JNDI Adapter Guide

Windchill 7.0

December 2003

Licensed Third-Party Technology Information

Parametric Technology Corporation, 140 Kendrick Street, Needham, MA 02494 USA

Contents

About This Guide

This *JNDI Adapter Guide* documents the use of the Info*Engine JNDI adapter software. It contains the following chapters:

- *About This Guide* is the section you are currently reading. It details the helpful documentation associated with this adapter, the contact information for technical support and documentation comments, as well as the documentation conventions specific to this manual.
- *Architectural Overview* describes the general Info*Engine architecture.
- *Installing and Configuring the Adapter* describes how to install and configure the Info*Engine JNDI adapter.
- *The JNDI Webject Library* details each of the webjects available for use with the adapter.

This guide assumes you are familiar with the basics of HTML, XML, and JSP as defined by the World Wide Web Consortium (http://www.w3c.org).

To take advantage of the advanced functionality of Info*Engine, you must have expert knowledge of HTML, XML, and JSP.

Related Documentation

The following Info*Engine documents may be helpful to you:

- The *Info*Engine Installation and Configuration Guide* details the procedure for installing and configuring the Info*Engine Server.
- The *Info*Engine User's Guide* details the main functionality of the Info*Engine Server.
- The *Info*Engine Java Adapter Development Kit Programming Reference* describes how to develop your own native Info*Engine adapters using the Java programming language if the PTC adapter library does not contain an adapter that suits your needs.
- The *Info*Engine C Adapter Development Kit Programming Reference* describes how to develop your own non-native Info*Engine adapters using the C programming language if the PTC adapter library does not contain an adapter that suits your needs.
- The *Info*Engine Read This First* document provides additional information about Info*Engine (including the adapter) that is not in the Info*Engine guides.

Technical Support

Contact PTC Technical Support via the PTC Web site, phone, fax, or email if you encounter problems using this adapter.

For complete details, refer to Contacting Technical Support in the *PTC Customer Service Guide* enclosed with your shipment. This guide can also be found under the Support Bulletins section of the PTC Web site at:

```
http://www.ptc.com/support/index.htm
```

The PTC Web site also provides a search facility that allows you to locate Technical Support technical documentation of particular interest. To access this page, use the following link:

```
http://www.ptc.com/support/index.htm
```

You must have a Configuration ID before you can receive technical support. If you do not have an ID, contact PTC License Management using the instructions found in your *PTC Customer Service Guide* under Contacting License Management.

Documentation for PTC Products

PTC provides documentation in the following forms:

- Help topics
- HTML books
- PDF books

All books are available in HTML and PDF formats, or both, on the Info*Engine CDs. To view and print PDF books, you must have the Adobe Acrobat Reader installed.

All Windchill documentation is included on the CD for the application. In addition, books updated after release (for example, to support a hardware platform certification) are available from the Reference Documents section of the PTC Web site, at the following URL:

`http://www.ptc.com/cs/doc/reference`

Comments

PTC welcomes your suggestions and comments on its documentation: send comments to the following address:

documentation@ptc.com

Please include the name of the application and its release number with your comments. For online books, provide the book title.

Documentation Conventions

Info*Engine documentation uses the following conventions:

Convention	Item	Example
Bold	Names of elements in the user interface such as buttons, menu paths, and dialog box titles. Required elements and keywords or characters in syntax formats. The beginning of cautionary statements and similar notes to the user.	Click **OK**. Select **File > Save**. **License File** dialog box **create**_*tablename*.**sql** **CAUTION:** When prompted…
Italic	Variable and user-defined elements in syntax formats..	**create**_*tablename*.**sql**

Monospace	Examples
	
	Messages
		`http://localhost/Windchill/…`
		
		`Processing completed.`

"Quotation marks"	Strings	The string "UsrSCM" . . .
	The CAUTION symbol indicates potentially unsafe situations which may result in minor injury, machine damage or downtime, or corruption or loss of software or data.	When you add a value to an enumerated type (for example, by adding a role in the RolesRB.java resource file), removing that value can result in a serious runtime error. Do not remove a role unless you are certain there is no reference to it within the system.

1

Architectural Overview

This chapter provides an overview of the Info*Engine architecture.

Topic **Page**

Identifying the Info*Engine Components

The following components make up the Info*Engine architecture:

- The ***Info*Engine servlet*** provides an interface between the Web server and Info*Engine.
- The ***Info*Engine server*** provides a mechanism for retrieving and manipulating the data that users or custom applications want to view or receive.
- The ***Naming Service*** is the software that supports the operation of Info*Engine components. In the Info*Engine Naming Service, you can identify the LDAP directory servers where entries for the network addresses of Info*Engine components and entries for configuration properties reside.
- The ***Info*Engine Service Access Kit*** *(SAK)* is an application program interface (API) that facilitates the development of Java applications, including JSP pages, that directly utilize the functions and features of Info*Engine. For example, high-level Info*Engine components such as the Info*Engine servlet, the Info*Engine server, and the E-Mail Broker use the SAK to invoke tasks and individual webjects.
- The ***native adapters*** provide a direct interface between Info*Engine and information systems.
- The ***non-native adapters*** provide an indirect interface between Info*Engine and information systems. These adapters use a different protocol from the protocol used by Info*Engine and therefore cannot connect directly to Info*Engine.
- ***Gateways*** provide an interface between Info*Engine and non-native adapters.
- The ***Info*Engine SOAP RPC servlet*** catches and processes Info*Engine SOAP requests that are made over the Web. SOAP (Simple Object Access Protocol) is a lightweight protocol that can be used by non-Java applications. By using this protocol, non-Java applications can send requests to execute Info*Engine code and return the output that is generated.
- The ***E-Mail Broker*** provides a process by which users can e-mail Info*Engine requests to a mailbox. Using the SAK, the messages in the mailbox are then passed on to the Info*Engine server for processing.

The remainder of the chapter describes the relationships among the components.

Identifying Basic Configurations

Info*Engine components can be used in many different software and hardware configurations to meet your business requirements for accessing, managing, and presenting data from many different information systems.

Setting up your Info*Engine environment can be accomplished by:

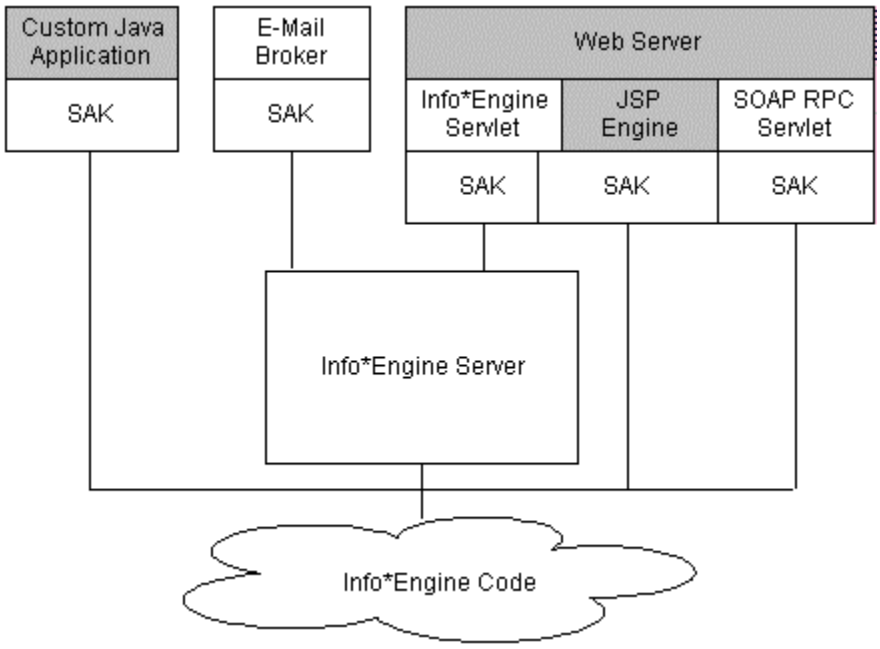
- Establishing interactions with Info*Engine.
- Managing the execution of Info*Engine tasks.
- Starting and managing Info*Engine components.
- Managing connections to the information systems where the data of interest resides.

Interacting with Info*Engine

Initiating an interaction with Info*Engine can be accomplished by using one or more of the following:

- Custom Java applications, including JavaServer Pages (JSP).
- Web Servers that process Info*Engine requests. The requests can come from applications, Web browsers, or wireless devices such as cell phones and personal digital assistants (PDAs).
- E-mail requests that contain formatted messages sent to a predefined Info*Engine mailbox.
- Java Message Service (JMS) events and messages that queue Info*Engine tasks for execution.
- Custom non-Java applications that make requests to execute Info*Engine tasks. These applications can use the Info*Engine SOAP RPC servlet.

The following diagram shows how the Info*Engine components and other customer software components can interact to execute Info*Engine code.



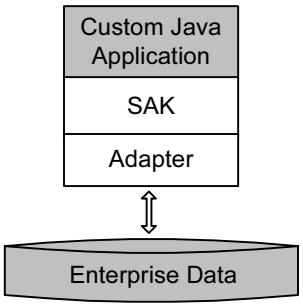
Info*Engine code consists of Java classes that are accessed through the Info*Engine API. The API is available through the SAK and externalizes predefined functions called webjects and tasks. The webjects and tasks can be easily instantiated and invoked as Java objects from a Java application or in a text file. Info*Engine text files can be accessed using requests or code within an application.

