

# Lycia Development Suite



**Lycia II Getting Started**  
**Version 2.2 – April 2014**  
**Revision number 1.07**





# Querix Lycia II

---

## Getting Started with Lycia

Part Number: 012-022-107-014

Elena Krivtsova/Olga Gusarenko  
Technical Writers

Last Updated April 2014

## **'Lycia' Getting Started**

Copyright © 2006-2014 Querix Ltd. All rights reserved.

Part Number: 012-022-107-014

### **Published by:**

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

### **Publication history:**

July 2009:	Beta edition
June 2010:	Beta 2 edition
November 2010:	First edition
June 2011:	Updated for Lycia II
April 2014:	Updated for Lycia 2.2

### **Last Updated:**

April 2014

### **Documentation written by:**

Elena Krivtsova, Olga Gusarenko

### **Notices:**

The information contained within this document is subject to change without notice. If you find any problems in the documentation please submit your comments by email to [documentation@querix.com](mailto:documentation@querix.com).

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose without the express permission of Querix (UK) Ltd.

Other products or company names used within this document are for identification purposes only, and may be trademarks of their respective owners.



## Table of Contents

INTRODUCTION.....	1
CHAPTER 1 .....	2
LYCIA REQUIREMENTS .....	2
<i>System Requirements</i> .....	2
<i>Downloading Software</i> .....	2
<i>Documentation and Demonstrations</i> .....	2
Documentation .....	2
Demonstration Applications.....	3
CHAPTER 2 .....	4
INSTALLING LYCIA.....	4
<i>Installation for Windows</i> .....	4
Installation modules .....	4
GUI Server Configuration.....	5
<i>Installation for UNIX/Linux</i> .....	8
<i>Checking the GUI Server Service and other Services</i> .....	9
Verifying that the GUI application server service is running.....	9
Verifying that the GUI application server will start automatically on reboot.....	10
Verifying connection to the GUI application server service .....	10
Verifying that 'LyciaStudio' can connect to the GUI application server service .....	11
Apache Tomcat Service .....	12
<i>GUI Server Troubleshooting</i> .....	12
Firewall.....	12
Starting the GUI Server Service (manually).....	13
CHAPTER 3 .....	15
GETTING A LICENSE .....	15
<i>A Trial License</i> .....	15
Getting a Trial License Automatically.....	15
Getting a Trial License Manually .....	16
<i>Trial License Restrictions</i> .....	17
CHAPTER 4 .....	18
DATABASE CONNECTIVITY .....	18
<i>Connecting to a database</i> .....	18
Setting up Informix Connection .....	18
Setting up Oracle Connection.....	21
Setting up SQL Server Connection.....	22
<i>Database Configuration File</i> .....	24
Precedence of Parameters .....	25
<i>Setting the Default Database Driver</i> .....	26
For compilation using LyciaStudio.....	26
For running applications using the application server .....	27
For the command line environment.....	27
CHAPTER 5 .....	29
USING LYCIASTUDIO .....	29
<i>Running the Studio</i> .....	29
<i>Importing a legacy 'HydraStudio' project</i> .....	31

4GL and C programs.....	35
<i>Importing other legacy projects.....</i>	<i>36</i>
<i>Importing demonstration applications.....</i>	<i>39</i>
<i>Creating a new project.....</i>	<i>41</i>
<i>LyciaStudio Functionality and Debugging.....</i>	<i>48</i>
Explaining the LyciaStudio toolbar.....	48
The Debugger / Debug As.....	49
Run As.....	51
The Form Editor.....	51
<b>CHAPTER 6 .....</b>	<b>55</b>
USING THE COMMAND LINE.....	55
<i>qfgl.....</i>	<i>55</i>
<i>qform.....</i>	<i>56</i>
<i>qmsg.....</i>	<i>56</i>
<i>qlink.....</i>	<i>56</i>
<i>qrun.....</i>	<i>57</i>
<b>CHAPTER 7 .....</b>	<b>58</b>
LOOK AND FEEL OF YOUR APPLICATIONS .....	58
<i>Launching the Theme Designer.....</i>	<i>58</i>
Changing the background.....	59
Changing the font.....	60
Moving objects around.....	61
Finishing the Editing.....	62
<i>Responsive Design.....</i>	<i>63</i>
Full-screen Mode.....	63
Orientation.....	65
<b>CHAPTER 8 .....</b>	<b>67</b>
GRAPHICAL REPORTS .....	67
Creating a report.....	67
Defining the report data.....	68
Populating the report.....	69
Deploying and viewing the report .....	72





---

## Introduction

---

Welcome to the world of Lycia, the essential tool set for 4GL, ESQL/C & New Era developers.

This 'Lycia – Getting Started' guide will take you through the actions required to get your applications working with our new development suite and LyciaStudio, our Eclipse-based development studio.

Lycia is designed to make the development and deployment process simpler and easier. Lycia compiles your projects into pseudo-machine code (or P-Code), which means platform independence for your projects. You will be able to compile a project and export it to a different platform than that it was developed on and it will work.

LyciaStudio is an integrated Eclipse-based development suite with a built-in graphical form editor, allowing adjustments to be made quick and easily. With detachable tabs to rearrange your editing environment, LyciaStudio is easily customisable to meet your needs.

There is also the introduction of a debugger, which allows developers to delve into their coding and explore where faults may be occurring using breakpoints and watchpoints, all easily executed from within the Studio.



# CHAPTER 1

## Lycia Requirements

---

### System Requirements

- Java 1.7 and above
- C Compiler (only required if you include C or EsqIC files within your Lycia project):
  - Solaris: SUN Works Pro C, gcc
  - AIX: Visual Age, gcc
  - HP-UX: native HP-UX C compiler, gcc
  - Windows: Microsoft Visual Studio 2003, Microsoft Visual Studio .NET, Microsoft Visual C++ Toolkit 2005 and above
  - Linux: gcc

### Downloading Software

Querix software, user guides and other documentation can be downloaded from the Querix Web site at: <http://www.querix.com>.

## Documentation and Demonstrations

### Documentation

This installation includes a set of documentation, provided as PDFs. This comprises of:

#### The 'Lycia Developer's Guide' Guide

This guide shows you the working steps involved to develop with our new compiler and Studio and how to import old projects. It will also explain the command line tool, the new graphical form editor and developments within the language.

#### The 'Lycia Getting Started' Guide

This guide shows you the basics of getting Lycia up and running, how to build and develop programs and get working with our new Studio.





### **The ‘Lycia Release Notes’**

This document contains all the known issues which will be fixed in the next releases

### **The ‘Lycia Upgrade Guide’**

This document contains all of the changes made between the latest version and the previous release.

### **Demonstration Applications**

You can download demonstration applications for Lycia from our CVS repository. They can be checked out using the built-in functionality of LyciaStudio. The instructions on how to download and use the demo applications can be found in chapter 4 of this manual, [“Importing demonstration applications”](#).




## CHAPTER 2


### Installing Lycia

The installation of Lycia on Windows and Unix has been designed to guide you step-by-step through each decision. This chapter will highlight key differences between the installation processes and key issues that may be encountered.

#### Installation for Windows

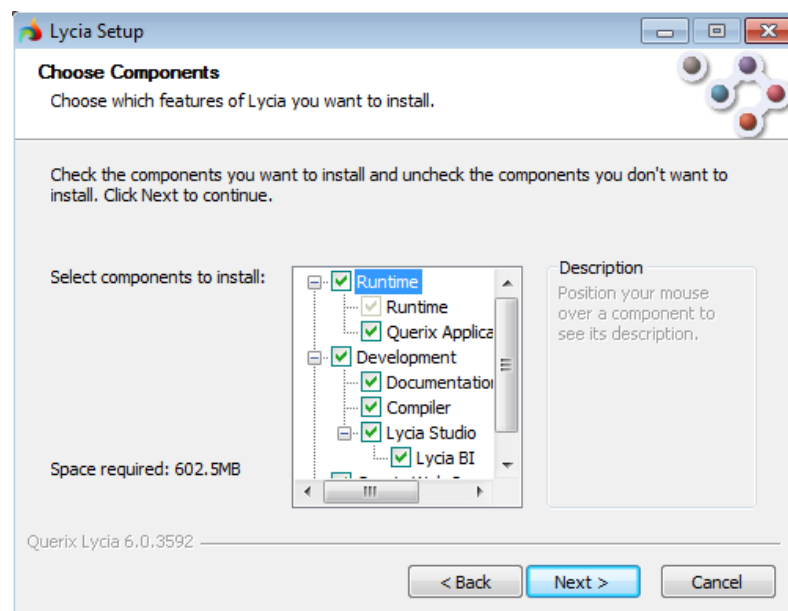
	<p>Note: to install Lycia on Windows, the user <b>must</b> have administrator privileges.</p>
---	---

Basic installation can be completed by following each step after running the installation package. We will now discuss any extra issues that may be encountered.

	<p>Note: The installation path for Lycia must not contain special symbols such as @, %, #, &lt;, &gt;, ", !. If it does, Lycia Studio will fail to start.</p>
---	---

#### Installation modules

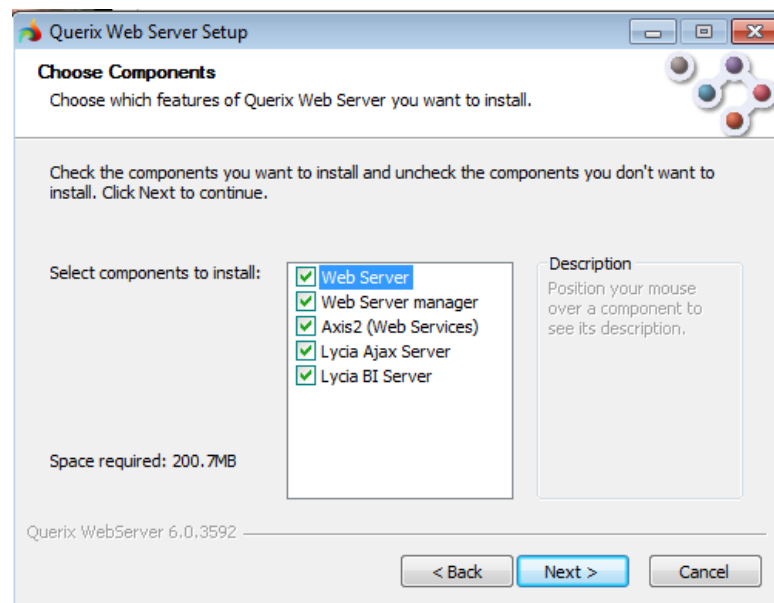
Lycia II Development Suite installation package includes a number of optional modules.





There are three big modules: Lycia Runtime with the Application Server, Lycia Compiler and development environment and Querix Web Application Server:

- Lycia Runtime includes:
  - The Runtime proper, this is the only required module which must be installed
  - Lycia Application Server - the application server is now independent from Lycia version and inherits any of the other application servers that were previously installed.
- Development
  - Documentation
  - Compiler
  - LyciaStudio
    - LyciaBI - an additional plug-in for LyciaStudio which allows you to develop Business Intelligence report templates.
  - Querix Web Server - this is an independent installation which should be ticked off, if you want to install any of the following features which are optional and can then be checked or unchecked in the Web Server installation dialog:
    - Web Server Proper - this is an obligatory module, if you want any of the other components to work.
    - Web Server Manager - this is a third party tool used for web server configuration.
    - Axis2 - this is a third party service used for deploying web services
    - LyciaWeb Server - this is LyciaWeb Client
    - LyciaBI Server - this is the server for deploying and executing Business Intelligence reports




## GUI Server Configuration

There are 2 points during the installation process on Windows where the user's input is required relating to GUI Server configuration. The first decision that will be encountered is if you have a previous Querix installation and have a GUI server listener configuration already in place. If you wish to keep your existing configuration, then select the "Keep Existing Configuration" button and continue, else a new configuration will be created and replace any that was already present. If you choose to

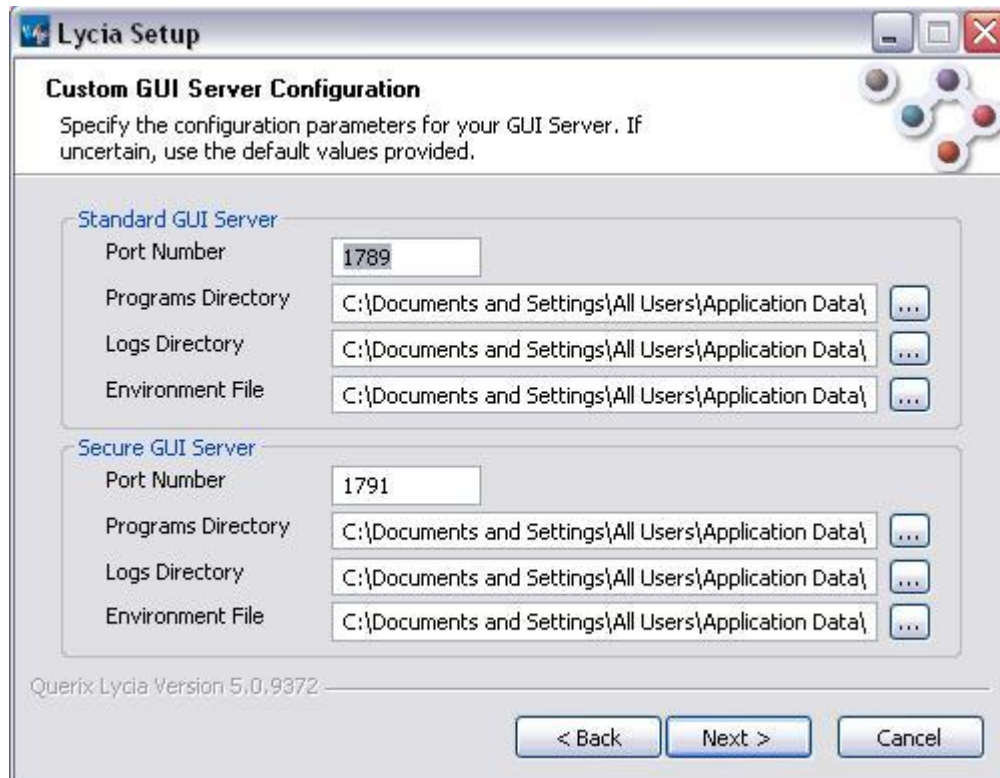


"Keep Existing Configuration", you will not be presented with the second screen of this process. There is also an "Append to Existing Configuration" option, which allows previous ports to remain active whilst new ones are added.



	<p>Note: if you choose 'New Configuration', your previous settings <b>will be replaced</b> – if you previously had 'Hydra4GL' Version 4 installed, the GUI server settings will be overwritten.</p>
---	---


If you choose to create a new configuration, then you will be presented with a screen in which you can either use the default values for the GUI server listener (1889 for a basic connection or 1891 for an authenticated connection) or whether you wish to set up your own custom values. If you choose to customise this, you will see the following screen:




Here, you can specify the port number for both your standard and secure ports, which directory programs are deployed to, where logs are to be kept and which environment file to use.



## Installation for UNIX/Linux

	<p>Note: to install Lycia on Unix/Linux, the user <b>must</b> be <i>root</i>.</p> <p>It is also recommended that you create a user called <i>querix</i> or <i>informix</i> and a group called <i>querix</i> or <i>informix</i>, which can act as the owner for the Querix installation</p>
---	--


	<p>Note: Installation of Lycia for Mac OS, which is also a UNIX system differs from installation on other UNIX systems due to its graphical interface.</p> <p>For more details about the installation for Mac OS see the Developers Guide.</p>
---	--

To install Lycia on Unix, the installer should be copied into a temporary location just for the installation and the process run as root. The package should then be unzipped and extracted and then run the install script as follows:

```
gunzip Lycia-<platform>-<buildtype>-<date>.tar.gz  
tar xvf Lycia-<platform>-<buildtype>-<date>.tar  
./install.sh
```

After reading and agreeing to the EULA, you will be able to install Lycia into /opt/Querix by following a number of simple steps. Points to note:

- Step 2 allows you to change the directory into which your Lycia install is placed. By default, this is /opt/Querix


	<p>Note: The installation path for Lycia must not contain special symbols such as @, %, #, &lt;, &gt;, ", !. If it does, Lycia Studio will fail to start.</p>
---	---

- Step 4 allows you to select and deselect which elements of the Lycia package to install. You cannot disable the installation of the runtime components, but you can disable installation of the documentation, thin client support through the application server and the compiler.
- Step 5 will overwrite any previous environment files if you select 'yes' – note that this cannot be undone.
- Step 6 sets the Application server directory. If an application server is already installed in the location specified, you are informed of this, and selecting the option 'yes' will replace any settings already in place



- . environ needs to be run before using Lycia to ensure the correct environment is set.

After installation, you will find in /opt/Querix (or the location set in Step 2) a folder called Lycia, where the software has been installed to.

	<p>Note: to run the Studio on Linux, GTK must already be installed on the system. This must include the GTK2 bindings for Java package (for example, the libgtk-java package).</p> <p>This is because LyciaStudio is built using the Eclipse framework, which in turn includes the SWT toolkit which uses GTK+ on Linux. Installation of this package varies across platforms, so please refer to your distribution's manuals for how to use the package tools.</p> <p>Also note that if GTK is not present, the Studio may not start and will produce no error messages.</p>
---	---

## Checking the GUI Server Service and other Services

The GUI AppServer is the service qxinetd2 which is located in the AppServer directory (/usr/local/AppServer by default in UNIX/Linux and C:\Program Files\Querix\AppServer by default in Windows). This service is started when the machine is booted by the script file qxinetd2. This script file can be called with four arguments:

- start            to start the service
- stop            to stop the service
- restart        to stop and start the service
- status         to display the status of the service

It should be noted that even if this service is running, the firewall can still block any communication between clients and the GUI AppServer. For more information on your firewall, please ask your system administrator to enable the AppServer ports (default installation values 1889 & 1891).

### Verifying that the GUI application server service is running

#### UNIX/Linux

It is now possible to test if the GUI application service is running and whether you are able to connect to the Querix listener service. The easy way to do this is to use the following code at the command line:

```
/etc/init.d/qxinetd2 status
```

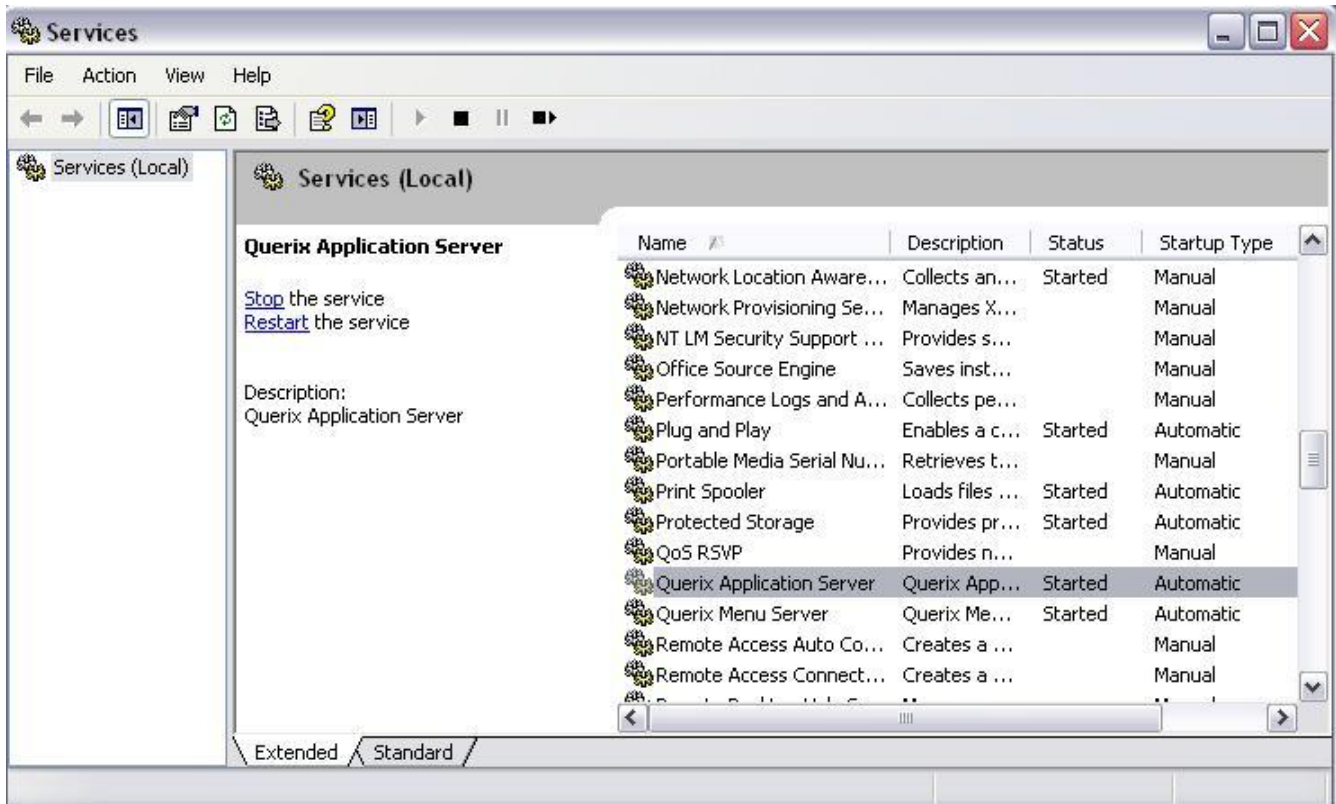
If the GUI server is running, the message 'The GUI Server is running' will be displayed.

