



DEVELOP 10 TIMES FASTER 

TABLE OF CONTENTS

Introduction

Preliminary points	17
Overview of the tutorial	17
How do I access the online help?	18
Legend of symbols.....	20
If you are familiar with WinDev 17... ..	21
What is WinDev used for?	21

PART 1 - DISCOVERING WINDEV

Lesson 1.1. Discovering WinDev

Overview	26
Starting WinDev.....	26
Development environment.....	27

Lesson 1.2. My first window

Overview	31
Opening the project.....	31
My first window: entering and displaying data.....	32
Overview	32
Create the window.....	32
Input and display of the value entered	34
Improving the window	37

Lesson 1.3. My first windows

Overview	40
My first window: a counter	40
Overview	40
Implementation.....	40
Managing the counter.....	41
Creating a button to exit the window	44
Displaying the date and time in real time	46
Create the window.....	46
Implementing the timer	48
Managing the input of a numeric value to perform a calculation	49
Using a procedure	52
Creating a window with a menu	52
Describing the menu	52
Associating source code with the menu options	55
WinDev: Main concepts and terminology.....	57
Main concepts.....	57
Terminology	58

Remember to visit the download section of www.windev.com on a regular basis to check whether upgraded versions are available.

Free technical support address: freesupport@windev.com

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

All product names or other trademarks mentioned in this publication are registered trademarks of their respective owners.
© PC SOFT 2013: This publication may not be reproduced in part or in whole in any form without the express permission of PC SOFT.

Lesson 1.4. Programming concepts

Introduction.....	60
Declaring the different types of variables	60
The different types of variables	60
Declaring the variables and their scope.....	60
Main statements of WLanguage.....	61
Conditional statements	61
Loop statements	62
Comments	62
Practical example on the different main statements.....	63
Main WLanguage operators	63
Logical operators.....	64
Comparison operators	64
Indirection operators.....	64
Procedures and functions.....	65
Definition	65
Local procedure.....	66
Global procedure and set of procedures.....	66
How do I decide whether a procedure must be global or local?.....	66
About passing parameters	66
Calling a procedure	66
Creating a procedure.....	67
A practical example about the procedures and functions	67
Processing strings	68
Practical example	68
Details.....	68
Processing numeric values.....	69
Practical example	69
Details.....	70
Processing currencies.....	71
Practical example	71
Details.....	71
Processing dates and times	73
The dates.....	73
Practical example	73
Input mask and returned value	73
What is today's date?	75
What is today's date, with the day and the month in letters?.....	75
How many days between two dates?.....	75
What is the day of the week corresponding to a given date?	76
Practical exercise	76
Handling Date types variables.....	76
The times.....	77
Practical example	77
What time is it?	78
How much time has passed between two given times?	78
Calculations with dates and times.....	79

Lesson 1.5. Questions/Answers

Questions/Answers.....	81
How do I view the element to which the current process belongs?	81
How do I print the source code?	81
How do I find and/or replace a variable in the code?	81
What is the meaning of the "+" and "-" signs found in the code editor?	81
Can the person who wrote a piece of code be identified?	82
Is it possible to find out the line number of a code line?	82
How do I easily display the syntax or the help for a function?	82
What are the useful shortcuts in the code editor?	83
How do I communicate with the user?	83

PART 2 - APPLICATION WITH DATA

Lesson 2.1. Overview

Overview of the application created in this section	88
--	-----------

Lesson 2.2. Project and analysis

Overview	90
Creating the project.....	90
Creating the analysis.....	91
Creating the description of the data files	92
Creating a data file and its items in the editor	93
Importing a CSV file	99
Importing files from the dictionary	102
Direct import of existing data files	103
Creating the links.....	103
Configuring the analysis for RAD.....	105
Configuring the RAD in the description of the data files	106
Configuring the RAD in the description of items	107
Generate the analysis	107

Lesson 2.3. The full RAD

What is RAD?	110
Generating RAD.....	110
Testing the application	113
Quick modification: Locking the application	114

Lesson 2.4. Edit window

Overview	116
Creating an edit window	116
Managing the addition of a record.....	118
Creating the add button	118
Adding an input mask	119
Erasing the data after the addition	120
Viewing the records	121

Managing the search and the modification	122
Exact-match search.....	122
Generic search	124
Modifying the form displayed.....	125
Browsing the forms	125

Lesson 2.5. Table based on a query

Overview	128
Creating a query with parameters	128
Creating a window with a table based on the query	130
Printing the content of the table	132
Creating an automatic report on the table.....	133

Lesson 2.6. Report with embedded query

Overview	135
Creating a report.....	135
Testing a report	138

Lesson 2.7. User groupware

Overview	141
Including the user groupware.....	141
Configuring the user groupware	143

Lesson 2.8. The dashboard

Overview	146
Automatic tests.....	146
Static audit and query optimization.....	151

Lesson 2.9. Deploying the application

Overview	154
Creating the executable.....	154
Creating the setup program.....	157
Conclusion.....	159

Lesson 2.10. Questions/Answers

How do I create the main menu of my application?.....	161
How do I automatically insert the '?' menu?.....	161
How do I create a popup menu?.....	161
How do I automatically close a window after a preset duration?.....	162
How do I retrieve the parameters passed by command line to an executable?.....	162
How do I uninstall an application developed with WinDev?	162
How do I create a setup via CD-ROM?	163
How do I install an application?.....	163
How do I associate an icon with my executable?	163
How do I associate a splash screen with my executable?	163
How do I display the icon of my application in the taskbar?.....	164
How do I install a shortcut for an application on the desktop?.....	164
How do I share the WinDev framework among the applications installed on the same computer?.....	164
How do I detect the elements not used by my application?	165

PART 3 - WINDOWS AND CONTROLS

Lesson 3.1. Windows

How do I create a window?	170
Description of a window: 8 tabs are available	171

Lesson 3.2. Controls

Introduction.....	175
Practical example.....	175
The standard controls	176
Type of control: Static.....	176
Type of control: Edit control.....	178
Type of control: Button	183
Type of control: Image	187
Type of control: Radio button	193
Type of control: Check box	194
Type of control: List box.....	196
Type of control: ListView.....	201
Type of control: Combo box.....	204
Type of control: Table.....	206
Type of control: Looper	211
Type of control: TreeView.....	213
Type of control: TreeView Table.....	214
Type of control: Progress bar	214
Type of control: Chart	215
The specialized controls.....	219
Type of control: Calendar	220
Type of control: Organizer	220
Type of control: Scheduler	220
Type of control: Organization Chart	221
Type of control: TreeMap.....	221
Type of control: Carousel.....	222
Type of control: Cube.....	222
Type of control: HideShow.....	222
Type of control: Shape.....	222
Type of control: Bar Code	223
Type of control: Scrollbar.....	223
Type of control: Range Slider	224
Type of control: Slider.....	224
Type of control: Spin	224
Type of control: Rating.....	225
Type of control: Web camera.....	225
Type of control: HTML.....	226
Type of control: OLE, ActiveX.....	226
Type of control: Sidebar	228
Type of control: Toolbar	228
Type of control: Ribbon.....	229
Type of control: Pivot table	229

Type of control: Multiline zone	230
Type of control: TimeLine	230
Type of control: Xaml	230

Lesson 3.3. Ergonomics of the application

Improving the ergonomics of the windows and controls	232
Style/Skin template	232
Resizing the windows	233
Anchoring.....	233
Tabulation/Tab order	234
Rulers/Interface checker/Alignment.....	235
Plane/Tab	236
Splitter	237
Animation of windows	238
Dim disabled windows	238
Totals in the tables	238
Images of controls configurable in the style	239
Transparency of controls, transparency of windows	239
Saving the position of the columns in the tables.....	239
Ergonomics of the application.....	240
Example of program used to manage the multi-windowing of free windows (non MDI)	240
Example of program used to manage the MDI windows	243

Lesson 3.4. Reusability

What does the reusability mean?	249
Re-using a set of controls.....	249
The supercontrol	249
Internal window	250
Control template.....	252

Lesson 3.5. Questions/Answers

How do I duplicate a control found in a window by programing?	254
How do I delete a control found in a window by programing?	254
How do I manage the planes of a window?	254
How do I manage the click on a tab pane?	255
How do I display the progress of a process?	255
How do I display an Internet site in a window?.....	255
How do I modify the color of a control?	256
How do I make a button invisible?.....	256
How do I create the main menu of my application?.....	257
How do I create a popup menu?	257
How do I modify the rollover cursor for a control or a window?	257
How do I pass parameters to a window?	258
How do I group controls in order to modify their properties by programming?	258
How do I transform a check box into a radio button?	259
How do I give the same size to the buttons?	259
How do I manage the persistence of controls?.....	259
How do I add a background image to a window?.....	259
How do I manage the transparency of a window?	259

PART 4 - DATABASE AND ANALYSIS

Lesson 4.1. Introduction

Overview	264
The different modes for accessing databases.....	264
Native Access	264
Direct ODBC access	265
OLE DB access	265
ODBC access via OLE DB	265

Lesson 4.2. Browsing the data files and queries

HReadFirst browse	267
HReadSeek browse	267
FOR EACH browse.....	268
FOR EACH WITH browse	268
What type of browsing command should I choose?.....	269

Lesson 4.3. Error management

Overview	271
Managing the errors in automatic mode.....	271
Managing the errors in advanced programmed mode	273

Lesson 4.4. Data encryption

What is data encryption used for?	276
How do I encrypt the data files?	277
Implementing the encryption of the data files	277
Managing the encryption in WLanguage	279

Lesson 4.5. Advanced concepts

Overview	281
The log process	281
Transactions.....	281
A simple example of the need for transactions in some types of processes.....	282
The transactions on HyperFileSQL	282
The replication	283
Triggers	283

Lesson 4.6. External databases

Overview	285
Connecting to a database in the data model editor	285
Connecting to a database by programming.....	288
Creating the connection: HOpenConnection	288
Associating the connection with the data files: HChangeConnection	289

PART 5 - MANAGING A HYPERFILESQL CLIENT/SERVER DATABASE**Lesson 5.1. Introduction**

Overview	294
Why switch an application to HyperFileSQL Client/Server?	295

Lesson 5.2. Implementing a Client/Server application

Overview	297
Installing a local HyperFileSQL server	297
Creating an application in HyperFileSQL Client/Server mode	297
Adapting a HyperFileSQL Classic application to support the Client/Server mode	298
Overview	298
Migrating the example	298
Features available in HyperFileSQL Client/Server mode	300

Lesson 5.3. Managing a Client/Server database

Overview	302
Configuring the computers	302
The HyperFileSQL Control Center	302
Creating a user account in the HyperFileSQL Control Center	304
Saving the database	307
Conclusion	307

Lesson 5.4. Setup on the user computers

Overview	309
Starting the wizard for setup creation	309

PART 6 - QUERIES**Lesson 6.1. Creating a Select query**

Introduction	314
Your first query	315

Lesson 6.2. Queries with count

Creating a query with count	321
-----------------------------------	-----

Lesson 6.3. Sum queries

Creating a "Sum" query	324
------------------------------	-----

Lesson 6.4. The SQL language

Source code of a query	327
The SQL commands	328
The SELECT statement	328
The FROM statement	328

The WHERE statement	328
The GROUP BY statement	329
The ORDER BY statement	329

Lesson 6.5. Using queries

Using a query from a report	331
Using a query from a control	331
Using a query from WLanguage	331

PART 7 - REPORTS**Lesson 7.1. Your first report**

Basic vocabulary	336
Overview	338
Printing with the report editor or printing in WLanguage	338
The reports to create	339
Creating your first report	339
Creating the report: List of customers sorted by city	339
What is a break?	342

Lesson 7.2. Statistical reports

Overview	348
Step by step	348

Lesson 7.3. Crosstab reports

Overview	358
Step by step	358

Lesson 7.4. Special controls

Overview	363
Operations for creating a Link control	363
Operations for creating a clickable control	364

Lesson 7.5. Running reports

Running reports in WLanguage	367
iPreview and iDestination	367
iPrintReport function	368

Lesson 7.6. Distributing "Reports & Queries" with your applications

Starting "Reports & Queries"	370
Configuring the project	371
Configuring the analysis	371
Configuring the reports	372
Configuring the queries	372
Creating the executable and distributing the application	372
Installing the application	374
Testing the application	375

PART 8 - ADVANCED PROGRAMMING**Lesson 8.1. Automatic management of errors**

Overview	380
Implementation.....	380
Implementation.....	380
Types of errors	381
Using the automatic error management.....	381

Lesson 8.2. Import/Export

Overview	383
Practical example	383
Handling text files, CSV files,	383
Practical example	384
Handling XML files.....	385
Practical example.....	385
Handling XLS files.....	386
Practical example	386

Lesson 8.3. Dynamic compilation

Overview	388
Example.....	388

Lesson 8.4. Windows event

Introduction.....	391
Example: Detecting the click performed on a list	392

Lesson 8.5. The threads

Definition.....	395
Example.....	395

Lesson 8.6. Sockets

Overview	397
Practical example.....	397
Server Application (simplified).....	398
Creating the socket	398
Exchanging data	398
Closing the socket	398
Practical example.....	398
Client Application	399
Connecting to the server.....	399
Exchanging data	399
Ending the communication.....	399
Practical example.....	400

Lesson 8.7. FTP

Overview	402
Connect to an FTP server	402
Sending a file	402
Retrieving a file	403
Disconnecting from an FTP server.....	403

Lesson 8.8. Emails

Overview	405
POP3/SMTP protocol: Principle	405
Practical example.....	406
Sending an email: the steps	406
Opening an SMTP messaging program.....	406
Preparing the email.....	407
Send the email	407
Disconnect.....	407
Other possibilities.....	408

Lesson 8.9. OOP

Concepts.....	410
The classes.....	410
The objects	410
The members	410
The methods.....	410
Concept of inheritance.....	410
Constructor and Destructor	411
Data encapsulation.....	411
Example	411
Creating an object-oriented program	411
Declaring a class	412
Describing the methods.....	412
Declaring and handling objects.....	413

PART 9 - ADVANCED PROJECT MANAGEMENT**Lesson 9.1. Dashboard**

Overview	418
Example	418
The different elements of the dashboard.....	418
The lists.....	419
The lights	420
The counters in relation with the Control Centers	422
Configuring the dashboard	423

Lesson 9.2. SCM

Introduction	425
SCM (Source Code Manager)	425
Principle of SCM	425
Creating the SCM database.....	426
Including a project in the SCM.....	427
Opening a project from the SCM.....	429
Configuring the SCM.....	430
Checking out an element.....	431
Modifying the checked-out element.....	432
Checking in the checked-out element.....	432
Synchronizing the project	433
Off-line mode (or mobile mode)	434
SCM administrator	434

Lesson 9.3. Control Centers

Overview	436
The Project Monitoring Center	436
Managing your time	439
Monitoring the user feedback with the Control Centers	440
The other Control Centers	440

Lesson 9.4. Business rules

Overview	442
Example	442
Application on a real example	442
Creating a business rule	442

Lesson 9.5. Multi-configuration

Overview	446
Creating a project configuration	446
Managing the project configurations	447
Multiple generation	448

Lesson 9.6. External components

Overview	450
Teamwork	450
Large projects.....	450
The databases accessed by several projects	451
The processes used in several projects	451
The ability to distribute a feature or set of features.....	451
Multi-product external component	451
Step by step	452
Step 1: Creating an external component	452
Step 2: Using the external component	455
Distributing an external component	457
Standard distribution	457
Professional distribution.....	457

Lesson 9.7. The debugger

Overview	459
Debugging a window	459
Starting the debugger	459
Viewing the information in the debugger	460
Trace window of the debugger	461
Debugging a project	461
Advanced options of the debugger	462

Lesson 9.8. The performance profiler

Overview	464
Using the performance profiler	464
Optimizing a process with the performance profiler	465
Reading the result of the performance profiler.....	465
Choosing a process to optimize.....	465
Optimizing a process.....	465

Lesson 9.9. Import/Export

Importing elements	468
Exporting elements	469
Specific import operations	470
Importing a WebDev project	470
Importing a window	470
Access import.....	471
Visual Basic import	471

Lesson 9.10. Installing an application

Overview	473
Network update/Web update	473
Setup with network update	473
Setup with Internet update	475
Setup editor	477
Questions/Answers	478
How do I uninstall an application developed with WinDev?	478
How do I create a setup via CD-ROM?	478
How do I install an application?.....	478

Lesson 9.11. UML and 3-tier

What is the UML?	480
The diagrams managed by WinDev	480
Class diagram.....	480
Use case diagram.....	480
Object diagram	480
Component diagram	480
Activity diagram	480
Sequence diagram	480
Collaboration diagram	480
State-transition diagram.....	481
Deployment diagram.....	481

3-tier	481
What is the 3-tier?	481
How do I implement 3-tier?	481
Handling the different layers	482

Lesson 9.12. Flexible modeling

Principle	484
Operations.....	484

Lesson 9.13. Multilingual

What is a multilingual application?	488
Choosing the project languages.....	488
Localizing the analysis	489
Localizing the project elements	491
Characteristics to translate	491
How do I translate this information?	493
Direct input of the translations	493
Special cases.....	494
Translation with WDMMSG and WDTRAD	497
Programming the change of language.....	497
Using a parameter file	497
Using menu options	498
Managing the Unicode and the specific character sets	499

Conclusion

INTRODUCTION

Preliminary points

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

The aim of the tutorial is to help you discover WinDev, become familiar with the editors and teach you the WinDev concepts.

This manual does not cover all the features of WinDev.

You should plan on spending a few hours to follow this course and to learn WinDev: you'll find it was well worth it!

If you try to develop an application before practicing, you will lose time, and a lot more than a couple of days.

This course was designed so you can approach it in two different ways:

- either you follow all the detailed exercises in each lesson (recommended method).
- or, if you are in a hurry and already experienced, you can read through it without doing the exercises, as all the exercises have screen shots. However, in order to quickly assimilate the main concepts, we recommend that you follow the course step by step.

WinDev evolves all the time, so the screen shots found in this guide may differ from the ones found in your product.

Overview of the tutorial

The tutorial has been designed to progressively teach you how to use WinDev. By following this course:

- 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 (directly accessible from the editors or from the guide).

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

Don't forget to also take a look at the dozens of examples supplied with WinDev: they are very instructive!



Notes

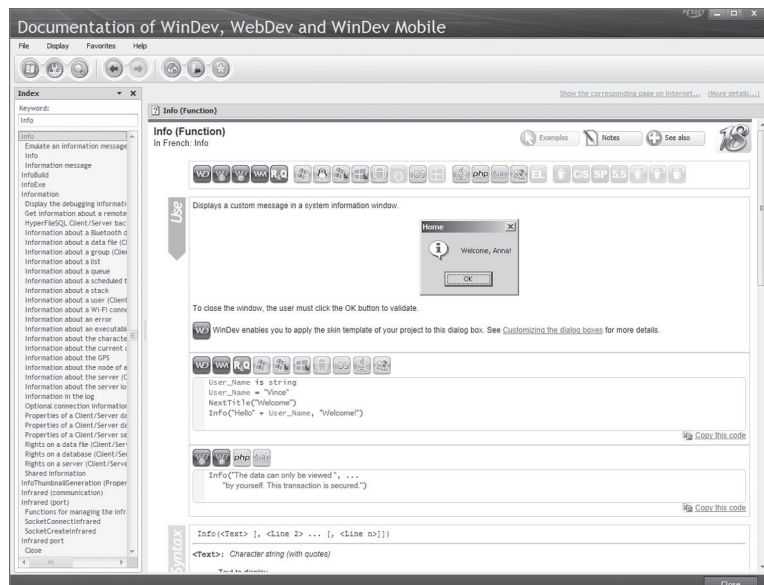
The Tutorial may have evolved since this document was published. Don't hesitate to consult the electronic version of the tutorial (PDF file accessible from the WinDev menu: on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Tutorial (PDF)").

How do I access the online help?

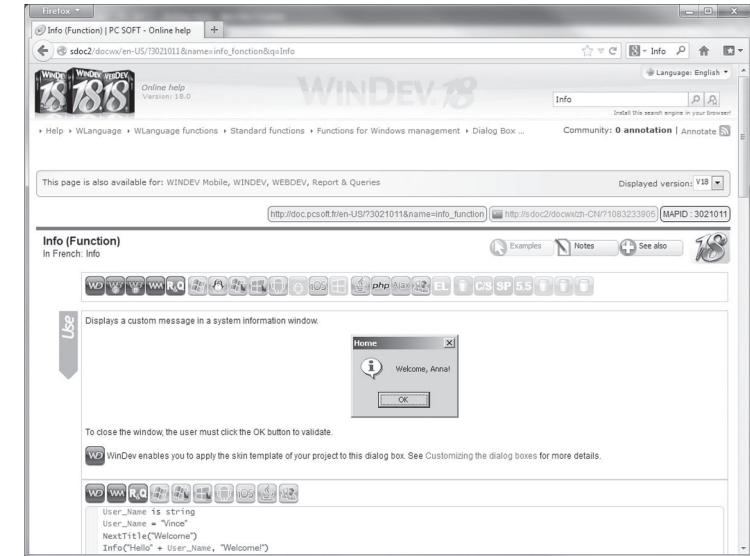
The online help of WinDev enables you to get detailed information about the 2500 WLanguage functions. The online help also contains the help about the editors, controls, tips, ...

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 proposes a button allowing you to access the corresponding help page.
- The help menu of the editors ("Help" option available on the "Home" pane, in the "Online help" group of the WinDev menu) allows you to start the online help.
- ▶ The help can be displayed:
 - in a specific "help browser":



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



Notes

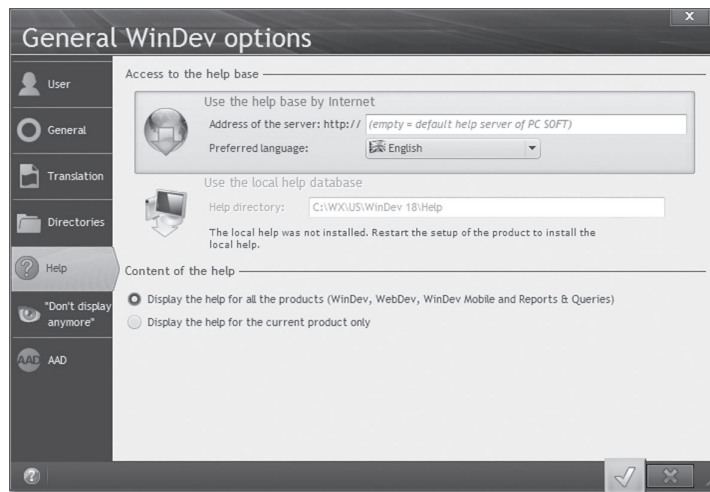
The online help of WinDev, WebDev and WinDev Mobile is available on Internet from any computer equipped with an Internet access, without the product being necessarily installed. This help is updated on a regular basis.

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

To start the Internet online 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 database.



- the content of the help: help common to WinDev, WebDev and WinDev Mobile or help for the product currently used.

Legend of symbols



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 "Wizards, Examples and Components" pane of WinDev.



This symbol introduces a "Tip", we advise you to read the associated text.



This symbol introduces a "Warning", reading the associated text is extremely important.



This symbol introduces a "Note", we advise you to read the associated text.



This symbol gives the result of a "Test", we advise you to read the associated text.

If you are familiar with WinDev 17...

If you are familiar with WinDev 17, following this course will do no harm: it's a good opportunity to "revise" the features of WinDev!

What is WinDev used for?

WinDev is an IDE (Integrated Development Environment). It enables 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 for developing an application.

Unlike some other programming languages, you don't need to find and add modules to be able to design, test and install an application.

The 5GL (5th Generation Language) of WinDev, the 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!

PART 1

Discovering
WinDev

18



DEVELOP 10 TIMES FASTER PC SOFT

LESSON 1.1. DISCOVERING WINDEV

This lesson will teach you the following concepts...

- Starting WinDev



Estimated time: 5 min

Overview

WinDev is an IDE (Integrated Development Environment) allowing you to develop Windows applications in several fields: management, industry, health care, ... The applications can provide access to information stored in databases.

In this tutorial, you will learn how to create your applications (with or without database) and how to improve them by using the features proposed by WinDev.

Starting WinDev

- ▶ Start WinDev 18 (if not already done).
- ▶ A wizard starts if WinDev 18 was never started before. This wizard is used to:
 - If you own an earlier version of WinDev, retrieve the existing configurations.
 - If you are a new user, configure your environment. This way you can choose the screen configuration used and configure the Control centers (we'll see the Control Centers in more details in "Control Centers", page 435).
- ▶ If you've already run WinDev 18, identify yourself if needed. The development environment starts. The home window is opened.

Let's take a closer look at the development environment of WinDev.

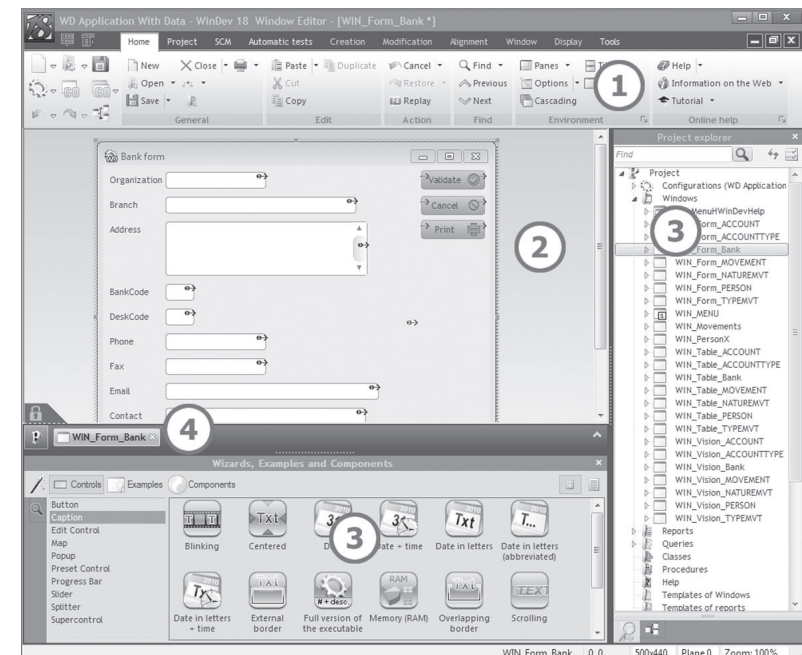
Development environment

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, ...

All the editors use the same environment:



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

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

3. Panes. The WinDev interface includes several panes allowing you to quickly access different types of information. Some examples:

- The "Project explorer" pane (displayed on the right) is used to list all the project elements by category.
- The "Wizards, Examples and Components" pane (at the bottom) is used to quickly access the full examples, sample pages, controls that can easily be included in your applications.

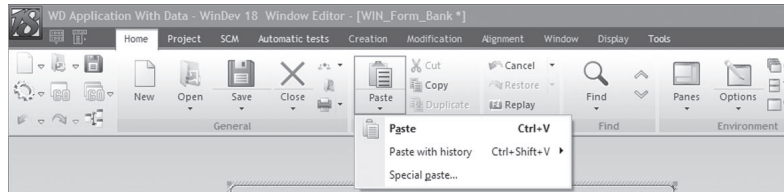
If necessary, these panes can be hidden by pressing [CTRL] + [W].

4. Bar of opened documents. This bar is used to quickly view all the 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 options of the editors are grouped.

We are going to take a closer look at the main elements of the ribbon, as well as how we will interact with it in this tutorial.



The different ribbon elements

The ribbon includes three areas:

- the button area, on the left.
- the pane area, at the top
- the option area.

Let's take a closer look at these areas.

The button area



The button area groups the **quick access buttons**. These buttons are used to perform the most usual operations, common to all the editors: save, open, create, ...

The 3 buttons at the top of this area are also specific:

- The 18 logo is used to display the "About" window and the custom-menus.
- The 2 other icons are used to restore the toolbars and the menus found in the earlier versions.

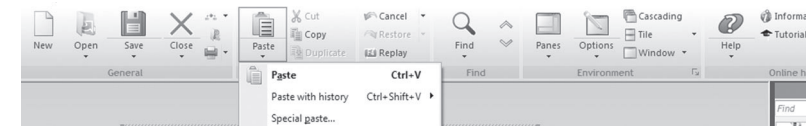
The pane area



The ribbon panes are used to access the options of the different editors. Several types of panes are available:


- the current pane: The pane tab appears in light gray and an orange line is displayed at the top of the tab.
- the popup panes, specific to the current element: The name of the pane is displayed in orange.
- the available panes: The name of the pane appears in white.

The option area



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 name and it can have a group button . This button is used to perform a specific action according to the current group: displaying the description of the current element, displaying the help, ...

In this tutorial, to identify a menu option, we shall be talking about panes, groups and options.

For example:

To display the help, on the "Home" pane, in the "Online help" group, click "Help".

LESSON 1.2. MY FIRST WINDOW

This lesson will teach you the following concepts...

- How do I create a window?
- How to enter text and display it?



Estimated time: 30 min

Overview

To start working with WinDev, we are going to create some windows.



Note

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

These examples will enable you to get familiar with the programming concepts used in WinDev.

The lessons found in this first part will allow you to:

- create simple windows,
- handle strings, numeric values, currencies,
- handle dates and times.

Opening the project

- ▶ Start WinDev 18 (if not already done). Close (if necessary) the current project to display the home window.
- ▶ Open the "WD My First Windows" project.
To do so, in the home window, click "Tutorial" and select the first project "My first windows (Exercise)".
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "My first windows (Exercise)".



Important

In this part, we will focus on creating simple windows. The project named "My First Windows" is an empty project that has already been created. The creation of a project will be presented in another lesson.



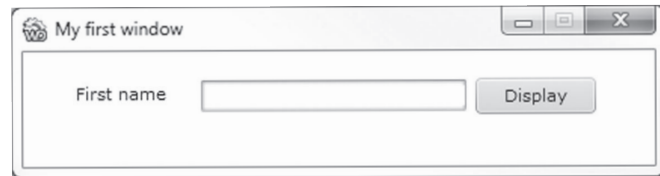
Answers

A corrected project is available. This project contains the different windows created in this lesson. To open the corrected project, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "My first windows (Answer)".

My first window: entering and displaying data

Overview

Here's the window you're going to create:




This window allows the user to enter his or her first name and display a welcome message back with the "Display" button.

You may think this is too basic, too simple, but we recommend that you 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 some principles that are fundamental for the rest of this tutorial.

Create the window

► To create the window:

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



2. A window shaped like a wheel is displayed. This window is used to create all the elements that can be associated with a project.
3. Click "Window". The wizard for window creation starts.
4. Select "Blank" in the list of "standard" windows displayed on the left. In the list of skin templates found on the right, the "Elegant" skin template is selected by default. You can choose another skin template proposed in the list.



Note

The skin templates allow you to quickly create outstanding interfaces. A skin template defines the style of the window but also the style of all the controls that will be used in this window. No ugly interface anymore.

5. Validate. The window is automatically created in the editor.

- We're going to define the window's main characteristics: its name, the title that will display in the title bar.
Double-click the window you've just created. The description window opens on the "General" tab. The window name offered by default is "WIN_NoName1".



Note

Study the name of the window proposed by WinDev: this name starts with the letters "WIN_". This prefix is automatically added because the project uses a programming charter.

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


- a window starts with WIN,
- a button starts with BTN,
- etc.

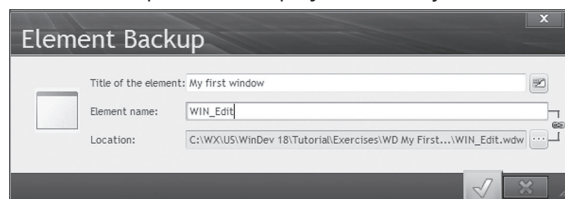
If you don't want to use this charter, all you have to do is disable it: on the "Project" pane, in the "Other actions" group, expand "Charter" and uncheck "Use the charter".

- Enter the following information:



1. The window's "logical" name: replace "WIN_NoName1" with "WIN_Edit".
This name will correspond to the name used to save the window on disk (with the "WDW" extension) in the directory of the project sources. This name will also be used in programming to handle the window.
 2. The window's title: enter "My first window". This title is displayed in the title bar of the window.
- Validate (green button). This title is displayed in the window's title bar.

- Save the window by clicking  among the quick access buttons. By default:
 - the element's title corresponds to the title we entered.
 - the name of the element is the logical name of the window.
 - the proposed location corresponds to the project directory.

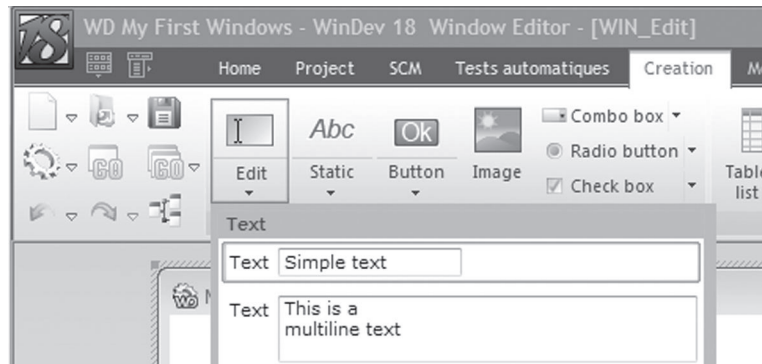


- Click the green button to validate.

Input and display of the value entered

To manage the input and display of the value entered, you're going to create:

- a control where the user will enter his or her name. Therefore, this type of control is an "edit control".
 - a button used to display the name entered.
- To create the edit control:
1. On the "Creation" pane, in the "Usual controls" group, expand "Edit" (click the arrow):



The list of edit controls proposed by default is displayed.

2. Click "Simple text" in the control list. The control currently created follows the movement of the mouse.
3. Move the mouse in the window toward the position where the control must be created (at the top of the window for example). To drop the control in the window, all you have to do is perform a new left mouse click.
4. Perform a right mouse click on the control that was just created. The control's popup menu comes up. Select "Description" from the popup menu. The edit control's description window comes up.

5. Modify the characteristics of the control by entering the following information:




- The name of this control is: "EDT_FirstName".
- The caption of this control is: "First name"
- This control is a text control.

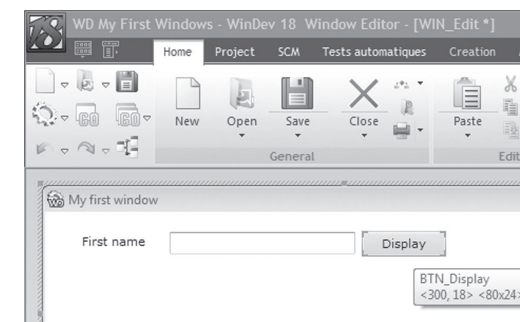
6. Validate the description window of the control (green button). The control is displayed in the window editor.

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

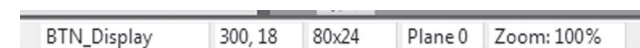
1. Select the control (simply click it with the mouse).
2. Select "Resize" from the popup menu (right mouse click).
3. The control is instantly enlarged.

- To create the "Display" button:

1. On the "Creation" pane, in the "Usual controls" group, click .
2. Position the control inside the window. Click where you want to create the button (for example to the right of the edit control).
3. Click the button you've just created. The text displayed in the button becomes editable. Enter the caption: "Display". The name of the button automatically becomes "BTN_Display". You can see the button's name:
 - in a tooltip, displayed when the control is hovered by the mouse cursor:



- in the status bar of the editor:



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



Notes

WLanguage is the programming language that comes with WinDev. It's a 5th generation language (5GL) that uses highly sophisticated commands.

1. Select the "Display" button.
2. Display the popup menu of the control (right mouse click).
3. Select the "Code". This option opens the code editor of WinDev, in which all the WLanguage statements can be entered.
4. Enter the following code in the "Click BTN_Display" process:

```
Info("Hello " + EDT_FirstName)
```

Note on the assisted input: As soon as the first three characters are typed, WinDev proposes all the words in the WLanguage vocabulary containing these characters. The aided development is very a powerful feature. No more mistake when typing the name of an element: the syntax errors are reduced to a minimum. All you have to do is select the requested word and press [Enter] to validate. You can focus on the algorithm.



Notes

This is the reason why the programming charter is so important. All the elements handled in the code of the application use the same standard so they can be easily found when entering the source code.



Notes

When entering this code in the code editor, you have noticed that different colors are used by the different elements. This is the syntactic coloring. The code editor enables you to easily identify the different elements handled by the code:

- 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. Our message is built from the text "Hello " and the value of the "EDT_FirstName" control. The "+" sign indicates a "concatenation" operation between two strings.



Notes

In this example, the text entered is displayed in a system window, but you can also display it in a window created using WinDev.

- Let's save the window and then test it:
1. Save the modifications by clicking among the quick access buttons (on the left of ribbon), or by pressing [Ctrl]+[S].
 2. Click among the quick access buttons (or press [F9]).
 3. The created window is started in execution. Enter your first name.
 4. Click the "Display" button.
 5. Validate the system window that is displayed.

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 the window, report or procedure while you are creating it. This is both simple and fast!

- Click the "x" button found in the title bar to close the window.
- The WinDev editor is redisplayed. Close the code editor (on the "Home" pane, in the "General" group, click "Close").

Improving the window

During this first test, you have noticed that:

- the window is too large,
- your window can be resized since it only contains two controls
- the first name is displayed using the same case as when entered. We're going to force the use of an uppercase letter at the beginning of the name.

We are going to improve the appearance of this window.

- To reduce the size of the window in the editor:
1. Click the window: blue handles are displayed around the window (depending on the skin template used, the color of these handles may be different).
 2. Click the handle found in the bottom right corner and reduce the size of the window by keeping the 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 "GUI" tab, clear the "Resizable" option (simply click the option)
 3. Validate.



Notes

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

- Run the test of this window to view the result at run time.
- Now we're going to force the use of an uppercase letter when entering and displaying the first name.

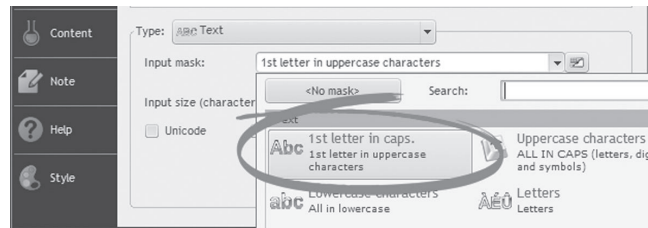


Notes

By default, the characters will be entered in uppercase or lowercase according to the setting of the [CAPSLOCK] key on the keyboard. This type of input may be very annoying 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 entered, without any programming.

- To modify the input mask:
 1. Double-click the "FirstName" control. The description window is displayed.
 2. In the "General" tab, expand the "Input mask" list and choose "1st letter uppercase".



3. Validate the description window of the control.
 4. In the editor, the name of the control automatically comes up in the control.
- Run the test of this window to view the result at run time.



You've just quickly and successfully created your first window. In the rest of this lesson we're going to discover new WinDev concepts.

- Validate the message, and close the test window (click the X in the top right corner of the window). The WinDev editor is redisplayed.
- Close the "WIN_Edit" window displayed in the editor (on the "Home" pane, in the "General" group, click "Close").
- Close the project (on the "Home" pane, in the "General" group, expand "Close" and select "Close the project").

LESSON 1.3. MY FIRST WINDOWS

This lesson will teach you the following concepts...

- How do I create a counter window
- How to create a window displaying the date and time in real time
- How do I perform a calculation
- Creating a window with menus



Estimated time: 30 min

Overview

In the previous lesson, we've already seen how to enter and display text.

In this lesson, we're going to create the following windows:

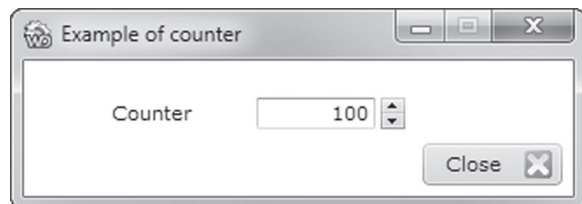
- A counter window,
- A window displaying the date and time in real time,
- A calculation window,
- A menu to group the windows that were previously created.

These windows will be grouped in a project for practical reasons.

My first window: a counter

Overview

We are going to create the following window:



The numeric counter will be incremented and decremented with the arrow buttons.

Implementation


- Open the "WD My First Windows" project.
To do so, in the home window, click "Tutorial" and select the first project "My first windows (Exercise)".
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "My first windows (Exercise)".



Answers

A corrected project is available. This project contains the different windows created in this lesson. To open the corrected project, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "My first windows (Answer)".

- To create the window:

1. Click  among the quick access buttons of the WinDev menu.
2. In the window shaped as a wheel that comes up, click the "Window" option. The wizard for window creation starts.
3. Select "Blank" in the list of "standard" windows displayed on the left. In the list of skin templates found on the right, the "Elegant" template is offered automatically.
4. Validate. The window is automatically created in the editor.
5. Right-click the window that was just created and select "Description" from the popup menu. The description window is displayed.
6. In the "General" tab, specify the name of the window you've just created ("WIN_Counter") as well as its title ("Counter example"). Validate the description window.
7. Save the window (CTRL + S).



Notes

It is very important to save the window as soon as it is created.

Saving the window allows WinDev to:

- automatically propose the name of the controls in the code.
- propose the automatic renaming of controls in the code.

Managing the counter

To manage the numeric counter, you are going to create:

- a control where the numeric value will be displayed,
- a spin box control (made of two buttons) that will be used to add 1 to the counter or subtract 1 from it.

WinDev allows you to create these different controls in a single operation via the preset controls proposed during the creation.

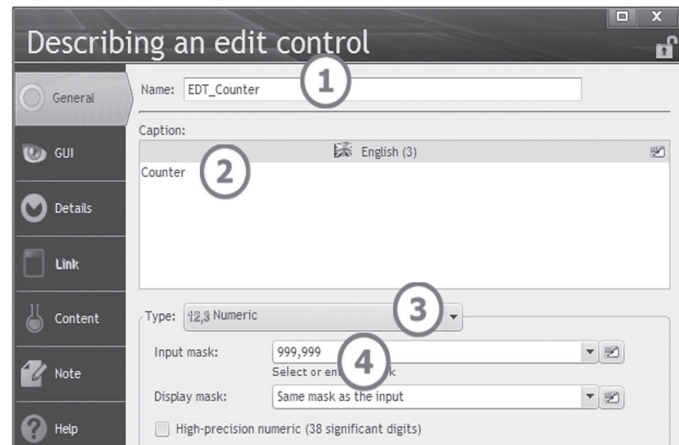
- To create the different controls of the counter:

1. On the "Creation" pane, in the "Usual controls" group, expand "Edit" (click the arrow). The list of preset edit controls is displayed. The "Integer + Spin" numeric control corresponds to our requirements.
2. Click the "Integer + Spin" control.
3. In the window, click the location where the control must be created.
4. Save the window.

Now, we are going to modify the characteristics of the created controls.

- Click in the window. The control we just create is no longer selected.
- Double click the edit control to display the control's description window. This window contains the information generated by default.

We are going to modify this information:



1. Enter the name: "EDT_Counter".
2. Enter the caption: "Counter".
3. This control is a numeric control.
4. Modify the format ("Input mask" option): expand the list of default formats and select "999,999" (you can also directly type the value "999,999" in the "Input mask" field). The "999,999" mask means that:
 - the number can contain up to 6 digits.
 - the number contains no decimal part.
 - the thousand separator is a comma.



Notes

By default, the input mask of the control corresponds to the numeric mask defined for the project. This information is selected in the "Languages" tab of the project description. To display the project description, on the "Project" pane, in the "Project" group, click "Description".

Therefore, the same mask is automatically used in all the numeric controls of the application. This feature is very useful in multilingual applications.



Notes

Two types of masks are available:

- **The display mask:** it corresponds to the format used when the value is displayed in the control.
- **The input mask:** it corresponds to the format used when the user enters a value in the control.

We want to give an initial value to the counter. To do this, initialize the content of the control with this value ("Content" tab of the control description window).



Notes

Many developers would perform this operation by programming: a code line must be written to assign a value to this control.
No need to write code lines in WinDev. This type of information is part of the control characteristics.

- ▶ Click the "Content" tab and enter the initial value ("100" for example).
You also have the ability to use the traditional method by entering the following WLanguage code line in the initialization code of the control:

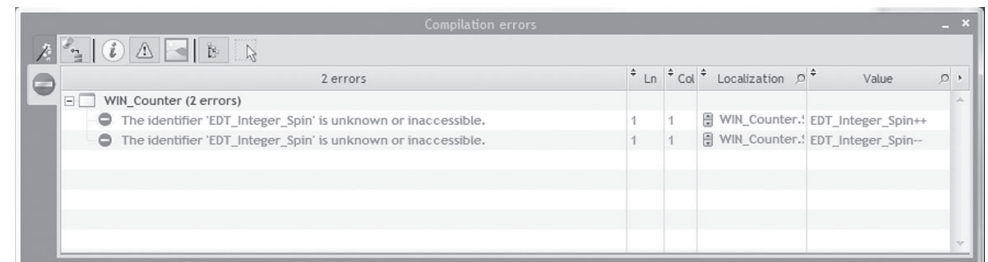
```
EDT_Counter = 100
```



Notes

In this code line, the "=" sign is used to assign the specified value to the "EDT_Counter" control. The following syntax is used:
<Destination> = <Value to assign>

- ▶ Validate the information about the control (green button of the description window).
In the editor, the edit control contains the value 100. This concept is called "Live Data": you see the data in real time! You will come across this concept later when handling data files.
- ▶ Save the window. A "Warning" and several compilation errors automatically come up in the "Compilation error" pane.



These errors indicate that the "EDT_Integer_Spin" is unknown. This name is used in one of the processes of the "WIN_Counter" window. WinDev has detected that a control was renamed (in our case, the name of the edit control was modified) and a compilation error is displayed for each use of the control.

Display the error's popup menu (right click the error). This menu offers to automatically rename this control wherever it is used in the window (notably in the WLanguage processes). Select "Rename all the references" from the error's popup menu.

- ▶ The "Spin" control (the two small arrows) has been automatically created next to the edit control. This control displays two small arrows allowing you to modify the value found in the edit control. We are going to modify it to manage the value of the counter.

- Display the code of the Spin control:
 - Display the popup menu of the control (right mouse click).
 - Select the "Code" option from the popup menu.
- This control contains the initialization code as well as two specific pieces of code:
 - The increment code: this code is run when a user handles the spin box control to increase a value.
 - The decrement code: this code is run when the user handles the spin box control to reduce a value.

- The following code is displayed in the increment code:

```
EDT_Counter++
```

This code line means that 1 is added to the value of the EDT_Counter control.
The EDT_COUNTER control can then be handled as a variable.





Notes You also have the ability to use the following syntax:

```
EDT_Counter += 1
EDT_Counter = EDT_Counter + 1
```

The principle is the same for the decrement code:

```
EDT_Counter--
```

This code means that 1 is subtracted from the value of the EDT_Counter control.

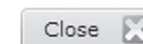
- Save the window and test it (to do so, click  then  among the quick access buttons).
- Click the "x" button found in the title bar to close the window. The WinDev editor is redisplayed.
- You can now reduce the size of the window using the handles, and make it non-resizable (we saw how to do this in the previous lesson).

Creating a button to exit the window

A window always contains a button or an option used to exit from it. If no button or option is added to the window, it still can be closed by clicking the "X" icon found in the title bar, even if this method is not very user-friendly.

- To create the closing button:
 1. On the "Creation" pane, in the "Usual controls" group, expand "Button" (click the arrow). The list of preset buttons is displayed.
 2. Click the "Close" button.
 3. In the window, click the position where the button must be created.

- Let's check the description of the created button:
 1. Display the control description (select "Description" from the popup menu or double-click the button).
 2. In the "General" tab:
 - The name of the button complies with the programming charter.
 - An action is associated with the button: this action is used to close the window.
 - The images corresponding to the skin template used are automatically associated with the button.



- 3. In the "GUI" tab, the button is a "Cancel" button.



Notes

WinDev proposes several types of buttons. They will be presented later in the chapter regarding ergonomics.
A "Cancel" button must be used to close a window.

- 4. Close the description window of the button.

- The close button can be easily moved:
 1. Click the close button with the left mouse button.
 2. Keep the mouse button down and move the mouse (and the control) to the requested location.
- Run the test of this window to view the result at run time.

Our counter is now entirely created!


Displaying the date and time in real time

In a new window, you will now display the date and time. Simple! However, the time will be updated in real time. To do so, WinDev offers "Timers".

A timer is used to run in parallel a task that must be periodically started (get data in background task, refresh data automatically, trigger a program at a given time, ...). Only the procedures (global or local to the window) and the global class methods can be called by a timer.

But let's create our window first!

Create the window

- ▶ Create a new blank window:
 1. Click  among the quick access buttons. Click "Window" in the wheel that is displayed. The wizard for window creation starts.
 2. Select "Blank" and the "Elegant" skin template.
 3. Validate. The window is automatically created in the editor.
 4. Right-click the window and select "Description".
 5. Name the window "WIN_Timer" and give it the "Refreshing the time by timer" title. Validate the description window.
 6. Save the window.

Now let's add a date type control in our window to display today's date.



- ▶ To create the date type edit control:
 1. On the "Creation" pane, in the "Usual controls" group, expand "Edit" (click the arrow). The list of available edit controls is displayed.
 2. Select the "Date" control (without the small calendar) and position the control inside the window.
 3. Right-click the control and select "Description".
 4. In the "General" tab of the description window:
 - Enter the control's name: "EDT_Day".
 - Enter the caption: "Today's date is: ".
 - Select the display mask "DDDD DD MMMM YYYY". This display mask allows you to display the date spelled out, for instance "Wednesday 26 September 2012".
 5. This control is used only to display the date. No input should be accepted. To do so, simply specify that this control is in "Read only" mode in the "GUI" tab of the control description window.
 6. Validate.

To display today's date in the date type control, simply initialize the control with this date. This operation is performed via programming.

- ▶ Display the code associated with the control you've just created:
 1. Select the edit control: simply click it with the mouse.
 2. Display the popup menu (right mouse click) and select "Code". The code editor appears.
 3. In the "Initialization of EDT_Day" process, enter the following code:

```
Myself = Today()
```

Let's take a look at this initialization code:

- **Myself** is a keyword which stands for the name of the current control. **MySelf** enables you to write local code (processing a control, etc.) or global code (global procedure, class, etc.) that remains independent from the current control.
- **Today** is a WLanguage function used (as its name implies) to get today's date.
- ▶ Save the window and test it (to do so, click  then  among the quick access buttons). The edit control is initialized with today's date.





Now let's add a new edit control for displaying the time in real time.

- ▶ To create the time type edit control:
 1. On the "Creation" pane, in the "Usual controls" group, expand "Edit" (click the arrow). The list of available edit controls is displayed. Select the "Time" control and position the control inside the window.
 2. Right-click the control and select "Description".
 3. In the "General" tab of the description window:
 - Enter the control's name: "EDT_Time".
 - Enter the caption: "It is: ".
 - The display mask selected by default displays the hours, minutes and seconds. Don't change anything.
 4. As with the Date control, this control only allows you to display the time. Select the "Read only" option in the "GUI" tab of the control description window.
 5. Validate.

- ▶ Similarly to the Date control, we're going to initialize the Time control with the current time.
 1. Display the code associated with the control you've just created.
 2. In the "Initialization of EDT_Time" process, enter the following code:

```
Myself = Now()
```

The function named **Now** is a WLanguage function used (as its name implies) to get the current time.

- ▶ Save the window and test it (to do so, click  then ) . The date and time are displayed. Now we're going to refresh the time so it stays current.

Implementing the timer

To update the Time control, we're going to call a procedure every second; this procedure will update the time control with the actual time.

- ▶ To create a local procedure:
 1. In the "Project explorer" pane, select the name of the window: "WIN_Timer".
 2. Click the arrow on the left to display the different elements.
 3. Click "Local procedures". the folder is selected.
 4. In the "Local procedures" popup menu (right click), select "New local procedure".
 5. In the window that opens, specify the name of the procedure, "RefreshTime", and validate.
 6. The procedure is displayed in the code editor. Enter the following code:

```
EDT_Time = Now()
```

7. This code updates the control that displays the time. We are now going to call it every second via the timer.

- ▶ To implement the timer:
 1. Go back to the "WIN_Timer" window. You can for example click the "WIN_Timer" button at the bottom.



2. Click inside the home window of the window (green background). Display the popup menu (right mouse click) and select "Code".
 3. The processes associated with the window are displayed.
 4. In the "Initialization of EDT_Timer" process, enter the following code:
- ```
TimerSys(RefreshTime, 100, 1)
```
5. This code specifies that the "RefreshTime" procedure is going to be called every second (100 represents the hundredth of seconds).

- ▶ Our window is now created. Save the window and test it (by clicking  then  among the quick access buttons).



Notes

We've seen how to create a timer via programming. You can also create a timer automatically from the code editor. For more details, see the online help (keyword: automatic procedures).


## Managing the input of a numeric value to perform a calculation

In a new window, we're now going to:

- Create two numeric edit controls.
- Calculate and display the IOT value of the entered BT amount.

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

- ▶ Create a new blank window:

1. Click  among the quick access buttons. Click "Window" in the wheel that is displayed. The wizard for window creation starts.
2. Select "Blank" and the "Elegant" skin template.
3. Validate. The window is automatically created in the editor.
4. Right-click the window and select "Description".
5. Name the window "WIN\_Calculation" and give it the title "Example of calculation". Validate the description window.
6. Save the window.

- ▶ To create the control used to enter 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. Select the "Currency" control and position the control in the window.
2. Right-click the control and select "Description".
3. In the description window:
  - Enter the name of the control: "EDT\_PriceBT".
  - Enter the caption: "Price BT".
  - Select the type: "Currency + Euro".
4. Validate.

- ▶ To create the 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. Select the "Currency" control and position the control in the window.
2. Enter the control information: right-click the control and select "Description".
  - Specify the name of the control: "EDT\_PriceIOT".
  - Modify the caption to "Price IOT".
  - Select the type: "Currency + Euro".
3. The result displayed in the control must not be modifiable. To ensure this, click the "GUI" tab and choose the initial status as "Read only".
4. Validate.
5. Save the window.

Some "Numeric" edit controls are also available (see the online help for more details).

The IOT amount will be calculated whenever the control containing the amount before tax is modified.




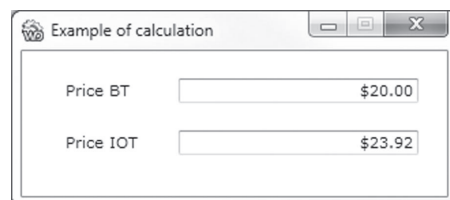
► To calculate the IOT amount:

1. Display the code of the "EDT\_PriceBT" control ("Code" from the popup menu of the control).
2. Enter the following code in the "Whenever modified" code:

```
// Tax rate set at 19.6% for the example
// This could have been any variable
// coming from a database
EDT_PriceIOT = EDT_PriceBT * 1.196
```

3. Close the code window ("x" icon from the code window).
4. Save the window.

5. Test the window: click  among the quick access buttons. Enter a value in the "Price BT" control. The result is automatically displayed in the "Price IOT" control.



6. Close the test window.

This calculation uses a VAT with a fixed rate. To make our example a little bit more complicated, we will give the ability to select the VAT rate in a drop-down list box (also called "combo box").

► To create the combo box for selecting the VAT:

1. On the "Creation" pane, in the "Standard controls" group, click "Combo" then click inside the window at the position where the control must be created (next to the "Price" control).
2. The wizard for creating a combo box starts. This wizard is used to define the main characteristics of the control.
3. Select "Fill the combo box by programming or enter a list of values". Display the next screen.
4. Keep the options proposed by default. Display the next screen.
5. Enter the values of possible VAT values:
  - 5.5
  - Press the ENTER key
  - 19.6
  - Press the ENTER key
  - 25.
6. Display the next screen; enter the name of the control (COMBO\_VAT) as well as its caption (VAT).
7. Validate.

We are now going to modify the code of the "Price BT" control in order to take into account the VAT rate selected in the combo box.

► To take the selected VAT rate into account:

1. Display the code of the "Price BT" control ("Code" from the popup menu).
2. Modify the code as follows:

```
EDT_PriceIOT = ...
 EDT_PriceBT * (1 + Val (COMBO_VAT..DisplayedValue) / 100)
```

This code calculates the IOT amount by using the value selected in the combo box (returned by **.DisplayedValue**). The content of the combo box being a character string, **Val** is used to get a numeric value in order to calculate the IOT amount.



Notes

WLanguage includes functions and properties. The functions may accept parameters and they return results. The properties are directly applied to the controls via the following syntax:

<Control Name>..<Property Name>.

At any time you can consult the online help of a function by using the F1 key (for more details about the online help, see "How do I access the online help?", page 18).

3. Run the test of your window. Enter a value in the "Price BT" control. The result is automatically displayed in the "Price IOT" control according to the selected VAT rate. However, the value of the "Price IOT" control is not modified when the VAT rate changes. We are now going to fix this problem.
4. Close the window. The code editor is redisplayed.

► To take the VAT rate into account whenever the VAT rate is modified in the combo box, the "Whenever modified" code of the "Price BT" button must be copied to the "Selecting a row" process of the "COMBO\_VAT" combo box:

1. Display the code of the "Price BT" control ("Code" from the popup menu).
2. Select the code found in the "Whenever modified" process of the "Price BT" control.
3. Press "CTRL" + "C".
4. Display the code of the "VAT" combo box.
5. Go to the "Selecting a row" process and press "CTRL" + "V".
6. Run the test of your window and select different values in the combo box.



Now, your window operates properly. **However, the same code is used at 2 different locations! How do I manage the modifications and apply them? The easiest method is to use a procedure. That's what we are going to do now.**

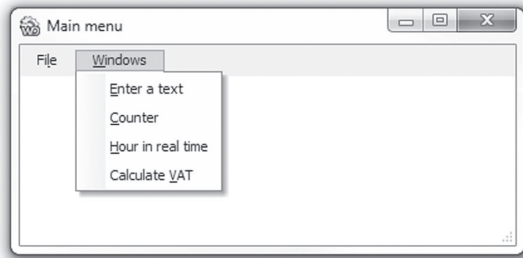
## Using a procedure

The method for creating a procedure from an existing code is very easy: a menu option takes care of everything.

- To create the procedure for calculating the VAT:
  1. Select the code found in the "Selecting a row" process of the "VAT" combo box.
  2. Display the popup menu of the selected control (right mouse click).
  3. Select "Create a procedure .. Create a local procedure containing the selected code".
  4. Enter the name of the procedure to create: CalcVAT. Validate.
  5. The local procedure named "CalcVAT" is automatically created from the selected code. Your code was replaced by the call to the "CalcVAT()" procedure.
  6. Then, all you have to do is replace the second code for calculating the VAT (found in the "Whenever modified" code of the "Price BT" control) by the call to the "CalcVAT()" procedure.
  7. That's it, you're done. You can save your window and test it to check that everything is working the way you want.

## Creating a window with a menu


We have just created some windows. We are now going to create the main window of this example. This window will contain the menu providing access to the other windows of the application.



WinDev gives you the ability to create drop-down menus.

A menu is always associated with a window. To create a menu, the window that will be used to display this menu must be created beforehand.

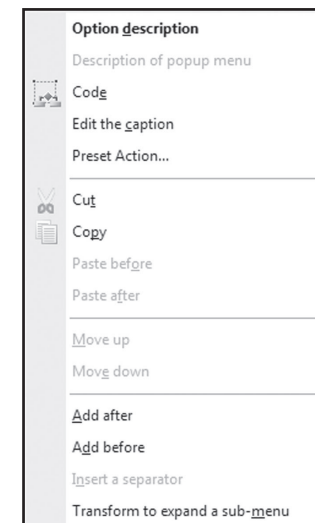
## Describing the menu

- To create a window containing a menu:
  1. Click  among the quick access buttons for creating a window.
  2. Click "Window" in the wheel that is displayed.
  3. In the wizard, select "Blank" and validate.
  4. Give a name and a title to this window. The name of this window is "WIN\_Menu" and its title is "Main menu".
  5. Save the window.

6. On the "Window" pane, in the "Bars and menus" group, expand "Main menu" and select "Add the main menu". A menu is inserted into the window, below the title bar. This menu contains a default option named "Menu".

**Note:** Each menu option is directly entered in the menu. Each option contains a shortcut. This shortcut gives direct access to the option by pressing [Alt]+Letter. The '&' character must be found in front of the shortcut ("&Windows" for example).

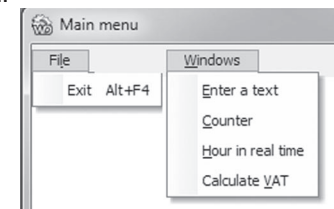
- To create a menu option and to modify it, use the popup menu of the menu or the popup menu of the menu option. To display the popup menu of a menu option:
  1. Select the menu.
  2. Click with the right mouse button.
  3. The following popup menu is displayed:



Several choices are available. The main options for menu management are:

- "Option description" to modify the caption and the shortcut of the option.
- "Code" to enter the source code corresponding to the process that will be run when this option is clicked.
- "Add after" to add a menu option **after** the current option.
- "Add before" to insert an option **before** the current option.
- "Transform to expand a sub-menu" to add a sub-menu into the current option (tree structure).

- Create the following menu:



## ► For the "File .. Exit" menu:

1. Display the description of the "Menu" option inserted by default and change the caption to "Fi&le". Validate the description window.



Notes

Reminder: The "&" character is used to define the shortcut for the menu option.

2. In the popup menu of the "File" option, select the "Transform to expand a sub-menu" and type "Exit".

## ► For the "Windows" menu:

1. Select "File".
2. In the popup menu of the "File" option, select "Add after" and type "&Windows".
3. In the popup menu of the "Windows" option, select "Transform to expand a sub-menu" and enter "&Enter text".
4. In the popup menu of the "Enter text" option, select "Add after" and enter "&Counter".
5. In the popup menu of the "Counter" option, select "Add after" and type &Hour in real time".
6. In the popup menu of the "Hour in real time" option, select "Add after" and enter "Calculate &VAT".

## ► We are now going to create the [ALT + F4] shortcut for the "Exit" option.

1. Select "Exit".
2. Display the description of the option ("Option description" from the popup menu).
3. In the "Keyboard shortcut" area, expand the combo box, browse through the elements and select "F4". Then, check the "Alt" box.
4. Validate.

Display the window once all the menu options have been entered (click "Preview" on the "Modification" pane or use the [SHIFT] + [F9] key combination). This display mode enables you to get an overall view of the menu. To close this window, click the "Close" button found on the remote control of the preview.



Tip

In your applications, we recommend that you follow the Windows standard for menus. Pay special attention to the following points:

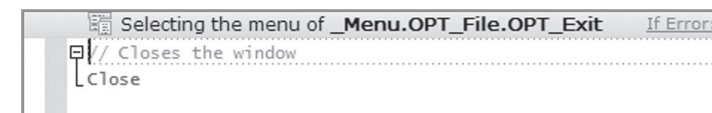
- Each option of the main menu must propose at least one sub-option.
- The first letter of each option must be written with an uppercase character, even if it is not the shortcut. The shortcut is underlined (use the "&" character).
- If a menu choice calls a configuration window before running the chosen option, the caption must end with three dots (...).
- The "Exit" option must be the last sub-option of the menu bar's first option (example: "File .. Exit").
- If there is a help option, it must be found at the end of the first line of the menu (the right-most option).

**Associating source code with the menu options**

Each ending menu option must contain an associated process. In most cases, this process is used to open a window but any type of process can be run.

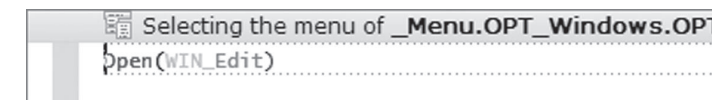
## ► We are going to enter the code for the different options:

1. Select "Exit". This option will be used to exit from the application. Select "Code" from the popup menu. Enter the following code in the code window:



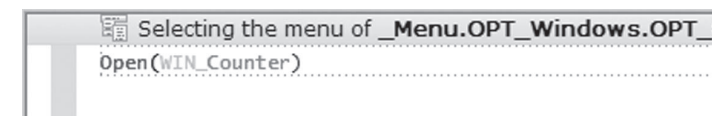
2. Then, close the code window. The window currently edited is displayed.

3. In the window, select the "Enter text" menu option. This option will be used to open the "WIN\_Edit" window. Select "Code" from the popup menu. Enter the following code in the code window:



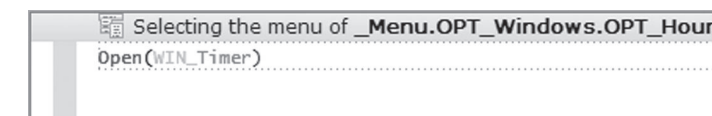
4. Close the code window.

5. Repeat this operation for the "Counter" option and enter the following code:

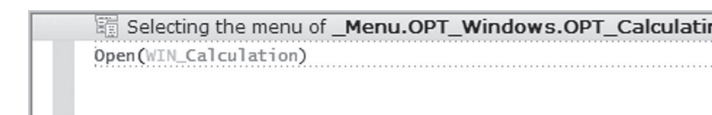


6. Close the code window.

7. Repeat this operation for the "Hour in real time" option and enter the following code:



8. Repeat this operation for the "Calculate VAT" option and enter the following code:



You've guessed it: **Open** is used to display a window allowing you to perform an input. **Close** is used to close the current window.

If the window containing the menu is the first window displayed by the application, closing the window containing the menu (via "File .. Exit" in our example) is equivalent to quitting the application.



Notes

In this code, the WLanguage functions used are in English. If you are a French speaker, you can view and enter code using the French version: on the "Code" pane, in the "Languages" group, expand "Convert the code" and select "Translate into French".

Only the English terms will be used in this documentation. For each term, the online help indicates its translation in French.

- Save the window and run the window's test. During the test:
  - Select one of the menu options. Close the window that comes up.
  - Click "File .. Exit" to exit the program.



Notes

#### Handling the menu options by programming

WLanguage offers several functions and properties for handling menu options.

You have the ability to add menus, menu options or separators and to clone a menu option by programming.

The properties enable you to find out whether an option is visible or not, checked or not, ...

For more details, see the online help (Keyword: "Menu").

## WinDev: Main concepts and terminology

After these hands-on exercises let's go over some of WinDev's main concepts and specific terminology.

### Main concepts

WinDev enables you to easily create an application. But what is an **Application**?

An **application** is a tool allowing you to performs tasks, actions automatically. An application contains 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, and so on). It will be started by the end user of an application.

To create an executable WinDev offers to create a **project**. A project links and organizes the different elements of the program. The executable program will be created from the project.

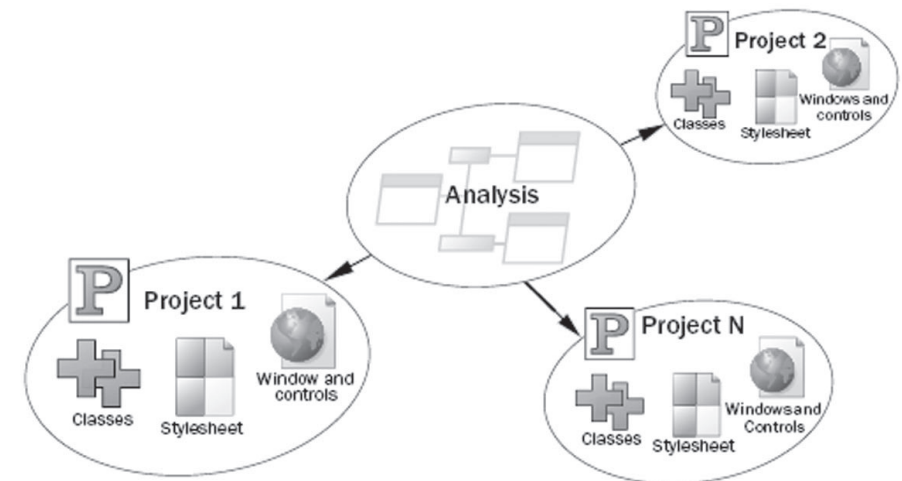
If your application handles data, WinDev enables you to define the structure of the database via the **analysis**. The WinDev analysis contains the description of the files (also called "Tables" in several databases). These files will contain the data of the application.



Notes

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

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 uses the same analysis (the executables can also use the same data files at run time).



## 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 differ from the ones used in other tools.

In the **analysis**, the terms used are as follows:

- **File:** the analysis describes the structure of the database files. A "File" may correspond to a "table" in other databases.  
In WinDev, the term "**Table**" is reserved for specifying a graphical object used to visualize a data file in a table format, and/or to enter rows. A table can be used to enter the details of an order for example.
- **Record:** A record is sometimes called row. A data file record corresponds to all the items defined for the file.
- **Item:** In the analysis, the term item means a zone in the data file. All the items found in a data file are used to define the structure of a record.
- **Key/Index:** With WinDev and its HyperFileSQL database, the concept of index is linked to the concept of key. The concept of key is part of the characteristics of an item. The keys are used to improve the data access speed and to simplify the reading of the data files. In WinDev, if a HyperFileSQL data file contains several key items, a single index file will be created at run time.

In **windows and reports**, the terms used are as follows:

- **Window:** windows are used to display or capture information. Windows are also called "Screens" or "Dialog boxes". The user can directly act on the windows via controls, buttons, ...
- **Report:** reports offer a custom view of the information available. 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 in a report.
- **Skin template:** The skin template is used to define the "appearance" of the application: visual appearance of the 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 1.4. PROGRAMMING CONCEPTS

### This lesson will teach you the following concepts...

- The different types of variables
- Main statements of WLanguage
- The main operators of WLanguage
- Procedures and functions
- Processing strings, numeric values and currencies
- Processing dates and times



Estimated time: 30 min



## Introduction

In the previous lesson, we have used the code editor of WinDev and its language, WLanguage. This lesson enables you to discover the programming concepts of WLanguage by presenting the following topics:

- Declaring the different types of variables
- Main statements of WLanguage
- Procedures and functions
- Processing strings
- Processing numeric values
- Processing dates and times

## Declaring the different types of variables

### The different types of variables

WinDev proposes several types of variables (boolean, integer, real, currency, string, date, time, duration, datetime, variant, array, structure, ...).

The syntax for declaring a variable is very simple: all you have to do is specify the name of the variable and its type:

```
VariableName IS VariableType
```

Some examples:

```
Subscript is int
SupplierName is string
ArrayPrice is array of 10 currencies
I, J, K are int
Counter is int = 120
B1 is boolean = False
```

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

### Declaring the variables and their scope

WLanguage enables you to handle two types of variables:

- The global variables
- The local variables

The global variables can be global to a project or to a window. These variables must be declared:

- in the initialization code of the project, to declare the global variables of the project. These variables can be used in all the processes of the project and project elements (windows, reports, ...)
- in the declaration code of the global variables of the window, to declare the global variables of the window. These variables can be used in all the processes of the window and window elements.

All the variables declared elsewhere are local variables.



Caution !

Most developers are tempted to declare all their variables as "global" to a project. This type of programming is quite "easy". All the variables being global variables, they can be handled from any process.

But this type of programming often causes overwritten variables and side-effects problems.

Therefore, we advise you to declare a limited number of global variables and to declare local variables as necessary.

If you want to share values between 2 windows, we highly recommend that you use the method for "passing parameters" that we shall see later in this tutorial.

## Main statements of WLanguage

WLanguage is a 5GL made of:

- WLanguage functions.
- WLanguage properties.
- WLanguage keywords.
- WLanguage statements.

You will discover all these statements as you progress through this tutorial.

The main statements (used to create conditions and loops or to enter comments) will be presented in this lesson.

### Conditional statements

WLanguage enables you to manage the conditional statements such as:

- **IF, ELSE, END** to run a test on a condition.
- **SWITCH, CASE, END** to perform one or more actions depending on the result of a test on a condition.

### Some examples

```
IF Maximum > Random(1, 999) THEN
 Info("Well done, you won!")
ELSE
 Info("Sorry, you lost!")
END
```

```
SWITCH Day
CASE "Monday"
 // First day of the week
CASE "Wednesday"
 // Third day of the week
CASE "Friday"
 // Fifth day of the week
CASE "Saturday", "Sunday"
 // It's the weekend
OTHER CASE
 // It's another day
END
```

## Loop statements

WLanguage proposes several methods for managing loops:

- **FOR, END** for a set number of iterations.
- **WHILE, END** for a specified number of iterations whose exit condition is checked at the beginning of the loop.
- **LOOP, END** for an undefined number of iterations whose exit condition is checked inside the loop. **The BREAK statement enables you to exit from this loop.**



Notes

WLanguage also proposes loops (FOR EACH, END) used to browse the control elements, the character strings, the records of a data file, ... These advanced loops will be presented later in this tutorial.

## Some examples

```
Sub is int = 0
WHILE Sub<Maximum
 Sub ++
END
```

```
Sub is int = 0
LOOP
 Sub ++
 IF Sub>Maximum THEN EXIT
END
```

```
FOR Sub = 1 TO 100 // There is no need to declare Sub
 Cpt = Cpt + 1
END
```

## Comments

To enter comments in your code, the code line must start with // (two "/" characters/).

Example:

```
// This is a comment line
```



Tip

Several code lines can be set in comment; to do so, select the code lines with the mouse or with the keyboard, then press [Ctrl] / (on the numeric keypad).  
To reverse this operation (to remove the comments), select the code lines with the mouse or with the keyboard, then press [Ctrl] [Shift] / (on the numeric keypad).

## Practical example on the different main statements


Now that you are familiar with the main statements, let's try to run a test!

- ▶ Start WinDev 18 (if not already done). Close the current project. On the "Home" pane, in the "General" group, expand "Close" and select "Close the project".
- ▶ In the home window, click "Tutorial" and select the project named "Programming concepts".  
Tip: You can also go to the "Home" tab, then in the "Online Help" group, expand "Tutorial" then select "Programming concepts".
- ▶ Open the "WIN\_Code.wdw" window (on the "Project explorer" pane, in the "Windows" folder, double-click the window's name). This window presents several examples.



Tip

To quickly find a window in the current project, press [CTRL] + [E]. A window is displayed, allowing you to perform a search on all the windows containing the letters entered in the search control. Simply select the window you want and validate and this window comes up in the window editor.

- ▶ Run the test of this window (  among the quick access buttons).
  1. Click the "Loop" tab.
  2. Run the test of the different examples.
  3. Click the "IF/SWITCH" tab.
  4. Run the test of the different examples.
  5. Close the window and go back to the editor.

## Main WLanguage operators

WLanguage offers several types of operators:

- the logical operators (AND, OR, NOT, ...)
- arithmetic operators used to add, subtract, ...
- the comparison operators used to compare values. We've seen a few of these in the previous paragraph (>, <, =, ...)
- the binary operators that allow you to perform operations on binary values,
- the assignment and exchange (=, <=>) operators
- the indirection operators, used to build the name of a control from an expression.
- the operators for character strings (which we'll see in the "Processing strings", page 68 paragraph).
- ...

We're going to detail here the use of the logical operators, comparison operators and indirection operators. We'll see the other types of operators as we need them in this tutorial. For more details about a type of operator, see the online help (keyword: "Operators").


## Logical operators

WLanguage offers 5 logical operators used to perform logical operations and to build conditions:

- AND and `_AND_` which correspond to the logical multiplication.
- OR and `_OR_` which correspond to the logical addition.
- NOT which corresponds to the logical negation.

### Practical example

Nothing is better than an example to understand the subtlety of these operators.

- In the "WD Programming Concepts" project, open the "WIN\_Operator" window.
- Run the test of this window (  among the quick access buttons) and test the logical operators.


## Comparison operators

WLanguage offers a large number of comparison operators that let you perform a lot of operations:

- equality operators: equality (`=`), flexible equality (`~=`) or very flexible equality (`~~`).
- comparison operators: different (`<>`), lower and greater than (`<`, `<=`, `>`, `>=`) or start with (`[=`, `[=~`, `[=~~`).

### Practical example

An example page is available for testing various use cases for the comparison operators.

- In the "WD Programming Concepts" project, open the "WIN\_Operator" window.
- Run the test of this window (  among the quick access buttons) and test the comparison operators.

## Indirection operators

The "indirection" mechanism is used to build the name of a control, the name of a file item or the name of a variable from a string expression.

This enables you to create generic processes independent of the names of controls, variables, file items...

**The indirection is performed by the `{ }` operators.**

To optimize the speed of your applications, the type of the element handled should be specified when using the indirection syntax.

Therefore, if you are using a control, the corresponding type will be *IndControl* (see the example below).

Here are some indirection examples:

- Example of simple indirections:

```
{ "NAME",indControl} = CustName
//is equivalent to NAME=CustName
{ "NAME",indControl} = { "CU.CUSTNAME" }
//is equivalent to NAME=CU.CUSTNAME
{ "WINCUST.NAME",indControl} = CustName
//is equivalent to WINCUST.NAME=CustName
{ "WINCUST"+"NAME",indControl} = CustName
//is equivalent to WINCUST.NAME=CustName
```

- Example of indirection with a procedure:

```
ControlName is string
ControlName = "EDIT1" // EDIT1 is the name of the control
//call to a procedure used to make a control invisible
INVISIBLE(ControlName)

PROCEDURE INVISIBLE(NControl)
{NControl,indControl}..Visible = False
```



Example

See the example named "Handling the indirections" (supplied with WinDev) for more details. This example is accessible from the "Wizards, Examples and Components" pane.

## Procedures and functions

### Definition

As already seen in the previous lesson, when a process is called several times in a project or in a window, it may be interesting to create a procedure containing this process. Then, all you have to do is call the procedure whenever required.

The procedures and the functions are available in programming:

- The **functions** return a result.
- The **procedures** are used to run a specific process.

WinDev enables you to easily manage the two types of processes: in WinDev, there is no difference between a procedure and a function. This is the reason why we shall talk about "procedure" in the rest of this lesson.

You have the ability to create "local" procedures and "global" procedures.

## Local procedure

A "local" procedure is linked to a single window.

A local procedure can only be used in the processes of the window where it was declared (and in the processes of the controls found in this window). It is part of the window.

## Global procedure and set of procedures

The "global" procedures are stored in "sets of procedures". Each "set of procedures" is a file that contains all the global procedures that are associated with it. The extension of this file is ".WDG". For example, a set is used to group the procedures according to a topic: CalcVAT, CalcExpenses, ...

You can create as many sets of procedures as required in a project.

A set of procedures is used to:

- share the global procedures among several developers, for the same project.
- share the global procedures among several projects. Indeed, the same set of procedures can be used in several projects.

## How do I decide whether a procedure must be global or local?

To choose whether a procedure should be global or local, ask the following question: "Will the procedure be used only in this window or can it be called from another window?"

- If the procedure is called "from this window only", the procedure can be "local".
- If the procedure can be called "from other windows", the procedure must be "global".

## About passing parameters

In the "WD My First windows" project, you saw that a procedure could support parameters. The parameters can be mandatory or optional.

The "mandatory" parameters are always defined before the "optional" parameters. The declaration of an "optional" parameter is performed by assigning a default value when declaring the parameter. For example:

```
PROCEDURE MyProcedure (Param1, OptionParam = "Default Value")
```

A window can also return a value to the process that called it. See the online help about `..ReturnedValue` for more details.

## Calling a procedure

To call a procedure in a process, all you have to do is write its name in the code editor and specify parameters if necessary.

WinDev knows how to manage the calls to nested procedures.



Tip

In the code editor, when you are positioned on a procedure name, press the [F2] key if you want to view the code of this procedure. Press [CTRL]+[F2] simultaneously to go back to the name of the procedure in the previous code.

## Creating a procedure

A procedure can be created from the main editor of WinDev. Simply use the "Project explorer" pane.

- ▶ To create a local procedure:
  1. In the "Project explorer" pane, select the name of the window.
  2. Click the arrow on the left to display the different elements.
  3. Select "Local procedures".
  4. Select "New local procedure" from the popup menu of "Local procedures".
  5. In the window that opens, specify the name of the procedure and validate.
  6. The procedure is displayed in the code editor. Enter the code of the procedure.
- ▶ To create a global procedure:
  1. Select the "Project explorer" pane.
  2. Select "Procedures".
  3. Select the set of procedures where the global procedure must be created.
  4. Select "New global procedure" from the popup menu.
  5. In the window that opens, specify the name of the procedure and validate.
  6. The procedure is displayed in the code editor. Enter the code of the procedure.

**Reminder:** A procedure can also be created from the code selected in the code editor ("Create a procedure .. Create a procedure containing the selected code" from the popup menu). That's what we've done in the previous lesson.




Tip

### When should I use procedures?

- When a process is used several times in the same window, we recommend that you use a procedure local to the window that will contain this process.
- When a process is used several times in a set of windows, we recommend that you use a procedure global to the project that will contain this process.

## A practical example about the procedures and functions

Enough theory, let's get down to work!

- ▶ Open the "WD Programming concepts" project if you haven't already done so:
  1. Close the current project if necessary.
  2. In the home window, click "Tutorial" and select the project named "Programming concepts".
- ▶ Open the "WIN\_Code.wdw" window (double-click its name in the project explorer). This window presents several examples.
- ▶ Run the test of this window (  among the quick access buttons).
  1. Click "Function" to check a function's implementation.
  2. Click "Procedure" to check a procedure's implementation.
  3. Close the window.
- ▶ Display the list of local procedures (accessible from the "Project explorer" pane). When you double-click the name of the procedure, the code of this procedure is displayed in the code editor.


## Processing strings

The ability to handle character strings is one of the most important features of a programming language.

WLanguage offers several features for handling character strings: WLanguage functions, extraction operators, concatenation operators, ...

The most common functions for handling character strings will be presented in this tutorial. See the online help (keyword: "Character string") for more details.