The following sections provide more details about how to use the Info*Engine components with your software.

Using a Custom Java Application

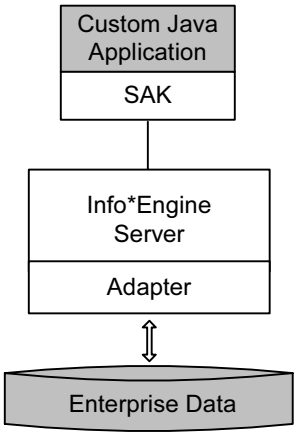
By coding a custom application in Java, you can have quick and easy access to Info*Engine without the added complexity of a Web server. By using the API defined in the SAK, you can execute Info*Engine webjects, tasks, and other Info*Engine code in the Java Virtual Machine (JVM) where the application resides.

The following diagram shows the SAK and adapter classes being used in the application to access data in a remote database.



Within a Java application, you also have the flexibility of executing Info*Engine tasks that are maintained outside of the application. An Info*Engine task consists of a set of webjects and surrounding code that supports the processing of the webjects. These tasks can then be processed either in the JVM of any Info*Engine server or in the JVM of the application.

The following diagram shows the Info*Engine components that are used when an application executes a task in an Info*Engine Server. In this case the application requests that a task be executed in the server that accesses data in a remote database.



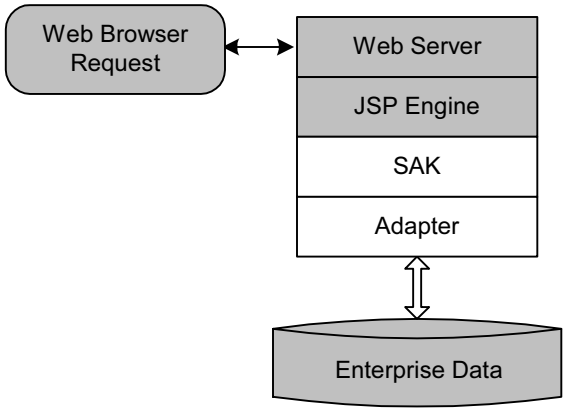
Using a Web Server to Process Info*Engine Requests

The installation process steps you through a procedure that deploys Info*Engine as a Web application. Going through the installation process sets up your Web server and its servlet engine to identify Info*Engine requests and pass those requests on to Info*Engine components for processing. After the installation is complete, your Info*Engine environment is set up so that Info*Engine requests to execute JSP and HTML pages coming from Web browsers are processed correctly.

By doing some additional Info*Engine configuration steps, you can set up your Info*Engine environment to process requests from the following additional sources:

- If you configure Info*Engine to identify wireless communication protocols, requests can come from wireless communication devices such as a cell phone.
- If you implement the Info*Engine SOAP RPC servlet, Info*Engine SOAP requests can come from non-Java applications.

The following diagram shows the relationships among the components that process Web browser requests for JSP pages.



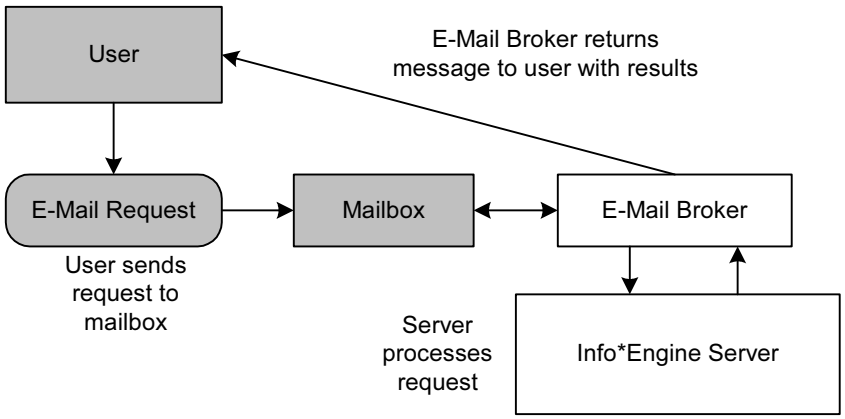
This diagram shows the components that are used when the request specifies that Info*Engine execute a JSP page. By default, Info*Engine and Web server configuration specifies that JSP pages are processed in the JSP engine of the servlet engine installed on your Web server. The JSP engine creates an instance of the SAK, which is then used to execute the Info*Engine-specific code on the page. For example, if a user clicks a link or uses a URL in a browser window that serves as a JSP request for information from Info*Engine, the JSP engine and the SAK work together to manage the request.

The SAK processes the request and, as needed, connects to specialized Info*Engine adapters that communicate with external applications such as Oracle databases, PDM systems, various legacy systems, and ERP systems. After the requested information is obtained from the external applications, the process reverses itself and ultimately displays information in the user's browser window.

Making E-Mail Requests to Info*Engine

The E-Mail Broker allows users to make Info*Engine requests by e-mail.

The E-Mail Broker provides a process that monitors a mailbox for requests to execute Info*Engine templates and tasks. When a request arrives in the mailbox, the E-Mail Broker connects to the Server and passes the request to the Info*Engine Server for processing. It also captures output from the processed template or task, and returns the output in an e-mail message to the address specified in the From or Reply-To heading of the original request.



Managing the Execution of Info*Engine Tasks

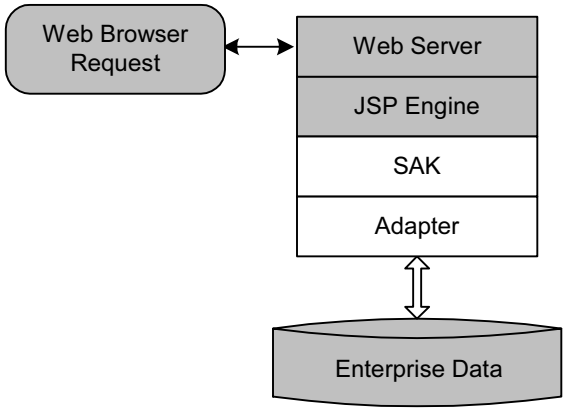
Info*Engine tasks control the retrieval and manipulation of data. Tasks consist of the following:

- Info*Engine webjects that retrieve and manipulate data.
- Surrounding Info*Engine custom tags that manage the execution of the webjects.

There are two basic ways to execute tasks:

- Incorporate tasks directly into any Java application, including JSP pages, using Info*Engine custom tags.
- Put the tasks in individual text-based documents, specify which tasks to execute in the Info*Engine custom tags within a Java application (or JSP page).

The decisions about how and where to execute Info*Engine tasks depend on your system requirements. For example, if you have a dedicated environment where one system contains both your Info*Engine application and all of the required software components, and the tasks to execute do not require any complex processing, you may choose to execute your tasks from within JSP pages that are also used to display the results. In this case, the environment used could be similar to the following:



The JSP engine depicted in the diagram instantiates an instance of the SAK within the JVM of the JSP engine. The SAK is then used to process the Info*Engine custom tags. Some of the Info*Engine tags can execute webjects that extract data from enterprise systems through an adapter, while others can display the data. In this example, all of the webjects are contained in the same JSP page.

In a more complex environment where you have a large Java application that executes complex tasks, you can manage the tasks more efficiently by separating them into individual documents, rather than coding them directly into the application. When a task is contained in its own document, it is called a standalone task. For a standalone task, the following processing options are available:

- You can specify where you want a standalone task to execute, whether it is in the same JVM as the application or in the JVM of any Info*Engine Server that is part of your environment.
- You can specify how you want to execute standalone tasks that do not execute in the same JVM as the application. There are three ways to execute these standalone tasks:

–Requesting, through a TCP/IP connection, that the task executes in a specific Info*Engine Server. Each Info*Engine Server listens for task requests and executes them upon arrival.

–Implementing a specific event that executes tasks. Establishing events through an Info*Engine Web Event Service allows you to execute tasks based on specific actions that can occur in your environment.

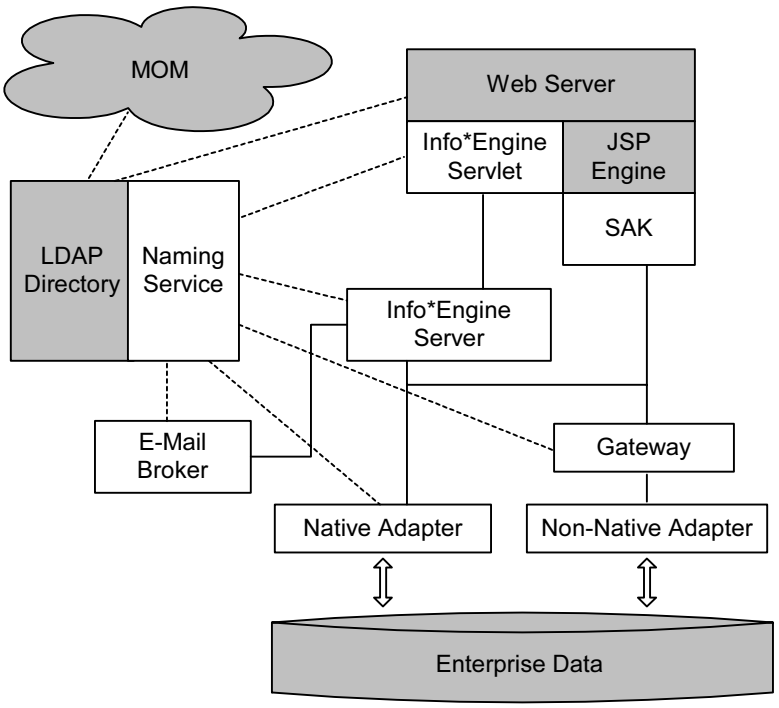
–Queuing a task for execution. After you queue a task, you can disconnect from your application. Any results are queued for later retrieval either by you or others. By queuing a task, you can also guarantee that the task will be completed, even if it is interrupted due to a system problem.