#### Windows

To test whether the GUI application service is running in a Windows environment, simply open the Control Panel, then Administrative Tools (in Windows 7 they are in System and Security section), and



finally Services. You should see in the list of services 'Querix Application Server'. If the GUI Server is running, this will say 'Started' under the Status.



## Verifying that the GUI application server will start automatically on reboot

### UNIX/Linux

You should have the link `S*xinetd2` in `/etc/init.d/rc3.d`, `/etc/init.d/rc2.d` and `/etc/init.d/rc5.d` pointing at `/etc/init.d/xinetd2`. To check this, in the `/etc/init.d/rc3.d` directory, type `ls -l` and look for the `S99xinetd2` file.

### Windows

In a Windows environment, in the Control Panel -> Administrative Tools -> Services screen, look under Startup Type and check that it reads 'Automatic'.

## Verifying connection to the GUI application server service

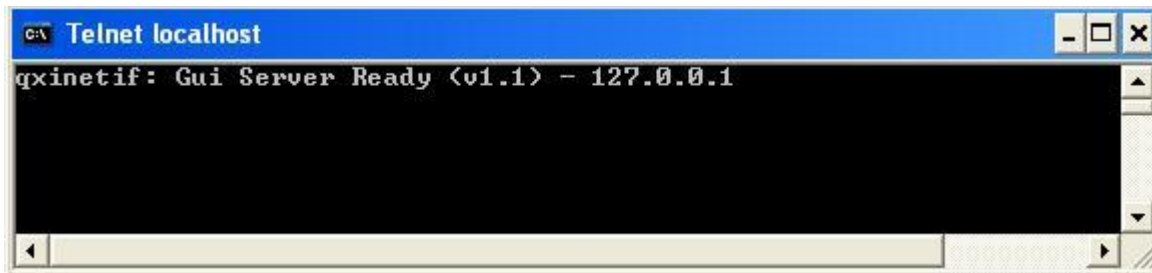
If the service is running correctly, we test whether the user can connect to it by typing the following line in the Lycia command line environment:

```
telnet localhost 1889
```

(assuming the default port configuration with the port 1889 (and secure 1891) were specified earlier during the installation)

If the connection is successful you will see the following response:

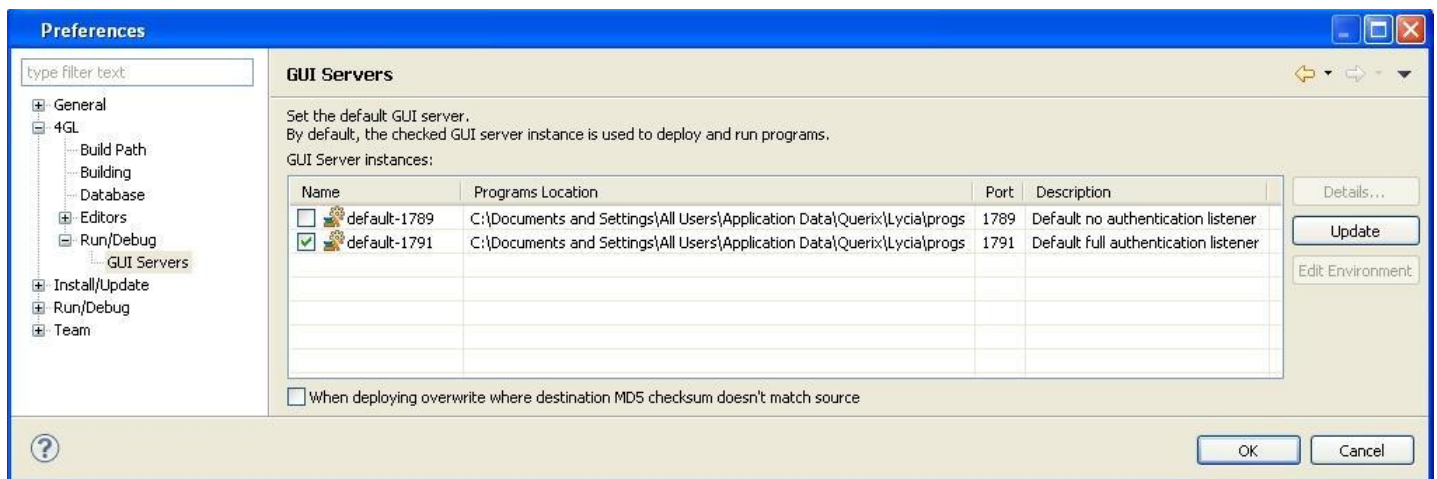




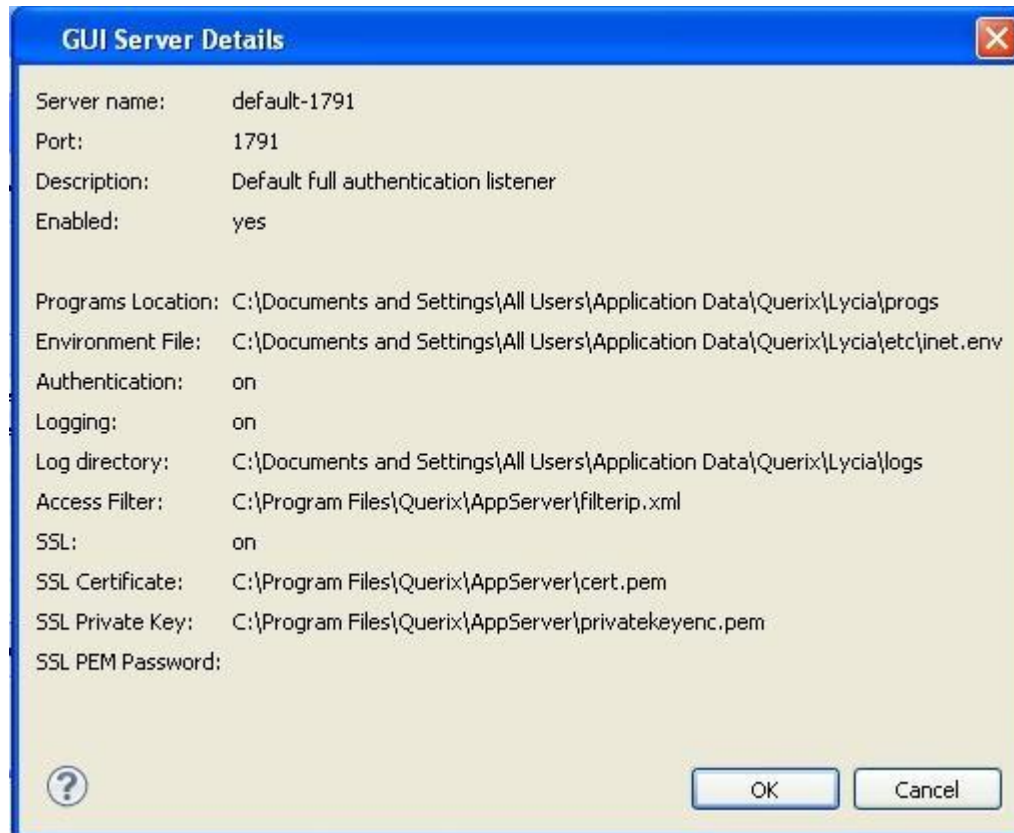
This is the same for Unix/Linux users – perform the same actions as above in a command line to check that you can connect to the GUI application server service.

### Verifying that 'LyciaStudio' can connect to the GUI application server service

1. Start 'LyciaStudio'
2. Go to the **Window -> Preferences -> 4GL -> Run/Debug -> GUI Servers** preferences page. Here you will see the list of installed GUI servers:



3. You can verify whether a server is enabled by highlighting it and pressing the **Details** button. The server information will be displayed:



If you do not see this response it may be because the GUI application server service is not working or because a firewall is not allowing external communication. Check with your system administrator to ensure that the ports 1889 / 1891 allow connections.

### Apache Tomcat Service

Lycia is shipped together with a Tomcat instance used for LyciaBI server and for Lycia Administration Tool. It can also be used for deploying and running web services created in Lycia.

The service is called Apache Tomcat QxTomcat6. It can be verified and adjusted in the same way as the GUI Application Server service. If this service is not running, you will not be able to use Querix tools, because it is crucial for license managing.

## GUI Server Troubleshooting

### Firewall

If you do not see any response (or you get some kind of error message such as 'could not connect...') when trying to telnet the GUI application server, but the services are running, it may be because a firewall is not allowing external communication. Check with your system administrator to ensure that port 1889 allows a connection.

If you are on a standalone system, you may try to temporarily disable the firewall to see if it shows any effect.



### **Starting the GUI Server Service (manually)**

The GUI Application Server (AppServer) is dependent upon the system inetd/qxinetd2 service manager. If your system does not start the qxinetd2 service manager on system boot, you can start the qxinetd2 service manager manually by doing the following:



## UNIX/Linux

1. Login as root.

```
su -
```

2. Change to the directory /etc/init.d

```
cd /etc/init.d
```

3. Start the GUI application server service qxinetd2 with the command

```
./qxinetd2 start
```

You should see a response similar to "Starting the Querix AppServer..."

```
linux:/ # cd /etc/init.d
linux:/etc/init.d # ./qxinetd2 start
Starting the Querix AppServer...
linux:/etc/init.d #
```

If, having seen such a response, no further activity is seen you may need to press the Enter key to return to the prompt.

## Windows

In the same screen as before (Control Panel -> Administrative Tools -> Services), in the upper-left corner of the screen, you should see the option to 'Start' the service. Click this to start the service.

When you have 'LyciaStudio' running, in order to update the listener status you will need press the **Update** button in the **Window -> Preferences -> 4GL -> Run/Debug -> GUI Servers** preferences page.



## CHAPTER 3

### Getting a License

To be able to use Lycia II you need a license. It can be either a full license bought from Querix or our reseller, or a trial license. The last one has a number of critical restrictions described later in this chapter. As to the full licenses, they differ in a number of runtime seats and a period of validity.

If you are interested in getting a normal license, please, contact Querix team or our reseller. After you obtain the full license, refer to Lycia Development Guide for the information about activating and using it.

This section will cover the receiving and usage of a trial license.

#### A Trial License

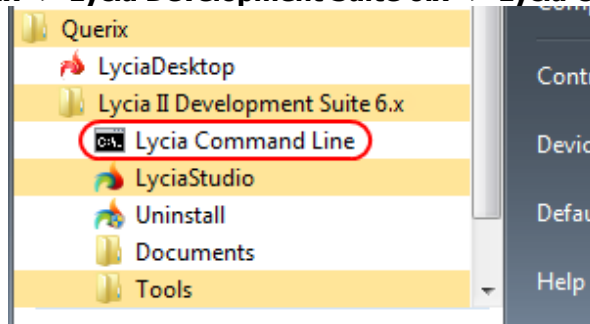
A trial license is a license which is available to anyone who has downloaded and installed Lycia free of charge. If the machine where you are installing Lycia has an active Internet connection, you do not need to contact the Querix support team to get it. In this case you will get the license automatically. To do it follow the instructions given in the section "Getting a Trial License Automatically".

If an internet connection is not available on your working machine follow the instructions given below in the section named "Getting a Trial License Manually".

#### Getting a Trial License Automatically

To get a trial license with a working Internet connection do the following:

1. Go to **Start -> Querix -> Lycia Development Suite 6.x -> Lycia Command Line**

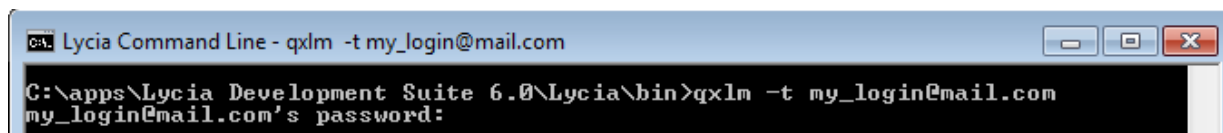


2. In the command line environment type this command:

```
qxlm -t <login>
```


Where the login stands for your login on Querix web-site.

3. Enter the password you use to login to Querix web site when prompted:





4. The license will be received from the license server and activated automatically. You will see the following message:



```
ca: Lycia Command Line - qxlm -t my_login@mail.com

C:\apps\Lycia Development Suite 6.x\Lycia\bin>qxlm -t my_login@mail.com
my_login@mail.com's password:
Trial License added successfully
```

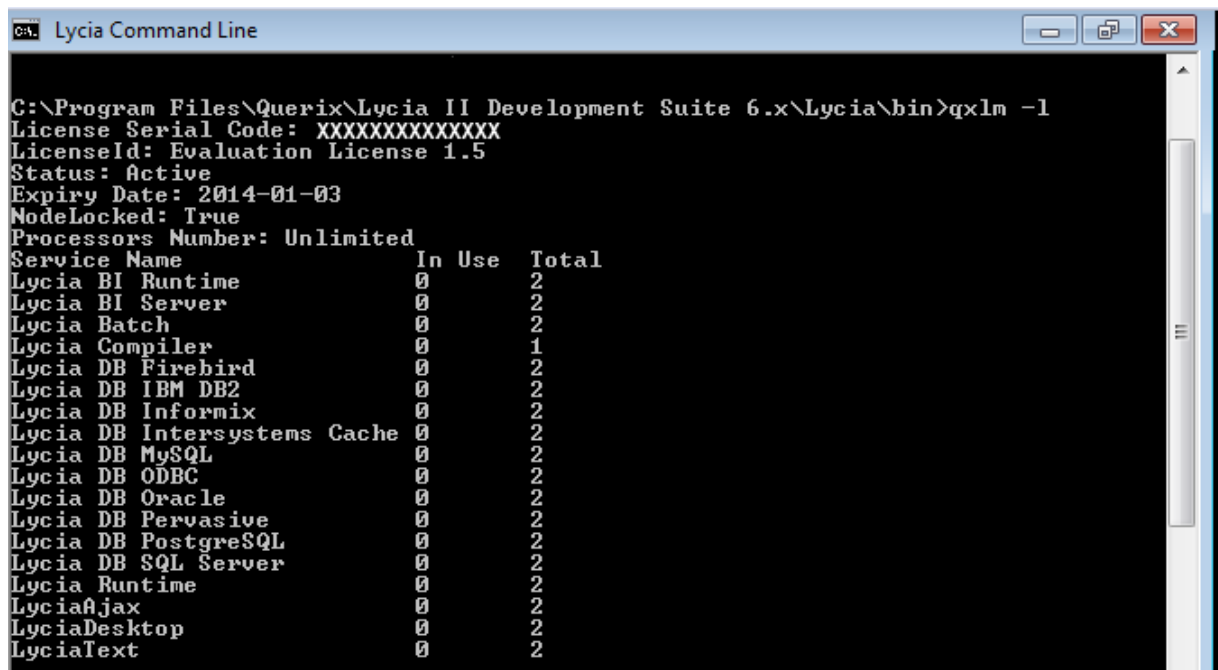
If instead of success after taking the described steps the error message appears, make sure that your login and password were put correctly. Note, that the possible causes of an error (besides incorrect login/password) may also be the following:

- you are trying to get the trial license for the second time for the same machine;
- the server is not available;
- the local file system is not available.

If you cannot reach success in getting a trial license following the instructions below on your own, please, contact Querix support team to handle a problem.

To retrieve the information about the successfully received license from the command line, you may use the `qxlm -l` command.

It is displayed to the console as shown on the screenshot below:



```
ca: Lycia Command Line

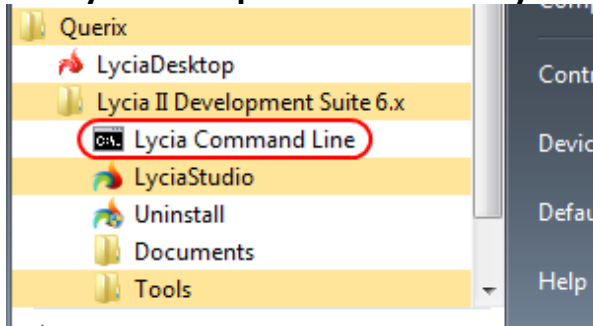
C:\Program Files\Querix\Lycia II Development Suite 6.x\Lycia\bin>qxlm -l
License Serial Code: XXXXXXXXXXXXXXXX
LicenseId: Evaluation License 1.5
Status: Active
Expiry Date: 2014-01-03
NodeLocked: True
Processors Number: Unlimited
Service Name      In Use  Total
Lycia BI Runtime  0       2
Lycia BI Server   0       2
Lycia Batch       0       2
Lycia Compiler    0       1
Lycia DB Firebird 0       2
Lycia DB IBM DB2  0       2
Lycia DB Informix 0       2
Lycia DB Intersystems Cache 0       2
Lycia DB MySQL    0       2
Lycia DB ODBC     0       2
Lycia DB Oracle   0       2
Lycia DB Pervasive 0       2
Lycia DB PostgreSQL 0       2
Lycia DB SQL Server 0       2
Lycia Runtime     0       2
LyciaAjax         0       2
LyciaDesktop      0       2
LyciaText         0       2
```

## Getting a Trial License Manually

If the machine where you want to install the license is not connected to the Internet, please, follow the following instructions:



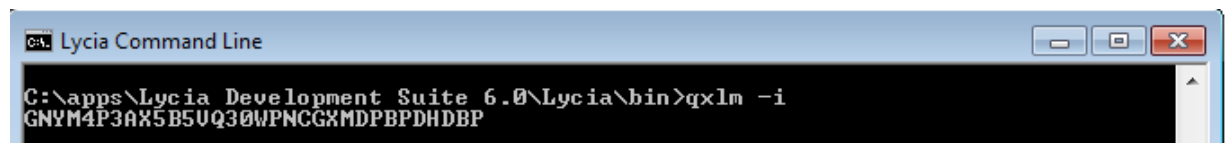
1. Go to **Start -> Querix -> Lycia Development Suite 6.x -> Lycia Command Line**



2. In the command line environment type this command:

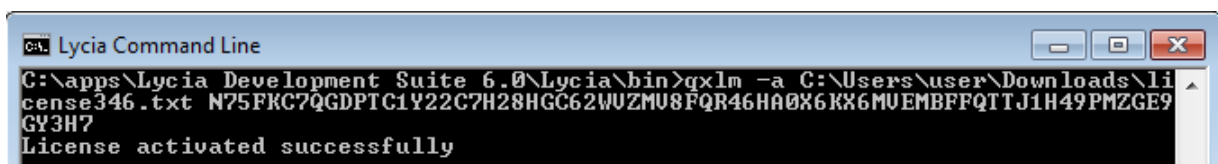
```
qxlm -i
```

It will return the hardware ID of your computer consisting of letters and numbers, e.g.:



3. Copy the hardware ID and send it by e-mail to Querix support team ([support@querix.com](mailto:support@querix.com)) together with the request for a trial license.
4. Querix Team will send you a license file in .txt format and the activation code for the license.
5. After you get the file, run the command line tool again and enter the following command to activate the license:

```
qxlm -a <path to the license file> <activation code>
```



Now the license information can be viewed with the `qxlm -l` command.