### Practical example

- ▶ If needed, open the "WD Programming Concepts.WDP" project.
- ▶ Open the "WIN\_String.WDW" window (double click its name in the project explorer). This window illustrates the various operations that we're going to see in this lesson.
- ▶ Run the test of the window (  among the quick access buttons).

### Details

A text control (a static control for example) can be initialized:

- with the string directly:

```
EDT_TEXT1 = "WinDev is a great tool"
```

- with a string variable:

```
str is string
str = "I'm learning how to use WinDev"
EDT_TEXT2 = str
```

A string can be built from several other strings. We talk about **string concatenation** then. To concatenate two strings, use the "+" operator.

```
// Info is used to display the result on the screen
Info(EDT_TEXT2 + EDT_TEXT1)
```

A section of a string can be extracted by:

- the [[ and ]] operators (caution: no space must be found between the [[ and ]] brackets).

```
Info(EDT_TEXT1[[1 to 6]]) //displays "WinDev"
```

- the function named **ExtractString**, which extracts a sub-string from a string:

```
Info(ExtractString(EDT_TEXT1,1," ")) //displays "WinDev"
```

- the function named **Middle**, which extracts a section of a string from a string:

```
Info(Middle(EDT_TEXT2,19)) //displays "WinDev"
```

- the function named **Left**, which returns the left section of a string:

```
Info(Left(EDT_TEXT2,12)) // displays "I'm learning"
```

- the function named **Right** that returns the right section of a string:

```
Info(Right(EDT_TEXT1,9)) //displays "fantastic"
```

The size of a string is returned by **Length**:

```
Info(Length(EDT_TEXT2)) //displays 30
```

A string can be converted into uppercase characters by **Upper** or into lowercase characters by **Lower**:

```
Info(Upper(EDT_TEXT2))
Info(Lower(EDT_TEXT2))
```

A string can be sought in another string by **Position**:

```
SoughtString is string = "WinDev"
Pos is int
Pos = Position(EDT_TEXT2, SoughtString)
IF Pos = 0 THEN
 Info(SoughtString + " not found in text2")
ELSE
 Info(SoughtString + " found in text2")
END
```



Tip

You can also find the position of a character string inside another one while ignoring the case. Simply specify the **IgnoreCase** constant with the **Position** function.

Example:

```
Pos = Position(EDT_TEXT2, SoughtString, 1, IgnoreCase)
```

To find out the number of occurrences of a given character string inside another character string, use **StringCount**:

```
NbOccurrences is int
NbOccurrences = StringCount("anastasia", "a") // Returns 4
```

## Processing numeric values

The calculations on numeric values can be performed from numeric edit controls or by directly using the typed variables (integer, real, numeric, currency, ...)

### Practical example

- ▶ If needed, open the "WD Programming Concepts.WDP" project:
  1. Close the current project if necessary.
  2. In the home window, click "Tutorial" and select the project named "Programming concepts".
- ▶ Open the "WIN\_Numeric.WDW" window (double-click its name in the "Project explorer" pane).
- ▶ Run the test of the window. This window presents the operations that can be performed on the "numeric" values.



## Details

A numeric edit control can be initialized:

- with the numeric value directly:

```
EDT_NUM1 = 3.14
```

- with a numeric variable:

```
int1 is int
int1 = 1234
EDT_NUM1 = int1
```

The "+" operator is used to concatenate a string and a numeric value:

```
Info("The EDT_NUM1 control contains the value: "+ EDT_NUM1)
```



Tip

If you run the following code:

```
Info("A calculation: " + 1 + 2)
```

The dialog box will display "A calculation: 12".

To display the result of the calculation, use the following code:

```
Info("A calculation: " + (1 + 2))
```

The dialog box will display "A calculation: 3".

Some examples of numeric operations performed in WLanguage:

- The integer part of a number is returned by **IntegerPart** and the decimal part of a number is returned by **DecimalPart**:

```
Info("Integer part of "+EDT_NUM1+": "+...
 IntegerPart(EDT_NUM1), ...
 "Decimal part of "+EDT_NUM1+": "+...
 DecimalPart(EDT_NUM1))
```

- The absolute value of a number is returned by **Abs**:

```
Info("Absolute value of "+EDT_NUM2+": "+Abs(EDT_NUM2))
```

- The rounded value of a number is returned by **Round**:

```
Info("Rounded value of "+EDT_NUM2+": " + Round(EDT_NUM2,1))
```

- The function named **Root** is used to calculate the Nth root of a number:

```
EDT_Root = Root(EDT_Power, 2)
```

- The function named **Power** is used to raise a number to the power of N:

```
EDT_Power = Power(EDT_Root, 2)
```

## Note: the type of the numeric edit controls

When a control is defined as a numeric control, its type may be undefined (integer, real, double real, ...). Its type is automatically defined according to the mask selected for the control.

To force the type of a control, all you have to do is use a typed variable. For example:

```
Val1 is int
Val1 = 123456789 //assign the control
EDT_NUM1 = Val1
Val1 = EDT_NUM1 //retrieve the control
```

## Processing currencies

### Practical example

- ▶ If needed, open the "WD Programming Concepts.WDP" project:
  1. Close the current project if necessary.
  2. In the home window, click "Tutorial" and select the project named "Programming concepts".
- ▶ Open the "WIN\_RealCurrency.wdw" window (double click its name in the "Project explorer" pane).
- ▶ Run the test of the window. This window is used to run the test of the different source codes presented in this paragraph.

### Details

The "Currency" type is a real coded on 10 bytes. It is used to give the solution to the two problems not solved by the reals:

- more than 15 significant digits (the double reals support up to 15 significant digits), the "Currency" type supports 23 significant digits.
- avoid an error resulting from the binary coding of reals:
  - A real supports up to 15 significant digits. In fact, the binary coding of the reals does not allow to code all reals with 15 significant digits.
  - The "Currency" type uses a different system for coding reals that causes no rounding error. Indeed, the following code returns an invalid result:

```
// The calculation is false with a Real variable
x is Real
x = 18.6 - 8.6 - 10
Error ("18.6-8.6-10="+x)
```

On the other hand, with the following code, the result is correct:

```
x is Currency
x = 18.6-8.6-10
Info(x)
```

To perform divisions on currencies, we recommend that you use intermediate variables of "Currency" type.

```
x, y are Currencies
x = 12345678901234567.123456
y = 12345678901234567.123456
x = x/y
Info ("21345678901234567.123456", ...
 "/", "12345678901234567.123456", "= "+x)
```



Note

For advanced calculations, WLanguage also offers the **Numeric** type. By default, the Numeric type corresponds to 32 digits for the integer part and to 6 digits for the decimal part (like the currency type). The numeric type gives you the ability to configure the number of digits for the integer part and the number of digits for the decimal part.

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

### Mixing strings and numeric values

The numeric values and the strings can be mixed together. WinDev is very flexible in assigning variables. For example, a string of digits can be assigned into a numeric variable (and conversely). For example:

```
i is int
c is string
i = 123
c = i //the variable c contains the "123" string
c = "456"
i = c //the variable i contains the value 456
```

To transform a number into a character string while respecting a specific format, all you have to do is use **NumToString**. For example:

```
NumToString(1234.567, "012,3f") // returns "00001234,567"
```

- ▶ Open the "WIN\_NumToString.wdw" window. Double click its name in the project explorer for instance.
- ▶ Run the test of the window.

Whenever you need to call **NumToString** and you're not sure which parameters to use, you can run this window: it will inform you of the syntax to use according to the requested result.



Notes

**Val** is the "reverse" function of **NumToString**. This function is used to convert a string into a numeric value.

## Processing dates and times

To easily manage dates and times in your applications, WinDev includes:

- a **Date, Time or Duration edit control**. With this control, no more hassle to specify a valid date or a valid time.
- a **Date, Time or Duration static control**. With this control, no more hassle to display a valid date or a valid time.
- **Date, Time, DateTime and Duration variables**. These variables allow you to easily handle the dates and times by programming and to perform various calculations.

## The dates

### Practical example

- ▶ If needed, open the "WD Programming Concepts.WDP" project:
  1. Close the current project if necessary.
  2. In the home window, click "Tutorial" and select the project named "Programming concepts".
- ▶ Open the "WIN\_Date.WDW" window. This window illustrates the explanations given in the rest of this lesson.

### Input mask and returned value

Dates (like times) are strings with a preset format.

In a "Date" (or "Time") type edit control, we must distinguish between:

- The **edit mask**: this is the format used to enter the date or time.
- The **display mask**: this is the format used to display the date and time.
- The **returned value** (or stored value): this is the value returned by the control to the program.

This information is entered in the description window of the edit control.

- ▶ In the "WIN\_Date" window, display the control description window of the "Enter a date" control ("Description" option of the control's popup menu). Display the "General" tab of this window. The information displayed are as follows:

For example, for a "Date" type control:

- the input mask will be "MM/DD/YYYY". The date entered by the user will have the following format: "11/01/2009".
- the display mask will be "DDDD DD MMMM YYYY". The date displayed in the control will have the format "Tuesday 13 June 2000".
- by default the returned value will be "YYYYMMDD" (for example: the value entered in "23/04/2009" format will return "20090423" to the program).



Note

You have the ability to choose a "System date" input mask. In this case, the format displayed will be the one defined in the "Regional settings of the Windows control panel" of the computer currently run.

Let's take a look at the following examples to understand the relationships between the input mask and the returned value:

- If the input and display mask of a date control is "DD/MM/YY" and if the returned value is "YYYYMMDD":

```
DDATE = "19981225"
// Displays the date in the "25/12/98" format
DDATE = "981225"
//Display the date in an incorrect format
```

The date displayed will be incorrect.

- If the input and display mask of a date control is "MM/DD/YYYY" and if the returned value is "YMMDD":

```
DDATE = "981225"
// Displays the date in the "12/25/98" format
DDATE = "19981225"
// Displays the date in an invalid format
```

The date displayed is incorrect.

**You should pay great attention to the format of the value returned in a date or time control.**

The WLanguage functions that handle:


- the dates use the "YYYYMMDD" format.
- the times use the "HHMMSSCC" format (some of them use the "HHMMSSCCMMM" format to manage the milliseconds).



Notes

**Reminder:** By default, the control format (input mask) corresponds to the date mask defined in the "Languages" tab of the project description window (on the "Project" pane, in the "Project" group, click "Description"). Therefore, the same mask is automatically used in all the date controls of the application.

This feature is also very useful in the multilingual applications.

- Now let's see how to work with dates in your programs. If needed close the control description window and test the window "WIN\_Date" (  among the quick access buttons).

## What is today's date?

To find out today's date, all you have to do is use **Today** (or **DateSys**). **Today** returns the system date of your computer as a character string in "YYYYMMDD" format. Example:

```
Info("Today is " + Today())
```

- To check this feature in our example, click the "Today" button.  
You will notice that the date is displayed in "YYYYMMDD" format.

To display the date in a more explicit format, all you have to do is use **DateToString**:

```
Info("Today is " + DateToString(Today(), maskSystemDate))
```

**DateToString** transforms a string in "YYYYMMDD" format into a string in the selected format. The **maskSystemDate** constant allows you to use the date format defined in the "Languages" tab of the project characteristics.

Reminder: To display the project's characteristics, on the "Project" pane, in the "Project" group, click "Description".

- To check this feature in our example, click the "DateToString" button.

## What is today's date, with the day and the month in letters?

To spell out the date, use **DateToString** and use a specific date format:

```
Info("Today is " + DateToString(Today(), "DDDD DD MMMM YYYY"))
```

In this code:

- the "DDDD DD" string spells out the date. For example: Monday 17.
- the "MMMM" string spells out the month. For example: September.
- the "YYYY" string provides the year. For example: 2012.

- To check this feature in our example, click the "In letters" button.

## How many days between two dates?

You want to find out how many days have passed between two dates? It's quite simple: use **DateDifference**:

```
NumberDays is int
NumberDays = DateDifference("20100101", Today())
InfoBuild(...
 "The number of days between %1 and %2 is: %3", ...
 DateToString("20100101", maskSystemDate), ...
 DateToString(Today(), maskSystemDate), nNumberDays)
```

In this piece of code, **InfoBuild** is used to display the result. This function allows you to build a characters string based on parameters. These parameters are specified in the string as %1, %2, ...



Notes

#### Why not directly build a string with the result from the functions?

Of course, you could have built the message's content as follows:

```
Info("The number of days between" + ...
 DateToString("20010101", maskSystemDate) + " and the " + ...
 DateToString(Today(), maskSystemDate), ...
 nNumberDays)
```

This notation presents several drawbacks:

- the readability of the code: a string is easier to read.
- the translation: a string can be entirely translated. The words and the parameters can be reversed. The translator has only one obligation: put back the %1, %2, %3.

► To check this feature in our example, click the "DateDifference" button.

#### What is the day of the week corresponding to a given date?

As easily you can find out the day corresponding to a given date by using `DateToDayInAlpha`:

```
D is string
D = DateToDayInAlpha("17890714")
Info("July 14th 1789 was a " + D)
```

► To check this feature in our example, click the "DateToDayInAlpha" button.

#### Practical exercise

Now that you are a bit more familiar with dates: what is the purpose of the following source code?

```
DateToString(Today(), "DDDD DD MMM YYYY")
```

Answer: Displays the literal date (the day and the date in everyday's language).

#### Handling Date types variables

WLanguage offers specific variable types for Date, DateTime and Duration type values.

These variables can be handled as "strings". Each one contains a value according to the following table:

| Type     | Default format   |
|----------|------------------|
| Date     | YYYYMMDD         |
| DateTime | YYYYMMDDHHMMSSCC |
| Duration | YYYYMMDDHHMMSSCC |

The "Date" type supports the dates from 01/01/0001 to 12/31/9999 (that should keep us going for a while!).

More seriously, to handle the value of these variables, you can use the following syntaxes:

```
MyDate is Date = "20121021"
Info(DateToString(MyDate)) //Displays "21/10/2012"
MyDate..Year = MyDate..Year + 1
MyDate..Month = MyDate..Month + 1
MyDate..Day = MyDate..Day + 1
Info(DateToString(MyDate)) //Displays "22/11/2013"
```

In this code, **Year**, **Month** and **Day** are WLanguage properties.

► In the "WIN\_Date" window, the "Code example" button calculates the date of the next January 1st by using a Date type variable.

The code used is as follows:

```
MyDate is Date
// MyDate is automatically initialized to today's date

//Calculate the date of the next January 1st
NewYear is Date
NewDate..Year++
NewYear..Day = 1
NewYear..Month = 1
MyDuration is Duration = NewYear - MyDate

Info(StringBuild(...
 "There are %1 days left before the next New Year's Eve", ...
 " on %2.", MyDuration..InDayss, DateToString(NewYear))
```

## The times

#### Practical example

- If needed, open the "WD Programming Concepts.WDP" project.
  1. Close the current project if necessary.
  2. In the home window, click "Tutorial" and select the project named "Programming concepts".
- Open the "WIN\_Time.WDW" window. This window illustrates the explanations given in the rest of this lesson.
- Run the test of this window.

### What time is it?

To find out the time, all you have to do is use **Now** (or **TimeSys**). **Now** returns the current time of your system as a character string in "HHMMSSCC" format:

```
Info("It is " + Now())
```

► To check this feature in our example, click the "Now" button.

► You want to display the time in "HH:MM:SS:CC" format? The code becomes:

```
Info("It is " + TimeToString(Now()))
```

**TimeToString** transforms a time in "HHMMSSCC" format into a string in "HH:MM:SS:CC" format.

► To check this feature in our example, click the "TimeToString" button.

If you do not want to display the hundredths of a second, the code is:

```
Info("It is " + TimeToString(Now(), "HH:MM:SS"))
```

Simply specify the time's display format using **TimeToString**.

► To check this feature in our example, click the "Without the hundredths" button.

### How much time has passed between two given times?

It is 17:25. How much time has passed since 12:15? The code is as follows:

```
// Displays the time passed since 12:15
Diff is int = TimeDifference("1215", Now())
Time_Sign is string
Result_Time is string = IntegerToTime(Abs(Diff))

IF Diff < 0 THEN Time_Sign = "-"
InfoBuild(...
 "Time passed since 12:15 -> %1 %2 h %3 min %4 s ", ...
 Time_Sign, Result_Time..Hour, ...
 Result_Time..Minute, Result_Time..Second)
```

**IntegerToTime** transforms an integer (that corresponds to the number of hundredths of a second since midnight (or 00:00)) into a time in "HHMMSSCC" format.

**TimeToInteger** performs the reverse operation.



Caution!

- No ":" should be included in the time passed in parameter to **TimeToInteger** otherwise the result would be incorrect.
- To calculate durations exceeding 24 hours, use the Date, Time types, ...

► To check this feature in our example, click the "1st example" button.

The same calculation can be done by using Time and Duration type variables.

Specific types of variables are available for the Date, Time, DateTime and Duration values. These variables can be handled as "strings". Each one contains a value according to the following table:

| Type     | Default format |
|----------|----------------|
| Time     | HHMMSSCC       |
| Duration | YYMMDDHHMMSSCC |

The code can become:

```
In12H is Hour
In12H..Hour+= 12
In12H..Minute+= 30
InfoBuild("In 12 hours and 30 minutes, it will be %1", ...
 TimeToString(In12H, "HH:MM"))
```

► To check this feature in our example, click the "2nd example" button.

### Calculations with dates and times

Several WLanguage functions can be used to manage the dates and times and to perform calculations.

See the online help (keyword: "Date, WLanguage functions") for more details.



## LESSON 1.5. QUESTIONS/ANSWERS

### This lesson will teach you the following concepts...

- Questions/Answers



Estimated time: 5 min

### Questions/Answers

#### Question How do I view the element to which the current process belongs?

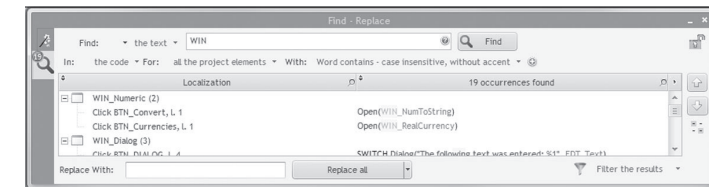
To view the element corresponding to the current process, on the "Code" pane, in the "Navigation" group, click "Go to object" (CTRL + TAB). The window containing the requested element is displayed.

#### Question How do I print the source code?

To print the current source code, click  on the "Home" pane or press [CTRL] + [P].

#### Question How do I find and/or replace a variable in the code?

The functions for performing searches or replacements in the code can be accessed from the "Home" pane, in the "Find" group or in the "Find - Replace" pane:



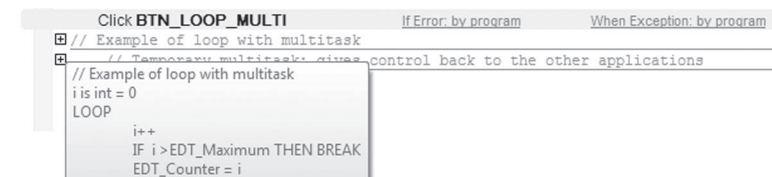
The search can be accessed at any time by pressing [CTRL]+[F].

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

The code editor lets you 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 section of code, on the "Display" pane, expand "Collapse" and select the "Collapse all" option (or use the [CTRL] + [SHIFT] + \* (on numeric keypad) key combination).

Only the comments remain visible. The associated code is displayed in a tooltip when the comment line is hovered by the mouse cursor:



Press [CTRL] + \* (on the numeric keypad) to expand the entire code. A click performed on the "-" or "+" symbol enables you to collapse or to expand the corresponding section of code.

**Question Can the person who wrote a piece of code be identified?**

Press [F6] to display the information (name, date of creation/modification) about each code line.

**Question Is it possible to find out the line number of a code line?**

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

**Question How do I easily display the syntax or the help for a function?**

When typing a function, the syntax of the function is displayed:

- in a tooltip found below the current line. An information tooltip is displayed for each parameter (including for the result of the function).  
If several syntaxes are available, press [ALT] + Right Arrow or [ALT] + Left Arrow to switch from one syntax to another.
- in the status bar of the editor.

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

For the 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 use of the function and automatically generates the corresponding WLanguage code.

A help page is associated with all the WLanguage functions and properties. This help page can be directly accessed from the code editor: to do so, press [F1] on the name of the requested function or property.

**Question What are the useful shortcuts in the code editor?**

- [CTRL]+[L] deletes the current line.
- [CTRL]+[D] duplicates the current line (or the selected lines) on the line below.
- [TAB] 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).
- [F2], when positioned over the name of a control, class, procedure or report block, displays the process of this object.
- [CTRL]+[F2] is used to go back to the initial process.  
To move from one process to another one, press [F2] repeatedly. To go back to the initial process, press [CTRL]+[F2] the same number of times.
- [CTRL]+[R] is used to automatically indent the code displayed.

**Question How do I 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 buttons containing the text of the action to perform.
- the immediate input (**Input**), by allowing the user to enter the requested value in the dialog box.

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

To test the different modes for communicating with the user:

- ▶ If needed, open the "WD Programming Concepts.WDP" project: on the "Home" pane, in the "Online help" group, expand "Tutorial" then select "Programming concepts".
- ▶ Open the window named "WIN\_Dialog.WDW". This window presents the different dialog modes.
- ▶ Run the test of this window ("GO" button).

PART 2

Application with  
data

18



DEVELOP 10 TIMES FASTER 

## LESSON 2.1. OVERVIEW

**This lesson will teach you the following concepts...**

---

- Overview of the application created in this section



Estimated time: 5 min

## Overview of the application created in this section

In this part, we are going to study the databases that can be handled by WinDev. For teaching purposes, we are going to develop an application from A to Z, from the creation of the analysis to the distribution of the application.

You will see the main points for developing an application.

The application that will be created is used to manage bank accounts: you will be able to follow your accounts in real time once the application is developed. The database used is HyperFileSQL Classic, the free database supplied with WinDev. Later in this tutorial, we'll study the HyperFileSQL Client/Server database.

**Notes**

The application developed in this tutorial is a teaching application that does not contain all the elements required to manage a bank account. The purpose of this application is to help you discover the features of WinDev.

Let's see what you are going to do first. The full project of the application is supplied with this tutorial:

1. On the "Home" pane, in the "Online help" group, expand "Tutorial".
2. Select "Application with data (Executable)".
3. Use the application. This application was created by the automatic application generator of WinDev (RAD).
4. Close the application.

## LESSON 2.2. PROJECT AND ANALYSIS

### This lesson will teach you the following concepts...

- Creating a project
- Creating an analysis



Estimated time: 40 min



## Overview

To create an application with a database, you must:

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

We will then create the application via the RAD module (Rapid Application Development).

## Creating the project


► To create the project:

1. Start WinDev (if not already done). Close the current project if necessary.
2. In the home window, click the "Create a project" button, then the "Windows application (desktop)" option. The wizard for project creation starts. The different wizard screens help you creating your project. The information specified in this wizard can be modified later.

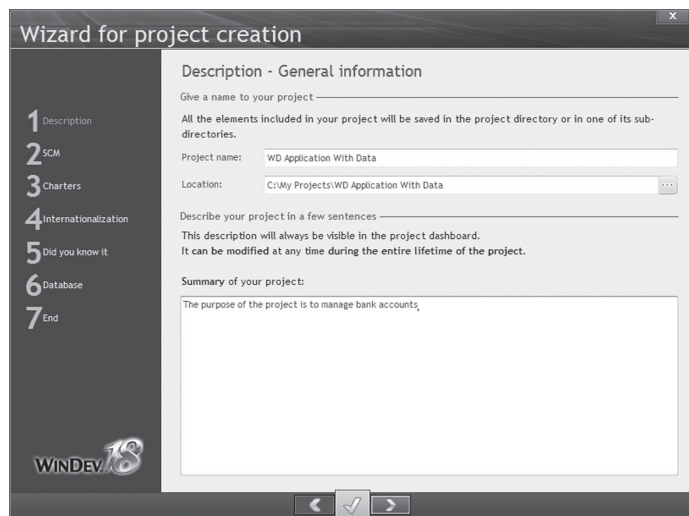


Notes

Tip: To create a project, you can also:

1. Click  among the quick access buttons of the WinDev menu.
2. Click "Project" in the wheel that is displayed.

3. The first wizard screen is used to enter the name of the project, its location and its description. In our case, this project will simply be named "WD Application With Data". By default, WinDev offers to create this project in the "\\My Projects\\WD Application With Data" directory. You can keep this location or modify it via the [...] button. For the summary of the project, type "The purpose of the project is to manage bank accounts".



4. The different steps of the wizard are specified on the left side of the wizard. These steps can be clicked directly. The other screens of step 1 ("Description") are not essential, so click "3 Charters" directly.
5. This step is used to define the programming charter. Don't modify the suggested options. Go to the next screen via the arrows found at the bottom.
6. This step is used to define the style book. Select "ActivUbuntu". Go to the next screen via the arrows found at the bottom.
7. A size of 800x600 will be chosen for the screens as our application will contain no large windows and it will be suitable for most resolutions. The choice would be different for an application used to manage images for example.
8. Click step "5 Did you know it". This screen is used to manage the display of a "Did you know it" type window. We won't display this window: select "Don't display the 'Did you know it' window when starting the application". Go to the next screen.
9. We will now specify the information regarding the database.
10. Select "Yes, create a new database" and validate. The wizard for analysis creation starts.



Notes

To better follow the lessons in this section and optimize your WinDev learning, we advise you to create the "WD Application With Data" project.

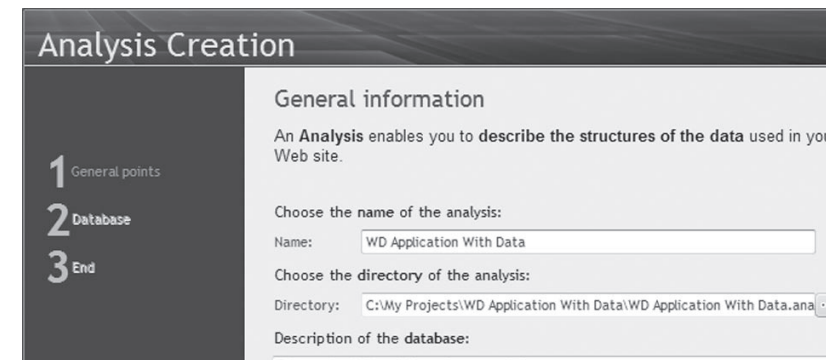
A corrected example can be access at any time to check the validity of the operations performed.

To open the corrected project, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Application with data (Answer)".

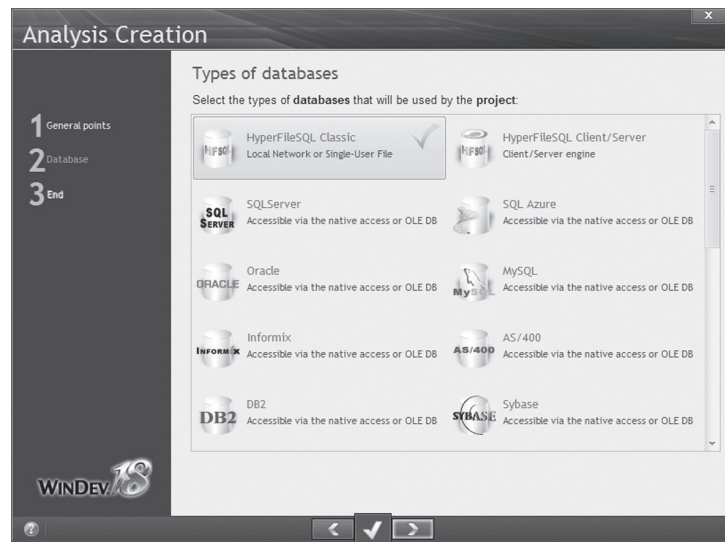
## Creating the analysis

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

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



2. You have now the ability to choose the types of the databases handled by the project. Select HyperFileSQL Classic (the database proposed by default with WinDev).



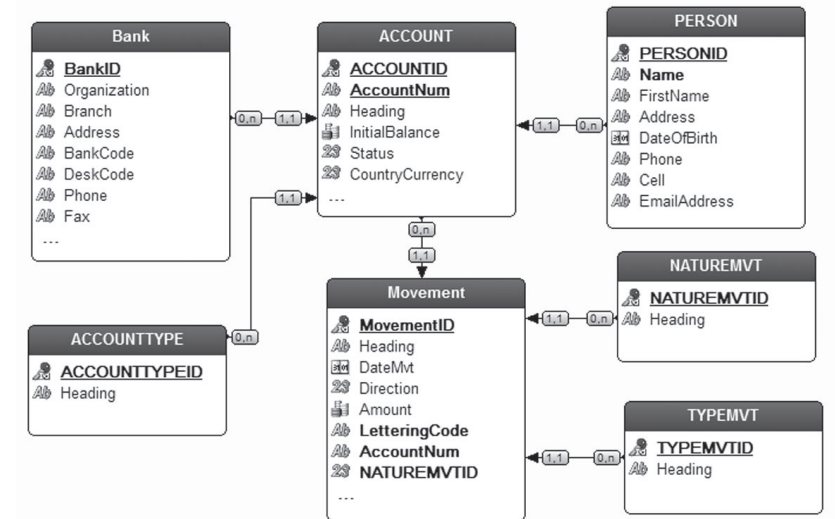
Go to the next wizard screen.

3. Validate. The wizard for creating a data file is automatically started.

## Creating the description of the data files

Our application for account management will be associated with the following analysis. This analysis includes seven data files:

- BANK
- ACCOUNT
- ACCOUNTTYPE
- PERSON
- MOVEMENT
- NATUREMVT
- TYPEMVT



To create the files of this analysis, we are going to use the different methods available in WinDev.

## Creating a data file and its items in the editor

► The steps of the wizard for creating a data file are as follows:

1. In the wizard, select "Create a new description of data file". Go to the next wizard screen.
2. The data file that will be created is the "ACCOUNT" file. Its name is "ACCOUNT". This name will be used:
  - to handle the data file in your code. The variable associated with the file will be account.
  - to build the name of the associated physical data file (ACCOUNT.fic file).

The caption and description of the elements represented by the records found in the data file are automatically displayed.



Notes

In the wizard, the "A record represents" field specifies the information that will be used to describe the links among data files. This information must be specified with great care!

3. Keep "The data file includes an automatic identifier". This option indicates whether the data file must include a unique key, automatically managed by WinDev.



Notes

To create the identifier (an identifier is a unique key), you can create a numeric item whose type is "Automatic identifier".

This identifier is automatically managed by WinDev. Whenever a record is added into the data file, WinDev automatically assigns a value to the identifier of the file. This value is unique.

This choice can be unchecked if no automatic identifier is required (if no unique key is required or if a unique key already exists in the data file).

### Creating a data file

**General parameters**

Specify the name of the data file.

Name:

Caption:

A record represents:

☒ The data file includes an automatic identifier

4. Go to the next screen and select the type of the database associated with the data file. We are going to work on HyperFileSQL Classic data files. Go to the next screen.
5. Keep the default options and go to the next screen.
6. Click the green button to validate. The data file is automatically created in the analysis. The item description window opens up.

**Describing the items and indexes of a data file**

**ACCOUNT**

Number of items and indexes: 1 Size in bytes: 13 Display in physical order

| Key       | Name                  | Caption      | Type         | Size |
|-----------|-----------------------|--------------|--------------|------|
| ACCOUNTID | Identifier of ACCOUNT | Automatic ID | Automatic ID | 4    |

Item not linked to a metatype

Sub-type: Automatic Identifier (4 bytes)

Key: ☒ Unique key ☐ Not key ☐ Key with duplicates

Search direction: ☒ Ascending ☐ Descending

Index parameter and search parameter for text key:

☐ Case sensitive ☐ Accent sensitive ☐ Space, punctuation and special char. sensitive

☐ Array Dimension: 0 Actual size: 0

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

- First, we are going to create the "AccountNum" item. This item will contain the account number.
  1. In the description window of the items, click the "Name" column of the first empty line twice. This column automatically becomes editable. Enter "AccountNum".
  2. Click the "Caption" column. The name of the item is automatically displayed. We're going to modify the caption for the item by clicking it: enter "Account number". In the "Type" column, the "Text" type is automatically selected. Don't change anything.
  3. We are going to resize the item. Click the "50" box and replace "50" by "25". Click the next line. The values are automatically updated.

Number of items and indexes: 1 Size in bytes: 13 Display in physical order

| Key       | Name                  | Caption        | Type         | Size |
|-----------|-----------------------|----------------|--------------|------|
| ACCOUNTID | Identifier of ACCOUNT | Automatic ID   | Automatic ID | 4    |
|           | AccountNum            | Account number | Text         | 25   |

4. This item will be a key item in our data file: the keys are used to improve the speed for accessing the data and to simplify the browse operations performed on the data files. You will be able to perform search or read operations on this item.



Notes

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

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

5. To define the key, you must: reselect the line corresponding to the "AccountNum" item in order to enable the description controls found on the right of the screen. Then, all you have to do is specify the type of the key. In our case, the account number is a unique key.

- Then, create the following items (these items are not key items):

| Name           | Caption         | Type     | Size                                 |
|----------------|-----------------|----------|--------------------------------------|
| Heading        | Heading         | Text     | 50                                   |
| InitialBalance | Initial balance | Currency | The size is automatically set to 10. |

- We are now going to create the "Status" item and the "CountryCurrency" item.

- Position on a new table row. Enter:
  - the name: Status
  - the caption: Status of the account
  - the type: Radio button, list box, combo box. The window that opens enables you to select the type of control used by default for this item. It will be a radio button in our case. Validate the window.

- In the lower section of the screen, click the double arrow to display the parameters of the control linked to the selected item.

The information entered here will be automatically used when creating the windows linked to the data file. You will find here the type of the control and the caption.

We are going to enter the following options in the "Content" tab:

- Click the "Content" tab.
- Click the "+" button to add the first option.
- The option 1 corresponds to Enabled. Enter "Enabled" in the edit control on the right of the check box.
- Click the "+" button to add the second option.
- Enter "Suspended" instead of "Option 2".
- Click "+" again.
- Enter "Closed" instead of "Option 3".

- Click the double arrow again.

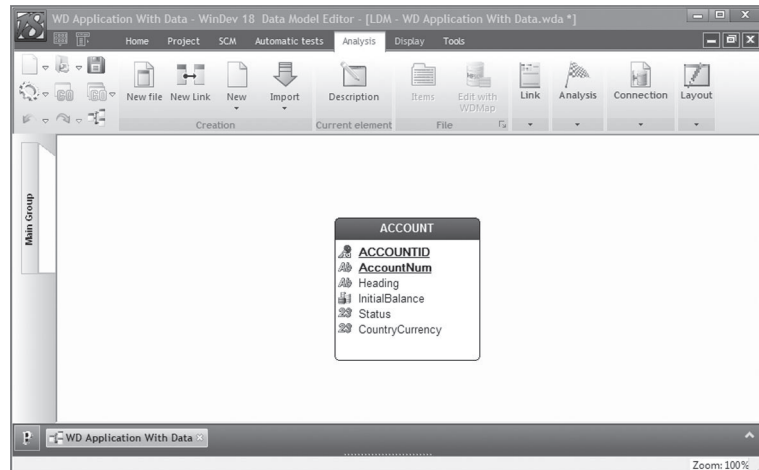
- Similarly, position yourself on a new row in the table and create the "CountryCurrency" item. This item is a "Radio button, list box, combo box" type item. The default control linked to this item is a combo box. The initial content for this combo box must be entered in the "General" tab of the lower section of the screen. In the "Initial content" area, enter the following information:

- EUR - Euro
- USD - American Dollar
- JPY - Japanese Yen
- AUD - Australian Dollar
- CAD - Canadian Dollar

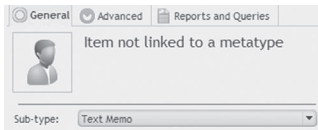
Note: After each value, press [ENTER] to go to the next line.

- That's it, the ACCOUNT file is described. Validate the description window of the items. In the window that is opened, select "Go back to the WinDev editor". Then select the "WinDev editor" option.

6. The ACCOUNT file is displayed in the data model editor. You can enlarge the display of the file. To do so, click the file, select the black handle at the bottom of the file and move the mouse toward the bottom.



Now that you are becoming an "expert" in creating data files, let's create the "PERSON" file. To start the file creation wizard, on the "Analysis" pane, in the "Creation" group, click "New file". The PERSON file includes an automatic identifier and it contains the following items:

| Name        | Caption                  | Type and size                                                                                                                                                                           |
|-------------|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | Name of the person       | Text, 50, key with duplicates                                                                                                                                                           |
| FirstName   | First name of the person | Text, 50                                                                                                                                                                                |
| Address     | Address                  | Text, "Text memo" sub-type<br>The sub-type must be selected in the right section of the window.<br> |
| DateOfBirth | Date of Birth            | Date, 8                                                                                                                                                                                 |
| Phone       | Phone                    | Text, 20                                                                                                                                                                                |
| Cell        | Cell                     | Text, 20                                                                                                                                                                                |

We are going to add a new feature to this data file. We are going to create an email item. Easy! To do so, we are going to use the metatypes.



Notes


### Metatype

A metatype is a combination of characteristics for an item and for its linked control. For example, a "Fax" metatype will contain the type and length of the item, the input mask, the alignment, ...

You can use the metatype items supplied with WinDev or create your own metatypes.

To use a metatype item, click the "Metatypes" button in the description of the analysis items.

► To add an item created from a metatype:

1. Click .
2. Select "Email address".
3. Validate.
4. This item is a unique key.
5. Validate the description window.

### Importing a CSV file

Another method will be used to create the BANK file (that contains the characteristics of the banks): importing a CSV file. From the CSV file containing the data, WinDev will create the description of the data file in the analysis as well as the corresponding HyperFileSQL data file with the data found in the CSV file.

► To import a CSV file into the analysis:

1. In the Windows file explorer, open the following WinDev sub-directory: "\Tutorial\Exercises\WD Application With Data".
2. Select the "Bank.csv" file.
3. "Drag and Drop" the "Bank.csv" file to the data model editor of WinDev. The wizard for importing files starts.
4. The content of the CSV file will be converted to the HyperFileSQL format. Check "Convert the data to the HyperFileSQL Classic or HyperFileSQL Client/Server format". Go to the next wizard screen.
5. Select the format of the files to import. Select "Text file". Go to the next wizard screen.
6. WinDev indicates the path of the file to import. Go to the next wizard screen.



7. Indicate the following import parameters:

**Parameters for importing a text file**

Specify the text file format

The records are delimited by:

The columns are delimited by:

The strings are delimited by:

Decimal separator:

Preview

| Organization   | Branch     | Address                   | BankCode | DeskCode |
|----------------|------------|---------------------------|----------|----------|
| BARCLAY'S      | LONDON E12 | Post Avenue, London       | 5432     | 1001     |
| BARCLAY'S      | LONDON E18 | Money Street, London      | 5432     | 5504     |
| ABBEY NATIONAL | Manchester | Rose Avenue, Manchester   | 7004     | 6460     |
| ABBEY NATIONAL | Liverpool  | Stadium Avenue, Liverpool | 9127     | 9670     |

☒ The first row contains the names of the columns

Don't forget to select the "The first row contains the names of the columns" option.

8. Go to the next screen.

9. The structure of the data file that will be created is displayed. We are going to perform some modifications:


| Item         | Type | Size |
|--------------|------|------|
| Organization | Text | 20   |
| Branch       | Text | 50   |
| Address      | Text | 33   |
| BankCode     | Text | 5    |
| DeskCode     | Text | 5    |
| Phone        | Text | 20   |
| Fax          | Text | 20   |
| Email        | Text | 260  |
| Contact      | Text | 50   |

10. Validate the wizard. The wizard for data conversion starts.

11. Validate the creation directory of the HyperFileSQL data files. Go to the next wizard screen.

12. Uncheck "Save the parameters of the conversion" and validate. The data file appears in the analysis.

► Let's see the description of the imported data file:

1. Select the "Bank" file then "Description of data file" from the popup menu.
2. This data file contains no automatic identifier: check "Automatic identifier".
3. Modify the caption of the file: remove "(Imported)".
4. Click  to display the description of the items found in the data file.
5. We are going to modify some information: the sub-type of the Address item must be "Text memo".
6. We get the following data:

**Bank**

Number of items and index: 10 Size in bytes: 460 ☐ Display in physical order

| Key          | Name               | Caption      | Type         | Size |
|--------------|--------------------|--------------|--------------|------|
| BankID       | Identifier of Bank | Automatic ID | Automatic ID | 4    |
| Organization | Organization       | Organization | Text         | 20   |
| Branch       | Branch             | Branch       | Text         | 50   |
| Address      | Address            | Address      | Text         | 33   |
| BankCode     | BankCode           | BankCode     | Text         | 5    |
| DeskCode     | DeskCode           | DeskCode     | Text         | 5    |
| Phone        | Phone              | Phone        | Text         | 20   |
| Fax          | Fax                | Fax          | Text         | 20   |
| Email        | Email              | Email        | Text         | 260  |
| Contact      | Contact            | Contact      | Text         | 50   |

► To simplify the searches in the data file, we are going to create a composite key.



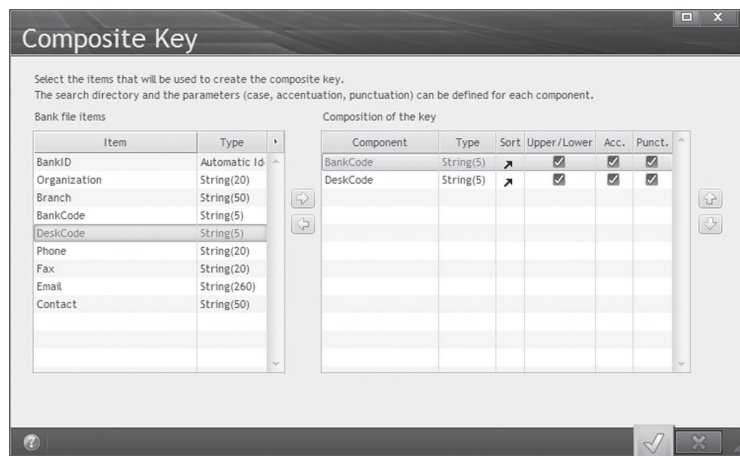
Notes

A composite key is a key that includes several items found in the same data file.  
This type of key is used to browse the data file according to complex criteria or to perform specific searches.

This key will contain the bank code and the desk code.

1. Click .
2. Select "BankCode" and click ">".

3. Select "DeskCode" and click ">".



4. Validate. The composite key appears in the description of the data file.  
 5. Validate the description of the "BANK" file.  
 6. Validate the description of the data files.



## Notes

The data model editor enables you to add full-text keys. These keys are used to find a word or a sentence in one or more text items (character strings, text memo, ...).

This search will be performed via a query and the result of this search can be displayed in a control.

See the online help (keyword: "Full-text index") for more details.

## Importing files from the dictionary

To simplify the creation of the data files in the analysis, a file description was prepared and included in a dictionary beforehand. Let's see how this data file can be imported. A specific WinDev pane, the "Dictionary" pane, enables you to easily handle the dictionaries.

- To import elements from a dictionary:
1. Display the "Dictionary" pane if necessary: on the "Home" pane, in the "Environment" group, expand "Panels" and select "Dictionary".
  2. Click the pane and select "Open" from the popup menu.
  3. Select the "Dictio\_TUT.dic" file found in the following WinDev sub-directory: "\Tutorial\Exercises\WD Application With Data". The dictionary is displayed in the pane. This dictionary contains the "Movement" file.
  4. "Drag and Drop" the "Movement" file to the data model editor.
  5. The editor asks whether a subscription is required. Answer "No". The dictionary will be presented later in this tutorial.
  6. Validate. The description of the "Movement" file is included in the analysis.

## Direct import of existing data files

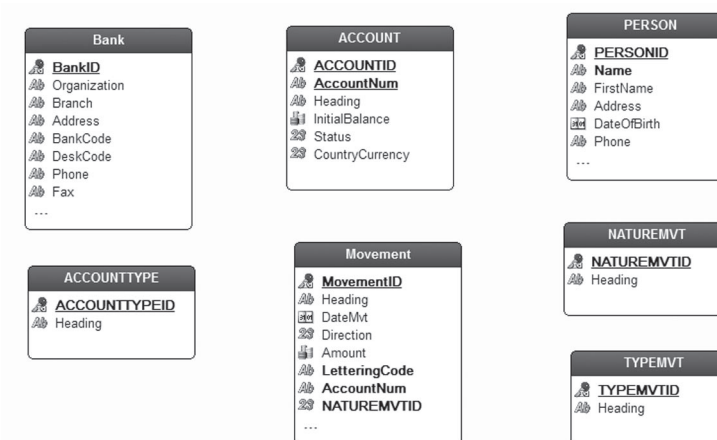
The last method for creating data files consists in importing the existing HyperFileSQL data files. The last files were prepared in this format.

- To import the HyperFileSQL data files:
1. In the Windows file explorer, open the following WinDev sub-directory: "\Tutorial\Exercises\WD Application With Data".
  2. Select the "TypeMvt.fic" file.
  3. "Drag and Drop" the "TypeMvt" file to the data model editor of WinDev.
  4. The import wizard starts. Validate the various screens. The data file appears in the data model editor.
  5. Repeat this operation for the AccountType.fic and NatureMvt.fic files.
  6. To handle the imported data files, the "xxx.fic" and "xxx.ndx" files (found in the "\Tutorial\Exercises\WD Application With Data" directory) must be copied to the EXE directory of your project.

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

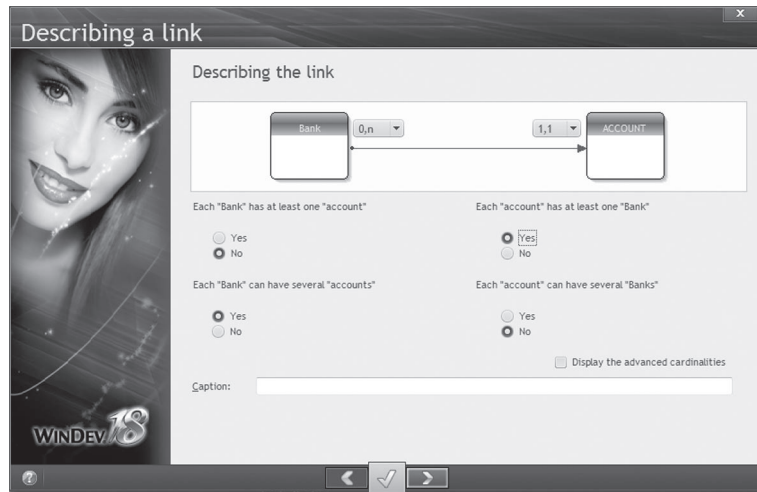
## Creating the links

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



We are now going to create the links between the data files.

- Let's create the link between the BANK file and the ACCOUNT file.
1. On the "Analysis" pane, in the "Creation" group, click "New link". The mouse cursor turns into a pen.
  2. Click the "BANK" file, then click the "ACCOUNT" file.
  3. The wizard for link creation starts.
  4. Answer the questions asked by the wizard:



- Each Bank has at least one account: No
- Each Bank can have several accounts: Yes
- Each Account has at least one bank: Yes
- Each Account can have several banks: No



## Notes

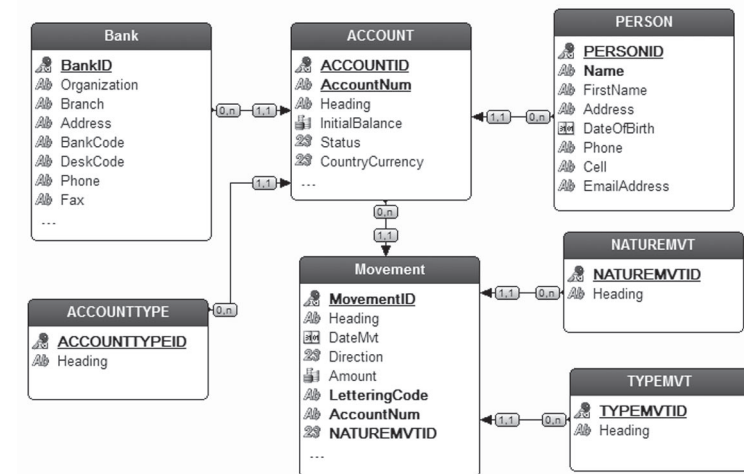
You also have the ability to enter the cardinalities of the link in the wizard.

5. Go to the next screen. The wizard automatically proposes the key used by the link. Display the next wizard screen.
6. The wizard proposes to create a new key in the Account file. Accept this option by going to the next screen.
7. Validate the integrity rules by going to the next screen.
8. Click the green arrow. The link is automatically created in the data model editor.

- You can create the following links:

| Source file | Linked file | Cardinalities |
|-------------|-------------|---------------|
| Person      | Account     | 0,n - 1,1     |
| Account     | Movement    | 0,n - 1,1     |
| AccountType | Account     | 0,n - 1,1     |
| NatureMvt   | Movement    | 0,n - 1,1     |
| TypeMvt     | Movement    | 0,n - 1,1     |

- The analysis is as follows:




## Configuring the analysis for RAD

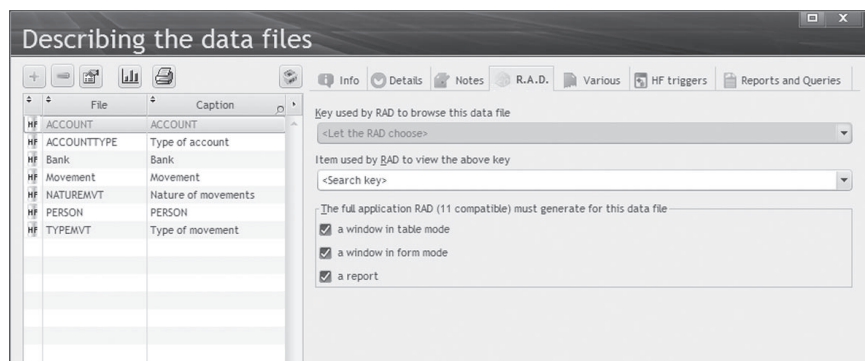
When generating the application, the window or the reports, the RAD module (Rapid Application Development) uses the options specified in the analysis. We recommend that you take a little bit of time to configure these options in order to get the best possible result.

The elements used by RAD are defined:

- in the description of the data files (RAD tab)
- in the description of the items (RAD tab)
- in the shared information specified for each item.

## Configuring the RAD in the description of the data files

- To configure the RAD options of the data files:
  1. Display the data file description: on the "Analysis pane", click the group button of the "File" group (  ).
  2. Display the "RAD" tab.



The following information must be configured for each data file:

- The item to display for the viewing.
- The elements generated by RAD for the data file.

The following information must be configured for each data file:

| Data file   | Item to view | Element to generate |
|-------------|--------------|---------------------|
| ACCOUNT     | Heading      | Table, Form, Report |
| ACCOUNTTYPE | Heading      | Table, Form, Report |
| BANK        | Organization | Table, Form, Report |
| MOVEMENT    | Heading      | Table, Form, Report |
| NATUREMVT   | Heading      | Table, Form, Report |
| PERSON      | Name         | Table, Form, Report |
| TYPEMVT     | Heading      | Table, Form, Report |

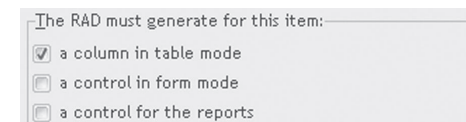
- To perform these modifications:
  1. Select the file in the table.
  2. Perform the modifications in the "RAD" tab.
  3. Go to the next file.

Validate the window at the end of the operation.

## Configuring the RAD in the description of items

- To configure the RAD options of items:
  1. For each data file, display the description of items ("Item description" from the popup menu).
  2. Display the "Advanced" tab.
  3. Configure the RAD options.

For each data file, **the identifiers of the files** (BankID, AccountID, PersonID, AccountTypeID, MovementID, NatureMvtID and TypeMvtID items) must have the following characteristics:



Indeed, the identifier must not be displayed in the windows in form mode and in the reports.

**Leave everything checked for the other items** (including the other identifiers found in the files).

4. Validate the description of the items found in the data file.

The analysis is ready to be used by RAD.

## Generate the analysis

Generating the analysis is used to make the information about the data files available to the other modules of the project. These data files can be handled in the programs.

Without generation, even though the description of the data file exists, you would not be able to use the data file in your programs.

When generating the analysis, all the modifications performed in the analysis and in the data files will be automatically applied to the entire project (windows, linked controls, reports, ...).



Caution!

The generation must be performed whenever you want the modifications made to the analysis to be taken into account in the programs that use this analysis. If the analysis is modified several times, without any programming between each modification, there is no need to generate the analysis whenever it is modified. Generate the analysis once all the modifications have been made and go to the programming step.

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

The descriptions of the data files found in the analysis have been modified (BANK file).

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 screens until the automatic data modification is performed.  
Note: WinDev detects an ambiguity on the copied data files and the imported data files: select the files to update them.
- ▶ Close the data model editor. We can now start programming the application.

## LESSON 2.3. THE FULL RAD

### **This lesson will teach you the following concepts...**

---

- What is RAD?
- Generating RAD
- Testing the project



Estimated time: 20 min



## What is RAD?

R.A.D. stands for "Rapid Application Development".

RAD is used to automatically build an application, which means all the necessary windows, reports and queries.

As already seen in a previous lesson, to develop an application in WinDev, a project and an analysis (if necessary) must be created beforehand. The analysis contains the definition of the structures of the data files handled in the processes.

The RAD module of WinDev uses this analysis. The RAD module contains a wizard allowing you to choose the application template to generate (the RAD pattern) and the main options for your applications.



Note

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 (keyword: "RAD pattern") for more details.

The windows, reports, queries and source 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's the window RAD. The Window RAD is available when a new window is created in your application.

We shall now see how to use the Project RAD module.



Note

### RID (Rapid Interface Design)

WinDev also enables 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.

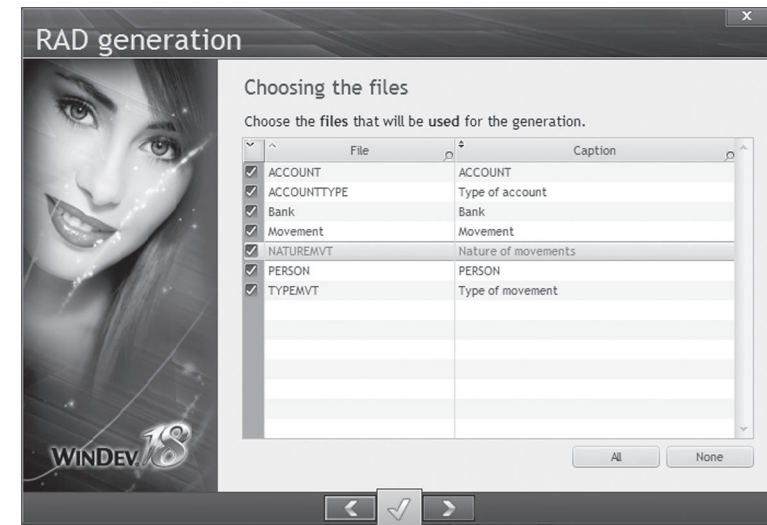
See the online help (keyword: "RID") for more details.

## Generating RAD

► To start generating RAD:

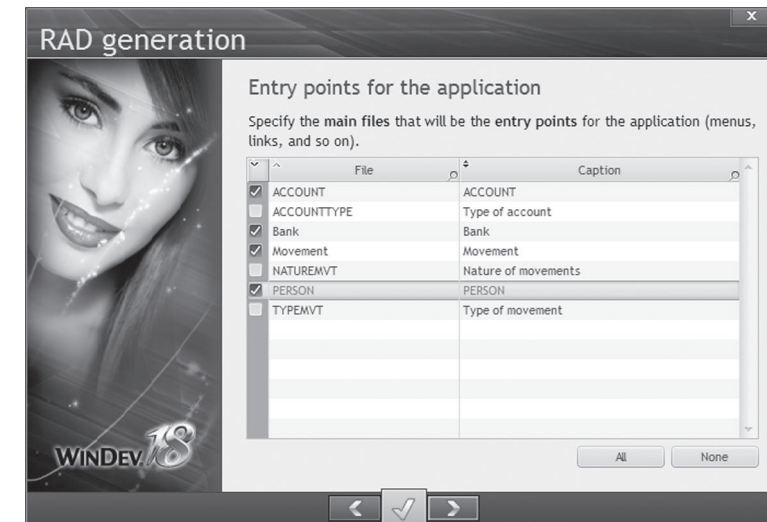
1. On the "Project" pane, in the "Generation" group, click "Full Application RAD". The wizard for generating the RAD application starts.
2. Select the pattern that will be used for the RAD generation: "Simple RAD" for example. Go to the next wizard screen.

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



Bring up the next screen.

4. The entry points of the application correspond to the entries available in the menu.



Keep the Bank, Account, Movement and Person files only. Display the next screen.

5. Specify whether the tables generated in the windows of the application must be editable or not. In our example, the tables will allow the user to enter new information. Select "Yes: Allow the input in the tables". Display the next screen.

6. Specify whether the user groupware must be used in the application. It will be included later in the application: select "No: Don't include the management of user groupware". Go to the next screen.

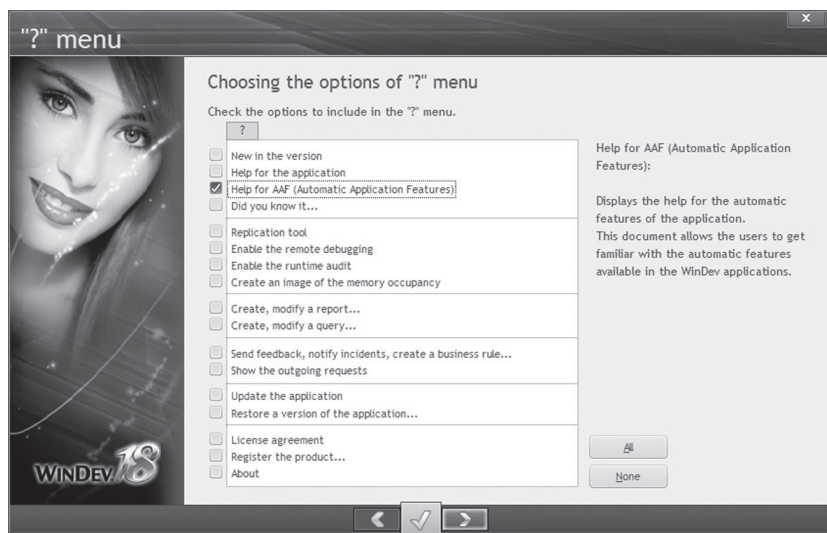
7. Specify whether the automatic menu must be included in the application. Answer "Yes: Include the '?' menu". Go to the next screen. The wizard for RAD generation is over and the wizard for generating the automatic menu starts.



Note

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 steps of the wizard for generating the automatic menu are as follows:



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



Notes

When creating the "?" menu, "? .. Help for the automatic features" has automatically added to your application:

- the "CCMenu" component. Indeed, this option requires a procedure found in this component to work properly.
- the "WinDev AAF 18.PDF" file. This file will be automatically opened when the menu option is used.

The application is generated and its test can be run.



Notes

GUI 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.

## Testing the application

Let's now run the test of the generated application.

► To run the test of the application:

1. Click among the quick access buttons of the WinDev. The application starts.
2. In the menu, select "Person .. List of person".
3. The list of persons is displayed.
4. To add a new person, click the "New" button. An input form is displayed. Enter your personal details for instance and validate.
5. Close the list of persons.
6. In the menu, select "Account .. List of account". The list of accounts is displayed.
7. Click the "New" button. Enter the information regarding the new account. For example:

| Account number        | Heading               | Initial balance |
|-----------------------|-----------------------|-----------------|
| 0123456L030           | THA                   | 1500            |
| Country currency      | Organization          | Person          |
| USD                   | BARCLAY'S             | You for example |
| Status of the account | Heading (AccountType) |                 |
| Enabled               | Bank account          |                 |

8. Similarly, create several movements. For example:

| Heading                | Petrol      | Rent        |
|------------------------|-------------|-------------|
| Date of movement       | 03/03/2013  | 03/04/2013  |
| Direction              | Debit       | Debit       |
| Amount                 | 56          | 737         |
| Account number         | 0123456L030 | 0123456L030 |
| Nature of movement     | Fuel        | Rent        |
| Type                   | Credit card | Check       |
| Heading of the account | THA         | THA         |

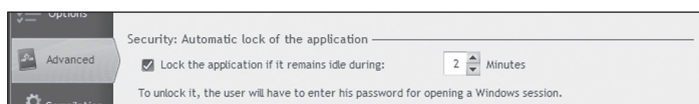
**Note:** To select the dates, use the calendar found on the right of the controls.

## Quick modification: Locking the application

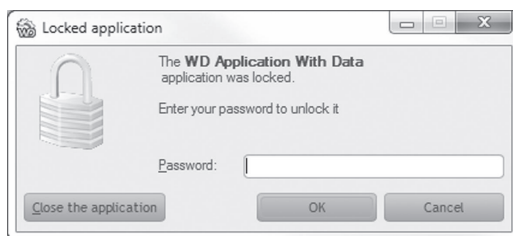
In most cases, when using an application, this application remains opened on the user computer while the user is away (meeting, lunch, ...). To prevent the application from being used by unauthorized persons, the application can be automatically locked. To reconnect, the user will have to specify his identifier and his Windows password.

A few mouse clicks allow you to implement this feature.

- To lock an application:
  1. Display the project description: on the "Project" pane, in the "Project" group, click "Description".
  2. Display the "Advanced" tab.
  3. Specify the lock options (2 minutes for example).



The following window will be displayed when the application is not used for 2 minutes:



The user will have to enter his Windows password to continue to use the application.



**Notes** If the User Groupware is used by the application (see "User groupware", page 140), the password to use is the one defined for the User Groupware.

- To check the lock of the application, run the test of the application and wait for 2 minutes. The password request window comes up. Enter your Windows password and validate.
- To avoid our next tests from being too heavy, we are going to delete this option.
- Display the project description again.
  1. On the "Project" pane, in the "Project" group, click "Description".
  2. In the "Advanced" tab, clear the "Lock application ..." option. Validate.

The rest of this lesson uses the same project. Keep this project open.

## LESSON 2.4. EDIT WINDOW

### This lesson will teach you the following concepts...

- Creating an edit window
- Managing the addition of a record
- Managing the search and the modification

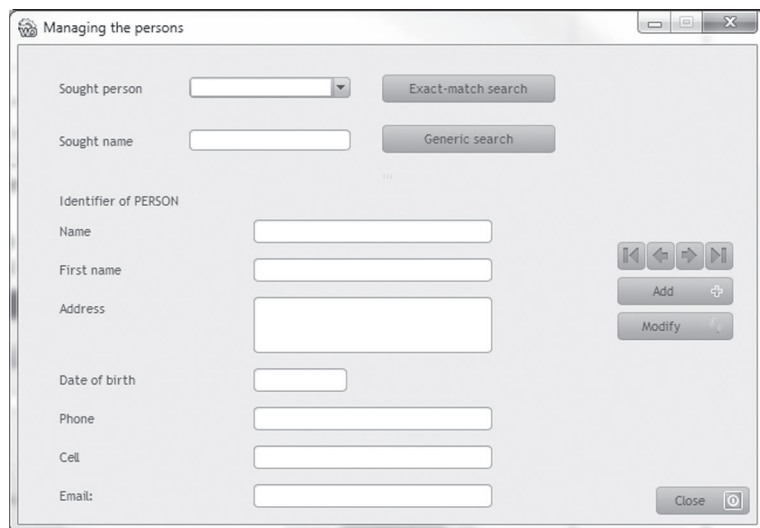


Estimated time: 20 min

## Overview

We have seen the different windows automatically created by RAD. We are now going to create a window used to manage the addition and the modification of records. These operations will allow you to discover several topics regarding the management of the data files and will also enable you to use some features of WinDev.

The window that will be created is used to manage the input, the search and the modification of persons. This window is as follows:




## Creating an edit window

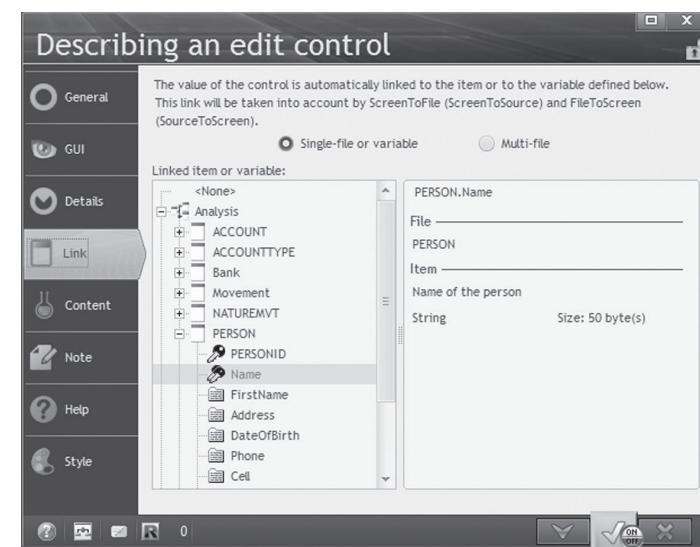
To create the edit window on the Person file, we could use the wizard for creating RAD windows and create a "Form" window.


But we want to create this window from a blank window: you will see all the steps required to program such window.

► To create an edit window:

1. If needed, open the "WD Application With Data" project you've created in the previous lessons.
2. Create a new blank window. Click  among the quick access buttons. In the wheel that is displayed, click "Window". Choose "Blank" and validate the wizard.
3. In the description window, specify:
  - the name of the window: "WIN\_PersonX".
  - the title and description of the window: "Managing the persons".
4. Validate the description window.

5. Display the "Analysis" pane if necessary: on the "Home" pane, in the "Environment" group, expand "Panels" and select "Analysis". The different data files described in the "WD Application With Data" analysis appear in the pane.
6. Click the "+" icon found on the left of the "PERSON" file: the file items are listed.
7. With the mouse, select all the items displayed in the pane and "Drag and Drop" these items to the window that was just created.
8. Different controls are automatically created in the window. These controls are automatically linked to the corresponding item in the data file. To check this, display the "Link" tab found in the description of one of the controls ("Description" from the popup menu).




9. Close the description.
  10. Save the window.
- Run the test of the window (  among the quick access buttons). The window is displayed with empty controls.

## Managing the addition of a record

We are now going to manage the addition of a record in our window. At the moment, the window contains controls linked to the data file but we want to enter information in these controls and we want to save this information in the "Person" data file.

Two buttons will be added into our window: an "Add" button that will be used to save the data and a "Close" button that will be used to exit from the window without saving.

### Creating the add button

- To create the add button:
  1. Create a new button: on the "Creation" pane, in the "Usual controls" group, click .
  2. The caption of this button is "Add" and its name is "BTN\_Add".
- To enter the code of the button:
  1. Display the "BTN\_Add" button (for example, select the button and use the F2 key).
  2. Enter the following code:

```
ScreenToFile ()
HAdd (PERSON)
```


Let's take a look at this code:


- **ScreenToFile** is used to initialize the items with the values of the linked controls. This function is equivalent to the following code lines:

```
Person.Name = EDT_Name
Person.FirstName = EDT_FirstName
Person.Address = EDT_Address
Person.DateOfBirth = EDT_DateOfBirth
...
```

Our window uses 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 the assignments!

- **HAdd** adds the record into the data file. This function takes the values in memory and writes the content of the file items into the data file itself. The indexes are automatically updated. In our case, the "Person" data file is updated.

3. Save your window (  among the quick access buttons).

- Run the test of the window (  among the quick access buttons). Fill the controls and click the "Add" button.

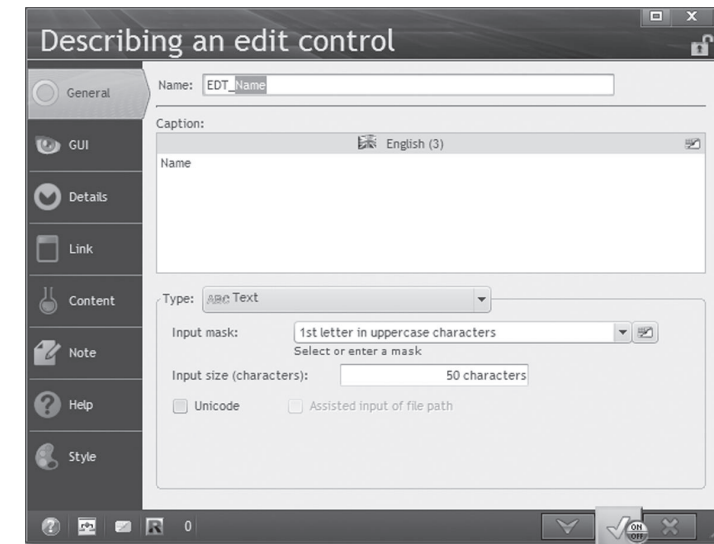
Several notes:

- No mask is used. You have the ability to enter uppercase and lowercase characters in the different controls.
- The controls are not reset after the addition.
- No closing button: the cross found at the top of the window must be used.

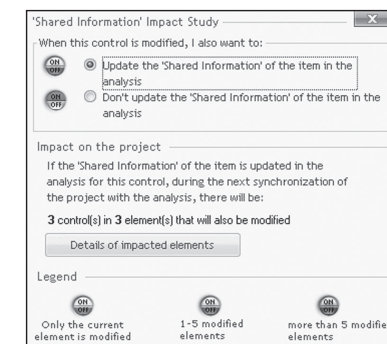
We are going to perform the necessary modifications.

## Adding an input mask

- First of all, the input mask. We're going to add an input mask to the "Name" control. Select "1st letter in uppercase characters" in the control description.



Once the mask has been modified, the style of the validation button found in the description window changes. An "On/Off" button is displayed. The following screen is displayed when you click this button:



This screen is used to manage the modification of the shared information in the analysis (the information regarding the control associated with the "Name" item). If the shared information is modified in the analysis, this modification will be applied to all the controls linked to the item when the analysis is regenerated. Validate the control description window.

- Define the same mask for the FirstName control.



- Similarly, define an input mask for the "EDT\_DateOfBirth" control. The format of this mask is "MM/DD/YYYY". The format of the returned value is "YYYYMMDD".



Note

For the Date or Time edit controls, the input mask is used to define the format for entering the information and the returned value is used to indicate the format that will be used in programming. This format will be used to store the value in the data file for example.

## Erasing the data after the addition

- We are now going to manage the erasing of the data in the controls, once the "Add" button has been used. Two new WLanguage functions will allow you to perform this operation on all the window controls.

1. Display the code of the "Add" button (F2 on the button for example).
2. Complete the code as follows:

```
ScreenToFile()
HAdd(PERSON)
Reset()
HReset(PERSON)
```

**Reset** resets all the controls for the next input. This is also used to specify to the user that the record was added.

**HReset** resets all the file variables (avoids storing the previous buffer when items are partially added into a record).

3. Save the window.

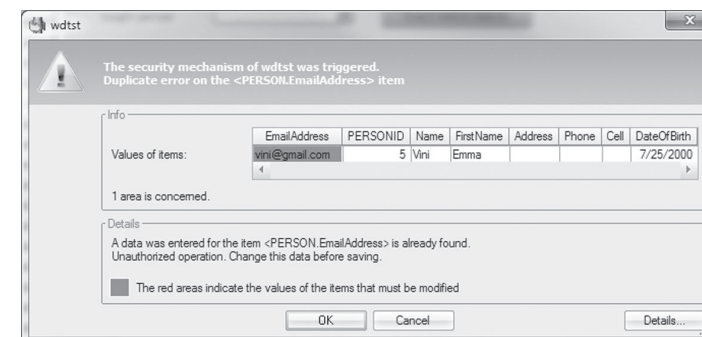
- To add a closing button:

1. Create a "Close" type button: on the "Creation" pane, in the "Usual controls" group, expand "Button" and select a "Close" button.
2. Save the window.

- Run the test of the window (among the quick access buttons). Fill the controls and click the "Add" button. Several controls can now be entered one after another. Some examples:

| Name   | First name | Date of birth | Email            |
|--------|------------|---------------|------------------|
| Morgan | Frank      | 11/03/1945    | morgan@gmail.com |
| Taste  | Flora      | 07/19/1970    | test@yahoo.com   |
| Vini   | Hans       | 12/01/1965    | vini@gmail.com   |
| Vini   | Emma       | 07/25/2000    | vini@gmail.com   |

- A special window is displayed when the last record is validated:



This window signals that a duplicate was found: the email address (that is a unique key) is identical for two persons. This window is used to modify the value of the email address: enter "vini2@gmail.com" for example.

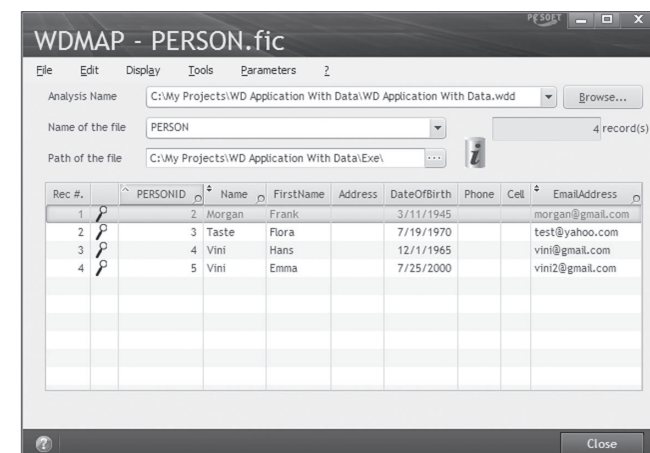
This window is one of the windows for automatic management of HyperFileSQL errors. We will see later in this tutorial how to manage by programming the different types of errors that may occur on the databases.

## Viewing the records

Several records have been added to our window. How can I view their content? WinDev comes with a tool used to view the content of the data files while developing the application (when the viewing windows have not been created yet). This tool is named WDMAP. We will be using it to view the content of the Person file.

- To start WDMAP:

1. On the "Tools" pane, in the "Database" group, click "WDMAP".
2. Select the "Person" file. The content of the data file is displayed.



## Managing the search and the modification

We've managed the addition of records into the PERSON file. It's fine. It would be even better to be able to find and modify a record. That's what we are going to do now, on the same window.

This window will allow you to perform a search on the name. Two search modes will be implemented:

- an exact-match search
- a generic search.

Then, we will have the ability to modify the record found.

### Exact-match search

To perform an exact-match search, we are going to select the name of the user from a combo box. The "Find" button will be used to display the form of the corresponding person. Only one person corresponds to the selected name.

In this first case, the search control is a combo box.

► To create the search control:

1. Create a combo box control: on the "Creation" pane, in the "Usual controls" group, click "Combo box".
2. The wizard for creating combo boxes is displayed. We are going to create a combo box based on the "PERSON" data file. Select "Display the data found in a file or in an existing query".



Go to the next wizard screen.

3. Select the "PERSON" data file. Go to the next screen.
4. We are going to display the last name and the first name of the person.
  - Clear the "PERSONID" item.
  - Select the "Name" item.
 Go to the next screen.

5. The "Name" item will also be used to sort the list. Select the "Name" item. Go to the next screen.
6. The item returned by the combo box will be the "PERSONID" identifier. This is the value that will be sought in the PERSON file. Select the "PERSONID" item and go to the next screen.
7. The combo box will be linked to no item. Keep the "No" option and go to the next screen.
8. Validate the next screen with the yellow arrow.
9. Specify the name and caption of the combo box, "COMBO\_PERSON" and "Sought person" for example.
10. Validate. Position the combo box in the window (top left corner for example).

► To create the search button:

1. Create a button: on the "Creation" pane, in the "Usual controls" group, click .
2. Position the button beside the combo box that was just created.
3. Specify the name and caption of this control ("BTN\_ExactMatch" and "Exact-match search").
4. Enter the following code:

```
HReadSeekFirst(PERSON, PersonID, COMBO_Person)
IF HFound(PERSON) = True THEN
 FileToScreen()
END
```

**HReadSeekFirst** is used to perform an exact-match search. In this example, the search is performed on the PERSON file and on the PersonID item. The sought value corresponds to the last parameter of the function. In this case, the sought value corresponds to the value selected in the combo box. This value is obtained by using the name of the combo box (COMBO\_Person).

**HFound** is used to check the result of the search. 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 using **FileToScreen**.



Note

**FileToScreen** performs the reverse operation of **ScreenToFile**: the data found in the file items is displayed in the corresponding controls.

5. Save the window.


► Run the test of the window. Enter a value in the combo box and click the search button. The result is instant.

## 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 are going to create an edit control that will be used to enter the sought name and a button that will be used to perform this search.

► To create the search control:

1. Create an edit control: on the "Creation" pane, in the "Usual controls" group, click .
2. Enter the name and caption of this control ("EDT\_Sought\_Name" and "Sought name").

► To create the search button:

1. Create a button: on the "Creation" pane, in the "Usual controls" group, click .
2. Position the button beside the edit control that was just created.
3. The name of this control is "BTN\_Generic" and its caption is "Generic search".
4. Enter the following code:

```
HReadSeek(PERSON, Name, EDT_Sought_Name)
IF HFound(PERSON)=True THEN
 Reset()
 FileToScreen()
ELSE
 Error("No person matches")
END
```

**HReadSeek** is used to perform a generic search. In this example, the search is performed on the PERSON file and on the "Name" item. The sought value corresponds to the value entered in the EDT\_Sought\_Name control. This value is obtained by using the name of the control.



**Note** An exact-match search can be performed by **HReadSeek**: to do so, use the **hIdentical** constant.

5. Save the window.

► Run the test of the window. Enter a value in the edit control and click the search button. The result is instant. However, if several records correspond to the search, only the first one is displayed.

## Modifying the form displayed

When the result of the search is displayed, it may be interesting to modify the information displayed. Modifying the values of the edit controls is child's play but they must also be taken into account in the data file. To do so, we are going to create a modification button.

► To create the modification button:

1. Create a "Modify" type button: on the "Creation" pane, in the "Usual controls" group, click "Button".
2. Position the button below the "Add" button.
3. The name of this control is "BTN\_Modify" and its caption is "Modify".
4. Enter the following code:

```
ScreenToFile()
HModify(PERSON)
ListDisplay(COMBO_PERSON, taCurrentSelection)
```

In this code:

- **HModify** is used to modify the current record with the data found in the screen.
- **ListDisplay** is used to update the search combo box (if the name is modified for example).



**Note**

When modifying a record, integrity errors, duplicate errors, ... may occur. The mechanism for the automatic management of errors is enabled by default (as already seen during the addition).

5. Save the window and run the window's test.

## Browsing the forms

We are now going to add buttons used to browse the different records.

► Create four buttons named "BTN\_First", "BTN\_Previous", "BTN\_Next" and "BTN\_Last". The code of these buttons will be:

```
// BTN_First button: call the first one
HReadFirst(Person)
IF HOut(Person) = True THEN
 Info("No form to view")
ELSE
 FileToScreen()
END
```

```
// BTN_Previous button: call the previous one
HReadPrevious(Person)
IF HOut(Person) = True THEN
 Info("Reached beginning of file")
ELSE
 FileToScreen()
END
```

```
// BTN_Next button: call the next one
HReadNext(Person)
IF HOut(Person) = True THEN
 Info("Reached end of file")
ELSE
 FileToScreen()
END
```

```
// BTN_Last button: call the last one
HReadLast(Person)
IF HOut(Person) = True THEN
 Info("No form to view")
ELSE
 FileToScreen()
END
```

In this code:

- **HReadFirst** is used to read the first record of the data file, according to the key used for the last search.
- **HReadLast** is based on the same principle, but this function reads the record with the greatest key value.
- **HReadNext** reads the record whose key value is immediately greater than the one of the current record.
- **HReadPrevious** reads the record whose key value is immediately less than the one of the 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.

► Save the window and run its test. Click each one of the buttons to browse the data file.

## LESSON 2.5. TABLE BASED ON A QUERY

### This lesson will teach you the following concepts...

- Creating a query with parameters
- Creating a window with a table based on the query
- Creating an automatic report on the table



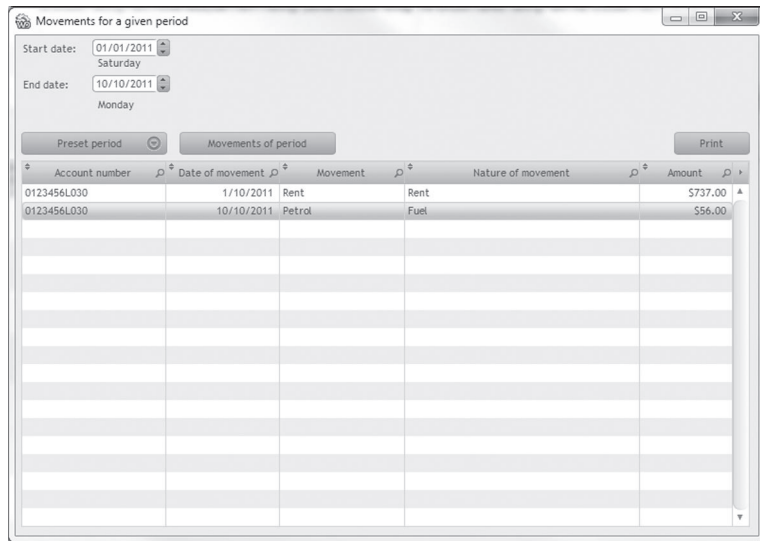
Estimated time: 20 min

## Overview

In this lesson, you will learn how to handle tables. In our example, we are going to create a window containing a table used to list the movements performed during a given period.

The table is based on a query that will be created. The query is used to select the records displayed in the table.