By performing the basic Info*Engine installation, the Info*Engine server is set up to receive task requests. To use either queues or events for executing tasks, you must install and configure additional Message-Oriented Middleware (MOM) software and then update your Info*Engine configuration.

Starting and Locating Info*Engine Components

The Naming Service uses an LDAP directory to provide the Info*Engine servlet, the Info*Engine server, the native adapters, and the Info*Engine gateways with a means of locating each other, acting as a traffic director of sorts.

In the following diagram, dashed lines represent the communication between the Naming Service, Info*Engine components, and third party software that could be installed.



Additionally, there is an Info*Engine SOAP RPC servlet entry in the Naming Service.

The Naming Service can be used to automatically start Info*Engine components residing on the same hardware system. By default, the Naming Service is set up during the installation to start the Info*Engine Server and the E-Mail Broker. Depending on where you install adapters and gateways, you may want to configure the Naming Service to start them as well.

Setting Up Connections Through Adapters

Adapters provide a connection between the Info*Engine server and information systems. One side of the adapter communicates with the Info*Engine server and the other side communicates with the information system. The adapter translates Info*Engine server requests into information system requests.

Info*Engine provides two types of adapters:

- **Native adapters** are implemented in the Java language and conform to the formal Info*Engine interface specification. For example, the JNDI and JDBC adapters are native adapters.
- N**on-native adapters** are implemented in a non-Java language or do not conform to the formal Info*Engine interface specification. Because the implementation is different from Info*Engine, you must also define a gateway for each non-native adapter you install. Gateways translate Info*Engine requests so that the adapters can process them. After an adapter receives a request, the adapter sends it to the associated database or data repository. The adapter also returns any information obtained from the data repository to the gateway where it is translated and passed back to the Info*Engine server. For example, the Metaphase and Oracle adapters are non-native adapters.

The adapters you must use are determined by the information systems from which you want to retrieve information. Info*Engine provides a unique adapter for each information system. For example, to retrieve information from a Metaphase database, you must install and configure the Metaphase adapter and, because this adapter is a non-native adapter, you must also configure a gateway for the adapter.

Native adapters can be installed as follows:

- Residing in the same Java Virtual Machine as the Info*Engine webject that accesses the adapter (known as the in-process adapter).
- Distributed in their own Java Virtual Machine on the same hardware system or on remote hardware systems (known as out-of-process adapters).

How to install native adapters is determined by your site.

Gateways usually reside in the same Java Virtual Machine as the calling webject since the code for gateways is installed as part of Info*Engine.

Non-native adapters are always distributed in their own environment and are run as out-of-process adapters.

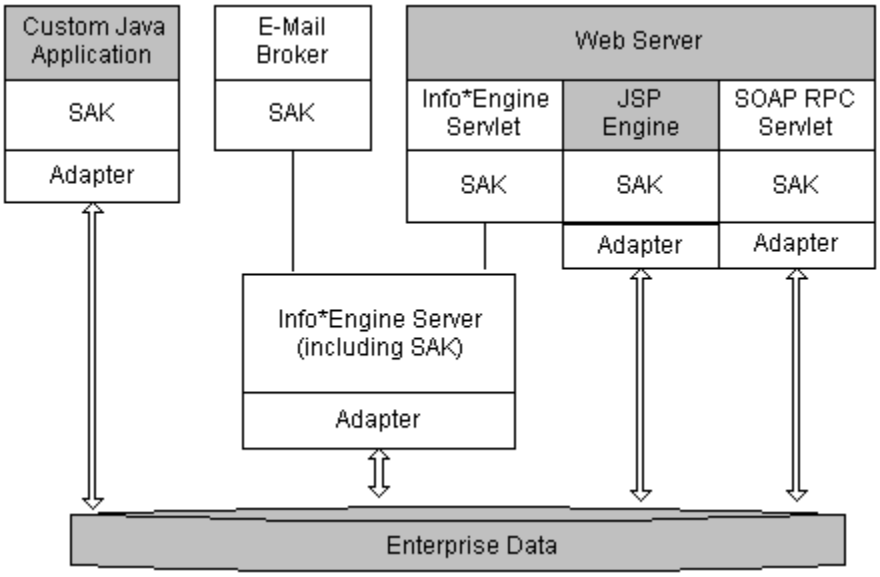
The following sections expand on the installation options.

Using In-Process Adapters and Gateways

In-process adapters and gateways are installed and run in the same Java Virtual Machine as the calling webject. Only native adapters and gateways can be configured to run in the same JVM as the calling webject. The SAK determines which classes are required when processing webjects for an in-process adapter or gateway, and instantiates the classes in the JVM. Therefore, the communication between the webject and the adapter or gateway is very efficient.

Configuring in-process adapters and gateways minimizes communication delays and resource usage; however, the total resource usage of the machine hosting the Info*Engine code may be increased because of the additional load burden of running the adapter or gateway.

When an adapter is configured to be an in-process adapter, the adapter classes can be instantiated by any SAK that executes adapter webjects. The following diagram shows adapter classes residing in the JVM of a custom Java application, the Web server, and the Info*Engine server:



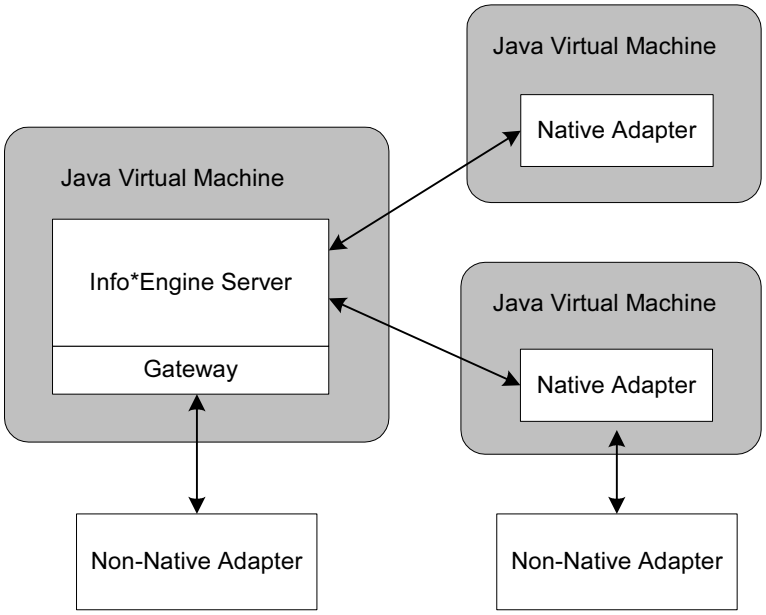
As shown in the diagram, no external communication is needed between the SAK and the adapter when the adapter is in the same process.

Running in-process native adapters and gateways is generally the preferred configuration if the resource usage on a single system is not excessive.

Using Out-of-Process Adapters and Gateways

Distributing adapters across multiple hardware systems reduces the overall resource usage on the machine hosting the Info*Engine code; however, it does introduce some delay and resource usage associated with using a TCP/IP connection for communicating between Info*Engine components and each adapter.

The following diagram shows the communication lines that are used when three adapters and one gateway are distributed.



Distributed native adapters and gateways are installed and run in their own Java Virtual Machine. These virtual machines can be on the same hardware system as the Info*Engine Server or on a different hardware system. Non-native adapters can only be configured as out-of-process adapters, and they always run as separate processes. Although gateways for non-native adapters are typically configured as in-process gateways to minimize the communication delays, they do not need to be in the same process.

The deployment of distributed adapters at your site may be determined by a company policy that requires the adapter to be located near the application it accesses, or it may be based on administrative reasons. One reason for running a native adapter in its own Java Virtual Machine could be to better manage the resource usage of the virtual machine.

2

Installing and Configuring the Adapter

This chapter describes all of the installation and configuration procedures for the JNDI Adapter.

Topic **Page**

About the Adapter

The Java Naming and Directory Interface (JNDI) adapter allows the user to connect to the various naming and directory interfaces accessible using the JNDI system.

The JNDI Service Provider Interface (SPI) provides the means by which naming and directory services are integrated into the JNDI framework. To connect to a directory, the JNDI adapter requires the appropriate JNDI Service Providers.

Using action and query webjects, the adapter communicates with both Info*Engine and the JNDI system, which controls the directory. The adapter makes the transition effectively seamless.

Installation

The JNDI adapter is installed automatically when the Info*Engine components are installed. A standard installation for these components is in <ie_dir>/codebase/WEB-INF/lib/ie.jar file, where <ie_dir> is the Info*Engine installation directory.

Note: The JNDI adapter classes are automatically used by Info*Engine to access the Info*Engine component LDAP entries stored in the Aphelion Directory. No additional configuration is necessary.

To use the JNDI adapter to access other information stored in the Aphelion Directory or stored in other naming or directory systems, create an LDAP entry for the JNDI adapter that provides access to that system.