## Trial License Restrictions

A trial license has a number of restrictions:

- It can be claimed only once for each computer.
- It grants 2 seats for each service. Which means that you have 2 runtime seats and you can run only up to 2 4GL processes simultaneously.
- It is valid for 6 weeks (42 days) starting from the day it was claimed.
- It is node-locked, which means applications compiled with this license can be run only from the application server located on the same machine and cannot be transferred to other application servers.



# CHAPTER 4

## Database connectivity

Nearly all 4GL programs require access to a database server, and so it is necessary to provide Lycia II with the connection details for that server. There are several different methods for this; which one you use is dependent on your platform.

Since Lycia 4GL is a complete implementation of Informix-4GL, all database connections are made to emulate Informix connection conventions, which ensures that your 4GL source code requires no changes to access multiple RDBMSs.

Database connections are discussed in more detail in the various dedicated Database Migration guides as well as in the Lycia II Developer Guide, which can be downloaded from our web-site [www.querix.com](http://www.querix.com). These Migration guides cover such databases as Informix, Oracle, MySQL and other. In this chapter we outline only the main steps required for connecting to the main supported databases, we do not discuss these steps in detail.

### Connecting to a database

Lycia II allows your application to connect to several databases at a time. This is achieved by the means of the database configuration file which stores all the valid database connections and the CONNECT TO statement which unlike the DATABASE statement can enable connection to several databases at the same time.

If you migrate your applications from the earlier versions of Querix products and do not plan to use the multi-database connection, you do not need to perform any changes either in your environment or in your code. For the legacy projects migrated to Lycia the corresponding environment variables and the Connect Tool settings will be used, if the database is not found in the database.cfg file.

To connect to a database in Lycia II you will need:

1. Set up connection between the database server and the database client. This guide describes how to do it for [Informix](#), [Oracle](#) and ODBC connections with the [SQL Server](#) as an example.
2. Add the database connection details to the [database.cfg](#) file.
3. Use the DATABASE statement in the program, if you plan to connect to one database and use the CONNECT TO statement to connect to several databases. For more information about these statements refer to the "Querix 4GL Reference" guide.
4. (Optional) Set up the database driver. It will be used at compilation and at runtime, if the database connection details for the database name referenced by the application were not found in the database configuration file.

### Setting up Informix Connection

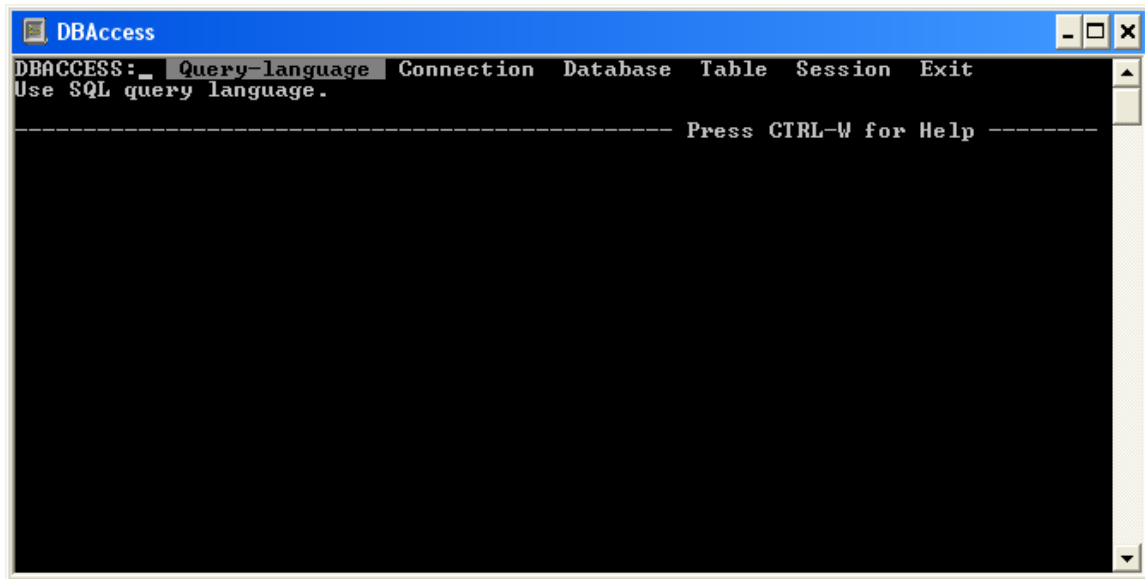
You can use an existing Informix database to work with 'Lycia' or you can create a new one. Here are the simple steps which can be used to create an Informix database and to connect to it using 'Lycia'.





## On Windows

1. Install Informix as per instructed in the Informix documentation and configure the Informix server
2. Create your Informix database using the DBAccess tool:



3. Install Informix Client SDK.
  - a. Use the Setnet32 tool to set up the server and make it the default server at the 'Server Information' tab.
  - b. Then set up the host at the 'Host Information' tab. The user name and the password must be valid on the database server specified. It is advisable not to use the usernames that do not require password for security reasons.
  - c. In the tab 'Environment' set the INFORMIXSERVER variable to the name of the server you made default. INFORMIXSERVER value should match one of the servers listed in the file \$INFORMIXDIR/etc/sqlhosts.
  - d. Save and close.
4. Record the database connection details into database.cfg file.
5. For the GUI server set the following environment variables in the inet.env file. The inet.env file can be opened and edited using the Studio. To open it in LyciaStudio, start the Studio, go to **Window-> Preferences -> 4GL -> Run/Debug -> GUI Servers**, select the GUI server from the list and press the **Edit Environment** button. The file will be opened in the editor area of the Studio. Set the following variables:

```
INFORMIXDIR = $INFORMIX_DIR  
  
INFORMIXSERVER = server_name  
  
LOGNAME = login  
  
INFORMIXPASS = password
```

Since the login and password will be stored in plain text, it is advisable not to set them in the inet.env file, but to pass to the program at runtime by means of the CONNECT TO statement.



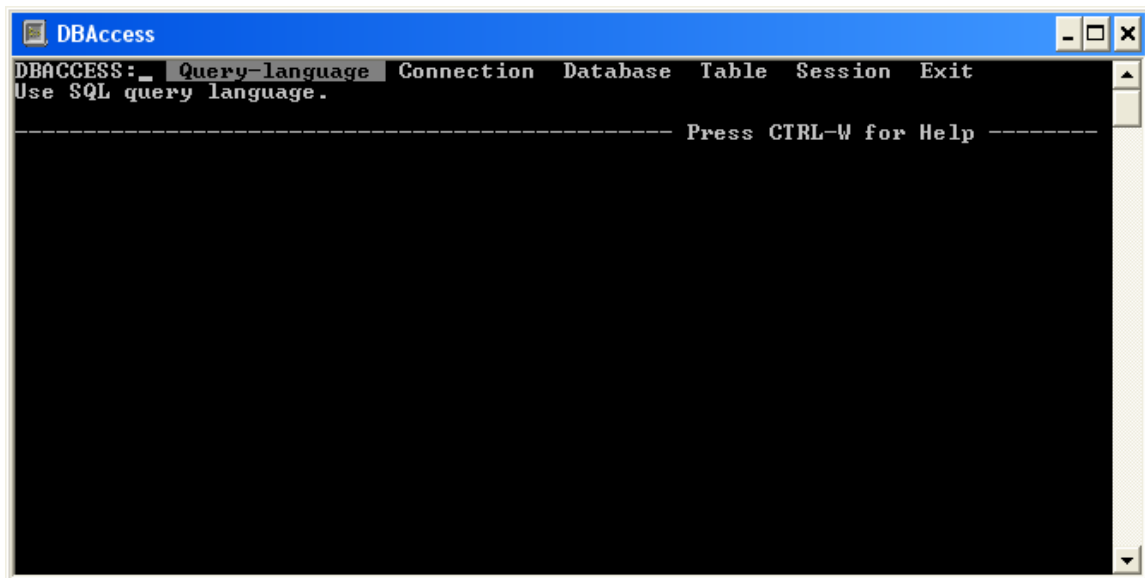
**Note:** Connecting to a database in GUI mode is rather tricky, if your application is running on an application server without authentication (i.e. 1889).

An application using this post is run as the service owner (LOCALSYSTEM) and this means that all database connections are using the same user ID. The Informix database bases the authentication on the user ID of the process and stores the connection information under the user's registry settings, thus it is advisable to use the full authentication port (1891).

6. (Optional) Set the database driver to "informix" as described in the "[Setting the database driver](#)" section below.
7. Restart LyciaStudio using the **File -> Restart** main menu option. LyciaStudio loads the environment variables from the files when it is started, they are not modified dynamically. Thus each modification of the environment variables which influence LyciaStudio must be followed by the Studio restart. If you use Lycia Command Line Environment, you do not need to restart it after having set the variables.

### On UNIX/Linux


1. Install Informix as per instructed in the Informix documentation and configure the Informix server.
2. Create your Informix database using the DBAccess tool:



3. Set Lycia process environment variables. The values of the variables will be used only if the database will not be found in the database.cfg file.
  - a. INFORMIXSERVER - should match one of the servers listed in the file \$INFORMIXDIR/etc/sqlhosts.
  - b. INFORMIXDIR - should point to the base location of the Informix product directory.
  - c. DBPATH (only for Informix Standard Engine) - specifies the search path that Informix will use to locate a database.



- d. Set the variables specified in step 3 also in the env.properties file. This file is located in \$LYCIA\_DIR\LyciaIDE\etc directory. The values of the variables will be used only if the database will not be found in the database.cfg file.
4. Record the database connection details into database.cfg file.
5. If you use the GUI server, set the same environment variables (specified in step 3) in the inet.env file. The inet.env file can be opened and edited using the Studio. To open it in LyciaStudio, go to **Window-> Preferences -> 4GL -> Run/Debug -> GUI Servers**, select the GUI server from the list and press the **Edit Environment** button. The file will be opened in the editor area of the Studio.

	<p><b>Note:</b> Connecting to a database in GUI mode is rather tricky, if your application is running on an application server without authentication (i.e. 1889).</p> <p>An application using this post is run as the service owner (querix product owner) and this means that all database connections are using the same user ID. The Informix database bases the authentication on the user ID of the process, thus it is advisable to use the full authentication port (1891).</p>
---	---

6. Set the database driver to "informix" as described in the "[Setting the database driver](#)" section below.


## Setting up Oracle Connection

You can use an existing Oracle database to work with Lycia II or you can create a new one. Here are the simple steps which can be used to create an Oracle database and to connect to it using Lycia II.

### On Windows

1. Install and configure the Oracle RDBMS. You must have the Oracle client installed regardless of whether you want to create a new database or to connect to an existing one.
2. Create a user under Oracle and provide the username and password by running the command 'sqlplus', and connect as a user with DBA privileges. Refer to Oracle documentation for the details on creating and managing a database.
3. Record the database connection details into database.cfg file.
4. For the GUI server running an authenticated port, set the following environment variables in the inet.env file:
  - a. ORACLE\_HOME (the installation directory of Oracle).
  - b. ORACLE\_SID (the default server instance for this connection).
  - c. DBTEMP (the directory for storing temporary files).

The inet.env file can be opened and edited using the Studio. To open it in LyciaStudio, go to **Window-> Preferences -> 4GL -> Run/Debug -> GUI Servers**, select the GUI server from the list and press the **Edit Environment** button.


	<p><b>Note:</b> Connecting to a database in GUI mode is rather tricky, if your application is running on an application server without authentication (i.e. 1889).</p> <p>An application using this post is run as the service owner and this means that all database connections are using the same user ID. Thus it is advisable to use the full authentication port (1891).</p>
---	--



5. Set the database driver to "Oracle" as described in the "[Setting the database driver](#)" section below.

### On UNIX/Linux

1. Install and configure the Oracle RDBMS. You must have the Oracle client installed regardless of whether you want to create a new database or to connect to an existing one.
2. Create a user under Oracle and provide the username and password by running the command 'sqlplus', and connect as a user with DBA privileges.
3. Set Lycia environment variables:
  - a. Set the environment variables ORACLE\_HOME (the installation directory of Oracle) and ORACLE\_SID (the default server instance for this connection). You may also need to set DBTEMP (the directory for storing temporary files) variables. The values of the variables will be used only if the database will not be found in the database.cfg file.
  - b. Set the same environment variables (as specified in step 3a) in the env.properties file. This file is located in \$LYCIA\_DIR\LyciaIDE\etc directory. The values of the variables will be used only if the database will not be found in the database.cfg file.
  - a. For the GUI server, set the same environment variables (as specified in step 3a) in the inet.env file. The inet.env file can be opened and edited using the Studio. To open it in LyciaStudio, go to **Window-> Preferences -> 4GL -> Run/Debug -> GUI Servers**, select the GUI server from the list and press the **Edit Environment** button. The file will be opened in the editor area of the Studio.

	<p><b>Note:</b> Connecting to a database in GUI mode is rather tricky, if your application is running on an application server without authentication (i.e. 1889).</p> <p>An application using this post is run as the service owner and this means that all database connections are using the same user ID. Thus it is advisable to use the full authentication port (1891).</p>
--	--

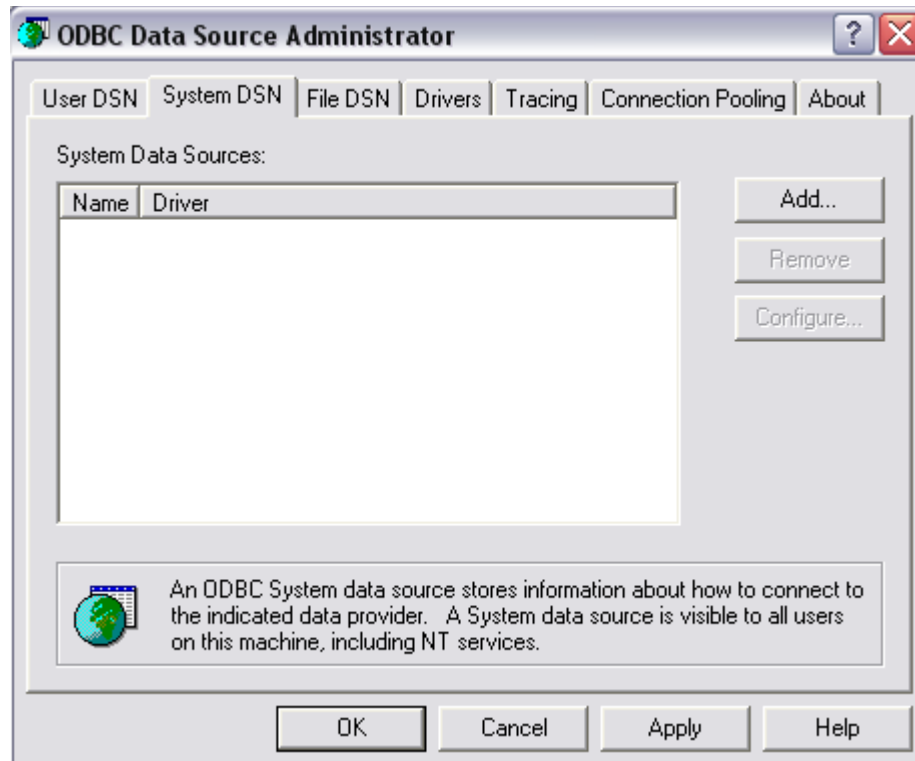
- c. Record the database connection details into database.cfg file.
4. (Optional) Set the database driver to "oracle" as described in the "[Setting the database driver](#)" section below.

### Setting up SQL Server Connection

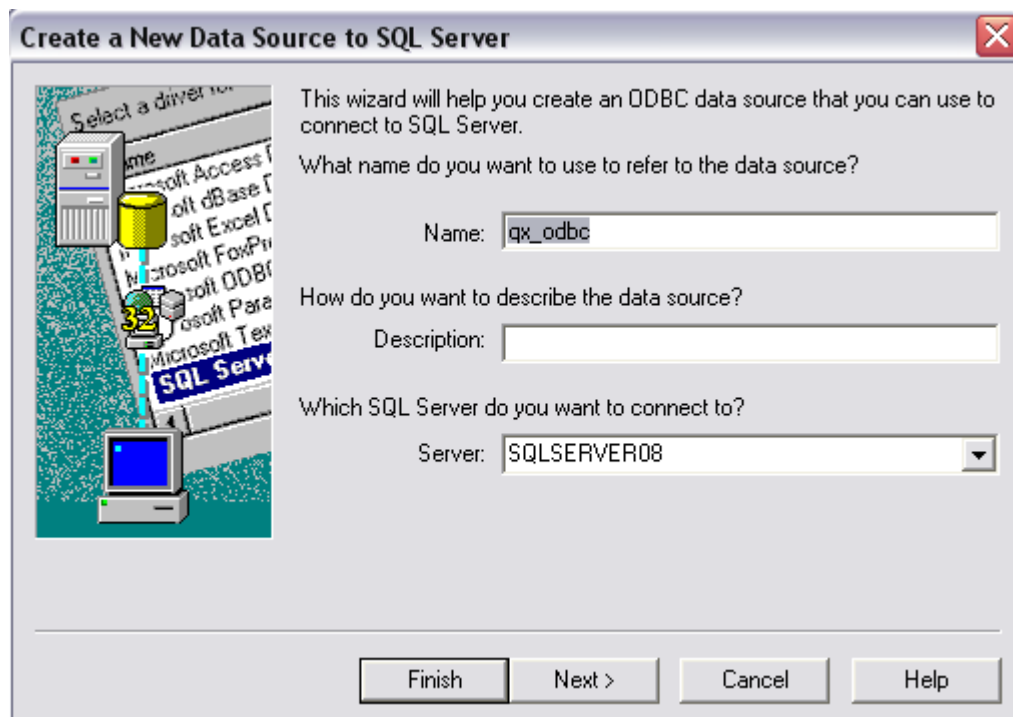
You can use an existing SQL Server database to work with 'Lycia' or you can create a new one. Lycia uses the ODBC connection to connect to SQL Server. Here are the simple steps which can be used to create an SQL Server database and to connect to it using 'Lycia'.

The steps below can be used for Windows only. To connect to SQL Server from UNIX/Linux use the FreeTDS ODBC driver for UNIX, which is a third party product used to provide SQL Server connectivity tools from UNIX.

1. Install SQL Server as instructed in the SQL Server documentation and create your database.
2. Configure your DSN using the Administration Tools/Data Sources (ODBC) manager located within 'Control Panel'.
3. Create a DSN using "System DSN" tab:




- Press the **Add** button and select the driver for the DSN.
- Then specify the name of the DSN (e.g. qx\_odbc) and select the server:





- c. Select the authentication mode, test the connection and close the ODBC Administrator window as well as the Connect tool by pressing **OK** buttons.
4. Record the details of the database connection to database.cfg file
5. Set the database driver to "sserver" as described in the "[Setting the database driver](#)" section below.
6. Restart LyciaStudio. using the **File -> Restart** main menu option LyciaStudio loads the environment variables from the files when it is started, they are not modified dynamically. Thus each modification of the environment variables which influence LyciaStudio must be followed by the Studio restart. If you use Lycia Command Line Environment, you do not need to restart it after having set the variables.

	<p><b>Note:</b> Connecting to a database in GUI mode is rather tricky, if your application is running on an application server without authentication (i.e. 1889). An application using this post is run as the service owner and this means that all database connections are using the same user ID. Thus it is advisable to use the full authentication port (1891).</p>
---	---

In some cases you may need to set the ODBC\_DSN variable. This variable should contain the name of the DSN you want to connect to, e.g. ODBC\_DSN=qx\_odbc. It is not required and you can omit it. The following actions are performed by the 4GL depending on the availability of this variable:

- If this variable is not set, 4GL considers the database name specified in the DATABASE statement to be the DSN. E.g. if you have DATABASE *stores* and no ODBC\_DSN variable, you will connect to DSN called "stores".
- If this variable is set, the keyword specified after the DATABASE statement is considered to be a catalogue within the database. E.g. If you have DATABASE *stores* and ODBC\_DSN=qx\_odbc, 4GL will connect to DSN called "qx\_odbc" and switch to the catalogue named "stores" after that.