The window is as follows:



Note

This lesson "gives an overview" of query creation. The first part of this tutorial discuss managing queries ("Creating a Select query", page 313).

## Creating a query with parameters

► The query editor will be used to create the query.

1. Click among the quick access buttons. Click "Query" in the wheel that is displayed. The wizard for query creation starts.

2. Select "Select".

Indeed, this query will be used to select records. Go to the next screen.

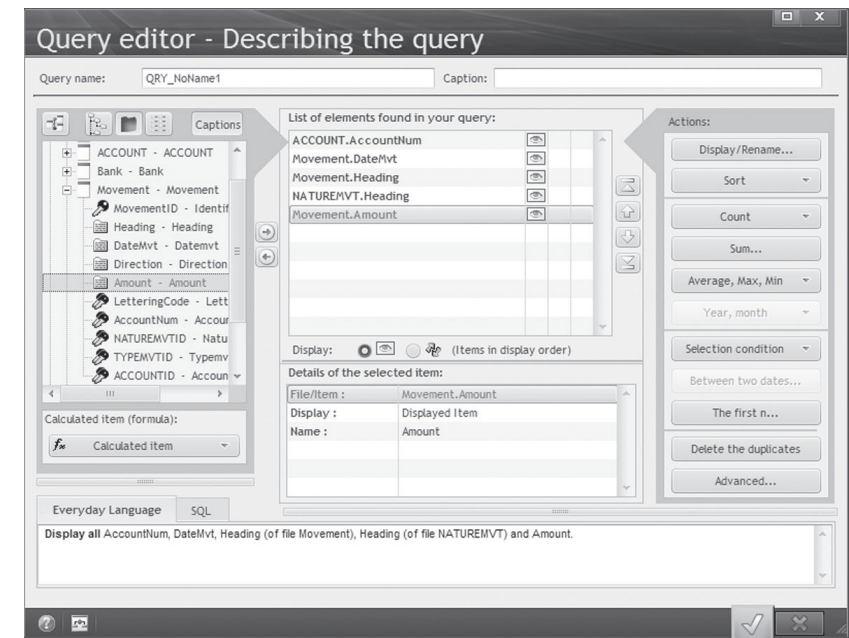
3. The description window of the query is displayed. To build the query, we are going to select the elements that will be displayed in the result.

4. Double-click the items found in the analysis on the left of the description window. The items taken into account are displayed in the center of the screen.

We want to display:

- the account number (AccountNum item of ACCOUNT file),
- the date of the movement that was performed (DateMvt item of MOVEMENT file),
- its heading (Heading item of MOVEMENT file),
- the nature of the movement (Heading item of NATUREMVT file)
- its amount (Amount item of MOVEMENT file).

The description window of the query is as follows:

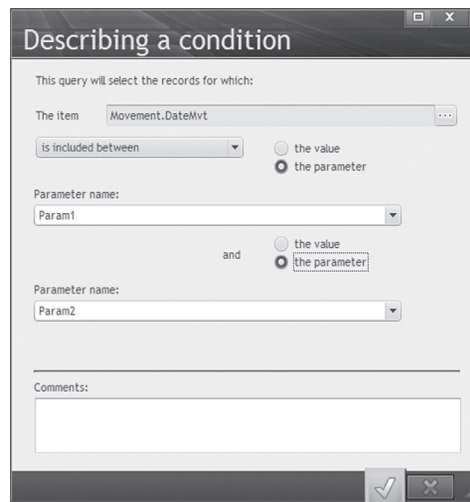


5. The data will be sorted by date. Select the "MOVEMENT.DateMvt" item, then click the "Sort" button and select "Sort on the selected item". Specify an ascending sort on the item and validate.

6. We are now going to specify that the date corresponds to a period entered by the user. To do so, select the "MOVEMENT.DateMvt" item and click the "Between two dates" button.



7. The two dates will be given by the user: specify that the value is included between two parameters and validate.



8. Our query is over. Give a name to the query (QRY\_Movement for example, top left of the screen). Validate the description window of the query.

9. Let's now run the test of the query: click . The window that opens asks for the parameters of the query.

- Select the "Param1" parameter in the table and enter its value in the edit control: 03/03/2013 for example.
- Select the "Param2" parameter in the table and enter its value: 03/04/2013 for example.


10. Validate. The result is automatically displayed.

We are now going to create the window containing the table associated with this query.

## Creating a window with a table based on the query

This window will allow the user to enter the interval of dates for which the list of movements performed must be displayed.

► To create the window with the table control:

1. Create a new window: click  among the quick access buttons, then in the wheel that comes up, click "Window". In the wizard, select "blank" and validate.
2. Give a name and a title to your window: WIN\_Movements and "Movements for a given period".
3. Create a table control: on the "Creation" pane, in the "Data" group, expand "Table and list box" and select "Table (Vertical)". The wizard for table creation starts.
4. We are going to create a table based on the query that was just created. Select "Display the data found in a file or in an existing query". Go to the next wizard screen.
5. Select the query that will be used as data source for the table: QRY\_Movement. Go to the next wizard screen.

6. Select all the proposed items. Go to the next wizard screen.

7. Select the search item: "DateMvt". Go to the next wizard screen.

8. Enter the different parameters of the table: keep the options proposed by default and uncheck "Generate the initialization code of the query parameters".



Caution!

The "Generate the initialization code of the query parameters" option is used to generate the initialization code of the Table control. This option must be used with great care because the generated code uses the default values specified during the test of the query.

Go to the next wizard screen

9. Select the orientation of the table: Vertical. Go to the next wizard screen.

10. Give a name and a title to the table ("TABLE\_Movements" and "Movements performed during the given period" for example). Validate.

11. The table is automatically created. It can be resized or the window can be resized.

12. Save the window.

► To create the controls used to enter the interval of dates, we will be using a supercontrol supplied with WinDev.

1. Display the "Wizards, examples and components" pane if necessary: on the "Home" pane, in the "Environment" group, expand "Panels" and select "Wizards, Examples and Components".

2. Select "Controls".

3. Select "Supercontrol".

4. Select the supercontrol named "Entering a period" and "Drag/Drop" the supercontrol toward the window: position the supercontrol above the table for example.

► We are now going to enter the code that will be used to initialize the table. This code will be entered in a specific button.

1. Create a new button. The name of this button is "BTN\_Movement" and its caption is "Movements of period".

2. Display the code of the button ("Code" from the popup menu of the button).

3. Enter the following code in the click code of the button:

```
QRY_Movement.Param1 = SC_SelectPeriod.EDT_StartDate
QRY_Movement.Param2 = SC_SelectPeriod.EDT_EndDate
HExecuteQuery(QRY_Movement)
TableDisplay(TABLE_Movements, taCurrentFirst)
```

This code is used to initialize the parameters of the QRY\_Movement query with the values entered in the Date controls.


The query is run, then the table is displayed.

► We must now save the window and run its test.

## Printing the content of the table

The window displays the movements performed between two dates. Why not create a report to print this information? But how do I create a report?

Nothing is simpler with WinDev! The data is displayed in a table? The table automatically offers a popup menu containing a "Print" option.

- To automatically print the content of the table:
  1. Run the test of the window (GO icon).
  2. Enter the requested period to display the movements. Click the "Movements of period" button.
  3. When the data is displayed in the table, display the popup menu of the table (  in the top right corner or right mouse click on the table). Click "Print". The following window is displayed:



4. Select "Print directly". Choose (if necessary) to print the content of the table in "Landscape" mode. The report corresponding to the table is displayed in a preview window.



Note

In test mode, you have the ability to print the content of the table or to create a report based on table. This option automatically creates the corresponding report in the report editor.

At run time, the end user will be able to print directly or to start "Reports and Queries" to create the corresponding report. See "Distributing "Reports & Queries" with your applications", page 369 for more details about the Reports and Queries program.

5. Close the preview window and stop the test of the application.

Can you imagine anything easier? In fact, you've been using a WinDev application's AAF. These three letters stand for Automatic Application Features. Indeed, a WinDev application contains by default a set of automatic features: no additional development is required. WinDev helps you simplify your development tasks.



Note

The entire list of AAF (Automatic Application Features) is available in the WinDev AAF 18.PDF file. This file is automatically included in your application if you choose the "Help for the AAF" option when creating the automatic menu.

You want to customize the report proposed by default? All you have to do is create an automatic report on the table. This report (if it exists) will be automatically used by the option for printing the table.

## Creating an automatic report on the table

To create a report used to print the data found in a table, all you have to do is create a "Report on Table button" and the report corresponding to the table is automatically created.

- To add a "Report on table" button:
  1. On the "Creation" pane, in the "Data" group, expand "Table and list box" and select "Report on table".
  2. The report is automatically created.
  3. A window proposes to add the report to the project. Validate.
  4. Go back to the "WIN\_Movements" window (click the "WIN\_Movements" button found in the bar containing the opened elements for example).
  5. The "Print" button was positioned in the top left corner of the window. Move it above the table for example.
  6. Save the window and run its test. Use the print button or select "Print" from the popup menu of the table: the report that was just created is used in both cases.

## LESSON 2.6. REPORT WITH EMBEDDED QUERY

### This lesson will teach you the following concepts...

- Creating a report with an embedded query.
- Modifying a report with an embedded query.
- Testing a report with an embedded query.



Estimated time: 20 min

### Overview

A report was created in the previous lesson. In this lesson, we are going to create a report based on a query. However, instead of creating the query then the report, we are going to create the query during the report description. In this case, the query is embedded in the report: it cannot be used elsewhere in the project.

The report that will be created is quite simple: we want to display the operations performed for each account of each bank.

05/25/2009

Operations per bank account

Organization : BNP Paribas

Account number : 0123456789

|                 | Nature of movement | Date of movement | Amount   | Direction |
|-----------------|--------------------|------------------|----------|-----------|
| Fuel            |                    | 10/10/2007       | -26.00   | Debit     |
| Rent            |                    | 10/01/2007       | -737.28  | Debit     |
| Cash withdrawal |                    | 09/21/2007       | -30.00   | Debit     |
| Miscellaneous   |                    | 10/02/2007       | 2,350.52 | Credit    |
|                 |                    |                  | 1,557.24 |           |

Number of Rows: 4

Powered by WinDev




Note

This lesson "gives an overview" of report creation. A full section of this tutorial is dedicated to managing report ("Your first report", page 335).

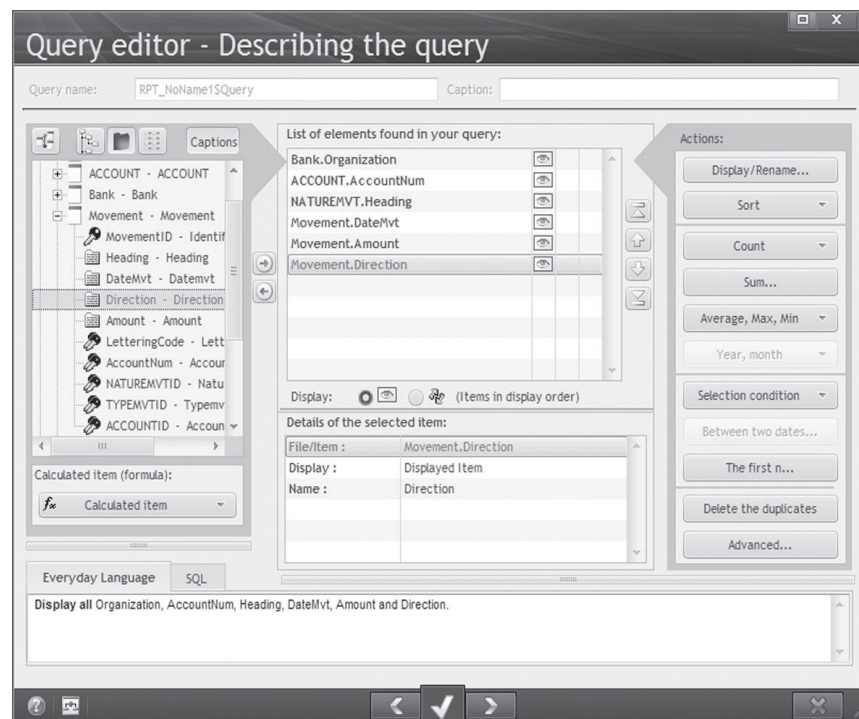
### Creating a report

► To create a report:

1. Click  among the quick access buttons. Click "Report" in the wheel that is displayed. The wizard for report creation starts.
2. Select a "Table" report. Go to the next wizard screen.
3. The report will be based on a query that we are going to create: select "From a new query". Go to the next wizard screen.

4. The description window of the query is displayed. We are going to define the items that will belong to the query:

- The "Organization" item of BANK file: the name of the bank
- The "AccountNum" item of ACCOUNT file: the account number
- The "Heading" item of NATUREMVT file: the heading of the operation
- The "DateMvt" item of MOVEMENT file: the date of the operation
- The "Amount" item of MOVEMENT file: the amount of the operation
- The "Direction" item of MOVEMENT file: the direction of the operation.



5. The data will be sorted:

- select the Organization item and define an ascending sort ("Sort" button).
- select the AccountNum item and define an ascending sort ("Sort" button)

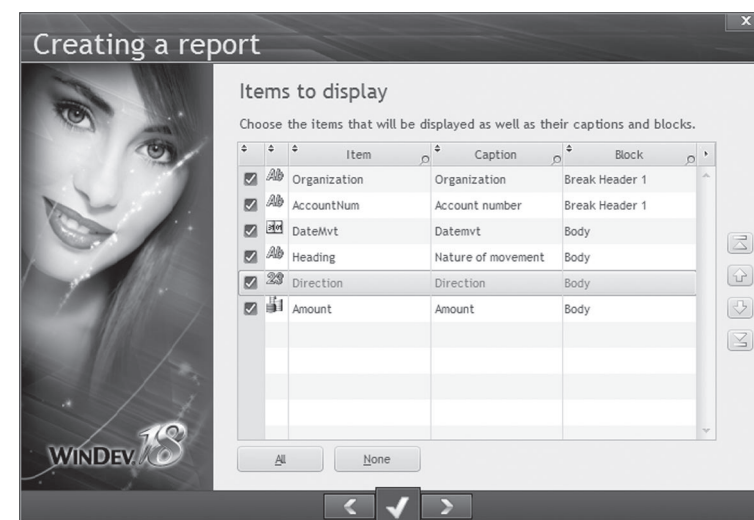
6. Validate the query. The report creation wizard continues.

7. The report will contain a break (check "Yes"). The breaks are used to group the data. In the report, the data will be grouped by bank organization. Go to the next screen.



8. The wizard automatically proposes the **Organization** item and the **AccountNum** item as break items. Those are the items for which a sort was defined in the query. Select only the **Organization** item and go to the next screen.

9. **This screen is very important.** Indeed, it enables you to associate the different items associated with the report with the different sections of the report. In this example, the organization and the account number will be displayed in the break header. The order of the items will also be changed: the date, the heading, the direction and the amount will be displayed in the "body" block of the report (the main block). Use the arrows beside the table to arrange the items.



Go to the next screen.

10. The next screen proposes to calculate a sum on the Amount item for each end of break. Accept and go to the next screen.

11. For the paper format, choose the landscape mode. Go to the next screen.

12. Choose a skin template for your report (ActivUbuntu for example).

13. Give a name and a title to the report ("RPT\_Operation" and "Operations per bank account" for example).

14. Validate.

15. The report is automatically created.

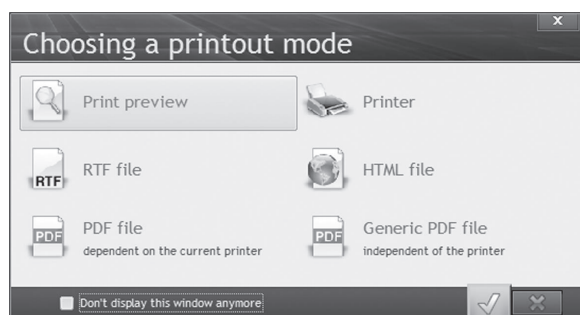
16. Save the report.

## Testing a report

► To run test of a report:

1. Run the test (among the quick access buttons).

2. Choose the "Print preview" mode:



3. The report is displayed in a preview window.

► Let's take a look at this report, you will notice that it is incorrect! Indeed, the sum of calculated amounts does not take the movement direction into account. We are now going to modify the report.

1. Edit the code of the BODY block (position the mouse cursor in the BODY block of the report and select "Block code" from the popup menu).

2. In the "Before printing" code of the block, add the following code:

```
// Display negative amounts in case of debit
IF ITEM_Direction = 1 THEN
 ITEM_Amount = ITEM_Amount*-1
END
```

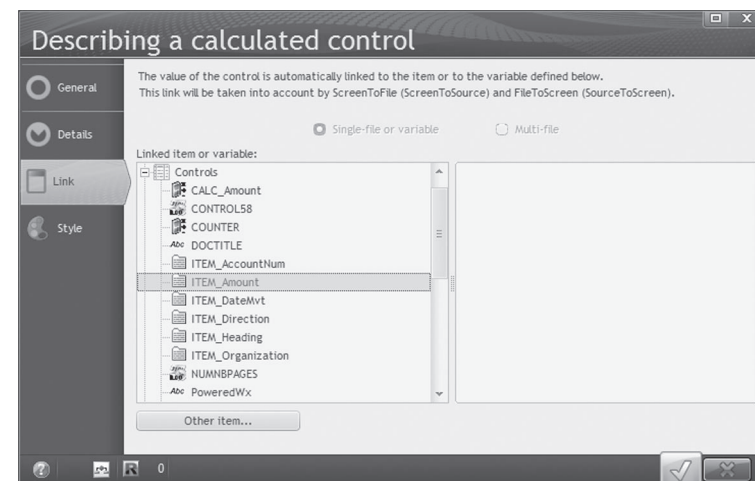
This code allows you to take negative amounts into account.

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

4. Edit the description of the "CALC\_Amount" control. This control corresponds to the amount found in the BREAK FOOTER block. This control displays the total amount. By default, it is linked to the "Amount" item of the query used by the report.

5. Modify the method for calculating the total: the total must be calculated on the "ITEM\_Amount" control.

6. Display the "Link" tab in the description window of the control and select the control of the "ITEM\_Amount" report:



7. Validate.

8. Save the report and run its test.



## LESSON 2.7. USER GROUPWARE

### This lesson will teach you the following concepts...

- What is the user groupware?
- Including the user groupware
- Configuring the user groupware
- Running the test of the user groupware



Estimated time: 20 min

### 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: when implementing an application for sales management, the application 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 salespeople can see the price list, place the orders and manage the new customers.
- the sales directors have access to all the options.

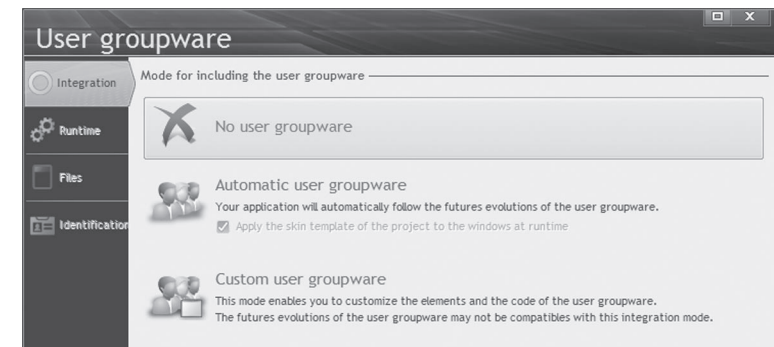
WinDev enables you to easily manage these access levels via the user groupware.

We're going to include the user groupware in our "WD Application with Data" application then configure it.

### Including the user groupware

► To include the user groupware in the "WD Application With Data" 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 modes are available:

- **Automatic user groupware:** all the elements of the user groupware are automatically included in your application. This way you benefit from all the updates to the user groupware. In addition, you can also apply the project skin template to the groupware windows.
- **Custom user groupware:** all the elements of the user groupware are included in your application via an internal component. You can customize the various elements of the user groupware. However updates are not taken into account.

3. Select the "Automatic user groupware" option.
4. In the "Runtime" tab, two modes are available:
  - Auto run: the groupware is started as soon as the application is started.
  - Manual start: the user 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 the location of the data files for the user groupware. In our case, we will be using the HyperFileSQL Classic data files, in the location specified by default.



**Note** If you (or the end user) uses **Windows Vista** (or a more recent operating system), we recommend that you use the directory of data.

7. Select "Enable the history of connections". This option allows the supervisor to get information about the connected users.
8. Select the "Identification" tab. This tab is used to define the user identification mode:
  - managed by the user groupware
  - use a LDAP directory. In this case, when installing the application, users will be able to enter the parameters of their LDAP directory.
  - use Windows authentication.
9. Validate. A message indicates that by default a Supervisor user is created.



**Note** A single user exists by default, the supervisor. When starting the application for the first time, connect by using the name: SUPERVISOR. You'll then be able to define the password used by the supervisor.

10. Validate this message. The user groupware is included in the application.

► Let's now run the test of our application:

1. Run the test of the project ( ). A login window is displayed.
2. Login as supervisor.
3. A new menu is displayed, allowing you to run the test of the application or to configure the application.
4. Choose "Configuring the groupware". We are going to configure the user groupware.

## Configuring the user groupware

Configuring the groupware consists in defining the different users of the application as well as their rights on the different windows and controls of the application.

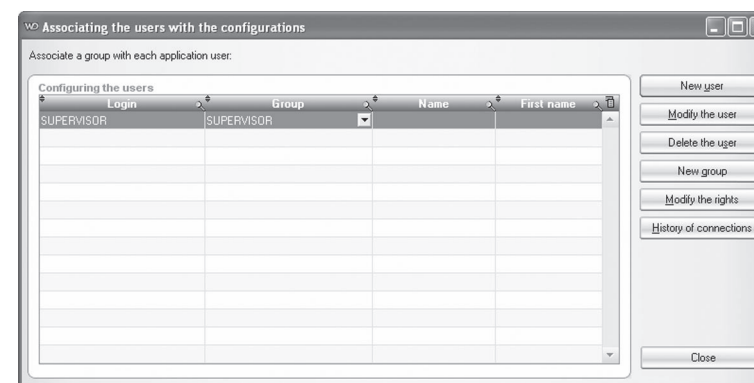


**Note**

The configuration of the 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.

► To configure the user groupware:



1. Create the users who will be able to access the application ("New" button of the "User" area). Only the supervisor exists by default.  
Create the new TEST user.  
Click "Password" and specify the password: "test".  
Click "Save".
2. Associate (if necessary) the users with a group. The "new" button of the "groups" area lets you create groups.

3. Configure the rights of the users ("Right management" option on the left of the window). Select the "TEST" user and click "Next". The window that opens is used to select each window of the application. For each window element, you can define whether the control will have the behavior of the application (default) or whether it will be disabled, invisible or grayed.

4. Close the configuration window.

5. If you choose to start the application, the application starts normally.

6. Close the application and go back to the editor.

- **The user groupware will no longer be used in the rest of this tutorial.** It can be disabled. To do so, on the "Project" pane, in the "Project" group, click "User Groupware". In the window that comes up, in the "Integration" tab, select "No user groupware". Validate.



Note

**If automatic tests are run 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 "Execution" tab of the user groupware configuration window.

## LESSON 2.8. THE DASHBOARD

**This lesson will teach you the following concepts...**

- What is the dashboard?
- Automatic tests
- Optimizing the queries




Estimated time: 20 min

## Overview

The project dashboard is a main element for managing the WinDev projects. The project dashboard gives an overall view of the progress status of a project.

The dashboard includes several progress bars, lights and counters that give an overall view of the status of a project.

In this section, we will present the management of automatic tests and the optimization of queries. We'll look into the dashboard in more details in "Dashboard", page 417.

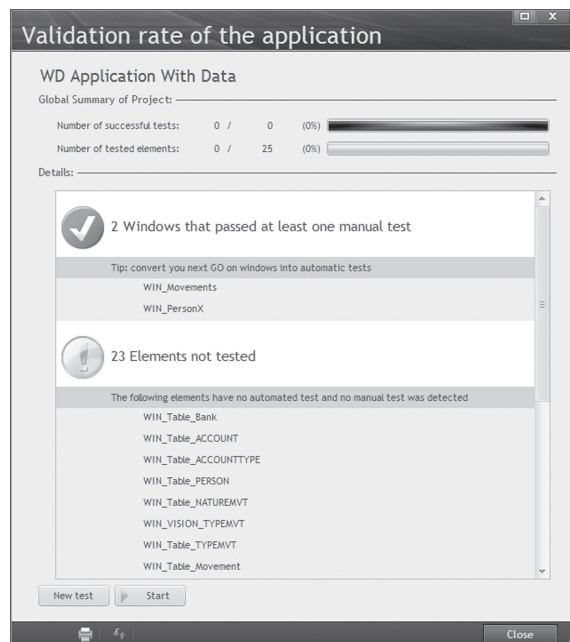
- To display the project dashboard (if not already done), on the "Project" pane, in the "Project" group, click .

## Automatic tests

One of the most interesting features of the dashboard is to give information about the tests that were run on the application.

Several tests have already been run while developing our application.

Click the Tests light. A window displays the validation rate of the application.



This window returns the list of all the tests that have been run on the application:

- Manual tests (the tests that have already been run)
- Automatic tests.

The automated tests are a category of specific tests. The automated 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 automated 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 automated test on the edit window that was created at the beginning of this lesson.



Note

The automated tests can be run on windows, procedures, classes.

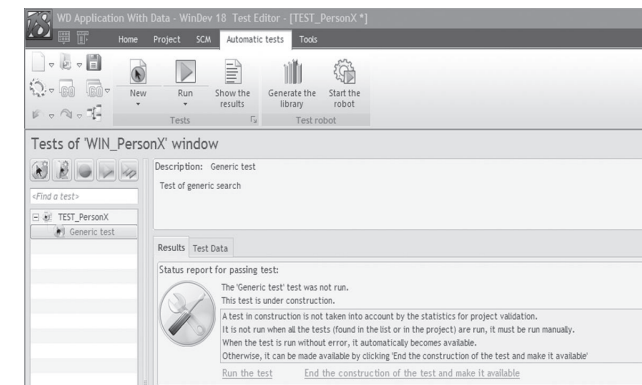
- To create an automated test on a window, all you have to do is run the test of the window:
  1. Open the "WIN\_PersonX" window that was previously created.
  2. On the "Automatic tests" pane, in the "Tests" group, expand "New" and select "Save a new test".
  3. In the window that is opened, click "Start recording".
  4. The test of the window is run.
  5. Enter a name in the "Sought name" area and click the "Generic search" button.
  6. Click the "Close" button.
  7. The test is over.
  8. The test editor proposes to save a description for the test. The test is named "Generic test" and its description corresponds to "Test of generic search". Validate.
  9. The code of the test in WLanguage is displayed in the code editor. Close this code window.



Note

The WLanguage functions used to perform automated tests are the EmulateXXX functions.

- 10. The test editor is displayed:



The test is being built.

11. We are going to make the test available (click the link named "End the construction of the test and make it available"). The test can now be run at any time. The test editor indicates that the test has never been run before. Click the "Run test" link.

12. The test is automatically run and the test editor displays the result (the test is successfully run in our case).

13. Close the test editor and record the test if necessary.

► We are now going to modify our window and run the test again. The modification affects the "EDT\_Sought\_Name" search control. A check code will be added into the "Whenever modified" code.

1. Add the following code to the "Whenever modified" code of the EDT\_Sought\_Name control:

```
IF Length(EDT_Sought_Name) >= 2 THEN
 BTN_GENERIC..State = Grayed
ELSE
 BTN_GENERIC..State = Active
END
```

2. Save the window.

3. Run the test associated with the window: on the "Automatic tests" pane, expand "Run" and select the "Start tests" option.

4. The test fails.

5. Modify the code of the search control as follows:

```
IF Length(EDT_Sought_Name) <= 2 THEN
```

6. Re-run the test. The test is successfully run.

Another feature of the test editor is the ability to use a set of tests: this is the notion of iterations. In our example, we have run the test regarding the search for a person. We are going to modify our test to search for the different persons of our file. This test is not a real test but it is used to check the feature regarding the iterations.

1. Display the test editor (double-click "TEST\_PersonX" in the project explorer for example).

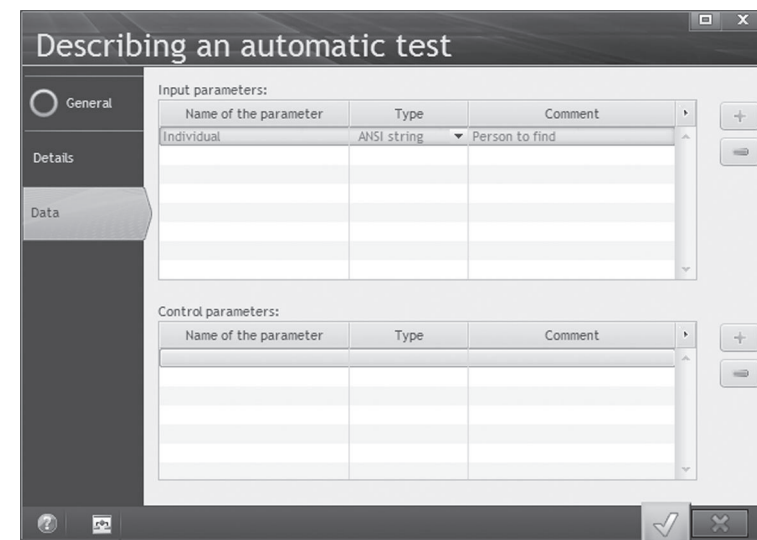
2. Display the test description ("Description" from the popup menu of "Generic test").

3. The "Data" tab is used to enter the parameters that must be taken into account in the test.

In our example, we will only have the sought name in input parameter:

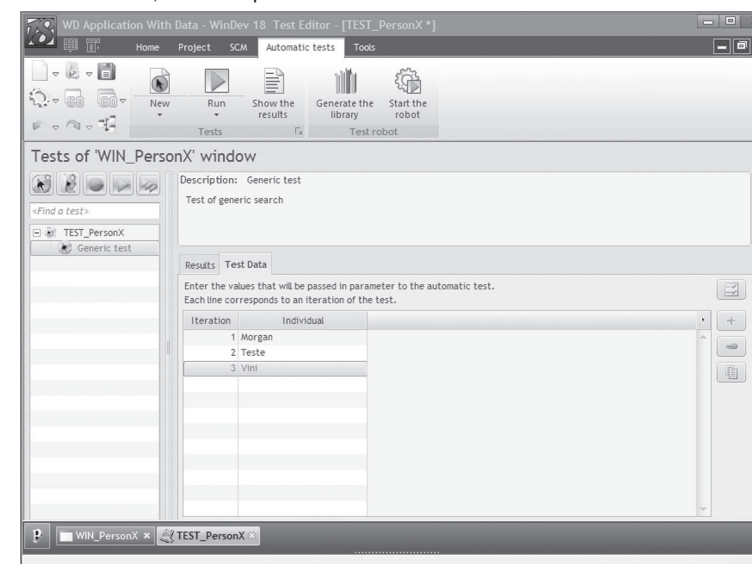
- Click the "+" button found on the right of the "Input parameters" table.
- Replace "Parameter1" by "Individual".
- The parameter type is "Ansi String" (select this type in the list).

- In the comments, specify "Person to find".



Validate the description window.

4. In the test editor, click the "Test data" tab. We are going to enter the names of the persons found in our data file, for example:

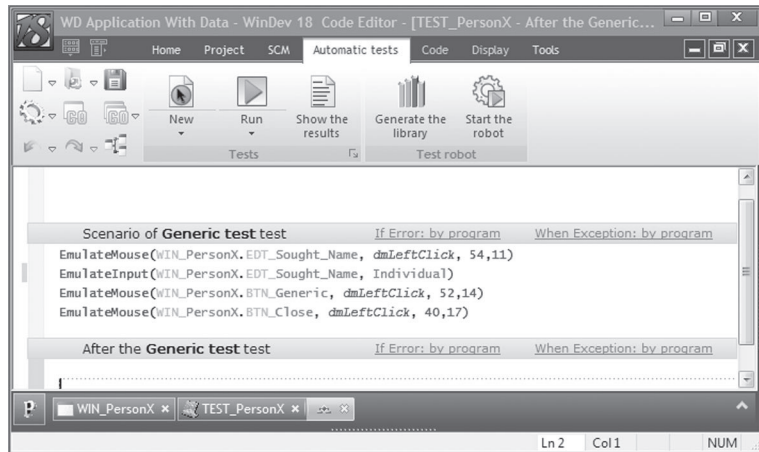


To do so:

- Click the "+" button found on the right of the table.
- Enter Morgan in the "Individual" column.
- Click the "+" button found on the right of the table.



- Enter Test in the "Individual" column.
  - Click the "+" button found on the right of the table.
  - Enter Vini in the "Individual" column
5. We are now going to edit the code of the test to take the parameter into account. Select "Code" from the popup menu of the test named "Generic Test". The name used during the test must be replaced by the "Individual" variable



#### 6. Run the test.


The test editor proposes several features that will not be presented in this tutorial:

- the definition of the 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 instance.

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

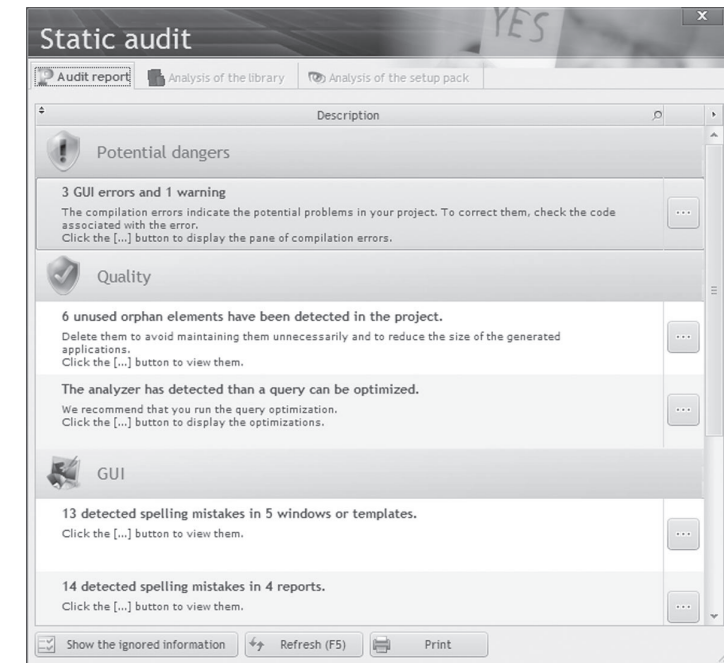
- Close the editor of automatic tests.

## Static audit and query optimization

Let's go back to our application's dashboard: on the "Project" pane, in the "Project" group, click .

We're going to run a static audit of our application. The static audit is used to quickly detect all the improvements that can be performed: correct misspellings, optimization of queries, ...

- In the dashboard, click the "Static audit" button. In the wizard, select "Run the static audit on the entire project". The report window of the audit is displayed:

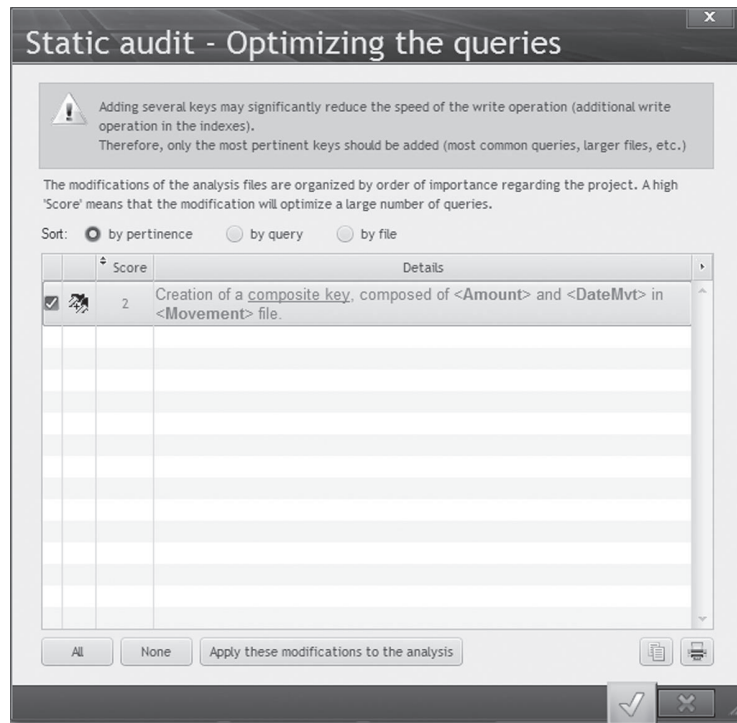


Note: The information displayed in the static audit can be configured via the button .

Let's take a look at the audit summary:

- Start the profiler (also called the performance analyzer). The performance analyzer will be presented later in this tutorial.
- Optimization of queries: the static audit has detected that one of the application queries can be optimized. We are going to perform this operation now.

- To optimize the query:
1. Click the [...] button of the corresponding line in the static audit.
  2. The optimization window is automatically displayed.



To optimize the project queries, WinDev proposes to create composite keys. These composite keys must be created in the analysis. These modifications will be automatically performed by WinDev.

3. Click the "Apply these modifications to the analysis" button and validate.
4. To take into account the modifications performed, the analysis must be regenerated then the automatic file modification must be run. Validate the windows that come up. The project is now consistent with the analysis.

## LESSON 2.9. DEPLOYING THE APPLICATION

**This lesson will teach you the following concepts...**

- Creating the executable
- Help for the new features
- Creating the setup program



Estimated time: 20 min

## Overview

Our first application is now created. 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.

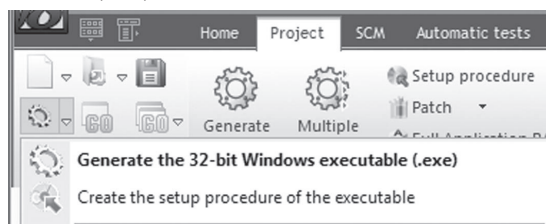
## Creating the executable

Creating the executable is child's play: a menu option and a few mouse clicks are sufficient.

We will now present in details all the steps for creating the executable. You have the ability to click the green button at any time to validate all the wizard screens.

► To create the executable:

1. In the quick access button area of the ribbon, expand the wheel and select the "Generate the 32-bit Windows executable (.exe)".



Note

WinDev also enables you to create 64-bit executables, Linux executables, services, Java applications, ...

2. The wizard for creating an executable starts. A first screen presenting the status of the automatic tests is displayed. In our case, all the automatic tests have been successfully run. Go to the next screen.

3. The next screen is used to describe the main options of the executable:

- the name of the executable: we will keep the default name.
- the name of the icon associated with the executable: you can select an icon found in the image catalog.
- the splash screen. WinDev proposes several types of splash screens. A splash screen with animated text is selected by default. The "Options" button is used to configure it.

4. Go to the next screen. For the operating mode of the executable, we will keep the default options. Go to the next screen.

5. The next screen is used to customize the error message of the application. We will keep the message proposed by default. Go to the next screen.

6. The next screen is used to define the use of UMC in our application.



Note

The UMC (User Macro Code) allows the end user to create his own procedures in WLanguage in order to modify the operating mode of a control, window, ... If the UMC is included in the application, an additional button (found in the title bar) will allow the user to enter the additional source codes.

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

We will keep the default options. Go to the next screen.

7. The next screen is used to specify whether the patches will be taken into account by the executable.



Note

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 "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 is started.

See the online help for more details.

We will keep the default options. Go to the next screen.

8. The next screen is used to manage the languages of the executable. The multilingual feature will be presented in another lesson. We will keep the default options. Go to the next screen.

9. The next screen displays all the files that will be included in the library of the executable. Those are all the project elements that can be handled by the end user. We will keep the default options. Go to the next screen.

10. The next screen is used to manage the components used by the executable. We'll see how to create and use components later in this tutorial. Go to the next screen.

11. The next screen affects the directory of the data files. If your application is going to be deployed on Windows Vista, we advise you to choose "Directory of the application data".



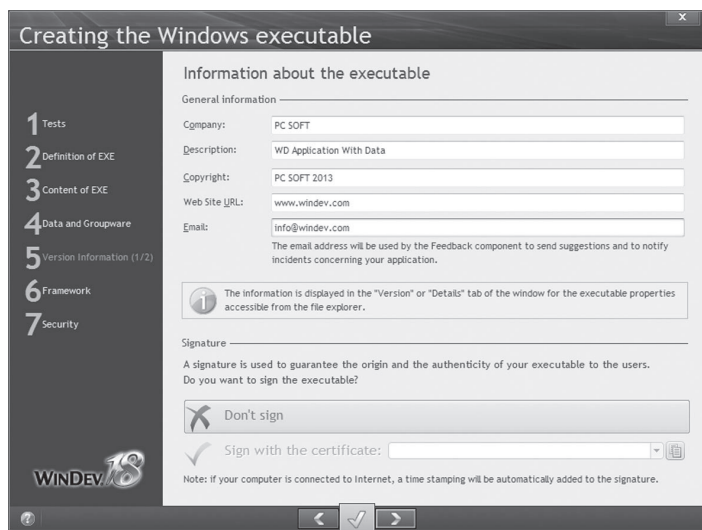
Note

In Windows Vista and later, the UAC (User Account Control) implies the use of the Windows programming standard.

See the online help for more details.

Select the option corresponding to your system. Go to the next screen.

**12.** We are now going to enter the executable information. This information is displayed in the Windows explorer by selecting the properties of the file.  
Enter the following information for instance:



This screen can also be used to sign the executable if necessary.  
Go to the next screen.

**13.** The next screen concerns the version number of the executable. You can use:

- a format compatible with the earlier WinDev versions,
- a standard Windows format. Select this option.

**14.** The next screen concerns the WinDev framework.



**Note** The framework corresponds to the different libraries required for the executable to operate.

To use the necessary libraries only, select "Use the framework specific to the application".  
To use the framework common to all the WinDev applications installed on the computer, select "Use the common WinDev framework". With this option, the framework is installed once only on the computer (it can also be downloaded via Internet) and it is used by all the WinDev applications. Check this option.

Go to the next screen.

**15.** A screen regarding Windows Vista is displayed. This screen is used to include a manifest for a use in Windows Vista.

Go to the next screen.

**16.** Don't save the project and validate the wizard.

**17.** The executable is created. It can be started immediately to check that it's working properly. To do so, click the "Run the executable" button).

That's it, the creation of the executable is completed. Several steps are required but as soon as a first configuration is performed, you have the ability to validate all the steps from the start.



**Note**

You also have the ability to click the steps specified in the wizard in order to directly go to a screen. The default options of the other screens will be automatically validated.

## Creating the setup program

Creating the setup program is child's play: a wizard helps you 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 won't go into details about its use in this lesson. See "Installing an application", page 472 for more details.

We will now present in details all the steps for creating the setup program. You have the ability to click the green button at any time to validate all the wizard screens.

► To create the setup program:

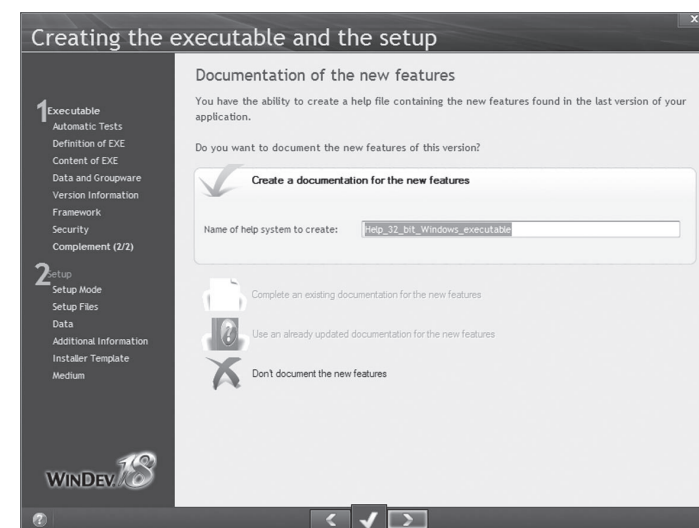
1. On the "Project" pane, in the "Generation" group, expand "Setup procedure" and select "Create the setup procedure". The wizard for creating the executable and the setup starts.



**Note**

We've already created the executable; the options chosen are stored. Directly click "Complement" of the "Executable" section in the wizard.

2. A screen proposes to save the projet. Keep the option "No: don't save the project" and go to the next screen.
3. A screen proposes to create the page for the new features.



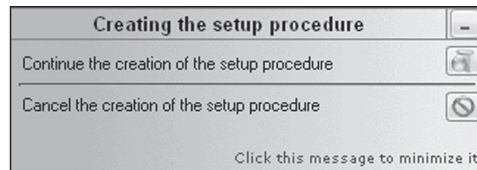
This option is used to create a help file in order to present the new features of the version to the end users. During a first setup, this file may correspond to the help for the software. Select "Create a documentation for the new features" and go to the next screen.

4. The next screen is used to define the elements that will be automatically included in the help file. Keep the default options and go to the next screen.



**Note** If the application already contains a help system, it can be used in order to include a page for the new features.

5. Validate the message. The executable is automatically created (with the options that were defined when creating the executable) as well as the help system. Specific information can now be entered in the help system. In the example, leave the information created by default. To resume the creation of the setup program, click "Continue the creation of the setup procedure".



The help is automatically compiled.

6. 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. When an update is performed, a single update is required on the server. The applications installed on the computers will be automatically updated. Go to the next screen.

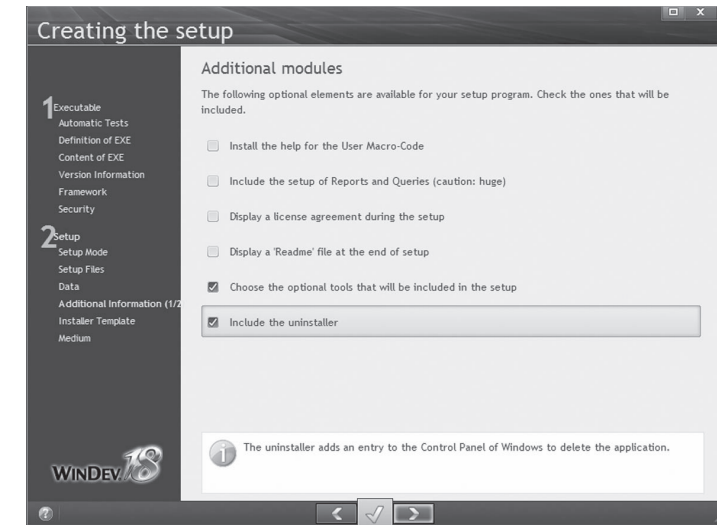
7. Choose a standard setup. Go to the next screen.

8. We will not change the default setup directory. Go to the next screen.

9. Keep the files proposed in the list of files installed. Go to the next screen.

10. Uncheck all the options in the parameters of the database. Go to the next screen.

11. Keep the following options in the additional modules:



We're going to choose the optional tools to install and we will include the uninstaller. The uninstaller will allow the users to uninstall the application from the program manager of Windows. Go to the next screen.

12. We will select WDOptimizer. This tool is used to reindex the data files of the application.

13. Click "Medium" to the left of the wizard. The setup will be generated in a single directory. By default, the setup program is created in the "Install" sub-directory of the project. A password can be specified for the setup. In this case, only the user who knows the password will be able to install the application.

14. Validate. The setup program is automatically created.

15. Run the test of the setup program.

## Conclusion

That's it, we have created an application from its first design in the data model editor to its distribution. You are now familiar with several features of WinDev. The following chapters will allow you to examine some of these features in details.



## LESSON 2.10. QUESTIONS/ANSWERS

### This lesson will teach you the following concepts...

- Tips and tricks

#### Question How do I create the main menu of my application?

Perform the following operations:

1. Create a new window or open an existing window.
2. On the "Window" pane, in the "Bars and menus" group, expand "Main menu" and select "Add the main menu". An option is automatically created.
3. Right-click this option.
4. Click "Option description" to modify the selected option.
5. Click "Add after" to add a new option after.
6. Click "Add before" to insert a option before.
7. Click "Insert a sub-menu" to add a sub-menu to the tree structure of the menu.

#### Question How do I link a window to an option of my main menu?

**Open** is used to associate a window with a menu option. Enter the following code in the click code of your menu option:

```
Open (WIN_MyWindowName)
```



Notes

To associate a report with a menu option, use **iPrintReport**:  
`iPrintReport (RPT_MyReportName)`

#### Question How do I automatically insert the '?' menu?

To automatically insert a "?" menu in your application, on the "Window" pane, in the "Bars and Menus" group, expand "Main menu" and select "Add the '?' menu". In the wizard that comes up, select the options you want.

#### Question How do I create a popup menu?

A popup menu can be added:

- at window level.
- at control level.

For a window:

1. Right-click the window and select "Description".
2. Click the "GUI" tab and click  beside the "Popup Menu" combo box.

For a control:

1. Right-click the control and select "Description".
2. Click the "GUI" tab and click  beside the "Popup Menu" combo box.

To find out or modify the popup menu of a control or window by programming, use the property named **..PopupMenu**.

### Question How do I automatically close a window after a preset duration?

To do so, use a button and **DelayBeforeClosing**. For example:

```
DelayBeforeClosing("WINDOW", "BUTTON", 200)
```

See the online help (keyword: "Close, Automatic closing (button)") for more details.

You also have the ability to use the "Automatic validation" option found in the "Details" tab of the window description.

### Question How do I retrieve the parameters passed by command line to an executable?

To do so, use **CommandLine** in the initialization code of the project.

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



Tip

Passing parameters by command line to your project can be simulated in test mode. In the editor:

1. On the "Project" pane, in the "Test mode" group, expand "Test mode" and select "Configure the test mode".
2. Enter the parameters of the command line.

### Question How do I uninstall an application developed with WinDev?

When creating the setup program, you can allow the users to uninstall the application.

The uninstall program is automatically created if this option is chosen. Your application was registered toward Windows so that it can be uninstalled later.

To uninstall an application:

1. Click the "Start" menu.
2. Select "Control panel".
3. Select "Add/Remove programs".
4. Select the application and click "Uninstall".

### Question How do I create a setup via CD-ROM?

When creating the setup program ("Setup procedure" option in the "Generation" group on the "Project" pane), you have the ability to choose various setup delivery methods.

When selecting "Support with autorun", WinDev will create a folder in which all the necessary files for a setup via CD will be created (AUTORUN file, setup files, ...).

Then, all you have to do is burn the content of this folder on a blank CD and distribute it!

### Question How do I install an application?

Once your executable has been created, your project's directory contains a "EXE\<Project name>" sub-directory, which holds all the elements needed for your application to run.

To prepare a setup for your application:

1. On the "Project" pane, in the "Generation" group, expand "Setup procedure" and select "Create the setup procedure". The setup creation wizard starts.
2. Follow the instructions given on the screen.

### Question How do I associate an icon with my executable?

The icon that is associated with your executable can be defined when creating the executable. This icon must be in ICO format.



Notes

A catalog of preset icons is supplied with WinDev. This catalog is accessible when selecting the icon.

### Question How do I associate a splash screen with my executable?

The splash screen associated with your executable can be defined when creating the executable. The format of this image must be recognized by WinDev (BMP, WMF, GIF, JPEG, TIFF, ...)

A catalog of preset images is supplied with WinDev. This catalog is accessible when selecting the image.

WinDev gives you the ability to customize this image when creating the executable. This enables you to write the text of your choice (with the requested formatting) on this image.

**Question    How do I display the icon of my application in the taskbar?**

Use **SysIconAdd**. Example:

```
Res = SysIconAdd("C:\Icons\Icon.ICO", "PopupMenu", ...
 "Rollover message", "Name Associated Procedure")
```

To restore your application, use **SysIconDelete**.

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

**Question    How do I install a shortcut for an application on the desktop?**

**CreateShortcut** is used to create the shortcut of an application by programming. This function is used to create a shortcut on the desktop or in a program group.

For example, to install a shortcut on the desktop:

```
CreateShortcut(shortcutDesktop, "Caption of my icon", ...
 "C:\Program Files\Accounting\Accounting.EXE")
```

See the online help (keyword: "Create, A shortcut") for more details.

**Question    How do I share the WinDev framework among the applications installed on the same computer?**

When creating the setup program of the application, you have the ability to specify whether your application shares the WinDev framework with the other applications installed on the computer. In this case, the WinDev Framework will be installed in the "C:\Program Files\Common Files\PC SOFT\18.0\Framework" directory.

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

**Question    How do I detect the elements not used by my application?**

After months or years of development and maintenance, the directory of your project often contains several files that are not used anymore but that you don't dare delete.

Test files and windows, useless images, ... It's time to clean up!

A WinDev tool is used to automatically detect the unused elements and to delete them from the project. The elements deleted from the project will be archived (in ZIP format or in a backup directory) so that they can be restored later if necessary...

- ▶ To use this wizard, on the "Project" pane, in the "Other actions" group, expand "Save/Restore" and select "Clean the project directory".

**Note:** You can find out:

- the project's dead code: on the "Project" pane, in the "Audit and performances" group, expand "Edition audit" and select "Dead code".
- a project's orphan elements: on the "Project" pane, in the "Audit and performances" group, expand "Edition audit" and select "Orphan elements".

PART 3

Windows and  
controls

18



DEVELOP 10 TIMES FASTER 

## LESSON 3.1. WINDOWS

**This lesson will teach you the following concepts...**

---

- How do I create a window?
- The characteristics of a window



Estimated time: 10 min



## How do I create a window?

We've created several windows using various methods since starting this tutorial:

- creation of blank windows based on a skin template (part 1 of this tutorial).
- creation of different types of windows (form, table, ...) based on data files (part 2 of this tutorial).

Other methods can also be used to create windows. These methods are grouped in the window creation wizard.

**Reminder:** To start the window creation wizard:

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



2. A window shaped like a wheel is displayed. This window is used to create all the elements that can be associated with a project.
3. Click "Window". The wizard for window creation starts.

The wizard for window creation allows you to:

- create **RAD windows** (Rapid Application Development): these windows are based on the data files described in the analysis and they contain the code required for them to operate. These windows are associated with a RAD pattern (that defines the features included in the window and in the interface) and with a skin template (that defines the appearance of the window). These windows can be used straightaway. Several types of windows are proposed: form window, window with table, window with loop, ...
- create **RID windows** (Rapid Interface Development): these windows are based on the data files described in the analysis. They contain the controls and the buttons only as well as the code required by the elements included by the associated RID pattern. The corresponding code must be entered by the developer. These windows are linked to a RID pattern and to a skin template if necessary. These windows can be used straightaway. Several types of windows are proposed: form window, window with table, ...
- create standard windows: This tab is used to create standard blank windows.
- create **internal windows**. The internal windows are a specific type of window. Their operating mode will be presented later in this section.
- create **windows based on a window template**. The window templates are used to define a set of criteria (graphic, control, code) that must be re-used in each window of the application. The creation and the use of a window template will be presented later in this section.
- **import an existing window** (non-WinDev window). This option is used to "copy" the interface of a window into your WinDev application. All you have to do is select the requested window and WinDev does it all for you. The "Generate images for the unrecognized controls" option is used to make the interface closer to the interface of the imported window.

You still have the ability to create a blank window, without control, that uses (or not) a skin template.

Let's now present the main characteristics of a window in details.

## Description of a window: 8 tabs are available

Several windows have been created since the beginning of this tutorial. The first element displayed was the description window in order to enter the title of the window, its name and its description. The description window includes several useful settings, saving you from writing several code lines.

Some of these features will be presented in a simple example.

### Practical example

To handle the different tabs found in the windows, we will be using a sample project created for this purpose. To open this project in WinDev:


1. Close (if necessary) the current project to display the home window.
2. In the home window, click "Tutorial" and select the project named "Windows and controls".  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Windows and controls".
3. Open the "WIN\_EditControl" window (found in the "Standard controls" folder of the project explorer).



Notes

#### Opening a window in the editor

Several methods can be used to open a window in the editor. Summary of the different methods:

- Press [CTRL] + [E] and select the window to open. The advantage of this method is the preview of the window.
- In the "Project explorer" pane, double-click the name of the window to open.
-  among the quick access buttons and selection of the file corresponding to the window to open.

4. Display the window description ("Description" from the popup menu of the window).

### "General" tab

The first tab of the description window is the "General" tab. This tab is used to specify the general parameters of the window, for example:

- **the logical name of the window:** this name will be used in programming to handle the window.
- **the description of the window:** this description provides information to the developer. It gives additional information about the purpose of the window. It is also used to describe the window when printing the documentation.
- **the title of the window:** as already seen, the title is displayed in the window and it is used to provide information to the user.

## "GUI" tab

The second tab of the description window is the "GUI" tab. This tab is used to define the parameters regarding the interface of the window. This tab groups the characteristics regarding:

- the size and position of the window
- the popup menu of the window,
- the rollover cursor and the action of the right click performed in the window, ...

Several options are interesting in this tab:

- **Move by the background:** This option allows the user to move the window by clicking anywhere in the window (and not only in the title bar). This option can be checked with the description window.
- **Resizable:** this option allows the user to resize the window. However, this option requires a specific management of the resize operation performed on the controls (called anchoring). We shall devote a whole chapter to the ergonomics of the windows later in this tutorial.
- **Store the size and position of the window:** the window position defined by the user will be automatically saved and re-used the next time the window is opened. When the user is working on several screens, the display of the window on such or such a screen will be automatically stored.
- **Dim the window when it becomes inaccessible:** another very popular option. If dialog boxes are displayed by the application, the user immediately knows where he can click: the window found in the background is automatically grayed so that the user can focus on the message displayed on the screen.
- **Adapt the size according to the content:** this option is used to automatically adapt the size of the window according to the controls found in this window. The empty areas automatically disappear.

## "Details" tab

This tab contains the parameters regarding the operating mode of the window. It groups the characteristics regarding:

- The type of the window (this characteristic will be presented later in another lesson),
- The advanced parameters of the window (HyperFileSQL context, ...),
- The animation of the window when it is opened and/or closed,
- Automatic execution of a window button.

Let's see these two last points in details.

**The window animation** is used to give a dynamic style to your application. You have the ability to configure the opening and/or the closing of your window.

To configure the animation? Nothing's easier, all you have to do is click the animation's characteristics. The setting window is displayed. Enter your parameters and run their test (even on the setting window).

The **automatic closing of the window or the automatic validation of a button** allows you not to leave your application locked on a window. Indeed, a user may display a Customer form and leave it open while he is away. To avoid locking the other users, you can define a duration and the action that will be performed after this duration (click the Validate button, click the Cancel button, ...).

## "Image" tab

This tab is used to define the images that will be used by the window:

- The background image of the window and its display mode.  
You can use an image that defines the shape of your window (to create a window that looks like a remote control for example). In this case, select "Clip the window" and follow the instructions. You can also use an image that will be resized when your window is resized. To do so, define the associated "9-image mode".
- The image for the sizing handle of the window.
- The icon associated with the window: this icon is visible in the top left corner of the window (near the title bar). It customizes the window. The icon associated with the project will be visible if no icon is specified.

## "Language" tab

This tab is used to specify the languages that will be supported in the window (for the "multilingual" projects).

## "Note" tab

This tab is used to enter a text. This text is printed in the documentation. You can, for example, enter information about the operating mode of the window, the rules for managing the processes.

## "Help" tab

This tab is used to specify the name of the context-sensitive help file associated with the window. This help file is used when pressing the help button of the window to enable a context-sensitive help.

## "Style" tab

This tab groups the parameters regarding the style of the window. You can select:

- the skin template
- the icons displayed (maximize, minimize, help, system menu, ...)
- the type of border
- the background color
- the XP theme
- the opacity (used to define the transparency of a window)
- the presence of the status bar, ...

## LESSON 3.2. CONTROLS

### This lesson will teach you the following concepts...

---

- The different types of controls
- The standard controls
- The specialized controls



Estimated time: 45 min

### Introduction

---

WinDev proposes more than 40 types of controls for communicating with the end user. They are used to enter or view values.

The displayed values can come from a calculation performed by programming, from a file found in a database or from an assignment.

The values entered can be used to perform calculations, they can be saved in a file found in a database or assigned to other controls.

This lesson will allow you to discover the different types of controls, to run their test and to program them.

To simplify this lesson, the different controls have been divided into two categories:

- the standard controls: these controls are the most frequently used.
- the special controls, used to display a special interface or to manage special features.

### Practical example

To learn how to use this various types of controls, we're going to use the "WD Windows and controls" example built for this purpose. To open this project in WinDev:

1. Close (if necessary) the current project to display the home window.
2. In the home window, click "Tutorial" and select the project named "Windows and controls".

Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Windows and controls".

## The standard controls

The "Standard controls" are the controls that are frequently used in the applications created in WinDev. These controls are as follows:

- Static control
- Edit control
- Button
- Image
- Click area
- Radio button
- Check box
- List box
- Combo box
- ListView
- Table
- Looper
- Treeview table
- TreeView
- Progress bar
- Chart

These controls will be presented one by one. For each type of control, a specific window is available allowing you to test the main functionality of the control. These pages are grouped in the "Standard controls" folder in the project explorer.

### Type of control: Static

#### Summary

The static controls are used to display a static text. No input is allowed at run time. They can be modified by programming (like any other control).

The static controls are used to display information, a title in large characters for example. The content of a static control can be defined in the editor when describing the static control or by programming.

The static controls can also be used to display a text according to the language used in the project without additional programming.

#### Practical example

- Open the "WIN\_StaticControl.WDW" window and run the test of this window.  
This window presents:
  - the various types of static controls that can be used.
  - the various operations that can be performed on a static control.

- Check the different operations for the static controls.
  - The static control is displayed in green.
  - The "Color" button changes the color of the static control.
  - The "Retrieve" button retrieves the static control.
  - The "Fill" and "Build" buttons modify the static control.
- Go back to the editor.

## The different types of static controls

WinDev proposes several types of static controls.

In the "General" tab of the control description, you have the ability to select the preset types:

- **Multi-lingual static control:** this type of static control lets you manage the displayed static controls over several lines.
- **RTF static controls:** this type of static control is used to manage all the characteristics of the RTF format (bold, italic, underlined, colors, ...).  
Note: In order for a static control to be in RTF format, you can also check "Text with formatting (RTF)" in the "Details" tab of the control description.
- **Numeric static control:** this type of static control is used to display numeric values. You can specify the input mask and the display mask used by the static control.
- **Currency static control:** this type of static control is used to display a currency value. You can specify the input mask and the display mask used by the static control. You have the ability to display the currency for example.
- **Date, Time, Duration static controls:** these types of static controls are used to display a date, time, duration in the format specified by the input/display mask. In this case, you can also specify the returned value (used if the control is handled by programming for example).

You also have the ability to define specific static controls:

- **animated static controls** Make a static control blink or scroll is child's play. To do so, choose the animation mode and configure the corresponding parameters in the "Details" tab of the control description.
- **static controls with soft shadow:** a soft shadow is displayed behind the static control to create a raised effect. The configuration of the soft shade is performed in the "Style" tab of the control description.

## Handling the static controls by programming

We will study the code of the buttons used to modify the Static control.

To build the static control named "STC\_StaticHandled" ("Build" button), you must use the following syntax (in this example, the string is built from strings and from the result of **Today**, which returns today's date).

```
STC_StaticHandled = "Hello" + CR + "Today is " + ...
 DateToString(Today(), maskSystemDate)
```

To retrieve the caption of "STC\_StaticHandled" ("Retrieve" button), the syntax is:

```
// Retrieve a string
Str is String
Str = "The caption is " + CR + STC_StaticHandled

// Other method: display the information in a dialog box
// Info("The caption is " + CR + STC_StaticHandled)
```

You can even change the color of the text displayed in the static control ("Color" button) with the **..Color** property.

```
STC_StaticHandled..Color = PastelRed
```

**PastelRed** is a WLanguage constant corresponding to the Pastel Red color.



Notes

The properties enable you to modify some characteristics of the controls by programming: color, caption, font, ... We will not present all the properties. We will discover some of the main properties as we progress through this lesson. All the properties are relative to a given control. Their syntax is as follows:  
`Control_Name..Property_Name`  
 See the online help (keyword: "Properties, Properties of the window controls") for more details.

## Type of control: Edit control

### Summary

The edit controls are the first controls that you have been using. They are used to enter data or to display data coming from variables, calculations or data files.

They can be used to enter a password, a quantity to order, the name of a customer, an address, ... In most cases, the edit controls are necessarily found in the created windows. Now let's take a look at their operating mode.

Regardless of the information to enter or to display, you have the ability to choose the corresponding type:

- text, RTF, HTML, password,
- numeric,
- time,
- date,
- duration,
- currency,
- currency+Euro.

Several input masks (or display masks if the control is read-only) correspond to each type of control.

We have already seen why masks are useful when developing the application named "My accounts".



Reminder

The numeric edit controls have no defined type. The input mask selected for the control will define whether the control is a real, a double real, an integer, ...

### Practical example

- ▶ Open the "WIN\_EditControl.WDW" window in the editor. This window presents the management of the text and numeric edit controls.
- ▶ Run the test of the window.
- ▶ Enter values in the "Editable" controls. No input can be performed in the other controls. They have been defined as read-only. Click the different buttons and see what happens.

### All the details

A simple assignment is all it takes to display and retrieve a value in an edit control, regardless of the type of the edit control.

#### Some initialization examples:

- Initialization code of a text edit control:

```
EDT_ReadOnly = "Hello" // Displays Hello in the text control
```

- Initialization code of a numeric edit control:

```
// Displays the value 20.6 in the numeric control
EDT_Numeric = 20.6
```



Reminder

You also have the ability to initialize the value of an edit control by using the "Content" tab of the control description.

#### Some examples of retrieval processes:

- Click code of a button to retrieve the value of a text edit control:

```
Str is String
Str = EDT Editable // Retrieves the content of the text control
```

- Click code of a button to retrieve the value of a numeric edit control:

```
Value is Real
// Retrieves the content of the numeric control
Value = EDT_Numeric Editable
```

### Managing the color in the edit controls

In the input area, the text color and the background color can be modified by programming with **..Color** and **..BrushColor**.

```
// Changes the color of the text
EDT_Oper..Color = LightRed
// Changes the background color
EDT_Oper..BrushColor = LightYellow
```



## Characteristics of an edit control

A description window is available for the controls. This description window is used to configure several options of the control (style or operating mode).

We are now going to describe the characteristics of the edit control, keeping in mind that several options are available in the description of other controls. We won't go into details about all the available options in this tutorial.

### "General" tab

As already seen, the "General" tab of the edit control is used to define:

- the name of the control
- its caption
- its type
- its input mask.

Some types can be very useful:

- **RTF**: This type of edit control is used to write words in italic, bold, color and with different sizes and fonts, ... To do so, copy/paste a content in RTF format ("Rich Text Format") into the control. A formatting toolbar can be associated with the RTF edit controls.
- **HTML**: This type of edit control is used to display a text in HTML format and to enter a text in HTML format. A formatting toolbar can be associated with the HTML edit controls.
- **Password**: If this type of control is selected, the characters typed in the controls are replaced by star characters on the screen.



**Tip** This type of control is ideal for entering a user password in a login page.

- **Multiline text**: this type allows you to enter text over several lines. CR characters (Carriage Return) can be inserted into the text.



**Tip** This type of control is perfect for entering comments or notes. Using this parameter is essential for the "Memo" controls (presented later).

The "Unicode" option lets you enter a Unicode format string in the edit control. This option is useful for applications that use non-latin character sets.

### "GUI" tab

The GUI tab is used to define the characteristics of the control interface.

Let's take a look at the following points:

- Initial state
- Management of TAB key
- Eye magnet
- Management of the shortcut and right click

Anchor management is detailed in the "Anchoring", page 233 paragraph.

initial status of the control corresponds to the status of the control when running the window. This characteristic is available for nearly all the types of controls.

When running a window, an edit control can be:

- **In edit mode**: the control is active, it'll be in edit mode in the window. The user will be able to enter information in this control.
- **Display only**: the control is disabled, no input is possible. This control will behave like a static control. However, the content of the control can be selected with the mouse if "With selection in read-only" is checked in the "Details" tab.
- **Grayed**: the element is disabled and it appears grayed out. This status is used to specify to the user that the control exists but that it cannot be currently accessed.
- **Visible**: As its name indicates, the control "exists" in the window. This control can be visible or not. This status can be very useful in programming when a control must be displayed or not according to conditions!  
This option is independent from the 3 previous options. This enables you to make the control visible or not while keeping the display parameters.

The status of the control can be modified later by programming with:

- the property named **..State** for the "Editable", "Read-only" and "Grayed" options,
- the property named **..Visible** for the "Visible" option.

The **management of the TAB key** is used to configure how the TAB key will be taken into account in the window. In most cases, the TAB key allows the user to move between the different controls of the window ("Control accessible by TAB"). This enables you to define the tab order in the window (this topic will be presented later in this part).

However, WinDev allows you not to manage the TAB key or to accept TAB in edit. This last option may be interesting in the multiline controls used to enter long texts.

The **"eye magnet"** (also called conditional formatting) consists in modifying the visual aspect of a control in order to attract the attention of the user to this control. WinDev allows you to automatically modify this visual aspect according to a set of conditions. For example, a control for which the input is mandatory will be displayed on a red background as long as its content remains empty.

The **"shortcut"** gives direct access to the control by pressing a combination of keys on the keyboard ([ALT] + [Letter] for instance).

In the caption of an edit control, any letter preceded by the "&" character automatically becomes the "shortcut".

For example: "&Customer name" describes [Alt] [N] as the shortcut.

The **"right click"** process is used to associate an action with the right mouse click. The process can be, for instance, "Validate the input", "Display the help", ... The process is chosen among the processes of buttons (text and graphic) found in the window.



**Notes**

Sometimes, it may be easier to use the shortcuts rather than the mouse. WinDev enables you to easily create interfaces (GUI) where the mouse is not required. For example, for point of sales applications (not enough room next to the cash register!) or in industrial settings: interactive kiosks, machine-tool, ...

**"Details" tab**

The tab is used to define the different features that apply to the edit control.

Some important points:

- For a multiline input, you have the ability to select the mode used to scroll the information in the control ("Details" tab of the description window).  
If the control is set as **"Horizontal AND Vertical Scroll"**, the move to the next line is performed by [CTRL] + [ENTER] (Windows standard) or by [ENTER].



Notes

**Did you know it?** The maximum size of a text variable is 2 GB. Assuming that you type 3 letters per second, it would take, ... over 22 years to fill the control!

You can also display and enable the links (URL for example) found in the multi-line text.

- For the numeric controls, you have the ability to enter a **lower bound and an upper bound** (in the "Details" tab of the description window of the control). During the input in this control, WinDev will automatically check whether these values are respected. You can inform the user about the values to enter by displaying the bounds in the tooltip or in an indication text.
- **Mandatory input:** if the mandatory input is defined for this control, the move to another control will not be allowed as long as no value is entered.
- **Assisted input:** if the control is linked to a key item, the value found in the data file and corresponding to the first three letters typed is automatically proposed,
- **Check the spelling:** if Open Office and its dictionaries are installed on the user computer, the spelling checker will be automatically proposed to the user.
- **Store the value:** used to store the last value entered in the control before closing the window. The value will be automatically proposed during next opening of the window.
- **History of inputs:** used to propose the last values entered by the user for this control.
- **Return NULL if empty:** returns the value NULL if the control has no value (Caution! Not 0, nor an empty string!).



Tip

The **NULL** value is very useful to pass parameters to a query and to make the conditions optional for multi-criteria searches. This feature will be presented later in the lesson about the query editor.

- **Ellipse mode:** Used to truncate the content of the control if necessary. If the control is truncated "at the end", the "..." characters will replace the end of the text.

**Type of control: Button****Summary**

The buttons are used to start processes. The process associated with the button will be run when the button is enabled (when the button is clicked by the user).

The buttons can contain a simple text and/or an image.

You've already created several buttons. Let's go back to the types of buttons proposed by WinDev.

**Practical example**

- Open the "WIN\_ButtonControl.WDW" window in the editor. This window presents the different types of buttons. We will study all the available types of buttons.

**The different types of buttons**

A button can have 5 different types (this information must be specified in the description window, "GUI" tab):

1. Normal button
2. Validation button
3. Interruption button
4. Cancel button
5. Help button

**NORMAL button**

A **Normal** type button is a button like the ones you've already created. This type of button has no specific preset action. The exit code of the current control when validating the button is run before the process of the button.

- To check this, click the "Normal" button. The message "Exit code of NAME control" is displayed first then the message "Click on Normal" button" is displayed.



Tip

The **NORMAL** buttons must not be used in the windows containing controls. The **NORMAL** buttons can be used in the toolbars, in the icon bars or in the windows without edit controls.

**VALIDATION button**

When a button is a **Validation** button, pressing the [CR] key or the [Enter] key runs the process of the button, and the exit code of the current control is run.

- To check this, go back to the "Name" control and click the "Validation" button.



Tip

A VALIDATION button must be used to validate an input in a window.  
A VALIDATION button must be found in the window as soon as an input can be performed in an edit control. This is the type used for the "OK" buttons.

**INTERRUPTION button**

When a button is an **Interruption** button, the exit code of the current control is not run when the button is enabled. The code of the button is run, then the cursor goes back to the control.

- To check this, go back to the "Name" control and click the "Interruption" button. The code of the button is run but the exit code of the control is not run.



Tip

An INTERRUPTION button will be used to call a window (to interrupt the current input without validating) then to resume the input.  
**This is the type used for the "Help" buttons.**

**CANCEL button**

When a button is a **Cancel** button, pressing the [ESC] key runs the process of the button, and the exit code of the current control is not run.

- To check this, go back to the "Name" control and press the "Esc" key.



Tip

A CANCEL button will be used to "exit" from the window currently in edit without validating.  
In an edit window, a CANCEL button is required to cancel the current input without validating.  
This is the type used for the "Cancel" buttons.



Caution!

The "cross" and the Cancel button have the same effect. When the "cross" is clicked, the window is closed by running the code of the Cancel button found in the window. The closing code of the window is then called. The "cross" and [Alt]+[F4] have the same action.

**HELP button**

When a button is a **Help** button, the button validation automatically displays the help associated with the current control (in most cases enabled via [F1]) without running the exit code and the entry code of the current control.

- To check this, click the "Help" button.

| Summary                                           |                                                              |
|---------------------------------------------------|--------------------------------------------------------------|
| If your window contains no edit control           |                                                              |
| Type of button to use                             | All the buttons can be NORMAL buttons                        |
|                                                   |                                                              |
| If your window contains at least one edit control |                                                              |
| Type of button to use                             | A VALIDATION button                                          |
|                                                   | A CANCEL button                                              |
|                                                   | a HELP button (if the controls have associated help windows) |
|                                                   | Any other INTERRUPTION button                                |

**Characteristics of a button****Action associated with a button**

A button can be associated with a preset action. WinDev proposes several preset actions: open a project window, close a window, preview a report, start an Internet site, ... No code line is required to perform these actions: WinDev takes charge of everything.

The actions can be associated with the buttons in the "General" tab of the description window.

**Shortcut of a button**

A button can be associated with a shortcut. This shortcut may correspond to a function key, an arrow key or a key combination.

The shortcut associated with the button is specified in the information about the GUI of the control. The button must be an "Interruption" button (or a "Normal" button in a window where no input is allowed). This enables you to run a process, to open a window, ... via the action of a key.



Notes

**Automatic Application Feature (AAF):** You cannot remember a keyboard shortcut? At run time, the user has the ability to choose the keyboard shortcut associated with a button. To do so, he can select "Choose the keyboard shortcut" from the popup menu of the button.

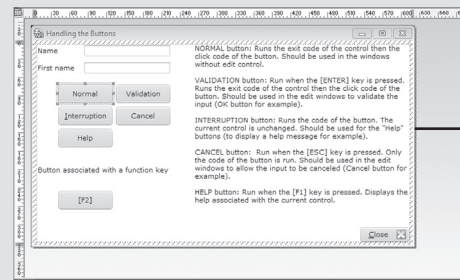
- In our window, press [F2]: an "Info" window is displayed.

In some applications, the processes are triggered by the function keys to avoid using the mouse. To do so, the buttons must be created off-screen and a shortcut must be associated with them.

To create an off-screen button, this button must be located in the host window of the window in the editor. The host window corresponds to the window where the WinDev window is displayed.



Tip



Home window

### Timer button

A button found in your window can be defined as timer button: the button will be automatically run after a specific duration. This type of button is very useful to avoid locking the records (the form of a customer at lunch time for example): to do so, the Cancel button must be defined as a timer button.



- ▶ A timer button is found in the test window. This button times out after 5 minutes. The user can modify it himself. Simply right click this button and select the "Automatic validation..." option. A window comes up allowing the user to set the time out length for the automatic validation.

### On/Off button

The On/Off buttons (also call push button) are buttons that work like an On/Off switch. The On/Off buttons go from a "normal" state to a "down" state and vice versa when a user clicks on this type of button.

### Image of a button

You have the ability to associate images with the buttons. Two types of images can be used:

- the image that illustrates the button. In this case, the button is a graphic button: 
- the image that defines the shape and background of the button: 

The image associated with the button can be used to manage:

- the states of the button (normal, pressed, grayed, focus, rollover). In this case, the image contains the different drawings associated with each status. These images are arranged horizontally.
- the animation of the button. In this case, the image of the button contains the different drawings of the animation for each status. These images are arranged vertically.

The image of the button can be:

- an image created by yourself.
- chosen among the ones proposed in the catalog of WinDev (in the "Cliparts" or in the "Graphic buttons" for example).

You can also use an image with a transparent background. In this case, you can:

- create an image whose background color is "Light magenta".
- specify that the color of the top left pixel in the image is the transparent color.



Tip

We advise you to:

- give no caption to a graphic button whose image contains a caption, otherwise the button will contain two captions.
- avoid using graphic buttons and text buttons in the same window.

The caption of a text or graphic button can be modified by programming with the property named **..Caption**. If the caption contains the "&" character, the letter that follows it will be underlined and it will become the shortcut of the button:

```
BTN_Text..Caption = "New &caption"
```

Similarly, the image of a graphic button can be modified by programming:

```
BTN_Image..Caption = "MyImage.BMP"
```

The property named **..Caption** automatically recognizes whether the caption or the image must be modified.

## Type of control: Image

### Summary

The image controls are used to display static or animated images. You have the ability to display images coming from a product catalog, personal pictures, ...

These images must have one of the following formats: BMP, JPEG, GIF, PCD, PSD, TGA, TIF, PNG, ICO (Windows icons), EMF, WMF, CUR, OS2 (BMP in OS/2), PCX, SVG.

All the modes are supported, from "256 colors" to "true colors" (16-bit, 24-bit or 32-bit).

The image controls can also be used as click areas.

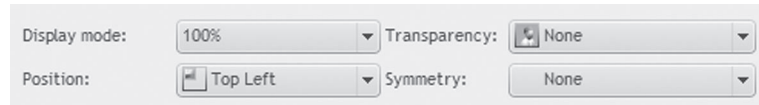
### Practical example

- ▶ Open the "WIN\_ImageControl.WDW" window in the editor. This window presents different types of images:
  - static image.
  - animated gif and animated image.
  - clickable image.
- ▶ Run the test of this window.

## The different types of images

### Static images

The static images are the most common images. To use a static image in a WinDev application, all you have to do is create an image control and associate this control with an image file. This image can be selected in one of the directories of your disk or it can be chosen from the image catalog. WinDev allows you to configure the display mode of the image ("General" tab of the description window):



| Display mode        | Example | Description                                                                                                                                                                             |
|---------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 100 %               |         | The image is displayed with its initial size. The size of the control can be modified but it does not affect the image.                                                                 |
| 100% Centered       |         | The image is centered in the image control. The size of the control can be modified. If the image is smaller (in width or in height) than the control, bars are displayed on the sides. |
| Stretched           |         | The image is stretched to occupy the entire surface of the control. The size of the image is adapted to the size of the control.                                                        |
| Tiled               |         | The image, in its initial size, is repeated as many times as possible to occupy the entire surface of the image control.                                                                |
| Homothetic          |         | The image is proportionally enlarged so that it is entirely displayed in the image control. The image is aligned to the top and to the left.                                            |
| Homothetic extended |         | The image is proportionally enlarged so that the smallest side of the image is entirely displayed in the image control.                                                                 |
| Homothetic centered |         | The image is stretched or reduced (according to the case) while keeping the width/height ratio to occupy the greatest possible surface of the control. The image is not distorted.      |

|                                         |  |                                                                                                                                                                     |
|-----------------------------------------|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Homothetic centered extended            |  | The image is proportionally enlarged so that the smallest side of the image is entirely displayed in the image control. The image is centered in the image control. |
| Homothetic without enlargement          |  | The image can be reduced if necessary but it will not be enlarged above its initial size if the control is too large.                                               |
| Homothetic centered without enlargement |  | The image is centered. The image can be reduced if necessary but it will not be enlarged above its initial size if the control is too large.                        |

The "High Quality Display" option is used to improve the quality of the image in homothetic mode when the image size is reduced from its initial size.

WinDev supports the images with a transparent background. In this case, you must:

- create an image whose background color is "Light Magenta" (RGB color: 255, 0, 255). The magenta is automatically considered as being the transparency color.
- create an image whose color of the top left pixel is the reference color for the transparency. You can define the color that will be the "transparent" color.



Caution!

The "transparent" color is applied to the entire image. Therefore, transparent areas may appear in the middle of the image. This may cause unexpected effects so don't forget to check the images!

