyword: "Automatic test").

► Close the editor of automatic tests.

Lesson 4.14. Deploying the application

This lesson will teach you the following concepts

- Creating the executable.
- Help about the new features.
- Creating the setup.

Estimated time: 30 mn

Overview

A full application was created, allowing you to discover several WINDEV features. We must now generate the executable and install the application on the user computers. That's what we are going to do now. So, you will be familiar with the main topics for developing a WINDEV application.

Answer

If you did not perform the operations in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)"

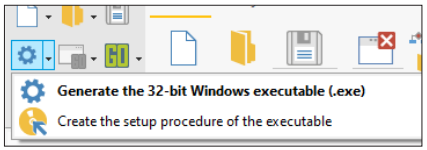
Creating the executable

Creating the executable is child's play: a menu option and a few mouse clicks are sufficient.

We will now present in detail the different steps for creating the executable. You can click the "Finish" button to validate all the steps of the wizard at any time.

▸ To create the executable:

1. Expand among the quick access buttons, and select "Generate the 32-bit Windows executable (.exe)".



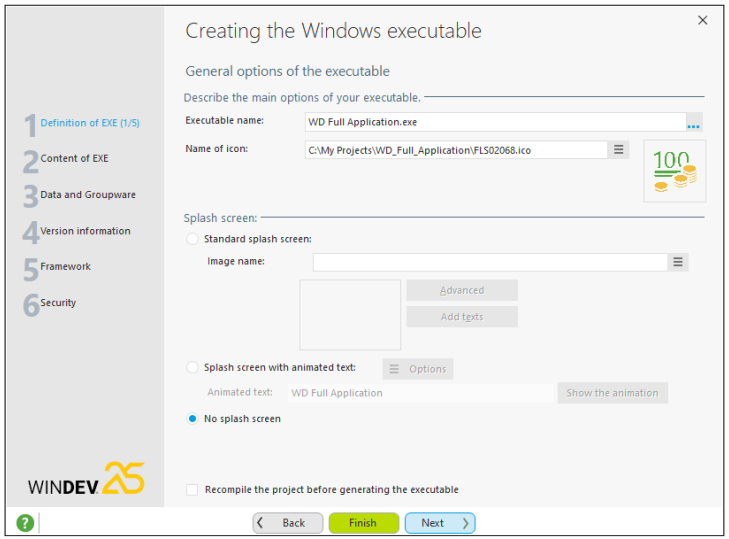
Remark

WINDEV also allows you to create 64-bit executables, Linux executables, services, Java applications, ...

2. The wizard for creating an executable starts.

3. Describe the general options of executable:

- the executable name: we will keep the default name.
- the name of icon associated with the executable: you have the ability to select an icon in the image catalog.
- the splash screen. Several types of splash screens are available. A splash screen with animated text is selected by default. The "Options" button is used to configure it.



Choose "No splash screen" and go to the next step.

4. For the operating mode of executable, we will keep the default options. Go to the next step.

5. You have the ability to customize the error message of application. We will keep the message proposed by default. Go to the next step.

6. The wizard proposes to enable the telemetry in the application. We won't be using this feature. Go to the next step.

Remark

Telemetry allows you to get detailed statistics about the use of your application by the end users. Telemetry returns various information about your deployed applications, giving you the ability to improve them.

See the online help for more details about implementing and configuring the telemetry.

7. This step is used to specify whether the executable will take the patches into account.

Remark

When a modification is performed in the application, to avoid having to provide the entire executable, the additional resources (windows, reports, ...) can be supplied as patches. These patches are additional libraries.

If the option "Yes: the executable will take these updates by patch into account" was checked when the executable was created, the elements found in the patch will replace the elements found in the application library when the application starts.

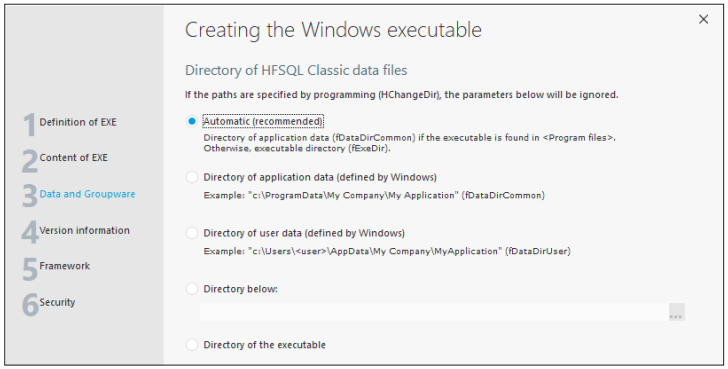
See the online help for more details.

We will keep the default options. Go to the next step.

8. This step is used to manage the executable languages. The multilingual feature will be presented in another lesson. We will keep the default options. Go to the next step.

9. This step displays all files that will be included in the executable library. Those are the project elements that can be handled by the end user. We will keep the default options. Go to the next step.

10. This step concerns the directory of HFSQL Classic data files used by the application.

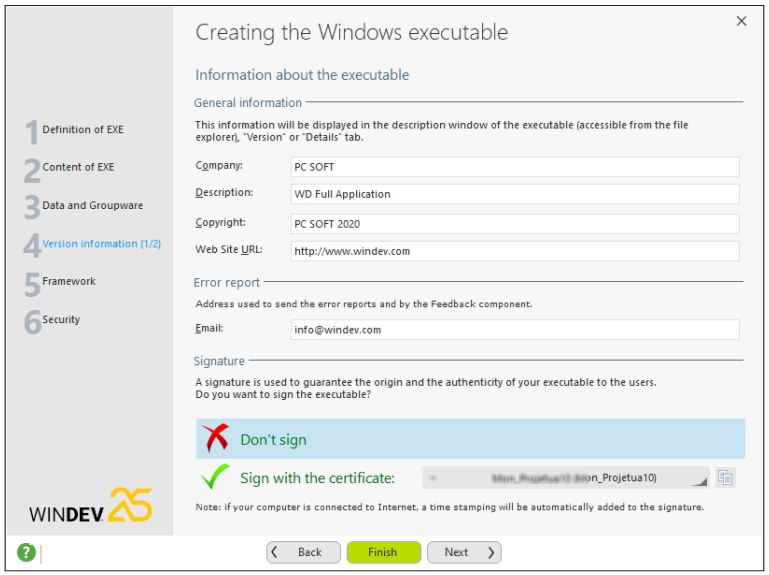


Select "Automatic (recommended)" if necessary. This option is used to install the data files:

- in the data directory of application (if the executable is installed in "Program files").
- in the executable directory.

11. Go to the next step.

12. We are now going to type the information about the executable. This information is displayed in the Windows explorer by selecting the file properties.
Type the following information:



This step can also be used to sign the executable if necessary.
Go to the next step.

13. This step concerns the version number of executable. You can use:

- a format compatible with the earlier WINDEV versions,
- a standard Windows format. Select this option.

14. Go to the next step of the wizard.

15. We are now going to define the options for integrating the WINDEV framework.

> **Remark**
> The framework corresponds to the different libraries required for the executable to operate.

The option "Framework DLL beside the executable" allows you to use the necessary libraries only. These DLLs are copied into the executable directory.
The option "Framework included in the executable" allows you to distribute the executable only (the DLLs being found in the executable).
The option "Common framework" allows you to use the framework common to all the WINDEV applications installed on the computer. With this option, the framework is installed once only on the computer (it can also be downloaded by Internet) and it is used by all WINDEV applications.

16. Select "Common framework" and choose "Common WINDEV framework".
Go to the next step.

17. This step concerns Windows Vista (and later). You have the ability to include a manifest for a use in Windows Vista (and later).
For our example, check "Don't include a manifest for Windows Vista and later".
Go to the next step.

18. WINDEV proposes to perform a project backup. This backup is used to keep a project version whenever the executable is created. In our case, select "No: Don't make a backup copy of the project" and validate the wizard.

19. The executable is created. It can be run immediately to check its operating mode. To do so, click the "Run the executable" button.

That's it, the executable creation is ended. A lot of options to fill but after a first configuration of your choices, you will be able to validate all wizard steps from the beginning.

> **Remark**
> You also have the ability to click the steps specified in the wizard in order to reach a wizard screen directly. The default options of other screens will be automatically validated.

Creating the setup

The setup is created via a wizard. This wizard allows you to define the main choices. You also have the ability to use the setup editor if the options proposed by the wizard are not suitable. We will not see its use in detail in this lesson. See the online help for more details.
We will now present the different steps for creating the setup program. You can click the "Finish" button to validate all the steps of the wizard at any time.

▸ To create the setup program:

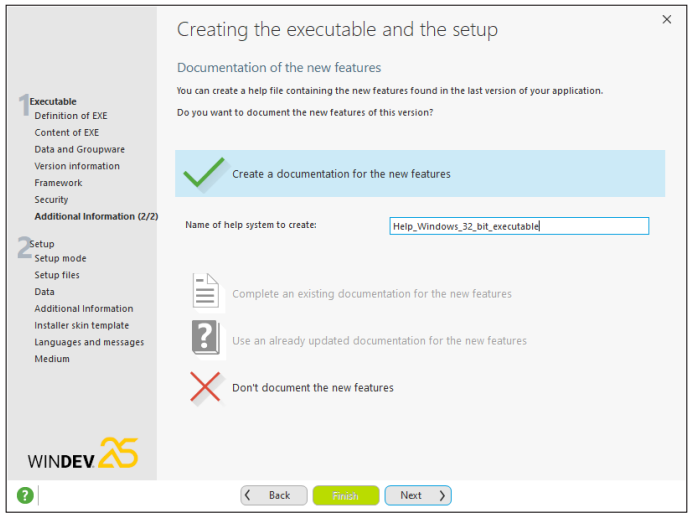
1. On the "Project" pane, in the "Generation" group, expand "Setup procedure" and select "Create the setup procedure". The wizard for creating the executable and the setup starts.

2. The executable was already created: the selected options are stored.

3. In the wizard, click on "Additional information" in the "Executable" section.

4. The wizard proposes to perform a project backup. Select "No: Don't make a backup copy of the project" and go to the next step.

5. The wizard proposes to create the page of new features.

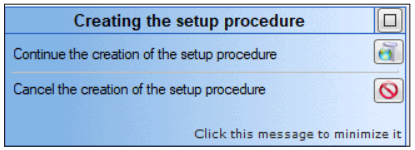


This option is used to create a help file in order to present the new features to the end users. During a first setup, this file can correspond to the software help.
Select "Create a documentation for the new features" and go to the next step.

6. You have the ability to define the elements that will be automatically included in the help file. Keep the default options and go to the next step.

> **Remark**
>
> If the application already contains a help system, it can be used in order to include a page for the new features.

7. Validate the message. The executable is automatically created (with the options defined when creating the executable) as well as the help system. You now have the ability to type specific information in the help system. In the example, leave the information created by default.
To resume the creation of the setup program, click , which corresponds to "Continue the creation of the setup procedure".



The help is automatically compiled.

8. Let's now go to the "Setup" section of the wizard. In this first screen, choose the setup mode:
   - Individual setup for an independent application, installed and started on each computer. We will choose this option.
   - Setup with automatic update, for a setup on a server. The applications will be installed from the server. In case of update, only the server must be updated. The applications installed on the computers will be automatically updated.

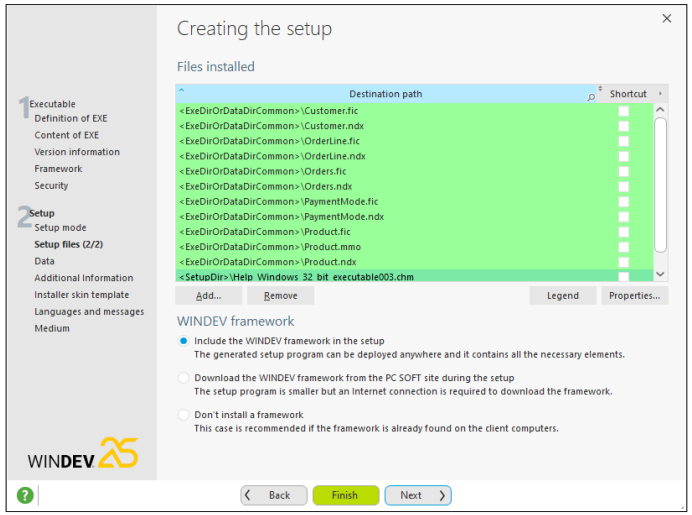
   Go to the next step.

9. Choose a standard setup. Go to the next step.

10. We will not change the default setup directory. Go to the next step.

11. Keep the files proposed in the list of installed files. By default, WINDEV proposes the Executable file and the help file (created by the help of new features). We are going to add the data files:
   - Click the "Add" button. The Windows explorer displays the content of the generation directory of application.
   - Select the Customer, Orders, OrderLine, PaymentMode and Product data files (files with ".fic", ".ndx" and ".mmo" extension).

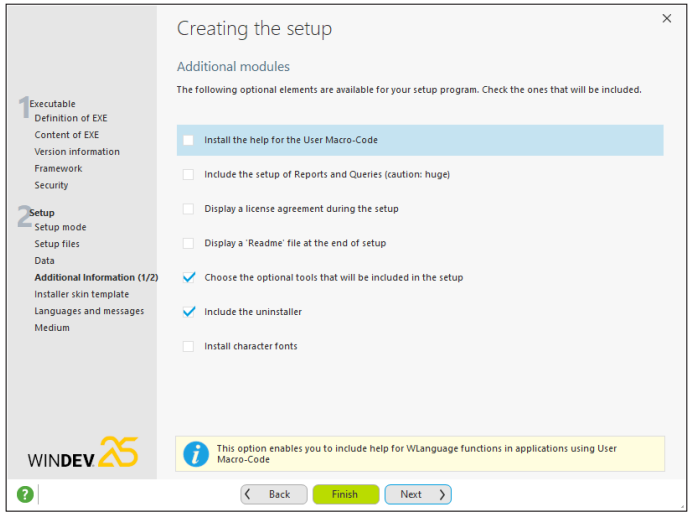
   - Validate. The files are automatically positioned in the data directory of application.



12. Go to the next step.

13. Uncheck all options from the database parameters. Go to the next step.

14. Keep the following options in the additional modules:



We will choose the optional tools to install and we will include the uninstaller. The uninstaller will allow the users to uninstall the application from the Windows program manager. Go to the next step.

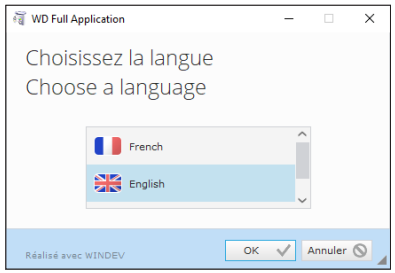
15. We will select WDOptimizer. This tool is used to optimize the data files of application on a regular basis.
16. Click "Medium" on the left of wizard. The setup will be generated in a single directory. By default, the setup program is created in the "Install" subdirectory of project. You can:
    - Specify a password for the setup. In this case, only the user who knows the password will be able to install the application.
    - Sign the setup with a certificate. This option is used to ensure the integrity of setup pack.
17. Validate. The setup program is automatically created.
18. A screen is displayed, allowing you to check the setup or to open the generation directory.

Installing an application

In the window for the end of setup creation, click the "Test" button. If this window was closed, run the "Install.exe" program found in the directory of setup program. The application setup starts.

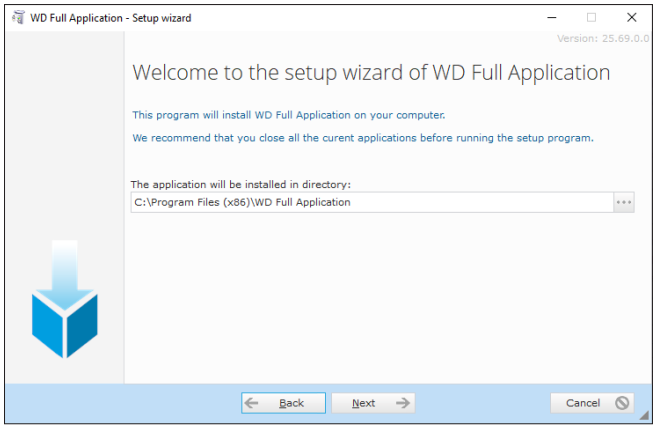
▶ We are going to follow the different steps of setup program for the "WD Full Application" application.

1. If you are using an operating system corresponding to Windows Vista (or later), a window requests the authorization to run the program. Validate this window.
2. Choose the language of setup program and click "OK".



3. The setup wizard starts. Two setup modes are available:
    - Quick setup: The setup is performed with the parameters specified by the application provider.
    - Custom setup: The wizard asks the user to define the parameters of application setup.

4. Choose "Custom setup". The wizard asks for the setup directory of application.



5. Validate the setup directory of application.
6. Go to the next step and end the application setup.
7. The application setup starts. Validate the different setup steps.

The different types of deployment

We have performed a simple application deployment. In the wizard for setup creation, WINDEV also proposes setups with automatic update.

Overview

Several setup modes are available for a WINDEV application:

- **Stand-alone setup**:
  This type of setup is used to create a unique setup program. This setup program will be run by the end user on his computer.
  To update the application, you will have to re-create a setup for the application. Then, the end user will have to install this new program.
  This is the type of setup that was just performed.
- **Setup with automatic update**:
  This type of setup is used to automatically detect the updates when starting the application. If an update is available, the user can perform this update immediately.
  This type of setup is available via the network or via the Web. You also have the ability to perform a multi-site setup.

We are now going to present the operating mode of a setup with update.

Setup with network update

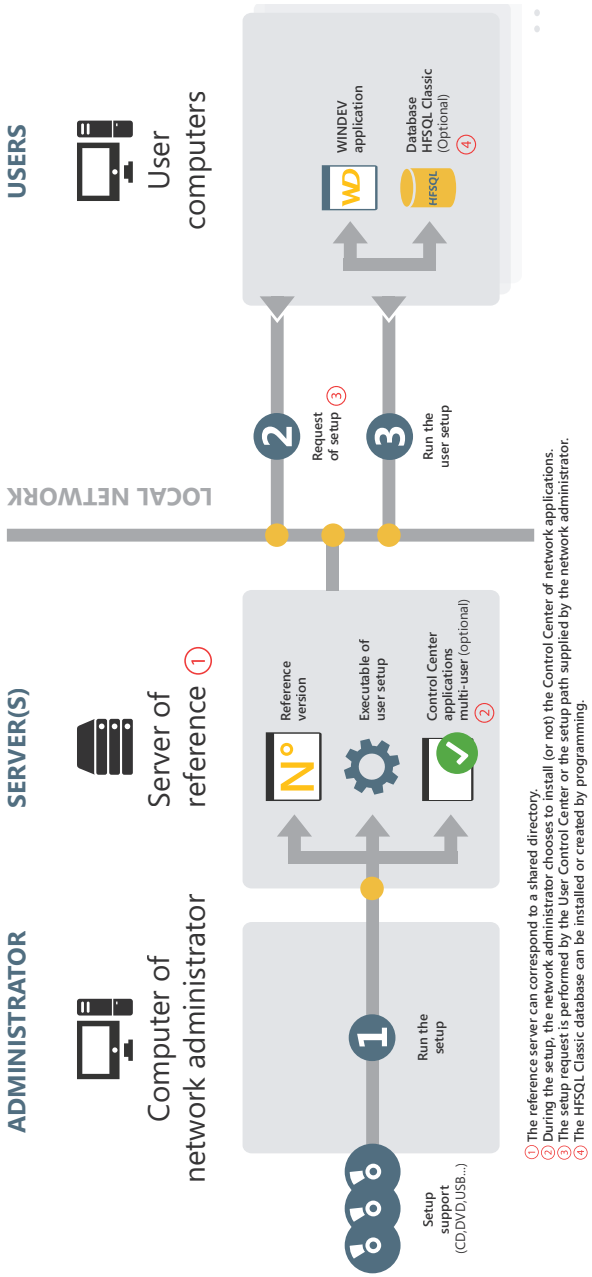
The creation of the setup program is performed via the wizard for creating the setup program ("Setup procedure" option in the "Project" pane).

The setup program obtained at the end of the wizard includes two setups:

- the setup of the reference application. The reference application must be installed on the network (in a shared directory or on a server for example), accessible to all the end users of the application.
- the application setup itself. This setup is included in the reference application. It can be accessed once the reference application is installed.

The diagram below presents the setup mode of an application that is using a local HFSQL Classic database, with a network update.

The version of reference application is automatically checked whenever the application is started by the end user. If this version was modified (if the reference version was updated for example), an update is automatically proposed for the final application.



Setup with Internet update

The same principle is used by the setup with Internet update.

The creation of the setup program is performed via the wizard for creating the setup program ("Setup procedure" option in the "Project" pane).

The setup program obtained at the end of the wizard includes:

- the setup of the reference application and the Web page used to download the client setup. These elements must be installed on an HTTP server.
- the application setup itself. This setup is included in the reference application. It can be accessed once the reference application is installed, via the Web page for download.

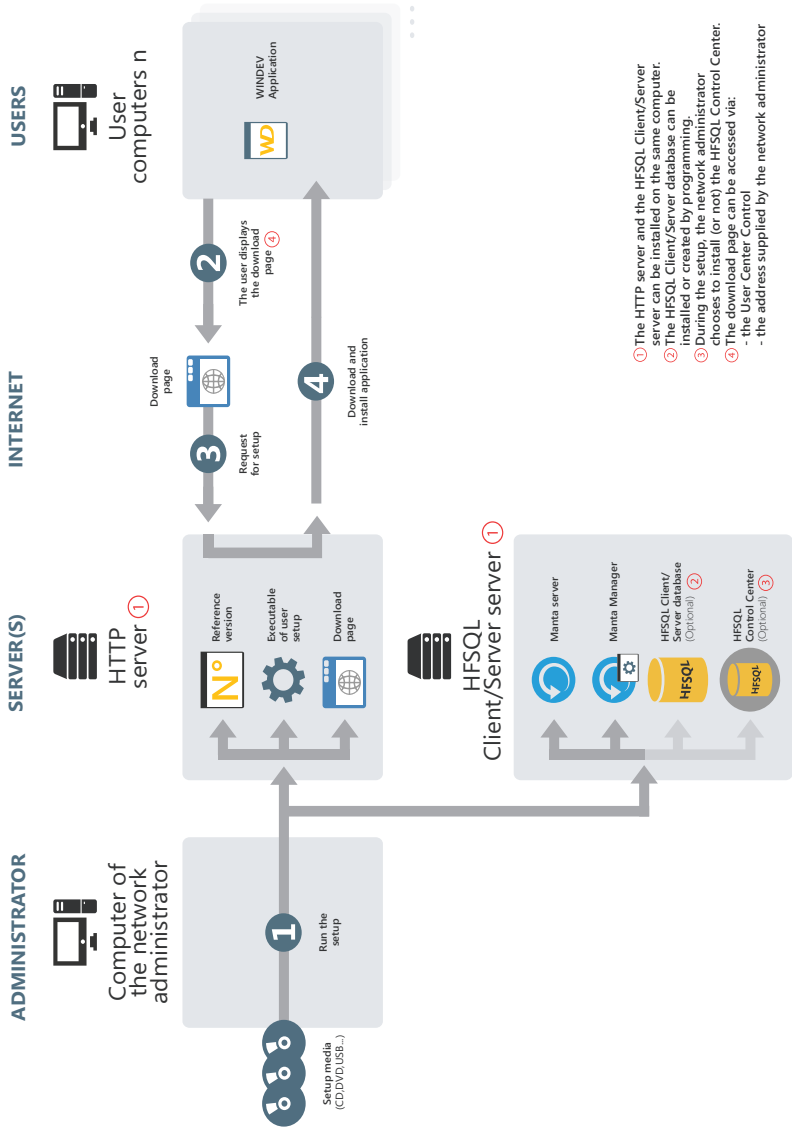
The diagram below presents the setup mode of an application that is using a HFSQL Client/Server database, with an update by Internet.

Remark: This type of setup can also be performed via PCSCloud (if you have an account).

Multisite setup

The multi-site setup combines:

- a reference network setup deployed on a local network.
- an HTTP setup used to update the reference setup on the local network.



Lesson 4.15. Distributing "Reports and Queries" with your applications

This lesson will teach you the following concepts

- What is "Reports & Queries" used for?
- How to distribute "Reports and Queries".
- How to use "Reports and Queries".

Estimated time: 10 mn

Overview of "Reports and Queries"

"Reports & Queries" allows the users to modify and create the reports and queries found in your applications.

The user thinks that some information is missing in the standard report? "Reports and Queries" allows the user to add this information from the report viewer. This modification may be useful for the other users? The report can be made available to all users who are using the application in network. Same for the queries.

Remark

"Reports and Queries" is free and it can be distributed with your applications developed with WINDEV. See the license for more details about the distribution and use conditions.

Starting "Reports and Queries"

In order for the end users to customize the reports of your application or to create their own reports and their own queries, all they have to do is start "Reports & Queries".

To allow the end users to start "Reports & Queries":

- display your reports in the report viewer. The end users will be able to start "Reports and Queries" by clicking  .
- add the automatic help menu '?' into the main window of your application: on the "Window" pane, in the "Bars and menus" group, expand "Main menu" and select "Add the '?' menu". In the wizard for creating this menu, check "Create, modify a report" and "Create, modify a query".
- use ***RunReportsAndQueries*** in your application.

Distributing "Reports and Queries" with your applications

To distribute "Reports and Queries" with your own WINDEV application, you must:

- specify in the project that the application allows "Reports and Queries" to be run.
- in the analysis, define (if necessary) the files and items that can be used in "Reports and Queries".
- define the reports and queries that can be modified in "Reports and Queries".
- create the executable and the setup program including "Reports and Queries".

We are going to present these different steps in details by using the "WD Full Application" project.

Answer

If you did not perform the operations in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

Configuring project

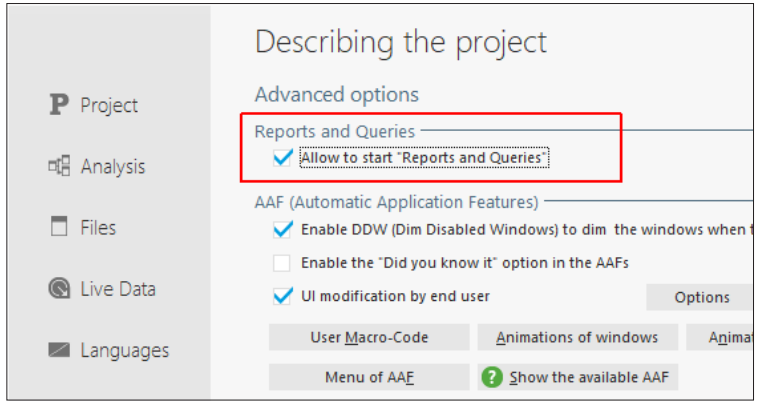
▸ To use "Reports and Queries" from your application, you must configure the project associated with your application.

1. Open (if necessary) the "WD Full Application" example: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full Application (Answer)". The project is loaded.

Remark

If no report was created in the "WD Full Application" project, open the corrected application project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

2. Display the project description.
Reminder: To display the project description, go to the "Project" pane, "Project" group, and click on "Description".
3. In the "Advanced" tab, check "Allow to start "Reports and Queries"".



4. Validate.

Configuring analysis

We are going to open the analysis in order to configure the files and items that can be used in "Reports and Queries".

