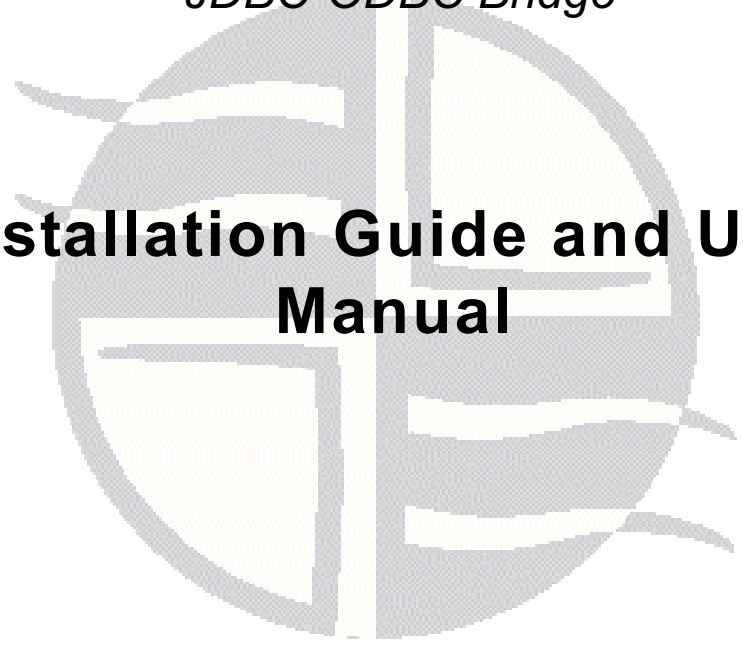
Easysoft Data Access

JDBC-ODBC Bridge



Installation Guide and User Manual

Version 17.

This manual documents version 1.x of the Easysoft JDBC-ODBC Bridge.

Publisher: Easysoft Limited

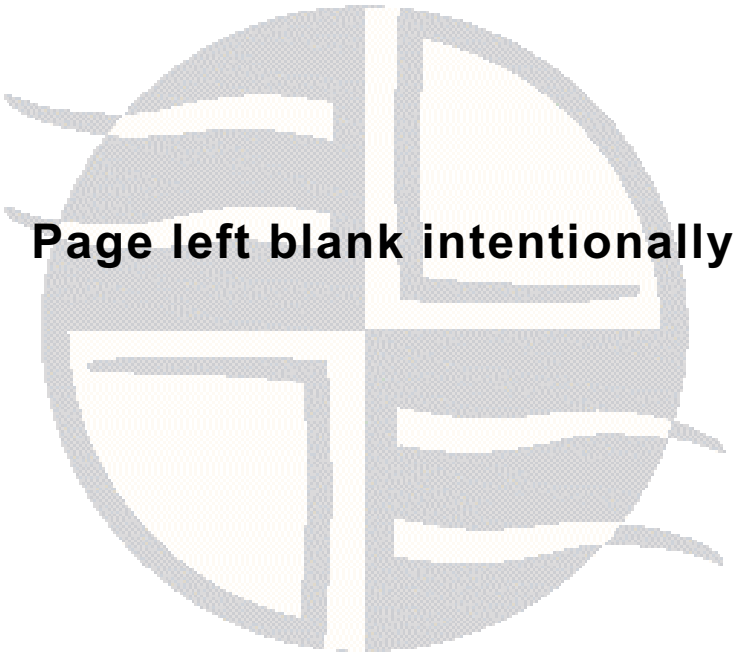
Thorp Arch Grange

Thorp Arch

Wetherby

LS23 7BA

United Kingdom



Page left blank intentionally

PREFACE

About this manual

This manual is intended to cover the full range of requirements for anyone wishing to install, use, or configure the Easysoft JDBC-ODBC Bridge.

Please note that this is not a JDBC programming manual.

Chapter Guide

Intended Audience

The sections written for the Microsoft Windows platform require some familiarity with the use of buttons, menus, icons and text boxes. If you have any experience of Apple Macintosh computers, Microsoft Windows or the X Window System, you will have no difficulty with these sections.

The Unix sections require that you are experienced at using a Unix shell, and can perform basic functions like editing a file. More complex activities are detailed more clearly.

Displaying the Manual

This manual is available in the following formats:

- Portable Document Format (PDF), which can be displayed and printed using the Acrobat Reader, available free from Adobe at **http://www.adobe.com**.
- HTML (the format Easysoft recommend for viewing onscreen).

Notational Conventions

Across the range of Easysoft manuals you will encounter passages that are emphasized with a box and a label.

A *note box* provides additional information that may further your understanding of a particular procedure or piece of information relating to a particular section of this manual:

NB	Note boxes often highlight information that you may need to be aware of when using a particular feature.

A *reference box* refers to resources external to the manual, such as a useful website or suggested reading:

REF	For more manuals that use this convention, see the rest of the Easysoft documentation.

A *platform note* provides platform-specific information for a particular procedure step:

Linux	In Linux you must log on as the `root` user in order to make many important changes.

A *caution box* is used to provide important information that you should check and understand, prior to starting a particular procedure or reading a particular section of this manual:

Caution!	Be sure to pay attention to these paragraphs because Caution boxes are important!

Information has also been grouped within some chapters into two broad classes of operating system, Windows and Unix, for which side tabs are used to help you turn to the section relevant to you.

Typographical Conventions

To avoid ambiguity, typographic effects have been applied to certain types of reference:

- User interface components such as icon names, menu names, buttons and selections are presented in bold, for example:

  Click **Next** to continue.

  Where there is a chain of submenus, the following convention is used:

  Choose **Start > Programs > Command Prompt**.

- Commands to be typed are presented using a `monotype` font, for example:

  At the command prompt type `admin`.

- Keyboard Commands

  It is assumed that all typed commands will be committed by pressing the *<Enter>* key, and as such this will not normally be indicated in this manual. Other key presses are italicized and enclosed by angle brackets, for example:

  Press *<F1>* for help.

- File listings and system names (such as file names, directories and database fields) are presented using the `monotype` plain text style.

Contents

Trademarks

Throughout this manual, *Windows* refers generically to Microsoft Windows 7, 8, 2012 R2, 10, 2016, 2019 or 2022, which are trademarks of the Microsoft Corporation. The X Window system is specifically excluded from this and is referred to as *The X Window System* or just *X*.

Note also that although the name UNIX is a registered trademark of The Open Group, the term has come to encompass a whole range of UNIX-like operating systems, including the free, public Linux and even the proprietary Solaris. Easysoft use Unix (note the case) as a general term covering the wide range of Open and proprietary operating systems commonly understood to be Unix ‘flavors’.

Easysoft and Easysoft Data Access are trademarks of Easysoft Limited.

1 INTRODUCTION

Introducing the Easysoft JDBC-ODBC Bridge

The Easysoft JDBC-ODBC Bridge (JOB) is a Type 3 JDBC Driver that enables 100% pure Java programs to make calls across the network from any Java-enabled platform to an ODBC data source running on Windows or Unix.

The interface at the client end is JDBC, allowing the Easysoft JDBC-ODBC Bridge to be incorporated into ready-written Java database-aware programs.

The Easysoft JDBC-ODBC Bridge implements all the functionality required for working with Java 2, Enterprise Edition (J2EE) compatible products and is "Certified for J2EE" under the certification program managed by KeyLabs, Inc. on behalf of Sun Microsystems, Inc. (see **JDBC Certification on page 120**).

Chapter Guide

Why ODBC?

ODBC is an API (*Application Programming Interface*) definition, compliant with ANSI SQL and X/Open's SQL Call Level Interface (CLI).

Applications can be written to this API standard without considering the intricacies of the various database engines, as the ODBC *driver* takes care of all the database-specific code, if necessary mapping the structure of the underlying system to a relational framework.

ODBC permits the DBMS-specific parts of the program to be separated from the part that fulfils the business requirement.

The result is that the completed application can be attached to any Database Management System (DBMS) that has a corresponding driver.

REF

For the definitive SQL CLI document consult the **Open Group CAE Specification C451, ISBN 1-85912-081-4** at **http://www.opengroup.org/pubs/catalog/c451.htm**.

The **Microsoft ODBC 3.0 Programmer's Reference** explains ODBC usage in some detail.

Why JDBC?

Given that you are using Java and you need access to a database, JDBC is really your only option. JDBC is the standard database Application Program Interface (API) for Java, supported by Sun and many large database vendors.

JDBC is modelled on ODBC but in addition provides an object-oriented model for accessing databases, permitting use of Java methods as well as SQL for querying and updating data.

The JDBC standard means that applications can be written without considering what driver will be used in the final deployment, and gives system managers the freedom to change database engines without requiring a change in program logic.

With the Easysoft JDBC-ODBC Bridge, you can connect your Java programs to any ODBC-conformant database without waiting to see whether the relevant JDBC drivers become available.

JDBC driver types

The Sun website at **http://java.sun.com/products/jdbc/datasheet.html** identifies four JDBC Driver types to help customers select the correct driver (these explanations use Sun's own terminology and diagrams):

TYPE 1: *A JDBC-ODBC Bridge (not the Easysoft JDBC-ODBC Bridge)*

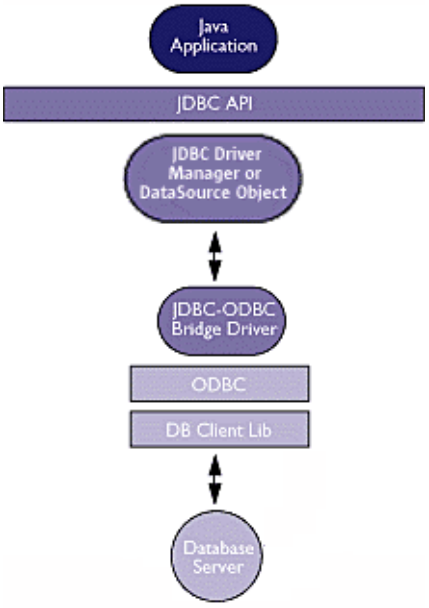


Figure 1: Type 1 JDBC Driver architecture

Type 1 JDBC drivers are always written in platform-specific code and thus limit the potential of the application for cross-platform deployment.

This simple JDBC driver is provided by Sun with the Java Development Kit and implements the JDBC API by making ODBC calls, allowing connection to any ODBC data source available on the local machine.

TYPE 2: *Native-API Partly Java technology-enabled driver*

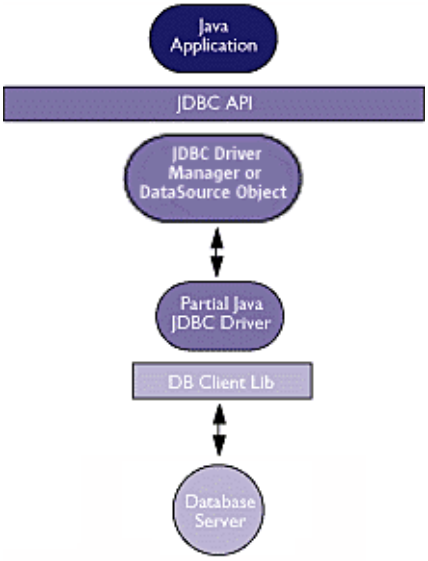


Figure 2: Type 2 JDBC Driver architecture

Type 2 JDBC drivers talk directly to the API of the DBMS, rather than any mapping layer such as ODBC, and can be accessed from Java.

This implies that the driver either does not provide the complete JDBC API (but provides enough to drive the native database API) or is not written completely in Java (thus losing out in cross-platform functionality).

Technically, this type of driver is the most efficient user of machine resources, but this advantage is far outweighed by the need to write a different JDBC driver for each DBMS and for each platform.

TYPE 3: *Net-protocol fully Java technology-enabled driver*

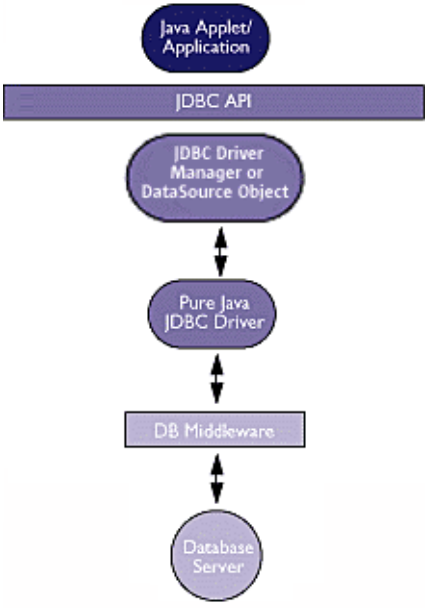


Figure 3: Type 3 JDBC Driver architecture

Type 3 JDBC drivers are client-server pairs which use a database-independent network protocol to bridge the network.

The client is written in Java, enabling cross-platform deployment of the application.

Sun cites this as generally "the most flexible JDBC API alternative", as the location of the database is not confined to the same machine as the Java program, and the client can be written in pure Java.

Platform and database dependent code is moved to the server running the ODBC driver, allowing these drivers to run over the internet.

TYPE 4: *Native-protocol fully Java technology-enabled driver*

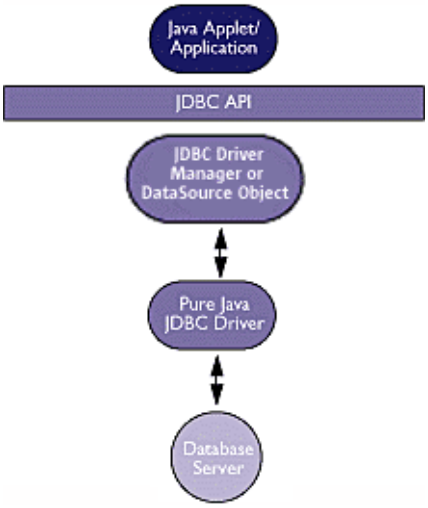


Figure 4: Type 4 JDBC Driver architecture

Type 4 JDBC drivers are written entirely in Java and transmit database-specific messages direct to the DBMS, which cannot be changed without modifying or replacing the driver.

Architecture

At the server end, although the Easysoft JDBC-ODBC Bridge has a Type 3 architecture, it produces ODBC calls which can be picked up by any ODBC driver and used to drive the database, rather than DBMS-specific ('native') calls.

The JOB Server is dependent only on the platform and the database engine can be replaced without users, application programmers or the Easysoft JDBC-ODBC Bridge knowing and as long as the chosen DBMS supports ODBC the code can remain unchanged.

At the application end, the driver is pure Java and therefore completely portable.

The following table summarizes the advantages of the Easysoft JDBC-ODBC Bridge:

	Type 1	Type 2	Type 3	Type 4	Easysoft
Can connect to ODBC data sources	Yes	No	Not always	No	Yes
Application has "write once/run anywhere" platform independence	No	No	Yes	Yes	Yes
Can connect across internet	No	No	Subject to firewall	Subject to firewall	Subject to firewall
Can replace database engine without changing JDBC driver	Yes	No	Not normally	No	Yes

Figure 5: Advantages of the Easysoft JDBC-ODBC Bridge

NB	In practice, the SQL syntax and semantics vary slightly between database vendors. By carefully partitioning your code however, the database-specific functions can be stored on the server, so that the deployed programs do not need to be modified.