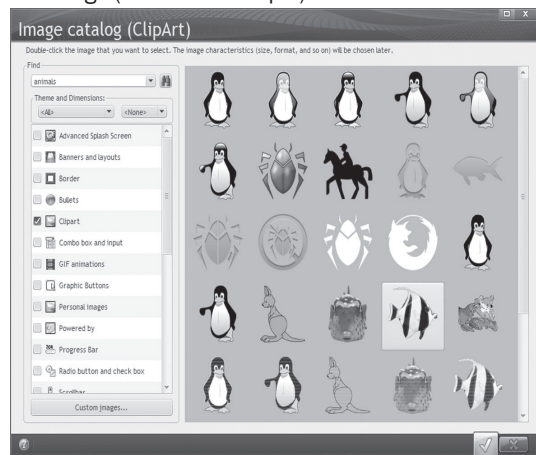


## The image catalog

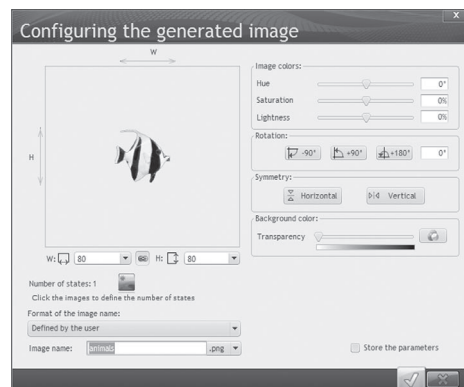
As soon as an image can be displayed in a control or in a page, WinDev proposes to use the image catalog via the "Catalog" button. This catalog contains hundreds of images, cliparts, ...

To select a clipart, you have the ability to perform a search according to a keyword. In the search window:

- Enter a keyword ("Animals" for instance)
- Specify (if necessary) the theme, the dimensions and the category.
- Click the search button (the magnifier). The images related to the keyword are displayed in the window.
- Click the requested image (a fish for example) and validate.



- Specify the characteristics of the image to generate: size, lightness, extension, ...



The name of the specified image is displayed in the description window of the image and the image file (in the requested format) is copied into the project directory.

## Animated GIF and animated image

Two methods can be used to animate an image:

- use an image in animated GIF format: the animation is contained in the image file.
- use an image containing the different drawings of the animation.

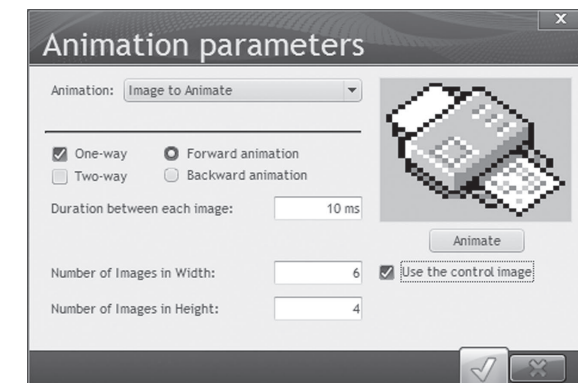
► In the editor, open the "WIN\_ImageControl.WDW" window:

1. Click the "Animation, zoom, click, 9-image mode" tab.
2. Display the description of the control containing the animated GIF. The "General" tab only contains the name of the image to display. This file is in GIF format. The animation is contained in the file.

► The "WIN\_ImageControl.WDW" window also uses an image animated by WinDev. The first Image control contains the entire image: this image is made of 6 drawings in width and 4 drawings in height. We will present the setting used to create the animated image.

► Display the description of the image control containing the image of the animated printer. In the "Details" tab of the control, "Animated at creation" is checked. This option is used to specify that the control contains an image that must be animated as soon as the control is displayed.

Click the [...] button found beside the "Animation: Image to animate" option. The window for configuring the animation is displayed.



This window contains all the parameters of the animation. The number of drawings used in width and in height is also found in this window. To test these parameters on the current image, check "Use the control image". The parameters of the animation can be modified and their effect can be checked immediately.

► Close the description of the Image control.



Example

For more details about all the proposed parameters, see the "WD Animated" example provided with WinDev (accessible from the "Wizards, Examples and Components" pane).



Tip

The animation parameters can be described in the control but they can also be modified by programming with the WLanguage functions: **AnimationSetParameter**, **AnimationSetMode** and **AnimationGetParameter**. See the online help (keyword: "Animation, Animate an image") for more details.

### Clickable image

Your images can be converted into clickable images via a mouse click. To do so, you must:

1. Display the description of the image.
  2. In the "Details" tab, select:
    - "This image is a click area": this option is used to specify that the image can be clicked
    - "Priority for mouse clicks": this option is used to manage the click on the image regardless of the z-order position of the control (in the foreground or not).
  3. A new process is then associated with the image: Image click code. The effect of the click can be managed in this code.
- In the "WIN\_ImageControl.WDW" window, the click areas are:
- the image containing the head of the lion.
  - various image controls located on top of the lion's head. These Image controls have no associated image and they are used to define the click areas.

### 9-image mode

The 9 image mode allows an image found in an Image control to automatically adapt to the control's size.

Let's talk a simple example: a resizable window. To harmonize the position of the controls when resizing a window, WinDev offers the anchoring mechanism (we'll see this later in this chapter). In this case, the Image control can change size with the window. The 9 image mode allows you to optimize the image's behavior.

- Run the test of the "WIN\_ImageControl" window. Click the "Animation ..." tab and increase the size of the window. The image control becomes larger and a new palm tree appears.

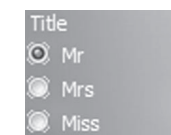
## Type of control: Radio button

### Summary

The radio buttons are used to select a single option among the proposed ones.

Depending on the selected display mode, each option can contain:

- a selection box and a caption
- a selection box, a caption and a sub-caption
- an image, a caption (and a sub-caption if necessary).



The different elements included in the option (image, caption and sub-caption) must be entered when describing the radio button in the editor. They can be modified by programming.

### Practical example

- Open the "WIN\_RadioButtonControl.WDW" window in the editor.
- Run the test of the window: this window contains two types of radio buttons:
  - a standard radio button
  - a radio button in Vista style with sub-captions.
- Click the different buttons, modify the radio buttons. See what happens.
- Go back to the editor.

### Characteristics of the radio buttons

WinDev proposes four display modes for the radio buttons:

- standard radio button.
- standard radio button with sub-captions.
- radio button in Vista style (ability to use an image for each option).
- radio button in Vista style with sub-captions (ability to use an image for each option).

For all these types of radio buttons, you have the ability to associate with each option:

- **an integer** (that can take a value from 1 to N, N being the total number of options found in the radio button).

The first option is number 1, the second number 2, ... The options are numbered from top to bottom and from left to right.

In our window:

- "RADIO\_Title = 1" means that "Mr" is selected,
- "RADIO\_Title = 2" means that "Mrs" is selected
- and "RADIO\_Title = 3" means that "Miss" is selected.

- **a specific returned value.** This returned value is specified in the "Content" screen. In our example:
  - RADIO\_TodaysDish = "Moussaka" means that "Moussaka" is selected.
  - RADIO\_TodaysDish = "Pie" means that "Pie" is selected.
  - RADIO\_TodaysDish = "Duck" means that "Duck" is selected.



Notes

#### Using the numbering or the returned values?

The method for programming radio buttons that return an integer or a value differs from the standard programming method. The benefit of using returned values is visible when inserting or moving an option in the radio button:

- If the radio button returns an integer, inserting or moving an option in the radio button involves a full check of the code used to manage the value of the radio button (to manage the offset of numbers).
- If the radio button returns a specific value, inserting an option into the radio button involves the management of the new option only. No modification is performed when inserting an option.

- Study the code of the different actions proposed by the window: the code used to manage the two radio buttons presents some differences.

### Type of control: Check box

#### Summary

The check boxes are used to specify whether the value of each option is "True" ("Yes") or "False" ("No").

Depending on the selected display mode, each option can contain:

- a box and a caption
- a box, a caption and a sub-caption
- an image, a caption (and a sub-caption if necessary).

The different elements included in the option (image, caption and sub-caption) must be entered when describing the check box in the editor. They can be modified by programming.

#### Practical example

- Open the "WIN\_CheckBoxControl.WDW" window in the editor.
- Run the test of the window and click the various buttons, modify the check boxes. See what happens.
- Go back to the editor. Let's take a look at the code associated with the various buttons.

#### Characteristics of the check boxes

WinDev proposes four display modes for the check boxes:

- standard check box
- standard check box with sub-captions
- check box in Vista style (ability to use an image for each option)
- check box in Vista style with sub-captions (ability to use an image for each option)

Each option found in a check box is "identified" by a subscript (that can take a value from 1 to N, N being the total number of options found in the check box).

The first option is option #1, the second option is option #2, ... The options are numbered from top to bottom and from left to right:

- "CBOX\_Hobbies[2] = True" means that "Write" is checked.
- "CBOX\_Hobbies[3] = False" means that "Dance" is not checked.

When several options are found, the check box operates like an array of options.

To access an option, all you have to do is pass the subscript of the option by using the syntax with the [ ] brackets.

To initialize a check box ("Initialize the check box" button in our example), the corresponding choice must be initialized with "True" or "False" depending on the case.

```
CBOX_Hobbies[2] = True // to check the option 2 "Write"
```

To find out the value of an option ("Retrieve the options" button in our example), all you have to do is retrieve the value of the control ("True" or "False").

```
IF CBOX_Hobbies[2] = True THEN
 // option checked
ELSE
 // option not checked
END
```



Notes

No option is checked by default in a check box.



Tip

The programmers who are used to handling boolean expressions may omit the test with "True" or "False":

```
IF CBOX_Hobbies[2] THEN
 //option checked
ELSE
 //option not checked
END
```

You have the ability to modify:

- the caption of a check box or the caption of an option found in a check box with the property named **..Caption**. However, no option can be added by programming. In our example, this operations is performed with the "Caption" and "Options" buttons. For example:

```
// Changes the caption
CBOX_Hobby..Caption = "You prefer to"
// Changes the second option
CBOX_Hobbies[2]..Caption = "Write with a pen"
```

- the sub-caption of the option (if it exists) with the property named **..SubCaption**.
- the image of the option (if it exists) with the property named **..Image**. For example:

```
CBOX_Folder[2]..Caption = "Film"
CBOX_Folder[2]..Image = "Film.png"
CBOX_Folder[2]..SubCaption = "Digitized films (holidays)"
```

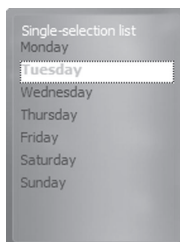
## Type of control: List box

### Summary

The list boxes are always expanded. This is the reason why they are also called "drop-down list boxes". They are used to display several elements and to select one or more elements. A list box can contain text and/or images.

When creating a "List box" control, a wizard proposes to create:

- a list box whose content is linked to a data file or to a query (browsing list box).
- a list box whose content is linked to a variable (list box on source).
- a list box whose content is defined in the editor or by programming (memory list box).



#### Rules for defining the type of list box to create

- Use a list box linked to a file or to a query when you want to propose to the user a list of values that are contained in a data file. If the list box is linked to a data file, the entire content of the data file is proposed directly. If the list box is linked to a query, the list box will contain a selection of records found in the data file.
- Use a list box whose content is defined in the editor if the values are "STATIC". They do not change during the execution of the program.
- Use a list box filled by programming if the content of the list box is the result of calculations or if the elements that must be included in the list box cannot be selected by a query.
- Use a list box whose content is linked to a variable when the content of the list box comes from calculations performed via a variable.

Caution!

### Practical example

- ▶ Open the "WIN\_ListBoxControl.WDW" window in the editor.
- ▶ Run the test of this window. This window contains the 3 types of list:
  - Single selection memory list, filled by programming. This list is empty and can be filled by the "Fill" button.



Notes

#### Single-selection list box

A single-selection list box is equivalent to a combo box or to a radio button. A single choice is possible. Depending on the interface of your application, choose the control that will allow the user to easily select his choice.

- Browsing list box containing the list of contributors. This list box is automatically filled with the content of the associated data file.
  - List on source displaying a list of months. This list box is automatically filled with the content of the associated variable.
  - The two lists at the bottom are multi-selection memory list boxes.
- ▶ Click the different buttons and see what happens. At run time, in a multi-selection list (using the Windows standard):
    - to select several elements one by one, press the [CTRL] key while selecting the elements with the mouse.
    - to select several consecutive elements at the same time, click the first element, press the [SHIFT] key, keep this key down and click the last element of the selection.
  - ▶ Go back to the editor.

### Mode for filling a list box

The wizard for creating a list box proposes to create several types of list boxes:

- list box linked to a data file or to a query (called browsing list box)
- list box filled by programming (called memory list box)
- list box filled from a variable (called list box on source)

Each type of list box corresponds to a specific fill mode.

#### Filling a memory list box

Two methods can be used to fill a memory list box:

- in the editor,
- by programming.

**In the editor:** the content of a list box can be defined in the description window of the control ("General" tab, "Initial content" control). Several values can be entered. Press [Enter] to go to the next row.

To fill a list box **by programming**, you must use **ListAdd**.

- In the practical example, the initialization of the memory list box is performed in the "Fill" button. The code used is as follows:

```
// Adds Monday to LIST_Day
ListAdd(LIST_Day, "Monday")
```

### Filling a list box from a data file or from a query

For a browsing list box, the list box is filled from the elements specified when creating the list box. This information is also available in the "Content" tab. This information is as follows:

- the analysis file or the query that will be used as data source for the fill operation.



Notes

If the query does not exist when the control is created, you have the ability to create the query at the same time as the control. The query will be embedded into the window that contains the control.

Caution: if you are using an embedded query, this query will be used for this control only. It cannot be re-used in your project.

This data file or this query will be automatically read without having to write a single code line.

- the item to display in the list box.
- the search key used: this item will be used for sorting when filling the list box; it specifies the order in which the values will be displayed in the list box.
- the retrieved value: when selecting an element of the list box, the retrieved value can be:
  - the subscript of the selected row,
  - a value of the data file.

In most cases, an item of the data file is chosen as retrieved value. It is the unique key in most cases, giving you the ability to perform searches in the data files for example.

- In the practical example, see the "Content" screen of the "Browsing" list box to check the characteristics of the browsing list box used.



Notes

### The different types of browsing list boxes

Two types of browsing list boxes are available:

- Browsing list box "Loaded in memory": The content of the list box is loaded in memory, then it is displayed in the list box. The access to the different elements of the list box is faster. This mode is reserved for the data files containing less than 100 000 records.
- Browsing list box with "Direct access (no limit)": The list displays the content of the data file or query directly. The current record corresponds to the selected element. Each move in the list box involves an access to the linked file. This type of list box is reserved for files containing more than 100,000 records.

### Filling a list box on source

A list box can be filled with the information found in a WLanguage array for example. The array is a global variable of the window:

```
MyMonthArray is array of 12 strings
MyMonthArray = ["January", "February", "March", "April", ...
 "May", "June", "July", "August", "September", ...
 "October", "November", "December"]
```

For a list box on source, the list box is filled from the elements specified when creating the list box. This information is also available in the "Content" tab. This information is as follows:

- the name of the source variable,
- the variable to display in the list box,
- the stored variable.

- In the practical example, see the "Content" screen of the "On source" list box to check the characteristics of the list box based on source used.

### Characteristics of the list box

Once the list box is created, its characteristics can be modified in the description window of the control ("Description" from the popup menu). The different tabs are used to specify all the options of the list boxes.

For example, the "Details" tab is used to:

- define whether the list box is multi-selection.
- define whether the list box is sorted.
- define whether the move must be performed to the pixel if a tooltip is displayed by the scrollbar, ...

### Programming a list box

The method for programming the list box controls is very easy in WinDev: all you have to do is use the WLanguage functions that start with LIST. These functions can be used on the memory list boxes, browsing list boxes or list boxes on source.



Notes

You don't know whether the function can be used on a memory list box, on a browsing list box or on a list box on source? See the online help! Use the F1 key on the function name.

### Modifying a list box (memory list box only)

Two methods can be used to modify an element in a list box:

- the function named **ListModify**
- the direct assignment by using the subscript of the element to modify.

```
ListModify(LIST_Day, "MONDAY") // Modifies the current element
ListModify(LIST_Day, "MONDAY", 1) // Modifies the element 1
LIST_Day[1] = "MONDAY" // Modifies the element 1
```

Note: To access an element in a list box, its subscript must be specified. The subscript of the first element is set to "1".



### Retrieving the selected element

In a single-selection list box, to retrieve the selected element, you must know the subscript of the selected element. The function named **ListSelect** returns the subscript of the selected element. If no element is selected, **ListSelect** returns "-1".

```
Subscript is int
Day is string
Subscript = ListSelect(LIST_Day)
Day = LIST_Day[Subscript] //Retrieves the selected element
```

**Note:** for a browsing list box, if the retrieved value corresponds to a value of the data file, directly use the name of the list box to find out the selected value.

In a multi-selection list box, to retrieve all the selected options, you must use **ListSelect**. As long as **ListSelect** does not return "-1", this function returns the subscript of the selected option.

```
// Rows selected in a multi-selection list box
i is int = 1
SelectedRow is int = ListSelect(List1,1)
WHILE SelectedRow <> -1
 Trace("The row #" + iSelectedRow + " is selected")
 i++
 SelectedRow = ListSelect(List1,i)
END
```

To find out the number of selected elements, you must use **ListSelectCount**.

To find out the number of elements in the list box, you must use the **..Occurrence** property or the **ListCount** function:

```
IF LIST_Day..Occurrence = 0 THEN //empty list
```

### Deletion (in the memory list boxes only)

To delete a value, use **ListDelete**.

To delete all the values from the list box, use **ListDeleteAll**.

```
//Deletes the selected element
ListDelete(LIST_Day)
```

### Search

To find an element in a list box, use **ListSeek**.

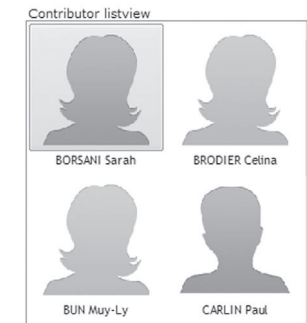
```
IF ListSeek(LIST_Day, Value) = -1 THEN
 // doesn't exist
ELSE
 // already exists
END
```

## Type of control: ListView

### Summary

A listview is a list used to display both text and images. An element found in a listview contains:

- a text
- a thumbnail (or image)



Two methods can be used to display a listview:

- listview mode: each element in the listview contains a caption and an image.
- standard mode: only the caption is displayed in the listview.

When creating a "Listview" control, a wizard starts and offers to create:

- a listview whose content is linked to a data file or to a query (browsing listview).
- a listview whose content is linked to a variable (listview on source).
- a listview whose content is defined in the editor or by programming (memory listview).

The same rules will be applied to choose the fill method.

This paragraph will only present the features specific to listviews. See the paragraph about list boxes for more details regarding the operating mode of listviews.

### Practical example

- ▶ Open the "WIN\_ListViewControl.WDW" window in the editor. This window presents a memory listview and a browsing listview.
- ▶ Run the test of this window.
- ▶ Click the different buttons and see what happens.
- ▶ Go back to the editor and study the processes of the buttons.

## Mode for filling the listview

A listview can be a browsing listview (linked to a data file or to a query), a memory listview or a listview on source.

### Filling a memory listview

A memory listview must be filled by programming with **ListAdd**. For example:

```
ListAdd(LSV_ListView1, "abort.ico image", "abort.ico")
// Add the "abandon.ico" image and its caption "Abandon.ico image"
// to the listview named LSV_ListView1
```

### Filling a listview from a data file or from a query

When creating a listview control filled from a data file or from a query, you must specify the analysis file or the query that will be used as data source to fill the listview. This data file (or this query) must contain:

- an item that will correspond to the caption of the element displayed in the listview.
- an item that will correspond to the image of the element displayed in the listview.



Notes

If the query does not exist when the control is created, you have the ability to create the query at the same time as the control. The query will be embedded into the window that contains the control.

Caution: if you are using an embedded query, this query will be used for this control only. It cannot be re-used in your project.

The other elements that must be specified in the wizard for listview creation are identical to the ones that must be specified for a list box.

### Filling a listview on source

A listview can be filled with the information found in an array, a class, a WLanguage structure. This element is a global variable of the window or project.

For a listview on source, the listview is filled from the elements specified when creating the listview. This information is also available in the "Content" tab. This information is as follows:

- the name of the source variable
- the variable that will correspond to the caption of the element displayed in the listview.
- an item that will correspond to the image of the element displayed in the listview

## Characteristics of the listview

Once the listview is created, its characteristics can be modified in the description window of the control ("Description" from the popup menu).

All the characteristics specified in the wizard are available as well as many other interesting options.

The "General" screen is used to configure the size of a listview cell.

The "Details" screen is used to:

- define whether the listview is multi-selection.
- define whether the listview is sorted.
- define the orientation of the control.

## Programming

The method for programming the ListView controls is very easy in WinDev: all you have to do is use the WLanguage functions that start with LIST. These functions can be used with memory listviews and/or browsing listviews.



Notes

You don't know whether the function can be used on a listview? See the online help! Use the F1 key on the function name.

Furthermore, some properties specific to listviews can also be used to handle the images or the display mode.

### Modifying a memory listview by programming

To modify an element in a listview, you can use:

- the function named **ListModify**.

```
//Modifies the current element
ListModify(LSV_ListView1, "abort.ico image", "abort.ico")
//Modifies element 1
ListModify(LSV_ListView1, "abort.ico image", "abort.ico", a1)
```

- the properties named **..Value** and **..Miniature** on the element to modify.

```
// Modifies element 2
LSV_ListView1[2]..Value = "Help.ico image"
LSI_ListView1[2]..Miniature = "Help.ico"
```

### Retrieving the value of a listview

The following elements can be retrieved from a listview:

- the caption of the element via the property named **..Value**.
- the image associated with the element via the property named **..Miniature**.

## Type of control: Combo box

## Summary

A combo box is also called "drop-down list". When creating a "Combo box" control, you have the ability to create:

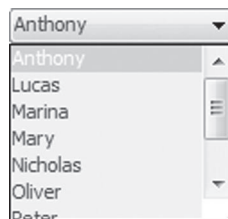
- a "Browsing" combo box whose content is linked to a data file or to a query.
- a "Memory" combo box whose content is defined in the editor or by programming.
- a combo box on source, linked to a WLanguage variable.

The same rules will be applied to choose the fill method.

The expanded part of the combo box may correspond to:

- a list box
- a listview
- a table
- a popup window

The choice of the expanded type is performed in the "Content" tab of the combo box description.



## Characteristics of a combo box

The combo box is expanded when the control is selected. Two types of combo boxes are available:

- the combo boxes where an element can be selected in the list. These combo boxes are called "**non-editable combo boxes**".  
The user expands the list to select an element.
- the combo boxes where an element can be selected in the list or directly entered in the control of the combo box. These combo boxes are called "**editable combo boxes**".  
The user can choose to enter a value or can expand the list to select an element.

The type of the combo box is defined in the description window of the combo box ("General" tab).



Caution!

Unlike the list boxes, the combo boxes are not multi-selection: a single element can be selected in the combo box.

The combo boxes can contain text and images.

This paragraph only presents the features specific to combo boxes. See the paragraph about the list boxes for more details regarding the operating mode of combo boxes.

## Practical example

- ▶ Open the "WIN\_ComboBoxControl.WDW" window. This window presents the features of the combo boxes.
- ▶ Run the test of this window. This window contains:
  - A sorted non-editable combo box,
  - A sorted editable combo box,
  - A browsing combo box.
- ▶ Click the different buttons and see what happens.
- ▶ Go back to the editor and study the processes of the buttons.

## Characteristics of the combo box

Once the combo box is created, its characteristics can be modified in the description window of the control ("Description" from the popup menu).

All the characteristics specified in the wizard are available as well as many other interesting options.

The "General" screen is used to define whether the combo box is editable or not.

The "Details" screen is used to:

- define the characteristics of the editable combo box
- define whether the combo box is sorted.
- propose the history of selections, in the first values of the drop-down list box.
- allow the user to modify the width and height of the combo box when it is expanded!

## Programming combo boxes

The method for programming combo boxes is identical to the method for programming list boxes. The WLanguage functions that must be used start with LIST. Several functions specific to the combo boxes (starting with COMBO) are available.

The only special aspect concerns the type of the combo box: "**editable**" or "**not editable**".



Caution!

No element is selected by default in a combo box, so it is empty. No combo box should be left empty except if it is an editable combo box.

## For an editable combo box

To assign or retrieve the control of an editable combo box, the syntax is the same as the syntax used to assign or retrieve an edit control:

```
// Assigns the selection of the combo box
COMBO_FirstName = "Julia"
```

```
// Retrieves the selection of the combo box
Str is String
Str = COMBO_FirstName
```

### For a non-editable combo box

The method for retrieving the element selected in the combo box is identical to the method for retrieving the element selected in a list box. To retrieve the subscript of the selected element, use the function named **ListSelect**. Then, this subscript must be used to retrieve the value.

```
// Retrieves the selected element
Subscript is int
Str is string
Subscript = ListSelect(COMBO_Combol)
Str = COMBO_Combol[Subscript]
// Retrieves the 3rd element of the list
Str is String
Str = COMBO_Combol[3]
```

To select an element, use the function named **ListSelectPlus**.

## Type of control: Table

### Summary

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

The term **Table** is used with SQL database.

A **Table control** is used to view the data in a table. In this case, the data can be entered or it may come from a data file (or from a table). Got it!

The table control of WinDev is used to view or modify the data:

- in memory: we talk of **memory table** or table filled by programming.
- coming from data files or queries: we talk of **browsing table**.
- coming from WLanguage variables: we talk of **table on source**.

The tables allow you to view several data in a single window.

The method for managing the tables depends on the type of the table (memory table, browsing table or table on source). The type of the table is defined in the wizard when creating the table.

We are going to present the different types of table controls.

### Practical example

The "Windows and controls" project offers two windows introducing the characteristics of tables.

- Open the "WIN\_TableControl.WDW" window in the editor. This window presents several processes that can be performed on the memory tables and on the browsing tables. Run the test of this window.
- Open the "WIN\_AdvancedTableControl" window in the editor. This window presents how to use browsing tables with breaks as well as tables on source.

### Mode for filling the table

A table can be a browsing table (linked to a data file or to a query), a memory table or a table on source.

### Filling a memory table

A memory table can be filled by programming with **TableAddLine**. This function accepts the name of the table and the content of the row in parameter. The content of the row is given column by column: each column value is separated by "a comma".

► In the "WIN\_TableControl" window, the filling of the table is done via the "Fill" button:

```
// Add an entire row
TableAddLine(TABLE_Menu, "Monday", "Scrambled eggs", ...
 "Irish stew", "Vanilla ice cream")
```

### Table linked to a data file or to a query

Various information must be specified when creating a table control filled from a data file or from a query:

- the data file or the query that will be used as data source for the fill operation.



Notes

If the query does not exist when the control is created, you have the ability to create the query at the same time as the control. The query will be embedded into the window that contains the control.

Caution: if you are using an embedded query, this query will be used for this control only. It cannot be re-used in your project.

This data file or this query will be automatically read without having to write a single code line.

- the items displayed in a row of the table.
- the search key used: this item will be used for sorting when filling the table; it specifies the order in which the values will be displayed in the table. For example, the products as well as their caption can be displayed in a table.



Notes

#### The different types of browsing tables

Two types of browsing tables are available:

- Browsing table "Loaded in memory": The content of the file is loaded in memory, then it is displayed in the table. The access to the different elements of the file is faster. The data being loaded in memory, all the table columns can be sorted. This mode is reserved for the data files containing less than 100 000 records.
- Browsing table with "Direct access (no limit)": The table displays the content of the data file or query directly. The current record corresponds to the selected element. Each move in the table involves an access to the linked data file. This type of table is reserved for the files containing more than 100,000 records.

This information is also available in the "Content" tab of the browsing table description window.

- In the "WIN\_TableControl" window, bring up the "Content" tab of the browsing table description window.
  - The file browsed is the "Contributor" file.
  - The search key used is the file identifier (ContributorID item)

#### Special case: Table linked to a query with parameters

If a table is linked to a query with parameters, the wizard for table creation proposes to "Generate the initialization code of the query parameters". This option automatically generates the initialization code of the table with default parameters. If some test data is specified, the data is automatically used (as already seen in part 2).



Notes

#### Refreshing a table based on a query

To refresh the display of a table based on a query (with or without parameters), use **TableDisplay** associated with the **taReExecuteQuery** constant.

#### Filling a table on source

The following information must be specified when creating a table control filled from a variable:

- the array variable that will be used as data source for filling the table.  
This variable will be read automatically without you having to write a single line of code.
- the variables used for each table column.

#### Characteristics of the table

Once the table is created, its characteristics can be modified in the description window of the control ("Description" from the popup menu).

All the characteristics specified in the wizard are available as well as many other interesting options.



Notes

The description window of a table contains two sections:

- the upper section, presenting the name of the table, the name of the columns as well as their type
- the lower section, containing the different description tabs.

If the name of the table is selected, the lower section presents the characteristics of the table.

If a column is selected, the lower section presents the characteristics of the columns.

Some useful information:

- You have the ability to perform automatic totals in the tables ("General" tab of the table description and "Details" tab of the columns). This feature will be presented in details in "Totals in the tables", page 238.
- The tables can be single-selection or multi-selection tables ("GUI" tab of the table description). The selection can be performed by row, by column or by cell.
- The "Style" tab is used to define the colors of the odd and even rows as well as the color of the selected row (or the background image used), ...

Regardless of the type of the table, browsing table, memory table or table on source, the columns can be:

- moved by the user (with the mouse)
- searched: a magnifier will be displayed, allowing you to search for a value in the column.
- sortable. Arrows will be displayed in the title of the columns, specifying that the column can be sorted and indicating the sort direction.



Notes

The magnifier or the sort arrow are visible on all the columns found in a memory table, in a table on source or in a browsing table loaded in memory. The magnifier and the arrow are visible on the columns linked to a key (index) for the browsing tables with "Direct access (no limit)".

#### Programming a memory table

In a memory table, each table row is identified by a subscript.

To handle a row or a column, the subscript of the relevant row must be specified. If the subscript is not specified, the selected row is used.

- A memory table can be handled row by row or column by column.
- A memory table can be editable or read-only.
- A memory table can be single-selection or multi-selection.

#### Functions used to handle a memory table

The method for programming the Table controls is very easy in WinDev: all you have to do is use the WLanguage functions that start with **TABLE**. These functions can be used on the memory tables, tables on source and/or browsing tables.



Notes

You don't know whether the function can be used on a memory table, on a table on source or on a browsing table? See the online help! Use the F1 key on the function name.

We will now present useful functions for handling a memory table:

- **TableModifyLine** modifies a table row (either the current row, or the row whose subscript number is specified).
- **TableSelect** returns the subscript of the current row (selected row) or the subscript of the selected rows for a multi-selection table.
- **TableSelectPlus** is used to select a table row by programming according to its subscript and/or to select several rows.
- **TableDelete** is used to delete a table row.
- **TableDeleteAll** empties the memory table.

- The "WIN\_TableControl" window lets you perform several operations on a memory table. See the processes associated with the "Select" and "Modify" buttons.



### States of a memory table

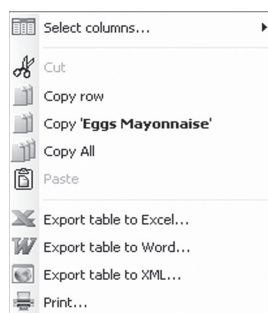
A table can be:

- read-only: no column is in edit.
  - editable: at least one column is in edit.
- These options can be defined for each column ("Details" tab of each column in the description window of the table).

The status of a table or the status of each table column is defined in the editor; it can be modified by programming with the property named **..State**.

### Tips and AAF on the tables

- Run the test of the "WIN\_TableControl.wdw" window. A popup menu is displayed when a right click is performed in one of the tables:



The "Chart" and "Automatic calculations" options are available only if the right click was performed on a numeric column.

Tips to optimize the appearance of your tables:

- **To use a multi-line column title**, check "Multiline column title" in the "Details" tab of the description window of the Table control.
- **To use a gradient background color**, display the "Style" tab of the description window of the control. Select the "Outside border" element, then select the "Gradient" border.

### Advanced used of tables: table with breaks

To improve the view mode of data in a table, WinDev proposes the **tables with breaks**. This allows you to add an additional sort on the displayed data.

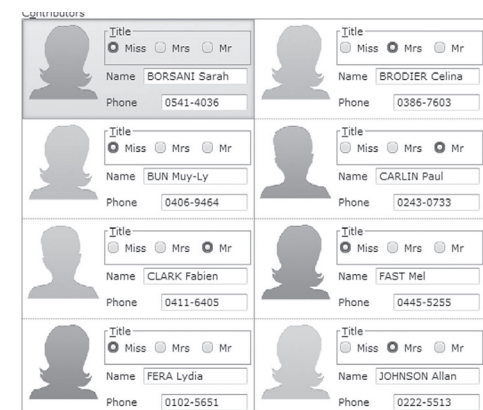
- Open the "WIN\_AdvancedTableControl" window in the editor. This window presents a browsing table with breaks.
  - Run the test of this window.
  - Different buttons are used to handle the breaks.
  - Stop the test and go back to the editor.
- In the editor, open the description window of the table: the characteristics of the data file browsed and the characteristics of the break are defined in the "Content" tab.

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

### Type of control: Looper

### Summary

Loopers are used to repeat "N" times a set of controls. Different types of information can be displayed in the looper controls. This enables you to display, in each row, the records found in a database.



Several types of loopers are available:

- **Memory looper**: the looper is filled by programming.
- **Browsing looper**: the looper is automatically filled from a data file or from a query, without a single code line.
- **Looper on source**: the looper is filled from a WLanguage variable without a single code line.

Several characteristics can be modified when the looper controls are repeated: value, caption, color of the text, status, ...

The looper can be moved like a control. When it is moved, all the controls associated with the looper are moved at the same time.

When creating a looper, you have the ability to define the number of columns that will be displayed. This option is used to get an horizontal repetition that does not exceed the specified number of columns. When the number of columns is reached, a new vertical repetition is added.

### Practical example

- Open the "WIN\_LooperControl.wdw" window. This window uses a looper containing a list of contributors. The data found in this looper comes from a HyperFileSQL data file.

The method for managing the loopers differs for:

- a memory looper,
- a looper linked to a data file or to a query.
- a looper linked to WLanguage data.

Let's take a look at the different types of looper controls.

## All the details

### Memory loopers

A looper can contain several controls (static control, edit control, image, ...). Each control found in the looper can be associated with an attribute.

This attribute is used to modify a property of the control (its value, its caption, its color, its visibility, ...). You also have the ability to create an attribute in the looper without associating this attribute with a control: we talk of free attribute (this attribute may contain the value of a counter for instance).

A memory looper behaves more or less like a table control.

In a memory looper, each looper row is identified by a subscript.

To add a row into a looper, you must add the function named **LooperAddLine**. This function expects in parameter:

- the name of the looper,
- the values of the attributes (in order) separated by commas:

```
// Add a row into the looper
// The looper contains 4 attributes, respectively associated
// with the controls: STCMODEL, IMGMODEL, QUANTITY, UNITPRICE
LooperAddLine(LOOP_Prototype, "Model 1", "IMAGE_MODEL1.JPG", ...
10, 45.65)
```

**LooperModifyLine** modifies a looper row.

A looper row is deleted by **LooperDelete**.

### Looper linked to a data file or to a query

When creating the browsing looper, you must specify the data file or the query that will be used as data source for the fill operation.

This data file or this query will be automatically read without having to write a single code line.

The selected search key is used for sorting when filling the looper, so it is used to specify the order in which the values will be displayed in the looper.

For example, a looper can be used to display:

- the photo of a product.
- the description of a product.
- the price of a product, ...

- ▶ In our example, see the "Content" tab of the looper control description window. The characteristics of the link with the data file are displayed.

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

## Type of control: TreeView

### Summary

A "Treeview" control is similar to a list box whose content is organized hierarchically. In other words, the data is organized in the same way as in the Windows file explorer.

### Practical example

- ▶ Open the "WIN\_TreeViewControl.wdw" window.
- ▶ Run the test of this window.
- ▶ Click the different buttons and see what happens.
- ▶ Go back to the editor and study the code of the buttons.

### All the details

In this example, the Treeview control is managed by programming. The WLanguage functions used to program a treeview start with **TreeXXX**.

These functions are similar to the functions for managing the tables filled by programming. For example, **TreeAdd**, **TreeDeleteAll**, **TreeDelete**, **TreeSelect**, ...

The TreeView control can also be handled row by row to modify the properties of a row. You have the ability to use:

- the property named **..Color** to modify the color of a row,
- the property named **..Brush Color** to modify the background color of a row,
- the property named **..Font** to modify the characteristics of the font for a row, ...

Similarly to the List box, Table, Combo Box or Looper controls, the treeview controls can have different types:

- **Memory treeview** the treeview control is filled by programming.
- **Browsing treeview**: the treeview control is automatically filled from a data file or from a query, without a single code line.
- **Treeview on source**: the treeview control is filled from a WLanguage variable without a single code line.

We will not go into details about the characteristics of Treeview controls or the functions used to handle them. See the online help (keyword: "TreeView, Handling a treeview by programming") for more details.

## Type of control: TreeView Table

## Summary

A treeview table is a cross between a Table control and a TreeView control.

This control is based on the structure of a Table control (column, row, cell) but it also contains a TreeView column that introduces the notion of tree structure. This notion doesn't exist in a table.

A treeview table can be:

- filled by programming
- based on a data file.

| TreeView Table |                |           |                        |
|----------------|----------------|-----------|------------------------|
|                | Name           | Phone     | Email                  |
| B              |                |           |                        |
| B              | BORSANI Sarah  | 0541-4036 | SarahBorsani@yahoo.com |
| B              | BRODIER Celina | 0386-7603 | BrodierC@yahoo.com     |
| B              | BUN Muy-Ly     | 0406-9464 | BunMuyLy@gmail.com     |
| C              |                |           |                        |
| C              | CARLIN Paul    | 0243-0733 | Carlin.Paul@gmail.com  |
| C              | CLARK Fabien   | 0411-6405 |                        |
| F              |                |           |                        |
| F              | FAST Mel       | 0445-5255 |                        |
| F              | FERA Lydia     | 0102-5651 | Fera@gmail.com         |
| J              |                |           |                        |
| J              | JOHNSON Allan  | 0222-5513 | johnson.a@gmail.com    |

## Practical example

- Open the "WIN\_TreeViewTableControl.WDW" window in the editor. This window presents the processes that can be performed on the browsing and memory treeview tables.
- Run the test of this window.

Note: since treeview tables are a special type of tables, they can also be handled by programming with the TableXXX functions. Specific functions allow you to manage the treeview (for example **TableAddChild** to add a child element) as well as give you the ability to collapse or expand a level (for example **TableCollapse** or **TableExpand**).

## Type of control: Progress bar

## Summary

The progress bars are used to represent the progress of a process. The value of the progress bar can be modified by programming. A progress bar can be horizontal or vertical. The colors and the style can be configured.

## Practical example

- Open the "WIN\_ProgressBarControl.wdw" window and run the window test.

## Programming tips

- To assign a value to a progress bar, all you have to do is perform an assignment like for any numeric control.

```
PROGBAR_Copy = 75
```

- To define the lower bound and the upper bound of a progress bar, use the properties named **..MinValue** and **..MaxValue**:

```
PROGBAR_Copy..MinValue = 0
PROGBAR_Copy..MaxValue = 100
```

- The code of the "Execution" button (used to move the progress bars) is as follows.

```
n is int
FOR n = 1 TO 100
 PROGBAR_Horizontal = n
 PROGBAR_Vertical = n
 PROGBAR_Image = n
 PROGBAR_Blue = n
 PROGBAR_LCD = n
END
```

## Type of control: Chart

## Summary

The Chart control allows you to easily include a chart in a window. The data source of this control can be:

- defined by programming.
- a data file or a query.
- a table column.
- a List Box control.
- a WLanguage array.

Several types of charts can be created in 2D or in 3D:

- Pie chart,
- Bar chart,
- Stock chart
- Line chart,
- Scatter chart, ...

The display options are used to produce different visual effects for each type of chart. See the online help (keyword: "Chart control") for more details.

## Practical example

- Open the "WIN\_ChartControl" window and test it. This window presents the different modes used to fill a chart control:

- Table (browsing or memory table).
- Data supplied by programming.
- List of values.

A Surface type chart can also be displayed.

- In each case, you can directly modify the information displayed in the chart. These modifications are immediately taken into account.



**Tip** The popup menu of the chart control is used to modify several display options of the chart.

- Go back to the WinDev editor. We are going to create a window that contains a chart control based on a query. This query (QRY\_TO\_per\_year) is used to find out the turnover per year.

1. Open the "QRY\_TO\_per\_year" query in the editor.
2. Run the test of the query.

Preview of QRY\_TO\_per\_year

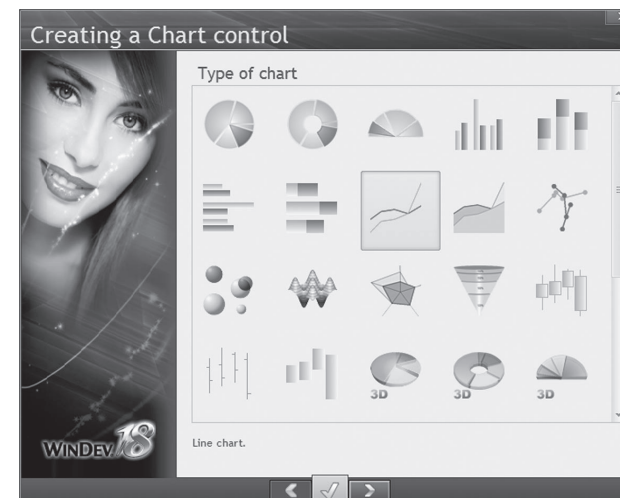
6 records correspond to your query.

| Year | the_sum_TotalSales |
|------|--------------------|
| 2006 | 55,325.00          |
| 2007 | 59,524.00          |
| 2008 | 33,046.00          |
| 2009 | 23,248.00          |
| 2010 | 55,995.00          |
| 2011 | 24,085.00          |

- We are now going to create the Chart control used to display the query's data.

1. Create a new window.
2. In this window, create a Chart control: on the "Creation" pane, in the "Graphic controls" group, click "Chart" then click inside the window to create the control. The wizard for chart creation starts.

3. Select the type of chart to create (line for example)

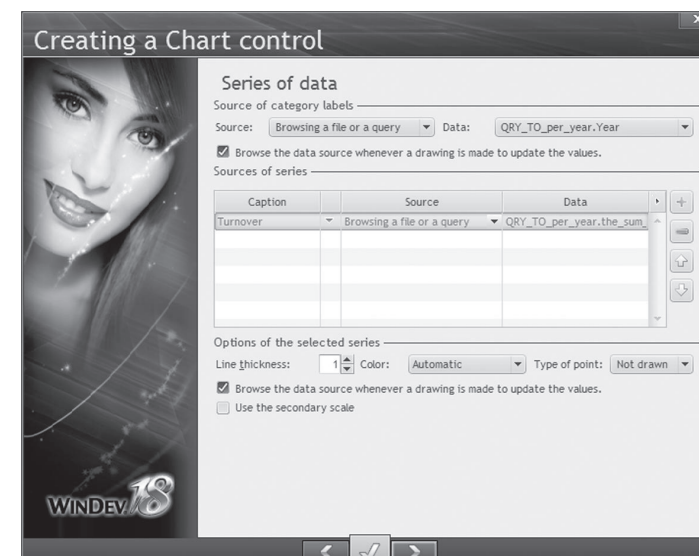


4. Specify the characteristics of the chart:

- Title of the chart: Turnover per year
- Display the legend to the right
- Horizontal gridlines

5. Indicate the series of the chart. In our case, we will be using a single series.

- Specify the category label that will be used: source corresponds to "Browse a file or a query" and data corresponds to the Year item of the QRY\_TO\_Per\_Year query.
- Click the first row of the table.



- Specify the caption: "Turnover".
  - Specify the source: select "Browse a file or a query". This option enables you to select the file or the query that will be used as base for the series.
  - Specify the data: click the "Data" column. Select the "QRY\_TO\_per\_Year" query, then the item: the\_sum\_TotalSales. In the search item, select "Year".
6. Enter the name of the chart and validate.
  7. Resize the chart control in your window and run the test.

### The chart functions

Several functions allow you to create and configure a chart. These functions start with "gr". These functions can be used:

- in an image control. The chart is drawn by the grXXX functions.
- in a chart control. The functions are used to modify by programming some characteristics defined in the chart control.

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

### The specialized controls

The "Specialized controls" group the controls that are used for specific features. These controls are as follows:

- Calendar
- Organizer
- Scheduler
- Organization chart
- TreeMap
- Carousel
- Cube
- HideShow
- Shape
- Bar code
- Scrollbar
- Range Slider
- Slider
- Spin
- Rating
- WebCam
- HTML
- OLE
- Sidebar
- Toolbar
- Ribbon
- Pivot table
- Multiline zone
- TimeLine
- ActiveX
- Xaml

We are going to study these different types of controls. The "Windows and controls" project contains an example for each type of control. We advise you to run the test of the window associated with the control, to study the source code found in this window and to see the online help for more details.

In the project explorer of the example project, the windows containing these controls are grouped in the folder named "Specialized controls".



## Type of control: Calendar

## Summary

The Calendar control is used to create a calendar. This calendar allows the user to select a date.



Notes

If a Date edit control is found at run time, the user will have the ability to open a calendar to select the date via "Calendar" from the popup menu.  
When describing a Date edit control, you also have the ability to display a Calendar button. This button allows the user to select the requested date.

## Practical example

- Open the "WIN\_CalendarControl" window. This window presents the different modes for using a calendar.

## Type of control: Organizer

## Summary

The Organizer control is used to display and handle the organizers. The display can be done on a daily, weekly or monthly basis.

## Practical example

- Open the "WIN\_OrganizerControl.WDW" window. This window uses an Organizer control. The different buttons present some operations that can be performed on an Organizer control.

## Type of control: Scheduler

## Summary

The Scheduler control is used to display and manage a schedule. The main difference between a Scheduler control and an Organizer control is as follows: an Organizer control only applies to one resource while a Scheduler control applies to several resources.

## Practical example

- Open the "WIN\_SchedulerControl" window. This window uses a Scheduler control. This Scheduler control is used to manage several appointments for several people. Run the test of this window to discover the possibilities of the Scheduler control.

## Type of control: Organization Chart

## Summary

The Organization Chart control is used to automatically display an organization chart or a hierarchy of elements. The data displayed can come from a database.

## Practical example

- Open the "WIN\_OrganizationChartControl" window. This window uses an Organization Chart control. Run the test of this window to discover the possibilities of the Organization Chart control. The popup menu of the control is used to create an image that can be printed.

## Type of control: TreeMap

## Summary

The TreeMap control is used to graphically represent data in a tree structure. It can be used to view the disk space occupied by the directory files for example.

## Practical example

- Open the "WIN\_TreeMapControl" window. This window presents the data displayed in a browsing table with break in the format of a TreeMap.
- Run the test of this window.
- Click the different controls and buttons and see what happens.
- Go back to the editor and study the code of the buttons.

## All the details

The TreeMap control is managed by programming. The WLanguage functions used to program a TreeMap start with TreeMapXXX.

These functions are similar to the functions for managing the treeviews. For example, **TreeMapAdd**, **TreeMapInfoXY**, **TreeMapCount**, ...

We will not go into details about the TreeMap functions. See the online help (keyword: "TreeMap") for more details.

**Type of control: Carousel****Summary**

The Carousel control is used to create outstanding menus and lists.

**Practical example**

- Open the "WIN\_CarouselControl" window. This window uses a Carousel control. The different buttons present some operations that can be performed on a Carousel control.

**Type of control: Cube****Summary**

The Cube control is a control that automatically performs rotations around the different axes. This control can be found in a corner of a window for example.

**Practical example**

- Open the "WIN\_CubeControl" window.  
Caution: the Cube controls uses the .Net 3.0 framework. Starting this framework may take quite a long time.  
Note: This window requires Windows XP or later.

**Type of control: HideShow****Summary**

The HideShow control can be expanded in order to display (or not) the included controls. This type of control is used to propose original menus.

**Practical example**

- Open the "WIN\_HideShowControl" window.

**Type of control: Shape****Summary**

The shapes are used to draw elements with geometric shapes (circle, ellipse, rectangle, square, line or arrow). A wizard for creating shape controls enables you to define all the characteristics of a shape control:

- colors.
- shapes, sizes, ...

**Practical example**

- Open the "WIN\_ShapeControl.wdw" window and run its test.

**Type of control: Bar Code****Summary**

The Bar Code control is used to easily display bar codes in your windows.

You have the ability to read the bar code of a product via a specific device, to display the bar code on the screen and to save its value in a data file.



Notes

The bar codes can be printed from the report editor or by programming (*iPrintBarCode*).

See the online help for more details.

**Type of control: Scrollbar****Summary**

In some specific processes, it may be interesting to create "Scrollbar" controls.

The principle for managing scrollbars is as follows:

- the value of the scrollbar box (the little square that goes up and down!) is retrieved in a variable.
- a value can be assigned to this box.

To move the scrollbar, the user can click on one of the arrow buttons or move the box directly. The user can even click between the box and the arrows to move faster.

**Practical example**

- Open the "WIN\_ScrollbarControl.wdw" window and run its test.  
The scrollbars are used to move the image inside the window.  
By default, the scrollbar moves between 1 and 100. Each click performed on one of the arrow buttons moves the box by 1 and each click performed between the box and the button moves the box by 10.  
These values can be modified in the description window of the scrollbar or by programming with the properties named *..MinValue*, *..MaxValue* and *..ScrollValue*

## Type of control: Range Slider

**Summary**

The Range Slider control is used to easily select a data interval or a zoom.

**Practical example**

- ▶ Open the "WIN\_RangeSliderControl.wdw" window and run its test.  
This window allows you to better understand the use of a Range Slider control:
  - the first example is used to handle the box of the Range Slider and see the different values of the associated properties.
  - the second example uses the Range Slider controls to manage a zoom area on an image.

Don't hesitate to take a look at the code of the different elements in this window to understand it works.

See the online help for more details.

## Type of control: Slider

**Summary**

A Slider control corresponds to a linear or circular graphic object used to show variation in a value. The current value of the slider can be retrieved by programming in a variable and it can be used in calculations.

**Practical example**

- ▶ To assign a value to a slider, all you have to do is perform an assignment like for a numeric control.

```
SLD_Sound = 75
```

- ▶ To define the lower bound and the upper bound of a slider, use the properties named **..MinValue** and **..MaxValue**:

```
SLD_Sound..MinValue = 0
SLD_Sound..MaxValue = 100
```

## Type of control: Spin

**Summary**

A Spin control is used to increment and decrement a value. A Spin is made of two buttons. Unlike a button, the click event is automatically repeated when clicking a Spin control while keeping the mouse button down.

**Practical example**

- ▶ Open the "WIN\_SpinControl.wdw" window and run its test.  
Spin boxes allow you to:
  - increment/decrement a value.
  - move the image in the window.

## Type of control: Rating

**Summary**

A Rating control allows:

- the user to give a rate.
- the application to give a mark.

The rating can be viewed via stars or via any other image.

**Practical example**

- ▶ Open the "WIN\_RatingControl.wdw" window and run its test.

## Type of control: Web camera

**Summary**

A Web Camera control is used to view a video source coming from an external camera connected to the computer.



Example

The "Web Camera" example, supplied with WinDev, presents the use of a Web Camera control in a window. This example is accessible from the "Wizards, Examples and Components" pane of WinDev.

**Practical example**

The functions used to manage the Web Camera control start with VideoXXX:

```
// Save the video broadcasted in the Web Camera control
// "CAM_WebCamera" as an AVI file of 60 seconds
Res is boolean = VideoCapture(CAM_WebCamera, ...
 "C:\Temp\MyVideo.AVI", ...
 viVideoCapture, 60)
IF Res = True THEN
 Info("Video successfully saved")
ELSE
 Error("Cannot save the video")
END
```



Notes

Only the video sequence currently broadcasted can be viewed in a Web Camera control.

To view:

- an image captured by **VideoCapture**, use an Image control.
- a video sequence captured by **VideoCapture**, use an ActiveX control that can read videos.
- ▶ **VideoDisplay** enables you to display a video straight from the Web camera.
- ▶ To understand the operating mode of the Web Camera control, you are going to load a sample window and to run it.
- ▶ Open the "WIN\_WEBCameraControl.WDW" window and run its test.  
Caution: A Web Camera must be accessible from the current computer.

See the online help (keyword: "WEB, Web Camera control") for more details.

## Type of control: HTML

### Summary

An HTML control is used to include an HTML page in a WinDev window.

**Note:** The HTML control can only be used to display an HTML Page while the HTML edit control can be used to display and modify the HTML code.

### Practical example

- ▶ To understand the operating mode of a HTML control, open the "WIN\_HTMLControl.wdw" window and run its test.

## Type of control: OLE, ActiveX

### ActiveX control

#### Summary

The ActiveX control is used to easily manage the ActiveX technology. The ActiveX controls are reusable components that use the ActiveX technology and that can be included in a window.

#### All the details

An ActiveX is associated with properties and methods. These properties and methods can be used in WLanguage to handle the ActiveX control directly. To do so, use the Automation syntax of WLanguage.

For example:

```
// AX_AVIRReader is an ActiveX control
AX_AVIRReader>>Zoom = 12 // Assign a property
AX_AVIRReader>>Pause(10) // Call a method
```

To find out the properties and the methods associated with an ActiveX, you can:

- see the documentation of the ActiveX control.
- use WDXVIEW supplied with WinDev.



Example

The "Controlling Excel", "Controlling Word" and "Search on Internet" examples, supplied with WinDev, present the use of an ActiveX control. These examples are accessible from the "Wizards, Examples and Components" pane of WinDev.

## OLE control

### Summary

The OLE control is used to handle an OLE object.

**Reminder:** OLE is a data exchange and command protocol whose principle consists in nesting and linking objects. The Client/Server connection is established as soon as the embedded object is opened.

An **OLE object** is a file containing:

- the reference of the OLE server that created the object.
- the object itself (image, document, ...).

### All the details

Technically speaking, an OLE object encapsulates data containing text, images, sounds, ... Data encapsulation in OLE objects allows these objects to be embedded without knowing anything about their structure.

WinDev is an OLE client. The applications developed in WinDev request the data, display and store the OLE objects.

The OLE servers provide the objects to the clients that request them. These objects can be PaintBrush, Excel, WinWord, ...

As a client, WinDev supports the OLE objects. It is used to:

- draw an object in an OLE control. An object can be an Excel worksheet, a Paint Brush image, ...
- edit an OLE object in an OLE control. The OLE object can be:
  - chosen in the editor or by programming.
  - defined according to a server or to a file.

**Notes:**

- In programming, an OLE control is handled by specific functions. A WinDev application **cannot** be an OLE server.
- OLE version 1 and OLE version 2 are supported (later versions are not supported).
- An OLE object cannot be printed in WinDev. The object is updated upon request.
- An OLE control can contain a single OLE object.

We won't go into details about the operating mode of these two types of controls.  
See the online help (keywords: "OLE", "OLE control", "ActiveX (control)") for more details.

**Type of control: Sidebar****Summary**

A Sidebar control includes several panes. Each pane can contain any type of control (including another sidebar).

**Practical example**

- Open the "WIN\_SidebarControl.wdw" window and run its test.

**Type of control: Toolbar****Summary**

The Toolbar control is an advanced type of control, linked to a window. A toolbar contains several controls (button, edit control, combo box, ...) allowing the user to quickly start some features of his application. A toolbar can be:

- **Anchored:** the toolbar is "Stuck" to one of the window sides.
- **Floating:** the toolbar is an independent window.

The user can change the position of the toolbar with the mouse.

**All the details**

To associate the existing controls with the toolbar:

**Method 1:** Drag and drop the control into the toolbar.

**Method 2:** In the description window of the toolbar ("General" tab), click the "Add" button and select the control that must be added to the toolbar. Apply the modifications, the selected control is automatically included in the toolbar.

To associate new controls with the toolbar:

**Method 1:** Drag and drop the control into the toolbar.

**Method 2:** In the description window of the toolbar ("General" tab), click the "Create" button and select the type of control that must be added to the toolbar. The selected control is automatically created in the toolbar. To modify it, click the "Edit" button: the description window of the new control is displayed.

**Practical example**

- Open the "WIN\_ToolbarControl" window and run its test. This window uses two toolbars: a docked toolbar and a floating toolbar.

**Note:** The spacing between the toolbar controls can be configured in the "Details" screen of the toolbar description.

**Type of control: Ribbon****Summary**

The Ribbon control is used to display menus in an original way. It is used to present in a small section of the screen an important number of controls arranged logically in order to perform choices.

A ribbon control includes several panes used to group the controls.

**Practical example**

- Open the "WIN\_RibbonControl" window and run its test. Enlarge and reduce the window to see the different options of the ribbon.

**Type of control: Pivot table****Summary**

The Pivot Table controls dynamically displays the data coming from the crosscheck of different files found in a database.

This enables you to display for example the volume of sales according to families of products, products or regions, as time goes by, with or without details.

**Practical example**

- Open the "WIN\_PVTControl" window and run its test.



**Type of control: Multiline zone****Summary**

The Multiline zone control lets you build original interfaces. This control is used to group several controls:

- options on a category,
- group of information about a contact, ...

This control can contain at the same time:

- lines defined in edit (static lines).
- lines defined at run time, by programming (dynamic lines).

**Practical example**

- Open and test the "WIN\_MultilineZoneControl" window.

**Type of control: TimeLine****Summary**

The TimeLine control is used to represent chronologies of close events that can be parallel.

This control is useful, for example, for:

- creating a video montage.
- represent elements on an automate.
- building an animation, ...

**Practical example**

- Open and test the "WIN\_TimelineControl" window.

**Type of control: Xaml****Summary**

The Xaml control enables you to easily manage the Xaml language.

To use this control, we strongly advise you to see the Microsoft documentation.

## LESSON 3.3. ERGONOMICS OF THE APPLICATION

### This lesson will teach you the following concepts...

- Improving the ergonomics of the windows and controls
- Improving the ergonomics of the applications



Estimated time: 20 min

## Improving the ergonomics of the windows and controls

WinDev proposes several tools for improving the ergonomics of your windows and controls. We will present simple solutions to improve the usability of your windows.

### Style/Skin template

Each WinDev project is associated with a style sheet. The style sheet is used to define the characteristics of each type of control found in the project.

To simplify the use of styles, WinDev is supplied with several skin templates. The skin templates use a specific style sheet, containing the description of the interface for all the types of controls that can be used.

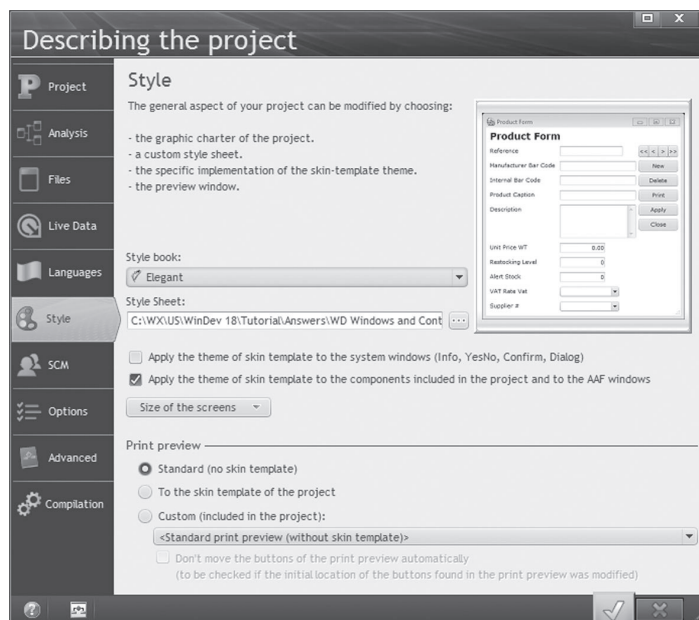


Notes

When modifying the style of a control ("Style" tab), WinDev proposes to overload the properties of the current style with the modified options: this is the mechanism of style overload. See the online help for more details.

When creating the project, you have the ability to define the style book of the project. This style book corresponds to the skin template that will be used. The style book can be modified at any time:

- directly in the project's options. To modify the programming charter:
  1. On the "Project" pane, in the "Project" group, click "Description".
  2. In the "Style" tab, modify the graphic charter. The change is performed for all the project windows.



- directly from the window editor.
  1. On the "Window" pane, in the "Style and Skin templates" group, click "Apply a skin template".
  2. In the window that opens, select the skin template to apply and its options. The change is performed for the current window.



Notes

WinDev enables you to create your own skin templates. This gives you the ability to entirely define the style book of your application. See the online help for more details.

### Resizing the windows

The windows found in the WinDev applications can be resized. In this case, the user will have the ability to use the sizing handle to resize the window. The sizing handle is found at the bottom of the window.

To allow a window to be resized, select "Resizable" in the "GUI" tab of the window description.

The size and position of the window controls must automatically adapt when the window is resized. This feature is called control anchoring. That's what we are going to study now.

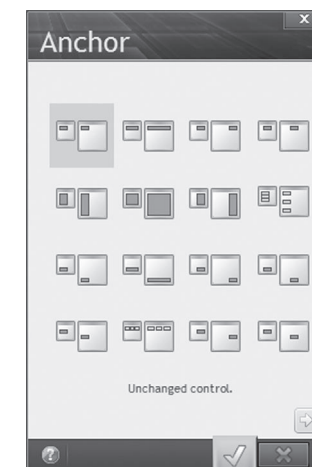
The windows should be defined as resizable only if they contain controls that can be resized: table, list box, image, sidebar, loop, splitter, tab, treeview, listview, HTML, ...

### Anchoring

The anchoring is used to link the size and/or position of the controls to the size of the window. If the user of the application has the ability to modify the size (width and height) of the window, some of the windows controls can be linked to this resize operation.

The control will be able to move, be enlarged, or remain the same.

To implement the anchor, right-click a control and select "Anchor" from the popup menu:





## Notes

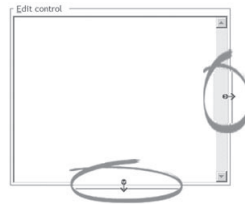
All the window controls do not necessarily have to be anchored. You have the ability to choose the controls and the windows to which anchors will be applied.

For example, in a window containing a Table control, it may be interesting to resize the table when the window is resized.

Another example: an image containing a photo that will be resized when the window is enlarged.

On the contrary, you should avoid resizing the edit controls, the buttons, the check boxes and the radio buttons.

The anchoring is represented by red arrows on the sides of the control:



Note: If your window is resizable and if your controls are not anchored, an AAD (Aided Application Development) is automatically displayed once your window was analyzed (about 10 seconds) to signal the problem and to help you anchor the controls.

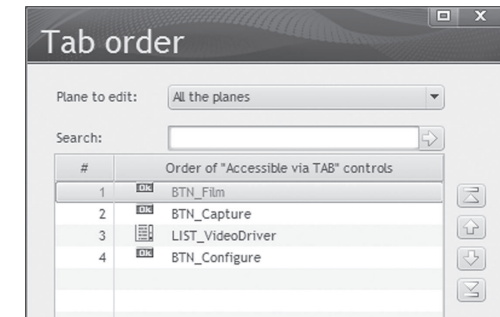
### Tabulation/Tab order

The tab order of the controls is the order in which the user can enter the values in the different window controls. The move from a control to another one is performed via the [TAB] key at run time.

The default tab order corresponds to the order in which the controls were created. It can be modified:

- by specifying an automatic tab order: the first control in edit will be the control found in the top left part of the window, the second one will be the one immediately to its right or below, ...
  - by specifying a tab order by selection.
- To define an automatic tab order, on the "Window" pane, in the "Order" group, expand "Navigation" and select "Define automatically".
- To edit the tab order of the controls, on the "Window pane, in the "Order" group, expand "Navigation" and select "Edit".
- The window that comes up presents the tab order used in the current window. You can also see the tab order used for the controls of a specific plane or search for the name of a control (for windows containing a lot of controls).

For example:



- To define a tab order by selection:

1. Select the controls in the requested tab order while keeping the [CTRL] key down.
2. On the "Window" pane, in the "Order" group, expand "Navigation" and select "Define from selection".

To find out the tab order of the controls found in your window, no need to run the test of your window: press [F5].

### Rulers/Interface checker/Alignment

One of the most important points when creating an application is the style of the interface. An important point is how the different controls are standardized, how they are aligned in the window. Several WinDev tools help you create interfaces that comply with the programming standard: grid, rulers, alignment options, automatic positioning, ... A few tips to choose the best suitable tool.

When creating our windows in the previous lessons, you have probably noticed that dotted lines were displayed when moving your controls in the window. This is the **automatic positioning**. These "rulers" help you position your controls in real time. You can see straightaway whether the control is aligned with the previous control.

To configure the automatic positioning:

1. On the "Display" pane, click the grouping button ( ) of the "Options" group.
2. In the window that opens, select the "Magnetism" tab. You can then enable or disable the automatic positioning.



## Notes

To move your controls without using the magnetism or the rulers, keep the [SHIFT] key down during the move.

You may also want to align some window controls once they have been created (once they have been moved for instance). You have the ability to use the alignment options. These options are all grouped in the "Alignment" pane.

To use the features of this pane, select several controls (with the lasso or with the CTRL key) and choose one of the preset alignments. The control taken as reference to perform the alignment is the first selected control.



Notes

WinDev lets you view the effect of the alignment option without actually applying it. To do so, simply select the controls to align and hover over the alignment icon you want. To apply the option, just click the alignment icon.

If none of the preset alignments suits you, you have the ability to define a custom alignment: On the "Alignment" pane, in the "Other alignments" group, the "Custom alignment" option lets you configure all the options.

The rulers allow you to position your controls to the pixel. To display the rulers, press [CTRL] + [R].

If your interfaces must comply with all the options defined for the Windows interfaces (control size, and so on), you also have the ability to use the interface checker. To start it, on the "Alignment" pane, in the "Other alignments" group, click the "Interface checker" option. This checker analyzes your interface and proposes all the necessary modifications. It is rarely used.

## Plane/Tab

Your window contains a large number of controls? Why not use planes or tabs to group the information.

### Tabs

The tabs are used to group the information by theme. The user directly accesses a theme by clicking the requested "pane".

The tabs look familiar to you? Indeed, this system is used in all the description windows found in WinDev.

The principle for handling tabs is very easy.

1. Create the tab control. On the "Creation" pane, in the "Containers" group, expand "Tab and associated" and click "Tab".
2. Create as many panes as necessary and give a caption to each pane ("General" tab of the control description).
3. Associate the controls with the tabs you want. A control can be common to all the tabs or associated with a specific tab.

► Two methods can be used to associate the control with a tab:

1. By moving the control:
  - Click the requested tab to enable one of its panes.
  - Click the control to associate and move it to the active pane.

The control is directly associated with the tab and with the tab pane.
2. By using the popup menu:
  - Move (if necessary) the relevant control to the tab.
  - Right-click the control to associate and choose "Associate with a tab..." from the popup menu.
  - Select the tab and the tab pane, and validate.

The control is automatically associated with the tab and with the specified tab pane.

A control can be associated with a single tab pane. The control will be active and visible when the tab pane is enabled.

Your tab is ready to operate, no programming is required.

## The planes

Planes are also used to group the controls while limiting the number of windows. The developer is the one who decides to display the controls in the planes.

In most cases, planes are used in "Wizard" windows. The "Next" and "Previous" buttons are used to move from a plane to another one.



Example

The "Wizard" example, supplied with WinDev, presents the use of planes in a window. This example is accessible from the "Wizards, Examples and Components" pane of WinDev.

The planes are very easy to handle in the window editor of WinDev. Indeed, when you want to describe a new screen without changing window, press the [PageDown] key to go to the next plane and press the [PageUp] key to go to the previous plane. The first plane of a window is named "Plane 0" by default. Each following plane is numbered: "Plane 1", "Plane 2", ...



Notes

To simplify the use of planes in the window editor, the number of the current plane is displayed in the window editor (in the home window and in the status bar of the editor).

To change plane by programming, use **..Plane**. For example:

```
// Click code of PREVIOUS button
MyWindow..Plane = MyWindow..Plane -1
```

```
// Click code of NEXT button
MyWindow..Plane = MyWindow..Plane +1
```



Notes

**MyWindow** is a keyword that returns the name of the window containing the process that runs **MyWindow**. It can be used in the processes of the window controls, in the processes of the window and in the local procedures of the window.

## Splitter

The splitter is a control used to divide a window into several sections. These sections can be resized by the end user. The splitting can be horizontal, vertical or a combination of both.



Notes

Implementing splitters in a window requires to implement the management of anchors: the dimension of the controls must be adapted in the different areas of the window.

Splitters allow the user to customize his work environment according to his own needs.

## Animation of windows

To give your applications an outstanding look or to impress users when starting your applications, the windows can be animated when they open and/or when they close: a window can turn around, it can come from the background of the screen (a little bit like video effects).

The effect can be fixed or chosen at random.

This feature can be defined in the options of the project, in the window description or by programming.

If a window loads several elements, an "animation" (performed independently of the current processes) allows the users to wait for the window to be ready!

This feature must be used sparingly to avoid annoying users. You can for example animate the first window of your applications or limit this animation to a specific number of windows.

You can animate:

- all the windows of the application. In this case, the animation is defined at the project level:
  - On the "Project" pane, in the "Project" group, click "Description".
  - In the project description window, click "Advanced".
  - Click the "Window animations" button. This screen lets you define the animation for the first window of the project and/or for all the windows of the application.
- one or more given windows ("Details" tab in the description window of each window).

## Dim disabled windows

Your application opens several windows at the same time and users don't know which one to use? The inactive windows can be automatically grayed. This feature allows you to always know which window to use. This feature is called DDW, which stands for Dim Disabled Windows.

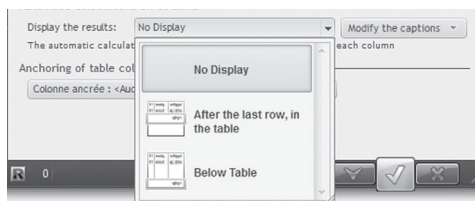
To enable this feature:

- On the "Project" pane, in the "Project" group, click "Description".
- In the project description window, click "Advanced".
- Select the "Enable DDW" option.

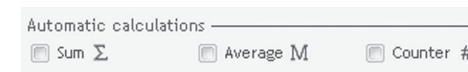
## Totals in the tables

You use memory tables or browsing tables in your applications? You want to display a total but you don't know how? You want to browse the file linked to the table? You want to perform a query? A simple solution is available: use the automatic calculations of columns: sum, count or average. To do so:

- Select the display position of the calculation ("General" tab of the table description):



- Select the column on which the calculation will be performed.
- In the "Details" tab of this column, check the requested calculation.



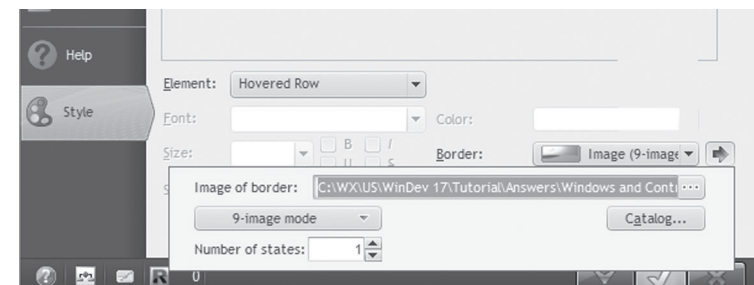
Notes

The user has the ability to add an automatic calculation on the table via the popup menu of the table.

## Images of controls configurable in the style

WinDev allows you to configure the interface of your controls without having to re-create the style sheet or the skin template of the application. How? Easy, all you have to do is specify the images that must be used in the control's style.

For example, in the "Style" tab of a table, you have the ability to configure the background image of the selected row:



## Transparency of controls, transparency of windows

The transparency enables you to give special effects to your controls and to your windows. The transparency can be used to display an alert window or a warning window "above" another window without interrupting the current use.

The transparency is defined:

- in the editor for the windows ("Opacity" in the "Style" tab of the window description) and for the controls ("Opacity" in the "GUI" tab of the control description)
- by programming for the windows and controls with the property named **..Opacity**.

## Saving the position of the columns in the tables

Another option will be very appreciated by the users, without any specific development: the automatic backup of the configuration of the table columns. The last configuration performed for the columns is automatically retrieved whenever the application is started: position, width of columns, ...

No programming is required: check "Store the column configuration" in the "Details" tab of the description window of the table.



## Ergonomics of the application

We have presented how the ergonomics of your controls and windows could be improved. Let's now talk about the general ergonomics of the application. Indeed, elements such as the menu or the opening mode of the application windows are very important elements regarding the appreciation of the end users. We've already seen how to create a menu so we will now present the management modes of multi-windowing.

WinDev proposes several modes for using the multi-windowing:

- using an MDI interface
- multi-windowing from free windows.

An example will be presented for each one of these management modes.

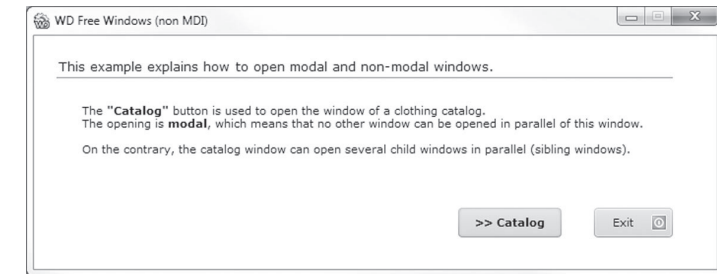
### Example of program used to manage the multi-windowing of free windows (non MDI)

To illustrate the different opening modes, we will be using the "WD Free Windows" project.

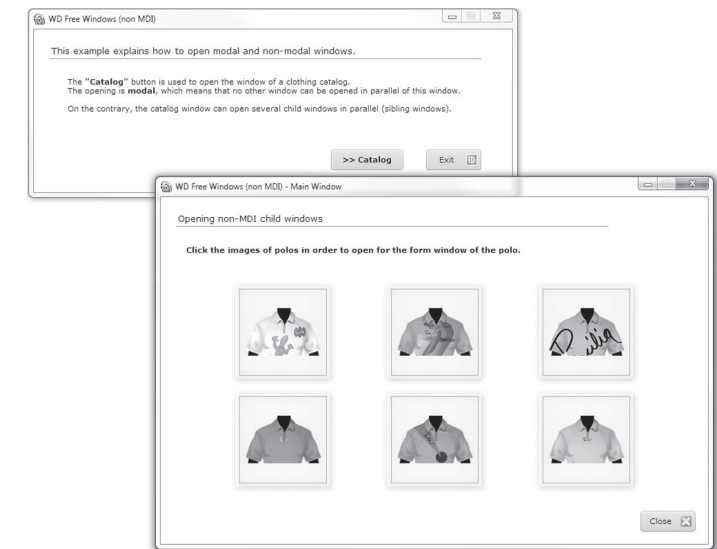
- Open the "WD Free Windows" project.
    1. Close (if necessary) the current project to display the home window.
    2. In the home window, click "Tutorial" and select the project named "Management of multi-windowing".

Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Management of multi-windowing".
  - Run the test of this project by pressing GO (🚀) among the quick access buttons.
- This example is deliberately simple: it is used to open and close windows. It supports no input and no MDI window. Its purpose is to illustrate the opening modes of the free windows.

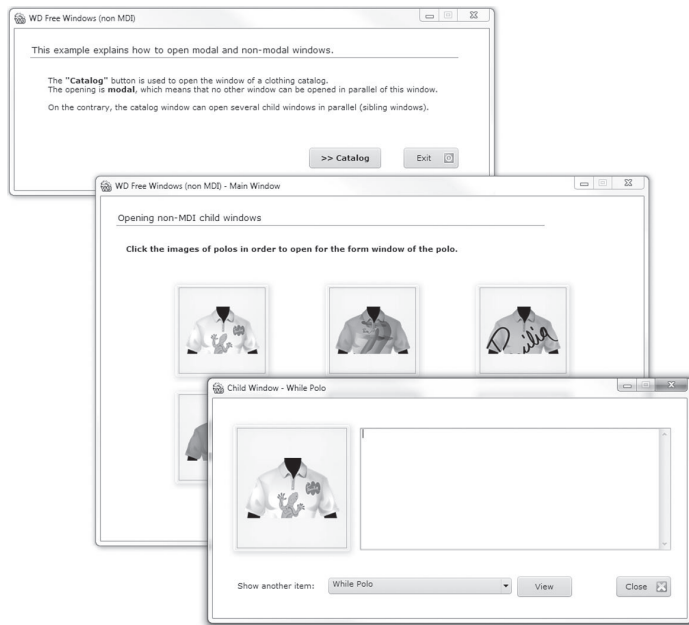
- The first window contains a "Catalog" button. This button opens a window in modal opening.



- Click this button.  
You will notice that you cannot click outside the child window (the catalog) and that the introduction parent window cannot be in edit as long as the child window is opened.



- Click one of the shirt images, the form of the shirt (child window of the catalog) is displayed. You will notice that the catalog window (parent window) remains in edit.



- Click the form window so that it becomes editable. All the forms can be opened together by clicking each shirt found in the catalog.
- From a shirt form, you have the ability to open another form (sibling window) by selecting the shirt in the combo box and by clicking the "View" button.
- Only the form is closed when you click the "Close" button of a shirt form. However, when you click the "Close" button of the catalog, all the forms are closed as well as the catalog window. Indeed, all the child windows are closed when the parent window is closed.

#### Details of the processes

- End the application's test and go back to the editor.
- Open the "WIN\_Main.WDW" window. The processes associated with each Image control are as follows:

```
-- Clicking IMG_CLICK1
sImageName is string = Myself
sAliasName is string = White Polo"
IF WinStatus(sAliasName) = NotFound THEN
 OpenChild(sAliasName + "=WIN_Child,0,0", sImageName)
END
```

The form window of the shirt is displayed by the function named **OpenChild**. Therefore, several forms can be opened in parallel.

In order for the window of a shirt not to be opened several times, the window is opened only if it does not exist. **WinStatus** is used to find out whether a window is open or not.

The polo's form window whose name is "WIN\_Child" can be opened several times. Therefore, you must differentiate between each opened window. To do so, use an alias.

#### What is an alias?

An alias is used to differentiate between the same window opened several times.

The alias is given when the window is opened by **OpenChild** or **OpenSister**.

In a polo's form window (WIN\_Child), the process of the button used to open another form is as follows:

```
nSubscript is int
sImageName is string


// Checks whether an element is selected
IF COMBO_Shirt[COMBO_Shirt]=-1 THEN Info("No shirt is selected")
nSubscript = ListSelect(COMBO_Shirt)
sImageName = COMBO_Shirt[COMBO_Shirt]+".png" // Selected shirt

// Opens the form window of the selected polo
// once you've checked that it was not already opened
IF WinStatus(COMBO_Shirt[COMBO_Shirt])= NotFound THEN
 OpenSister(COMBO_Shirt[COMBO_Shirt]+...
 "=WIN_Child,"+(50*nSubscript)+","+(50*nSubscript), sImageName)
END
```

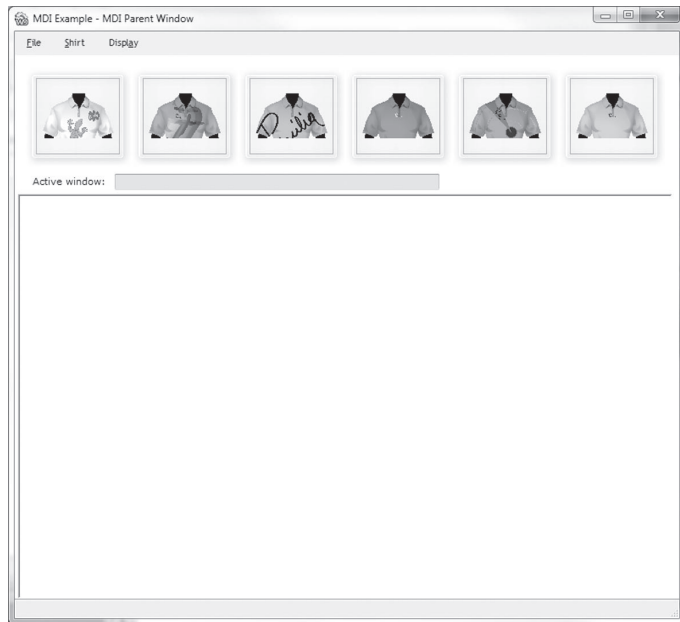
The form window of the polo is displayed by **OpenSister**. Therefore, several forms can be opened in parallel.

#### Example of program used to manage the MDI windows

To illustrate the management of MDI windows, we're going to use the "MDI Windows.WDP" project.

- Open the "WD MDI windows" project.
  1. Close (if necessary) the current project to display the home window. In the home window, click "Tutorial" and select the project named "Management of MDI windows".  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Management of MDI windows".
- Run the test of this project by pressing GO  among the quick access buttons).

The first window is an MDI parent window (it is named "WIN\_MDIParent").  
It contains an icon bar section.



The icon bar section includes:

- 1 edit control
- 6 click areas

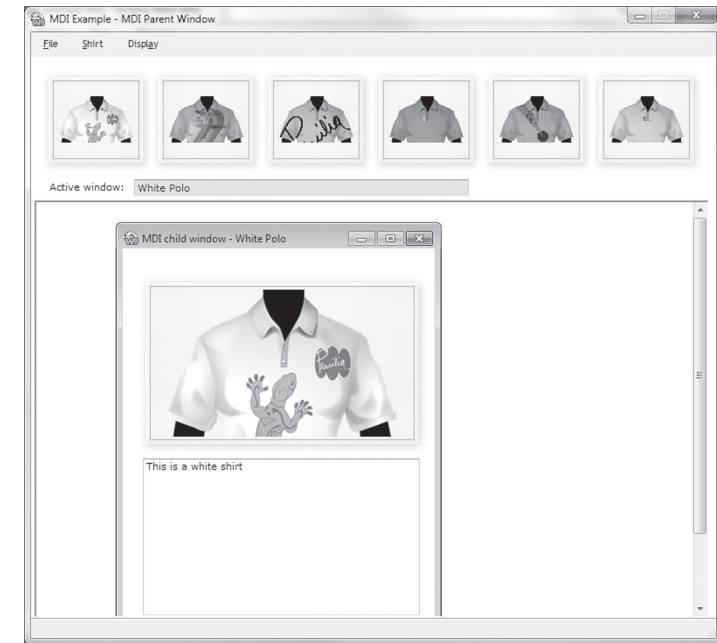
Each click area is used to open a child window.