▸ Open the analysis of your project: click among the quick access buttons of WINDEV menu.

By default, all data files and all items are visible and usable in "Reports & Queries". If your application contains sensitive information (passwords, ...), these data files or these items can be made invisible in "Reports and Queries".

Remark

"Reports & Queries" does not allow the user to add, modify or delete records (***HAdd***, ***HModify*** and ***HDelete*** are not allowed). It is also not possible to create or re-create data files (***HCreation*** and ***HCreationIfNotFound*** are not allowed).

▸ To define an invisible data file in "Reports and Queries":

1. In the data model editor, select the requested data file.
2. Display the description window of the data file ("Description of data file" in the popup menu)
3. In the "Reports and Queries" tab, uncheck "Visible by the end user in "Reports and Queries"".
4. Validate.

▸ To define in invisible item in "Reports and Queries":

1. In the data model editor, select the requested data file.
2. Display the file description window ("Description of items" in the popup menu).
3. Select the requested item in the table.
4. In the "Reports and Queries" tab, uncheck "Visible by the end user in "Reports and Queries"".
5. Validate.

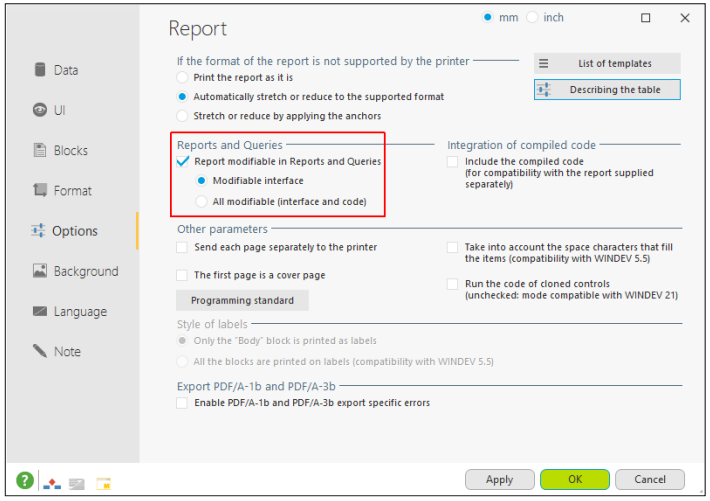
▸ To take into account the modifications performed in the analysis, you must regenerate the analysis (on the "Analysis" pane, in the "Analysis" group, click on "Generation"). An automatic modification of data files is proposed to take the modifications into account.

Configuring reports

When creating a report, you have the ability to specify whether this report can be modified in "Reports and Queries".

▸ To allow the "RPT_QRY_FindOrders_table" report (found in the "WD Full Application" project) to be modified in "Reports & Queries":

1. Open the "RPT_QRY_FindOrders_table" report in the report editor.
2. Display the report description ("Report description" from the popup menu).
3. In the "Options" tab, check "Report modifiable in Reports and Queries".
4. Next, check one of the following options:
   - "Modifiable interface" for the report interface to be modifiable in Reports and Queries.
   - "All modifiable (interface and code)" for the entire report to be modifiable in Reports and Queries.



5. Validate and save the report.

Configuring queries

By default, a query can be modified in "Reports and Queries".

▸ To allow the "QRY_OrderForm" query (found in the "WD Full Application" project) to be modified in "Reports & Queries":

1. Open the "QRY_OrderForm" query in the query editor.
2. Display the query description ("Query description" from the popup menu of query graph).
3. Click the "Advanced" button (found among the "Actions").
4. In the "Reports and Queries" tab, check "Visible in 'Reports Queries'".
5. Validate and save the query.

Creating the executable and distributing the application

The method for creating and distributing an application that is using "Reports and Queries" is the same as the method for creating and distributing a standard application. You can specify the parameters specific to the "Reports and Queries" program when creating the setup procedure.

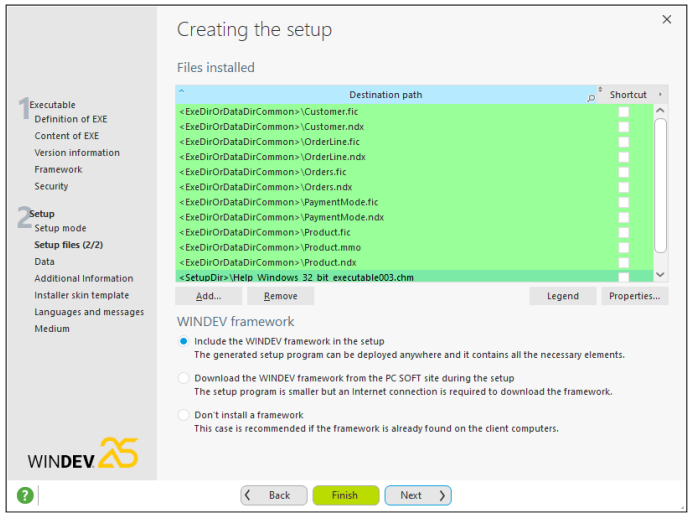
▸ To create the setup program of your WD Full Application application:

1. On the "Project" pane, in the "Generation" group, click "Setup procedure". The wizard for setup creation starts. Click the "Setup" link on the left of wizard.
2. Select "Create the executable now". The executable is created.

Remark

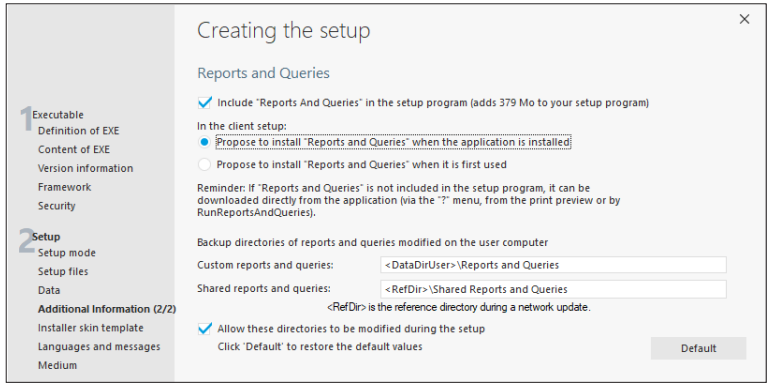
If the wizard proposes to create the help file, validate the different messages.

3. The wizard for setup creation starts.
4. Choose an "Individual setup". Go to the next step.
5. Choose a "Standard setup". Go to the next step.
6. Validate the proposed setup directory and go to the next step.
7. In the screen named "Files installed", select (if necessary) the data files (.fic, .mmo and .ndx) found in the executable directory. To do so, click "Add" and select these files.



8. Click on "Additional information" on the left.
9. In the screen "Additional modules" screen, make sure that "Include the setup of Reports and Queries (caution: huge)" is checked.
10. Go to the next step.
11. In the step entitled "Reports and Queries":
    - Specify whether the setup of "Reports and Queries" must be included in your setup program.
      In this case, the setup package of "Reports and Queries" (located in the subdirectory "Reports and Queries" of the WINDEV setup directory) will be used.
      You can:
      - Propose to install "Reports and Queries" with the application.
      - Propose to install "Reports and Queries" when it is first used.

      Caution: Including "Reports and Queries" significantly increases the size of your setup (about 350 MB).



- Specify (if necessary) the directories corresponding to the reports and queries modified or created by the end user:
  - The directory of custom reports and queries corresponds to the directory of reports and queries visible by the user who created them.
  - The directory of shared reports and queries corresponds to the directory of reports and queries visible by all application users.

  These directories can be modified when installing the application: all you have to do is check "Allow these directories to be modified during the setup".

12. Click "Medium" on the left of wizard. This step allows you to select the setup media, the directory for creating the setup program if necessary, and also to protect the setup with a password.

13. Keep the default options and validate the wizard. The setup program is generated. Don't close the window indicating the end of setup creation.

Installing and using "Reports and Queries"

Installing the application

To check the setup and use of "Reports and Queries", we are going to install the "WD Full Application" application.

The setup program of this application was generated in the previous paragraph. The setup program was created in the "Install" subdirectory of the current project.

▸ In the window for the end of setup creation, click the "Test" button. If this window was closed, run the "Install.exe" program found in the directory of setup program. The application setup starts.

▸ We are going to follow the different steps of setup program for the "WD Full Application" application.

1. If UAC is enabled, a window appears and requests authorization to run the program. Validate this window.

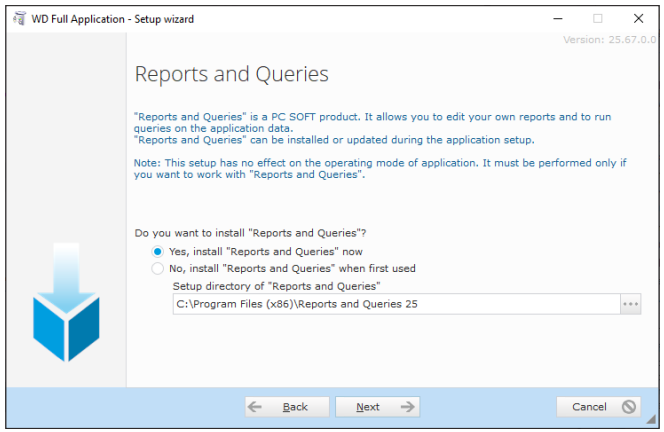
2. Choose the language of setup program and click "OK".

3. The setup wizard starts. Two setup modes are available:
- Quick setup: The setup is performed with the parameters specified by the application provider.
- Custom setup: The wizard asks the user to define the parameters of application setup.

4. Choose "Custom setup". The wizard asks for the setup directory of application.

5. Validate the setup directory of application.

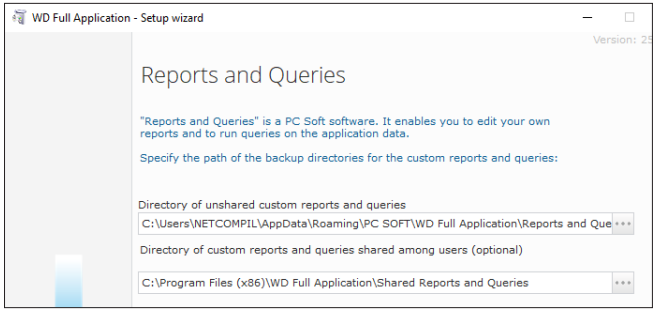
6. The screen named "Reports and Queries" is used to install "Reports and Queries" (only if this program is not already installed on the current computer).



Remark: If an earlier version of "Reports and Queries" is installed on the computer, it is possible to make a backup copy of that version.

7. Go to the next step.

8. The next screen allows you to define the directories that will be used to save the reports and queries (if the option "Allow these directories to be modified during the setup" was checked in the wizard for creating the setup program).



9. Go to the next step and end the application setup.

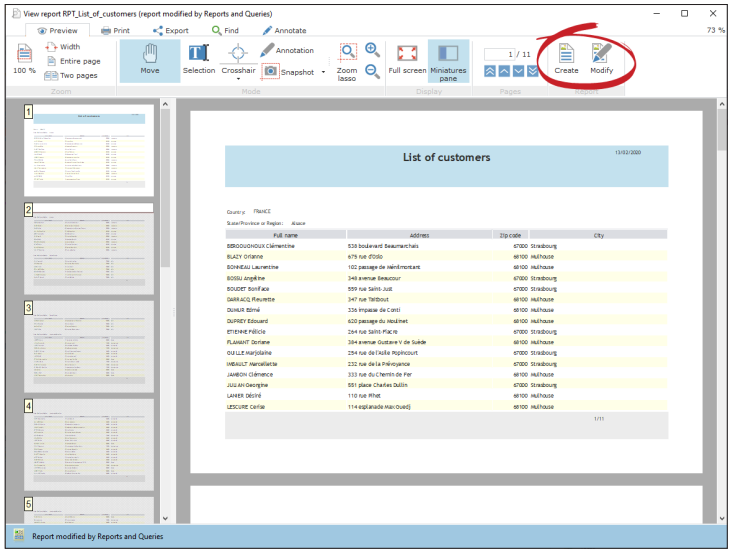
10. The application setup, then the setup of "Reports and Queries" start.

Application test

We are now going to take the place of the application user (not easy, but worth a try).

► To run the application test:

1. Start the "WD Full Application" application.
2. Select "Reports .. List of customers". The report is displayed in the report viewer. Click the "Preview" pane.



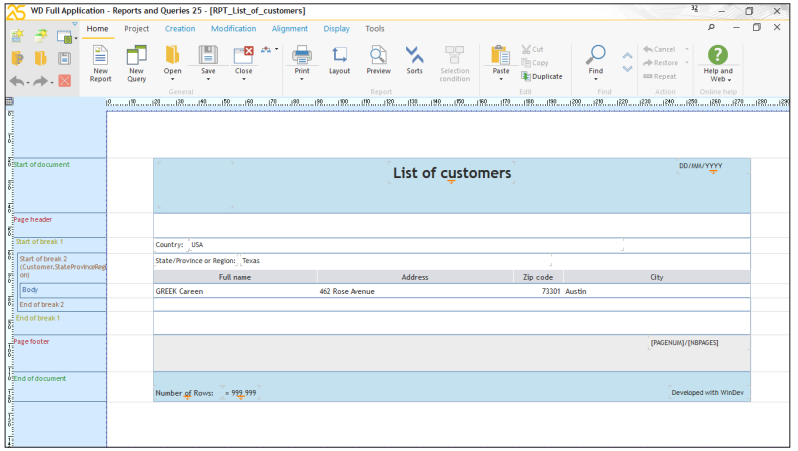
3. Two new icons are displayed on this screen:

 Used to modify the current report in the report viewer.

 Used to create a new report.

► We are going to modify this report:

1. Click  .
2. The report is displayed in edit mode in "Reports and Queries".



3. The user can perform the requested modifications:
   - on the style of the displayed information (change the color of a control for example).
   - on the content of the report (add a control for example). If the report is linked to a query, the user can modify this query.
     - If the report is linked to an embedded query, the modification is directly performed in the report.
     - If the report is linked to an independent query, the query is also modified. The modification will be taken into account when running the report (if a window is also using this query, the modification will not be taken into account by the window).

► In this example, we want to highlight the date in red. To do so:

1. Select the date control in the start of document.
2. On the "Modification" pane, in the "Quick edit" group, click the "Color" icon and select the red color.
3. Save your report. The directory proposed by default corresponds to the setup directory of your application.
4. Close "Reports and Queries".
5. Select "Reports .. List of customers" in the application. The report is displayed in "Report viewer" and the modification is applied.

In this example, the modified report is available to you only. For a network application, the user has the ability to make a modified report available to all the users.

You are now familiar with "Reports & Queries".
See the online help for more details (keyword: Reports and Queries).

Conclusion

The development step of our application is ended. The following lessons will allow you to discover:
- how to transform your application into a multilingual application,
- how to include your application in the Source Code Manager (SCM).

Lesson 4.16. Managing multiple languages

This lesson will teach you the following concepts

- What is a multilingual application?
- Creating a multilingual application, step by step.

Estimated time: 20 mn

What is a multilingual application?

A multilingual application is an application that can be run in different languages (English, French, German or any other language).
Therefore, the same application can be used in several languages. But how is it possible? That's what we will see in this lesson.
We are going to handle a project that can be run in English or in French, according to the user's choice.

The main steps of a multilingual application are:
- Choosing the project languages.
- Localizing the analysis.
- Localizing the project elements (windows, reports, controls, help system, ...).
- Localizing the messages found in the code.
- Programming the change of language in the application.

These different steps will be applied to the "WD Full Application" project. This project, available in English, will be translated into French.

Answer

If you did not perform the operations in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

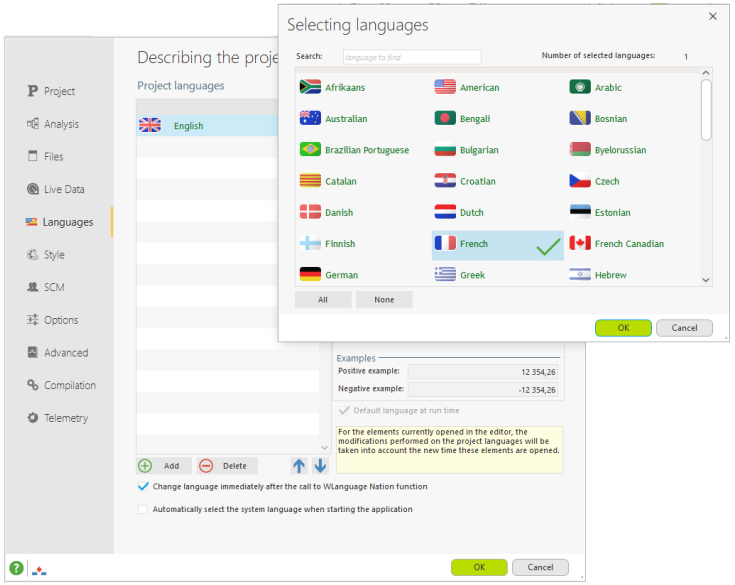
Choosing the project languages

▶ The first operation consists in choosing the project languages.

1. Display the project description: on the "Project" pane, in the "Project" group, click "Description".

2. Click the "Languages" tab. Our application will support English and French.

3. Click the "Add" button. The window for selecting languages is displayed.



4. Click "French". A checkmark is displayed on the right of language.

5. Validate. The "French" language appears in the list of project languages.

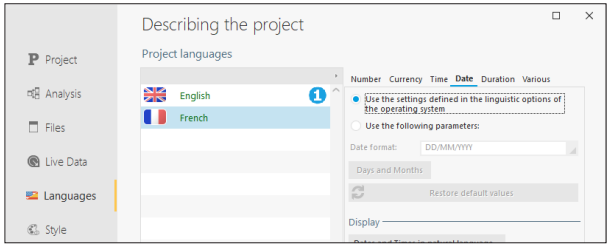
▶ The "Languages" tab can also be used to configure the linguistic options regarding the numbers, the currencies, the dates, ... for the selected language. Let's see an example:

1. Click the "French" language.

2. Select the "Date" tab.

3. The linguistic options of Windows are used by default. Select "Use the following parameters": you now have the ability to define the date format as well as the translation used for the days and months.

4. Keep the option "Use the parameters defined in the linguistic options of the operating system".



Remark

In the linguistic options, you have the ability to choose the text direction for the language ("Various" tab, "Text direction" option). This allows you to create interfaces with a language written from right to left.

► Validate. A message proposes to synchronize the different project elements. Answer "Yes".
All project elements opened in the editor (windows, reports, ...) are closed and the additional languages are added to these elements.
Remark: The captions that exist in the base project language are automatically copied into the added languages.

Localizing the analysis

By default, an analysis is created in a specific language and it cannot be translated.

However, some information can be typed in several languages (notes in the documentation, shared information, ...).

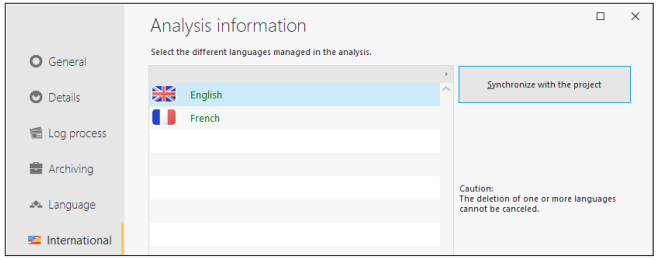
If your application uses Reports and Queries, the names of the data files and items can also be translated. This translation can be done in the "Reports and Queries" tab (this tab is both in the data file and the item description window).

By default, the controls created from the analysis items have the item caption specified in the analysis. If a caption was specified in the shared information of item, this caption will be used when creating the control.

When changing the language of project linked to the analysis, this change is not automatically applied to the analysis. Indeed, an analysis can be shared between several projects.

► To support several languages in an analysis:

1. Display the data model editor: click among the quick access buttons of WINDEV menu.
2. In the analysis description ("Analysis description" from the popup menu), select the "International" tab.
3. The list of languages supported by the analysis is displayed. The French language not being supported:
   - Click the "Add" button.
   - Select "French".
   - Validate the window for adding languages.



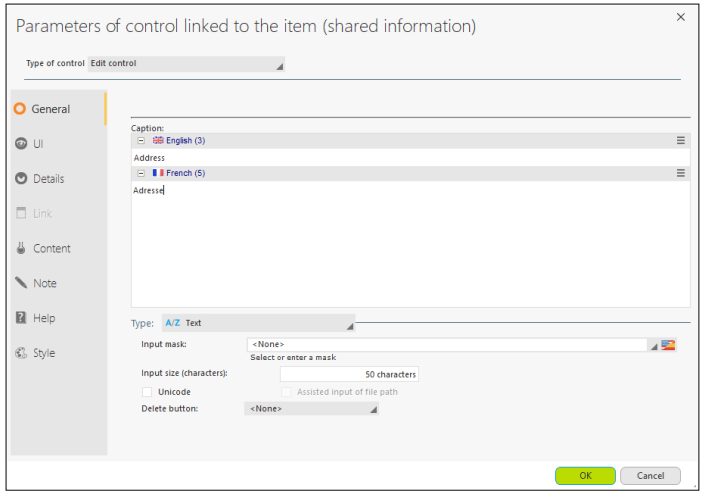
4. Validate the description window of analysis.

Remark

To take into account all languages of the project linked to the current analysis, you also have the ability to click the "Synchronize with the project" button.

► To type shared information in several languages, let's take a simple case: entering the caption of the control linked to the "Address" item of "Customer" data file:

1. Select the Customer data file.
2. Display the description of items of a data file ("Description of items" in the data file popup menu).
3. Select the "Address" item and display the parameters of control linked to the selected item (shared information). To do so, click the link at the bottom of the screen. The shared information is displayed in a new window.
4. In the "Caption" area, type the caption for the French language: "Adresse".



5. Validate the window for typing shared information.
6. Validate the description window of items.

Remark

The shared information of analysis can be translated:
- when creating the analysis.
- at any time via the editor.
- at any time via WDMSG and WDTRAD, tools used to extract, translate and reintegrate the different elements.

► To take into account the modifications performed in the analysis, you must generate the analysis: on the "Analysis" pane of the ribbon, in the "Analysis" group, click "Generation".

Localizing the project elements

All project elements can become multilingual elements: windows, reports, help, ...

We are going to modify some elements of WIN_Menu window to present the different methods that can be used.

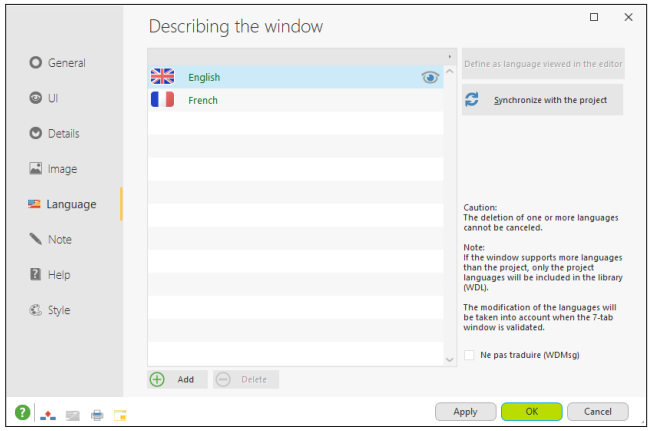
We are going to modify:

- the image of the bar used in the WIN_Menu window.
- the captions of controls found in the WIN_Menu window.
- the menu options.
- the message displayed by the WLanguage code when closing the application.

▶ Open the "WIN_Menu" window in the editor (double-click its name in the "Project explorer" pane for example).

▶ First of all, check whether the WIN_Menu window is associated with the different languages defined in the project:

1. Display the window description ("Description" from the popup menu of window).
2. Select the "Language" tab: the two languages selected in the project are displayed.

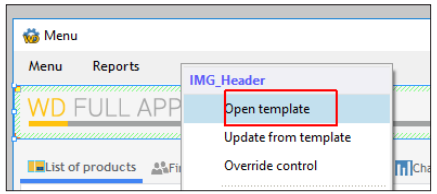


3. Validate the window.

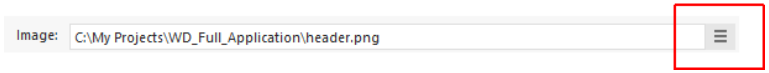
Localizing an image

▶ To change the image of the bar used in the WIN_Menu window according to the runtime language:

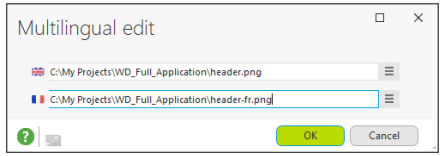
1. Open the "WIN_Menu.WDW" window.
2. The bar image is found in the template associated with the window. Therefore, the associated window template must be opened:
   - Click the "WD FullApplication" control and display the popup menu.
   - Select "Open the template".



   - The window template appears, enclosed in an orange border.
3. Display the description window of the bar:
   - Select the "WD FullApplication" control and display the popup menu.
   - Select "Description".
4. In the "Image" area, click the ≡ button.



5. Select "Multilingual" from the popup menu that is displayed.
6. The window for managing multilingual images is displayed. A different image can be used for each language. This feature is very useful if you are using images containing text.



7. For our example, in the "French" area, select the "header-fr.png" file:
   - Click the ≡ icon.
   - Select "Browse".
   - Select the requested file.
8. Validate. The "Multilingual value" caption appears in the "Image" area of description window.
9. Validate the control description window.
10. Save the window template (or Ctrl + S).
11. Update the windows that use the window template by clicking the icon in the orange bar. Validate the update window.
12. Close the window template displayed in the editor.

Localizing controls

A control can display various information to the user:
- a caption,
- a help message,
- an image, ...

This information must be translated. This information is accessible in the different tabs of the description window of control.

▸ For our example, we are going to translate the first pane of Tab control.

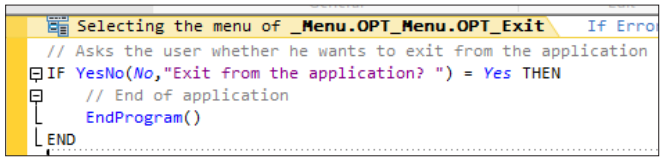
1. Double-click the Tab control to display its description window.
2. In the "General" tab of description window, select the first tab "List of products".
3. In the "Description of static pane" section, you can:
   - type the translation for the pane caption: "Liste des produits".
   - define a specific image for the tab pane according to the language (as already done before).
   - define a tooltip for each language.
4. Close the description window of control.

Localizing a programming message

All messages found in your program can be translated into several languages. In our example, selecting the "Exit" menu displays the message "Exit from the application". We are going to translate this message.

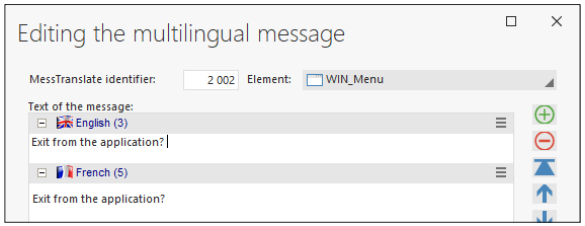
▸ To translate the message of menu option:

1. Display the code of menu option:
   - Expand the menu in the editor.
   - Select the "Exit" option.
   - Select "Code" from the popup menu (right mouse click).