Note that if you want to utilize the support of the Easysoft JDBC-ODBC Bridge for JDBC 2 functionality within your applet then you must amend your HTML to work with the Java Plug-in, which users may install on their client in order to override their web browser's default virtual machine and take full advantage of the latest capabilities and features of the Java 2 SDK.

The Java Plug-in is available as a component of Sun's Java 2 Runtime Environment and you can obtain the Java Plug-in HTML Converter and supporting documentation from **http://java.sun.com/products/plugin**.

In order to develop and test for JDBC 2 functionality you need to:

- use the Java Plug-in HTML Converter to convert the applet tags on your HTML web pages into a format that can be used with the Java Plug-in.
- obtain the Java Plug-in from Sun.

Driver Managers

The barest ODBC system would include an ODBC-conformant driver accessing some data, and an ODBC-conformant application, linked to the driver library.

If commercial applications were distributed in this way, users would need to re-link their applications to their chosen driver whenever they wanted to access a different data source.

Instead, the application program is linked to a *driver manager*, which is responsible for loading the required ODBC driver at runtime and then initializing the driver with a stored set of attributes:

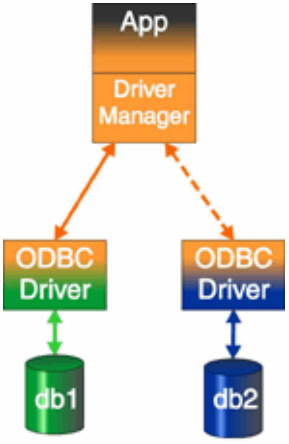


Figure 6: The Driver Manager as a dynamic linker

To support user-transparent replacement of the back end the driver manager provides a mapping from an abstract *Data Source Name* to a driver along with a default set of *attribute=value* pairs which are passed to the driver on connect.

Although it is possible to use ODBC without a driver manager by linking directly with the ODBC driver at build time, this option needlessly removes some beneficial functionality and the method of linking programs with the driver manager shared object (Unix) or DLL (Windows) provides three key results:

- Applications written by developers to satisfy a business requirement can be 'plugged in' to any compatible database.
- Administrators can connect a variety of applications (such as query tools) to their databases to browse and investigate data.
- *Data access middleware* can be inserted between the ODBC driver manager and driver to add strategic functionality such as joining heterogeneous databases into one data source or bridging a network.

On Unix, the unixODBC driver manager is included in the Easysoft JDBC-ODBC Bridge distribution.

Windows users should have the Windows ODBC driver manager included with the ODBC Driver for their database application.

REF

For the Microsoft Windows driver manager, go to **http://support.microsoft.com** and search for MDAC, *Microsoft Data Access Components*.

See **http://www.unixodbc.org** for the full set of unixODBC data access components.

Components

The Easysoft JDBC-ODBC Bridge is installed on the server (i.e. the host where the ODBC data source is installed) and comprises of:

- The Easysoft JDBC-ODBC Bridge Server

  This is a native-code program that services incoming requests for data and executes code in the ODBC driver.

- The Easysoft JDBC-ODBC Bridge Driver class library

  This allows your JDBC application to transmit these requests across the network.

- The Web Administrator

This allows the client computer to download the Easysoft JDBC-ODBC Bridge classes at runtime and configure the JOB Server.

- The unixODBC Driver Manager (Unix distributions only)

Deployment

SCENARIO 1: JAVA APPLICATION

Scenario 1 shows a Java application accessing two remote ODBC data sources using the Easysoft JDBC-ODBC Bridge.

The Easysoft JDBC-ODBC Bridge Client is 100% Java and therefore the only requirements of the client device are a Java Virtual Machine and TCP/IP connectivity.

One database is hosted on the same platform as the Easysoft JDBC-ODBC Bridge Server and another on a separate device.

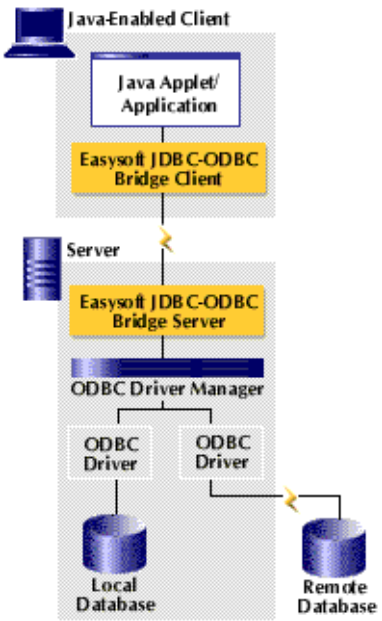


Figure 7: A Java application using JOB

SCENARIO 2: ZERO INSTALLATION JAVA APPLET

Scenario 2 illustrates the loading of a Java applet from a Web Server.

No client installation is required.

Applet rules require that the applet and driver are served from the same machine on which the Easysoft JDBC-ODBC Bridge Server is running.

The target database can be local to the Easysoft JDBC-ODBC Bridge Server or on a remote database server.

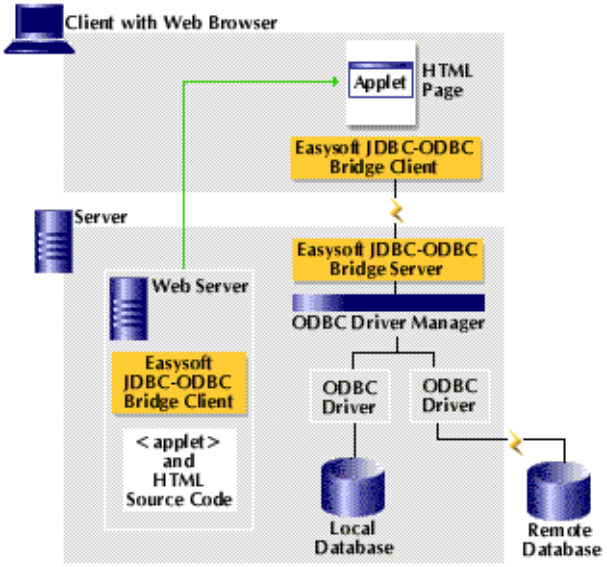


Figure 8: A Zero Installation Java applet using JOB

SCENARIO 3: ZERO INSTALLATION JAVA APPLET FROM A DIFFERENT MACHINE

Scenario 3 is a variation on scenario 2 and demonstrates how the HTML page containing the applet tag can be served from a different machine to the applet code and Easysoft JDBC-ODBC Bridge Client.

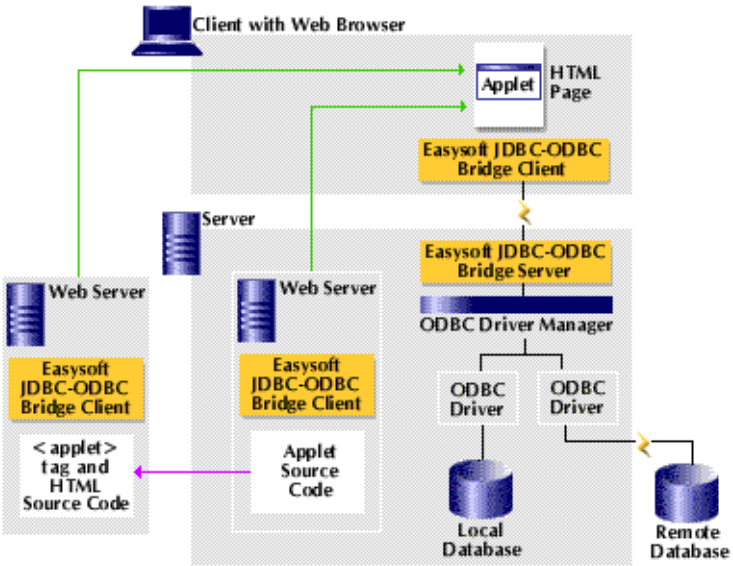


Figure 9: A Zero Installation Java applet using JOB with multiple servers

SCENARIO 4: HETEROGENEOUS DATABASE ACCESS

Scenario 4 introduces the Easysoft ODBC Join Engine as a device to simplify and optimise multi-database queries.

To the Java application on the client platform the three existing ODBC data sources on the Easysoft JDBC-ODBC Bridge Server appear as a single heterogeneous data source.

Queries are passed to the Easysoft ODBC Join Engine and resolved on the Easysoft JDBC-ODBC Bridge Server platform, reducing both execution times and network traffic.

The Easysoft ODBC Join Engine can be used in this way with both Java applications and Java applets.

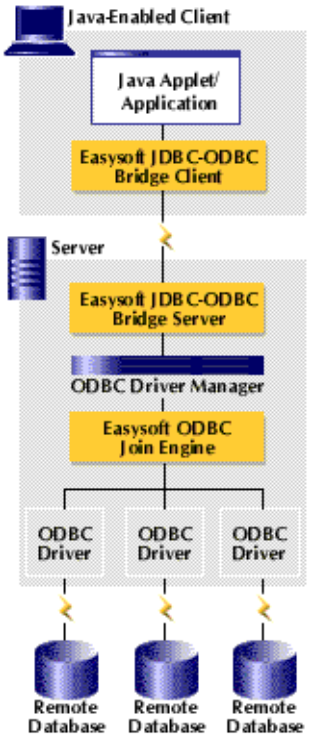


Figure 10: Heterogeneous database access using the ODBC Join Engine

INSTALLATION

2

Installing the Easysoft JDBC-ODBC Bridge

This section explains how to install, license and remove the Easysoft JDBC-ODBC Bridge on supported Windows and Unix platforms.

The Windows installation can be carried out by anyone with local administrator privileges for the target machine.

The Unix installation assumes you are, or have available for consultation, a system administrator.

Chapter Guide

Obtaining the Easysoft JDBC-ODBC Bridge

There are three ways to obtain the Easysoft JDBC-ODBC Bridge:

- The Easysoft web site is available 24 hours a day at **http://www.easysoft.com** for downloads of definitive releases and documentation.

  Select **Download** from the Easysoft JDBC-ODBC Bridge section of the website and then choose the platform release that you require.

  First time visitors must complete the new user form and click **Register**. Note that your personal Internet options may require you to login and click **Continue** if you have previously registered.

- The Easysoft FTP server is available 24 hours a day at **ftp://ftp.easysoft.com**, containing free patches, upgrades, documentation and beta releases of Easysoft products, as well as definitive releases.

  Change to the `pub/jdbc-odbc-bridge` directory and then choose the platform release that you require.

What to install

The Easysoft JDBC-ODBC Bridge consists of the following product components:

- the JOB Driver (also known as the JOB Client)
- the JOB Server
- the unixODBC Driver Manager (on Unix platforms only)

The JOB Driver is 100% Java and can be installed on any machine supporting Java by copying it to a suitable place in the filestore and setting the appropriate value in the `CLASSPATH` environment variable.

The JOB Server is platform-specific, so be sure to download the correct distribution.

NB

You only need to perform the download and install procedure on the server host (i.e. the machine where the ODBC data source is located).

For the client side, copy the file `EJOB.jar` from the server host into the `CLASSPATH` on the client host. You can normally find this by pointing your browser to the URL http://*serverhost*:8031/`EJOB.jar` (on Windows you specify `CLASSPATH` as an environment variable via the Control Panel). If you download `EJOB.jar` from the Easysoft Web site, ensure that you save it with the `.jar` file extension and that it is NOT saved as `EJOB.html`.

In the distribution are the driver classes, the JOB Server (including the Web Administrator), shared components needed to license and run the JOB Server, and documentation files.

The Unix distribution also includes the unixODBC Driver Manager.

Windows users should have already received a driver manager with their ODBC Driver.

REF	If you need the Microsoft Windows driver manager, go to **http://support.microsoft.com** and search for MDAC, *Microsoft Data Access Components*.. See **http://www.unixodbc.org** for the full set of unixODBC data access components.

The name of the install file varies from platform to platform, but is of the form:

- `jdbc-odbc-bridge-x_y_z-platform.exe` (Windows)

– OR –

- `jdbc-odbc-bridge-x.y.z-platform.tar.gz` (Unix)

where "*x*" is the major version number, "*y*" is the minor version number and "*z*" is the build index, which is incremented when minor changes are made.

"*platform*" will vary depending on the operating system distribution you require and you may come across files of the form:

- `jdbc-odbc-bridge-x.y.z-platform-variation.tar`

within specific Unix platforms, where "*platform-variation*" refers to alternative versions available for a single platform.

NB	Select the highest release available for your platform within your licensed major version number (installing software of a different major version number requires a new Easysoft license).

Unix filenames may also be suffixed with `.gz` for a "`gzip`ped" archive, or `.Z` for a "compressed" archive.

NB If you download a Unix file using Windows, the browser may corrupt the filename. For example, if you download a `.gz` file and Windows corrupts the filename, it may not be obvious that the file is `gzip`ped. Use `file` *filename* to find out the file type of the downloaded file.

You can now download a file and begin the installation process.

Caution!

On Unix, as long as you stop any software either from Easysoft or using Easysoft drivers, it is safe to reinstall or upgrade the Easysoft JDBC-ODBC Bridge without uninstalling.

If you do uninstall, you should first back up any configuration data that you still need, as uninstalling some Easysoft products will result in this information being deleted (license details remain in place).

Refer to the section relevant to your platform to continue:

- **“Installing on Windows” on page 36**
- **“Uninstalling on Windows” on page 47**
- **“Installing on Unix” on page 48**
- **“Uninstalling on Unix” on page 57**

Installing on Windows

- Execute the file distribution that you downloaded in “Obtaining the Easysoft JDBC-ODBC Bridge” on **page 32**.

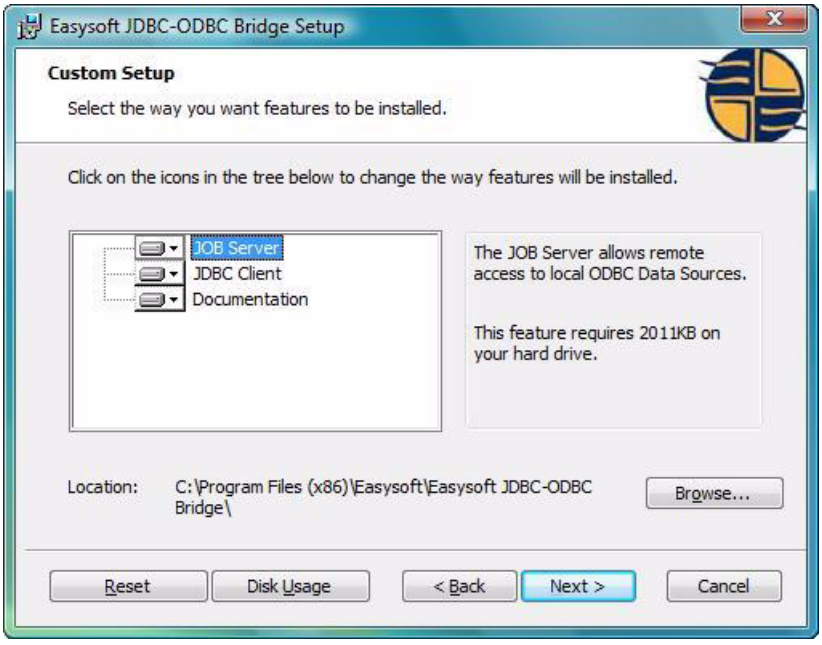
  Follow the on screen instructions.