For example, for the first click area, the associated process will contain:

```
// -- Initialization code
MySelf = "White Polo.png"

// -- Click code
ImageName is string = Myself
AliasName is string = "White Polo"
IF WinStatus(AliasName) = NotFound THEN...
 MDIOpen(WIN_MDIChild, AliasName, "White Polo", ImageName)
```

The child windows are MDI child windows, they are displayed in the parent window.



The options of the "Display" menu allow you to:

- Open all the child windows (the child windows are stacked)
- Close all the opened child windows:

```
MDIWindowMenu (mdiCloseAll)
```

- Display the opened child windows in tile:

```
MDIWindowMenu (mdiTile)
```

- Display the opened child windows in cascade:

```
MDIWindowMenu (mdiCascade)
```

- Minimize all the opened child windows:

```
MDIWindowMenu (mdiMinimizeAll)
```

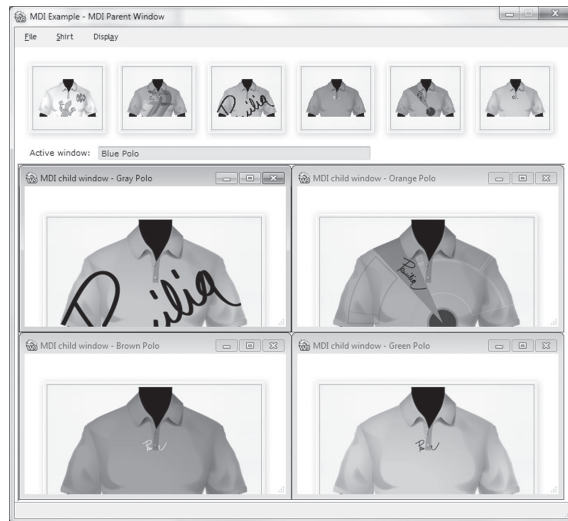
- Restore all the minimized child windows:

```
MDIWindowMenu (mdiRestoreAll)
```

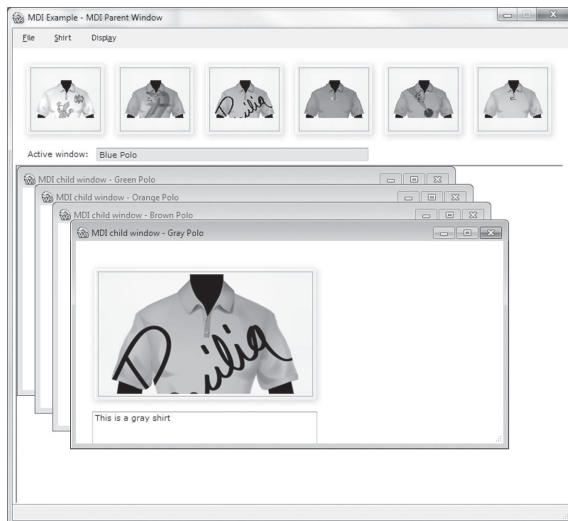
- Align the icons in the parent window:

```
MDIWindowMenu (mdiArrange)
```

## Tiled Display:



## Cascading display:



The "current" control indicates which child window is active. The active child window is the one found in the foreground, the one whose z-order position is the highest compared to the other windows.

**MDIActive** returns the name of the child window displayed in the foreground.

In our example, a procedure called by "timer" displays the name of the active child window.

The use of timers is explained in "Displaying the date and time in real time", page 46.

**MDIMother** is used to find out the name of the MDI parent window.

## LESSON 3.4. REUSABILITY

### This lesson will teach you the following concepts...

- Re-using the project elements



Estimated time: 10 min

### What does the reusability mean?

You want to re-use the same directory picker in all your projects? You want to use the same "Close" buttons at the same location in your applications?

WinDev proposes several methods for re-using the existing features.

### Re-using a set of controls

You want to re-use a set of controls? The following tools are available in WebDev:

- Supercontrols
- Control template
- Internal windows

We are going to present these three tools in details, with a summary table for each one of these tools to help you choose the most suitable one.

For each tool, the "Windows and controls" project contains a window that allows you to see and test the described feature.

### Practical example

To open this project in WinDev:

1. Close (if necessary) the current project to display the home window.
  2. In the home window, click "Tutorial" and select the project named "Windows and controls".
- Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Windows and controls".


### The supercontrol

The supercontrol is an advanced type of control. A supercontrol groups a set of controls for a specific purpose (directory picker, file picker, ...). The supercontrol contains:

- the necessary controls
- the necessary code.



## Summary table

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Creation    | <b>Creating a supercontrol in a window:</b> <ol style="list-style-type: none"> <li>1. Create the supercontrol: on the "Creation" pane, in the "Containers" group, click "Supercontrol" .</li> <li>2. Include the controls in the supercontrol.</li> </ol> <b>Refactoring:</b> <ol style="list-style-type: none"> <li>1. Select the controls.</li> <li>2. On the "Modification" pane, in the "Transformations" group, expand "Refactoring and swapping" and select "Create a supercontrol with the selection".</li> </ol> |
| Reusability | Via the dictionary                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Update      | Via the dictionary (via the subscription mechanism)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Teamwork    | Sharing the dictionary by network.<br>Source Code Manager not available.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

## Practical example

The "Windows and controls" project contains a window that uses a supercontrol:

1. In the project explorer, open the "WIN\_SupercontrolControl" window (found in the "Other controls" custom-folder). This window uses a supercontrol to select a file.
2. Run the test of the window.

## Internal window


The Internal Window control is used to include a window (and its code) in another window.

At run time, the internal window will be dynamically merged to the host window.

An internal window is a specific window (no title bar, no menu, and so on). All types of controls can be included in this window. An internal window is a file with the "WDW" extension.

The "Internal Window" control is used to dynamically share the same part of the interface among several applications.

## Summary table

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Creation    | <b>Creating an internal window:</b> <ol style="list-style-type: none"> <li>1. Create the internal window. To create an internal window, simply: <ul style="list-style-type: none"> <li>• click  found among the quick access buttons of the WinDev menu</li> <li>• hover the "Window" option in the wheel that comes up.</li> <li>• click "Internal window".</li> </ul> </li> <li>2. Define the controls and the code.</li> <li>3. Save.</li> </ol> <b>Refactoring:</b> <ol style="list-style-type: none"> <li>1. Select the controls.</li> <li>2. On the "Modification" pane, in the "Transformations" group, expand "Refactoring and swapping" and select "Create an internal window with the selection".</li> </ol> |
| Reusability | Via the "internal window" control.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Update      | Via the dictionary (via the subscription mechanism) or via the SCM.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Teamwork    | Share via the network or via the SCM                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Benefits    | Ability to dynamically modify (by programming) the internal window used in the Internal Window control.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Drawbacks   | <ul style="list-style-type: none"> <li>- No improvement is allowed in the interface: no overload, no move of controls.</li> <li>- Rectangular area.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**Use example:** window whose interface will never change regardless of the project: Option window.

## Practical example


To test the use of internal windows, we are going to use the "Windows and controls" project. To do so:

1. In the project explorer, open the "WIN\_InternalWindowControl" window (found in the "Other controls" custom-folder). This window uses several internal windows to view addresses.
2. Run the test of the window.

## Control template

A control template is a set of controls that can be re-used in several windows. 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".

### Summary table

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Creation    | <b>Creating a control template:</b> <ol style="list-style-type: none"><li>1. Create the control template. To create a control template, simply:<ul style="list-style-type: none"><li>• click  found among the quick access buttons of the WinDev menu</li><li>• hover the "Window" option in the wheel that comes up.</li><li>• click "Control template".</li></ul></li><li>2. Define the controls and the code.</li><li>3. Save.</li></ol> <b>Refactoring:</b> <ol style="list-style-type: none"><li>1. Select the controls.</li><li>2. On the "Modification" pane, in the "Transformations" group, expand "Refactoring and swapping" and select "Create a control template with the selection".</li></ol> |
| Reusability | Via the "Control template" control.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Update      | Via the dictionary (via the subscription mechanism) or via the SCM.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Teamwork    | Share via the network or via the SCM                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Benefits    | The control templates can be overloaded: source code can be added, the controls can be moved in the window that uses the control template. The controls can be modified.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

In most cases, we recommend that you use a control template.

### Practical example

To see the benefits of control templates, we are going to use the "Windows and controls" project. To do so:

1. In the project explorer, open the "WIN\_ControlTemplate" window (found in the "Other controls" custom folder). This window uses a control template to enter a time period. Two use modes are available for the control template:
  - control template without overloading. The control template is used as is.
  - control template with overloading. The blue squares are used to identify the overloaded controls.
2. To view the overloaded controls and the properties that have been overloaded, on the "Window" pane, in the "Templates" group, expand "Templates" and select "List of overloaded controls".
3. Run the test of the window.

## LESSON 3.5. QUESTIONS/ANSWERS

### This lesson will teach you the following concepts...

- Tips for handling the controls



Estimated time: 10 min

**Question How do I duplicate a control found in a window by programming?**

With WinDev, you can already clone controls. **ControlClone** is used to duplicate a control found in a window or in a report by programming.

The control is duplicated with the same characteristics but with a different name.

**Question How do I delete a control found in a window by programming?**

**ControlDelete** is used to delete a control from a window or from a report by programming.

**Question How do I manage the planes of a window?**

The planes found in a window are used to arrange the controls in "layers" to avoid overcrowding the screens and to keep the number of project windows down.

To associate a control with a plane:

1. Right-click the control.
2. Select "Associate with a plane".
3. Choose the number of the plane to which the control must be associated.

The [Page Up] and [Page Down] keys allow you to go from one plane to another in the editor. The number of the current plane is displayed:

- in the status bar of the editor (bottom left).
- in the home window of the current window (top right).



Tip

To avoid duplicating the same control in a window, the control can be associated with "no plane". The control will be visible in all the planes.

Only the controls of the current plane and the controls that belong to no plane are visible in edit and at run time.

You also have the ability to use **..Plane** to:

- finding out and changing the current plane in a window.
- finding out and changing the plane associated with a control.

The "Plane mode" pane is used to edit all the captions of the controls found in all the window planes in order to view them and to modify them if necessary. To display this pane, on the "Home" pane, in the "Environment" group, expand "Panels" then select "Plane mode".

**Question How do I manage the click on a tab pane?**

A tab is a kind of button divided into several click areas called "Panels".

To manage and define the pane that was clicked, use the following syntax in the click code of the tab:

```
SWITCH NameTabControl
 CASE 1 // first pane
 //...Process to Perform...
 CASE 2 // second pane
 //...Process to Perform...
 OTHER CASE
 //...Other processes to perform...
END
```

**Question How do I display the progress of a process?**

In most cases, a progress bar is used to display the progress of a process.

To do so, use a Progress Bar control in a window ("Creation" pane, "Graphic controls" group).

In the initialization code of the Progress Bar control:

1. Initialize the minimum value of the progress bar:

```
NameProgressBar..MinValue = MinimumValue
```

2. Initialize the maximum value of the progress bar:

```
NameProgressBar..MaxValue = MaximumValue
```

In the code of the requested process, increment the progress bar at each step of the process:

```
NameProgressBar ++
// or NameProgressBar = NameProgressBar + 1
```

**Question How do I display an Internet site in a window?**

WinDev enables you to easily insert an Internet link into your applications. To do so:

1. Insert an HTML control into a window ("Creation" pane, "Other controls" group).
2. Initialize the HTML control with the address of the site to which you want to connect:

```
NameHTMLControl = "http://www.windev.com"
```

Your computer (as well as the end-user computer) must be equipped with an active Internet connection.

**Question How do I modify the color of a control?**

The color of a control is defined in the control style ("Choose a style" option from the control popup menu). However, the color of this control can be modified by programming. The syntax is as follows:

```
// Color the text in red
ControlName..Color = LightRed

// Color the background of the static in green
ControlName..BrushColor = LightGreen

// Restore the initial color (the one of the style)
ControlName..Color = iDefaultColor
```



Notes

This syntax applies to all types of controls.

**RGB** is used to define a color from the values of the Red, Green and Blue components.

```
<ColorCode> = RGB(<red>,<green>,<blue>)
```

You also have the ability to modify the colors of the rows, columns or cells in a Table control. The syntax is as follows:

```
// Modify the color of a column
ColumnName..Color = <ColorCode>

// Modify the color of a row
TableName[RowSubscript]..Color = <ColorCode>

// Modify the color of a cell
ColumnName[RowSubscript]..Color = <ColorCode>
// or
// TableName[RowSubscript, ColumnSubscript]..Color = <ColorCode>
```

**Question How do I make a button invisible?**

A button can be made invisible by programming with the following syntax:

```
ButtonName..Visible = False
```

Set the value to **"True"** to make the control visible.

This syntax can also be applied to all types of controls and to groups of controls.

**Question How do I create the main menu of my application?**

Perform the following operations:

1. Create a new window or open an existing window.
2. On the "Window" pane, in the "Bars and menus" group, expand "Main menu" and select "Add the main menu". An option is automatically created.
3. Right-click this option.
4. Click "Option description" to modify the selected option.
5. Click "Add after" to add a new option after.
6. Click "Add before" to insert a option before.
7. Click "Transform to expand a sub-menu" to add a sub-menu into the tree structure.

**Question How do I create a popup menu?**

A popup menu can be added:

- at window level.
- at control level.

For a window:

1. Right-click the window and select "Description".
2. Click the "GUI" tab and click  beside the "Popup menu" combo box.

For a control:

1. Right-click the control and select "Description".
2. Click the "GUI" tab and click  beside the "Popup menu" combo box.

To find out or modify the popup menu of a control or window by programming, use the property named **..PopupMenu**.

**Question How do I modify the rollover cursor for a control or a window?**

The rollover cursor can be defined:

- at window level.
- at control level.

To define the rollover cursor:

1. Right-click the element (window or control) and select "Description".
2. Click the "GUI" tab and select the requested rollover cursor in the "Cursor" combo box.

You can choose one of the preset cursors or you can select a cursor that you've created.

To modify the rollover cursor by programming, use the property named **..MouseCursor**.

**Question How do I pass parameters to a window?**

The method for passing parameters to a window is similar to the method for passing parameters to a procedure.

In the declaration code of the global variables of the window, enter the following WLanguage code:

```
PROCEDURE WindowName (NameParam1, NameParam2, ...)
```

When the window is opened by **Open**, pass the parameters after the name of the window, for example:

```
Open(WindowName, ValueParam1, ValueParam2, ...)
```

If a parameter is initialized during the declaration in the window, this parameter becomes optional:

```
// NameParam2 is an optional parameter
PROCEDURE WindowName (NameParam1, NameParam2 = "Test")
```



Notes

We advise you to pass parameters to a window rather than declaring global variables in the project.

**Question How do I group controls in order to modify their properties by programming?**

Perform the following operation:

1. Select several controls with the mouse.
2. On the "Window" pane, in the "Edit" group, expand "Groups" and select "Associate the selection".
3. You can:
  - Associate the selection with an existing group.
  - Associate the selection with a new group ("New" button). All you have to do is specify the name of the group and validate.

The groups of controls can be used in the windows and in the reports.

The controls will be associated with this group. Then, you can modify the properties of the controls found in this group with the following syntax:

```
GroupName..<PropertyName> = Value
```



Caution!

Only the properties common to all the controls can be modified.

**Question How do I transform a check box into a radio button?**

On the "Modification" pane, in the "Transformations" group, expand "Refactoring and swapping" and select "Radio button/Check box" to swap the two types of controls.



Notes

This operation can also be performed between a combo box and a combo box with table or between a slider and a round slider.

**Question How do I give the same size to the buttons?**

- ▶ Select the button that will be used as reference for the size (height and width), then the other buttons to resize.
- ▶ Select "Same width, Same height" in the alignment tools of the "Alignment" pane.

**Question How do I manage the persistence of controls?**

- ▶ To manage the persistency of controls, select "Store the value" in the "Details" tab of the description window of the control.

Note: the user can also store the value of a control via the popup menu of the control.

**Question How do I add a background image to a window?**

On the window:

1. Right-click and select "Description".
2. Select the "Image" tab. You can choose an image and its display mode.

**Question How do I manage the transparency of a window?**

On the window:

1. Right-click and select "Description".
2. Select the "Style" tab.
3. Use the slider to define the opacity of the window.

You also have the ability to define the opacity of a window or control via the WLanguage property named **..Opacity**.



PART 4

Database and  
analysis

18



DEVELOP 10 TIMES FASTER 

## LESSON 4.1. INTRODUCTION

**This lesson will teach you the following concepts...**

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



Estimated time: 20 min

## Overview

You may have to handle data when designing an application. To store the data, you must create a "database".

In WinDev, when creating a project that handles data, an "analysis" must be created beforehand. An analysis contains the description of the files (or tables) containing application data.

When the application is run, these descriptions will be used to create the database and/or the data files. The data will be stored in this database or in these files.



Notes

Several tools allowing you to perform maintenance operations on the HyperFileSQL databases are supplied with WinDev. They can be accessed from the HyperFileSQL Control Center.

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

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

Several methods (also called "access modes") can be used to access the data:

- Native Access
- OLE DB access
- Direct ODBC access
- ODBC access via OLE DB

## The different modes for accessing databases

### Native Access

A native access handles a database format directly and exclusively. This type of optimized access is developed specifically for each database format.

A native access is available for the following types of databases:

- HyperFileSQL 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)

Other native accesses will be available soon, contact our sales department for more details!

The WLanguage functions **SQL\*** and **HRead\*** can be used with this type of access. The code is portable and independent of the database.

### Direct ODBC access

An access via direct ODBC uses a multi-database access standard. The 32-bit ODBC layer must be installed on your computer. In most cases, this layer is already installed in the recent versions of Windows. This can be checked in the control panel of Windows via "ODBC Administrator".

Caution: some databases may not be accessible via this method. Check whether an ODBC driver exists before you use this type of access.

Only the **SQL\*** functions of WLanguage 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).



Caution!

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. Check whether an OLE DB driver exists before you use this type of access.

The **SQL\*** and **HRead\*** functions of WLanguage can be used with this type of access.

### ODBC access via OLE DB

In summary, this 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 **SQL\*** and **HRead\*** functions of WLanguage can be used with this type of access.

## LESSON 4.2. BROWSING THE DATA FILES AND QUERIES

### This lesson will teach you the following concepts...

- HReadFirst browse
- HReadSeek browse
- FOR EACH browse
- FOR EACH WITH browse



Estimated time: 20 min

### HReadFirst browse

A **HReadFirst** browse is used to browse a data file according to a search key item (or index). In this case, **HReadFirst** is combined with **HReadNext**, that is used to read the next record.

For example, the following code is used to read the Customer file according to the "Name" key item, from the first record to the last one.

```
HReadFirst(Customer, Name)
WHILE NOT HOut(Customer)
 // Process the record
 HReadNext(Customer, Name)
END
```

In this code, the read loop is used to test the value of **HOut**. **HOut** is used to find out whether the end of the data file has been reached.

When browsing a query, **HReadFirst** automatically re-runs the query.



Note

The same method can be used to browse the file from the last record to the first one. To do so, use **HReadLast** and **HReadPrevious**.

### HReadSeek browse

A **HReadSeek** browse is used to position in the data file in order to browse the records corresponding to a condition. This condition is applied to the search key (all the orders passed by a customer for example). **HReadNext** is used to read the next record corresponding to the condition. For example, the following code is used to find the customers whose name starts with "Smith":

```
HReadSeek(Customer, Name, "Smith")
WHILE HFound(Customer)
 // Process the record
 HReadNext(Customer, Name)
END
```

In this code, the browse loop is used to test the value of **HFound**.

**HFound** is used to find out whether a record corresponding to the condition was found.



Note

**HReadSeek** performs a generic search by default: all the records that start with the specified value will be sought (in our example, Smith and Smither will be found).

To perform an exact-match search:

- use **HReadSeekFirst**.
- use the *hIdentical* constant associated with **HReadSeek**.

## FOR EACH browse

The FOR EACH browse is used to entirely browse a data file, a query or a HyperFileSQL view.

The browse can be performed:

- on the best key of the data file, automatically detected by the HyperFileSQL engine.
- on a specific key.

For example, the following code is used to browse the Customer file according to the "Name" item, from the first record to the last one.

```
FOR EACH Customer ON Name
 // Process the records
END
```

This code is equivalent to the code presented for the *HReadFirst* browse:

```
HReadFirst(Customer, Name)
WHILE NOT HOut(Customer)
 // Process the records
 HReadNext(Customer, Name)
END
```

The reading of the file can be performed from the last record to the first one by specifying the direction:

```
FOR EACH Customer ON Name FromEnd
 // Process the records
END
```

## FOR EACH WITH browse

The FOR EACH WITH browse is used to browse a HyperFileSQL data file according to a condition.

The browse can be performed:

- on the best key of the data file, automatically detected by the HyperFileSQL engine.
- on a specific key.

For example, the following code is used to browse the Customer file according to the customer number and by specifying a condition on the customer name.

```
// Browse with filter
FOR EACH Customer ON CustomerNum WHERE "Name = 'Smith'"
 // Add customers into the list
 ListAdd(CustomerList, Customer.CustomerNum)
END
```

This code is equivalent to the code presented for the *HReadSeek* browse:

```
HReadSeek(Customer, Name, "Smith")
WHILE HFound(Customer)
 // Process the record
 HReadNext(Customer, Name)
END
```

## What type of browsing command should I choose?

All the types of browsing commands are equivalent. The type of browse depends on the keys found in the data file, on the size of the file, on the type of search performed ("Contains" search or not).



## LESSON 4.3. ERROR MANAGEMENT

### This lesson will teach you the following concepts...

- The automatic management of errors
- The programmed management of errors



Estimated time: 20 min

### Overview

When managing a database, several types of errors may occur when adding or modifying a record:

- Password error on a data file
- Error of mandatory input
- Duplicate error
- Integrity Error, ...

WinDev proposes several modes for managing these errors:

- **the automatic mode:** a specific window is displayed to the user whenever an error occurs when managing the database records. This window allows the user to modify his data.
- **the advanced programmed mode:** a custom procedure or window for error management is called whenever an error occurs when managing the database records.

Let's take a look at these modes for error management via a simple example. This example includes two data files (Product and VAT) allowing us to test the duplicate errors and the integrity errors.

- Open the "WD Managing HFSQL errors.WDP" project. To do so:
  1. Close (if necessary) the current project to display the home window.
 In the home window, click "Tutorial" and select the project named "Management of HyperFileSQL errors".
 

Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Management of HyperFileSQL errors".

### Managing the errors in automatic mode

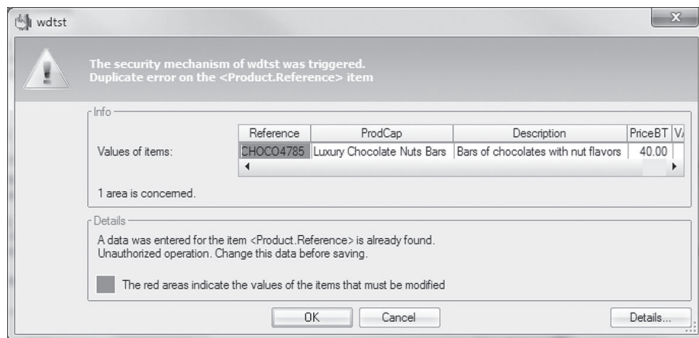
The automatic mode is enabled by default. In this mode, WinDev proposes standard windows for each type of error. These windows indicate the problem to the user.

No specific code is required in the application.

Let's now test the type of error that can be displayed in our simple example.

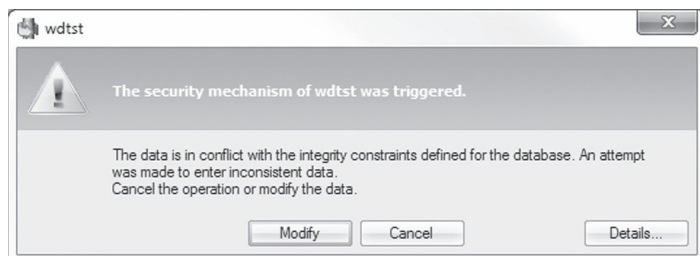
- Click the "GO" button (found with the quick access buttons) to run the "WD Managing HFSQL errors" project.
- Click "Manage the duplicate and integrity errors in automatic mode".
- We are now going to test the duplicate errors. To do so, click the "Duplicates Fill the controls" button. A new product is automatically entered in the form at the bottom of the window.

- Click the "OK" button to add data to the database. The following screen is displayed:



This window explains the problem: data that already exists was entered for the "Reference" item. A duplicate error occurs. This window allows the user to correct the value. The "OK" button lets you validate the input in the error window. The "Cancel" button is used to go back to the screen that triggered the error. Click the "Cancel" button.

- Let's now test the integrity errors. To do so, click the "Integrity: Fill the controls" button.
- Click the "OK" button to add data to the database. The following screen is displayed:



This window indicates that an integrity error occurred (in our case, no VAT rate was entered). The "Modify" button is used to go back to the screen currently in edit.

In the automatic mode, the help windows are displayed as soon as a database management error occurs. In some cases, it may be interesting to manage these errors manually. In this case, the developer can take a more precise action on the errors that occurred.

- Click the "Cancel" button twice. The menu of the example is displayed.

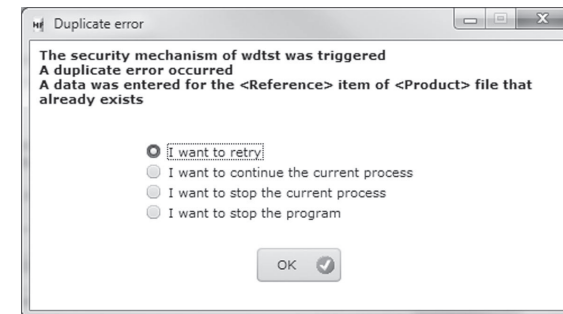
## Managing the errors in advanced programmed mode

The management of errors in advanced programmed mode consists in customizing the error process by programming. WinDev gives you the ability to use:

- a custom window for managing the errors.
- a specific procedure.

In our example, we will be using a specific window.

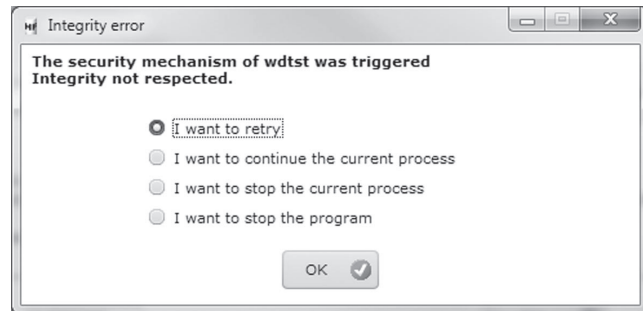
- Click the "GO" button (found with the quick access buttons) to run the "WD Managing HFSQL errors" project.
- Click "Manage the duplicate and integrity errors in advanced programmed mode".
- We are now going to test the duplicate errors. To do so, click the "Duplicates: Fill the controls" button. A new product is automatically entered in the form at the bottom of the window.
- Click the "OK" button to add data to the database. The following screen is displayed:



This window displays the reason of the problem with a custom interface. Select "I want to continue the current process" and validate.

- Let's now test the integrity errors. To do so, click the "Integrity: Fill the controls" button. A new product is automatically entered in the form at the bottom of the window.

- Click the "OK" button to add data to the database. The following screen is displayed:



Select "I want to continue the current process" and validate.

- Let's take a look at the code that was used to display these messages:
1. In the editor, open the "WIN\_Program\_Errors" window (double-click the name of the window in the Project explorer pane).
  2. Display the processes of the window ("Code" from the popup menu of the window). The initialization code of the window contains the following code lines:

```
HOnError("...", hErrDuplicates, WIN_Duplicate)
HOnError("...", hErrIntegrity, WIN_Integrity)
```

Once again, **HOnError** is used to configure the mode for managing the database errors. The parameters of the function are used to specify the range of the modification:

- "\*" to indicate that all the data files found in the analysis are affected.
- the **hErrDuplicates** constant to specify that only the duplicate errors are affected. Then, the name of the window to use must be specified (WIN\_Duplicate).
- the **hErrIntegrity** constant to specify that only the integrity errors are affected. Then, the name of the window to use must be specified (WIN\_Integrity).

3. Close this code window.

We won't go into details about the code of the windows used. See the online help and the "WD Detecting Errors" example (supplied with WinDev) for more details.

## LESSON 4.4. DATA ENCRYPTION

### This lesson will teach you the following concepts...

- What is data encryption used for?
- Using data encryption.



Estimated time: 15 min

## What is data encryption used for?

One of the WinDev features is to guarantee the confidentiality of the data found in the HyperFileSQL files.

This confidentiality is guaranteed by the encryption of the data files.

The encryption of the data files is used to make the content unreadable to any person who does not own the decryption key ("Open Sesame!"). Indeed, a data file is encrypted according to a key (also called "password").

This password is (and must be) known by nobody but you.

When a data file is described in the data model editor, you have the ability to specify whether an encryption must be performed:

- on the data file itself
- on the index
- on the memo files linked to the data file.

Different types of encryption are available:

- 128-bit encryption (high performance)
- RC5 encryption (128 bits, 12 rounds, less performance but more secured)
- RC5 encryption (128 bits, 16 rounds, less performance but more secured)

The specified key (or "password") will be valid for the data file only (and not for the entire analysis). You have the ability to define an encryption for all the data files or for some of the data files found in your analysis.

The data found in the file (and/or in the index and/or in the memo files) will be "encoded" according to the key (or "password") that was defined when generating the files.

When decoding the data file, the password can be:

- requested from the user via a dialog box automatically managed by WinDev. In this case, the user must know the password in order to use the data file.
- found in the code of the program. Any person who owns the program will be able to use the data file, but only for the processes defined by yourself. In this case, the user will not know the password.
- enabled by an advanced operating mode: password read in a file, enabled by a specific process...

The password must be managed by the developer, in other words you! WinDev takes care of encoding and decoding based on the password you specify.

## How do I encrypt the data files?

### Implementing the encryption of the data files

We're going to set up data encryption for one of the examples provided in this tutorial, "Wd HFSQL Error Management".


- ▶ Open (if necessary) the project named "WD Managing HFSQL errors.WDP":

1. Close (if necessary) the current project to display the home window.

In the home window, click "Tutorial" and select the project named "Management of HyperFileSQL errors".

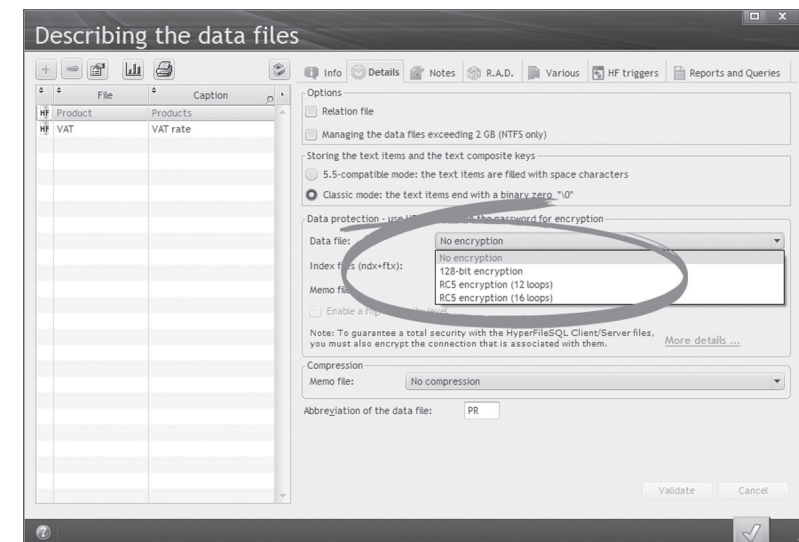
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Management of HyperFileSQL errors".

- ▶ To use the encryption on data file:

1. Load your project's analysis in the data model editor: click  among the quick access buttons of the WinDev menu.

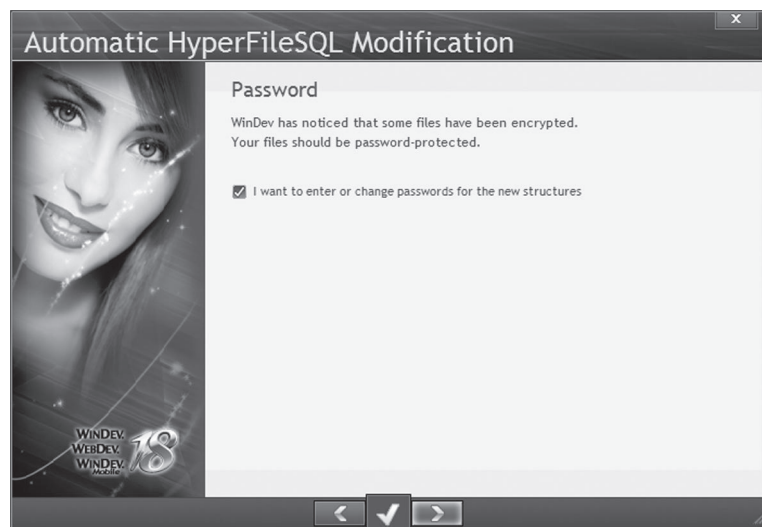
2. Select the data file that must be encrypted. In our example, we're going to encrypt the "Product" data file.

3. In the file description ("Description of data file" option of the popup menu, "Details" tab), choose the encryption option for your data file, your memo file or your index file.



You also have the ability to enable a higher security. In this mode, the encryption password must be entered whenever an automatic data modification is performed (new setup, update, generation...).

When generating your analysis, the wizard proposes to enter or to modify (if your data file was already encrypted) the encryption password.



The list of data files for which an encryption was requested is displayed. You have the ability to select the data files to encrypt and to enter the passwords for each one of the data files.



## Managing the encryption in WLanguage

To manage an encryption password in WLanguage, you can:

- indicate the password with the functions for opening and creating the data files (**HCreation**, **HCreationIfNotFound**, **HOpen**).  
Example for using **HCreationIfNorFound**

```
HCreationIfNotFound("Product", "Password")
IF HErrorPassword() THEN
 Error("Invalid password")
END
```

- use **HPass** before the first read or write operation performed in the encrypted data file.  
Use example for the **HPass** function:

```
// Open a file with password and error check
HPass(CUSTOMER, "Password")
HCreationIfNotFound(Product)
IF ErrorOccurred THEN
 Error("HyperFileSQL error: " + HErrorInfo())
RETURN
END
```

See the online help (keyword: "HPass", "FicCryptMethod", "MmoCryptMethod", "NdxCryptMethod") for more details.



## LESSON 4.5. ADVANCED CONCEPTS

### This lesson will teach you the following concepts...

---

- The log process
- Transactions
- The replication
- Triggers



Estimated time: 15 min

### Overview

---

This lesson is going to present important concepts for database management. The goal of this lesson is to present these concepts and their implementation with a HyperFileSQL database.

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

### The log process

---

The log file is a specific file that will allow you to fulfill the following requirements:

- restore the database to its previous state following a physical problem, errors in processes or errors in input, ...
- keep track of the operations performed on a "sensitive" data file (who modified or read the file for example).
- manage the replication (see "The replication", page 283)

The log file acts as a security mechanism for your database. It enables you to quickly restore your database to its last status before the problem occurred, since the last backup, without having to re-enter anything.

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

### Transactions

---

A transaction is a set of indissociable operations performed on one or more data files:

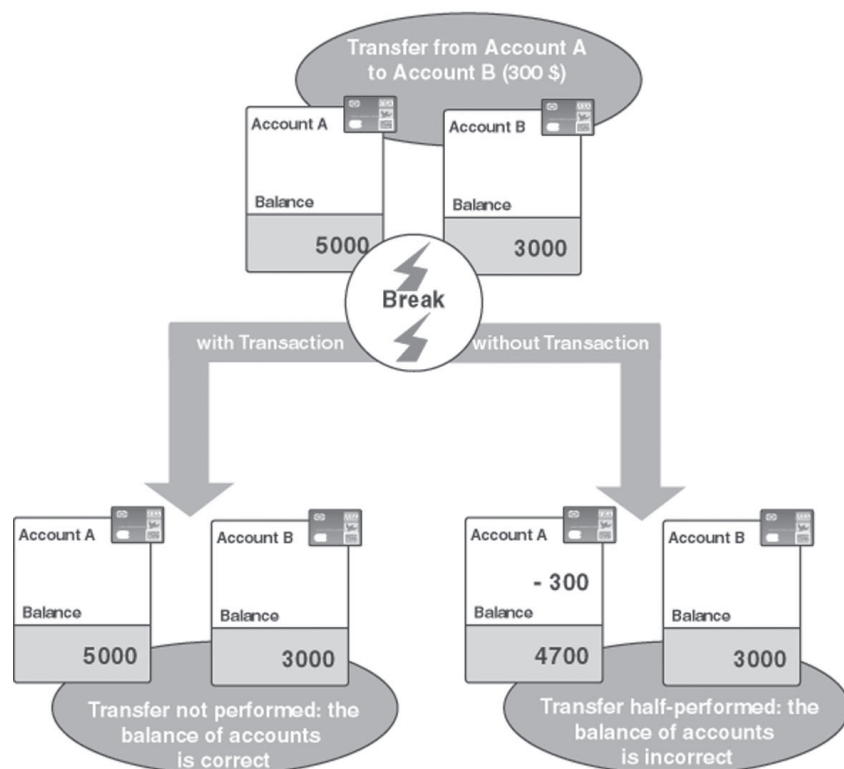
- either all the operations of the transaction are performed.
- or no operation is performed.

The management of transactions is the best way to ensure the integrity of a set of indissociable write operations performed on HyperFileSQL data files.

The transactions are also a reliable way for securing your processes on HyperFileSQL files.

### A simple example of the need for transactions in some types of processes

During a bank wire transfer, an account is debited while another account is credited. A failure may occur on the network while the operations are performed (power outage, computer crash). This is the reason why this process must be managed by a transaction. If a breakdown occurs, all the operations to perform are canceled. This way, you don't end up with an account being debited without the other one being credited!



### The transactions on HyperFileSQL

Each write operation performed during a transaction is stored in a specific file. The transaction can be canceled at any time: all the operations performed since the beginning of the transaction will be canceled.

The transactions are canceled in the following cases:

- program error.
- end of program.
- power failure or application shutdown.

When the application is restarted, the consistency of the database can be restored:

- via "WDTRANS".
- during the first call to **HTransactionStart**.
- during the first call to **HTransactionCancel**.

Once the write operations included in the transaction are completed, the program can validate the operations of the transaction.

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

### The replication

The data replication is a very powerful feature. The replication is the operation allowing the maintenance of remote databases with identical structures. Each one of these databases evolves independently.

Via the replication, the operations performed on each one of the databases are applied to all the other databases.

WinDev enables you to easily perform these operations.

Two types of replication are available:

- **The logged replication** (based on the log process). This type of replication is used to replicate the HyperFileSQL databases between themselves. This type of replication can be implemented by the WLanguage functions or by WDReplic.
- **The universal replication** that is used to replicate any type of database (a HyperFileSQL database with an Oracle database for example).

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

### Triggers

A trigger is used to automatically trigger an action when an addition, a modification or a deletion is performed in a HyperFileSQL data file.

During an operation (addition, modification or deletion) on a HyperFileSQL data file, an action can be run before or after the execution of this operation. For example, when performing a deletion in a data file, you have the ability to ask for the confirmation before deleting the record.

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

## LESSON 4.6. EXTERNAL DATABASES

### This lesson will teach you the following concepts...

- Configuring the connection to a database in the analysis
- Configuring the connection to a database by programming



Estimated time: 15 min

### Overview

Any database management system other than HyperFileSQL will be called an "External database". Any program created in WinDev can easily access a database other than HyperFileSQL. To do so, all you have to do is describe the database in an analysis (as for HyperFileSQL).

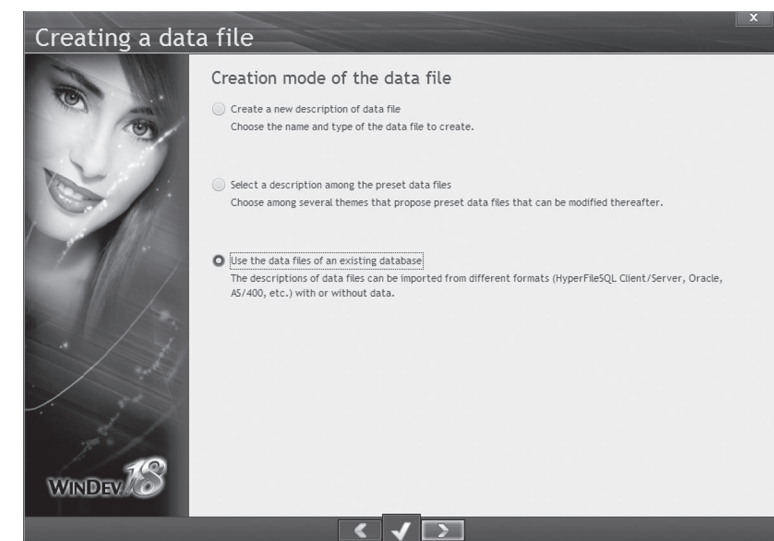
The modes for accessing the databases are:

- native access
- access via ODBC
- access via OLE DB
- ODBC access via OLE DB

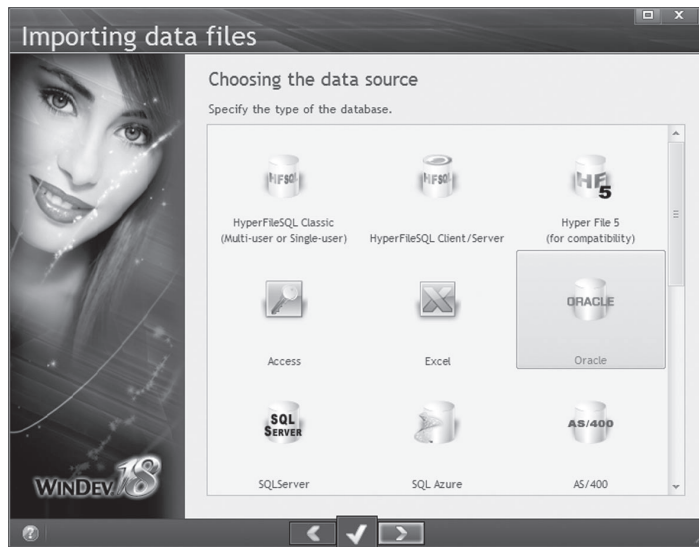
The programming will be performed by the functions of WLanguage (HRead\*) and/or by using the SQL language.

### Connecting to a database in the data model editor

When creating a new file ("New file" button on the "Analysis" pane), choose the "Use data files from an existing database" option:



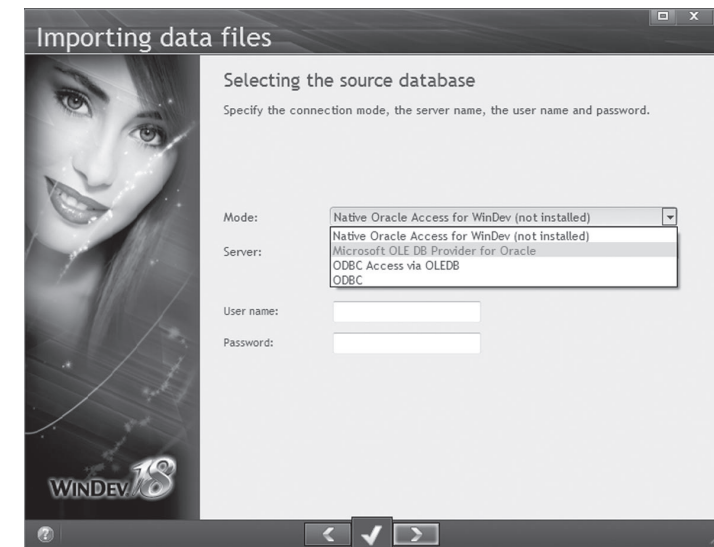
Then, select the type of the database:



Specify whether the data will be converted to HyperFileSQL Classic or whether it will remain in its current format:



The parameters for connecting to this external database must be described (name of server or service, database, login and password, type of access Native, OLE DB ODBC).



This information is used to define a connection in the analysis. This connection will be automatically used when generating RAD (full application RAD or window RAD). All the data files will be linked to this connection.

The use of a connection specified in the analysis is very convenient for the development of the application.

However, in most cases, the parameters used during the development correspond to the parameters of an administrator while the client application must use a more secure connection mode (with less rights).

Therefore, the connection must be redefined. Two solutions are available:

- Solution 1: Redefining the specified connection in the analysis. This solution forces you to keep "hard-coded" and non-configurable data in the application.
- Solution 2: Redefining the specified connection by programming. This solution forces you to use **HOpenConnection** and **HChangeConnection**.

## Connecting to a database by programming

Connecting to a database by programming is performed by two WLanguage functions:

- **HOpenConnection**, which allows you to describe the different connection parameters and to open the connection.
- **HChangeConnection**, which allows you to associate a connection with the data model files.

Let's study the operations required to implement the connection by programming.

### Creating the connection: HOpenConnection

The easiest method for using **HOpenConnection** is to use the wizard of this function: answer the different questions and the corresponding code is automatically generated in your process.

To use the wizard of **HOpenConnection**:

1. In the code editor, type the name of the function followed by an opening bracket:

```
HOpenConnection (
```

2. Click the "Wizard for function HOpenConnection" option that comes up. The wizard is automatically started. The wizard asks for the different parameters of the connection:

- Connection name
- Provider
- Encryption
- User and password
- Server and database



Caution!

The user and the password specified in the wizard will be displayed in the generated code and they will be "hard-coded".  
You may have to create a login window in order to identify the user and to use variables for the user name and password. These operations can be performed afterwards.

3. The wizard is over. Validate. The corresponding code is automatically included in the current process.

The function named **HOpenConnection** was used to define a connection but this connection is not yet associated with the data files.



Caution!

Remember to test the result of the **HOpenConnection** function. This function returns False if the connection cannot be opened.

### Associating the connection with the data files: HChangeConnection

To associate a connection with a file, all you have to do is use **HChangeConnection**. The syntax of this function is quite simple:

- the first parameter corresponds to the name of the file ("\*" for all the data files found in the analysis).
- the second parameter corresponds to the name of the connection to use

Example:

```
HChangeConnection ("*", "MyNewConnection")
```



Notes

You can describe as many connections as necessary.  
A file can use a connection that differs from the connection used by the other files found in the analysis.



PART 5

Managing a  
HyperFileSQL  
Client/Server  
database



DEVELOP 10 TIMES FASTER 

## LESSON 5.1. INTRODUCTION

**This lesson will teach you the following concepts...**

---

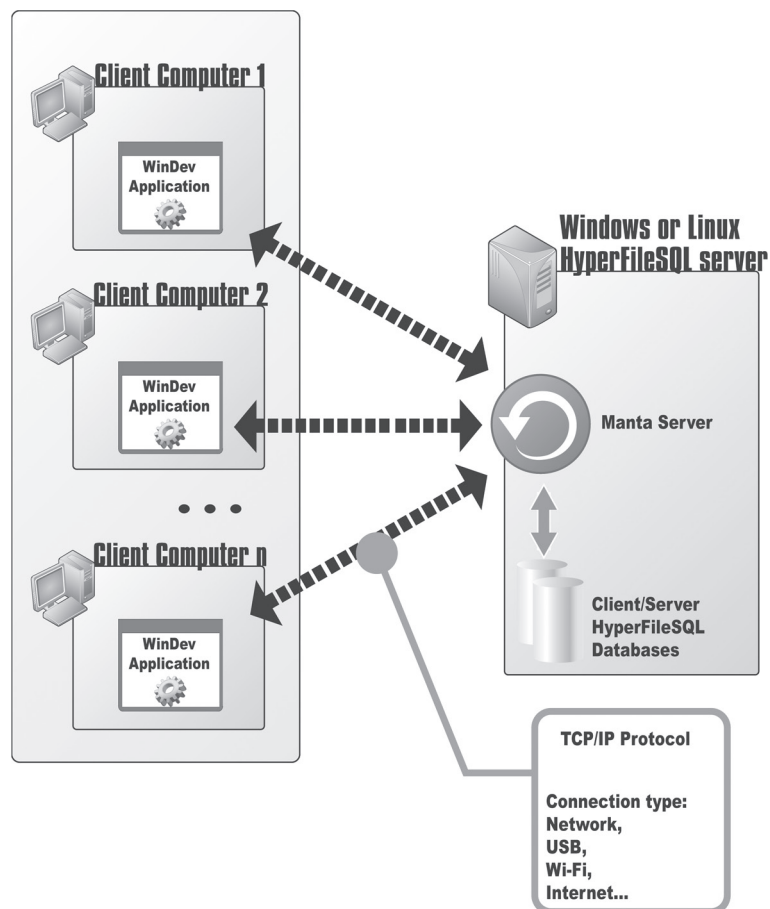
- Client/Server principle
- Why switch an application to HyperFileSQL Client/Server?



Estimated time: 5 min

## Overview

WinDev enables you to create applications that access the HyperFileSQL Client/Server databases. A HyperFileSQL Client/Server application runs an application on several computers (called client computers) and stores the databases and processes on a server. This operating mode makes response times faster and more reliable, and it simplifies the maintenance of the database.



WinDev allows you to:

- create a HyperFileSQL Client/Server application from scratch.
- modify an existing application into a HyperFileSQL Client/Server application.

## Why switch an application to HyperFileSQL Client/Server?

The main benefits of an application in HyperFileSQL Client/Server compared to an application in HyperFileSQL Classic:

- The use of HyperFileSQL Client/Server is more secured (use of a login and password and definition of the 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 by an Internet connection.

## LESSON 5.2. IMPLEMENTING A CLIENT/SERVER APPLICATION

### This lesson will teach you the following concepts...

- Installing a local HyperFileSQL server
- Creating an application in Client/Server mode
- Adapting an application to support the Client/Server mode
- Features available in Client/Server mode



Estimated time: 10 min

### Overview

In this lesson, we are going to perform all the operations required to develop and deploy a HyperFileSQL Client/Server application.

### Installing a local HyperFileSQL server

The first operation before developing a HyperFileSQL Client/Server application consists in installing a HyperFileSQL 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 setup program of the HyperFileSQL server is available from the WinDev CD. If you do not own this CD, the setup of the HyperFileSQL server is also available from our Web site ([www.windev.com](http://www.windev.com)).

To install the HyperFileSQL server locally:

1. Start the setup program of WinDev.
2. Choose "Install the HyperFileSQL C/S server".
3. Then, select "Install a HyperFileSQL Client/Server server".
4. Accept the license agreement.
5. Choose the platform ("For Windows on this computer").
6. Choose "Install a new server".
7. Select the setup directory and specify the name of the server and the port. Port 4900 will be used by default. The HyperFileSQL Control Center can be installed if this one is not found or accessible from your computer.



Caution!

The HyperFileSQL Control Center is required to manage the HyperFileSQL Client/Server database.

8. The setup is performed. By default, to connect to the server in administrator mode, use the "admin" user without password.

### Creating an application in HyperFileSQL Client/Server mode

The method for creating a HyperFileSQL Client/Server application is identical to the method for creating a WinDev application.

You must:

1. Create the project by requesting to create a new database.
2. Create the analysis by specifying that the databases used by the project will be "HyperFileSQL Client/Server" databases.
3. Specify the characteristics of the connection to the HyperFileSQL Client/Server server that will be used.
4. When creating a file in the analysis, indicate that this file is in Client/Server mode and specify the connection used.

## Adapting a HyperFileSQL Classic application to support the Client/Server mode

### Overview

Migrating a HyperFileSQL Classic application to the Client/Server mode is the most common operation.

Several methods can be used to perform this migration on the development computer:


- perform the migration from the data model editor.
- perform the migration from the HyperFileSQL Control Center.

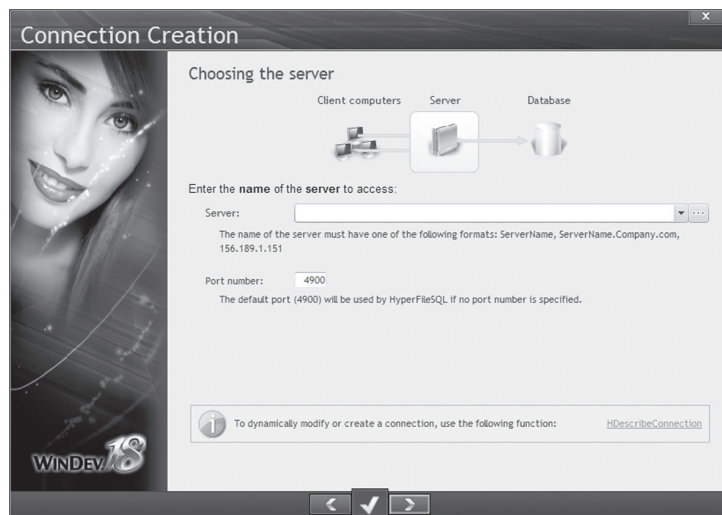
To better understand the different steps, we are going to migrate the application for account management that was created in part 2 of this book by using the first method.

### Migrating the example

A corrected version of the project studied in part 2 is available with this tutorial. We are going to migrate this project and to run its test in Client/Server mode.

To migrate the project:

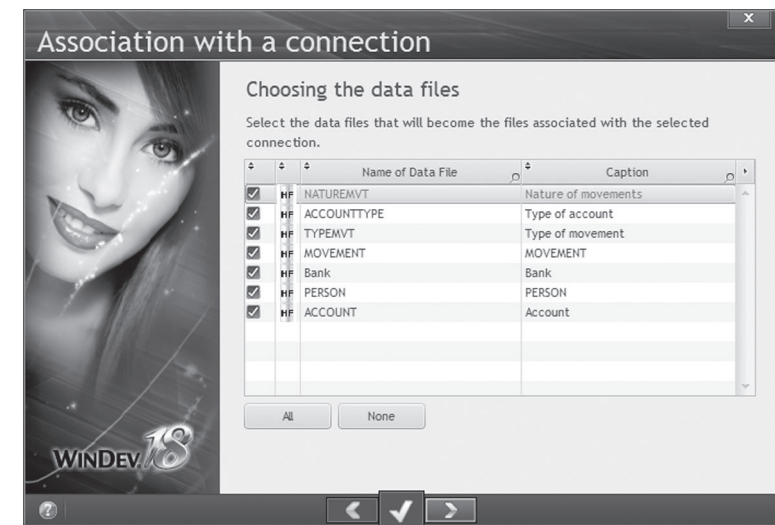
1. Close the current project if necessary.
2. In the welcome window, click "Tutorial" and select "WD Application with data (Answer)".  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "WD Application with data (Answer)".
3. Load your project's analysis in the data model editor: click  among the quick access buttons of the WinDev menu. The data model editor is displayed.
4. On the "Analysis" pane, in the "Connection" group, click "New connection". A wizard comes up, enabling you to create a connection.
5. Select the type of connection to create: "HyperFileSQL Client/Server". Go to the next screen:



6. In the following planes, specify:

- the name of the server (name of your computer for example) and the port number,
- the name of the user and his password (leave this information empty to use the administrator),
- the name of the database ("WD Application with data" in our example)
- the name of the connection (keep the proposed name)

7. Go to the next screen. The connection to the database is automatically created. The wizard proposes to associate the different data files found in the analysis with the created connection. Validate and select all the proposed files:



Go to the next screen.

8. The wizard proposes to copy the data files on the server. Validate ("Copy now").
9. Select the data files that will be copied onto the server: in our case, all the data files found in the EXE directory. Go to the next screen and validate.
10. The data files of the analysis are automatically transformed into HyperFileSQL Client/Server data files and associated with the selected connection.
11. Generate the analysis.



#### Caution!

Check the code of your project: in HyperFileSQL Client/Server mode, **HSubstDir**, **HChangeDir**, ... are useless.



#### Notes

According to the parameters specified when creating the connection, the connection defined in the analysis can be modified by **HOpenConnection** and **HChangeConnection**. This operation was presented in "External databases", page 284.





Notes

**HOpenConnection** can be used to go back to the HyperFileSQL Classic mode: all you have to do is specify the path of the directory containing the HyperFileSQL Classic data files.

12. The development project was migrated. You may also have to migrate the deployed application (if the deployed application uses HyperFileSQL Classic files for example). This operation is configured when creating the setup program of the application.

### Features available in HyperFileSQL Client/Server mode

HyperFileSQL Client/Server proposes several features:

- Transactions
- Logs
- Stored procedures
- Triggers,
- Hot automatic data modification,
- Hot reindexing,
- Scheduled backups,
- Incremental backups,
- Universal replication.

We shall not describe these features here (some of them have been presented in this tutorial in HyperFileSQL 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 HyperFileSQL Control Center
- Creating a user in the HyperFileSQL Control Center
- Saving the database



Estimated time: 20 min

## Overview

Now that we know how to create and/or migrate an application in HyperFileSQL Client/Server, let's see how the associated database can be managed.

Indeed, a Client/Server database requires:

- a specific configuration of the computers (installing a HyperFileSQL server, ...)
- a management performed via the HyperFileSQL Control Center.

## Configuring the computers

To use a HyperFileSQL Client/Server database, a HyperFileSQL server must be installed on the server. Several HyperFileSQL 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 HyperFileSQL server that includes a test database and a production HyperFileSQL server that uses a different port can be installed on the same computer.

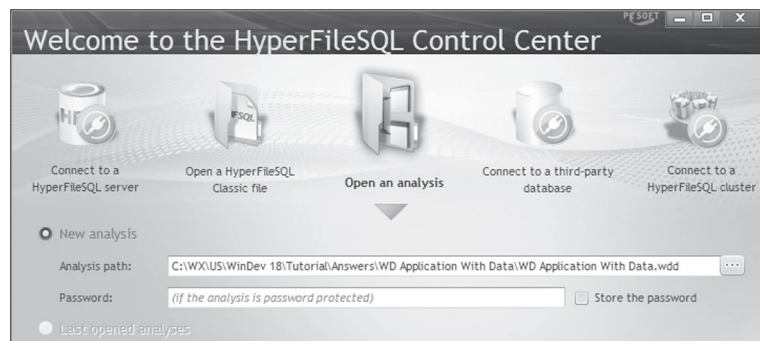
## The HyperFileSQL Control Center

The HyperFileSQL Control Center is used to perform all the operations for managing the HyperFileSQL Client/Server servers and databases.

We are going to present the most important features.

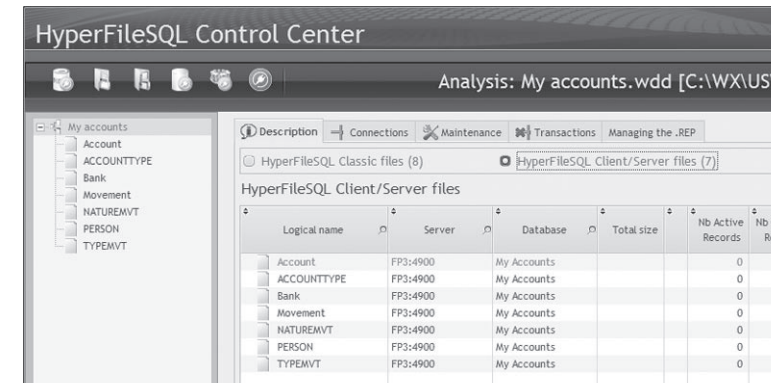
To start the HyperFileSQL Control Center and to access the data:

1. In the WinDev menu, on the "Tools" pane, in the "Database" group, click "HyperFileSQL". The home window of the HyperFileSQL Control Center is displayed. The analysis of the current project is automatically selected.



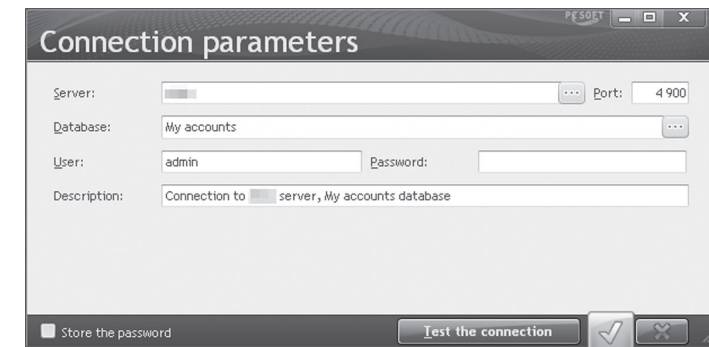
Validate the screen. The HyperFileSQL Control Center is displayed.

2. Click "HyperFileSQL Client/Server file". The list of data files in HyperFileSQL Client/Server format is displayed.



The Control Center lists the data files found in the analysis linked to the current project. No connection is established.

3. To view the data files, double-click one of the data files in the list on the left. The HyperFileSQL Control Center displays a connection window used to establish the connection to the HyperFileSQL Client/Server server used.



Specify the password if needed, and validate.

4. The information about the different data files that use this connection is displayed.
  - The "Content" tab displays the records found in the data files.
  - The "Description" tab gives information about the data files (number of records, ...).

The entire HyperFileSQL Client/Server database can be managed from the HyperFileSQL Control Center.

## Creating a user account in the HyperFileSQL Control Center

A single user account is created when installing a HyperFileSQL server and when creating a database: the administrator account ("Admin" login without password).

Using a user account enables you to secure the access to the data. Indeed, all the users of the application are not administrators. Specific rights can be granted to each user (or group of users).




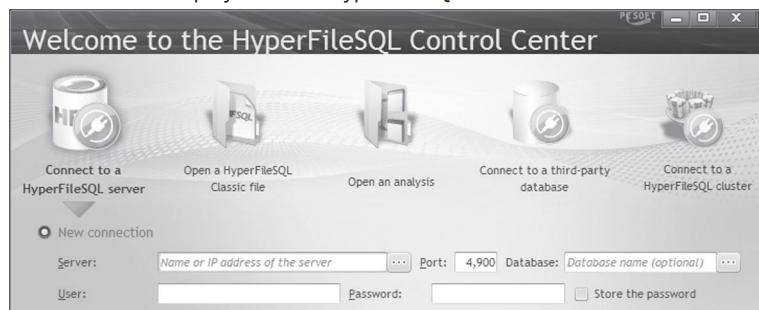
Caution!

The user rights specified in the HyperFileSQL Control Center are granted for the database (and not for the application).  
Don't 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 files for example.

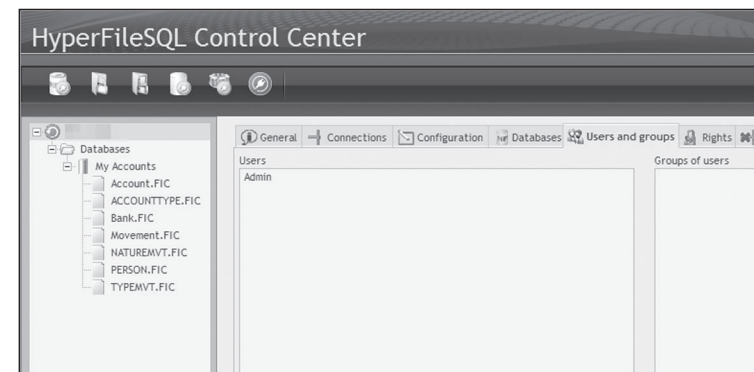
To run a simple test, we are going to create a user and allow this user to view the records found in the Account file.

1. In the HyperFileSQL Control Center, click  and validate the closing of connections if necessary.
2. The home window is displayed in the HyperFileSQL Control Center.

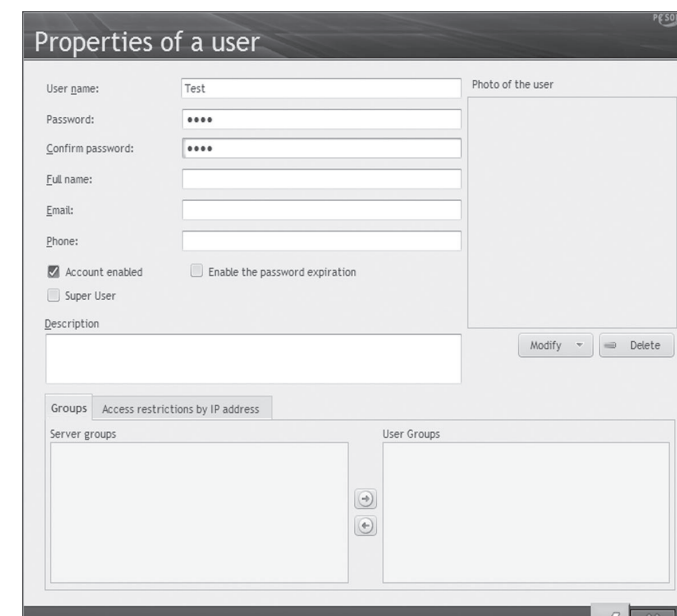


3. The "Connect to a HyperFileSQL server" option is selected by default. Indicate the characteristics of the server installed in the previous lesson.
4. The characteristics of the HyperFileSQL server are displayed:
  - in the panel on the left, the computer, the name of the HyperFileSQL server and the list of databases found on this server are displayed.
  - in the right section of the screen, the different tabs used to manage the HyperFileSQL server are displayed.
5. In the right section of the screen, select the "Users and Groups" tab. This tab is used to manage the users of the server.

Only the "Admin" user exists at this time.



6. To create a new user, click the "New" button (on the left). The screen used to define the characteristics of the user is displayed.  
Enter the following information for instance:



(use "Test" for example as the password).



Notes

Several characteristics can be noticed:

- **Super User:** The users defined as "Super user" are allowed to perform all the actions on the server, on the databases and on all the files.
- **Account enabled:** If this option is not checked, the user exists but he is not enabled (collaborators on holiday for example).
- **Password expiration:** You have the ability to specify a password valid for a few days only.

7. Validate the creation of the user. By default, no rights are granted to this user.

We are now going to grant rights to the user: the user named "Test" can connect to the database and view the ACCOUNT file.

To grant the rights to connect to the database:

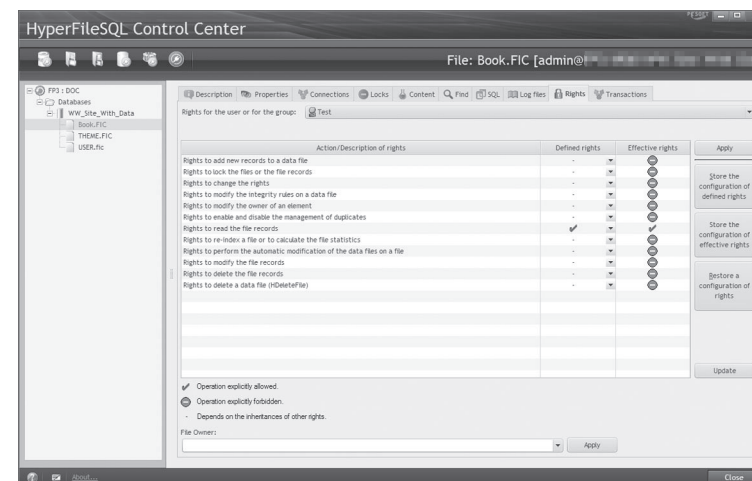
1. In the HyperFileSQL Control Center, select the "WD application with data" database.
2. Click the "Rights" tab.
3. Select the "TEST" user in the list of users.
4. In the list of rights, select "Rights to connect to the server (encrypted and unencrypted connection). Select the green checkmark in the "Defined rights" column.
5. Click the "Apply" button found on the right of the list of rights. The rights become effective.

To grant the rights to read the ACCOUNT file:

1. In the HyperFileSQL Control Center, select the "WD application with data" database, the ACCOUNT file.
2. Click the "Rights" tab.
3. Select the "TEST" user in the list of users.
4. In the list of rights, select "Rights to read the file records" and select the green checkmark in the "Defined rights" column.
5. Click the "Apply" button found on the right of the list of rights. The rights become effective.

Similarly, the rights can be defined:

- on the HyperFileSQL server
- on the database
- on the database files.



In our example, the user named "Test" will be able to browse the records found in the ACCOUNT file. If this user tries to perform another action, a message will be displayed: "The Test user does not have 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).



Notes

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

The backup of the database can be performed via the "Backups" tab. This tab is available when selecting the database in the left pane of the HyperFileSQL Control Center.

## Conclusion

The HyperFileSQL Control Center is a tool for managing the 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 backups of the database.

The HyperFileSQL Control Center is a redistributable tool that can be installed on the computers of the users who work with HyperFileSQL Client/Server databases. The HyperFileSQL Control Center must be used by the person who manages the database.

## LESSON 5.4. SETUP ON THE USER COMPUTERS

### This lesson will teach you the following concepts...

- How do I install a Client/Server application on the user computers?



Estimated time: 5 min

### Overview

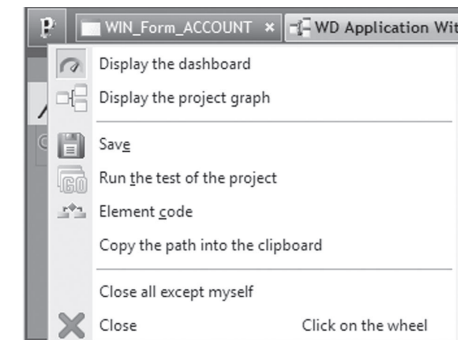
Installing a Client/Server application requires several specific options. These options are taken into account by the wizard for creating the WinDev setup.

We're going to create the setup procedure of our "WD Application with data" application.

### Starting the wizard for setup creation

To start the setup procedure of a HyperFileSQL Client/Server application:

1. Display the project dashboard (right click the "P" icon in the open document bar, and select "Display the dashboard").

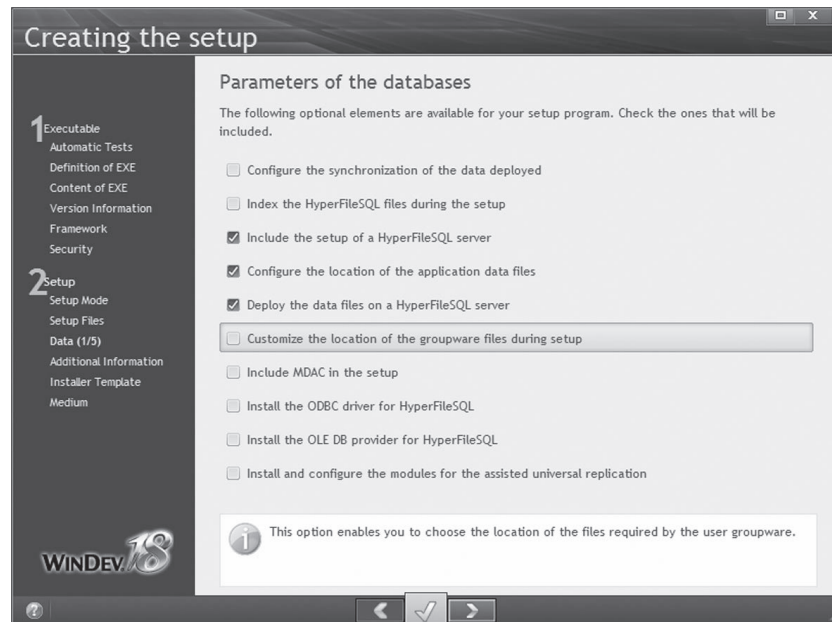


2. On the "Project" pane, in the "Generation" group, click "Setup procedure".
3. Create the executable and validate the creation of the help.



4. Go to the screen named "Parameters of the databases". In this plane, you can:

- Include the setup of a HyperFileSQL Client/Server server
- Configure the HyperFileSQL Client/Server connections
- Deploy the HyperFileSQL Client/Server files



If these options are checked, the following screens are used to configure the different elements that must be taken into account for the setup on the user computers. See the online help for more details.

## PART 6

### Queries



DEVELOP 10 TIMES FASTER PCSOFT

## LESSON 6.1. CREATING A SELECT QUERY

**This lesson will teach you the following concepts...**

---

- Operations performed in the query editor.
- Creating a query step by step.



Estimated time: 30 min

## Introduction

A query is used to define an interrogation on data files. The query editor enables you to create (without programming) the following types of queries:

- the list of suppliers located in a city starting with the letter "P".
- the number of classified ads per city.
- the list of customers who ordered most.

And this, for all types of databases (HyperFileSQL, Oracle, ...) !

We will now take a look at a simple example to explain how the query editor can be used. An advanced use of the query editor will be presented later in this tutorial.

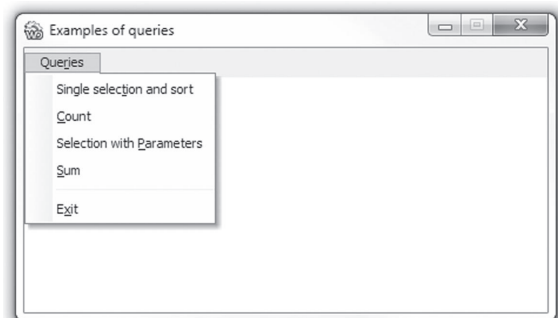
In this lesson, you will learn how to:

- Create a simple query corresponding to a list with selection and a sort.
- Create a count query.
- Create a multi-file query.

► To open the example project:

1. Close (if necessary) the current project to display the home window.
2. In the home window, click "Tutorial" and select the project named "Run queries (Exercise)".  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Run queries (Exercise)".

► Perform a "GO" of the "WD Running queries.WDP" project. The program starts. The following window is displayed.



You can browse the menu options and choose samples of queries.



### Answers

A corrected project is available. This project contains the various queries created in this lesson. To open the corrected project, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Using queries (Answer)".



EXAMPLE


### Example

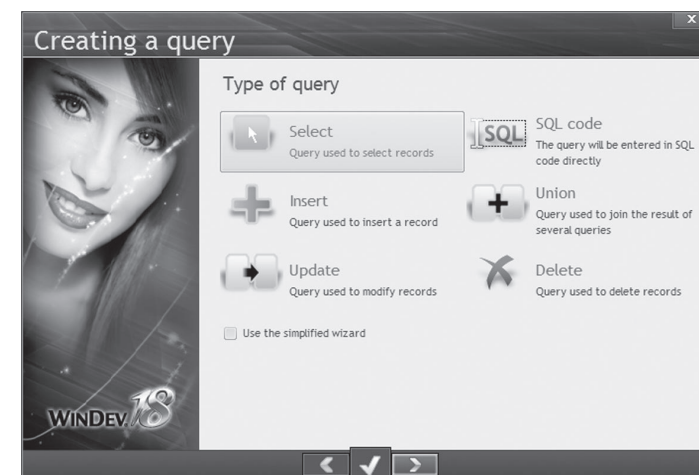
The "Queries" example, supplied with WinDev, presents the use of queries with WinDev. This example is accessible from the "Wizards, Examples and Components" pane of WinDev.

## Your first query

We are going to list the suppliers located in a town whose name starts with the letter "P".

► Create a new query:

1. Click  among the quick access buttons.
2. Click "Query" in the wheel that is displayed. The wizard for query creation starts:



You can use the wizard for query creation or you can enter the SQL code of the query ("Enter the SQL code").

The wizard enables you to create the following types of queries:

- "Select": interrogation
- "Insert": addition into a data file
- "Update": modification in a data file
- "Delete": deletion from a data file
- "Union": combination of Select queries

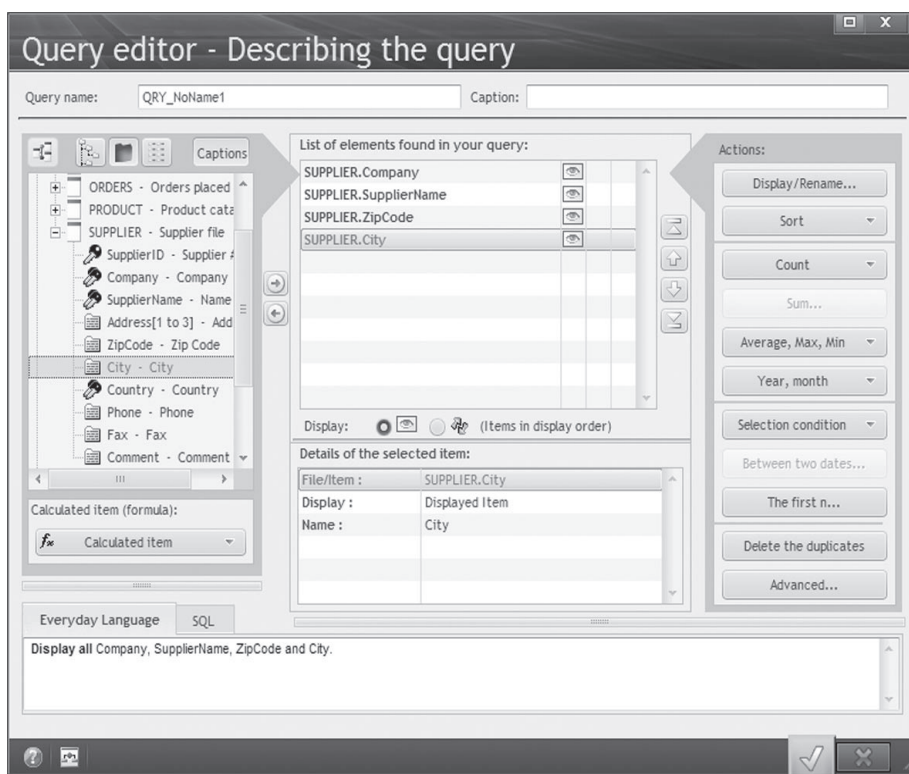


### Caution!

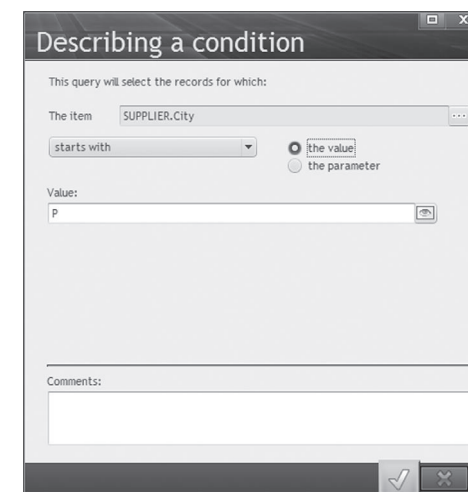
The direct input of SQL code is intended for users who are familiar with the SQL language! Consult the online help to get a detail list of SQL functions recognized by WinDev (keyword: "SQL, SQL Language").

You will learn to use the query editor by performing simple queries: select customers, perform counts... you will have the ability to improve your knowledge by using all types of queries.

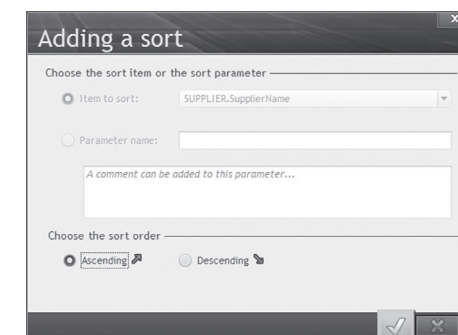
- ▶ To list the suppliers located in a city whose name starts with the letter "P", we must create a Select query. Go to the next screen.
- ▶ Select the items that will be displayed in the result of the query.  
In the left-side list:
  1. Double click the "SUPPLIER" file.
  2. Double-click the "Company", "SupplierName", "ZipCode" and "City" items. These items are displayed in the middle of the window.



- ▶ To find out the suppliers located in a city whose name starts with the letter "P", we are going to apply a selection condition on the city. The selection condition is as follows: "We want the suppliers located in a city starting with P".
- ▶ Select the "City" item in the middle. In the "Actions" to the right, click "Selection condition" then "New condition".  
The "SUPPLIER.City" item is automatically selected.  
Select the condition: "Starts with".  
Select the "the value" option.

