Lycia Development Suite

Lycia 2.2 Upgrade Guide

Version 2.2 – April 2014

Revision number 3.00

Querix

Querix Lycia

Lycia Upgrade Guide

Part Number: 011-022-003-014

Elena Krivtsova / Svitlana Ostnek

Technical Writers
Last Updated April 2014

'Lycia' Release Notes

Part Number: 011-006-003-2014

Published by:

Querix (UK) Limited. 50 The Avenue, Southampton, Hampshire,
SO17 1XQ, UK

Publication history:

September 2010: First edition

March 2012: Second edition

April 2014: Updated for Lycia 2.2

Last Updated:

April 2014

Documentation written by:

Elena Krivtsova / Svitlana Ostnek / Olga Gusarenko / Daria Sakhno

Notices:

The information contained within this document is subject to change without notice. It is a stable document and may be used as reference material or cited from another document. If you find any errata or omissions in the documentation, please, report errors to documentation@querix.com

Table of Contents

About This Document

This document is intended for use, primarily, by the 4GL application developers as it is assumed, that the reader has basic knowledge of Informix 4GL, the databases and the operating systems on which they function.

This document describes new features and functionalities implemented for the release of the Lycia Development Suite for the corresponding product version. For general information on getting started, installation requirements, and other issues, please consult 'Lycia Getting Started Guide' and 'Lycia Developers Guide'.

This guide mostly covers the issues which differ Lycia from Lycia 2.2. It does not focus on the difference between Lycia 2.2 and previous product versions released before Lycia I.

What's new in Lycia?

Lycia 2.2 presents a number of additional modules and functionality, if compared to its previous versions. The basic functionality - the eclipse-based IDE and the compiler functionality remain largely the same though the compiler has been refined and perfected due to the fact, that the Lycia compiler created P-Code using to a different improved protocol.

Lycia 2.2 is not only a compiler, it is a family of products comprised together for better performance and widened functionality.

The new products and add-ons which go together with Lycia are:

- **Apache Tomcat instance:** shipped together with Lycia 2.2. to fulfil a number of auxiliary tasks
- **Lycia DC .HTML** thin client
- **LyciaWeb Client:** a web-client to run 4GL applications in web environment; combines Ajax and HTML5 technologies
- **Lycia Theme Designer.NET:** a new graphical tool to manage 4GL applications appearance and functionality
- **Graphical Form Designer**: a completely new designer developed on the base of Window Builder. It is created to make the process of GUI application development easier and less time-consuming. Graphical Form Designer works with the new *.fm2* graphical forms
- **Major 4GL extensions:** Java and XML interface, new classes and methods
- **BIRT® Report Engine & Design Support:** the Eclipse-based API designed to create and manage graphical interactive reports

- The possibility to manipulate elements at runtime by means of the **graphical API functionality**
- The possibility of **multiple database connections**
- **Genero ® compatibility** and Import/Migration Utilities
- **Web API** provides you with enhanced approach to creation, development and deployment of modern web services using Querix 4GL.

Lycia Form Designer

Lycia 2.2 includes a completely new powerful form designer, based on the Window Builder. It provides the user with a wide range of tools and possibilities, that allow to create graphical forms and adjust them to the most complicated tasks and layout preferences.

The Form Designer is able to work with forms of the three following formats:

1. the new graphical **.fm2** forms,
2. text **.per** format forms (legacy forms),
3. graphical **.4fm** format forms.

The forms available in the previous versions of the product (.per and .4fm format files) will be imported and converted to remove their limitation without loosing any properties or functionality. An .fm2 form is an XML-based one. The Designer allows to manipulate the XML code via a graphical designer, that includes a palette with a variety of tools, as well as manually via XML Source Editor.