SETUP TYPES

Figure 11: The Choose Setup Type dialog box

When prompted to choose a Setup type, do one of the following:

- Select **Typical** or **Complete** to install the Easysoft JDBC-ODBC Bridge Client and Server, documentation, tutorials and examples (all components).
- Select **Custom** if you want to choose which components to install or choose where the the software is installed.



The Custom Setup dialog box has two panes. The left pane shows a tree view of the features that you can install and you can expand a feature to view its subfeatures. The right pane displays feature descriptions. When you click a feature or subfeature, the Feature description pane displays a description of the selection and its disk space requirements.

To add or remove a feature, click the arrow next to the feature name, and then choose one of the following options from the drop-down list:

- **Will be installed on local hard drive** Install the selected feature in the location shown under **Location**.
- **Entire feature will be installed on local hard drive** Install the selected feature and any subfeatures.
- **Entire feature will be unavailable** Do not install the selected feature and any subfeatures.

The default installation directory is *drive*:\Program Files\Easysoft\Easysoft JDBC-ODBC Bridge. To change the installation directory, click **Browse**.

To restore the default list of features to install, click **Reset**. To check that you have enough disk space to install the features you want, click **Disk Usage**.

EASYSOFT JDBC-ODBC BRIDGE SERVER CONFIGURATION SETTINGS

If you chose to install the Easysoft JDBC-ODBC Bridge Server, the **Initial Server Configuration** dialog box is displayed:

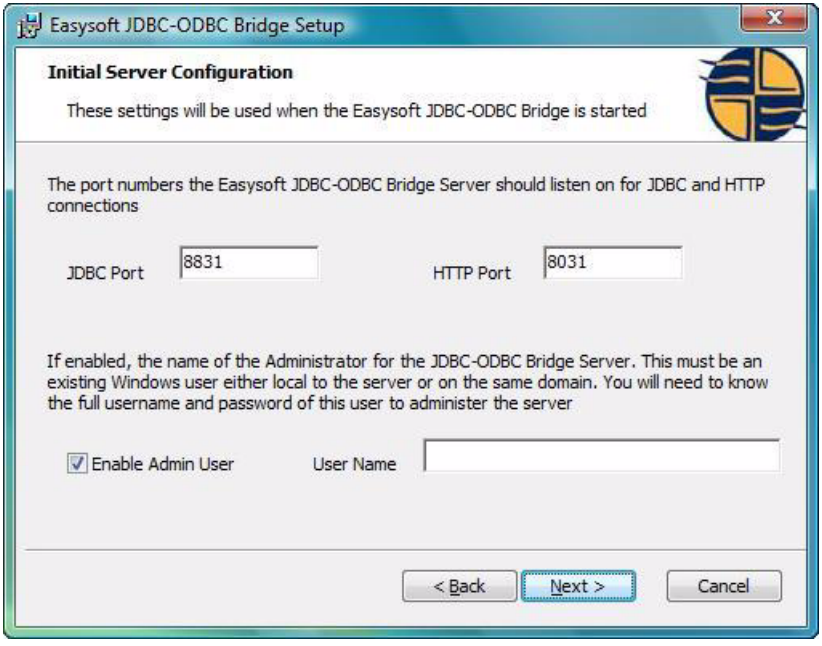


Figure 12: The Initial Server Configuration dialog box

JDBC Port defines the port where the Easysoft JDBC-ODBC Bridge Server will listen for Easysoft JDBC-ODBC Bridge Client connections.

NB	Changing the value of **JDBC Port** will mean changing the JDBC URL used in your application.

HTTP Port defines the port on which the Easysoft JDBC-ODBC Bridge Server Web Administrator will listen for HTTP requests. Accept the default values unless you have a port conflict.

In the **User Name** box, enter the login name of an existing Windows user account as the Server Administrator user name.

You also need to know the password for this user, because it will be required when you run the Web Administrator (see “The Web Administrator” on **page 73**).

To grant group access to the Web Administrator, you may wish to create a specific "`Administrator`" user.

To allow anyone to have access to the Web Administrator, leave the **Username** field blank or click to clear **Enable Admin User** to remove the requirement to log in.

Caution! You should consider carefully whether you wish to allow *anyone* on your network to have access to the Web Administrator in order to change Easysoft JDBC-ODBC Bridge settings.

Updating Files That are in Use

To avoid rebooting your computer, the Easysoft JDBC-ODBC Bridge installer prompts you when files that it needs to update are in use by another application or service. This frees the locked files and allows the installation to complete without a system restart.

On Windows Vista, the Easysoft JDBC-ODBC Bridge installer uses the Restart Manager to locate the applications that are using files that need updating. These applications are displayed in the Files in Use dialog box. To avoid a system restart, choose **Automatically close applications and attempt to restart them after setup is complete**. The Easysoft JDBC-ODBC Bridge installer then uses the Restart Manager to try to stop and restart each application or service in the list. If possible, the Restart Manager restores applications to the same state and with the same data that they were in before it shut them down.

On earlier versions of Windows, when the Files in Use dialog is displayed, manually shut down each application in the list and then click **Retry** to avoid a system restart.

Licensing the Easysoft JDBC-ODBC Bridge Server

Only the Easysoft JDBC-ODBC Bridge Server needs to be licensed.

By default, the install program starts the Easysoft License Manager (documented in the **Licensing Guide**), because you cannot use the Easysoft JDBC-ODBC Bridge Server until a license is obtained.

The following types of license are available:

- A *free time-limited trial license*, which gives you free and unrestricted use of the product for a limited period (usually 14 days).
- A *full license* if you have purchased the product. On purchasing the product you are given an authorization code, which you use to obtain a license.

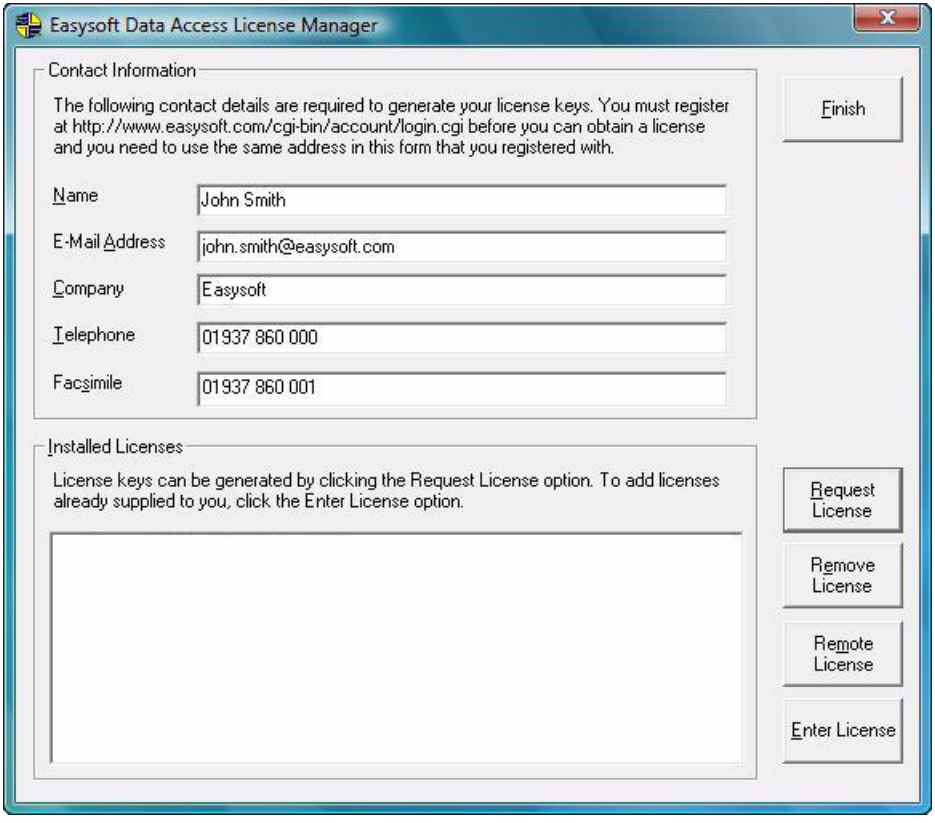


Figure 13: The License Manager window

1. Enter your contact details.

   You MUST enter the **Name**, **E-Mail Address** and **Company** fields. The e-mail address that you enter here must be the same as the one that you registered with.

   The **Telephone** and **Facsimile** fields are important if you require Easysoft to contact you by those methods.

2. Click **Request License**.

   You are asked for a license type.

3. For a trial license click **Time Limited Trial** and then click **Next**.

   The License Manager asks what software you are licensing.

   Select your required version of the Easysoft JDBC-ODBC Bridge from the drop-down list and then click **Next**.

   – OR –

   If you have obtained an authorization code for a purchased license, select **Non-expiring License** and then click **Next**.

   The License Manager requests your authorization code.

   Enter the authorization code and then click **Next**.

   The License Manager displays a summary of the information you entered and allows you to choose how to apply for your license.

4. Choose **On-line Request** if your machine is connected to the internet and can make make outgoing connections on port 8884.

The License Manager then sends a request to the Easysoft license server to activate your license key automatically. This is the quickest method and results in your details being entered immediately into our support database.

> **NB** Only your license request identifier and contact details as they are displayed in the main License Manager screen are sent to Easysoft.

The remaining three options (**Email Request**, **Print Request** and **View Request**) are all ways to obtain a license if your machine is offline (i.e. does not have a connection to the internet).

Each of these methods involves providing Easysoft with information including your machine number (a number unique to your machine) and then waiting to receive your license key.

Instead of emailing, faxing or telephoning your details to Easysoft, you can enter them directly at the Easysoft web site and your license key will be emailed to you automatically.

To use this method, click **View Request**, and then visit:

- **http://www.easysoft.com/support/licensing/trial_license.html**
  (trial licenses)
- **http://www.easysoft.com/support/licensing/full_license.html**
  (purchased licenses)

In the Licensing page, enter your machine number (and authorization code for purchased license), click **Submit** and your license key will be emailed to you.

> **NB** You can copy your machine number from the **View Request** dialog box using `CTRL-C` and then paste it into the License Generator by using `CTRL-V`.

When you receive the license key, you can activate it either by double-clicking the email attachment or by clicking **Enter License** on the License Manager main screen and pasting the license key into the dialog box.

A message tells you how many licenses have been added.

NB	If you use the **Email Request** option, the license key is emailed to the email address as displayed on the License Manager screen, not the `from:` address of your email.

For more information about the licensing procedure refer to the **Licensing Guide**.

NB	If you add a new license, the Easysoft JDBC-ODBC Bridge Server service needs to be restarted in order to access the new details. During Setup, the Easysoft JDBC-ODBC Bridge Server service is restarted automatically after a license has been added. If you add a license outside of Setup, you need to restart the service manually. For more information about restarting the Easysoft JDBC-ODBC Bridge Server service, see “Setting up the JOB Server on Windows” on **page 60**.

5. Click **Finish** to quit the License Manager.

.

	If you connect to the Internet via a proxy server you should disable the use of the proxy server for local addresses in your browser, as this will cause the connection to fail.
NB	Disable the proxy server via **Tools > Internet Options > Connections > LAN Settings > Bypass proxy server for local addresses** under Microsoft Internet Explorer and via **Edit > Preferences > Advanced > Proxies > Direct connection to the Internet** under Netscape Navigator.

CHANGING OR REPAIRING THE EASYSOFT JDBC-ODBC BRIDGE INSTALLATION

The Easysoft JDBC-ODBC Bridge installer lets you add or remove Easysoft JDBC-ODBC Bridge components. You can add or remove the Easysoft JDBC-ODBC Bridge Server, Easysoft JDBC-ODBC Bridge Client or Easysoft JDBC-ODBC Bridge documentation.

The installer can also repair a broken Easysoft JDBC-ODBC Bridge installation. For example, you can use the Easysoft JDBC-ODBC Bridge installer to restore missing Easysoft JDBC-ODBC Bridge files or registry keys.

1. Do one of the following:
   - In Windows Vista, in **Control Panel**, open **Programs and Features**.
   - In earlier versions of Windows, in **Control Panel** open **Add or Remove Programs**.

2. Do one of the following:

   - In Windows Vista, right-click **Easysoft JDBC-ODBC Bridge**, and then click **Change** or **Repair**.
   - In previous of Windows, select **Easysoft JDBC-ODBC Bridge** and click **Change/Remove**.

Uninstalling on Windows

This section explains how to remove the Easysoft JDBC-ODBC Bridge from your system.

1. Do one of the following:

   - In Windows Vista, in **Control Panel**, open **Programs and Features**.
   - In earlier versions of Windows, in **Control Panel** open **Add or Remove Programs**.

2. Do one of the following:

   - In Windows Vista, double-click **Easysoft JDBC-ODBC Bridge**.
   - In earlier versions of Windows, select **Easysoft JDBC-ODBC Bridge** and click **Remove**.

   The uninstall process is complete.

   Any licenses you obtained for the Easysoft JDBC-ODBC Bridge and other Easysoft products are held in the Windows registry.

   When you uninstall, your licenses are not removed so you do not need to relicense the product if you reinstall or upgrade.

Installing on Unix

These instructions show how to install the Easysoft JDBC-ODBC Bridge on Unix platforms.

You should only install this software onto a machine that holds ODBC drivers that you want to access over the JDBC-ODBC Bridge from other machines.

If the ODBC data sources to which you would like to gain remote access from any platform resides on other platforms (such as Linux or Windows), please obtain and install the Easysoft JDBC-ODBC Bridge appropriate for those platforms.

If you wish to gain access to your ODBC Data sources via JDBC from applets, you can use the build in HTTP server to serve them.

For production use it is recommended that you take the Easysoft JDBC-ODBC Bridge Driver classes from `<install_path>easysoft/job/EJOB.jar` and make them available via your own Webserver.

If you are developing or deploying applications you can download the JDBC-ODBC Bridge Driver classes directly from the machine that runs the server. See the page:

`http://<your machine>:<EJOB HTTP PORT>/`

If you have ODBC drivers on this machine which you want to access over the JDBC-ODBC Bridge from other machines you should install the server. You must be `root` to install the server properly as it requires write access to `/usr/local` tree.

If you are installing the JOB Server to access ODBC drivers on this machine from other machines via the JOB, then Easysoft recommends that you install the unixODBC Driver Manager.

Please remove any old installation tree before reinstalling or installing a new version in the same tree. See the file `<install_path>/easysoft//job/doc/uninstall.txt` for instructions on how to do this.

REQUIREMENTS

The installation script has a minimal set of requirements:

- Bourne shell in `/bin/sh` (if your Bourne shell is not located there you may need to edit the first line of the install file).
- Various commonly used UNIX commands such as:

  `grep`, `awk`, `test`, `cut`, `ps`, `sed`, `cat`, `wc`, `uname`, `tr`, `find`, `echo`, `sum`, `head`

  If you are missing any of these commands they can generally be obtained from the Free Software Foundation (**http://www.fsf.org**).

Depending on the platform, you will need up to 5Mb of disk space free for the installed programs and up to 4Mb temporary space of the installation files themselves. If you install the unixODBC Driver Manager as well, these numbers increase by approximately 1.5Mb.

THE INSTALLATION PROCESS

Untar the tar distribution file. On some platforms this is gzipped or compressed. This should create a directory called "`jdbc-odbc-bridge-`*`p.q.r.s.os`*" (where *`p.q.r.s`* is the version and *os* is the operating system name) containing some further tar files, some `checksum` files and an installation script ("`install`").