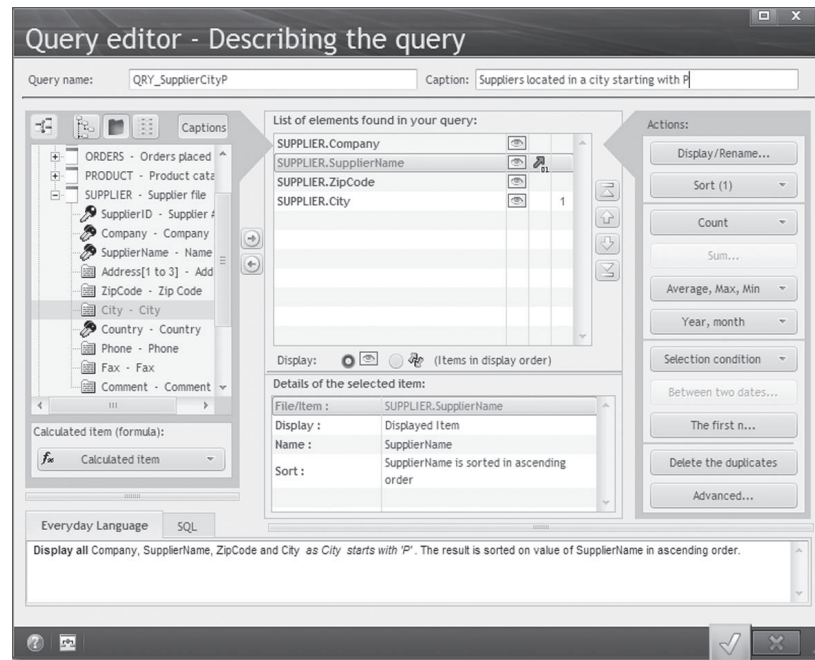


- ▶ Enter the value "P" and validate.
- ▶ We are now going to specify the sort criterion.  
We're going to sort the result by "SupplierName".
- ▶ Select the "SupplierName" item in the middle then click in the "Actions" on the "Sort" button and select "Sort on the selected item".  
We'll sort in ascending order:

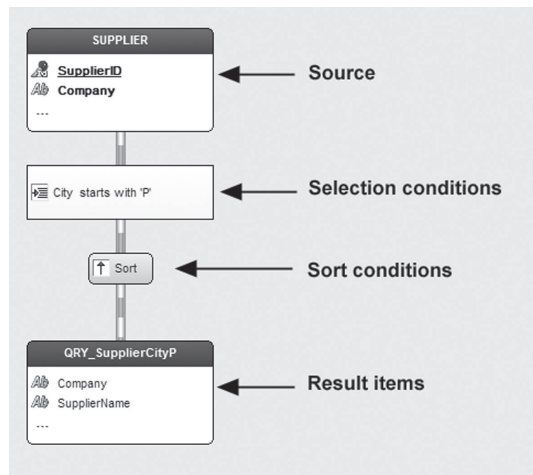



- ▶ Validate. A red arrow numbered 01 is displayed to the right of the "SupplierName" item. This arrow indicates that this item will be sorted in ascending order. The number "01" indicates that it will be the first sort performed.
- ▶ The query is created. It must now be named and saved.
  1. At the top of the screen, enter the name: "QRY\_SupplierCityP".
  2. Enter the caption: "Suppliers located in a city starting with P".
  3. Validate.

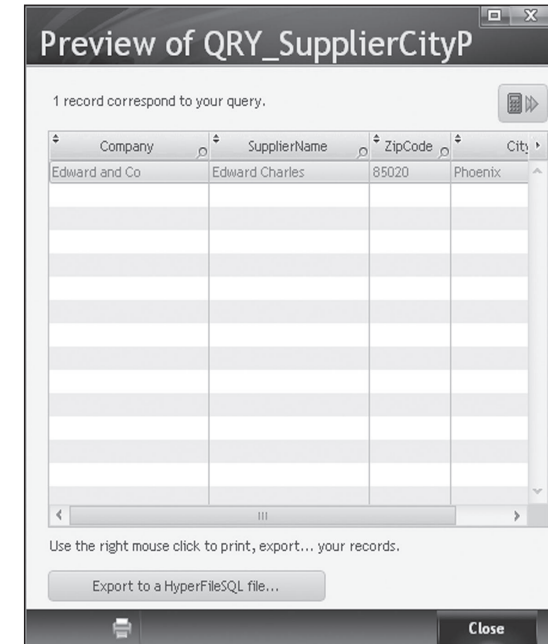




► The graphic representation of the query is as follows:



- Click  (among the quick access buttons) to save the query. Validate the backup window. Let's now run the test of the query.
- Click the "GO" button among the quick access button. The result is displayed in a table:



A popup menu is displayed when you right-click the table displaying the result of the query. The result can be exported to:

- Microsoft Excel.
- an XML file (*eXtensible Markup Language*)
- Microsoft Word.



## LESSON 6.2. QUERIES WITH COUNT

### This lesson will teach you the following concepts...

- Operations performed in the query editor.
- Creating a query with count step by step.




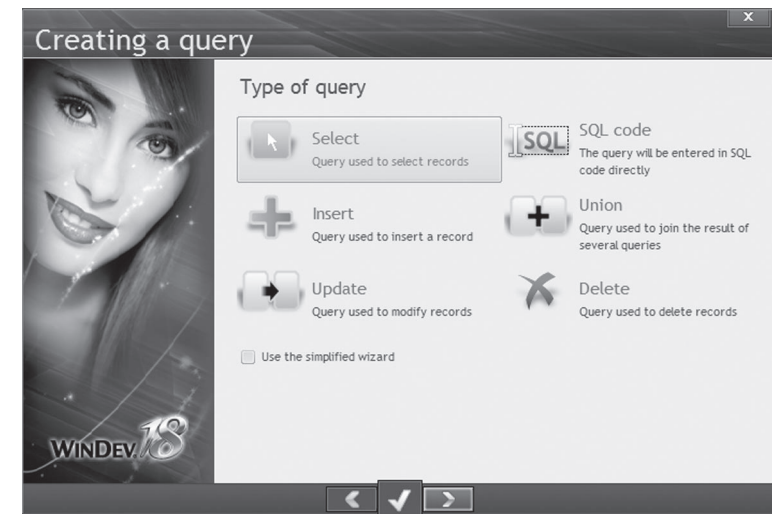
Estimated time: 20 min

### Creating a query with count

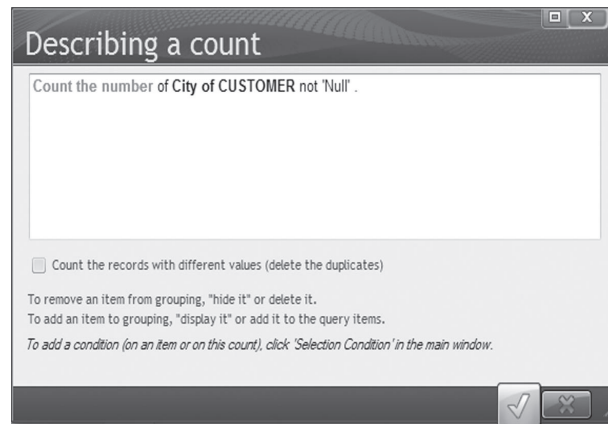
We will now create a query used to perform a count. This query is used to count the number of occurrences found for a value in a data file.

Let's count the number of customers per city.

- ▶ If needed open the example project, "WD Running queries": on the "Home" pane in the "online help" group, expand "Tutorial" then select "Run queries (Exercise)".
- ▶ Create a new query:
  1. Click  among the quick access buttons.
  2. Click "Query" in the wheel that is displayed. The wizard for query creation starts. We are going to create a "Select" query ("Select" option).



3. Go to the next screen. We are going to select the items that must be displayed in the result of the query.
- ▶ We want to display the city. In the list on the left (list of available data files and queries), select the "City" item found in the "CUSTOMER" file (via a double click).
  - ▶ To create a count item:
    1. Select the "City" item (in the middle)
    2. In the "Actions", click the "Count" button.
    3. In the menu, select: "Count on the selected item".
    4. Validate.

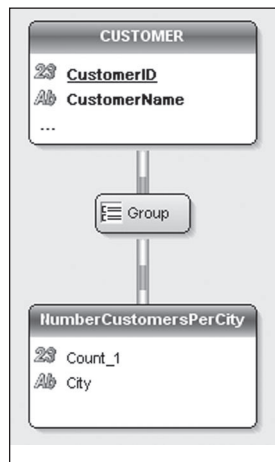


The "Count the records" action is added to the result of the query.



As we want to display all the cities, select the "CITY" item of "CUSTOMER" file once again (via a double click). The item is displayed in the middle of the screen.

- The query is created. It must now be named and saved.
  1. At the top of the screen, enter the name: "QRY\_NbCustomerPerCity".
  2. Enter the caption: "Number of customers per city".
  3. Validate.

The graphic representation of the query is as follows:



The "Count\_1" item contains the count of the above-mentioned records.

- Click  to save the query. Let's now run the test of the query.
- Click .

## LESSON 6.3. SUM QUERIES

### This lesson will teach you the following concepts...

- Operations performed in the query editor.
- Creating a "Sum" query step by step.




Estimated time: 20 min

## Creating a "Sum" query

We are now going to create a query used to perform a sum. We are going to perform a cumulative sum of a value found in a data file.

We are going to calculate the total of orders per customer (gross sales per customer).

- ▶ If needed open the example project, "WD Running queries": on the "Home" pane in the "online help" group, expand "Tutorial" then select "Run queries (Exercise)".
- ▶ Create a new query:
  1. Click  among the quick access buttons.
  2. Click "Query" in the wheel that is displayed.
  3. We are going to create a Select query. Select "Select". Go to the next screen.

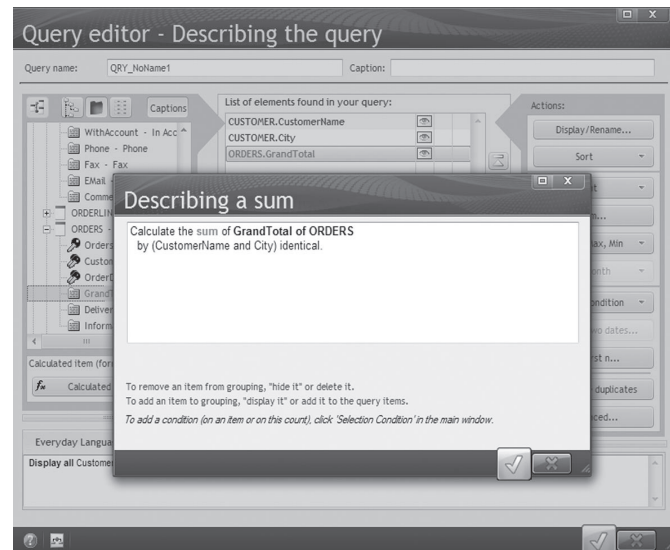
We are going to select the items that must be displayed in the result of the query.

We are going to display the name and city of the customer.

- ▶ On the left, select the "CustomerName" and "City" items found in "CUSTOMER" file (via a double click).

We are going to select the item on which the sum must be performed (GrandTotal in this example).

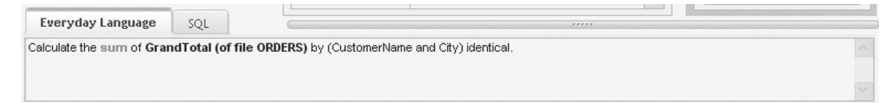
- ▶ On the left, select the "GrandTotal" item found in the "ORDERS" file (via a double click).
- ▶ To calculate the sum of values of "GRANDTOTAL":
  1. Select the "GRANDTOTAL" item in the middle.
  2. In the "Actions" on the right, select "Sum".



3. Validate.

The sum of "GRANDTOTAL" was added to the list of the query result.

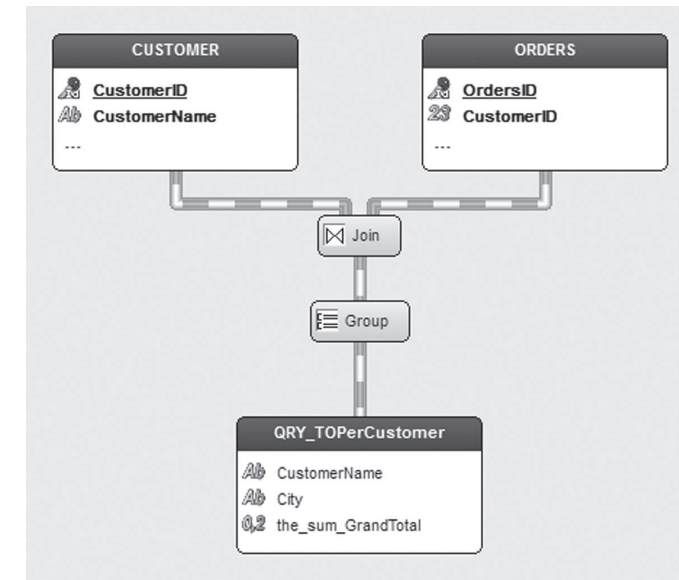
You will notice that the WinDev query editor creates the queries in everyday's language (and also in SQL language).





This gives you the ability to check that your query accomplishes what you set out to do.

- ▶ The query is created. It must now be named and saved.
  1. At the top of the screen, enter the name: "QRY\_TOPerCustomer".
  2. Enter the caption: "Sum of orders per customer".
  3. Validate.

The graphic representation of the query is as follows:



The graphic representation corresponds to a "join". The query uses two data sources (the "CUSTOMER" file and the "ORDERS" file).

- ▶ Click  to save the query. Let's now run the test of the query.
- ▶ Click  to run the test of the query.

## LESSON 6.4. THE SQL LANGUAGE

### This lesson will teach you the following concepts...

- The main commands of SQL language.



Estimated time: 15 min

### Source code of a query

In WinDev, the query is graphically represented in the editor. We shall now discover the source code corresponding to a query. The language used to represent a query is the SQL language. This is just an overview of SQL, not a full course (entire books are devoted to SQL!).

Let's take a look at the SQL code of the query that was created in the previous lesson.

- ▶ Open the query named "QRY\_TOperCustomer".  
If this query was not previously created, open the query named "QRY\_OrderAmountPerCustomer".
- ▶ Display the popup menu of the query's chart and select "SQL Code".

```
SQL Code of QRY_TOperCustomer
SELECT
 CUSTOMER.CustomerName AS CustomerName,
 CUSTOMER.City AS City,
 SUM(ORDERS.GrandTotal) AS the_sum_GrandTotal
FROM
 CUSTOMER,
 ORDERS
WHERE
 CUSTOMER.CustomerID = ORDERS.CustomerID
GROUP BY
 CUSTOMER.CustomerName,
 CUSTOMER.City
```

The query in SQL language contains different keywords:

- the SELECT statement
- the FROM statement
- the WHERE statement
- the GROUP BY statement



Caution!

Make sure that you follow the sequence of commands. This sequence cannot be reversed.

WinDev enables you to easily switch from the graphic description of a query to the SQL code of this query (and conversely).

To display the SQL code of a query created in the query editor, select "SQL Code" from the popup menu of the query's chart.

To display the graphic representation of a query created in SQL code, select "Reverse-analysis of the query" from the popup menu of the query's code.

## The SQL commands

### The SELECT statement

The SELECT statement is used to define the list of items and calculations that must be displayed in the result. The syntax is as follows:

```
SELECT NAMEITEM1, NAMEITEM2...
```

Each item can be renamed (we talk of alias) by using the AS keyword. For example:

```
CUSTOMER.CUSTOMERNAME AS NAME
```

In this example, the *CustomerName* item (found in CUSTOMER file) is renamed to *Name*. *NAMEITEMXX* can be either an item associated with a data file, or a calculation formula.

### The FROM statement

The FROM statement is used to define the list of data files from which the items and the calculations will be checked out. The syntax is as follows:

```
FROM FILE1 , FILE2...
```

*FILEXX* corresponds to one of the data files found in the analysis. You can use **AS** as well.

### The WHERE statement

The WHERE statement contains the selection conditions as well as the join conditions. These conditions are used to filter the records coming from the database files. Only the records that match the conditions will be displayed in the result.

A join is used to associate a record found in a file with a record found in a linked file. For example, a join between the CUSTOMER file and the ORDERS file will be used to find out all the orders of each customer.

The syntax is as follows:

```
WHERE CONDITION1 AND/OR CONDITION2
```

The AND and OR operators are used to perform a logical operation between two conditions.

- The AND operator indicates that the two conditions must be matched at the same time.
- The OR operator indicates that one of the conditions or both conditions must be matched.

The WHERE statement is not mandatory. If this keyword is not specified, all the records found in all the data files will be selected.

### The GROUP BY statement

The GROUP BY statement is used to specify the group items or the group formulas when calculation is performed. For example, in the query that was previously created, the calculation was performed on the sum of orders per customer; which means that the orders were cumulated for each customer. Therefore, the orders were grouped by customer.

The syntax is as follows:

```
GROUP BY ITEM1, ITEM2...
```

*ITEMXX* corresponds to an item from the data files or a formula.

### The ORDER BY statement

The ORDER BY statement is used to specify the sort criteria for the query. The sort criteria will be used to sort the result of the query.

The syntax is as follows:

```
ORDER BY ITEM1, ITEM2...
```

*ITEMXX* corresponds to an item from the data files or a formula.



Notes

The query editor automatically generates the SQL code. This is an easy way to get familiar with this language!

## LESSON 6.5. USING QUERIES

### This lesson will teach you the following concepts...

- Using queries in the reports.
- Using queries in the controls (list boxes, combo boxes, tables, loopers).
- Using queries in WLanguage.



Estimated time: 15 min

### Using a query from a report

You have the ability to choose the data source when creating a report. The data source can be a query, a data file, ... You can create a report that directly operates on the result of a query (for more details, see the lesson about reports, page 335).

### Using a query from a control

When creating a table, a list box, a combo box or a looper, you have the ability to define the data source used to fill the control. This source can be a data file or a query.

### Using a query from WLanguage

A query can also be used like any data file. All you have to do is initialize the execution of your query with **HExecuteQuery**. When you're done reading the file, use **HCancelDeclaration**. This optional function is used to cancel the definition of the query. Your browse can be performed by **HReadFirst** and **HReadNext**:

```
// Initializes the query
HExecuteQuery(MyQuery)
IF ErrorOccurred THEN
 Error("Error while initializing the query"+CR+...
 HErrorInfo())
RETURN
END
// Reading the query's record
FOR EACH MyQuery
 // Process on the query record
END
HCancelDeclaration(MyQuery) //optional
```



Notes

Give the users of your applications the ability to create their own reports and queries (or to modify the existing reports and queries)! All you have to do is supply "Reports and Queries" with your application. See "Distributing "Reports & Queries" with your applications", page 369 for more details.



PART 7

Reports

18



DEVELOP 10 TIMES FASTER 

## LESSON 7.1. YOUR FIRST REPORT

**This lesson will teach you the following concepts...**

---

- Basic vocabulary.
- Creating a report step by step.



Estimated time: 30 min

## Basic vocabulary

Before going into details about printing, we would like first to remind you about basic printing terms!

- **Typeface:** The typeface defines the shape of the character (thickness of the line, spacing between characters, ...). For example, this document uses the "Franklin Gothic" typeface (the programming examples are in "Courier").  
Typefaces are often called **fonts** in everyday's language: we will talk about fonts rather than typefaces.
  - **Style:** The style can be *italic*, bold, underlined, ... It does not modify the height of the characters.
  - **Character height:** the character height is expressed in "Didot" points. A point is equivalent to 0.3759mm. This is the unit used in the printing industry. For example, the height of the characters in this guide is set to 10 points. A height of 10 or 12 points is the standard height for documents. The height of the characters is independent of the printer definition.  
A height of 16 or 24 points can be used to highlight a title.  
A height of 6 points will be hard to read, especially if the printer resolution is poor.
  - **Font:** The font is the shape used to print the text. It corresponds to the combination of typeface, style, height and color of the character.
  - **Printer definition:** Each printer has its own definition. **The higher the resolution is, the better the print will be.** For example, a printer with a resolution of 600 DPI (dots per inch) will produce better printouts than a printer with a resolution of 300 DPI. The type of printing (laser, ink jet, matrix, ...) also affects the quality of the printout. And let's not forget the quality of the paper!
  - **Proportional/Fixed:** A "fixed" font will always present the same character width, regardless of the character. An "i" will have the same size as an "m". A well-known fixed font is "Courier".  
A "proportional" font will use a different size depending on the letters. This documentation uses a proportional font.
- A **Report** is the name given to the representation of a print job. WinDev enables you to create reports. A document is created when the report is run (or edited).

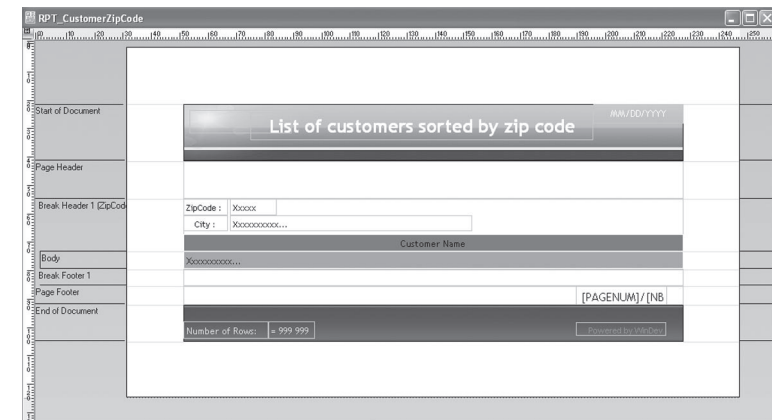
Several types of reports are available:

- Blank
- Form
- Label
- Array
- Crosstab
- Multicolumn report
- Mailing
- Based on form
- Composite
- Organizer or schedule



A **Block** is the name given to a section of the report. A report includes a set of blocks. The different blocks found in a report are as follows:

- start of document
- page header
- break header
- body
- break footer
- page footer
- end of document



Other blocks are available but we won't go into details about them:

- iteration block
- body complement

Some of these blocks are optional. They can be deleted when the report is created. Only the "Body" block is mandatory.

You also have the ability to print reports with bar codes (automatically generated by WinDev), reports with labels, multi-column reports, ...

## Overview

### Printing with the report editor or printing in WLanguage

WinDev proposes two methods for printing:


- the report editor allows you to create reports via a "user-friendly" interface.
- by programming in WLanguage.

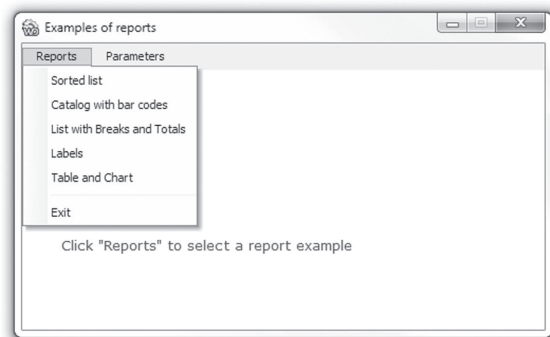
For example, it may be useful to print a list whose data comes from a file.

To print the content of a data file, we recommend that you use the report editor. You will find out later on in this guide how a report can be printed by programming.

Let's take a look at some examples of reports created with the report editor, then we will see how they can be created.

#### Some examples of reports

- Close the current project if necessary. The home window is opened.
- In the home window, click "Tutorial" and select "Print reports (Exercise)". The project is loaded.  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Print reports (Exercise)".
- Run the test of the project (  among the quick access buttons). The program starts. The following window is displayed.



Browse the menu options and check the various report outputs.

- End the test and go back to the editor.



#### Answers

A corrected project is available. This project contains the various queries created in this lesson. To open the corrected project, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Using reports (Answer)".

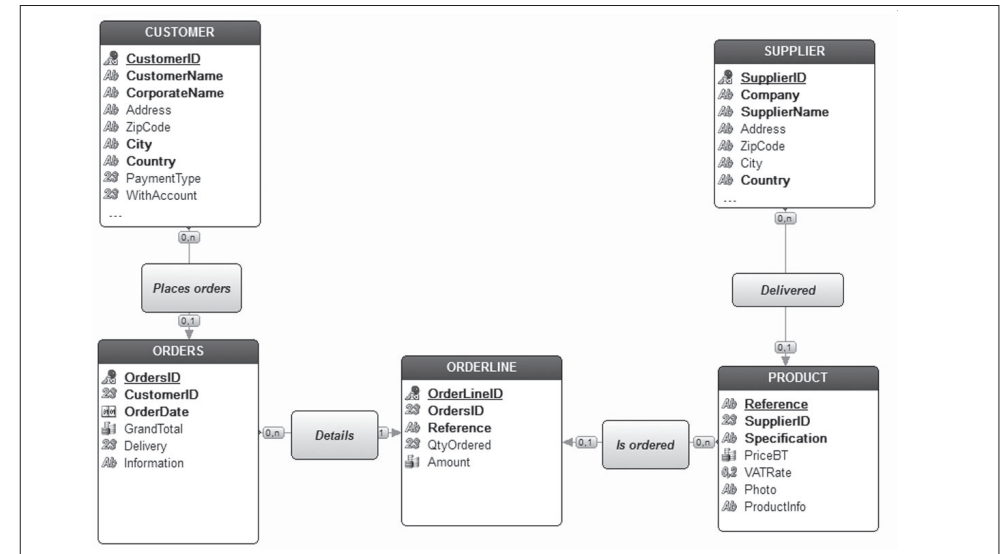
### Let's get down to work now!

To learn how to use the report editor, we will create some printed reports in the project that was just opened.

This project uses data files. Simple data files have been used in our example:

- a CUSTOMER file
- an ORDERS file
- an ORDLINE file
- a PRODUCT file
- a SUPPLIER file

The logical modeling (MERISE) of the data used (called LDM) is presented below:



### The reports to create

To learn how to use the report editor, we will create the following examples:

- List of customers sorted by city
- Printing the orders per customer


## Creating your first report

### Creating the report: List of customers sorted by city

We are going to print the list of customers sorted by city. To do so, we will use a table report in order to clearly represent a list of data.

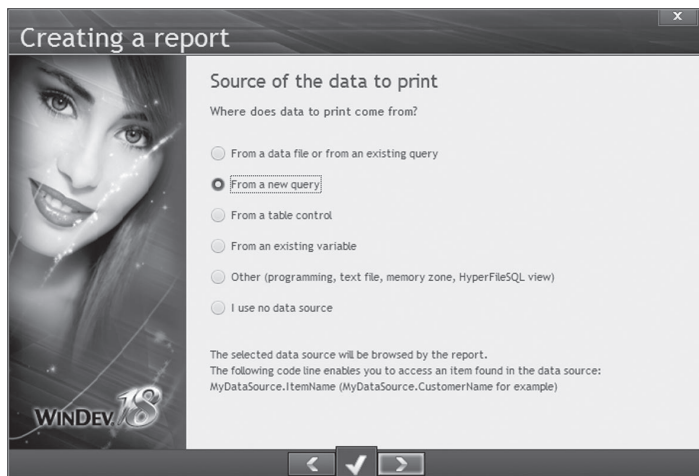


► To create a report:

1. Click  among the quick access buttons.
2. Click "Report" in the wheel that is displayed. The report creation wizard starts. The report creation wizard proposes several types of reports:



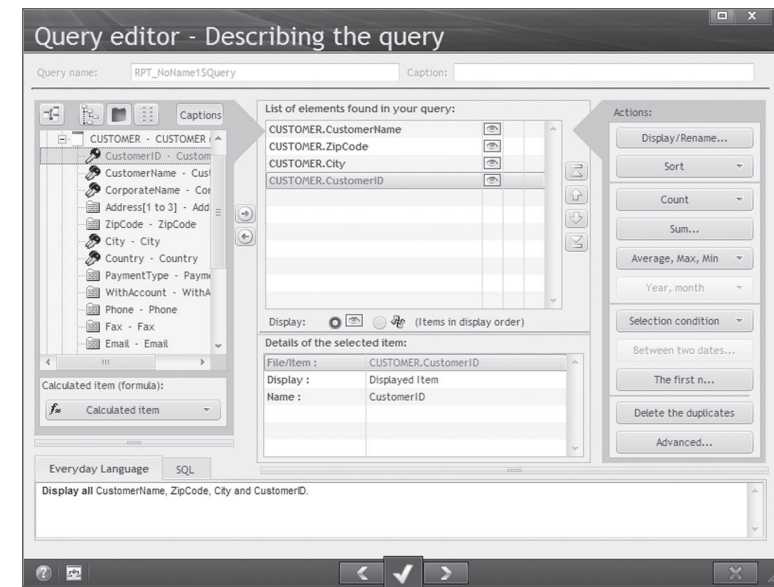
- Select "Table". Go to the next screen.
- Select the data source of the report.



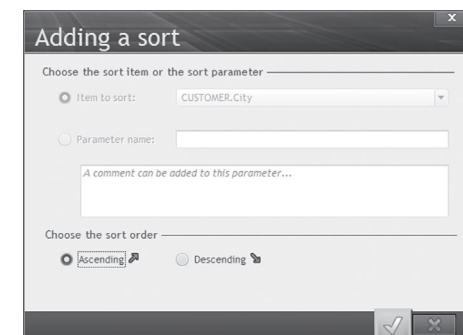
The report will be based on a query that does not exist yet. Select "From a new query". Go to the next screen.

- The query creation wizard starts. This screen is used to select the files containing the data that will be printed. We are going to print the data coming from the CUSTOMER file.

- Click the "+" sign found to the left of the CUSTOMER file. Select the items that will be printed (via a double click):
- CustomerName
  - ZipCode
  - City
  - CustomerID



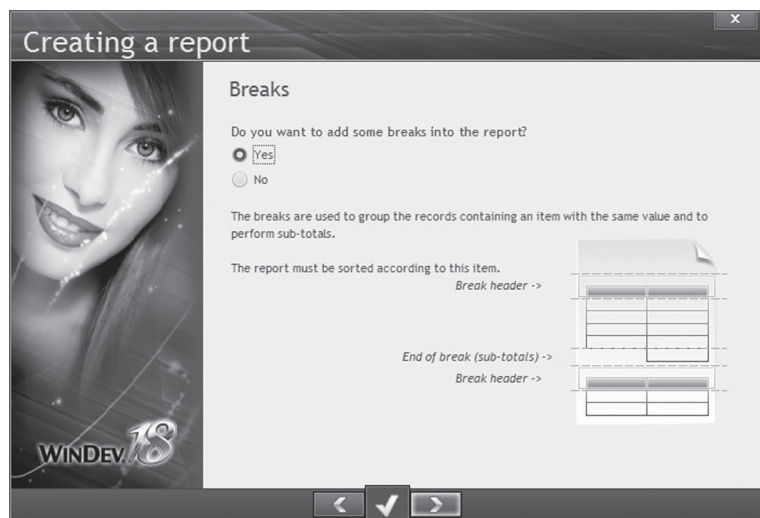
- To sort the customers by city, we must choose a search criterion:
1. Select the sort item: click "City" in the middle section of the screen.
  2. Click the "Sort" button (top right of the screen). Choose "Sort on the selected item".



3. Choose "Ascending" and validate. A red arrow is displayed to the right of the "City" item as well as the number "01". This arrow indicates that this item will be sorted in ascending order. The number "01" indicates that this sort will be the first sort performed.



- We are now going to validate this screen. Go to the next screen.  
The wizard asks if there is a break.



### What is a break?

A **Break** is used to group records (or rows) according to one or more criteria. Caution, the records (or the rows) will be printed.

**A break must ABSOLUTELY be linked to a sort.**

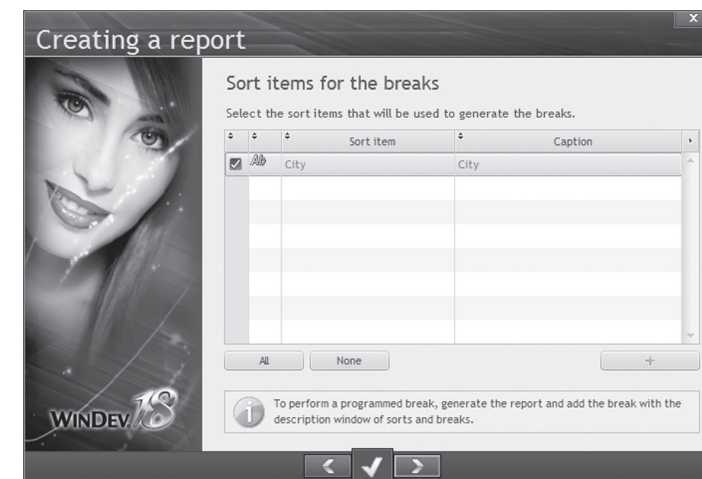
You will notice the presence of a break header and a break footer.

The information common to the different rows will be grouped in the break header.

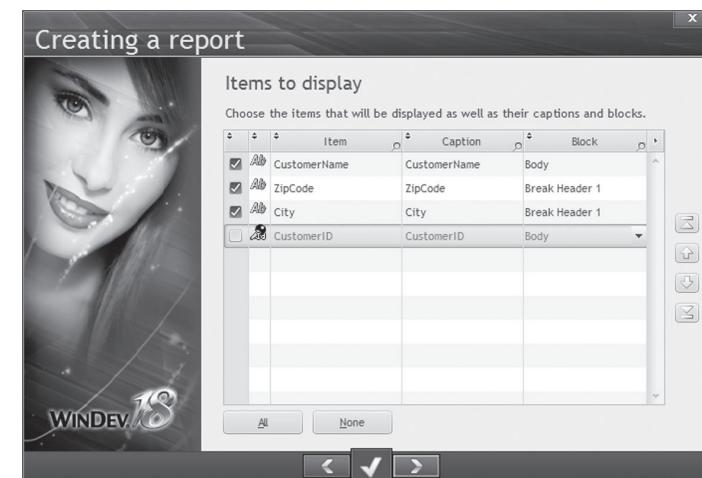
The totals, counters, ... are found in the break footer.

This break is used to group a set of records according to the same criterion. In our example, the break is performed on the zip code and it is used to group all the customers who live in the same city.

- Answer "Yes". Go to the next screen.

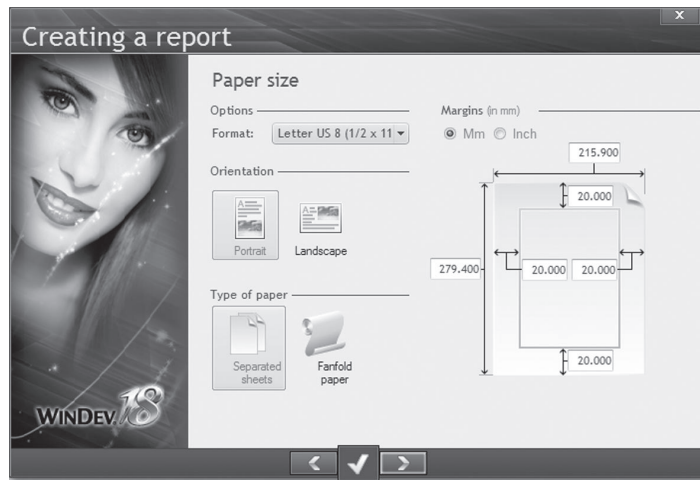


- The break is performed on the city. Go to the next screen.
- You will now specify the order in which the items will be printed and how they will be arranged in the blocks.  
In our example, the zip code and the city will be printed in the "Break header 1" block and the name in the Body block.  
In the column on the right, select "Break header 1" for "ZipCode".  
Clear the "CustomerID" item.



The arrows found to the right of the table are used to modify the order of the items. You can for example display the city before the name.

- Go to the next screen. This screen is used to define the layout of the report.



We will keep the default values as well as the "Portrait" orientation.

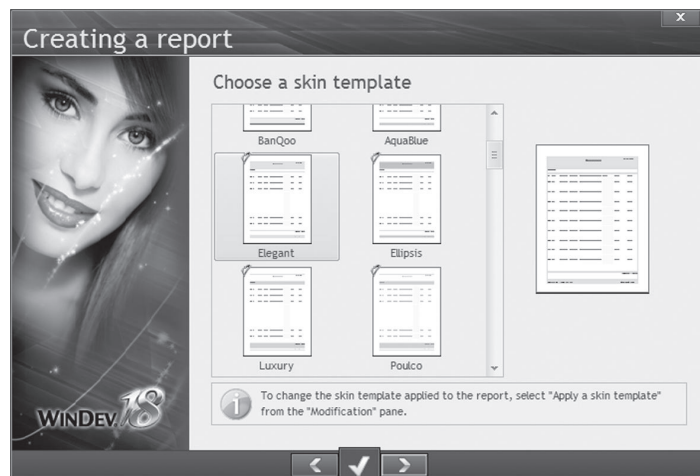


Notes

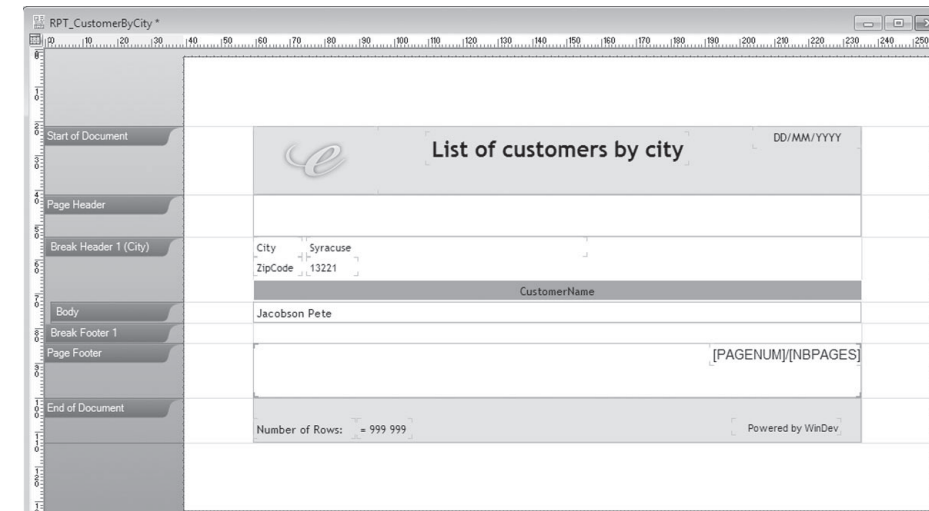
#### Print margins

When choosing the print margins, don't forget to take into account the physical margins of the printer. The physical margins of the printer are margins in which no printing is allowed. Furthermore, the physical margins differ according to the type of printer.

- Go to the next screen.
- Select the "Elegant" skin template for example. Go to the next screen.



- We must now enter a name and a caption for the report and save it.
  1. Enter the "RPT\_CustomerByCity" name.
  2. Enter the title: "List of customers by city".
  3. Validate.



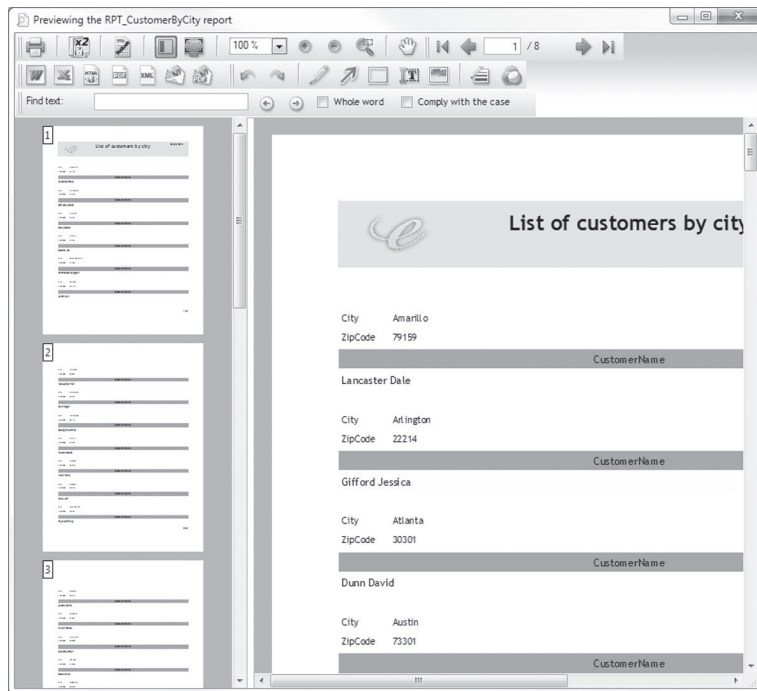
- 4. Save the report in the default folder. Validate the backup window. The report is completed. Let's now run the test of the report.

- To run this report, click  among the quick access buttons.



The print destination can be:



- Choose "Print preview" and validate.  
The report's execution is displayed on the screen.



You can:

- Print the current page or the entire document by clicking the printer .
- Create a duplicate copy.
- Select a different zoom value with the icons .
- 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 the message body.
- Create an email with the report in PDF format in attachment.
- Annotate the document.
- Perform a search inside a document.
- Add watermarks.

## LESSON 7.2. STATISTICAL REPORTS

**This lesson will teach you the following concepts...**

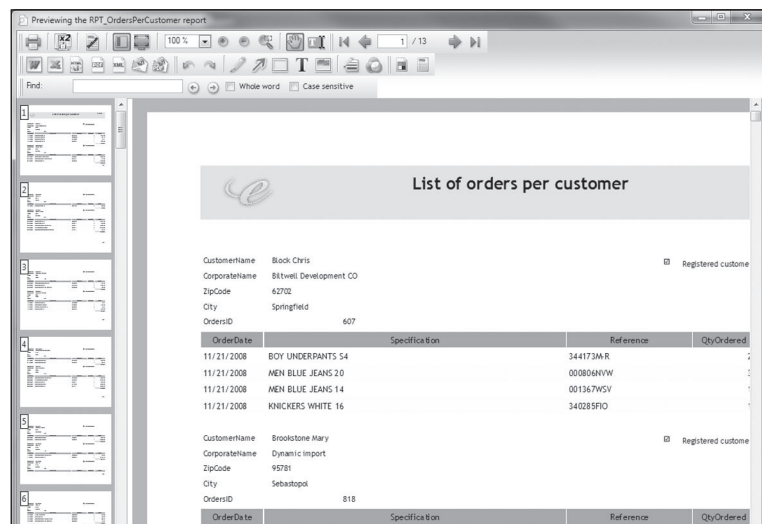
- Linking several data files in a report.
- Grouping elements in a report.



Estimated time: 30 min

## Overview

We are now going to edit the list of orders per customer. This list will be sorted by customer name. We are going to print, for each customer, the order and the details of the order (lines) with a sub-total for each order. The result should be similar to the representation below:




The report that will be created uses several data files as data source.

## Step by step

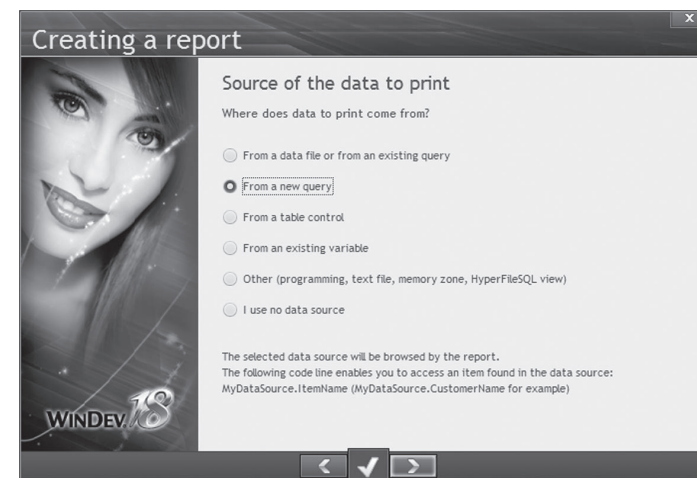
- If needed, open the example project, "WD Printing Reports": on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Print Reports (Exercise)". The project is loaded.

- To create a new report:

1. Click  among the quick access buttons.
2. Click "Report" in the wheel that is displayed. The report creation wizard starts.



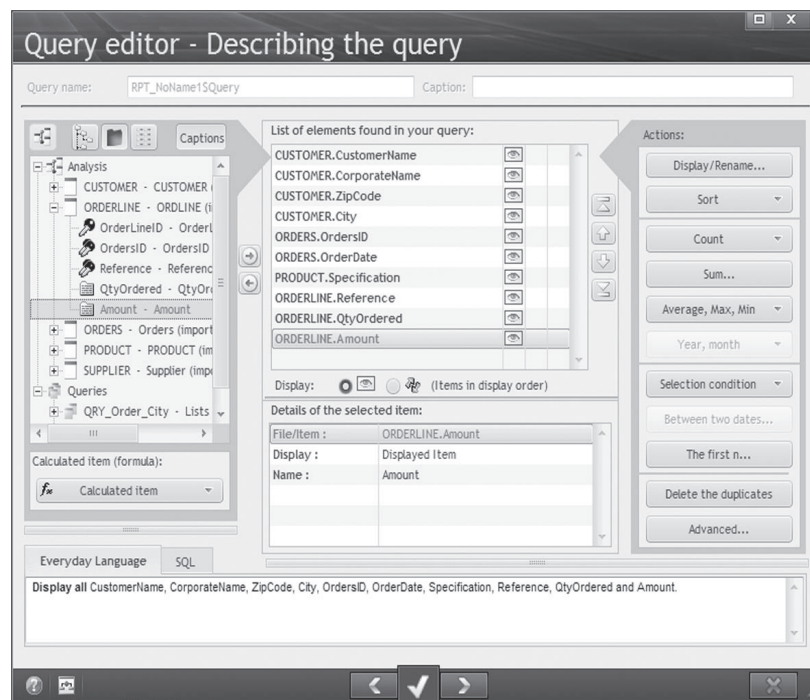
- Select "Table" and go to the next screen.
- Select "From a new query" and go to the next screen. The query does not exist so let's create it.



We are going to print the data coming from the CUSTOMER, ORDERS, PRODUCT and ORDERLINE files.

► Select the items that will be printed (via a double click):

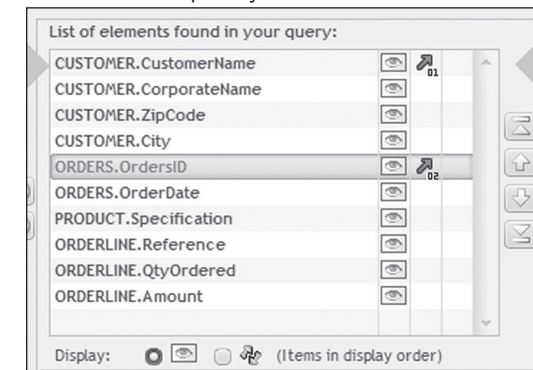
- The CUSTOMER file:
  - CustomerName
  - CorporateName
  - ZipCode
  - City
- The ORDERS file:
  - OrdersID
  - OrderDate
- The PRODUCT file:
  - Specification
- The ORDERLINE file:
  - Reference
  - QtyOrdered
  - Amount



The results will be sorted by customer name and by order number.

- In the list found in the middle, select the first item on which a sort must be performed:
1. Click the "CustomerName" item. In the popup menu (right mouse click), select "Sort the item .. Sort by ascending order".
  2. Click the "OrdersID" item. In the popup menu (right mouse click), select "Sort the item .. Sort by ascending order".

A red arrow is displayed to the right of the "CustomerName" and "OrdersID" items. The number found beside each arrow indicates the priority order of the sorts.



- We are now going to validate this screen. Go to the next screen.
- We are now going to specify the breaks. Choose "Yes". Go to the next screen.



A break is proposed by default on the "CustomerName" and "OrdersID" items (already defined as sort criteria).



- Deselect the CustomerName item (click the checkmark).

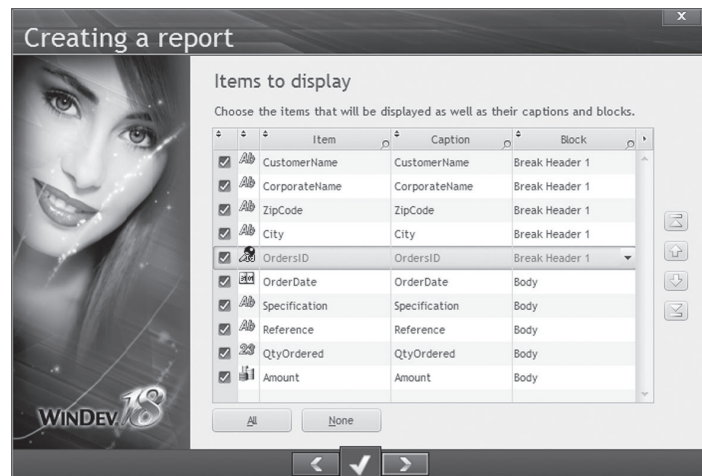


Go to the next screen.

You will now define the location of the items.

The information regarding the customer will be displayed in the break header 1. The body block contains the details of each order.

- Organize the items as specified in the image below:



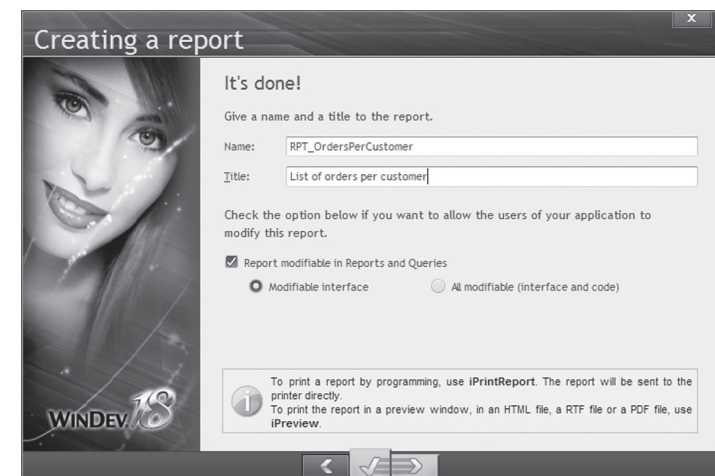
- Go to the next screen.

We must now specify whether totals or counts are found in the report. We are going to add up the amount of the order lines.

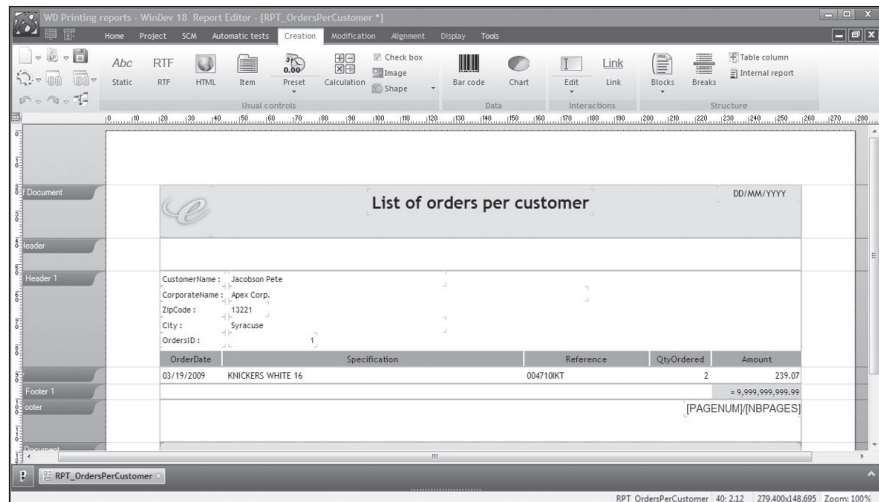
This sum will be included in a break. We are going to keep the sum on the amount. Specify that no calculation will be performed on the quantity by choosing "No calculation" in the "Calculation" column.



- Go to the next screen.
- We will keep the default parameters for the page format.
- Go to the next screen.
- Choose a skin template and go to the next screen.
- Give:
- a name to the report: "RPT\_OrdersPerCustomer".
  - a caption: "List of orders per customer"

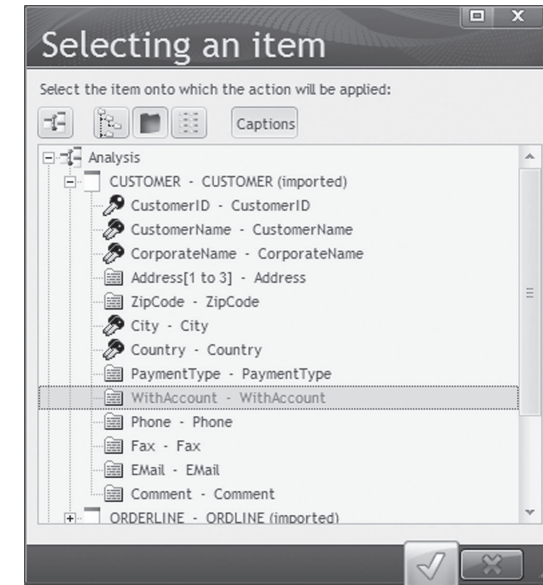


- ▶ Validate. The report being too big, the report editor proposes to change its format or to reduce it.
- ▶ Select the landscape mode and validate.
- ▶ The report is displayed in the editor:



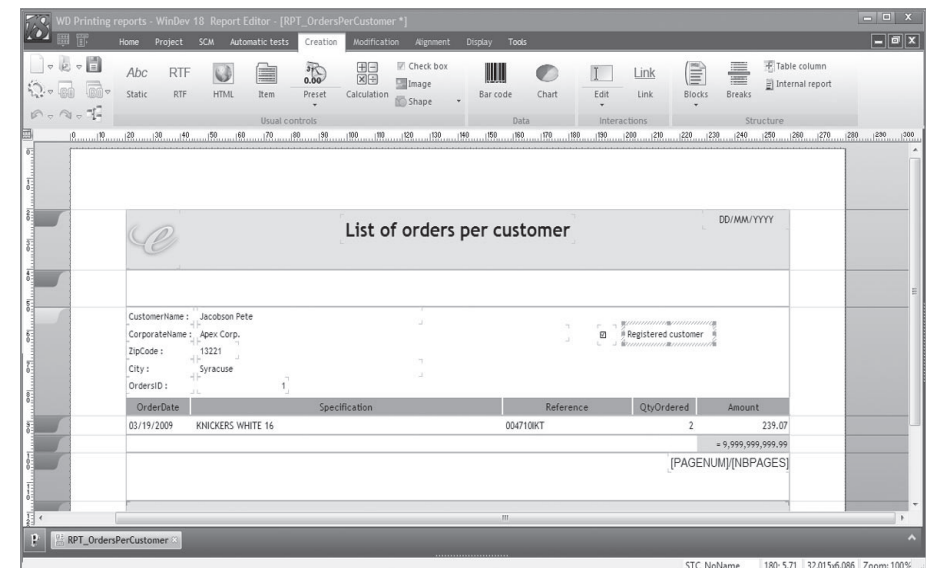
- ▶ Modify (if necessary) the layout of the controls.
- ▶ We are now going to add a check box to this report.  
Check boxes in reports are mainly used in pre-printed forms (such as boxes for "Mr", "Mrs" and "Ms", ...).  
An item in the "CUSTOMER" file indicates whether the customer has an account with the supplier (us in this case). The value of this item can be "Yes" or "No", "True" or "False".
- ▶ On the "Creation" pane, in the "Usual controls" group, click "Check box". The check box appears under the mouse. The check box must be included in the "Break header 1" block of the report.
- ▶ Display the description of the check box control (right click, "Description"):
  1. In the "General" tab, enter the name of the control, "CBOX\_WithAccount".
  2. In the "Link" tab:
    - Click "Other item".


- Select the "WithAccount" item of the "CUSTOMER" file.

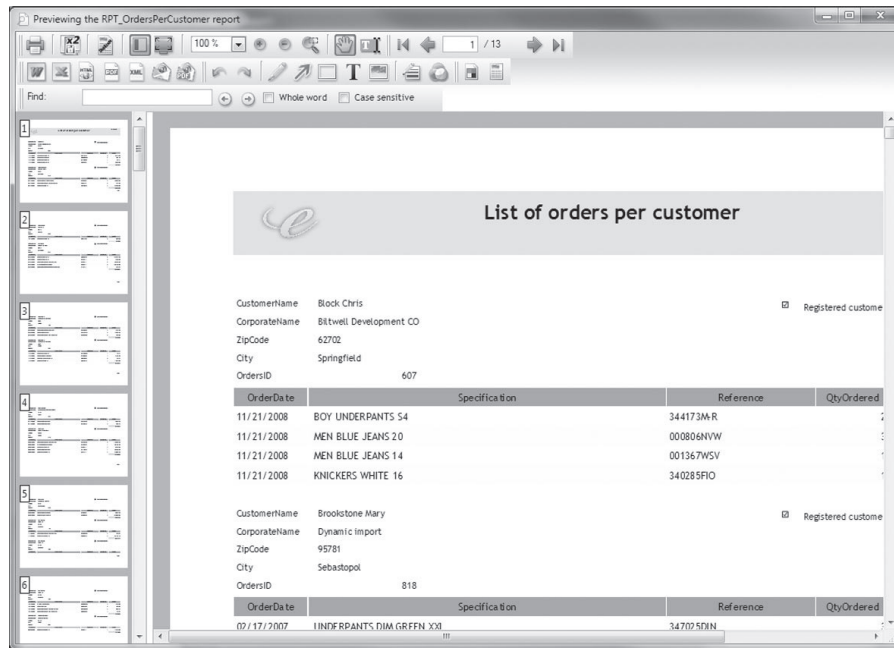


- Validate. The item is automatically added to the report's base query.
3. Validate the description window of the Check Box control.

- ▶ Add a Static control: on the "Creation" pane, in the "Usual controls" group, click "Static". Position this static next to the check box and enter the text "Registered customer".
- ▶ This is the report as it is displayed in the editor:



► Save the report and run its test with the button .



## LESSON 7.3. CROSSTAB REPORTS

**This lesson will teach you the following concepts...**

- Creating a crosstab report step by step.



Estimated time: 20 min

## Overview

The report editor gives you the ability to create "Crosstab" reports. In this case, the report contains a double-entry table.

Example:

|                   | 2006       | 2007       | 2008 | 2009       |
|-------------------|------------|------------|------|------------|
| Bock Chris        |            | 15,497.47  |      | 15,497.47  |
| Brookstone Mary   |            | 128,879.86 |      | 128,879.86 |
| Chan Edward       | 434.46     |            |      | 434.46     |
| Chase John        |            | 348,760.45 |      | 348,760.45 |
| Cheng Sheila      | 19,785.06  | 16,754.35  |      | 36,539.41  |
| Collins Michelle  | 10,376.72  |            |      | 10,376.72  |
| Daniel Nina       |            | 332,246.60 |      | 332,246.60 |
| Davis Paula       |            | 193,023.70 |      | 193,023.70 |
| Dell Melanie      | 2,924.88   | 45,519.15  |      | 48,444.03  |
| Dorevan Alexander | 303,950.74 |            |      | 303,950.74 |
| Dowell Ralph      |            | 46,179.77  |      | 46,179.77  |
| Dunn David        | 7,793.40   |            |      | 7,793.40   |
| Edward Charles    | 259,118.61 |            |      | 259,118.61 |
| Finch Patrick     | 1,309.14   |            |      | 1,309.14   |
| Flores Robert     | 149,999.15 |            |      | 149,999.15 |
| Fong Richard      |            | 79,398.10  |      | 79,398.10  |
| Gaines Jason      |            | 229,580.40 |      | 229,580.40 |
| Gallagher Arthur  | 7,928.16   |            |      | 7,928.16   |
| Gettlen Kristina  |            | 20,570.90  |      | 20,570.90  |
| Gibson Helen      | 179.80     | 216,417.52 |      | 216,597.32 |
| Gifford Jessica   |            | 201,333.60 |      | 201,333.60 |
| Gordon Ed         | 53,616.33  | 2,413.28   |      | 56,029.61  |
| Haden Julianne    | 123,494.08 | 26,263.76  |      | 149,757.84 |
| Heidelberg Steve  |            | 106,581.72 |      | 106,581.72 |

Unlike the other types of reports, the "Crosstab" report is always based on an **embedded query**. This query is created via the wizard for creating a "Crosstab" report. This query can be modified once your report is created.

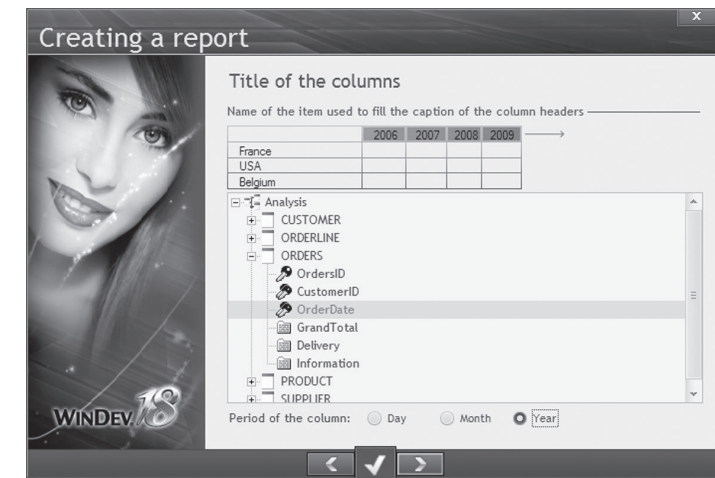
## Step by step

We are going to create a report used to summarize the turnover per customer and per year. This table can be represented in several ways:

- by specifying the different years and the corresponding gross sales for each customer.
  - by creating a crosstab, in order to sort the results by customer (vertically) and by year (horizontally). We are going to use this method.
- If needed, open the example project, "WD Printing reports": on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Print reports (Exercise)". The project is loaded.
- Create a new report:
1. Click among the quick access buttons.
  2. Click "Report" in the wheel that is displayed. The report creation wizard starts.
- Select "Crosstab". Go to the next screen.

The wizard proposes to choose the item that will be used as data source for the headers of columns. This enables you to define the first entry (row) of your crosstab.

- Select the "OrderDate" item in the "ORDERS" file.  
The wizard offers you three choices then (day, month, year).  
Select "Year". Go to the next screen.



- The wizard asks you whether the dates must be "bounded". Don't check anything as all the orders will be used. Go to the next screen.
- The wizard asks you to choose the item that will be used as data source for the rows. This enables you to define the second entry (column) of your crosstab.  
Select the "CustomerName" item in the "CUSTOMER" file.

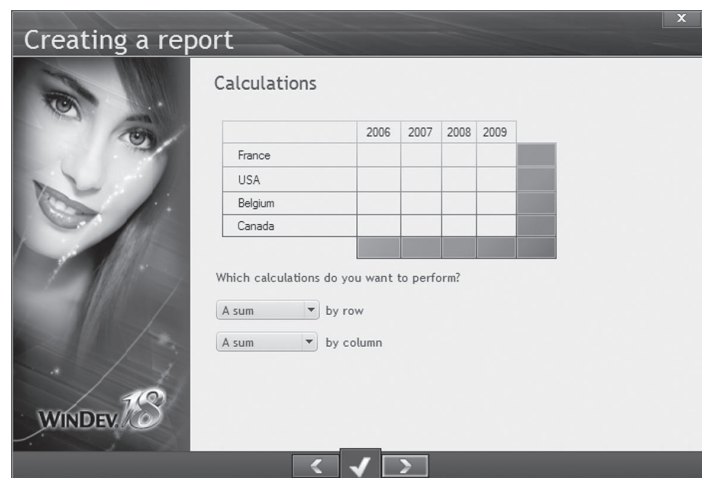




- ▶ Go to the next screen.  
The wizard asks you to select the type of information that will be displayed in the cells of the crosstab. As we want to display the gross sales per customer and per date, we are going to request a sum of the total amounts of each order placed by each customer for each year.
- ▶ Select "The sum of the item:". In the list that is displayed on the side, choose the "GRANDTOTAL" item of "ORDERS" file:

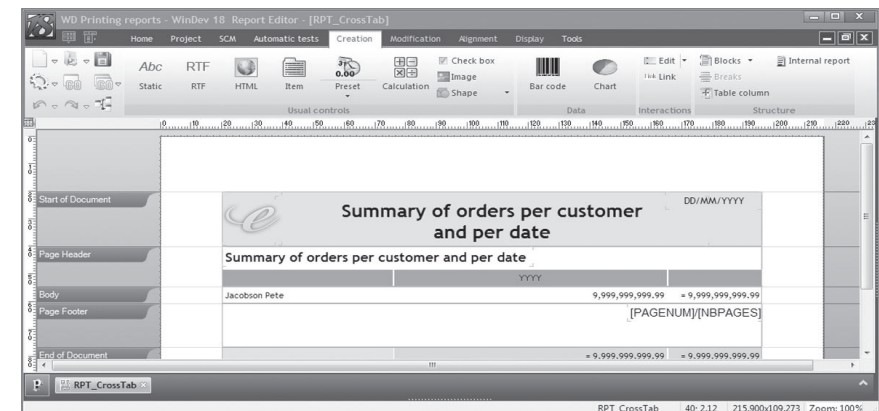


- ▶ Go to the next screen.  
The wizard proposes to calculate a sum per row and a sum per column. Accept this calculation:

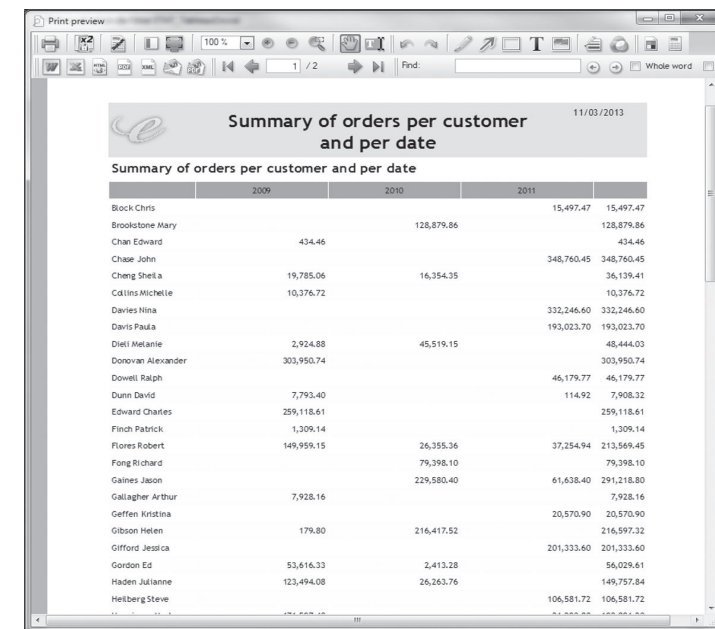


- ▶ Go to the next screen. Don't change the page layout. Go to the next screen.
- ▶ Select a skin template ("Elegant" for example). Go to the next screen.

- ▶ Give a name to this report:
  1. Enter the name: "RPT\_CrossTab".
  2. Enter the description: "Summary of orders per customer and per date".
  3. Validate. Since the report is too big, the report editor proposes to change its format or to resize it: select "Reduce the table". The report is displayed in the report editor.
- ▶ Modify (if necessary) the title and the size of the column title to get the same layout as below.



- ▶ Save the report. To run its test, click .



You now know how to create a crosstab report. This type of report can be used for other purposes such as performing comparisons, statistics...



## LESSON 7.4. SPECIAL CONTROLS

### This lesson will teach you the following concepts...

- Creating a link in a report.
- Creating a clickable control (in the preview)



Estimated time: 20 min

### Overview

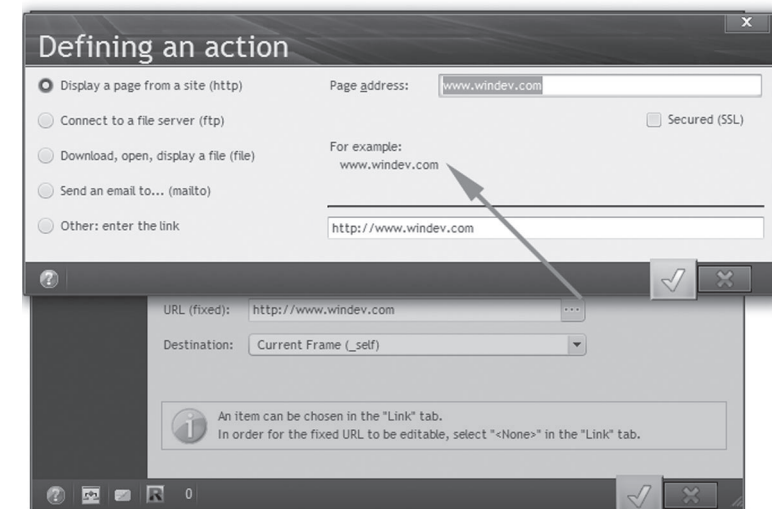
The report editor of WinDev proposes two interesting features for the controls:

- **The Link controls**, that can be used both in reports generated in HTML and in reports displayed in the print preview.  
The Link controls are used to directly open a site from a report. This would be useful to create a summary of orders placed by email. This report can contain a link to the company site, a link for sending an email to the company, ...
- **The clickable controls**, only available in print preview.  
In this case, the click performed on one of the report controls triggers a specific process, such as displaying the entire form of the selected customer.

These two types of controls can be easily identified in the print preview via small stars.

### Operations for creating a Link control

- If needed, open the example project, "WD Printing Reports": on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Print reports (Exercise)". The project is loaded.
- Open the report named "RPT\_OrderWithLink.WDE" (by using the project explorer for example). This report is found in the "Answers" custom-folder.
- Display the description of the "Link" control. This link will allow the Web user to visit the site of the company that sent the invoice.  
In the "General" tab, click the [...] button located next to the URL control.

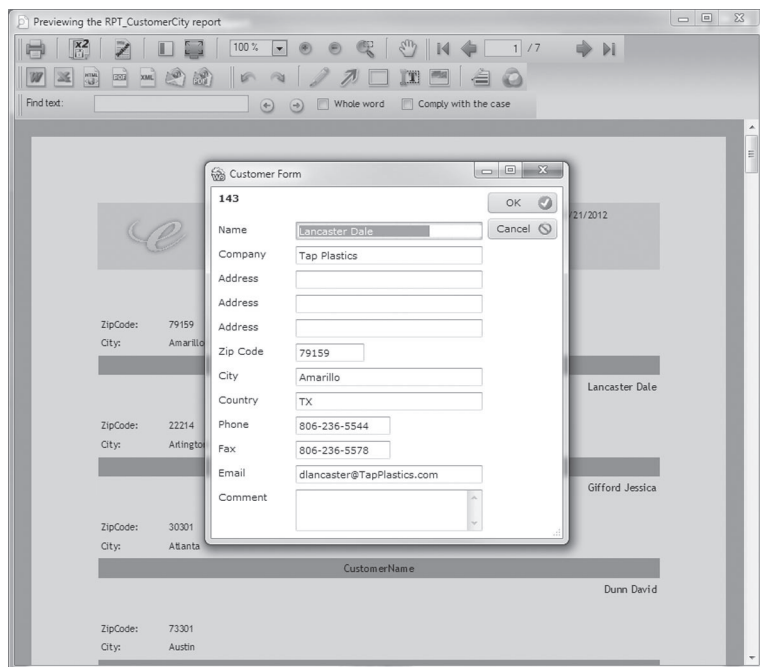


Note: A Link control in a report can be used to display the content of an item or the content of a URL. In this case, you can:

- display a page of a site
  - connect to an FTP server
  - download, open, display a file
  - send an email
- Test the report ([GO] button in the quick access buttons). This link operates in print preview (a star displayed beside the link indicates that this link can be clicked) or in HTML mode.

## Operations for creating a clickable control

- If needed, open the example project, "WD Printing Reports": on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Print reports (Exercise)". The project is loaded.
- Open the report that was previously created "RPT\_CustomerByCity" (or the report named "RPT\_CustomerCity" found in the "Answers" folder). This report returns the list of customers by city. We will allow the user to click the name of the customer to display his personal details in a form window.



- The project already contains a window to open: "WIN\_Customer\_Form". This window is found in the "Answers" folder.

- Open this window in the window editor (press CTRL E to find the window and to open it). This window was created by RAD. Its code for declaring the global variables was modified in order to display the customer whose identifier is passed in parameter.
- Display the global declaration code of the window:
- Right click inside the home window.
  - "Code" option of the popup menu.
- The code is as follows. The lines in bold represent the lines modified in order to display the customer whose identifier is passed as parameter.

```

CASE ~="Modif"
 IF CustomerId<>0 THEN
 HReadSeekFirst (CUSTOMER, CustomerId, CustomerId)
 END
 IF HNbRec (CUSTOMER)=0 OR HOut (CUSTOMER) THEN
 // open a dialog box to inform the user
 Info("The file contains no record.")
 // close the window
 Close
 END

 // Fill the controls
 FileToScreen()
END

```

- Close the code window and the "WIN\_Customer\_Form" window.
- To directly open the "WIN\_Customer\_Form" window from print preview:
1. In the report editor, in the report "RPT\_CustomerPerCity" (or the "RPT\_CustomerCity" report), click the "ITEM\_CustomerName".
  2. Display the code of this control ("Code" from the popup menu).
  3. Enter the following code in the "Click on ITEM\_CustomerName" process:

```
Open (WIN_Customer_Form, "Modif", MySource.CustomerId)
```

This code is used to:

- open the "WIN\_Customer\_Form" window in Modification mode.
  - pass the identifier of the customer clicked in parameter to the window (this identifier is found in the query included in the report).
4. Save your report and run its test.

## LESSON 7.5. RUNNING REPORTS

### This lesson will teach you the following concepts...

- Running reports in WLanguage.



Estimated time: 10 min

### Running reports in WLanguage

The tests of the created reports have been run from the editor. We will now see how a report can be run from a program.



Notes

WinDev enables you to directly run the reports from an application via the WLanguage functions. You also have the ability to add source code in order to perform specific processes inside each report block (perform calculations, display a dialog box, ...).

The general syntax for running a report is as follows:

```
// Preview the "CustomerLabel" report
iPreview(100)
// Print Report
iPrintReport(CustomerLabel)
```

A preview screen will be proposed when this code is run. To directly print the report without using a preview window, delete the line:

```
iPreview(100)
```

#### iPreview and iDestination

The functions named **iPreview** and **iDestination** are used to select the print destination.

The syntax of **iPreview** is as follows:

```
iPreview(<print format>)
```

**iPreview** can also be used to display the printed report on the screen. In this case, the <print format> parameter can take the following values:

- **iPrint**: to print on the current printer directly
- **iPage**: to display the entire page on the screen
- **i2Pages**: to display 2 pages on the screen
- **i100**: to display the page with a 100% zoom

Other constants (that can be used with **iDestination** or **iPreview**) can be used to get more specific print formats:

- **iFile**: to print the report in a text file
- **iHTML**: to print the report in HTML format
- **iHTMLWithoutCSS**: to print the report in HTML format without style sheet
- **iPCL**: to print the report in PCL format
- **iPDF**: to print the report in PDF format
- **iRTF**: to print the report in RTF format
- **iXLS**: to print the report in XLS format
- **iXML**: to print the report in XML format
- **iFax**: to print onto a fax directly.

For more details about the *iPreview* or *iDestination* functions, see the online help (keyword: "iPreview" or "iDestination").

### iPrintReport function

*iPrintReport* is used to run a report. The syntax of this function is as follows:

```
iPrintReport (<Report Name>)
```

#### Note

Other syntaxes are available for these functions, see the online help (keywords: "Printing a report", "iPreview", "iPrintReport") for more details.



#### Notes

Give the users of your applications the ability to create their own reports and queries (or to modify the existing reports and queries)!  
All you have to do is distribute "Reports & Queries" with your application. See the next lesson for more details.

## LESSON 7.6. DISTRIBUTING "REPORTS & QUERIES" WITH YOUR APPLICATIONS

### This lesson will teach you the following concepts...

- What is "Reports & Queries" used for.
- How do I distribute "Reports & Queries".
- How do I use "Reports & Queries".



Estimated time: 10 min

## Overview of "Reports & Queries"

"Reports & Queries" allows the users of your applications to modify and to create their own reports and queries for your application.

A user thinks that some information is missing from the standard report? "Reports and Queries" allows the user to add this information directly from the print preview. This modification may be useful to other users? The report can be made available to all the users who are using the application on the network. Same for the queries.



Notes

"Reports & Queries" is free and it can be distributed with your applications created in WinDev. See the license for more details about the distribution and the conditions.

## Starting "Reports & Queries"

To allow the end users to customize the reports of your application or to create their own reports and their own queries, all you have to do is start "Reports & Queries".

To allow the end users to start "Reports & Queries":

- display your reports in the preview window. The end users will be able to start "Reports & Queries" by clicking .
- add the "?" in 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 & Queries" with your application

To distribute "Reports & Queries" with your own WinDev application, you must:

- specify in the project that the application allows "Reports & Queries" to be run.
- in the analysis, define (if necessary) the files and the items that can be handled by "Reports & Queries".
- define the reports and the queries that can be modified in "Reports & Queries".
- create the executable and the setup program for "Reports & Queries".

These different steps will be presented below. The following operations can be performed on the "WD Printing Reports.WDP" project that was already used in this lesson.

## Configuring the project

To use "Reports & Queries" from your application, the project associated with your application must be configured.

- ▶ If needed, open the example project, "WD Printing Reports": on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Print reports (Exercise)". The project is loaded.
- ▶ Display the project description  
Remember: to display the project description, on the "Project" pane, in the "Project" group, click "Description".
- ▶ In the "Advanced" tab, check "Allow to start "Reports and Queries"". Validate.

## Configuring the analysis

Now, we are going to open the analysis in order to configure the files and the items that will be used in "Reports & Queries".

- ▶ Open your project's analysis: click among the quick access buttons of the WinDev menu. By default, all the data files and all their 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 & Queries".



Notes

"Reports & Queries" does not allow the user to add, modify or delete records (*HAdd*, *HModify* and *HDelete* are not allowed). No data file can be created or re-created (*HCreation* and *HCreationIfNotFound* are not allowed).

- ▶ To define an invisible file in "Reports & Queries":
  1. In the data model editor, select the requested data file.
  2. Display the description window of the file ("Description of data file" from the popup menu).
  3. In the "Reports and Queries" tab, uncheck "Visible to the end user in "Reports and Queries"".
  4. Validate.
- ▶ To define an invisible file item in "Reports & Queries":
  1. In the data model editor, select the requested data file.
  2. Display the description window of the file ("Item description" from the popup menu).
  3. Select the requested item in the table.
  4. In the "Reports and Queries" tab, uncheck "Visible to the end user in "Reports and Queries"".
  5. Validate.
- ▶ To take into account the modifications made to the analysis, you must regenerate the analysis ("Generation" option on the "Analysis" pane). An automatic data file modification is offered to apply the modifications performed.

## Configuring the reports

When creating a report, you have the ability to specify whether this report can be modified in "Reports & Queries".

- ▶ To make a report of your application modifiable in "Reports & Queries":
  1. Open the report in the report editor.
  2. Display the report's description ("Report description" from the popup menu).
  3. In the "Details" tab, check "Report modifiable in Reports and Queries".
  4. Check:
    - "Modifiable interface" in order for the interface of the report to be modifiable in Reports & Queries.
    - "All modifiable (interface and code)" in order for the entire report to be modifiable in Reports & Queries.
  5. Validate and save the report.

## Configuring the queries

By default, a query cannot be modified in "Reports & Queries". When creating a query, you can specify whether this query can be modified in "Reports & Queries".

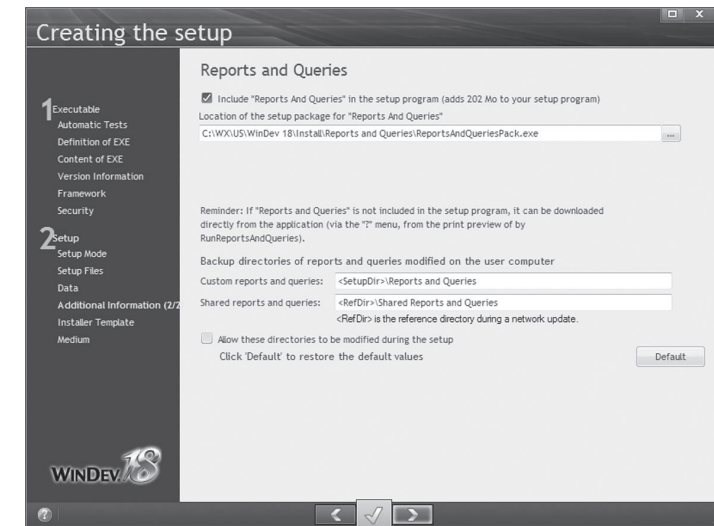
- ▶ To make a query from your application modifiable in "Reports & Queries":
  1. Open the query in the query editor.
  2. Display the query's description ("Query Description" option in the popup menu of the query's chart).
  3. Click the "Advanced" button.
  4. In the "Reports and Queries" tab, check "Visible in Reports and Queries".
  5. Validate and save the query.

## Creating the executable and distributing the application

The method for creating and distributing an application that uses "Reports & Queries" is the same as the method for creating and distributing a standard application. The parameters specific to "Reports & Queries" can be specified when creating the setup program.

- ▶ To create the setup program for your application, 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 the wizard window.
  1. Select "Create the executable now". The creation of the executable is performed.
  2. The wizard for setup creation starts.
  3. Choose an "Individual setup". Go to the next screen.
  4. Choose a "Standard setup". Go to the next screen.
  5. In the screen named "Files installed", select the data files (.fic, .mmo and .ndx) found in the directory of the executable. To do so, click "Add" and select these files. Go to the next screen.
  6. In the screen named "Additional modules" (accessible by clicking the "Additional information" link on the left), check whether "Include the setup of Reports and Queries" is checked.

7. In the screen named "Reports and Queries", specify whether the setup of "Reports & Queries" must be included in your setup program.  
In this case, the setup package of "Reports & Queries" will be sought. By default, this setup pack is found in the "Reports and Queries" sub-directory of the setup directory of WinDev.  
Caution: Including "Reports & Queries" significantly increases the size of your setup (about 200 MB).



8. Specify (if necessary) the directories corresponding to the reports and queries modified or created by the end user.  
The directory of the custom reports and queries corresponds to the directory of the reports and queries visible only by the user who created them.  
The directory of the shared reports and queries corresponds to the directory of the reports and queries visible by all the users of the application.  
These directories can be modified (or not) when installing the application. Check this option ("Allow these directories to be modified during the setup").
9. Follow the wizard until the setup program of your application is generated (the "WD Printing reports" application in our example). Don't close the window notifying you of the end of the setup procedure.