The JNDI adapter uses the following .jar files:

ie.jar
servlet.jar
ldap.jar
jndi.jar
providerutil.jar
CommonCoreMeta.jar

Configuration

To configure the adapter involves the following steps:

- setting an LDAP entry using the Property Administrator
- setting the appropriate properties for in-process or out-of-process use
- modifying a start file for out-of-process use

Configuring the Adapter Through the Property Administrator

Use the Info*Engine Property Administrator to add or modify existing Info*Engine adapter service LDAP entries. Adapter properties are maintained as attributes in Info*Engine adapter LDAP entries.

For general information about using the Info*Engine Property Administrator, see the *Info*Engine Installation and Configuration Guide* and the online help provided in the administrator.

To create a new adapter service LDAP entry, select the adapter form from the **Create Entry** list on the Info*Engine Property Administrator main page, and complete the form.

All forms include the following set of common fields that are located at the top of the form:

Service Name
Distinguished Name
Runtime Service Name
Service Class
Host
Port
Serialization Type

When the form displays, Property Administrator populates the **Service Name**, **Distinguished Name**, and **Runtime Service Name** fields with suggested names. These names are based on information provided when you logged in to the administrator and also information that is stored in the form. You can change these names to match the criteria set up for your site LDAP entries. For additional information about these fields, view the online help for your form or talk to your Info*Engine administrator. Additional information about setting up the criteria your site should use for creating LDAP entries can be found in the *Info*Engine Installation and Configuration Guide*.

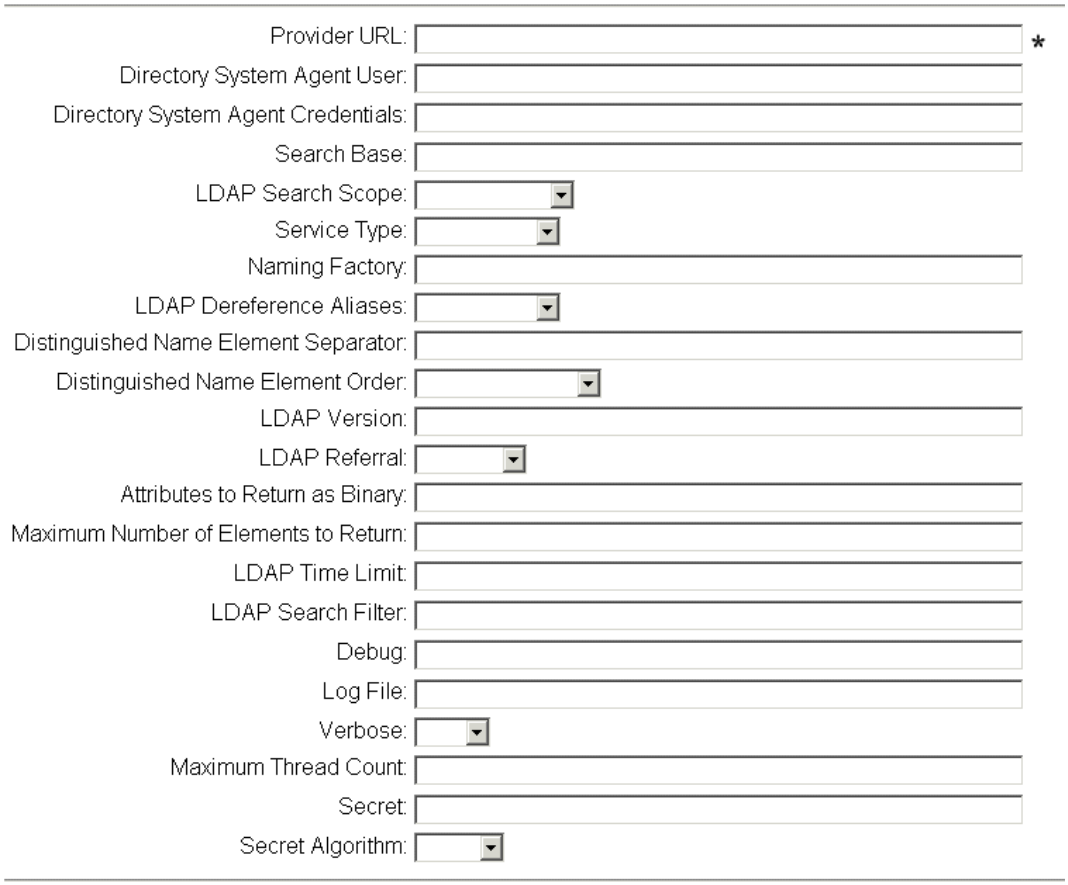
Service Class contains the service class name used for the adapter. If you are using the adapter as an in-process adapter, leave the default name. If the adapter will only be used out of process, you must delete the name in the **Service Class** field and add the host and port used to access the adapter in the **Host** and **Port** fields.

The **Serialization Type** field allows you to change the type of data serialization Info*Engine uses when passing data to an out-of-process JNDI adapter. By default, Info*Engine components use Java serialization when passing data between components. Java serialization preserves data type information so that the data can be easily manipulated from within an Info*Engine custom application, Java Server Page, or task. To pass only XML, you would change the type to **XML**.

For help on **Logging, Additional Properties, Co-resident Services and Additional Services** options click the **Help** link on the main Property Administrator page.

To create a new JNDI adapter entry:

1. Select **JNDI Adapter**. A form that includes the following displays:



2. Enter values on the form. Required fields on the form are indicated with an asterisk (*). Click the field heading to display information about the JNDI properties on the form. For additional information see the section titled JNDI Adapter Properties later in this chapter.

3. Click **Create Adapter** to complete creation of the JNDI adapter entry.

Using the Adapter In Process

If you are using the adapter as an in-process adapter, leave the default service class name in the **Service Class** field.

Using the Adapter Out of Process

If you are using the adapter only out of process, you must delete the name in the **Service Class** field and add the host and port used to access the adapter in the **Host** and **Port** fields.

Creating Your Own Start File

It is only necessary to create a start file if you are going to be using the adapter out of process.

Installed JNDI Start Files

PTC.Setup provides the following JNDI start files:

startjndi.bat
startjndi.sh

These files can be used for starting the JNDI adapter as an out-of-process adapter. The files are located in the bin/infoengine directory where Info*Engine is installed. Each of the start files executes another file named runieservice to actually start the service.

The start files use the ie.properties file that PTC.Setup generates as the value for the DpropFile parameter on the Java start command.

Modifying the Installed Start File

On a Windows system, PTC.Setup generates a file similar to the following to start the JNDI adapter:

```
set JAVA_HOME=<javaHome>
set WEB_APP_HOME=<webAppHome>

set IEPROPFILE=<propUrl>
set IEMYNAME=<domain>.jndiAdapter-oop
set IENAMINGSERVICENAME=<domain>.namingService
set IE_OPTS=-Xms64m -Xmx64m "-DpropFile=%IEPROPFILE%"
                            "-DmyName=%IEMYNAME%"
                            "-DnamingServiceName=%IENAMINGSERVICENAME%"

%JAVA_HOME%\bin\java -cp %WEB_APP_HOME%/codebase/WEB-INF/lib/ie.jar
                  "-Dcom.infoengine.javaCmd=%JAVA_HOME%/bin/java"
                  "-Dcom.infoengine.javaOptions=%IE_OPTS%"
                  "-Dcom.infoengine.webAppHome=%WEB_APP_HOME%" com.infoengine.Loader
                                                  com.infoengine.jndi.JNDIAdapter
```

Replace jndiAdapter-oop with the service class name of the JNDI LDAP entry you created.

In the generated file, all of the variables contain actual values that were determined by PTC.Setup during the installation:

<javaHome>	Specifies the location of the installed Java SDK.
<webAppHome>	Specifies the location of the Info*Engine installation directory.
<propUrl>	Specifies the location of the ie.properties file, which is described in the next section.
<domain>	Specifies the host domain where Info*Engine is installed.

For more information on modifying start files see the *Info*Engine Installation and Configuration Guide*.

ie.properties Location and Contents

The ie.properties file is located in the codebase/WEB-INF directory where Info*Engine is installed and contains a reference to the LDAP branch that contains the Info*Engine properties. This reference also contains validation information to ensure that the services can access the LDAP directory.

PTC.Setup generates the contents of the ie.properties file based on the values that were entered when Info*Engine was installed. For more information about this file, see the *Info*Engine Installation and Configuration Guide*.

JNDI Adapter Properties

binaryAttributes

Specifies which attributes have non-string syntax. Attributes with non-string syntax are stored as byte arrays (byte[]). Specify the value of this property as a string of comma separated attribute names. All attributes to store as binary should be included (ie: userPassword).

If you do not specify this property, all attributes are assumed to have string syntax with the expection of "userPassword". Examples of attributes that might be stored as binary in the LDAP are: photo, personalSignature, audio, jpegPhoto, javaSerializedData, thumbnailPhoto, thumbnailLogo, userPassword, userCertificate, cACertificate, authorityRevocationList, certificateRevocationList, crossCertificatePair, x500UniqueIdentifier.