NB	Make sure you read the license agreement (`Server-Client-License.txt`).

Installing the software shows your acceptance of the terms and conditions in the license agreement.

Throughout the installation you may be asked to supply the answer to some questions. In each case the default will be displayed in square brackets and you need only press <return> to take the default. If there are alternative responses these will be shown in round brackets; to pick one of these type them and press <return>.

Change into the created directory and run the install program with

```
./install
```

You will be prompted to read the license file appropriate for your installation and you must answer yes or no to accept the license. If you do not accept the license, the installation will be aborted.

Once the installation has checked you have the necessary shell tools it will autodetect the operating system and check any required dependencies (e.g. version of the C runtime library). Next the install checks the encapsulated `tar` package(s) by comparing them with a `checksum` file (do not worry if the `md5 checksum` is not found as some platforms do not come with `md5sum`). If this fails you will be given the opportunity to abort the installation.

You will next be prompted for an installation path. All files are installed in a subdirectory of your specified path called "`easysoft`" e.g. if you pick the default of `/usr/local`, the JOB will be installed in `/usr/local/easysoft` and below. Some files outside the installation path will be modified (but not without prompting you first).

If you choose an install path different from the default then the installation will try to symbolically link `/usr/local/easysoft` to the `easysoft` in your chosen path. This allows us to distribute binaries with built in dynamic linker run paths. If you are not `root` or the path `/usr/local/easysoft` already exists and is not a symbolic link this will fail and you may have to fix the link yourself or define `LD_LIBRARY_PATH`/`LD_RUN_PATH` to contain the path `<install_path>/easysoft/lib`.

You should set `LD_LIBARY_PATH`/`LD_RUN_PATH` in the `<install_path>/easysoft/job/startjob` script.

The JOB is now distributed as 3 or 4 packaged tar files. These tar files are for common elements, EJOB.jar JDBC Driver, and the server. The tar files are all untarred into subdirectories of `<install_path>/easysoft`. You can consult `<install_path>/easysoft/job/INVENTORY.txt` for a description of all the files installed.

If the distribution for your platform contains the unixODBC driver manager you will now be asked whether you would like to install it. Easysoft strongly recommend you use the unixODBC Driver Manager because:

- The installation is designed to work with unixODBC and can automatically add ODBC drivers and DSNs during the install.
- Most applications and interfaces that can use ODBC know about unixODBC. This means any new ODBC drivers or data sources you add with this installation will automatically become available to your applications and interfaces.
- unixODBC is currently maintained by Easysoft developer Nick Gorham. This means there is much greater experience with unixODBC within Easysoft and we will be able to provide better support for OOB running under unixODBC. It also means that if you find a problem in unixODBC it is much easier for us to facilitate a fix.

The installation will next untar the common tar file. Some files and shared objects are common to multiple Easysoft products. If files shared between Easysoft products are included in the JOB install the installation will check to see if those files already exist or are older. Only if these files do not exist or are older will the shared files be installed. You may see messages like:

```
versioned shared file <filename> does not exist -
installing
```

– OR –

```
versioned shared file <file> is old - replacing
```

The remainder of the installation installs the JOB server and requires the installer to be the `root` user. If you are not `root` the installation will terminate now.

The JOB Server requires a license in order to accept JDBC connections from the JOB driver.

The installation will create the `<install_path>/easysoft/license` directory if it does not exist and also a licenses template file in that directory.

You can obtain a trial license or register a full license automatically during the installation. When prompted to request an JOB Server license, enter '`y`' and the Easysoft License Client program will be started for you.

To obtain a license automatically you will need to be connected to the Internet and allow outgoing connections to `license.easysoft.com` on port 8884. If you are not connected to the Internet or do not allow outgoing connections on port 8884 then the License Client can create a license request file which you can mail, fax or telephone to Easysoft.

Once the License Client has started you are presented with a menu of options which allow you to:

```
[0] exit

[1] view existing license

[2] show product list

[n] obtain a license for the desired product
```

Obviously, if you have not got any other Easysoft products licensed then option `[1]` will not show any existing licenses. To obtain a license select one of the options from `[2]` onwards for the product you are installing, e.g. currently, option `[3]` for licencing the JDBC-ODBC Bridge. The License Client will then run a program that was installed for that product which generates a key which is used to identify the product and operating system (we need this key to license you).

NB	Some products in the list may not be licensable during the install, e.g. if you are installing the JOB and do not have the Easysoft ODBC-ODBC Bridge installed then you cannot obtain a license for the ODBC-ODBC Bridge (in this case the License Client will probably output a warning that the siteinfo command for the chosen product was not found).

Assuming you have picked a product which is installed (or in the process of being installed) then you need to supply:

1. Your full name
2. Your company name
3. An email contact address (you need to specify this if you email the license request to us).
4. Your telephone number (you need to specify this if you telephone the license request to us).
5. Your fax number (you need to specify this if you fax the license request to us).
6. A reference number. When applying for a trial license just hit <return> on this field as this field is used to enter a reference number we will supply you for full (paid) licenses.

You will then be asked for a method of obtaining the license where the choices are:

[1] Automatically by contacting the Easysoft License Daemon (this requires connection to the Internet and the ability to support an outgoing TCPIP connection to `license.easysoft.com` on port 8884).

[2] Write information to file so you can fax, telephone it (the license request is output to license_request.txt and you can telephone or fax this to Easysoft).

[3] Cancel request

If you choose to obtain the license automatically, the License Client will start a TCPIP connection to `license.easysoft.com` on port 8884 and send the details you entered at the prompts above and your site number. No other data is sent. The data sent is transmitted as plain text so if you do not want this information possibly intercepted by someone else on the net you should choose `[2]` and telephone or fax the request to us. The License daemon will return the license key, print it to the screen and make it available to the installation script in the file licenses.out.

If you choose option `[2]` the license request is written to the file `license_request.txt` and you should exit the License Client via option `[0]` and complete the installation. Once you have mailed, faxed or telephoned the license request to us we will return a license key which should add to the end of the file `<install_path>/easysoft/license/licenses`.

If during this process any warnings or errors are output please mail the output to **support@easysoft.com** and we will rectify the problem.

If you ran the License Client, the install script will detect whether any licenses were automatically obtained and ask you if they are to be installed in the `<install_path>easysoft/license/licenses` file. You should answer `y` to this or the JOB Server will not be licensed and will not accept connections from JOB clients. If you choose not to have the licenses added to the licenses file then you can do this manually later by copying the lines in licenses.out to `<install_path>/easysoft/license/licenses`..

NB	The shared objects installed are NOT stripped. You may strip them if you wish but this may affect any support Easysoft can supply.

STARTING THE JOB SERVER

Start up the server by:

```
# cd <INSTALL PATH>/easysoft/job
```

and then run:

```
# ./startjob
```

It should respond by saying that:

```
"Starting HTTP Server
The port is set to 8031 in this installation."
```

STOPPING THE JOB SERVER

To stop the job server, ensure that you are logged in as the `root` user, then use the following script:

```
# ./stopjob
```

Response with '`y`' to kill the processes shown.

Uninstalling on Unix

To remove the Easysoft JDBC-ODBC Bridge from your Unix system, you need to:

- stop the JOB Server (see **"Stopping the JOB server" on page 56** for details).
- remove the install directory tree.

REMOVING THE INSTALL DIRECTORY TREE

1. Log in as the user who performed the installation (usually `root`).
2. Enter the following:

   `cd` *<InstallDir>*`/easysoft/`

   `pwd`

   The system displays the current directory.

Double-check that this is the directory under which you installed the Easysoft JDBC-ODBC Bridge:

```
ls
```

```
(Check that you are in the right directory).
```

Caution!

Be very careful issuing the following `rm -r` command as `root`. Normally `rmdir` will not remove directories that contain files, but `rm -r` will remove all subdirectories along with their contents. It is possible to effectively destroy your system and/or lose all user files by removing the wrong directory.

```
rm -r job
```

The system may ask you to confirm deletion for some files.

CONFIGURING THE SERVER

3

Configuring the Easysoft JDBC-ODBC Bridge Server

There are certain configurable parameters that affect the JOB Server, irrespective of the DSN to which you are connecting.

In Windows, these are stored in the registry.

In Unix they are held in the `esjobserver.ini` initialization file.

On either platform you can use the Web Administrator to modify these parameters.

Chapter Guide

Setting up the JOB Server on Windows

In Windows, data sources are made visible to applications through the Microsoft driver manager.

You set up a data source for the JOB Server in the same way as any ODBC application.

The JOB Server for Windows can connect to any *System Data Source Name* (System DSN) configured on your machine, given the necessary information.

If you do not already have a System DSN on your machine then now is the time to create one. You should use the ODBC driver suitable for your data source.

This section consists of a worked example where a System DSN is created for the Microsoft Northwind database, which is shipped with Microsoft Access.

The example is based around a 32-bit ODBC data source, as by default, the JOB installation program starts the 32-bit version of the JOB Server. (The JOB Server and ODBC driver architecture must be the same.) If you must use a 64-bit ODBC data source, for example, because your ODBC driver is 64-bit only, then you need to stop the 32-bit JOB Server in Windows Services, and start the 64-bit Server instead (`Easysoft JDBC-ODBC Bridge Server x64`).

You can follow the example as an exercise on your own computer, providing:

- You have the Microsoft ODBC driver for Access (almost all Microsoft Access installations have this).
- You have an Access database to connect to, such as `northwind.mdb` (provided by Microsoft).

The first step is to open the Microsoft Data Source Administrator on the machine where the JOB Server is installed:

1. In **Control Panel**, double-click **Administrative Tools** and then **Data Sources ODBC Data Sources (32bit)**.

   The **ODBC Data Source Administrator** dialog box is displayed:

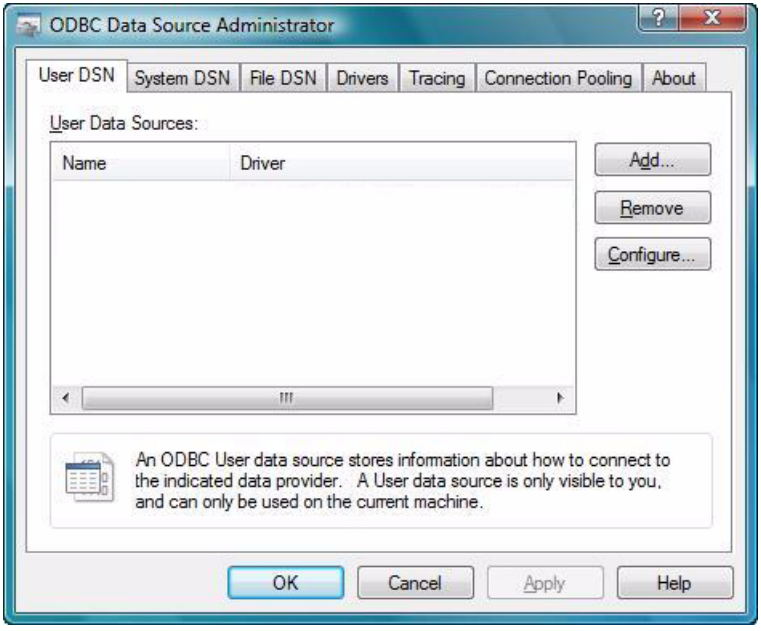


Figure 14: The ODBC Data Source Administrator User DSN Tab

2. Select the **System DSN** tab:

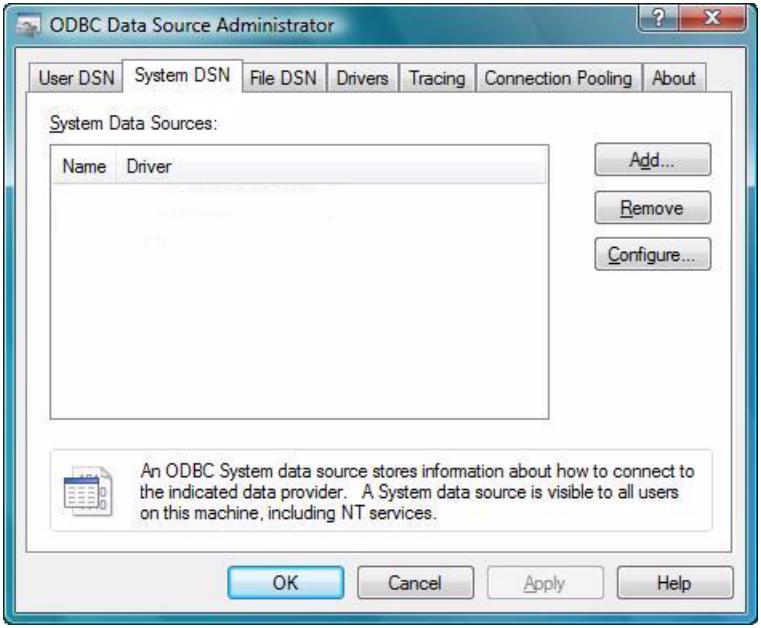


Figure 15: The ODBC Data Source Administrator System DSN Tab

It is important to create a system DSN rather than a user DSN, which is only visible to the desktop user who created it.

Since the JOB Server runs as a service, User DSNs are not available to it.

3. Click **Add...** to add a new DSN.

The **Create New Data Source** dialog box is displayed, containing a list of drivers:

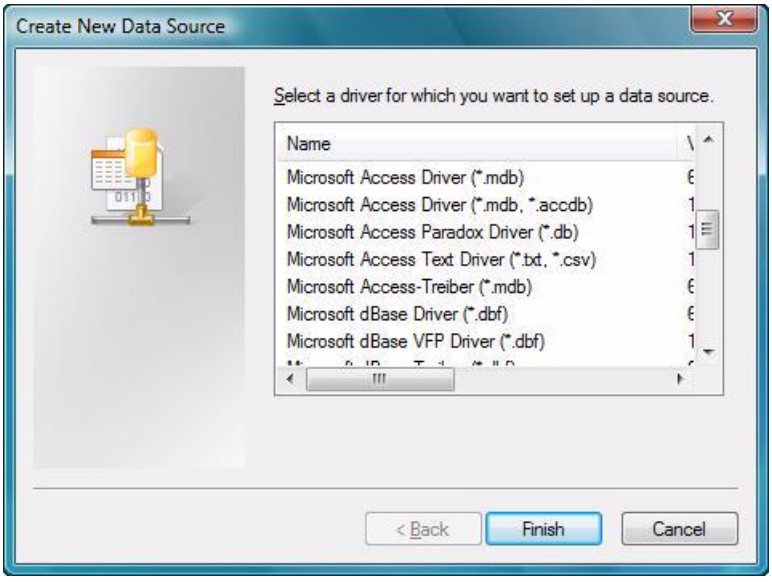


Figure 16: The Create New Data Source dialog box

4. Select `Microsoft Access Driver` and click **Finish**.

   The ODBC driver for Microsoft Access displays a dialog box for configuring the Data Source (this step differs from one database to another).

5. Enter a name for this data source in the **Data Source Name** box ("`NorthwindDSN`", for example).

   The text that is entered into the `Description` field may be displayed to a user if they choose to connect interactively.