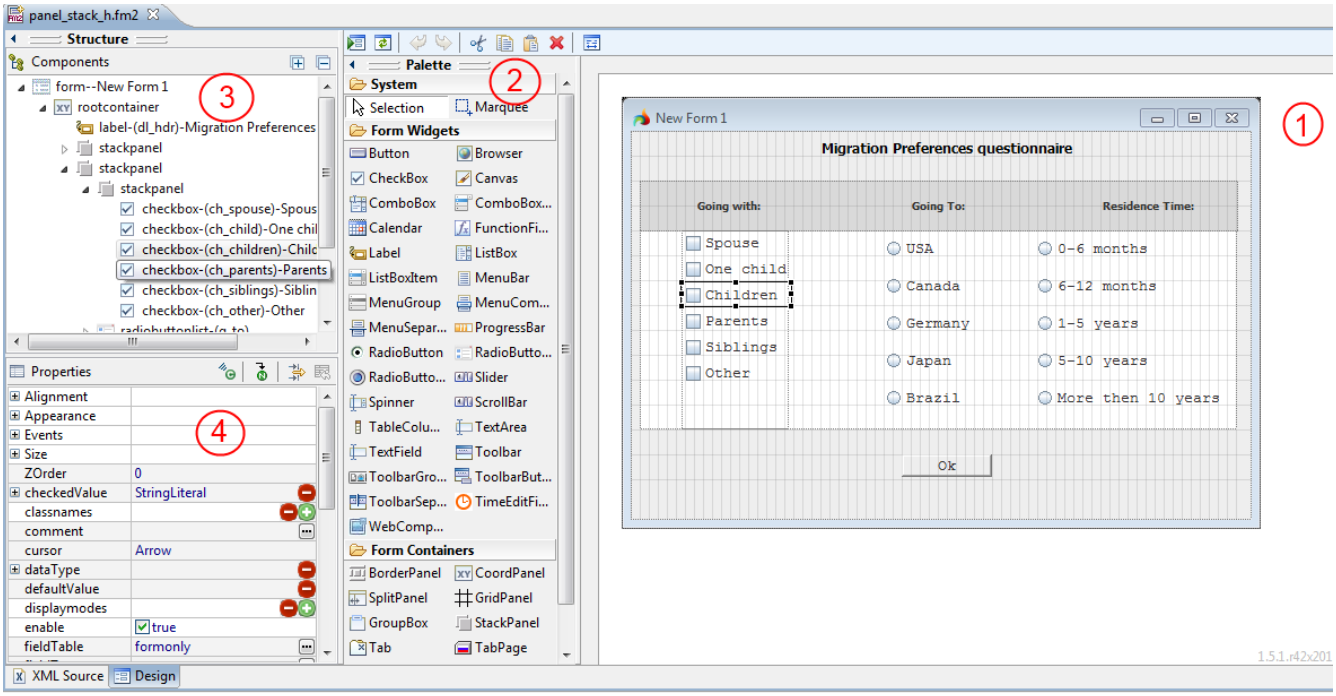
Besides the availability of the widened range of form widgets, the new Form Designer introduces a set of form containers - the areas of user-specified size to which the graphical form elements can be added. Different containers locate and treat the form widgets according to their purpose, and have different behaviour during the form resizing.

Window Builder is an open source Eclipse plug-in, it is not the property or responsibility of Querix. Similar to any other advanced graphical HTML/XML designer, the preview result can differ to the thin clients representation.

The Properties view lists an object-specific set of behavioural and graphical properties for each of the form elements. The developer can set up the objects borders, backgrounds, fonts and fore colours, location, reactions on different user actions, etc. It is also possible to set up some context-sensible properties.

The new Form Designer also allows to create form-specific menus and toolbars. The set of the available toolbar buttons can be adjusted according to the currently active fields.

The screenshot below illustrates the layout of the Form Designer. It consists of four main views that are numbered and briefly described below:



1. **Form Editor Area:** displays the graphical form itself, including all its visible elements and widgets, the way they will look, when the application is run.

2. **Palette:** contains the set of the graphical elements, that can be added to the form.

3. **Structure:** represents the hierarchical structure of the form.

4. **Properties:** gives the user the full control over the graphical and behavioural properties of the form objects.