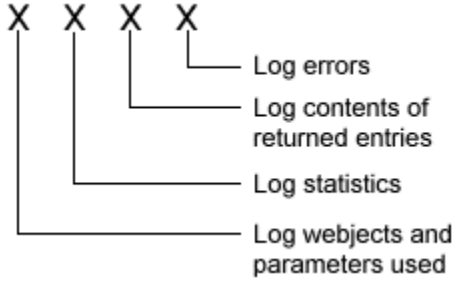
binaryMode

Specifies which attributes have non-string syntax. Attributes with non-string syntax are returned as byte arrays (byte[]).Specify the value of this property as a string of space-delimited attribute names.

If you do not specify this property, then only attributes already recognized by the LDAP Directory as having non-string syntax are returned as byte arrays. All other attributes are assumed to have string syntax. Examples of attributes that the LDAP directory might automatically consider to have non-string syntax are: photo, personalSignature, audio, jpegPhoto, javaSerializedData, thumbnailPhoto, thumbnailLogo, userPassword, userCertificate, cACertificate, authorityRevocationList, certificateRevocationList, crossCertificatePair, x500UniqueIdentifier, and any attribute ID with the ".binary" option.

debug

Determines whether debugging information is logged. Valid values are any combination of four bits that represent what debugging output information is placed in the adapter log file for troubleshooting purposes. Each bit corresponds to the logging of certain information, as shown in the following diagram:



All four bits must be set for the debugging option to work properly. For example, to turn on the logging of webjects, statistics, and errors, set the value to 1101.

The default for this property is 0000; no debugging information is placed in the log file.

Use this property rather than setting the **Logging** options that are provided later on the form.

dsaUser

User name of the default user used to log in to the directory service. More information about credential mapping can be found in the *Info*Engine User's Guide*.

There is no default for this property. If the user name is not specified here, it must be specified directly in webjects which will be accessing your chosen directory services or defined through credentials mapping.

DsaCredentials

Specifies the default password used for connections.

environment.java.naming.ldap.derefAliases

Search alias control. Valid values are NEVER, FINDING, SEARCHING, and ALWAYS:

ALWAYS always dereferences aliases.

NEVER never dereferences aliases.

FINDING dereferences aliases only while locating the target entry.

SEARCHING dereferences aliases once the target entry is located.

The default for this property is ALWAYS.

java.naming.factory.initial

Java class name of the factory class. The default for this property is com.sun.jndi.ldap.LdapCtxFactory.

java.naming.provider.url

The address of the naming or directory server.

jndi.syntax.direction

Direction in which object names in a distinguished name are ordered. Valid values are RIGHT_TO_LEFT, LEFT_TO_RIGHT, and FLAT.

The default for this property is RIGHT_TO_LEFT.

jndi.syntax.seperator

Character used to delimit the names of objects in a distinguished name. The default for this property is a comma (,).

ldapVersion

The LDAP protocol used. Valid values are 3 for LDAPv3 and 2 for LDAPv2. The default for this property is 3.

logfile

The fully qualified path of the log file that is used when the adapter is out of process. Use this property rather than the **Logging Directory**

property that is provided later on the form. The log messages for in-process adapters are written to an Info*Engine server log file and not to the file named in this property.

referral

Determines how the LDAP server processes referrals. Referrals are a mechanism by which an LDAP directory contacts other physical LDAP directories to obtain the results requested by a search. Valid values for this property are FOLLOW, IGNORE, and THROW:

FOLLOW directs the LDAP server to process referrals.

THROW directs the underlying code to throw an LdapReferralException if a referral is encountered.

IGNORE directs the underlying code to ignore any referrals and return only results obtained without referral processing.

If this property is not specified, the property will default to the setting in the LDAP server.

searchBase

Distinguished name of the entry at which a search will start.

searchFilter

Default search filter for the Query-Objects webject if no search filter is specified on the webject. The search filter allows a query to select a subset of all the entries in the scope of the query.

searchScope

Sets the scope of the search. Valid values for the property are BASE, ONELEVEL, and SUBTREE:

BASE indicates that only the current entry should be searched.

SUBTREE indicates that the search should start at the current level and search all levels of the complete LDAP hierarchy below the current level.

ONELEVEL limits the search to entries contained in the current LDAP directory hierarchy level.

The default for this property is ONELEVEL.

secret.algorithm

Algorithm used to encrypt secrets. Valid values are SHA-1, MD2, and MD5. The default for this property is SHA-1.

secret.text

Secret used to sign and validate requests. Information about validating requests can be found in the *Info*Engine User's Guide*.

secret.text2

Specifies a string used to sign and validate requests to a task processor or adapter. The secret.text2 property generates a more comprehensive request signature than the secret.text property.

serviceType

Determines what service type is used. Valid values are NAMING and DIRECTORY:

NAMING performs an LDAP one-level search operation of the named entry using the filter "(objectclass=*)" to retrieve the names of the entries immediately below the named entry.

DIRECTORY performs an LDAP search operation according to the specified search controls.

The default for this property is DIRECTORY.

sizeLimit

Maximum number of entries to be returned as a result of a query. If the specified limit is reached and the search is not complete, the found entries are returned and an IEPartialResultsException is thrown. If 0 is specified, all entries found are returned. The underlying LDAP directory may also impose a size limit restriction.

The default for this property is 0; all entries found are returned.

socketAccess.maxThreadCount

Maximum number of concurrent request threads against each adapter instance. The default for this property is 100.

timeLimit

Maximum time limit in seconds allowed for this search. If this limit is reached and the search has not finished, the found entries are returned and an IEPartialResultsException is thrown. If 0 is specified, the search will not time out.

The default for this property is 0; search will not time out.

verbose

Determines whether a detailed description sent is to the log file. Valid values are TRUE and FALSE. Selecting TRUE sends detailed descriptions to the log file.

The default for this property is FALSE; detailed descriptions are not sent to the log file.

Naming the Adapter in Webject INSTANCE Parameters

You define the adapter name to use in the INSTANCE parameter when you configure the adapter through the Info*Engine Property Administrator.

An adapter name can be one the following forms:

- A simple name, which is defined in the **Service Name** field on the Info*Engine Property Administrator service form. Simple names are stored in the ptcServiceName attribute of the adapter LDAP entry. To use a simple name, the adapter LDAP entry must reside within the Naming Service search path. For example, assume that "com.myCompany.myHost.jndi" is the ptcServiceName attribute of the adapter LDAP entry in the Naming Service search path. Then, the following INSTANCE parameter can be used:

  ```
  <ie:param name="INSTANCE" data="com.myCompany.myHost.jndi"/>
  ```

- A fully-qualified distinguished name. When configuring the adapter, you specify the distinguished name on the Info*Engine Property Administrator form. This name consists of the ptcServiceName attribute and the other attributes that define the location of the LDAP entry.

  For example if the "com.myCompany.myHost.jndi" entry is located on "host1" at "dc=IeProps,dc=myHost,dc=myCompany, dc=com,ou=Applications,o=myCompany", then the distinguished name is used in the INSTANCE parameter in the following form:

  ```
  <ie:param name="INSTANCE" data="ldap://host1/
  ptcServiceName=com.myCompany.myHost.jndi,
  dc=IeProps,dc=myHost,dc=myCompany,dc=com,
  ou=Applications,o=myCompany"/>
  ```

- A domain-based reference name. This name is just another way of identifying the distinguished name when the LDAP directory that has the Info*Engine entries is constructed using dc=com as a root-level entry and other dc attributes for subtree entries.

  The format of a Info*Engine domain-based reference name is:

  ptcServiceName@*dc_attributes*

  In this format, *ptcServiceName* is the value of the ptcServiceName attribute and *dc_attributes* are the dc attributes that make up the domain location of the LDAP entry, where each attribute is separated from the next attribute using a period.

  Note: The domain-based reference name can only be used when the LDAP directory that has the Info*Engine entries is constructed using dc=com as a root-level directory or when the Naming Service .serviceDomainBase property is set to include those attributes beyond the domain that are used in the distinguished name of the entry.

For example, if the ptcServiceName attribute value is "com.myCompany.myHost.jndi" and the entry is located in the "dc=myHost,dc=myCompany,dc=com,ou=Applications, o=myCompany" branch, then the following domain-based reference name could only be used in the INSTANCE parameter if the .serviceDomainBase property is set to "ou=Applications,o=myCompany":

```
<ie:param name="INSTANCE" data="com.myCompany.myHost.
jndi@myHost.myCompany.com"/>
```

Use the Info*Engine Property Administrator to set the .serviceDomainBase property. For more information, see the property help in the Property Administrator.

Testing the JNDI Installation

Use the following steps to test the JNDI installation.

1. Create a JNDI adapter LDAP entry to access a specific directory.
2. If your adapter is running out of process, start the adapter listening for requests.
3. Resolve any error messages that appear when attempting to start the adapter.

   The following table details the most common error messages that may appear when the adapter does not appear to start up correctly:

Error Message	Meaning	Action

Windows:
`The name specified is not recognized as an internal or external command, operable program or batch file.`
 	
UNIX:
`java: not found`
	The Java executable cannot be found.	Option 1—add the full path to your Java or jre executable to your PATH environment variable.
		
		Option 2—add the full path to your Java or JRE executable to the startup script.

`Unable to initialize threads: cannot find class java/lang/Thread`	The Java executable is not able to find classes.zip or rt.jar file in its classpath.	Ensure the classpath is set and that the path and file name specified are correct.
`Can not find class com.infoengine.JNDI .JNDI Adapter`	The Java executable cannot find the ie.jar file in its classpath.	Ensure the classpath is set and that the path and file name specified are correct.
`java.naming. factory.initial was not set`	The Java executable is not able to find a value for the java.naming.factory.initial property.	Ensure this property is set either in the startup script as a command line option or in the adapter LDAP entry.
`java.naming.provider.url was not set`	The Java executable is not able to find a value for the java.naming.provider.url property.	Make sure that this property is set either in the startup script as a command line option or in the adapter LDAP entry.

