# A crash course for development with InstantSolutions Framework of Ethea Srl

**Revision of 09/02/2024 referring to InstantSolutions 8.6.1**

## General Index

# 1. Introduction: what is InstantSolutions

## 1.1. *Introduction to development with InstantSolutions Framework*

InstantSolutions is one wayinnovativeto conceive the development of applications based at all levels on OOP paradigms. It combines the power of SQL engines with the flexibility of object-oriented development in a simple and above all transparent way for the developer. All this is possible thanks to a mechanism called Object Persistence Framework (OPF), which allows the storage of objects within relational databases, such as**InstantObjects**and thanks to the use of an object-oriented programming language such as Delphi.

The data, in the form of classes, is described in a singleData Dictionary integrated into both the development environment and the application, from the name**ISWorkbench**.ISWorkbench is the heart of the model of an application developed with InstantSolutions.

But InstantSolutions is not just a development system based on an OPF. It makes 2 available to developersReady-made application frameworks to create applications of different types and with different user targets.

The open architecture allows the integration of third-party technologies, while its layered structure allows you to clearly separate the business logic from the presentation part (user interface).

Instant Solutions8.5works with Delphi (versionsDelphi 11and Delphi 12) and supports all Microsoft operating systemscurrently under maintenance heretolile various versions ofWindows 10and Windows 11.

## 1.2. *OO development and the goal of InstantSolutions*

The objective of Ethea's InstantSolutions project is to use an application development framework that helps (and, if necessary, forces) the developer to always be consistent with the principles of OOP, from the user interface level up to persistence of data in SQL database.

## 1.3. *Object storage and the relational model*

In object-oriented technology, the complexity of data is contained (encapsulation) in the object, whose data (attributes) are accessible through simple and coherent interfaces (methods). On the other hand, the relational model itself provides a coherent interface (for example the SQL language) but, since it does not manage the complexity of the data in any way, it is the developer who must constantly refer to this abstract complexity. Furthermore, some strictly OO paradigms (such as inheritance) are difficult to implement and complicated to maintain within a relational database.

The framework reduces the difficulties relating to data access, providing the developer with a coherent interface towards the data, which are always "seen" as object classes, and automatically managing the translation of data access operations into SQL language. In particular situations, to achieve the performance achievable only from server-side processing by the SQL engine, InstantSolutions provides support for mapping logical views and stored procedures to classes.

## 1.4. *Layering and inheritance versus "spaghetti basic" development*

InstantSolutions has chosen the logic of code stratification and inheritance as a basic principle, accustoming the developer to staying within the boundaries suitable for the code being written. Here is the scheme:

| Level | Development activities | Maintenance worker |
|---|---|---|
| **Custom GUI** | **Custom GUI development for advanced functions** | **Developer** |
| Basic GUI | MultiFramework,ConsoleFramework | Ethea |
| **Custom business logic** | **Specific business logic development** | **Developer** |
| Basic business logic | Auto-generated business logic from**ISWorkbench** | Ethea |
| Data access | InstantObjects Broker Development | InstantObjects |
| Physical level | SQL Relational Database Development | Database Vendors |

It is possible to notice how the developer intervenes only at 2 levels (highlighted in bold) based on as many levels already implemented by Ethea and therefore is naturally led not to "trespass" and to correctly follow the separation of the code.

# 2. Instant Solutions 8.6 versions

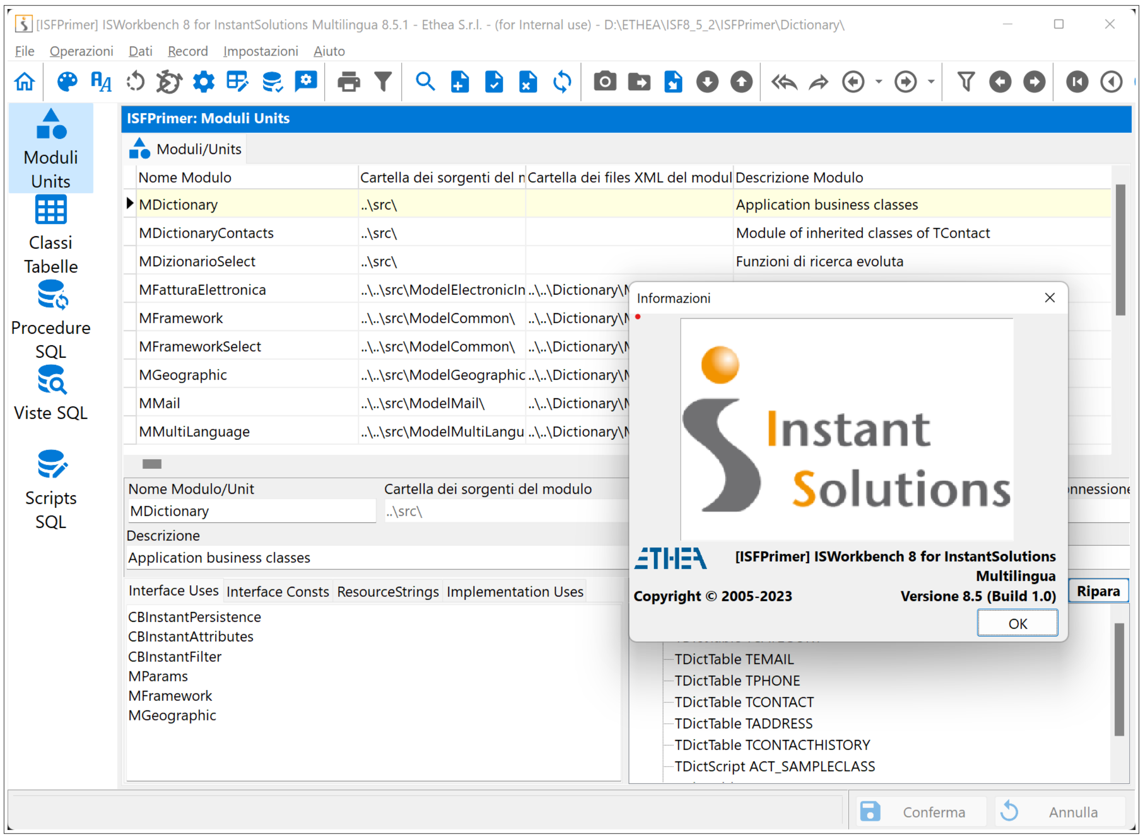
The version 8.6 exists in2versions:

1) Basic version (team development)

3) Multilingual version (user interface in multiple languages and multilingual data management).

For details on multilingual, read the dedicated section.

The heart of the development is the ISWorkbench application, through which all the main organizational functions of an InstantSolutions project are brought together.



[ISFPrimer] ISWorkbench 8 for InstantSolutions Multilingua 8.5.1 - Ethea S.r.l. - (for Internal use) - D:\ETHEA\ISF8_5_2\ISFPrimer\Dictionary\

ISFPrimer: Moduli Units

Moduli/Units

| Nome Modulo | Cartella dei sorgenti del n | Cartella dei files XML del modul | Descrizione Modulo |
|---|---|---|---|
| MDictionary | ..\src\ | | Application business classes |
| MDictionaryContacts | ..\src\ | | Module of inherited classes of TContact |
| MDizionarioSelect | ..\src\ | | Funzioni di ricerca evoluta |
| MFatturaElettronica | ..\..\src\ModelElectronicIn | ..\..\Dictionary\ | |
| MFramework | ..\..\src\ModelCommon\ | ..\..\Dictionary\ | |
| MFrameworkSelect | ..\..\src\ModelCommon\ | ..\..\Dictionary\ | |
| MGeographic | ..\..\src\ModelGeographic | ..\..\Dictionary\ | |
| MMail | ..\..\src\ModelMail\ | ..\..\Dictionary\ | |
| MMultiLanguage | ..\..\src\ModelMultiLangu | ..\..\Dictionary\ | |

Nome Modulo/Unit: MDictionary

Cartella dei sorgenti del modulo: ..\src\

Descrizione: Application business classes

CBInstantPersistence
CBInstantAttributes
CBInstantFilter
MParams
MFramework
MGeographic

TDictTable TEMAIL
TDictTable TPHONE
TDictTable TCONTACT
TDictTable TADDRESS
TDictTable TCONTACTHISTORY
TDictScript ACT_SAMPLECLASS

Informazioni

Instant Solutions

[ISFPrimer] ISWorkbench 8 for InstantSolutions Multilingua

Copyright © 2005-2023

Versione 8.5 (Build 1.0)

## 2.1. *Features of InstantantSolutions8.6*

The **version 8.6** use only **FireDAC** as InstantObjects Brokers for accessing data both within ISWorkBench and applications: in this way the DB license is no longer requiredFormerpress which was needed in previous versions.

The supported delphi versions are the**11and the 12th.** Previous versions are no longer supported.

AND'support has been added to**MarkDownHelpViewer**a new help system integrated into its ISF applications.

Support has been added for**StyledComponents**, a set of visual/graphical components such as Buttons, Toolbars,DbNavigatorand the new TaskDialogs including animations with Skia4Delphi.

Asubstantial difference compared to ISF7.6concerns thethe new management of icons based onthecomponentAnddevelopedorfrom Etheaand the use of**ImageName**rather than**ImageIndex**:

**- SVGIconImageCollection**(which exploitsTheSVG format) also coloredpaired with**VirtualImageList**.

No.B. IconFontsImageList support has been removed!

PFor details, see the chapter:***Icon ManagementSVGand IconName in the dfm***

The **version 8** contains some newsalready present in 7.6such as the management of the "HomePage" and the new FlexcelDocProducer, as well as a refactoring of the main forms which no longer use the "docking" system but a "splitview" to host the menu on the left and the SQL monitor (optional) on the right.

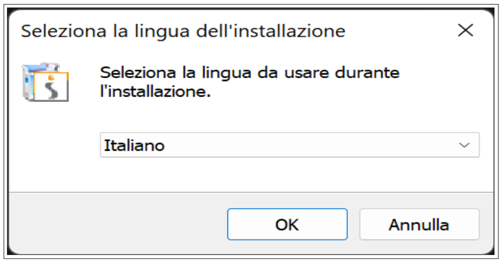
**For more informationon the passage of previous versionsconsult the document Migration_from_ISF_7_a_ISF_8.pdf.**

## 2.2. *MARS Server framework added*

Starting from version 7.3.2, a new development framework for Server/REST applications has been introduced using the MARS and delphi-neon libraries: see dedicated chapter.

# 3. Installing the framework

## 3.1. *Framework installation procedure and language choice*



Seleziona la lingua dell'installazione

Seleziona la lingua da usare durante l'installazione.

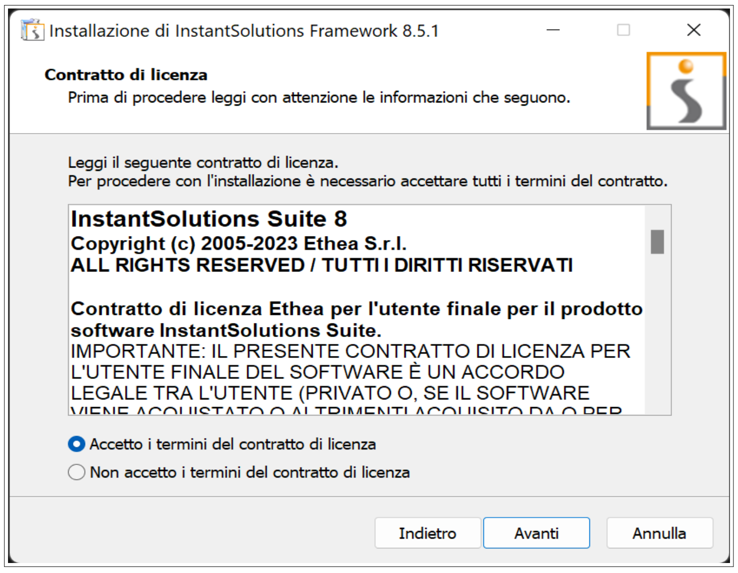
Italiano

Installing the framework is very simple. It is not invasive, but simply consists of creating the development environment.

*Notice: The choice of language at the beginning of the installation affects the set of framework .dfm files that are installed and used (in English or Italian). In this way in the Delphi IDE you will be able to see the basic forms of the framework in Italian or English.*

However, there is a backup of these dfms in the ITA_dfm and ENG_dfm folders, so it is always possible to use one or the other set of files, copying them into the Src folder of the framework and overwriting those present.

**The installation includes the following modules:**



Installazione di InstantSolutions Framework 8.5.1

**Contratto di licenza**

Prima di procedere leggi con attenzione le informazioni che seguono.

Leggi il seguente contratto di licenza.
Per procedere con l'installazione è necessario accettare tutti i termini del contratto.

**InstantSolutions Suite 8**
**Copyright (c) 2005-2023 Ethea S.r.l.**
**ALL RIGHTS RESERVED / TUTTI I DIRITTI RISERVATI**

**Contratto di licenza Ethea per l'utente finale per il prodotto software InstantSolutions Suite.**
IMPORTANTE: IL PRESENTE CONTRATTO DI LICENZA PER L'UTENTE FINALE DEL SOFTWARE È UN ACCORDO LEGALE TRA L'UTENTE (PRIVATO O, SE IL SOFTWARE

[✓ Accetto i termini del contratto di licenza]
Non accetto i termini del contratto di licenza

- OpenSource component sources:
  - SynEdit, ChromeTabs, HTMLViewer, OnGuard
- Documentation
- InstantSolutions Workbench
- Styled Components
- Markdown Components
- CBLib component library sources
- ISFLib component library sources
- Sources Application frameworks
  - MultiFramework Template (form in Italian or English)
  - ConsoleTemplate (console applications)
- Demo sources of the ISFPrimer application
- Framework class dictionary (XML)
- License management template

The installation takes place in a folder of your choice. You can use multiple releases at the same time on the same machinein a simple way. To do this your projects must reside within the framework folder.

## 3.2. *Installing components in the Delphi IDE*

To install all the components in the Delphi IDE you can use the convenient project group provided (first yesruns the buildthen some “run-time” packagesthey are installedthe “design-time” packages.

In the IDE there will be new components. Some are the components of CBLib, a basic library of components developed by Ethea (IS pages.....).

Among these components, the "doc-producer" stands out, a highly advanced technology for generating prints and documents and CBXDbMultiEdit, a componentAndwidely used in ISF for dynamic "form" design.

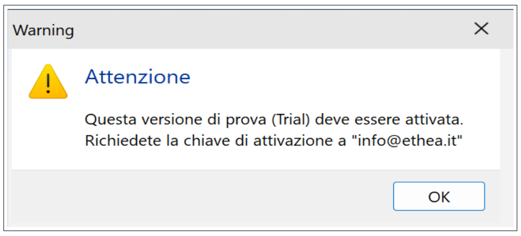
Then there is the InstantObjects native components page and the related extension page of the same contents in theISFLib.

Notice:All installation steps are well explained in the document Installazione_ISF8.rtf provided.

## 3.3. *Using ISWorkbench 8 and the related repository*

The installation also provides in the Windows menu access to ISWorkbench for supported Delphi versions: ISFTemplate. ISWorkbench is just the GUI that operates on a repository in XML format present in the project folder. By configuring the parameter it starts directly by opening the correct repository.

The functioning of ISWorkbench can vary depending on the repository on which it is made to work: inside the repository folder there is the ISWorkbench.ini file which defines some important parameters for the functioning of ISWorkbench. Refer to the ISWorkbench help in the configuration chapter for further details.



Warning

Attenzione

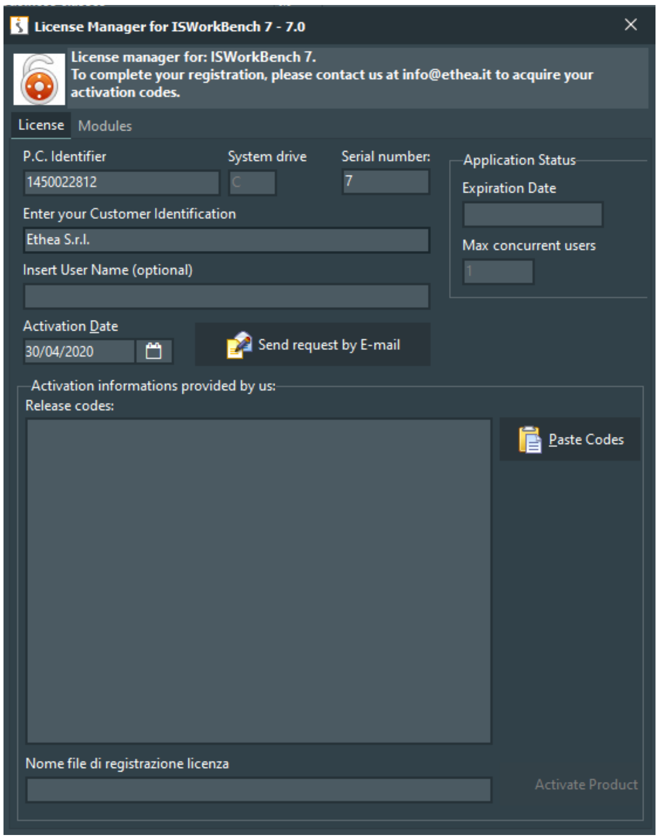
Questa versione di prova (Trial) deve essere attivata.
Richiedete la chiave di attivazione a "info@ethea.it"

### 3.3.1. ISWorkbench registration

To use ISWorkbench it is necessary to register the product, upon first use there will be this message, then the map for registration will be proposed.

Enter the denomination of Town company, the optional username and send the email with the registration data to info@ethea.it.

In the response email there will be the activation codes to copy in the box below, and click on "Activate Product" for activation.

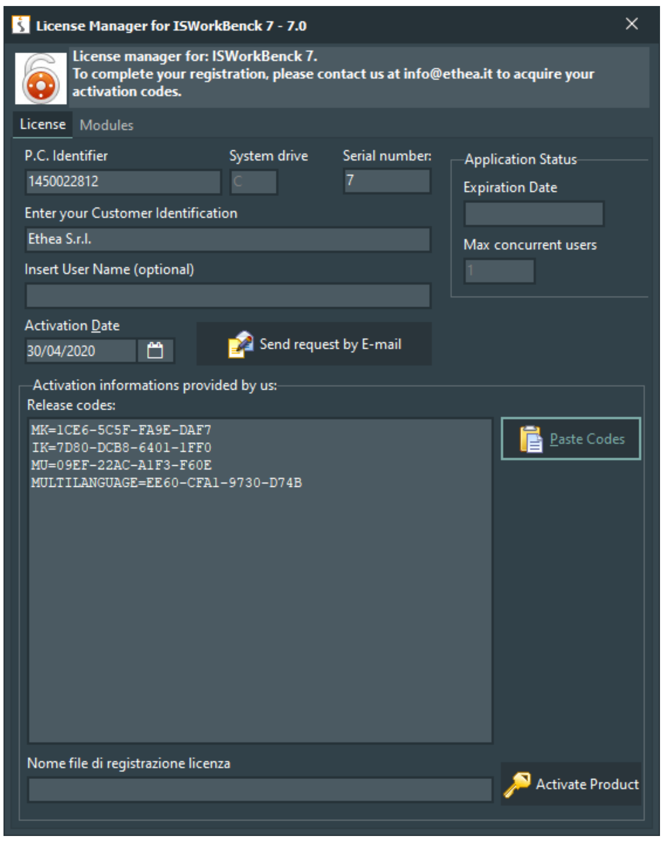


License Manager for ISWorkBench 7 - 7.0

License manager for: ISWorkBench 7.
To complete your registration, please contact us at info@ethea.it to acquire your activation codes.

License | Modules

P.C. Identifier: 1450022812
System drive: C
Serial number: 7
Enter your Customer Identification: Ethea S.r.l.
Insert User Name (optional):
Activation Date: 30/04/2020
Application Status
Expiration Date:
Max concurrent users: 1
Activation informations provided by us:
Release codes:
Nome file di registrazione licenza:



License Manager for ISWorkBenck 7 - 7.0

License manager for: ISWorkBench 7.
To complete your registration, please contact us at info@ethea.it to acquire your activation codes.

License | Modules

P.C. Identifier: 1450022812
System drive: C
Serial number: 7
Enter your Customer Identification: Ethea S.r.l.
Insert User Name (optional):
Activation Date: 30/04/2020
Application Status
Expiration Date:
Max concurrent users: 1
Activation informations provided by us:
Release codes:

```
MK=1CE6-5C5F-FA9E-DAF7
IK=7D80-DCB8-6401-1FF0
MU=09EF-22AC-A1F3-F60E
MULTILANGUAGE=EE60-CFA1-9730-D74B
```

Nome file di registrazione licenza:

## 3.4. *Markdown as Help system*

Ethea has abandoned the use of HelpScribble as a help generation system, replacing it with the more modern Help systembased on the MarkDown format. The application Help viewer is available in an Ethea Open-Source project: https://github.com/EtheaDev/MarkdownHelpViewer.

In this project there is a convenient setup for installing the Help display in Markdown format.

ISWorkBench generates the basic structure of an ISF application's Help files.

## 3.5. *Editor for Help in MarkDown format*

In another OpenSource project (https://github.com/EtheaDev/MarkdownShellExtensions) a convenient editor is available that allows you to easily edit Help pages in MarkDown format.

### 3.5.1. MarkDownHelp Viewer (new format)

Ethea has developed a new Help display mechanism based on filesmarkdownexported fromISWorlbench.

To use the new system you need to modify the project file by replacing the unitISHTMLHelpviewer thererigto:

```
MarkDownHelpViewer in '..\..\..\ext\MarkDownHelpViewer\Source\AppInterface\MarkDownHelpViewer.pas',
```

This uses the unit of the new interface and the Help display form based onLcomponentAnd MarkDownHelpViewer.

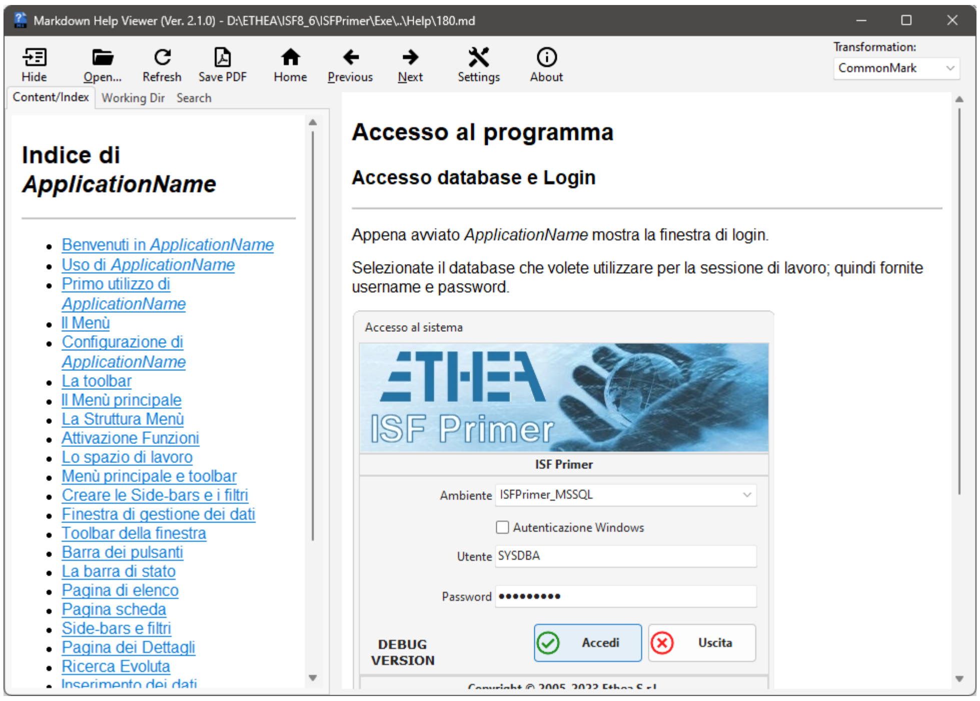
To ensure that the application loads the right file, you need to set the name of the help file inside the .ini file like this:

**HelpFile={app}\..\Help\ISFPrimerHelp.md**

To test the operation of the help, by pressing F1 the user is redirected to the specific page of the class/attribute he is editing or managing.

The help window remembers where it was last moved/positioned, even after you close the application.

*In this image an example of the markdownhelpviewer invoked from the Login window:*



Markdown Help Viewer (Ver. 2.1.0) - D:\ETHEA\ISF8_6\ISFPrimer\Exe\..\Help\180.md

Transformation: CommonMark

**Indice di *ApplicationName***

- Benvenuti in *ApplicationName*
- Uso di *ApplicationName*
- Primo utilizzo di *ApplicationName*
- Il Menù
- Configurazione di *ApplicationName*
- La toolbar
- Il Menù principale
- La Struttura Menù
- Attivazione Funzioni
- Lo spazio di lavoro
- Menù principale e toolbar
- Creare le Side-bars e i filtri
- Finestra di gestione dei dati
- Toolbar della finestra
- Barra dei pulsanti
- La barra di stato
- Pagina di elenco
- Pagina scheda
- Side-bars e filtri
- Pagina dei Dettagli
- Ricerca Evoluta
- Inserimento dei dati

**Accesso al programma**

**Accesso database e Login**

Appena avviato *ApplicationName* mostra la finestra di login.

Selezionate il database che volete utilizzare per la sessione di lavoro; quindi fornite username e password.

Accesso al sistema

ISF Primer

| | |
|---|---|
| Ambiente | ISFPrimer_MSSQL |
| | ☐ Autenticazione Windows |
| Utente | SYSDBA |
| Password | ••••••••• |

## *3.6. Testing the development environment*

To check if we have installed the environment correctly, let's try to compile and start itTheprojectorof ISF examplePrimerswhich are located in the ISF foldersPrimers\Projects\DXXX (XXX is the Delphi version).

# 4. Notes on the CBLib (ISRtl, ISVcl) and ISFLib libraries

One of the building blocks of InstantSolutions are the CBLib library components.

ISF is the collection of InstantObjects, CBLib and ISFLib

Let's see some features of CBLib, which is a Delphi component library developed by Ethea, of which you have all the sources.

## *4.1. The components of the library*

The CBLib library (consisting of an RTL part (ISRTL) and a VCL part (ISVCL), contains many components, which can be used for your own InstantSolutions applications, but also for different applications.

Based on the setting of two global variables (CbFlatControls and CBNoBorderControls) it is possible to change the appearance of all the components of an application (to make it more modern).

Let's see a very quick overview of the components.

### 4.1.1. ISControls components

All the visual components that have been customized to have some very convenient features are part of this family:

– All "editing" components have an "EditCaption" property which is a label anchored to the editing component that facilitates the construction of "forms".
– Transparencies are handled in this waydto work properly with Windows 10/11and enabled themes.
– Some editing components not present in Delphi (editing of numbers from the right, of dates/times with calendar, with a button associated with the edit.
– A “dataaware” component that displays data from a dataset in a DbListView (TCBXDbListView)
– A “Data diffing” component for managing comparisons and merges between data from two datasets (TCBDataDiffing).

Here is the list of components available in the “ISControls” palette:

```
TCBChart, TCBXActionList, TCBXBitBtn, TCBXButton, TCBXButtonEdit, TCBXCheckBox, TCBXCheckListBox,
TCBXColorBox, TCBXColorGrid, TCBXComboBox, TCBXControlBar, TCBXCurrencyEdit, TCBXEdit, TCBXFormResize,
TCBXGroupBox, TCBXImage, T CBXImageList, TCBXLabel, TCBXListBox, TCBXListView, TCBXMainMenu, TCBXMaskEdit,
TCBXMemo, TCBXPageControl, TCBXPanel, TCBXPopUpMenu, TCBXRadioButton, TCBXRadioGroup, TCBXRichEdit,
TCBXScrollBox, TCBXSpeedButton, TCBXSpinEdit, TCBXSplitter, TCBXStringGrid, TCBXTabControl,
TCBXThinTrackBar, TCBXTitle, TCBXTreeView, TCBXDBListView, TCBDataDiffing, TCBClock
```

### 4.1.2. TCBXBitBtn

For reasons of incorrect "rendering" of images on Delphi's TBitBtn, the TCBXBitBtn component actually inherits from TButton, so that it can correctly display IconFonts or SVGIcon icons drawn with the Alpha channel (transparent) via GDI+.

### 4.1.3. ISDataAccess components

Among the data access components, the TCBClientDataSet stands out, which has some features to work directly on files, saving the data in "formatted, ordered and indented" XML format, to facilitate its use with versioning systems. Furthermore there are a series of useful components for exporting and importing data in various formats (Excel, CSV, Text, XML).

Here is the list of components available in the “ISDataAccess” palette:

```
TCBDatasetToExcel, TCBDatasetToFlexcel, TCBClientDataSet, TCBDataSource, TCBDatasetToCSV, TCBDatasetToText,
TCBDatasetToXML, TCBDatasetFromExcel, TCBDataSetFromFlexCel, TCBDatasetFromCSV, TCBDatasetFromText,
TCBDatasetFromXML, TCBDatasetFromExcel, TCBADOQuery, TCBADOTable, TCBADODataSet
```

### 4.1.4. ISDataControls components

The "dataaware" components of CBLib provide the same features as the editing components (in particular the EditCaption and specialized data input modes, for amounts, numbers, dates/times, etc...).

In particular, the TCBXDbImageLink component allows the input of a string "linked" to an image.

All these DataAware components are then "exploited" by one of the most used components in InstantSolutions applications, namely the**TCBXDbMultiEdit**: this component (as we will see later) allows a very simplified but at the same time very powerful management of a data entry map, using the "Custom Layout" concept.

Here is the list of components available in the “ISDataControls” palette:

```
TCBDbChart, TCBXDBButtonEdit, TCBXDBCheckBox, TCBXDBComboBox, TCBXDBCurrencyEdit, TCBXDBEdit,
TCBXDBExtendedMemo, TCBXDbGrid, TCBXDBImage, TCBXDbImageLink, TCBXDBLabel, TCBXDBListBox,
TCBXDBLookupComboBox, TCBXDBLookupListBox, TCBXDBMemo, TCBXDBMultiEdit, TCBXDBNavigator,TCBXDBRadioGroup,
TCBXDBText, TCBDbBlob, TCBDBMultiValue, TCBDBReferenceButtons, TCBDBMultiValue,TCBDBNumberBox, TCBNumberBox
```

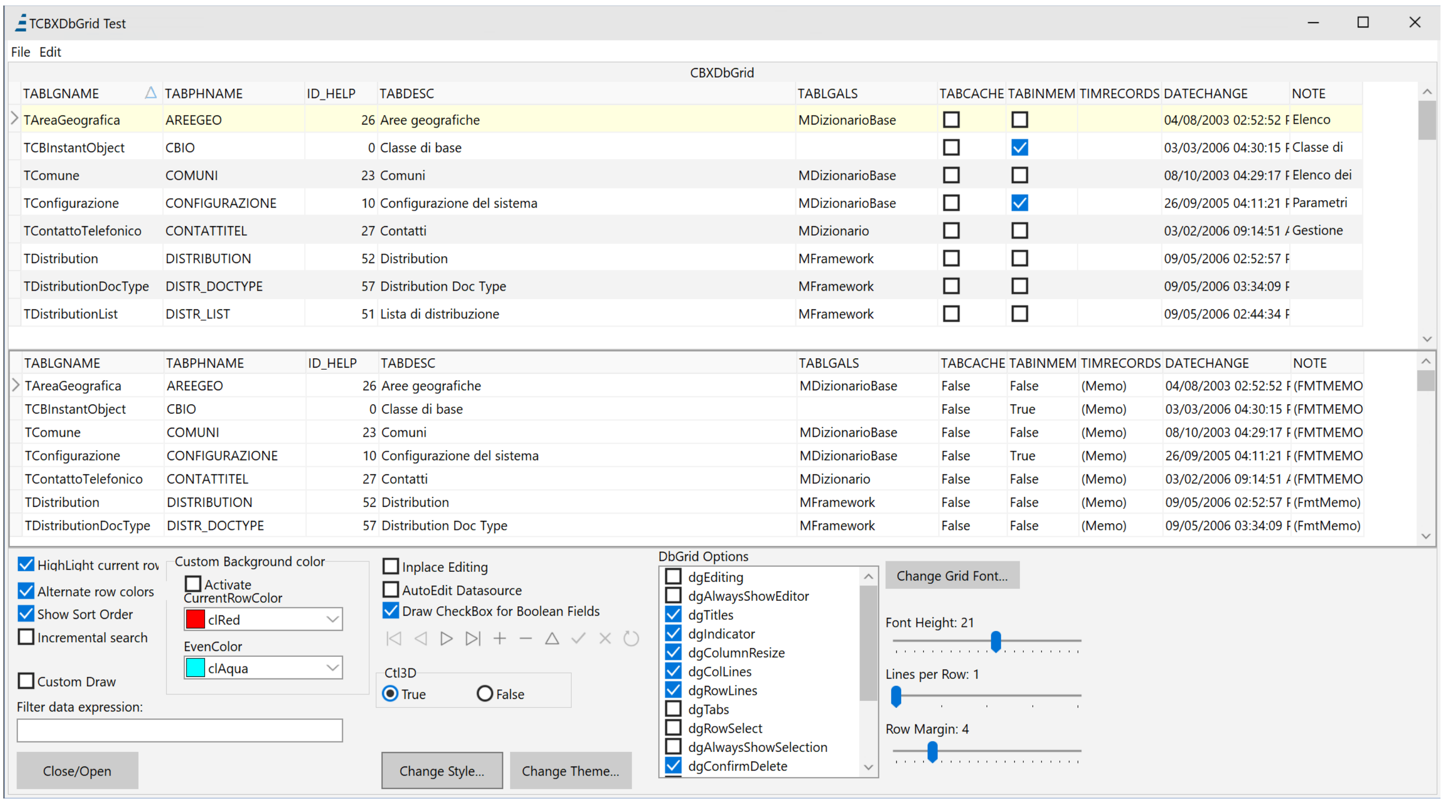
### 4.1.5. Editing components with BoundCaption

All CBLib editing components have the characteristic of having a**"Bound Caption"**(TCBXBoundLabel) or a label attached to the editing component in a position "around" the component, among those available: lpTopLeft (the default), lpTopCenter, lpTopRight, lpBottomLeft, lpBottomCenter, lpBottomRight, lpLeftTop, lpLeftMiddle, lpLeftBottom, lpRightTop, lpRightMiddle, lpRightBottom. This allows you to always keep the component's caption label attached and aligned, even if the component is moved.

### 4.1.6. The new TCBXDbGrid

As well as providingLsame concept as "Custom Layout",the newTCBXDbGrid, also provides a series of very advanced features, such as the ability to sort data, perform incremental searches, apply custom colors to rows, etc...

From ISF 7.6 it is now also possible to control the default height of the rows, defining margins, or define that in each "record row" "multiple rows of data" must be shown, when for example you want to show the contents of a "Memo" field: here is an example ofTCBXDBGrid with margins and 3 rows for each record:



| TABLGNAME △ | TABPHNAME | ID_HELP | TABDESC | TABLGALS | TABCACHE | TABINMEM | TIMRECORDS | DATECHANGE | NOTE |
|---|---|---|---|---|---|---|---|---|---|
| TAreaGeografica | AREEGEO | 26 | Aree geografiche | MDizionarioBase | ☐ | ☐ | | 04/08/2003 02:52:52 F | Elenco |
| TCBInstantObject | CBIO | 0 | Classe di base | | ☐ | ✓ | | 03/03/2006 04:30:15 F | Classe di |
| TComune | COMUNI | 23 | Comuni | MDizionarioBase | ☐ | ☐ | | 08/10/2003 04:29:17 F | Elenco dei |
| TConfigurazione | CONFIGURAZIONE | 10 | Configurazione del sistema | MDizionarioBase | ☐ | ✓ | | 26/09/2005 04:11:21 F | Parametri |
| TContattoTelefonico | CONTATTITEL | 27 | Contatti | MDizionario | ☐ | ☐ | | 03/02/2006 09:14:51 / | Gestione |
| TDistribution | DISTRIBUTION | 52 | Distribution | MFramework | ☐ | ☐ | | 09/05/2006 02:52:57 F | |
| TDistributionDocType | DISTR_DOCTYPE | 57 | Distribution Doc Type | MFramework | ☐ | ☐ | | 09/05/2006 03:34:09 F | |
| TDistributionList | DISTR_LIST | 51 | Lista di distribuzione | MFramework | ☐ | ☐ | | 09/05/2006 02:44:34 F | |

| TABLGNAME | TABPHNAME | ID_HELP | TABDESC | TABLGALS | TABCACHE | TABINMEM | TIMRECORDS | DATECHANGE | NOTE |
|---|---|---|---|---|---|---|---|---|---|
| TAreaGeografica | AREEGEO | 26 | Aree geografiche | MDizionarioBase | False | False | (Memo) | 04/08/2003 02:52:52 F | (FMTMEMO |
| TCBInstantObject | CBIO | 0 | Classe di base | | False | True | (Memo) | 03/03/2006 04:30:15 F | (FMTMEMO |
| TComune | COMUNI | 23 | Comuni | MDizionarioBase | False | False | (Memo) | 08/10/2003 04:29:17 F | (FMTMEMO |
| TConfigurazione | CONFIGURAZIONE | 10 | Configurazione del sistema | MDizionarioBase | False | True | (Memo) | 26/09/2005 04:11:21 F | (FMTMEMO |
| TContattoTelefonico | CONTATTITEL | 27 | Contatti | MDizionario | False | False | (Memo) | 03/02/2006 09:14:51 / | (FMTMEMO |
| TDistribution | DISTRIBUTION | 52 | Distribution | MFramework | False | False | (Memo) | 09/05/2006 02:52:57 F | (FmtMemo) |
| TDistributionDocType | DISTR_DOCTYPE | 57 | Distribution Doc Type | MFramework | False | False | (Memo) | 09/05/2006 03:34:09 F | (FmtMemo) |

[✓] HighLight current rov
[✓] Alternate row colors
[✓] Show Sort Order
[ ] Incremental search
[ ] Custom Draw
Filter data expression:

Custom Background color
[ ] Activate CurrentRowColor: 🟥 clRed
EvenColor: 🟦 clAqua

[ ] Inplace Editing
[ ] AutoEdit Datasource
[✓] Draw CheckBox for Boolean Fields

Ctl3D: [✓ True] False

DbGrid Options
[ ] dgEditing
[ ] dgAlwaysShowEditor
[✓] dgTitles
[✓] dgIndicator
[✓] dgColumnResize
[✓] dgColLines
[✓] dgRowLines
[ ] dgTabs
[ ] dgRowSelect
[ ] dgAlwaysShowSelection
[✓] dgConfirmDelete

Font Height: 21
Lines per Row: 1
Row Margin: 4

### 4.1.7. ISOpenOffice components

They are components for interfacing with OpenOffice. The most complete is theTOOoWriter, which together with TOOoDocProducer allows you to use OpenOffice as an editor and report generator.

Here is the list of components available in the "ISOpenOffice" palette:

TOOoDocument, TOOoWriter, TOOoCalc, TOOoDraw, TOOoImpress.

### 4.1.8. ISDocProducer components

They are components, "compatible" with each other, for the management, editing and generation of documents in various formats: Openoffice, ReportBuilder, HTML, FO (Formatting Object), XML, etc...

A new "DocProducer" for FlexcelReport has also been added since version 7.6 of ISF: refer to the document **"Using Flexcel Report with ISF.pdf"** for more information.

Here is the list of components available in the "ISDocProducer" palette:

```
TCBRBuilderDocProducer, TCBHtmlDocProducer, TCBFopDocProducer, TCBOooWriterDocProducer,
TCBXMLDocProducer,TCBFlexcelDocProducer.
```

### 4.1.9. IS componentsFireDAC

In previous versions of ISF, support was given to DBEpress, in the new version some components have been reported with the same previous functionality but replaced by FireDAC, some ad-hoc components are provided, in particularTCBFDSimpleDataSet, which allows client-side "caching" of a one-way dataset, and TCBFDMetadata, which allows you to extract metadata from DBs in a simple way.

Here is the list of components available in the “ISFireDAC”:

```
TCBFDMetadata, TCBFDConnection, TCBFDDataSet, TCBFDQueries, TCBFDStoredProc, TCBFDTable,TCBFDSimpleDataSet.
```

## *4.2. ISFLib*

### 4.2.1. InstantObjects ISF components

In the ISF library there are some components used directly by the InstantSlutions framework, most of which are extensions of the native InstantObjects components to be able to access the data contained in the class dictionary managed with ISWorkbench.

Here is the list of componentsnot visual:

```
TCBInstantExposer, TCBInstantSelector, TCBInstantClientDataSet,
TCBXMLFileAccessor, TCBJSONFileAccessor,
```

Here is the list of componentsvisuals,available in the “ISF” palette:

```
TCBInstantExplorer
```

### 4.2.2. Component TCBInstantClientDataSet for native forms

To optimize performance with InstantObjects, a "special" dataset has been developed which derives from TClientDataSet but which is able to remain synchronized with an InstantObject object, always aligned to the current record of the dataset.

This component is exploited internally of the “native forms”.

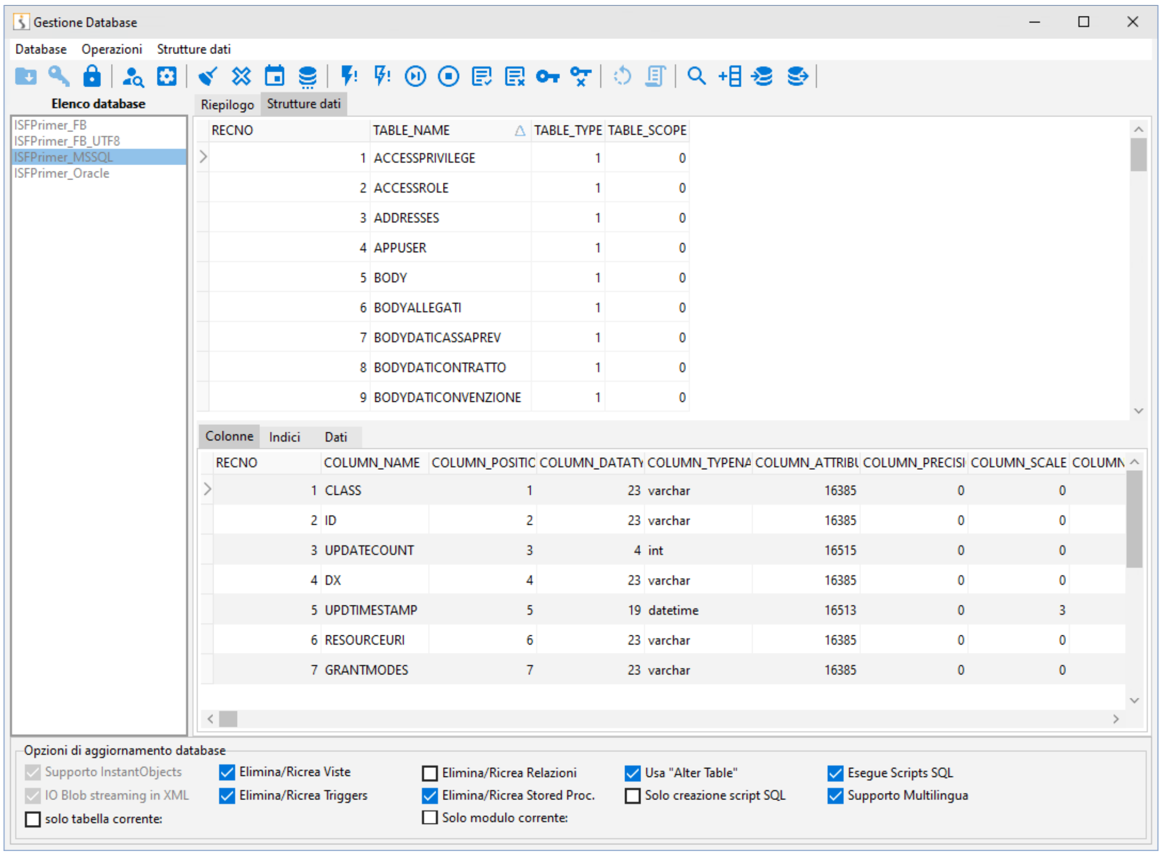
### 4.2.3. TCBInstantExplorer component

The TCBInstantExplorer visual component is a very advanced "explorer" compared to the standard InstantObjects one, because it is able to host both a TCBXDBGrid (when it must show a grid) and a TCBXDBMultiEdit (when it must show a data mask) within it .