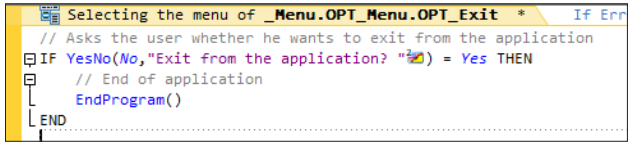


2. To translate this type of message, position the cursor in the "Exit from application?" string and press Ctrl + T. On the "Code" pane, in the "Languages" group, you can also expand "Translate strings" and select "Translate messages".

3. The following window is displayed:



4. This window allows you to translate all messages of your program into all project languages.
5. In the "French" area, type "Quitter l'application?" and validate.
6. The icon as well as a digit appear in the code editor.



These icons indicate that the multilingual message exists in 2 languages.

7. Close the code editor.

Localizing menus

The menu options can be translated like the other controls via the description window of the option, or from the window editor directly.

▸ To translate the menu of "WIN_Menu" window:

1. On the "Display" pane, in the "Options" group, expand "Language displayed" and select the language that will be viewed in the editor (French in our case).
2. The menu options are displayed in the selected language. If no translation corresponds to the selected language, the menu options are displayed in English.
3. Expand the "Menu" option.
4. Select "Send an email".
5. Press the Space key on the keyboard: the caption becomes editable.
6. Type the caption in French: "Envoyer un email" and validate.
7. Switch the displayed language back to English: on the "Display" pane, in the "Options" group, expand "Language displayed" and select "English".

The translation tools

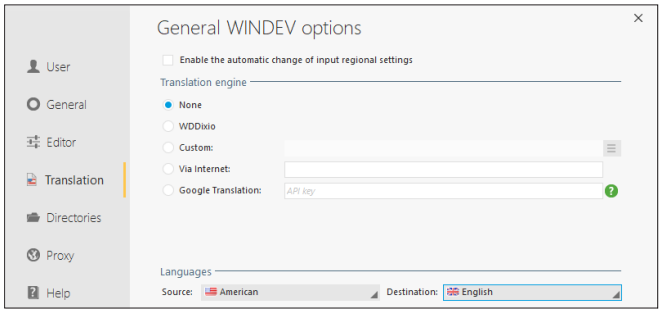
Some application elements have been translated manually.
Several methods can be used to translate this information:

- a direct translation of messages performed in the different editors. This translation can be performed via a translation tool, Google Translate (providing that you own a license), ...
- a translation performed via an external tool (WDMSG and WDTRAD).

Direct input of translations

The translations are typed in the product interface directly. For example, the caption of "New" button becomes "Nouveau" in French. All you have to do is open the description window of control and type the corresponding translation in the requested language.
If you want to use a translation software or a translation site, WINDEV can be configured to use this software:

1. On the "Home" pane, in the "Environment" group, expand "Options" and select "General options of WINDEV".
2. Display the "Translation" tab.



3. Specify:
   - Whether the regional settings must be automatically enabled according to the language used for the input. In this case, if the language requires a specific character set, this character set will be automatically selected.
   - The software or the site that will be used for translation. You have the ability to use WDDixio, translation dictionary supplied with WDMSG (see next paragraph), a specific translation software or site, or Google Translate.
     See the online help for more details.
   - The supported languages.
4. When the translation parameters are defined, you have the ability to click the button found in the different description windows of project elements: this button allows you to use the software defined for translation.

Translation with WDMSG and WDTRAD

A tool named **WDMSG** (not supplied with WINDEV) can be used to:

- check out all project messages (caption of controls, code message, title of windows, ...) in order to translate them,
- check the translated messages back in.

The messages to translate are checked out:

- in a text format that can be configured to be used by most translation tools
- in HFSQL format.

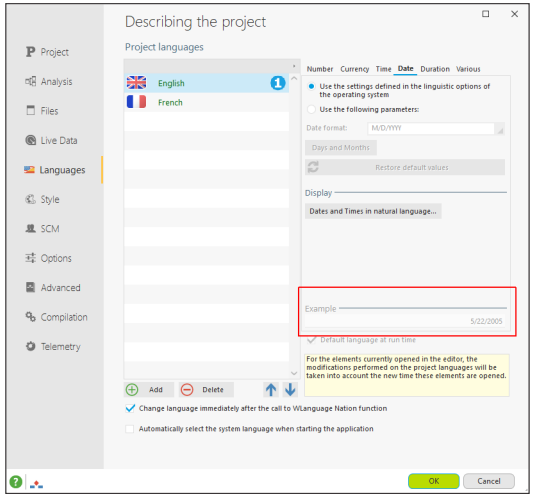
WDMSG is also supplied with WDTRAD, a computer-assisted translation tool. WDTRAD is used to easily type all translations for the multilingual information of a project.

Contact PC SOFT Sales Department for more details about **WDMSG** and **WDTRAD**.

Other elements to translate: the framework messages

Various information and messages are found in the WINDEV framework. For example, the names of days and months used by the functions for date management come from the WINDEV framework. To translate one or more libraries of this framework, you must use WDINT (not supplied with WINDEV).
This program is used to get a file whose extension is WDM. To use this file in your application:

- you have the ability to use ***LoadError***.
- you have the ability to include the file to the project description in the "Languages" tab. All you have to do is select the requested language and select the "Various" tab.



Contact PC SOFT Sales Department for more details about WDINT.

Programming the change of language

By default, the project is run in the runtime language defined for the project, in the "Languages" tab of project description ("Description" in the "Project" pane).

In an application, the language can be chosen via a menu option. You can change the language of the application being run using **Nation** in the event associated with the menu option.

Adding a menu option

▶ To add a menu option:

1. Open the "WIN_Menu" window in the editor if necessary (double-click its name in the "Project explorer" pane).
2. Click the "Menu" option of window. The menu is expanded.
3. Select "Send an email".
4. Display the popup menu (right click). Select "Add after". Type the caption ("Languages") and validate.
5. Select the "Languages"' option you have just created.
6. Display the popup menu (right mouse click) and select "Transform to expand a sub-menu".
7. Type the caption of first sub-option: "English".
8. Press the Enter key twice and type the caption of second option ("French").

We are now going to type the WLanguage code required to change language.

Programming

To type the code for managing languages:

1. Select "Menu .. Languages .. French" in the editor.
2. Display the popup menu (right click). Select "Code".
3. Write the following code:

```
Nation(nationFrench)
```

4. Select "Menu .. Languages .. English" in the editor.
5. Display the popup menu (right click). Select "Code".
6. Write the following code:

```
Nation(nationEnglish)
```

Nation is used to change the runtime language of application. The constants passed in parameter allow you to specify the language to use.

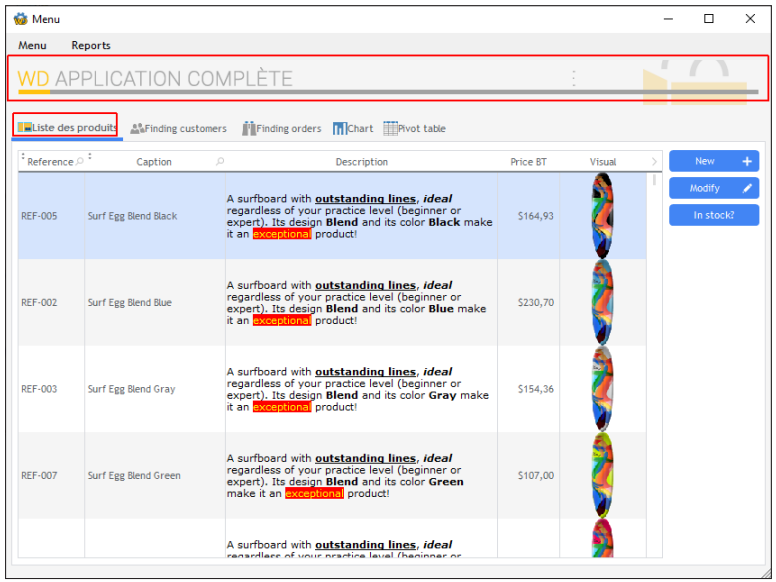
The change of language performed by ***Nation*** is immediately effective.

Project test

Some application elements being translated, we are now going to check the change of language.

▶ To run the application test:

1. Run the project test (GO among the quick access buttons). The window is displayed in test mode in English.
2. Select "Menu .. Languages .. French".
3. The elements that have been translated are displayed in French:



4. End the test and go back to the editor.

Lesson 4.17. SCM

This lesson will teach you the following concepts

- Overview.
- The Source Code Manager.
- Using the Source Code Manager.

Estimated time: 30 mn

Introduction

The development of a large IS system requires the participation of several developers. These developers must work on a single WINDEV project while sharing the different resources (windows, classes...).

WINDEV is supplied with a Source Code Manager named "SCM" used to share the source codes of different projects between developers and to find out the full history of modifications performed (in the code, in the interface, ...).

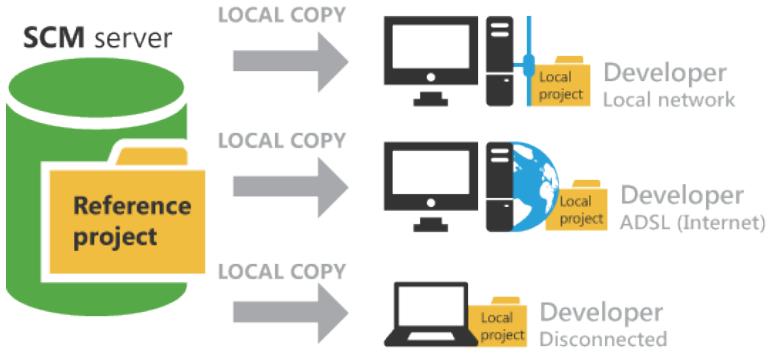
SCM (Source Code Manager)

Principle of SCM

The Source Code Manager is used to store and share the projects and their elements.

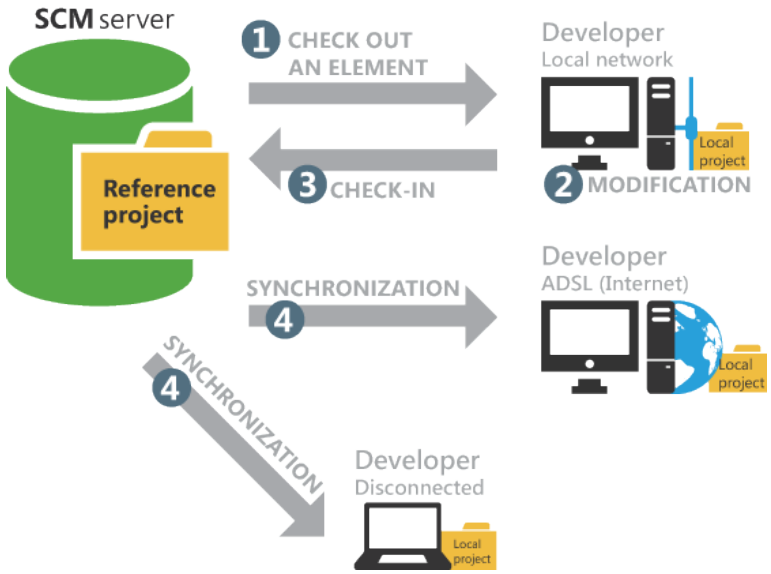
The principle is as follows:

- A reference version of each one of your projects is found on a server. This set of versions is called "Repository".
- Each developer has a local copy of different projects on his computer.



- Whenever a developer wants to modify a project element (window, report, query, ...), he informs the SCM that he is becoming the temporary owner of this element. To do so, the developer will check out the element from the repository.
- This developer gets exclusive rights on this element: all requested modifications can be performed on this element.
- The other developers are still working on the copy of the reference version of this element (found in the repository).
- Once the developer has finished, the checked-out element is checked back into the repository.

- The other developers are automatically notified of this check-in operation. They can now update their local copy.



The SCM supports teamwork and it allows you to find out the history of all modifications. The SCM can also be used to manage and control the elements shared between several projects.

Creating the repository

To share a project via the Source Code Manager, a repository must be created. This repository must be created once only on a server.

This repository can be created:
- when installing WINDEV.
- when creating a project that is using the SCM.
- when importing a project into the SCM.
- whenever you want, from WINDEV directly or from the SCM administrator.

The repository can be installed in the following modes:
- HFSQL Classic,
- HFSQL Client/Server,
- PCSCloud. The PCSCloud mode allows you to access the sources of projects from anywhere and at any time. This mode corresponds to a private Cloud and it proposes several options (dedicated platform, use of Control Centers, ...). Visit www.pcscloud.net for more details.
- SCM Drive. The SCM Drive mode allows you to access the sources of projects from anywhere and at any time. Visit www.pcscloud.net for more details.

▸ Our repository will be created when a project is imported into the SCM (next step).

Remark

We advise you to make backup copies of the repository on a regular basis. To do so, you must:
- connect as administrator to the management tool of SCM.
- on the "Management" pane, in the "Backups" group, select "Full backup".

Integrating a project in SCM

Adding the project into SCM

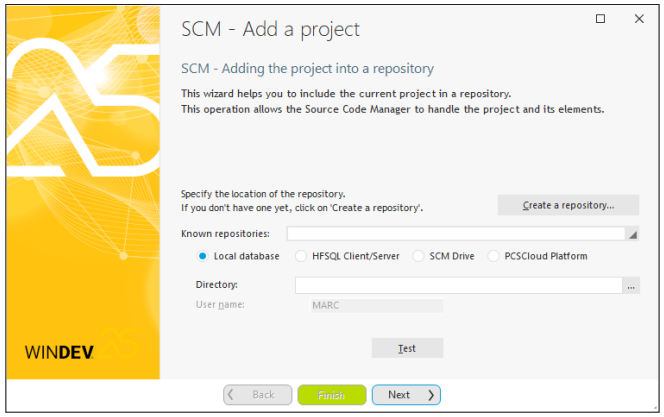
To use an existing project with the Source Code Manager, simply include it in the repository.

Answer

If you did not perform the operations in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)"

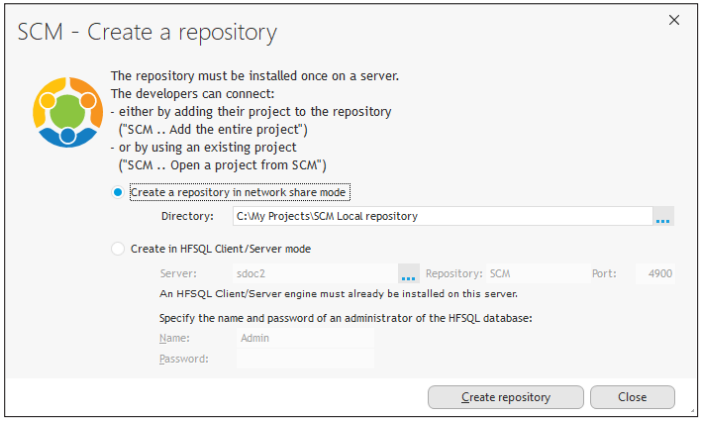
▸ We are now going to include the "WD Full Application.WDP" project in the repository:

1. On the "SCM" pane, in the "Add project" group, click "Add project to SCM". The wizard for adding projects into the SCM starts:



The repository was not created yet. We are going to create one.
Remark: We are going to create a "local" repository (on the development computer). The operation would be similar for another type of repository.

2. Click"Create a repository...".
3. The repository creation window appears.

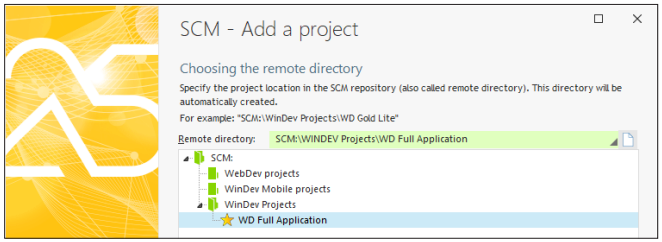


The repository can be in HFSQL Classic format (local or network) or in HFSQL Client/Server format. We are going to create a repository in HFSQL Classic format.

Remark

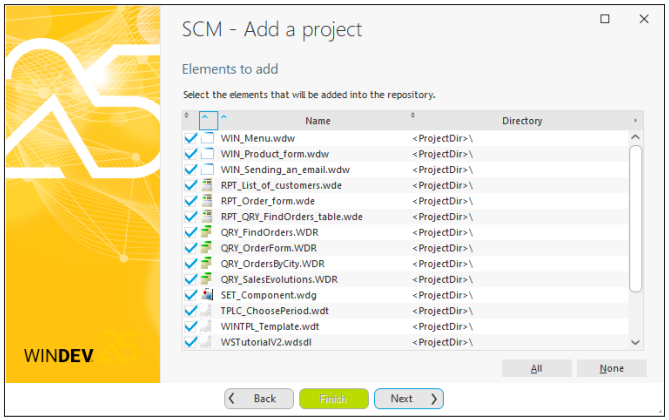
If the repository is in HFSQL Client/Server format, it can be used remotely.

4. Select "Create a repository in network share mode " and specify the directory of this repository ("C:\My Projects\Local repository" for example).
5. Validate the creation of the repository ("Create repository").
6. The repository has now been created. We are going to include our project in this repository.
7. Go to the next step.
8. The wizard proposes to locate the project in the "WINDEV projects" subdirectory of the repository.



Accept this location. Go to the next step.

9. The wizard asks you to select the project elements that will be added into the repository.



We want to add all project elements. Go to the next step.

10. The wizard asks you to select the project dependencies that will be added into the repository. These dependencies correspond to all external elements required by the project (images, style sheets, etc).
We want to add all project dependencies. Go to the next step.

11. Validate the project integration in the SCM. The project and its elements have now been added into our repository.
A help window about the SCM toolbar is displayed. Read and validate this window.

Remark

Sharing project elements

When the projects that share the same resources (same analysis, same windows, etc.) are included in the SCM, the relevant elements can be shared between the different projects. Therefore, the same element is checked in once only into the SCM and the modifications are automatically applied to the other projects.

Opening a project from SCM

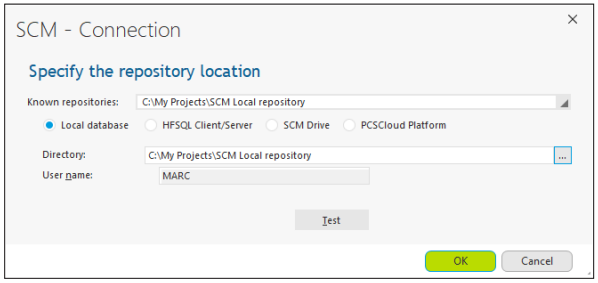
In our example, the project is integrated in the SCM and it can be used directly.

In a real case, in order for other developers to work on a project found in the Source Code Manager, they must get a copy of this project locally.

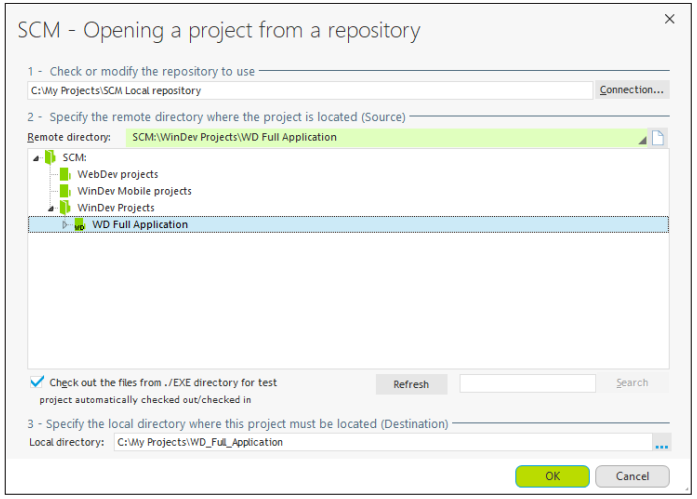
To do so, the following operations must be performed:

1. Open the project from the Source Code Manager: on the "Home" pane, in the "General" group, expand "Open" and select "Open a project from the SCM".

2. Specify the location parameters of the repository and validate (this step is required only if the current project in the editor does not belong to the SCM):



3. In the screen that is displayed, specify (if necessary) the connection and the local directory:



Remark: If the project was already opened from SCM, the SCM proposes to open the project as usual or to overwrite the content (to retrieve the entire project). This operation must be performed once only by each developer who is using the project. The developer who added the project into the Source Code Manager (you in this case!) has no operation to perform.

Remark

The next openings of a project managed by SCM are identical to the openings of a project not managed by SCM: all you have to do is open the project (".WDP" file) corresponding to the local copy.

Configuring SCM

Before you start working on the elements of project found in the SCM, you must configure the mode for checking out the project elements.

When working on the elements of a project in the SCM, the element must be checked out from the repository before it can be modified, then it must be checked back in once the changes have been made. Therefore, the modified element becomes available to all SCM users.

WINDEV proposes two modes for checking out the project elements:

- **the standard mode**: if you display a SCM element that is not checked out, a panel indicates that the element must be checked out before it can be modified. The element can be checked out immediately (check-out button found in the dialog box).
- **the automatic mode**: if you try to modify a SCM element that is not checked out, the SCM automatically proposes to check out this element. Once the check-out is validated, the element can be modified.
  Remark: this mode is not recommended when using SCM with a slow Internet connection.

The automatic check-out will be used in this tutorial.

▸ To make sure that the automatic check-out is enabled, on the "Home" pane, in the "Environment" group, expand "Options" and select "General options of WINDEV". In the "General" tab, check (if necessary) "Check out elements during the first modification".

Handling the project via SCM

We are now going to work with SCM in real conditions by performing the following operations:

- Modify a project parameter.
- Modify a project window.

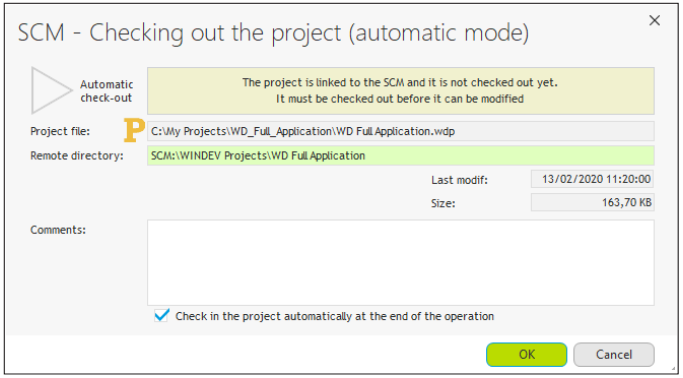
Modifying a project parameter

▸ We are going to modify the project by asking to display the skin template on the system windows:

1. Display the project description: on the "Project" pane, in the "Project" group, click "Description".
2. Click the "Style" tab.
3. Check "Customize the system windows (Info, YesNo, Confirm, Dialog)".
4. Validate the description window of project.

▸ Several SCM windows appear:

1. First of all, the window for automatic project check-out is displayed. Indeed, we want to modify a project characteristic therefore the project must be checked out.



2. "Check in the project automatically at the end of the operation" allows you to automatically check in the project once all the changes have been made. Keep this option.

3. Validate this window.

4. Different check-in and check-out windows open allowing you to add the internal component "WDFAA.wci" and its elements to the project in the repository and extract them to the local project. This component contains the different system windows to customize. Validate these different windows.

5. The project description window is closed and the project is automatically checked back into the repository.

Modifying a project window

We are now going to modify the "WIN_Product_form" window: a click on the image must allow to modify the image (like the "Modify" button).

The method for modifying a checked-out element (UI, code, ...) is the same as the method for modifying an element in a project not managed by SCM.

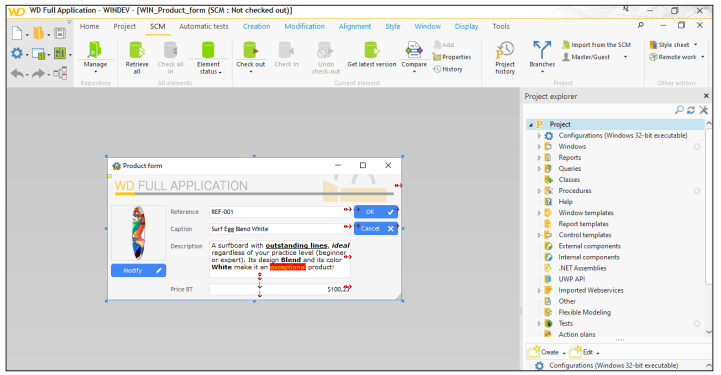
However, the modifications performed on a checked-out element are not visible to the other developers.

If another developer runs the checked-out element, the element that is currently found in the repository will be used.

This allows you to make an application evolve while keeping a stable version in the repository.

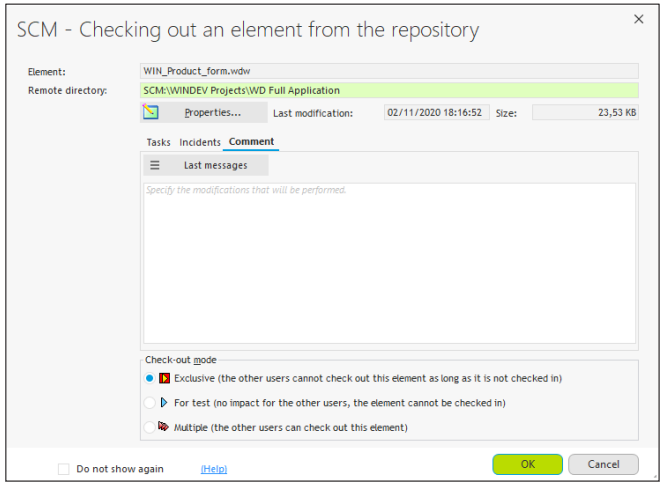
▸ To modify the "WIN_Product_form" window:

1. Select the "WIN_Product_form" window in the project explorer and double-click the element to open it in the window editor.



2. The automatic check-out is enabled during the first modification: all you have to do is move a control to check the element out. You also have the ability to click the "Check out" icon found in the ribbon ( ).

3. The check-out window is displayed:



4. The SCM proposes three check-out modes:
   - **Exclusive** (recommended mode): nobody can check out this element until it is checked back in. The element can be checked out for test only.
   - **For test**: the element can be modified but the modifications will not be checked back in.
   - **Multiple**: the element can also be checked out by other users. In this case, the differences between the different element versions can be viewed when the element is checked back in. This mode is reserved to specific cases and to experienced developers.
5. The window will be checked out in exclusive mode. Keep the "Exclusive" option checked.
6. Type a comment ("Image modification" for example). This comment will be useful for the other developers.
7. Validate the check-out. The window is checked out.
8. Display the description window of Image control ("Description" from the popup menu).
9. In the "UI" tab, modify the cursor used: select the "System hand" cursor.
10. Validate the control description window.
11. Display the code of Image control: select the Image control and press F2 for example.
12. Write the following code in the event "Click...":

```
// Runs the click code of button that modifies the image
ExecuteProcess(BTN_Modify, trtClick)
```

13. Close the code window.
14. Save your window (Ctrl + S).

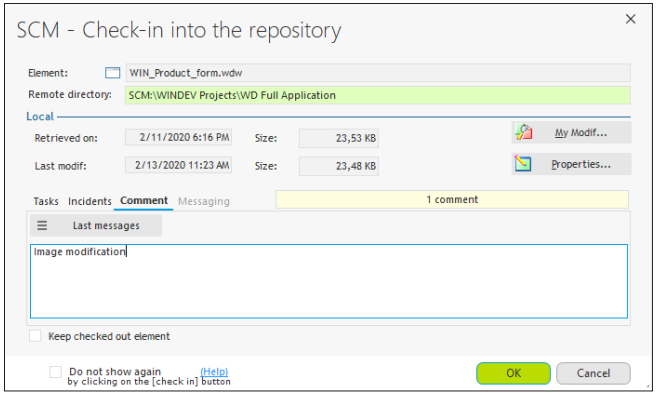
▸ Test your modifications.