Lycia Web Application Server

Lycia 2.2 installation package goes together with an instance of a Web Server that is based on the Apache Tomcat service. It runs automatically, if installed. It was included into the package for a number of reasons:

- Now you can create web services without worrying about installing and configuring a web server. You can use the Tomcat instance shipped with Lycia to deploy your web services.
- LyciaWeb requires a Web server to be used.

The default setting of the web server configuration can be changed by means of Lycia Web Server Manager: **Start**-> **All Programs** -> **Querix** -> **Lycia Web Server** -> **Lycia Web Server Manager**. Launching it requires administrator permissions. This tool is a native Apache Tomcat tool for web server configuration, so for any documentation on it or regarding any issues found contact the vendor.

Since Apache Tomcat is an open source web server and servlet container, Querix cannot guarantee its canned behaviour. Please, refer to the Apache Tomcat support, if any profound consultation is required.

Graphical Clients

Lycia 2.2 introduces two completely new thin clients, that are Lycia DC .HTML and Lycia Web, and a revolutionary tool for customization of an application appearance, that is Lycia Theme Designer.

Lycia DC .HTML

The Lycia DC .HTML is the Querix multi-platform thin-client. It can be run both under Windows™ and Linux™ OSs. It is developed to allow the users to render and use 4GL applications as desktop ones.

Lycia DC .HTML gets with the most recent developments and upgrades, it fully supports Microsoft WPF technology.

Besides, it easily combines graphical user interface, dynamic documents and media contents.

Lycia Web

Lycia Web is a web client based on the HTML5 and Web technologies which makes your 4GL applications available for a larger number of users, since any system which utilizes a web browser can run your applications via Lycia Web.

It is intended to replace Chimera on Linux/UNIX platforms as well as to introduce 4GL applications to other platforms including the mobile devices, where 4GL applications could not be run before, for example, on iOS and Android devices.

Lycia Web also does not require any additional software installations on the client side. At the same time the web client is more flexible and mobile; it manages to reproduce the functionality of the desktop clients practically without exceptions.

It utilizes the graphical themes created with the help of the Theme Designer, which contributes a great deal to its customizability.

Lycia Web also allows you to run 4GL application on an isolated web page or to embed 4GL applications into your existing web pages.

Theme Designer

Theme Designer is a standalone tool introducing a new way of an application feel and look customization.

It records the appearance modifications into a theme file, which can be applied to applications run both by Lycia DC .HTML and Lycia Web.

Theme Designer has a number of advantages, if compared to the script file usage:

- **Adjustable**
  Theme Designer is opened from within the application, so every user can adjust it according to their preferences.

- **Understandable**

The interface of Theme Designer is intuitive and easy to understand, there is no need to memorize the names of the script options and the precise address of the object you want to modify, because the object can be selected from the object list present in the program.

- **Fully-variable**

You can select what set of display options must be applied at what time of the program execution. An XML file can be loaded at runtime overriding the previous display settings, so one program can have several display patterns.

- **Traced**

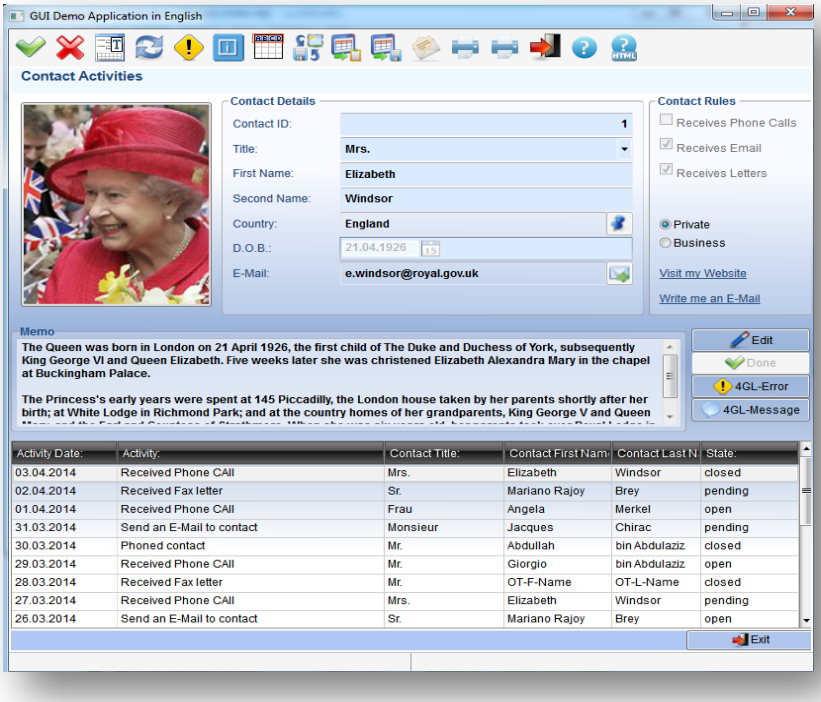
The creation of an XML file, using Theme Designer, is visual. You can see the changes made taking effect right away.