## *4.3. Advanced CBLib editors*

In addition to the various components, there are also "advanced editors", in relation to the functioning of some components:

### 4.3.1. Ethea Database Manager



| RECNO | TABLE_NAME | TABLE_TYPE | TABLE_SCOPE |
|---|---|---|---|
| 1 | ACCESSPRIVILEGE | 1 | 0 |
| 2 | ACCESSROLE | 1 | 0 |
| 3 | ADDRESSES | 1 | 0 |
| 4 | APPUSER | 1 | 0 |
| 5 | BODY | 1 | 0 |
| 6 | BODYALLEGATI | 1 | 0 |
| 7 | BODYDATICASSAPREV | 1 | 0 |
| 8 | BODYDATICONTRATTO | 1 | 0 |
| 9 | BODYDATICONVENZIONE | 1 | 0 |

| RECNO | COLUMN_NAME | COLUMN_POSITIC | COLUMN_DATATY | COLUMN_TYPENA | COLUMN_ATTRIBU | COLUMN_PRECISI | COLUMN_SCALE | COLUMN |
|---|---|---|---|---|---|---|---|---|
| 1 | CLASS | 1 | 23 | varchar | 16385 | 0 | 0 | |
| 2 | ID | 2 | 23 | varchar | 16385 | 0 | 0 | |
| 3 | UPDATECOUNT | 3 | 4 | int | 16515 | 0 | 0 | |
| 4 | DX | 4 | 23 | varchar | 16385 | 0 | 0 | |
| 5 | UPDTIMESTAMP | 5 | 19 | datetime | 16513 | 0 | 3 | |
| 6 | RESOURCEURI | 6 | 23 | varchar | 16385 | 0 | 0 | |
| 7 | GRANTMODES | 7 | 23 | varchar | 16385 | 0 | 0 | |

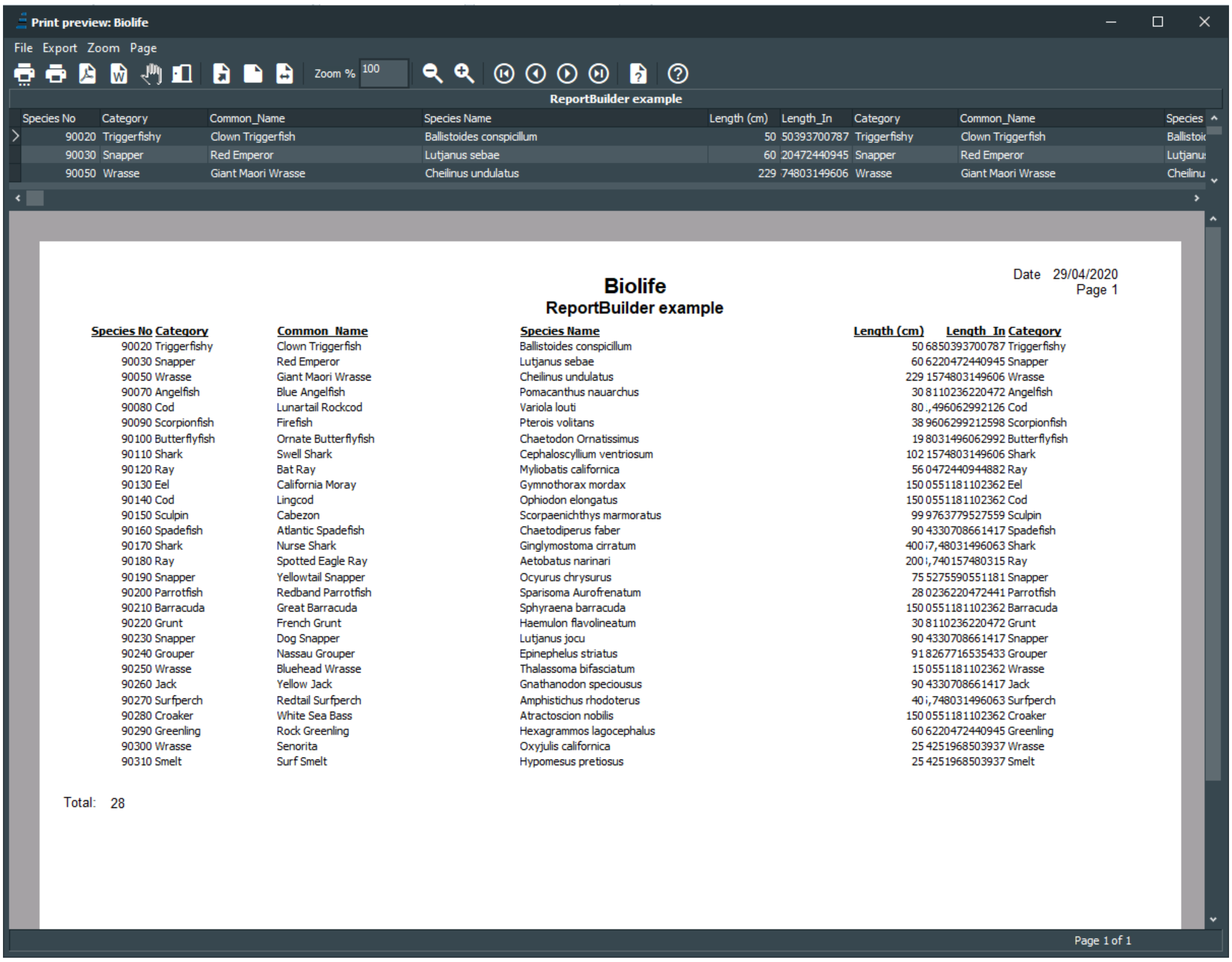
Opzioni di aggiornamento database

- ✓ Supporto InstantObjects
- ✓ IO Blob streaming in XML
- ☐ solo tabella corrente:
- ✓ Elimina/Ricrea Viste
- ✓ Elimina/Ricrea Triggers
- ☐ Elimina/Ricrea Relazioni
- ✓ Elimina/Ricrea Stored Proc.
- ☐ Solo modulo corrente:
- ✓ Usa "Alter Table"
- ☐ Solo creazione script SQL
- ✓ Esegue Scripts SQL
- ✓ Supporto Multilingua

It is a data source management window”FireDAC”: through a configuration file it is possible to access different databases, see their structure, and within an “SQL editor” it is possible to execute select, insert, update or delete queries, even in scripting mode.

### 4.3.2. Ethea ReportBuilder Preview (and Table Preview)

Ethea recommends ReportBuilder as a report generator, and also provides advanced support for this report generator, such as the customized preview (the standard RB one is very limited), but above all a custom preview to generate "tabular" reports automatically, by of the user: just provide the dataset and it does everything by itself: it is possible to move columns, define their width, to obtain the desired print layout, all "WYSIWYG". It is also possible to export the report to PDF.



ReportBuilder example

| Species No | Category | Common_Name | Species Name | Length (cm) | Length_In | Category | Common_Name | Species |
|---|---|---|---|---|---|---|---|---|
| 90020 | Triggerfishy | Clown Triggerfish | Ballistoides conspicillum | 50 | 50393700787 | Triggerfishy | Clown Triggerfish | Ballistoic |
| 90030 | Snapper | Red Emperor | Lutjanus sebae | 60 | 20472440945 | Snapper | Red Emperor | Lutjanu: |
| 90050 | Wrasse | Giant Maori Wrasse | Cheilinus undulatus | 229 | 74803149606 | Wrasse | Giant Maori Wrasse | Cheilinu |

**Biolife**

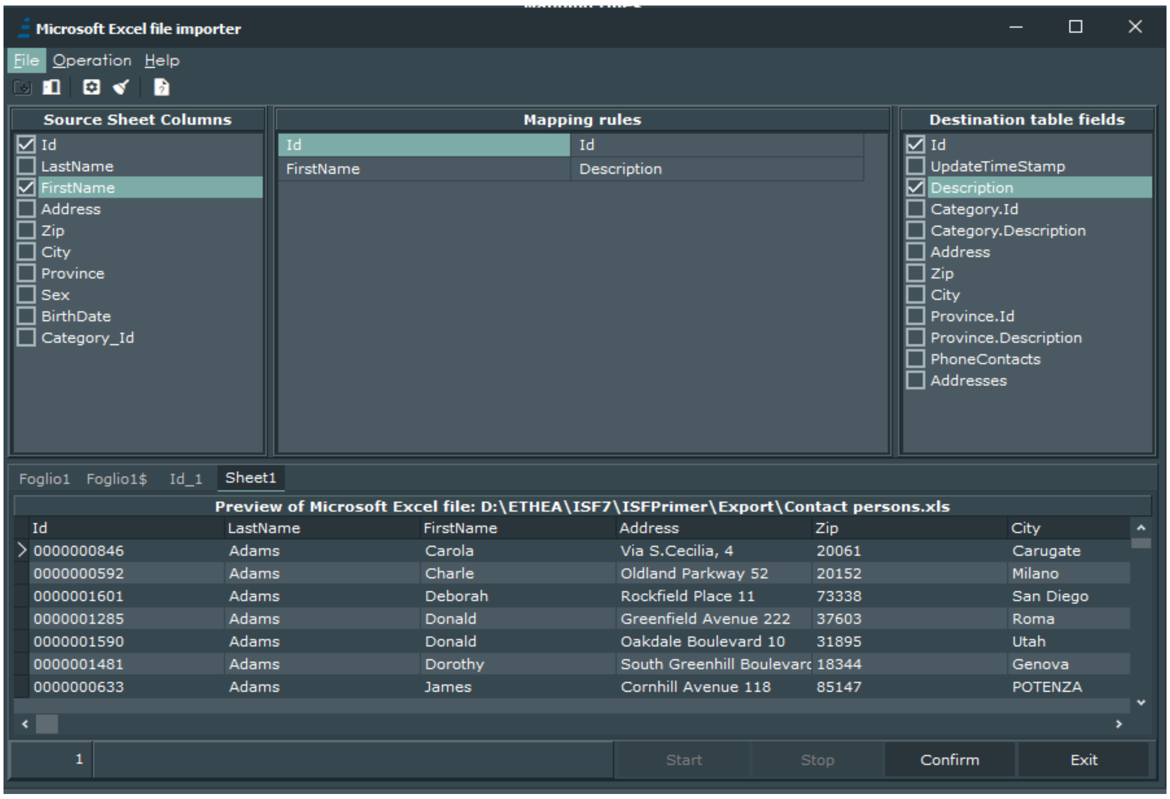
**ReportBuilder example**

Date 29/04/2020
Page 1

| Species No | Category | Common_Name | Species Name | Length (cm) | Length_In | Category |
|---|---|---|---|---|---|---|
| 90020 | Triggerfishy | Clown Triggerfish | Ballistoides conspicillum | 50 | 6850393700787 | Triggerfishy |
| 90030 | Snapper | Red Emperor | Lutjanus sebae | 60 | 6220472440945 | Snapper |
| 90050 | Wrasse | Giant Maori Wrasse | Cheilinus undulatus | 229 | 1574803149606 | Wrasse |
| 90070 | Angelfish | Blue Angelfish | Pomacanthus nauarchus | 30 | 8110236220472 | Angelfish |
| 90080 | Cod | Lunartail Rockcod | Variola louti | 80 | .,496062992126 | Cod |
| 90090 | Scorpionfish | Firefish | Pterois volitans | 38 | 9606299212598 | Scorpionfish |
| 90100 | Butterflyfish | Ornate Butterflyfish | Chaetodon Ornatissimus | 19 | 8031496062992 | Butterflyfish |
| 90110 | Shark | Swell Shark | Cephaloscyllium ventriosum | 102 | 1574803149606 | Shark |
| 90120 | Ray | Bat Ray | Myliobatis californica | 56 | 0472440944882 | Ray |
| 90130 | Eel | California Moray | Gymnothorax mordax | 150 | 0551181102362 | Eel |
| 90140 | Cod | Lingcod | Ophiodon elongatus | 150 | 0551181102362 | Cod |
| 90150 | Sculpin | Cabezon | Scorpaenichthys marmoratus | 99 | 9763779527559 | Sculpin |
| 90160 | Spadefish | Atlantic Spadefish | Chaetodiperus faber | 90 | 4330708661417 | Spadefish |
| 90170 | Shark | Nurse Shark | Ginglymostoma cirratum | 400 | i7,48031496063 | Shark |
| 90180 | Ray | Spotted Eagle Ray | Aetobatus narinari | 200 | :,740157480315 | Ray |
| 90190 | Snapper | Yellowtail Snapper | Ocyurus chrysurus | 75 | 5275590551181 | Snapper |
| 90200 | Parrotfish | Redband Parrotfish | Sparisoma Aurofrenatum | 28 | 0236220472441 | Parrotfish |
| 90210 | Barracuda | Great Barracuda | Sphyraena barracuda | 150 | 0551181102362 | Barracuda |
| 90220 | Grunt | French Grunt | Haemulon flavolineatum | 30 | 8110236220472 | Grunt |
| 90230 | Snapper | Dog Snapper | Lutjanus jocu | 90 | 4330708661417 | Snapper |
| 90240 | Grouper | Nassau Grouper | Epinephelus striatus | 91 | 8267716535433 | Grouper |
| 90250 | Wrasse | Bluehead Wrasse | Thalassoma bifasciatum | 15 | 0551181102362 | Wrasse |
| 90260 | Jack | Yellow Jack | Gnathanodon speciousus | 90 | 4330708661417 | Jack |
| 90270 | Surfperch | Redtail Surfperch | Amphistichus rhodoterus | 40 | i,748031496063 | Surfperch |
| 90280 | Croaker | White Sea Bass | Atractoscion nobilis | 150 | 0551181102362 | Croaker |
| 90290 | Greenling | Rock Greenling | Hexagrammos lagocephalus | 60 | 6220472440945 | Greenling |
| 90300 | Wrasse | Senorita | Oxyjulis californica | 25 | 4251968503937 | Wrasse |
| 90310 | Smelt | Surf Smelt | Hypomesus pretiosus | 25 | 4251968503937 | Smelt |

Total: 28

### 4.3.3. Ethea Excel Import Mapping Tool



Microsoft Excel file importer

File Operation Help

| Source Sheet Columns | Mapping rules | | Destination table fields |
|---|---|---|---|
| ✓ Id | Id | Id | ✓ Id |
| LastName | FirstName | Description | UpdateTimeStamp |
| ✓ FirstName | | | ✓ Description |
| Address | | | Category.Id |
| Zip | | | Category.Description |
| City | | | Address |
| Province | | | Zip |
| Sex | | | City |
| BirthDate | | | Province.Id |
| Category_Id | | | Province.Description |
| | | | PhoneContacts |
| | | | Addresses |

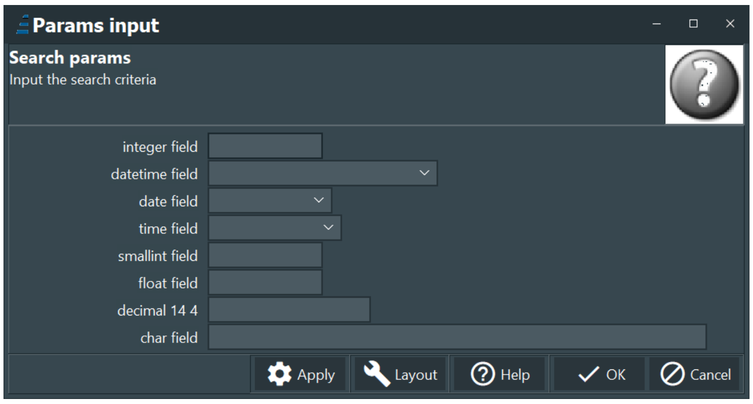
Foglio1 Foglio1$ Id_1 Sheet1

Preview of Microsoft Excel file: D:\ETHEA\ISF7\ISFPrimer\Export\Contact persons.xls

| Id | LastName | FirstName | Address | Zip | City |
|---|---|---|---|---|---|
| 0000000846 | Adams | Carola | Via S.Cecilia, 4 | 20061 | Carugate |
| 0000000592 | Adams | Charle | Oldland Parkway 52 | 20152 | Milano |
| 0000001601 | Adams | Deborah | Rockfield Place 11 | 73338 | San Diego |
| 0000001285 | Adams | Donald | Greenfield Avenue 222 | 37603 | Roma |
| 0000001590 | Adams | Donald | Oakdale Boulevard 10 | 31895 | Utah |
| 0000001481 | Adams | Dorothy | South Greenhill Boulevarc | 18344 | Genova |
| 0000000633 | Adams | James | Cornhill Avenue 118 | 85147 | POTENZA |

In addition to the import/export components for Excel, Ethea also provides a very advanced interface for importing the data contained in an Excel sheet: by selecting the "range" and defining the "mapping" of the fields it is very simple to import the data into a Dataset.

### 4.3.4. Ethea Param input form



Params input

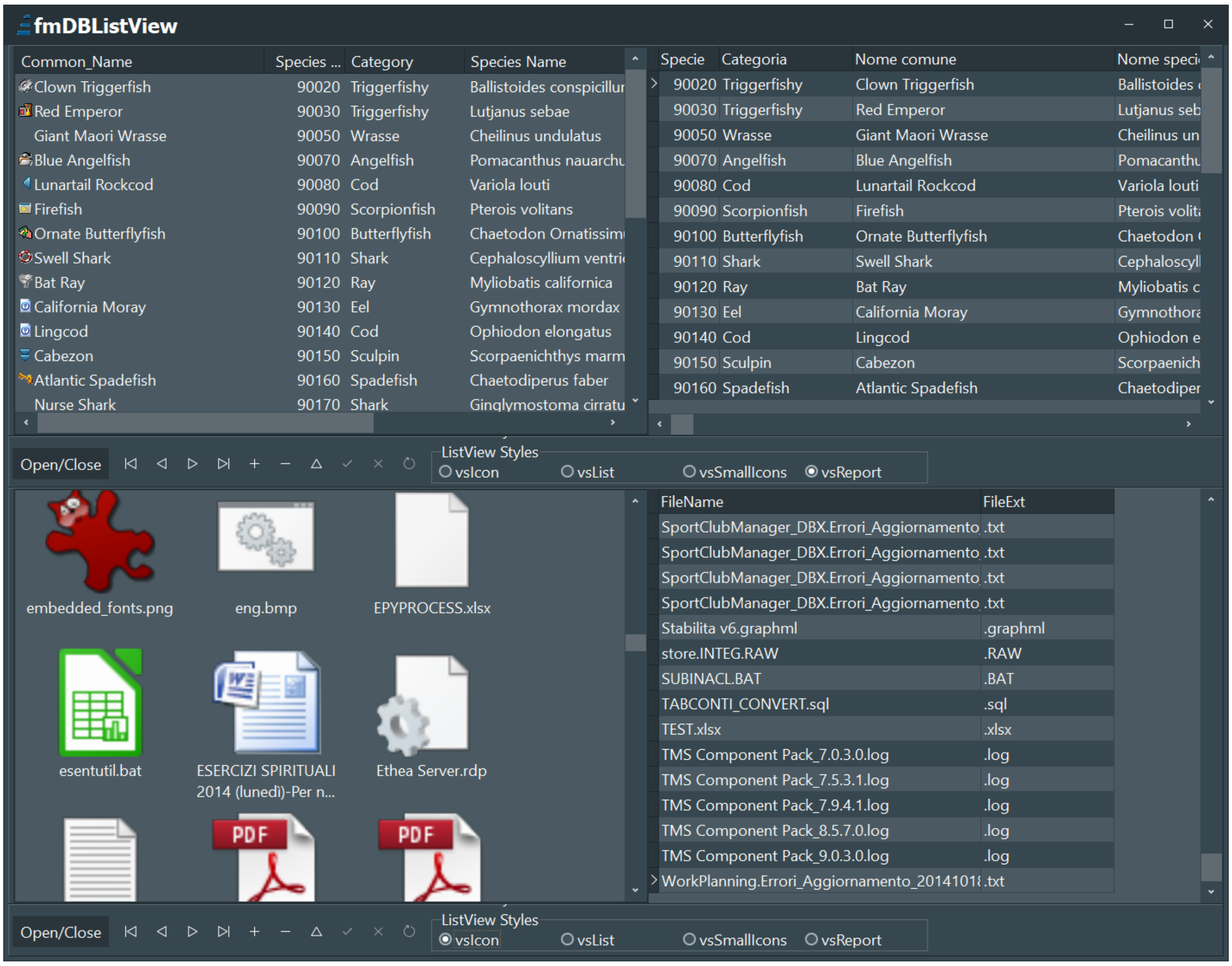
**Search params**

Input the search criteria

| | |
|---|---|
| integer field | |
| datetime field | |
| date field | |
| time field | |
| smallint field | |
| float field | |
| decimal 14 4 | |
| char field | |

There is also the possibility of easily providing the user with a window for inputting the parameters of a SQL query, via this editor: the input is defined by the type of parameters, so it takes advantage of the CBLib components, as well as the possibility of defining a layout “custom” for each set of parameters.

### 4.3.5. DbListView: an "alternative" way to the classic DbGrid.



fmDBListView

| Common_Name | Species ... | Category | Species Name |
|---|---|---|---|
| Clown Triggerfish | 90020 | Triggerfishy | Ballistoides conspicillu |
| Red Emperor | 90030 | Triggerfishy | Lutjanus sebae |
| Giant Maori Wrasse | 90050 | Wrasse | Cheilinus undulatus |
| Blue Angelfish | 90070 | Angelfish | Pomacanthus nauarchu |
| Lunartail Rockcod | 90080 | Cod | Variola louti |
| Firefish | 90090 | Scorpionfish | Pterois volitans |
| Ornate Butterflyfish | 90100 | Butterflyfish | Chaetodon Ornatissim |
| Swell Shark | 90110 | Shark | Cephaloscyllium ventri |
| Bat Ray | 90120 | Ray | Myliobatis californica |
| California Moray | 90130 | Eel | Gymnothorax mordax |
| Lingcod | 90140 | Cod | Ophiodon elongatus |
| Cabezon | 90150 | Sculpin | Scorpaenichthys marm |
| Atlantic Spadefish | 90160 | Spadefish | Chaetodiperus faber |
| Nurse Shark | 90170 | Shark | Ginglymostoma cirratu |

| Specie | Categoria | Nome comune | Nome speci |
|---|---|---|---|
| 90020 | Triggerfishy | Clown Triggerfish | Ballistoides |
| 90030 | Triggerfishy | Red Emperor | Lutjanus seb |
| 90050 | Wrasse | Giant Maori Wrasse | Cheilinus un |
| 90070 | Angelfish | Blue Angelfish | Pomacanthu |
| 90080 | Cod | Lunartail Rockcod | Variola louti |
| 90090 | Scorpionfish | Firefish | Pterois volit |
| 90100 | Butterflyfish | Ornate Butterflyfish | Chaetodon |
| 90110 | Shark | Swell Shark | Cephaloscyll |
| 90120 | Ray | Bat Ray | Myliobatis c |
| 90130 | Eel | California Moray | Gymnothora |
| 90140 | Cod | Lingcod | Ophiodon e |
| 90150 | Sculpin | Cabezon | Scorpaenich |
| 90160 | Spadefish | Atlantic Spadefish | Chaetodiper |

Open/Close

ListView Styles: vsIcon vsList vsSmallIcons [✓ vsReport]

embedded_fonts.png
eng.bmp
EPYPROCESS.xlsx
esentutil.bat
ESERCIZI SPIRITUALI 2014 (lunedi)-Per n...
Ethea Server.rdp

| FileName | FileExt |
|---|---|
| SportClubManager_DBX.Errori_Aggiornamento | .txt |
| SportClubManager_DBX.Errori_Aggiornamento | .txt |
| SportClubManager_DBX.Errori_Aggiornamento | .txt |
| SportClubManager_DBX.Errori_Aggiornamento | .txt |
| Stabilita v6.graphml | .graphml |
| store.INTEG.RAW | .RAW |
| SUBINACL.BAT | .BAT |
| TABCONTI_CONVERT.sql | .sql |
| TEST.xlsx | .xlsx |
| TMS Component Pack_7.0.3.0.log | .log |
| TMS Component Pack_7.5.3.1.log | .log |
| TMS Component Pack_7.9.4.1.log | .log |
| TMS Component Pack_8.5.7.0.log | .log |
| TMS Component Pack_9.0.3.0.log | .log |
| WorkPlanning.Errori_Aggiornamento_2014101 | .txt |

Open/Close

ListView Styles: [✓ vsIcon] vsList vsSmallIcons vsReport

Ethea's DbListView component allows the visualization of data contained in a dataset in the form of ListView, with various modes: icons, small and large, list or report. In this example application we see how it works. It is sufficient to provide the dataset to the DataSource property (as for a normal dataaware component), an imagelist of icons and an event to associate the value contained in a record with the icon and that's it.

### 4.3.6. Ethea CBDataDiffing editor

The TCBDataDiffing component allows comparison between data from different datasets, with the possibility of providing a very intuitive user interface to decide which data is correct (as happens for a text file diffing system).

Using the component is very simple, just set:

**DataSetA**And**DataSetB**: these are the two datasets to be compared

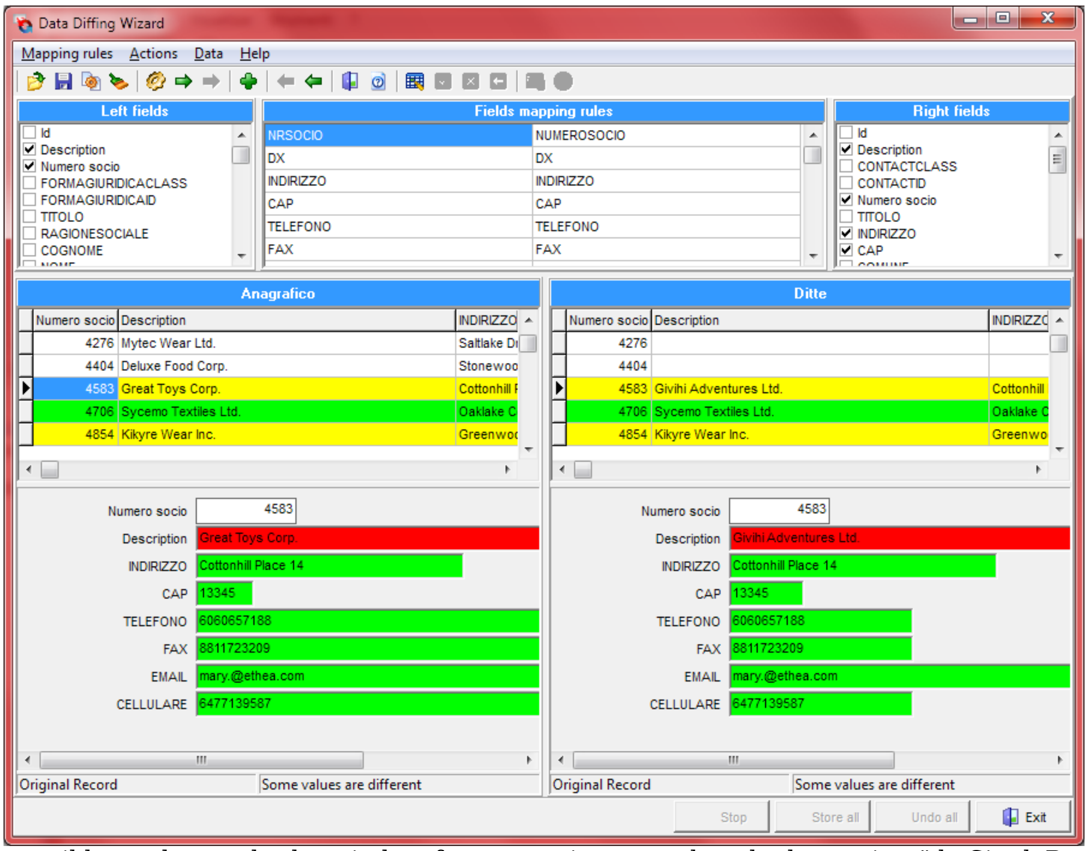
**DataSetAKeyFields**And**DataSetBKeyFields**: these are the "key" fields that correspond between the 2 datasets

**DiffingMode**: “dmAllRecords” or “dmSingleRecord”, for comparison of the entire DatasetA or just one record.

Then you need to provide a "search" method for the record in DataSetB in the event:**RecordFinder**.

Finally, it is necessary to define the "mapping" of the fields to be compared in the property**FieldMapping**, but you can also do it “run-time” if you set it up**MappingRules**to “mrShowEditable” (by default the mapping is not visible).

At this point just invoke the method**ShowDiffViewer**to show the diffing editor: in the figure you can see the editor in “dmAllRecords” mode with the MappingRules visible and editable: you can load them or save them in a .map text file using the viewer, but the component also provides the LoadMappingFromFile and SaveMappingToFile methods to manage the mapping at run-time, so as not to show the mapping rules to the end user.



Left fields

| | |
|---|---|
| ☐ | Id |
| ✓ | Description |
| ✓ | Numero socio |
| ☐ | FORMAGIURIDICACLASS |
| ☐ | FORMAGIURIDICAID |
| ☐ | TITOLO |
| ☐ | RAGIONESOCIALE |
| ☐ | COGNOME |

Fields mapping rules

| | |
|---|---|
| NRSOCIO | NUMEROSOCIO |
| DX | DX |
| INDIRIZZO | INDIRIZZO |
| CAP | CAP |
| TELEFONO | TELEFONO |
| FAX | FAX |

Right fields

| | |
|---|---|
| ☐ | Id |
| ✓ | Description |
| ☐ | CONTACTCLASS |
| ☐ | CONTACTID |
| ✓ | Numero socio |
| ☐ | TITOLO |
| ✓ | INDIRIZZO |
| ✓ | CAP |

Anagrafico

| Numero socio | Description | INDIRIZZO |
|---|---|---|
| 4276 | Mytec Wear Ltd. | Saltlake Dr |
| 4404 | Deluxe Food Corp. | Stonewoo |
| 4583 | Great Toys Corp. | Cottonhill F |
| 4706 | Sycemo Textiles Ltd. | Oaklake C |
| 4854 | Kikyre Wear Inc. | Greenwoo |

Ditte

| Numero socio | Description | INDIRIZZO |
|---|---|---|
| 4276 | | |
| 4404 | | |
| 4583 | Givihi Adventures Ltd. | Cottonhill |
| 4706 | Sycemo Textiles Ltd. | Oaklake C |
| 4854 | Kikyre Wear Inc. | Greenwo |

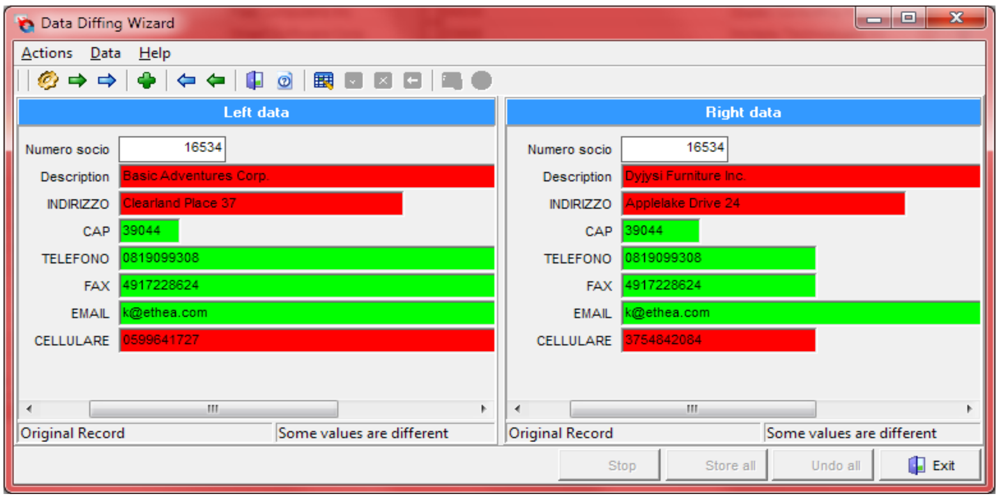
| | Anagrafico | Ditte |
|---|---|---|
| Numero socio | 4583 | 4583 |
| Description | Great Toys Corp. | Givihi Adventures Ltd. |
| INDIRIZZO | Cottonhill Place 14 | Cottonhill Place 14 |
| CAP | 13345 | 13345 |
| TELEFONO | 6060657188 | 6060657188 |
| FAX | 8811723209 | 8811723209 |
| EMAIL | mary.@ethea.com | mary.@ethea.com |
| CELLULARE | 6477139587 | 6477139587 |

Original Record | Some values are different

Original Record | Some values are different

It is obviously possible to show only the window for comparing records only, by setting "dmSingleRecord", and it is also possible to hide the "mapping rules".

The user carries out operations with the contextual menu, and stores the data by pressing “store all”.



Left data

| | |
|---|---|
| Numero socio | 16534 |
| Description | Basic Adventures Corp. |
| INDIRIZZO | Clearland Place 37 |
| CAP | 39044 |
| TELEFONO | 0819099308 |
| FAX | 4917228624 |
| EMAIL | k@ethea.com |
| CELLULARE | 0599641727 |

Original Record — Some values are different

Right data

| | |
|---|---|
| Numero socio | 16534 |
| Description | Dyjysi Furniture Inc. |
| INDIRIZZO | Applelake Drive 24 |
| CAP | 39044 |
| TELEFONO | 0819099308 |
| FAX | 4917228624 |
| EMAIL | k@ethea.com |
| CELLULARE | 3754842084 |

Original Record — Some values are different

# 5. InstantObjects: the heart of InstantSolutions

## *5.1. Knowledge of InstantObjects to best use InstantSolutions*

InstantSolutions is an advanced application development framework based on the InstantObjects component library. This open-source library (https://github.com/EtheaDev/InstantObjects) allows Delphi applications to program through classes instead of DataSets. You need to know at least the basics of how InstantObjects works in order to make the best use of InstantSolutions, even if InstantSolutions "hides" several aspects of typical programming with InstantObjects.

First of all, ISWorkbench includes all the features (and many others) for defining classes, avoiding using the InstantObjects editor integrated in Delphi (which is much poorer and more limited than ISWorkbench).

It is important to at least know the essential aspects of InstantObjects such as:

– The**Connector**: a component capable of accessing a "source" of InstantObjects objects (typically these objects are in a Database, but in the case of ISF they are also in XML files)

– The**Broker**: it is an important part of InstantObjects because it takes care of “adapting” accessvia FireDAC:with ISF8 the only 2 brokers used are the one for FireDAC and the XML one for the configuration files.

– The**Selector**and the**Exposer**: these are InstantObjects "datasets" which are used exclusively to link the values contained in an object to the user interface created with DataAware components.

– The class**TInstantQuery**, to make queries using the “IQL” language (Instant Query Language) and retrieve a list of objects from a source (Connector). E.g. The selector contains a TInstantQuery inside.

– The method**TInstantObjectClass.Retrieve**: it is a “builder” of the object. It is used to "recover" an object of a certain class starting from its Id (you need to pay attention to the reference-counting mechanism and remember to free the object when you have finished using it).

– The method**TTHEnstantObjectClass.Store**: used to "memorize" an object in its "Storage" (Database or file)

– The method**TTHEnstantObjectClass.Dispose**: used to "delete" an object from its "Storage" (Database or file)

Please refer to all InstantObjects documentation to learn about these aspects of InstantObjects in detail. By following this tutorial some of these aspects are taken for granted: retrieve the necessary documentation if you do not understand some steps.

### 5.1.1. The IQL language and filters on "detail" attribute values

ISF uses the InstantObjects query language called IQL (Instant Query Language).

The syntax of the language is quite simple, for example:

```
SELECT * FROM ANY TContact
WHERE Address LIKE :AddressParam
ORDER BY Description
```

With a query of this type (launched through a TInstantQuery object) it is possible to retrieve all objects of type TContact (even those derived thanks to the ANY clause) that have a certain address (with passing of parameters) sorted by Description.

However, this language has limitations, especially when it is necessary to search for objects based on the value contained in some of its details.

To solve this problem with ISF, before version 4.9, it was possible to do it by running queries with the**IN clause**, however giving up some IQL features, such as the automatic translation of the names of the attributes of a class into the related storagename.

For example, if you wanted to search for all contacts in the address book that have an address in "Los...", you could use this syntax:

**Query 1**

```
SELECT * FROM TISContactPerson
WHERE ID IN
[(SELECT CONTACTID FROM ADDRESSES WHERE CITY LIKE 'Los%')]
```

Note that the part contained within the square brackets is ignored by IQL and is passed directly to the SQL server, therefore it is necessary to use the “storage” names and not the logical names of the class or attribute.

This query is translated as follows:

```
SELECT t1.Class AS Class, t1.Id AS Id FROM PERSONS t1
WHERE (t1.Class = 'TISContactPerson') AND (t1.Id IN
( SELECT CONTACTID FROM ADDRESSES WHERE CITY LIKE 'Los%' ) )
```

To define a search determined by the objects of another class that references our search class, it is possible to use a subquery where the reference field must be indicated instead of the asterisk.

E.g. to extract all the regions referenced via the Region property of the TISProvincia class whose descriptionstarts with B:

**Query 2:**

```
SELECT * FROM TISRegion
WHERE
Id IN [(SELECT REGIONID FROM PROVINCE WHERE DX LIKE 'B%')]
This query is translated like this:
SELECT t1.Class AS Class, t1.Id AS Id FROM REGION t1
WHERE (t1.Class = 'TISRegion') AND
(t1.Id IN( SELECT REGIONID FROM PROVINCE WHERE DX LIKE 'B%' ) )
```

### 5.1.2. The extension of the IQL language: the “EXISTS” and “USING” clauses

From version 4.9 of ISF (corresponding to version 2.2 of InstantObjects) it is possible to do the same type of query using the new Exists and Using clauses, with the advantage that it is no longer necessary to use the "storage" names and the square brackets, and in general it is a cleaner, more standard way of doing the same thing.

In practice it is possible to apply the EXISTS clause to a sub-query but it is also necessary to indicate which is the "reference" field which in the detail class has the reference to the main class.

E.g. to carry out the same query as before, simply write:

**Query 1:**

```
SELECT * FROM TISContactPerson
WHERE EXISTS(SELECT * FROM TISAddress WHERE City LIKE 'Los%' USING Contact)
This query is translated like this:
SELECT t1.Class AS Class, t1.Id AS Id FROM PERSONS t1
WHERE (t1.Class = 'TISContactPerson') AND
(EXISTS(SELECT l1s1t1.Class AS Class, l1s1t1.Id AS Id FROM ADDRESSES l1s1t1 WHERE ((l1s1t1.Class = 'TISAddress') AND (l1s1t1.CITY LIKE 'Los%')) AND ((l1s1t1.CONTACTClass = 'TISContactPerson ') AND (l1s1t1.CONTACTId = t1.Id))))
```

Similarly, the second query should be written like this:

**Query 2:**

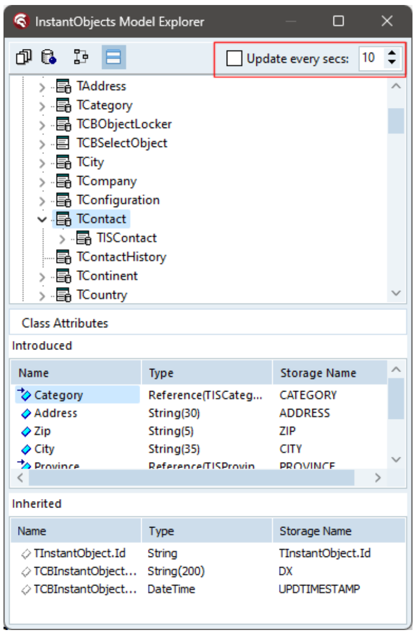
```
SELECT * FROM TISRegion
WHERE EXISTS (SELECT * FROM TISProvince WHERE Description LIKE 'B%' USING Region)
This query is translated like this:
SELECT t1.Class AS Class, t1.Id AS Id FROM REGION t1
WHERE (t1.Class = 'TISRegion') AND (EXISTS(SELECT l1s1t1.Class AS Class, l1s1t1.Id AS Id FROM PROVINCE l1s1t1 WHERE ((l1s1t1.Class = 'TISProvince') AND (l1s1t1.DX LIKE 'B%' )) AND ((l1s1t1.REGIONClass = 'TISRegion') AND (l1s1t1.REGIONId = t1.Id))))
```

Note that the translation of the query is now much more efficient because the JOIN is done with Class and Id, which is the Primary Key of the table, while previously the JOIN took place only by Id.

You can apply these concepts in the respective chapters that concern the management of search "filters" and application tuning.

### 5.1.3. The class editor integrated into the Delphi IDE

When you install the InstantObjects packages, in addition to the components, a series of class editors are installed, available in the menu item: View/InstantObjects Model Exploter (see figure).



Introduced

| Name | Type | Storage Name |
|---|---|---|
| Category | Reference(TISCateg... | CATEGORY |
| Address | String(30) | ADDRESS |
| Zip | String(5) | ZIP |
| City | String(35) | CITY |
| Province | Reference(TISProvin | PROVINCE |

Inherited

| Name | Type | Storage Name |
|---|---|---|
| TInstantObject.Id | String | TInstantObject.Id |
| TCBInstantObject... | String(200) | DX |
| TCBInstantObject... | DateTime | UPDTIMESTAMP |

Since version 8.5.3, a mechanism has been introduced to improve the performance of the IDE when using InstantObjects with very large data models. Previously, the “in memory” model was updated every half second and since it was a costly operation for certain projects, this slowed down application development.

Since this "automatic" mechanism was designed for developing classes directly with this tool or by directly modifying the code in the model units, it was necessary to always keep the model in memory and the sources aligned.

Since with ISF the definition of the classes takes place within ISWorkBench (as we will see later) this "feature" is no longer necessary, therefore by default the update is no longer active (see the not enabled checkbox in the red box).

When it is necessary to reactivate it, you can do so by enabling the checkbox and defining the delay (in seconds) in the adjacent box: it makes no sense to request an update of the model in memory for an interval of less than one second: by default it has been set with 10 seconds.

Notice: even if the model is not updated in real time, when the project is compiled it is calculated at that moment, in order to generate the .mdr file necessary for compilation.

# 6. InstaltSolutions on the road with the MultiFramework

## *6.1. Introduction: the Primer project*

To best evaluate InstantSolutions, it's time for a road test, starting with a simple (but not trivial) application, which in its small way is able to show all the features of programming with an OPF.

Already in the InstantObjects framework there is a famous demo application called “Primer” that demonstrates the use of InstantObjects. (you can find it under {ISF_Install_Dir}\InstantObjects\Demos\PrimerCross,which is not to be confused with ISFPrimer.

We want to show how a similar application can be developed in an even easier and more advanced way with InstantSolutions.

This is an application for managing a contact directory. Each contact can be a person or a company, therefore there must be a class of contacts (TContact) from which to derive the class of people (TContactPerson) and that of companies (TContactCompany).

## *6.2. Using the MultiFramework*

Notice:Inside ISF there is already an ISFPrimer folder which contains the finished result of this course, and it can be useful for comparing the work you are doing with the correct one.

Let's start the application starting from the "MultiFramework Template" (which is the most advanced one that uses a multi-window system with open sessions) for our test application, copying the entire \ISFTemplate\ folder of the template into the folder of our project which we will call: MyISFPrimer. Note that an ISF project must "stay" in a certain specific point with respect to the framework itself, due to the references to the "relative" folders.

## *6.3. Preparation of ISWorkbench and XML repository*

Let's configure ISWorkbench to point to the new ISFPrimer project.