4. Execute a JSP page that contains a JNDI webject.

5. Resolve any error messages that appear when attempting to use any webjects.

   Sometimes the adapter will appear to have started correctly, but certain webjects will appear to fail. The following table details the most common error messages that are displayed when the adapter appears to be running properly but webjects appear to fail:

Error Message	Explanation and Correction
`java.lang.NoClassDefFoundError: com/sun/JNDI /toolkit/ComponentDirContext`	Although the adapter will appear to start properly, requests sent to the adapter will fail and this message will be displayed in the adapter startup window. This error indicates the Java executable cannot find the providerutil.jar file in its classpath. Ensure the classpath is set and the path and file name specified are correct.
`DirContext error! [Root exception isjava.lang.ClassNotFoundException:com.sun.JNDI .ldap.LdapCtxFactory]javax.naming.NoInitialContextException:`	Cannot instantiate the com.sun.JNDI .ldap.LdapCtxFactory class. Although the adapter will appear to start properly, requests sent to the adapter will fail. No error messages will be displayed in the adapter startup window, but this error will appear as the STATUS of an action webject request. This error indicates the Java executable cannot find the ldap.jar file in its classpath. Ensure the classpath is set and the path and file name specified are correct.
`java.lang.NoClassDefFoundError: javax/naming/directory/InitialDirContext`	Although the adapter will appear to startup properly, requests sent to the adapter will fail and this message will be displayed in the adapter startup window. This error indicates the java executable cannot find the JNDI .jar file in its classpath. Ensure the classpath is set and the path and filename specified are correct.

3

JNDI Webject Library

This chapter describes the webjects that are available through this adapter. Because display and task webjects are available for all adapters, they are described in the *Info*Engine User's Guide*. Each webject description includes the description, syntax, parameters, and, in some cases, an example.

Topic **Page**

Compare-Attribute

Description

Compares the value of an attribute name to the attribute value parameter for an entry in the directory.

Syntax

```
<ie:webject name="Compare-Attribute" type="ACT">
  <ie:param name="ATTNAME" data="attribute_name"/>
  <ie:param name="ATTVALUE" data="attribute_value"/>
  <ie:param name="BASE" data="base_context_name"/>
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="GROUP_OUT" data=group_out"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="OBJECT" data="object_value"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
ATTNAME	BASE
ATTVALUE	CONNECTION ATTEMPTS
INSTANCE	CONNECTION_ATTEMPT_INTERVAL
	DBUSER
	GROUP_OUT
	OBJECT
	PASSWD

ATTNAME

Specifies the attribute name of the entry to be compared with a given value. This is a required parameter.

ATTVALUE

Specifies the value of the attribute for the entry being compared. This is a required parameter.

Specifies the base distinguished name of the entry at which to start searching. If a value is specified for this parameter, then the webject value takes precedence over the property value specified in the adapter LDAP entry. The default for this parameter is the value specified for the searchBase property in the adapter LDAP entry. This is an optional parameter.

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

GROUP_OUT

Identifies the name of the object set returned by the action. Currently, GROUP_OUT only returns a status message. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

OBJECT

Specifies the context name of the entry. One object may be specified. If more than one is specified, the first will be used and the remainder ignored. This is an optional parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
          prefix="ie"%>
<ie:webject name="Compare-Attribute" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="ATTNAME" data="location"/>
  <ie:param name="ATTVALUE" data="Pontiac"/>
  <ie:param name="OBJECT" data="uid=pperson, ou=people"
  <ie:param name="BASE" data="uid=pperson, ou=people"/>
  <ie:param name="GROUP_OUT" data="Compare"/>
</ie:webject>
```

The Compare-Attribute webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected, Info*Engine checks the object "uid=pperson, ou=people" to see if it has an attribute "location" set to "Pontiac".

Create-Object

Description

Creates directory objects with specified attributes.

Syntax

```
<ie:webject name="Create-Object" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser"/>
  <ie:param name="FIELD" data="attributes"/>
  <ie:param name="GROUP_OUT" data="group out"/>
  <ie:param name="INSTANCE" data="instance name"/>
  <ie:param name="OBJECT" data="context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
FIELD	CONNECTION ATTEMPTS
INSTANCE	CONNECTION_ATTEMPT_INTERVAL
OBJECT	DBUSER
	GROUP_OUT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

FIELD

Specifies the attributes and associated values to set for the target object. This parameter is specified using the following forma:

attname=value

where *attname* is the attribute name, and *value* is the name of the attribute value. The values of this parameter are case-sensitive. Multiple parameters can be specified for this parameter. This parameter is required.

GROUP_OUT

Identifies the name of the object set returned by the action. Currently, GROUP_OUT only returns a status message. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

OBJECT

Specifies the context name of the entry. One object may be specified. If more than one is specified, the first will be used and the remainder ignored. This is a required parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Create-Object" type="ACT">
  <ie:param name ="INSTANCE" data="jndiAdapter"/>
  <ie:param name ="OBJECT"
                data="uid=PPerson,ou=people,o=organization.com"/>
  <ie:param name ="FIELD" data="objectclass=top"/>
  <ie:param name ="FIELD" data="objectclass=person"/>
  <ie:param name ="FIELD"
                        data="objectclass=organizationalPerson"/>
  <ie:param name ="FIELD" data="objectclass=inetorgperson"/>
  <ie:param name ="FIELD" data="sn=Person"/>
  <ie:param name ="FIELD" data="cn=People Person"/>
  <ie:param name="GROUP_OUT" data="Create"/>
</ie:webject>
```

The Create-Object webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected, Info*Engine creates a new object with the attributes submitted.

Delete-Objects

Description

Deletes an object in a directory.

Syntax

```
<ie:webject name="Delete-Objects" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="GROUP_OUT" data="group_name"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="OBJECT" data="context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
INSTANCE	CONNECTION_ATTEMPTS
OBJECT	CONNECTION_ATTEMPT_INTERVAL
	DBUSER
	GROUP_OUT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

GROUP_OUT

Specifies the name of the object-set returned by the update action. Currently, GROUP_OUT only returns a status message. This parameter is optional.

OBJECT

Specifies the context name of the entry. One object may be specified. If more than one is specified, the first will be used and the remainder ignored. This is a required parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Delete-Objects" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
                data="uid=pperson,ou=people,o=organization.com"/>
  <ie:param name="GROUP_OUT" data=delete/>
</ie:webject>
```

The Delete-Objects webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected, the object matching the specific context name is deleted.

In-Subtree

Description

Tests to see if the distinguished name for a given user is in a subtree of a directory service. If the user name exists, then a status message is returned. If the user does not exist, then an exception is returned.

Syntax

```
<ie:webject name="In-Subtree" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="GROUP_OUT" data="group_name"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="OBJECT" data="context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
  <ie:param name="SUBTREE_OBJECT" data="context_name"/>
</ie:webject>
```

Parameters

Required	Optional
INSTANCE	CONNECTION_ATTEMPTS
OBJECT	CONNECTION_ATTEMPT_INTERVAL
SUBTREE_OBJECT	DBUSER
	GROUP_OUT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

GROUP_OUT

Specifies the name of the object-set returned by the update action. Currently, GROUP_OUT only returns a status message. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

OBJECT

Specifies the context name of the entry. One object may be specified. If more than one is specified, the first will be used and the remainder ignored. This is a required parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

SUBTREE_OBJECT

Specifies the context name of the node under which a user must exist within a subtree. This is a required parameter.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject NAME="In-Subtree" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="SUBTREE_OBJECT
                              data="ou-people,o=organization.com"/>
  <ie:param name="OBJECT"
                 data="uid=PPeople,ou=people,o=organization.com"/>
  <ie:param name="GROUP_OUT" data="Create"/>
</ie:webject>
```

The In-Subtree webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected, the adapter determines if the submitted object exists in the named subtree.

Put-Blob-Stream

Description

Stores Binary Large OBject (BLOB) data on a specified database object. BLOBs are used by database administrators to refer to any random large block of bits that needs to be stored in a database, such as a picture or sound file.

Syntax

```
<ie:webject name="Put-Blob-Stream" type="ACT">
  <ie:param name="ATTRIBUTE" data="attribute_name"/>
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="FILENAME" data="filename"/>
  <ie:param name="GROUP_OUT" data="group_out"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="OBJECT" data="context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
ATTRIBUTE	CONNECTION_ATTEMPTS
INSTANCE	CONNECTION_ATTEMPT_INTERVAL
OBJECT	DBUSER
	FILENAME
	GROUP_OUT
	PASSWD

ATTRIBUTE

Specifies the name of the attribute that will hold the BLOB data. This is a required parameter.

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

FILENAME

Specifies a path to a file on the file system local to the adapter that will be stored in the directory. This parameter will only be utilized if a file has not been uploaded through the web browser. This parameter is optional.