- **Easy to group**

Theme Designer has convenient object grouping such as child-parent relation, classes and pseudo-classes. It is also possible to reference each object by its ID.

Theme Designer can also be used as a part of Lycia DC .NET with some peculiarities caused by the fact, that in this case, it is considered to be the part of it, but not a standalone tool. For more details, please, refer to "Theme Designer Guide" included to the package.

The screenshot below illustrates how an application can be customized by a theme file application.



Lycia BI. BIRT Integration

BIRT is an Eclipse-based open source reporting platform for server, desktop and web based applications. BIRT has two main components: a visual report designer (available as an Eclipse integrated plug-in and standalone designer), and a runtime components that you can deploy to your application server.

Application implementing in BIRT becomes possible with the report designer. The Eclipse-based interface provides an HTML page-oriented design model to create reports, that are easy to create and integrate into web based applications.

BIRT reports are complex reports which include visual graphs, labels and other objects. The BIRT Engine enables you to produce professional looking reports.

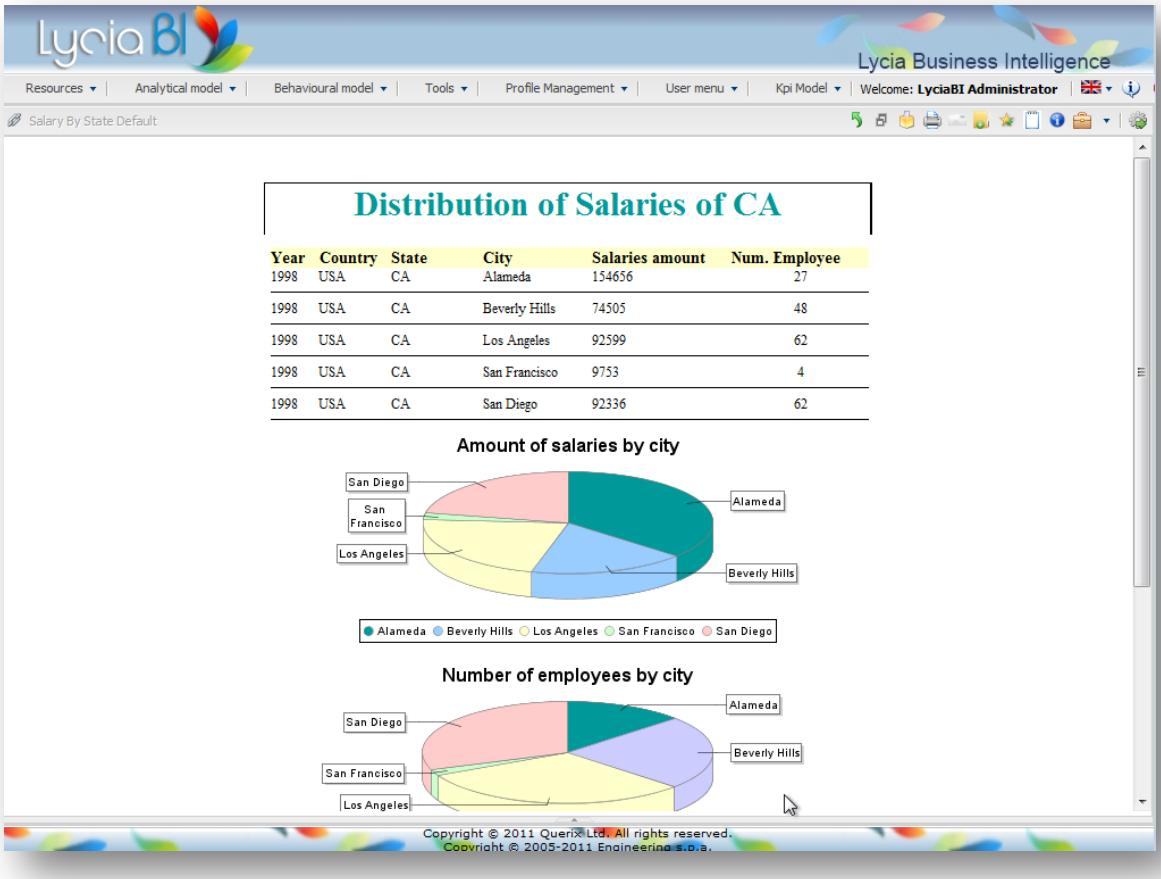
It supports the following operations:

- Adding a rich variety of report types to your application: lists, charts, grids, letter and documents, compound reports.
- Managing data sources and data sets.
- Running a report to produce HTML/Paginated HTML, WORD, XLS, PS, ODT, ODS, ODP or PDF format.
- Making editing quick and easy with the most commonly used properties in a convenient format.
- Exporting Report data to CSV.
- Generating and rendering report content.

The BIRT provides two types of designer: a plug-in for the Eclipse IDE and a Rich Client Platform (RCP) version.

BIRT Report Design Files are XML files with the default extension *.rptdesign* . BIRT Reports can contain single or multiple files. The files are categorized as either library files or resource files.

Here is what a deployed report on the server looks like:



This document only gives the general description of BIRT integration. For more information refer to "LyciaBI. BIRT Integration Guide".

4GL Modifications

Lycia 2.2 introduces a number of small, but rather significant innovations in the Lycia 4GL language which simplify the developing process.

The Java Interface

Lycia offers the possibility to use Java classes and Java objects as well as methods applicable to them within Lycia 4GL code.

Java classes can be imported easily by using corresponding 4GL statement within the source code of your application.

As with native 4GL objects, Java classes can have different scopes of reference depending on the place in the source code and the module they were imported to. Therefore, you can specify GLOBAL, MODULE and LOCAL Java classes depending on the needs of your application.

The implementation of Java interface required the introduction of set of new 4GL operators which allow you to operate Java elements easily and effectively. Therefore, java elements within a 4GL program are controlled with 4GL statements and Java class and Object methods which can be invoked within the application.

The ability to use Java elements in Lycia also means that you can pass 4GL variables to Java methods and constructors as parameters. Java variables can also be passed to Lycia 4GL. Not only simple, but also structured data types can be converted from 4GL to Java and vice versa.

Java implementation in Lycia is described in details in the 'Lycia Java Interface' manual.

The XML Interface

Lycia includes an interface for parsing XML files. This interface allows a programmer to read the content of the XML files, change it or create completely new XML data and then save the files to the disc. It includes both DOM and SAX parsers and introduces a number of specific data types such as XMLDOCUMENT, XMLNODE, XMLATTRIBUTE, etc. that are used to manipulate XML data.

Parsing XML files is much more efficient than using text configuration files for your 4GL programs, since an XML file has a definite structure where each element has its own unique position.

The XML Interface is described in details in 'Lycia XML Interface' guide.

Action Extensions

In Lycia one can specify program actions for a set of application details performed within an application window as a whole or in a specified field during an input or any other event which can have ON ACTION clause.