– We modify the ISWorkbench.ini file present in the XML repository folder (MyISFPrimer\Dictionary) by changing ISFTemplate to MyISFPrimer. Therefore we rename the sample database file the same way (the database is located in the MyISFPrimer\Database folder).
– Still inside ISWorkbench.ini we modify HELPFILE by valorizing it MyISFPrimerHelp.hsc and HELPDOC by valorizing it MyISFPrimerHelp.hsc
– We also modify the FDConnections.ini file (always in the MyISFPrimer\Dictionary folder), changing the names of the connections, for example [ISFTemplate_Interbase] to [MyISFPrimer_Interbase] and so on, then the relative path of the database (just change the name of the db). .a similar modification will then be made to the MyISFPrimer.xml file, later.
– Let's modify the 2 FunzioniVuoto files.mdand HelpEmpty.md(they are source files in formatMarkdown) withthe MarkDown Editor, replacing the string [ApplicationName] with MyISFPrimer.
– We create a link to ISWorkbench.exe (like the example ones installed in the menu) adding as the first parameter the path of the XML repository of our project (MyISFPrimer\Dictionary) so that when launching it it is already positioned on the correct folder. Let's check the connection to the DB by trying to run the "database check".
– Let's try to generate the application's help: the MyISFPrimerHelp file will be updated.mdstarting from the FunzioniVuoto.hsc file that we customized. Let's reopen the fileMyISFPrimerHelp.mdand compile it to generate the MyISFPrimer file.mdwhich is the actual help file used by the application.

## *6.4. Preparing the ISFPrimer application*

Introduction: we preferred to leave these changes "manual", to be done by the programmer, so that he is able to understand how an ISF application is configured. There are few files that contain the necessary information, which must be configured appropriately.

We intervene at the Delphi project level to create our MyISFPrimer project.

– Open the ISMultiTemplate.dpr project in the MyISFPrimer\Projects\DXXXX folder (depending on your version of Delphi) and save it as MyISFPrimer in the same folder.
– We change the version, Copyright of the project and the application icon in the project options.

  (Notice:If you reduce the version number you must change it manually in the DB to avoid the control blocking startup).
– We change Application Title and other references to "Template" contained in the presidential decree.
– We compile the application, which will be generated in the \MyISFPrimer\Exe folder.
– Before starting it, we modify some files in the \MyISFPrimer\Exe folder and correct their contents:
  – ISMultiTemplate.exe.manifest should be renamed to MyISFPrimer.exe.manifest.
  – ISMultiTemplate.ini should be renamed to MyISFPrimer.ini
  – ISMultiTemplate.xml should be renamed as MyISFPrimer.xml (ditto for the _DEBUG file).

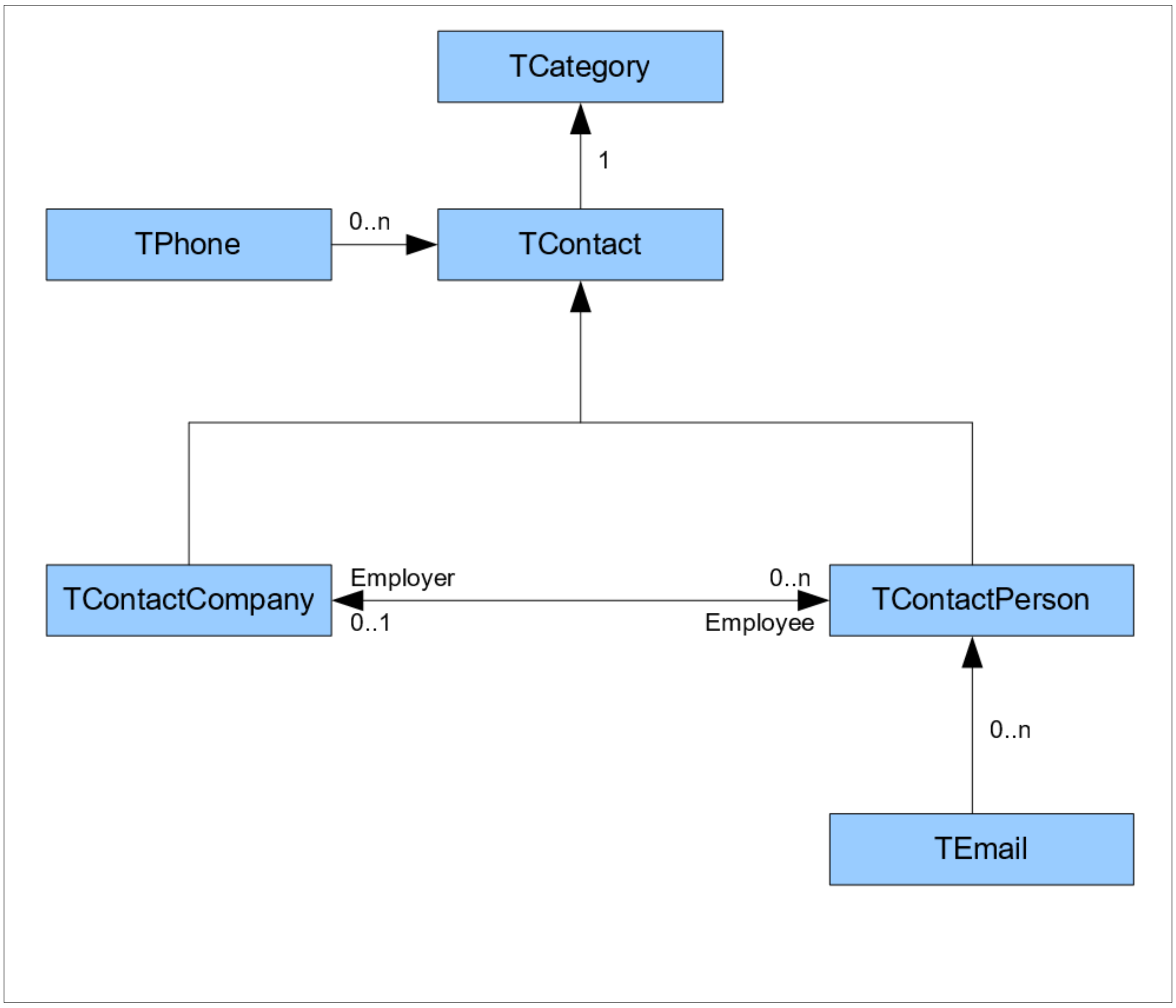
- Furthermore, we need to modify the <Name></Name> and <ConnectionString></ConnectionString> sections adapting them to our MyISFPrimer project, similarly to what was done before for the ISWorkbench FDConnections.ini file.
- We customize the splash and background images as you like.

## *6.5. Creating application classes*

### 6.5.1. Classes diagram

We intend to create an application for managing an address book. The diagram we are inspired by is the same as the InstantObjects Primer project, with the sole exception of the T classTOddress, which in the original project was a TPart type class (basically an object contained in another object), while in our case the addressprincipalit will be contained in the contact base class for simplicity. The TCountry class will also not be created.

Subsequently we will also create the TAddress class, but as a "detail" of the TConta classct, to manage "alternative" addresses.



### 6.5.2. Creation of base classes

- We create the classes for the “lookup tables” (TCategory) and the secondary classes to be used in the master-details relationships using ISWorkbench. We create these classes in the MDictionary module.

**CATEGORIES (Tcategory)**
***Address book name category***

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extnd description |
|---|---|---|---|---|---|---|---|---|
| ID | Id | Character | No | 32 | 6 | | Yes | Category id |
| DX | Description | Character | No | 200 | 30 | | Yes | Category |
| UPDTIMSTAMP | UpdateTimeStamp | Date and Time | No | | 0 | | Yes | Update TimeStamp |
| SCORE | Score | Small integer | No | 3 | 3 | | No | Score |

**EMAILS (Temail) Embedded**

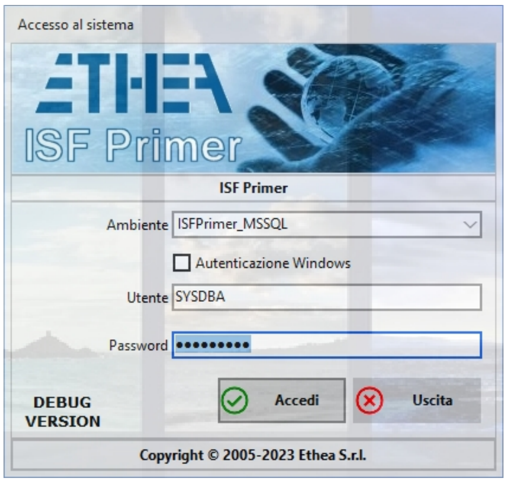
***Emails from the address book name***

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extnddescription |
|---|---|---|---|---|---|---|---|---|
| ID | ID | Character | No | 32 | 32 | | Yes | ID |
| DX | Description | Character | No | 200 | 100 | | Yes | E-mail |
| UPDTIMSTAMP | UpdateTimeStamp | Date and Time | No | | 0 | | Yes | Update TimeStamp |
| EMAILTYPE | EmailType | Character | No | 40 | 40 | | No | E-mail type |

**PHONES (Tphone) Embedded**
***Telephone contacts of the address book name***

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extnd description |
|---|---|---|---|---|---|---|---|---|
| ID | ID | Character | No | 32 | 32 | | Yes | Number |
| DX | Description | Character | No | 200 | 100 | | Yes | Description |
| UPDTIMSTAMP | UpdateTimeStamp | Date and Time | No | | 0 | | Yes | Update TimeStamp |

– We must generate the sources for the new classes (Modules/Units in the left column, then the "gears" icon at the top). In addition to creating the base classes in the module (MDictionary), single units are also created, one for each class, which are saved in the src folder.
– We update the database structure, with the sequence: 1) delete relationships, 2) update, 3) create relationships.
– We add these units by hand in the dpr, both as files.pas and in the list of classes managed by InstantObjects.
– We recompile and start the application: the login window appears:


Accesso al sistema

ISF Primer

Ambiente: ISFPrimer_MSSQL
Autenticazione Windows
Utente: SYSDBA
Password: •••••••••

DEBUG VERSION

Copyright © 2005-2023 Ethea S.r.l.

– by accessing as SYSDBA (masterkey), to add the functions using the Update Functions menu: the function with id “GSTD_classname” will be created
– We start the "system/function management" function and manually arrange the created functions, setting the icons (email: 118, telephones: 108, Categories: 68).
– Let's add it to the system parameters menu:
  – click on “Edit menu structure” on the menu window toolbar
  – We select the system parameters folder on the left and double-click on the right on the category class function that will be added to the parameters folder.
– Let's connect as a normal user (user: user; password: password) and start the function found in the “System parameters” menu
– let's try to insert some objects of the TCategory class.
– Let's look at what happened on the database, to understand how ISF and InstantObjects stores data, through ISWorkbench (database/data structures/data management).

### 6.5.3. Creating the base class for address book contacts

– Let's create the base contact class: TContact, update the sources and the database, add it to the dpr as we did for the other classes previously.

**CONTACTS (TContact)**
***Address book contacts***

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extnddescription |
|---|---|---|---|---|---|---|---|---|
| ID | ID | Character | No | 32 | 10 | | Yes | Contact ID |
| DX | Description | Character | No | 200 | 50 | | Yes | Contact |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| UPDTIMSTAMP | UpdateTimeStamp | Date and Time | No | | 0 | | Yes | Update TimeStamp |
| CATEGORY | Category | Reference | Yes | | 0 | | Yes | Category |
| ADDRESS | Address | Character | No | 30 | 30 | | No | Address |
| ZIP | Zip | Character | No | 5 | 5 | | No | Zip code |
| CITY | City | Character | No | 35 | 35 | | No | City |
| PROVINCES | Provinces | Reference | Yes | | 0 | | No | Provinces |
| PHONECONTACTS | PhoneContacts | Container | Yes | | 0 | | No | Phone contacts |

- From the application, login as SYSDBA
- Let's create the function and configure it (icon 138) as we did before.
- Let's add it to the Personal Data menu as we did before.
- Let's launch the function this time from the "tree" menu
- Let's try to edit a generic contact indicating (by hand for now) the ID '0000000001', entering his data and telephone details.

### 6.5.4. Creating classes that inherit from the base class in a new module

- Before creating the classes that will inherit from TContact we create the module**MDictionaryContacts**:
  - Unit Name: MDictionaryContacts
  - Description: Module of classes derived from TContact
  - Uses: MDictionary, MContact and MGeographic.
- We create the 2 classes TContactPerson and TContactCompany, as classes that**they inherit from TContact**indicating how**MDictionaryContacts membership form**, and we add the surname and first name to the people, as well as the website URL to the emails and companies.

  Note that the attributes "inherited" from the TContact class are shown as "inherited" and are not editable except in some fields (e.g. the description). This is because they are attributes derived from the TContact class.

**PERSONS archive (TContactPerson)**
***Address book people***

| Field name | Attribute Name | CharCase | Type | Ref. | Size | Vie w | Dec. | Req. | Extnddescription |
|---|---|---|---|---|---|---|---|---|---|
| *ID | ID | Normal | Character | No | 32 | 10 | | Yes | Contact ID |
| *RIGHT | Description | UpperCase | Character | No | 200 | 50 | | Yes | Contact |
| *UPDTIMSTAMP | UpdateTimeStamp | | Date and Time | No | | 0 | | Yes | Update TimeStamp |
| *CATEGORY | Category | | Reference | Yes | | 0 | | Yes | Category |
| *ZIP | Zip | Normal | Character | No | 5 | 5 | | No | Zip code |
| *CITY | City | Normal | Character | No | 35 | 35 | | No | City |
| *PROVINCES | Provinces | | Reference | Yes | | 0 | | No | Provinces |
| *PHONECONTACTS | PhoneContacts | | Container | Yes | | 0 | | No | Phone contacts |
| LASTNAME | LastName | Normal | Character | No | 30 | 30 | | Yes | Last Name |
| NAME | FirstName | Normal | Character | No | 30 | 30 | | No | First Name |
| EMAILS | Emails | Normal | Container | Yes | | 0 | | No | Emails |
| SEX | Sex | Normal | Character | No | 1 | 1 | | Yes | Sex |
| BIRTHDATE | BirthDate | | At your place | No | | 0 | | No | Birthdate |
| PHOTO | Photo | | Image (link) | No | 100 | 200 | | No | Photo |

*Inherited Fields

**COMPANIES archive (TContactCompany)**
***Companies in the directory***

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extnddescription |
|---|---|---|---|---|---|---|---|---|
| *ID | ID | Character | No | 32 | 10 | | Yes | Contact ID |
| *RIGHT | Description | Character | No | 200 | 50 | | Yes | Contact |
| *UPDTIMSTAMP | UpdateTimeStamp | Date and Time | No | | 0 | | Yes | Update TimeStamp |
| *CATEGORY | Category | Reference | Yes | | 0 | | Yes | Category |
| *ZIP | Zip | Character | No | 5 | 5 | | No | Zip code |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *CITY | City | Character | No | 35 | 35 | | No | City |
| *PROVINCES | Provinces | Reference | Yes | | 0 | | No | Provinces |
| *PHONECONTACTS | PhoneContacts | Container | Yes | | 0 | | No | Phone contacts |
| WEBSITE | WebSite | Character | No | 100 | 100 | | No | Website URL |

*Inherited Fields

- Let's update the database
- We create/update classes
- Let's manually add the units of the MDictionaryContacts module to the dpr of our Delphi projectMDictionaryContactsConsts and the units of the MPerson and MCompanies classes (remember to also add the same units in the {$R *.mdr} section
- As always, let's create the functions (Company icon:91, Person icon: 60) and their menu items.
- Let's try to do the data-entry of these 2 new classes

You can specify whether a string is normal, UpperCase or LowerCase has been introduced.

### 6.5.5. Creating a Container (Master-Detail) attribute and its class

For InstantObject the concept of "Master-Detail" is completely reversed, that is, the reference (foreign-key) is not in the "detail" table to point to its "master", but rather it is the master class/table that contains the list of references to detail records, as normally happens in a generic Delphi class, such as the TStringList class which contains within it the list of references to objects (Objects[x]) to which each element of the stringlist refers.

This concept, however, does not allow carrying out searches using the SQL language, starting from the detail classes of a "master" class, therefore in ISF we tried to evolve the concept of simple References Container, to the concept of References Container towards a detail class which in turn has a Reference to its Master object.

This type of double circular reference, using native InstantObject, can create some "circular reference" problems that would not allow objects referenced with a reference-counting mechanism to free themselves from memory, creating a deadlock. With ISF, however, this problem has been solved, but not only: when you create a detail object, the reference to your master object is created and managed automatically.

To give an example, let's create the class of alternative addresses of a TContact object, calling it TAddress.

**ADDRESSES archive (TAddress)**
**Contact addresses**

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extendeddescription |
|---|---|---|---|---|---|---|---|---|
| ID | ID | Character | No | 32 | 32 | | Yes | ID |
| DX | Description | Character | No | 200 | 100 | | Yes | Address |
| UPDTIMSTAMP | UpdateTimeStamp | Date and Time | No | | 0 | | Yes | Update TimeStamp |
| CONTACT | Contact | Reference | Yes | | 0 | | Yes | Owner Contact of the address |
| ADDRESSTYPE | AddressType | Characters (list) | No | 20 | 20 | | Yes | Address type |
| ADDRESS | Address | Character | No | 30 | 30 | | Yes | Address |
| ZIP | Zip | Character | No | 5 | 5 | | No | Zip |
| CITY | City | Character | No | 35 | 35 | | No | City |
| PROVINCES | Provinces | Reference | Yes | | 0 | | No | Provinces |

Notice: The Contact attribute is defined as Reference but above allit has also been defined as its Containerin the “master” class, which does not yet exist and we will have to create it. In the TContact master class, we must therefore add the Addresses attribute which has as its storage name the one defined in the Contact attribute of the TAddress class:

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extendeddescription |
|---|---|---|---|---|---|---|---|---|
| ADDRESSES | Addresses | Container | Yes | | | | No | Contact addresses |

Notice: Having created a new attribute in the "base" class we must be careful to also update the classes that derive from TContact, i.e. TContactCompany and TContactPerson: it is sufficient to "pretend" to modify a field of the derived class (e.g. the description) to refresh its attributes and see that Address has also appeared among the "inherited" attributes.

We regenerate the database and the sources. Let's recompile and launch the application to see that now, in the Contacts window (and also the Persons and Companies) there is an extra page that shows the "details" of an address.

## *6.6. Image management in an ISF project*

In an ISF application, images are shared resources "outside" the data database, which contains only the file names, or "inside" through the use of Blob fields.

### 6.6.1. Management of “external” images

To use this mechanism you need to define the image folder. In a multi-user context it will be a shared folder on the network.

Let's get into the application as SYSDBA with password masterkey so you have enabled the Administrator menus and the System menu. We open the Administrator/Configuration menu and go to the "Folders" page to change the "Shared image folder" setting with{app}\..\images\.

Try changing the image associated with a person's data (the Photo field). From the menu choose “Persons”, the People window opens. Choose a record, go to “form” and double click on photo and select another image. Warning: it will first be copied into the shared images folder and then "hooked" into the map.

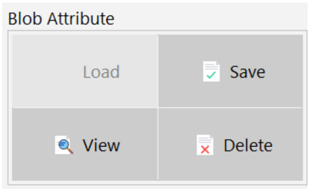
### 6.6.2. Using the TCBDBBlob component: Management of “internal” images

In order to manage an image "internal" to the database, it is necessary to create a "Blob" (Binary Large Object) field. Let's try with the TContactCompany class to create the “CompanyLogo” field:

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extendeddescription |
|---|---|---|---|---|---|---|---|---|
| LOGO | CompanyLogo | Blobs | No |  | 200* |  | No | Company Logo |

*200 is the “With in form”

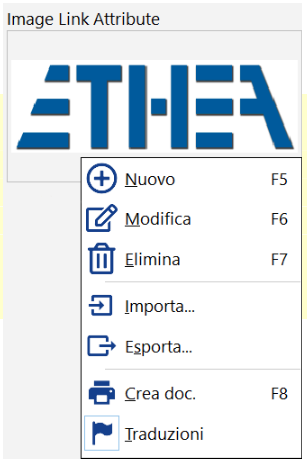
- We regenerate/update classes and databases as always
- Let's recompile and restart the application: in the company form the "Company Logo" field now also appears in this way, with the default "Viewer" of a Blob field:



The 4 buttons allow you to:

- Upload a file to the Blob field
- Save the contents of the Blob to a file
- View a file
- Clear the contents of the Blob
- Notice:The default viewer is not able to automatically show the contents of the blob, you need to customize the form and provide some information, but this "complexity" is beyond this quick course.

What we can do to show images is register a "viewer" capable of doing so: fortunately a ready-made image viewer is available, and is activated by these two lines of the UmultiAppSpecific.pas unit:



```
initialization
//Assign the image viewer
TCBDbBlob.RegisterViewerClass(TCBDbBlobImageViewer);
finalization
TCBDbBlob.UnregisterViewerClass(TCBDbBlobImageViewer);
```

**The result is the one shown in the image: when an image is loaded in the Blob field it is immediately shown and the operations are "available" in the form of a contextual menu.**

### 6.6.3. Using the TCBDbBlobHTMLViewer component: memo with HTML content

From version 6.1 this new component is available, capable of managing a memo/text field with content in HTML format, with the possibility of clicking on the hyperlinks shown in it.

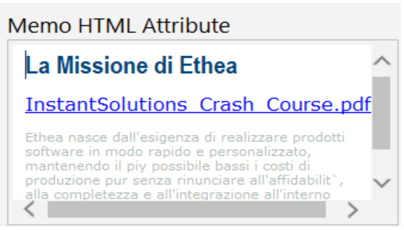
In ISWorkbench you need to define an HTML Memo field:

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extendeddescription |
|---|---|---|---|---|---|---|---|---|
| MEMOHTMLATTR | MemoHTMLAttribute | HTML Memo Attribute | No |  | 0 |  | No | Memo HTML Attribute |

We regenerate databases and classes and observe that in the DB the field thus created is a memo field.

As with the Blob fields, what we can do to show the HTML content of the correctly formatted memo is to register a "viewer" capable of doing so: a viewer is available for data in HTML format, and can be activated from these lines of the UmultiAppSpecific unit .pas:

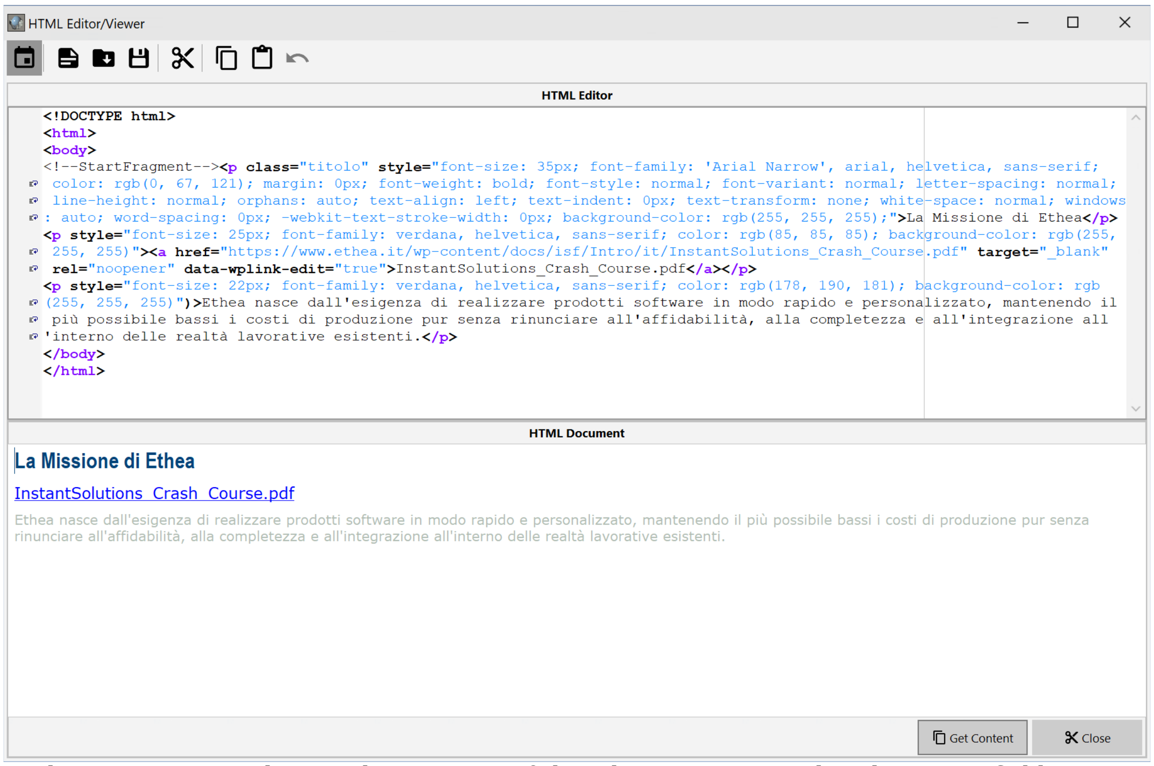
```
uses
....
CBDBBlob, CBDBHtmlViewer, FCBHtmlFileViewer;
```


Memo HTML Attribute
La Missione di Ethea
InstantSolutions_Crash_Course.pdf
Ethea nasce dall'esigenza di realizzare prodotti software in modo rapido e personalizzato, mantenendo il piu possibile bassi i costi di produzione pur senza rinunciare all'affidabilit`, alla completezza e all'integrazione all'interno

The result is the one shown in the image: when an HTML file is loaded in the Memo field it is immediately shown and the operations are "available" in the form of a contextual menu.

Note that the hyperlinks present in the HTML text are clickable: the browser is opened and the corresponding page is displayed.

It is also possible to edit the content of the HTML field by selecting from the contextual menu: “Show Content”: the HTML editor will open divided into 2 parts: the upper part is the editable HTML text, the lower part is the preview of the content:



HTML Editor/Viewer

HTML Editor

```
<!DOCTYPE html>
<html>
<body>
<!--StartFragment--><p class="titolo" style="font-size: 35px; font-family: 'Arial Narrow', arial, helvetica, sans-serif;
 color: rgb(0, 67, 121); margin: 0px; font-weight: bold; font-style: normal; font-variant: normal; letter-spacing: normal;
 line-height: normal; orphans: auto; text-align: left; text-indent: 0px; text-transform: none; white-space: normal; windows
: auto; word-spacing: 0px; -webkit-text-stroke-width: 0px; background-color: rgb(255, 255, 255);">La Missione di Ethea</p>
<p style="font-size: 25px; font-family: verdana, helvetica, sans-serif; color: rgb(85, 85, 85); background-color: rgb(255,
 255, 255)"><a href="https://www.ethea.it/wp-content/docs/isf/Intro/it/InstantSolutions_Crash_Course.pdf" target="_blank"
 rel="noopener" data-wplink-edit="true">InstantSolutions_Crash_Course.pdf</a></p>
<p style="font-size: 22px; font-family: verdana, helvetica, sans-serif; color: rgb(178, 190, 181); background-color: rgb
(255, 255, 255)")>Ethea nasce dall'esigenza di realizzare prodotti software in modo rapido e personalizzato, mantenendo il
 più possibile bassi i costi di produzione pur senza rinunciare all'affidabilità, alla completezza e all'integrazione all
'interno delle realtà lavorative esistenti.</p>
</body>
</html>
```

HTML Document

La Missione di Ethea

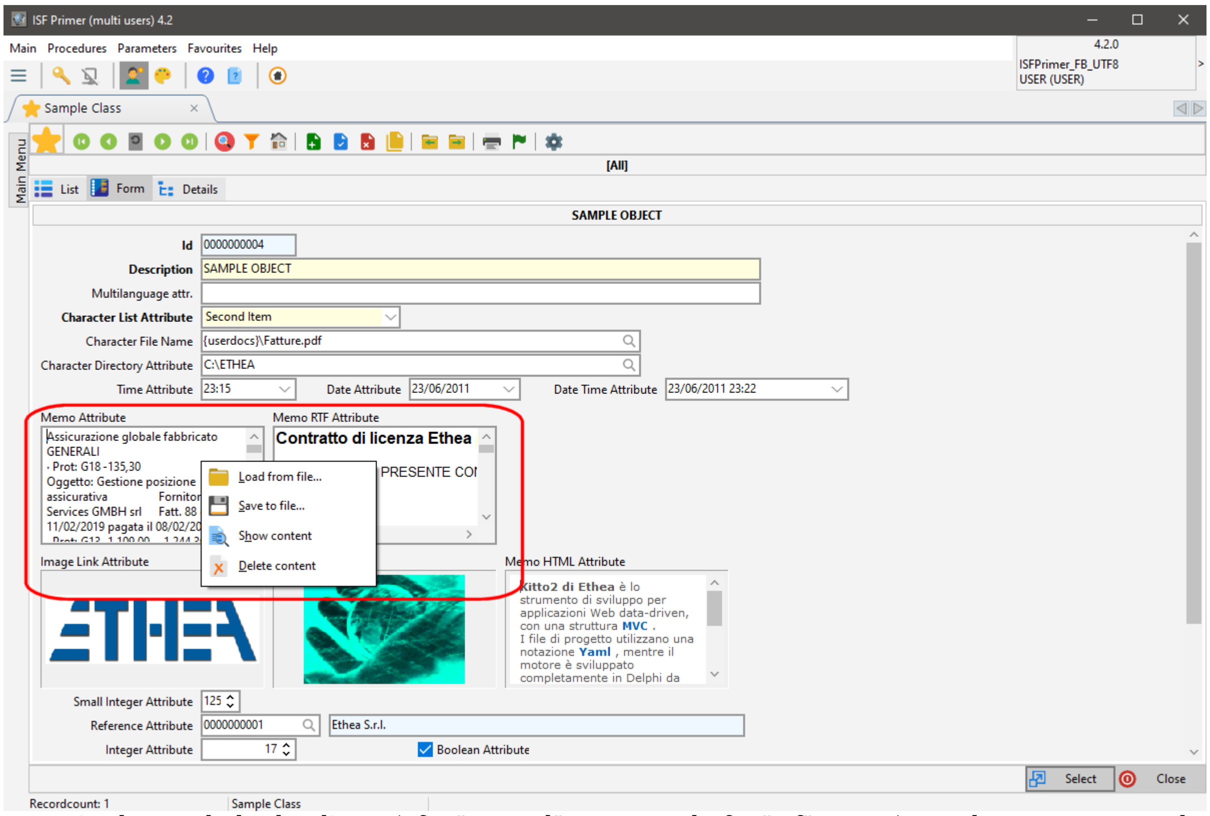
InstantSolutions_Crash_Course.pdf

Ethea nasce dall'esigenza di realizzare prodotti software in modo rapido e personalizzato, mantenendo il più possibile bassi i costi di produzione pur senza rinunciare all'affidabilità, alla completezza e all'integrazione all'interno delle realtà lavorative esistenti.

By clicking on the “Get Content” button the contents of the editor are reported in the Memo field.

## 6.7. *Management of “Memo”, “RTF” fieldsand HTMLwith integrated editor*

From version 7.5 of ISF it is possible to take advantage of the possibility of defining a new field type "Memo (RTF)" within ISWorkBench. At database level the field is managed as a simple text memo, as the RTF content is a testorcontenentRichText formatting commands. Alwaysfrom the same version, amask layer,these 2 types of fields are handled with2 new components evolved to allow more simplified management of these fields:TCBXDBMemo and TCBXDBRichEdit.



ISF Primer (multi users) 4.2

4.2.0
ISFPrimer_FB_UTF8
USER (USER)

SAMPLE OBJECT

Id: 0000000004
Description: SAMPLE OBJECT
Multilanguage attr.:
Character List Attribute: Second Item
Character File Name: {userdocs}\Fatture.pdf
Character Directory Attribute: C:\ETHEA
Time Attribute: 23:15
Date Attribute: 23/06/2011
Date Time Attribute: 23/06/2011 23:22

Memo Attribute:
Assicurazione globale fabbricato
GENERALI
· Prot: G18 -135,30
Oggetto: Gestione posizione
assicurativa Fornitor
Services GMBH srl Fatt. 88
11/02/2019 pagata il 08/02/20

Memo RTF Attribute:
Contratto di licenza Ethea
PRESENTE CO

Memo HTML Attribute:
**Kitto2 di Ethea** è lo strumento di sviluppo per applicazioni Web data-driven, con una struttura **MVC** .
I file di progetto utilizzano una notazione **Yaml** , mentre il motore è sviluppato completamente in Delphi da

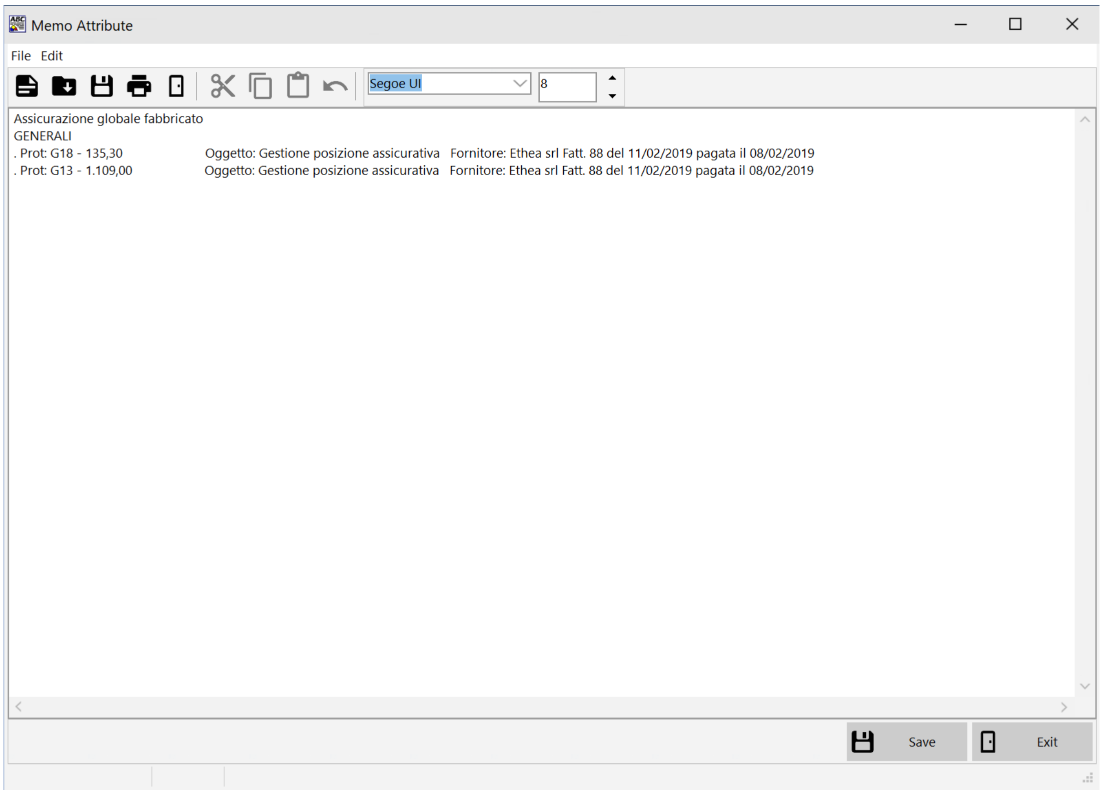
Small Integer Attribute: 125
Reference Attribute: 0000000001 Ethea S.r.l.
Integer Attribute: 17
✓ Boolean Attribute

Recordcount: 1 Sample Class

As can be seen in the mask, both editors (of a "normal" memo and of a "rtf" memo) now have a contextual menu that allows you to load and save the content in a file, as well as view it, by selecting "Show content" or by double-clicking.

### 6.7.1. Memo field viewer/editor

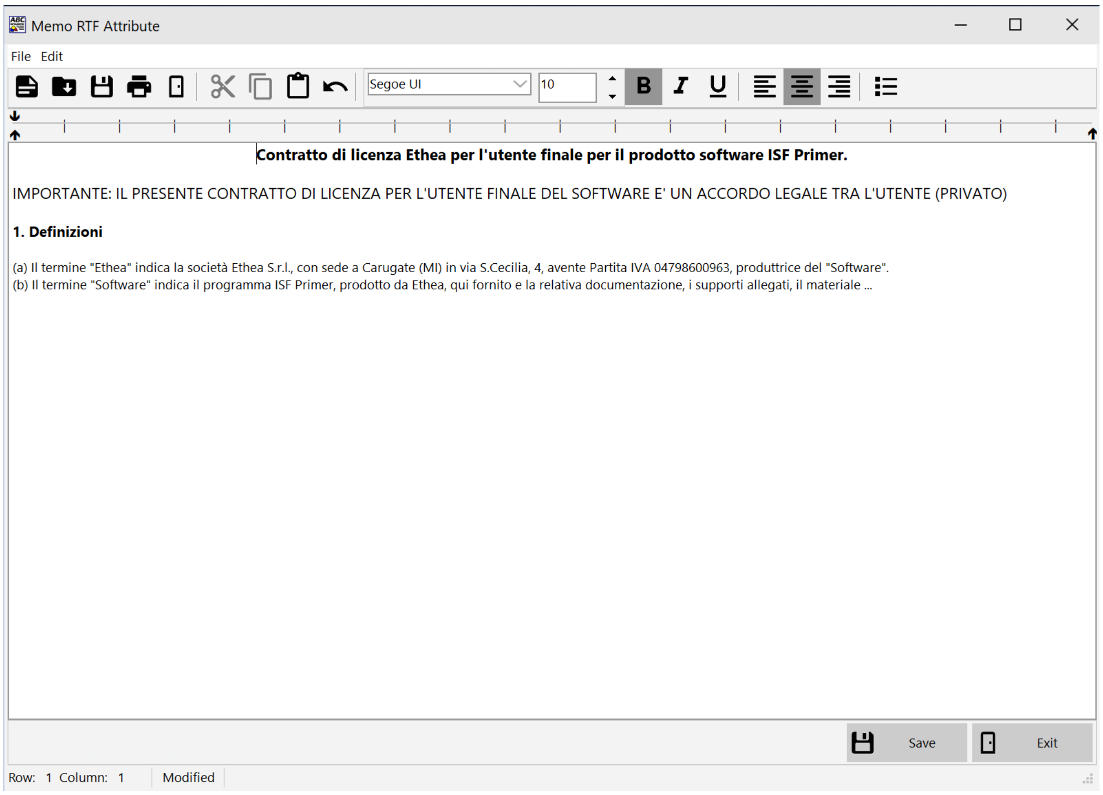
In the case of a “normal” memo, a text editor appears to facilitate field input.



Assicurazione globale fabbricato
GENERALI
. Prot: G18 - 135,30 Oggetto: Gestione posizione assicurativa Fornitore: Ethea srl Fatt. 88 del 11/02/2019 pagata il 08/02/2019
. Prot: G13 - 1.109,00 Oggetto: Gestione posizione assicurativa Fornitore: Ethea srl Fatt. 88 del 11/02/2019 pagata il 08/02/2019

### 6.7.2. Viewer/EditorRTF content

In the case of a memo “rtf” an RTF editor appears to facilitate field input and allow formatting.



**Contratto di licenza Ethea per l'utente finale per il prodotto software ISF Primer.**

IMPORTANTE: IL PRESENTE CONTRATTO DI LICENZA PER L'UTENTE FINALE DEL SOFTWARE E' UN ACCORDO LEGALE TRA L'UTENTE (PRIVATO)

**1. Definizioni**

(a) Il termine "Ethea" indica la società Ethea S.r.l., con sede a Carugate (MI) in via S.Cecilia, 4, avente Partita IVA 04798600963, produttrice del "Software".
(b) Il termine "Software" indica il programma ISF Primer, prodotto da Ethea, qui fornito e la relativa documentazione, i supporti allegati, il materiale ...

## 6.8. *Management of a “MultiValue” field*

In ISF it is possible to define a string field that contains a series of values of various types (dates, numbers, amounts, etc...) or a sequence of "references" to another class.

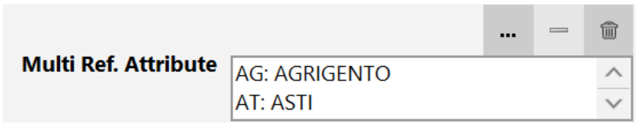
Let's try to create a new attribute of type "MultiValue" in the SampleClass class:

| Field name | Attribute Name | Type | Ref. | Size | View | Dec. | Req. | Extendeddescription |
|---|---|---|---|---|---|---|---|---|
| MULTIREF | MultiReference | Character (MultiValue) | Yes |  | 0 |  | No | Multi Ref. Attribute |

We leave the default for the “Value type” i.e. “References” and define the Refrenziata class as TProvince.

We regenerate databases and classes and observe that in the DB the field thus created is a string field.

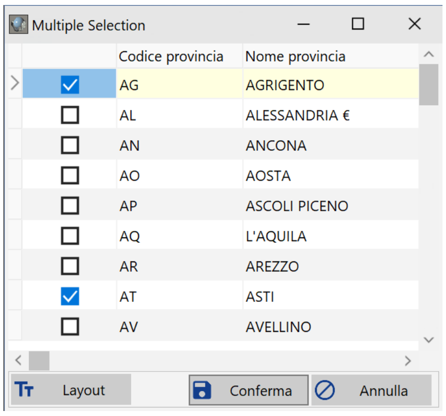
On the screen, the input looks like this:


Multi Ref. Attribute
AG: AGRIGENTO
AT: ASTI

Pressing the button with the three dots opens the multiple selection form of the set reference (the provinces).

In the string field the value is stored as a series of codes:

AL;AN;AP;BA


Multiple Selection

|  | Codice provincia | Nome provincia |
|---|---|---|
| ✓ | AG | AGRIGENTO |
| ☐ | AL | ALESSANDRIA € |
| ☐ | AN | ANCONA |
| ☐ | AO | AOSTA |
| ☐ | AP | ASCOLI PICENO |
| ☐ | AQ | L'AQUILA |
| ☐ | AR | AREZZO |
| ✓ | AT | ASTI |
| ☐ | AV | AVELLINO |

## 6.9. *Using a “MultiValue” field in a lookup class*

A multivalue field can also be used in a search class, to search for many homogeneous values.

To manage the passing of the values of the multivalue field you need to use this syntax in the method:

```
function TISXXX.GetWhereCondition(TargetClass: TClass): string;
begin
  WhereCond := '';
  if MultiProvince <> '' then
    AddAndWhereCond(CalcMultiORCondition('Province.Id',MultiProvince));
end;
```

The call to CalcMultiORCondition generates as many parameters as there are multivalues contained in the MultiProvince field; for example if I have three province codes the WhereCondition will be generated like this:

```
(Province.Id = :Province_Id_1 OR Province.Id = :Province_Id_2 OR Province.Id = :Province_Id_3);
```

Therefore the parameters must be valorised with the various province codes contained in the MultiProvince field: to provide these values that is necessary to inherit another method:

```
procedures TISXXX.SetParamValues(TargetClass: Tclass; ParamValues: TParams);
begin
inherited;
  SetMultiORValues('Province.Id',MultiProvince, mcReference, ParamValues);
end;
```

Pay attention to the third parameter of type TmultiValueContentType: this type is defined in CBClasses and is used to indicate the type of value contained in the MultiValue field.

For example, if we assume we have a search field containing a series of "Dates" we will have:

```
  SetMultiORValues('BirthDate',MultiDates, mcDate, ParamValues);
```

By simply implementing these 2 methods in your search class you can do multi-valued searches.

## 6.10. *Updated application help*

- Let's now try to update the application help.
- From ISFWorkbench we update the help file (thus creating the MyISFPrimerHelp.md)Thatwill end up in theHelp folder of our project.
- From the application we press F1: the help is contextual to the point we are at!

## 6.11. *Writing Delphi code for business rules*

Let's start writing some Delphi code in classes to implement some business rules.