GROUP_OUT

Identifies the name of the object set returned by the action. Specifies the name of the object-set returned by the update action. Currently, GROUP_OUT only returns a status message. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

OBJECT

Specifies the context name of the entry you wish to store the BLOB data on. You may specify only one object. If you specify more than one object, the first one will be used and the remaining ones ignored. This is a required parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in

the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Put-Blob-Stream" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
              data="uid=PPerson, ou=People, o=organization.com"/>
  <ie:param name="ATTRIBUTE" data="jpegPhoto"/>
  <ie:param name="GROUP_OUT" data="Create"/>
  <ie:param name="FILENAME" data="/usr/file.gif"/>
</ie:webject>
```

The Put-Blob-Stream webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected, Info*Engine stores the submitted BLOB data on the object "uid=PPerson,ou=People,o=organization.com" under the attribute "jpegPhoto".

Put-Bulk-Stream

Description

Saves an uploaded file to the file system that is local to the adapter.

Syntax

```
<ie:webject name="Put-Bulk-Stream" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="FILENAME" data="filename"/>
  <ie:param name="GROUP_OUT" data="group_name"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
FILENAME	CONNECTION_ATTEMPTS
INSTANCE	CONNECTION_ATTEMPT_INTERVAL
	DBUSER
	GROUP_OUT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

FILENAME

The fully qualified pathname to save the file as. This is a required parameter.

GROUP_OUT

Specifies the name of the object-set returned by the action. Currently, GROUP_OUT only returns a status message. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Put-Bulk-Stream" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="FILENAME" data="/usr/file.pdf"/>
  <ie:param name="GROUP_OUT" data="group_name"/>
</ie:webject>
```

The Put-Bulk-Stream webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected, the submitted data is stored as the file "/usr/file.pdf".

Query-Objects

Description

Queries for objects based on attribute values.

Syntax

```
<ie:webject name="Query-Objects" type="OBJ">
  <ie:param name="ATTRIBUTE" data="attributes"/>
  <ie:param name="BASE" data="base_context_name"/>
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="DEREF"
                   data="[SEARCHING | FINDING | ALWAYS | NEVER]"/>
  <ie:param name="FILTER"
                     data="[character_string | "objectclass=*"]"/>
  <ie:param name="GROUP_OUT" data="group_out"/>
  <ie:param name="INSTANCE" data="application"/>
  <ie:param name="PASSWD" data="dbpassword"/>
  <ie:param name="SCOPE" data="[BASE | ONELEVEL | SUBTREE]"/>
  <ie:param name="SIZELIMIT" data="number of entries"/>
  <ie:param name="TIMELIMIT" data="number_of_seconds"/>
</ie:webject>
```

Parameters

Required	Optional
INSTANCE	ATTRIBUTE
	BASE
	CONNECTION_ATTEMPTS
	CONNECTION_ATTEMPT_INTERVAL
	DBUSER
	DEREF
	FILTER
	GROUP_OUT
	PASSWD
	SCOPE
	SIZELIMIT
	TIMELIMIT

ATTRIBUTE

Specifies which attributes from the queried objects to return. An LDAP entry contains two types of attributes, user and operational. A user attribute is normally data that can be modified by an application that has sufficient privileges. An operational attribute is created and maintained by the directory and normally cannot be modified (i.e creatorsName). To view operational attributes, the specific attribute must be specified in the "ATTRIBUTE" parameter. If you specify "*", all user attributes are returned. Mulitple parameters can be specified using the value "*" and the explict operational attribute names to return. This combination will return all user attributes and the specified operational attributes.

If this parameter is not specified, all user attributes are returned. This parameter is optional.

BASE

Specifies the base distinguished name of the entry at which to start searching. If a value is specified for this parameter, then the webject value takes precedence over the property value specified in the adapter LDAP entry. The default for this parameter is the value specified for the searchBase property in the adapter LDAP entry. This is an optional parameter.

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

DEREF

Specifies how aliased objects should be handled during a search with respect to the base object of the search. Valid values of this parameter are SEARCHING, FINDING, ALWAYS, and NEVER.

The SEARCHING value aliases subordinates of the base object when searching the specified database, but not while locating the base object of the search. The FINDING value aliases while finding the base object of the search but not when searching subordinates of the base object. The ALWAYS value aliases both in searching the database and in locating the base object of the search. The NEVER value never aliases in searching nor in locating the base object of the search.

If a value is specified for this parameter, then the webject value takes precedence over the property value specified in the adapter LDAP entry. The default for this parameter is the value specified for the environment.java.naming.ldap.derefAliases property in the adapter LDAP entry. If the property is not specified in the adapter LDAP, then the LDAP server default is used. This parameter is optional.

FILTER

Specifies the character string representing the search filter specific to an LDAP directory search. If a value is specified for this parameter, then the webject value takes precedence over the property value specified in the adapter LDAP entry. The default for this parameter is the value specified for the searchFilter property in the adapter LDAP entry. This parameter is optional.

GROUP_OUT

Identifies the name of the object-set that is returned by the webject. Use this name as the value of the GROUP_IN parameter in a display webject. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

SCOPE

Indicates the scope of the search to be performed. Valid values are BASE, ONELEVEL, and SUBTREE. Setting the scope to BASE returns only the

base entry of the search. Setting the scope to ONELEVEL returns the base entry of the search and all of its children. Setting the scope to SUBTREE returns the base entry of the search and the subtree.

If a value is specified for this parameter, then the webject value takes precedence over the property value specified in the adapter LDAP entry. The default for this parameter is the value specified for the searchScope property in the adapter LDAP entry. This parameter is optional.

SIZELIMIT

Specifies the maximum number of entries to be returned as a result of the query. If the specified limit is reached and the search is not complete, the found entries are returned and an IEPartialResultsException is thrown. If 0 is specified, all entries found are returned. The underlying LDAP directory may also impose a size limit restriction.

If a value is specified for this parameter, then the webject value takes precedence over the property value specified in the adapter LDAP entry. The default for this parameter is the value specified for the sizeLimit property in the adapter LDAP entry. If the property is not specified in the adapter LDAP, then the LDAP server default is used. This is an optional parameter.

TIMELIMIT

Specifies the maximum time in seconds allowed for the search. If the specified limit is reached and the search has not finished, the found entries are returned and an IEPartialResultsExcepetion is thrown. If 0 is specified there are not time limit restrictions in effect for the search.

If a value is specified for this parameter, then the webject value takes precedence over the property value specified in the adapter LDAP entry. The default for this parameter is the value specified for the timeLimit property in the adapter LDAP entry. If the property is not specified in the adapter LDAP, then the LDAP server default is used. This parameter is optional.

Examples

With SCOPE set to ONELEVEL:

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Query-Objects" type="ACT">
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="BASE" data="ou=people, o=organization.com"/>
  <ie:param name="ATTRIBUTE" data="cn"/>
  <ie:param name="ATTRIBUTE " data="sn"/>
  <ie:param name="GROUP_OUT" data="query"/>
  <ie:param name="SCOPE" data="onelevel"/>
</ie:webject>
```

With SCOPE set to SUBTREE:

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Query-Objects" type="ACT">
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="BASE" data="ou=people, o=organization.com"/>
  <ie:param name="ATTRIBUTE" data="cn"/>
  <ie:param name="ATTRIBUTE " data="sn"/>
  <ie:param name="GROUP_OUT" data="query"/>
  <ie:param name="SCOPE" data="subtree"/>
</ie:webject>
```

With SCOPE set to BASE:

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Query-Objects" type="ACT">
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="BASE" data="ou=people, o=organization.com"/>
  <ie:param name="ATTRIBUTE" data="cn"/>
  <ie:param name="ATTRIBUTE " data="sn"/>
  <ie:param name="GROUP_OUT" data="query"/>
  <ie:param name="SCOPE" data="base"/>
</ie:webject>
```

Rename-Object

Description

Changes the context name of an object.

Syntax

```
<ie:webject name="Rename-Object" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="GROUP_OUT" data="group_name"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="NEW_NAME" data="new_context_name"/>
  <ie:param name="OBJECT" data="context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
INSTANCE	CONNECTION_ATTEMPTS
NEW_NAME	CONNECTION_ATTEMPT_INTERVAL
OBJECT	DBUSER
	GROUP_OUT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

GROUP_OUT

Identifies the object-set returned by the action. Currently, this parameter only returns a status message. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

NEW_NAME

Specifies the new context name for the entry you are updating. This is a required parameter.

OBJECT

Specifies the context name of the entry. If more than one is specified, the first will be used and the remainder ignored. This is a required parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Rename-Object" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
                 data="uid=pperson,ou=people,o=organization.com"/>
  <ie:param name="NEW_NAME"
                data="uid=p.person,ou=people,o=organization.com"/>
  <ie:param name="GROUP_OUT" data="Rename"/>
</ie:webject>
```

The Rename-Object webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected to the system, an object matching the specified context name is identified and its context name is changed to "uid=p.person,ou=people,o=organization.com"

Send-Blob-Stream

Description

Retrieves Binary Large OBject (BLOB) data from a directory service and streams it back to the browser. Blobs are used by database administrators to refer to any random large block of bits that needs to be stored in a database, such as a picture or sound file.