   Enter a description, such as `"JOB demonstration DSN"`:

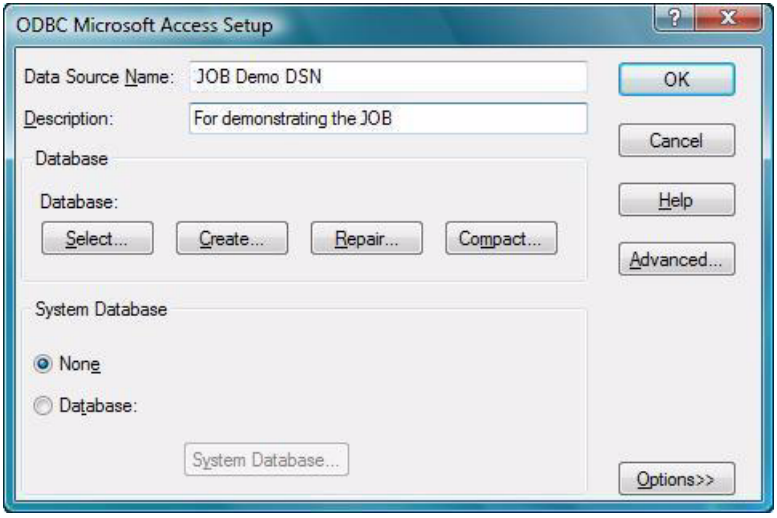


Figure 17: The Microsoft Access ODBC driver configured

6. Click **Select...** to browse for the target database, select your chosen database and click **OK**.

   This example uses the database in `Microsoft Office\Office\Samples\Northwind.mdb`, but this database may not exist or may be in a different location on your system. In this case, use any database you have to hand, preferably a small one.

NB	Remember the Data Source Name, because it will be required when you come to connect through the Easysoft JDBC-ODBC Bridge.

7. Click **OK**.

   You are returned to the ODBC Data Source Administrator window.

Note that:

- the window now contains a line with your new DSN in it
- the **System DSN** tab should be selected (if it is not, then you must remove the DSN, select the **System DSN** tab, and return to **step 3 on page 62**).

8. Click **OK**.

   You have now set up a system-wide Data Source Name on your machine to a local database, making it visible to the JOB Server (and all ODBC programs).

STARTING THE JOB SERVER ON WINDOWS

For your Java program to connect to the local DSN, the JOB Server must be running.

The installation program configures the JOB Server to start automatically as an Windows Service.

Alternatively, use the command-line interface and run `server.bat`, which stops any running JOB Server and Web Administrator and then starts them both again.

Go to **"Connecting from Java" on page 94** to connect a client.

Setting up the JOB Server on Unix

The Easysoft JDBC-ODBC Bridge is shipped as a binary executable linked with the unixODBC driver manager.

There are two ways to make your data source visible to the Easysoft JDBC-ODBC Bridge:

- create a System DSN on the server (see **"Using JOB with unixODBC" on page 66**).
- remove or rename the shared object `libodbc.so` (the suffix may differ from platform to platform) and replace it with a link to your chosen ODBC driver.

If you choose the second approach, then all Easysoft JDBC-ODBC Bridge connections will have to connect to your chosen ODBC driver.

If you use unixODBC you will be able to choose which ODBC driver to connect to at runtime.

NB	unixODBC is a project creating free data access components for Unix platforms. It is not an Easysoft product, but its development and maintenance are sponsored by Easysoft and other industry bodies. For further information about using unixODBC, please visit **http://www.unixodbc.org**.

USING JOB WITH UNIXODBC

If it is supplied with the necessary information, the JOB Server can connect to any system data source configured on a Unix machine.

Easysoft recommend that you use the unixODBC driver manager supplied with the Easysoft JDBC-ODBC Bridge for setting up data sources on Unix.

NB	This section explains how to set up data sources using unixODBC, as installed by Easysoft. If you choose to use a different driver manager, you should refer to the documentation with that driver manager for details of setting up data sources on Unix.

With unixODBC, you can create a data source by either:

- directly adding the data source and its attributes into a configuration file (`odbc.ini`)
- if you are running an X server you can create a data source using the graphical ODBC Data Source Administrator (`ODBCConfig`).

CREATING A DSN BY EDITING A CONFIGURATION FILE

With unixODBC, data sources are stored in a configuration file called `odbc.ini`. If you accepted the default Easysoft JDBC-ODBC Bridge installation, system data sources will be stored in `/etc/odbc.ini`. If you have built unixODBC yourself, the directory is specified in the `--sysconfdir=`*directory* configuration option.

You usually have to be logged in as `root` to edit the system `odbc.ini`.

Each section of the `odbc.ini` file starts with a data source name in square brackets [ ] followed by a number of *`attribute`*`=`*`value`* pairs.

NB	Attribute names in `odbc.ini` are not case sensitive.

The attributes that you need to specify vary depending on which ODBC driver you are using to connect to the local database.

For example, a data source using the PostgreSQL driver will be in the format:

```
[MAIN]
Description = Main data on Admin box
Driver = PostgreSQL
Trace = No
TraceFile = sql.log
Database = main
Servername = localhost
UserName =
Password =
Port = 5432
Protocol = 6.4
ReadOnly = No
RowVersioning = No
ShowSystemTables = No
ShowOidColumn = No
FakeOidIndex = No
ConnSettings =
```

unixODBC uses the `DRIVER` attibute to look up the driver in the `odbcinst.ini` file and locate the shared object to use as the ODBC driver.

Refer to the documentation with your ODBC driver for full details of the attributes it requires to define a data source.

CREATING A DSN USING THE ODBC DATA SOURCE ADMINISTRATOR

To create a data source using the graphical ODBC Data Source Administrator supplied with unixODBC:

1. Run an X session connecting to your Unix machine, ensuring that you log in as `root`.
2. Change into the `<InstallDir>/easysoft/unixODBC/bin` directory.
3. Type `./ODBCConfig` <*Enter*>.

   The ODBC Data Source Administrator opens.
4. Select the **System DSN** tab.

   This will create a data source which is available to any user or service that logs into this machine.
5. Click **Add** to create a new data source.

   The **Adding a New Data Source** dialog box displays a list of the drivers which are available.
6. Select the driver that you want to use to connect to the database.
7. Click **OK**.

   A driver-specific **New Data Source** dialog box is displayed.

The dialog box for setting up a PostgreSQL data source will be in the format:

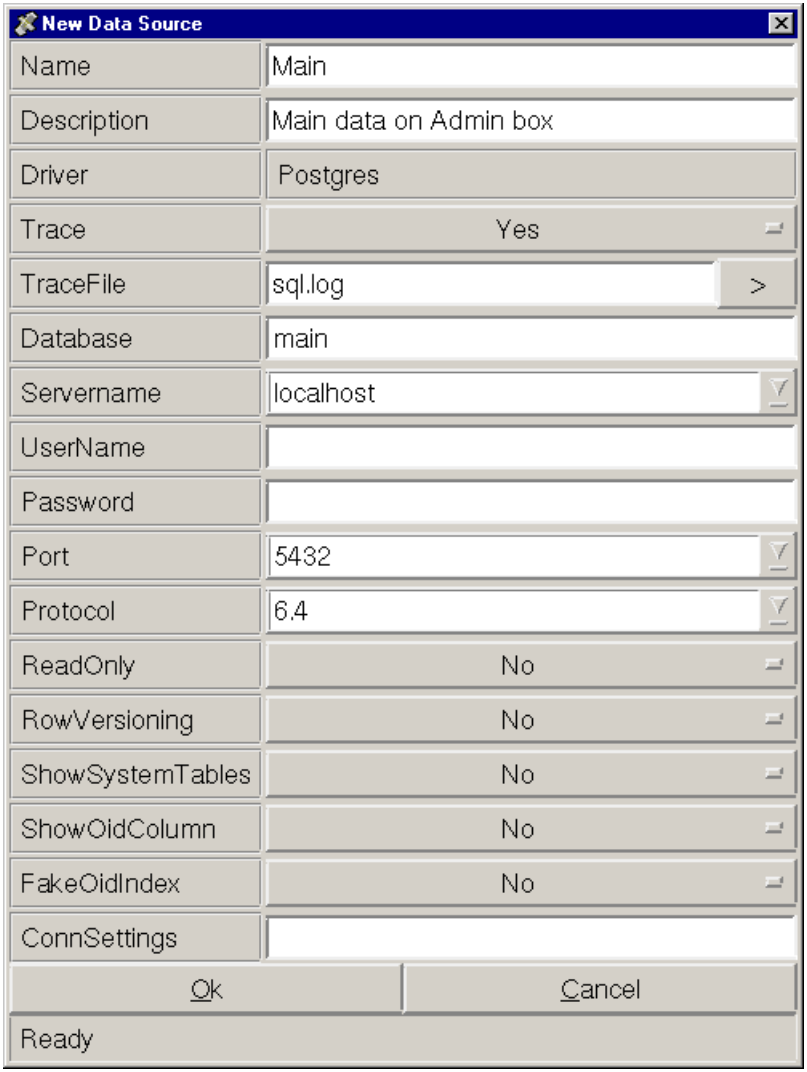


Figure 18: The Configuration dialog box for a PostgreSQL data source

Refer to the documentation with your ODBC driver for full details of the attributes you need to specify on this dialog box.

8. Click **OK** when you have specified all the data source attributes and then close the ODBC Data Source Administrator.

TESTING THE DATA SOURCE

You now have a data source connecting to the database on your server.

Before attempting to connect via the Easysoft JDBC-ODBC Bridge, you should test that this data source is working so that you can be confident that the server side is functioning correctly.

You can do this using any ODBC application available on your Unix machine, or using the `isql` utility provided with unixODBC.

To use `isql` to test the data source:

1. Change into the `<InstallDir>/easysoft/unixODBC/bin` directory
2. Type:

   ```
   ./isql -v data_source_name
   ```

   For example, to connect to the PostgreSQL data source illustrated earlier in this section, you would type:

   ```
   ./isql -v main
   ```

   If the DBMS on your server requires authentication, you should type:

   ```
   ./isql -v data_source_name [user_name] [user_password]
   ```

3. Once connected, type an SQL statement to query the data.

   e.g.

   ```
   select * from table;
   ```

where *table* is a table in that database, or type `help` to get a list of tables in the database.

4. To leave `isql` and return to the system prompt, press `<Enter>`.

Once you have established that the data source on your server machine is accessing data correctly, you can establish a connection to the data source across the Easysoft JDBC-ODBC Bridge.

STARTING THE SERVER IN UNIX

For your Java program to connect to the local DSN, the JOB Server must be running.

Normally the JOB Server will be started at the end of the installation procedure, but if you need to start or stop it manually, change into the `<InstallDir>/easysoft/job` directory and then:

- to start the JOB Server, type:

  ```
  ./startjob
  ```

- to stop the JOB Server, type:

  ```
  ./stopjob
  ```

You should consider putting these scripts into your system's startup and shutdown scripts or into your path.

After using `stopjob` you may need to wait for the socket to time-out before trying to start the JOB Server using `startjob`.

NB	To stop the JOB Server, you must be logged in as the user who started the JOB Server. Usually this will be `root`.

See **"Starting the JOB server" on page 56** for more details.

The Web Administrator

This section provides a tour of the Web Administrator interface and shows you how to change the configurable parameter settings in the JOB Server.

You can also edit these settings through:

- the JOB registry settings on Windows (see **"The Windows Registry entry" on page 89**).
- the JOB initialization file on Unix (see **"The Unix initialization file" on page 90**).

In order to follow this section you will need:

- a running JOB Server.
- a user name and password for the Web Administrator on Windows, as entered on the **User Information** screen during the installation of the Easysoft JDBC-ODBC Bridge (see **"Installing on Windows" on page 36**).

Win	A user name and password must be entered during the installation of the Easysoft JDBC-ODBC Bridge or the Web Administrator JOB Server configuration facility will be disabled.

– OR –

the `root` user name and password on Unix.

NB	The Web Administrator user name is displayed in the case sensitive `HTTPAdmin` field on the **Configuration** screen (see **"The Configuration Screen" on page 79**), which is hidden by default, but can be amended. Update access will be denied if `HTTPAdmin` does not specify a valid user on the server system (setting it to "disabled" disallows HTTP Authentication).

Open the URL `http://`*serverhost*`:8031/` (where *serverhost* is the name of the host running the JOB Server) from a web browser.

Depending on your server configuration, you may need to specify a port address different from the Easysoft default of `8031`.

If you click on a link to a protected page you will be prompted by your browser for the Web Administrator user name and password for the ESJOBServer realm:

Figure 19: The Enter Network Password dialog box

The web page returned is generated by the server process.

The Statistics Screen displays runtime statistics for the latest run of the server and allows access to the following additional screens:

- **The Configuration Screen** allows a user to define JOB Server features.
- **The Security Screen** allows a user to define from which machines the JOB Server allows access.
- **The Information Screen** provides access to various Easysoft locations providing documentation and user support.

- **The Client Hosts Screen** displays details of the different clients which have connected to the JOB Server.

THE STATISTICS SCREEN

The Web Administrator **Statistics** screen contains the following fields:

- **Server up time**

  The time in days, hours, minutes and seconds since the JOB Server was started.

- **Server CPU time (s)**

  This field is only visible if the `ShowProcessTime` flag is set to on (see **ShowProcessTime on page 85**).

  One or more values will be shown:

  – If only one value is shown it is the total CPU time consumed by the JOB Server.

  – If two times are shown the first is user time and the second is kernel time.

  Note that:

  – when the JOB Server is running multi-threaded any of the CPU times shown include CPU time consumed by the ODBC driver manager, any ODBC drivers loaded and any child processes or threads.

  – the process time is only updated when the JOB Server is idle. If the JOB Server is extremely busy servicing incoming connections then the process time will not be updated.

- **Total Connections**

  The total number of connections (or attempted connections) to the JOB Server.

  This will include connections dropped due to no license or insufficient license slots, port scanners or anyone using `telnet` to access the JOB Server port.

- **Total Threads/Processes**

  The total number of threads or processes that the JOB Server has created during its execution.

  Connections denied access because of an access control rule or `MaxThreadCount/MaxClientConnect` being exceeded are not included, because the JOB Server does not start a thread or process for these.

- **Active Threads/Processes**

  The total number of active threads or processes the JOB Server has created to handle connections.

  This number may exceed the actual active count as the JOB Server only looks for exited threads and processes when five seconds has elapsed without any connections (this is done to give preference to incoming connections).

  Note that if `MaxThreadCount` or `MaxClientConnect` is set to anything other than 0 then the JOB Server has to reap exited threads and processes every time a new connection arrives.

Also note that this is not a limit and may therefore exceed your maximum licensed connection slots.

- **Peak concurrent Threads/Processes**

  The highest value ever seen in the `Active Threads/Processes`.

- **Last Connection time**

  The time the last connection occured.

- **Last Disconnect time**

  The time the last disconnection occured.

- **Number of different client hosts**

  The number of different client machines which have connected to the JOB Server (where a client machine is identified by its IP address).

  You can click on this link to get a list of IP addresses or machine names. Machine names are only displayed if you have `ReverseLookup` enabled.

Changing the refresh frequency

The Web Administrator uses a set of template files into which the dynamic data is inserted before sending it back to your browser.

The template file for the **Statistics** screen is `index.html`, which is located in the `/admin` directory wherever you installed the JOB Server.

Edit the `index.html` file and near the top you will see:

```
meta http-equiv="refresh" content="60"; URL=/index.html
```

Change the 60 (the refresh time in seconds) to your preferred setting.

