

Chapter 4.

Configuring Application Properties

About This Chapter

This chapter describes how to configure ClickHR applications properties, including DBHandlers.

This chapter introduces the concept of DBHandlers, and how they are used to integrate ClickHR into the client's existing *Enterprise Resource Planning System (ERP)*. This chapter:

- ✓ defines DBHandlers.
- ✓ specifies skills required to create DBHandlers.
- ✓ provides the information needed to create DBHandlers.
- ✓ explains how the DBHandlers interact with the client's database and ClickHR.

What are DBHandlers?

DBHandlers are sets of *Java classes* that contain open calls to datasources. The language in those calls is often *SQL* (Standard Query Language). They are open to modification by the client and can be configured to perform such functions as:

- connecting ClickHR to the client's database.
- sending data to and retrieving data from the client's database.
- implementing *business logic* specific to the client.

DBHandlers are layered between the ClickHR software and the client's system. The ClickHR code is not open to the client and does not change,

but the DBHandler layer is open to modifications, so it is possible for a client to change the client's HR Database without having to make any modifications to the ClickHR software itself.

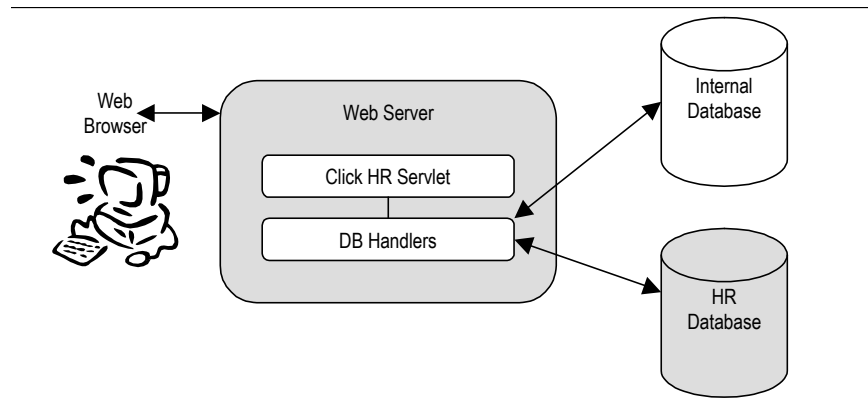
The implementation consultant configures DBHandlers after ClickHR is installed. But later, the code is owned by the client and can be modified as needed.

DBHandlers may use any Java-based open connectivity tool, including *JDBC* (Java Database Connectivity), to connect to the client's database. This allows clients to:

- ✓ access virtually any tabular data source using drivers available for JDBC, including files or spreadsheets.
- ✓ apply rules across multiple databases.
- ✓ invoke sophisticated transaction processing like two-phase commit functionality.

Functionality of DBHandlers

The function of the DBHandlers is to translate and transfer information back and forth between ClickHR and the Client's database. The figure below illustrates this concept. Descriptions of the databases represented in the diagram follow it.



Internal Database

The internal database maintains only that information required by ClickHR, including (for some installations) user login and password information, as well as *business rules*. *SQL* scripts that create the tables and fields required for the internal database are loaded to the Samples directory when the software is installed. The ClickHR implementation team or the client, as part of the configuration process, runs the *SQL* script that's appropriate for the client's database (sql.Oracle, sql.Sybase, sql.Informix, sql.DB2) against the database to automatically build all of the internal database tables that ClickHR requires to operate.

If there isn't an *SQL* script that's appropriate for the client's database, the team can try running the sql.Oracle script against the database, or it can create the required tables and fields manually.

HR Database

The HR Database is the client's database. It could have a variety of formats and data could reside in multiple databases. The developer must be familiar with the schema of the client's database management system in order to properly code the DBHandlers that connect to it.

DBHandler Features

Multithreading

DBHandler classes do not have to be concerned with multithreading issues. Their methods will only be called by one thread at a time. The constructor for DBHandler classes should take one parameter, a **com.iclick.servlet.ProcessorGroup**. That ProcessorGroup provides many useful, single threaded Services, including an SQL Connection that is often needed to access an SQL database.