Syntax

```
<ie:webject name="Send-Blob-Stream" type="ACT">
  <ie:param name="ATTRIBUTE" data="attribute_with_BLOB_data"/>
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="MIMETYPE" data="attribute_with_mimetype"/>
  <ie:param name="OBJECT" data="context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
ATTRIBUTE	CONNECTION_ATTEMPTS
INSTANCE	CONNECTION_ATTEMPT_INTERVAL
MIMETYPE	DBUSER
OBJECT	PASSWD

ATTRIBUTE

Specifies which attribute on the found object contains the BLOB data to return. This is a required parameter.

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

MIMETYPE

Describes the name of the attribute that has a mimetype. If placed in single quotes, the string typed here will be used as the exact mimetype for the data. If no quotes are used, the string represents the attribute which contains the mimetype. This is a required parameter.

OBJECT

Specifies the context name of the entry. If more than one is specified, the first will be used and the remainder ignored. This is a required parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Send-Blob-Stream" TYPE="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
                data="uid=pperson,ou=people,o=organization.com"/>
  <ie:param name="MIMETYPE" data="'image/gif'"/>
  <ie:param name="ATTRIBUTE" data="people_person"/>
</ie:webject>
```

The Send-Blob-Stream webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected to the system, The binary data stored in the specified object is streamed back to the browser.

Send-Bulk-Stream

Description

Retrieves a file from the file system that is local to the adapter and streams it back to the browser.

Syntax

```
<ie:webject name="Send-Bulk-Stream" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="FILENAME" data="file_name"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="MIMETYPE" data="mimetype"/>
  <ie:param name="OBJECT" data="object_context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
FILENAME	CONNECTION_ATTEMPTS
INSTANCE	CONNECTION_ATTEMPT_INTERVAL
MIMETYPE	DBUSER
	OBJECT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence

over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

FILENAME

Describes the name of the attribute that contains the filepath. If the OBJECT parameter is specified, then the value specifies a distinguished name to an LDAP entry which contains the filepath. This is a required parameter.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

MIMETYPE

Describes the name of the attribute that has a mimetype. If placed in single quotes, the string typed here will be used as the exact mimetype for the data. If no quotes are used, the string represents the attribute which contains the mimetype. This is a required parameter.

OBJECT

Specifies the context name of the entry. If more than one is specified, the first will be used and the remainder ignored. This is an optional parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
          prefix="ie"%>
<ie:webject name="Send-Bulk-Stream" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="FILENAME" data="'/usr/file.gif'"/>
  <ie:param name="MIMETYPE" data="'image/gif'"/>
</ie:webject>
```

The Send-Bulk-Stream webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected, Info*Engine returns a stream of data representing the file /usr/file.gif. The gif image can be displayed in the browser.

Update-Object

Description

Updates the attributes of an object in a directory.

Syntax

```
<ie:webject name="Update-Object" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="FIELD" data="attribute_name=value"/>
  <ie:param name="GROUP_OUT" data="group_name"/>
  <ie:param name="INSTANCE" data="instance_name"/>
  <ie:param name="MODIFICATION" data="[ADD | REPLACE | DELETE]"/>
  <ie:param name="OBJECT" data="context_name"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
FIELD	CONNECTION_ATTEMPTS
INSTANCE	CONNECTION_ATTEMPT_INTERVAL
MODIFICATION	DBUSER
OBJECT	GROUP_OUT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence

over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

FIELD

Specifies the attributes and associated values to set for the target object. The values of this parameter are case-sensitive.

This parameter is used in conjunction with the MODIFICATION parameter to define updated attributes. Depending on how MODIFICATION is specified, this parameter is specified using one of the following formats:

- If the MODIFICATION parameter is specified as ADD or REPLACE, then specify this parameter in the following format:

  attname=value

  where *attname* is the attribute name and *value* is the name of the attribute value.

- If the MODIFICATION parameter is specified as DELETE, and this parameter is specified in the above format, then the attribute value specified here is deleted from the attribute.

- If the value of the MODIFICATION parameter is set to DELETE, and this parameter is specified in the following format:

  attname

  where *attname* is the attribute name, then the entire attribute is deleted.

This parameter is required.

GROUP_OUT

Specifies the name of the object-set returned by the update action. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

MODIFICATION

Specifies the update action to take against the attributes and attribute values specified in the FIELD parameter. Valid values are ADD, REPLACE, and DELETE:

ADD appends the attribute values specified in the FIELD parameter, creating an attribute if necessary.

REPLACE substitutes the attribute values specified in the FIELD parameter for the existing attribute values, creating an attribute if necessary.

DELETE removes the attribute values specified in the FIELD parameter from an entry. Because attributes can have multiple values, either the entire attribute or one value can be deleted. If only the attribute is specified in the FIELD parameter, then the entire attribute is removed.

This is a required parameter.

OBJECT

Specifies the context name of the entry. If more than one is specified, the first will be used and the remainder ignored. This is an optional parameter.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Update-Object"  type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
              data="uid=pperson,ou=People,o=organization.com"/>
  <ie:param name="MODIFICATION" data="REPLACE"/>
  <ie:param name="FIELD" data="location=Pontiac"/>
  <ie:param name="GROUP_OUT" data="update"/>
</ie:webject>
```

The Update-Object webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected to the system, the attribute "location" is changed to "Pontiac" on the specified object.

Example of an Update Through Addition

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Update-Object" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
                        data="cn=Wes,ou=Sales,ou=dtr,o=aux,c=US"/>
  <ie:param name="MODIFICATION" data="add"/>
  <ie:param name="FIELD" data="sn=Shiminator"/>
  <ie:param name="FIELD" data="sn=Shimano"/>
  <ie:param name="FIELD" data="mail=junk"/>
  <ie:param name="GROUP_OUT" data="leaf_data"/>/
</ie:webject>
```

Example of an Update Through Replacement

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Update-Object" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
                        data="cn=Wes,ou=Sales,ou=dtr,o=aux,c=US"/>
  <ie:param name="MODIFICATION" data="replace"/>
  <ie:param name="FIELD" data="sn=Shimrep1st"/>
  <ie:param name="FIELD" data="sn=Shimrep2nd"/>
  <ie:param name="GROUP_OUT" data="leaf_data"/>
</ie:webject>
```

Example of an Update Through Deletion

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Update-Object" type="ACT">
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="OBJECT"
                        data="cn=Wes,ou=Sales,ou=dtr,o=aux,c=US"/>
  <ie:param name="MODIFICATION" data="delete"/>
  <ie:param name="FIELD" data="sn=Shiminator"/>
  <ie:param name="FIELD" data="sn=Shimano"/>
  <ie:param name="FIELD" data="mail=junk"/>
  <ie:param name="GROUP_OUT" data="leaf_data"/>
</ie:webject>
```

Validate-User

Description

Validates the identity of a directory service user.

Syntax

```
<ie:webject name="Validate-User" type="ACT">
  <ie:param name="CONNECTION_ATTEMPT_INTERVAL" data="interval"/>
  <ie:param name="CONNECTION_ATTEMPTS" data="attempts"/>
  <ie:param name="DBUSER" data="dbuser_name"/>
  <ie:param name="GROUP_OUT" data="group_name"/>
  <ie:param name="INSTANCE" data="application"/>
  <ie:param name="PASSWD" data="dbpassword"/>
</ie:webject>
```

Parameters

Required	Optional
INSTANCE	CONNECTION_ATTEMPTS
	CONNECTION_ATTEMPT_INTERVAL
	DBUSER
	GROUP_OUT
	PASSWD

CONNECTION_ATTEMPTS

Defines the maximum number of times to attempt establishing a connection to an adapter before returning an error. The default value is 1. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPTS defines the maximum number of times to iterate through the list of adapter instances.

CONNECTION_ATTEMPT_INTERVAL

Defines the amount of time, in seconds, to delay between connection attempts. The default value is 60 seconds. This parameter is optional.

If multiple INSTANCE parameter values are specified, the value of CONNECTION_ATTEMPT_INTERVAL defines the number of seconds to wait between the attempts to iterate through the entire list of adapter instances.

DBUSER

Specifies the name to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

GROUP_OUT

Specifies the name of the object-set returned by the update action. This parameter is optional.

INSTANCE

Specifies the name of the adapter that executes the webject. Adapter names are defined when the adapter is configured for use in your Info*Engine environment. This parameter is required.

In order to provide the ability to connect to other adapters if a specific adapter is not available, this parameter can be multi-valued. Info*Engine attempts to connect to the adapters in the order given. If the first adapter specified in not available, the next adapter listed is tried, and so on, until a connection is made. If a connection cannot be established with any listed adapter, an error is returned.

In conjunction with this parameter, you can include two other parameters, CONNECTION_ATTEMPTS and CONNECTION_ATTEMPT_INTERVAL.

PASSWD

Specifies the password to use when logging in to the data repository. If this parameter is specified in this webject, the webject value takes precedence over any value specified in the credentials mapping settings or in the adapter LDAP entry. If this parameter is not specified here, it must be specified in the credentials mapping settings or in the adapter LDAP entry. For more information about credentials mapping, see the *Info*Engine User's Guide*.

Example

```
<%@page language="java" session="false"%>
<%@taglib uri="http://www.ptc.com/infoengine/taglib/core"
         prefix="ie"%>
<ie:webject name="Validate-User" type="ACT"/>
  <ie:param name="INSTANCE" data="jndiAdapter"/>
  <ie:param name="DBUSER"
                data="uid=pperson,ou=people,o=organization.com"/>
  <ie:param name="PASSWD" data="password"/>
  <ie:param name="GROUP_OUT" data="validate"/>
</ie:webject>
```

The Validate-User webject shown in this example instructs Info*Engine to connect to the appropriate adapter instance. Once connected to the system, Info*Engine validates that the specified user has permission to connect to the naming or directory service.