### 6.11.1. Create progressive code for a class

- We use a mechanism to generate a progressive code for the contact code (and make the field read-only): this mechanism will also be inherited by the other 2 classes!

```
procedures TISContact.AfterCreate;
begin
  inherited;
  Id := TISIdGenerator.CalcNewProgress('TISContact', GetIDSize);
end;
```

### 6.11.2. Create validation rules

- Let's create a field validation rule: the e-mail value must necessarily contain @.

```
procedures TISEmail.SetDescription(const Value: string);
begin
  //Email validation
  if (Value <> '') and (Pos('@',Value)=0) then
    raise Exception.Create('email must contain @');
  inherited;
end;
```

### 6.11.3. Search for objects using InstantQuery

- In the TISCity class there is a mechanism for retrieving other information on the municipality and province starting from the postcode (Zip code).**This example is important to understand the use of the TInstantQuery object:**

```
function TISCity.RetrieveCityFromZip(const Zip: string): TISCity;
var
  InstantQuery : TInstantQuery;
begin
  InstantQuery := InstantDefaultConnector.CreateQuery;
  Try
    //cerco direttamente il comune con il CAP passatomi
    InstantQuery.Command := 'SELECT * FROM TISCity WHERE Zip = :ZIP';
    //Se nella TinstantQuery ho dei parametri devo fare questo, altrimenti non "vede" i parametri:
    InstantQuery.FetchParams(InstantQuery.Command, InstantQuery.Params);
    InstantQuery.Params.ParamByName('ZIP').AsString := Zip;
    InstantQuery.Open;
    if InstantQuery.ObjectCount = 0 then
    begin
      //Se non trovo il CAP direttamente provo ad "azzerare" le ultime due cifre
      InstantQuery.Close;
      InstantQuery.Params.ParamByName('ZIP').AsString := Copy(Zip,1,3)+'00';
      InstantQuery.Open;
    end;
    if InstantQuery.ObjectCount > 0 then
    begin
      Result := InstantQuery.Objects[0] as TISCity;
      Result.AddRef; //è un funzione di retrieve
    end
    else
      Result := nil;
  Finally
    InstantQuery.Free;
  End;
end;
```

We use it in the contact class and the TISContactPerson and TISContactCompany classes will benefit from it:

```
procedure TISContact.SetZip(const Value: string);
var
  ISCity : TISCity;
begin
  //valido il Value
  if (Value <> '') and (Length(Value)<>5) then
    raise Exception.Create(Format('Il valore di %s deve essere lungo %d caratteri',
      [CONTACT_ZIP_TITLE,5]));
  inherited;
  //Recupero la città e la provincia a partire dal CAP
  if (Self.City = '') and (Self.Province = nil) and (Value <> '') then
  begin
    ISCity := TISCity.RetrieveCityFromZip(Value);
    try
      if Assigned(ISCity) then
      begin
        Self.Province := ISCity.Province;
        Self.City := ISCity.Description;
      end;
    finally
      ISCity.Free;
    end;
  end;
  UpdateAttributes;
end;
```

### 6.11.4. Automatically calculate the description in a class

As we have seen, all classes that inherit from TCBInstantObject will always have a "description" attribute. This attribute is very useful for describing an object (e.g. when showing it in a form with code+description). Often, however, the Description attribute must not be entered by the user but must be calculated automatically based on the content of other attributes.

We take the TISContactPerson class and define the Description attribute as “read-only” inside ISWorkbench and regenerate the classes.

Let's write a private CalcDescription method in the TISContactPerson class.

```
procedures TISContactPerson.CalcDescription;
begin
  Description := Trim(LastName+' '+FirstName);
end;
```

This method must be called every time the elements that determine it change, in this case when setting Surname and Name. Let's write the "setters" of these attributes (in override) in the protected section and implement them like this:

```
procedure TISContactPerson.SetLastName(const Value: string);
begin
  inherited;
  CalcDescription;
end;

procedure TISContactPerson.SetFirstName(const Value: string);
begin
  inherited;
  CalcDescription;
end;
```

We also apply this principle to the class of addresses: the description of an address is the composition of the address itself.

We add the address calculation rule by writing a private CalcDx method in the TISAddress class.

```
procedure TISAddress.CalcDx;
var
  ProvId : string;
begin
  if Assigned(Province) then
    ProvId := '('+Province.Id+')'
  else
    ProvId := '';
  Description := Address+' '+Zip+' '+City+' '+ProvId;
end;
```

We write in the protected section (in override) the "setters" of all the attributes that contribute to the address and implement them like this:

```
procedure TISAddress.SetZip(const Value: string);
var
  ObjCity : TISCity;
begin
  inherited;
  //ricerco il City tramite il Zip
  if (Value <> '') and (City = '') then
  begin
    ObjCity := TISCity.RetrieveCityFromZip(Value);
    Try
      if ObjCity <> nil then
      begin
        City := ObjCity.Description;
        Province := ObjCity.Province;
      end;
    Finally
      ObjCity.Free;
    End;
  end;
  CalcDx;
end;

procedure TISAddress.SetCity(const Value: string);
begin
  inherited;
  CalcDx;
end;
procedure TISAddress.SetAddress(const Value: string);
begin
  inherited;
  CalcDx;
end;
procedure TISAddress.SetProvince(Value: TProvince);
begin
  inherited;
  CalcDx;
end;
```

The SetZip method contains the code to retrieve the municipality from the postcode, as already seen for the TContact class.

### 6.11.5. Synchronize data between the master object and its details

The concept that we have introduced into the structure of the classes as "alternative" addresses poses the problem of having among the alternative addresses also the one contained on the master object, i.e. the main address, and of keeping its values synchronized.

Find the implementation of these methods in the example provided with ISF, in the methods:

```
TISContact.RetrieveDefaultAddress
TISContact.UpdateContactFromDefaultAddress
TISContact.AddOrUpdateDefaultAddress
```

They are used to synchronize data with each other, and are invoked in the AfterStore methods of both TContact and TAddress.

### 6.11.6. Master-Detail: with the classic approach

ISF always remains open to a classic Master-Detail relationship approach, especially in situations where both classes have such a value that one class cannot always be considered a detail of another, and vice versa. For this reason in ISF the TCBInstantSelector class also allows the Master-Detail relationship with another Selector or Exposer. Simply indicate the datasource of the master dataset in the DataSource, and indicate the where condition in the WhereCondition property with the parameters necessary for the master-detail relationship (their name must correspond to the name of the key fields present in the master dataset). The master TCBInstantSelector automatically takes care of updating its detail (closing it, assigning new parameters and reopening it) every time the record changes.

### 6.11.7. UpdateAttributes and RefreshLayoutAttributes

One of the problems that often annoys the development of applications and user interfaces is the need to reflect the conditions of the current record (object in this case) at the layout level of a map. Typically, some fields (attributes, in this case) are displayed or made mandatory or not based on the content of others.

With ISF it is necessary to intervene at 2 levels:

1) Implement the method at class levelUpdateAttributes (override) and write the logic for changing attribute values based on the value of another attribute.
2) Provide the TCBInstantSelector orTCBInstantExposer (connected to a TCBXDBMultiEdit editing panel), the "RefreshLayoutFields" property indicating the list of fields whose value affects the value of other attributes. In this way, when the selector/exposer applies the field modification to the underlying object it knows that it must "refresh" the attributes of all its fields (e.g. visible, readonly or required). Consequently, the connected editing panel will automatically readapt to these new settings.

Let's try to implement this by adding an UpdateAttributes method in the TISContact class:

```
procedure TISContact.UpdateAttributes;
begin
  if Zip <> '' then
    _City.Required := True
  else
    _City.Required := False;
end;
```

Now trying to upload a person or a company we cannot leave the municipality field empty if we have specified the postcode (Zip attribute). The problem remains that at the user interface level this "change" is not visible. You need to set the RefreshLayoutFields property of the Selector/Exposer as already mentioned.

If you use the Multi GUI Framework (see specific chapter) just indicate this value in the function map in "autorefresh fields".

Notice:From version 5.3 it is also possible to define this information at class level, in the functionRefreshLayoutAttributes. This will ensure that any connected Selector/Exposer will be able to retrieve this information automatically, so it is no longer necessary to indicate it in the associated function.

```
class function TISContact.RefreshLayoutAttributes: string;
begin
  Result := 'Zip';
end;
```

In this way, in all maps that show a class derived from TISContact (Companies and Persons) as soon as you modify the Zip field (whitening it or inserting a value) the color of the Common field will change (reflecting the fact that it has become mandatory or less). It is also possible to make a field disappear or appear during editing (by setting Attribute.Visible to True or False).

### 6.11.8. Extend a framework class

Sometimes there is a need to extend a framework class, for example by adding attributes. Since it is not possible to intervene directly on the framework class itself, under penalty of future backward compatibility, it is possible to exploit a "notification" mechanism that InstantObjects makes available. In practice, a class is registered that receives notifications from another class (the framework class) every time an object of that class is manipulated.

An example of how to operate is present in the UObjectNotifier.pas unit present in the src folder of ISFPrimer.

## *6.12. Using Messages within InstantSolutions*

To show messages to users, InstantSolutions provides some simplified functions:

**ISInform**or**ISInformFmt**for an information message

**ISWarning**or**ISWarningFmt**for a warning message

**ISError**or**ISErrorFmt**for a blocking error message

the version**Fmt**allows the passing of parameters to the formatted message as normally happens for Format.

For confirmation request messages you can use: ISConfirm or ISConfirmFmt

If you want more control over the buttons or the type of dialog you can call:

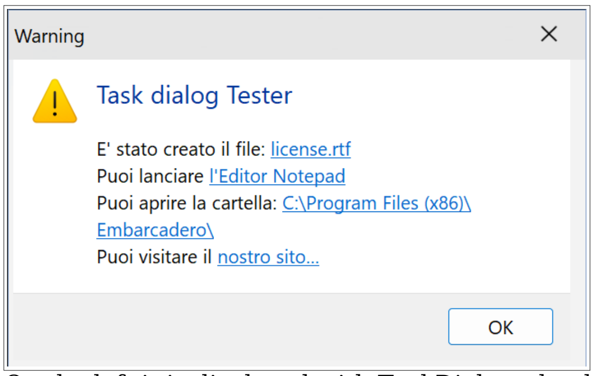
**ISMessageDlg**or**ISMessageDlgFmt**.

It is important to use these functions as they call library functions capable of independently managing whether to use the task dialog (available from Windows Vista onwards) or the message dialog.

Furthermore it is also possible to format the message through hyperlinks: with the use of taskdialogs the hyperlinks are active and the user can click on them.

For example, with this message:

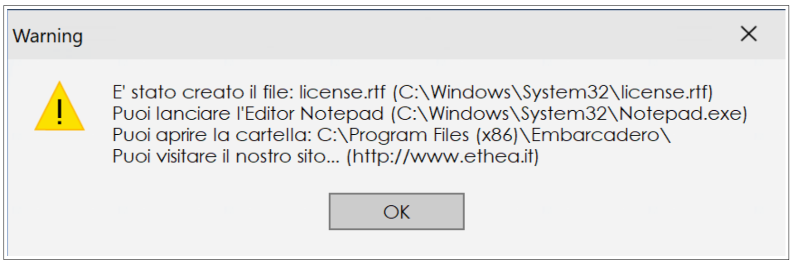
```
Msg :='The file has been created: '+StringToHRef('C:\Windows\System32\license.rtf','license.rtf')+sLineBreak+
'You can run '+StringToHRef('C:\Windows\System32\Notepad.exe','l ''Notepad Editor')+sLineBreak+
'You can open the folder: '+StringToHRef('C:\Program Files (x86)\Embarcadero\')+sLineBreak+
'You can visit '+StringToHRef('http://www.ethea.it','our site...');
```



Warning

Task dialog Tester

E' stato creato il file: license.rtf
Puoi lanciare l'Editor Notepad
Puoi aprire la cartella: C:\Program Files (x86)\Embarcadero\
Puoi visitare il nostro sito...



Warning

E' stato creato il file: license.rtf (C:\Windows\System32\license.rtf)
Puoi lanciare l'Editor Notepad (C:\Windows\System32\Notepad.exe)
Puoi aprire la cartella: C:\Program Files (x86)\Embarcadero\
Puoi visitare il nostro sito... (http://www.ethea.it)

On the left it is displayed with TaskDialog, thanks to hyperlinks, on the right with the old message dialog.

With the taskdialog the user can click on hyperlinks, open the file, access a folder or a site, etc...

Note the use of the StringToHRef function which simply formats two strings making them become a Hyperlink such as:

```
StringToHRef('c:\windows\system32\Notepad.exe','Editor')
```

produces a string:

```
<A HREF="c:\windows\system32\Notepad.exe”>Editor</A>
```

In the same way it is possible to extract the elements of a Href string (ExtractHrefValues) or clean it (HRefToString).

## *6.13. The management of the "function"(TISFunction)in the MultiFramework*

By default, the functions created have a series of preset parameters that can be easily modified and adapted. The important thing is to understand that the MultiFramework starts the windows through the rules defined in the functions and that these rules are initially customizable at the function level and in any case modifiable at the application level. For example we can change the sorting of people with “Name”.

Let's see which parameters can be indicated in the "Function" object:

- **ID:**Function code, default**GSTD_*TISCLASSNAME***
- **Description:**Description of the function
- **HelpContext:** is the index of the help page
- **FormClass:** it is the class of the Framework Form that is used
- **ModalForm:** “Modal” window
- **EditFormClass:** if specified, it is the class of the Framework Form that is used in case of insertion/modification: sometimes TformEdit is used, which is designed for editing a single object, without details.
- **ModalEdit**:“Modal” Edit Window
- **InsertFormClass:** if specified, it is an alternative to the EditFormClass in case of insertion: it often indicates a Wizard that guides the user in entering data.
- **ModalInsert**:“Modal” Insert Window
- **IOClassName:** is the InstantObject class handled in the function
- **RequestedLoadMode:** Type of data loading(choice with dropdown):the default is defined in the .ini file
- **FirstAction:** it is the first action that the form performs (you choose from onecurtain: if it is empty it is equivalent to “Browse”)
- **CustomParameters:** they are optional parameters: you can write any “attributes=value” as long as the attribute is a “Published” property of the form.
- **Extended Description:** is an extended description (200characters)
- **Flag_Menu:** Visible in the tree menu
- **ImageIndex:** Index of the image associated with the function (choose from onecurtain)
- **ImageName:** Name of the image associated with the function (choose from a drop-down menu)
- **WhereCond:** Filter condition “global”: if the form is native it must be written in SQL otherwise in IQL
- **WhereCondDesc:** Description of the “fixed” filter condition
- **OrderBy:** Sorting condition(SQL or IQL)
- **RefreshFields:** List of fields that trigger map layout updates
- **ConfigMode:**
- **ReferenceStopLevel:** Reference detail level: by default it is 1, only first level references
- **AllDetailFields:** Show all fields of referenced objects (default no)
- **LargeTable:** Flag to indicate that the table is large and that it should not be opened immediately, but only

after a search

- **FilterPages:** definition of the filters that will be shown as pages in the list(it is a container of TISFuncFilter objects)
- **Roles:** definition of the roles that will be shown with the “SideBar” (it is a container of objectsTISFuncRole
- **DetailPages:** definition of the pages of the "detail" objects that you want to show (it is a container ofTISFuncDetails)
- **ActiveFilterPage:** Page filter active by default
- **ActiveRole:** Default active role
- **ActiveDetailPage:**
- **Context:** Data context identifier

## 6.14. *The management of the "Context" in the functions*

When you have to manage a lot of data at the application level, organized in different contexts, it can be useful to divide the functions in order to automatically "filter" the data based on the context.

For example, in a utility management software (WATER, ELECTRICITY, GAS), where the "TipoFornitura" information is transversal to almost all archives, it is useful to be able to define a "Context" of the function at the ISFunction level, for example " WATER" and in the global filter REF_TIPO_FORNITURA_REF.ID='ACQUA': in this way the window will only show the data referring to WATER consumption, but above all, in the "lookup" phase on the references it will search for the existence of a "special" function " from the name**WATER_TISCLASS*NAME***: if it finds it, it invokes this function instead of the standard GSTD_TIS*CLASSNAME*.By doing so the user always navigates the filtered data in the context in which he finds himself.

At the object level, there is a class variable “FCurrentContext” that is kept synchronized by the framework, and it is therefore possible to know in which context an object is operating at that moment.

## 6.15. *How objects are created and stored: DemoData.pas*

Taking inspiration from the similar units of the Primer demo (DemoData.pas and RandomData.pas) of instantObjects, we build a unit to create test data for the application. It is useful to see the written code: note that you work on classes and objects, no concept of "DataSet" is needed to write the processing processes.

In the Functions unit (which is also used to customize the application icons) we try to create N contacts after connecting to the database:

```
uses DemoData;

procedure TdtmdFunzioni.AfterConnect(ConnectionDef : TInstantConnectionDef);
begin
  ............
  CreateRandomContacts(10);
end;
```

## 6.16. *How to export data*

### 6.16.1. Export via code

The CBLib library provides some ready-made classes for exporting any "DataSet" in various formats, such as CSV, TXT, XML, Excel. To export data from code, just call a simple function, passing a series of specific parameters and a series of event handlers for choosing the records or fields to export: e.g.

```
uses CBExport;

...
  ExportDataSetToCSVFile(nil,DataSet,FileName,
    IncludeFieldNamesChecked,Delimiter,QuotedChecked,
    AcceptRecord,AcceptField,GetFieldValue);
```

### 6.16.2. Excel Export using a template (xlt o xltx)

When exporting to Excel, it is also possible to specify a "template" Excel sheet (xlt or xltx) to be used for the export, so that the result is formatted according to the specifications of the model, e.g.:

```
uses CBExport;

...
  ExportDataSetToExcelFile(DefaultProgressHandler,ExportExposer,FileName,
    EXCEL_SHEET_NAME,True,AcceptRecord,AcceptField,GetFieldValue,
    True, TemplateFileName);
```

The template must have particular characteristics:

1) must have a “Range” defined with the name of “Sheet1” or “Sheet1” (the constant EXCEL_SHEET_NAME)

2) this range must be 2 lines: the first must contain the name of the field, the second an example value formatted in the correct way (string or number or date, etc...).

The simplest way to build a template is to export an Excel file with the "Wizard", selecting the fields that interest you. Then open the generated file; delete all the records, leaving only one, and “reset” the values of the first record by placing empty values; save the file as xlt (in the project's templates folder).

In this way the Wizard will propose it as the default template file for export (see as an example the “Persons_to_export.xlt” template in the DocTemplates folder of ISFPrimer).

Export through the standard GUI FExportWizard

In ISF (MultiTemplate) there is already a data export Wizard form, already made available to export any DataSet from any window.

In this form it is also possible to define export rules and save them to be able to reload them at a later time.

### 6.16.3. Import through the standard GUI FImportWizard

In ISF (MultiTemplate) there is already a data import Wizard form, already made available to import any DataSet from any window.

In this form it is also possible to define the mapping of the import fields and the import rules can be saved to be able to reload them at a later time.

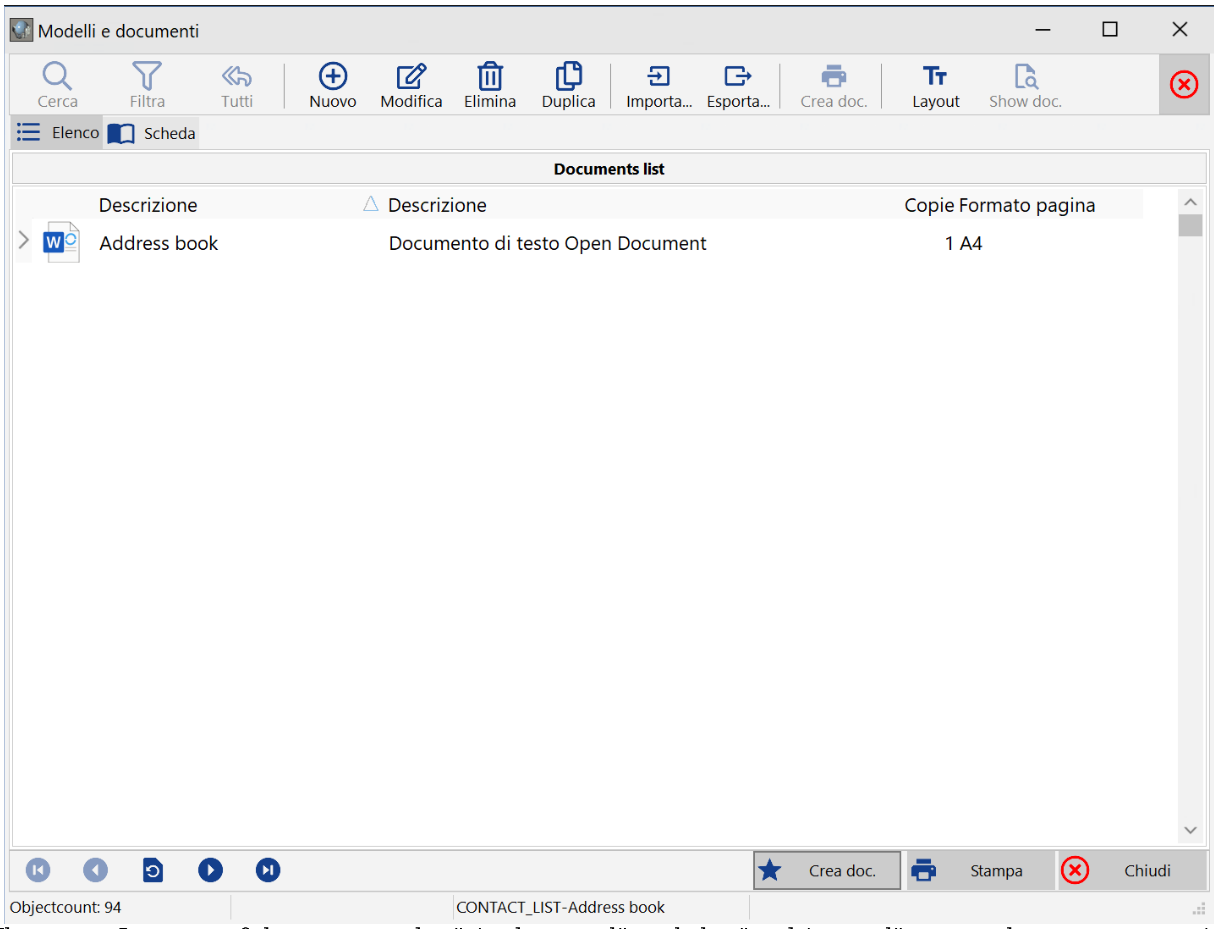
## *6.17. How to create prints with ISF*

In ISF it is very easy to create printouts using different technologies implemented in the CBLib “CBDocProducer” class. The interesting thing is that the work is carried out mainly at run-time with tools that can also be entrusted to the end user or to a person who does not have Delphi development skills.

Let's try to create the same type of print with different technologies.

### 6.17.1. The “print selection” map

The map that allows the user to select the document to "generate" or "print" shows the list of printouts compatible with the calling "class": to "generate" a document (the related file) without printing it press "Create" doc.” while to print it press “Press”.



Documents list

| Descrizione | Descrizione | Copie | Formato pagina |
|---|---|---|---|
| Address book | Documento di testo Open Document | 1 | A4 |

Objectcount: 94 | CONTACT_LIST-Address book

Notice: There are 2 ttypes of documents, the "singlerecord" and the "multirecord" ones: when you are positioned on a "singlerecord" printout, the "Generate" and "Print All" buttons are also activated which allow you to generate/print many documents, one for each element of the list from which the printing choice was activated.

### 6.17.2. Generation of a list (Report)

**List with OpenOffice:**

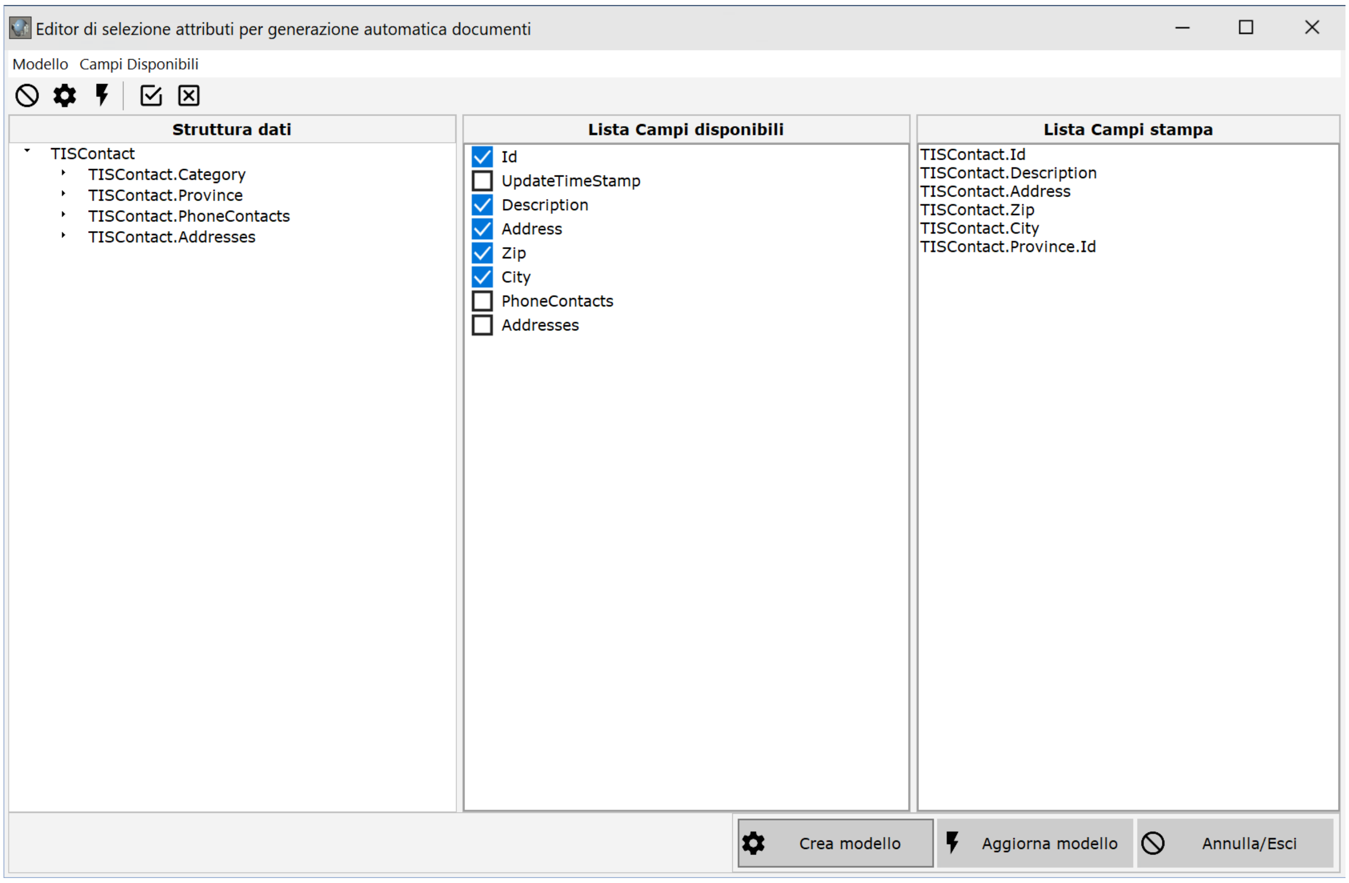
Let's try to create contact list printouts (Login as SYSDBA).

We enter the "Address Book" function, click on the printer to access the window shown in the figure which shows the list of printouts available for this class. Initially it is empty.

Let's create a new record like this:

```
Model: CONTACT_LIST
Description: Address book
File Name: Contact_List
Document type: D
Copies: 1
Editable: No
Page format: A4
Landscape: Yes
Owner Class: TISContact
Use Source Data: No
Records number: -1
```

We confirm the data and move the cursor to the “Field for the document” field: the layout button is activated. We activate the layout editor and select the fields for printing:



Struttura dati:
- TISContact
  - TISContact.Category
  - TISContact.Province
  - TISContact.PhoneContacts
  - TISContact.Addresses

Lista Campi disponibili:
- [✓] Id
- [ ] UpdateTimeStamp
- [✓] Description
- [✓] Address
- [✓] Zip
- [✓] City
- [ ] PhoneContacts
- [ ] Addresses

Lista Campi stampa:
```
TISContact.Id
TISContact.Description
TISContact.Address
TISContact.Zip
TISContact.City
TISContact.Province.Id
```

```
TISContact.Id
TISContact.Description
TISContact.Address
TISContact.Zip
TISContact.City
TISContact.Province.Id
```

Always from inside the “Layout editor” generate the template and exit.

Try running the print to see the result.

**List with FOP:**

Let's clone the template just created to try with FOP, changing:

```
Template: CONTACT_LIST_FOP
Document Type: F
```

We generate the template and try to generate the print.

### 6.17.3. Generation of a “sheet” with details

This time we work on the "people", and from the relevant map, we click on the printer to access the list of prints available for this class.

**Let's create a tab with OpenOffice:**

```
Model: PERSON_FORM
Description: Person Form
File Name: Person_Form
Document type: D
Copies: 1
Editable: No
Page format: A4
Landscape: No
Owner Class: TISContactPerson
Use Source Data: No
Records number: 1
```

We confirm the data and move the cursor to the “Field for the document” field: the layout button is activated. We activate the layout editor and select the fields for printing:

```
TISContact.Id
TISContact.Description
TISContact.Address
TISContact.Zip
TISContact.City
TISContact.Province.Id
```

Always from inside the “Layout editor” generate the template and exit.

Try running the print to see the result. With OpenOffice it is also possible to manage details, but only at the first level.

**Linking a photo in OpenOffice:**

To add the photo you need to create an image in the template and link it to an external image.

Making properties on the image, we go to the "Link" page and indicate the name of the field as the Name of the image, in this case: TISContactPerson.Photo

When creating the document, the external image will be automatically linked.

**Let's create a board with FOP:**

Let's clone the template just created to try with FOP, changing:

```
Template: PERSON_FORM_FOP
Document Type: F
```

We generate the template and try to generate the print.

With an XML editor we modify the template (the xsl file) to obtain the desired layout.

**Image management with ReportBuilder**

For images in ReportBuilder it is necessary to include the uses raIDE (enables RAP) in the UmultiAppSpecific unit to be able to add an OnGetPicture event on a DbImage at run-time like this:

```
procedures DBImage1OnGetPicture(aPicture: TPicture);
begin
aPicture.LoadFromFile('..\site_Template\images\'+TISContactPerson_pipe['Photo']);
end;
```

Notice: With ReportBuilder it is essential to have purchased the enterprise version which includes the RAP which allows you to do everything at run-time through a scripting language.

### 6.17.4. Show the printer selection window

It is possible to decide whether to always show the printer setting/choice window when generating a report that has a printer as destination.

To do this, you need to modify the GSTD_TISDOCREPORT function (by accessing the program with system privileges) and set the "optional parameters" property:

ShowPrinterSettings=True

### 6.17.5. DocProducerEvent: to control particular events related to reports

By inheriting the basic form it is possible to intervene on some aspects of the reports, thanks to the DocProducerEvent event. For example, it is possible to change the default printer for some specific reports (because for example they must be redirected to a virtual printer/fax), or to perform an operation as soon as a report has been generated (such as updating a "printed" flag on the object), etc...

The DocProducesEvent event is invoked many times, during the process of generating a report, with this sequence:

1) etSetMasterDataSet

2) etBeforeCreateTemplate
3) etAfterCreateTemplate
4) etOnPrinterSettings (only if the report destination is a printer).
5) etBeforeCreateDocument
6) etAfterCreateDocument

Based on what you intend to do, you should always test which "phase" of report generation you are in (as seen in the example below).

Furthermore, a DocProducer type class can work using a dataset and its nested-dataset fields (as a TInstantSelector or TInstantExposer object normally does). Sometimes it is necessary to “pass” more than one dataset to the “producer”: you can do this through the “DocProducerEvent” event, as shown in this example.

```
procedure TfmMiaForm.DocProducerEvent(DocReport: TISDocReport; EventType: TISDocEventType);
begin
  inherited;
  if (EventType = etSetMasterDataSet) and (DocReport.Id = 'XXX') then
  begin
    if DocReport.DocProducer.DetailDataSets.Count = 0 then
    begin
      //Aggiungo al DocProducer il dataset di dettaglio
      With DocReport.DocProducer.DetailDataSets.Add do
      begin
        DetailDataSet := MioSelector;
        DetailAliasName := GetDataSetAliasName(DetailDataSet);
      end;
    end;
  end;
end;
```

### 6.17.6. Customize the data to print

The "automatic" mechanism of ISF provides that the prints that are generated are prints that are connected to the "DataSet" active on the window that recalled the printing operation.

Often, however, there is the need to launch a print whose data is "elsewhere" and is not directly linked to the DataSet active on the calling window: in these cases it is possible to intervene to tell the framework that the dataset to be used for the print is a other. This possibility, for now, is only available for the “MultiWindows” Framework and we refer you to the example later in the chapter:**“personalize an ad-hoc print”**.

## *6.18. Enable filters and searches with the MultiFramework*

### 6.18.1. Definition of page filters, roles and search classes

In the MultiFramework there is a concept of search/filter on objects that crosses 3 elements: page filters, roles, search class filters.

### 6.18.2. Enabling filters and roles

We can try using a more advanced framework form, TformDataSideBar and create three "roles" for males and females or all in the people window.

– Role Description: Male – Filter Expression: Sex='M' – Image Index: 104
– Role Description: Female – Filter Expression: Sex='F' – Image Index: 84

We can define roles in the contact form using the TformDataSideBar form:

– Role Description: All – Filter Expression: 1=1 – Image Index: 138
– Role Description: Companies – Filter Expression: Class = 'TISContactCompany' – Image Index: 91
– Role Description: Companies – Filter Expression: Class = 'TISContactCompany' – Image Index: 91
– Role Description: Persons – Filter Expression: Class = 'TISContactPerson' – Image Index: 68

### 6.18.3. Defining a default search class: WhereCondition and OrderByCondition

– Role Description: All – Filter Expression: 1=1 – Image Index: 60

A search class in InstantSolution is a special class that provides a Where and Orderby condition.

To develop a search class you need to define the class in ISWorkbench, inheriting from TCBSelectObject. In Delphi code you need to implement the method **GetWhereCondition** of the research class. The search class is a class that can also be reused on multiple "target" classes (which is a parameter of the GetWhereCondition.

In practice GetWhereCondition must return an IQL or SQL query (depending on whether you use it in standard or native forms) to the objects of the "target" class, therefore the search class can be the same but depending on the target class it can have a different syntax, with different field names.

A "default" search class does not require logical operators to be entered (and, or, etc...) but the logic is internal to the GetWhereCondition method. E.g. if I have to do a search between a range of 2 dates I define a "from" field and a "to" field in the search class and in the GetWhereCondition I will create the condition >= value of From and <= value of To, testing whether the Dal and Al camps were charged. For values such as dates, since each database engine formats the value differently, you need to use parameters:

```
'DateStipula >= :FromDateStipula
```

and provide the field value through the SetParamValues method.

To define the ordering of the search result, you need to write the method**GetOrderByCondition**. Normally the order is established by the "Function" (default is the description) and the user can change the order by clicking on the titles of the list grid. When you force orderby in a search class the default order is changed, otherwise the pre-existing one remains.

For example, the TISSelLocalita search class allows you to search for objects of a class that has ifields: Postcode, Municipality and Province.Id, exactly like our TContact class.

To make a predefined search class available, we need to "register" the search class to our class, in the initialization:

```
//search class registration
CBRegisterSelectClasses(TISContact,[TISSelLocalita]);
```

At this point, the "filter" button (funnel) is available in the contact form to enable the search.

A very convenient thing about the search is the possibility of memorizing the criteria and recalling them later.

The search filter is "combined" with the concepts of "role" and "page filter" already seen.

### 6.18.4. Example: Create a lookup class for addresses

For**define the Where clause**based on the user input, it is sufficient to inherit from TCBSelectObject and implement the method in the class unitGetWhereConditionalso using the following methods:

```
function CalcLikeCondition(const PropName, Value: string): string;
function OperatorCondition(const PropName, Operator, Value: string): string; overload;
function OperatorCondition(const PropName, Operator : string; Value: integer): string; overload;
procedures AddAndWhereCond(const Condition: string);
```

present in the search base classes, to more easily construct the where condition.

For example:

```
function TISSelLocation.GetWhereCondition(TargetClass: TClass): string;
begin
  WhereCond := '';
  if Zip <> '' then
    AddAndWhereCond(CalcLikeCondition('Zip',Zip));
  if City <> '' then
    AddAndWhereCond(CalcLikeCondition('City',City));
  if Province <> nil then
    AddAndWhereCond('Province.Id = '+QuotedValue(Province.Id));

  Result := WhereCond;
end;
```

Notice:It is possible to construct the IQL query also through the use of parameters (e.g. DataStipula < :Today). The value of the parameter must be set in the class Procedure SetFilterParamValue class.

### 6.18.5. The mechanism linked to SetFilterParamValue

In the TCBInstantObject base class there is a widely used method to set search values specified in the where conditions of a search class, or of a page or role filter.

The bass class already implements the setting of some "standard" parameters such as:

```
'No' o 'False' <- False
'Si' o 'Yes' o 'True' <- True
'Oggi''Today' <- Date
'Adesso' o 'Now' <- Now
'DataVuota' o 'EmptyDate' <- 0
'InizioMese' o 'StartOfMonth' EncodeDate(YearOf(Date),MonthOf(Date),1)
'FineMese' o 'EndOfMonth' EncodeDate(YearOf(Date),MonthOf(Date),DaysInMonth(Date));
```

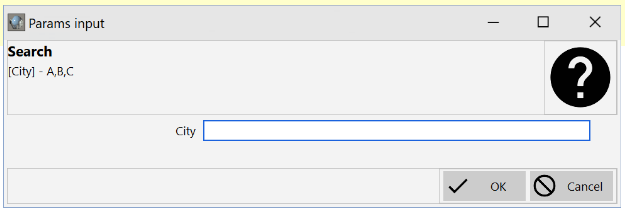
Other "non-standard" parameters must be managed within the single class, inheriting SetFilterParamValue.

### 6.18.6. Use parameters in searches/filters/search classes

It is possible to have the value of the search parameters entered directly by the user. Previously it was necessary for a parameter to always be "resolved" within the methodSetFilterParamValueof the class on which the search is carried out. To give the user the possibility to indicate some search parameters on a certain class, the class function must be implementedGetHiddenFilterParam(by default this function returns True, because all the parameters are not editable by the user).

Let's try to grant the possibility of indicating search parameters to the contact class. We will write:

```
class function TISContact.GetHiddenFilterParam(const ParamName: string): boolean;
begin
  //On this class I can apply "free" searches with "user" parameters
  Result := False;
end;
```


Params input
Search
[City] - A,B,C
City

At this point we try to add some "contacts" to the GSTD_TISCONTACT functionof the address book" a new "Role" with the name "City" with the filter expression:(Municipality LIKE :Municipality).

When we click on the "city" icon on the map, a request for the parameter value appears (as seen in the figure).

Using the % it is possible to carry out partial searches.

It is also possible to use this technique for page filters, as well as for roles, or even at the application level, when using a TCBInstantSelector component.

The event must be setTCBInstantSelector.OnGetParamValues and in the event-handler call the CBEditParams function (the example is available in FData.pas).

The same thing was done in the FQueryView form (free searches): if you set a search with parameters the input mechanism is triggered.

### 6.18.7. Use a customized parameter passing mechanism

The standard parameter input form (see figure) is just one of the possible ways to provide the query that uses generic searches to provide the value of the parameters. It is possible to hook to the event handlerTCBInstantSelector.OnGetParamValues any method that provides the necessary parameters directly or through prompting the user for input.

### 6.18.8. Use a "generic" search class

Using the same parameter concept seen for page filters and roles, it is also possible to use a "generic" search class to enable searches without necessarily having to create an ad-hoc class.

**Advantages**: it is not necessary to recompile the application to have new search classes available (provided you have "enabled" a class to receive "user" search parameters, as explained in the previous point.

**Disadvantages**: The management of search parameters is less advanced

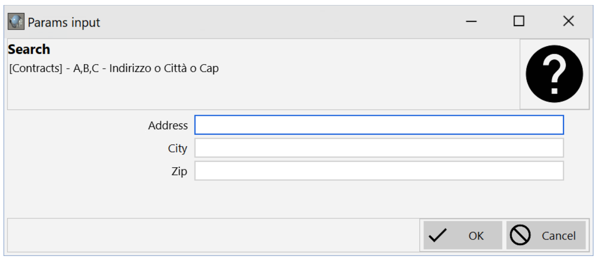
The TISSelGeneric class (unit SSelGeneric.pas) and its detail class TISSelGenericDetail are used to manage this type of "generic" search.

By launching the "System parameters/Special search conditions" menu you enter the generic search classes management window.

Simply insert a new record, indicate a description for the user, a filter and the class on which to activate the search: for example:

```
Description:Address or City or Postcode
Where IQL condition:(Address = :Address) OR (City LIKE :City) OR (Zip = :Zip)
Search object: TISContact
```

At this point, just launch the contact management window and press the search/filter button (the funnel): choose "Generic/customizable search conditions" and select "Municipality or province or postal code".


Params input
Search
[Contracts] - A,B,C - Indirizzo o Città o Cap
Address
City
Zip

The same map for entering the search parameters will appear, showing the list of available search fields.

To reset the "free" search filter, press the "Restore complete list" button (the little house).

*Notice:The parameter request form is "cumulative": if there are multiple parameters relating to free searches or page or role filters, they are requested all together.*

### *6.18.9. Use a "generic" search class in a "composite" way*

An evolution introduced in version 5.8 is the possibility of composing the generic search class with multi-level AND or OR logical operators. Detail records of type TISSelGenericDetailof a generic search class can be defined in a hierarchical structure with operators and can also hook records of search classes already registered. It is therefore possible to use the generic search class to compose complex searches and subject them to inclusion or exclusion logics.

### 6.18.10. Customize search parameters

As we were already saying, if within a class you want to provide "automatic" search parameter values (without the user being able to modify them) you need to provide the value of the parameter in the SetFilterParamValue class procedure. Here's how to do it:

1) Let's add a “role” to the GSTD_TISContactPerson function.
   1. Description = “In the Province”
   2. Filter expression = “Province = :Province_Company”
   3. Image index = 53

By doing so, since we have enabled the TContact class to receive input parameters from the user, the parameter request window would always appearProvince_Society, but we want to provide this parameter with a fixed value, calculated by the application.

2) In the TContact class, inherit the SetFilterParamValue class procedure and write:

```
class procedure TISContact.SetFilterParamValue(Param: TParam);
var
  Societa : TISSocieta;
begin
  inherited;
  if SameText(Param.name,'Contact_Province') then
  begin
    Societa := RetrieveDefaultSocieta;
    try
      if Assigned(Societa.Provincia) then
        Param.AsString := Societa.Provincia.Id;
    finally
      Societa.Free;
    end;
  end;
end;
```

3) In the TContact class, modify the class procedure GetHiddenFilterParam and write:

```
class function TISContact.GetHiddenFilterParam(
  const ParamName: string): boolean;
begin
  if SameText(ParamName,'Provincia_Societa') then
  begin
    //Questo parametro non è modificabile da parte dell'utente
    Result := True;
  end
  else
  begin
    //Di default i contatti consentono ricerche "libere"
    Result := False;
  end;
end;
```

In this way we provide 2 pieces of information: the first is the value of the parameter (taken from the province of the default company registered in the database), the second that the parameter is "hidden" and therefore input must not be requested.

The same mechanism will also work on the TISContactCompany class.

### 6.18.11. Search classes for “Native” forms

In the case of "Native" forms, all the logic related to the search classes works equally as long as the "Where" condition of the search class is written in SQL and not IQL.

### 6.18.12. Filter the data visible in the Explorer

On the details page there is the InstantExplorer to show the detail data which can be containers or objects.From version 6.4 it is possible to organize this page into subpages, if there are many nodes to show (see the specific chapter below).

It is also possible to apply filter rules on the objects shown on the explorer (of the TCBInstantExplorer type), both at the container node and object level.