This variable is set like all the other environment variables for the Studio, for the GUI and for the command line environment. For the detailed information about how and where to set the ODBC\_DSN variable see the "Database connectivity" chapter of the "Lycia Developers Guide".

## Database Configuration File

database.cfg file is created empty during Lycia installation. The file structure should be as follows:

```
dbname {  
  driver = "dbtype"  
  source = "databse"  
  username = "user"  
  password = "pwd"  
}
```

The options listed above should have the following values:

- database\_alias - the alias that will be used in 4GL code as the database name, for example, in the DATABASE statement or in the CONNECT TO statement



- driver - its values should be the database type. This parameter is optional, if it is not set the value of the LYCIA\_DB\_DRIVER variable will be taken as the driver. It can be one of the following keywords:
  - informix
  - oracle
  - odbc
  - sserver
  - db2
- source - its values should be the name of the database as it is specified on the database server. For Informix it can be database@server specification, for Oracle it should be tnsname. For the format of the database name see Source section in the 'Database Connectivity' chapter of the Lycia II Developer guide. This parameter is optional, if it is not set for the given database connection, the database alias is treated as the source.
- username - should contain the login of the user. This parameter is optional. If it is not set
- password - should contain the access password for the given user login. Note, that the password is stored as plain text. This parameter is optional. For the security reasons use the USING clause of the CONNECT TO statement instead.

The database.cfg file can contain a number of such database specifications, their number is not limited. Here is an example of this file containing two database specifications:

```
or_db {  
  driver = "oracle"  
  source = "stores"  
  username = "john"  
  password = "123456"  
}  
ifx_db {  
  driver = "informix"  
  source = "stores@ixserv"  
  username = "jane"  
  password = "567890"  
}
```

## Precedence of Parameters

The database connection can be set in the database.cfg file and then overridden by the CONNECT TO statement completely or partially. Since all the clauses of the CONNECT TO statement and the parameters of the database connection in the configuration file are optional, there may be a set of cases when some parameters are missing and some are set twice.

Generally speaking CONNECT TO has higher precedence than the database.cfg and database.cfg has higher precedence than the environment variables (i.e. LYCIA\_DB\_DRIVER) and flags (-d). The -d flag



and the environment variables have the lowest precedence in order to enable the usage of different database drivers throughout the program.

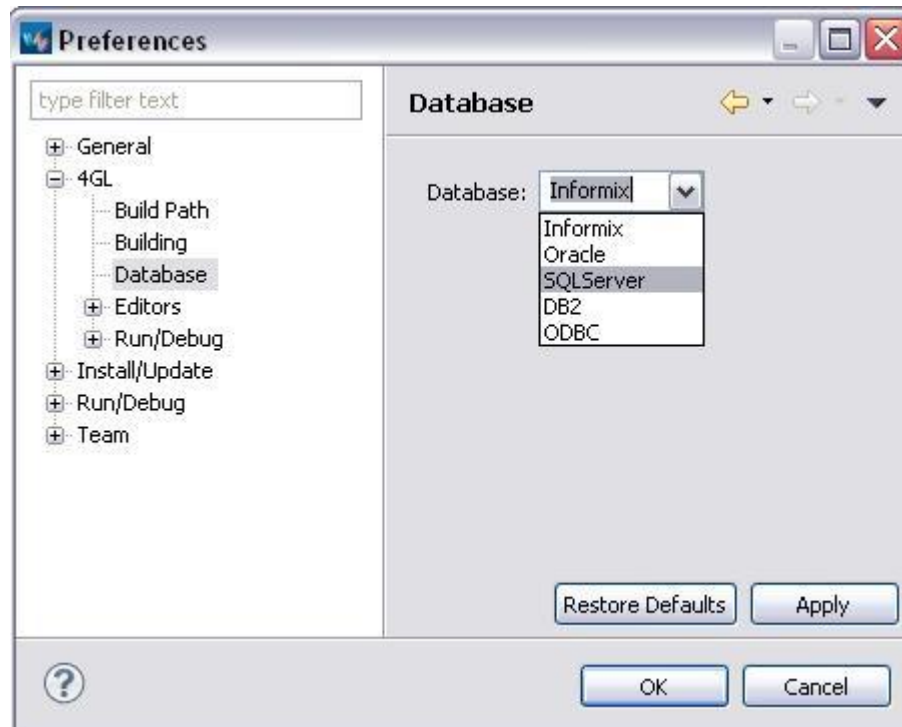
## Setting the Default Database Driver



**Note:** With the introduction of the multi-database connection in Lycia II the default database driver for runtime and compilation takes effect only if there is no corresponding driver specification in the database configuration file for the used database name.

### For compilation using LyciaStudio

To set the database driver, start LyciaStudio and go to **Window->Preferences -> 4GL -> Database** option as shown below:



From here, the developer can choose between a number of databases without the need to recompile after every change. This dialogue window modifies the env.properties file which stores the environment variables used during Studio compilation. The LYCIA\_DB\_DRIVER variable acquires the following values depending on the option selected:

- Oracle option – <oracle> value
- Informix option – <informix> value
- SQLServer option - <sserver> value
- BD2 option -<db2> value
- ODBC option - <odbc> value





## For running applications using the application server

For the applications executed directly via the application server (i.e. for the applications run in GUI mode), the setting will be taken from the inet.env file associated with that particular listener.

To set the database driver you should open the inet.env file and modify the LYCIA\_DB\_DRIVER variable. This variable should have one of the following values:

- <oracle> – Native connection to Oracle databases
- <informix> – Native connection to Informix databases
- <sserver> – ODBC connection to Microsoft SQL Server databases (Windows only)
- <db2> – ODBC connection to IBM DB2 databases
- <odbc> – General ODBC interface

To open the corresponding inet.env file do the following:

1. Go to **Window-> Preferences -> 4GL -> Run/Debug -> GUI Servers** preferences page.
2. Select the GUI required server from the list.
3. Press the **Edit Environment** button.
4. The inet.env file will be opened in the editor area of the Studio.
5. Edit the LYCIA\_DB\_DRIVER or add it, if it is absent. For example:

```
LYCIA_DB_DRIVER=sserver
```

6. Save the file and close it.



**Note:** Connecting to a database in GUI mode is rather tricky, if your application is running on an application server without authentication (i.e. 1889).

An application using this post is run as the service owner and this means that all database connections are using the same user ID. Thus it is advisable to use the full authentication port (1891).

## For the command line environment

If you compile and run your programs using the command line environment, you must set one of the following values to the LYCIA\_DB\_DRIVER environment variable to specify which database you want to connect to.

- <oracle> – Native connection to Oracle databases
- <informix> – Native connection to Informix databases
- <sserver> – ODBC connection to Microsoft SQL Server databases (Windows only)
- <db2> – ODBC connection to IBM DB2 databases
- <odbc> – General ODBC interface



To set this variable on Windows you can edit the `environ.bat` file located in `$LYCIA_DIR\bin` directory. In this case the specified database driver will be loaded each time Lycia Command Line Environment tool is launched. You can also set the value for this variable using the Lycia Command Line Environment, however, it will not be saved and used for another instance of the Lycia Command Line Environment. Use the "set" command to set a variable:

```
set LYCIA_DB_DRIVER=db2
```

On UNIX/Linux you should run the following command within a terminal/shell session or window from the Querix directory:

```
. environ
```

```
["dot" "space" "environ"]
```

Here you can change the `LYCIA_DB_DRIVER` variable.

The database driver may also be set by appending the `-DB` flag to an application on the command line, this overrides the variables in the environment settings.

For environment settings, please, see the "Configuration settings" chapter of the Lycia Developers Guide.



# CHAPTER 5

## Using LyciaStudio

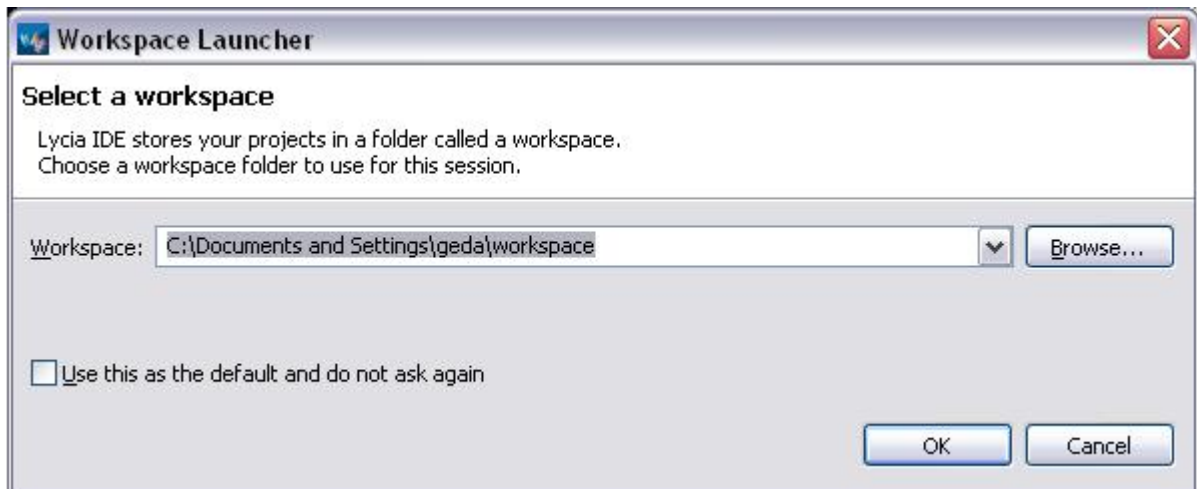
LyciaStudio is a simple to use, clean and organised development environment for both the development of new 4GL projects or on going maintenance of legacy 4GL projects. LyciaStudio runs from a Java environment, meaning it can be run on any platform which supports Java.

### Running the Studio

After installing Lycia from the installer, on Windows a link to the Studio is made available under Start Menu -> Programs -> Querix -> Lycia. From here, the user can either run "LyciaStudio" or "Lycia Command Line Environment". LyciaStudio will load the Studio interface, whilst the Command Line Environment will open up a command prompt.

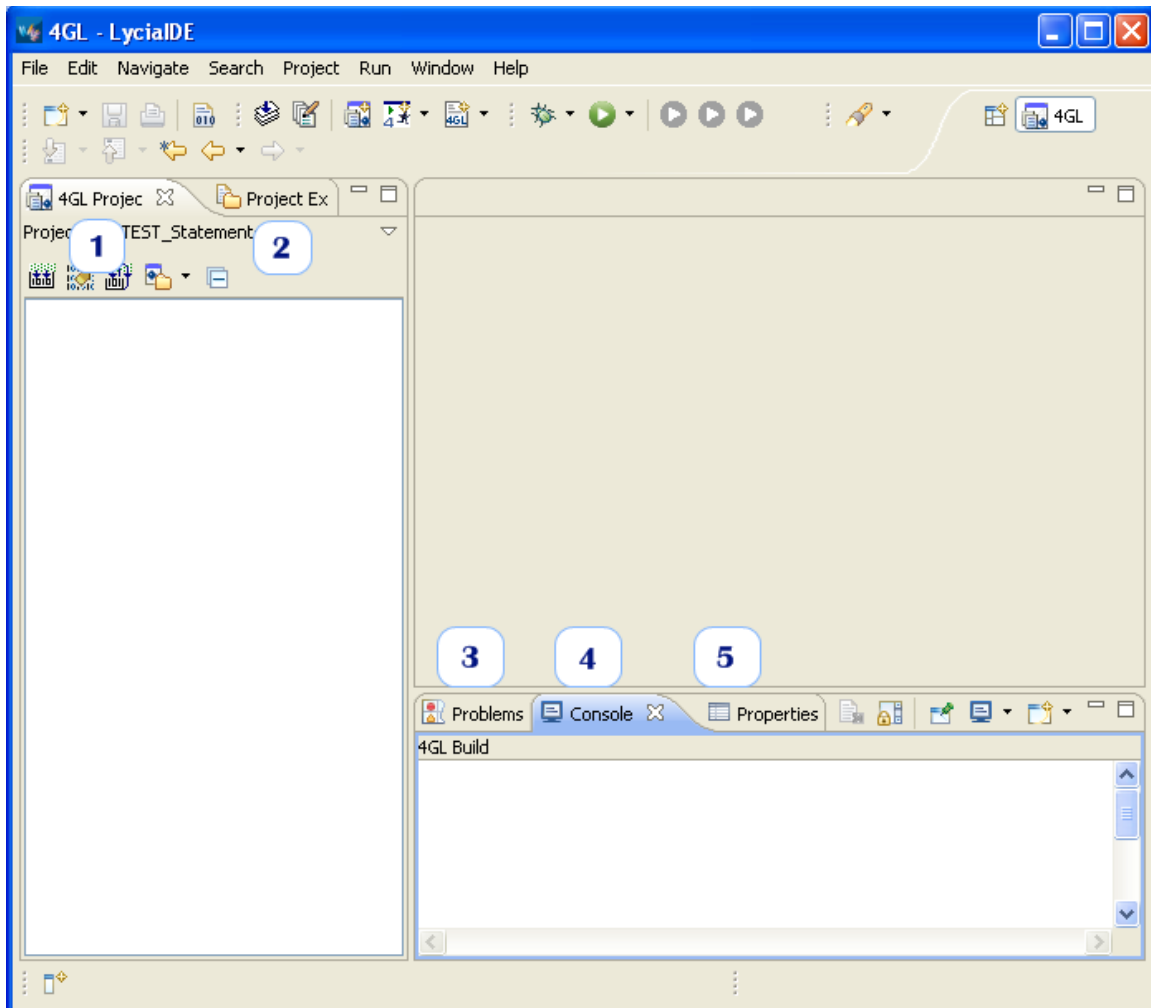
On Unix, simply change to the \$LYCIA\_DIR directory and execute lyciaide/lycia-ide.

After running the LyciaStudio program, you will be presented with the following popup:



The default path here is where Lycia recommends you install your 'workspace', which is where the projects and source code the developer wishes to work on will be stored. Developers can have more than one workspace which can aid project organisation, or all projects can be stored within just one workspace.

Click OK and the Studio will load the specified workspace, which will look like the following:



As you can see, this has the same basic layout as previous Querix versions but with additional features, such as a source code viewer in the Project Explorer, a console where the developer is kept up to date with compilation issues and the flexibility of more than one project can be opened within one instance of the Studio.

- 1) 4GL Project – allows the developer to see the contents of each of their programs and libraries
- 2) Project Explorer – allows the developer to see both the output and source files for a program without their relation to a program being visible. Many projects can appear here. The Project Explorer allows the developer to see the programs in a structure similar to Windows Explorer.
- 3) Problems – any errors during compilation appear here
- 4) Console – any information messages from the build system are displayed here
- 5) Properties – lists any properties of a selected item within the Studio

By default, the 4GL Project View tab obscures the Project Explorer View tab. This is because, for most developers, they will not need to use the Project Explorer tab as most of their work is in 4GL and everything 4GL related is worked on in the 4GL Project View tree. The Project Explorer tree and tab are built in parts of Eclipse, and whilst being needed for some non-4GL projects, will not be used by most developers when working with 4GL alone.

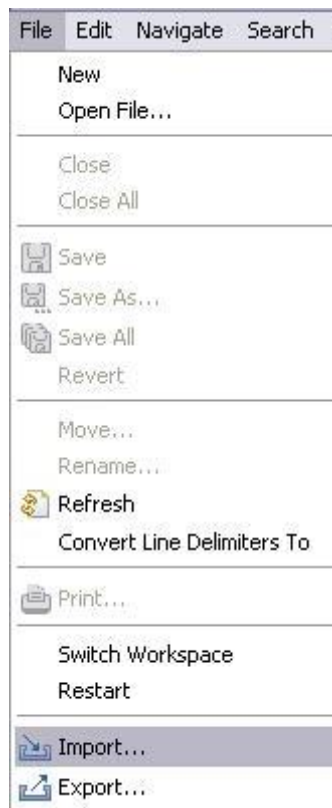


When the developer double clicks on an open tab, such as the currently worked on 4GL file, the tab will be displayed full screen. Another set of double clicks on the same tab will restore the view to its original size.

## Importing a legacy 'HydraStudio' project

To work with legacy 'HydraStudio' projects, a developer would need to import the project into the LyciaStudio. This is a simple process, as follows:

1. Start the Studio
2. Click on File -> Import



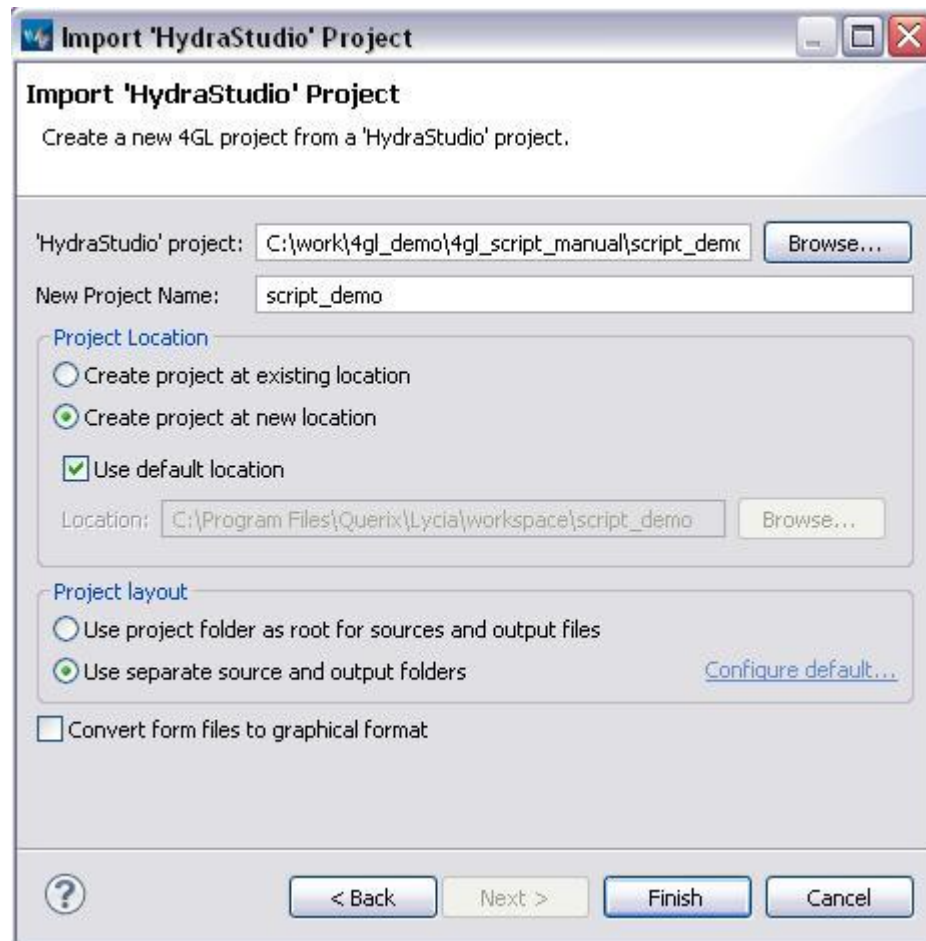


3. Select "4GL/Legacy HydraStudio Project" and click Next





4. Browse for the project you wish to import (for this example, we will pick the script\_demo from the Querix online store)

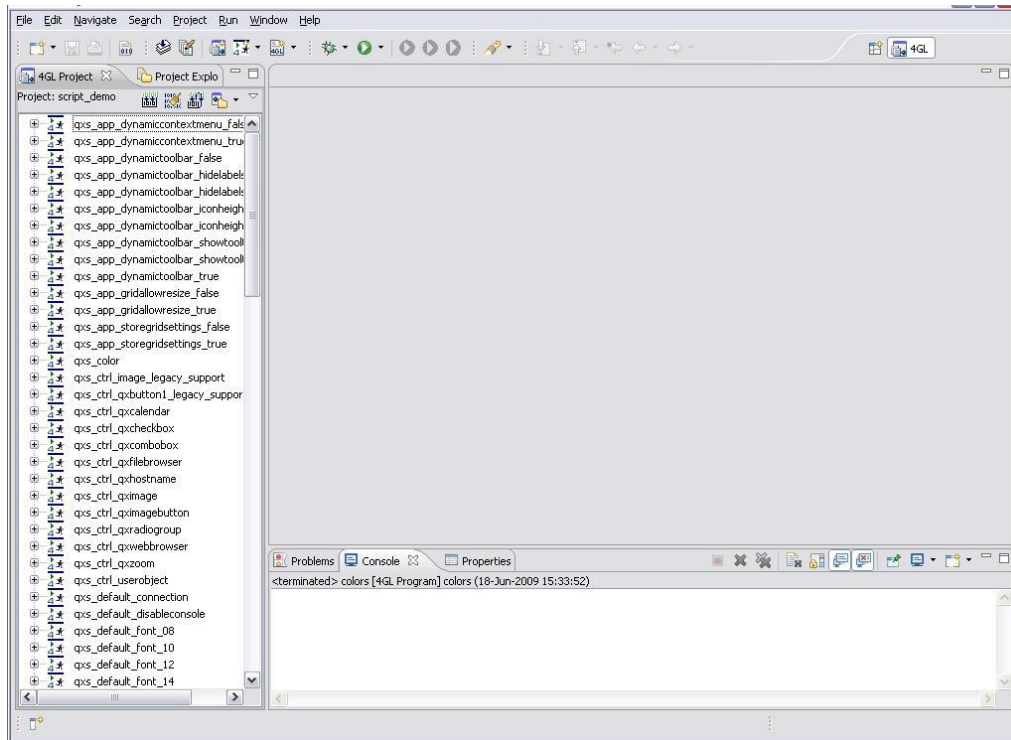


Note: form files are **NOT** automatically converted into a graphically editable format by default (.4fm). If you wish to edit them graphically, you will need to convert them.