The actions, that the application must perform after these manipulations can be specified in ON ACTION clause of different input and display statements:

For example, the INFIELD clause lets you specify in which fields the doubleclick action can be activated. If you omit the INFIELD clause, the doubleclick action will be invoked after a double click anyplace in the application window:

```
ON ACTION ("doubleclick")[INFIELD (fieldname)]
```

Among other options, there are OnMouseWheel, OnResize, OnDrop and other. Find the more detailed information in the 'Lycia Form Designer' guide that is available on the Querix website.

Pre-processor

Lycia includes a pre-processor. The pre-processor allows you to define different kinds of macros and to include other files. In general, 4GL pre-processor behaves very similar to the C Pre-processor, though it has some peculiarities.

A macro is defined in the code using dollar ($) sign. E.g.:

```
$define mac "this is macros body"

-- Do not use quotation marks in your code.
```

The pre-processor can do the following:

- Define and expand macros:
- Simple macros: the pre-processor replaces the macro name found in the code with the macro body;
- Function macros: serves the same purpose as the simple macro, but accepts arguments.

- Include files: the included file is processed together with the source file into which it is included resulting in the output of the two (or more) files combined in one. The syntax of the function is as follows:

  ```
  $include
  ```

- Perform conditional compilation: some code may be omitted during compilation depending on the given conditions.
- Add multiline by using the slash symbol (\) in the end of the definition line

Application Manipulation Methods (Graphical API)

Lycia 4GL now includes the following 4GL constructs which both facilitate the process of 4GL application development and make the applications easier and more pleasant to use:

- New classes with the sets of corresponding methods:

- **base class:** provides the general application execution controlling interface for character string manipulation, retrieving application details, serialization of program variables, etc.

- **util class:** provides an interface for mathematical functions;

- **path class:** provides interface for files and directories manipulations

User Interface Manipulation from 4GL Code

Lycia 2.2 offers the unprecedented freedom in manipulating the user interface from within the program. Basically any UI object - a form widget, a window or even such things as the background of a window or a font - can be addressed from within the 4GL code. All the properties of such object can be manipulated and set from within the 4GL code.

All these objects are grouped with the help of the 'ui' class. This class incorporates a number of objects with methods. Some methods are unique for the objects. These methods and objects are described in the "Functions" guide.

For example ui.Dialog object has methods that allow you to manipulate fields during input:

```
DEFINE my_dial ui.Dialog
...
INPUT a,b,c,d FROM scrrec.*
BEFORE INPUT
      LET my_dial = ui.Dialog.getCurrent()
ACTION fieldtouched
#This method called from within input sets the field status
CALL my_dial.setFieldTouched("city", TRUE)
...
ACTION val
#The following code validated all the fields in the record
LET err = dialog.validate("scrrec.*")
            IF err!= 0 THEN
             LET err_mess = "Validation failed \n", err_get(err)
             DISPLAY BY NAME err_mess ATTRIBUTE (RED)
            END IF
END INPUT
```

There are other objects that belong to the 'ui' class which have a number of properties that can be set through 4GL code or whose values can be retrieved. Such objects are called UI entities and no other manipulations can be performed for these objects. The properties can be set using the *Set<property>()* method and retrieved using the *Get<property>()* method.

Here is an example of how they can be used:

```
MAIN
DEFINE newColor ui.SystemColor
DEFINE airport ui.ComboBox
```

```
DEFINE items DYNAMIC ARRAY of ui.ComboBoxItem
DEFINE newItem ui.ComboBoxItem
DEFINE str STRING

OPEN WINDOW wd AT 2,2 WITH FORM "myform2"
INPUT str FROM airport
BEFORE INPUT
      LET airport = ui.ComboBox.forName("airport")
      CALL newColor.SetSystemColorName("green") --define a colour
      CALL airport.SetForeColor(newColor)--set font colour
      CALL airport.SetEditable(true) -- allow editing in combobox
      # then we create a set of combobox elements
      CALL airport.SetText("Test")
      CALL items[1].SetText("First")
      LET str = items[1].GetText()
      CALL items[2].SetText("Second")
      LET str = items[2].GetText()
      CALL items[3].SetText("Third")
      LET str = items[3].GetText()
      # and add them to the combobox
      CALL airport.SetComboBoxItems(items)
      LET str = airport.GetText()
END INPUT
END MAIN
```