1. Run the project test (GO among the quick access buttons).
2. Select a product and click "Modify".
3. In the form that is displayed, click the product image: the file picker is opened to modify the product image. This is the expected operating mode.
4. Close the test window.

Checking the checked-out element back in

Now that the modifications have been made and tested, we are going to check the window back into the repository. Then, your modifications will be made accessible to the other developers.

▸ On the "SCM" pane, in the "Current element" group, click the "Check in" button. The following window is displayed:



This window is used to:

- find out the modifications made by comparing the element from the repository with the checked-out element ("My Modif...").

> **Remark**
>
> **Merging code**
> You have the ability to compare an element to one of its earlier versions. This allows you to compare the code in order to retrieve a code section that was "lost" or accidentally deleted by another developer.

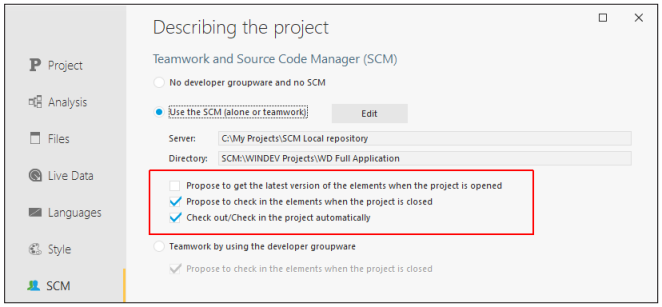
- access the history of the element in the repository ("Properties...").
- type a comment about the modifications performed. By default, WINDEV proposes the comment typed during the check-out.
- send a message to the other developers.
- check in the modifications made to the element while keeping the element checked out ("Keep checked out element").

> **Remark**
>
> If you are using the Control Centers, the current task can be ended when the element is checked back into the Source Code Manager. This feature is useful to follow the monitoring of tasks, the corrections of bugs, ...

▸ Validate the check-in. The window is displayed in the editor.

Synchronizing the project

Several options can be used to configure a project handled by SCM. These options are grouped in the "SCM" tab of project description (to display it, click "Description" on the "Project" pane).



These options are as follows:

- **Propose to get the latest version of the elements when the project is opened.**
  When opening a project found in the SCM, this option proposes to get the latest version of the project elements.
- **Propose to check in the elements when the project is closed.**
  When the project is closed, this option is used to display the list of elements that are currently checked out in order for some of them (or all of them) to be checked back in.
  By default, the checked-out elements are not checked back in when the project is closed.
- **Check out/Check in the project automatically.**
  This option allows you to automatically check out or check in the project when using an element.
  This option is selected by default.

Offline mode

The SCM allows you to work in offline mode. This mode allows a developer (who is using a laptop, for example) to continue working on a project found in the repository while being disconnected.

The principle is simple:

- before the disconnection, on the "SCM" pane, in the "Other actions" group, expand "Remote work" and select "Disconnect to work offline".
- during the reconnection, on the "SCM" pane, in the "Other actions" group, expand "Remote work" and select "Reconnect and synchronize". Then, all you have to do is check the modified elements back in.

In offline mode, there are two solutions to check out elements:

- No element is checked out from the SCM. The other developers will be able to work on the same elements as you while you are working in offline mode. When you reconnect to the SCM, the modifications made by yourself to the element will have to be merged with the modifications made by the other developers.
- The elements that you want to modify are checked out in exclusive mode. No one else can use the element while you are working in offline mode.

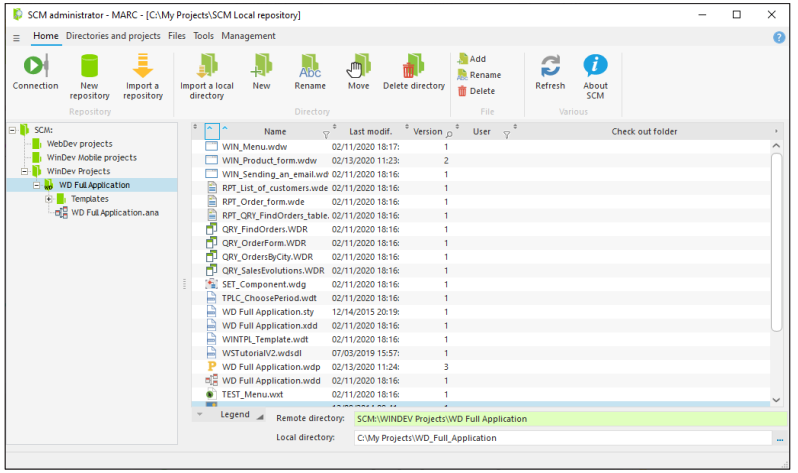
SCM administrator

The SCM administrator is used to directly handle the different projects included in the source code manager.

It allows you to:

- manage the repositories (creation, connection to a repository).
- manage the files and directories found in a project of repository (add, delete, rename, ... files and directories).
- manage the different files of the repository (check-in, check-out, share, etc).
- start some tools (options, maintenance, etc).
- show the history of an element.
- show the status of elements.
- perform backups.
- grant rights to the different SCM users.
- list the projects in which you are taking part in order to dissociate from them (if necessary).

▶ Start the SCM administrator: on the "SCM" pane, in the "Repository" group, click "Manage". All project elements are listed in the administrator.



See the online help for more details (keyword: "SCM").

Disconnecting from SCM

In the rest of this tutorial, we will be using the "WD Full Application" application. To simplify the operations, we advise you to disconnect from SCM:

1. Display the description window of project: On the "Project" pane, in the "Project" group, click "Description".
2. In the "SCM" tab, select "No developer groupware and no SCM".
3. Validate the description window of project.

Conclusion

We have presented the main steps for developing an application.

WINDEV proposes several tools to optimize your applications. See "Project audits" for more details.

Lesson 5.1. Introduction

This lesson will teach you the following concepts

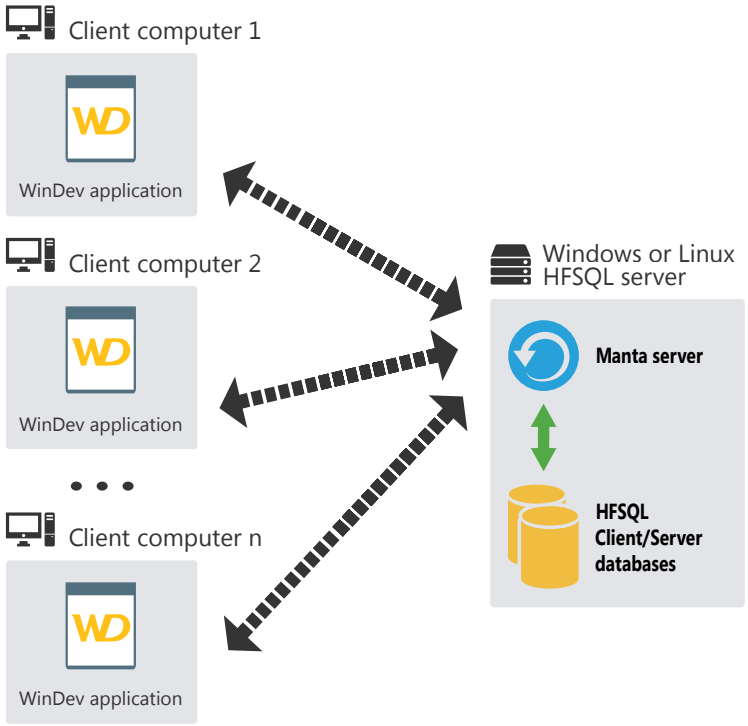
- Principle of Client/Server.
- Why switch an application to HFSQL Client/Server?

Estimated time: 5 mn

Overview

WINDEV allows you to create applications accessing HFSQL Client/Server databases.
A database in Client/Server mode allows you to host the databases on a server.
This operating mode:

- increases the security of data,
- allows you to easily manage the multi-user mode,
- simplifies the maintenance,
- ...



WINDEV allows you to:

- create an entire application using a HFSQL Client/Server database.
- modify an application that is using a HFSQL Classic database so that it can use a HFSQL Client/Server database.

Why switch an application to HFSQL Client/Server mode?

The main benefits of an application in HFSQL Client/Server mode compared to an application in HFSQL Classic mode are as follows:

- The use of HFSQL Client/Server is more secure (use of a login and password and definition of rights granted to the users).
- No management of directories: all the database files are grouped at the same location.
- The end users do not see the data files in the explorer and they cannot access them directly.
- The databases in Client/Server mode can be used via an Internet connection.
- Management of native multi-user mode: the performances are optimized in multi-user mode.

Lesson 5.2. Implementing a Client/Server database

This lesson will teach you the following concepts

- Installing a local HFSQL server.
- Creating an application that is using a HFSQL Client/Server database.
- Adapting an application to manage a HFSQL Client/Server database.
- Features available in Client/Server mode.

Estimated time: 10 mn

Overview

In this lesson, we are going to perform all operations required to develop and to deploy an application that is using a HFSQL Client/Server database.

Installing a local HFSQL server

The first operation to perform consists in installing a HFSQL server.

This server can be installed locally on the development computer (that's what we are going to do). In deployment, this server can be installed on a specific computer.

The HFSQL server setup program is available on the WINDEV setup media. If you do not have this media, the HFSQL server setup is also available on our site (www.windev.com).

▶ To install the HFSQL server locally:

1. Start the WINDEV setup program.
2. Select "HFSQL Server Setup".
3. Then, select "Install or update an HFSQL server".
4. Accept the license agreement.
5. Choose the platform ("For Windows on this computer").
6. If HFSQL servers are already installed on the current computer, select "Install a new server".
7. Select the setup directory and specify the server name and port. The port 4900 will be used by default.

Remark

Don't forget to open this port on the firewall in order to connect to the HFSQL server from another computer.

8. Specify the name and password of the user with management that will be created by default on the HFSQL server. By default, this user is "Admin" and has no password.

Remark

For security reasons, don't forget to change the administrator password.

9. Install the HFSQL Control Center if it is not already present or accessible from your computer.

Caution!

The HFSQL Control Center is required to manage the HFSQL Client/Server database.

10. The wizard proposes to configure the sending of notifications in order to identify the server dysfunctions in real time. In our example, go to the next step and indicate that the setting will be performed "Later".

11. The wizard proposes to enable the authentication via Active Directory. Enable this option if necessary.

12. Validate (or not) the participation in product improvement by allowing us to collect information regarding the use of product. This optional and anonymous collect allows PC SOFT to improve the product features.

13. The setup is performed. By default (if you have not changed the administrator settings), to connect to the server in administrator mode, you must use the "Admin" user without a password.

Creating an application that is using a HFSQL Client/Server database

Creating a WINDEV application that is using a HFSQL Client/Server database is child's play. You must:

1. Create the project by asking to create a new database.

2. Create the analysis by specifying that the databases used by the project will be "HFSQL Client/Server" databases.

3. Specify the characteristics of the connection to the HFSQL Client/Server server that will be used.

4. When creating a data file in the analysis, indicate that this data file is in Client/Server mode and specify the connection used.

Remark

You can also describe the connection to the HFSQL server by programming. See the online help for more details: ***HDescribeConnection***.

Adapting an application to use a HFSQL Client/Server database

Overview

Switching a database from HFSQL Classic mode to Client/Server mode is the most common operation.

WINDEV proposes several methods to perform this switch:

- perform this adaptation in the data model editor.
- perform this adaptation from the HFSQL Control Center.

To better understand the different steps, we are going to switch the application that was created in Part 3 to Client/Server mode by using the first method, the data model editor.

Adapting the example

Answer

If you did not perform the operations in the previous lessons, you can follow this lesson by opening a corrected project: in WINDEV's home page (Ctrl + <), click "Tutorial" and select "Full application (Answer)".

► To adapt the project:

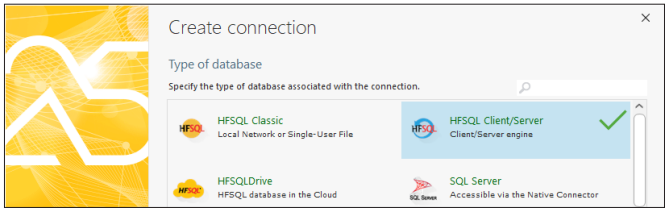
1. Open the "WD Full Application" project if necessary.

2. If necessary, go to the "Project explorer" pane and select the "Windows 32-bit executable" project configuration.

3. Load the analysis of your project in the data model editor: click  among the quick access buttons. The data model editor is displayed.

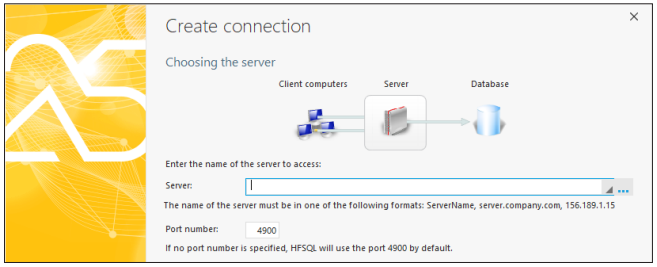
4. On the "Analysis" pane of the ribbon, in the "Connection" group, click "New connection". A wizard is opened, allowing you to create a connection.

5. Select the type of connection to create: "HFSQL Client/Server".



6. Go to the next step.

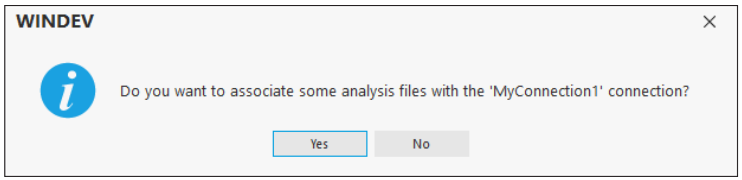
7. In the following steps, specify:



- the server name (name of your computer for example) and the port number. Go to the next step.
- the user name and password (leave this information empty to use the administrator). Go to the next step.
- the database name ("WD Full Application" in our example). Go to the next step.

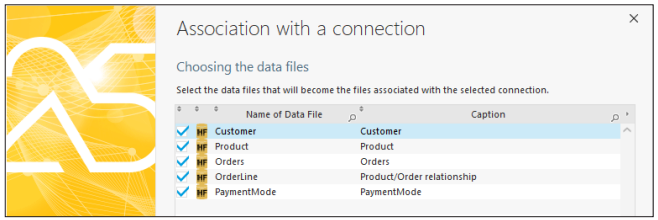
8. Type the connection name (keep the proposed name).

9. Go to the next step and validate. The connection to the database is automatically created. The wizard proposes to associate the different data files found in the analysis with the connection that was just created.



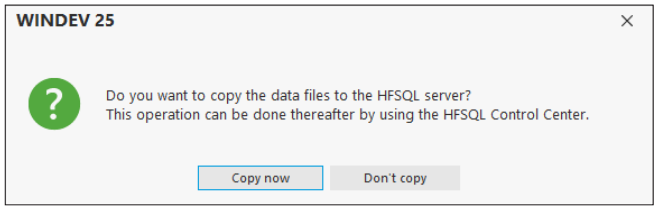
10. Click "Yes".

11. In the next step, select all the data files proposed:

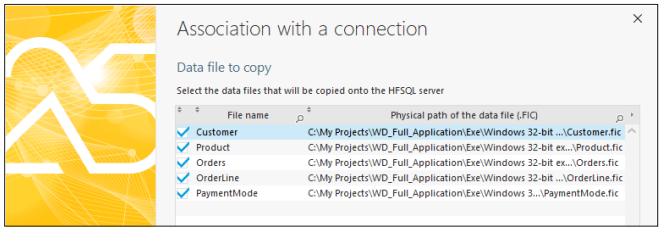


Go to the next step.

12. Then, the wizard proposes to copy the data files onto the server. Validate ("Copy now").

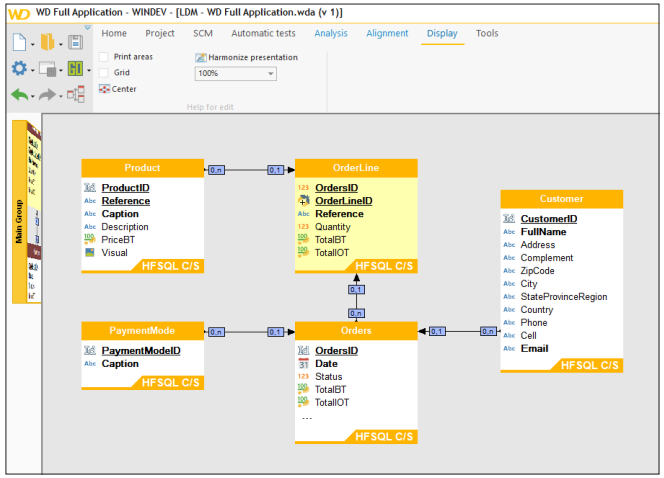


13. Select the analysis files that will be copied onto the server: in our case, all data files found in the EXE directory.



14. Go to the next step and validate.

15. The data files in the analysis are automatically changed into HFSQL Client/Server data files and associated with the selected connection.



16. Generate the analysis: on the "Analysis" pane of the ribbon, in the "Analysis" group, click "Generation". An automatic modification of data files is automatically proposed. If all the data files are up to date, the automatic modification of data files can be canceled.

Remark

Switching to Client/Server mode: tips

- Check the code of your project: in HFSQL Client/Server mode, the ***HSubstDir*** statements, ... are useless.
- Depending on the parameters specified when creating the connection, the connection defined in the analysis can be modified by ***HOpenConnection*** and ***HChangeConnection***.
- ***HOpenConnection*** is used to go back to HFSQL Classic mode: all you have to do is specify the path of directory containing the HFSQL Classic data files.

17. The development project was successfully adapted. You may also have to adapt the deployed application (if the deployed application is using HFSQL Classic data files for example). This operation is configured when creating the setup program of the application.

Features available in HFSQL Client/Server mode

HFSQL Client/Server proposes several features:

- Transactions,
- Logs,
- Stored procedures,
- Triggers,
- Hot automatic data modification,
- Hot reindexing,
- Scheduled backups,
- Incremental backups,
- Universal replication.

These features will not be described here (some of them have been presented in this tutorial in HFSQL Classic mode).

See the online help for more details.

Lesson 5.3. Managing a Client/Server database

This lesson will teach you the following concepts

- The HFSQL Control Center.
- Creating a user in the HFSQL Control Center.
- Saving the database.

Estimated time: 20 mn

Overview

Now that we know how to create and/or adapt an application so that it operates in HFSQL Client/Server, let's see how to manage the associated database.

Indeed, a Client/Server database requires:

- a specific configuration of computers (setup of a HFSQL server, ...).
- a management performed via the HFSQL Control Center.

Configuring the computers

To use a HFSQL Client/Server database, a HFSQL server must be installed on the server. Several HFSQL servers that use different ports can be installed on the same computer. One or more databases can be installed on each server.

For example, a test HFSQL server that includes a test database and a production HFSQL server that is using a different port can be installed on the same computer.

The HFSQL Control Center

The HFSQL Control Center is used to perform all management operations on the HFSQL Client/Server servers and databases.

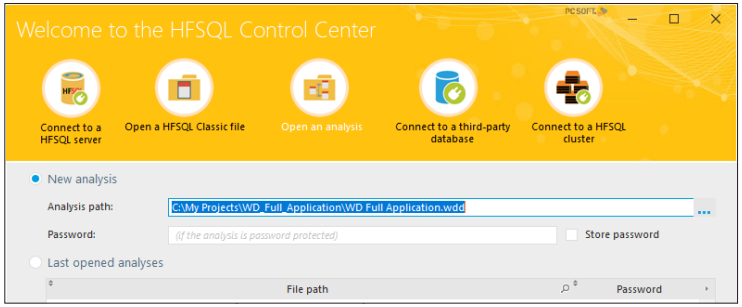
We are going to present the most important features.

First of all, we are going to start the HFSQL Control Center from the WINDEV project.

► To start the HFSQL Control Center and to access data:

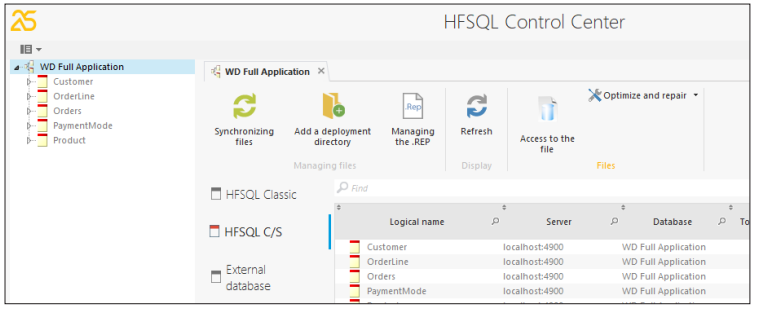
1. In the WINDEV menu, on the "Tools" pane, in the "Database" group, click "HFSQL". The HFSQL Control Center is displayed.

2. The home window of HFSQL Control Center is displayed. The analysis of current project is automatically selected.



3. Validate the screen. The HFSQL Control Center is displayed. This mode allows you to see the different files linked to the analysis of the current project.

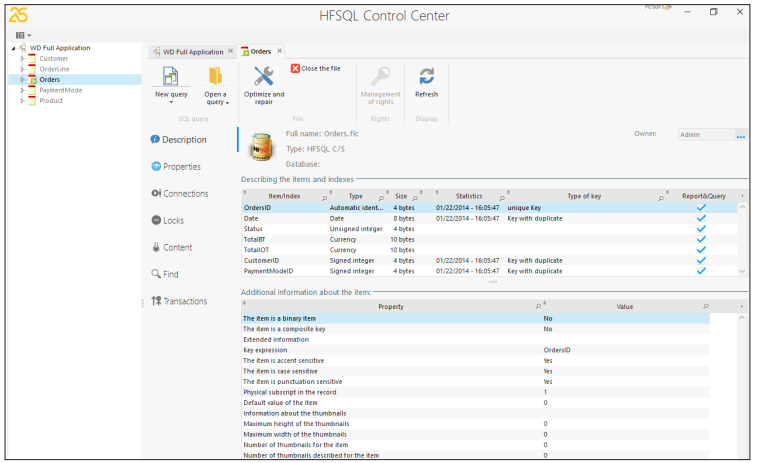
4. Click the vertical tab named "HFSQL C/S" if necessary. The list of data files in HFSQL Client/Server format is displayed.



The Control Center lists the Client/Server data files found in the analysis linked to the current project. No connection is established: the data files appeared grayed.

5. To see the data found in the files, double-click one of the data files in the list on the left ("Orders" for example). If the HFSQL Control Center does not recognize all connection parameters, a connection window is used to perform the effective connection to the HFSQL Client/Server server used. If this window is displayed, specify the password and validate.

6. The information about the selected data file that uses this connection is displayed in a new "Orders" pane. This pane presents this information in several vertical tabs:



- The "Description" tab gives information about the data files (data file items, etc.).
- The "Content" tab displays the records found in the data files.

The entire HFSQL Client/Server database can be managed from the HFSQL Control Center.

Creating a user account in the HFSQL Control Center

A single user account is created when installing a HFSQL server and when creating a database: the administrator account ("Admin" login without password).

Using a user account allows you to secure the access to data. Indeed, all application users are not administrators. Specific rights can be granted to each user (or group of users).

Caution!

The user rights specified in the HFSQL Control Center are granted for the database (and not for the application).

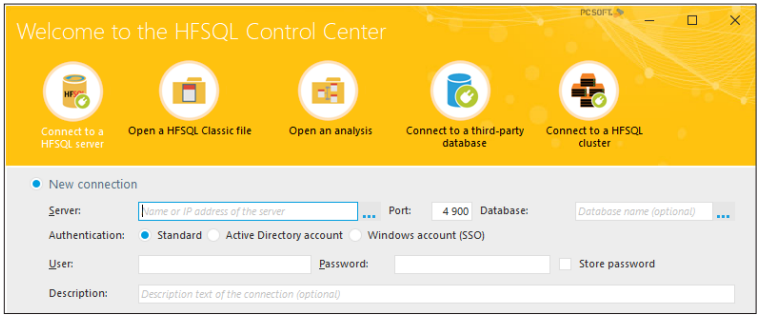
Do not confuse the management of rights for the Client/Server databases with the user groupware.

Some users may not have the rights to write into some data files for example.

To run a simple test, we are going to create a user and allow this user to see the records in the Customer data file.

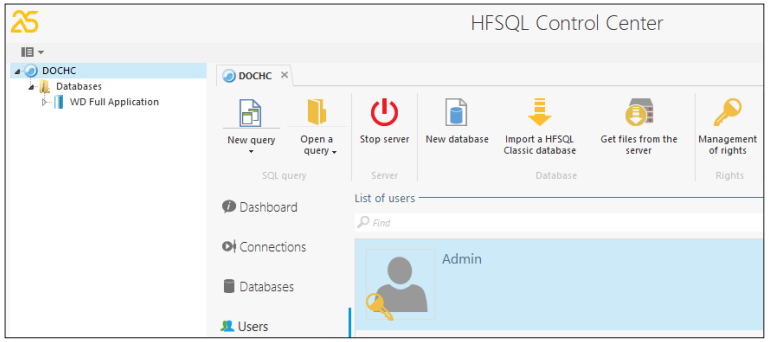
▸ To directly connect to the database found on the server:

1. Expand the menu found at the top left of HFSQL Control Center and select "Connect to a HFSQL server".
2. The home window of HFSQL Control Center is displayed.



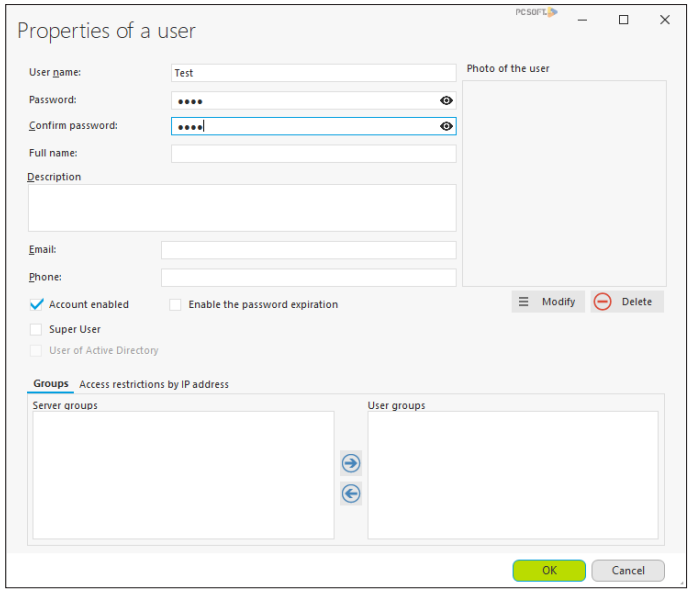
3. The option "Connect to a HFSQL server" is selected by default. Specify the characteristics of the server that was installed in the previous lesson and validate.
4. The characteristics of HFSQL server are displayed:
   - the name of HFSQL server as well as the list of databases found on this server are displayed in the left panel.
   - in the right section of the screen, a new tab allows you to see the characteristics of HFSQL server.