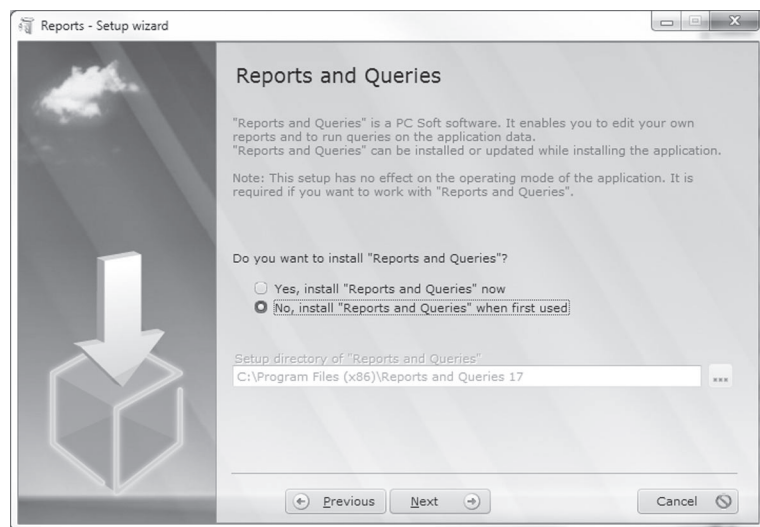
## Installing and using "Reports & Queries"

### Installing the application

To test the setup and use "Reports & Queries", we are going to install the "WD Printing Reports" application.

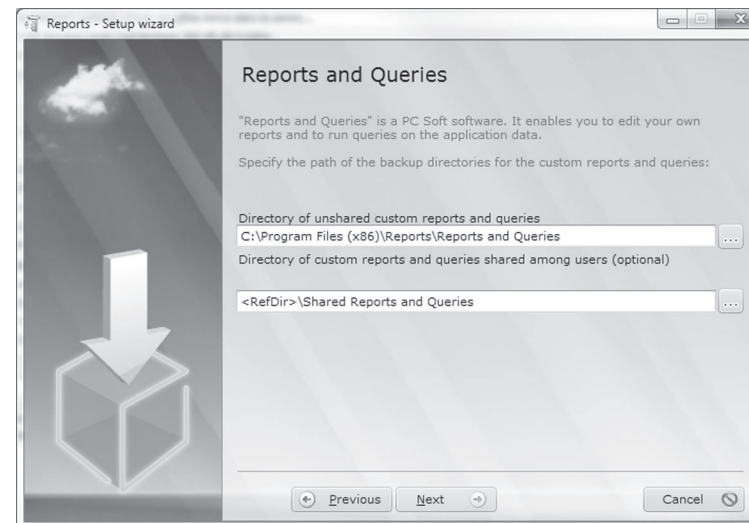
By following the steps of the previous paragraph, you have now generated the setup program of this application. The setup program was created in the "Install" sub-directory of the current project.

- ▶ In the window notifying you of the end of the setup's creation, click the "Test" button. If this window was already closed, run the "Install.exe" program found in the directory of the setup program. The setup of the application starts.
- ▶ We are going to follow the different steps of the setup program for the "WD Printing Reports" application.
- ▶ The screen named "Reports and Queries" is used to install "Reports & Queries" (only if this program is not already installed on the current computer).



Go to the next screen.

- ▶ The next screen enables you to define the directories that will be used to save the reports and the queries.



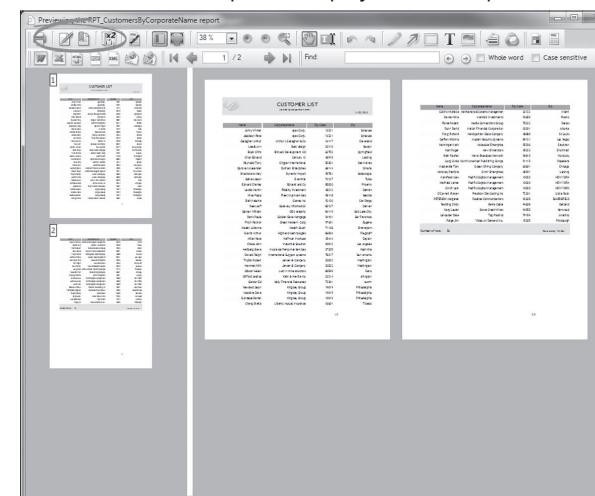
Go to the next wizard screen and end the application setup.

- ▶ The setup program for the application and then for "Reports & Queries" starts.


### Testing the application

We are now going to take the place of the end user of your application (not easy, but worth a try).

- ▶ Start the "WD Printing Reports" application.
- ▶ Select "Reports .. Sorted list". The report is displayed in "Print preview" mode.




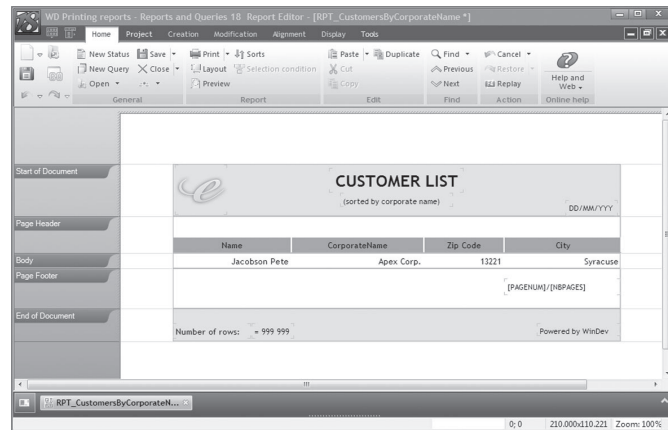
► In this screen, two new icons are introduced:

 Used to modify the current report in the print preview.

 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 & Queries".




3. The user can perform the requested modifications:

- on the style of the information displayed (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 only when the report is run (if a window is also using this query, the modification will not be taken into account by the window).

► In this example, the date will be highlighted in red. To do so:

1. Select the date control in the start of document.
2. On the "Modification" pane, in the "Fast edit" group, click the "Color" icon () and select the red color.
3. Save your report. The setup directory of your application is proposed by default.
4. Close the "Reports & Queries" program.
5. Select the "Reports .. Sorted list" option in the application. The report is displayed in "Print preview" mode and the modification becomes effective.

In this example, the modified report is available only to you. 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 (keyword: Reports and Queries) for more details.

## PART 8

# Advanced programming



DEVELOP 10 TIMES FASTER **PC SOFT**

## LESSON 8.1. AUTOMATIC MANAGEMENT OF ERRORS

**This lesson will teach you the following concepts...**

---

- What is the automatic error management?
- Using the automatic error management.



Estimated time: 10 min

## Overview

The errors can be automatically managed by WinDev. This feature helps you reduce the number of code lines while centralizing the error management.

The use of this feature also makes the code easier to read.

## Implementation

Two operations are performed when an error is detected by a WLanguage function:

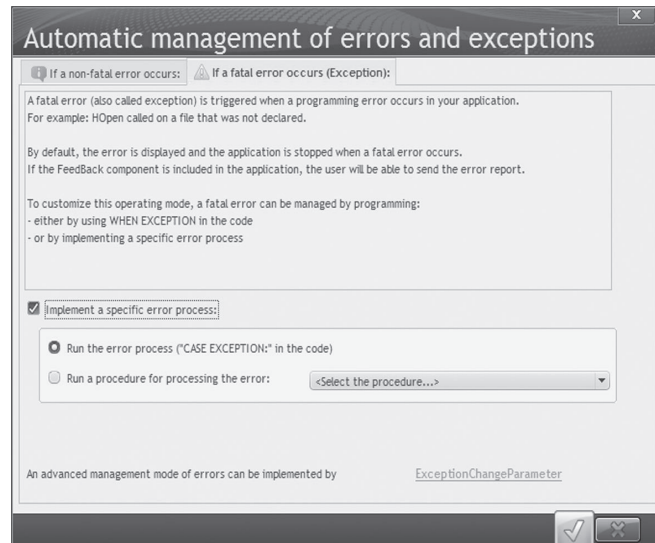
- an error value is returned by the function (**fOpen** returns "-1" if the selected file cannot be opened for example).
- the error is detected by the WLanguage (the **ErrorDetected** variable corresponds to True); the details of this error can be retrieved via **ErrorInfo**.

This second operation can be automatically managed via WinDev error management.

## Implementation

The automatic error management can be configured:

- in the code editor: all you have to do is click the "If error: by program" link in the code editor:



- by programming with **ErrorChangeParameter**.

## Types of errors

Two types of errors can occur in WLanguage:

- the "non-fatal" errors (or 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 hardware problem (not enough disk space, loss of network connection, ...) or to important development problems (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 the errors is used to manage these two types of error according to different methods so that you can specify behaviors adapted to the errors that occur.

## Using the automatic error management

- ▶ Close the current project if necessary. The home window is opened.
- ▶ In the home window, click "Tutorial" and select the project named "Management of errors". The project is loaded.  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Management of errors".

This project presents the management:

- of a non-fatal error (opening an archive that does not exist).  
When running the code line that triggers the error, an error message is displayed to the user offering to retry the operation, cancel the operation or stop the application.
- of a fatal error (division by 0).  
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.
- of an error on several levels (process calling a procedure that opens a non-existing archive).  
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 stop the process.

- ▶ To run the test of the project, click  (among the quick access buttons).



Example

See the "Error detection" example (unit example) supplied with WinDev for more details. This example is accessible from the "Wizards, Examples and Components" pane of WinDev.



## LESSON 8.2. IMPORT/EXPORT

### This lesson will teach you the following concepts...

- CSV file and text files
- XML file
- XLS file



Estimated time: 20 min

### Overview

WinDev allows you to easily handle most data formats: text files (INI files, CSV files, ...), XML files, XLS files, ... Several groups of WLanguage functions allow you to read and create these files. This enables you to easily read the data generated by other software in your WinDev applications or to create files that required a specific formatting.

Three file formats will be presented in this chapter:

- the Text files (text files, INI files and CSV files)
- the XML files
- the XLS files

### Practical example

To simplify the presentation of the various file formats, we've created an example containing the main possible operations.

- ▶ Close the current project if necessary. The home window is opened.
- ▶ In the home window, click "Tutorial" and select "Handle external files (XLS, XML, ...)". The corresponding project is loaded.  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Handle external files (XLS, XML, ...)".

### Handling text files, CSV files, ...

The external files are also called text files or files with direct access. In most cases, this type of file contains text but it may also contain binary information such as images, sounds, ...

In this lesson as in all the WinDev documentation, we will talk about **external files**.

WinDev enables you to handle external files by programming. The WLanguage functions are used to create, read... external files. All the 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 be able to handle an external file, you must know its structure, i.e. how the data is organized inside the file.

The CSV files are text files that use a specific structure. Therefore, they can be handled as the text files.

The .INI files are also text files that use a specific structure. Two specific functions are used to handle them: **INIRead** and **INIWrite**.



## Practical example

- Open the "WIN\_TextFile.WDW" window and run its test. This window presents the different operations that can be performed in WinDev.

We advise you to study the "Text Files" example (supplied with WinDev): this example presents the different operations that can be performed on the external files:

- creating an external file
- reading an external file (by block, by line or all at once)
- writing into an external file
- finding out information about a file
- locking a file
- checking the existence of a file
- selecting a file
- copying a file
- deleting a file
- listing the files found in a directory

This example also presents the operations that can be performed on the disks and on the directories:

- creating a directory
- finding out the current directory
- checking the existence of a directory
- returning the list of directories
- copying a directory
- deleting a directory



Example

The educational example named "WD Text Files" (supplied with WinDev) presents the operations that can be performed on the files, disks and directories with WLanguage. This example is accessible from the "Wizards, Examples and Components" pane of WinDev.

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

## Handling XML files

XML (Extensible Markup Language) is the language that will follow the HTML language on the World Wide Web. Like HTML (HyperText Markup Language), it is a tag-based language, i.e. a language that presents information between tags. Unlike HTML that presents a limited number of tags, XML is a **meta-language** that can create new tags in order to isolate the elementary information that can be found in a Web page.

XML is used to structure a document containing data. A HyperFileSQL data file containing several items and records can be exported to an XML file (**XML** \* functions).



Example

See the "WDXML" example (educational example) supplied with WinDev for more details. This example is accessible from the "Wizards, Examples and Components" pane of WinDev.

WinDev supports:

- the files in XML format via a native access supplied with WinDev. See the online help for more details.
- the export to XML (**TableToXML**, **TextToXML**, **HExportXML**)
- the 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 directly handled in the code editor. To do so:

1. "Drop" the XML file in the project explorer, "XML description" folder.
2. "Drop" the XML file from the project explorer to 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 nodes of the variable by their names. These names are automatically proposed by the mechanism for automatic completion of the code editor.

## Practical example

- Open the "WIN\_XML\_Example.WDW" window and test it. This window is used to:
  - create an XML file. This file is created by using XML functions.
  - read an XML file by using the file description loaded in the code editor.
- Test this window and check the code associated with each button.

## Handling XLS files

To handle the Excel files, WinDev proposes:

- the xlsXXX functions. These functions enable you to read the data found on the different worksheets of an Excel file.
- the advanced variables (xlsDocument, xlsRow, xlsColumn, xlsCell).

### Practical example

Let's see the features of these functions?

- ▶ Open the "WIN\_XLS\_Example" window. This window contains two tabs used to test the advanced variables as well as the xls functions.
- ▶ Run the test of this window.
- ▶ In the "Handling the XLS worksheets dynamically" tab, open the "TestFile.xls" file found in the EXE directory of the example. You can select a cell found in the XLS worksheet and see its content.
- ▶ In the "Reading an XLS worksheet (compatible)" tab, open the "TestFile.xls" file found in the EXE directory of the example. The file contains 3 worksheets. You have the ability to display the content of the three worksheets.
- ▶ Open the file ("Open the file with Excel" button) to check the information displayed by WinDev.
- ▶ Go back to the editor and check the code of the various buttons.



Notes

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 8.3. DYNAMIC COMPILATION

### This lesson will teach you the following concepts...

- Overview
- Drawing a line using dynamic compilation



Estimated time: 20 min

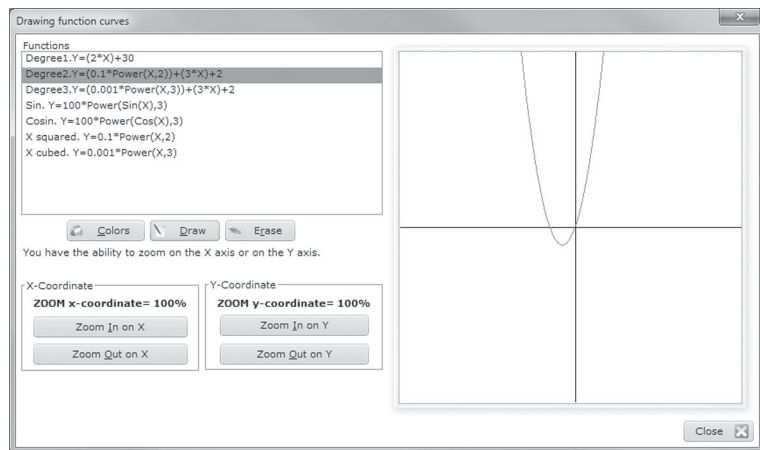
## Overview

The dynamic compilation is used to compile 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 supply the executable again: the modification of a text file is sufficient.

## Example

In the "WD Advanced programming" project, the dynamic compilation is used to create mathematical curves.

- Close the current project if necessary. The home window is opened.
- In the home window, click "Tutorial" and select the project named "Advanced programming". The project is loaded.  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Advanced programming".
- We are going to run the test of this example:
  1. Open the "WIN\_Draw" window (press CTRL + E for example).
  2. Run the test of this window:
    - Choose the function to draw.
    - Choose the colors of the drawing ("Colors" button).
    - Click the "Draw" button to validate.



- Let's go back to the code editor to study the code used to draw this line. Close the window.
  1. Display the code of the "Draw" button (right click the "Code" button). This code calls the procedure named **DisplayLine**.
  2. Position the mouse cursor over the name of the procedure and press [F2]. The code of the procedure is automatically displayed in the code editor.

The code of this procedure can be divided into several sections:

1. initializing the variables.
2. drawing the chart background and axes.
3. managing the errors (this topic will be presented later in this chapter).
4. declaring the dynamically compiled code.

```
// Code of the function that was compiled dynamically
FunctionCode is string = [
Function YValue(Funct,X)
Y is int
%1
Result IntegerPart(Y)
]
```

This code highlights several important topics:

- A dynamically compiled function is contained in a string.
- The code used handles a multiline string. This string uses no quotes as the standard strings. The = [ and ] characters are used to define the beginning and the end of this string. This string is highlighted in pink in the code editor.
- The code that will be compiled is a YValue function that expects two parameters: Funct (the function chosen by the user) and X (the x-coordinate of the point to draw). The function will return the corresponding Y coordinate.

5. compiling the function and using it.

```
// Compile the function used to calculate the value of Y
ErrCompile = Compile("YValue", ...
StringBuild(FunctionCode, Funct))
IF ErrCompile = "" THEN
Xold = -201 // We calculate the previous values
Yold = YValue(Funct,-201)
FOR X=-200 TO 200 // Read the X coordinates and trace
Y = YValue(Funct,X)
dLine(XScale*Xold)+X0, ...
-(YScale*Yold)+Y0, (XScale*X)+X0, -(YScale*Y)+Y0)
Xold = X // Storing the previous values
Yold = Y
END
ELSE
Error("The function couldn't be compiled "+ ...
"dynamically." + ErrCompile)
END
```

The compilation of the function is performed by **Compile**. The dynamically compiled function becomes usable as soon as this function is used (and if no error is returned).

## LESSON 8.4. WINDOWS EVENT

### This lesson will teach you the following concepts...

- Programming the Windows events.



Estimated time: 10 min

### Introduction

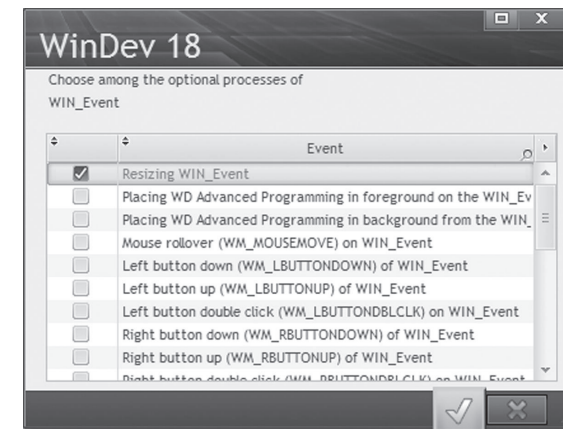
Each action performed by Windows corresponds to a Windows event. Different types of events can occur, for example:

- 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 the most frequently used events. For example, the following events are proposed for an edit control:

- Initialization of the control
- Entry in the control
- Modification of the control
- Exit from the control

Several other events can also be easily managed. A toolbar is available in the code editor:



All you have to do is add the requested event. Example: "Key Down".

To manage more "specific" events, use the WLanguage function named **Event**. **Event** is used to trigger a procedure in response to a Windows event. This enables you to associate a WLanguage procedure to any type of Windows event.



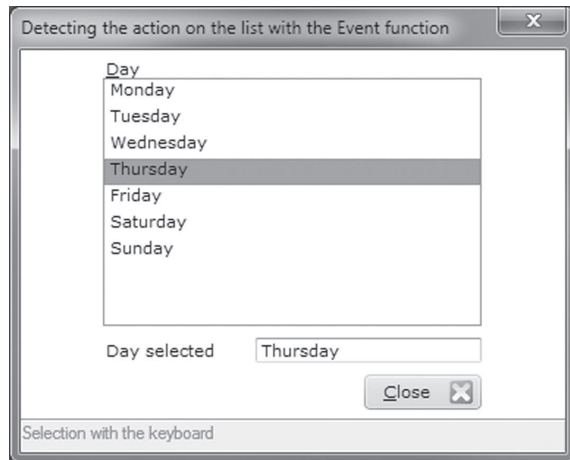
Notes

To use **Event**, you must be familiar with the Windows programming and the Windows events.

To find out the non-exhaustive list of Windows events, see the WinDev online help (keyword: "Value of the constants for the Windows API").

## Example: Detecting the click performed on a list

- ▶ If needed, open the example project, "WD Advanced programming": on the "Home" pane, in the "Online help" group, expand "Tutorial" then select "Advanced programming". The project is loaded.
- ▶ Open the "WIN\_Event" window and run its test. This window detects whether the list is handled by the mouse or by the keyboard.



- ▶ Stop the test and go back to the editor.
- ▶ Let's take a look at the code used, especially in the code for declaring the global variables of the WIN\_Event window.

```
EXTERN "WINCONST.WL"
```

First of all, the WINCONST.WL file is included in the code of the application via the EXTERN keyword. This file contains the declaration and the values of the Windows constants. When compiling, the entire code found in the Winconst.WL file will be automatically included in the code of the application.

Then, all the supported events are declared:

```
// Events on the LIST_Day control
// Keyboard key down
Event ("MouseOrKeyboard", LIST_Day..FullName, WM_KEYDOWN)
// Left mouse click
Event ("MouseOrKeyboard", LIST_Day..FullName, ...
 WM_LBUTTONDOWN)
```

The MouseOrKeyboard procedure is called whenever the keyboard is used on the list (corresponding Windows event: WM\_KEYDOWN) or whenever the left mouse click is used (corresponding Windows event: WM\_LBUTTONDOWN).

The code of the procedure is as follows: the global variable Keyboard is set to True or to False according to the tool that was used (keyboard or mouse).

```
PROCEDURE MOUSEORKEYBOARD()
// Defines whether the validation of the element was performed
// with the keyboard or with the mouse
SWITCH _EVE.wMessage
 CAS EWM_KEYDOWN // It's the keyboard
 Keyboard=True
 CASE WM_LBUTTONDOWN // It's the mouse (click down)
 Keyboard=False
END
```

The message is then displayed by the process associated with the LIST\_Day control.



## LESSON 8.5. THE THREADS

### This lesson will teach you the following concepts...

- What are threads?



Estimated time: 10 min

### Definition

The threads are used to run a code (or processes) in parallel of the main application. Several processes can be run in background task without locking the main application.

The threads replace some types of timers.

An efficient thread is a thread that waits for an event such as a user action, an incoming email, an incoming phone call, ...

Examples for using the threads:

- Retrieving emails in background task while a new email is typed.
- Application for communication: management of phone calls, communications via socket, ...

WinDev enables you to:

- easily manage the threads (functions: **Thread\***, **ThreadExecute**, **ThreadStop**, ...)
- use the threads with the "semaphores" in order to limit their simultaneous action to a given time (**Semaphore\*** functions)
- manage the "signals"; in order to synchronize several threads (functions: **Event\***, **EventCreate**, **EventDestroy**, ...)

### Example



Example

WinDev is supplied with an example that presents the interest and the use of threads: "Pool of threads" (this example is accessible from the "Wizards, Examples and Components" pane of WinDev).

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

## LESSON 8.6. SOCKETS

### This lesson will teach you the following concepts...

- Overview
- Server Application
- Client Application



Estimated time: 20 min

### 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 can be used, for example, to establish a communication between computers connected by Internet.

A WinDev application can manage the sockets according to different modes:

- Client WinDev/WebDev application: the application connects to a standard server and exchanges data via a socket.
- "Simplified server" WinDev application: the WinDev application is a server, exchanging information via a socket with a single client computer (WinDev socket recommended on the client computer but not mandatory).
- WinDev "Standard server" application: the WinDev application is a server that exchanges information by sockets with several client computers.



#### Example

WinDev is supplied with an example that presents the use of sockets: "WD Using sockets" (this example is accessible from the "Wizards, Examples and Components" pane of WinDev).

### Practical example

The "WD Advanced programming.WDP" project contains a window corresponding to both a "socket server" type application and a client application.

- ▶ If needed, open the example project, "WD Advanced programming": on the "Home" pane, in the "Online help" group, expand "Tutorial" then select "Advanced programming". The project is loaded.
- ▶ Open the "WIN\_Sockets" Window.
- ▶ Run the test of this window.
  1. Enter a message.
  2. Click the "Send" button. The message is sent to the socket server.
  3. The message is read by the socket server and displayed in the list.

## Server Application (simplified)

WinDev gives you the ability to create a simplified socket server. **This server enables you to communicate with a single client computer at a time.** This type of application is very useful when two remote applications must communicate between themselves.

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 uses **SocketCreate**. A socket is associated with a specific port. To simplify the use of the socket by programming on the server, specify the name of the socket.

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 use the same socket, a communication stream is established between these two computers. These two computers can read and write character strings on the socket.

**Note:** To avoid locking applications, the management of the incoming messages is often performed by a specific thread.

To read the socket and write to the socket, the WinDev server application must use **SocketRead** and **SocketWrite**.

**Caution:** to perform a read operation, a write operation must have been performed. For example:

1. The client computer writes onto 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 returns a response to it.

### Closing the socket

To end the communication, the server closes the socket with **SocketClose**.

**Note:** the socket can be closed by the client computer.

### Practical example

- In the "WIN\_Sockets" window, the socket server corresponds to a **pListen** procedure. This procedure performs the following operations:
  - Socket creation.
  - Read the messages sent to the socket.
  - Process the messages.

The corresponding code is as follows:

```
PROCEDURE pListen()
 sIncomingMessage is string
 SocketCreate ("S1", 9000)

 LOOP
 sIncomingMessage = SocketRead("S1", False, 100)
 IF sIncomingMessage <> "" THEN
 ListAdd(LIST_message, sIncomingMessage)
 END
 END
END
```

- This procedure is performed in a thread, in the "Global declarations" process of the window. The code is as follows:

```
ThreadExecute("TH_Listen", threadNormal, pListen)
```

## Client Application

A client application of a socket server connects to a standard server in order to exchange information via a socket.

Example: A WinDev client 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, you must 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 use the same socket, a communication stream is established between these two computers. These two computers can read and write character strings on the socket.

**Note:** To avoid locking applications, the management of the 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**.

**Note:** you can also end the communication from the server computer.

**Practical example**

- ▶ In the "WIN\_Sockets" window, the client operations are performed when pressing the "Send" button.
- ▶ The operations performed are:
  - The connection to the socket
  - Sending a message to the socket
- ▶ The corresponding code is as follows:

```
IF NOT SocketConnect("S2", 9000) THEN
 Error("Cannot connect to the socket server")
 RETURN
END
SocketWrite("S2", "EDT_message")
SocketClose("S2")
```

Note: in this lesson, we've seen a "simple" communication between a server and a client: the client sends messages and the server processes the messages. Of course you can also create more complex applications.

You can for example, have two applications communicate with each other, both acting as client and server. In this case, thread management becomes very important to manage the sending and receiving for the applications.

## LESSON 8.7. FTP

**This lesson will teach you the following concepts...**

- Presenting the WinDev FTP functions.



Estimated time: 20 min

## Overview

FTP (File Transfer Protocol) is a standard protocol used to transfer files from a computer to another computer. 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 FTP. A standard FTP server is required.

## Connect 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.

Example of code:

```
UserName is string = "User"
Password is string = "Password"
FTPAddress is string = "ftp.mysite.com"
Res is int
Res = FTPConnect(FTPAddress, UserName, Password)
IF Res = -1 THEN
 Error("Unable to connect." + CR + ...
 ErrorInfo(errMessage))
ELSE
 Info("Connection successful.")
END
```

You can also specify the port number for establishing the connection 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 (keyword: "FTP, Functions") for more details.

## Sending a file

To send a file to an FTP server, all you have to do is use **FTPSend**:

```
// When establishing the connection to the server with FTPConnect,
// we have retrieved the connection number in the Res variable
// Transferring the "C:\MyDocuments\File.DOC" file to
// the "Temp" directory on the server.
Result is boolean = FTPSend(Res, ...
 "C:\MyDocuments\File.DOC", "/Temp")
```



Caution !

Pay great attention to the case (uppercase, lowercase characters) regarding 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, if a directory named "MyDirectory" is found on the FTP server and if you try to access "mydirectory", an error ("Path not found") will be returned by the FTP server.

## Retrieving a file

To retrieve a file from an FTP server, all you have to do is use **FTPGet**:

```
// When establishing the connection to the server with FTPConnect,
// we have retrieved the connection number in the Res variable
// Downloading the "/Document/File.doc" file located
// on the FTP server to the "D:\Temp" directory on
// the current computer
Res is boolean = FTPGet(Res, ...
 "/Document/File.DOC", "D:\Temp")
```

## Disconnecting from an FTP server

A WLanguage function enables you to disconnect from an FTP server. This function is named **FTPDisconnect**:

```
// When establishing the connection to the server with FTPConnect,
// we have retrieved the connection number in the Res variable
// Disconnection
FTPDisconnect(Res)
```

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 (keyword: "FTP, Functions") for more details.



## LESSON 8.8. EMAILS

### This lesson will teach you the following concepts...

- Discovering the Email functions of WinDev.



Estimated time: 20 min

### Overview

Several WLanguage functions allow you to manage the incoming and outgoing Internet messages (called "emails"). Furthermore, WinDev enables you to find out all the characteristics of an email:

- sender, recipients
- outgoing date, subject, message
- attachments etc.

WinDev proposes several methods for managing emails:

- The **POP3 and SMTP** protocols (most frequently used method). These protocols are protocols for email management recognized by all the service providers. These protocols allow you to directly communicate with the server, available at your ISP.
- The "<strong>Simple Mail API</strong>" (also called SMAIL or Simple MAPI): this email management mode is used by most of the Microsoft applications, primarily by Microsoft Exchange.
- The **Lotus Notes or Outlook messaging software**: these programs allow you to send and receive emails.
- The **IMAP protocol**: this incoming email protocol enables you to leave the emails on the server so they can be consulted from different email client programs or webmail.

In this lesson, we shall only present the management method via the POP3/SMTP protocols. Indeed, this mode is commonly used all over the world.

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

All the functions for email management start with "Email".



Example

See the "WD Mail" example (complete example) provided with WinDev, for more details about email management. This example is accessible from the "Wizards, Examples and Components" pane of WinDev.

### POP3/SMTP protocol: Principle

The POP3 protocol is used to receive the emails.

The SMTP protocol is used to send the emails

We won't go into details about the operating mode of these protocols.

To send or read messages via the POP3/SMTP protocols, you must:

1. Connect to the service provider (if necessary).
2. Start an email session with **EmailStartSession**.
3. Send and read the messages.
4. Close the messaging session with **EmailCloseSession**.
5. Disconnect (if necessary).

To send and read messages, WinDev offers two email management modes:

- the email structure. This structure is a preset WLanguage structure.
- an email type variable. This type allows you to handle several messages at the same time.

### Practical example

The "WD Advanced programming.WDP" project contains a window allowing you to send emails (most common case).

- ▶ If needed, open the example project, "WD Advanced programming": on the "Home" pane, in the "Online help" group, expand "Tutorial" then select "Advanced programming". The project is loaded.
- ▶ Open the "WIN\_Manage\_Email" window.
- ▶ Run the test of this window.
  1. Enter the characteristics of the email: SMTP server, sender address, email text, recipient address, ...
  2. Click the "Send" button for the chosen method:
    - "Send method 1" uses the email structure
    - "Send method 2" uses the email variable
  3. Check your favorite email client program to confirm that the message was sent.

## Sending an email: the steps

### Opening an SMTP messaging program

The message is sent using the SMTP protocol. To open a messaging session, you can:

- use **EmailStartSMTPSession** and specify the address of the SMTP server to use. In our example, this function is used in the code of the "Send method 1" button. The code used is as follows:

```
IF NOT EmailStartSMTPSession(sUserName, sPassword, ...
 EDT_SMTPServer) THEN
 Error("Cannot connect to the SMTP server")
 RETURN
END
```

- use **EmailStartSession** with an EmailSMTPSession type variable. In our example, this function is used in the code of the "Send method 2" button. The code used is as follows:

```
MySession is EmailSMTPSession
MySession.Name = EDT_User
MySession.Password = EDT_Password
MySession..ServerAddress = EDT_SMTPServer
IF NOT EmailStartSession(MySession) THEN
 Error("Cannot connect to the SMTP server")
 RETURN
END
```

### Preparing the email

To send a message with WinDev, all you have to do is specify the main information of an email. This information is found:

- in the preset Email structure of WLanguage. In our example, the "Send method 1" button uses this structure. The code to fill the structure is as follows:

```
EmailReset()
Email.Sender = EDT_Sender
Email.Subject = EDT_Subject
Email.Message = RTFToText(EDT_BODY)
Email.HTML = RTFToHTML(EDT_BODY)
Email.NbRecipient = 1
Email.Recipient[1] = EDT_Recipient_Addr
IF SC_FilePicker.EDT_FILE <> "" THEN
 Email.NbAttach = 1
 Email.Attach[1] = SC_FilePicker.EDT_FILE
END
```

- in the advanced Email variable. In our example, the "Send method 2" button uses this variable. The code to fill the variable is as follows:

```
MyMessage is Email

EmailReset(MyMessage)

MyMessage..Sender = EDT_Sender
MyMessage..Subject = EDT_SUBJECT
MyMessage..Message = RTFToText(EDT_BODY)
MyMessage..HTML = RTFToHTML(EDT_BODY)

Add(MyMessage..Recipient, EDT_Recipient_Addr)

IF SC_FilePicker.EDT_FILE <> "" THEN
 EmailLoadAttachment(MyMessage, ...
 SC_FilePicker.EDT_FILE)
END
```

Note: This example supports sending an attachment. Of course, you can send several attachments. See the online help for more details (keyword: "Send an email").

### Send the email

To send the email, simply use **EmailSendMessage**.

### Disconnect

To disconnect, all you have to do is use **EmailCloseSession**.

## Other possibilities

---

You can also:

- perform mailshots. See the WD Mail" example (educational example found in the "Wizards, Examples and Components" pane) for more details".
- handle the Outlook data (via the email functions of WLanguage). See the WD Outlook" example (full example found in the "Wizards, Examples and Components" pane) for more details".
- handle the Lotus Notes data (via the Notesxxx functions of WLanguage). See the Accessing the Notes and Outlook databases" example (educational example found in the "Wizards, Examples and Components" pane) for more details".

See the online help (keyword: "Email, Managing the emails") for more details.

## LESSON 8.9. OOP

### This lesson will teach you the following concepts...

---

- Object-oriented programming concept.
- Examples of object declaration.



Estimated time: 30 min

## Concepts

Object-Oriented Programming (OOP) is designed for better code reusability. The programs developed in OOP are structured: they are made of modules, each one of these modules being used to manage a feature of the software. These modules can easily be re-used in other software. They contain a set of procedures and they encapsulate the data structure on which the procedures will act.

To use "object" 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) used to 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 being an instance of the class.

Once 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 the 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 the standard procedures of WLanguage.

### 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 enables you to create a new type of data 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.

A **derived class** inherits from the members and methods of its parent class (which can, itself, be a sub-class of another parent class), in addition to its own members and methods (and also from the members and methods of the first parent class if it is a multiple inheritance). The members and the methods of the main class(es) do not have to be duplicated.

## Constructor and Destructor

The notion of **Constructor** and **Destructor** is important because it allows an automatic call to initialization methods when creating an object or when destroying it.

The Constructor method associated with a class is automatically called when declaring an object of the class.

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 make sure that the data belonging to the object is not accidentally modified by functions (methods) external to the object.

This enables 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 use a simple example to apply the concepts we just saw:

- The class is the PERSON class.
- Florence is an instance of the PERSON class.
- The last name, first name and date of birth can be members of the PERSON class.
- The Age[] method can be a method of the PERSON class. It would calculate the age based on the "date of birth" member.
- 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 members of the class
2. Specify all the methods of the class
3. Declare the objects by associating them with a class ("instantiate a class")

► If needed, open the example project, "WD Advanced programming": on the "Home" pane, in the "Online help" group, expand "Tutorial" then select "Advanced programming". The project is loaded.

► Open the "WIN\_OOP\_Example1.WDW" window and test it.

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 (keyword: "OOP, Class") and the programming guide of WinDev for more details.

## Declaring a class

To create a class:

1. In the project explorer, select the "Classes" folder.
  2. Display the popup menu of the option (right mouse click) and select "New class".
  3. In the window that opens, specify the name of the class and validate.
  4. The code of the class comes up in the code editor.
- Consult the code of the CFile class used in the example.
1. In the project explorer, select the "Classes" folder.
  2. Open the "Classes" folder (by clicking the arrow found in front of the folder).
  3. Double click the CFile class.
  4. The code of the class comes up in the code editor. The class declaration code has the following format:

```
CFile is class
 m_sLongName is a string
 m_sShortName is string
 m_sExtension is string
 m_sShortPath is string
 m_sFullPath is string
END
```

"CFile" is the name of the class.

"m\_sLongName", "m\_sShortName", "m\_sExtension"... are the members of the class.

## Describing the methods

To create a method:

1. Right-click your class found in the project explorer.
  2. Choose "New method" from the popup menu.
  3. In the window that opens, specify the name of the method and validate.
  4. Enter the code of the method in the code editor.
- To display the **FileSelection** method of the **CFile** class:
1. Click with the mouse on your class found in the project explorer.
  2. Click the little arrow in front of the name of the class: the different methods of the class are displayed.
  3. Double-click the method name:

```
PROCEDURE FileSelection()
sSelectedFile is string

//Open the file picker
fSelectedFile = fSelect(fCurrentDir(), "", ...
 "Image selection", fGraphicFilter(), "png", ...
 fselOpen + fselExist)
```

```
//Assign the members of the current object
:m_sLongName = fLongName(sSelectedFile)
:m_sShortName=fShortName(sSelectedFile))
:m_sExtension = fExtractPath(sSelectedFile, fExtension)
:m_sFullPath=fLongPath(sSelectedFile)
:m_sShortPath=fShortPath(sSelectedFile)

RETURN sSelectedFile
```

## Declaring and handling objects

In the processes of the window, an object is declared at the same time as the other variables:

```
GLOBAL
 File1 is object CFile
```

To refer to a member of the "File" object, use the following syntax

```
<ObjectName>.<member name>
```

- In our example, the object is handled during the file selection. The code that uses the method's classes is placed in the button used to select the file.
- Display the code of this control (right click the button, "Code" option of the popup menu). The object is handled as follows:

```
//Call to the method
sFile = gclMyFile.SelectFile()
IF sFile ~= "" THEN
 RETURN
ELSE
 EDT_File = sFile
END

GR_FILEINFORMATION..Visible = True
GR_FILEINFORMATION..State = DisplayOnly

EDT_LongName =gclMyFile:m_sLongname
EDT_ShortName = gclMyFile:m_ShortName
EDT_Extension = gclMyFile:m_sExtension
EDT_FullPath = gclMyFile:m_sFullPath
EDT_ShortPath = gclMyFile:m_sShortPath
EDT_Size = gclMyFile:FileSize()
EDT_Date = gclMyFile:FileDate()
EDT_Time = gclMyFile:FileTime()
```

- Open the "WIN\_OOP\_Example2.WDW" window and test it. This window presents an example of object-oriented programming that handles the inheritances and the Constructor methods.

We won't go into details about OOP in this tutorial.



PART 9

Advanced project  
management

18



DEVELOP 10 TIMES FASTER PC SOFT

## LESSON 9.1. DASHBOARD

**This lesson will teach you the following concepts...**

---

- Overview
- View mode
- Dashboard options



Estimated time: 20 min

## Overview

The project dashboard is a main element for managing the WinDev projects. The project dashboard gives an overall view of the progress status of a project.

Some features of the dashboard were already presented in part 2.

We are now going to study it in details and see how it can interact with the Control Centers.

### Example

To work with the dashboard, we're going to go back to the "WD Application with data" project. If this application was not created beforehand, a corrected version is available. To open the corrected project, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Application with data (Answer)". The project is loaded.

## The different elements of the dashboard



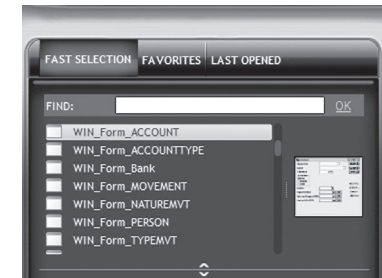
The project dashboard includes:

- lists of elements
- lights
- counters

## The lists

The dashboard is made of various lists. These lists are used to optimize the access to the project elements.

For example, the **"Fast selection"** list is used to quickly find a project element. All you have to do is enter some letters found in the name of the sought element. The selection is automatically performed and a preview enables you to choose the requested element.



To find the WIN\_PersonX window: type "Per" and make your choice. A double click performed on the name of the element enables you to open it in the associated editor.



Tip

You want to find an element without being positioned in the dashboard? press [CTRL] + [E]... and the same features will be available.

**Another interesting list: the favorites.** Who does not have a window or a piece of code they use frequently in their application? A window that groups the main features for example? To avoid losing time searching for this object, all you have to do is include it in the favorites.

For example, the "WIN\_PersonX" window found in our application will now be included in the favorites:

1. In the dashboard, click the "Favorites" button and click the link for displaying the favorites pane.

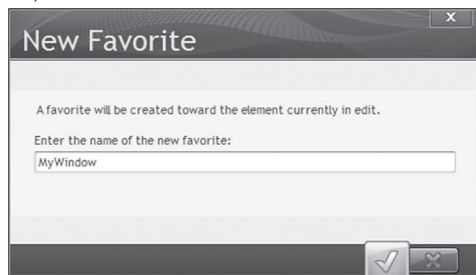


Note

To display the favorites pane, you can also on the "Home" pane, in the "Environment" group, expand "Panels" and select the "Favorites" option.

2. Display the "WIN\_PersonX" window in the editor (double-click its name in the WinDev "Project explorer" pane for instance).

3. In the "Favorites" pane, click the "+" button. You can now enter the name of the favorite (MyWindow for instance).



Validate. The favorite is displayed.

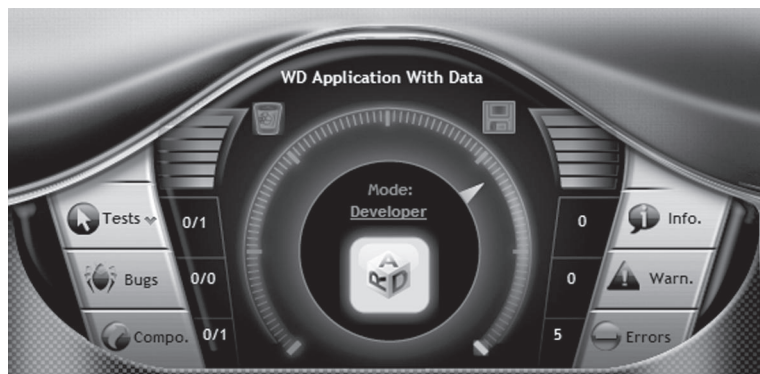


**Note** If several developers work on the same project, you have the ability to view "All the favorites".

4. In the dashboard, you can also see your favorite by pressing F5.

## The lights

The light regarding the tests was presented in part 2. Let's take a look at the other lights. The information regarding the **global quality of the project** is grouped in a specific counter.



In the counter, you will find:

- A progress bar indicating the overall quality of the project,
- A progress bar for backup. All you have to do is click the progress bar to save the project.
- A progress bar for cleaning. All you have to do is click the progress bar to clean the project.
- Warning lights: components that are not updated, compilation errors that occurred, detected information, detected warnings.

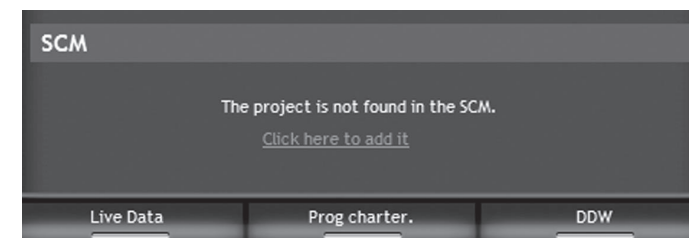
The dashboard also includes optimization lights:

- **Static audit:** If this light is red, this means the static audit has not been run for a long time on your project. The static audit is used to get the status of your project. It allows you to:
  - **detect dead code**, that is procedures unused in your application.
  - **orphan elements**, which means list the unused elements of your project (window used to run the test of the application during the development step for example).
  - **detect whether the project profiler was run:** the static audit informs you if the performance profiler has not been run recently on your application. We'll look into the performance profiler in the following chapter, "The performance profiler", page 463.
  - **detect whether query optimization is needed:** the static audit informs you if your project uses queries that can be optimized by a simple modification of the analysis (adding a composite key for example).

Don't hesitate to run the static audit: it can really improve the performances of your application.

- **the infrastructure audit:** If this light is red, it means that the infrastructure audit has not been run since a long time on your project.
- **the dynamic audit:** If this light is red, it means that the dynamic audit has not been run since a long time on your project. The dynamic audit is used to perform an audit during the test or the execution of the application. You can find out:
  - the memory footprint of the different elements used by the application.
  - the images not found
  - the patches not taken into account
  - ...

The dashboard also enables you to find out and modify some characteristics of the project. These elements are also represented as lights: the light is green? It means that the feature is enabled in your project. Otherwise, a click performed on the light is used to enable or configure the feature. Let's take a look at the available elements:



- **SCM:** Enables you to find out whether the project is shared via the Source Code Manager. This feature will be presented in the next lesson ("SCM", page 424).
- **Live data:** Allows you to use the content of the data files found on the development machine in the windows, reports, ... handled in the editor. This feature is very useful to define the size of the controls found in a window.
- **Programming charter:** As already seen in part 1, the programming charter is used to standardize the names of the elements found in a project. As a programming charter is used by our project, this light is green.
- **DDW (Dim Disabled Windows):** This option indicates whether the inaccessible windows of your application will be automatically grayed or not. This feature is very appreciated by the users. When several windows are stacked, the inaccessible windows are automatically grayed.



- **Languages:** enables you to find out the number of languages supported by your project. A single language is supported in our case. A click performed on this light enables you to add new languages to your project. The management of multilingual projects will be presented later in this tutorial ("Multilingual", page 487).
- **Skin template:** Skin templates are used to easily define the style book of your application. A click performed on this light enables you to access the screen for configuring the skin template of your application.
- **Reports and Queries:** As we've already seen, Reports and Queries allows the end user to create his own reports and his own queries. To include it in your application: a light is all you need. See "Distributing "Reports & Queries" with your applications", page 369 for more details.

### The counters in relation with the Control Centers

WinDev is supplied with several Control Centers. The Control Centers are used to control a set of features. You are already familiar with the HyperFileSQL Control Center, that is mainly used to control the applications and the data files found in the HyperFileSQL Client/Server applications.

WinDev also proposes the Project Monitoring Center. This center is used to manage the projects, from design to distribution and even maintenance.

We will only present the elements in relation with the dashboard. For more details about Control Centers and their configuration, see "Control Centers", page 435.

#### The bug counter

The bug counter represents the number of bugs currently found in the project. These bugs are referenced in the Quality Control Center.

These bugs have been reported by the users of the application, via the "? .. Send feedback..." option of their application. This option is automatically proposed if you include the automatic menu, "?" in your application.

#### The task counter

The task counter represents the number of tasks currently in progress in the project. These tasks are referenced in the Project Monitoring Center.

These tasks correspond to:

- features that must be added to the project
- the forthcoming development, ...

#### The request counter

The request counter represents the number of requests currently found in the project. These requests are referenced in the Quality Control Center.

These requests have been made by the users of the application, via "? .. Send feedback..." option of their application. This option is automatically proposed if you include the automatic menu, "?" in your application.

#### The message counter

The message counter represents the number of messages associated with the project. These messages are visible in the messaging of WinDev. These messages are automatically sent when working with the Source Code Manager for example. This feature enables you to be automatically notified when an element is updated in the SCM.

#### The rule counter

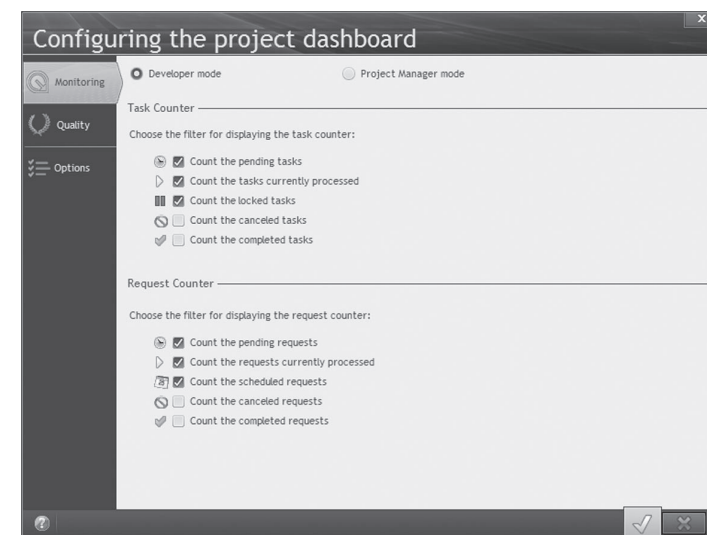
The rule counter is used to find out the number of business rules defined for the project. The business rules will be presented later in this tutorial. The business rules can be entered:

- by yourself, the developer of the application
- by the users, via "? .. Send feedback..." option of their application. This option is automatically proposed if you include the automatic menu, "?" in your application.

### Configuring the dashboard

We have presented the main categories of information displayed in the dashboard. WinDev proposes two modes for viewing this information: Developer mode and Project Manager mode. To change mode, click "Mode" found in the round gage of the dashboard. The levels of alert for the dashboard can be configured for each mode.

To configure the dashboard, select "Dashboard options" from the popup menu of the dashboard. The following elements can be configured for the two modes:



- the counter of tasks and the counter of requests,
- the cleaning,
- the backup,
- the quality of the project,
- the position of the dashboard, ...



## LESSON 9.2. SCM

### This lesson will teach you the following concepts...

- Overview
- The Source Code Manager
- Using the Source Code Manager



Estimated time: 30 min

### 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 resources (windows, classes...).

WinDev is supplied with a Source Code Manager named "SCM" used to share the source code of different projects among the developers and to find out the full history of the modifications performed (in the code, in the interface, ...).

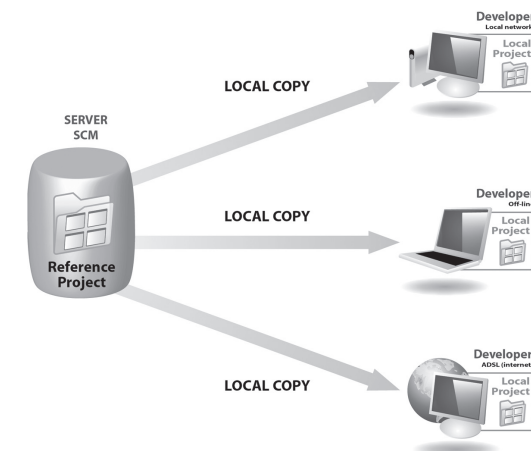
### SCM (Source Code Manager)

#### Principle of SCM

The Source Code Manager is used to store and share the projects and their elements.

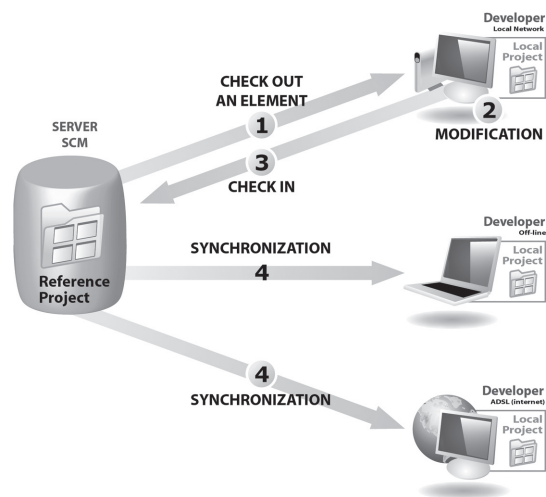
The principle is as follows:

- A reference version of each one of your projects is found on a server. All these versions are called a "SCM database".
- Each developer has a local copy of the 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, this element will be checked out from the SCM database by the developer.
- Each developer gets exclusive rights on this element: all the requested modifications can be made to this element.
- The other developers are still working on the copy of the reference version of this element (found in the SCM database).
- Once the modifications have been made by the developer, the checked-out element is checked in into the SCM database.

- The other developers are automatically notified of this check-in operation. They can now update their local copy.



The SCM supports teamwork and it enables you to find out the history of all the modifications. The SCM can also be used to manage and control the elements shared among several projects.

### Creating the SCM database

To share a project via the Source Code Manager, a SCM database must be created. This SCM database must be created once only on a server.

This SCM database can be created:

- when installing WinDev.
  - when creating a project that uses the SCM.
  - when importing a project into the SCM.
  - in the SCM administrator directly.
- Our SCM database will be created when a project is imported into the SCM (next step).



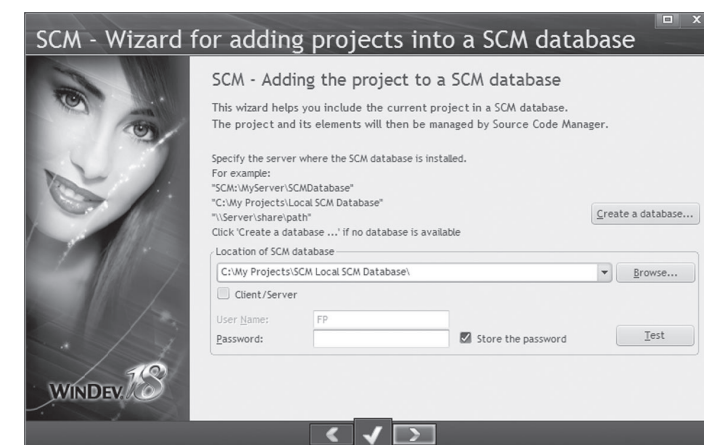
Notes

We advise you to perform backups of the SCM database on a regular basis. To do so, connect as administrator to the tool for SCM management and select "Tools .. Management .. Full database backup".

### Including a project in the SCM

To use an existing project with the Source Code Manager, all you have to do is include this project in the SCM database.

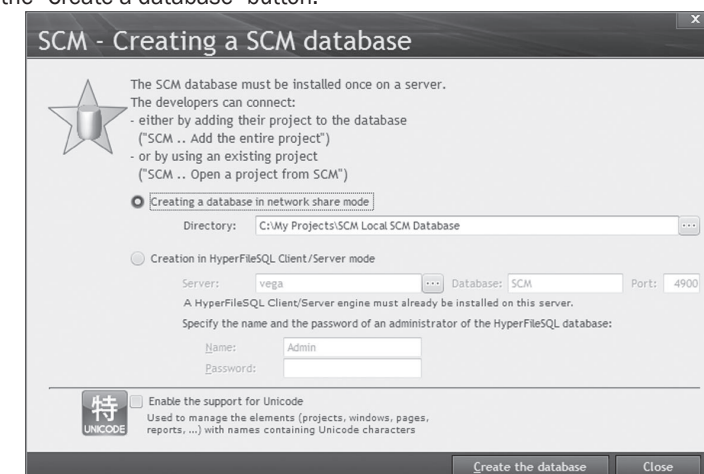
- We are now going to include the "Windows and Controls.WDP" project in the SCM database:
1. Open the "WD Windows and Controls.WDP" example project: on the "Home" pane, in the "Online" group, expand "Tutorial" then select "Windows and controls". The project is loaded.
  2. On the "SCM" pane, in the "Project" group, click "Add the project". The wizard for adding projects into the SCM starts:



The SCM database has not been created yet. We are going to create one.

Note: We are going to create a "local" SCM database (on the development computer). The operating mode would be similar for a network SCM database.

3. Click the "Create a database" button.

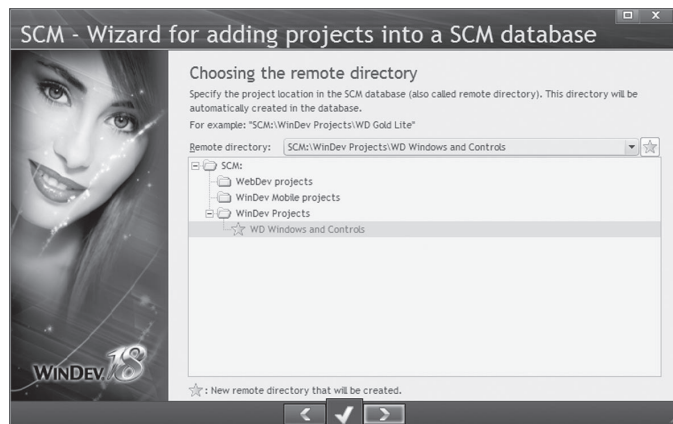


The SCM database can be in HyperFileSQL Classic format (local or network) or in HyperFileSQL Client/Server format. We are going to create a SCM database in HyperFileSQL Classic format.



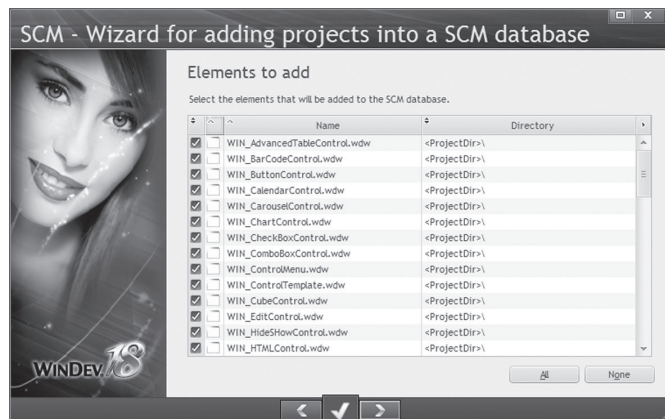
**Notes** If the SCM database is in HyperFileSQL Client/Server format, this SCM database can be used remotely.

4. Keep the "Creating a database in network share mode" option selected and specify the directory of this SCM database ("C:\My Projects\My SCM database" for example). Validate the creation of the SCM database ("Create the database" button). The SCM database is now created. Our project can be included in this SCM database.
5. Go to the next screen.
6. The wizard proposes to place the project in the "WinDev Projects" sub-directory of the SCM database.



Accept this location. Go to the next screen.

7. The wizard asks you to select the project elements that must be added into the SCM database.



We want to add all the project elements. Go to the next screen.

8. The wizard asks you to select the project dependencies that must be added into the SCM database. These dependencies correspond to all the external elements required by the project (images, style sheets, ...).

We want to add all the project dependencies. Go to the next screen.

9. Validate the inclusion of the project in the SCM. The project and all its elements have been added into our SCM database.

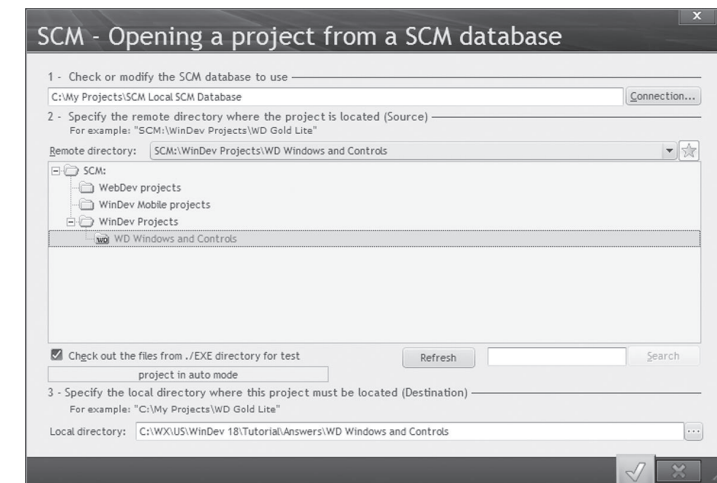
A help window on using the SCM toolbar comes up. Read and validate this window.



**Notes** When the projects that share the same resources (same analysis, same windows, ...) are included in the SCM, the relevant elements can be shared among the projects. Therefore, the same element is checked in once only into the SCM.

## Opening a project from the SCM

To work on a project found in the Source Code Manager, you must retrieve a local copy of this project. To do so, open the project from the source code manager: on the "Home" pane, in the "General" group, expand "Open" and select "Open a project from SCM".



This operation must be performed once only by each developer who uses the project.

The developer who added the project to the Source Code Manager (you in this case!) has no operation to perform.



**Notes** The method for opening a project managed by SCM is identical to the method for opening a project not managed by SCM: all you have to do is open the project (".WDP" file) corresponding to the local copy.

## Configuring the SCM

The SCM must be configured before we start working on the project elements found in the SCM.

The important points of this configuration affect:

- the check-out mode of the project elements.
- the check-out mode of the project (WDP file).

### Check-out mode of the project elements

When working on the elements of a project found in the SCM, the element must be checked out from the SCM database before it can be modified, then it must be checked back in once the modifications have been performed. The modified element is available to all the 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 dialog box indicates that this element must be checked out before it can be modified. The element can be checked out immediately (check-out button found in the panel).
- **the automatic mode:** if you try to modify a SCM element that is not checked out, the SCM automatically proposes to check this element out. Once the check-out is validated, the element can be modified.

**Note:** this mode is not recommended when the SCM is used with a slow Internet connection.

In this tutorial, we will be using the automatic check-out.

- 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 whether "Checking out elements during the first modification" is selected. If not, enable this option.

### Check-out mode of the project

The Project file (WDP file) contains the different options used to configure the project (initialization code of the project, list of linked elements, name of the first project window, ...).

WinDev proposes two management modes of the project:

- **Master/Guest mode:** Only the master can modify the project and apply these modifications to the SCM database. The master can also check in all the elements to create the executable and the setup program. The modifications made to the project by the guests will not be taken into account by the SCM database.



Notes

The Master/Guest mode is recommended when SCM is used by a single developer.

- **Automatic mode:** The project file is checked out only if the action performed requires this check-out (regardless of the user). Once the action has been performed on the project, the project file is automatically checked back in.



Notes

The "Automatic" mode quickly becomes essential for the large projects shared by several developers.

In this tutorial, we will be using the automatic check-out.

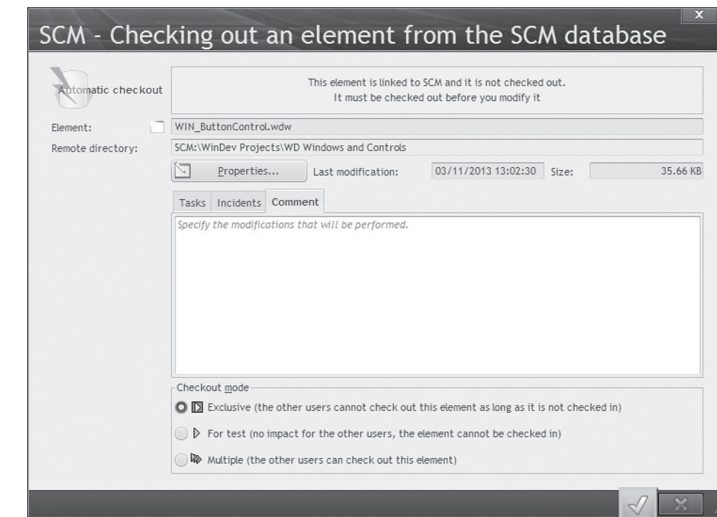
- To make sure that the automatic check-out is enabled, on the "SCM" pane, in the "Project" group, expand "Master/Guest" and make sure that the "Manage the project checkout automatically" option is checked.

## Checking out an element

We are now going to start working with the SCM.

To modify a project element, this element must be checked out.

- We are going to check out the "WIN\_ButtonControl" window:
  1. Select the "WIN\_ButtonControl" window in the project explorer. This window is found in the sub-folder named "Standard controls".
  2. Double-click the element to open it in the window editor.
  3. Display the code of the F2 button ("Code" from the popup menu).
  4. We are going to enter a comment before the line "Info ...". Position the cursor and press the ENTER key.
  5. The window for automatic check-out is displayed:



- 6. Three check-out modes are available:

- **Exclusive:** nobody else will be able to 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 versions of the element can be displayed when the element is checked back in. This mode is reserved to specific cases and to experienced developers.

- 7. The window will be checked out in exclusive mode. Keep "Exclusive" checked.

- 8. Enter a comment ("Add a comment for F2" for example). This comment will be useful for the other developers.

- 9. Validate the checkout.

The window is checked out. The code can be modified.



## Modifying the checked-out element

The method for modifying a checked-out element (GUI, code, ...) is the same as the method for modifying an element in a project not managed by the SCM.

However, the modifications made to 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 SCM database will be used.

This enables you to modify an application while keeping a steady version in the SCM database.

### ► Modify the checked-out window.

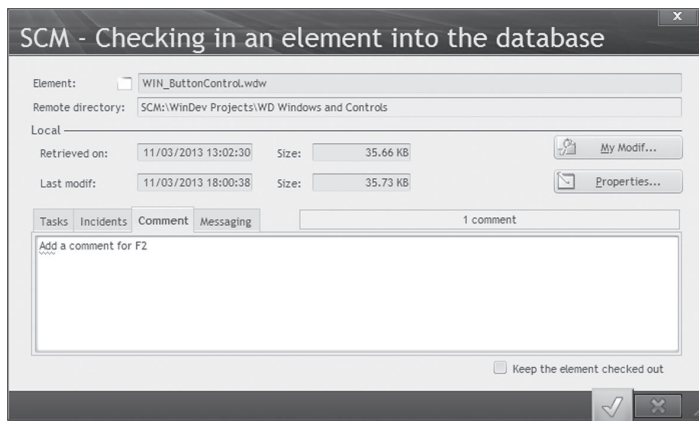
1. Enter the following comment: "// Display the key used". Close the code window.
2. Change the location of the controls, modify the captions, add controls or code, ...
3. Save your window (CTRL + S).

### ► Run the test of your modifications.

## Checking in the checked-out element

Now that the modifications are over, we are going to check in the window into the SCM. Then, your modifications will be accessible to the other developers.

- On the "SCM" pane, in the "Current element" group, click the "Check in" button. The following screen is displayed:



This screen allows you to:

- find out the modifications performed by comparing the element found in the SCM database with the checked-out element ("My Modif" button).
- access the history of the element found in the SCM database ("Properties" button).
- enter a comment about the modifications performed. By default, WinDev proposes the comment entered 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 the element checked out").



Notes

If you are using the Control Centers, the current task can be ended when the element is checked back in into the Source Code Manager. This feature is useful to follow the monitoring of the tasks, the corrections of bugs, ...

- Validate the check-in.

## Synchronizing the project

Several options can be used to configure a project handled by the SCM. These options are grouped in the "SCM" tab of the project description (displayed by clicking "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 database, this option proposes to retrieve the latest version of the project elements.  
By default, the latest version of the elements is automatically retrieved.
- *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 in when the project is closed.
- *Propose to check in and to get the latest version of the elements when generating executables, libraries, components, ...*  
This option enables you to display when creating an executable, a library, ... the list of checked-out elements in order to check them back in and to get the latest version of these elements.  
Therefore, the executable, the component or the library will contain the most updated elements.  
By default, the generation of the executable, the library, ... is performed with the project elements currently found on the local computer.



Notes

### Merging code

An element can be compared to one of its earlier versions. This enables you to compare the code in order to retrieve a section of code that was "lost" or accidentally deleted by another developer.

- *Check out/Check in the project automatically.*  
This option is used to automatically check out or check in the project when handling an element.  
By default, this option is selected.
- *Include the new elements only in the current configuration.*  
This option allows you to manage elements only in the current configuration.  
By default, this option is not selected.



**Off-line mode (or mobile mode)**

The SCM allows you to work in off-line mode.

This mode allows a developer who is using a laptop computer to continue to work on a project found in the SCM database while being disconnected from the SCM database.

The principle is straightforward:

- before the disconnection, on the "SCM" pane, in the "Other actions" group, expand "Remote work" and select "Disconnect for a mobile use".
- 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 in the modified elements.

In mobile mode, two solutions are available for checking out elements:

- No element is checked out from the SCM. Other developers will be able to work on the same elements as you while you are working in off-line mode. When you reconnect to SCM, the modifications you made 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. Nobody can use the element while you are working in off-line mode.

**SCM administrator**

The SCM administrator enables you to handle the different projects included in the Source Code Manager.

It allows you to:

- manage the scm databases (creation, connection to a scm database).
  - manage the files and the directories found in a project of the scm database (add, delete, rename, ... files and directories).
  - manage the different files found in the scm database (check-in, check-out, share, ...).
  - start some tools (options, maintenance, ...).
  - view the history of an element.
  - view the status of the elements.
  - perform backups.
  - grant rights to the different users of SCM.
  - list the projects in which you are involved in order to dissociate from them (if necessary).
- Start the SCM administrator: on the "SCM" pane, in the "SCM database" group, click "Manage". All the project elements are listed in the administrator.

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

## LESSON 9.3. CONTROL CENTERS

**This lesson will teach you the following concepts...**

- Overview
- The Project Monitoring Center
- Monitoring the user feedback with the Control Centers
- The other Control Centers



Estimated time: 20 min

## Overview

WinDev proposes several tools for monitoring the development of a project, from design to maintenance. These tools are called the Control Centers.

The HyperFileSQL Control Center was presented in a previous chapter. This Control Center (redistributable) is used to manage the deployed HyperFileSQL Client/Server applications.

In this part, we will focus on the Control Centers linked to the development of an application:

- the Project Monitoring Center.
- the Quality Control Center.

These two centers use a specific database. This database can be:

- a HyperFileSQL Classic database: the path of this database was specified the first time WinDev was started. By default this database is installed in a subdirectory of WinDev 18.
- a HyperFileSQL Client/Server database.

This information can be modified at any time from the options of WinDev:

1. On the "Home" pane, expand "Options" and select "General options of WinDev".
2. On the "User" pane of the WinDev options, click the "Control Center Parameters").

## The Project Monitoring Center

The Project Monitoring Center is the heart of the organization. The Project Monitoring Center helps you organize and schedule a project, from design to delivery. It allows you to define the tasks to perform in a project, the developers working on it, the durations, the status of progress, ...

To test the Control Centers, we will be using our application named "WD Application with data".

- Start the Project Monitoring Center. To do so, on the "Tools" pane, in the "Control Centers" group, click "Project monitoring").



Notes

If the Project Monitoring Center has never been started before, a screen allows you to define the options for time management.

The first thing to do is to define the working "Project". In our case, this project corresponds to a WinDev project but it could be any other project.



Notes

The Project Monitoring Center helps you manage and schedule any project or task (development, business or other).

1. In the Project Monitoring Center, create a new project ("Management .. Managing the projects", "+" button).
2. Enter:
  - the name of the project ("Tutorial" for example)
  - its description ("WD Application with data" for example)
  - the type of application: Windows
  - the path of the project. In our case, specify a project path outside the SCM (the path of the "WD Application With Data.wdp" project for example, in the "Tutorial\Answers\WD Application With Data" sub-directory of WinDev).



Note

If the project is available in SCM (Source Code Manager), you have the ability to enter the SCM database and the remote directory corresponding to the project.

- a group
  - the associated spool (a new one can be created). The spool will be used to retrieve the requests made by the users.
  - the color associated with the project (used to easily identify the different projects).
3. Validate.
  4. The screen used to select the contributors of your project is automatically displayed. Leave yourself as contributor and validate.
  5. Close the window for project management.
- You now have the ability to define tasks on the project. We are going to create a task in order to make our project multilingual:
    1. Select "Tasks .. Create a new task". This task will be associated with the "Tutorial" project that was just created.

## 2. Specify the following characteristics:

## 3. Validate. The task appears in the task list.

You can create as many tasks as necessary. These tasks represent your schedule. These tasks can also be fixed tasks (a meeting at a given day and time for example) or recurring tasks (a meeting every Friday for example).

To enable a task, select "Start this task" from the popup menu of the task list.

The use of the Project Monitoring Center is straightforward: as soon as you want to perform a task, you select this task in the Project Monitoring Center and you specify that you "start this task" (from the popup menu of the task list). If your project is linked to a WinDev project, the corresponding task is enabled as soon as the project is opened in WinDev.

## Managing your time

Time management is extremely important. Where is time spent in a day? How to find out without adding constraints, without requiring some fastidious time keeping from team members and without making people feel like somebody is watching over their shoulder?

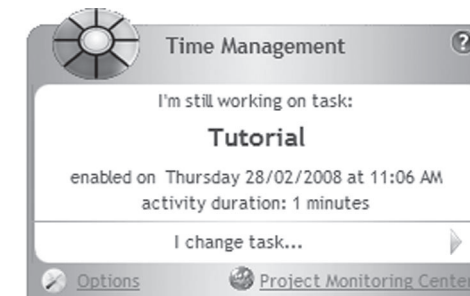
Entering the time spent on the tasks is an interesting feature of WinDev. This feature is linked to the use of the Project Monitoring Center.

**The principle is straightforward:** the list of tasks is entered in the Project Monitoring Center.

### ► To enable the help for time management:

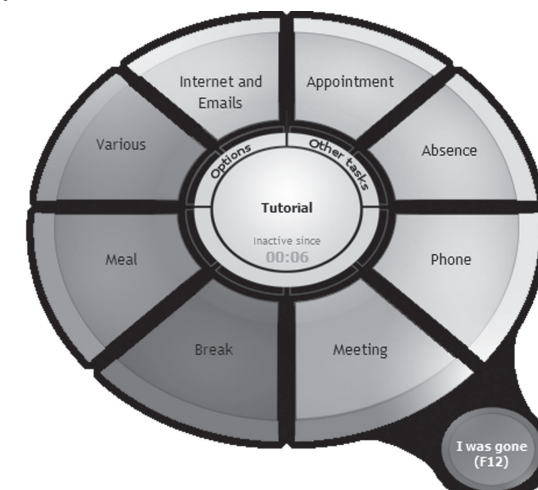
1. In the Project Monitoring Center, select "Parameters .. Options of Project Monitoring Center".
2. In the "Help for time management" tab, select "Enable the help for time management".
3. Validate.

The current task is called back at regular time intervals.



The user can validate the current task or change his current task if necessary. Furthermore, a "task wheel" is displayed if the computer is idle for a specified amount of time.

This task wheel enables you to select the interrupting task corresponding to the time spent while the wheel was displayed.



### Configuration

The time management can be enabled/disabled in the window for configuring the time management. This window can be displayed:

- in the Project Monitoring Center ("Parameters .. Options of Project Monitoring Center").
- in WinDev: to do so, on the "Project" pane, in the "Other actions" group, expand "Time management" and select "Options for time management".

### Monitoring the user feedback with the Control Centers

The purpose of the automatic menu was presented in part 2 of this tutorial. One of the options of this automatic menu is used to send suggestions, incidents or business rules to the developer of the application.

The management of the user feedback depends on whether the application is used in online mode or in offline mode.

**If the application is used in online mode** (the Control Center database is accessible to the users), the requests are directly saved in the Control Center database:

- the suggestions and the incidents are saved in the spool of the application.
- the business rules are automatically added to the business rules of the project.

**If the application is used in offline mode** (the Control Center database is not accessible to the users), the requests are emailed. The Project Monitoring Center must be configured to receive the emails (setting performed in the administrator options of the Project Monitoring Center):

- the suggestions and the incidents are saved in the spool of the application, then they are managed by the Quality Control Center.
- the business rules are automatically added to the business rules of the project.



Notes

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

### The other Control Centers

WinDev includes several other Control Centers, intended for the WinDev developers:

- **Reusability Center:** This center is used to centralize the resources (components, libraries, classes, ...) of your different projects in order to make them available to your development team.
- **Document Management Center:** This center is used to centralize the help systems and the documents generated for your projects.

WinDev also offers the Software Infrastructure Supervisor (SIS). This tool is used to generate a summary report about the status of your software infrastructure. You have access to various information regarding your servers and your applications: applications found, connected users, load, ...

## LESSON 9.4. BUSINESS RULES

### This lesson will teach you the following concepts...

- Overview
- Creating a business rule
- Validating a business rule



Estimated time: 20 min

## Overview

WinDev enables you to manage the business rules.

A business rule is used to define a specific operating mode or a specific process. For example: the calculation of a specific VAT rate, the rules for changing the status of a customer, the formula for calculating the shipping costs, ...

A business rule can be simple or complex, and it can affect one or more elements found in one or more projects.

### Example

To see the benefits of the business rules, we are going to create a business rule on the "WD Application with data" project created in part 2 of this tutorial.

If this application was not created beforehand, a corrected version is available. To open the corrected project, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Application with data (Answer)". The project is loaded.

## Application on a real example

### Creating a business rule

- Display the "WIN\_PersonX" window.
- We are going to modify this window in order to manage the persons under the age of 18. Indeed, specific conditions must be taken into account when a person under the age of 18 is associated with an account.
  1. Create a static control and move it beside the "Date of birth" control. This static control is named "STC\_NoName1"
  2. Enter the following code in the exit code of the "Date of birth" control:

```
Age_Individual is Date = Age(EDT_DateOfBirth)
IF Age_Individual..Year < 18 THEN
 // Indicate that the person is not over 18 years old
 STC_NoName1 = "Underage person" + CR + ...
 "Specific conditions"
 STC_NoName1..Color = DarkRed
ELSE
 // Reinitialize the static control
 STC_NoName1 = ""
END
```

This code calculates the age of the person, and if it is lower than 18, the static control displays in red the string "Person under 18".

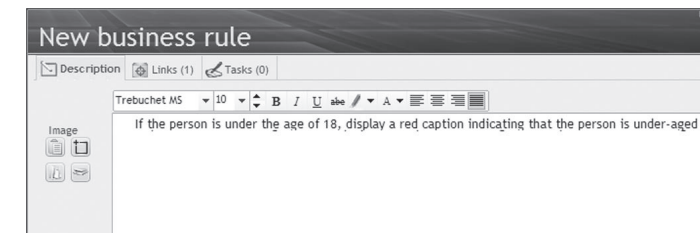
This code must be run:


- whenever the date of birth of a person is modified
- whenever the form of a person is displayed.

- We are going to create a procedure with this code in order to run it whenever required.
  1. In the code editor, select the code that was just entered.
  2. In the popup menu, select "Create a procedure .. Create a local procedure containing the selected code".
  3. Give a name to the procedure: CalculateAge.
  4. Copy the call to the **CalculateAge** procedure into the following elements:
    - Button for exact-match search.
    - Button for generic search.
    - Navigation buttons
- Run the test of the window. Perform a generic search on VINI and display VINI Emma.
- Now, we're going to create a business rule for this window.
 

In our case, the business rule will be applied to all the persons under the age of 18. In this case, a specific process must be performed: display a caption indicating that the person is under 18.

  1. Display the business rules pane. On the "Home" pane, in the "Environment" group, expand "Panels" and select "Business rules".
  2. By default, the new business rule will be created on the selected element: select "STC\_NoName1".
  3. Click the "+" button: the screen for entering the business rule is displayed.
  4. The description of the rule is as follows: "If the person is under the age of 18, display a red caption indicating that the person is under-aged".



5. We are now going to define the elements on which the business rule must be applied. Click the "Links" tab.
6. We are going to define an automatic link: the rule will be applied as soon as an element that respects the conditions of the rule is created. Click the "Conditions of automatic links" button (  ).
7. This business rule will be applied to the current project. Select "The current project only".



8. We are going to define the selection conditions of the business rule. Click "+" and specify the following information:

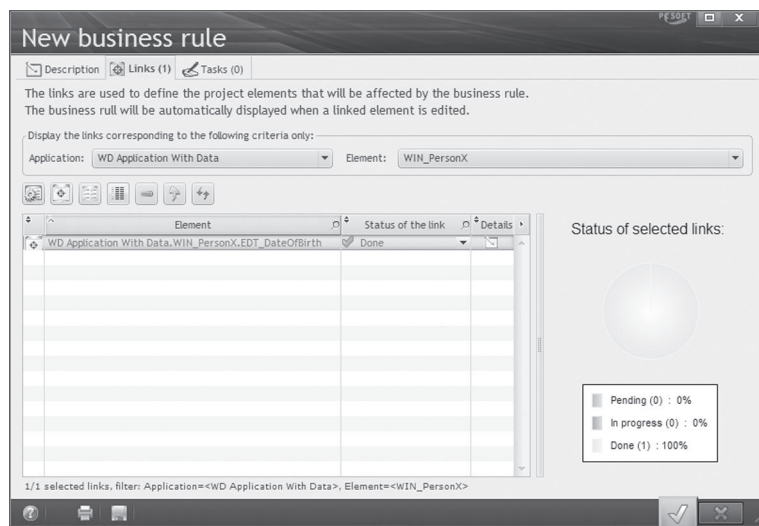


The rule will affect all the window controls linked to the "Person.DateOfBirth" item. Validate. The selection condition is displayed.

The existing project windows corresponding to the rule are automatically listed.

9. Validate. Then validate the definition window for the automatic link conditions.

10. The list of links for the "WIN\_PersonX" element is displayed. The status of the link can be marked as "Done" for the affected element in the "WIN\_Person\_X" window.



11. Validate the creation of the business rule.

12. Save your window.

## LESSON 9.5. MULTI-CONFIGURATION

### This lesson will teach you the following concepts...

- Overview
- Using the same project for different configurations



Estimated time: 20 min

## Overview

You have created an application and you want to create a component from some project windows? Your project contains procedures that could be used in a Web service? Part of your application can be used in Java?

How can I avoid the multiplication of projects? A single project and several destinations, that's the challenge that was resolved by the WinDev project configurations.

The project configurations are used to create several "targets" from the same project. You have the ability to choose the requested configuration at any time and to generate in a single operation all the elements for all the configurations of a project.

A practical example? We are going to test the project configurations on our full application.

## Creating a project configuration

We are going to create a project configuration in order to generate a 64-bit application from the "WD Application With Data" project that was created in part 2 of this tutorial. If this application was not created beforehand, a corrected version is available.

► To open this project in WinDev:

1. Close the current project if necessary. The home window is opened.
2. In the welcome window, click "Tutorial" and select "Application with data (Answer)". The project is loaded.

Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Application with data (Answer)".

► To create the project configuration:

1. In the "Project explorer" pane, select "Configurations (WD Application with data)".
2. Select "New configuration" from the popup menu. The wizard for creating a project configuration starts.

3. In the general information of the project configuration, specify the name (Test), the description (Test) and the type of generation ("Windows applications (desktop)" for our example).



Go to the next screen.