NB	Note that setting the refresh time to a very low value will increase the workload on the JOB Server process which handles HTTP requests. As this may reduce the reponse time to the JOB Server thread, times much less than 60 seconds are not recommended.

THE CONFIGURATION SCREEN

This section explains the parameters that are configurable via the Web Administrator **Configuration** screen.

These parameters apply either to the JOB Server itself, or as a default value for all DSNs accessed via the server.

Brief reminder notes are available by scrolling the browser window. Make your changes and click **Submit**. At the confirmation screen, click another menu option or wait a few seconds to return to the **Server Configuration** screen and see the changes you have made.

These settings, and some additional settings, can also be edited via the registry in Windows (see **"The Windows Registry entry" on page 89**) or in the `esjobserver.ini` file in Unix (see **"The Unix initialization file" on page 90**).

The Web Administrator **Configuration** screen displays the configurable parameters of the server process. Some sensitive settings may not be displayed.

You can modify the server settings by clicking **Change** and then typing the Web Administrator user name and password when prompted.

The Web Administrator user name is the network user entered during the installation process and the password the one required for that user on the system where the JOB Server is running.

> **NB** You are only asked for log in details if authentication has been enabled by entering a value other than `"disabled"` into the `HTTPAdmin` field on the Web Administrator **Change Configuration** screen.

The **Configuration** screen contains the following fields (click **Submit** to make your changes).

Values are case-dependent if the operating-system is, so it is best to match case where possible:

- **Port**

  The port on which the JOB Server listens for incoming JOB Client connections.

  The default port number is `8831`, but it may be any port number not in use on your JOB Server machine.

- **HTTPPort**

  The port on which the JOB Server listens for HTTP requests (i.e. runs the Web Administrator).

  The default port number is `8031`, but it may be any port number not in use on your JOB Server machine.

  If the `Flags` option bitmask has the second bit set (see **"HTTP_Server" on page 84**) the JOB Server starts listening on the specified port for HTTP requests in addition to acting in its normal role serving the JOB Client.

You may use the URL `http://machine_name:HTTPPort` where `machine_name` is the name (or IP address) of the JOB Server machine, and `HTTPPort` is the port number to communicate with the JOB Server from your browser.

- **Timeout**

  The inactivity timeout in seconds (the default is 7200 - two hours).

  The JOB Server starts a new thread (or process) for each client that connects and if there has been no communication in `Timeout` seconds the thread or process exits.

  This ensures clients which fail to closedown properly do not cause increasing resource usage on the server.

  To disable the timeout, set it to `0`.

- **MaxThreadCount**

  The maximum number of threads or processes the Easysoft JDBC-ODBC Bridge Server will allow at any time.

  One thread/process is created for every ODBC connection. If `MaxThreadCount` is set to 0, there is no limit. The default is 100.

  You can use this parameter to prevent too many simultaneous connections swamping your server.

- **RetryCount**

  The number of times the JOB Server will attempt to create a thread or process to handle a connection, or the number of times the the server will attempt to obtain a license slot for a new connection.

- **RetryPause**

  The time in seconds between each retry attempt (see **"RetryCount" on page 82**).

- **SingleStatement**

  By default, the Easysoft JDBC-ODBC Bridge will attempt to pool statement handles in the ODBC driver and open new connections to support multiple concurrent statements, even if the backend ODBC driver does not.

  If you know that you do not need this feature (i.e. you ensure that there is only a single result set open at any one time) set this to "`Y`".

- **HTTPAdmin**

  The user name of the person allowed to make changes to the JOB Server via the Web Administrator.

  This must be a valid user name in the operating system the server is running on. If set to the string "`disabled`" (omit the quotes) then authentication is not required (this does not mean the JOB Server stops authenticating incoming connections).

  The value is case-sensitive.

> **NB**  The "eye" symbol displayed against the `HTTPAdmin` parameter is used to denote that the value of the field is hidden from the user unless they gain access to the corresponding screen by clicking **Change** and entering user name and password details.

- **Path**

  The installation path of the Easysoft JDBC-ODBC Bridge.

  This is a read-only parameter for information only.

- **Logging**

  A bitmask telling the JOB Server what sorts of event to record in the log file.

  This should only be used as directed by Easysoft support and will slow the Easysoft JDBC-ODBC Bridge down considerably if set. You may specify the number as decimal or hexadecimal (e.g. `2047` or `0x7ff`).

- **LogDir**

  The directory where log files are created (see `Logging`).

  It defaults to *`drive`*`:\Program Files\Easysoft\Easysoft JDBC-ODBC Bridge\Logs\` on Windows and `/tmp` on Unix.

- **Flags**

  A bitmask of operational flags.

  The bitmasks are split into check boxes, one for each bit in Flags:

  – **Authentication_Disabled** (0x1) If set authentication is disabled in the JOB Server. Setting this parameter should be considered as a security risk. However, on some high hit servers in a controlled environment where you do not need to authenticate the connections this can save a considerable amount of time during connections (e.g. in Windows NT, 0.25 to 0.75 seconds of the connection time can be spent in authentication). The default is off.

  – **HTTP_Server** (0x2) If set the JOB Server listens for HTTP connections to the Web Administrator. The default is on.

  – **MultiProcess** (0x4) If set the JOB Server starts a new process rather than a new thread for each incoming connection. Use this setting for an ODBC driver which is not thread-safe or is leaking memory. The default is off.

Unix	The **MultiProcess** flag cannot be updated, as the server is not multi-threaded and therefore always starts new processes.

- **HideSensitive** (0x10) If set the Web Administrator hides sensitive parameters on the **Configuration** page. The `HTTPAdmin` and `Authentication_Disabled` parameters are hidden when `HideSensitive` is set. All parameters are always shown in full on the change configurable parameters page as this page is password protected. The default is off.
- **ReverseLookup** (0x20) If set the JOB Server calls `gethostbyaddr()` on the connecting client's IP address to obtain the client's machine name. On machines where DNS is not set up properly this can cause problems and in any case adds time to the connection. `ReverseLookup` currently only affects the number of different clients page on the Statistics screen, where `"unknown"` will be displayed instead of the machine name for the machine names of connecting clients if `ReverseLookup` is off. The default is off.
- **AuditODBCAccess** (0x40) If set the JOB Server audits all connections to a log file which may be viewed through the Web Administrator. The audit file is written to the `LogDir` directory as `esjob_access.log`. The default is off.
- **ShowProcessTime** (0x100) If set user and kernel CPU times for the main JOB Server process are displayed on the Web Administrator **Statistics** screen (see **Server CPU time (s) on page 76**). Values are updated every five seconds. The default is off.

THE SECURITY SCREEN

The Web Administrator **Security** screen displays and allows the user to view or change the set of hosts that are allowed to connect to the Easysoft JDBC-ODBC Bridge.

The Web Administrator user name and password are required if they have not yet been entered:

In addition to the user names and passwords of your system and of the database management system, the Easysoft JDBC-ODBC Bridge provides another layer of security with access control lists.

To add an IP address to a list, type the address or address-stem into either the **Allowed Access** or **Denied Access** boxes and click **Add**.

Unix administrators will recognize this mechanism from the `hosts.allow` and `hosts.deny` files.

NB	Although the approach is similar, the rules for determining whether or not a host should be allowed to connect are different from those for `hosts.allow` and `hosts.deny`.

Security for the Easysoft JDBC-ODBC Bridge takes the form of two lists of IP addresses.

When a host attempts to connect to the JOB Server, access is only granted if:

- the `allowed` list is empty and the IP address is not in the `denied` list.
- the IP address is in the `allowed` list.

The lists can be edited either in the registry (Windows), their relevant flat files (Unix), or via the Web Administrator (Windows and Unix when run standalone.)

Addresses must be entered using the IP ‘dot’ notation.

Entries which consist of fewer than four fields represent all the addresses that match the fields which are defined.

e.g.

`163.141.23.` (note the trailing dot) matches all IP addresses from `163.141.23.0` to `163.141.23.255`..

NB	Direct editing is not supported by Easysoft and will not allow access to changed values until the server has been rebooted.

THE INFORMATION SCREEN

The Web Administrator **Information** screen displays a list of links to Easysoft support resources:

- **README -** A document listing any recent updates to the Easysoft JDBC-ODBC Bridge product.
- **JOB Programming Guide -** A guide for Java developers who wish to add JDBC connectivity to their applications using the Easysoft JDBC-ODBC Bridge.
- **JOB Getting Started Guide -** A guide to the download, installation and licensing of the Easysoft JDBC-ODBC Bridge, and to using the sample Java test program to access ODBC compliant data.
- **EJOB.jar -** Download a copy of the Easysoft JDBC-ODBC Bridge `EJOB.jar` file.
- **JOB Product Page -** The Easysoft JDBC-ODBC Bridge page on the Easysoft web site.
- **Mail Easysoft Support -** A form allowing a user to email a support request for the Easysoft JDBC-ODBC Bridge.

THE CLIENT HOSTS SCREEN

The Web Administrator **Client Hosts** screen displays a list of individual clients which have connected to the JOB Server. These connection attempts may not have been fully successful (e.g. not authenticated or denied access).

The screen contains the following fields:

- **IP Address -** The IP addresses of connecting clients.

- **FQDN -** The Fully Qualified Domain Name of connecting clients, which are displayed as `"unknown"`, unless `ReverseLookup` is turned on (see **"ReverseLookup" on page 85**).

The Windows Registry entry

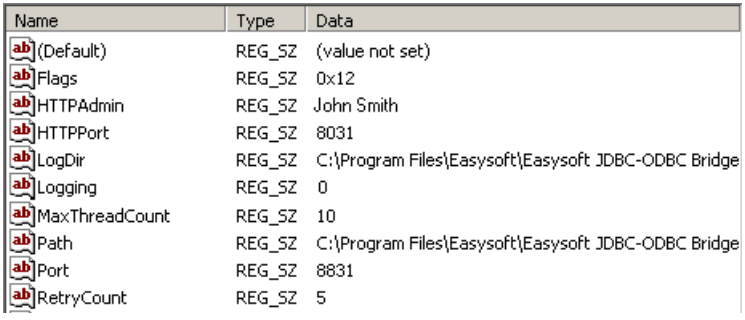
If you cannot remember the user name or you do not know the corresponding password, then you will not be able to modify the configuration of the JOB Server via the Web Administrator.

In this case you need to log in directly to the server host and edit the registry.

The registry key for the Easysoft JDBC-ODBC-Bridge is:

```
HKEY_LOCAL_MACHINE/SOFTWARE/Easysoft JDBC-
ODBC Bridge/Configuration/System/settings
```

Note that the registry key contains a string value for each of the Server Configurable Parameters (numeric parameters are stored in strings):



Name	Type	Data
(Default)	REG_SZ	(value not set)
Flags	REG_SZ	0x12
HTTPAdmin	REG_SZ	John Smith
HTTPPort	REG_SZ	8031
LogDir	REG_SZ	C:\Program Files\Easysoft\Easysoft JDBC-ODBC Bridge
Logging	REG_SZ	0
MaxThreadCount	REG_SZ	10
Path	REG_SZ	C:\Program Files\Easysoft\Easysoft JDBC-ODBC Bridge
Port	REG_SZ	8831
RetryCount	REG_SZ	5

Figure 20: The Windows Registry entry

See **"The Configuration Screen" on page 79** for full details of each of these settings.

The Unix initialization file

If you do not know the password then you will not be able to modify the JOB Server configuration via the Web Administrator.

In this case, for the you need to log in directly to the server host and edit <*InstallDir*>`/easysoft/job/server/esjobserver.ini`.

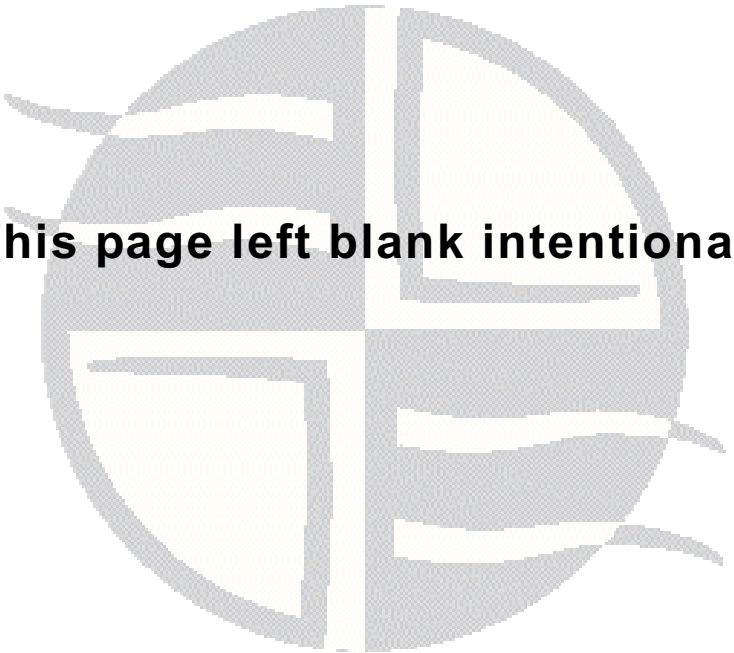
If <*Installir*> is anything other than `/usr/local/` then there will be a symbolic link `/usr/local/easysoft` to the real `Easysoft` directory.

An example `esjobserver.ini` file follows:

```
{Settings}
Port = 8831
HTTPPort = 8031
Timeout = 3600
LogDir = /tmp
Logging = 0
Flags = 38
RetryCount = 5
RetryPause = 3
AllowList =
DenyList =
MaxThreadCount = 0
MaxClientConnect = 0
HTTPAdmin = John Smith
```

See **"The Configuration Screen" on page 79** for full details of each of these settings.

NB	Lines beginning with `#` are comments. The first non-comment line should be `{Settings}` and the rest of the file should be *Key*=*Value* pairs. Bitmask values are given either in decimal or in hexadecimal. The `Flags` attribute must be included, and must be given a value of `38`, unless you are advised otherwise by Easysoft.



This page left blank intentionally

CONFIGURING THE CLIENT

4

Configuring the Easysoft JDBC-ODBC Bridge Client

This section takes you through connecting to an ODBC data source from an existing JDBC program.

Chapter Guide

Connecting from Java

In JDBC, you connect to data sources using an URL, the format of which is dependent on the driver being used.

The format for an Easysoft JDBC-ODBC Bridge URL is:

`jdbc:easysoft://`*hostname*`:`*port*`/`*[RemoteDSN]*{`:`*attr*`=`*value*}*

where *[ ]* denotes an optional item and { }* denotes zero-or-more occurences.

> **NB** You must omit an attribute altogether (rather than passing an empty string) if you do not wish it to apply.

hostname and *port* describe how to access the remote machine where the JOB Server is running and should be used in the same way as in the address box of a browser, except that the additional name `appletHost` is supported.

`appletHost` is understood by the JOB Driver class and refers to the server that the applet was loaded from. For an example of using `appletHost`, click the **Getting Started** link at **http://www.easysoft.com/products/data_access/jdbc_odbc_bridge/index.html** and look at `OutputApplet.java`.

RemoteDSN identifies a System DSN (Data Source Name) configured on that machine.

The DSN should be set up with all the attributes it requires on the remote machine, with the possible exceptions of `UID` and `PWD` which can be specified on connection.