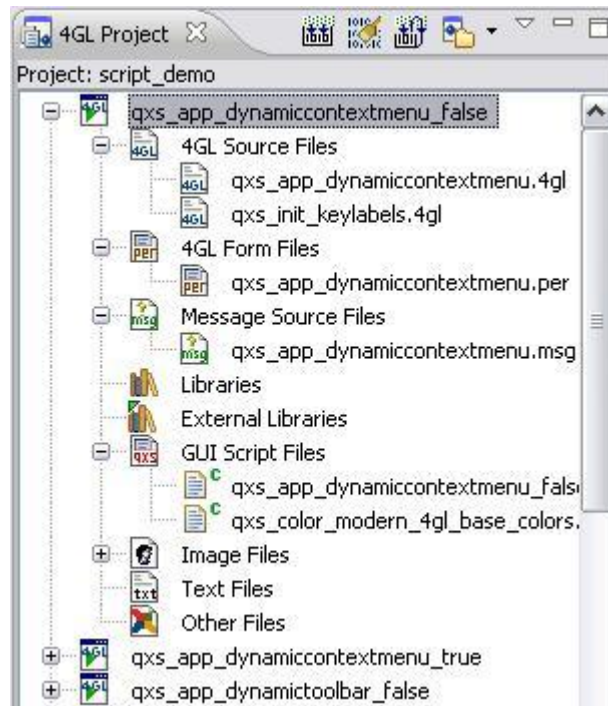
To do this, either click the tick box option "Convert form files to graphical format" as seen above, or individually within the 4GL Project viewer by right-clicking a form file and selecting from its menu "Convert to graphical format" at any point and can also be reverted to the .per format.



## 5. Click Finish



You can now work with your previous projects as you wish. The structure of an imported project will remain the same as it would have within 'HydraStudio', as shown below. Expanding a program in the '4GL Project' view will allow you to edit the files as you desire.

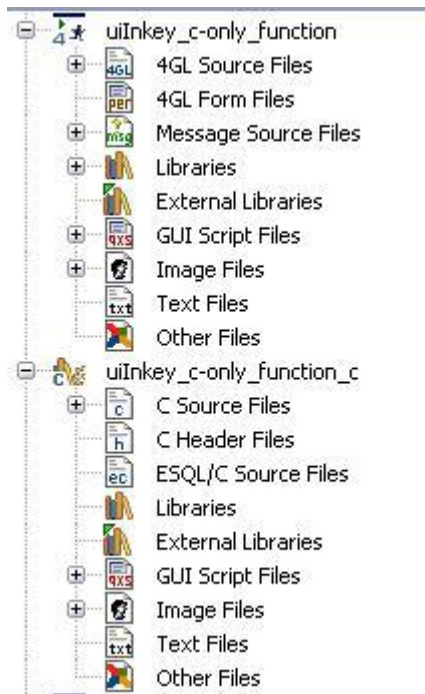






## 4GL and C programs

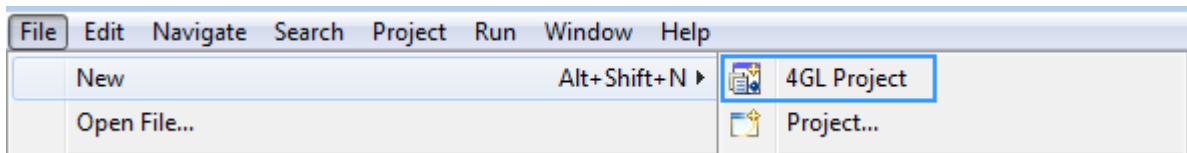
A legacy build target containing both 4GL and C will be split into two build targets when imported into the Studio. If your program contains both 4gl and C sources, the program will be split into both the original program and a C library. The C sources will be transferred to the C-library.





## Importing other legacy projects

1. Start the Studio
2. Create a new project from File -> New -> 4GL Project



This will create a project which can be seen under the Project Explorer tab.

3. Select File -> Import and change to "General"



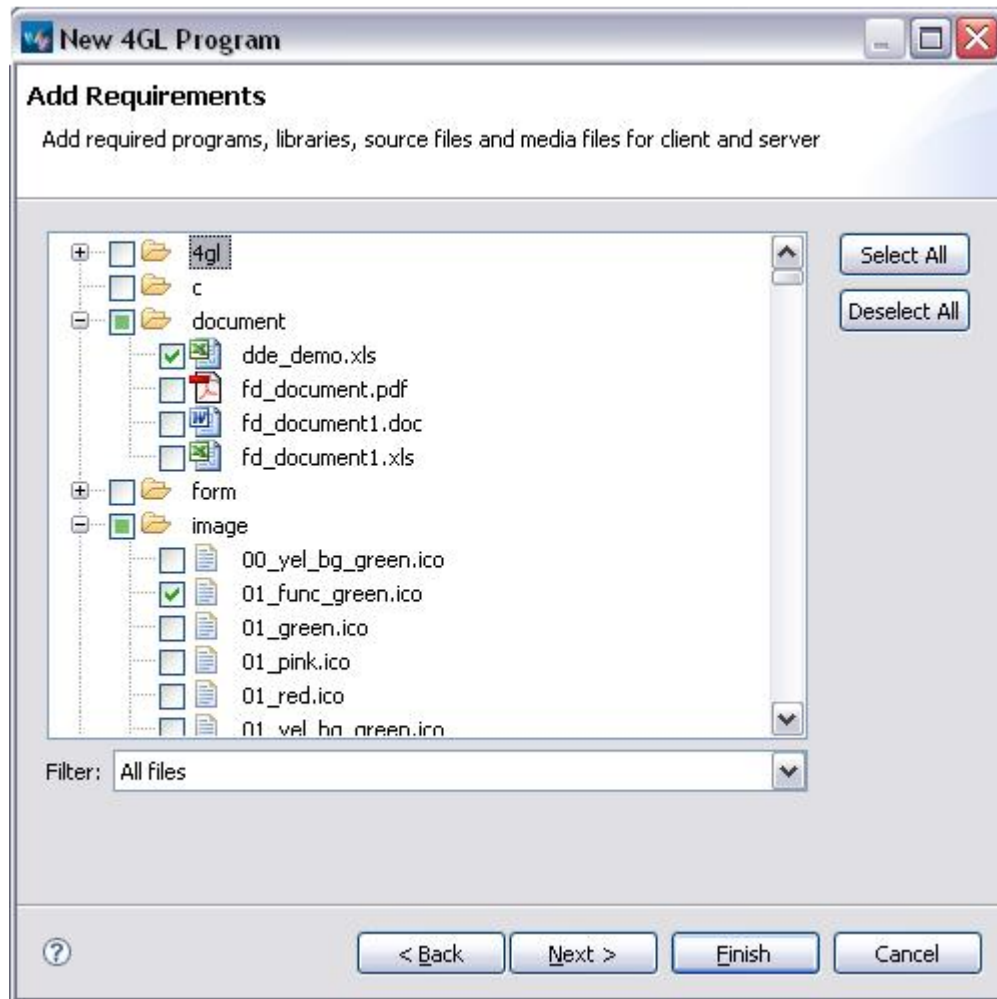


Note: you can either import a file at a time, or you can import an entire directory structure at once. To do this, use "File System" as your selection, and click on Next.

- Specify the directory of the legacy project in the "From Directory" field.  
Select which files are required for the import from the right hand list, ticking each required.  
Next specify the "Into Directory" (if not already set).  
Click on Finish.



- The Studio then takes these files and creates a project, maintaining the structure from the previous project.
- Create a new program within this project with File -> New -> 4GL Program
- Name your program, choose the same location as before for the source folder and click Next
- Under the Add Requirements section, select all files which will be needed by this project.



As you can see from above, we have selected a spreadsheet and an icon file to be added. This will then set a dependency between this project and the files selected.

9. Add any External Libraries if required on the following screen
10. Click Finish

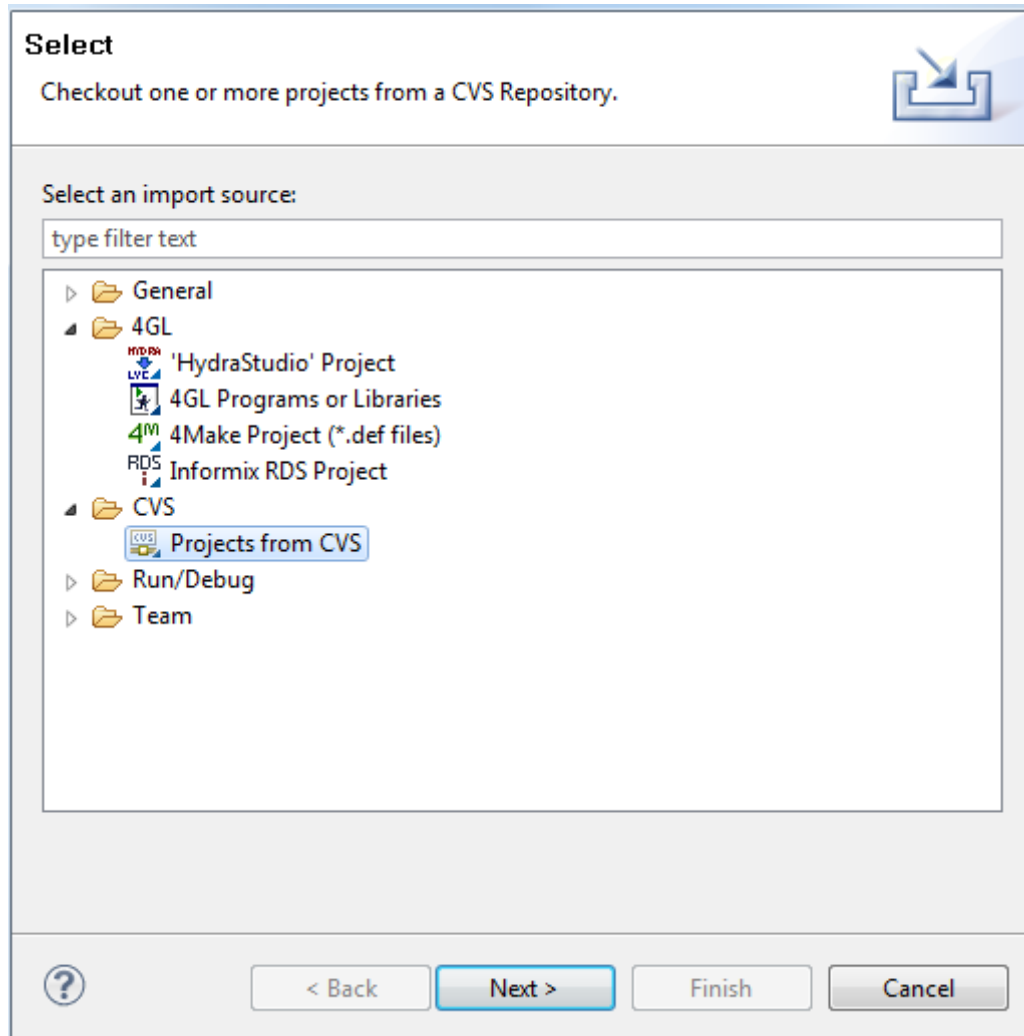


## Importing demonstration applications

You can obtain Lycia demonstration projects from a CVS repository located at [demo.querix.com](http://demo.querix.com). Lycia contains a built-in functionality allowing you to check out projects from a repository from within the Studio and with no additional tools needed.

To obtain the demonstration projects:


1. Start LyciaStudio
2. Go to **File -> Import** in the main menu.
3. Select **Projects from CVS** in the dialogue which will appear and press **Next**:



4. In the next dialogue you will need to enter the connection details for the Querix CVS. Enter the details as follows:
  - 'Host': `demo.querix.com`
  - 'Repository path': `/lycia`
  - 'User': `anonymous`
  - 'Password': `<blank>`
  - 'Connection type': `pserver`



### Enter Repository Location Information



Define the location and protocol required to connect with an existing CVS repository.

Location

Host:

Repository path:

Authentication

User:

Password:


Connection

Connection type:

☒ Use default port

☐ Use port:

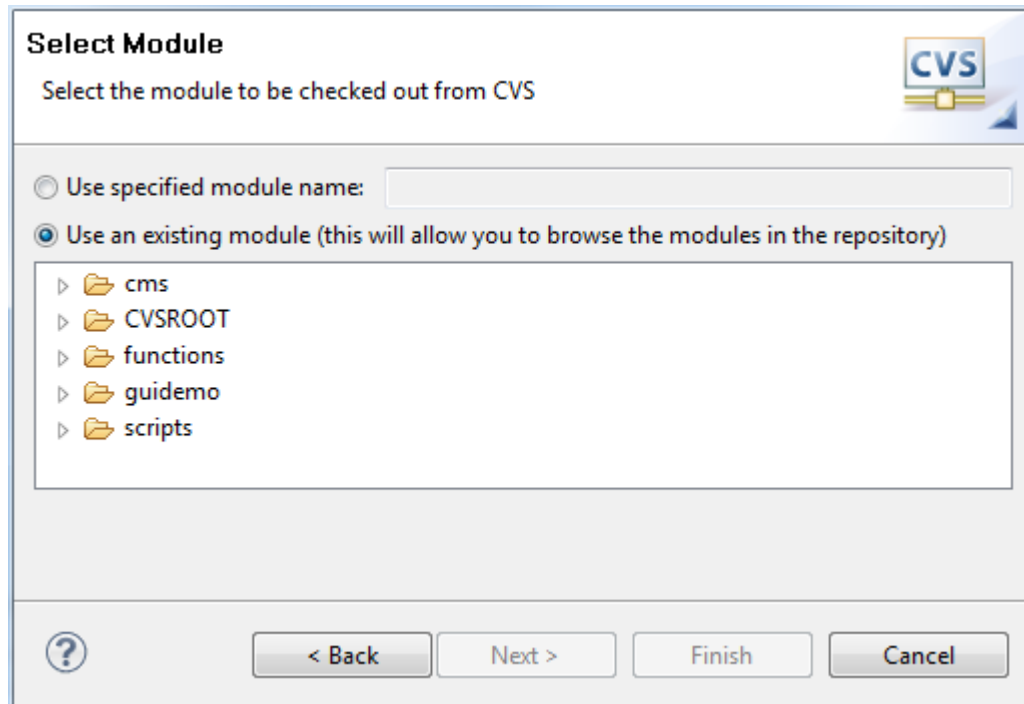
☐ Save password (could trigger secure storage login)  
org.eclipse.equinox.security.ui.storage not found  
[Configure connection preferences...](#)



5. Leave the port as default and click **Next**.
6. You will be presented with another dialogue where you need to select the module to import.



7. Select the "Use an existing module" option and you will see the list of modules available for downloading as shown below:



8. Select the modules you want to download by clicking on them. If you want to select several modules at once, hold Ctrl while clicking.

**Note:** do not select the CVSROOT module; this is a module managed by CVS and it is not a demonstration application.

9. Then you can click **Finish** (which is recommended), or click **Next** to select the project preferences. It is advisable to leave the settings default and click **Finish**.

Now you should have the selected demo projects copied to your workspace and available in the Project Explorer and 4GL Project navigation views.

## Creating a new project

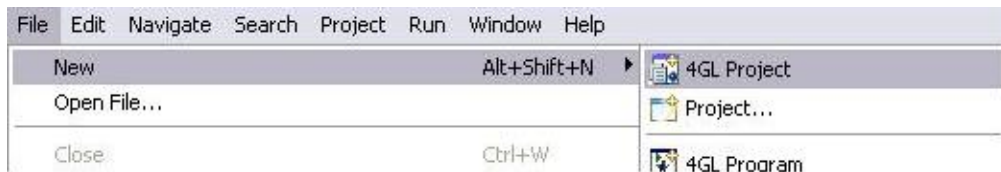
To demonstrate how to make a new project, we are going to use a simple example called hello.4gl, which will print to the screen "Hello World" and run using our thin client, LyciaDesktop.