To hide a certain container node, you need to inherit and implement:

procedures DoIncludeNode(Sender: TInstantExplorer;

NodeData: TInstantExplorerNodeData; var Include: Boolean); virtual;

For object nodes, you must not use the DoIncludeNode otherwise the objects would still be visible on the grid of a container node, but rather you need to filter the objects by activating Limited and using the OnLimit event of the

TCBInstantExplorer component.

e.g. in the FContacts custom form implement:

```
procedures TfmContacts.InitExposers;
begin
  inherited;
  //Filter the explorer data
  ioExplorer.Limited := True;
  ioExplorer.OnLimit := LimitExploperObjects;
end;
```

And

```
procedure TfmContacts.LimitExploperObjects(Sender, AObject: TObject;
var Accept: Boolean);
begin
  //I don't show Gmail emails
  if (AObject is TISEmail) and (Pos('gmail.com', TISEmail(AObject).Description) > 0) then
    Accept := False;
end;
```

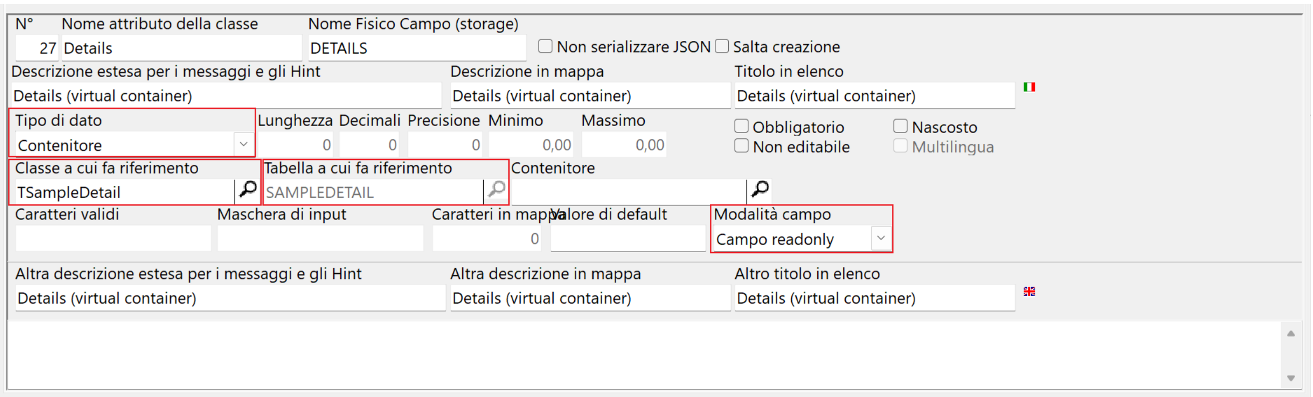
## 6.19. *“Virtual” container attribute*

In InstantSolution it is possible to use detail classes of a master class without necessarily having a container field in the master object, at Database level.

At the InstantObject class level, however, there will still be a "virtual" container that is able to access the detailed objects through the metadata information provided to it.

### 6.19.1. Management of “virtual” containers in ISWorkbench

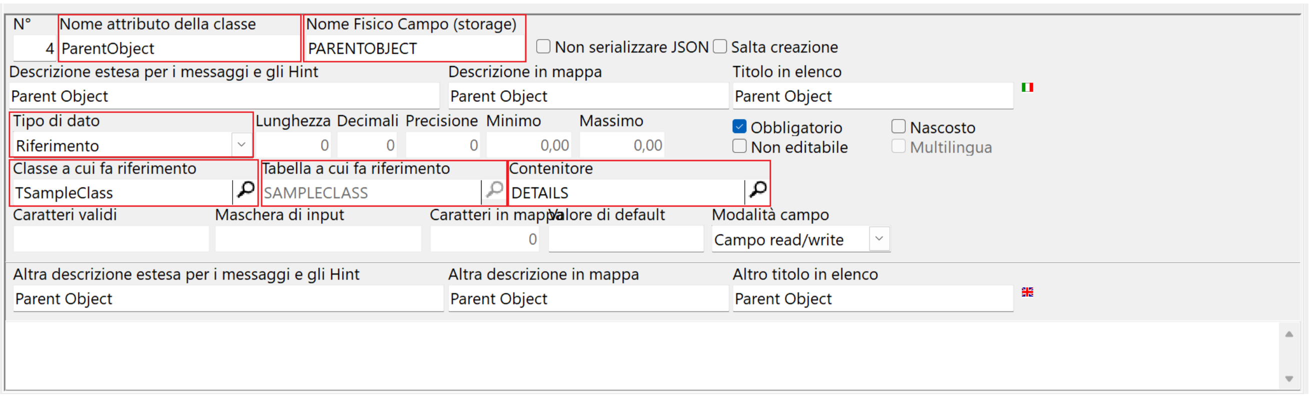
First you need to define a virtual container within ISWorkbench. This container is identical to the standard definition except for the fact that it is defined as a “Read-Only” field, as in the example of the Details attribute of the TISSampleClass class of the ISFPrimer project:



| | |
|---|---|
| N° | 27 |
| Nome attributo della classe | Details |
| Nome Fisico Campo (storage) | DETAILS |
| Non serializzare JSON | ☐ |
| Salta creazione | ☐ |
| Descrizione estesa per i messaggi e gli Hint | Details (virtual container) |
| Descrizione in mappa | Details (virtual container) |
| Titolo in elenco | Details (virtual container) |
| Tipo di dato | Contenitore |
| Lunghezza | 0 |
| Decimali | 0 |
| Precisione | 0 |
| Minimo | 0,00 |
| Massimo | 0,00 |
| Obbligatorio | ☐ |
| Non editabile | ☐ |
| Nascosto | ☐ |
| Multilingua | ☐ |
| Classe a cui fa riferimento | TSampleDetail |
| Tabella a cui fa riferimento | SAMPLEDETAIL |
| Contenitore | |
| Caratteri validi | |
| Maschera di input | |
| Caratteri in mappa | 0 |
| Valore di default | |
| Modalità campo | Campo readonly |
| Altra descrizione estesa per i messaggi e gli Hint | Details (virtual container) |
| Altra descrizione in mappa | Details (virtual container) |
| Altro titolo in elenco | Details (virtual container) |

Notice:the field defined in this way is not created in the database table, therefore the container is only "virtual" and managed within the InstantObject class.

Similarly, in the detail class (TISSampleDetail), it is necessary to define a reference to the master connected to the "virtual" container. This configuration is identical to that for non-virtual containers.



| | |
|---|---|
| N° | 4 |
| Nome attributo della classe | ParentObject |
| Nome Fisico Campo (storage) | PARENTOBJECT |
| Non serializzare JSON | ☐ |
| Salta creazione | ☐ |
| Descrizione estesa per i messaggi e gli Hint | Parent Object |
| Descrizione in mappa | Parent Object |
| Titolo in elenco | Parent Object |
| Tipo di dato | Riferimento |
| Lunghezza | 0 |
| Decimali | 0 |
| Precisione | 0 |
| Minimo | 0,00 |
| Massimo | 0,00 |
| Obbligatorio | ✓ |
| Non editabile | ☐ |
| Nascosto | ☐ |
| Multilingua | ☐ |
| Classe a cui fa riferimento | TSampleClass |
| Tabella a cui fa riferimento | SAMPLECLASS |
| Contenitore | DETAILS |
| Caratteri validi | |
| Maschera di input | |
| Caratteri in mappa | 0 |
| Valore di default | |
| Modalità campo | Campo read/write |
| Altra descrizione estesa per i messaggi e gli Hint | Parent Object |
| Altra descrizione in mappa | Parent Object |
| Altro titolo in elenco | Parent Object |

Note that you can define the name of the attribute and the physical field linked to the master table freely. This setting will have the effect of generating a new class definition in the Mdictionary.pas unit of this type:

```
Details: References(TISSampleDetail) virtual 'SAMPLEDETAIL;PARENTOBJECTCLASS;PARENTOBJECTID';
```

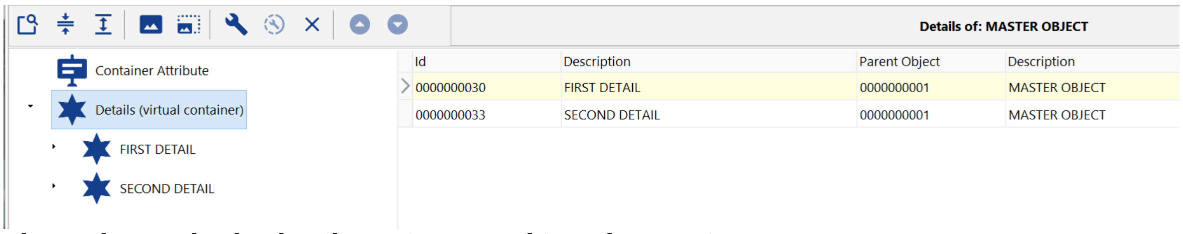
Note the new "virtual" keyword of a references type attribute and the storagename which contains a string composed of the detail table and the key fields used for the master-detail join.

### 6.19.2. “Virtual” container shown on Explorer

At the application level, there is no need to do anything else as the container (TCBInstantReferences) which contains the list of detail record objects continues to exist on the master class.

The immediate effect is to have the "virtual" container available on the explorer exactly as if it were stored in the memo field of the database.

The internal mechanism ensures that a detail access query is used to load the list of detail "ids", rather than reading the contents of the memo field.



Details of: MASTER OBJECT

| Id | Description | Parent Object | Description |
|---|---|---|---|
| 0000000030 | FIRST DETAIL | 0000000001 | MASTER OBJECT |
| 0000000033 | SECOND DETAIL | 0000000001 | MASTER OBJECT |

The image shows the result, the details are integrated into the treeview.

### 6.19.3. Sorting the details of a “virtual” container

By default the system always sorts the details based on the Description field, so the order corresponds to the value visible on the tree.

It is not possible to move the detail objects with the Move, but it is possible to define an ordering as desired also through the join with other tables.

The GetDetailsStatementValues method must be implemented at the class level of the container by setting one or more variables that are passed.

Since there can be multiple virtual containers in a class, it is more correct to test the name of the detail table that is passed in the FromClause.

If you only want to implement sorting, just update the OrderByClause variable. In the example the sorting is the descending description:

```
procedure TISSampleClass.GetDetailsStatementValues(
  var FromClause, SequenceNoFieldName, OrderByClause: string);
begin
  inherited;
  if SameText(FromClause,'SAMPLEDETAIL') then
  begin
    OrderByClause := 'COMPANY.DX DESC';
  end;
end;
```

It is also possible to establish a sorting based on a field belonging to a table referenced by the detail table. In the example the order is given by the description of the company referenced in detail.

```
procedure TISSampleClass.GetDetailsStatementValues(
  var FromClause, SequenceNoFieldName, OrderByClause: string);
begin
  inherited;
  if SameText(FromClause,'SAMPLEDETAIL') then
  begin
    OrderByClause := 'COMPANY.DX DESC';
  end;
end;
```

The SequenceNoFieldName variable is not managed for now: it will be used when you want to implement a sequential sorting based on a field, but for now there is no support for updating this field on the detail table.

# 7. The MultiFramework GUI

The MultiFramework is based on a concept of "visual inheritance" through which some basic forms useful for development have been created.

The framework is based on a main form, (FmainXPor FMainModern) and by “secondary” forms that are invoked when a function (TISFunction) connected to a menu item (TISMenuItem) is activated.

## 7.1. *Customization of the layout of the Main Form and other graphic elements*

In the same folder (exe) where the application executable is present there is an .ini file (with the same name as the application) which defines some attributes of the application: let's see the meaning of the most important ones:

ApplicationStyle=MDI or SDI or TDI (controls the layout of application windows)

ColorMap=Silver,Olive,Silver,Blue (toolbar color)

HideMenuStructure=0 or 1 (hides the menu tree)

HideActiveFunctions=1 or 0 (hides the active functions window)

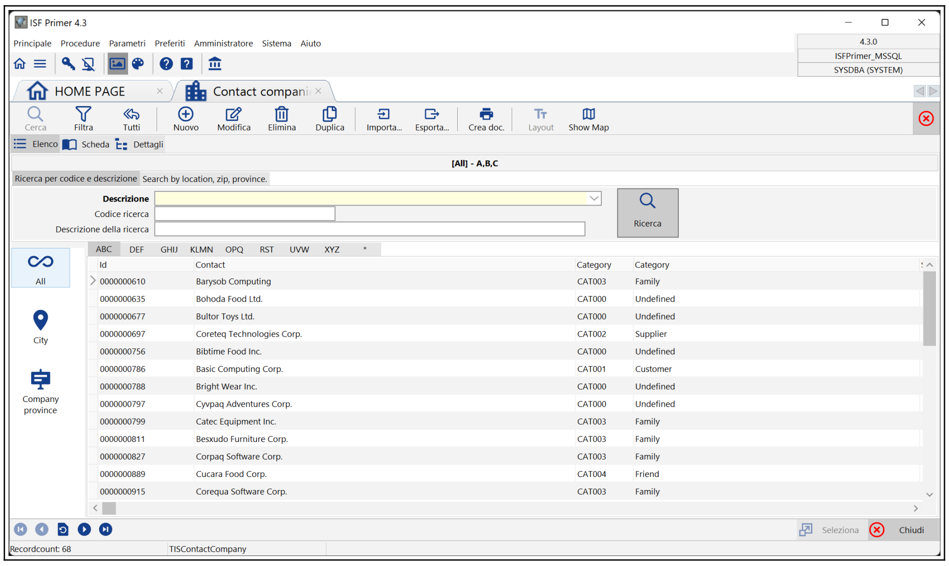
MenuStructureEmbedded=0 or 1 (pop-up or fixed tree menu)

ActiveFunctionsEmbedded=0 or 1 (pop-up or fixed active functions window)

SQLMonitorEmbedded=0 or 1 (pop-up or fixed SQL monitor window)

HideMainToolbar=0 or 1 (hides the main form toolbar).

Here is an example of a TDI application, with the windows inside the TABs:


ISF Primer 4.3
Principale Procedure Parametri Preferiti Amministratore Sistema Aiuto
4.3.0
ISFPrimer_MSSQL
SYSDBA (SYSTEM)
HOME PAGE
Contact compani
Cerca Filtra Tutti Nuovo Modifica Elimina Duplica Importa... Esporta... Crea doc. Layout Show Map
Elenco Scheda Dettagli
[All] - A,B,C
Ricerca per codice e descrizione
Search by location, zip, province.
Descrizione
Codice ricerca
Descrizione della ricerca
Ricerca
ABC DEF GHIJ KLMN OPQ RST UVW XYZ *
All
City
Company province
Seleziona
Chiudi

| Id | Contact | Category | Category |
|---|---|---|---|
| 0000000610 | Barysob Computing | CAT003 | Family |
| 0000000635 | Bohoda Food Ltd. | CAT000 | Undefined |
| 0000000677 | Bultor Toys Ltd. | CAT000 | Undefined |
| 0000000697 | Coreteq Technologies Corp. | CAT002 | Supplier |
| 0000000756 | Bibtime Food Inc. | CAT000 | Undefined |
| 0000000786 | Basic Computing Corp. | CAT001 | Customer |
| 0000000788 | Bright Wear Inc. | CAT000 | Undefined |
| 0000000797 | Cyvpaq Adventures Corp. | CAT000 | Undefined |
| 0000000799 | Catec Equipment Inc. | CAT003 | Family |
| 0000000811 | Besxudo Furniture Corp. | CAT003 | Family |
| 0000000827 | Corpaq Software Corp. | CAT003 | Family |
| 0000000889 | Cucara Food Corp. | CAT004 | Friend |
| 0000000915 | Corequa Software Corp. | CAT003 | Family |

Recordcount: 68 TISContactCompany

## 7.2. *Customizing the layout of maps and lists*

ISF provides a built-in editor for customizing maps. It is part of the CBLib components and is very convenient for easily and quickly defining a map or grid layout, which is stored at the dictionary level.

– Connect howuser “system”and customize maps of people and companies.
– Go to a "list" grid or a map, the "Layout" button also appears: pressing it activates the layout editor (for more information on using the layout editor, use the ISWorkbench help under " Layout editor for maps” and “Map layout rules” or “Layout editor for grids” and “Grid layout rules”. The result of the layout is stored in the ISWorkbench repository and is visible on the “Layout” page.
– You can also add backgrounds to the maps: try with maps of people and companies: just add a jpg or bmp or Gif file with the name of the class (e.g. TISContact.gif) to the "backgrounds" folder.
– It is possible to use the NunmberBox components as an alternative to the CurrencyEdit components by setting in the ini file in the Framework section:
    – NumberBoxEditorOptions=nbSmallInt,nbInteger,nbCurrency,nbFloat

Notice:some component sizes have a defa sizeult different from ISF7, to facilitate the creation of layouts based on "columns" multiples of 4: Currency=12; Date=12; DateTimeHHMM = 16;Time=8; TimeHHMM=8; Float=12, GUID=28, TimeStamp=16)

## 7.3. *The role of ActiveDataSet, ActiveClass, CurrentObject*

It is important to know that the framework works on a DataSet concept that is active in a certain context(and consequently of an active “ActiveClass”.

The basic forms of the framework are already created in such a way as to correctly set the active DataSet when, for example, you change pages or change nodes on the details page explorer.

In addition to ActiveDataSet, the system also acts based on the concept of ActiveClass (the InstantObject class active in a certain context which is almost always the class controlled by the ActiveDataSet).

Finally, you can always use the CurrentObject function to find out what the current object is (be careful that it could be nil)of the active class.

### 7.3.1. Exploit the visual inheritance of the MultiFramework

The concept of visual inheritance can be very useful for customizing framework functionality.

It should be remembered that in inherited forms the ActiveDataSet/ActiveClassis always consistent with the context in which you find yourself:for this reason automatic mechanisms have been set up.

### 7.3.2. Automatically set ActiveDataSet and ActiveClass

When developing "custom" forms that inherit it is necessary to always remember to provide the Framework with an indication of the ActiveDataSet (and consequently of the ActiveClass) active in a certain context. Since the new GUI is often based on DbGrid or DBMultiEdit, two ready-made events are provided to be hooked to the "OnEnter" event handler of the DbGrid and DBMultiEdit which are created in the "custom" form:

- DBGridEnterChangeDataSet

- MultiEditEnterChangeDataSet

it will therefore be sufficient to hook these events to change the ActiveDataSet.

### 7.3.3. Set ActiveDataSet on page change

If you create a new Selector/ExposerTo show its data in a detail page, it is important to ensure that when that page changes, the ActiveDataSet must be set in the page's OnChange event.

e.g. a page has been added to the FAccessRole form that shows the profiles used by the current role. This page shows a grid connected to the ioslProfiles Selector and has therefore been implemented:

```
procedure TformAccessRole.pgctDataChange(Sender: TObject);
begin
  inherited;
  if pgctData.ActivePage = tsProfile then
  begin
    OpenProfiles;
    ActiveDataSet := ioslProfiles;
  end;
end;
```

In this way, changing the page changes the ActiveDataSet (and consequently the ActiveClass and the CurrentObject) and the user interface is consistent with this change.

### 7.3.4. Example of a “custom” form

Instead, let's try to inherit from the TformDataSideBar form to create the contact form which must be able, in case of insertion or modification of an object, to decide which form to activate based on the class (TISContactPerson or TISContactCompany): let's save the FContact unit containing the TfmContact form . We remove the global “var” and register the class to make it available at run-time. (Notice:With this operation the dpr file is completely rewritten, losing all useful comments, it is best to restore it from the backup).

By default, when editing an object the form itself is used, but it is also possible to enable an alternative edit form by indicating it in the "function" to enable the "modal" editing form.

If the object to be edited is of a "derived" class, the framework is already able to enable editing on the correct class. What it cannot do is know when inserting which "derived" class the object is intended to be inserted.

In the customized form we must implement the method:

```
uses
  FCBSelectOptions, FChild, MContact, MPerson, MCompany;
{ TfmContacts }
function TfmContacts.GetInsertClassName: string;
var
  Selection : integer;
begin
  //Chiedo di quale classe si vuole creare l'ogetto del contatto
  if ActiveClass = TISContact then
  begin
    Selection := CBSelectOption(
      'Inserimento Persona'+sLineBreak+
      'Inserimento Azienda'+sLineBreak,
      Font, False, 'Seleziona il tipo di contatto');
    case Selection of
      0 : Result := TISContactPerson.ClassName;
      1 : Result := TISContactCompany.ClassName;
    else
      Abort;
    end;
  end
  else
    Result := inherited GetInsertClassName;
end;
```

### 7.3.5. Personalize an ad-hoc print

We also try to create a printout of all the provinces where at least one of our contacts exists. As we said in the previous chapter on prints, it is possible to change the default print mechanism.

In our new custom form, we add a selector with the name**"ioslProvince"**and we set the command property like this:

```
SELECT * FROM TISProvince
WHERE EXISTS (SELECT * FROM ANY TISContact WHERE Province <> '' USING Province)
ORDER BY Description'
```

This query extracts all provinces that are referenced by at least one contact from the "province" property.

We also set:

```
ContentDescription = 'Provinces where there are contacts'
```

We therefore want to create a Report with OpenOffice whose code will be: "PROVINCECONTATTI": since the "design" of the report all happens at run-time, we prepare our form to indicate that in the event that the report should be this, the data will be retrieved by the selector**"ioslProvince"**, both in the "Layout creation" phase and in the "printing" phase.

To do this we must "inherit" two methods and implement them in this way:

```
function TfmContacts.DoCreateDocument(const DocReportId: string;
  DataSet: TDataSet; Mode: TCBDocProducerMode; MultiDocument: boolean): string;
var
  DataSetToPrint: TDataSet;
begin
  if SameText(docReportId,'PROVINCECONTATTI') then
    DataSetToPrint := ioslProvince else
    DataSetToPrint := DataSet;
  Result := inherited DoCreateDocument(DocReportId,
    DataSetToPrint, Mode, MultiDocument);
end;

procedure TfmContacts.DoCreateTemplate(const DocReportId: string; DataSet: TDataSet);
var
  DataSetToPrint: TDataSet;
begin
  if SameText(docReportId,'PROVINCECONTATTI') then
    DataSetToPrint := ioslProvince else
    DataSetToPrint := DataSet;
  inherited DoCreateTemplate(DocReportId, DataSetToPrint);
end;
```

As you can see these 2 methods simply provide the appropriate DataSet. At Run-time follow the instructions on how to create a print (see below**"How to create prints with ISF"**), remembering to assign these values:

Model:"PROVINCECONTACTS" (is the Id field)

Document description: "Contact provinces"

File Name: "ProvinceContatti"

Type: "D" (Open Document text document).

Reference class:**"TISContact" Notice:It is the class of the form that started printing!**

Use data source:**"YES"** ***Notice:To avoid creating an alternative Selector/Exposer***

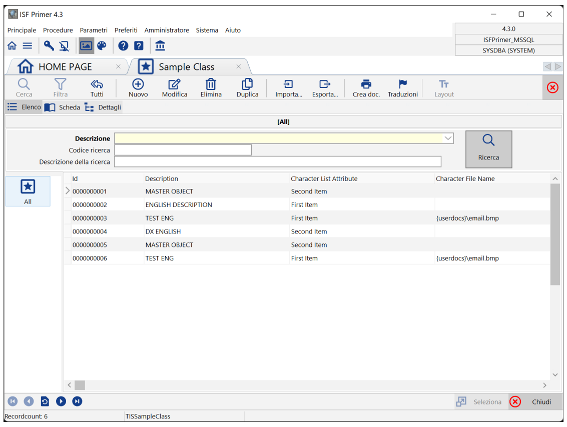
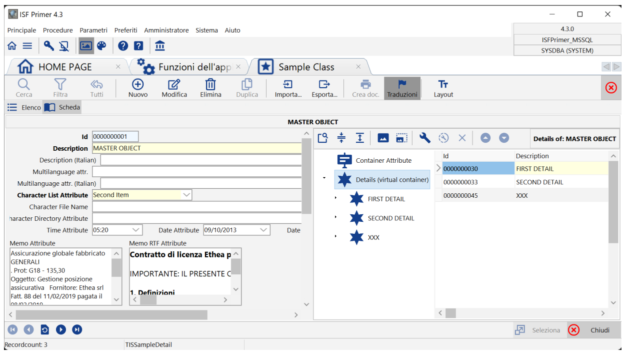
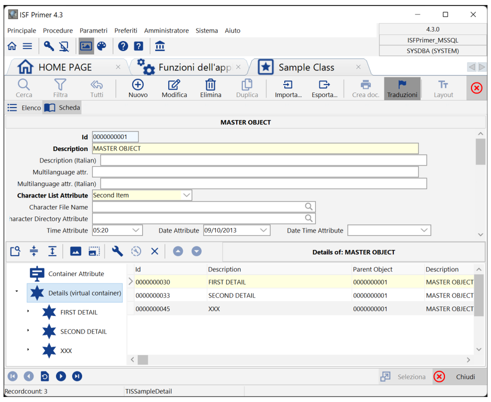
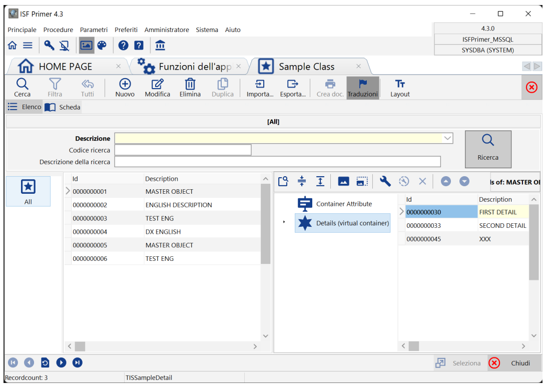
Printout:**"YES"**and Records Number:**-1*To make a horizontally oriented printout.***

Save and generate the template (by pressing the "Layout" button) by selecting all the fields of the Province. Then try to generate the OpenOffice document of all contact provinces.

### 7.3.6. Customize the layout of the standard TformDataStd form (unit FDataStd.pas)

The TformDataStd form included in the Framework exposes some published properties to customize the "layout" of the various parts. Normally the TformDataStd Form consists of 3 pages: List, Card, Details.

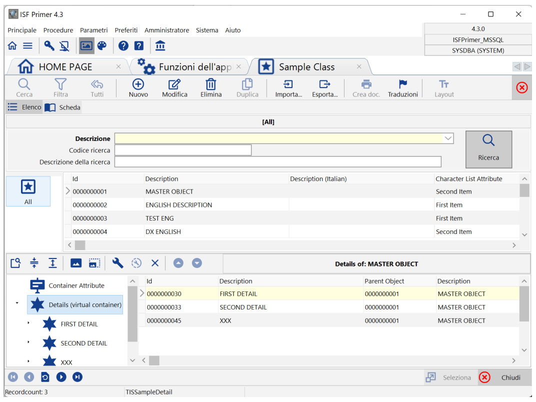
**EmbeddedExplorerPositioncan** take on these values:

| | |
|---|---|
| eepDetailPage:<br>Explorer on the Details page<br>(default) |  |
| eepDataRight:<br>Explorer on the data page to the right of the tab |  |
| eepDataBottom:<br>Explorer on the data page under the tab |  |
| eepListRight:<br>Explorer on the list page to the right of the list |  |

| Id | Description | Character List Attribute | Character File Name |
|---|---|---|---|
| 0000000001 | MASTER OBJECT | Second Item | |
| 0000000002 | ENGLISH DESCRIPTION | First Item | |
| 0000000003 | TEST ENG | First Item | {userdocs}\email.bmp |
| 0000000004 | DX ENGLISH | Second Item | |
| 0000000005 | MASTER OBJECT | Second Item | |
| 0000000006 | TEST ENG | First Item | {userdocs}\email.bmp |

Details of: MASTER OBJECT

| Id | Description | Parent Object | Description |
|---|---|---|---|
| 0000000030 | FIRST DETAIL | 0000000001 | MASTER OBJECT |
| 0000000033 | SECOND DETAIL | 0000000001 | MASTER OBJECT |
| 0000000045 | XXX | 0000000001 | MASTER OBJECT |

| | |
|---|---|
| eepListBottom: Explorer on the list page below the list |  |

ISF Primer 4.3
Principale Procedure Parametri Preferiti Amministratore Sistema Aiuto
4.3.0
ISFPrimer_MSSQL
SYSDBA (SYSTEM)
HOME PAGE
Funzioni dell'app
Sample Class
Cerca Filtra Tutti Nuovo Modifica Elimina Duplica Importa... Esporta... Crea doc. Traduzioni Layout
Elenco Scheda
[All]
Ricerca
All
Details of: MASTER OBJECT
Container Attribute
Details (virtual container)
FIRST DETAIL
SECOND DETAIL
XXX
Seleziona Chiudi

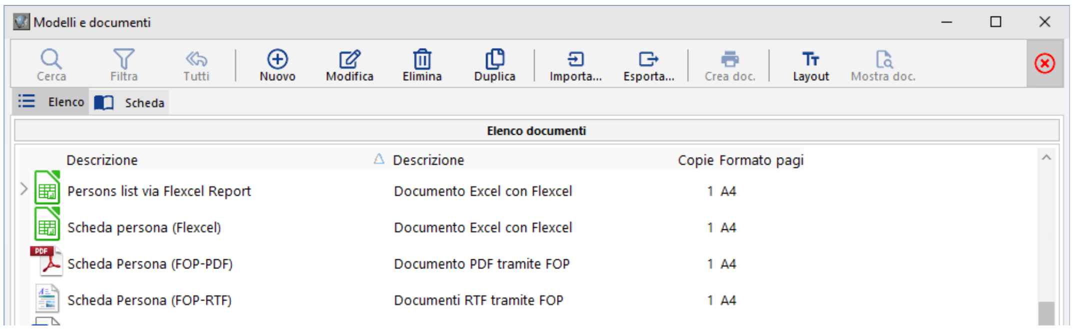
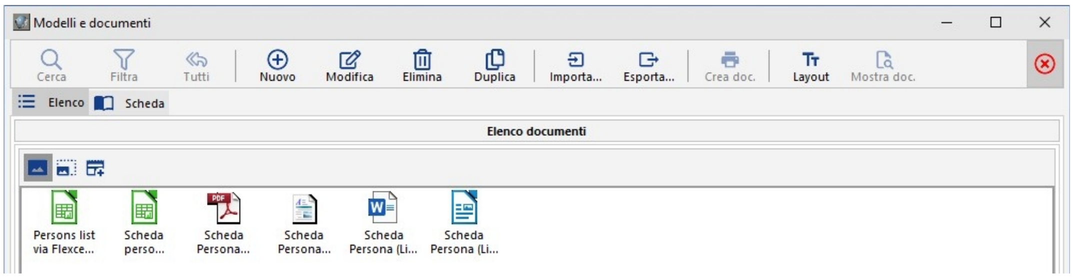
Descrizione
Codice ricerca
Descrizione della ricerca

| Id | Description | Description (Italian) | Character List Attribute |
|---|---|---|---|
| 0000000001 | MASTER OBJECT | | Second Item |
| 0000000002 | ENGLISH DESCRIPTION | | First Item |
| 0000000003 | TEST ENG | | First Item |
| 0000000004 | DX ENGLISH | | Second Item |

| Id | Description | Parent Object | Description |
|---|---|---|---|
| 0000000030 | FIRST DETAIL | 0000000001 | MASTER OBJECT |
| 0000000033 | SECOND DETAIL | 0000000001 | MASTER OBJECT |
| 0000000045 | XXX | 0000000001 | MASTER OBJECT |

Recordcount: 3 TISSampleDetail

It is also possible to use the DbListView component instead of the DbGrid in lists, for example for the GSTD_TISDOCREPORT class, using the parameter **BrowsingMode** That can take on these values:

| | |
|---|---|
| BmDbGrid:<br>List via the DbGrid<br>(default) |  |
| bmDbListViewReport<br>bmDbListViewSmallIcons<br>bmDbListViewLargeIcons:<br>List via DbListView (conDifferent ViewStyles) |  |

Modelli e documenti
Cerca Filtra Tutti Nuovo Modifica Elimina Duplica Importa... Esporta... Crea doc. Layout Mostra doc.
Elenco Scheda
Elenco documenti

| Descrizione | Descrizione | Copie | Formato pagi |
|---|---|---|---|
| Persons list via Flexcel Report | Documento Excel con Flexcel | 1 | A4 |
| Scheda persona (Flexcel) | Documento Excel con Flexcel | 1 | A4 |
| Scheda Persona (FOP-PDF) | Documento PDF tramite FOP | 1 | A4 |
| Scheda Persona (FOP-RTF) | Documenti RTF tramite FOP | 1 | A4 |

Modelli e documenti
Cerca Filtra Tutti Nuovo Modifica Elimina Duplica Importa... Esporta... Crea doc. Layout Mostra doc.
Elenco Scheda
Elenco documenti
Persons list via Flexce...
Scheda perso...
Scheda Persona...
Scheda Persona...
Scheda Persona (Li...
Scheda Persona (Li...

Once the form has been inherited, practically all aspects of an ISF application can be customized, using the appropriate virtual methods made available by the starting classes.

To change these settings there are 2 ways:

1. Define the values in the .ini file in the [FrameWork] section, for example:

EmbeddedExplorerPosition=eepDataRight

This setting becomes the default for every form in the framework.

2. Or indicate these values inside the ISFunction in the field: “Optional parameters”: just write for example

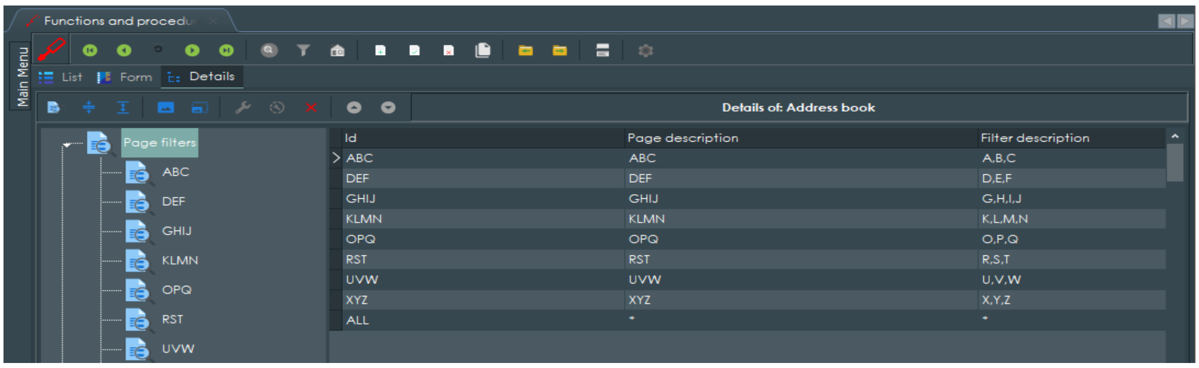
**EmbeddedExplorerPosition=eepDataRight**

## 7.3.7. Organize the detail page into subpages

In InstantSolutions is also possible to define in the functions that the Explorer is divided into different pages, indicating which top level nodes must be visible for each page.

It is necessary to use the function management and define the “DetailPages” container by indicating:

Name of the page, list of allowed containers (separated by ';') and any image to show, as seen in this example:



| Id | Page description | Filter description |
|---|---|---|
| ABC | ABC | A,B,C |
| DEF | DEF | D,E,F |
| GHIJ | GHIJ | G,H,I,J |
| KLMN | KLMN | K,L,M,N |
| OPQ | OPQ | O,P,Q |
| RST | RST | R,S,T |
| UVW | UVW | U,V,W |
| XYZ | XYZ | X,Y,Z |
| ALL | * | * |

### 7.3.8. Customize other form GUI attributes

It is also possible to customize many other graphic aspects of a MultiFramework window, using the "optional parameters" of the TISFunction class, used to start the window. Log in as a user with system privileges to be able to modify the TISFunction classes.

Try modifying the GSTD_TISContactPerson function by indicating in "optional parameters": Font.Color=255 (the decimal number of the color red). Now the “person” type contacts window will appear in red font.

## 7.4. *Use of the “native” form for quick access to data lists*

In some situations where performances are particularly critical, it is also possible to access data by "bypassing" InstantObjects. It is sufficient to use the "native" form provided with the framework.

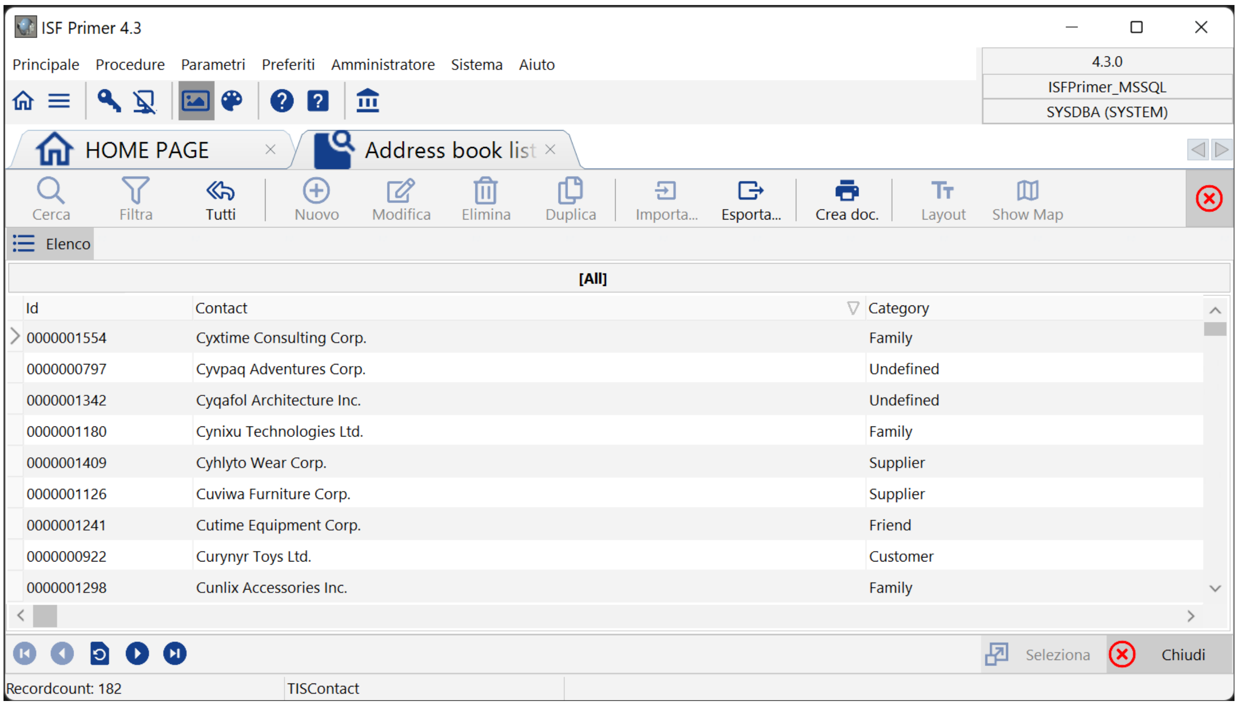
Let's create a new function:

```
Function Id=GSTD_CONTACTS_NATIVE
Description=Address book
Management form=TformNativeBrowse
Optional parameters=ClientDataSet.PacketRecords=100
InstantObjectClass=TISContact
ImageIndex=113
```

Let's then attach it to the menu in the "Address book contacts" section: we will have a form available that directly accesses the data throughFireDAC, using the same connection defined at ISWorkBench level.

In this "native" form it is also possible to specify "Page filters" and "Roles" as is done for the other "functions", but you must pay attention to the fact that the filter expressions must be made in "native SQL" and not in “IQL”.

This is what the "native" form of the example project looks like with the roles and page filters also set.



| Id | Contact | Category |
|---|---|---|
| 0000001554 | Cyxtime Consulting Corp. | Family |
| 0000000797 | Cyvpaq Adventures Corp. | Undefined |
| 0000001342 | Cyqafol Architecture Inc. | Undefined |
| 0000001180 | Cynixu Technologies Ltd. | Family |
| 0000001409 | Cyhlyto Wear Corp. | Supplier |
| 0000001126 | Cuviwa Furniture Corp. | Supplier |
| 0000001241 | Cutime Equipment Corp. | Friend |
| 0000000922 | Curynyr Toys Ltd. | Customer |
| 0000001298 | Cunlix Accessories Inc. | Family |

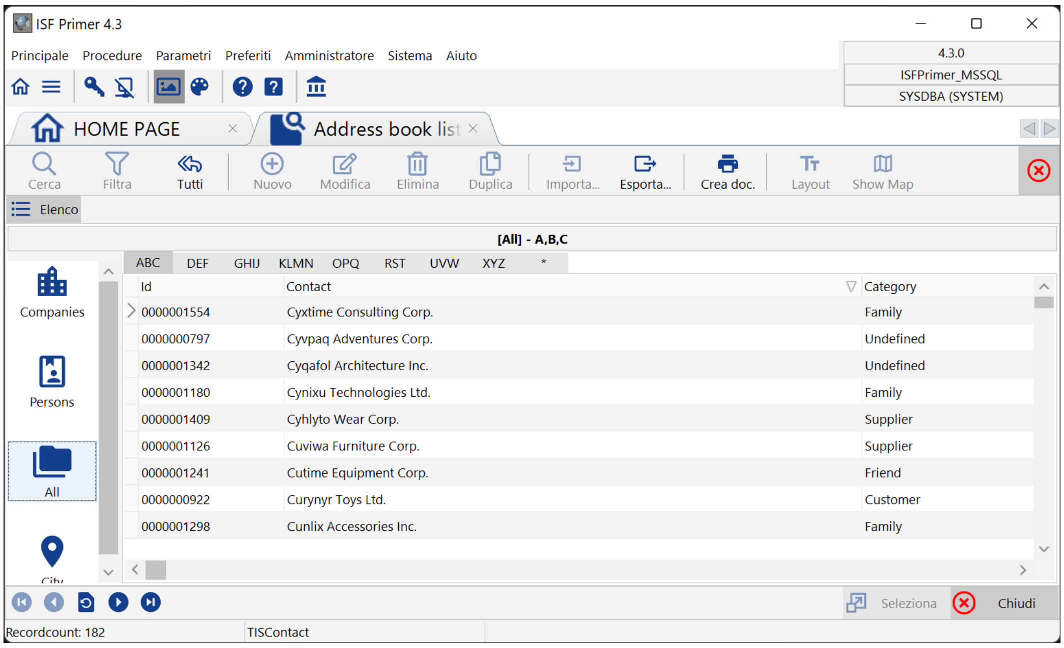
Recordcount: 182    TISContact

## 7.5. *QuickSearch mode*

The QuickSearch mode consists in the possibility of using the predefined and/or generic search classes directly on the "list" form, in the form of different pages (one for each search class set) plus possibly the "special" searches page, i.e. generic ones.

On each page it is possible to either select an already saved search (from the Description combobox) or manually enter the search values and press the Search button.

It is also possible to customize the QuickSearch layout because the panel has the layoutname “QUICK_SEARCH”, so you could also decide that only the most used fields are displayed at the QuickSearch level, or turn them all off and leave only the search selection combobox already memorized.



| Id | Contact | Category |
|---|---|---|
| 0000001554 | Cyxtime Consulting Corp. | Family |
| 0000000797 | Cyvpaq Adventures Corp. | Undefined |
| 0000001342 | Cyqafol Architecture Inc. | Undefined |
| 0000001180 | Cynixu Technologies Ltd. | Family |
| 0000001409 | Cyhlyto Wear Corp. | Supplier |
| 0000001126 | Cuviwa Furniture Corp. | Supplier |
| 0000001241 | Cutime Equipment Corp. | Friend |
| 0000000922 | Curynyr Toys Ltd. | Customer |
| 0000001298 | Cunlix Accessories Inc. | Family |

Recordcount: 182 TISContact

The Quick mechanism can be enabled in both standard and native forms in several ways:

1) global variable: vcAutoQuickSearch is a new global variable that you can set in your application's .ini file or from Delphi code. If it is set to True automatically all search classesthey are also shown as QuickSearch: vcAutoQuickSearchdefaults to False.
2) Via class procedure on the search "source" class: it is possible to implement the IsQuickSearch method to change the default with respect to the value ofvcAutoQuickSearch (e.g.vcAutoQuickSearch is false but I have a search class that I want to use as the default quicksearch).
3) Via class procedure on the "target" class of the search: regardless of the value of vcAutoQuickSearch and the IsQuickSearch method, you can inherit the CanQuickSearch class method and decide whether to enable/disable quick search for all search classes or just some. In the example, the TISContactPerson class enables QuickSearch for all its classes except TISselectId_Desc.

```
class procedure TISContactPerson.CanQuickSearch(
  SearchClass: TInstantObjectClass; var IsQuickSearch: boolean);
begin
  if SearchClass = TISSelectId_Desc then
    IsQuickSearch := False
  else
    IsQuickSearch := True;
end;
```

4) Since the mechanism is linked to the class I may need to turn off QuickSearch on different functions that use the same class, therefore there is also a Property at Form level (ShowQuickSearchFrame) to be able to turn it off by setting it in the Function parameters.