5. In the right section of the screen, select the "Users" tab. This tab is used to manage the server users.
6. Only the "Admin" user exists at this time.



7. To create a new user, in the ribbon, in the "Users" group, click the "New" button. The window used to define the user characteristics is displayed.

8. Type the following information:



(use "Test" as password for example).

Remark

Several characteristics can be noticed:

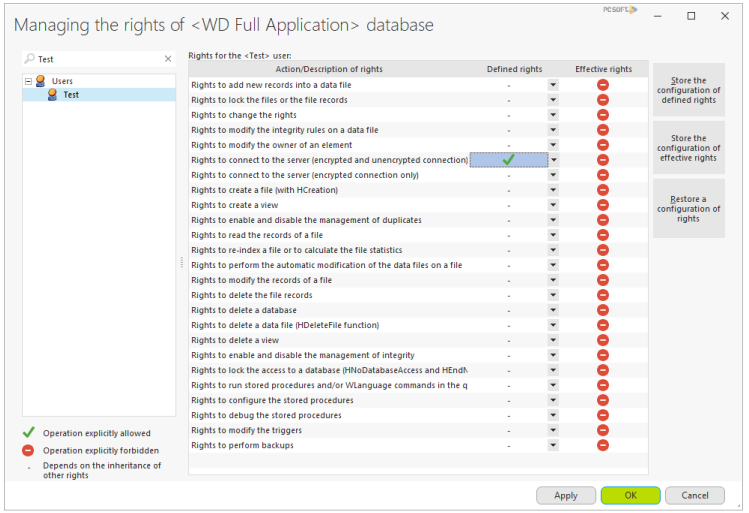
- Super User: The users defined as "Super user" are allowed to perform all types of actions on the server, on the databases and on all the data files.
- Account enabled: If this option is not checked, the user exists but he is not enabled (users on holiday for example).
- Password expiration: You can specify a password valid for a limited number of days. (configurable).

9. Validate the user creation. By default, no rights are granted to this user.

We are now going to grant rights to the user: the "Test" user can connect to the database and read the Customer data file.

▸ To grant rights to connect to the database:

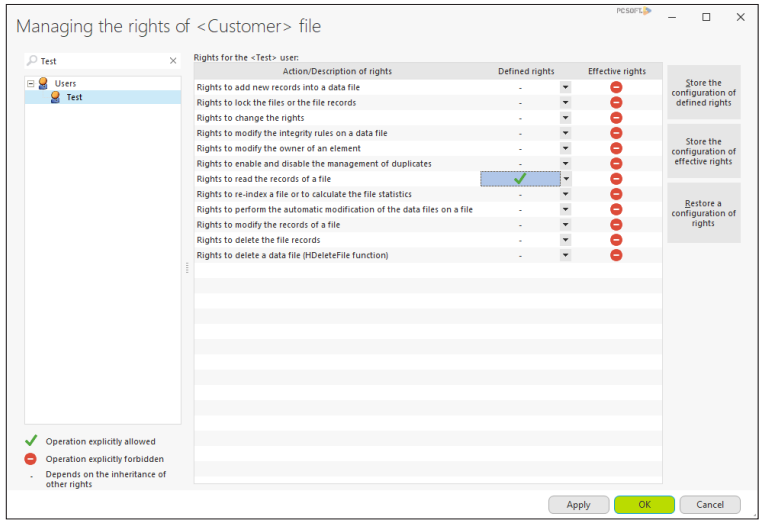
1. In the HFSQL Control Center, double-click the "WD Full Application" database.
2. On the "WD Full Application" pane, in the "Rights" group, click "Management of rights".
3. Select the "Test" user in the list of users.
4. In the list of rights, for the action named "Rights to connect to the server (encrypted and unencrypted connection)", click in the "Defined rights" column and select the green checkmark.



5. Click "Apply" at the bottom of window. The rights become effective.
6. Close the window for managing rights.

▸ To grant rights to read the Customer data file:

1. In the HFSQL Control Center, expand the "WD Full Application" database and double-click the Customer data file (on the left of the screen).
2. On the "Customer" pane, in the "Rights" group, click "Management of rights".
3. Select the "Test" user in the list of users.
4. In the list of rights, for the action named "Rights to read the records of a file", click the "Defined rights" column and select the green checkmark.



5. Click "Apply" at the bottom of window. The rights become effective.
6. Close the window for managing rights.

Similarly, rights can be defined:
- on the HFSQL server
- on the database
- on the database files.

In our example, the "Test" user can only browse the records in the Customer data file. If this user tries to perform another action, a message will be displayed: "The Test user has no sufficient rights to XXXX" (where XXXX corresponds to the action performed).

Once the account is created, it can be used when the application connects to the server (when ***HOpenConnection*** is used).

Remark

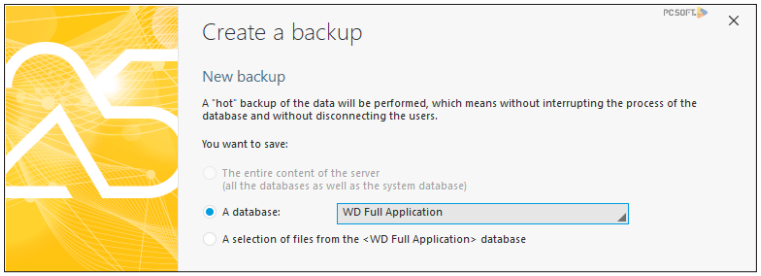
The users and their rights can also be managed by programming with the WLanguage functions.
See the online help for more details.

Saving the database

To make a backup copy of the database:

1. Go to the tab corresponding to the "WD Full Application" database.
2. Select the vertical tab named "Backups".
3. In the menu, in the "Backups" group, expand "New backup" and select "New hot backup".

Remark: This button is accessible in the "Backups" group:
- on the tab corresponding to the HFSQL server,
- on the tab corresponding to the database.



Note: The wizard can also be used to save a selection of data files.

4. Validate the backup.

Conclusion

The HFSQL Control Center is a tool for managing databases, allowing you to:
- stop or restart a server if a problem occurs,
- manage the users and their rights,
- reindex the data files if necessary,
- perform database backups.

The HFSQL Control Center is a redistributable tool that can be installed on the computers of users who are working with HFSQL Client/Server databases. The HFSQL Control Center must be used by the database administrator.

Lesson 5.4. Setup on end-user computers

This lesson will teach you the following concepts

- How to install a Client/Server application on the user computers?

Estimated time: 5 mn

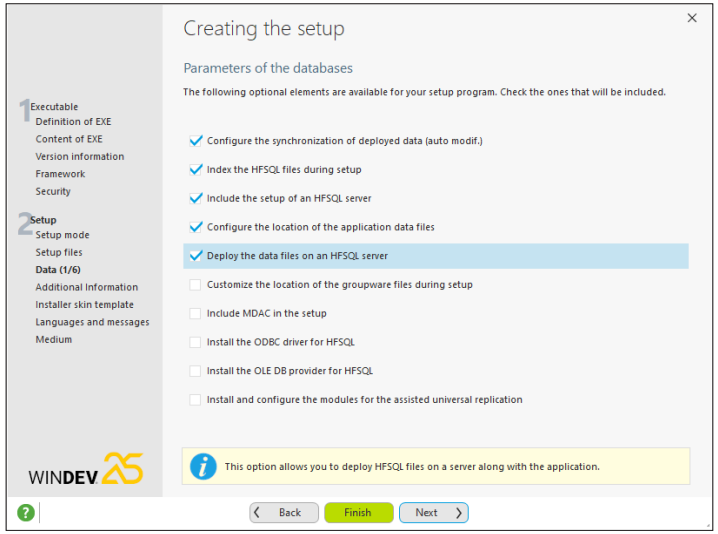
Overview

Installing a Client/Server application requires several specific options. These options are taken into account by the wizard for setup creation of WINDEV.
We are going to create the setup procedure of our "WD Full Application" application.

Starting the wizard for setup creation

To start the setup procedure of an HFSQL Client/Server application:

1. On the "Project" pane, in the "Generation" group, click "Setup procedure".
2. Create the executable and validate the help creation.
3. Click the "Data" option on the left side of wizard: the "Parameters of the databases" step is displayed. In this step, you can:
   - Include the setup of an HFSQL server,
   - Configure the location of the application data files,
   - Deploy the data files on an HFSQL server.



If these options are checked, the next steps are used to configure the elements that will be taken into account for the setup on end-user computers.
See the online help for more details.


DEVELOP 10 TIMES FASTER
PART 6
Optimizing and debuging a project

Lesson 6.1. Overview

This lesson will teach you the following concepts

- Why optimize an application?
- Example project.

Estimated time: 5 mn

Overview

Your application is created. It operates properly, several automatic tests have been created to avoid the regressions. You want to deploy it on the end-user computers.

Why not use the WINDEV tools to optimize your application? WINDEV proposes several tools and features allowing you to optimize your application and to avoid predictable bugs.

This part presents these tools and their use.

A project to optimize was prepared, allowing you to handle these features!

Opening project

► Start WINDEV 25 (if not already done).

► Display the WINDEV home page if necessary (Ctrl + <).

► Open the "WD Optimization" project.
To do so, in the home page, click "Tutorial" and select "Optimize a project".

Lesson 6.2. Project audits

This lesson will teach you the following concepts

- What is an audit and what is its purpose?
- Starting and studying the static audit.
- Starting and studying the dynamic audit.

Estimated time: 15 mn

What is an audit?

Audits provide a set of features that allow you to:
- automatically improve the quality and performance of a project.
- monitor the conditions in which it operates more precisely.

Two types of audits are available:
- **The static audit**. The static audit performs a detailed analysis of a project and elements. This audit is performed from the project editor.
- **The dynamic audit**. The dynamic audit analyzes the behavior of a project during its execution. This audit can be performed in test mode or in the production environment.

We are going to check these audits on the "WD Optimization" project.

Static audit

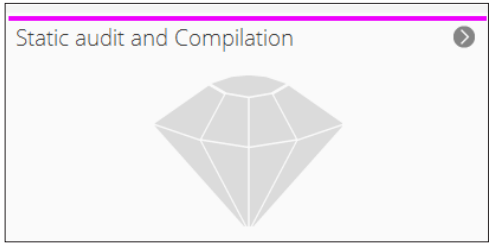
The static audit is an environment feature used to study the source codes of a project in order to detect different problems and to propose improvements.

► To start the static audit on the "WD Optimization" project:

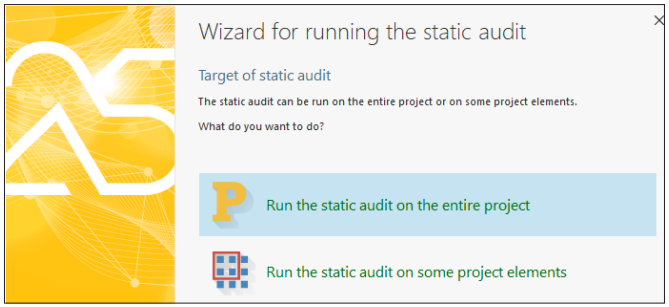
1. On the "Project" pane, in the "Audit and performance" group, expand "Edition audit" and select "Trigger the edition audit".

Remark

The static audit of project can also be started from the project dashboard, via the "Static audit and Compilation" widget.
To do so:
- enable the Widget if necessary (click the link "Click here to re-enable").
- click the arrow.



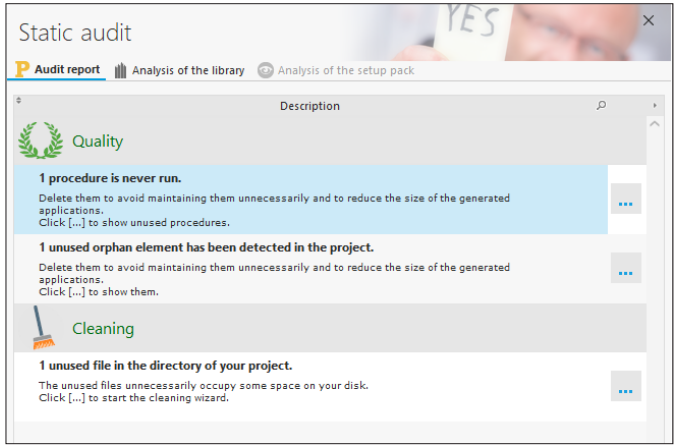
2. The wizard starts. We are going to define the target of static audit.



3. Select "Run the static audit on the entire project".

4. Validate the wizard.

5. The audit report is displayed:



The static audit includes:
- The static audit of project.
- The audit for the content of application library.
- The audit of setup content.

► Let's study the topics presented by this report.

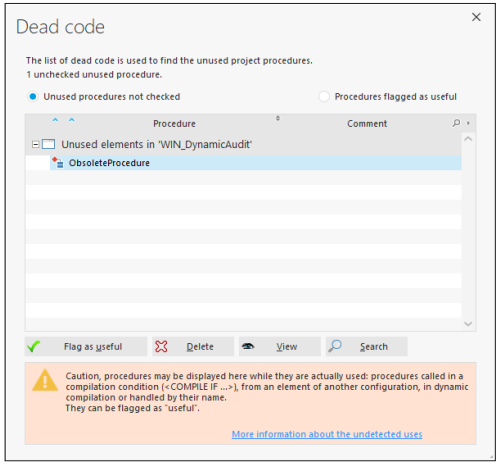
Procedure not run

In our project, the audit indicates that a procedure is never run.

In huge projects, you may want to create one or more procedures to perform a process then, further to a code reorganization, the procedure is no longer used but it remains in the project. The presence of unused procedures unnecessarily consumes the resources distributed to the end users.

▶ To fix this problem:

1. Click the [...] button to get more details. The window that lists the dead codes is displayed.



2. The "ObsoleteProcedure" procedure is never called. This window is used to:
   - specify that the procedure is still used ("Flag as useful"). In this case, the procedure will not be taken into account anymore during the audit.
   - delete the procedure if it is actually unused ("Delete").
   - see the procedure ("View").
   - find the use cases in the strings for example ("Search").
3. In our case, this procedure is actually unused, click "Delete".
4. A window is displayed, asking you to confirm the deletion. Click the "Delete" button to confirm the deletion.
5. Close the window of dead code (click the cross in the top right corner).

▶ In the window of static audit, click the "Refresh" button to update the audit report.

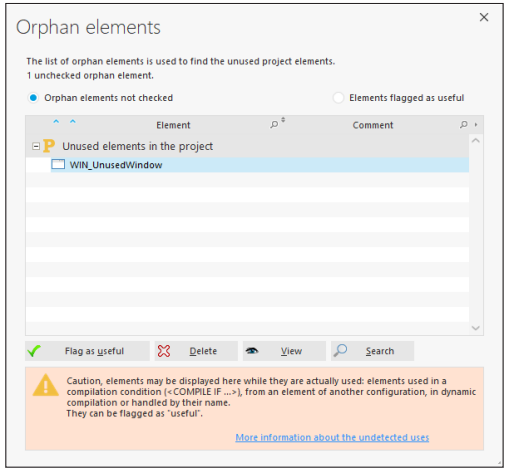
Orphan element

The audit indicates that our project contains an orphan element.

As for the procedures run, you may create windows or reports in order to run a quick test, save them and forget everything about them. The presence of orphan elements in the executable unnecessarily consumes the resources distributed to the end users.

▶ To fix this problem:

1. Click the [...] button to get more details. The window that lists the orphan elements is displayed.



2. The "WIN_UnusedWindow" window is never called. The window that lists the orphan elements is used to:
   - specify that the element is still used ("Flag as useful"). In this case, the window will not be taken into account anymore during the audit. This option can be interesting when using a test window specific to the development for example.
   - delete the element if it is actually unused ("Delete").
   - see the element ("View").
   - find the use cases in the strings for example ("Search").
3. In our case, this "WIN_UnusedWindow" window is actually unused, click "Delete".
4. Close the window of orphan elements (click the cross in the top right corner).

▶ In the window of static audit, click the "Refresh" button to update the audit report.

Cleaning the project

Our project contains several unused files. You have the ability to clean the project in order to keep the necessary elements only. The executable and the client setup is not weighted with images, external files, ... not used.

▸ To fix this problem:

1. Click the [...] button to get more details.
2. The wizard for cleaning a project starts. This wizard indicates the unused files that can be deleted.
3. Go to the next step.
4. Select the type of cleaning to perform. You can:
   - create a zip file with the useless files.
   - move the useless files into a specific directory.
5. Validate the option proposed by default and go to the next step.
6. End the wizard.

Our project was optimized according to the tips given by the static audit.

The static audit is used to get an overall status on the source code of your project. Our advice: run it on a regular basis!

Let's see what happens at run time by starting the dynamic audit.

Dynamic audit

The dynamic audit is used to study the application execution. The audit is used to detect problems such as:

- Excessive memory consumption,
- Slowness of algorithms used,
- Errors "hidden" at run time,
- ...

A dynamic audit can be performed in a test environment or on a live application.

The "WD Optimization" project contains a specific window triggering errors that can be detected by the dynamic audit.

The dynamic audit and the project test will be started at the same time.

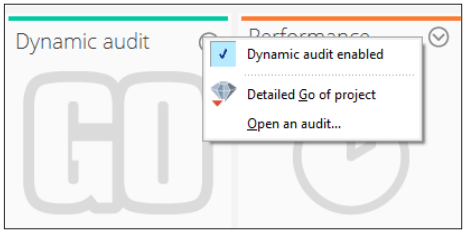
▸ To start the dynamic audit on the "WD Optimization" project:

1. On the "Project" pane, in the "Test mode" group, expand "Test mode" and select "Debug the project while the audit is enabled". The project test is run.

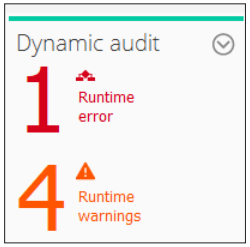
Remark

The dynamic audit of the project can also be started from the project dashboard, via the "Execution" widget. To do so:

- enable the Widget if necessary (click the link "Click here to re-enable").
- expand the arrow and select "Detailed Go of project".

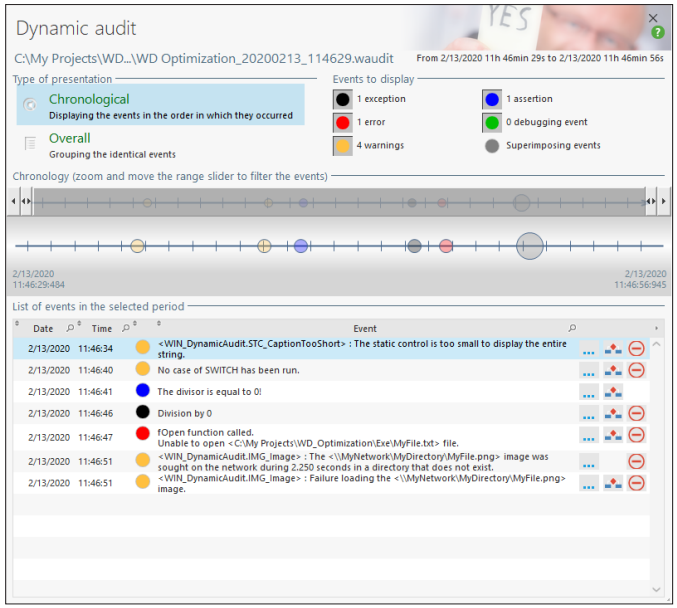


Remark: The dynamic audit is automatically run whenever the application is started by a project GO. A report is automatically displayed in the project dashboard:



2. Click the "Test window of dynamic audit" button.
3. Click the different buttons found in the window. At the end of each process, a toast message is displayed to specify that the process is over. For the "Assertion and Exception" option, an assertion is displayed: click "Continue" in order for the message to appear in toast format.
4. Stop the application test.

5. The report window of the dynamic audit appears.



► Let's study this window:

- The top section of this window is used to choose the display mode of data. You can:
  - choose a chronological display (respecting the order in which the events occurred) or an overall display, used to group the different types of problems. In this case, the chronological border allows you to see the position and importance of problems.
  - choose the type of problem to display (error, assertion, ...). This allows you for example to concentrate on the major errors, ...
- The bottom section of this window displays the different events that occurred and that may cause problems in the application.

► In this example, the dynamic audit detects several problems:

- An excessively long caption that is assigned by programming,
- A SWITCH loop for which no CASE is run,
- An assertion is triggered instead of an exception,
- The opening of a file that does not exist,
- The assignment of a nonexistent image file to an Image control.

For each problem, a "..." button is used to access the event details. If the event is linked to a specific line of code, the button allows you to open the code editor directly at the corresponding location in order to fix the problem.

► Close the window of dynamic audit.

Remark

The dynamic audit of a project can also be performed when an application is deployed on the user computers.

You can:

- modify the application and use ***dbgEnableAudit*** to start the audit.
- use a text file in the application directory. This solution allows you not to modify the executable. All you have to do is create, in the directory of the executable, a file named like the executable and whose extension is ".WX".

The audit generates a ".wdaudit" file, this file must be loaded in the development environment in order to study the result.

See the online help for more details (keyword: "Dynamic audit").

Lesson 6.3. Performance profiler

This lesson will teach you the following concepts

- Overview.
- Starting the performance profiler.
- Studying the result.

Estimated time: 15 mn

Overview

The performance profiler (also called Profiler) is a tool used to check and optimize the execution time of the processes found in your application.

The principle is straightforward:

- You run the test of your application.
- During this test, the performance profiler keeps track of all actions performed and saves the execution times of each one of the processes run.

At the end of test, the performance profiler displays:

- the 10 most time consuming operations,
- the duration and the number of calls for all processes run.

The "WD Optimization" project contains a specific window used to see the interesting results with the performance profiler.

Starting the performance profiler

The performance profiler can be started:

- **from the WINDEV editor**:
  In this case, the project is automatically run in test mode. You can use your application and start the processes of your choice.
  To go back to the WINDEV editor, all you have to do is exit from your application.
  Then, the performance profiler displays the monitoring result. This result is saved in a WPF file.
- **from one of your WLanguage processes**, with the following functions:

ProfilerStart	Starts "collecting data" for the performance profiler.
ProfilerEnd	Stops "collecting data" for the performance profiler.

In this case, only the code found between ***ProfilerStart*** and ***ProfilerEnd*** is studied. The result is saved in a WPF file.

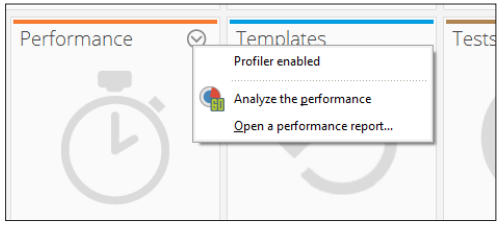
► The first method will be used in our example. To start the performance profiler on the "WD Optimization" project:

1. On the "Project" pane, in the "Audit and performance" group, expand "Analyze the performance" and select "Analyze the performance".

Remark

The performance profiler can also be started from the project dashboard, via the "Performance" widget. Simply:

- enable the Widget if necessary (click the link "Click here to re-enable").
- expand the button with arrow and select "Analyze the performance".



2. The project test is run.

3. Click the "Test window of performance profiler" button.

4. Click the "Process to analyze" button.

5. Validate the information window and stop the project test. The report window of performance profiler is displayed.

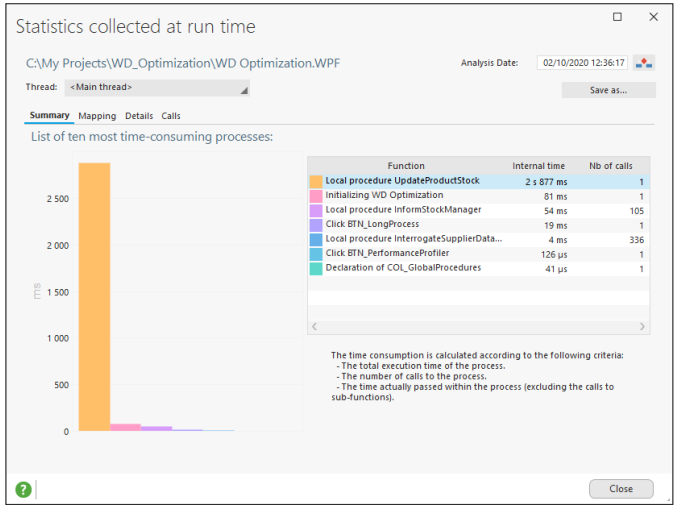
Studying the result

► Let's study the report window of the performance profiler. Results are displayed in several tabs:

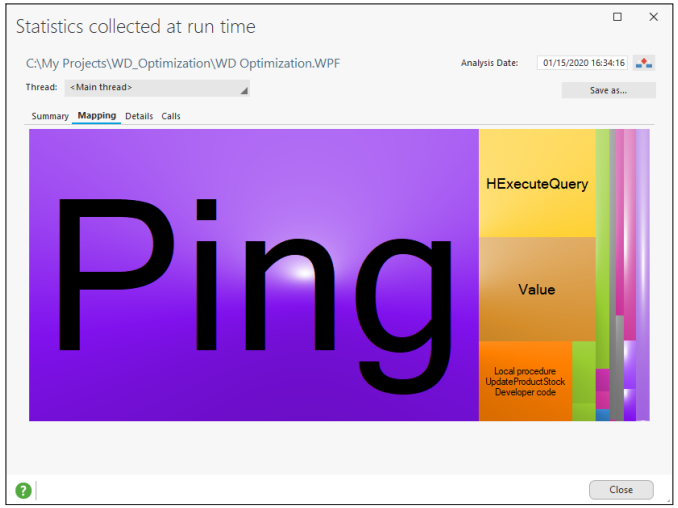
- the "Summary" tab presents the ten longest processes.
- the "Mapping" tab presents a graphical view of main processes.
- the "Details" tab presents all processes run during the application test (from the slowest one to the fastest one).
- the "Calls" tab is used to view the details of operations performed in a process.

► Let's present these different tabs in our example.

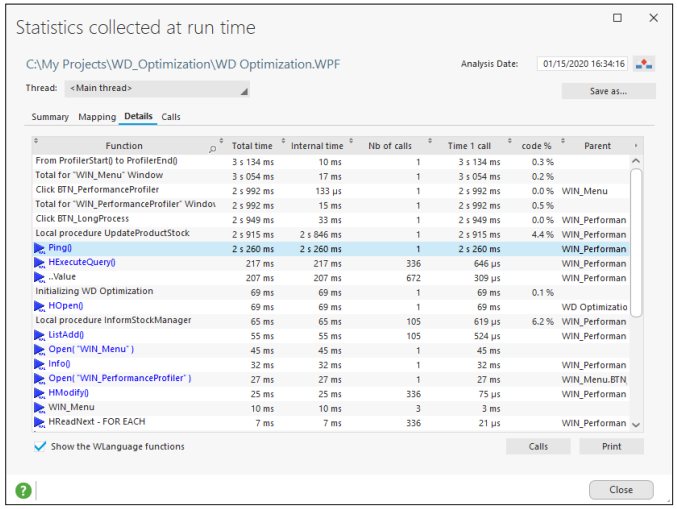
- The "Summary" tab presents the ten longest processes. In our example, you can see that the local procedure named "UpdateProductStock" takes more than 3 seconds to run.



- The "Mapping" tab is used to visually identify what took the longest time. In our case, it's a call to **Ping**:



- The "Details" tab presents all processes or events run, from the slowest one to the fastest one.