Transaction Processing

By default, ClickHR uses JDBC to handle transaction processing, and that is used in the majority of the installations. However, ClickHR fully supports a *Transaction Processing Monitoring* environment.

Database Connectivity

The DBHandlers use the JDBC (Java Database Connectivity) API (Application Programming Interface) to establish a connection object. (However, other connection types can also be used.) This connection will register as a single user on the DBMS. ClickHR makes multiple connections to the DBMS – one for each thread or object set it establishes. A mid-sized installation will establish about 10 concurrent connections.

In order to establish a connection, the JDBC requires a JDBC driver. There are over 100 drivers available for JDBC at the web link below including drivers for Oracle, Sybase, Informix and SQL Server. The implementation team must determine what databases the DBHandlers will be connecting to and install the proper JDBC drivers.

If you can't find a driver for the particular database, check to see if the database you are trying to connect to is *ODBC* compliant (Open Database Connectivity). If the database is ODBC compliant, a JDBC connection object can be created through either a database server like IDS or DB Anywhere, or by using the JDBC/ODBC bridge.

Downloading JDBC drivers

To obtain the JDBC drivers, go to the website indicated below:

<http://industry.java.sun.com/products/jdbc/drivers>

Portability of DBHandlers

Because the DBHandlers use JDBC drivers for transaction processing, the DBHandler code is portable and reusable. So regardless of which

database the DBHandler is communicating with, the Java code remains the same.

Additionally, this functionality is available across multiple connections. For example, lets say you're writing payroll information to two places and it is important that it gets written successfully in both or the transaction gets completely rolled back. Through our DBHandler and the JDBC this is entirely possible and quite common.

DBHandler

The process Data Transfer and Translation Process of data translation by the DBHandlers can be summarized as follows:

1. The user inputs data via the web browser and it is sent to the web server via the Internet or Intranet.
2. The web server must be running the ClickHR application under Java 1.1. The ClickHR application processes the data from the browser and the output goes to one or more of the DBHandlers that process that particular type of data.
3. The DBHandler translates the output to conform to the format requirements of the client's database and writes the output to the HR database. This is why DBHandler connections must have the proper permissions to write to the client's database.
4. The DBHandlers also read information from and write information to the internal database.

Like all *Java interfaces*, the DBHandler interfaces publish a contract or a list of methods. The implementer will have to satisfy the contract by returning the expected data from all the methods published by a DBHandler by coding and implementing classes.

Methods of Communicating With The Database

The following are the most common ways of communicating with the database.

- ✓ Embedded SQL (Standard Query Language script) that talks to a remote database via the JDBC (Java Database Connectivity).
- ✓ Accessing a remote stored procedure on the DBMS (Database Management System) via the JDBC.
- ✓ Accessing a Name Directory like *NDS* (Network Directory Service) or *LDAP* (Lightweight Directory Service) via the *JNDI* (Java Naming and Directory Interface).
- ✓ Calling the ERP's proprietary communications layer, such as the *JMAC* in PeopleSoft or the *BAPI* in SAP.
- ✓ Relinquishing control of the transaction to Transaction Processing Monitoring software (using the TP monitor's Java interface).

Modifying The DBHandlers

Skills Required to Configure DBHandlers

DBHandlers are Java classes that also contain embedded *SQL* (Standard Query Language) script. Therefore, a specific skill set is required in order to be able to create and modify them. This skill set includes:

- Java language and application development skills.
- SQL (Standard Query Language).
- *JDBC* (Java Database Connectivity).
- familiarity with the ClickHR product.

Information Needed to Configure DBHandlers

Early in the ClickHR implementation project, the project team meets with the client to establish the specific configuration required for the installation. The client communicates business rules such as whether they want authentication to be handled by the ClickHR software or their own system. The project team then compiles the information and communicates it to the developers, so it can be translated into the business logic incorporated into the DBHandlers.

Configuration of DBHandlers requires knowledge of ClickHR.

To configure or code DBHandlers, the implementation team needs the following information:

- ✓ Specifications of the business rules that need to be applied to the data before writing to the client's database.
- ✓ Documentation of the schema of the client's database including:
 - type of database.
 - tables and names of fields.
 - location of databases, url's, paths.
 - hardware where the database is located.
 - username and password to access the databases.