4. Specify the configuration platform (Windows in our case). Go to the next screen.
5. Specify the execution mode (64-bit in our case). Go to the next screen.
6. Specify the elements that will be included in the configuration. For our example, include all the elements. Go to the next screen.
7. Also include the external components. Go to the next screen.
8. Validate the creation of the configuration.
9. At the end of the wizard, the "Test" configuration is automatically selected in the "Project explorer" pane. Select the "WD Application with data" configuration then "Enable this configuration" from the popup menu.

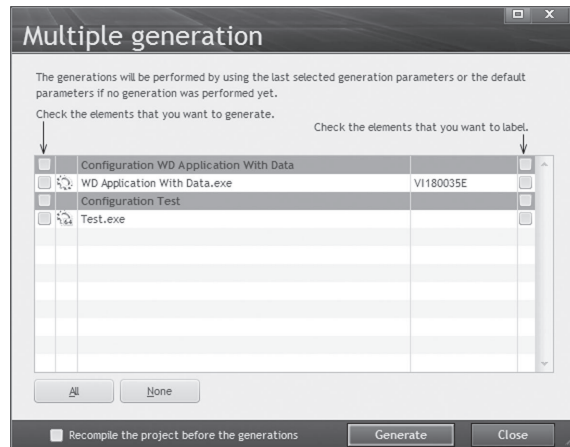
## Managing the project configurations

To define all the project's configuration options, on the "Project" pane, in the "Project configuration" group, click "Manage the configurations". You can:

- Create a new configuration or delete an existing configuration.
- Generate a configuration.
- Configure the test mode.
- Find out the description of the main element found in the configuration.
- Configure the actions that will be performed before and after the generation (copy the element to a specific directory for example).

## Multiple generation

We now know how to generate a configuration; however, after a modification, it is often necessary to generate the elements corresponding to all the configurations defined on the project. To do so, on the "Project" pane, in the "Generation" group, click "Multiple generation". Then, select the elements to generate. This generation will be performed in a single operation.



## LESSON 9.6. 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: 30 min

## Overview

A WinDev external component is a building block that can be re-used in an unlimited number of projects (and in executables).

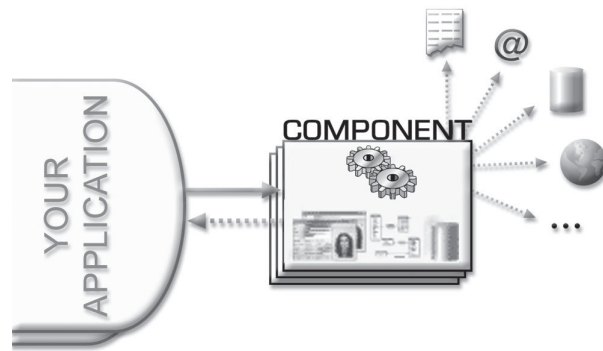
An external component enables you to have an identical process with a unique source code to perform a given operation even if this process must be performed by several projects.

The projects that use an external component have access in the WinDev editor to the name of the objects, procedures or methods made visible by the creator of the component. The projects cannot view or modify the source code. The confidentiality of the source code is guaranteed.

Creating an external component is **child's play**.

How do I proceed? Proceed as usual, create windows, procedures, classes. Then, instead of choosing the option for creating an executable, choose the option for creating a component and that's it!

A component may contain source code, windows, an analysis, data files, etc..!



Notes

Five methods can be used to share "code" in WinDev:

1. The sets of procedures
2. The classes
3. The supercontrols (via the use of a dictionary)
4. External components
5. Internal components

Let's present several cases where the external components can be useful.

### Teamwork

A developer creates and maintains an external component that is made available to the other developers. No risk of modifications "made by mistake"!

### Large projects

The external components allow you to have several small projects as well as a central project where the elements of the other projects are visible. Using external components is a lot more convenient than using 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 for accessing the database (at least in write mode) in a component, only a single source code must be checked and maintained; 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

You will often find more or less complex processes 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 the elements no longer compatible between themselves. Using external components prevents such out-of-sync modifications, preserves the compatibility between projects and facilitates the common modifications.

Furthermore, the PRIVATE keyword enables you to make sure that your source code remains confidential at all the levels of the external component. When your external component is re-used in another WinDev project, the source code cannot be edited but the documentation regarding the use of the functions (procedures for instance) will be displayed!

### The ability to distribute a feature or set of features

The external components are used 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 use 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 targeted for:

- a WinDev application
- a WebDev application
- a WinDev Mobile application
- the three types of applications.

In this last case, WinDev enables 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're going to create an external component using the "WD Printing reports" project. This project manages orders, products and suppliers. The external component that we're going to create is going to allow us to instantly get in an other application the customers corresponding to a given city.

When calling the external component, you will have the ability to:


- Pass a city in parameter.
- Retrieve a string containing the customer's name and the total amount of his/her orders.

To avoid having to develop the code required for the component to work, we have grouped all the necessary elements in the "WD Printing reports" project.

- ▶ Close the current project if necessary. The home window is opened.
- ▶ In the home window, click "Tutorial" and select the project named "Print reports (Exercise)". The project is loaded.  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Print reports (Exercise)".
- ▶ For the component's needs, the project contains:
  - a "QRY\_Order\_City". This query is a query with parameters that lets you obtain for a given city the corresponding customers and their total order amount.
  - a set of procedures "COL\_OrdersByCity". This collection of procedure contains the "GiveCityOrders" procedure, which for the city passed in parameter returns a string containing the customer's name and the total amounts of his or her orders.
  - an analysis used to described the data files storing the information.
- ▶ We are now going to create our component.
  1. On the "Project" pane, in the "Project configuration" group, expand "New configuration" and select "External component (WDK)".
  2. WinDev offers to create a "Component" type project configuration. Go to the next plane.
  3. Give a name to your project configuration: "CompOrdersByCity" for instance. Go to the next plane.
  4. Keep the default options. Go to the next screen.
  5. Select the elements that make up the project configuration. In this example, these are also the elements that will be in the component.
    - Click the "None" button.
    - Select the "QRY\_Order\_City" and "COL\_OrdersByCity" elements.
    - Select the data files (.fic, .ndx and .mmo files).



Notes

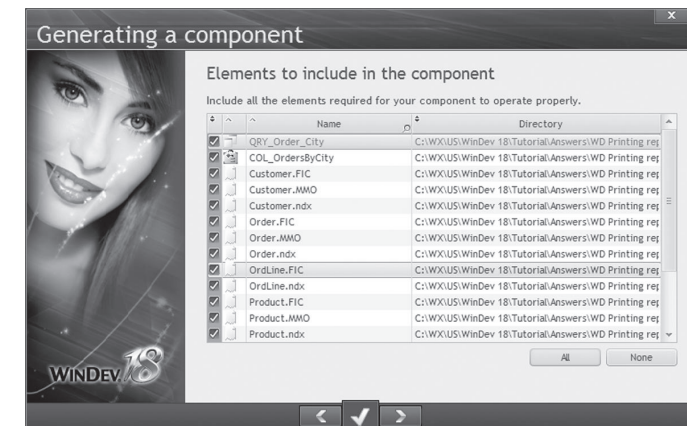
WinDev offers to automatically include in the component configuration the elements found in the list of the project's elements. To display the project's element list, on the "Project" pane, in the "Project" group, click .

In order for the data files to be automatically offered, these data files must be present in the list.

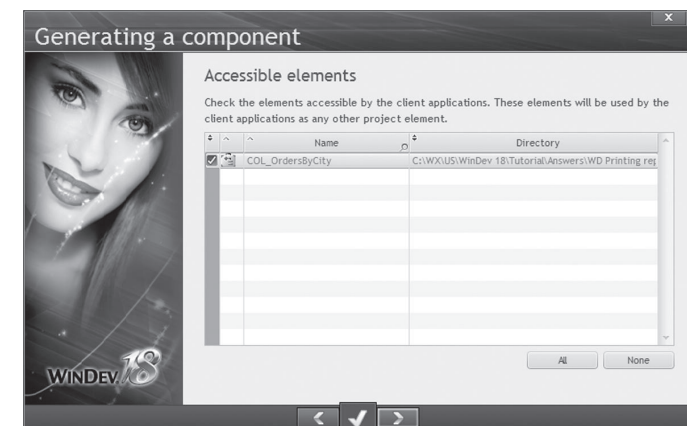
6. Go to the next plane and validate the creation of the configuration.

▶ Let's generate our component now.

1. On the "Project" pane, in the "Generation" group, click "Generate".
2. Go to the next wizard screen and specify the elements to include in the component. By default, WinDev offers the elements specified in the configuration and the analysis. Select all the elements ("All" button) and go to the next screen.

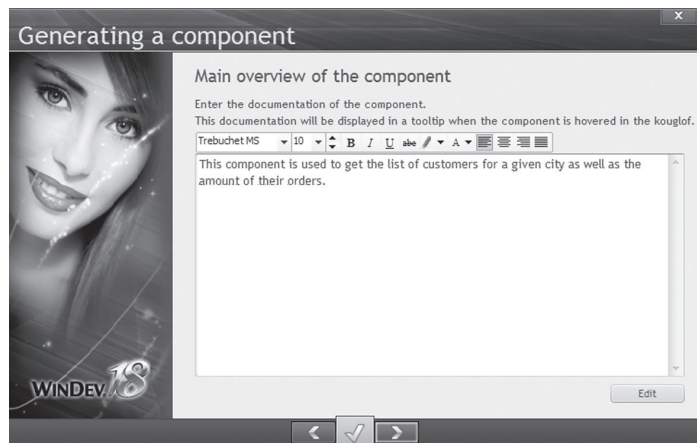


3. The wizard asks you to select the component elements that will be accessible from the client application. In our example, only the "COL\_OrdersByCity" set of procedures will be used:



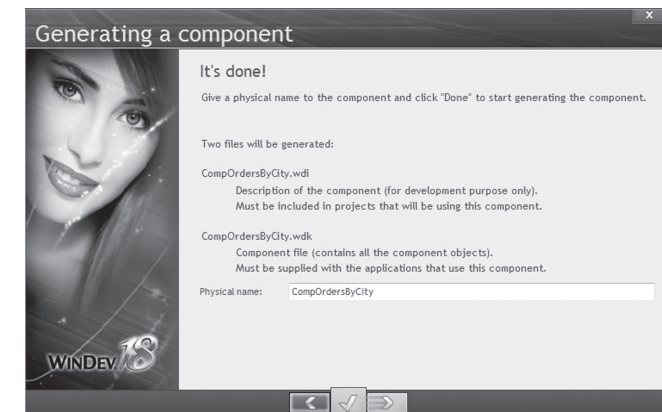


4. WinDev offers to authorize the translation of the component. This feature is interesting if your component must be used by multilingual applications.  
If "Allow the component to be translated" is checked, the specified component elements can be translated from the application that uses the component.  
This translation will be performed via WDMMSG, 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 screen.
5. Choose the languages of the component. Our example will contain the English language only. Go to the next screen.
6. The wizard proposes to manage the different versions of the component. In our example, the component was just created.  
Keep the default options. Go to the next screen.
7. Enter the information about the component:
  - Owner,
  - Caption, Copyright, ...
8. Go to the next screen.
9. An image can be associated with your component. Users will be able to easily identify the component in the "Catalog of examples" pane. Go to the next screen.
10. Specify the main overview of the component. This overview will allow the user to find out the purpose of the component. Enter for example:



11. Go to the next screen. The wizard will now automatically generate the documentation for the component. This documentation can be based on the comments inserted into your source code.
12. Go to the next screen. You have the ability to modify the generated documentation. Don't do anything. Go to the next screen.
13. 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's code.  
We won't generate any help file. Go to the next screen.

14. You can allow the end users of the component to enter their own WLanguage code in order to customize your processes. The suggestions can be directly sent to you. No input of "User Macro-Code" will be allowed in this example. Go to the next screen.
15. The component will not be saved in the GDS. Go to the next screen.
16. Now let's give a physical name to our component.



- Validate. Well done, you have just created your first ready-to-use component! The component is displayed in the "Wizards, Examples and Components" pane ("Components" button, "Custom components" item).

## Step 2: Using the external component

Once created, your component can be used in any other WinDev project. Let's see how this component can be re-used.

- To do so, we're going to create a new project and import our component into the project.
  1. Create a new project: click  among the quick access buttons. Click "Project" in the wheel that is displayed. This project is named "CompUse" and it has no analysis.
  2. On the "Project" pane, in the "Project" group, expand "Import" and select "An external component .. From a file".
  3. In the directory of the "WD Printing Reports" project, select the "EXE\CompOrdersByCity" sub-directory, then the "CompOrdersByCity.wdi" file. The "WD Printing Reports" project is available in the "Tutorial\Exercises\WD Printing reports" sub-directory.
  4. Click "Open", the description of our component is displayed. This description contains:
    - In the "General" tab, the elements entered when creating the external component as well as its location. You also have the ability to define the load mode of the component.
    - In the "Details" tab, the description of the component as well as the automatically generated help. This enables you to find out the re-usable elements of the component.



Notes

The description of the component can be accessed at any time. All you have to do is select the component in the project explorer and select "Description" from the popup menu.

5. Validate (green button). The component is included in your project.

► Now we're going to create blank window to use the component.

1. Create a blank window.

- Name the window "WIN\_CustomerByCity".
- Enter the "Customers by city" title.
- Save the window.

2. Add the following controls:

- An edit control whose caption is "City" and whose name is "EDT\_City".
- A table control "TABLE\_City" made of 2 columns:
  - a "Name" column, Text type.
  - a "Total sales" column, Numeric type.
- A button whose caption is "Search" and whose name is "BTN\_Search".

► Now edit the code of the "BTN\_Search" button. When clicking this button, we're going to run a search procedure found in the component. This procedure:

- expects the name of the city as parameter
- returns a string in the following format: Name of Customer 1 + TAB + Total sales 1 + RC + Name of customer 2 + TAB + Total sales 2 + ...

Therefore the code of the "BTN\_Search" button must:

- call the **GiveCityOrders** procedure of the component and pass it the content of EDT\_City as parameter.
- process the returned string to add it to the table.

Enter the following code:

```
S is string
S = CompOrdersByCity.GiveCityOrders(EDT_City)
FOR EACH STRING x OF S SEPARATED BY CR
 TableAdd(TABLE_City,x)
END
```

► Save your window and run its test (click the "GO" button). in the edit control, enter "Syracuse" then click "Search". The list of customers comes up.

► That's it! Easy, isn't it?

You now 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, to distribute them separately from your applications for instance.

## 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 work. 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 will be using 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 uses dependency files. This file contains the references to the different external files used in the component.

### Professional distribution

The distribution of components via a setup procedure consists in supplying a setup program to the users of the 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 the necessary tools (MDAC, ODBC driver for HyperFileSQL, ...).
- the automatic update of the data files used by the component (if necessary).
- the uninstall program of the component.

► Go back to the "WD Printing Reports" project. To create the setup program, on the "Project" pane, click "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 (keyword: "Component, Distributing a component") for more details.

## LESSON 9.7. THE DEBUGGER

### This lesson will teach you the following concepts...

- Overview
- Debugging a project element
- Debugging a project
- Advanced options



Estimated time: 20 min

### Overview

When developing the full application ("WD Application with data" project in part 2), several tests were run: tests on windows, tests on queries, tests on reports, ...

An error occurs and you don't know why? WinDev enables you to follow the execution of a program step by step: the debugger enables you to view and to validate all the code lines that are run.

During the execution of your application, the debugger enables you to find out the values of the variables used, to run some processes, to re-run code lines. You even have the ability to modify the code lines during the debug operation.

The debugger can be used during development (to find out the cause of a problem during the development step) or at run time: you have the ability to debug an executable started on the computer or on another computer, a component...

Interested in these features? Let's check these features directly.

### Debugging a window

#### Starting the debugger

To test a window, click the GO icon (or press [F9]).

Several methods can be used to debug a window.

- **1st method:** You want to start the debugger while running the test of your application, from a specific action: all you have to do is press [CTRL] + [PAUSE]
  - **2nd method:** You know which piece of code triggers the error in your window, you want to start the debugger from a specific code line: you can set a breakpoint or use the STOP statement.
- For example, the debugger will be started on one of the windows found in our application developed in part 2:
1. Open the "WD Application With Data" project. If the application was not developed, a corrected project is available. To open the corrected project, the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Application with data (Answer)".
  2. Open the "WIN\_PersonX" window.
  3. Display the code of the button for exact-match search (BTN\_ExactMatch).
  4. With the mouse, click the yellow column found before the first code line. A red dot is displayed: it's a breakpoint.

```
HReadSeekFirst(PERSON, PERSONID, COMBO_PERSON)
IF HFound(PERSON) THEN
 FileToScreen()
END
```

In test mode, the debugger will be enabled as soon as the lines containing a breakpoint are run.



Notes

The STOP keyword and the breakpoint have the same effect. Like the breakpoint, it is ignored in the executable.

5. Now that the breakpoint has been implemented, let's test our window (among the quick access buttons).
6. Select an element in the combo box.
7. Click the "Exact-match search" button.
8. The code editor is displayed in debug mode.

We are now going to present the information displayed in the debugger.

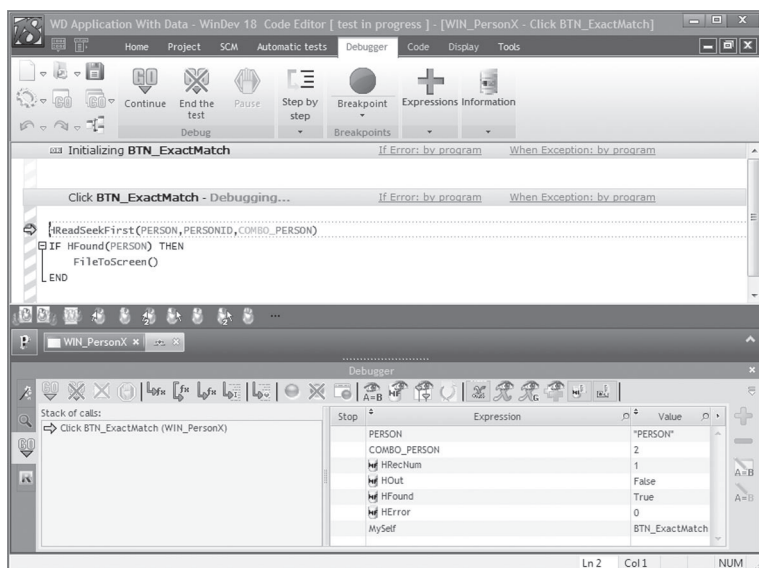
### Viewing the information in the debugger

The code editor is a little bit different in debug mode. A specific toolbar is displayed and the "Debugger" pane is automatically displayed.



The code editor will allow you to follow the execution of the code line by line.

The "Debugger" pane will allow you to follow the values taken by the different variables handled by the code.



Let's study the first debugging lines:

1. In the code editor, the yellow arrow indicates the line that will be run.
2. Press [F7]. The line is run. The yellow arrow moves to the next line.
3. Position the mouse cursor over the "END" line.
4. Press [F6]. All the lines up to the cursor are run.

We are now going to study the values of the variables used:

- The variables handled in the code are automatically viewed in the "Debugger" pane. Their value is automatically displayed.
- To find out the value of a variable found in the code, the variable must be hovered by the mouse cursor: the value is displayed in a tooltip.
- To add an expression into the "Debugger" pane, select the expression in the code editor and select "Add the expression to the debugger" from the popup menu of the code editor.

When the use of the debugger is no longer required, you can:

- press [F5] to continue the application without using the debugger.
- press [SHIFT + F5] to stop the application. In this case, the code editor will be re-displayed in standard mode on the code currently running.

The main topics of the debugger were presented here. Several other options are available, such as the auto-stop expressions, the code modification while debugging, ... See the online help (keyword: "Debugger") for more details.

### Trace window of the debugger

The WLanguage function named **Trace** can also be used to debug a project. This function is used to display the requested information in a "trace window". You can for instance display the value of a variable, a message to find out the way taken by the application, ...

When running a GO test (of the project, window, ...), the trace window is automatically displayed when **Trace** is reached (this window will also be displayed on the end-user computers). This window disappears when the test is over. However, the information displayed in the trace window can be retrieved via the "Trace of debugger" pane.

This pane displays all the information displayed during the test with the **Trace** function. A click performed on one of the lines found in the trace pane enables you to automatically access the corresponding code line.



Notes

An error occurs when running the test of your application? You didn't have time to write down the information displayed on the screen? Don't worry, this information was stored by the trace pane. A double click performed on the error message enables you to display the detailed information.

### Debugging a project

You also have the ability to debug the entire project. In this case, specific parameters can be defined: on the "Project" pane, in the "Test mode" group, expand "Test mode" and select "Configure the test mode".

## Advanced options of the debugger

---

The WinDev debugger is a powerful tool, used to debug several applications, in special cases. You have the ability to debug:

- a component, from a project that uses the component for instance.
- an executable (on the current computer or on a remote computer)
- an executable already started (on the current computer or on a remote computer)
- ...

## LESSON 9.8. THE PERFORMANCE PROFILER

### This lesson will teach you the following concepts...

---

- Overview of the performance profiler.
- Using the performance profiler.



Estimated time: 20 min



## Overview

The performance profiler enables you to check and optimize the execution time of your application. We recommend that you use the performance profiler to optimize your application (before it is distributed for example).

**Its principle is simple:** You test your application. During this test, the performance profiler keeps track of all the actions and processes run.

At the end of the test, the performance profiler displays:

- the 10 most time consuming processes
- all the actions performed in the application whose test was run, sorted by duration (from the longest one to the shortest one).

You have the ability to select a specific process in order to analyze the reasons for its duration.

## Using the performance profiler

The performance profiler can be started:

- **directly from the WinDev editor:** on the "Project" pane, in the "Audit and performances" group, click "Analyze the performance".

In this case, the project is automatically run in test mode. You can handle your application and start the processes of your choice.

To go back to the WinDev editor, simply exit your application.

The performance profiler displays the result of the analysis then. This result is saved as a WPF file.

- **from one of your processes in WLanguage,** via the following functions:

|                      |                                                        |
|----------------------|--------------------------------------------------------|
| <b>ProfilerStart</b> | Starts "collecting data" for the performance profiler. |
| <b>ProfilerEnd</b>   | Stops "collecting data" for the performance profiler.  |

In this case, only the code found between **ProfilerStart** and **ProfilerEnd** is analyzed. The corresponding WPF file can be opened in the performance profiler: on the "Project" pane, in the "Audit and performance" group, expand "Analyze the performance" and select "Open a performance report".

By default, the performance profiler saves the statistics performed on the code of the application in a <Project Name>.WPF file.

To change this name, click the "Save as..." button in the performance profiler.

**To open a specific statistics file** (a file created by programming for instance):

1. On the "Project" pane, in the "Audit and performances" group, expand "Analyze the performances" and select "Open a performance report".
2. Specify the path and the name of the statistical file.

The list of the last statistical files opened can be found in the menu opened by the "Analyze the performances" option.

## Optimizing a process with the performance profiler

### Reading the result of the performance profiler

The performance profiler presents the result of the analysis in two tabs:

- the "Synthesis" tab presents the 10 longest processes.
- the "Mapping" tab presents a graphical view of the main processes.
- the "Details" tab presents all the processes run during the test of the application (from the slowest one to the fastest one).

The following information is displayed for each process:

| Function                                  | Total time | Internal time | Nb of calls | Time 1 call | code % | Parent         |
|-------------------------------------------|------------|---------------|-------------|-------------|--------|----------------|
| From ProfilerStart() to ProfilerEnd()     | 5 s 300 ms | 1 ms          | 1           | 5 s 300 ms  | 0.0 %  |                |
| Total for "WIN_MENU" Window               | 5 s 299 ms | 2 s 662 ms    | 1           | 5 s 299 ms  | 50.1 % |                |
| Selecting the menu of MENU_MENUFILE_Bank. | 2 s 598 ms | 289 μs        | 1           | 2 s 598 ms  | 0.0 %  | WIN_MENU_MEN   |
| Total for "WIN_Table" Window              | 2 s 598 ms | 289 μs        | 1           | 2 s 598 ms  | 0.0 %  | WIN_MENU_MEN   |
| Click BTN_Print                           | 1 s 183 ms | 62 ms         | 1           | 1 s 183 ms  | 0.0 %  | WIN_Table_Banl |
| PrintReport()                             | 194 ms     | 194 ms        | 1           | 194 ms      | 0.1 %  | WIN_MENU_MEN   |
| Open("WIN_Table_Bank")                    | 145 ms     | 960 μs        | 1           | 145 ms      | 0.1 %  | WIN_Table_Banl |
| Click BTN_Modify                          | 144 ms     | 281 μs        | 1           | 144 ms      | 0.2 %  |                |
| Total for "WIN_Form_Bank" Window          | 111 ms     | 108 ms        | 1           | 111 ms      | 0.0 %  | WIN_Table_Banl |
| Open("WIN_Form_Bank")                     | 78 ms      | 78 ms         | 1           | 78 ms       | 0.0 %  | WIN_Table_Banl |
| Click BTN_CLOSE                           |            |               |             |             |        |                |

The elements displayed are as follows:

1. Function: Function, process or procedure run.
2. Total Time: Execution time of the function.
3. Internal time: Execution time due to the engine.
4. Nb calls: Number of calls made to the function (procedure or process)
5. Time 1 call: Time for running a call to the function (procedure or process)
6. % Code: Percentage of time spent processing the function or the procedure (developer code that could be optimized)
7. Parent: Element that contains the process.

### Choosing a process to optimize

The process to optimize is chosen according to several criteria:

- the execution time of the process. The longest processes must be optimized.
- the percentage of time spent processing the function or procedure. The higher this percentage is, the greater the number of processes that can be optimized in the code.

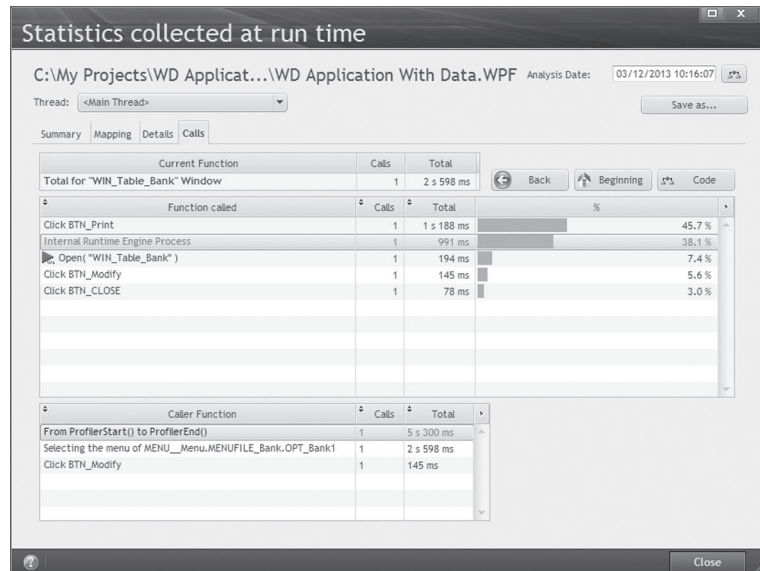
### Optimizing a process

Once the process to optimize is chosen, the performance profiler enables you to find out the details of the operations performed.

To display the details of the operations performed in a process:

1. Select the function to optimize in the "Details" tab.
2. Display the details of this function:
  - double-click the function.
  - click the "Calls" button.

The following tab is displayed:



This tab is divided into three sections:

- The selected function (with the number of calls to this function and the total processing time).
- The list of functions called by the selected function.
- The list of functions that call the selected process.

The list of functions called enables you to improve the search for the process to optimize. Indeed, you have the ability to view the different functions called as well as the processing time for each function.

All the processes that contain a WLanguage code (named "Developer code") can be optimized.

A double click performed on one of the functions found in this list ("Click Validate" for instance) enables you to view the details of the processes called by this function.

#### Notes:

- The "Internal runtime engine process" caption corresponds to the execution time of the function or procedure (for a WLanguage function for example). This time cannot be reduced and it cannot be optimized.
- The "Developer code" caption corresponds to the execution time of the code for the same function or procedure (excluding calls to sub-functions). This time can be reduced and it can be optimized.
- To quickly view the code of the current process, click the "Code" button. The profiler remains opened and the current code can be modified.
- The "Previous" button is used to go back to the calling function.
- The "Start" button is used to go back to the beginning process of the application.

## LESSON 9.9. IMPORT/EXPORT

### This lesson will teach you the following concepts...

- Importing elements from a project to another one.
- Exporting the elements of your project.
- Specific import operations (WebDev project, Access, Visual Basic, non-WinDev windows, ...).



Estimated time: 10 min

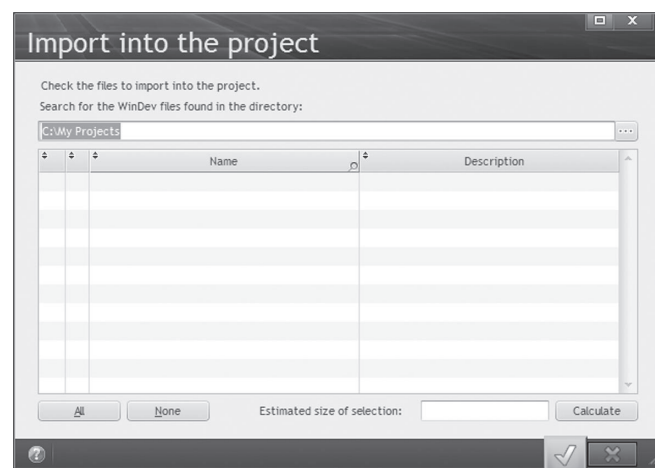
## Importing elements

You have the ability to import the WinDev elements found in your project. All types of WinDev elements can be imported:

- windows, reports,
- classes, components,
- procedures, ...

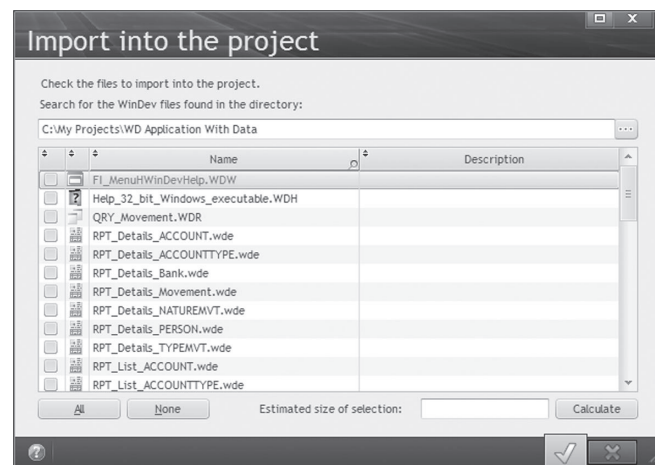
► To import existing elements into the current project:

1. On the "Project" pane, in the "Project" group, expand "Import" and select "WinDev elements and their dependencies...".



2. Click "..." and select the directory containing the elements to import (the directory must contain WinDev elements).

3. Validate. WinDev returns the list of the elements that can be imported and that were found in the selected directory (the sub-directories are ignored).



4. Select the elements to import and validate.

The elements (and all the files used by these elements: images, ...) are now part of the project.



Notes

The "Calculate" button in the import window is used to calculate the size of the selected elements and dependencies.

## Exporting elements

You also have the ability to export elements from your project to another directory for example. These elements can be re-used in other projects.



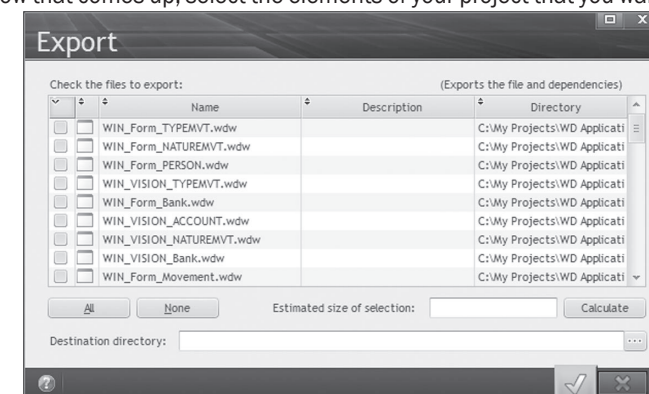
Caution!

Exporting elements is a lot more than a simple copy of elements. It is actually the only safe method for transferring a project element along with all its dependencies (images, icons, ...). A practical solution for transmitting windows along with all their dependencies by email for example.

► To export elements from your project:

1. On the "Home" pane, in the "General" group, expand "Save" and select "Export .. To a directory..".

In the window that comes up, select the elements of your project that you want to export.



2. Specify the name of the destination directory (or select it with the "..." button).

3. Validate.

The elements are exported to the specified directory. These elements are still available in your project.



Notes

The "Calculate" button in the import window is used to calculate the size of the selected elements and dependencies.

## Specific import operations

### Importing a WebDev project

WinDev enables you to import a page or a full WebDev project into a WinDev application.

During this import:

- The pages are changed into windows.
- The "Server" code and the "Browser" code are grouped together.
- The elements with no equivalent in WinDev (JavaScript code for example) are imported as comments or as separate elements.

#### To import a WebDev project into a WinDev project:

1. On the "Project" pane, in the "Project" group, expand "Import" and select "From WebDev .. A WebDev project". The wizard for importing a WebDev project starts.
2. Select the WebDev project to import. This project will not be modified and a new WinDev project will be created.
3. Specify the name and location of the WinDev project to create.
4. If an analysis is linked to the project, specify its name, its location, and whether it must be used by the WinDev project.
5. Specify the elements that will be shared between the WinDev project and the WebDev project (common elements such as reports, classes, ...). If the elements are shared, they will not be copied into the WinDev project.
6. Validate. The WebDev project is converted into a WinDev project.

#### To import WebDev elements into a WinDev project:

1. Open the WinDev project into which the element must be imported.
2. On the "project" pane, in the "Project" group, expand "Import" and select "From WebDev .. WebDev elements". The list of elements that can be imported is displayed.
3. Select the elements to import. Go to the next screen.
4. Specify the elements that will be shared among the WinDev project and the WebDev project. These elements will not be copied to the WinDev project.
5. Validate. The specified elements are automatically imported into the current WinDev project.

### Importing a window


You are interested by a window found in a non-WinDev application? You want to retrieve its interface? Nothing's easier.

WinDev lets you import an open window into the current environment.

The window is imported into the current project. Each window element is automatically changed into a WinDev element (static, edit control, button, and so on...).

Note: All the elements found in the window are imported but you may have to modify the window in order to get the requested interface.

#### To import a window into a WinDev project:

1. Open the WinDev project into which the window must be imported.
2. Click  among the quick access buttons. Click "Window" in the wheel that is displayed. The window creation wizard starts.
3. Display the "Import" tab. The windows currently displayed on the current computer are automatically listed. These windows are identified by their title (displayed in the title bar). **Note:** The "Generate images for the unrecognized controls" option is used to make the final interface closer to the interface of the window to import.
4. Validate. The preview of the window to import is displayed.
5. Modify if necessary the type of control created when importing the window ("Details" button).
6. Validate. The window is automatically included in the current WinDev project.

#### Limits: Are not imported:

- the code of the elements.
- the controls other than the following controls:
  - standard control of Windows.
  - Delphi controls.
  - Visual Basic controls.
- the images and the background images (except if "Generate images for the unrecognized controls" is checked).
- the color of the elements.

### Access import

WinDev enables you to import Access elements: analysis, form, query or full project.

To import elements from an Access application:

1. On the "project" pane, in the "Project" group, expand "Import" and select "Access project or elements".
2. Select the ".mdb" file corresponding to the Access project.
3. Select the elements to import. Specify (if necessary) whether the database must be migrated to HyperFileSQL format or kept in Access format.
4. Validate.

Caution: Importing elements from an Access application may take quite a long time.

### Visual Basic import

WinDev enables you to import Visual Basic projects and Visual Basic forms.

To import elements from a Visual Basic application:

1. On the "project" pane, in the "Project" group, expand "Import" and select "Visual Basic project or elements".
2. Select the files corresponding to the elements to import.
3. Validate.

Caution: Importing elements from a Visual Basic application may take quite a long time.

## LESSON 9.10. INSTALLING AN APPLICATION

### This lesson will teach you the following concepts...

- Setup editor
- Network update, Web update
- Questions/Answers



Estimated time: 20 min

### Overview

We have already deployed a simple application.

This lesson presents the setup of an application by focusing on:

- the setups with automatic update.
- the setup editor, used to create custom setups.

### Network update/Web update

Several setup modes are available for a WinDev application:

- **Stand-alone setup:**

This type of setup lets you create a single installation 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 type of setup was used with the "WD Application with data" application in part 2 of this tutorial.

- **Setup with automatic update:**

This type of setup enables you to automatically detect updates when starting the application. If an update is available, the user can immediately perform this update.

This type of setup is available both via a network or via the Web.

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 setup program creation wizard ("Setup procedure" option on the "Project" pane).

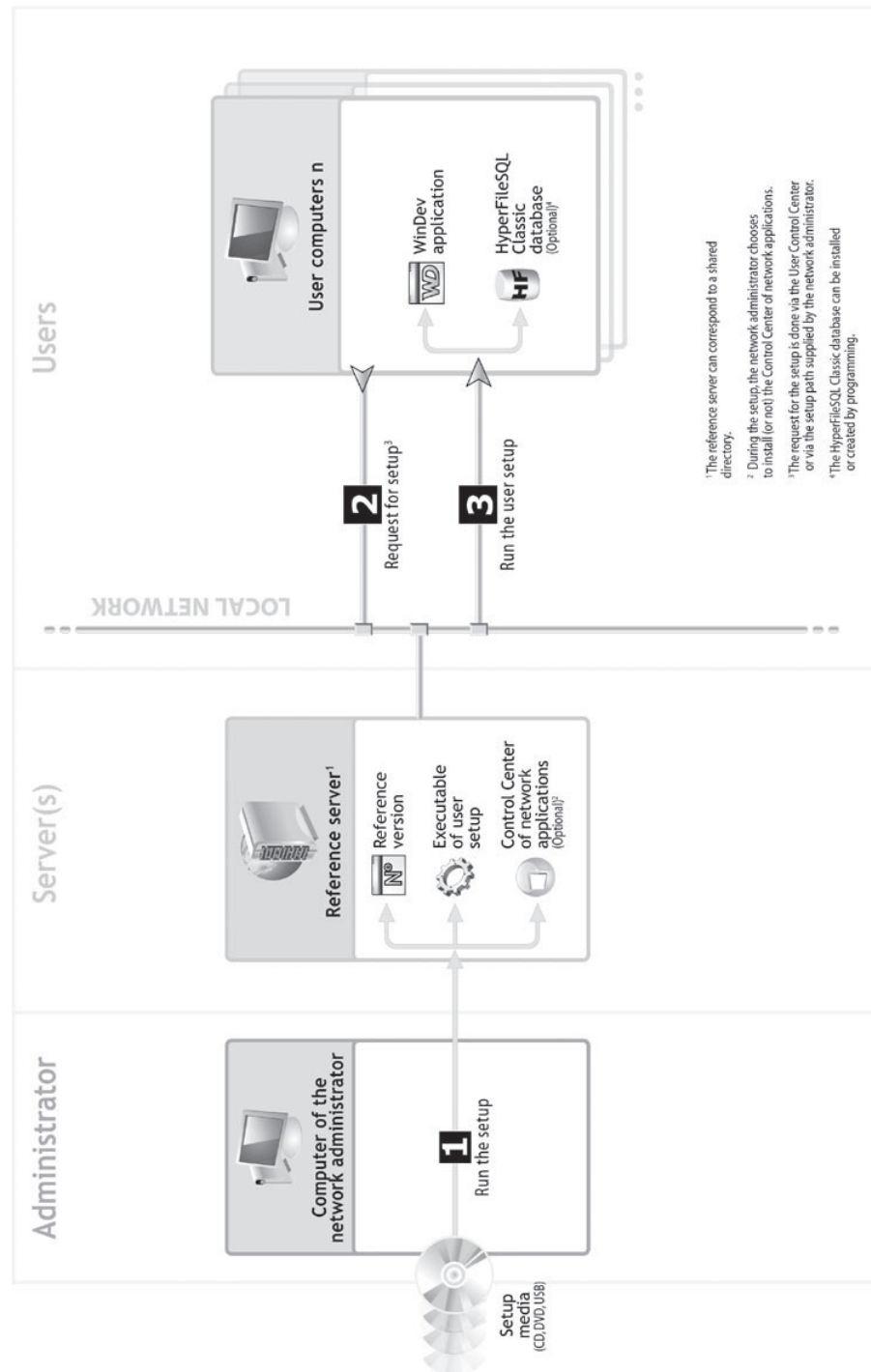
The setup program obtained at the end of the wizard is made of two setups:

- 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 setup of the application itself. This setup is included in the reference application. It can be accessed only when the reference application is installed.

The diagram below presents the setup mode of an application that uses a local HyperFileSQL Classic database, with a network update.

The version of the 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 instance), an update is automatically proposed for the final application.





### Setup with Internet update

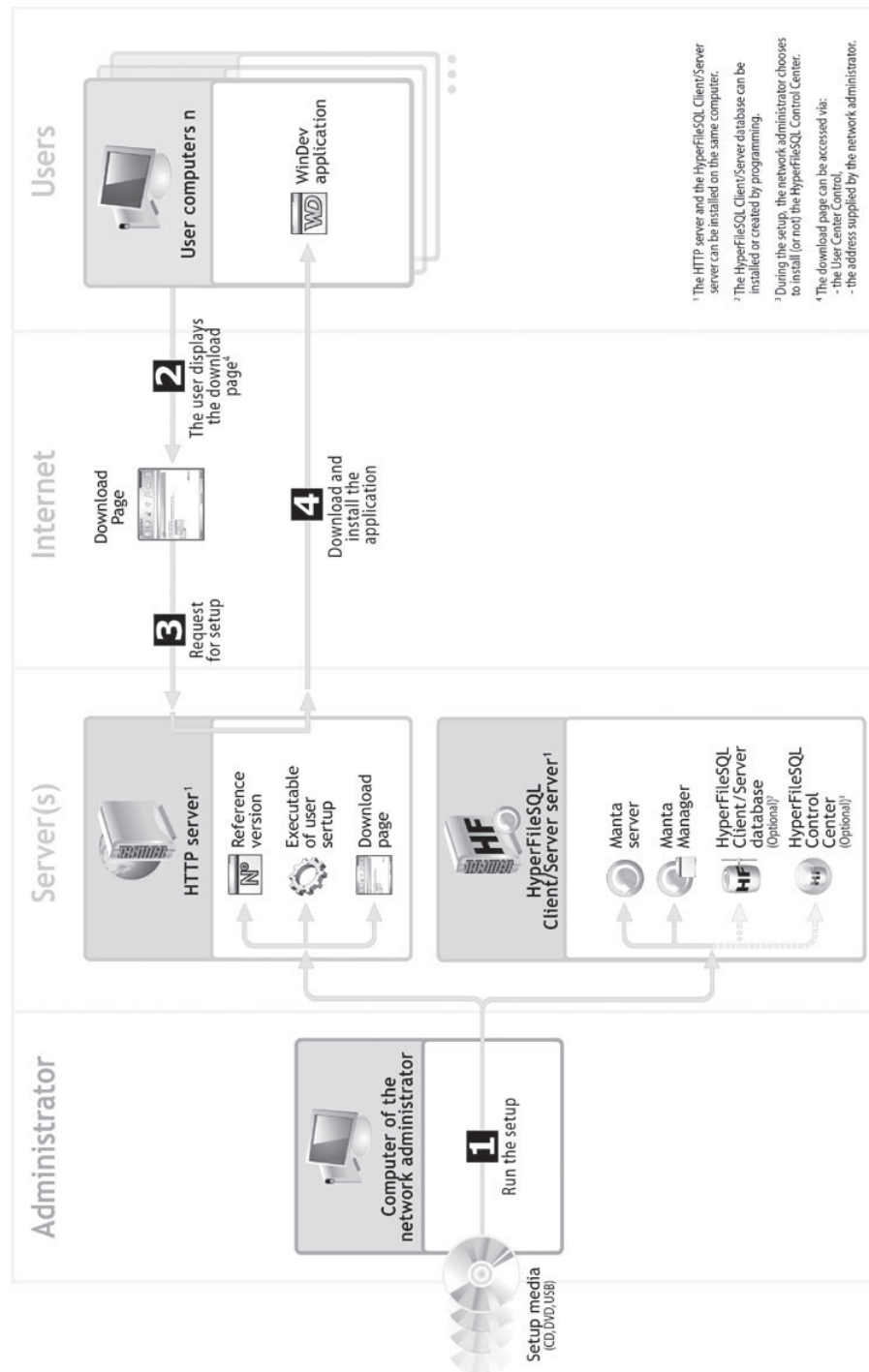
The same principle is used by the setup with Internet update.

The creation of the setup program is performed via the setup program creation wizard ("Setup procedure" option on 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 setup of the application itself. This setup is included in the reference application. It can be accessed only when the reference application is installed, via the Web page for download.

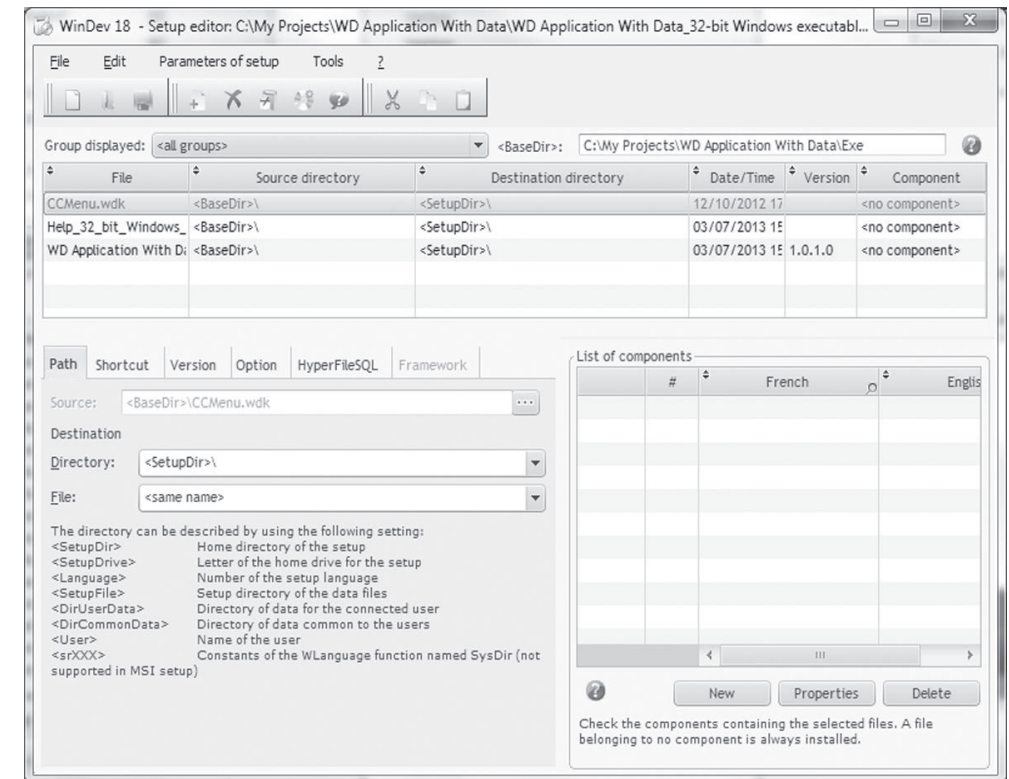
The diagram below presents the setup mode of an application that uses a HyperFileSQL Client/Server database, with an update by Internet.



## Setup editor

Setup programs were already created by the setup wizard. WinDev also proposes a setup editor, allowing you to configure your setup programs.

To start the setup editor, on the "Tools" pane, in the "Utilities" group, click "WDInst". The following screen is displayed:



## Questions/Answers

---

### Question How do I uninstall an application developed with WinDev?

When creating the setup program, you can allow the users to uninstall the application. The uninstall program is automatically created if this option is chosen. Your application was registered toward Windows so that it can be uninstalled later.

To uninstall an application:

1. Click the "Start" menu.
2. Select "Control panel".
3. Select "Add/Remove programs".
4. Select the application and click "Uninstall".

### Question How do I create a setup via CD-ROM?

When creating the setup program ("Setup procedure" option on the "Project" pane), you have the ability to choose various setup delivery methods. If you select "CD-ROM", WinDev will create a folder containing all the files required for a setup via CR-ROM (AUTORUN file, setup files, ...). Then, all you have to do is burn the content of this folder on a blank CD and distribute it!

### Question How do I install an application?

Once your executable has been created, the EXE directory located in your project's directory contains all the elements needed for your application to run.

To prepare a setup for your application:

1. On the "Project" pane, click "Setup procedure". The setup creation wizard starts.
2. Follow the instructions given on the screen.

## LESSON 9.11. UML AND 3-TIER

### This lesson will teach you the following concepts...

---

- What is the UML?
- The types of diagrams managed by WinDev.



Estimated time: 10 min

## What is the UML?

UML (Unified Modeling Language) is a concept for "modeling" processes.

## The diagrams managed by WinDev

### Class diagram

The class diagram is used for modeling the structure of a system and relationships between the different elements found in this system. It is mainly used in OOP.

With the diagram of an UML class, you can automatically generate the classes and the methods that derive from it.

Via "reverse engineering", you can then display the relationships between the different classes that were created.

### Use case diagram

Once the specifications have been defined, the use case diagram is used to establish in a simple and visual way the behavior of your project (correspondence between the implementations of user requests by the developers, ...). This type of diagram is sometimes called a "sequence diagram" (without specific chronology).

No application can be generated from this type of diagram. This type of diagram is only used when implementing the project management.

### Object diagram

An object diagram presents a set of objects and their relationships at a given time.

An object diagram is used to show a context (before or after an interaction between objects for example).

### Component diagram

A component diagram describes the physical and static architecture of a computer application. For example: source files, libraries, executables, ...

### Activity diagram

An activity diagram represents the behavior of a method or the flow of a use case.

### Sequence diagram

A sequence diagram represents the chronological order of the messages sent and received by a set of objects.

### Collaboration diagram

A collaboration diagram presents the structural organization of the objects that send and receive messages.

## State-transition diagram

A state-transition diagram presents a sequence of states that an object goes through during its lifecycle. It is used to describe the changes of states for an object or for a component.

A state is defined by its duration and by its stability.

A transition represents the instantaneous change from one state to another one.

A transition is triggered:

- by an event.
- automatically when no triggering event is specified.

## Deployment diagram

A deployment diagram presents the physical layout of the hardware devices (nodes) used in a system as well as the association between the executable programs and these devices.

We won't go into details about the use of the UML language with WinDev. See the online help (keyword: "UML") for more details.

## 3-tier

### What is the 3-tier?

The development in "3-tier" architecture is now simplified in WinDev.

The 3-tier architecture is designed to separate the 3 "tiers" of an application: GUI, processes and data.

An application will include 3 independent layers:

- a presentation tier
- a process tier
- a data access tier.

Specific functions (APIs in standard languages, advanced WLanguage functions) allow these 3 tiers to communicate between themselves.

The reason for separating the layers is to facilitate maintenance and future upgrades of the application (change of database system, transfer from a graphic environment to another one...).

This provides better security because the access to the database is allowed via the process tier only.

It also optimizes the teamwork and the multi-target development.

### How do I implement 3-tier?

The dialog between the different layers is performed via classes or structures that are automatically generated by WinDev.


#### 1. Choose the data sources (files or queries that will be used in 3-tier).

In the data model editor and in the query editor, you create the different files and queries.

**2. WinDev automatically builds the class diagram that corresponds to the classes and structures that will be generated.**

To do so, in the data model editor, on the "Analysis" pane, in the "Analysis" group, expand "Generation" and select the "Generate the UML diagram corresponding to the analysis" option.

You also have the ability to create, in the UML editor, a class diagram corresponding to the analysis:

- Create the UML model: Click  among the quick access buttons. In the wheel that is displayed, hover "Architecture" and click "UML".
- In the wizard that comes up, select the "Class diagram" option. Go to the next screen using the arrows.
- Select the "Build the class diagram corresponding to the analysis and queries" option.
- Choose the type of code to generate (procedural or object-oriented).
- Create then save your diagram.

**3. The classes and structures are generated in WLanguage.**

On the "Project" pane, in the "Other actions" group, expand "UML Modeling" and select "Generate the code". The sets of procedures and/or the classes are automatically generated.

**4. It can be "improved" by developing your own methods in these classes.**

The classes and the diagram are synchronized in real-time.

You can also modify the analysis and synchronize the UML diagram with the analysis.

**5. These classes or structures allow the "Process" layer and the "Presentation" layer to communicate between themselves.**

### Handling the different layers

- For the "Data access" layer:  
The standard data access functions can be used: read, write, queries, transactions, ...
- For the "Process" layer:  
Specific WLanguage functions can be used: *FileToMemory* and *MemoryToFile*. These functions are used to fill the class members from the items found in the corresponding files (and conversely).
- For the "Presentation" layer:  
The generated classes can be handled directly, no need to worry about the structure of the database.
- For the communication between layers:  
Depending on the communication protocol between the two layers (components, Web Services, HTTP, WEBDEV site,...), the classes can be serialized using XML or binary format, (using *Serialize* and *Deserialize*).

## LESSON 9.12. FLEXIBLE MODELING

### This lesson will teach you the following concepts...

- Principle
- Operation



Estimated time: 20 min




## Principle

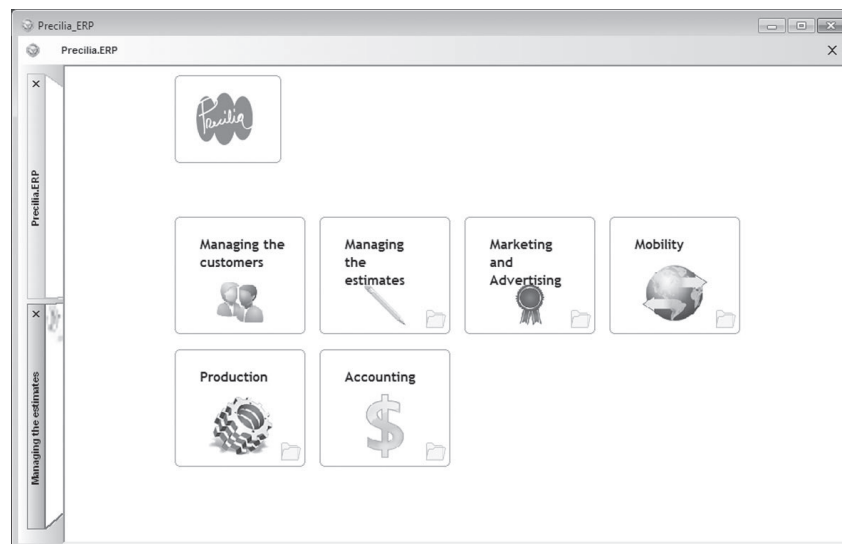
The modeling editor is used to represent all the organizations (existing ones or forthcoming ones). The modeling editor enables you to represent the organization of email management in your company, the organization of contacts with the customers, ...

A modeling example was created to help you discover the features of the modeling editor. This modeling represents part of the ERP (Enterprise Resource Planning) in a fictitious company called Precilia.

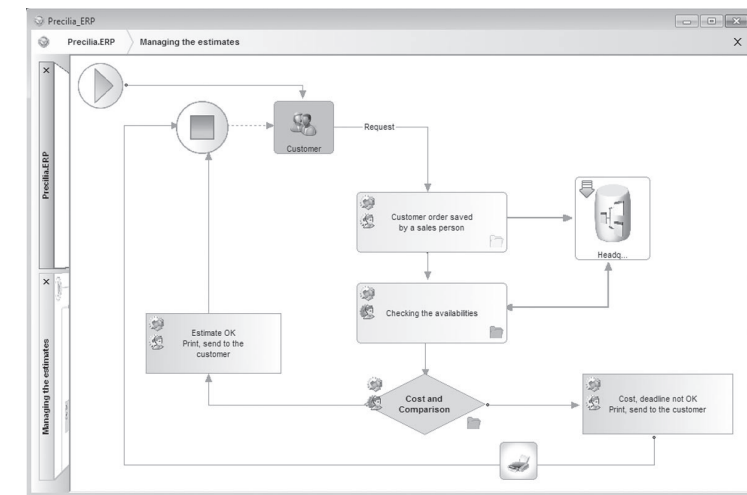
## Operations

- ▶ Close the current project if necessary. The home window is opened.
- ▶ In the home window, click "Tutorial" and select the project named "Flexible modeling". The project is loaded.  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Flexible modeling".
- ▶ Open the "Precilia\_ERP.wem" file (double-click the model in the project explorer). Several major fields are presented in this modeling: customer management, estimate management, marketing, ...

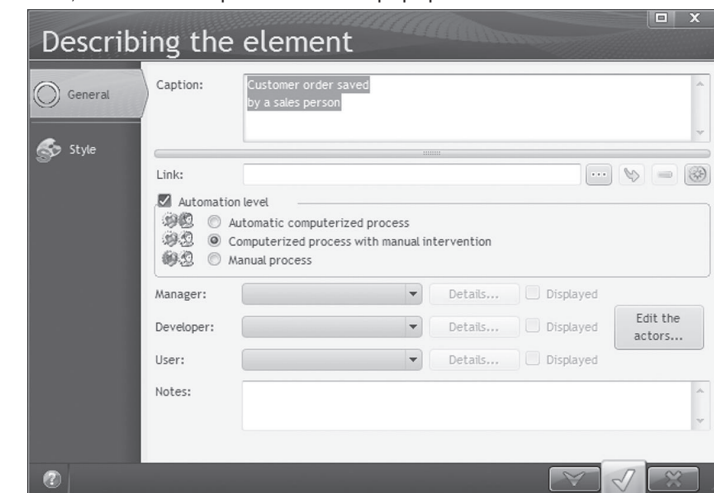
The icon  indicates that a sub-modeling is found. Double-click this icon if you want to access this sub-modeling.



- ▶ In our example, open the sub-modeling corresponding to the management of estimates. You will discover the different steps for managing the estimates:



- ▶ Let's take a closer look at the "Headquarter" element. An arrow is found in the top left corner of this element. This arrow indicates a link. The application analysis is automatically opened when you double-click the arrow. This is a main feature of the modeling editor: any element found in the modeling can be linked to an element of your project (window, code, ...)
- ▶ Close the data model editor.
- ▶ All the elements found in the modeling have their own characteristics. To display these characteristics, select "Description" from the popup menu of the element. For example:



This window is used to specify:

- the link between the element and an external file (analysis, window, ...)
- the automation level of the element: process performed automatically, manually, ...
- the actors taking part in this process (managers, users, developers, ...). This information may come from the contributors described in the Project Monitoring Center.

See the online help for more details.

## LESSON 9.13. MULTILINGUAL

### **This lesson will teach you the following concepts...**

---

- What is a multilingual application?
- Creating a multilingual application step by step.



Estimated time: 20 min

## 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.

The same application can therefore be used in several languages. How is it possible?

That's what we shall see in this lesson.

We are going to handle a project that can be run in English or in French, depending on the user's choice.

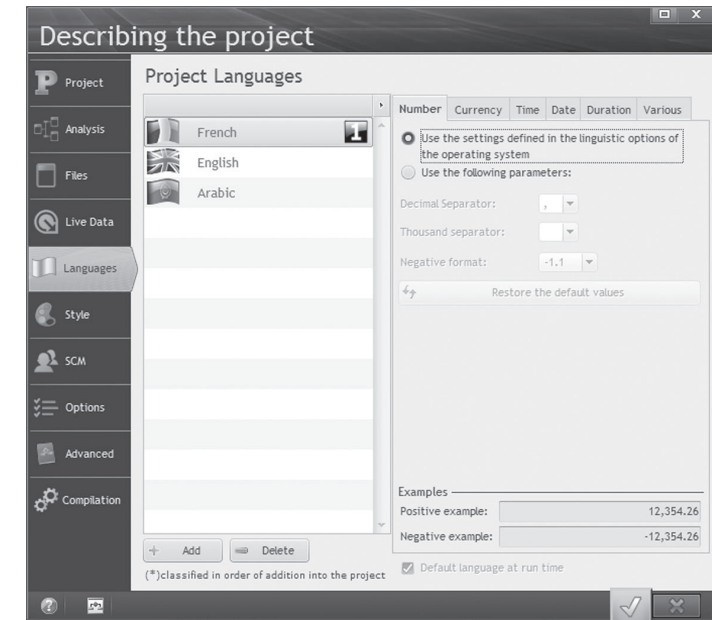
The main steps for a multilingual application are:

- Choosing the project languages.
- Localizing the analysis.
- Localizing the project elements (windows, reports, controls, help system, ...).
- Localizing the messages found in the source code.
- Localizing the WLanguage functions.
- Programming the change of language in the application.

We're going to apply these steps to the "WD Multilingual" project. This project, available in French, will be translated in English and in Arabic (the text direction will be reversed).

## Choosing the project languages

- Close the current project if necessary. The home window is opened.
- In the home window, click "Tutorial" and select the project named "Multilingual application". The project is loaded.  
Tip: if the home window is not displayed, on the "Home" pane, in the "Online help" group, expand "Tutorial" and select "Multilingual application".  
The associated analysis is "WD Multilingual". This information can be verified in the "Analysis" tab of the project description.
- The first operation consists in choosing the project languages.
  1. Display the project description if not already done: on the "Project" pane, in the "Project" group, click "Description".
  2. Click the "Languages" tab. Our application will support French, English and Arabic: select these three languages.
- This tab is also used to configure the linguistic options regarding the numbers, the currencies, the dates, ... for the selected language. Let's see an example:
  - Click the "English" language.
  - Click "Date".
  - The linguistic options of Windows are used by default. Select "Use the following parameters": you now have the ability to define the date format used as well as the translation for the days and for the months.
  - Keep "Use the parameters defined in the linguistic options of Windows".



- Validate.



Notes


In the linguistic options, you have the ability to choose the text direction of the language ("Various" box, "Text direction" option). This enables you to create interfaces with a language written from right to left.

## Localizing the analysis

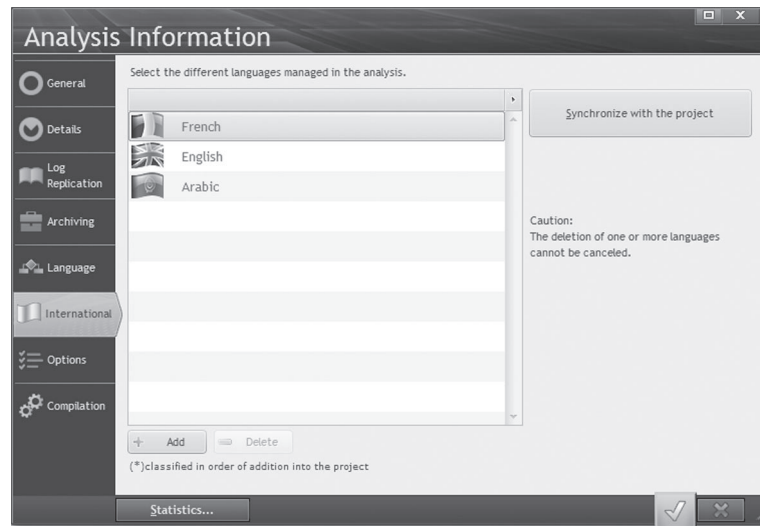
By default, an analysis is created in a language and it cannot be translated. However, some information can be entered in several languages (notes in the documentation, shared information, ...). If your application uses Reports and Queries, the file names and the item names can also be translated. This translation can be performed in the "Reports and Queries" tab (in the description window of the files or items).

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 the item, this caption will be used when creating the control.

To support several languages in an analysis:

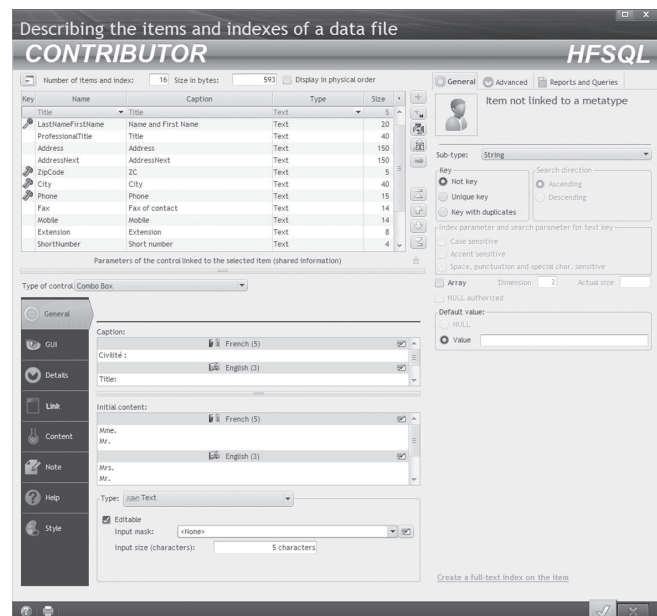
1. Display the data model editor: click  among the quick access buttons of the WinDev menu.
2. In the analysis description ("Analysis description" option of the popup menu), select the "International" tab.

3. Specify the languages supported by the analysis and validate.



To enter the shared information in several languages:

1. Display the description of the items found in a data file (select the "Item description" option from the popup menu).
2. For each item, display the parameters of the control linked to the selected item (shared information). To do so, click the double arrow at the bottom of the screen.

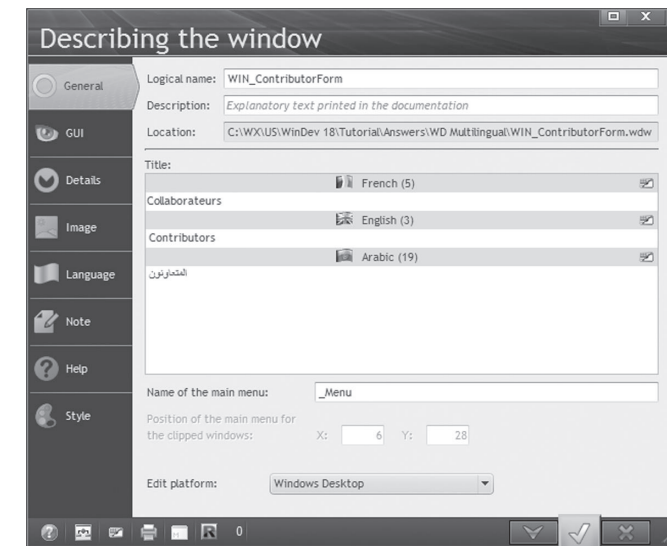


## Localizing the project elements

All the project elements can become multilingual elements: windows, reports, help system, ... Let's take a look at the characteristics of a window (and window controls).

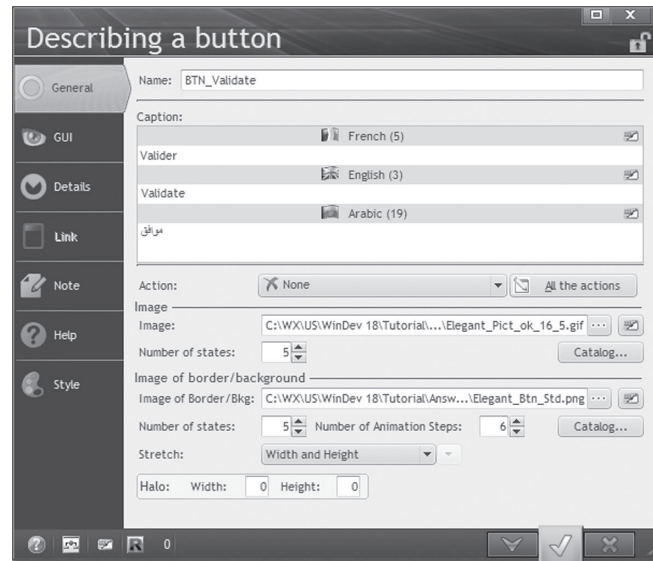
### Characteristics to translate

- Open the "WIN\_ContributorForm.WDW" window.
- Display the window description ("Description" from the popup menu of the window). Select the "Languages" tab: the three languages selected in the project are displayed. Select the "General" tab: the title of the window must be translated.




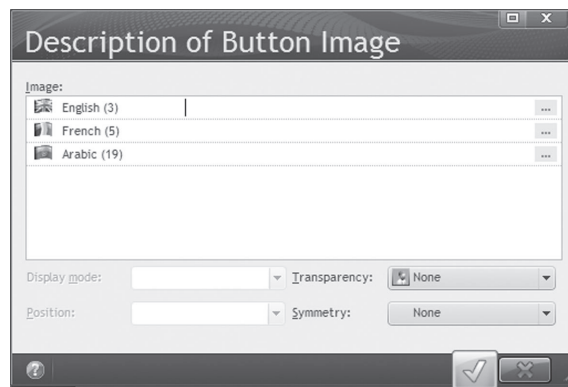
Close this window.

- Display the description of the "Validate" button.  
Let's take a look at the different areas regarding the management of the different languages



In the **"Caption"** area, the three languages are available. Scroll the "Caption" area. If the project was not yet translated, the content of the "French" language is copied into the other languages.

In the **"Image"** area,  is used to configure the multilingual management. Click this button.



In the window that opens, a different image can be used for each language. This feature is very useful if you are using images containing text. Close this window.  
Click the **Help** tab. This tab contains the different help messages associated with the control. This information must also be translated.

The same type of information must also be translated for:

- all the controls found in the windows
- the windows
- the reports,
- the controls found in the reports
- the text messages found in the code editor.

### How do I translate this information?

Several methods can be used to translate this information:

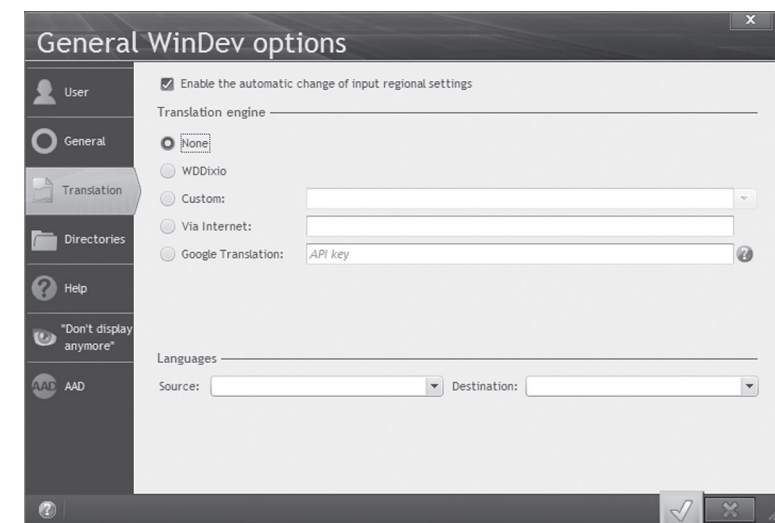
- a translation performed in the different editors.
- a translation performed via an external tool (WDMMSG and WDTRAD);

### Direct input of the translations

The translations are entered in the interface of the product. For example the translation of the "Validate" button, becomes "Valider" in French. All you have to do is open the description window of the control and enter 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 the translation.
  - 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 the project elements: this button will allow you to use the software defined for the translation.

## Special cases

### Direct translation of the menus

The translation of the menu options can be performed from the window editor. In our example, the "WIN\_ContributorForm" window displays a menu.

- To translate the menu of the "WIN\_ContributorForm" window:
  1. On the "Display" pane, expand "Language displayed" and choose the language to view in the editor.  
The menu options are displayed in the selected language. If no translation corresponds to the selected language, the menu options are displayed in French.
  2. Enter the caption of the menu options in the selected language.

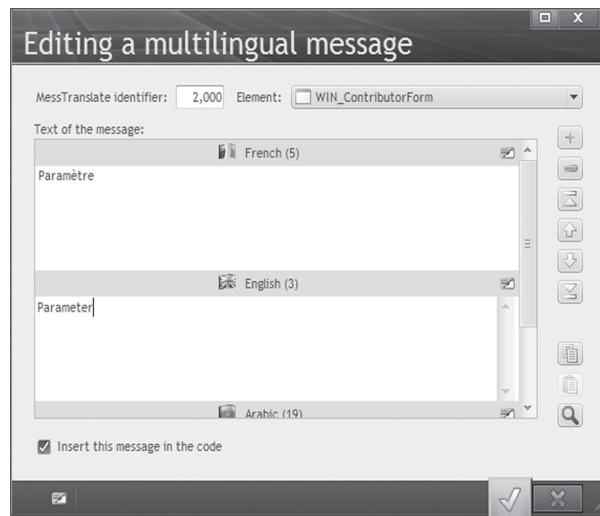
### Direct translation of the programming messages


All the messages found in your program can also be entered in several languages. Two types of messages can be found:

- the simple character strings, for example:

```
// Store the type of the chart in letters (BarCharts, candlestick, and so on)
INIWrite("Parameters", "NameUS", GraphType[GraphType], gsINIFile)
```

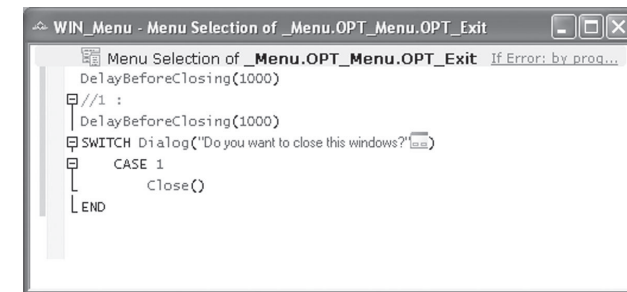
To translate this message, on the "Code" pane, in the "Languages" group, expand "Translate strings" and select "Translate messages". The following window is displayed:



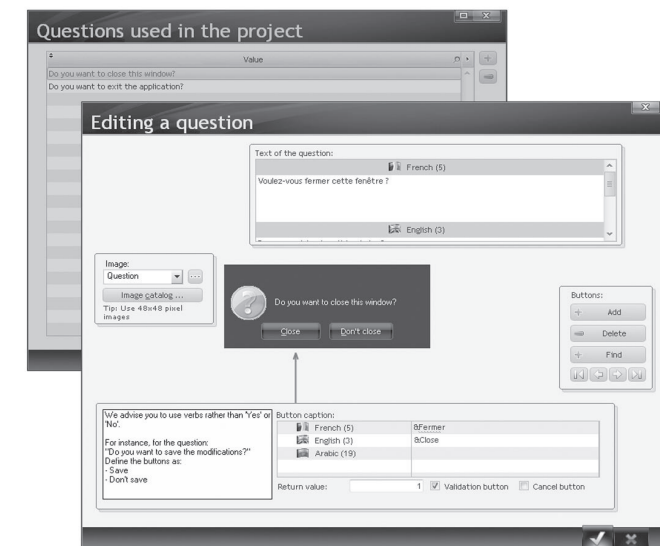
This window lets you enter all the messages from your program in all the languages supported by the project. A number is assigned to each message. When a translation is entered for the message, the icon  is displayed in the code editor.

```
// Store the type of the chart in letters (BarCharts, candlestick, and so on)
INIWrite("Parameters", "NameUS", GraphType[GraphType], gsINIFile)
```

- the directive questioning, for example:



To translate this message, on the "Code" pane, in the "Languages" group, expand "Translate the messages" and select "Directive questioning". The list of directive questioning is displayed. All you have to do is double-click the sentence to translate.



### Direct translation of the help files

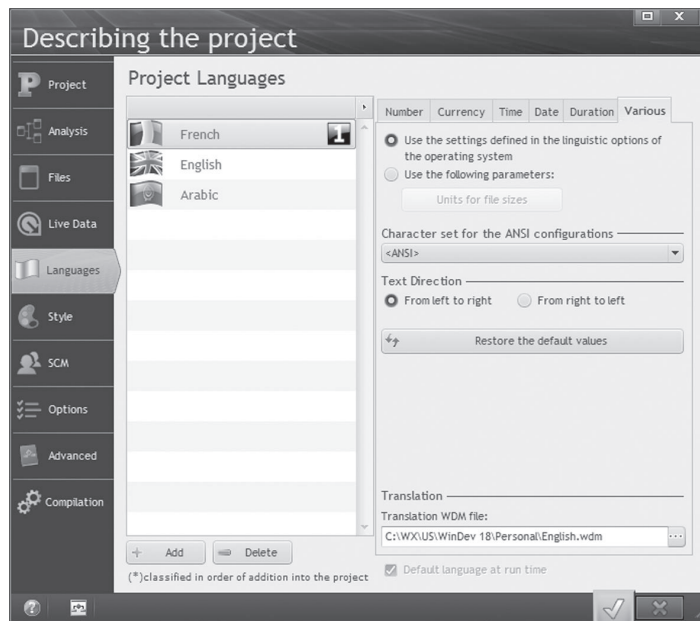
The translation of the help system can be performed in the help editor. All you have to do is change the display language. If the help system was not yet translated, the text will be displayed in the source language (French if the text was entered in French).

### Translating the result of the WLanguage functions and the error messages of the framework

Various information and messages are found in the WinDev framework. For example, the names of the 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 can use the function named **LoadError**.
- you can include the file in 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.

### Translation with WDMMSG and WDTRAD

A tool named **WDMMSG** (not supplied with WinDev) is used to:

- check out all the messages found in a project (caption of controls, code message, title of windows, ...) to translate them,
- check in the translated messages.

The messages to translate are checked out in a text format, which can be configured to be used by most translation software.

WDMMSG is also supplied with a tool for computer-aided translation, WDTRAD. WDTRAD is used to enter all the translations for the multilingual information found in a project.

Contact PC SOFT Sales Department for more details about **WDMMSG** and **WDTRAD**.

### Programming the change of language

By default, the project is run in the runtime language defined for the project, in the "Languages" tab of the project description ("Description" option on the "Project" pane).

The function named **Nation** is used to change the language.

In an application, the choice of language can be defined according to two methods:

- storing the language in a parameter file specific to the application.
- selecting the language via a menu option (method used in this example).

### Using a parameter file

With a parameter file, the choice of language is transparent for the user.

The language can be chosen when installing the project or when starting the application for the first time, and it can be modified afterwards.

To define the language in which the application will run, this parameter file must be read in the initialization process of the project.

The process is as follows:

```
Language is string; FileName is string
//Read the parameter file
FileName = "INTERNATIONAL.INI"
Language = INIRead("PARAMETER", "LANGUAGE", "FRENCH", FileName)
SWITCH Language
CASE "FRENCH":
 Nation(nationFrench)
 ChangeCharset(charsetAnsi)
CASE "ENGLISH":
 Nation(nationEnglish)
 ChangeCharset(charsetAnsi)
CASE "ARABIC":
 Nation(nationArabic)
 ChangeCharset(charsetArabic)
END
```

This process requires some explanations:

- The selected language is read in the "International.INI" file
- **Nation** is used to modify the runtime language of the project.
- The constant passed in parameter to **Nation** corresponds to the selected language.

**Nation** must be used in the initialization process of the project because the change of language will be effective in the next window that will be opened.

If **Nation** is called in the opening code of the first project window, this window will be displayed in the default language and the change of language will be effective for the other windows only.

On the contrary, if **Nation** is called in the initialization code of the project, the first project window will be displayed in the language selected by **Nation**.

This code also uses **ChangeCharset**. This function is needed to manage languages that use a specific character set.

### Using menu options

In the menu of the "WIN\_Menu.WDW" window, the "Languages" option lets you change the language. The code associated with these options is as follows:

```
- - Menu choice: &Languages..&French
Nation(nationFrench)
ChangeCharset(charsetAnsi)
Use(WIN_ContributorForm)
- - Menu choice: &Languages..&English
Nation(nationEnglish)
ChangeCharset(charsetAnsi)
Use(WIN_ContributorForm)

- - Menu choice: &Languages..&Arabic
Nation(nationArabic)
ChangeCharset(charsetArabic)
Use(WIN_ContributorForm)
```

**Use** enables you to reload the window passed in parameter. Therefore, the change of language performed by **Nation** is immediately effective.

- Run the test of the project.

## Managing the Unicode and the specific character sets

WinDev proposes two management modes:

- **the single charset mode:** In this mode, if your application manages languages that use specific character sets (Greek, Korean, ...), you have the ability to change by programming:
  - the character set used by all the screen fonts (**ChangeCharset**). This character set can also be used for the data saved in the HyperFileSQL files. See the online help (Keyword: multilingual) for more details.
  - the language of the keyboard handled by the users of the application (**ChangeKeyboard**).
- **the full support of Unicode:** This mode is used to manage the non-Latin character sets. In this mode, **ChangeCharset** and **ChangeKeyboard** are not required. The change of character set is automatically performed according to the language options defined in the project description. You also have the ability to mix different character sets.

The mode for managing the Unicode format is configured in the options of the current configuration ("Unicode" tab in the description window of the configuration).



Note

### Important

If you choose to switch your projects to UNICODE, the ANSI/Unicode cohabitation generates some constraints: the data must be translated when switching from an encoding system to the other one. The exchange of text strings must be adapted to take into account the new memory representation of the strings (in Unicode, 1 character is coded on 2 bytes while in ANSI, it is coded on 1 byte): calling APIs, reading or writing on disk to exchange data, reading or writing sockets containing strings, ...

If you choose to switch your projects to Unicode, a wizard measuring the impact of switching to Unicode is triggered on the project during the switch to Unicode.

## CONCLUSION

The tutorial is over now!

This course discussed a variety of subjects, but not all the features of WinDev, from far!

You are now familiar with the main concepts.

We recommend that you spend another day exploring the different menu choices of WinDev, for each one of the modules.

You can also explore the examples supplied with WinDev: some are simple and only address one topic, while others are more complex. These examples illustrate the different aspects of WinDev. Reading the source code is also a good way to learn.

It would take too much room to discuss all the available topics (there are hundreds, even thousands!). WinDev proposes several other features not presented in this tutorial:

- user groupware
- sockets, 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 18!**