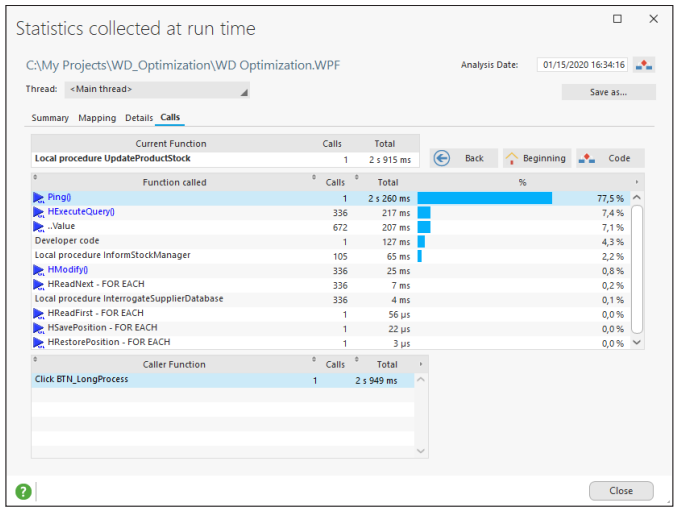
The following information is displayed for each process or event:

- Function: Function, event or procedure run.
- Total time: Execution time of function.
- Internal time: Execution time due to the engine.
- Nb of calls: Number of calls made to the function (procedure or event).
- Time 1 call: Execution time of a call to the function (procedure or event).
- Code %: Percentage of time spent in the process of function or procedure (developer code that can be optimized).
- Parent: Element that contains the process.

► In our case, the "Details" tab indicates that the call to the "Ping" function is one of the elements taking the longest time.

1. Select this line. We are going to check whether this slowdown is caused by a specific problem.

2. Click the "Calls" button to display the details of calls to the **UpdateProductStock** procedure. Select the "Ping" line and click the "Code" button: the corresponding code line is displayed in the code editor.



3. Close the performance profiler.

4. The following code line is run:

```
// Checks the accessibility of supplier server
Ping("supplier-addr")
```

The slowdown is caused by the fact that the address specified for **Ping** is not accessible.

► Let's check the operating mode of application by optimizing this code:

1. Replace the code line containing **Ping** by the following code line:

```
// Checks the accessibility of supplier server
Ping("www.google.fr")
```

2. Save the code (Ctrl + S).

▶ We are now going to restart the performance profiler:

1. On the "Project" pane, in the "Audit and performance" group, expand "Analyze the performance" and select "Analyze the performance".
2. The project test is run.
3. Click the "Test window of performance profiler" button.
4. Click the "Process to analyze" button.
5. Validate the information window and stop the project test. The report window of performance profiler is displayed.
6. In the "Mapping" tab, ***Ping*** does not appear with the same importance.

▶ Close the report window of performance profiler.

Lesson 6.4. Debugging a project

This lesson will teach you the following concepts

- Overview.
- Using the debugger.

Estimated time: 15 mn

Overview

Let's take a look at the debugger supplied with WINDEV.

What is the debugger?

The debugger is a powerful tool used to follow the code or application progress, step by step. Enhancing a process or even an application becomes child's play.

We are going to use the debugger on the long process found in the WIN_PerformanceProfiler window.

Using the debugger

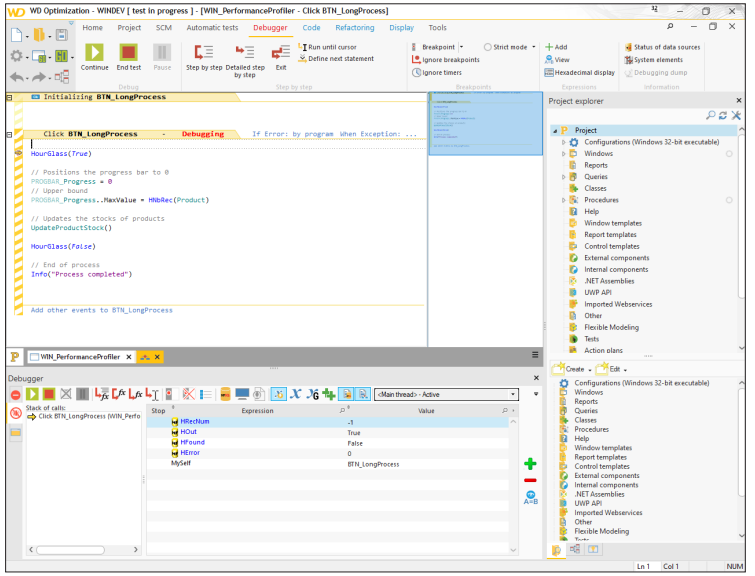
▶ To debug the WIN_PerformanceProfiler window:

1. Open "WIN_PerformancepRofiler" in the window editor (double-click its name in the "Project explorer" pane).
2. Run the test of "WIN_PerformanceProfiler" window (among the quick access buttons).
3. Start the debugger by using one of the following methods:
   - Press Ctrl + [Pause]: the debugger will be automatically started during the next user action in the window whose test is run.
   - Go back to the WINDEV editor and position a breakpoint in the WLanguage code associated to the Button control "Process to analyze" (click next to the first line of code of the "Click" event or press Ctrl + B; a red dot will appear). The debugger will be automatically started when the line preceded by the breakpoint is run.

Remark

Several methods can be used to start the debugger.
See the online help for more details (keyword: "Debugger, Run a test").

4. Click "Process to analyze". The debugger starts. The current line is preceded by a little arrow.



The "Debugger" pane appears in the lower section of the screen. This pane displays two distinct areas:

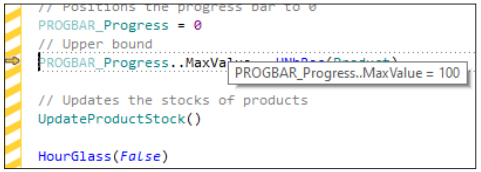
- the call stack: This area allows you to know the hierarchy of the events and processes displayed in the debugger. In our example, we are currently debugging the event "Click BTN_LongProcess".
- the list of expressions to evaluate. By default, the main variables used in the WLanguage code are displayed in this area. You have the ability to add variables in order to follow their evolution (this feature will be presented later).

We are going to perform some operations in the debugger to discover its capabilities.

▶ First of all, we are going to run the different statements step by step and see the content of variables:

1. Press the F8 key (or click the "Step by step" button found in the ribbon). The line following the current line is automatically run. The values of variables are modified (if necessary) in the "Debugger" pane (displayed at the bottom of the screen).

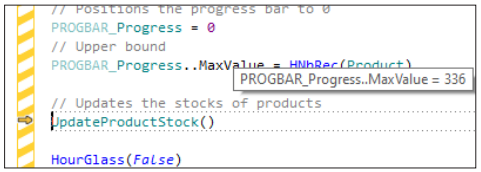
2. When the cursor shaped like an arrow reaches the "PROGBAR_Progress..MaxValue = ..." line, hover "PROGBAR_Progress" with the mouse. A tooltip is displayed with the expression value:



3. The value of "PROGBAR_Progress..MaxValue" is displayed in a tooltip. This value corresponds to 100 because the code line was not run.

4. Press F8 to run the line.

5. Hover "PROGBAR_Progress..MaxValue" again. The value of "PROGBAR_Progress..MaxValue" displayed in the tooltip corresponds to the result of HNbRec(Product).



► Let's continue our operations in the debugger. We are now going to run the ***UpdateProductStock*** procedure step by step in the debugger.

1. To run this procedure step by step, press the F7 key (or click the "Detailed step by step" button in the ribbon).

2. The procedure code is displayed in the debugger.

► We are now going to add an expression to monitor the evolution of its value in the "Debugger" pane. This expression can have any type: variable, function, operation on variables, ... The expression result is calculated and displayed.
This expression is used to perform a custom debugging. For example, you can find out the content of a variable while it is being used in the application.

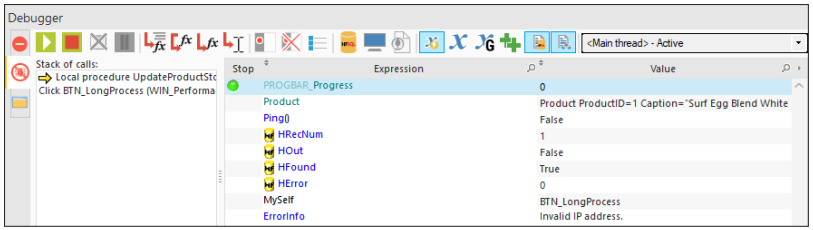
1. Press F7 to run the different lines of ***UpdateProductStock*** procedure until you reach the line:

```
PROGBAR_Progress++
```

2. Hover "PROGBAR_Progress". The value displayed in the tooltip is 0.

3. Select "PROGBAR_Progress" in the code editor and open the popup menu (right click). Select "Add the expression to the debugger".

4. The expression is automatically added into the debugger pane at the bottom of the screen.



► The debugger can also be used to run a set of code lines:

1. Press F8 until you reach the line:

```
HModify(Product)
```

The F8 key is used to run the code of ***InterrogateSupplierDatabase*** procedure without running it step by step in the debugger.

2. Click the following line with the mouse (caution: click inside the line and not in front of it):

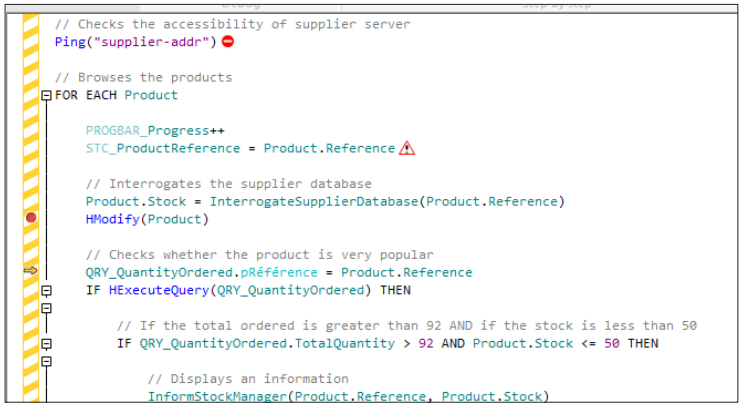
```
IF HExecuteQuery(QRY_QuantityOrdered) THEN
```

3. Press the F6 key (or click the "Run until cursor" button found in the ribbon).

4. The arrow indicating the line currently run moves until it reaches the code line where the cursor is positioned. The code lines found before the cursor are automatically run.

► We are now going to add a breakpoint and to run the code until it reaches the breakpoint:

1. Click in the hatched area with the mouse, in front of **HModify**. A breakpoint (red bullet) appears.



2. Press the F5 key (or click the "Continue" button found in the ribbon). The code is run until it reaches the breakpoint. The arrow used to identify the current line moves until it reaches the breakpoint.

3. Click the breakpoint to remove it.

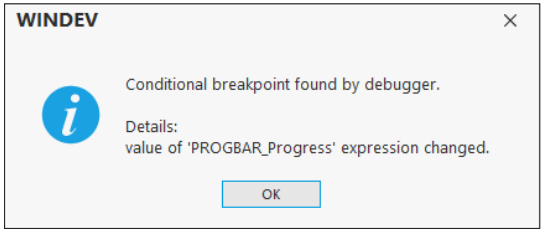
▸ To end this test, we will be using an "Auto-stop" expression. An "Auto-stop" expression is used to start the debugger as soon as a condition is checked or when the value of a variable is modified. In our example, the debugger will be started as soon as the value of progress bar is set to 50%:

1. In the "Debugger" pane, select the "PROGBAR_Progress" expression that was added beforehand.

2. Click the green circle.

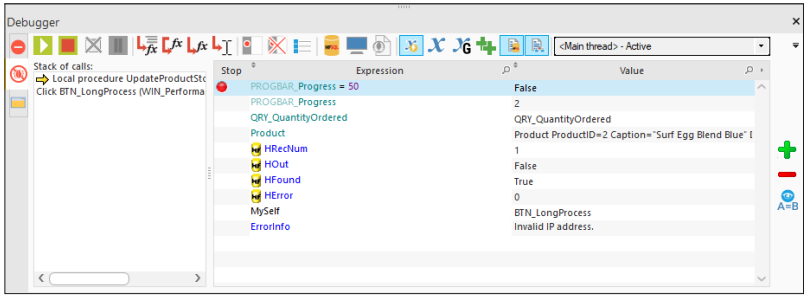
3. Press F5 to continue the test.

4. A message is displayed, indicating that the value of "PROGBAR_Progress" expression changed.



5. Validate.

6. In the "Debugger" pane, select the "PROGBAR_Progress" expression. Click the expression again: the "Expression" column becomes editable. In the "Expression" area, add "=50". You will get "PROGBAR_Progress = 50".



7. Press F5. The program continues to run. The debugger is started again when the value of the progress bar is set to 50.

▸ That's it, the main features of debugger have been presented here. To stop the test in the debugger, click "End the test" found in the ribbon.

Lesson 7.1. Overview

This lesson will teach you the following concepts

- Overview.
- How to open a training example?
- How to open a unit example?

Estimated time: 5 mn

Overview

This section presents several advanced features. You don't necessarily have to read it but it will allow you to discover some advanced features proposed by WINDEV.
The different lessons found in this section are based on examples supplied with WINDEV.

WINDEV proposes different types of examples:

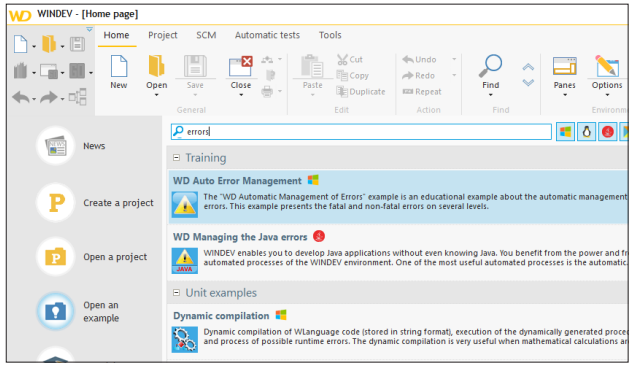
- **complete examples**: these examples correspond to a full application that is using one or more features.
- **training examples**: these examples correspond to a small application used to discover a feature.
- **unit examples**: these examples correspond to a window used to quickly check the use of function.

Practical example

To simplify the presentation of different features, we will be using unit examples or training examples supplied with WINDEV.

▶ To open a complete example, a training example or a unit example from the WINDEV home page:

1. Display the home page (Ctrl + <).
2. On the home page, click "Open an example". The list of complete examples, training examples and unit examples supplied with WINDEV is displayed. These examples are grouped by type of example (complete, training, ...).
3. You can enter a keyword in the search area (for example, enter "Errors"). Only the examples containing this word are listed.



4. To open an example, double-click the example name.
5. For the complete and training examples, the current project is automatically closed and the sample project is opened.
6. For the unit examples, the associated window is opened in the current project.

Lesson 7.2. Automatic management of errors

This lesson will teach you the following concepts

- What is the automatic management of errors?
- Using the automatic management of errors.

Estimated time: 10 mn

Overview

The errors can be automatically managed by WINDEV. This feature helps you reduce the number of code lines while centralizing the management of errors.
The use of this feature also makes the code easier to read.

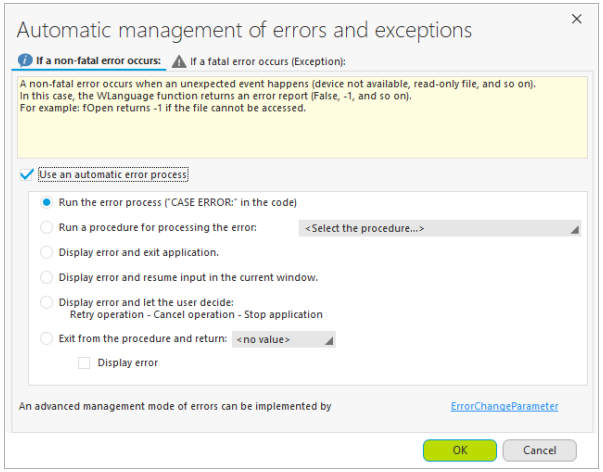
Operating mode

Two operations are performed when an error is detected by a WLanguage function:
- an error value is returned by the function (for example, ***fOpen*** returns "-1" if the specified file was not opened).
- the error is detected by WLanguage (the ***ErrorOccurred*** variable is set to True) and the error details are returned by ***ErrorInfo***.

This second operation can be automatically managed by the error management of WINDEV.

Implementation

The automatic management of errors can be configured:
- in the code editor: all you have to do is click the link "If error: By program" in the code editor:



- by programming with **ErrorChangeParameter**.

Types of affected errors

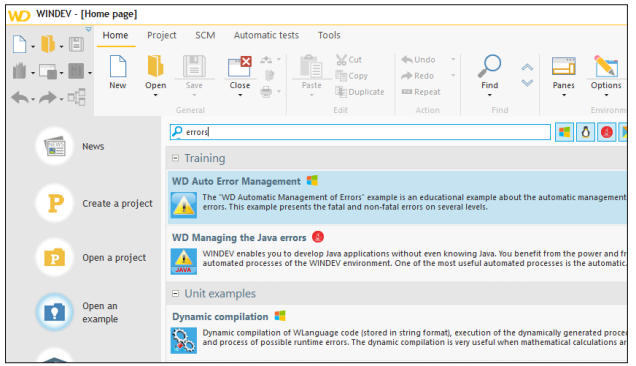
Two types of errors can occur in WLanguage:
- the "non-fatal" errors (also called runtime errors): in most cases, these errors are managed in the code and they do not stop the application. For example, opening a file that cannot be accessed or an archive that does not exist.
- the "fatal" errors (also called programming errors): in most cases, these errors are linked to a development problem (access to a non-declared file, use of non-existing controls ...). A "fatal" error can also occur after a "non-fatal" error that was not processed properly. In this case, the application will be stopped.

The mechanism for managing errors is used to manage these two types of errors according to different methods in order to specify behaviors adapted to the errors that occur.

Automatic management of errors: a training example

▶ To understand the different error cases, we will be using a training example supplied with WINDEV.

1. Display the WINDEV home page if necessary (Ctrl + <).
2. On the home page, click "Open an example". The list of complete examples, training examples and unit examples supplied with WINDEV is displayed.
3. Type "Error" in the search area. Only the examples containing this word are listed.

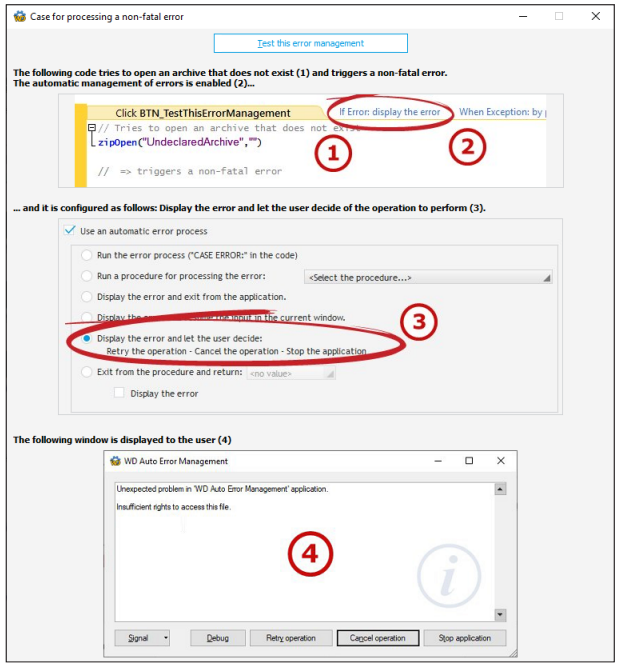


4. Select the "WD Auto Error Management" project. The project is loaded.

▶ This project presents the management:
- of a non-fatal error (opening an archive that does not exist).
- of a fatal error (division by 0).
- of an error on several levels.

► Run the project test by clicking GO (among the quick access buttons).

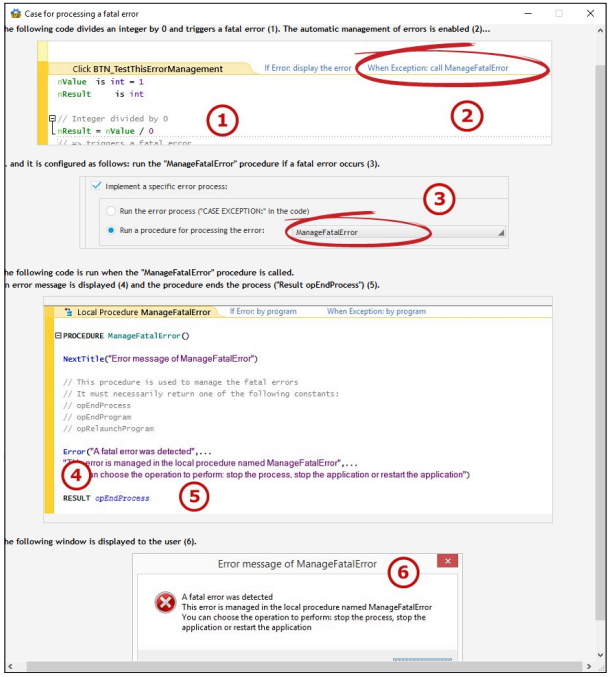
1. Click "Managing a non-fatal error".
2. The following window is displayed.



3. Click the "Test this error management" button.
4. When running the code line that triggers the error, an error message is displayed, allowing the user to retry the operation, cancel the operation or stop the application. The "Debug" option should be used:
   - in test mode, to directly debug the application.
   - in executable mode, to debug an executable directly from WINDEV.
5. Click "Cancel the operation" and close the window.

► Click "Managing a fatal error".

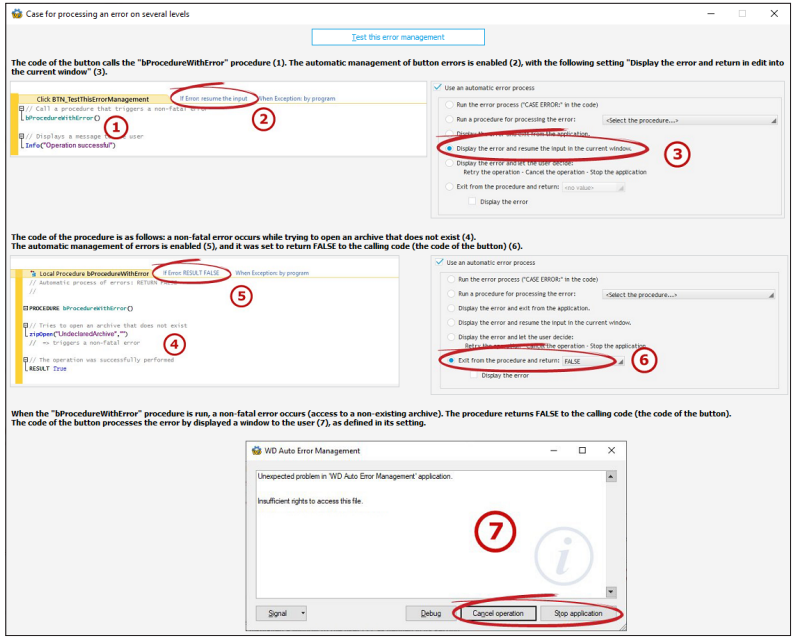
1. The following window is displayed. This window is used to check the error when an integer is divided by 0.



2. Click the "Test this error management" button.
3. When running the code line that triggers the error, a procedure is automatically called. This procedure is used to display the error message and to stop the current process.
4. Click "OK" and close the window.

▶ Click "Managing an error on several levels".

1. The following window is displayed. This window is used to test an error on several levels (process calling a procedure that opens an archive that does not exist).



2. Click the "Test this error management" button.

3. When running the code line that triggers the error:

- the procedure returns "False" to the calling process.
- the calling process displays an error message and stops the process.

4. Click the "Cancel the operation" button.

▶ Stop the application test.

Example

WINDEV also proposes an automatic management of HFSQL errors. See the "Detecting HFSQL errors" training example (supplied with WINDEV) for more details. This example is accessible from the WINDEV home page.

Lesson 7.3. Handling external files

This lesson will teach you the following concepts

- Handling text files (CSV , ...).
- Handling directories.
- Handling XML files.
- Handling XLS files.

Estimated time: 20 mn

Overview

WINDEV allows you to easily handle the most common data formats:
- text files (INI, CSV, ...),
- XML files,
- XLS files, ...

Several families of WLanguage functions can be used to read and create these files. This allows you to easily read data generated by another software in your WINDEV applications or to create files that require a specific formatting.

Furthermore, several WLanguage functions can be used to handle the directories containing the files.

In this chapter, we will focus on:
- Text files (text files, INI files and CSV files),
- XML files,
- XLS files,
- the management of directories.

Handling text or CSV files

Overview

The external files are also called text files or files with direct access. In most cases, this type of file contains text but it can also contain binary information such as images, sounds, ...

In this lesson as in the entire WINDEV documentation, we will be talking of **external files**.

WINDEV allows you to handle the external files by programming. The WLanguage functions can be used to create, read ... external files. All functions can easily be identified: they all start with the letter "f".

The notion of "record" does not necessarily exist in an external file. To handle an external file, you must know its structure, which means how the data is organized inside the file.

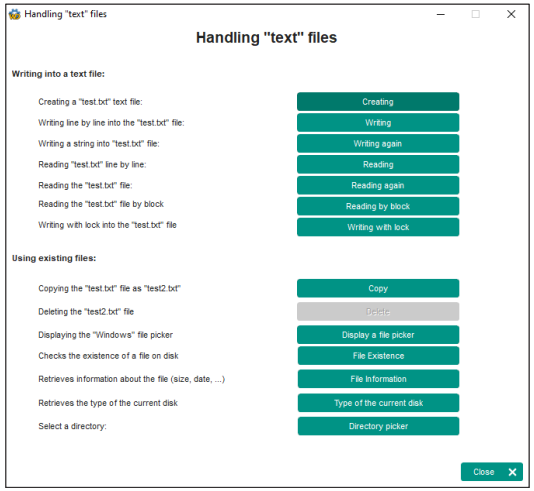
The CSV files are text files that use a specific structure. Therefore, they can be handled like the text files.

The .INI files are also text files that use a specific structure. To handle them, WINDEV includes two specific functions: **INIRead** and **INIWrite**.

Practical example

▶ Open the unit example named "Handling text files".

▶ Run the test of "WIN_Text_File" window. This window groups the different operations that can be performed on the external files with WINDEV.



The different window buttons propose to:
- Create a text file.
- Read and write in a text file.
- Operations performed on existing files (copy, delete, read information, ...).
- Copy a text file.

See the online help for more details (keyword: "External file").

Handling directories

Several WLanguage functions are used to handle the directories and their files.

Practical example

▶ Open (if necessary) the unit example named "Handling directories" and run the test of corresponding window.

▶ This window groups the operations that can be performed on disks and directories:
- create a directory,
- find out the current directory,
- check the existence of a directory,
- list the directories,
- copy and create a directory, ...

See the online help for more details (keyword: "External file").

Handling XML files

Overview

XML (Extensible Markup Language) is a markup language, which means a language that presents information enclosed in tags. XML is a **metalanguage** that is used to invent new tags to isolate the elementary information that may be found in a Web page.