### 7.5.1. Condition the use of the Search and QuickSearch classes

It is possible to condition the visibility and therefore the use of search classes only in certain contexts.

Using the IsQuickSearch and IsFilterSearch methods on the search class and the respective CanQuickSearch and CanFilterSearch on the target classes, you can have different combinations and establish whether a search class is visible only as Quick and IsQuickSearch or IsfilterSearch is used as a filter regardless of the target class. However, if the use is conditioned by the target class, CanQuickSearch and CanFilterSearch are used on the target class.

## *7.6. Customize actions on classes (CustomActions)*

It is possible to avoid inheriting forms if you only need to add actions to certain classes. There is a "Custom Actions" mechanism that can be implemented at class level: the custom action is made available on the actions toolbar, next to the standard ones.

For example, if we want to implement a "send e-mail" action from the TISEMail class, we must proceed to define 4 new methods in the MEMail.pas unit:

```
    class function GetCustomActionCount : integer; override;
    class procedure AssignCustomAction(CustomActionId: Integer; Action: TCBAction); override;
    procedure CustomActionOnExecute(CustomActionId : Integer; Action : TCBAction); override;
    procedure CustomActionOnUpdateDataSet(CustomActionId : Integer; Action : TCBAction;
      ActiveDataSet: TDataSet); override;
```

With these methods you inform the framework that a certain number of CustomActions are available.

It is a good idea to use a constant (e.g. ACTION_SEND_EMAIL = 0) to define an index relating to the CustomAction to be used in all methods like this:

```
class procedure TISEmail.AssignCustomAction(CustomActionId: Integer; Action: TCBAction);
begin
  inherited;
  if CustomActionId = ACTION_SEND_EMAIL then
  begin
    Action.Caption := 'Send E-Mail';
    Action.Hint := 'Send E-Mail to current address';
    Action.ImageIndex := IMG_POSTA;
    Action.ShortCut := ShortCut(VK_F12, []);
  end;
end;
```

In the Update method the action is activated based on some conditions: in this case the CustomActionOnUpdateDataSet method is used which also provides the ActiveDataSet:

```
procedures TISEmail.CustomActionOnUpdateDataSet(CustomActionId: Integer;
Action: TCBaction; ActiveDataSet: TDataSet);
begin
inherited;
if (CustomActionId = ACTION_SEND_EMAIL) then
begin
//Disable the action if there is no email address or if the dataset is being edited
Action.Enabled := (Description <> '') and (ActiveDataSet.State = dsBrowse);
end;
end;
```

Similarly the Execute method will be:

```
procedure TISEmail.CustomActionOnExecute(CustomActionId: Integer; Action: TCBAction);
var
  Msg: string;
begin
  inherited;
  //Apre il client di posta
  if CustomActionId = ACTION_SEND_EMAIL then
  begin
  Msg := 'mailto:'+Description;
    ShellExecute(0, 'open' , pChar(Msg), nil, nil, SW_SHOW );
  end;
end;
```

The framework will display the CustomAction as a button on the toolbar dynamically, whenever the ActiveClass changesor the current object.

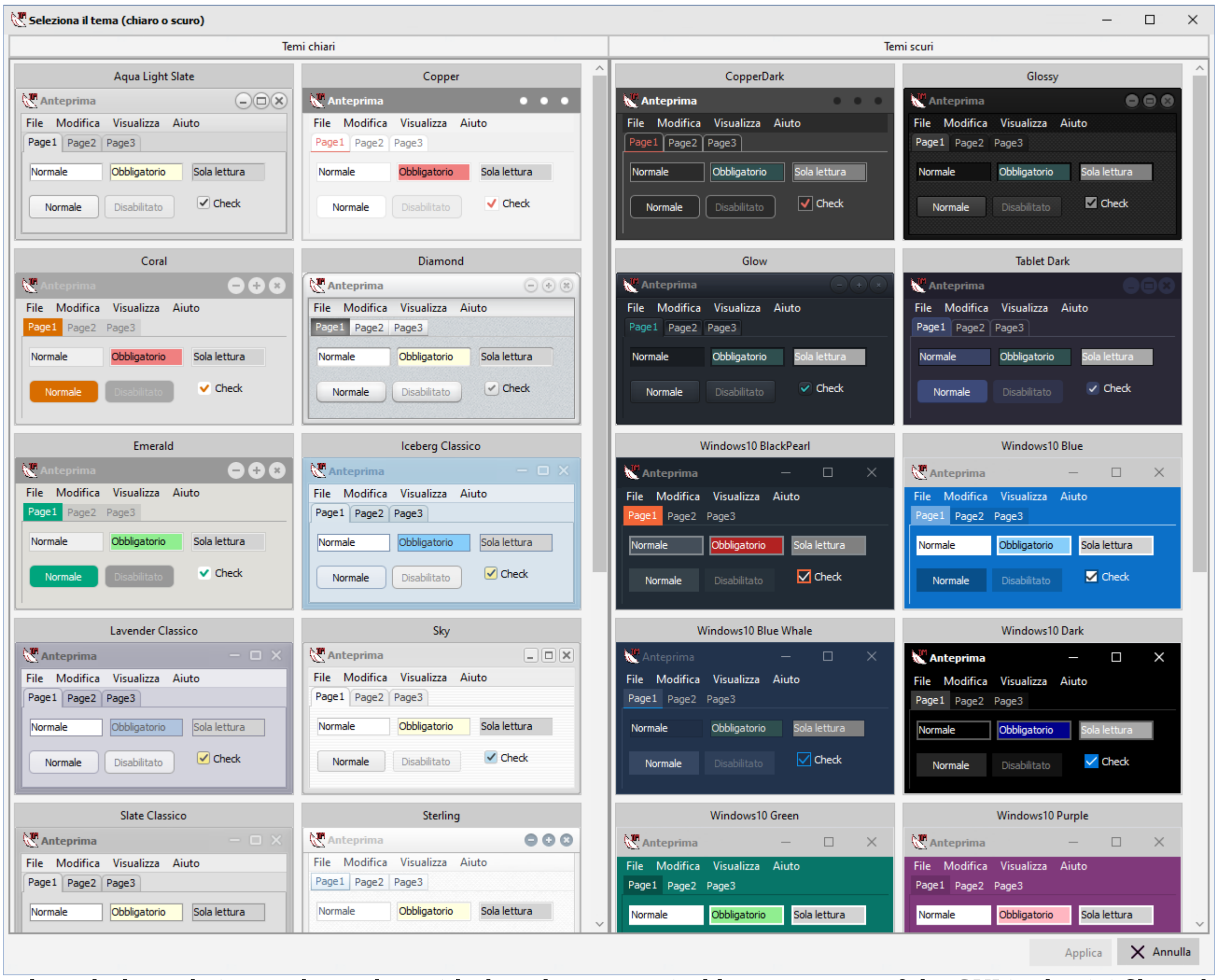
## 7.7. *Management of data locks*

Since version 6.5.6, the possibility of managing data locks has been introduced, using the TISCBObjectLocker class: refer to the document **Migration_from_ISF_6.5.3_to_ISF_6.5.6.pdf** to understand how it works.

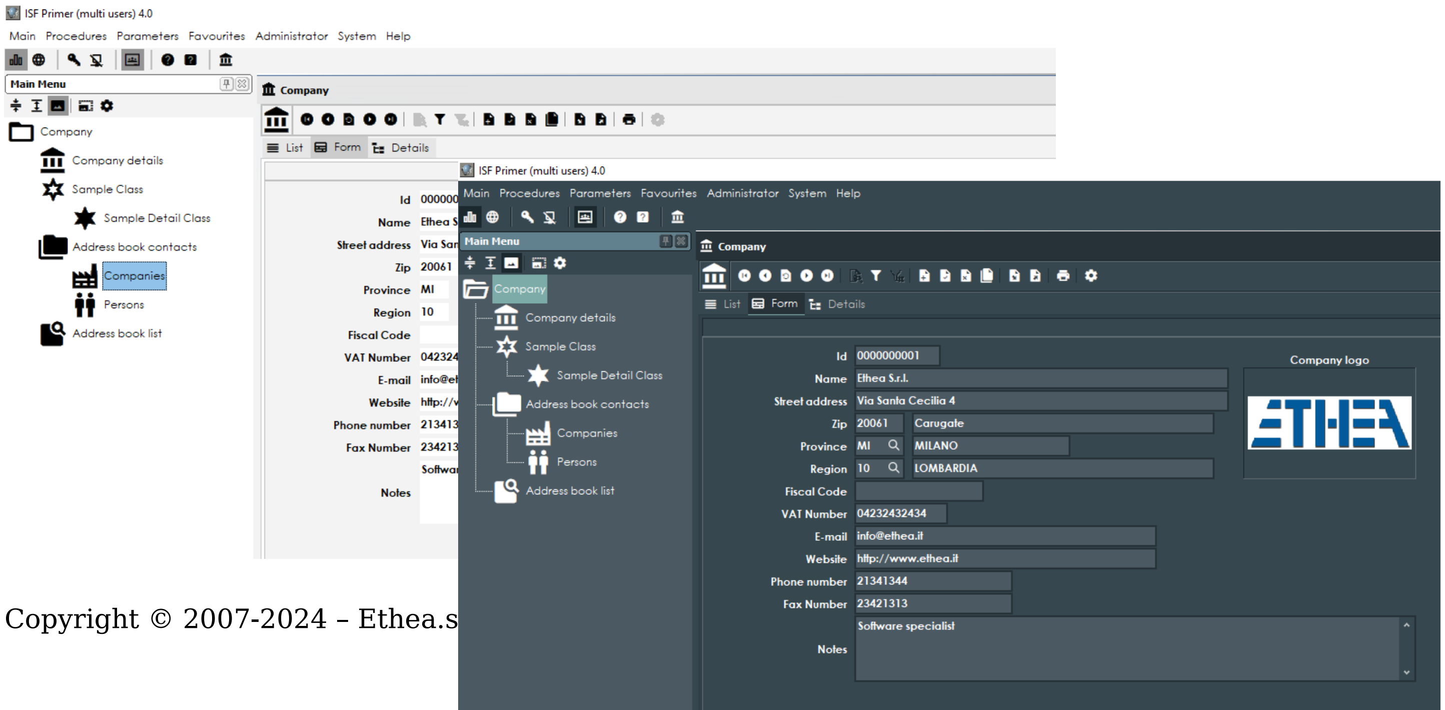
## 7.8. *Using “graphic styles”:light or dark theme*

With version 8 and icon support through the componentAnd SVGIconImageList, you can make the best use of them“graphic styles” for an application.(icon support via Font that was available with ISF7 has been removed).

To take advantage of this new feature in the MultiWindow framework you need to enable the application to use styles (Project/Options/Application/Appereance)and select the styles you want to make available to the end user. The application automatically provides the style selection in the Application Configuration window,or through thefunctionstyle selection available in the main menu bar(color palette button):



Once selected, the style is saved together with the other customizable parameters of the GUI in the .ini file under the VCLStyle item of the [Framework] section. This is what ISFPrimer looks like with 2 different stylesand the use of IconFonts icons that adapt by becoming white or black:



Id 0000000001
Name Ethea S.r.l.
Street address Via Santa Cecilia 4
Zip 20061 Carugate
Province MI MILANO
Region 10 LOMBARDIA
Fiscal Code
VAT Number 04232432434
E-mail info@ethea.it
Website http://www.ethea.it
Phone number 21341344
Fax Number 23421313
Notes Software specialist

## 7.9. *Icon ManagementSVGand IconName in the dfm*

With version8 Full High-DPI multimonitor support has been achieved.

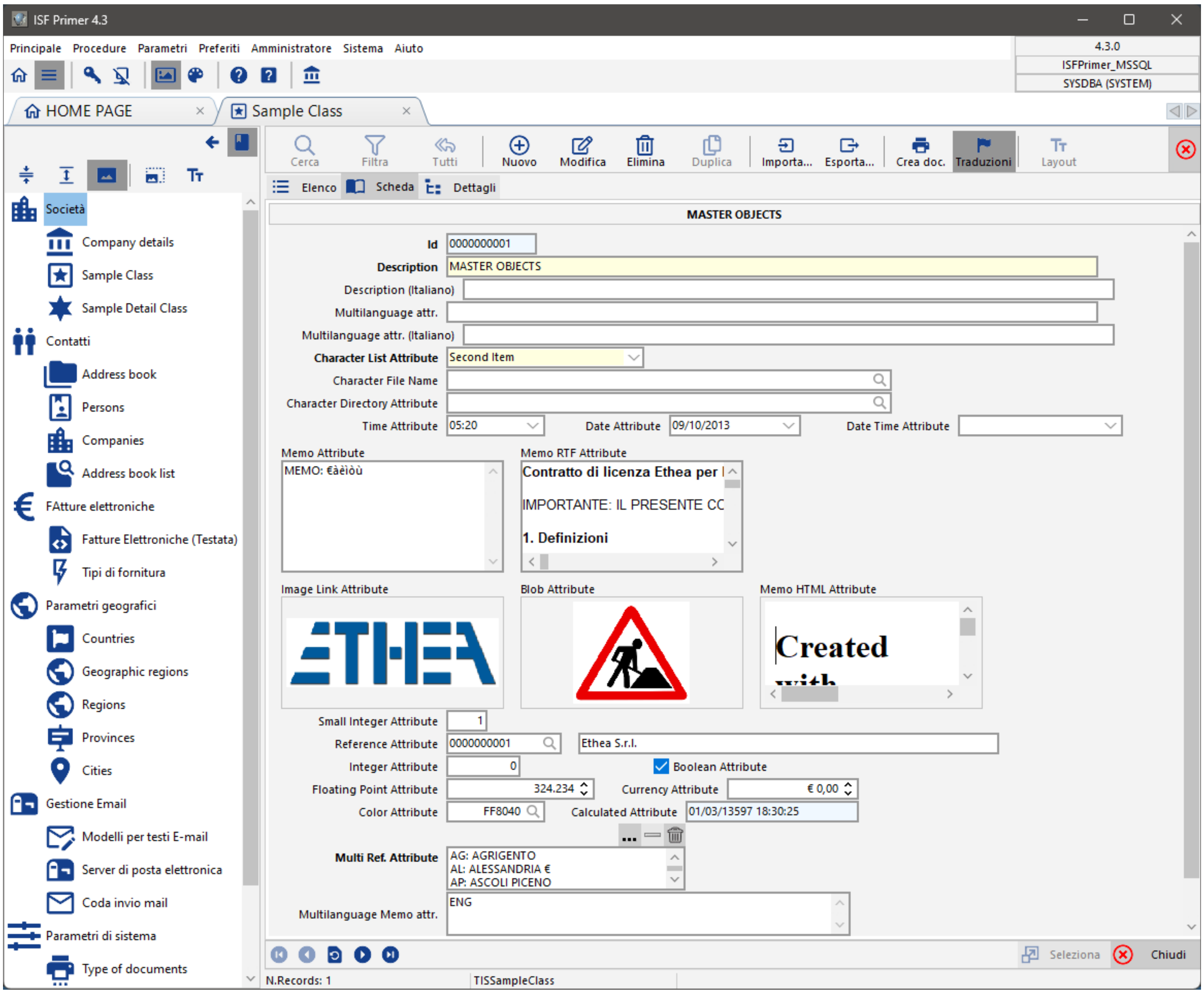
This aspect forced us to change the entire logic of the Icons which are now collected in the TdmISFResources datamodule (DISFEREsources.pas) through a TSVGIconImageCollection component.

***Notice: with version 8 you can no longer use Icon Fonts, therefore support for TiconFontImageList has been removed!***

To allow you to move the forms even on monitors with different DPIs, it was necessary, in all the forms of the framework and the CBLib library, to introduce the concept of "VirtualImageList", which collects only the icons necessary for the Actions of that form from the collection contained in the datamoduleTdmISFResources.

At the same time, all Actions and GUI elements no longer point to Icons by “ImageIndex” but by “ImageName”, and this concept has also been extended to framework classes (e.g. TISMenuItem and TISFunction) that previously used ImageIndex.

**Example of mask with SVG Icons(monochromatic)**



MASTER OBJECTS

| | |
|---|---|
| Id | 0000000001 |
| **Description** | MASTER OBJECTS |
| Description (Italiano) | |
| Multilanguage attr. | |
| Multilanguage attr. (Italiano) | |
| **Character List Attribute** | Second Item |
| Character File Name | |
| Character Directory Attribute | |
| Time Attribute | 05:20 |
| Date Attribute | 09/10/2013 |
| Date Time Attribute | |
| Memo Attribute | MEMO: €àèìòù |
| Memo RTF Attribute | Contratto di licenza Ethea per l<br>IMPORTANTE: IL PRESENTE CC<br>1. Definizioni |
| Image Link Attribute | |
| Blob Attribute | |
| Memo HTML Attribute | Created with |
| Small Integer Attribute | 1 |
| Reference Attribute | 0000000001 Ethea S.r.l. |
| Integer Attribute | 0 |
| Boolean Attribute | ✓ |
| Floating Point Attribute | 324.234 |
| Currency Attribute | € 0,00 |
| Color Attribute | FF8040 |
| Calculated Attribute | 01/03/13597 18:30:25 |
| **Multi Ref. Attribute** | AG: AGRIGENTO<br>AL: ALESSANDRIA €<br>AP: ASCOLI PICENO |
| Multilanguage Memo attr. | ENG |

N.Records: 1 TISSampleClass

## 7.10. *Using the MultiFramework Wizard form*

### 7.10.1. Result fromTFormWizard

The Wizard form present in the MultiFramework is a form designed to manage operations in "sequence". It is necessary to inherit from the TformWizard form and implement some virtual methods such as:

```
    procedure NextPage; override;
    procedure PriorPage; override;
    procedure ExecuteWizard; override;
    procedure OnPageChange; override;
    procedure ExecuteWizard; override;
    procedure ExecuteOK; override;
```

However, creating a Wizard is not a very simple operation and requires a little more development than usual ;-)

The important thing to know is that you must use TInstantSelector and TInstantExposer components to manipulate the objects to be presented on the user interface, in addition to the use of CBXDbMultiEdit and CBXDbGrid which offer all the flexibility of the map and grid editor already seen.

### 7.10.2. An example of a Wizard: the export form

We can analyze the data export form TformExportWizard to see how the basic Wizard form was used.

Each Tabsheet created must have Caption and Hint filled in, which are shown at the top of the wizard.

The NextPage, PriorPage and OnPageChange methods control movements between pages.

The ExecuteWizard method carries out the processing.

This Wizard uses the technologies for exporting a dataset into various formats contained in CbLib (see chapterrelating to data export).

## *7.11. Using the HomePage and Editing Profile Data*

The version7.6differs from the version7.5in some respects:

- Added Windows authentication support to connect to MS-SQL
- Added HomePage management to better organize the menu
- Added form to modify your account data

### 7.11.1. Added account editing form

With version 7.6 it isA menu item “Account/Profile” has been added to the “Main” menu to allow the user to modify some of his account data. For this reason this unit must be added to the project:

```
FAccount in '..\..\..\src\GUIMulti\FAccount.pas' {FormAccount},
```

and copy the file:

```
{ISF7_6}\ISFPrimer\Config\ISFUNCTION\TISFunction.GSTD_TISACCOUNT.1.xml
```

being careful to change the function descriptions to Italian (because in ISFPrimer they are in English).

### 7.11.2. Using a HomePage screen

From version 7.6 a new form of the framework is availablecontained in the unitFHomePage.pas which must be added to the dpr in order to compile:

```
FHomePage in '..\..\..\src\GUIMulti\FHomePage.pas' {fmHomePage},
```

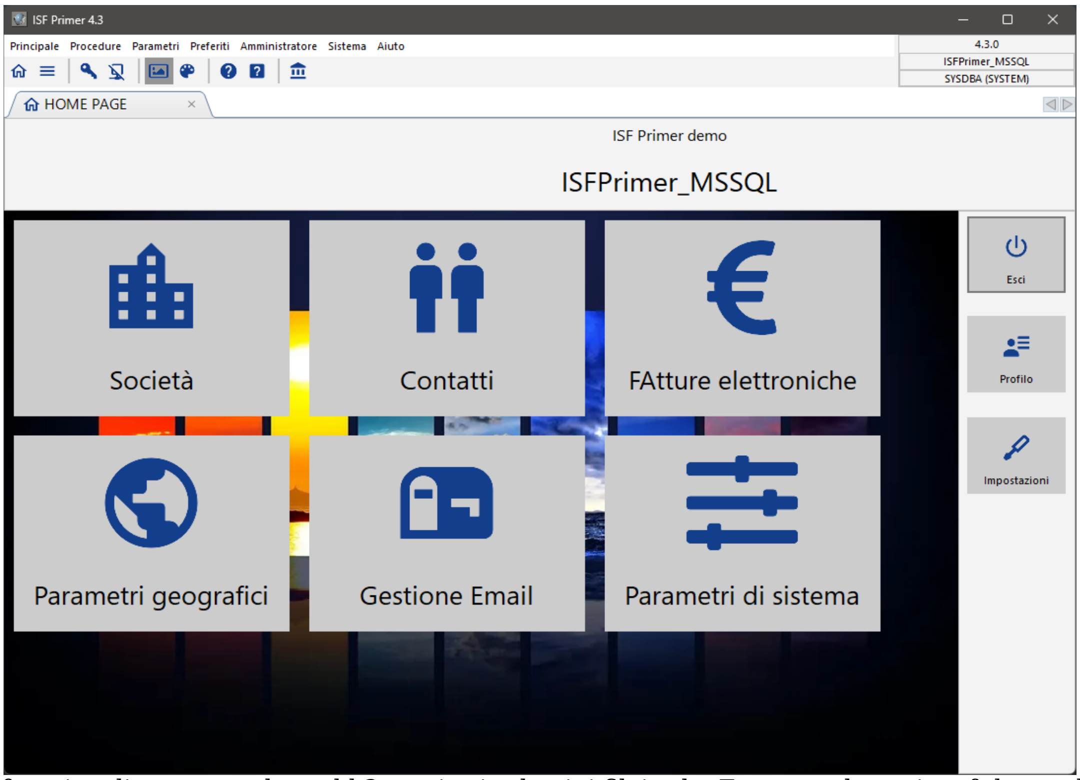
Andthe file must be copied:

```
{ISF7}\ISFPrimer\Config\ISFUNCTION\TISFunction.GSTD_TISHOME.1.xml
```

being careful to change the function descriptions to Italian (because in ISFPrimer they are in English).

Withthis form can show a series of buttons, one for each main "node" of the tree menu: by clicking on them you can open the filtered menu with only the items in this menu:in the example you see the filtered menu when you click on “Company”.

The “Profile” button starts the new form for editing your account data, while “Settings” launches the configuration window (if the user is an administrator).



To enable this functionality you need to add 2 entries in the .ini filein the Framework sectionof the application:

```
[FrameWork]
UseHomePage=1
HomePageButtonWidth=200
```

The size of the buttons must be “calibrated” based on which top-level menu items are in your menu.

The logo to be shown at the top left of the HomePage must be configured in the section:

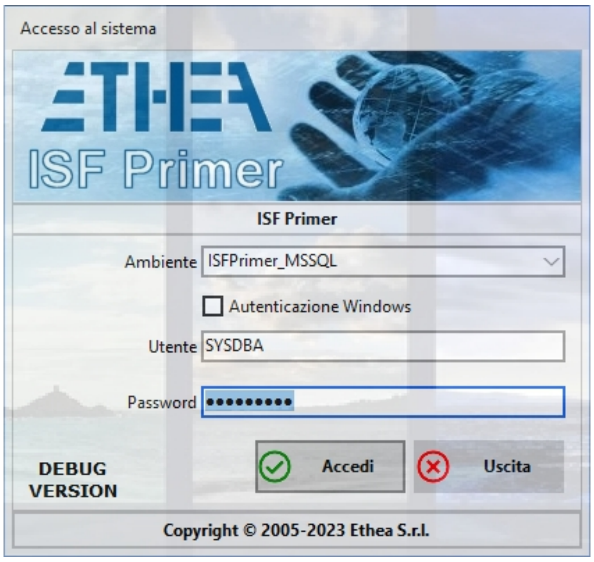
```
[Application]
HomePageBanner=FileName.png
HomePageWallPaper=FileName.png
```

## 7.12. Windows authentication(single sign)

You can now enable Windows authentication when connecting to an MS-SQL database.

The new login form includes a checkbox that is enabled whenyou choose an MS-SQL database: by clicking on “Windows Authentication” you disable the possibility of indicating the user name (which is recovered from the Windows one) and the password.

The system accesses the database with Windows Authentication, after which it still searches for the Domain\User user in the users table, ignoring the password indicated in the table itself.



Ambiente: ISFPrimer_MSSQL

☐ Autenticazione Windows

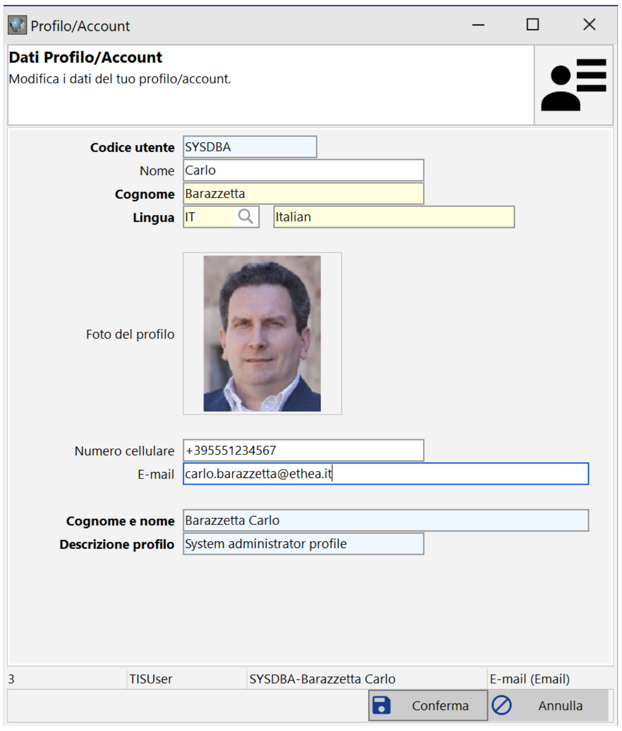
Utente: SYSDBA

Password: •••••••••

## 7.13. New “Account Profile” form

The possibility to modify the data of your user profile such as Name, Surname,Tongue,Photo, telephone number, email.

The same form is activated from the "Profile" button on the HomePage.



**Dati Profilo/Account**
Modifica i dati del tuo profilo/account.

| | |
|---|---|
| **Codice utente** | SYSDBA |
| Nome | Carlo |
| **Cognome** | Barazzetta |
| **Lingua** | IT Italian |
| Foto del profilo | |
| Numero cellulare | +395551234567 |
| E-mail | carlo.barazzetta@ethea.it |
| **Cognome e nome** | Barazzetta Carlo |
| **Descrizione profilo** | System administrator profile |

3 TISUser SYSDBA-Barazzetta Carlo E-mail (Email)

## *7.14. Using modules provided with the Framework*

Within the ISF framework there are also ready-made application modules, with a seriesof classes and libraries useful for widely used generic functionality.

Some examples are the modules:

- MGeographic: contains a series of geographic data classes (cities, provinces, regions, countries, etc.) useful for address management

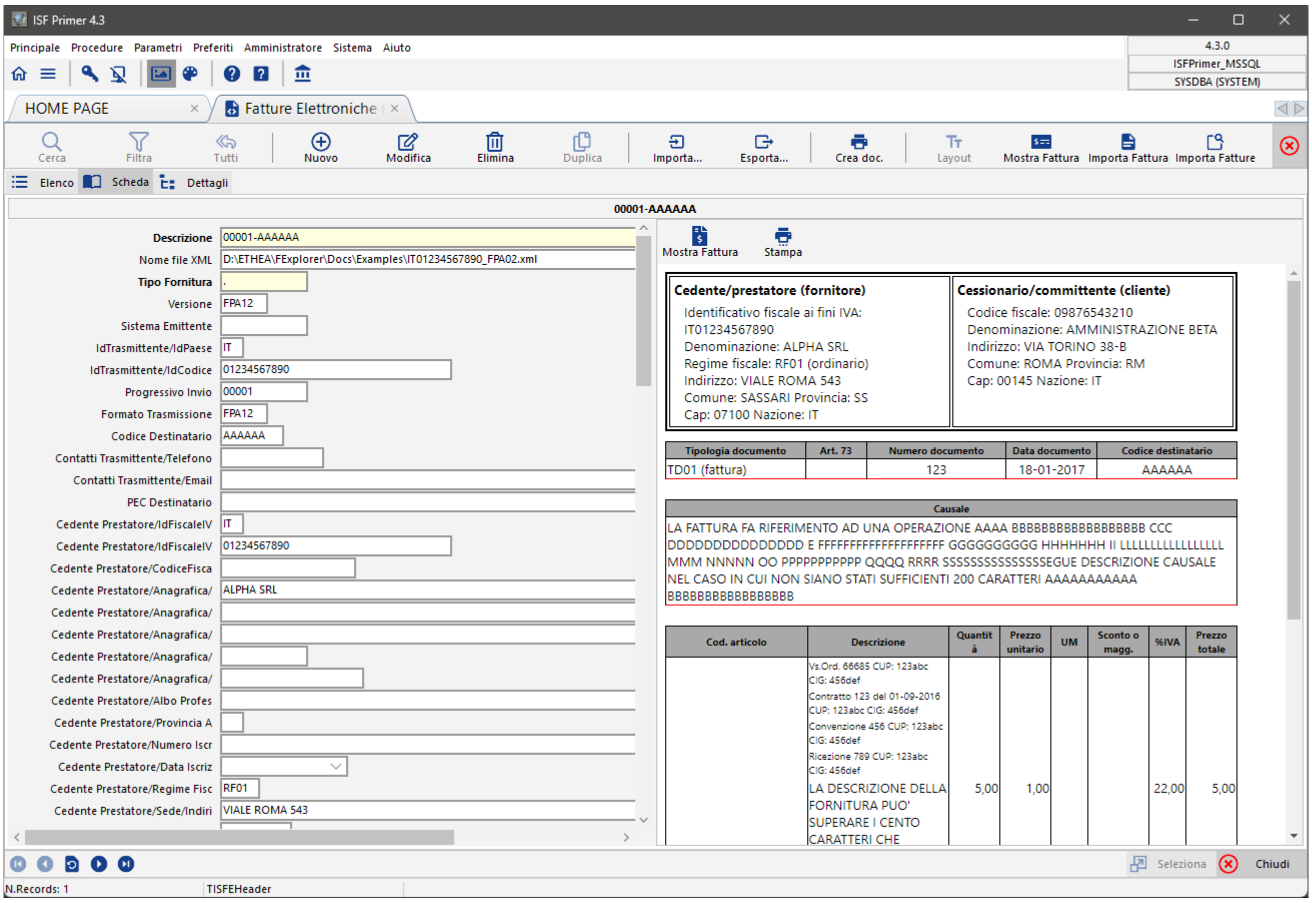
- MMultiLanguage: contains a class useful for translating application messages

- MFatturaElettronica: contains a series of classes and libraries for managing electronic invoices, released with version 8.5.

## *7.15. New FatturaElettronica Module (ISF version 8.5)*

The management of electronic invoice data in Italy now involves various applications that exchange xml or pm7 files containing electronic invoices and attachments. From version 8.5 of ISF, the new "Electronic Invoices" module is available complete with a data import and export library, as well as mapping to InstantObjects classes and display in the "Embedded" Browser thanks to the framework's new form: FNativeBrowseForm.

In the image, the module with the management of the Electronic Invoice integrated into ISFPrimer:


ISF Primer 4.3
Principale Procedure Parametri Preferiti Amministratore Sistema Aiuto
4.3.0
ISFPrimer_MSSQL
SYSDBA (SYSTEM)
HOME PAGE
Fatture Elettroniche
Cerca Filtra Tutti Nuovo Modifica Elimina Duplica Importa... Esporta... Crea doc. Layout Mostra Fattura Importa Fattura Importa Fatture
Elenco Scheda Dettagli
00001-AAAAAA
Mostra Fattura Stampa
Seleziona Chiudi
N.Records: 1 TISFEHeader

| | |
|---|---|
| **Descrizione** | 00001-AAAAAA |
| Nome file XML | D:\ETHEA\FExplorer\Docs\Examples\IT01234567890_FPA02.xml |
| **Tipo Fornitura** | . |
| Versione | FPA12 |
| Sistema Emittente | |
| IdTrasmittente/IdPaese | IT |
| IdTrasmittente/IdCodice | 01234567890 |
| Progressivo Invio | 00001 |
| Formato Trasmissione | FPA12 |
| Codice Destinatario | AAAAAA |
| Contatti Trasmittente/Telefono | |
| Contatti Trasmittente/Email | |
| PEC Destinatario | |
| Cedente Prestatore/IdFiscaleIV | IT |
| Cedente Prestatore/IdFiscaleIV | 01234567890 |
| Cedente Prestatore/CodiceFisca | |
| Cedente Prestatore/Anagrafica/ | ALPHA SRL |
| Cedente Prestatore/Anagrafica/ | |
| Cedente Prestatore/Anagrafica/ | |
| Cedente Prestatore/Anagrafica/ | |
| Cedente Prestatore/Anagrafica/ | |
| Cedente Prestatore/Albo Profes | |
| Cedente Prestatore/Provincia A | |
| Cedente Prestatore/Numero Iscr | |
| Cedente Prestatore/Data Iscriz | |
| Cedente Prestatore/Regime Fisc | RF01 |
| Cedente Prestatore/Sede/Indiri | VIALE ROMA 543 |

| Cedente/prestatore (fornitore) | Cessionario/committente (cliente) |
|---|---|
| Identificativo fiscale ai fini IVA: IT01234567890<br>Denominazione: ALPHA SRL<br>Regime fiscale: RF01 (ordinario)<br>Indirizzo: VIALE ROMA 543<br>Comune: SASSARI Provincia: SS<br>Cap: 07100 Nazione: IT | Codice fiscale: 09876543210<br>Denominazione: AMMINISTRAZIONE BETA<br>Indirizzo: VIA TORINO 38-B<br>Comune: ROMA Provincia: RM<br>Cap: 00145 Nazione: IT |

| Tipologia documento | Art. 73 | Numero documento | Data documento | Codice destinatario |
|---|---|---|---|---|
| TD01 (fattura) | | 123 | 18-01-2017 | AAAAAA |

| Causale |
|---|
| LA FATTURA FA RIFERIMENTO AD UNA OPERAZIONE AAAA BBBBBBBBBBBBBBBBBB CCC DDDDDDDDDDDDDDD E FFFFFFFFFFFFFFFFFFFF GGGGGGGGG HHHHHHH II LLLLLLLLLLLLLLLL MMM NNNNN OO PPPPPPPPPPP QQQQ RRRR SSSSSSSSSSSSSSSEGUE DESCRIZIONE CAUSALE NEL CASO IN CUI NON SIANO STATI SUFFICIENTI 200 CARATTERI AAAAAAAAAAA BBBBBBBBBBBBBBBBB |

| Cod. articolo | Descrizione | Quantità | Prezzo unitario | UM | Sconto o magg. | %IVA | Prezzo totale |
|---|---|---|---|---|---|---|---|
| | Vs.Ord. 66685 CUP: 123abc CIG: 456def<br>Contratto 123 del 01-09-2016 CUP: 123abc CIG: 456def<br>Convenzione 456 CUP: 123abc CIG: 456def<br>Ricezione 789 CUP: 123abc CIG: 456def<br>LA DESCRIZIONE DELLA FORNITURA PUO' SUPERARE I CENTO CARATTERI CHE | 5,00 | 1,00 | | | 22,00 | 5,00 |

To add the "Electronic Invoices" module into your application you need to do it in different areas:

1) Add the module into ISWorkBench

2) Add all model units into the .dpr

3) Add the deployment dlls to the executable folder (libeay32.dll, ssleay32.dll, webview2loader.dll) and install the Microsoft WebViewer (https://developer.microsoft.com/it-it/microsoft-edge/webview2/#download-section)

4) Add the path ..\..\..\ext\FatturaElettronica\Src to the project search paths

5) Add the project classes and units (see section "Electronic Invoice Models" and "Forms and datamodules of the Electronic Invoice library" in the ISFPrimer project file

6) Add the TPathConfiguration class (and the MPathConfiguration unit) to manage "custom" paths.

## *7.16. New ModuleSendMail(ISF version 8.5)*

Another new module added to version 8.5 of the framework concerns support for sending emails, including the management of outgoing queues and templates for defining the "bodies" of email messages.

To add the module "Sending Email" in your application you need to do this in various areas:

1) Add the module into ISWorkBench

2) Add all model units into the .dpr

3) Add the deployment dlls to the executable folder (libeay32.dll, ssleay32.dll)

4) Add the classes and project units (see section "SendMailModels"

# 8. Application configurationand data access

## 8.1. *The configuration file of the application*

Through the file **ApplicationName.ini** of the application it is possible to intervene on various aspects of the framework:

```
[Application]
Wallpaper=Wallpaper.bmp
Splash=Splash.jpg
LoginSplash=Splash.jpg
HelpFile={app}\..\WebHelp\Index.htm
DictionaryPath={app}\..\Dictionary\
CacheDictionaryClasses=1

[MultiLanguage]
;If you have Ethea Translation Tools for ISF applications, set your application languages
Languages=ENG;ITA;ESP
;UpdateRepository=yes

[FrameWork]
ApplicationStyle=TDI
FlatControls=0
NoBorderControls=1
ColorMap=Classic
HideMenuStructure=0
HideActiveFunctions=0
MenuStructureEmbedded=0
MenuStructureExpanded=1
ActiveFunctionsEmbedded=0
SQLMonitorEmbedded=0
HideMainToolbar=0
LargeMenuIcons=1
MenuHotTrack=1
LargeActiveFunctionIcons=0
LargeExplorerIcons=1
AutoEdit=1
AutoQuickSearch=0
AutoApplyChanges=1
EditColor=$FFFFC0
RequiredColor=$80FFFF
ReadOnlyColor=$E0E0E0
DecimalSeparator=,
DateSeparator=/
DateFormat=DMY
LoadFormBGImage=1
GridOddColor=$F0F0F0
ShowHintBalloon=1
DefaultLoadMode='lmFullBurst'
UseReferenceButtons=1
VCLStyle=Windows10
WindowMinWidth=500
WindowMinHeight=400
FontName=Segoe UI
FontSize=9
FontColor=clWindowText
ExcelDefaultExt=.xlsx
FontHeight=-12

[CustomColors]
ColorA=FFFFFF
ColorB=49B9BC
ColorC=2737C9
ColorD=DD2067
ColorE=60AAFD
ColorF=32ABA5
ColorG=4D5259
```

```
ColorH=B5A2B7
ColorI=FF00FF
ColorJ=C47D60
ColorK=23F8F2
ColorL=983D96
ColorM=B8ACF7
ColorN=9D2F55
ColorO=8CFDA3
ColorP=576391

[FOPEngine]
EnginePath={app}\..\..\FOPEngineNew
BatchFileName=fop.bat
TemplateFileName=FOPDocProducerTemplate.xsl
Debug=1

[CodeSite]
;LogPath=$(ISFPrimerLogPath)
;LogFileName=ISFPrimer.csl
;TCPHost=192.168.1.102
;TCPPort=3434
```

Another editable and customizable file is the TISConfiguration class, in the M unitconfigurationwhich must maintain the basic structure as that provided by the framework but can be extended to manage other configuration aspects that can be modified by the user and stored in the database.

## 8.2. *Database access emanagement/protection ofpassword*

To access the database in an ISF application you need to configure 2 different files: one is used by the InstantObjects technology, the other by the ISF DataDictionary technology for checking and updating the database.

### 8.2.1. Data access files

The two files that manage data access in an InstantSolutions application are:

**\Exe\AppName.xml**: Contains the definition of InstantObjects connectionbrokers

connection file example:

```
<TInstantConnectionDefs>
  <TInstantFireDACConnectionDef>
       <Name>ISFPrimer_FB_UTF8</Name>
    <Database>{app}\..\Database\ISFPrimer-UTF8.fdb</Database>
    <Port>-1</Port>
    <Properties></Properties>
    <DriverID>FB</DriverID>
    <UseDelimitedIdents>FALSE</UseDelimitedIdents>
    <User_Name>SYSDBA</User_Name>
    <EncryptedPassword>yekretsam</EncryptedPassword>
       <BlobStreamFormat>sfXML</BlobStreamFormat>
       <UseUnicode>True</UseUnicode>
  </TInstantFireDACConnectionDef>
  <TInstantFireDACConnectionDef>
       <Name>ISFPrimer_FB</Name>
    <Database>{app}\..\Database\ISFPrimer.fdb</Database>
    <Port>-1</Port>
    <Properties></Properties>
    <DriverID>FB</DriverID>
    <UseDelimitedIdents>FALSE</UseDelimitedIdents>
    <User_Name>SYSDBA</User_Name>
    <EncryptedPassword>yekretsam</EncryptedPassword>
       <BlobStreamFormat>sfXML</BlobStreamFormat>
    <DefaultStatementCacheCapacity>100</DefaultStatementCacheCapacity>
    <ReadObjectListWithNoLock>TRUE</ReadObjectListWithNoLock>
  </TInstantFireDACConnectionDef>
  <TInstantFireDACConnectionDef>
       <Name>ISFPrimer_MSSQL</Name>
       <DriverID>MSSQL</DriverID>
       <Server>127.0.0.1, 1434</Server>
       <OsAuthent>True</OsAuthent>
    <Database>ISFPrimer</Database>
```

```
      <BlobStreamFormat>sfXML</BlobStreamFormat>
  </TInstantFireDACConnectionDef>
</TInstantConnectionDefs>
```

**Dictionary\FDConnections.ini**: Contains the definition of data access viaFireDACand the services ofISWorkbench.Notice:if you do not want to integrate the verification/update of the database structure into the application, this file is not necessary.

Connection file example:

```
[ISFPrimer_FB_UTF8]
DriverID=FB
Database={DictionaryPath}\..\Database\ISFPrimer-UTF8.fdb
User_Name=SYSDBA
Password=masterkey
UseAlterTable=Yes
SQLScriptSeparator=;
BlobStreamFormat=sfXML
UseUnicode=True

[ISFPrimer_FB]
DriverID=FB
Database={DictionaryPath}\..\Database\ISFPrimer.fdb
User_Name=SYSDBA
Password=masterkey
UseAlterTable=Yes
SQLScriptSeparator=;
BlobStreamFormat=sfXML

[ISFPrimer_MSSQL]
DriverID=MSSQL
Server=127.0.0.1, 1434
Database=ISFPrimer
OSAuthent=yes
UseAlterTable=Yes
SQLScriptSeparator=GO
```

These two files share access names (Alias) and if you want to exploit native access to the database within the application, they must be configured in parallel and kept aligned.

### 8.2.2. Access password protectionto the database

It is possible to protect and "hide" the database access passwords saved in the NomeApp.xml and FDConnections.ini configuration files.

To avoid writing the password clearly in the AppName.xml file, you can define the EncryptedPassword entry (and not indicate the Password entry). In this case the framework looks for a function to decrypt this password which must be recorded in this way(it is recommendedin the initialization of the MConfiguration unit:

```
//Register a custom procedure for decrypting the access password
ISRegisterDecryptAuthProc(MyDecryptAuthProc);
```

MyDecryptAuthProc must be a procedure defined with these parameters:

```
proceduresMyDecryptAuthProc(var AUserName, APassword: string);
```

The framework will invoke the decryption procedure only if it finds EncryptedPassword defined and does not find Passwordin the configuration xml file.

Within the procedure it is necessary to decrypt the password, which will then be used to connect to the database. It is also possible to decrypt or change the username that accesses the database.The decryption algorithm that you want to adopt is the responsibility of the user of the framework, who will also have to worry about having a tool that allows him to encrypt the password to be indicated inEncryptedPassword.