The following attributes are recognized:

`:user=`*Database User ID*

`:password=`*Database Password*

`:logonuser=`*Remote System User ID*

`:logonpassword=`*Remote System Password*

`:cleartext=on` *or* `off`

`:limit=`*Row Limit*

`:trace=on` *or* `off`

`:unicode=on` *or* `off`

- `user` and `password` are given to the ODBC driver as `UID` and `PWD` and must be used if the back-end database requires authentication.
- `logonuser` and `logonpassword` specify an existing user on the target machine, but are only required if authentication is enabled on the JOB Server.

NB Set the `ClearText` attribute to true if you wish to be able to see the contents of these fields.

They may also be used if the database engine requires that the connecting process (i.e. the JOB Server) is logged on via a specific user.

Caution! Running the Windows service as a specified user (other than local admin) means that authentication must be disabled and that any user can then therefore connect, creating a security risk. You will also be unable to connect to trusted data sources (such as Microsoft SQL Server) as a system account.

- `cleartext` disables the default encrypted transmission of connection information (connection string, user names and passwords) over the network (useful for debugging purposes).

- `limit` fixes the maximum number of rows that will be returned in any result set, limiting the time and space requirements that can be generated by any single query.

  The limit is actually set in the ODBC driver by the JOB Server, so it is supported only if your ODBC driver supports it.

> **NB** The value of `Limit` cannot be changed during the lifespan of a connection, so it must be set either to a value which will accommodate the largest possible result set or be omitted (which means no limit is set).

- `trace` specifies whether the Driver (i.e. not the server) should write a detailed log to the Java Console.

  If set to `off` or omitted then no trace file is written. In the trace, return values, procedure calls (beginning `*`) and Remote Procedure Calls (beginning `-->`) with their return values are logged.

> **NB** Note that setting the `trace` attribute slows down performance considerably.

- `unicode=on` specifies that all character data is retrieved as Unicode.

  This setting must be specified in cases where back-end data sources support Unicode, but do not report that data conforms to the Unicode standard. The default is `unicode=on`, so that only columns containing Unicode data are retrieved as Unicode. Set to `unicode=off` to prevent any data from being retrieved as Unicode.

NB	You do not have to set up a DSN in order to make a connection. Instead, you can specify one or more connection attributes by using the *attr=value* clause within the Easysoft JDBC-ODBC Bridge URL.

WHERE TO PUT EJOB.JAR

Before you can connect your Java application to an ODBC data source, you must make the client JDBC driver (`EJOB.jar`) available to your application. You do this by doing either of the following:

- add `EJOB.jar` to your CLASSPATH
- unjar the `EJOB.jar` file into the directory where you are running your application, using the following command:

  ```
  jar xvf EJOB.jar
  ```

  This will create an `\easysoft` directory (and subdirectories) within your application directory.

CONNECTION DIFFICULTIES

If you experience difficulties when connecting your Java applet or application to the Easysoft JDBC-ODBC Bridge, check the following:

- ensure you have added `EJOB.jar` to your CLASSPATH or have copied to the directory where you are running your application
- check that you are using the appropriate ports in the JDBC URL. The defaults are `8831` for the JOB Server and `8031` for its web server
- when connecting from an applet, check that you have specified `archive=EJOB.jar` in the `<applet>` tag.

For example:

```
<APPLET
ARCHIVE=EJOB.jar
CODE="OutputApplet.class"
WIDTH=250
HEIGHT=200>
<PARAM NAME = "jdbcUrl"
VALUE = "jdbc:easysoft://appletHost:8831/northwind">
</PARAM>
</APPLET>
```

- when connecting from an applet, if you receive a `'no suitable driver'` error from the JDBC driver manager then the JDBC driver manager cannot find a JDBC driver that can recognise the JDBC URL with which you have tried to connect.

  This could be because the JDBC URL is not correct or because the driver classes are not loaded.

  Try using the following syntax to resolve this:

  ```
  Class.forName("easysoft.sql.jobDriver).newInstance();
  ```

- if you are running Windows 95/98, remember that you must start the JOB Server manually by selecting **Start > Program Files > Easysoft > Easysoft JDBC-ODBC Bridge > JOB Server** (see **"Starting the JOB Server on Windows" on page 65**).

Additional documentation and a connection example are provided with the Web Administrator, listening by default at port `8031`.

To access this information, run a web browser and go to:

`http://`*serverhost*`:8031/`

where *serverhost* is the name of the host running the JOB Server. If the serverhost is your local machine, you can type:

```
http://localhost:8031/
```

Getting Started provides details on how to make a connection.

Tuning

SIZE

The size of your Java applet affects download times, which can be critical in a consumer website.

The Easysoft JDBC-ODBC Bridge driver, in its compressed `.jar` file, is only 64k.

You could place the `.jar` file into the `CLASSPATH` on the client machines, so that it does not need to be downloaded at all, but this small benefit in reduced size is almost certainly outweighed by the loss of zero-installation functionality.

To reduce the size of your bytecode, try compiling the application or applet with Sun's `javac`, rather than whatever IDE you are using. Obfuscate the classes and place them in a `.jar` file.

SPEED

If you are returning large queries, but are only interested in the first few records, try setting the *limit* attribute in the connection, which reduces network traffic by restricting the maximum number of rows that can be returned in any single result-set.

Note that it is important to set the attribute to a large enough value so that no important behavior will be lost and that this feature will normally only be of use while developing your program.

To further reduce network traffic, get the DBMS to perform as much of filtering as possible.

The network is probably more of a bottleneck than CPU or I/O resources on the server, so it is better to design relatively complex queries that return a minimal result set rather than simple, naive queries that are then processed on the client side.

PROGRAMMING GUIDE

5

Easysoft JDBC-ODBC Bridge Programming Guide

This section is aimed at Java developers who wish to add JDBC connectivity to their applications using the Easysoft JDBC-ODBC Bridge.

It is assumed that the JOB Server has been successfully installed and configured and that you know the **hostname** for the server to which you wish to connect.

Chapter Guide

Introduction

The Easysoft JDBC-ODBC Bridge comes in two components:

- The JOB Driver: a 100% Java JDBC 1.2/2.1 compliant driver.
- The JOB Server: a server built specifically for the platform where your target ODBC System Data Source is configured.

The JOB Server incorporates a Web Administrator (see **"The Web Administrator" on page 73**) that provides access to administration, monitoring and test facilities through a standard web interface.

The classes that make up the driver are in the Java archive file `EJOB.jar`. The driver class is called `easysoft.sql.jobDriver`.

Java applications

Download `EJOB.jar` to the client machine from the JOB Server at `http://hostname:8031/jars/EJOB.jar` (where `8031` is the web server port).

On the client machine you must either amend or create the '`CLASSPATH`' setting in the **Environment Variables**, which are found under **Start > Settings > Control Panel > System > Advanced** in Windows 2000 (this may vary with other versions of Windows).

If '`CLASSPATH`' already contains a value, add a semi-colon and the full path and file name of your `EJOB.jar` file to the existing value.

To add a new '`CLASSPATH`', click on the **Environment Variables** tab and click **New**.

Put 'CLASSPATH' in the Variable Name, the full path and file name in the Variable Value and then click **OK**.

Please note that you need to update either the User or System Environment Variable list as your application requires.

Java applets

For applet deployment, place the EJOB.jar file with your applet in a directory readable by your web server and add the appropriate ARCHIVE=EJOB.jar attribute to the APPLET tag on your HTML page.

EJOB.jar can be downloaded from the link on the installed Easysoft JDBC-ODBC Bridge Web Administrator **Information** page:

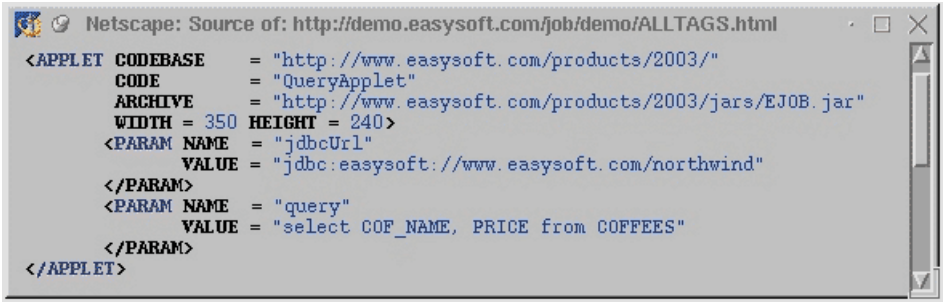


Figure 21: An example applet tag

Making applets portable

In **Figure 46 on page 103** the jdbcUrl explicitly specifies the host on which the JOB Server is running.

Applet restrictions dictate that applets can only open socket connections to machines from which they have been downloaded.

There is no browser independent way of determining from where a `jar` file has been downloaded.

The Easysoft JDBC-ODBC Bridge includes a facility that enables a short form of the URL to be used, eliminating the need to specify the **hostname** in your applet tag or Java code.

To achieve this the Web server must be installed on the same machine as the JOB Server.

Ensure the file `ez.class` is placed in the same directory as the HTML file used to run your applet, or in the directory specified as the `CODEBASE` attribute in your applet tag.

The driver will now attempt to discover the name of the host that it is loaded from, and will attempt a connection to the default `8831` port.

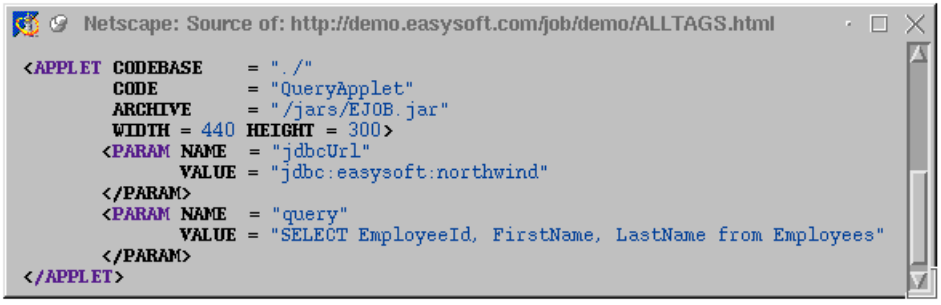


Figure 22: An example applet tag using ez.class

`ez` is a subclass of the `easysoft.sql.jobDriver` and can be used as an alternative short name in interactive applications which require you to enter the JDBC driver class name by typing `ez` instead of `easysoft.sql.jobDriver`.

You will find `ez.class` in `http://localhost:8031/demo/ez.class`:

```
import easysoft.sql.*
public class ez extendsjobDriver {
              public ez() throwsSQLException {
              }
}
```

Loading JOB Driver classes

You can load the JOB Driver classes in your code using:

```
Class.forName("easysoft.sql.jobDriver").newInstance();
```

Alternatively you could use:

```
Class.forName("ez").newInstance();
```

which in turn will instantiate the `easysoft.sql.jobDriver` class.

Connecting to the JOB Driver

An example connection to an URL with the Easysoft subprotocol:

```
String jobUrl= "jdbc:easysoft://myhost/myodbcdsn";
Connection con = DriverManager.getConnection(jobUrl);
```

The general form of a JOB URL is:

```
<url> ::= jdbc:easysoft:[<server spec>][<database>]
              {:<attribute>=<value>}*
<server spec> ::= //[<host name>][:<port>]/
<database> ::= <dsn> | DSN=<dsn> | FILEDSN=<filedsn>
              <DSNlessconnection string >
```

where | separates optional items, [ ] denotes an optional item and { }* denotes zero or more occurences.

DSNLESS CONNECTIONS

```
jdbc:easysoft://<host>/Driver={<driver
name>}[;<attr>=<value>]+
```

e.g.

```
jdbc:easysoft://<host>/Driver={Microsoft Access Driver
(*.mdb)};DBQ=C\:tsmall.mdb;trace=on
```

Note that when using DSNless connections the JOB JDBC URL attribute separator ':' is changed to ';' so that ':' can be used as part of the ODBC connection attributes (as in Windows driver names).

Easysoft JDBC URL attributes

Attribute	Description
:user	Database User ID
:password	Database Password
:logonuser	Remote System User ID
:logonpassword	Remote System Password
:clearText=on|off	Disable the default encrypted transmission of connection information (connection string, user names and passwords) over the network (useful for debugging purposes)
:limit = n	The maximum number of rows returned in any result set
:trace=on | off	Turn on/off tracing to stdout in the client
:traceFile=<filespec>	Turn on tracing and redirect output to a named file. Note that ':' is used as the attribute separator in JOB URLs. Hence you need to use the escape characters %3A for ':' in the file specs for `traceFile` attributes on Windows, e.g. `:traceFile=C%3A\logs\job.log`
:multi=on	Prefix each line in a trace with the thread ID
:fetchSize=0	Turn off retrieval of columns in blocks
:unicode=on | off	Turn on retrieval of Unicode data from Unicode fields

Figure 23: Easysoft JDBC URL attributes

Additional notes

host name and **port** describe how to access the remote machine where the JOB Server is running.

They must be explicitly specified in the JDBC URL if you wish to connect to a remote JOB Server.

user and **password** are given to the ODBC driver as UID and PWD.

They must be used if the database engine requires authentication.

logonuser and **logonpassword** specify an existing system user on the target machine.

They must be used if the database engine requires the connecting process (in this case the JOB Server) to be logged in as a particular user..

Caution!

Running the Windows service as a speciifed user (other than local admin) means that authentication must be disabled and that any user can then therefore connect, creating a security risk. You will also be unable to connect to trusted data sources (such as Microsoft SQL Server) as a system account.

limit is used to restrict the number of rows that will be returned in any result set.

The limit is actually set in the ODBC driver by the JOB Server as statement attribute `SQL_ATTR_MAX_ROWS`, so it is supported only if your ODBC driver supports it.

fetchSize should be set to zero to avoid the retrieval of far more columns (the default is 4k) in a results set than will actually be needed by the application.

Easysoft JOB URL examples

JOB Server	Platform
jdbc:easysoft:northwind	Driver will connect to the host from which the `ez.class` resource is obtained. This could be the local host for applications or the host from which the applet is downloaded, and will establish a connection to named ODBC data source '`northwind`' in the same way as the Sun JDBC bridge.
jdbc:easysoft:DSN=northwind	Equivalent to `jdbc:easysoft:northwind`.
jdbc:easysoft:	Driver will connect to the host as with `jdbc:easysoft:northwind`. Since no database is specified, a list of available DSNs will be displayed in a dialog box.
jdbc:easysoft:FILEDSN=myfiledsn	As with `jdbc:easysoft:northwind`., but connecting to a file DSN data source, rather than a system DSN data source.
jdbc:easysoft://:8899/northwind	Connect the JOB Server to a non-default port.

Figure 24: Easysoft JOB URL examples

TECHNICAL REFERENCE

A

Technical Reference for the Easysoft JDBC-ODBC Bridge

This section contains extra information relating to the deployment of the Easysoft JDBC-ODBC Bridge.

Appendix Guide

Java runtimes and browsers

Applications require the Java Runtime Environment (JRE) 1.2 or higher.

Windows platforms

The Easysoft JDBC-ODBC Bridge runs as a service under Windows.

Unix platforms

Log onto the Easysoft JDBC-ODBC Bridge Platforms page on the **Easysoft** website for details of currently available releases.

JDBC versions supported

In addition to support for all JDBC 1.22 methods, the Easysoft JDBC-ODBC Bridge also supports all JDBC 2.1 functions (such as batch updates, scrollable results sets, programmatic inserts, deletes, and updates), except SQL 3 data types.