	<p>Please note, within the Studio there is more than one method to adding new programs/files to a project. <b>This method takes you through the main menu system, but simply right-clicking on a program will also offer the same functionality.</b> This demonstration will show you both screen shots, first the menu-based option, followed by the right-click option.</p>
--	---

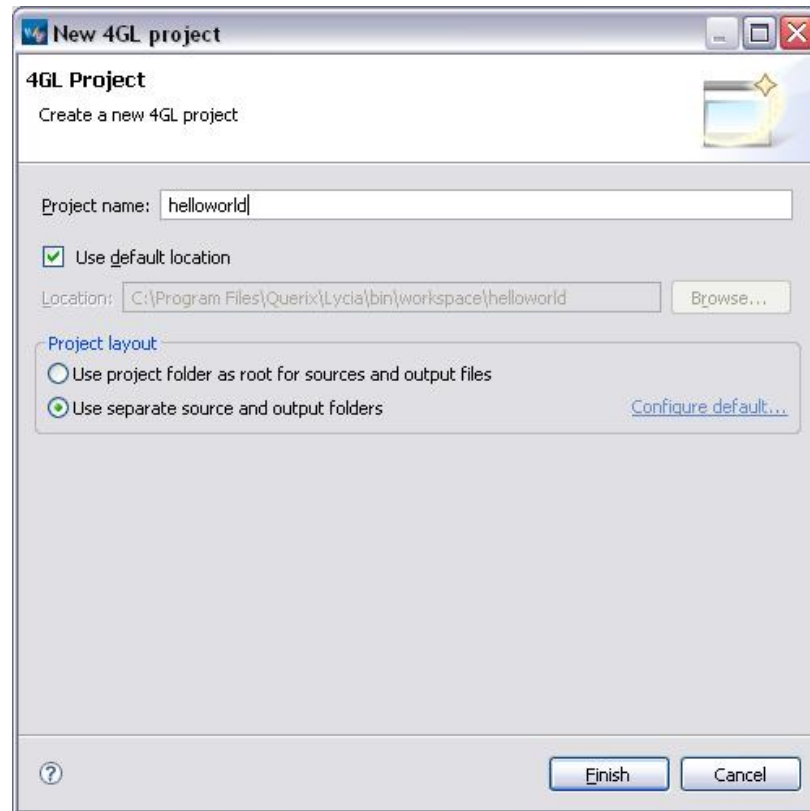
1. Start LyciaStudio



2. Click on File -> New -> 4GL Project

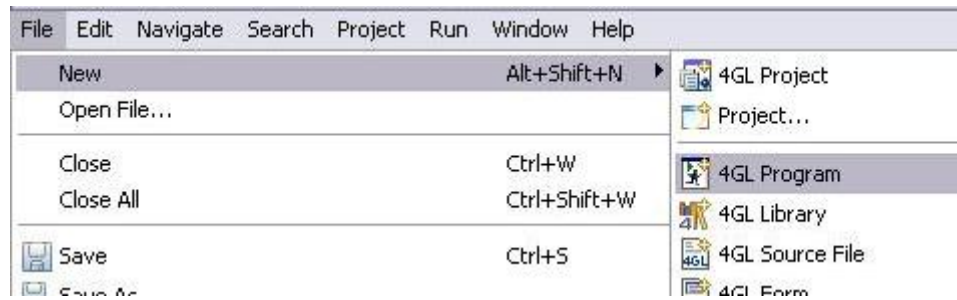


3. Give the project the name *helloworld*, leave the default options and click Finish



4. Click on File -> New -> 4GL Program







5. Give the program the name *hello*, leave the default options and click Finish



6. Click on File -> New -> 4GL Source File



7. Give the source file the name *hello*, leave the default options and click Finish

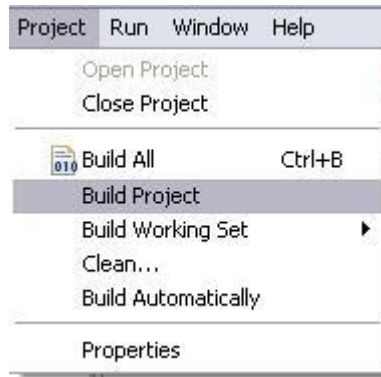


8. Add the following code to the source file and save:

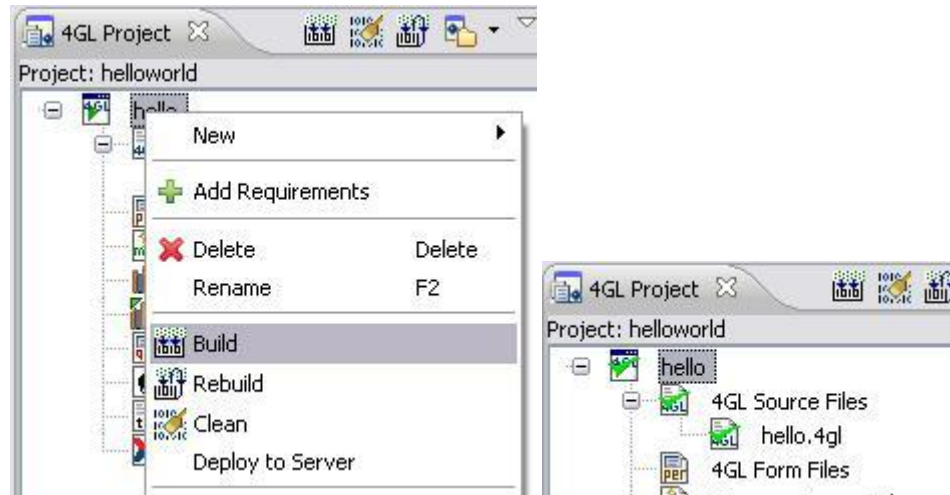
```
MAIN
DISPLAY "Hello World" AT 2,2
SLEEP 3
END MAIN
```




9. Select Project -> Build Project:



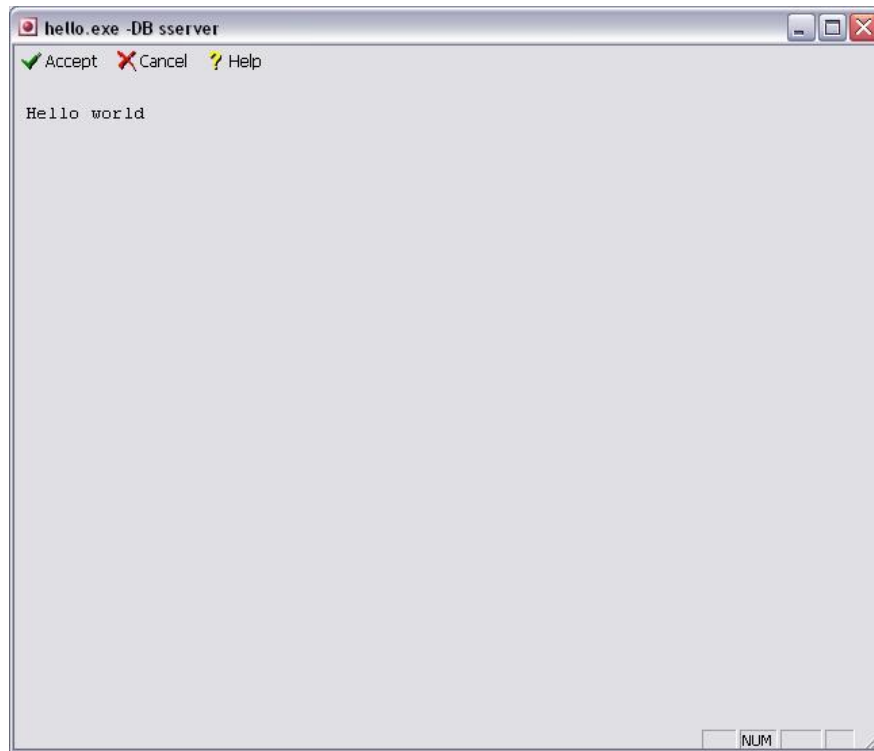
Or right-hand click on:



10. Run the program:

- Right-click on the program *Hello* and select **Run -> Run As -> 4GL Program with LyciaDesktop**
- Or click on  button in the main toolbar.
- Or right-click on the program in 4GL Project view and select **Run As -> 4GL Program with LyciaDesktop** option from the context menu.

11. You will get a LyciaDesktop window as below:

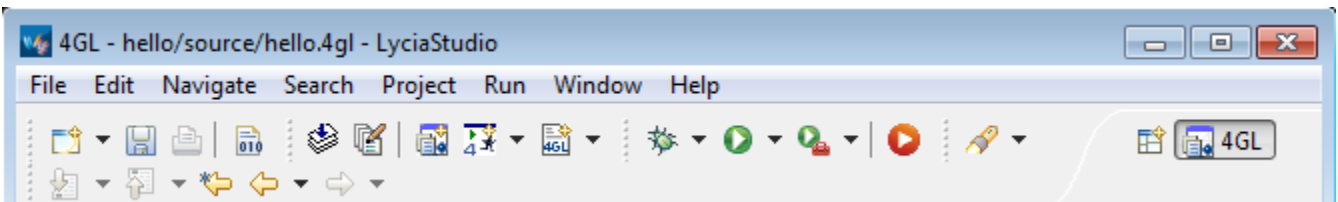




## LyciaStudio Functionality and Debugging
















### Explaining the LyciaStudio toolbar

The main toolbar of LyciaStudio looks like the following:






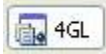


The main toolbar consists of the toolbar buttons and a tab with the buttons that switch perspectives.

We will now explain what each element of the toolbar's functionality below:

-  - The 'New' wizard (Alt-Shift-N) – the same as clicking on File -> New
-  - Save (Ctrl-S) – a shortcut to the 'save' functionality
-  - Print (Ctrl-P) – a shortcut to the 'print' functionality
-  - Build All (Ctrl-B) – this builds all programs in an open project
-  - Compile the selected source file – compiles the file open in the editor
-  - Build Automatically – enables automatic building
-  - Create a new 4GL Project
-  - Create a new 4GL Program – with options for files within this such as 4GL library, C Library and Web Service
-  - Create a new source file – can be 4GL, C, form file, message file etc
-  - Debug As (F11) – activates the debugger, which will be explained in its own section
-  - Run As (Ctrl-F11) – 'run as' settings can be changed here. Has a MRU top 5 list to choose from, explained more in its own section
-  - External Tools - allows you to create launch configurations for non-4GL programs (e.g notepad) and launch them using this button
-  - Run with LyciaDesktop – runs the current program in LyciaDesktop
-  - Search - opens the search dialog which allows you to search for lines of code and for files within your workspace
-  - Next Annotation (Ctrl+.) - takes you to the next annotation. The annotations can be selected from the drop-down menu of this button, these can be: breakpoints, bookmarks, tasks, errors, warnings, etc.




-  - Previous Annotation (Ctrl+,) - takes you to the previous annotation. The annotations can be selected from the drop-down menu of this button, these can be: breakpoints, bookmarks, tasks, errors, warnings, etc.
-  - Last edit location – takes you to the last place an edit was made
-  - Back to <file name> (Alt+Left) – opens and brings forward a previously viewed file
-  - Forward to <file name> (Alt+Right) – opens and brings forward the next file in the list of recently viewed files. It is active only if you have previously pressed the Back to <file name> button
-  - Open Perspective – opens a list of available perspectives
-  - The button which brings forward the 4GL perspective when there is more than one perspective opened. The buttons of other open perspectives can be located next to it.

## The Debugger / Debug As...

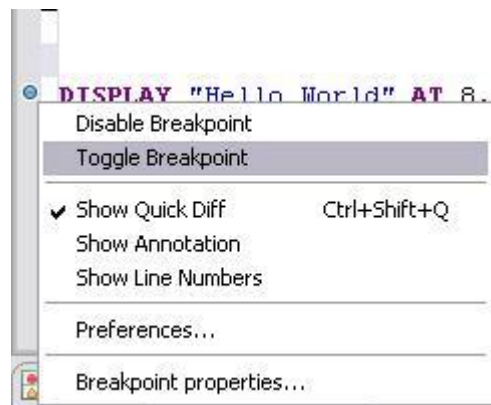
Any program compiled using Lycia can be debugged using the LyciaStudio. There are two ways to start debugging a compiled program through the Studio:

1. Right click the program in the '4GL Project' and select 'Debug As':

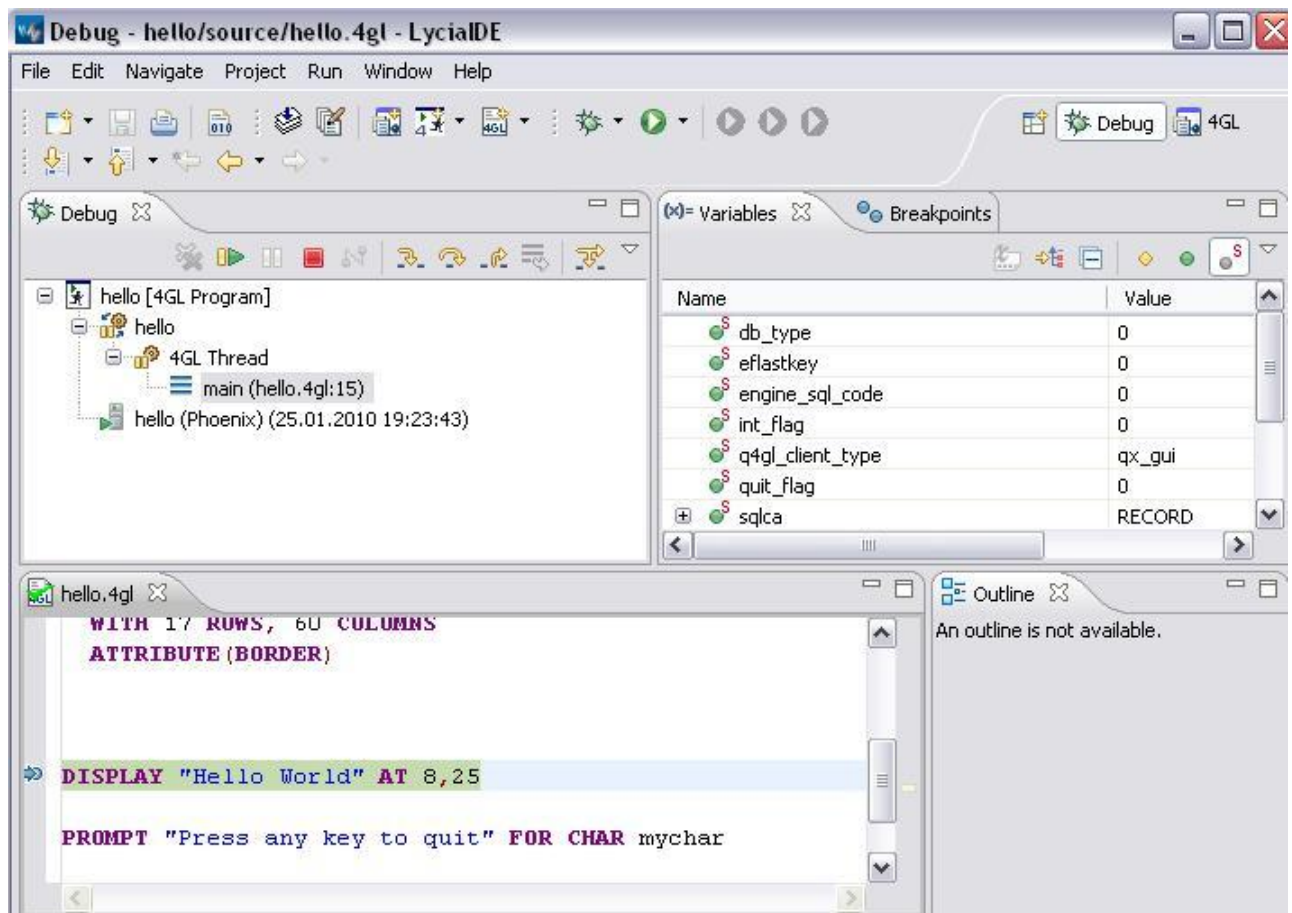


2. Use the  'Debug As...' button on the tool bar.

When debugging, breakpoints are placed in the source code where the developer requires them. You can place a breakpoint by double-clicking on the vertical ruler in the editor next to the line at which you want the execution to stop. To remove the breakpoint, double-click it once more or use the context menu of the breakpoint called by right-click:



The program is then run in debug mode, where the code will “stop” and the perspective changes to the debug view:



From here, you can see the processes running in the selected application and it will pause at the first breakpoint it finds. The example above shows that the execution of the hello program has been suspended at the line with the breakpoint.

The developer can then "step through" the program line by line until they can find the fault, or "step over", which executes a function without stepping into it.

The main functions within the debugger are:

1. Resume (F8) – resumes the execution of a suspended program
2. Suspend – suspends program execution when not waiting for user input
3. Terminate (Ctrl+F2) – kills the program being debugged
4. Step Into (F5) – steps into the function call on the current line – executes every line within a function before moving onto the next line
5. Step Over (F6) – steps over a function call on the current line – executes the function without stepping into the function.
6. Step Return (F7) – used to return from a function which has been stepped into

Watchpoints are similar to breakpoints. A watchpoint is attached to a variable or an expression which is evaluated continually as the program is executed. Program execution is suspended when the value





of the evaluated expression changes. To create a watchpoint on a variable, right-click a variable in the 'Variables' view and select 'Create Watchpoint'.

It is also worth noting that when a breakpoint is set on a variable, the value of the variable can be changed, so the developer can simulate certain situations to see what happens if a value is ever hit.

You can find more detailed information on the process of debugging in the "Lycia Developers Guide" in the chapter called "Using the debugger".

## Run As


Running a program from LyciaStudio can be as simple as clicking one button. The Run As... option stores the most recently used programs which will run in the last format you ran them in.

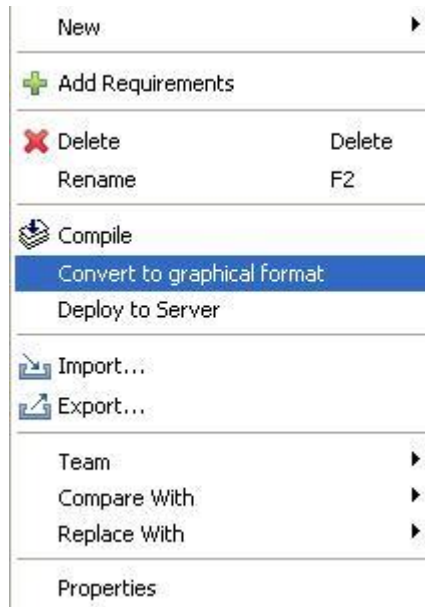
It is worth noting at this point, when double-clicking a program from within the 4GL Project viewer, it will run this program with the last known configuration – this means that if it was last run in LyciaDesktop, it will run in LyciaDesktop yet again. This can be easily changed by either right-clicking and choosing another option from "Run As..." or by changing the configurations from the "Run Configurations" menu option, as seen above.

## The Form Editor

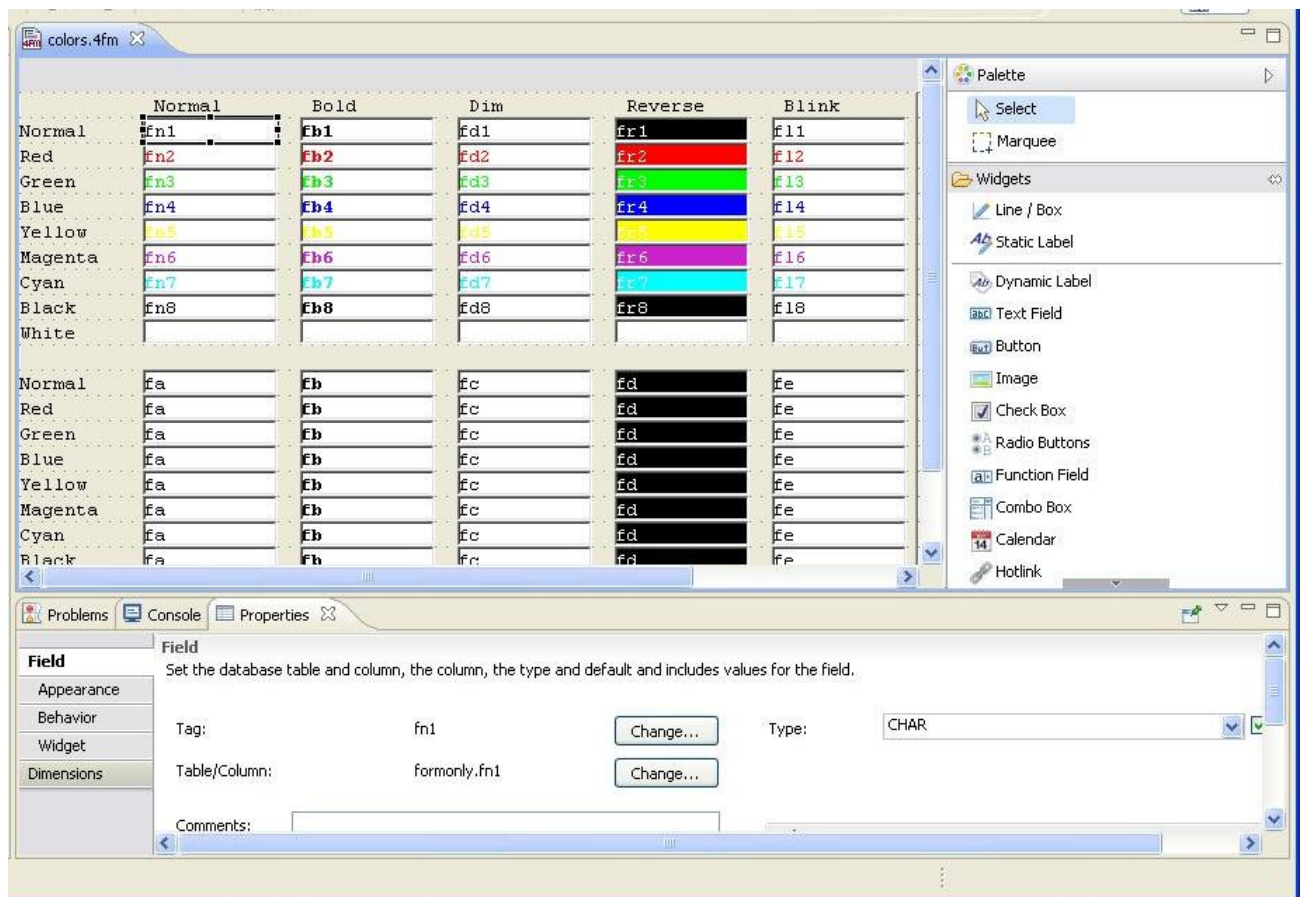
Lycia incorporates a new graphical form editor to allow developers to design or enhance their form files with ease into the development suite. There is still the option to create form files in a text format (.per files) but to edit files graphically (.4fm files) they will need to be either created originally in this format, or converted using Lycia's conversion tool, which will preserve all conditional data and form details and keep the form looking just as before.