It is also possible from version 6.6.5do not indicate any user and password in the FDConnections.ini file. In this case the framework is now able to provide the same user and password (possibly decrypted) used to connect to InstantObjects.

### 8.2.3. Protection of user access passwords

From version 6.6.5it is possible to protect the user password which is saved in the DB.

Since by default (for backward compatibility) the password is not protected, it is necessary to register the protection procedure, using the one already made available by the framework itself (which applies the hash to the password) or by providing one of your own.

Insertin the initialization section of MConfiguration:

```
//Register a custom user password protection procedure
//using the one already provided that applies the hash
ISRegisterProtectPasswordProc(ProtectPasswordWithHash);
```

Iswhenthe password comesupdatedtofromuser himself in the“change password” window,that when updated by the administrator in the user management window, the password **he comesprotectto by calling this procedure**.In a situation where passwords are not yet protected, it isYou can protect all your users' passwords by asking them to change their passwordor through the user management form by the administrator. The Framework is able to log in either via plaintext user password or using passwordprotected: in this way the protection of user passwords can be done progressively,without blocking operations.

### 8.2.4. User password validation rules

It is possible to establish specific rules for entering the passwordby the user. You need to register your own procedure that checks the validity of the password and provides the message with the rule it is applying, for example:

```
//Register a custom Password verification procedure
ISRegisterPasswordCheckProc(MyCheckUserPassword);

procedure MyCheckUserPassword(const AUserName, APassword: string;
  out APasswordIsCorrect: Boolean; out ARulesMsg: string);
var
  LLength: Integer;
begin
  LLength := Length(APassword);
  APasswordIsCorrect := LLength >= 8;
  if LLength < 5 then
    ARulesMsg := 'Indicare una password di almeno 8 caratteri'
  else if LLength < 8 then
    ARulesMsg := 'Password debole'
  else
    ARulesMsg := 'Password forte';
end;
```

Notice:the ARulesMsg message is displayed on the change password window to indicate to the user whether he is entering a correct password and possibly also the security level.

## *8.3. Performance and Isolation Level*

### 8.3.1. Isolation “DirtyReads”

With the use of the FireDAC broker it is also possible to control the isolation-level of the connection through the Isolation parameter contained in the connection xml file. By default, if it is not specified, the "unspecified" isolation level is used which at FireDAC level means using the default for the connection type (so in the case of MS-SQL or FireDAC it will be ReadCommitted).

If you set the parameter**isolation**to “xiDirtyReads” the connection will read the updated data on the DB but which are subject to an active transaction.

### 8.3.2. Dirty reads WITH(NOLOCK)

Only for connections with MS-SQL a mixed approach is possible, i.e. maintaining the isolation level at Read-Committed (to avoid reading data not yet updated by another transaction during the update phase) but using "dirty" reads only in case of selecting object keys. To do this you need to enable the parameter **ReadObjectListWithNoLock=True** so that InstantObjects adds the WITH(NOLOCK) keyword on all tables to key reads.

### 8.3.3. Increase performance with Statement Cache

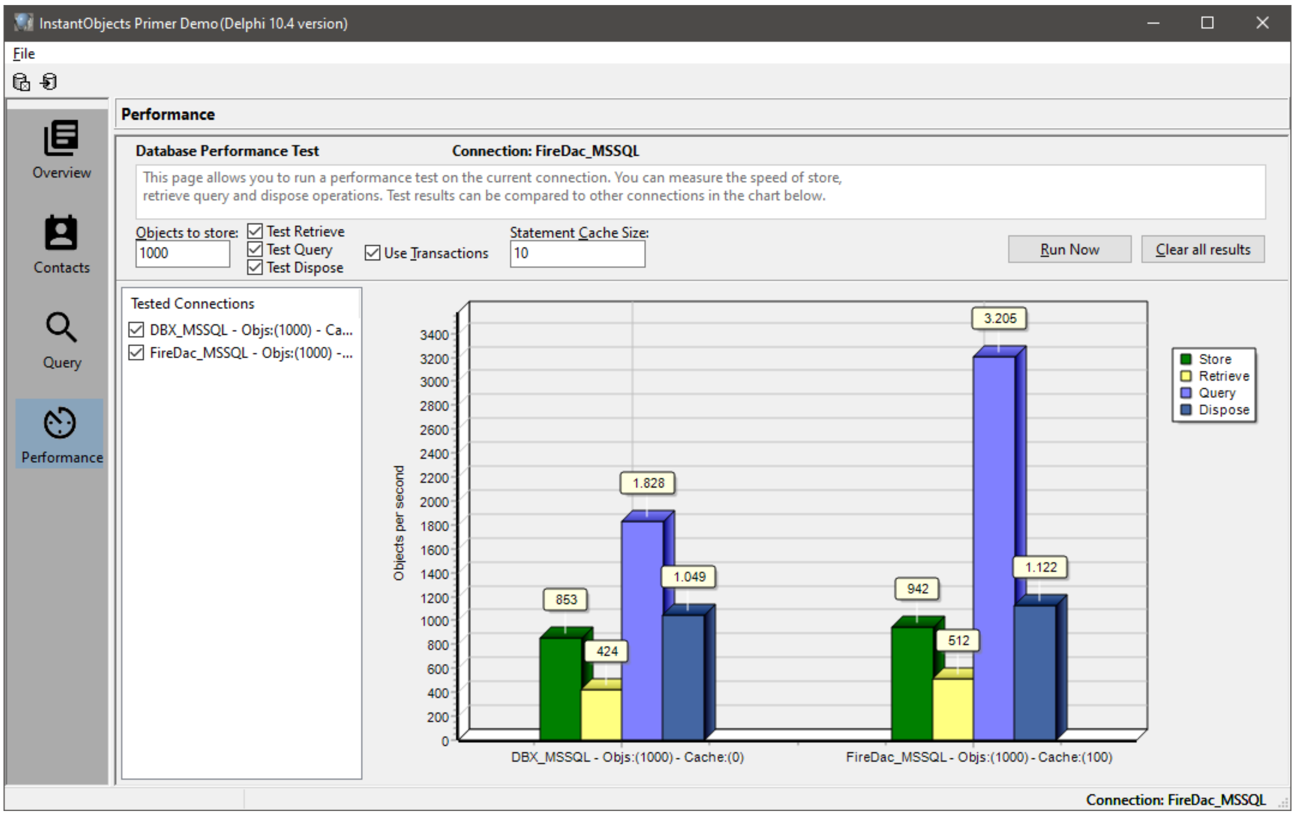
Using the parameter DefaultStatementCacheCapacity=N (where n can be 100 for example), InstantObjects is asked not to "throw away" the data access queries but to reuse them with the same "SQL statements": this allows you to reuse queries already created ( i.e. TFDQuery objects) that have already been "prepared", for which only the parameter values are changed: this technique improves performance, but also introduces a slowdown in the "search" for the identical statement, therefore the value of N must be defined with caution to avoid that the time needed to search for the already used query does not have a greater impact on performance benefits.

### 8.3.4. Measure performance with InstantObject Primer demo

In the InstantObjects folderframework there is a famous demo application called “Primer” that demonstrates the use of InstantObjects. (you can find it under {ISF_Install_Dir}\InstantObjects\Demos\PrimerCrossAndit is not to be confused with ISFPrimer.

In this application there is a useful form for measuring the performance of connections, both to see how the same broker behaves with local or remote connections, and to compare brokers of different technologies (e.g. DbExpress and FireDAC), even if DbExpress is no longer used as an ISF8 Broker, if there is better performance in DbExpress compared to FireDAC connected to the same database it means that the FireDAC broker is not efficient.

*QThis is a sample performance screen:*



InstantObjects Primer Demo (Delphi 10.4 version)

Performance

Database Performance Test — Connection: FireDac_MSSQL

This page allows you to run a performance test on the current connection. You can measure the speed of store, retrieve query and dispose operations. Test results can be compared to other connections in the chart below.

Objects to store: 1000

☑ Test Retrieve
☑ Test Query
☑ Test Dispose
☑ Use Transactions

Statement Cache Size: 10

Tested Connections
☑ DBX_MSSQL - Objs:(1000) - Ca...
☑ FireDac_MSSQL - Objs:(1000) -...

Connection: FireDac_MSSQL

## 8.4. *Database structure update*

It is possible to enable the application to automatically update the database structure when a new version of the product is released. The check is done via the CONFIGURATION.DBVERSION field and the executable version.

To do the automatic check you need to write these lines in the DFunctions datamodule:

```
procedure TdtmDFunctions.BeforeConnect(ConnectionDef : TInstantConnectionDef);
begin
CheckDatabase(ConnectionDef, dtmdMain.dictMain, dtmdMain.LoginEvent, '%d.%d.%d');
end;
```

The last parameter can be '%d.%d or %d.%d.%d: this is used to decide if you want to check only with Major Version and Minor Version or also for Release: if the version information of the The application and those inside the DB differ part of the automatic database update process.

# 9. Other configurations/application aspects

## *9.1. Show a dynamic application wallpaper*

By default the framework loads the application and homepage background configured in the .ini file in the entry:

```
[Application]
Wallpaper=Wallpaper.png
HomePageWallPaper=Wallpaper.png
```

by initializing the global variables vcApSfondo and vcApHomeWallpaper.

It is possible to modify these backgrounds once you have connected to the database, based on the name of the chosen environment or on any value contained in a class of the database to which you are connected.

For example, if you want to change the wallpaper based on the connection name, you need to edit Dfunction.pas in some places.

1) define a private variable to save the default background file name:

```
private
vcLoginBackground: string;
```

2) Add these lines of code into the method:

```
procedures TdtmDFunctions.AfterConnect(ConnectionDef : TinstantConnectionDef);
```

to change the global variables of the application background and HomePage:

```
//Change wallpaper based on database name
vcBackgroundLogin := vcApBackground;
vcApBackground := ExtractFilePath(Application.ExeName)+
ConnectionDef.Name+'.jpg';
vcApHomeWallpaper := vcApBackground;
```

3) Restore wallpaper when logging out:

```
procedures TdtmDFunctions.AfterDisconnect(ConnectionDef : TInstantConnectionDef);
begin
vcApBackground := vcBackgroundLogin;
end;
```

With these simple changes it is possible to customize the background in any way, even by reading values contained in the database.

## *9.2. Show the software license agreement*

It is possible to show the software license agreement within the information form. You need to provide information about the file to show and the title for the window:

```
procedure TdtmDFunctions.AfterConnect(ConnectionDef : TInstantConnectionDef);
var
  LicenseFileName: string;
  LicenseTitle: string;
  FileDate: Integer;
begin
{$IFNDEF ISFCONSOLE}
  //Registration of license file
  LicenseFileName := Format('%s\Licenza_ISFPrimer.rtf',
    [ExtractFilePath(ParamStr(0))+'..\Setup\']);
  if FileExists(LicenseFileName) then
  begin
    FileDate := FileAge(LicenseFileName);
    LicenseTitle := Format('Approved at %s',[DateToStr(FileDateToDateTime(FileDate))]);
    RegisterLicenseFile(LicenseTitle, LicenseFileName);
  end;
{$ENDIF}
end;
```

The program expects the license file to be called License_ISFPrimer.rtf and to be present in the ..\Setup folder relative to the location of the executable.

## *9.3. Role of UmultiAppSpecific*

Each project has a specific unit to define its peculiarities, in addition to the application's .ini file.

This unit is important for the configuration of third-party components supported by ISF but which are paid for, such as ReportBuilder, WPDF and WPView, EDocEngine, etc... In these units there are the uses to use these components, subjected to specific compilation directives.

## 9.4. *Advanced use of CBLib8*

The CBLib library8 supplied with InstantSolutions contains many advanced-use components. Let's see some examples.

### 9.4.1. Use the CBGoogleMapViewer component

The CBGoogleMapViewer component is available into CBLib, and is based on the TEdgeBrorwser and allows you to show a Google map, indicating a location or a route.

**Notice: In order to compile it is necessary to download the component from GetIt: EdgeView2 SDK.**

The new component must be added to a tab page of a custom form, aligned to the right, and the method must be implemented:

```
procedure TfmContacts.RefreshGoogleMap;
var
  Address: string;
  Contact: TISContact;
begin
  if (CurrentObject is TISContact) and (pgctData.ActivePage = tsScheda) then
  begin
    Contact := TISContact(CurrentObject);
    Address := format('%s, %s', [Contact.Address, Contact.City, Contact.Province]);
    MapViewer.GotoAddress(Address);
  end;
end;
```

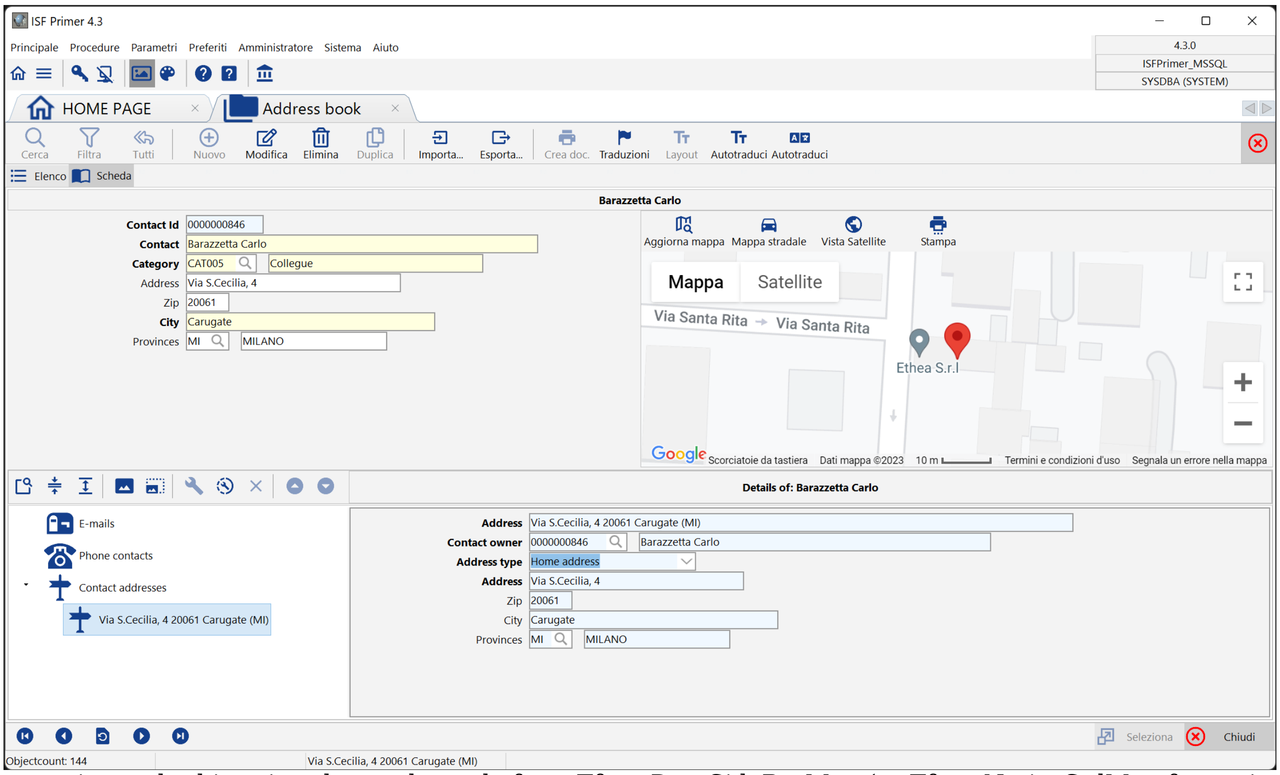
It is sufficient to calculate the complete address and call the "GotoAddress" method of the viewer to show the map synchronized with the data I am viewing: synchronization is guaranteed by having called RefreshGoogleMap in these 2 methods:

```
TfmContacts.ListDataSetAfterScroll and TfmContacts.pgctDataChange(Sender: TObject);
```

The effect obtained is to show the map with the possibility of changing zoom, display type, etc... This technique can be used in any custom form, just associate the data of an address with the component that shows it.

Notice:you need to acquire the license for the Google Maps API: it is a paid service from Google. Once your code has been acquired, it must be registered in the UmultiAppSpecific unit in the initialization section:

```
CBRegisterGoogleMapsApiKey('AIzaSyBNY0ARa4GdRU4LrOK………..');
```


ISF Primer 4.3
Principale Procedure Parametri Preferiti Amministratore Sistema Aiuto
4.3.0
ISFPrimer_MSSQL
SYSDBA (SYSTEM)
HOME PAGE
Address book
Cerca Filtra Tutti Nuovo Modifica Elimina Duplica Importa... Esporta... Crea doc. Traduzioni Layout Autotraduci Autotraduci
Elenco Scheda
Aggiorna mappa Mappa stradale Vista Satellite Stampa
Mappa Satellite
Via Santa Rita
Via Santa Rita
Ethea S.r.l
Google Scorciatoie da tastiera Dati mappa ©2023 10 m Termini e condizioni d'uso Segnala un errore nella mappa
E-mails
Phone contacts
Contact addresses
Via S.Cecilia, 4 20061 Carugate (MI)
Seleziona Chiudi

Barazzetta Carlo

| | |
|---|---|
| Contact Id | 0000000846 |
| Contact | Barazzetta Carlo |
| Category | CAT005 Collegue |
| Address | Via S.Cecilia, 4 |
| Zip | 20061 |
| City | Carugate |
| Provinces | MI MILANO |

Details of: Barazzetta Carlo

| | |
|---|---|
| Address | Via S.Cecilia, 4 20061 Carugate (MI) |
| Contact owner | 0000000846 Barazzetta Carlo |
| Address type | Home address |
| Address | Via S.Cecilia, 4 |
| Zip | 20061 |
| City | Carugate |
| Provinces | MI MILANO |

Objectcount: 144 Via S.Cecilia, 4 20061 Carugate (MI)

It is very easier to do this using the ready-made form TformDataSideBarMap (or TformNativeStdMap for native forms): just inherit from this form and implement a method in the object that returns the address:

```
function TISContact.GetMapAddress: string;
begin
  Result := format('%s, %s', [Address, City, Province]);
end;
```

### 9.4.2. Use the component CBBrowsers

From the version 8.4.0 the CB component is available from InstantSolutions Browsers which is based on TEdgeBrorwser and allows you to show a“Edge” browser inside a form.

**Notice: In order to compile it is necessary to download the component from GetIt: EdgeView2 SDK.**

The new componentit is already available in the FnativeWEBBrowser form and allows you to view any HTML content.

In the base class of CBInstantObject there is now a new object method that returns the HTML content of the object to be shown in the browser: this is very useful for example to show an electronic invoice, as shown in this example.

```
function TISFEHeader.GetBrowserContent(out ACaption, AHint, AImageName: string): string;
begin
  ACaption := SHOW_INVOICE;
  AHint := SHOW_INVOICE_WEB_HINT;
  AImageName := 'file-invoice-dollar-solid';
  if NomeFileXML <> '' then
  begin
    if FileExists(Self.NomeFileXML) then
    begin
      FInvoice := dmFEResources.LoadLegalInvoice(NomeFileXML);
      Result := dmFEResources.RenderInvoiceAsHTML(FInvoice, '');
    end
    else
    Result := Format('<html><p>%s</p></html>',
      [Format(FILE_NOT_FOUND,[NomeFileXML])]);
  end
  else
  Result := Format('<html><p>%s</p></html>', [FE_NOT_ASSIGNED]);
end;
```

# 10. “Console” applications with IinstantSolutions

## *10.1. Console Application Principles*

When it is necessary to carry out operations that do not require a user interface (for the management, for example, of scheduled processes) it is possible with Delphi to develop "console" applications that can be scheduled at system level and started. With InstantSolutions it is possible to use all the business logic developed at class level in a "Console" application following some guidelines:

- never make references to the Application object of the uses Forms
- never call “dialog boxes” of messages, but raise Exceptions or write to a log file
- use command line parameters to manage different operations

In the ISFTemplate\Projects folder there is the ISConsoleTemplate template which shows a small example of how to manage a Console application.

IS provides some basic services in the UConsoleFrameworkLib unit, such as:

- ReadIniFile
- ConsoleLogin
- WriteToLog

Notice: in a CONSOLE application, in order not to include visual components of the VCL, some special compilation directives must be specified:

IO_CONSOLE (to compile InstantObjects without the VCL)

CB_CONSOLE (to compile the CBLib without the VCL)

ISF_CONSOLE (to compile ISF without the VCL)

# 11. Server “REST” with InstantSolutions and M.A.R.S.

## *11.1. REST application principles*

A REST server allows you to provide data “resources” in JSON format (generally).

With ISF and the use of the MARS Curiosity library (by Andrea Magni) it is possible to easily build a REST application based on ISF.

The application must contain all the necessary datamodel classes and it is sufficient to implement a resource base class to be able to access a FireDAC-type InstantConnector to access the objects.

For further details, consult Ethea consultants.

# 12. Web application with IinstantSolutions and KITTO

## *12.1. Use with Kitto*

From version 7.3.7 it is possible to integrate the classes contained in ISWorkBench to generate and maintain the models of a Kitto application.

You must first configure the ISWorkBench.ini file to add this section:

```
[KITTO]
HOMEFOLDER={DictionaryPath}\..\Home
```

in which to specify the Home path of the Kitto application, within which there are the "models".

At the ISWorkBench menu level there is therefore a new Kitto Model generation/update function available to launch.

For more detailson how to build a WEB application with Kitto, please refer to the product information on the sitewww.ethea.it

## *12.2. Automatic generation of Kitto models*

From version 7.3.8 it is also possible to use ISWorkBench to create and update the models of a Kitto application by taking all the information contained in the ISWorkBench classes.

Start the procedure from the ISWorkBench toolbar “Automatic Kitto model generation”.

Notice:if in the Kitto model there are additional fields not present in ISWorkBench they are not removed to avoid eliminating the "expression" type fields. However, this means that if you remove "physical" fields from classes you have to delete them manually in the Kitto models.

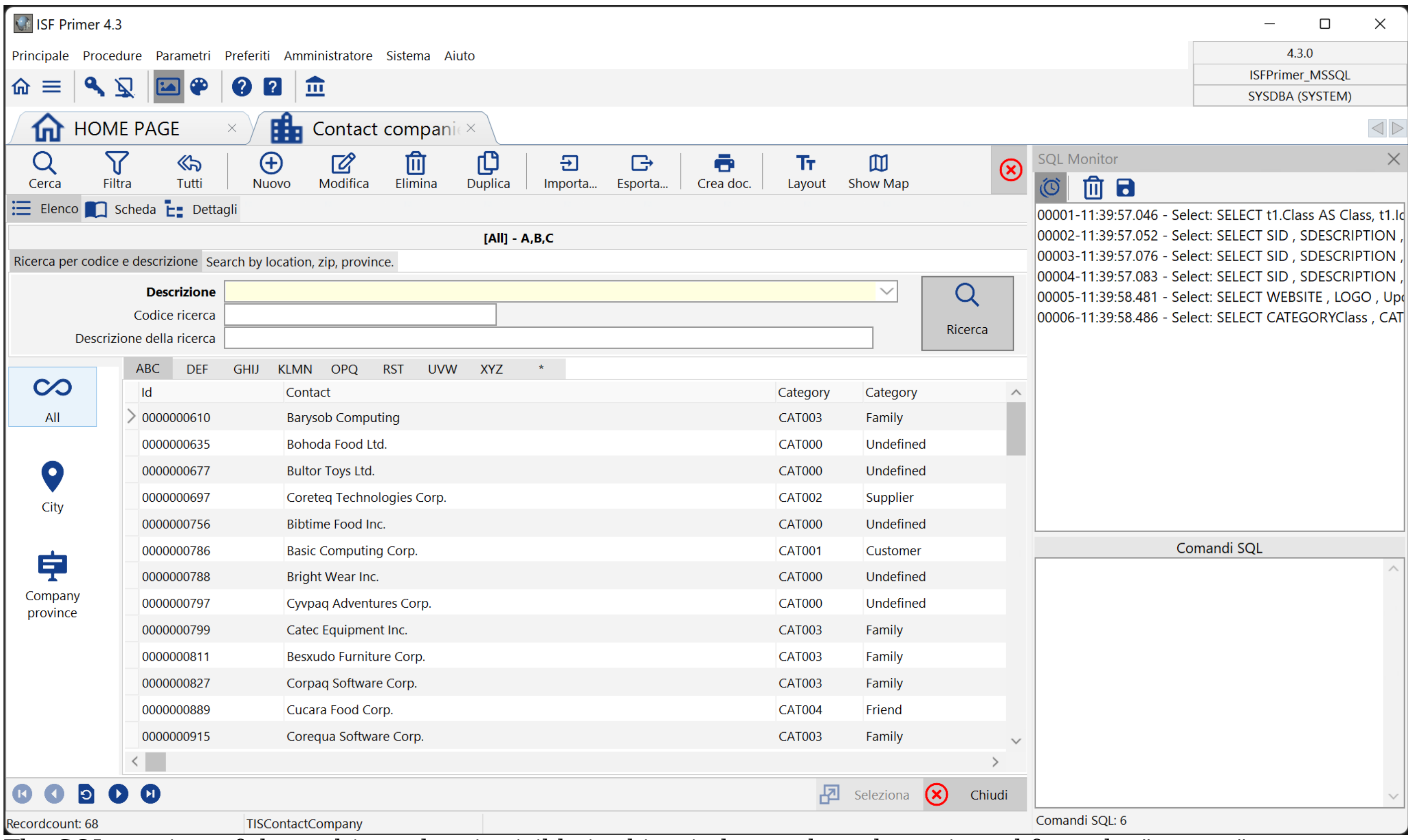
# 13. Application tuning and data integrity verification

An application developed with ISF, which is therefore based on InstantObjects, requires careful tuning, as the "real" SQL statements that are created and launched within the application are "invisible" to the programmer who uses classes and brokers access to data. However, knowing what happens "behind the scenes" is essential to ensuring good application performance.

For this reason, a “SQL monitor” form is available within the multiframework.

## 13.1. *Activation and use of the SQL monitor*

To have this form available, you need to compile the application with the directive: IO_STATEMENT_LOGGING and log in with a user who has system privileges (in the example databases this user is: SYSDBA - password: masterkey). Call up the form from the "system" menu.


ISF Primer 4.3
Principale Procedure Parametri Preferiti Amministratore Sistema Aiuto
4.3.0
ISFPrimer_MSSQL
SYSDBA (SYSTEM)
HOME PAGE
Contact compani
Cerca Filtra Tutti Nuovo Modifica Elimina Duplica Importa... Esporta... Crea doc. Layout Show Map
Elenco Scheda Dettagli
[All] - A,B,C
Ricerca per codice e descrizione
Search by location, zip, province.
Descrizione
Codice ricerca
Descrizione della ricerca
Ricerca
ABC DEF GHIJ KLMN OPQ RST UVW XYZ *
All
City
Company province
Seleziona
Chiudi
Recordcount: 68
TISContactCompany
SQL Monitor
Comandi SQL
Comandi SQL: 6

| Id | Contact | Category | Category |
|---|---|---|---|
| 0000000610 | Barysob Computing | CAT003 | Family |
| 0000000635 | Bohoda Food Ltd. | CAT000 | Undefined |
| 0000000677 | Bultor Toys Ltd. | CAT000 | Undefined |
| 0000000697 | Coreteq Technologies Corp. | CAT002 | Supplier |
| 0000000756 | Bibtime Food Inc. | CAT000 | Undefined |
| 0000000786 | Basic Computing Corp. | CAT001 | Customer |
| 0000000788 | Bright Wear Inc. | CAT000 | Undefined |
| 0000000797 | Cyvpaq Adventures Corp. | CAT000 | Undefined |
| 0000000799 | Catec Equipment Inc. | CAT003 | Family |
| 0000000811 | Besxudo Furniture Corp. | CAT003 | Family |
| 0000000827 | Corpaq Software Corp. | CAT003 | Family |
| 0000000889 | Cucara Food Corp. | CAT004 | Friend |
| 0000000915 | Corequa Software Corp. | CAT003 | Family |

```
00001-11:39:57.046 - Select: SELECT t1.Class AS Class, t1.Id
00002-11:39:57.052 - Select: SELECT SID , SDESCRIPTION ,
00003-11:39:57.076 - Select: SELECT SID , SDESCRIPTION ,
00004-11:39:57.083 - Select: SELECT SID , SDESCRIPTION ,
00005-11:39:58.481 - Select: SELECT WEBSITE , LOGO , Upd
00006-11:39:58.486 - Select: SELECT CATEGORYClass , CAT
```

The SQL monitor of the multitemplate is visible in this window and can be activated from the "system" menu.

Inside the SQL Monitor window you can see how many and which SQL statements are native and translated by the broker you are using while using the application (The counter of queries executed by the application is also visible at the bottom).

As already explained previously, it must be remembered that when executing a query with the IQL command, such as:

```
SELECT * FROM TISContactCompany ORDER BY Description
```

the mechanism that InstantObjects uses is to translate the query into a format compatible with the underlying SQL broker/database:

```
SELECT t1.Class AS Class, t1.Id AS Id FROM CONTACTS t1 WHERE (t1.Class = 'TISContactCompany') ORDER BY t1.DX
```

through which to download a list of keys.

Therefore, at each object "fetch", a "retrieve" of the object is performed which is in turn translated into one or more SQL statements, depending on whether the class has multiple levels of inheritance.

In this example we first take the attributes of the TISContactCompany class:

```
SELECT WEBSITE , UpdateCountFROM COMPANIES WHERE Class = :Class AND Id = :Id
```

Parameters:

```
Class: ftString = TISContactCompany
Id: ftString = 0000001542
```

then the attributes of the TISContact class are taken from which the TISContactCompany:

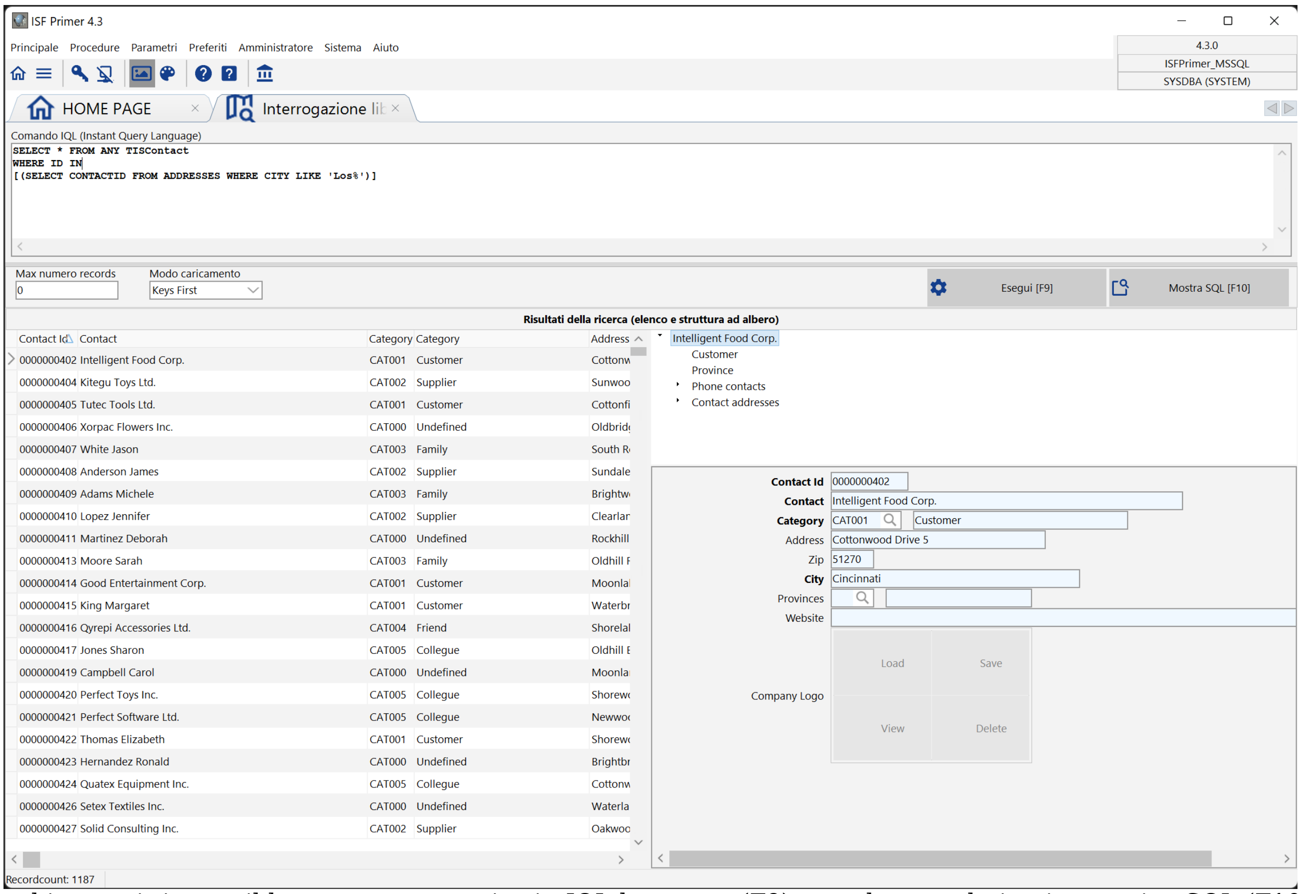
```
SELECT CATEGORYClass, CATEGORYId , ADDRESS , ZIP , CITY , PROVINCEClass, PROVINCEId , PhoneContacts,
Addresses , DX , UPDTIMESTAMP , UpdateCountFROM CONTACTS WHERE Class = :Class AND Id = :Id
Parameters:
Class: ftString = TISContactCompany
Id: ftString = 0000001542
```

...and so on, for each object whose key was downloaded with the first query.

As you can see, InstantObjects performs many queries to access objects, so you might think that the overall performance of the application is very low. In reality, by exploiting the object reference-counting mechanism, when you have to retrieve an object that is already in memory it is no longer reread by the database (unless you force a refresh), so there are better performances.

## 13.2. *Using the “free questions” window*

To learn how to use the IQL language, try to create complex queries and understand how they are translated into SQL format, there is the "free query" function, which can be started from the system menu, available if you log in with a user who has access privileges. system (in the example databases this user is: SYSDBA - password: masterkey).



Comando IQL (Instant Query Language)

```
SELECT * FROM ANY TISContact
WHERE ID IN
[(SELECT CONTACTID FROM ADDRESSES WHERE CITY LIKE 'Los%')]
```

Max numero records: 0
Modo caricamento: Keys First

Risultati della ricerca (elenco e struttura ad albero)

| Contact Id | Contact | Category | Category | Address |
|---|---|---|---|---|
| 0000000402 | Intelligent Food Corp. | CAT001 | Customer | Cottonw |
| 0000000404 | Kitegu Toys Ltd. | CAT002 | Supplier | Sunwoo |
| 0000000405 | Tutec Tools Ltd. | CAT001 | Customer | Cottonfi |
| 0000000406 | Xorpac Flowers Inc. | CAT000 | Undefined | Oldbrid |
| 0000000407 | White Jason | CAT003 | Family | South R |
| 0000000408 | Anderson James | CAT002 | Supplier | Sundale |
| 0000000409 | Adams Michele | CAT003 | Family | Brightw |
| 0000000410 | Lopez Jennifer | CAT002 | Supplier | Clearlar |
| 0000000411 | Martinez Deborah | CAT000 | Undefined | Rockhill |
| 0000000413 | Moore Sarah | CAT003 | Family | Oldhill F |
| 0000000414 | Good Entertainment Corp. | CAT001 | Customer | Moonlal |
| 0000000415 | King Margaret | CAT001 | Customer | Waterbr |
| 0000000416 | Qyrepi Accessories Ltd. | CAT004 | Friend | Shorelal |
| 0000000417 | Jones Sharon | CAT005 | Collegue | Oldhill E |
| 0000000419 | Campbell Carol | CAT000 | Undefined | Moonla |
| 0000000420 | Perfect Toys Inc. | CAT005 | Collegue | Shorew |
| 0000000421 | Perfect Software Ltd. | CAT005 | Collegue | Newwo |
| 0000000422 | Thomas Elizabeth | CAT001 | Customer | Shorew |
| 0000000423 | Hernandez Ronald | CAT000 | Undefined | Brightbr |
| 0000000424 | Quatex Equipment Inc. | CAT005 | Collegue | Cottonw |
| 0000000426 | Setex Textiles Inc. | CAT000 | Undefined | Waterla |
| 0000000427 | Solid Consulting Inc. | CAT002 | Supplier | Oakwoo |

Recordcount: 1187

- Intelligent Food Corp.
  - Customer
  - Province
  - Phone contacts
  - Contact addresses

| | |
|---|---|
| **Contact Id** | 0000000402 |
| **Contact** | Intelligent Food Corp. |
| **Category** | CAT001 Customer |
| Address | Cottonwood Drive 5 |
| Zip | 51270 |
| **City** | Cincinnati |
| Provinces | |
| Website | |
| Company Logo | |

Within this map it is possible to execute queries in IQL language (F9), see the translation into native SQL (F10) and observe/navigate on the results in the part below.

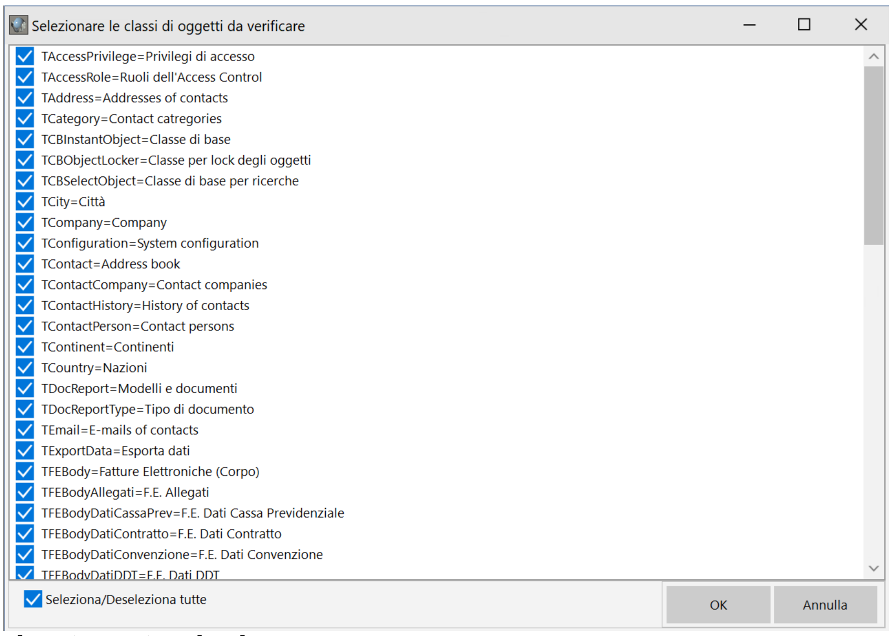
Notice:The translated query can only be shown if it is compiled with the directive:IO_STATEMENT_LOGGING

## 13.3. *Verification of data integrity*

Working for the first time with InstantSolutions (but in particular with InstantObjects) it is possible to write code that generates non-"congruent" objects, in particular "orphan" details of the parent object.

Just forget to do a "store" operation on the parent object, for example after having "hooked" a child object to its parent object, that the database is no longer "intact". It is a good idea, during development, to periodically check that the database istointact using the “Check integrity” function always from the “system” menu (always after logging in with system privileges).

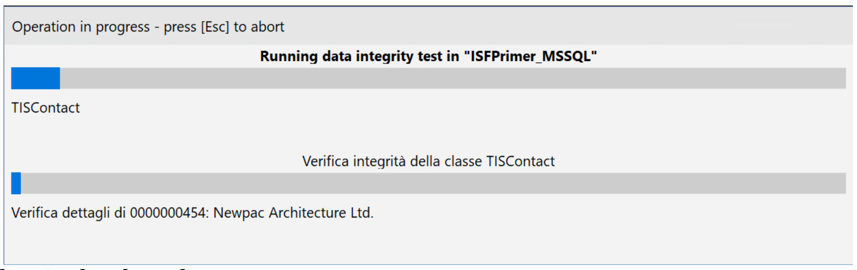
The window for selecting the classes on which to start the data integrity check will appear:


Selezionare le classi di oggetti da verificare

- ✓ TAccessPrivilege=Privilegi di accesso
- ✓ TAccessRole=Ruoli dell'Access Control
- ✓ TAddress=Addresses of contacts
- ✓ TCategory=Contact catregories
- ✓ TCBInstantObject=Classe di base
- ✓ TCBObjectLocker=Classe per lock degli oggetti
- ✓ TCBSelectObject=Classe di base per ricerche
- ✓ TCity=Città
- ✓ TCompany=Company
- ✓ TConfiguration=System configuration
- ✓ TContact=Address book
- ✓ TContactCompany=Contact companies
- ✓ TContactHistory=History of contacts
- ✓ TContactPerson=Contact persons
- ✓ TContinent=Continenti
- ✓ TCountry=Nazioni
- ✓ TDocReport=Modelli e documenti
- ✓ TDocReportType=Tipo di documento
- ✓ TEmail=E-mails of contacts
- ✓ TExportData=Esporta dati
- ✓ TFEBody=Fatture Elettroniche (Corpo)
- ✓ TFEBodyAllegati=F.E. Allegati
- ✓ TFEBodyDatiCassaPrev=F.E. Dati Cassa Previdenziale
- ✓ TFEBodyDatiContratto=F.E. Dati Contratto
- ✓ TFEBodyDatiConvenzione=F.E. Dati Convenzione
- ✓ TFEBodyDatiDDT=F.E. Dati DDT

✓ Seleziona/Deseleziona tutte

Pressing OK starts the data integrity check:


Operation in progress - press [Esc] to abort

**Running data integrity test in "ISFPrimer_MSSQL"**

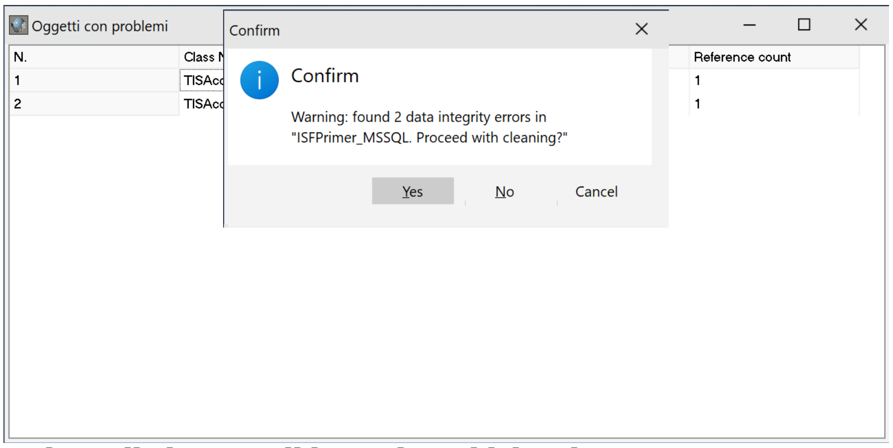
TISContact

Verifica integrità della classe TISContact

Verifica dettagli di 0000000454: Newpac Architecture Ltd.

If there are problems a list is displayed:

By closing this window you can automatically proceed with the removal of these "orphan" objects:


Oggetti con problemi

| N. | Class N | Reference count |
|---|---|---|
| 1 | TISAcc | 1 |
| 2 | TISAcc | 1 |

Confirm

Confirm

Warning: found 2 data integrity errors in "ISFPrimer_MSSQL. Proceed with cleaning?"

**At the end of the procedure all objects will be orphaned<u>deleted</u>!**

Notice: It is possible that an orphan object of a parent object is in fact simply "detached" from the parent object and therefore only needs to be "reattached". Be careful not to delete orphaned objects unless you are sure they are inconsistent data.