XML is used to structure a document containing data. A HFSQL data file containing several items and records can be exported to an XML file for example (**XML*** functions).

WINDEV supports:

- the files in XML format via a native access supplied with WINDEV. See the online help for more details.
- the exports to XML (***TableToXML***, ***TextToXML***, ***HExportXML***).
- import of XML data (***HImportXML***).
- the use of an XML document via the advanced ***XMLDocument*** type and via the WLanguage functions starting with XML.

The XML file can also be handled in the code editor directly. Simply:

1. Drag the XML file from the file explorer and drop it in the "Project explorer" pane tab, "External descriptions" folder.

2. Drag the XML file from the "Project explorer" pane and drop it in the code editor. The xmlDocument variable is automatically created as follows:

```
<Variable name> is xmlDocument, description =  <Document name>
```

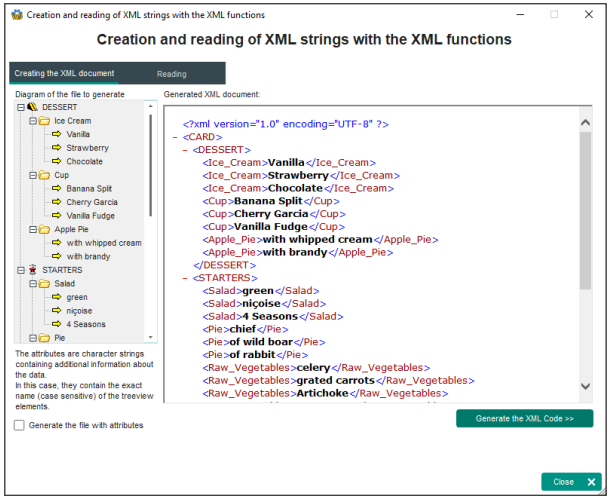
3. You now have the ability to access the variable nodes by their names. These names are automatically proposed by the mechanism for automatic completion of the code editor.

Practical example

▶ Open the unit example named "Reading and writing in XML format".

▶ Run the test of "WIN_HandleXML" window. This window is used to:

- create an XML file. This file is created by the XML functions.



- read an XML file.

▶ Study the code associated with each button.

Remark

We won't go into details about all the features proposed by the XML functions. See the online help for more details.

Handling XLS files

To handle the Excel files, WINDEV proposes:

- the xlsXXX functions. These functions are used to read the data found in the different worksheets of an Excel file.
- the advanced variables (xlsDocument, xlsRow, xlsColumn, xlsCell).

Practical example

A quick test of these functions?

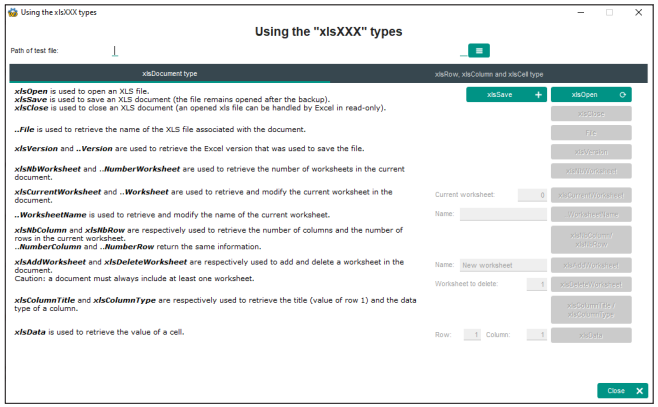
► To check the use of xlsXXX functions, open the unit example named "The Excel functions".

► Run the test of "WIN_XLSFunction" window. This window is used to:
- import data coming from an Excel worksheet.
- export data from a Table control to Excel.

Return to the editor to see the code of the different Button controls in the window.

► To check the use of advanced variables, open the unit example named "XLS type".

► Run the test of "WIN_XLS_Type" window.



► Go back to the editor and study the code of different Button controls in the window.

Remark

We won't go into details about all the features proposed by the XLS functions and the advanced types.
See the online help for more details.

Lesson 7.4. Dynamic compilation

This lesson will teach you the following concepts

- Overview.
- Drawing a line in dynamic compilation.

Estimated time: 20 mn

Overview

The dynamic compilation is used to compile a code at any time in the application. A common example? Your application contains a formula that can be configured. To change the parameters, there is no need to provide the executable again: the modification of a text file is sufficient.

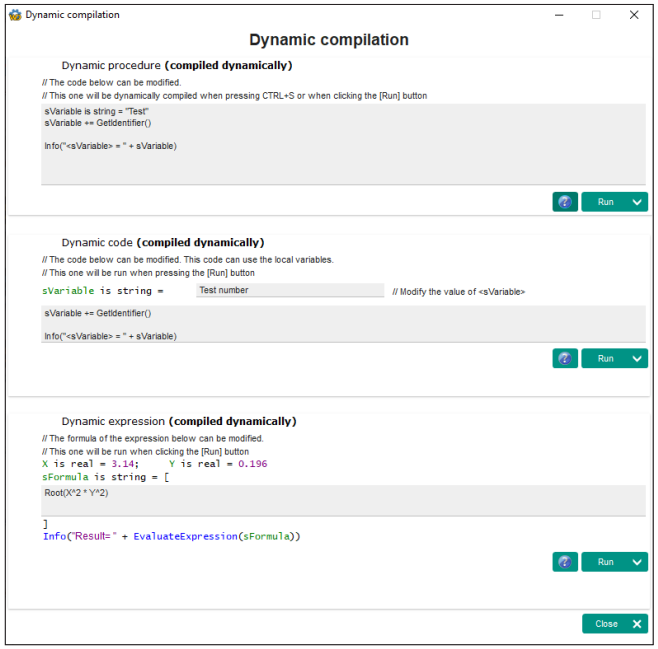
Example

The dynamic compilation will be presented via the unit example named "Dynamic compilation".

The "Dynamic compilation" window explains how to dynamically compile WLanguage code (stored in string format), run the procedure that was dynamically generated and process the possible runtime errors.

▸ To run the test of this window:

1. Open the unit example named "Dynamic compilation".
2. Run the test of "WIN_DYNAMIC_COMPILATION" window.



3. Click the different "Run" buttons to see the different cases.
4. Stop the test.

▸ Let's go back to the code editor and study the code of first "Run" Button control.

1. Display the code of the "Run" Button control (right click on the control, "Code" option). This code calls the ***CompileDynamicCode*** procedure.
2. Position the mouse cursor on the procedure name and press F2. The procedure WLanguage code is automatically displayed in the code editor.
The code of this procedure can be divided into several sections:
   1. Initializing variables.
   2. Compiling the code.

```
sCompilationResult = Compile(DYNAMIC_PROCEDURE, ...
 EDT_COMPIL_CODE)
```

This code contains several important points:

- The function is compiled by the WLanguage function ***Compile***. The function that is dynamically compiled becomes usable as soon as this function is called (and if no error is returned).
- This function expects two parameters: the name of compiled procedure ("DYNAMIC_PROCEDURE") and the code to compile. In this case, the code to compile is found in the EDT_COMPIL_CODE edit control.
- The compilation result is assigned to a sCompilationResult string.

3. Checking the compilation result.

```
// Checks the compilation result
SWITCH sCompilationResult
 // No error
 CASE ""
     bCompilationResult = True
     STC_ERROR_CODE_COMPILE = ""

 // Fatal compilation error
 CASE "ERR"
     bCompilationResult = False
     STC_ERROR_CODE_COMPILE = ErrorInfo()
 // Incorrect code
 OTHER CASE
     bCompilationResult = False
     STC_ERROR_CODE_COMPILE = sCompilationResult
END
```

3. Press Ctrl + F2. The "Click" event of the "Run" Button control re-appears in the code editor. In the rest of this code, you can see that the function that is dynamically compiled is run by ***Execute***.

Lesson 7.5. Windows event

This lesson will teach you the following concepts

- Programming the Windows events.

Estimated time: 10 mn

Introduction

Each action performed by Windows corresponds to a Windows event. Different types of events can occur, for example:
- A window is hovered by the mouse,
- The system is stopped,
- A dialog box is displayed,
- A software error,
- Etc.

When these events occur, they can be intercepted in order to prepare or to run a specific process.

WINDEV proposes an automatic management of most common events. For example, the following events are automatically proposed for an edit control:
- Initializing the control,
- Entry in the control,
- Modifying the control,
- Exit from the control.

To manage additional events, you can:
- use the optional events proposed by WINDEV.
- use the Windows events.

Practical example

The management of events will be presented via the unit example named "The Event function".

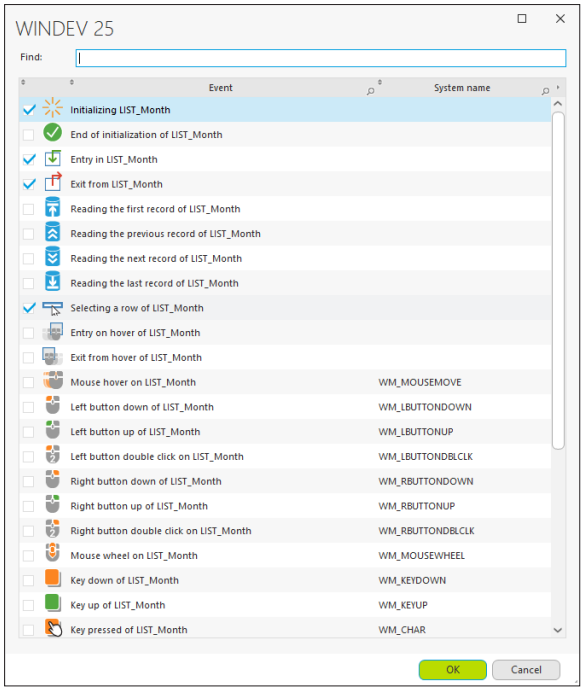
▶ Open the unit example named "The Event function".

Optional events proposed by WINDEV

WINDEV proposes many optional events for each element (window, control, etc.).

▶ To add an optional event:
1. Display the WLanguage events associated to the List Box control in the unit example window:
   - Select the List Box control.
   - Press F2.
   - The code editor is displayed.
2. Click the "Add other events..." link:

3. The complete list of available optional events is displayed:



4. To add an event, simply check the corresponding box and validate this window. Add the "Key Pressed" event for example.

Windows events

To manage more "specific" events, you have the ability to use the WLanguage **Event** function. **Event** is used to associate a WLanguage procedure to a Windows event.

Remark

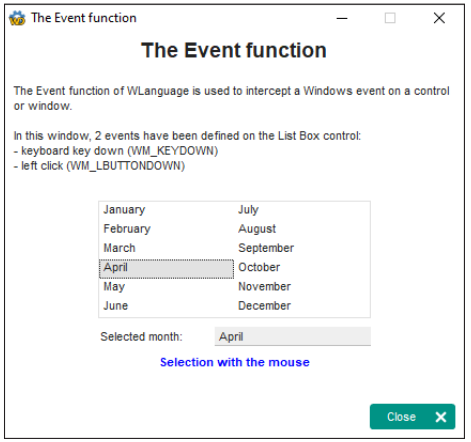
To use **Event**, you must be familiar with the Windows programming, especially the Windows events.

See the WINDEV online help to find out the non-exhaustive list of Windows events (keyword: "Windows API, Value of constants of Windows API").

Example: Detect the click on a List Box control

▸ Run the test of "WIN_Event_Function" window. This window detects if the list is manipulated with the mouse or with the keyboard.

1. Click the List Box control with the mouse.
2. Use the mouse to move the selection bar.
3. A message is displayed, specifying whether the mouse or the keyboard was used.



4. Stop the test and go back to the editor.

▸ Let's study the code used:

1. Click in the window.
2. Display the window events (press F2).
3. Let's study the event "Global declarations" of WIN_Event_Function.
   - First of all, the code line:

```
EXTERN "WINCONST.WL"
```

This code line is used to include the content of WINCONST.WL file in the application code via the EXTERN keyword. This file contains the declaration and values of Windows constants. During the compilation, the entire code found in the WINCONST.WL file will be automatically included in the application code.

   - Then, all supported events are declared:

```
// Events on LIST_Month control
// Keyboard key down
Event("MouseOrKeyboard", LIST_Month..FullName, ...
  WM_KEYDOWN)
// Left mouse click
Event("MouseOrKeyboard", LIST_Month..FullName, ...
  WM_LBUTTONDOWN)
```

The procedure ***MouseOrKeyboard*** is called whenever the keyboard is used on the List Box control (corresponding Windows event: WM_KEYDOWN) or whenever the left mouse click is used (corresponding Windows event: WM_LBUTTONDOWN).

4. Display the WLanguage code of the procedure:
   - Position the mouse cursor on "MouseOrKeyboard".
   - Press F2.
5. The procedure code is straightforward:
   - If the keyboard is used, the caption displayed below the List Box control contains "Selection with the keyboard".
   - If the mouse is used, the caption displayed below the List Box control contains "Selection with the mouse".

```
PROCEDURE MouseOrKeyboard()
// The _EVE.wMessage variable contains the message number
SWITCH _EVE.wMessage
  // Keyboard
  CASE WM_KEYDOWN
    // Message indicating that the keyboard is used
    STC_SelectionType = "Selection with the keyboard"
    STC_SelectionType..Color = LightRed

  // It's the mouse
  CASE WM_LBUTTONDOWN
    // Message indicating that the mouse is used
    STC_SelectionType = "Selection with the mouse"
    STC_SelectionType..Color = LightBlue
END
```


Lesson 7.6. The threads

This lesson will teach you the following concepts

- What are the threads?

Estimated time: 10 mn

Definition

The threads are used to run a code (or processes) in parallel of main application. Therefore, several long processes can be run in background task without locking the main application (also called "Main thread").
The threads replace some types of timers.

In most cases, a secondary thread is used to detect an event such as a user action, an incoming email, a phone call, ...

Examples for using threads:
- Retrieving emails in background task while typing a new email.
- Communication application: managing the phone calls, communication by socket, ...

WINDEV allows you to:
- manage the threads (***Thread****** functions).
- use "signals" in order to synchronize several threads (***Signal****** functions).
- use the threads with "semaphores" in order to manage the access to the resources shared by different threads (***Semaphore****** functions).

Example

Example

WINDEV is supplied with several unit examples allowing you to understand the benefit and the use of threads:
- The threads (Pool).
- The threads.

See the online help for more details (keyword: "Thread").

Lesson 7.7. The sockets

This lesson will teach you the following concepts

- Overview.
- Server application.
- Client application.

Estimated time: 20 mn

Overview

WINDEV proposes several functions used to perform an advanced management of sockets.

A socket is a communication resource used by the applications to communicate between computers regardless of the network type.

This communication mode is used to establish a communication between two applications found on different computers (connected by Internet or on the same local network) or on the same computer.

A WINDEV application can manage the sockets according to different modes:

- Client WINDEV application: the application connects to any server and it exchanges data via a socket.
- WINDEV, WEBDEV or WINDEV Mobile "Server" application: the WINDEV, WEBDEV or WINDEV Mobile application is a server, exchanging information by sockets with several client computers. The threads must be used to manage several simultaneous connections.

Example

WINDEV is supplied with a training example allowing you to understand the use of sockets: "WD Using sockets". This example is accessible from the WINDEV home page (Ctrl + <).

Server application: for a simplified server

WINDEV gives you the ability to create a simplified socket server. **This server allows you to communicate with a single client computer at a time**. This type of application is very useful when two remote applications must communicate.

The steps for creating a simplified server are as follows:

1. Creating the socket.
2. Exchanging data.
3. Closing the socket.

Creating the socket

To create the socket, the server is using **SocketCreate**. A socket is associated with a specific port. For a simplified use of socket by programming on the server, specify the socket name.

The client computer will connect to this socket in order to exchange data. The connection between the two computers will be actually established during the first exchange of data between the two computers (which means when the server reads information for the first time).

The connection is established during the first successful attempt of **SocketRead** on the server.

Exchanging data

When two computers are using the same socket, a communication stream is established between these two computers. These two computers can read and write character strings on the socket.

Remark: To avoid locking the applications, the management of incoming messages is often performed by a specific thread.

To read and write on the socket, the WINDEV server application must use ***SocketRead*** and ***SocketWrite***.

Caution: To perform a read operation, a write operation must have been done beforehand. For example:

1. The client computer writes into the socket: it sends a request to the server.
2. The server performs a read operation on the socket in a thread. As soon as a message is received, the message is processed by the server.
3. If a response to the message is required, the server identifies the client computer (***SocketClientInfo***) and sends a response to it.

Closing the socket

To end the communication, the server can close the socket with **SocketClose**.

Remark: the socket can also be closed by the client computer.

Client application

A client application of a socket server connects to a standard server in order to exchange information via socket.

Example: A client WINDEV application can connect to a standard news server on Internet.

The steps for creating a client application are as follows:

1. Connecting to the server.
2. Exchanging data.
3. Ending the communication.

Connecting to the server

To connect to a server socket, all you have to do is use **SocketConnect**. This function is used to perform a request for connecting to the server.

The socket is identified by its port and by its address.

Exchanging data

When two computers are using the same socket, a communication stream is established between these two computers. These two computers can read and write character strings on the socket.

Remark: To avoid locking the applications, the management of incoming messages is often performed by a specific thread.

To read and write on the socket, the WINDEV client application must use ***SocketRead*** and ***SocketWrite***.

Ending the communication

To end the communication, all you have to do is close the socket from the client computer with ***SocketClose***.

Remark: you also have the ability to end the communication from the server.

Practical example

The programming of sockets will be presented via the unit example named "Using sockets".

Example test

▶ Open the unit example named "Using sockets".

▶ Run the test of "WIN_Socket" window. A message will be sent from computer B to computer A. Computer A is the Server application and computer B is the Client application.

1. On computer A, click the "Create" button to create the socket.
2. Computer B can connect to the socket created by computer A. All you have to do is click the "Connect" button (in Computer B section).
3. Computer B sends a message to computer A:
   - Type message to send in the "Sentence to send to computer A" area.
   - Click the "Send" button found in the "Computer B" area.
4. To retrieve the message on computer A, click the "Get" button found in the "Computer A" area.
5. Click the "Disonnect" buttons to disconnect the two computers.

▶ Stop the window test to go back to the editor.

Studying the code used

▶ Let's study the code of the different buttons that have been used.

▶ First of all, we are going to study the processes performed by the socket server (computer A).

1. Display the code of "Create" button found in the "Computer A" area:
   - Select the "Create" button.
   - Press F2 to display the events.
2. In the "Click" event, you will see ***SocketCreate***, which is used to create the socket. Close the code editor.
3. Display the code of "Get" button found in the "Computer A" area:
   - Select the "Get" button.
   - Press F2 to display the events.
4. The following code is used in the "Click" event:

```
EDT_SentenceReceivedFromComputerB = SocketRead ...
("ComputerA", False, 2000)
IF EDT_SentenceReceivedFromComputerB <> "" THEN
  Info("Message received from IP address # "+...
      SocketClientInfo("ComputerA", SocketAddress))
END
```

You will notice the presence of ***SocketRead*** that is used to read the socket that was created beforehand. The message read is immediately displayed in the "EDT_SentenceReceivedFromComputerB" control. Close the code editor.

▶ Let's study the processes performed by the client (computer B).

1. Display the code of "Connect" button found in the "Computer B" area:
   - Select the "Connect" button.
   - Press F2 to display the events.
2. In the "Click" process, you will see ***SocketConnect***, which is used to connect to the socket created by computer A. Close the code editor.
3. Display the code of "Send" button found in the "Computer B" area:
   - Select the "Send" button.
   - Press F2 to display the events.
4. The following code is used in the "Click" event:

```
IF NOT SocketWrite("ForComputerA", ...
      EDT_SentenceToSendToComputerA) THEN
  Error(ErrorInfo(errMessage))
  RETURN
END
```

You will notice the presence of ***SocketWrite*** that is used to send a message onto the socket to which computer B is connected. Close the code editor.

Remark: In this lesson, we have seen a "simple" communication between a server and a client computer: the client sends messages and the server processes the messages. You have the ability to create more complex applications.

You have the ability to establish a communication between two applications, both client and server. In this case, the management of threads becomes very important to manage the send operations and the responses.

Lesson 7.8. The FTP

This lesson will teach you the following concepts

- Presenting the FTP functions of WINDEV.

Estimated time: 20 mn

Overview

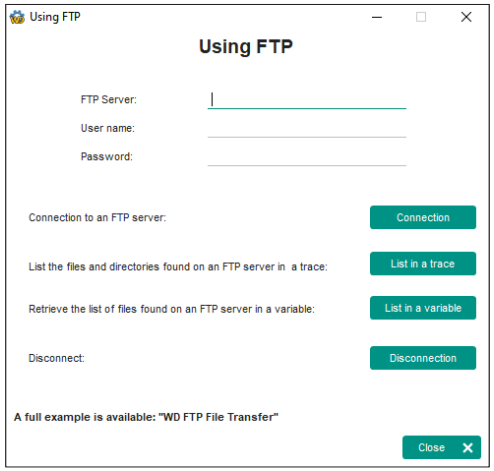
The FTP (File Transfer Protocol) is a standard protocol used to transfer files from a computer to another one. One of the computers must be an FTP server.

Several WLanguage commands allow you to transfer files by using this protocol with a server. These programming functions start with "FTP".

WINDEV only proposes "client" functions for the FTP. A standard FTP server is required.

The unit example named "The FTP functions" presents the operations that can be performed on the files found on an FTP server.

▶ Open the unit example named "The FTP functions". This example presents the main features that can be used on an FTP server.



Connecting to an FTP server

FTPConnect is used to connect to an FTP server. An FTP account (user name and password) is required to access an FTP server.

▶ Display the WLanguage events associated with the "Connection" Button control in "WIN_FTP":

1. Select the "Connection" Button control.
2. Press F2 to display the associated WLanguage code:

```
// 1 - FTP address: corresponds to the FTP site
// to which the application must connect. Ex: 192.108.10.2
sFTPAddress is string = EDT_FTP_Server
// 2 - User name: if this name is not specified,
// an "anonymous" connection will be used
sUserName is string = EDT_User
sUserPWD is string = EDT_Password
// Connection to FTP server
```

```
gnConnectionID = FTPConnect(sFTPAddress, sUserName, ...
sUserPWD)
IF ErrorOccurred THEN
  // An error occurred during the connection
  Error("The connection to the FTP server: " + ...
      sFTPAddress + " failed", ErrorInfo())
  RETURN
END
// The connection is established
Info("The connection to the FTP server is established", ...
  "connection identifier: " + gnConnectionID)
```

Remark: You also have the ability to specify the port number for connecting to the FTP server ("21" by default) as well as the connection mode ("True" for a "passive" connection, "False" for an "active" connection).
See the online help for more details (keyword: "FTP, Functions").

Sending a file

To send a file to an FTP server, all you have to do is use ***FTPSend***.
Let's see a code sample that can be used:

```
// When connecting to the server with FTPConnect, we have
// retrieved the connection number in the gnConnectionID variable
// Transfer the "C:\MyDocuments\File.DOC" file to
// the "Temp" directory found on the server.
bResult is boolean = FTPSend(gnConnectionID, ...
  "C:\MyDocuments\File.DOC", "/Temp")
```

Caution!

Pay great attention to the case (uppercase/lowercase characters) for the name of directories on the server. Indeed, some FTP servers operate under UNIX and are "case sensitive", which means that the case is taken into account for the name of files and directories.

For example, a directory named "MyDirectory" is found on the FTP server. If you try to access "mydirectory", an error such as "Path not found" will be returned by the FTP server because the case is incorrect.

Listing the files found on an FTP server

FTPListFile is used to list the files found on an FTP server. This function is using a "callback" procedure. The procedure is run for each file or directory found.

▸ Display the WLanguage events associated with the "List" Button control in "WIN_FTP":

1. Select the "List" Button control.
2. Press F2 to display the associated WLanguage code:

```
FTPListFile(gnConnectionID,"*.*", ...
  "CallBackFTPListFile", ftpFile+ftpDirectory)

// Check the function execution
IF ErrorOccurred THEN
  Error("Error while browsing the files found on the FTP server", ...
ErrorInfo())
  RETURN
END
```

3. In the example, the procedure called is used to display the files found in a trace window.

```
PROCEDURE CallBackFTPListFile(sFileName, nFileSize <useful>, ...
  sAttribute, sModifDate <useful>, sModifTime <useful>)

// Is it a file or a directory
IF sAttribute = "D" THEN
  // Directory or sub-directory
  Trace("Directory: " + sFileName)
ELSE
  // File
  Trace("File: " + sFileName)
END

// Continue to browse the files
RESULT True
```

Retrieving a file

To get a file from an FTP server, all you have to do is call ***FTPGet***.
Let's see a code sample that can be used:

```
// When connecting to the server with FTPConnect, we have
// retrieved the connection number in the gnConnectionID variable
// Download the "/Document/File.DOC" file found
// on the FTP server to the "D:\Temp" directory on
// the current computer
bResult is boolean = FTPGet(gnConnectionID, ...
  "/Document/File.DOC", "D:\Temp")
```

Disconnecting from an FTP server

To disconnect from an FTP server, use **FTPDisconnect**.

▸ Display the WLanguage events associated with the "Disconnection" Button control in "WIN_FTP":

1. Select the "Disconnection" Button control.
2. Press F2 to display the associated WLanguage code:

```
IF YesNo(No, "Do you want to close the connection to the FTP ser
ver?") = Yes THEN
  // Disconnect from the FTP server
  FTPDisconnect(gnConnectionID)
  IF ErrorOccurred THEN
      // Error while closing the connection
      Error("Unable to close the FTP connection", ErrorInfo())
      RETURN
  END

  // The connection is ended
  Info("The connection to the FTP server is ended")
END
```

Other FTP functions are available but we won't go into details about them in this tutorial. They are mainly used to:

- create, delete, modify directories on the FTP server,
- create, delete, modify files on the FTP server,
- retrieve information about a directory and/or a file,
- ...

See the online help for more details (keyword: "FTP, Functions").

Lesson 7.9. The OOP

This lesson will teach you the following concepts

- Concepts of Object-Oriented Programming.
- Examples of object declaration.
- Associated UML model.

Estimated time: 30 mn

Concepts

The Object-Oriented Programming (OOP) is designed for better code reusability. The programs developed in OOP are structured: they include modules, each one managing a software feature. These modules can easily be re-used in other software. They group a set of procedures (called methods) and they encapsulate the data structure on which the methods will act.

To use the object-oriented programming, you must declare the classes, the members and the associated methods.

The classes

A **class** contains the description of a data structure (the members) and the procedures (methods) that handle the members.

Therefore, a class defines a type of data and its behavior.

The objects

A class is used to create **objects**. Each created object owns the members described in its class and it can be handled via the methods of its class. An object is defined as a class instance.

When the class is declared, all you have to do is associate an object with a class in order for the object to be handled by all methods of this class.

The members

A **member** is a data (or parameter) of the object.

The methods

A **method** is used to act on the object, to modify its members for example.

A method is a procedure. Its operating mode is similar to the one of standard WLanguage procedures.

Concept of inheritance

The inheritance is used to include the characteristics of an existing class (base class) into a new class (derived class). The inheritance allows you to create a new data type from a known type in order to add features to it or to modify its behavior. Therefore, the **base class** will not be modified. A class can inherit from a class: it becomes a sub-class of this class.

The objects found in a **derived class** can access all methods, all members and all properties of ancestor classes ; it is as if the methods, the members and the properties of ancestor classes were part of the derived class.