ODBC version required

The JOB Server makes ODBC 2.0 and ODBC 3.0 calls.

The use of a driver manager means that you can connect it to ODBC 2.0 drivers and the driver manager will remap the calls.

As Easysoft recommend the Microsoft driver manager for Windows and the unixODBC driver manager for Unix, if you decide to use a different driver manager then you need to be sure that it supports ODBC 3.0 and ODBC 2.0 calls.

Supported data types

All standard JDBC data types are supported.

Classpath

The default Easysoft JDBC-ODBC Bridge driver classes are found in:

```
C:\Program Files\Easysoft\Easysoft JDBC-
ODBC Bridge\Jars\EJOB.jar
```

(Windows)

or

```
/usr/local/easysoft/job/EJOB.jar
```

(Unix)

> **NB** Alternate directories may be specified during installation.

This location must be added to the classpath in order to compile under Oracle's JDK or run under the JRE.

For further information about driver classes and class paths, see **http://java.sun.com/j2se/1.4/docs/tooldocs/windows/classpath.html** (Windows) or **http://java.sun.com/j2se/1.4/docs/tooldocs/solaris/classpath.html** (Unix).

Tracing

CLIENT

The standard Driver Manager methods:

```
DriverManager.setLogStream(java.io.PrintStream(java.io.Print
Stream out))
```

(for JDBC 1 driver managers)

or

```
DriverManager(java.io.PrintWriter out)
```

(for JDBC 2 driver managers)

can be used in your code to enable or disable tracing to a specified output file.

It is also possible to turn on tracing to the standard output (Java console) by adding the `:trace=on` attribute to the Easysoft JDBC URL.

Set the `clearText=on` attribute to disable the default encrypted transmission of connection information, which can make the tracing output unreadable.

For multithreaded applications and applets you should also set the `:multi=on` attribute, so that each line of the generated trace is prefixed with an indication of the thread that produced that line.

e.g.

```
jdbc:easysoft://host/dsn:trace=on:multi=on
```

SERVER

Use ODBC tracing on the machine where the JOB Server is installed.

On Windows :

Start the ODBC driver manager administration interface via **Start Menu>Control Panel>ODBC Data Sources**, click on **Tracing**, ensure the specified filename is valid and click **Start Tracing Now**.

See **ODBC Tracing** for further details.

On Unix:

If you are using the unixODBC driver manager then tracing is enabled in the `odbcinst.ini` file (usually `/etc/odbcinst.ini`).

To enable tracing you must add two attributes to the [ODBC] section (if you do not have an `[ODBC]` section, create one):

```
Trace = Yes
TraceFile = /path/filename
```

e.g.

```
[ODBC]
Trace = Yes
TraceFile = /tmp/sql.log
```

Tracing will only work if the user who is running the application to be traced has write permission to `TraceFile` (and to the directory containing it).

Unicode

Support for Unicode data in the Easysoft JDBC-ODBC Bridge can be enabled by setting the JDBC URL connection attribute `:unicode=on` (see **"Connecting from Java" on page 94**).

Use one of the two `getString` methods in the `ResultSet` class to read Unicode data and the `setString` method in the `PreparedStatement` class to write Unicode data.

Read and write statements to metadata table, row and column names containing Unicode characters are not supported, unless those names are restricted to ASCII characters only.

The `executeQuery` method in the `Statement` class does NOT support SQL containing Unicode character strings.

NB	Unicode data can also be be read by using one of the two `getUnicodeStream` methods in the `ResultSet` class and written by using the `setUnicodeStream` method in the `PreparedStatement` class in addition to the `getstring` and `setstring` methods.

Firewall and security issues

JOB Server connections are made through the port designated in the connection URL and any firewall must allow TCP protocol traffic on this port.

If the Java client program is an applet then the browser of a user will normally restrict network connections to the host from which the applet was served.

Note that the JOB Driver recognizes a special hostname in the URL, `appletHost`, which always refers to the host that the applet came from.

Using this hostname means that the name of the server may be changed or the ODBC data source and applet moved without breaking the reference.

For an example of using `appletHost`, follow the instructions in the **Getting Started** link at **http://www.easysoft.com/products/data_access/jdbc_odbc_bridge/index.html** and look at `OutputApplet.java`.

For general use in browsers, the applet must be stored on the machine where the ODBC data source is set up. If this is not the web server then for low-traffic applications the Web Administrator built into the JOB Server may be used to serve the single page that contains the `<APPLET>` tag.

For heavier-duty applications it may be advisable to install a fully-fledged web server (such as Apache) on the machine holding the data sources.

Although the JOB Server and ODBC Driver must both be on the machine that serves the applet, the actual database does not have to be running on that same machine. Oracle, DB2, Informix and SQL Server are all examples of such network-enabled databases.

This applet restriction can normally be overcome by providing a signed applet, which a user may choose to "trust", thus allowing access to a wider range of network hosts.

Threading

CLIENT

The JOB Client is safe to use in multi-threaded applications and applets. It fully meets the requirements of the ODBC 3.5 specification, with the proviso that the ODBC data source must support asynchronous statement execution (see **http://java.sun.com/products/jdbc/driverdevs.html**).

All operations on `java.sql` and `javax.sql` objects are required to be multi-thread safe able to cope correctly with having several threads simultaneously calling the same object.

In other words, a statement execution in one thread should not block an execution in another thread. In particular, JDBC drivers should operate correctly when used from multiple threads.

An example of a specific use of multi-threading is the way a long-running statement can be cancelled. This is done by using one thread to execute the statement and a second one to cancel it with the method `Statement.cancel`.

Although in practice most JDBC objects are accessed in a single-threaded way, there needs to be support for multi-threading.

Some database APIs, such as ODBC, provide mechanisms for allowing SQL statements to execute asynchronously. This allows an application to start up a database operation in the background and then handle other work (such as managing a user interface) while waiting for the operation to complete.

As Java is a multi-threaded environment, there is no need to provide support for asynchronous statement execution. Java programmers can easily create a separate thread if they wish to execute statements asynchronously with respect to their main thread.

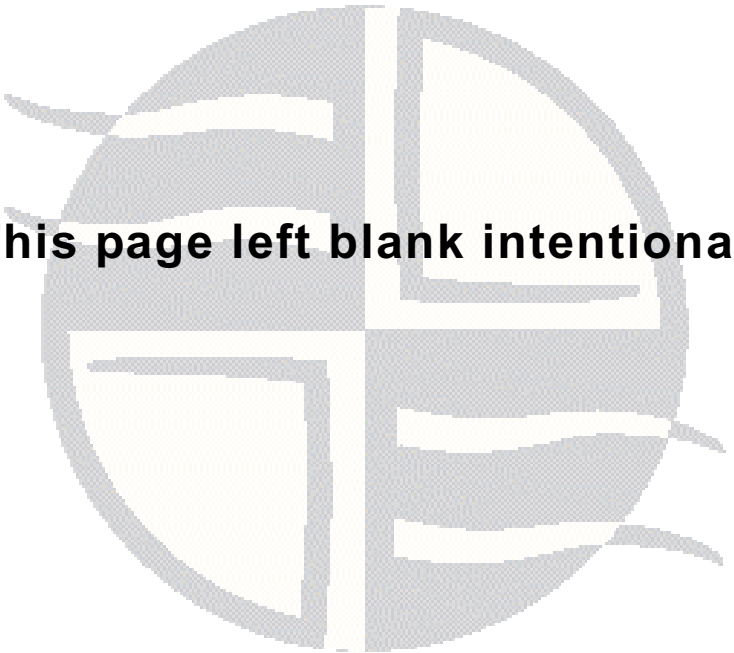
Some drivers may allow more concurrent execution than others, but developers should be able to assume fully concurrent execution. If the driver requires some form of synchronization, then the driver should provide it. In this situation, the only difference visible to the developer should be that applications run with reduced concurrency.

For example, two Statement objects on the same connection can be executed concurrently, and their result sets can be processed concurrently (from the perspective of the developer). Some drivers will provide this full concurrency, while others may execute one statement and wait until it completes before sending another one.

SERVER

Under Windows, JOB Server may run in either multi-threaded or multi-process mode:

- In the default multi-threaded mode, the JOB Server creates a thread for each connection from a JOB Client.
- In multi-process mode, the JOB Server creates a process for each connection from a JOB Client (see **"MultiProcess" on page 84**).
- If the ODBC driver you are using is not thread-safe (e.g. Microsoft Access Jet versions prior to 4.0), run the JOB Server in multi-process mode.
- Under non-Windows platforms, the JOB Server is not required to be thread-safe, because a new process is started to handle each connection, either by the JOB Server in standalone mode (without `inetd`) or by `inetd` itself (if JOB is started as an `inet` service). There can only be one thread of execution in these circumstances.



This page left blank intentionally

JDBC CERTIFICATION

B

Keylabs Certification Acceptance Report

KeyLabs JDBC Certified

Certification Acceptance Report

Date: 6th-Jan-2003

Congratulations! The product you have submitted to KeyLabs has successfully met the compliance criteria and has been accepted. Your product is now officially "JDBC Certified"! Below, you will find specific details concerning all of the testing performed.

VENDOR AND PRODUCT DESCRIPTION

Vendor: Easysoft Limited

Product: Easysoft JDBC-ODBC Bridge

Description: The Easysoft JDBC-ODBC Bridge allows JDBC access to any remote ODBC datasource.

PLATFORMS TESTED

The product submitted was successfully tested for certification compliance on each of the following platform(s) and environment(s):

- Sun Ultra 10 using Solaris 8, CDE 1.4, Solaris 8 with Recommended Cluster Patches

CERTIFICATION TOOLS USED FOR TESTING

- Java Platform Tested Against: JDK 1.4
- J2EE Version Used: 1.3.1
- JavaTest Version Used: 2.1.5

JAVATEST TESTCASE RESULTS

- All JavaTest JDBC Testcases completed sucessfully.

Keylabs

385 South 520 West

Lindon, UT 84042

Phone: 801-226-8200

Fax: 801-226-8205

E-mail: **info@applabs.com**

GLOSSARY

C

Terms and definitions

API

Application Programmer Interface. An API is a published set of function calls and constants allowing different programmers to utilize a ready-written library of subroutines.

Applet

A Java program that has certain restrictions placed upon it, such as restricted network and filesystem access. Applets are able to run in a web browser.

Application

By contrast to an Applet, a Java Application has unrestricted network and file access.

Authorization code

You must have an authorization code for the Easysoft product you wish to license in order to obtain a purchased license. When you purchase a product your authorization code is emailed to you. You do not need an authorization code to obtain a trial license.

Bitmask

A value which, when written out in binary, has a meaning assigned to each digit, which can be 0 or 1. This is a very efficient way of storing a number of *flags* in a small amount of memory. When viewed in decimal it is a single number resulting from adding up the values of the individual bits, worth 1, 2, 4, 8, 16, 32 and so on.

Client/Server

The name given to the architecture whereby one process (the *server*) keeps track of global data, and another (the *client*) is responsible for formatting and presenting the data. The client requests queries or actions be performed on the data by the server. Often these processes run on different *hosts* across a local-area network.

Column

The vertical dimension of a table. Columns are named and have a *domain* (or *type*). The term *column* might refer to only the *definition* of a column (i.e. its name and type), or to all the data in it.

Connection String

ODBC *driver managers* accept a connection string when a client connects. Ideally it contains all necessary attribute values to make the connection to a data source, but provision is made for the driver to negotiate with the application or the user for any missing information.

Data source

In the context of JDBC or ODBC, a data source is a database or other data repository coupled with an *ODBC driver*.

DLL

Dynamic Link Library. Windows' mechanism for shared object code. See also **"Shared Object" on page 126**.

to **Download**

To retrieve data from a remote machine (or "the internet") to your local machine. Mechanisms for achieving this include FTP and the World Wide Web.

Driver

In the context of JDBC or ODBC, software that accesses a proprietary data source, providing a standardized view of the data to ODBC.

Driver Manager

Software whose main function is to load drivers. ODBC or JDBC applications request a DSN through the Driver Manager. The Driver Manager loads the driver specified in the DSN's configuration file. In Windows, the ODBC Data Source Administrator is used to set up the Driver Manager.

DSN

Data Source Name. A name associated with an ODBC data source. Driver Managers, such as unixODBC or the Microsoft Windows Driver Manager, use the Data Source Name to cross-reference configuration information and load the required driver.

Field

A placeholder for a single datum in a record. For example, you can have a Surname field in a Contact Details record. Called a *cell* in Microsoft Access.

Flags

Single-bit values, representing 'Yes' or 'No'. Also called "True" and "False".

Host

A computer visible on the network.

HTTP

HyperText Transfer Protocol. The means of transferring web pages.

HTTPAdmin

An NT user name, valid on the machine the server is running on. This is the only user allowed to amend settings, or display certain items, through the Web Administrator.

JDBC (Java DataBase Connectivity)

A Java *API* for database access based on the *ODBC* API, but includes an object-oriented interface to the underlying data source.

JDK (Java Development Kit)

A software development environment for writing Java applets and applications.

JRE (Java RunTime Environment)

A subset of the Java Development Kit (JDK) which supports the execution (but not the development) of Java applications and consists of the Java Virtual Machine (JVM), the core classes, and supporting files.

JVM (Java Virtual Machine)

Software that interprets and executes the byte codes in Java class files like a microprocessor would execute machine code. There are many virtual machines available from different vendors and for different purposes.

Middleware

Software that is placed between the *client* and the *server* to improve or expand functionality.

ODBC (Open DataBase Connectivity)

A standard *API* for connecting application programs to relational database systems through a suitable *driver*. ODBC is available on a wide number of platforms and the Easysoft JDBC-ODBC Bridge allows the database and the application to reside on different machines across the network.

Operating System

Academics still haven't agreed on the actual definition of an operating system, but a working definition can be:

A collection of software programs, APIs and working practices that control and integrate the execution of system functions on behalf of application programs.

Platform

The term *platform* normally covers the hardware and operating system as a unit. For example, a PC running Microsoft Windows, a PC running BSD Unix, and a Sun system running Solaris are three different platforms.

Server

A computer, or *host*, on the network, designed for power and robustness rather than user-friendliness and convenience. Servers typically run round-the-clock and carry central corporate data.

– OR –

A process performing the centralized component of some task, for example extracting information from a corporate database. See also **"Client/Server" on page 123**.

Shared Object

A piece of object code (i.e. a program fragment) that can be loaded and executed by other programs.

SQL (Structured Query Language)

A standard language for interacting with relational database systems, based on Relational Theory.

System data source

In the context of ODBC under Microsoft Windows, a data source which can be accessed by any user on a given system. See also **"User data source" on page 127**.

Table

A data set in a relational database, composed of rows and columns. For example:

software	
vendor	name
Easysoft	Easysoft JDBC-ODBC Bridge
Mysoft	My JDBC Application

This table has two columns; `vendor`, and `name`. It has two rows: one corresponding to Easysoft JDBC-ODBC Bridge and the other corresponding to MySoft's Java application. The term *table* can also apply to just the definition of the table, without its data.

User data source

An ODBC Data Source with access limited to a specific user on a given system. See also **"System data source" on page 127**.

Web Administrator

A web server run by the Easysoft JDBC-ODBC Bridge which can be used to display, amend and configure selected parameters.

E

F

G

H

I

T

U

W

X