# 14. Structure of an ISF project: summary

## 14.1. Organization and location of files

– An ISF project must reside in a folder under {ISFInstallDir}\MyProject.
– An ISF project works through relative paths: here is an example of the defined paths

```
..\..\..\ext\CBLib\src;..\..\..\src\ISFLib\;
..\..\..\InstantObjects\Source\Core;
..\..\..\InstantObjects\Source\Brokers\FireDAC;
..\..\..\InstantObjects\Source\Catalogs\IbFb;*
..\..\..\InstantObjects\Source\Catalogs\MsSql;*
..\..\..\InstantObjects\Source\Catalogs\MySql;*
..\..\..\InstantObjects\Source\Catalogs\Oracle;*
..\..\..\InstantObjects\Source\Brokers\XML;
..\..\..\ext\HTMLViewer\source;
..\..\..\ext\SynEdit\source;
..\..\..\ext\ChromeTabs\Lib;
..\..\..\ext\ChromeTabs\GDIPlus;
..\..\..\ext\IconFontsImageList\Source;
..\..\..\ext\VCLStyleUtils\DDetours\Source;
..\..\..\ext\VCLStyleUtils\Common;
..\..\..\ext\OnGuard\source;
..\..\..\ext\OnGuard\Tools\LicenseManager;
```

*based onL databasesthat you intend to use

## 14.2. Compilation directives

The CBLib8.X supplied with ISF8.X has a double “base language”: Italian and English. By default the packages are in English.

To have the library (packages) and your application with the CbLib in Italian you need to add the CBLIB_ITA directive to the projects. Also pay attention to the forms files contained in the ISF Src folder: there are 2 folders for the framework dfms: ITA_dfm and ENG_dfm: the installer copies the dfms of these forms into the src folder.

If you want to develop an application with another base language, you need to manually copy the correct dfm files into the src folder.

Verify that you have these compilation directives:

```
NO_REPORTBUILDER;
NO_ISF_FLEXCEL;
OPENOFFICE;
FOPENGINE;
REGVERSION;
EFONGUARD;
VCLSTYLEUTILS;
CBLIB_ITA;
VCLSTYLEUTILS;
MAINXP;
```

By default OpenOffice and FOP are enabled (and compile), while ReportBuilderand Flexcelno: if you have ReportBuilderor Flexceljust activate the compile directive.

# 15. Deploy an application

Normally the deployment of an ISF application is similar to any other deployment of a Client/Server application througha Build process that must be done externally to the Delphi IDE ea setup file.

## *15.1. Application deployment: batch compilation*

In InstantSolutions some batch files have been prepared for compiling a project outside the IDE, very useful in the final deployment phase.

Within the IDE you often modify compilation directives (the use of the DEBUG directive is typical), or you compile using run-time packages to speed up compilation... During the deployment phase the compilation mode must always be unique , therefore it is advisable to use these batches.

## *15.2. How to compile an ISF application from the command line*

For this purpose, the examples provided also contain a BAT folder which contains BuildProject.bat: this batch is responsible for compiling the project, in fact it contains very important information: the compilation directives e.g.:

```
set Defines=OPENOFFICE;FOPENGINE;NO_REGVERSION;NO_EFONGUARD;IBFB_SUPPORT;VCLSTYLEUTILS;MAINXP
```

Inside the Project\{DelphiVer} folder there is the batch needed to compile the project for that specific version of Delphi called BuildAllProjects.bat: here is the one for the ISFPrimer projectfor Delphi 11contained in Projects\D11:

```
@echo off
call ..\..\..\bat\DelphiEnvironment
call ..\..\bat\BuildProject ISFPrimer D11SINGLE USER
echo.
echo End compilation with Delphi 11
pauses
```

## *15.3. Using InnoSetup6to create the application installer*

In the ISFPrimer example there is also an example Setup folder created with InnoSetup6: The file is SetupISFPrimer.iss

This setup shows how it is possible to generate different setups based on a condition, in this example “SINGLE USER” or “MULTI USER”.

To compile the setup directly from the InnoSetup IDE you need to set the line:

```
#define ProjectType = "SINGLE USER"
```

removing the semicolon in front.

This variable is then used in some points of the .iss project to influence the creation of the installation package, e.g.:

```
;Files of the "EmbeddedFirebird" group (except Multiuser version)
#if ProjectType = "SINGLE USER"
Source: .\FB2embedded\*; DestDir: {app}\ISF\ISFPrimer\Exe;
Source: .\FB2embedded\intl\*; DestDir: {app}\ISF\ISFPrimer\Exe\intl;
Source: .\FB2embedded\udf\*; DestDir: {app}\ISF\ISFPrimer\Exe\udf;
#endif
```

Notice: it is also necessary to install the data access "clients", such as the SQLNCLI11.dll client to access MS-SQL or FbClient.dll to access Firebird SQL, etc...

## *15.4. Start the Setup generation from the command line*

As for the compilation batches, in the same way it is also possible to launch the setup created with InnoSetup from the command line, through the Build_Setup.bat batch always present in the ISFPrimer\Setup folder.

This batch requires the "SINGLE USER" or "MULTI USER" directive to be passed to it in turn to pass it to InnoSetup in this way:

```
"c:\program files (x86)\Inno Setup6\iscc.exe" %1 /dProjectType="%2"
```

Notice:The batch assumes that Inno Setup6is installed belowc:\program files (x86)\Inno Setup6

This batch is called from the ISFPrimer\Bat\BuildProject.bat build batch like this:

```
echo.
CHOICE /C SN /M "Do you also want to generate the Setup (Y/N)?"
IF ERRORLEVEL == 2 goto Exit
IF ERRORLEVEL == 1 goto BuildSetup
goto Exit
:BuildSetup
call ..\..\Setup\Build_Setup ..\..\Setup\Setup_ISFPrimer.iss %3
pauses
goto exit
```

Since this batch in turn is called from ISFPrimer\Projects\D2007\BuildAllProjects.bat as we have seen above, the result will be to recompile the application from the command line and launch the setup passing the project type (in this example SINGLE USER).

This technique can be used to compile and create a conditional setup for different customers.

## *15.5. Deploy an application in the cloud*

To be able to deploy an InstantSolutions application in a cloud or terminal server environment, wanting to separate the application from the configuration and user data storage files, it is possible to use a fAndatureto manage local "Storage" in different folders on the same server.

### 15.5.1. Deploy steps to follow

- Edit the MyApplication.ini file to add the line in the [FrameWork] section:

```
StoragePath=D:\MyApplicationConfig
```

This folder must contain subfolders, one for each client company, configured to be accessible only to users of that company.

By hypothesis we will have:

```
D:\DossierManagerConfig\ClientCompany
```

In this folder, move the FDConnections.ini file and the MiaApplication.xml file, calling it SocietaClient.xml, and configure the connection parameters to that customer's database inside.

Start the MiaApplication program with a command line parameter (in this example it will be ClientSociety).

Licensing with EfOnGuard:

the license can be floating, i.e. the Registration.ini file can exist in the SocietaClient folder with the maximum number of users who can access it at the same time indicated.

Or it can be named, i.e. there will be many .ini files (e.g. SA.ini, USER.ini) with the single user license of that company.

Application Path Configuration:

From the Administrator / Configuration menu set the following folders:

```
Documents shared folder: {storagepath}\Documents\
Shared images folder: {storagepath}\Images\
Templates shared folder: {app}\..\DocTemplates\
Help folder: {app}\..\Help\
Document import folder: {storagepath}\Import\
Document export folder: {storagepath}\Export\
```

In this way, both the images and the documents produced, as well as those imported or exported, will be saved under the folder D:\MyConfig\SocietaClient folder (which corresponds to the alias {storagepath}).

# 16. Multilingual Support (Database and GUI)

From version 6, ISF multilingual support is provided (already present in version 5), to correctly manage all aspects, in particular the logic of the basic language of the application, introduced by version 5.

All features for developing multilingual applications with InstantSolutions are integrated with the "Ethea Translation Tools" application translation system.

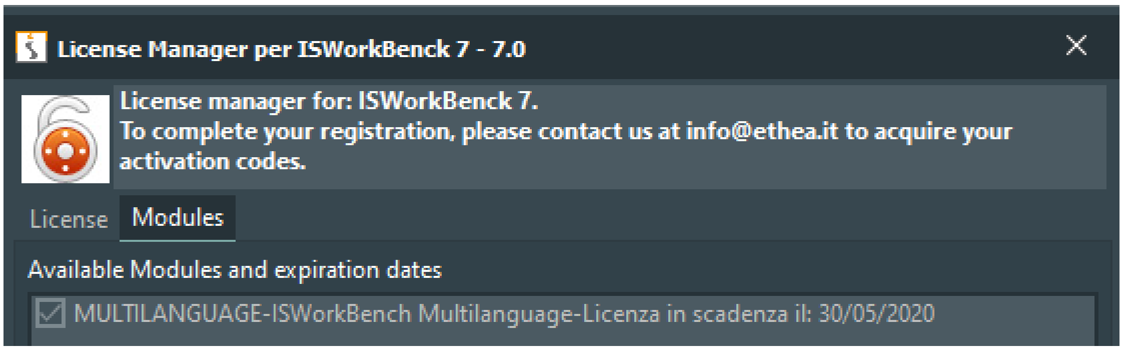
With multilingual support it is possible, within ISWorkbench, to identify some class attributes as "multilingual": it is also possible to provide a description to these attributes in an alternative language to the default one (only if the languages used include Italian and English). This option is only necessary if you intend to have a multilingual GUI. The concept of multilingual refers to 2 complementary but different aspects:

## 16.1. *Data management in language*

The multilingual system allows you to have an application with a base language, plus all the fields in the language.

At the application level it is not necessary to modify any program, map layout or report: the language that drives the attribute reading/writing mechanism is initialized based on the user's language, but it is possible to change it on the fly, based on the context ( e.g. before printing a report in your language).

1) You must request registration of the Multilingual moduleISWorkbench to manage multilingual applications


License Manager per ISWorkBenck 7 - 7.0

License manager for: ISWorkBenck 7.
To complete your registration, please contact us at info@ethea.it to acquire your activation codes.

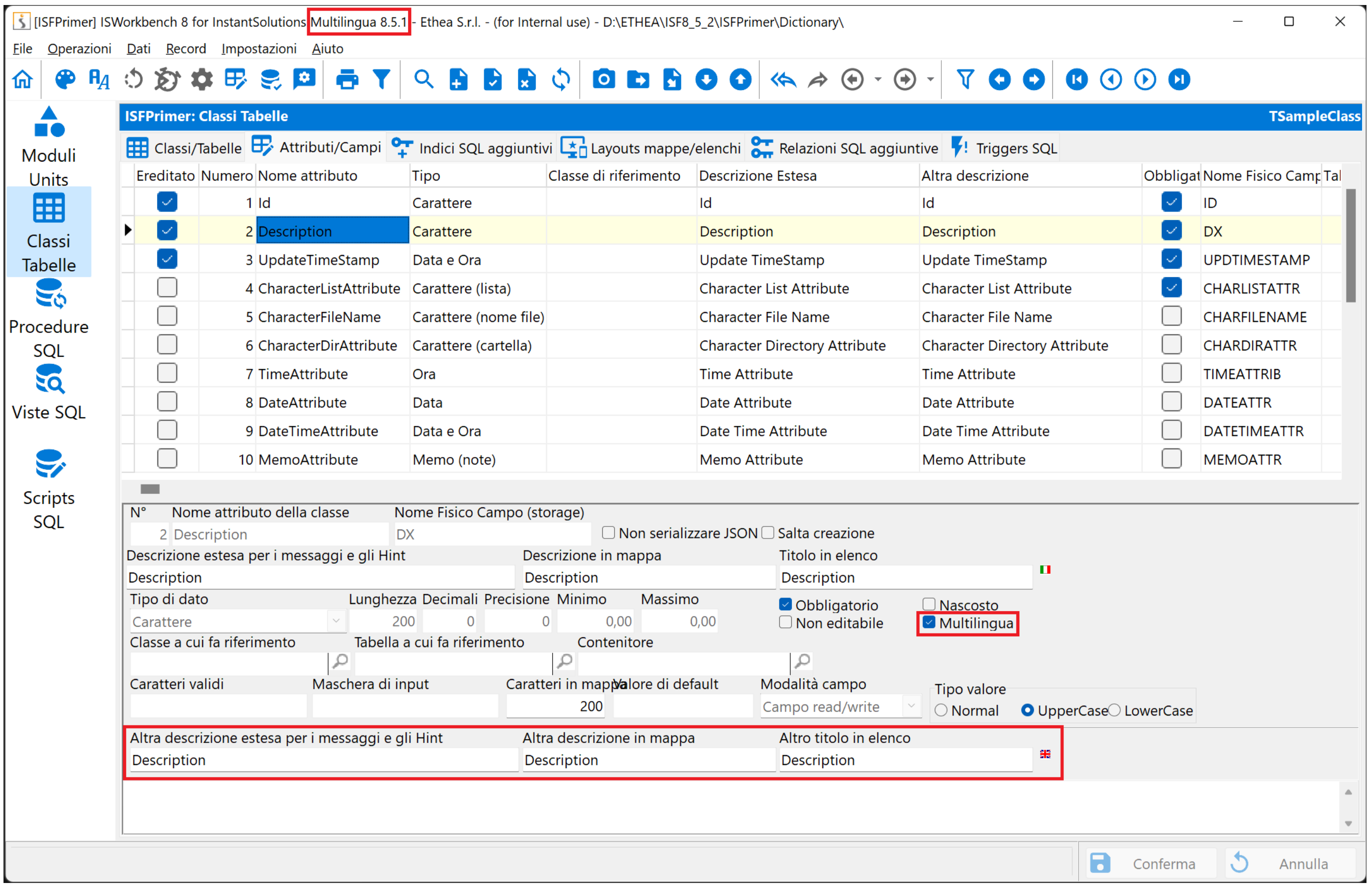
Available Modules and expiration dates

☑ MULTILANGUAGE-ISWorkBench Multilanguage-Licenza in scadenza il: 30/05/2020

```
2)To configure a Multilingual application you need to modify the ISWorkbench.ini file by setting the values
in the MULTILANGUAGE section:
[MULTILANGUAGE]
;If you have Ethea Translation Tools for ISF applications, set your application languages:
;For an English based application
LANGUAGES=ENG;ITA;ESP
```

In this example the application has English as its base language and must also manage data in Italian and Spanish. In the same way, the .ini file of the application must be modified, always in the MULTILANGUAGE section. The base language (variable IODbLanguage) is initialized by default by the CBLIB_ITA compilation directive, which determines the default language with which the application is being developed, but nothing prevents you from having an Italian GUI development language (CBLIB_ITA enabled) and a default database language in English: the parameter to indicate in the application's .INI file is **DB_LANGUAGE.**

```
[MULTILANGUAGE]
LANGUAGES=ENG;ITA;ESP
DB_LANGUAGE=ENG
```


[ISFPrimer] ISWorkbench 8 for InstantSolutions Multilingua 8.5.1 - Ethea S.r.l. - (for Internal use) - D:\ETHEA\ISF8_5_2\ISFPrimer\Dictionary\

| Ereditato | Numero | Nome attributo | Tipo | Classe di riferimento | Descrizione Estesa | Altra descrizione | Obbligat | Nome Fisico Camp | Tal |
|---|---|---|---|---|---|---|---|---|---|
| ✓ | 1 | Id | Carattere | | Id | Id | ✓ | ID | |
| ✓ | 2 | Description | Carattere | | Description | Description | ✓ | DX | |
| ✓ | 3 | UpdateTimeStamp | Data e Ora | | Update TimeStamp | Update TimeStamp | ✓ | UPDTIMESTAMP | |
| ☐ | 4 | CharacterListAttribute | Carattere (lista) | | Character List Attribute | Character List Attribute | ✓ | CHARLISTATTR | |
| ☐ | 5 | CharacterFileName | Carattere (nome file) | | Character File Name | Character File Name | ☐ | CHARFILENAME | |
| ☐ | 6 | CharacterDirAttribute | Carattere (cartella) | | Character Directory Attribute | Character Directory Attribute | ☐ | CHARDIRATTR | |
| ☐ | 7 | TimeAttribute | Ora | | Time Attribute | Time Attribute | ☐ | TIMEATTRIB | |
| ☐ | 8 | DateAttribute | Data | | Date Attribute | Date Attribute | ☐ | DATEATTR | |
| ☐ | 9 | DateTimeAttribute | Data e Ora | | Date Time Attribute | Date Time Attribute | ☐ | DATETIMEATTR | |
| ☐ | 10 | MemoAttribute | Memo (note) | | Memo Attribute | Memo Attribute | ☐ | MEMOATTR | |

N°: 2
Nome attributo della classe: Description
Nome Fisico Campo (storage): DX
☐ Non serializzare JSON ☐ Salta creazione
Descrizione estesa per i messaggi e gli Hint: Description
Descrizione in mappa: Description
Titolo in elenco: Description
Tipo di dato: Carattere
Lunghezza: 200
Decimali: 0
Precisione: 0
Minimo: 0,00
Massimo: 0,00
✓ Obbligatorio
☐ Non editabile
☐ Nascosto
✓ Multilingua
Classe a cui fa riferimento:
Tabella a cui fa riferimento:
Contenitore:
Caratteri validi:
Maschera di input:
Caratteri in mappa: 200
Valore di default:
Modalità campo: Campo read/write
Tipo valore: Normal [✓ UpperCase] LowerCase
Altra descrizione estesa per i messaggi e gli Hint: Description
Altra descrizione in mappa: Description
Altro titolo in elenco: Description

3) Finally, to make an attribute multilingual you need to tick the "Multi-language" item of the attribute.

**ISWorkbench automatically takes care of:**

to) generate InstantObjects classes with an automatic mechanism for reading/writing attributes from the database based on the active language: if it does not find the data, it uses the base field (in the default database language) and indicates it in square brackets.

b) Update the database structure by adding multilingual fields (preceded by the language prefix e.g. ITA_MULTILANGDESC, ENG_MULTILANGDESC, ESP_MULTILANGDESC).

## *16.2. GUI (user interface) language management*

Complementary, but not necessary, is the possibility of having a user interface in the language.

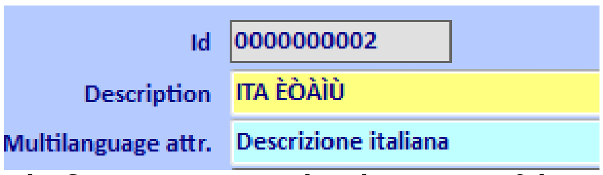
If in ISWorkbench you also define the extended/map/list descriptions in the alternative language to the basic one of the application (the figure shows these values in Italian).

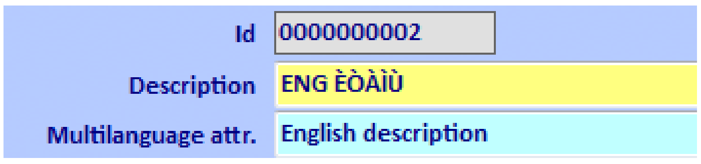
When generating class units, ISWorkbench automatically takes care of generating these descriptions in a way compatible with "Ethea Translation Tools". Then following the guidelines defined in the document **EtheaTranslationTools.pdf** and using the TranslationEditor and the RepositoryEditor it will be possible to create a multilingual user interface of the application.

Notice: If you develop with the same version of ISF on 2 different projects, one multilingual and the other not, when you move from one project to another it is always necessary to regenerate the classes with ISWorkbench, because the Framework units are different and obviously incompatible.

## *16.3. Application Impact: Data Visibility*

As already mentioned, no modifications are necessary at the application level. Given that ISWorkbench generates an intelligent mechanism for reading and writing the multilingual attribute, exploiting the potential of object-oriented programming, the final effect is to have an application that dynamically and transparently displays the data in the language.


Id: 0000000002
Description: ITA ÈÒÀÌÙ
Multilanguage attr.: Descrizione italiana

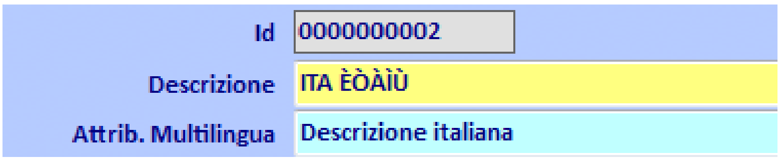

Id: 0000000002
Description: ENG ÈÒÀÌÙ
Multilanguage attr.: English description

In the figure you can see that the content of the same object is different depending on the active language.

In this case the effect is only that of the data in a different language.

*Notice:When the language value is not present in the database, the system still offers the description in the "base language", but in square brackets, indicating that the value still needs to be translated.*

However, if you change the language of the application GUI, the effect is to also automatically have the descriptions of the attributes:



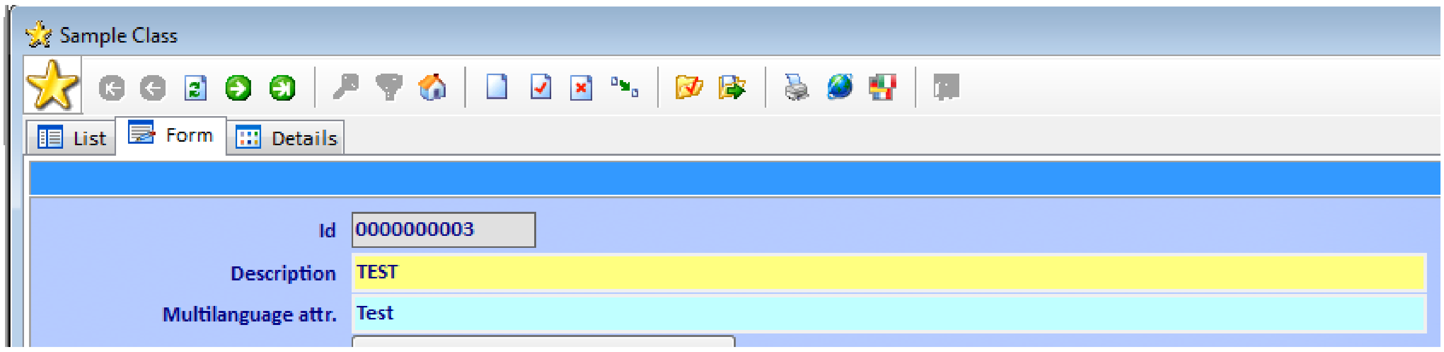
| | |
|---|---|
| Id | 0000000002 |
| Descrizione | ITA ÈÒÀÌÙ |
| Attrib. Multilingua | Descrizione italiana |

Obviously the rest of the application's GUI must also be translated, but since InstantSolutions makes great use of code reuse and an intelligent GUI mechanism at multiple inheritance levels, porting an application's GUI into a language is very fast and simple to do. achieve, through the tools provided by Ethea itself.

## 16.4. New action to show/hide translation attributes

From ver.5.10, a new action is available on all forms of the framework which is automatically enabled if the class you are working on has at least one multilingual attribute.

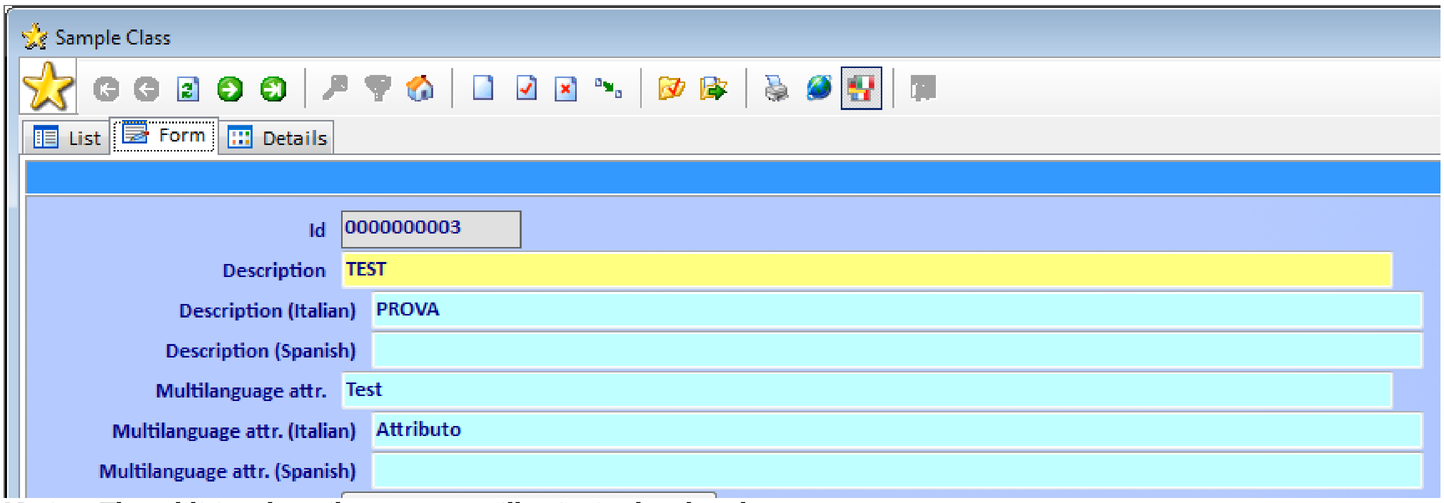
By default the other attributes in the languages to be translated are not visible (as seen in this image):



Sample Class

| | |
|---|---|
| Id | 0000000003 |
| Description | TEST |
| Multilanguage attr. | Test |

If you click on the “Show/Hide Translation attributes” action you will see all the translations:



Sample Class

| | |
|---|---|
| Id | 0000000003 |
| Description | TEST |
| Description (Italian) | PROVA |
| Description (Spanish) | |
| Multilanguage attr. | Test |
| Multilanguage attr. (Italian) | Attributo |
| Multilanguage attr. (Spanish) | |

Notice: The additional attributes must still exist in the class layout:

```
Id|-1|+0|20|-1|"EditCaption"|LeftMiddle|20
Description|-1|+0|20|-1|"EditCaption"|LeftMiddle|20
ITA_Description|-1|+0|22|-1|"EditCaption"|LeftMiddle|20
ENG_Description|-1|+0|22|-1|"EditCaption"|LeftMiddle|20
ESP_Description|-1|+0|22|-1|"EditCaption"|LeftMiddle|20
```

The button works like a checkbox: when it shows the translations it remains pressed, if it is pressed again it returns to its normal position.

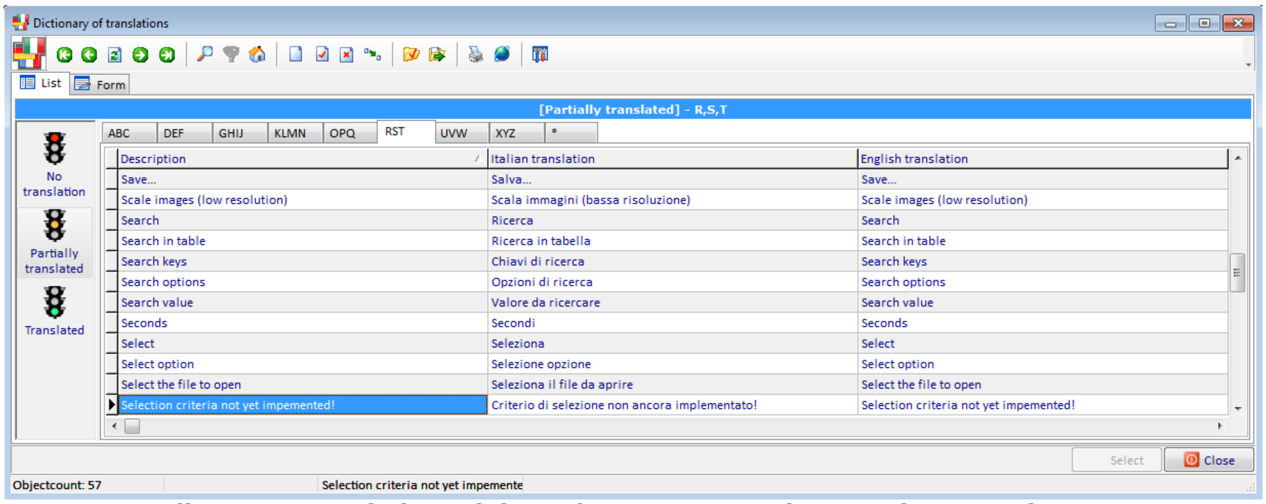
To change the display default (for example, if for a class you want the language attributes to be immediately visible) use the DefaultShowLanguageAttributes class method.

```
class function TISSampleClass.DefaultShowLanguageAttributes: boolean;
begin
  Result := True;
```

```
end;
```

## 16.5. *New Dictionary class for translations*

The TISTranslation class is used to collect a dictionary of strings translated into different languages, to facilitate the insertion of multilingual values. To add the class to your project follow the instructions in the document: [ISF]\Doc\Migration_from_ISF_5.3_to_ISF_5.10.pdf



[Partially translated] - R,S,T

| Description | Italian translation | English translation |
|---|---|---|
| Save... | Salva... | Save... |
| Scale images (low resolution) | Scala immagini (bassa risoluzione) | Scale images (low resolution) |
| Search | Ricerca | Search |
| Search in table | Ricerca in tabella | Search in table |
| Search keys | Chiavi di ricerca | Search keys |
| Search options | Opzioni di ricerca | Search options |
| Search value | Valore da ricercare | Search value |
| Seconds | Secondi | Seconds |
| Select | Seleziona | Select |
| Select option | Selezione opzione | Select option |
| Select the file to open | Seleziona il file da aprire | Select the file to open |
| Selection criteria not yet impemented! | Criterio di selezione non ancora implementato! | Selection criteria not yet impemented! |

Objectcount: 57 Selection criteria not yet impemente

This class automatically registers with the multilingual system to provide a translation mechanism.

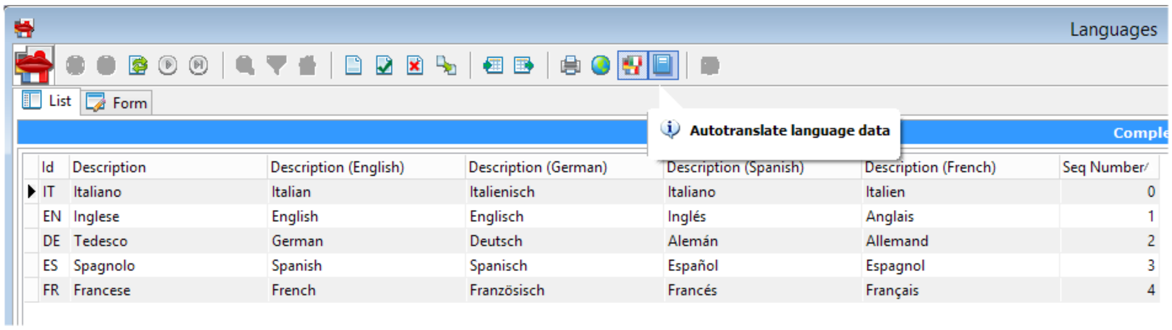
In this way it is possible to manage translations automatically by calling the TranslateValue translation service:

```
procedure TISSampleClass.SetENG_Description(const Value: string);
var
  TranslatedValue: string;
begin
  inherited;
  if (Value <> '') then
  begin
    if TranslateValue(Value,mlEnglish,mlItalian,TranslatedValue) then
      _ITA_Description.Value := TranslatedValue;
    if TranslateValue(Value,mlEnglish,mlSpanish,TranslatedValue) then
      _ESP_Description.Value := TranslatedValue;
  end;
end;
```

The example shows how to attempt automatic translation into Italian and/or Spanish of an English string that has been set in the Description attribute.

## 16.6. *New action to automatically translate data into language*

The action for automatic translation via Microsoft Translation Services is available on all forms of the framework, which is automatically enabled if the class you are working on has at least one multilingual attribute. To enable this action you must also configure the MSTranslation.ini file following the instructions in the document: **{ISF}\Doc\Microsoft_Translation_Services_Configuration.pdf.** Inside the multilingual application, the icon for data translation will appear:



| | Id | Description | Description (English) | Description (German) | Description (Spanish) | Description (French) | Seq Number |
|---|---|---|---|---|---|---|---|
| ▶ | IT | Italiano | Italian | Italienisch | Italiano | Italien | 0 |
| | EN | Inglese | English | Englisch | Inglés | Anglais | 1 |
| | DE | Tedesco | German | Deutsch | Alemán | Allemand | 2 |
| | ES | Spagnolo | Spanish | Spanisch | Español | Espagnol | 3 |
| | FR | Francese | French | Französisch | Francés | Français | 4 |

By clicking on the button you will be asked if you want to translate only the current record or all the records in the table. The translation will only affect data that has not yet been translated. The result of the automatic translation is already visible on the map.

## 16.7. *Dynamically enabling application languages*

While the ISWorkbench.ini file establishes supported languages at the database and class level, the application .ini file can contain a smaller subset of active languages than are available through the global AppLanguages variable. By setting this variable (e.g. by submitting it to the purchase of the language module) it is possible to enable the active languages dynamically.

## 17. CodeSite support

Support for the use of CodeSite for the advanced management of application logging has been introduced since version 6.3.4 of InstantSolutions. You can use both CodeSiteExpress and CodeSiteStudio.

To enable an application it is necessary to add the CODESITE compilation directive: this operation enables the CodeSite Live Viewer by default. InstantSolutions has been enabled to trackTheevents:

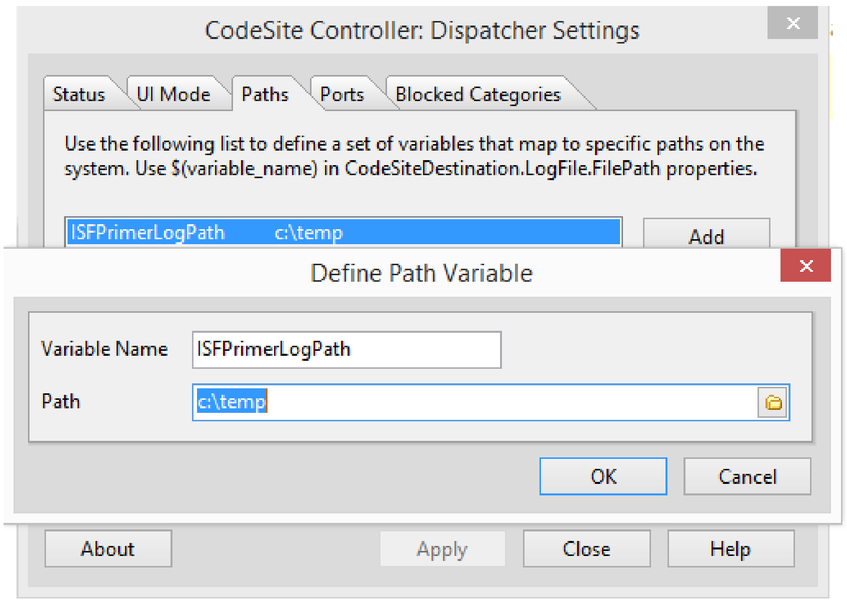
```
AbstractData.SetActiveDataSet (change of active dataset)
TformChild.SetActiveClass (changing active class)
TCBInstantObject.BeforeDispose (before deleting an object)
TCBInstantObject.BeforeStore (before storing an object)
TCBInstantExposer.DoAfterOpen (after opening an exposer in content mode)
TCBInstantSelector.DoBeforeOpen (before opening a selector)
TCBInstantSelector.StoreAllObjects (
TCBAction.Execute (EnterMethod and ExitMethod: execution of an action)
TCBXPageControl.Change (page change of a pagecontrol)
```

These are the most important events that determine the functioning of an application.

If you want to divert CodeSite's output to a file, you need to configure the application's ini file, as in this example:

```
[CodeSite]
LogPath=$(ISFPrimerLogPath)
LogFileName=ISFPrimer.csl
If you want to redirect the CodeSite output to a remote host via TCP-IP also add the lines:
TCPHost=192.168.1.102
TCPPort=3434
```

Notice:if you use the syntax $(ISFPrimerLogPath) it means that you want to delegate the definition of the location of the log file to the Dispatcher, using the CodeSite Controller:


CodeSite Controller: Dispatcher Settings

Use the following list to define a set of variables that map to specific paths on the system. Use $(variable_name) in CodeSiteDestination.LogFile.FilePath properties.

ISFPrimerLogPath c:\temp

Define Path Variable

Variable Name: ISFPrimerLogPath

Path: c:\temp

The new Ethea.CodeSite unit also provides the possibility of filtering the data that you want to send to CodeSite, through 3 events. Ex.

```
    CodeSite_RegisterIncludeFilter(IncludeFieldToLog);
    CodeSite_RegisterIncludeObject(IncludeObjectToLog);

procedure TdmFunction.IncludeFieldToLog(Sender: TObject; const MasterField: string;
  Field: TField; var Include: Boolean);
begin
  Include := SameText(Field.FieldName,FLD_ID) or SameText(Field.FieldName,FLD_DX) or
    (Field.DataType = ftTimeStamp);
end;

procedure TdmFunction.IncludeObjectToLog(Obj: TObject; var Include: Boolean);
begin
  Include := Obj.ClassName = 'TISContantPerson';
end;
```

These 2 examples show how to filter the fields of the dataset or indicate that the only object that must be saved in the log must be of the TISContactPerson class.

# 18. madExcept integration

The advanced exception management system (madExcept) provides support for the resolution of unexpected errors that occur within an application, with the possibility of contacting technical support in a simple and effective way.

Ethea selected it as an effective tool to solve this type of problem and therefore integrated it into InstantSolutions.

## *18.1. Purchase and installation of madExcept*

MadExcept is not bundled with the InstantSolutions development framework. It must therefore be purchased separately and installed. MadExcept is installed inside the Delphi IDE in the “projects” menu.

Website:http://madshi.net/madExceptDescription.htm

## *18.2. Configuration of madExcept in a Delphi application*

To integrate madExcept within an InstantSolutions application, refer to the document: Integrazione_madExcept_in_ISF.pdf.

## *18.3. Benefits of using madExcept*

Using madExcept integrated into InstantSolutions offers several advantages:

1) advanced error handling

2) the possibility of automatically sending detailed reports on the error and the screenshot without the user having to do any type of manual operation

3) the possibility of customizing error management: instead of registering dtmdMain.ShowMadExcept it is possible to register your own personalized procedure.

4) the possibility of having madExcept installed on a single team development machine

# 19. Advanced features: the use of Triggers

With ISF it is possible to integrate the use of Triggers into an application. It must be said that Triggers "escape" the model and business logic implemented through InstantObject with Delphi code, but when they are useful it is possible to exploit this potential of SQL engines.

“Historicization” trigger

Let's imagine we want to keep a history of records of an object, through a trigger mechanism. Every time a record is updated on the SQL server side, the previous data is saved in a "history" table so that the previous data can be traced back.

– To do this we create a SQL table that will contain the historicized data with the same structure as CONTACTS, and we call it STCONTACTS.
– Let's position ourselves on the CONTACTS table and go to the Triggers page.
– We create the new trigger indicating the name ST_CONTACTS and define the trigger Body: each database has a different syntax. In the Demo program we tried the same trigger implemented in 3 different ways, based on the type of database used:

Firebird trigger body

```
CREATE TRIGGER ST_CONTACTS FOR CONTACTS
ACTIVE AFTER UPDATE POSITION 0
AS
begin
  if (OLD.dx <> NEW.DX) then
  begin
    INSERT INTO STCONTACTS(
      CLASS,
      ID,
      UPDATECOUNT,
      DX,
      CATEGORIACLASS,
      CATEGORIAID,
      INDIRIZZO,
      CAP,
      COMUNE,
      PROVINCIACLASS,
      PROVINCIAID,
      ADDRESSES,
      CONTATTITELEFONICI
    )
    VALUES
    (
      OLD.CLASS,
    OLD.ID || '_' || CAST(OLD.UPDATECOUNT AS
VARCHAR(10)),
      OLD.UPDATECOUNT,
      OLD.DX,
      OLD.CATEGORIACLASS,
      OLD.CATEGORIAID,
      OLD.INDIRIZZO,
      OLD.CAP,
      OLD.COMUNE,
      OLD.PROVINCIACLASS,
      OLD.PROVINCIAID,
      OLD.ADDRESSES,
      OLD.CONTATTITELEFONICI
      );
  end
end
```

MS-SQL Trigger Body

```
CREATE TRIGGER ST_CONTACTS ON CONTACTS
FOR UPDATE
AS
BEGIN
  IF UPDATE(DX)
  BEGIN
    INSERT INTO STCONTACTS(
      CLASS,
      ID,
      UPDATECOUNT,
      DX,
      CATEGORIACLASS,
      CATEGORIAID,
      INDIRIZZO,
      CAP,
      COMUNE,
      PROVINCIACLASS,
      PROVINCIAID
      )
      SELECT
        DELETED.CLASS,

DELETED.ID+'_'+CAST(DELETED.UPDATECOUNT AS
VARCHAR(10)),
        DELETED.UPDATECOUNT,
        DELETED.DX,
        DELETED.CATEGORIACLASS,
        DELETED.CATEGORIAID,
        DELETED.INDIRIZZO,
        DELETED.CAP,
        DELETED.COMUNE,
        DELETED.PROVINCIACLASS,
        DELETED.PROVINCIAID
      FROM CONTACTS
      INNER JOIN DELETED ON
        (CONTACTS.CLASS = DELETED.CLASS) AND
        (CONTACTS.ID = DELETED.ID)
  END
END
```

Body Trigger Oracle

```
CREATE TRIGGER ST_CONTACTS
AFTER UPDATE ON CONTACTS
REFERENCING OLD AS OLD NEW AS NEW
FOR EACH ROW
  WHEN (OLD.DX <> NEW.DX)
BEGIN
    INSERT INTO STCONTACTS(
      CLASS,
      ID,
      UPDATECOUNT,
      DX,
      CATEGORIACLASS,
      CATEGORIAID,
      INDIRIZZO,
      CAP,
      COMUNE,
      PROVINCIACLASS,
      PROVINCIAID,
      ADDRESSES,
      CONTATTITELEFONICI
    )
    VALUES
    (
      :OLD.CLASS,
      :OLD.ID || '_' || CAST(:OLD.UPDATECOUNT
AS VARCHAR(10)),
      :OLD.UPDATECOUNT,
      :OLD.DX,
      :OLD.CATEGORIACLASS,
      :OLD.CATEGORIAID,
      :OLD.INDIRIZZO,
      :OLD.CAP,
      :OLD.COMUNE,
      :OLD.PROVINCIACLASS,
      :OLD.PROVINCIAID,
      :OLD.ADDRESSES,
      :OLD.CONTATTITELEFONICI
);
END;
```

This trigger historicizes the contact record only if its description changes.

Note that to avoid primary-key conflicts on the historicization table we have calculated the ID with the ID of the record to be historicized + the UpdateCount field.

# 20. Support for the development of Web-Services

From version 7.2.0 of ISF it is possible to easily develop Web-Services with library support **MARS Curiosity**, an open-source REST server written by **Andrea Magni** (https://github.com/andrea-magni/MARS) and library-based serialization/deserialization support for InstantObjects objects delphi-neon written by **Paolo Rossi** (https://github.com/paolo-rossi/delphi-neon).

- At the InstantObjects level, in addition to the "native" ObjectToJSON support (only for serialization) it is now possible to use**delphi-neon**for both serialization and de-serialization of InstantObjects objects, through a custom serializer/deserializer for InstantObjects:

ISF8\InstantObjects\Source\Core\**Instant.Neon.Serializers.pas**

- At the ISFLib level, a custom serializer/deserializer has been created for CBInstantObjects, the objects used by ISF:

ISF8\src\ISFLib\**CBInstant.Neon.Serializers.pas**

Notice:since this support is specific and requires a series of configuration steps, you must contact Ethea technical staff to include this support in your application.

## 21. Conclusions

Starting to work with ISF may seem very simple, but some complexities should not be underestimated, which can emerge as the level of complexity of an application increases.

First of all, you need to know the basics of InstantObjects which is a very powerful OPF framework but which still has some limitations, in particular in terms of performance on data in "list" format.

Then you have to try to learn to write "clean" code, giving each class the burden of its tasks and above all you have to learn to separate what is "business logic" from what is user interface.

ISF is the ideal tool to avoid immediately having to face the difficulties of a development approach based on an OPF, even if in order to make the best use of it, many other aspects need to be explored in depth which could not be touched upon in this short introductory course.

Ethea staff is always ready and available to intervene on the framework itself when the developers' needs converge towards common needs, with a view to offering a flexible and always updated tool.

**GOOD WORK WITH INSTANTSOLUTIONS!**