Constructor and Destructor

The notions of **Constructor** and **Destructor** are important because they allow for an automatic call to initialization methods when creating an object and when destroying it.

The Constructor method associated with a class is automatically called when declaring a class object.

The Destructor method associated with a class is automatically called when deleting the object (exit from the procedure where the object was declared).

Data encapsulation

The data encapsulation is used to ensure that the data belonging to the object is not accidentally modified by functions (methods) external to the object.

This allows you to prevent the user of an object from accessing some or all of its members. The members whose access is forbidden are called private members.

These private members can only be accessed from the methods designed for this purpose in the class.

Example

Let's take a simple example to apply these concepts:

- Let's consider the PERSON class.
- Florence is an instance of PERSON class.
- The last name, first name and date of birth can be members of PERSON class.
- The Age() method can be a method of PERSON class. It would calculate the age according to the "date of birth" member and according to todays' date (returned by **DateSys**).
- Inheritance: A contact can be either a person, or a company.
  - PERSON could be a class derived from CONTACT.
  - COMPANY could be a class derived from CONTACT.

Creating an object-oriented program

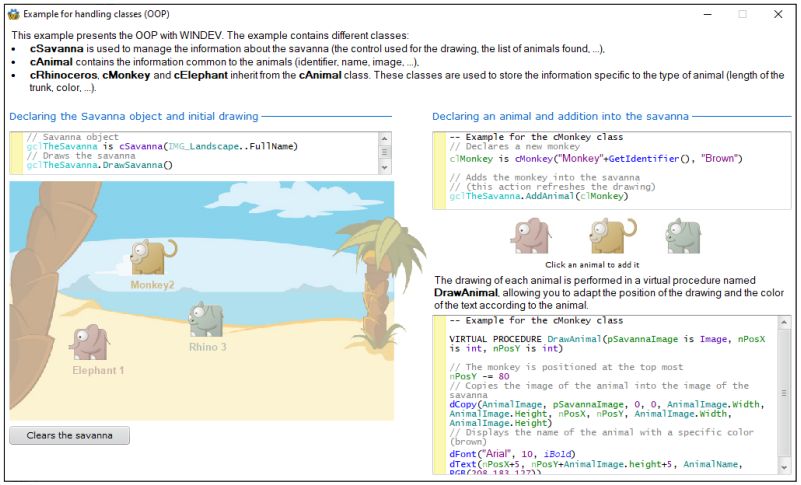
To create an object-oriented program in WLanguage, you must:

1. Describe the class and the class members
2. Specify all class methods.
3. Declare the objects by associating them with a class ("instantiate a class").

Simple example

▸ To illustrate these concepts, we will be using the training example named "WD Simple OOP".

1. Open the training example named "WD Simple OOP".
2. Run the test of this example.



3. Click the different buttons corresponding to the animals in order to add them.
4. Stop the example test to go back to the editor.Remark: We won't go into details about the syntax of OOP but we will present a simple example of an object-oriented program.

See the online help for more details (keyword: "OOP, Class").

Declaring a class

▸ WINDEV allows you to easily declare the classes from the "Project explorer" pane. To create a class:

1. In the "Project explorer" pane, select the "Classes" folder
2. Display the popup menu of this folder (right mouse click) and select "Create a class".
3. In the window that is displayed, specify the class name ("TestTUT" for example) and validate.

▸ Study the WLanguage code of ***cSavanna*** class used in the example.

1. In the "Project explorer" pane, select the "Classes" folder
2. Open the "Classes" folder (to do so, click the arrow found in front of the folder name).
3. Double-click the ***cSavanna*** class.

4. The class WLanguage code is displayed in the code editor. The declaration code of the class is as follows:

```
cSavanna is Class

  PROTECTED
      // Name of Image control used for the drawing
      m_sNameImageControlForDrawing is string
      // List of savanna animals
      m_arrTheAnimals is array of cAnimal dynamic
END
```

In this code:

- "cSavanna" is the class name.
- "m_sNameImageControlForDrawing" and "m_arrTheAnimals" are class members.
- The "PROTECTED" keyword is used to specify that these members can only be handled from a code of the class or from a code of a derived class.

Describing the methods

▸ WINDEV allows you to easily declare the classes from the "Project explorer" pane. To create a method:

1. Right-click your class in the "Project explorer" pane.
2. Select "New method" from the popup menu.
3. In the window that is displayed, specify the method name and validate.
4. Type the method code in the code editor.

▸ To display the ***AddAnimal*** method of ***cSavanna*** class:

1. Click your class in the "Project explorer" pane.
2. Click the little arrow found in front of the class name: the different class methods are displayed.
3. Double-click the method name:

```
PROCEDURE AddAnimal(pclAnAnimal is cAnimal dynamic)

// No more than 5 animals
IF m_arrTheAnimals..Count = 5 THEN
  Error("No more than 5 animals!")
  RESULT False
END

// Adds the animal to the list
Add(m_arrTheAnimals, pclAnAnimal)

// Draws the savanna
DrawSavanna()

RESULT True
```

Declaring and handling the objects

In the window WLanguage events, an object is declared at the same time as the other variables:

```
// Global declarations of WIN_OOP
PROCEDURE WIN_OOP()

// Savanna object
gclTheSavanna is cSavanna(IMG_Landscape..FullName)
```

To refer to a member of "cSavanna" object, use the following syntax

```
<ObjectName>.<member name>
```

▸ In our example, the object is handled:

- In the "Initializing" event of the window, to build the savanna.

```
// Draws the savanna
gclTheSavanna.DrawSavanna()
```

- When creating an animal, to position the animal in the savanna.

```
// Declares a new elephant
clElephant is cElephant("Elephant "+ GetIdentifier(), 13)

// Adds the elephant into the savanna (this action refreshes the
drawing)
IF gclTheSavanna.AddAnimal(clElephant) THEN
    // End message
    ToastDisplay("Animal added", toastShort, vaMiddle, haCenter)

END
```

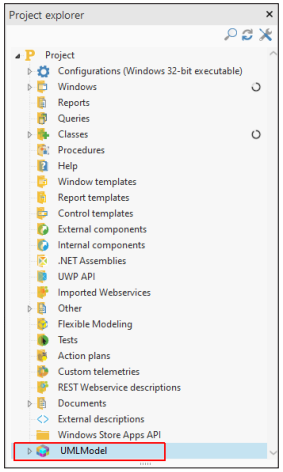
We won't go into details about OOP in this tutorial.

UML diagram

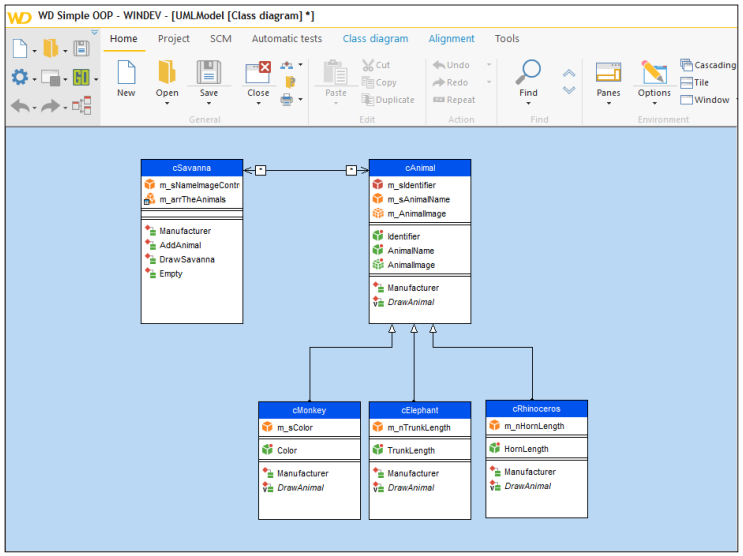
The "WD Simple OOP" example is associated with an UML diagram.

▸ To display the UML diagram linked to the project:

1. Double-click "UML Model" in the "Project explorer" pane:



2. The UML diagram (class diagram) linked to the project is displayed.

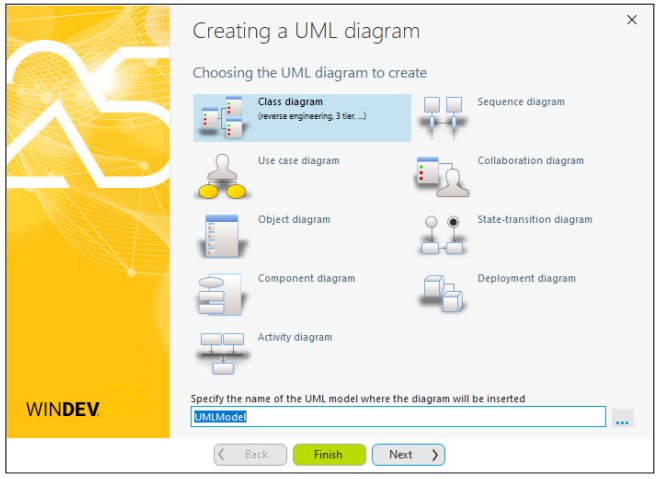


3. The different classes used by the "WD Simple OOP" project are found in this diagram.

Remark

WINDEV allows you to create the 9 types of UML diagrams.
To create a UML diagram:

1. On the "Home" pane, in the "General" group, click "New".
2. The window for creating a new element is displayed: click "Architecture" then "UML".
3. The wizard for creating a UML model starts, allowing you to choose the model to create:



We won't go into details about the use of UML language with WINDEV.
See the online help for more details (keyword: "UML").

Appendices 1. Vocabulary

This lesson will teach you the following concepts

- The terms used by WINDEV, WEBDEV and WINDEV Mobile.

Estimated time: 30 mn

Main terms used

AAF
Automatic Application Features.

Alignment
Method for organizing the controls in a window or page. For example, center a control in a page, define the same width for several controls, ...

Analysis
Description of structure of data files and relationships.

Anchoring
Mechanism that consists in defining positioning or resizing rules in order for the content of a window or page to adapt when resizing a window or a browser.

Application RAD
Fast development method of program from a program template.

Array
Type of variable that contains several values. The values can be accessed via a subscript. The [] characters are used to access the elements of an array.

Assignment
Operation that consists in assigning a value to a variable or control. For example:

```
// Assign the value MOORE to CustomerName variable
CustomerName = "MOORE"
```

The = sign is the assignment operator.

Break (report)
Mechanism that consists in grouping and separating the data according to a value.

For example, create a break in a report that is used to list the customers by city. The customers who live in the same city are grouped together. Visually, we separate the customers for each new city.

Class (OOP)
Element defined in Object-Oriented Programming. A class gathers methods (actions) and members (data).

Classic (data file)
Type of access to an HFSQL data file. An HFSQL data file is Classic when accessed directly in its directory.

Client/Server (data file)
Type of access to an HFSQL data file. An HFSQL data file is Client/Server when accessed by connecting to a server that has this data file via the HFSQL engine.

Control (window or page)
Graphic element used to build the interface of a program or site.

Control template
Container of one or more controls (with processes) that can be indefinitely re-used in pages.

Main template characteristic: if the initial template is modified, the modifications are automatically applied to the different template uses.

Database
Element containing the program data. The data is organized in data files (also called tables).

Data binding
Method used to associate a graphic interface element to a data (variable, item).

Deployment
Consists in installing a program on the user computer.

Editor
Program that is used to create a project element (window editor, page editor, ...).

Event-driven
Type of programming. A user action on a window or on a page induces a code to run. The code of the action to run is typed in the event representing the user action.

For example, the "Button click" event corresponds to the user clicking this button with a mouse.

External component
Software brick that is used to export one or more business rules in order to re-use them.

Data file (Table)
Element that constitutes a database. A data file is used to store the data entered into a program.

For example, a CUSTOMER file will contain the customer information that was entered into a program.

Global (variable or procedure)
Corresponds to the scope in memory of a variable or procedure. A global element is accessible from any other project element. The opposite is local.

GUI (also called UI)
Graphical User Interface (or User Interface). Description of windows or pages that constitute a program. This is what the user sees when using the program.

Homothetic
Method for resizing an image in order to maximize its display without distorting its content.

Index (data file)
Synonym: key

Integrity constraint
Rule associated with a data file item to ensure data consistency in a database.

Item
Element that belongs to the structure of a data file or table (found in an analysis). For example, a CUSTOMER data file can include the FirstName and LastName items.

Internal component
Container grouping elements from a project (window, page, query, report, class, ...) in order to allow and simplify the share with another project.

Key (data file)
Characteristics of a data file item. A key item is used to optimize the searches and the sorts in a data file.
Synonym: index

Link (analysis)
Used to describe the nature of relationship or the common point between 2 analysis data files. The integrity rules to write to the relevant data files can be defined depending on the specified link.

Live Data
Mechanism that consists in displaying real data coming from the database when creating the GUI. This mechanism is only used if the element is linked to the data file.

Local (variable or procedure)
Corresponds to the scope in memory of a variable or procedure. A local element can only be accessed in the process where it was defined. The opposite is global.

Member
Variable belonging to a class or structure.

Method
Procedure belonging to a class used to act on the class data (members).

Native Access (Native Connector)
Method for connecting to a database from a program.

n-tier
Programming method in layers. Each layer is independent and it can be changed without impacting the other ones.
Benefit: Simplified maintenance.
Drawbacks: Difficulty and development time.

Object-Oriented Programming (OOP)
Advanced programming method, opposed to procedural programming.
In OOP, we handle objects, which means grouped sets of variables and methods associated with entities that include these variables and these methods.
In procedural programming, we define functions that call each others. Each function or procedure is associated with a specific process that can be divided into sub-processes until we get basic functions.

OOP
Abbreviation of Object-Oriented Programming.

Parameter (window, page, procedure, method, ...)
Element expected in a window, page, procedure, method, ... during the call to this one. Each value passed in parameter must be assigned to a variable.

Popup
Type of window (or page). A popup is a window (or a page) that is opened above another window (or page). You can still view the content of the window (or page) underneath while performing input in the popup.

Popup menu
Drop-down menu containing possible actions according to the location where the right mouse click occurred and to the type of element on which this click was performed.

Private
Variable or procedure that can only used in the current element.

Project code
Code run when starting a program or a site.

Project configuration
Description of output format of project: Windows executable, Linux, JAVA, ...

Procedure
Project element containing the code of a process to run.

Project
Element that groups all the elements that constitute a program or a site. A project contains for example an analysis, pages, reports, queries, ...

Property (control, window, ...)
Keyword representing an element characteristic. The properties are used to handle and modify the characteristics of project elements by programming.

Public
Variable or procedure hat can be used from all the elements.

Query
Element written in SQL language that is used to access (in read-only or in read/write) the content of a relational database.

RAD
Acronym for Rapid Application Development
Fast development method of program from an analysis (description of data files).

Report
Project element that defines a printout.

Block (report)
Element constituting a report. For example, a Header block, a Footer block, a Body block.

Report template
Container representing a standard report that can be applied to one or more project reports.
Main template characteristic: if the initial template is modified, the modifications are automatically applied to the copies.

SCM
Source Code Manager. Tool for organizing, sharing project resources, managing rights, ...

Style
Element used to describe the graphic style of a control in a window or in a page. A style includes, for example, a font, the character size, the character color, etc.

Style sheet
Contains the list of styles used in a project.

Structure
Type of variable that includes several sub-variables.

Table (control)
Graphic element found in a window or page. A table control includes one or more columns and several rows.

Table (data file)
Element that constitutes a database. A table is used to store the data typed in a program. For example, a CUSTOMER table will contain the customer names and addresses that have been entered in a program.

User groupware
Tool for describing and managing the access rights to the interface for the users of a program or site. For example, prevent a user from clicking a "Delete" button according to his login or to his group.

Variable
Element used to store a program value in memory. Several types of variables are available. Each type corresponds to the nature of the value that will be stored. For example, a string variable to store the name of a person, a currency variable to store an amount.

Webservice
Program installed on a Web server whose processes are accessible via the Web.

XML
Language for organizing data in order to normalize and simplify the exchange of data (mainly used in the exchanges with the Webservices).

Appendices 2. Using SQL data

This lesson will teach you the following concepts

- Creating a project.
- Creating an analysis.

Estimated time: 40 mn

Overview

This lesson allows you to handle the SQL databases with WINDEV.

Remark

This lesson is intended to the users with a good knowledge of SQL databases. Some operations to perform on the SQL database will not be presented in details

To create an application that is using an SQL database, you must:

- Create the project linked to the application. This project will group all application elements (windows, codes, queries, reports, ...).
- Import the description of SQL data files.

Remark

To follow this lesson, you must:

1. Connect to your SQL server.

2. Import onto the SQL server the SQL script files and the image directory (found in the "\Exercises\SQLDatabase\" directory of directory containing the tutorial examples) with the available import tools (MySql, Oracle, SQL Server, ...). The image directory is used to get the images of Product data file.

3. Run the SQL script files on the SQL server.
Remark: The provided SQL scripts operate on Oracle only.

Your SQL database is ready for this lesson.

Creating the project

▶ To create the project:

1. Start WINDEV (if not already done).

2. Display the WINDEV home page (Ctrl + <).

3. In the home page, click the "Create a project" button and select "Windows or Linux application". The project creation wizard starts. The different steps of the wizard help you create your project. The information specified in this wizard can be modified later.

Tip

Tip: To create a project, you can also:

1. Click among the quick access buttons.

2. The window for creating a new element is displayed: click "Project".

4. The first step of the wizard allows you to select the operating system for which the project is intended. Select "Windows platform" and go to the next step of the wizard ("Next").

5. For this lesson, we are going to create a "blank" project. Select "Create a blank project" and go to the next step of the wizard.

6. WINDEV then proposes to specify the platform description. Keep the default options ("Windows 32-bit executable" and "Executable with windows"). Go to the next step of the wizard.

7. The wizard proposes to type the name of project, its location and its description. In our case, this project will be named "sqldatabase".

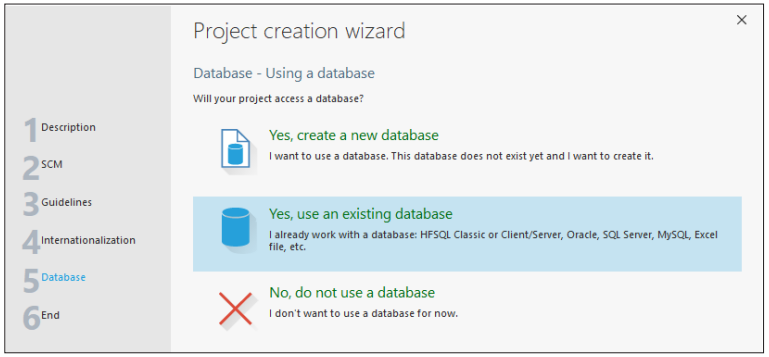
8. The different steps of the wizard are indicated in the menu on the left. These steps can be clicked directly. Since the other steps in the "Description" are not strictly necessary, you can click "Guidelines" directly.

9. This step is used to define the code style. Don't modify the suggested options. Go to the next step.

10. This step is used to define the style book. Select "Elegant". Go to the next step.

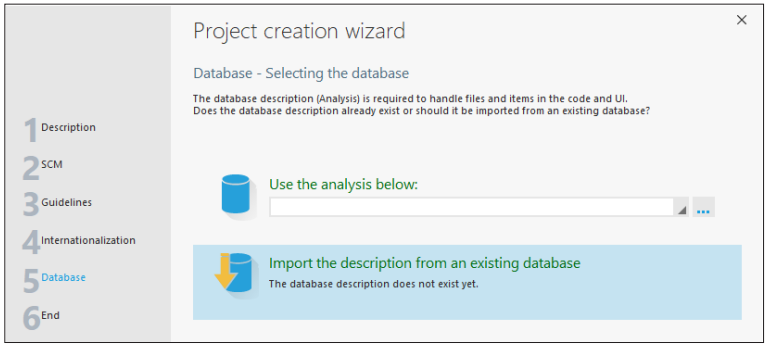
11. Click the "Database" step. This step is used to give information regarding the database.

12. Select "Yes, use an existing database".



Go to the next step

13. Select "Import the description from an existing database".

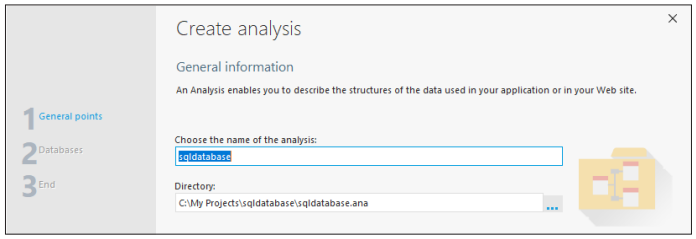


14. The analysis creation wizard starts.

Creating the analysis

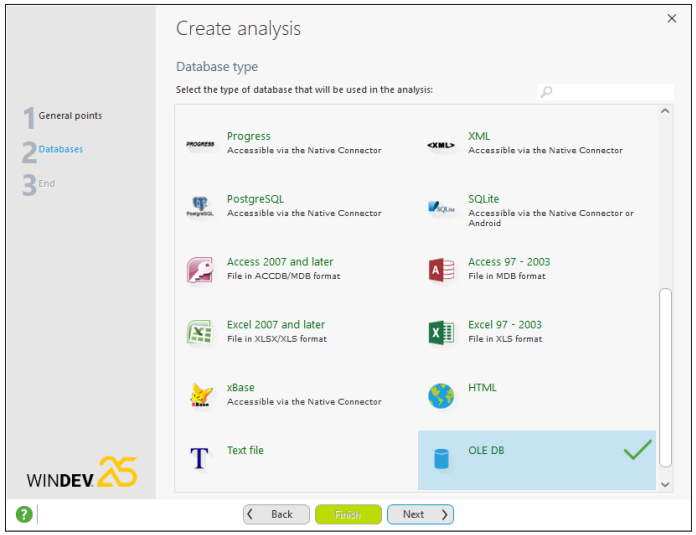
▸ The steps of the analysis creation wizard are as follows:

1. Specify the analysis name and directory. By default, the analysis name corresponds to the project name and the analysis directory is a ".ana" directory in the project directory. We will keep these default parameters.



Go to the next wizard step.

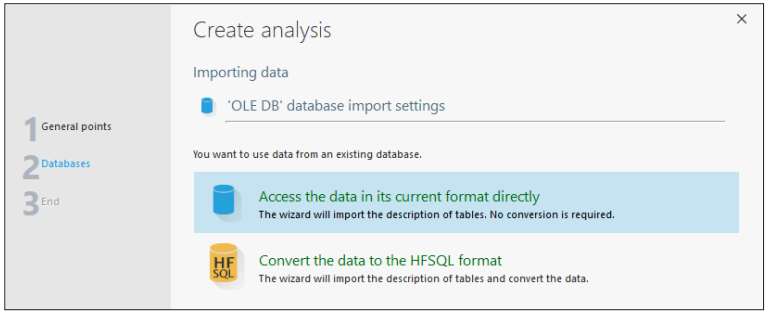
2. Next, you can choose the types of databases used by the project. Select OLE DB (or the SQL database use if you own the corresponding Native Connector).
Remark: Native Connectors are listed first.



Go to the next wizard step.

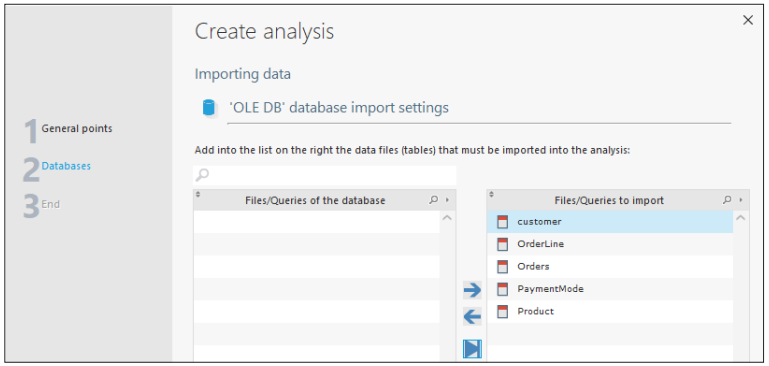
3. Select the OLE DB source corresponding to the type of your database.
Go to the next step of the wizard.

4. In the following steps, specify the parameters of your OLE DB connection (the steps are identical for a Native Access):
   - the connection name (the caption associated with the connection is optional). Go to the next step.
   - the name of database server. Go to the next step.
   - the user name and password. Go to the next step.
   - the database name if necessary. Go to the next step.
5. Specify the mode for accessing data.



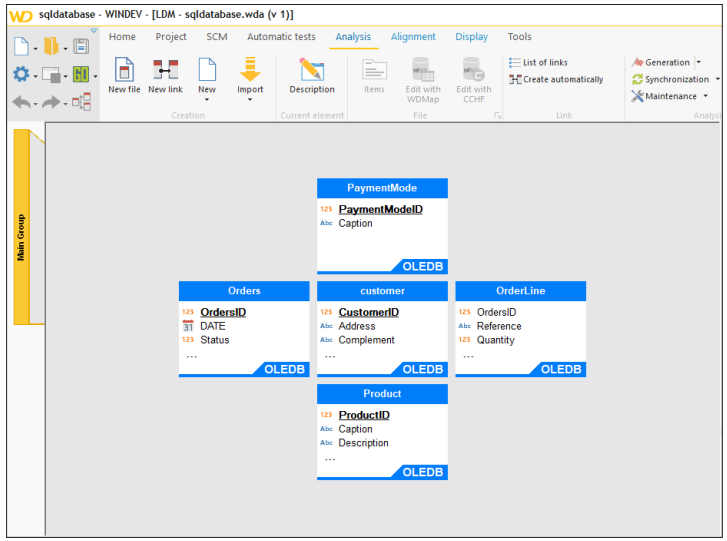
You can:
- access the data in its current format directly. Only the description of tables is imported into the analysis. No conversion is required.
- convert the data to HFSQL Classic format.

6. In our case, choose "Access the data in its current format directly". Go to the next step.
7. Select the tables (files) to import.



8. Validate and end the wizard. The tables are imported and viewed in the data model editor.

▸ The analysis is as follows:



Generating the analysis

Generating the analysis consists in validating the modifications performed in the analysis (creation of data files, addition or deletion of items, etc.) and to apply them to the entire project (pages, linked controls, reports, etc).

The generation is automatically proposed when closing the data model editor while modifications have been performed.

You also have the ability to generate the analysis manually. That's what we are going to do.

▸ To generate the analysis:

1. In the data model editor, go to the "Analysis" pane, "Analysis" group, and click "Generation".
2. The analysis generation is automatically started.

When your analysis is generated, the data can be handled like the HFSQL data. There is no difference. You can:
- Create a full application via RAD (Rapid Application Development). See "Lesson 3.4. The full RAD" for more details.
- Create a full custom application. See "Part 4" for more details.

Conclusion

The tutorial is over now!

This course covered a range of topics, but not all of WINDEV features, far from it!
You are now familiar with the main concepts.

Also explore the examples supplied with WINDEV: some are simple and only address one topic, while others are more complex. These examples will show you the different aspects of WINDEV. Reading the source code is also a good way to learn.

It would take too much time to discuss all available topics (there are hundreds, even thousands!). WINDEV proposes several other features not presented in this tutorial:

- HTTP and telephony functions
- creation of skin templates...
- nested reports, queries with parameters...
- dynamic compilation, calls to DLL, external languages...

See the online help for more details.

We wish you a great development experience with **WINDEV 25**!