To convert a form to the graphical format right-click the form file in the 4GL Project view and select **Convert to graphical format** option.

	<b>Note:</b> You cannot convert graphical form files back to the text format regardless of whether they were initially created in .4fm format or converted into it.
---	---



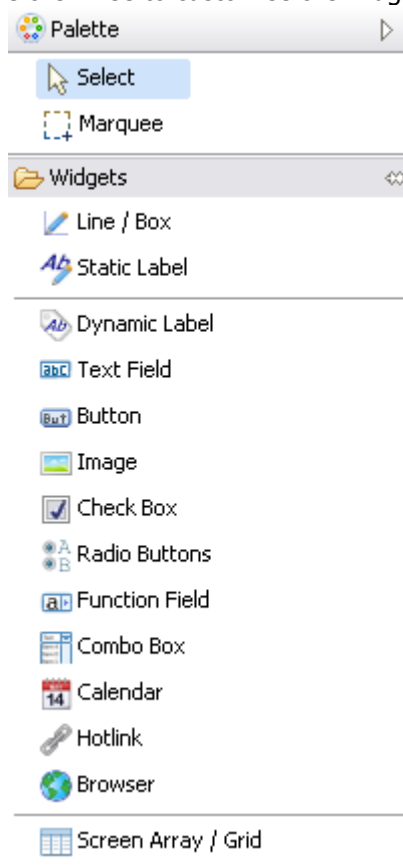
A form in the graphical format looks as the following:



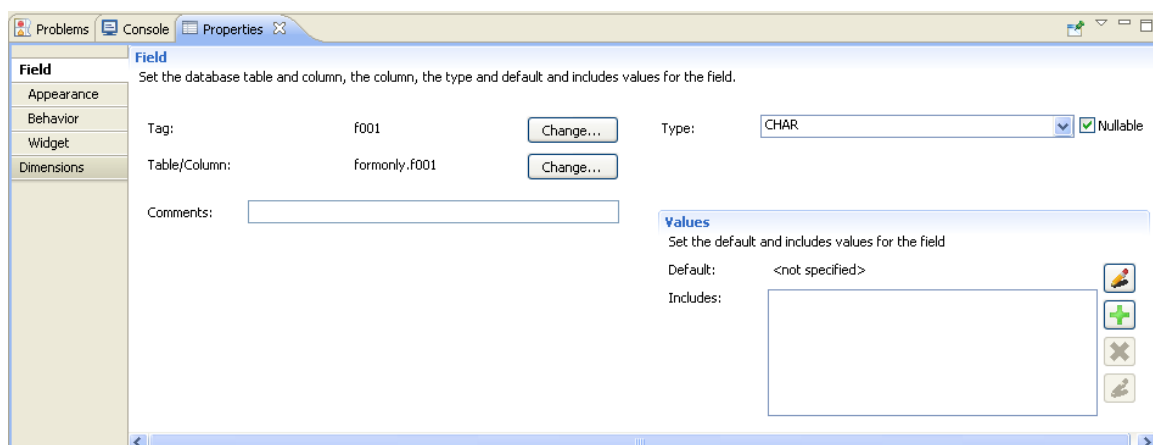
To add a new widget to a form in the graphical form editor, the palette needs to be used. The palette can be found to the right hand side of the form editor tab, and is shown below. To use the palette, the



developer simply clicks on the widget required once, then clicks where they would like the widget positioned upon the form. They are then free to customise the widget as they see fit.



The properties of a widget placed on the form can be viewed and changed using the Properties view which is opened below the form editor tab automatically, when a .4fm file is opened. The Properties view contains several tabs, the Form, Screen Records, Toolbar buttons and Database tabs refer to the form in general whereas other tabs appear when you select a widget. The Widget tab contains widget-specific options; other tabs are common for all the dynamic widgets and are used to specify the design, names and other common widget features. The Properties view looks like the following:





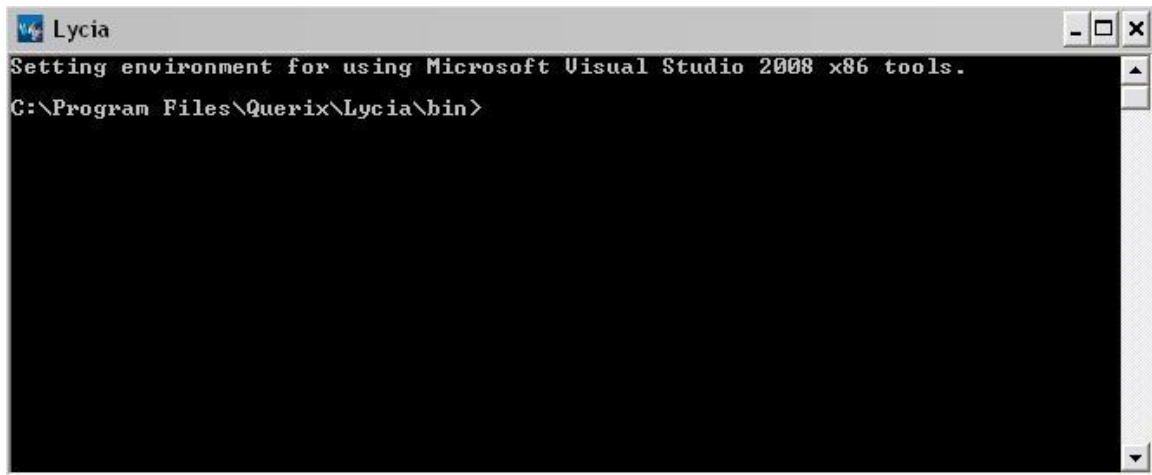
For details about how to use the new graphical form editor refer to the “Lycia Developers Guide” to the chapter called “Using the form editor”, which includes the detailed description of the widgets and the guide for creating a sample form file.



## CHAPTER 6

### Using the command line

The command line environment is invoked in Windows from the **Start Menu -> Querix -> Lycia**



On Unix, before working with the command line tool, the environment must be set. This can be done as follows:

```
cd <directory where Lycia is installed>  
./environ
```

taking care to note the space between the first "." and the second "."

### qfgl

The qfgl command is used to compile a 4gl source file into a p-code module. To compile a 4gl source code file, the following command would be called:


```
qfgl a.4gl
```

which would output a p-code module called a.4o

```
qfgl -o b.4o -DB sserver a.4gl
```



would output a p-code module called b.4o, and the 4gl file will be validated against a SQL Server database

	<p>Note: the <code>-DB</code> flag is only used for validation purposes during compilation. The compiled program can be run against any database server available, either by executing <code>qrun</code> with the <code>-DB</code> flag, or by setting <code>LYCIA_DB_DRIVER</code> environment variable.</p> <p>If a database is not specified, the system will attempt to use the default DB driver specified at installation.</p> <p>For full database support, please refer to our Database Connection guides, available to download from our website.</p>
---	--

## qform

The `qform` command is used to compile form files from the command line. From the command line, it compiles `.per` files to `.pic` files, `.per` files to `.4fm` files, and `.4fm` files to `.pic` files.

```
qform hello.per
```

would compile the file `hello.per` into the `hello.pic` file.

## qmsg

The `qmsg` command is the message compiler, it generates compiled message files of the name specified. Since there is no enforced convention on message file names, Lycia does not require the extension of the input file or output file to take any specific values.

```
qmsg a b
```

where `'a'` is the input file, and `'b'` is the output file.

## qlink

The `qlink` command is used to link files together. To link 4gl files together, the following command would be called:

```
qlink -o x.4a a.4o b.4o c.4o
```

where `a.4o`, `b.4o` and `c.4o` are statically linked together to form the p-code module `x.4a`

```
qlink -o x.4a a.4o b.4o -lib c.4o
```

would mean that `a.4o` and `b.4o` are statically linked into `x.4a`, whereas `c.4o` is dynamically linked to `x.4a`



## qrun

The qrun command is used to execute P-code modules using the P-Code runner. To run a 4gl program, the following command would be called:

```
qrun a.4a
```

which would run the application a.4a created after linking the object files

```
qrun -DB sserver a.4o
```

would run the application a.4o against the database SQLServer

The qrun tool can run .4o files if a program consists of one file and does not require linking.

By default, P-code modules will be linked with an appropriate loader for the system upon which they were compiled. This means that there is no need to invoke the P-Code runner on the platform that the P-Code module was compiled.

For example, under Windows we can run a P-Code executable simply as:

```
my_program.exe
```

Under Unix, we can run a P-Code executable as:

```
../my_program
```

Even though P-Code executables can be run in this manner, they are still portable between different operating systems. When moving a P-Code module between Windows/Unix, however, it will become necessary to run them directly through the P-Code runner, e.g.

```
qrun my_program
```



# CHAPTER 7

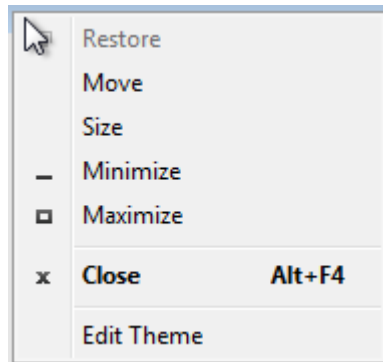
## Look and Feel of your Applications

Lycia II offers you a new tool named Theme Designer, that is aimed at easy customization of the look and feel of your applications. Compared to the script files used earlier, Theme Designer has a number of appreciable advantages.

The section below will provide you with a brief overview of Theme Designer possibilities. For more details concerning Theme Designer, please, refer to "Theme Designer Guide". Besides description of all the available elements, filters and properties, it includes the Theme Designer Tutorial section with the theme syntax description and practical examples of applying properties to elements.

### Launching the Theme Designer

You can launch Theme Designer from within Lycia Studio as a standalone tool or any application at runtime opened with Lycia Desktop .NET as a part of this client. If you chose the last variant, you should right-click the title bar of your application and select the **Edit Theme** option from the contextual menu:



In this way the Theme Designer will be opened to edit the default theme file which exists for any program created in Lycia II or imported into it. This file has .qxtheme extension and its name is the same as the name of the program. It is uploaded to the application server together with the program. There is also the possibility to use non-default theme files and more than one for a single program. For the details see the "Theme Designer Guide".

For the demonstration purposes you can use a simple 'Hello World' program below to see how the Theme Designer is used:

```
MAIN
DEFINE ch CHAR
OPTIONS PROMPT LINE LAST
DISPLAY "Hello World!" AT 7,15

# During the execution of this PROMPT you can right-click on the
# titlebar and select the EditTheme option.
```





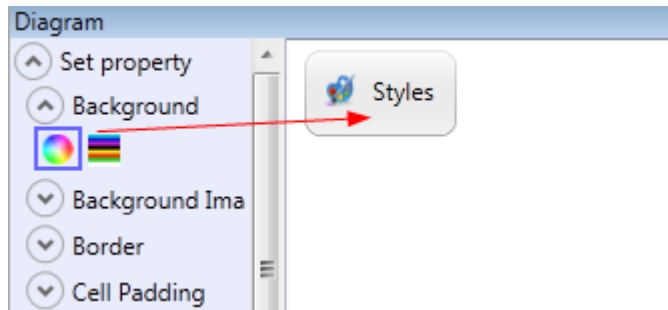
```
PROMPT "Press any key to open Theme Designer " FOR CHAR ch
END MAIN
```

Invoke Theme Designer to customize your application.

## Changing the background

To change the background colour of the whole program do the following:

1. Unfold the Background property, select the custom colour and drag it to the grey area called Styles.



2. From the drop-down menu select the background colour:

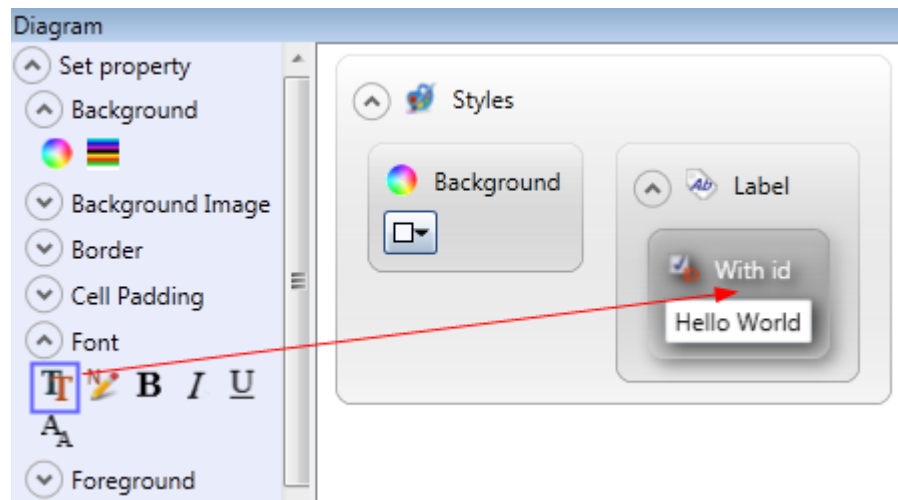





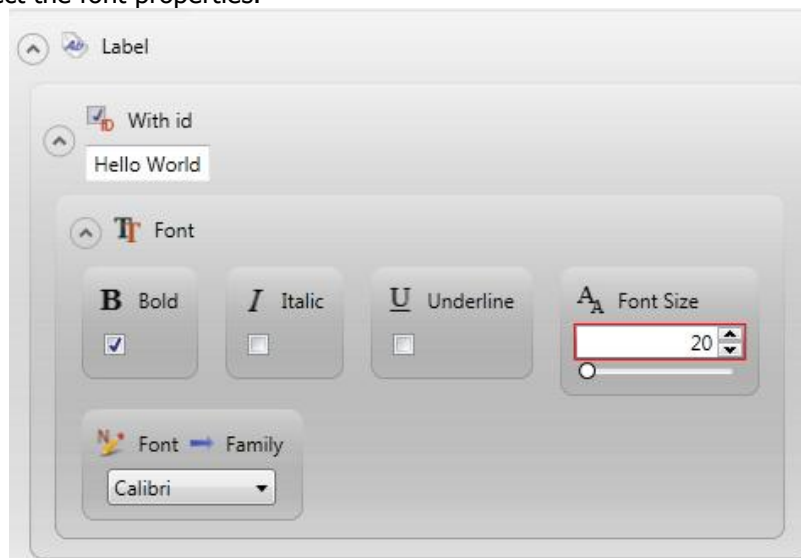
## Changing the font

You can change the font for the whole program: this will be the default font. You can also change the font for a particular element. In this case we will change the font of the "Hello World!" string whereas the font of the PROMPT statements will not be affected.

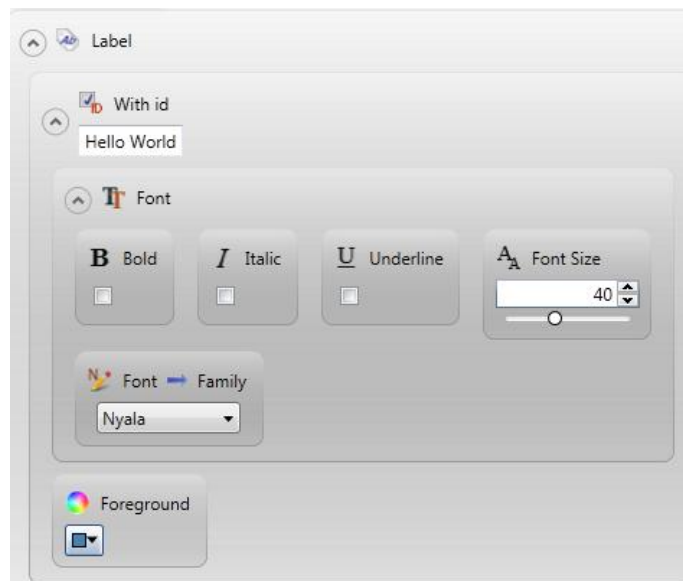
1. Add the Label element to the Styles view and place the With ID filter within it. Print "Hello World" in the empty field.
2. Unfold Font option and drag New Font to the With ID object:



3. Drag the Family option (  ) intoside the Font section to change the font itself and not only its weight or size.
4. Now select the font properties.




5. To change the font colour drag the Foreground colour option inside the With ID section, but outside the Font section. Select the colour.

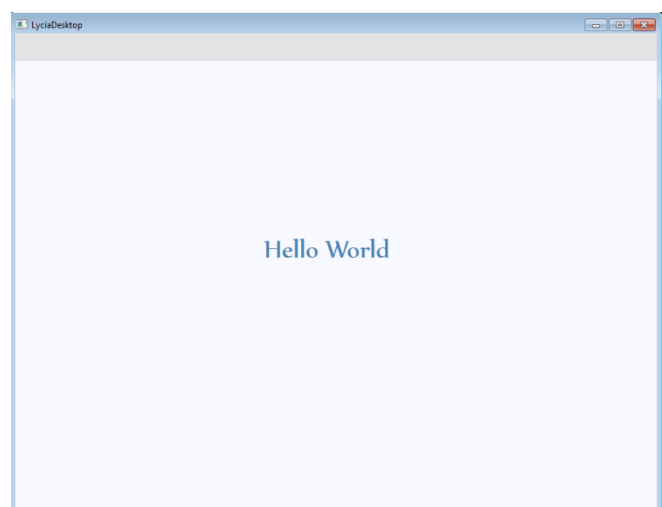
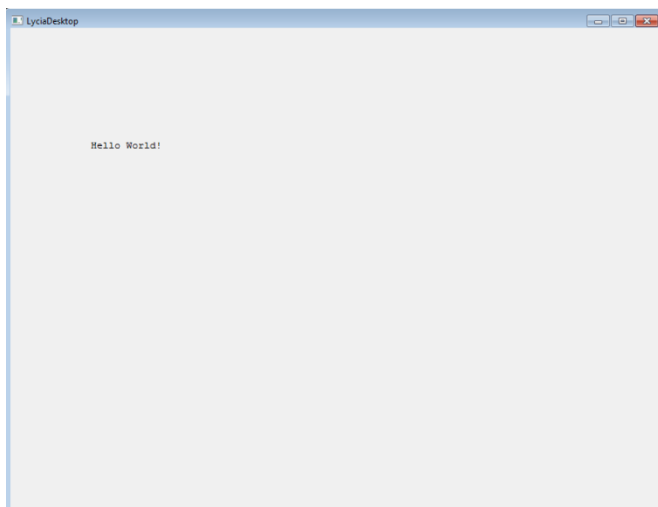


## Moving objects around

You can move any objects around, for example we can make the 'Hello World' string better centred:

1. Drag the Location option (  ) inside the Label -> With ID section.
2. Move the sliders for X and Y coordinate to move the string. The effect will be visible in the application window immediately. The coordinates should be given in pixels and not in rows and columns. The coordinates given are not relative to the current position of the element but to the screen or window.

The application will look as follows before and after all the changes made:





## Finishing the Editing

To close the Theme Designer, select **File -> Exit** menu option. You will be prompted to save the theme file.

If you select to save the theme, the modifications will be applied to the program. If you launch the application once more, the selected display features will be preserved. If you select the Edit Theme option again, the theme file with all the previously selected settings will be opened and you will be able to change them.

If you select to discard the changes, no display attributes will be applied when the program resumes and the theme file will remain empty.

The Theme Designer comprises much more display attributes and other features both inherited from script files and completely new ones. They are described in "Theme Designer Guide".



## Responsive Design

To make your application look and feel properly, when run on both desktop and mobile devices, you need to make some modifications within the process of its creating.

In order to achieve an optimal view of an application, that supposes easy-to-read and navigate through a close to an original program layout with a minimum of scrolling, resizing and panning the corresponding classes should be used.