Default DBHandlers

The standard set of DBHandlers that comes packaged in the ClickHR installation file is configured to communicate with the default database for a typical PeopleSoft installation. If the client's HR database is a PeopleSoft database that has not been modified, ClickHR can be tested for connectivity immediately after installation.

In most cases, however, the client's database will have undergone modifications, and may not be in the standard format. This will require

that some of the DBHandlers be modified or rewritten to handle the differences in database format.

Upgrading the ERP

If the client changes the Enterprise Resource Planning system, and reconfigures the HR database, the DBHandlers might need to be modified to reflect these changes. However, there is no need to modify the ClickHR application.

Determining Which DBHandlers Should be Implemented

Only certain DBHandlers need to be implemented. Some of them can be used out of the box in conjunction with the custom DBHandlers.

After the project team determines client requirements, the business rules and other information are translated into business logic that is programmed into the DBHandlers. The implementation developers must be conversant in the structure and functionality of the client's database and understand what requirements need to be satisfied by the DBHandlers.

Depending on the particular requirement, the developer must determine which DBHandler manages the particular piece of data that needs to be manipulated, and identify the DBHandler Interfaces that process the required information.

Java Documentation comes with the ClickHR software. Refer to "Chapter 5. Reference Documents" for more information about the "Java docs". DBHandler names are descriptive of their function and the Java docs provide details on the methods implemented by the DBHandler interfaces.

By analyzing the client's database and using the Java docs as reference material, the developer will determine which DBHandlers need to be implemented.

DBHandler Interfaces

DBHandler interfaces are the focal point of the configuration of the ClickHR application for a particular customer. Any aspect of the ClickHR application that may need to be customized for a customer's implementation is captured in a DBHandler interface.

In Java terms, an interface is a contract in the form of a collection of methods and constant declarations. When a class implements an interface, it promises to implement all of the methods declared in that interface.

To customize the ClickHR application, you need to implement the DBHandler Methods that satisfy the requirements of the DBHandler Interface. Then, you need to set the properties in the **iclick.props** file as described below.

Where are DBHandler Interfaces Found?

ClickHR Java classes are grouped into packages (corresponding to directories). These are listed in the Java documentation provided with ClickHR. Refer to the “Reference Documents” chapter for more information about the “Java docs”.

The best way to find the complete list of interfaces is to click on Index on the right frame of the Java docs main screen. Then scroll down to DBHandlers. Click on the DBHandlers link. You will see the entire list of interfaces. Click on each interface for details.

For the most part, each DBHandler interface comes with a standard class implementation, which is the default value of the property for that DBHandler. For some DBHandler interfaces, there are a number of class implementations already written, one of which may be appropriate for your environment.

Below is an illustration of the screen that lists all of the DBHandler interfaces.

The screenshot shows a web browser window displaying the Java documentation for the 'Interface DBHandler'. The left sidebar contains a navigation menu with the following items:

- IClick 4 Platform
- All Classes
- Packages
 - com.iclick.business
 - com.iclick.business.d
- All Classes
 - AbsEAHandleDisap
 - AbsLOAHandleDisa
 - AbsSepHandleDisap
 - AbstPrefabAdminCo
 - AbstractArchiveDBE
 - AbstractArchiveDB
 - AbstractArchiveDB
 - AbstractEAActivity
 - AbstractEAArchive
 - AbstractICanDBHan
 - AbstractLOAAActivity
 - AbstractLOANotify
 - AbstractLOAParAct
 - AbstractPAAActivity
 - AbstractPABuilder
 - AbstractPAMultipleC

The main content area is titled 'Interface DBHandler' and contains the following sections:

All Known Subinterfaces:

[AllPersonNameDBHandler](#), [AuthDBHandler](#), [BenefitsDBHandler](#), [BlobDBHandler](#), [BusinessRuleDBHandler](#), [CertificateDBHandler](#), [ClickHRDBHandler](#), [CompanyDBHandler](#), [CompensationInfoDBHandler](#), [ConfigDBHandler](#), [ConfRoomDBHandler](#), [ContactsDBHandler](#), [ContentDBHandler](#), [CostCenterDBHandler](#), [CPGroupDBHandler](#), [DepartmentDBHandler](#), [DependentsDBHandler](#), [DivisionDBHandler](#), [EducationDBHandler](#), [EidValidDBHandler](#), [EmailDBHandler](#), [EmployeeDBHandler](#), [ExtAuthDBHandler](#), [FiscalYearDBHandler](#), [GenderDBHandler](#), [GlobalDBHandler](#), [HireDateDBHandler](#), [HolidayDBHandler](#), [HomeDBHandler](#), [ICanDBHandler](#), [JobDescDBHandler](#), [JobInfoDBHandler](#), [LanguageDBHandler](#), [LicenseDBHandler](#), [LOAArchiveDBHandler](#), [LOAChangesDBHandler](#), [LocaleDBHandler](#), [LockDBHandler](#), [MapDBHandler](#), [OCNodeDBHandler](#), [PageViewCriteriaDBHandler](#), [PageViewDBHandler](#), [PATemplateAdminDBHandler](#), [PayCheckDBHandler](#), [PerfDBHandler](#), [PersonNameDBHandler](#), [PositionDBHandler](#), [PreferredNameDBHandler](#), [PRScheduleDBHandler](#), [ReviewDBHandler](#), [SeparateArchiveDBHandler](#), [SeparateChangesDBHandler](#), [SessionDBHandler](#), [SkillDBHandler](#), [SSNumberDBHandler](#), [StatusDBHandler](#), [TaskDBHandler](#), [TaxDBHandler](#), [TimeOffDBHandler](#), [TitleDBHandler](#), [TransferArchiveDBHandler](#), [TransferChangesDBHandler](#), [URLScriptDBHandler](#), [UserDBHandler](#), [UserGroupPrivDBHandler](#), [UserPrivDBHandler](#), [WFDBHandler](#), [WorkDBHandler](#)

All Known Implementing Classes:

[SerializableDBHandler](#)

Implementing DBHandlers by Setting Application Properties

Properties for DBHandlers are set in the ClickHR properties file, called iclick.props. ClickHR will look in the directory to which the iclick.dir parameter is set to find the iclick.props file (refer to “Step 2 - Receipt and Initial Installation of Software”, on page 11, for more information on creating the iclick.props file and setting the iclick.dir servlet parameter.) The sample.props file, located in the docs/samples directory, can be used

as a starting point to create this file. It contains the entire list of DBHandler settings with notations.

The DBHandler interface has a NAME, which is the property that must be set to plug-in a class for that DBHandler interface.

Refer to “ Sample iclick.props Properties File,” page 32, for an example of how to plug-in DBHandlers by setting properties.

For example, to plug-in the class:

com.yy.XXDBHandler

which implements the:

XXDBHandler interface NAMED xx.business.dbhandler,

you would set the following property:

xx.business.dbhandler=com.yy.XXDBHandler

Naming Convention for DBHandlers

DBHandler interface names are dot separated, indicating a hierarchy of property settings. A property set for a higher level in the hierarchy applies to all DBHandlers below it. For example:

business.dbhandler=com.yy.XXBusinessDBHandler

implies that the class com.yy.XXBusinessDBHandler implements all DBHandler interfaces whose NAME ends in business.dbhandler, including xx.business.dbhandler among others. In this way, one property setting can plug-in the same class for many DBHandler interfaces.

Steps to Configuring DBHandlers

Refer to “Chapter 2. Installation and Implementation,” page 1 for information on how to install ClickHR.

By now, the implementation team will have installed the software and has been able to access a login page.

Step 1 – Set CLASSPATH to location of user-defined DBHandlers.

Set your web server’s CLASSPATH environmental variable with the directory location of the new customized DBHandler classes that will be written. You can pick your own directory name. You should already have set the paths to the ClickHR Java classes(Refer to Chapter 2. for additional information on setting the classpath for ClickHR Java classes.)

Step 2 - Set the properties for the DBHandlers in the iclick.props file.

Type the name of the appropriate DBHandler interface (see DBHandler Interfaces above) followed by “=” and the name of the new DBHandler class you have written. For example:

business.dbhandler=com.yy.XXBusinessDBHandler

Step 3 – Determine the authentication method and rewrite DBHandlers as required.