The objects which have custom methods and are described in the Functions guide can also be used as UI entities. In the example above *ui.ComboBox* is used as an entity, though it has a number of custom methods. The *Set<property>()* method always takes only one parameter which is the value to be set. The *Get<property>()* method does not require parameters and returns one value.

Multiple Database Connections

In Lycia 2.2, a 4GL program can connect to several databases at a time, these databases can be located on different database servers and belong to different database vendors. This makes the task of migrating the data from one database to another extremely easy.

From the perspective of 4GL code, the CONNECT TO, SET CONNECTION, and DISCONNECT statements handle simultaneous connection to several databases. For the syntax and usage of these statements see the "Querix 4GL Reference" guide.

It is natural, that you should make adjustments to your code to make your application connect to several databases. However, you should also add the databases you want to connect to into the database configuration file. This file is called *database.cfg* and is located in %LYCIA_DIR%/etc/ directory. It should contain the information about all the databases you want to connect to regardless of the database type or vendor. More information about *database.cfg* can be found in the "Developer Guide".

Here is an example of the file contents:

```
ifx_cms {
driver = "informix"
source = "cms"
username = "John"
password = "12345"
}
ora_cms {
driver = "orcale"
source = "cms"
}
```

In this case the user credentials are not given for the second database which makes the connection far more secure. Once this is recorded to the configuration file, you can add the following lines to your application:

```
CONNECT TO "ifx_cms"
CONNECT TO "ora_cms+username='Jane', password='678912'"
```

If you want your application to connect to a single database, you still can use the DATABASE statement to specify the database, but you still will need to specify the database in the *database.cfg* file. The SET CONNECTION and DISCONNECT statements can be used to manipulate the multiple database connections effectively.

Genero® Compatibility

Lycia had the facilities for importing Hydra legacy projects as well as the native Informix RDS projects. Now, Lycia incorporates the facilities for importing and working with Genero® projects. The imported projects are fully incorporated into LyciaStudio and transformed into LyciaStudio native projects.

The Genero® compatibility includes the following features:

- Importing Genero® projects and transforming them into Lycia projects with the project and program structure retaining
- Importing Genero® graphical forms (.4fd format) and their converting into new form format.
- Importing the XML files containing start menu, toolbar or top menu and converting them into new form format.

The 4GL source files remain unchanged during the import: their syntax is fully supported by Lycia 2.2.

Web API

Lycia Web API is a framework for building web APIs using HTTP protocol, that can be consumed by a wide range of clients including browsers, mobiles, tablets, etc.

Lycia Web API provides customers with the great possibility to expose their data and service to different devices. It is an ideal platform for building REST-based services, that support simple and lightweight means for implementation of scalable distributed applications.

The advantages of using Lycia Web API:

- HTTP based: allows to use all service that are available for HTTP to be available for the interface,
- makes any 4GL application a web service accessible under any computing language,
- applications are standard Java servlet and can be run on any standard Java servlet container,
- it is implemented on top of the Spring MVC Framework, so customers can access the vast variety of tools shipped with the Spring MVC and its servlet container for application extension,
- provides means for services orchestration,
- a 4GL application plays resource role, not service, so it is typically stateless. But it can be stateful, if needed,
- integrates customers` business logic 4GL projects into the cloud,
- there is a possibility to call either services from 4GL applications or another ones, that can call 4GL.

Obsolete Features

Elements in the following list are completely obsolete, and must not be further used by the developers:

- `compat.ifx.input.display_to = true`
- `compat.ifx.input.display.nobind = 1`
- `gui.key.interrupt = "value"`
- `a<>NULL and a==NULL are not supported any more`