### Full-screen Mode

We provide you with a possibility of a user interface customization in several ways, without time-consuming recoding every time the user intends to utilize another device or thin client. These methods are described below.

#### The STYLE attribute

Adding a special argument to an ATTRIBUTE clause results in a full-screen view of a window it specifies, run both on desktop and mobile devices.

The syntax of such a clause is as follows:

```
ATTRIBUTE (BORDER, STYLE="full-screen")
```

#### where:

*full-screen*                      a STYLE attribute value, standing for a special class name, that enables the full-screen mode

#### Usage:

```
OPEN WINDOW w_full_screen WITH FORM "window_full_screen"  
ATTRIBUTE (BORDER, STYLE="full-screen")  
  
# opens the window titled "w_full_screen" in a full-screen mode
```

#### ui.Interface.getFrontEndName() method

This method is generally used to retrieve the type of the front-end. Depending on client or device an application is being run on, it may return different values.

If an application is run on a smartphone, tablet or any other mobile device, the method returns *lyciamobile* string.

The syntax of the method invocation is as follows:



```
CALL ui.Interface.getFrontEndName() RETURNING lyciamobile
```

**where:**

*lyciamobile* a returned value indicating, that an application is running on a mobile device

**Theme style modification**

The user interface may be adapted for the device utilized by the user by means of Theme Designer.

It allows an easy deployment of a number of special classes that adjusts an application layout depending on the client or device. Here is the list of the available classes with an indication of the front-end an application is expected to be opened in:

<i>lycia_normal</i>	browser or desktop client, NOT on a mobile device
<i>lycia_web</i>	browser
<i>lycia_desktop</i>	LyciaDesktop
<i>lycia_touch</i>	a standalone mobile application
<i>lycia_mobile</i>	a mobile device (browser or a standalone application)
<i>lycia_html5</i>	HTML client, that includes all three - web, desktop, mobile

for browsers:

<i>lycia_firefox</i>	Mozilla Firefox®
<i>lycia_opera</i>	Opera®
<i>lycia_safari</i>	Safari®
<i>lycia_ie</i>	Windows Internet Explorer®

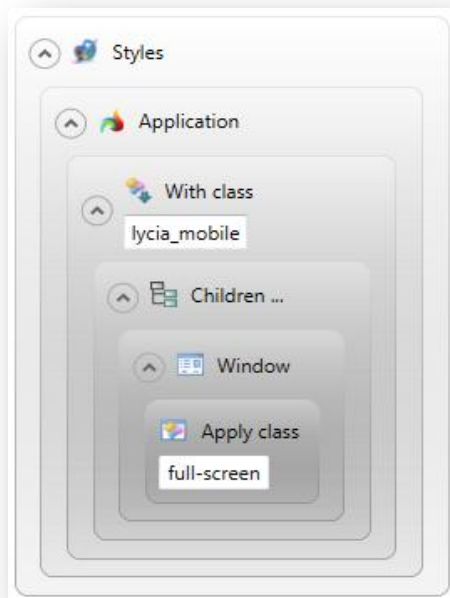
for mobile operating systems:

<i>lycia_iphone</i>	iOS
<i>lycia_android</i>	Android

Below is an illustration of how a *full-screen* class can be applied to all the windows of an application filtered with the *lycia\_mobile* class filter. Note, that such a modification, based on applying the *lycia\_mobile* class filter to the theme, effects an application only if it is run on a mobile device:



## Theme Styles diagram



## XML source code

```
<ElementFilter ElementName="Application">
  <StyleSheet>
    <WithClassFilter ClassName="lycia_mobile">
      <StyleSheet>
        <ChildFilter>
          <StyleSheet>
            <ElementFilter ElementName="Window">
              <StyleSheet>
                <DoStyleAction>
                  <ApplyClass Name="full-screen" />
                </DoStyleAction>
              </StyleSheet>
            </ElementFilter>
          </StyleSheet>
        </ChildFilter>
      </StyleSheet>
    </WithClassFilter>
  </StyleSheet>
</ElementFilter>
```

## Orientation

One of the most important aspects of an application building for various devices is taking into consideration possible screen orientation and user interface changes this may cause.



In order to save your time when coding, we advise you to set an `OnOrientationChanged` event, that is application scoped. It is triggered each time a mobile device is rotated by the user. For its handling you should use the `ON ACTION` clause. Its syntax is as follows:

```
ON ACTION ("orientation_name")
```

where *orientation\_name* is a string, standing for the orientation name and can be one of the following *landscape*, *portrait*, *portraitup*, *portraitdown*, *landscapecw* or *landscapeccw*.

In order an application could know how the user interface should be changed depending on a screen position (horizontal or vertical) the *fgl\_getproperty* function can be used.

Depending on the property name passed to it, the function can return the following values:

```
fgl_getproperty("gui", "application.deviceorientation")
```

- returns an integer, that stands for the mobile device rotation degree: 90, -90 for the landscape and 0, 180 for the portrait orientation.

```
fgl_getproperty("gui", "application.deviceorientationname")
```

- returns a string that stands for the orientation of a mobile device. The last can be one of the following:

<i>portraitup</i>	screen is rotated 0
<i>portraitdown</i>	screen is rotated 180
<i>landscapecw</i>	screen is rotated -90
<i>landscapeccw</i>	screen is rotated 90







## CHAPTER 8

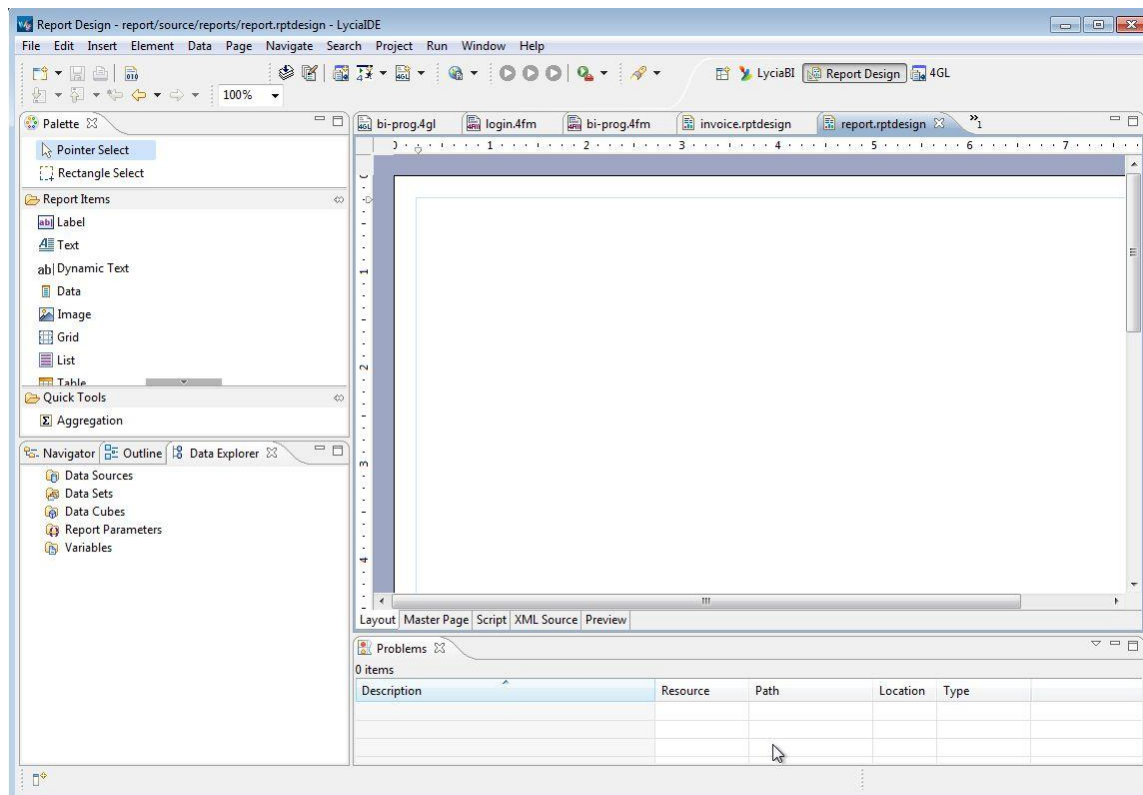
# Graphical Reports

LyciaBI is a tool used to create graphical reports based on your database data. It is built-in into LyciaStudio and LyciaBI server where the reports accessible via a web browser are stored is shipped together with Lycia. This document will only show how to create the simplest Birt report, upload it to the server and view it. For more details about LyciaBI see "LyciaBI Getting Started" guide.

### Creating a report

To create a report you will need LyciaStudio, a running LyciaBI server, and a database connection.

1. Launch LyciaStudio.
2. Go to **Window -> Open Perspective -> Others -> LyciaBI**.
3. On the main toolbar click New LyciaBI Project  button. Give the project a name and click **Finish**.
4. On the main toolbar click New Document  button and select **Birt** option.
5. Type the name of the file and click **Finish**. A Birt report will be created.
6. Go to **Window -> Open Perspective -> Others -> Report Design**. This perspective is used to edit Birt reports. Here is what it looks like:





## Defining the report data

We need to create a Data Source - the reference to the database which will be used. It can also be directory anywhere on the network. Then you will need a Data Set which can be the text of database query getting the data for the report or just a csv file, if the data source is a directory.

First of all create a csv file with the following contents:

```
Product,Average  
char,double  
product1,85  
product2,112  
product3,70  
product4,78  
product5,92
```

1. Go to the Data Explorer view. Right-click **Data Sources** element and select the option to create a new data source.
2. In the next dialog select **Flat File Data Source** from the list and click **Next**



3. In the next dialog select the directory where your csv files are located.
4. Check the second option. The second line of the file indicates the column data types, thus is this option is not checked the values will be misinterpreted.



**Select the folder that contains the flat files.**

Select folder:

Select charset:

Select flatfile style:

☒ Use first line as column name indicator.

☒ Use second line as data type indicator.

5. Click **Finish**.
6. Now in the Data Explorer right-click the **Data Sets** element and select New Data Set.
7. Click **Next**. From the list at the top select the csv file and move all the columns from the left to the right.

**Select the file and the columns for the data set.**

Select file:  File filter:

Product  
Average

➔

Name	Original Name	Type
Product	Product	String
Average	Average	String

Up  
Delete  
Down

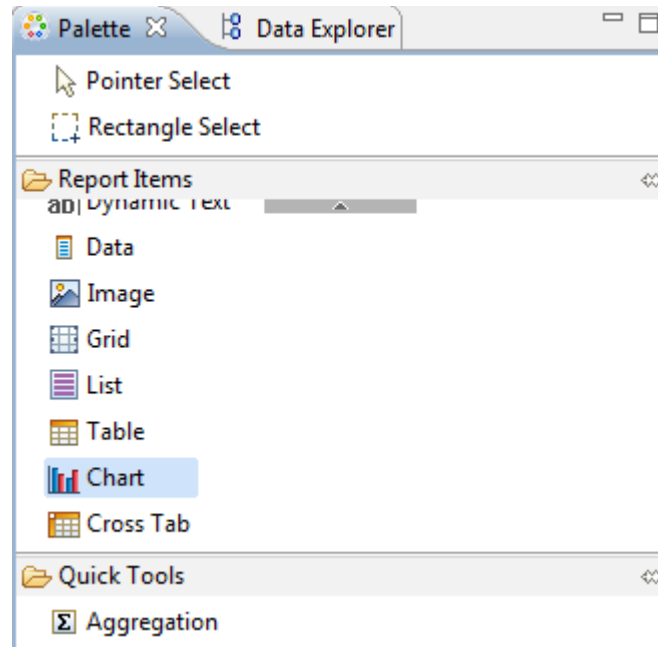
OK Cancel

8. Click **Finish** and then **OK**. The Data set was created.

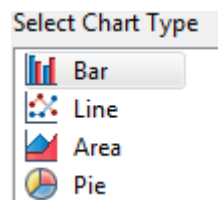
## Populating the report

Now we will create the simplest report with a chart:

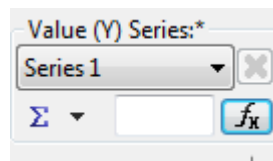
1. Go to Palette view of the Report Design perspective.
2. Select **Chart** and click in the editor area.



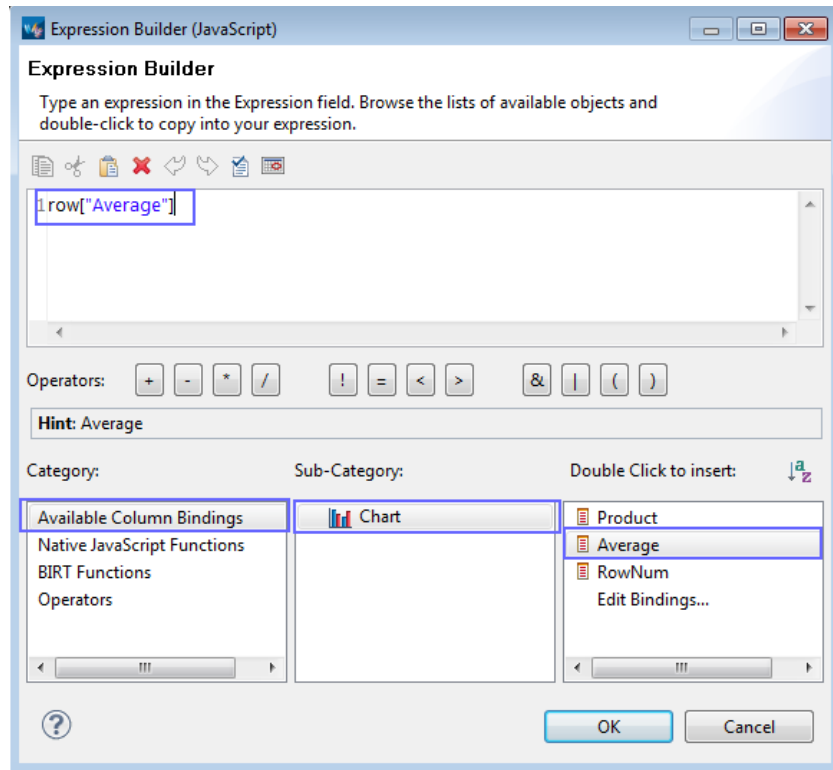
3. In the New Chart dialog select Bar chart and click **Next**.



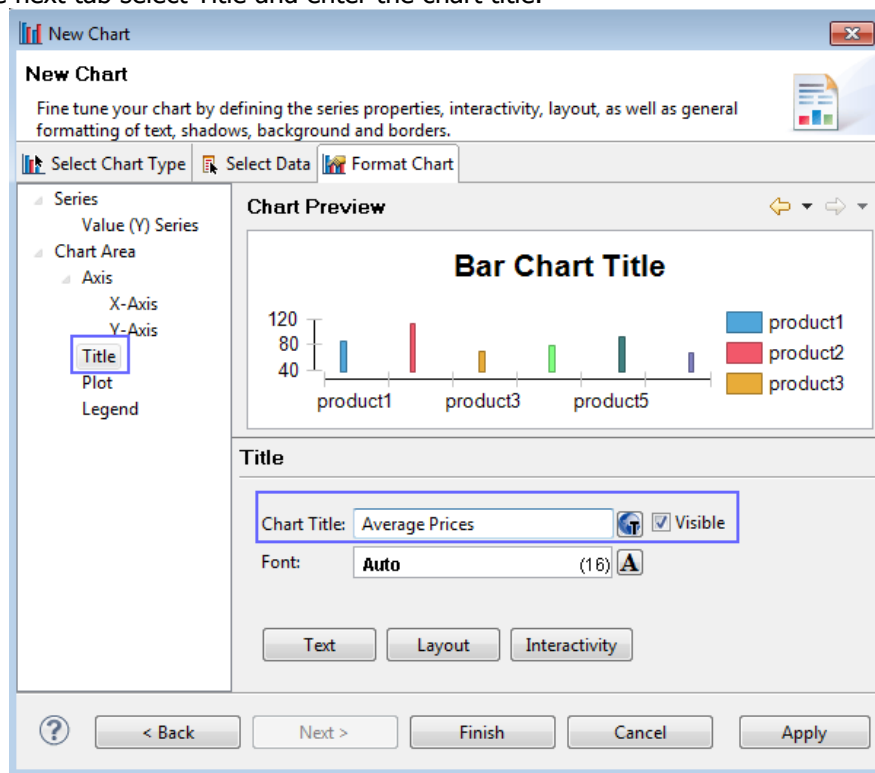
4. In the next tab click  $f_x$  button in the Value (Y) Series section:



5. Select the Average column and click **OK**.



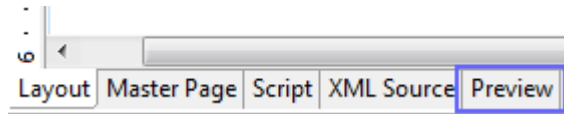
6. For Category (X) Series select the Product column in the same manner.
7. For the optional Grouping category also select the Grouping column and click **Next**.
8. In the next tab select Title and enter the chart title.





9. Then click **Finish**.

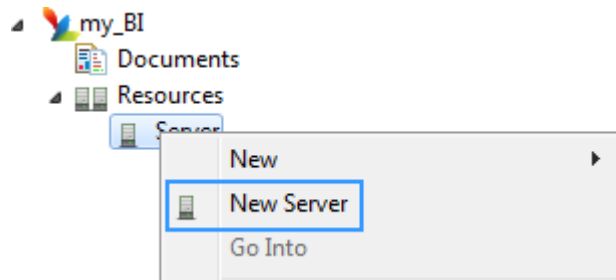
Now you can click **Preview** tab in the editor to see the results.



## Deploying and viewing the report

Now it is time to deploy the report to LyciaBI server. First you need to set LyciaBI server:

1. In LyciaStudio go to LyciaBI perspective.
2. In the BI project unfold the **Resources** section, then right-click the **Servers** and select **New Server** from the context menu:



3. Set LyciaBI server URL (e.g. <http://localhost:8080/LyciaBI>, if you installed the server on your local machine), the login (lbiadmin by default) and password (lbiadmin by default). Check the Active option if you want this sever to be used automatically for deployment. Click **Test** to test the connection and **Finish** to save the server configuration.



The 'New Server Wizard' dialog box is shown. It has a title bar with a question mark icon and standard window controls. The main title is 'New Server Wizard' with a subtitle 'This wizard lets you define a new server'. The form contains the following fields and controls:

- Server name: LyciaBI
- Url: http://localhost:9090/LyciaBI
- User: lbiadmin
- Password: (masked with dots)
- Active: ☒ (with a dotted box next to it)
- Test button
- Finish button
- Cancel button

Now you can deploy the document to the server:

1. In LyciaBI perspective right-click the report and select **Deploy** from the context menu.
2. In the new document dialog enter the document name and label, in the panel on the right select the folder on the server to which you want to deploy it.
3. Select the engine and the state and then click **Finish**.

The 'New Document' dialog box is shown. It has a title bar with a question mark icon and standard window controls. The form contains the following fields and controls:

- Label: my\_report
- Name: my\_report
- Description: (empty)
- Type: REPORT
- Engines: BirtEngine (dropdown)
- Dataset: (empty dropdown)
- Datasource: (empty dropdown)
- State: REL (dropdown)
- Refresh Seconds: 0 (spinners)
- Functionalities panel: Developer (selected)
- Finish button
- Cancel button

Now you can view the report on the server either logging in and navigating to it (using <http://localhost:9090/LyciaBI> link) or by using the following link in your browser or in the web browser of your 4GL program:



[http://localhost:8080/LyciaBI/servlet/AdapterHTTP?PAGE=LoginPage&NEW\\_SESSION=TRUE&OBJECT\\_LABEL=my\\_report](http://localhost:8080/LyciaBI/servlet/AdapterHTTP?PAGE=LoginPage&NEW_SESSION=TRUE&OBJECT_LABEL=my_report)

If your server has a different address, you should change it accordingly. "My\_report" is the report label created during deployment. If you entered a different label, change it accordingly.

The first time you use the ling - you will be asked to login. By default the login it "lbiadmin" the password is "lbiadmin". They can be changed by your LyciaBI administrator. For more details see "LyciaBI Getting Started" guide.

Your report will look like this:

