ApplinX User Guide

Administration

Version 9.7

October 2014

webMethods ApplinX

This document applies to ApplinX Version 9.7.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: APX-UG-ADMIN-97-20150616

Table of Contents

1 Managing the ApplinX Server

Starting ApplinX Server

Windows Environment

The ApplinX server can be started either as a Windows service or using a batch file.

Note: When selecting to run ApplinX as a service it will by default, start automatically when Windows is started.

▶ To start the ApplinX server as a Windows system service

1 In the installation process, define that ApplinX server will run as a Windows system service.

2 Select **Start>Settings>Control Panel>Administrative Tools>Services**.

3 Start the Software AG ApplinX Server service.

▶ To start the ApplinX server using a batch command file

1 Access the <ApplinX installation>/bin folder.

2 Double-click *startup.bat*.

It is possible to configure the server's system properties.

Refer to **ApplinX Server, System Parameters** for further details.

UNIX Environment

▶ To start the ApplinX server

- In the ApplinX installation directory, start the server using the *startup.sh* shell command.

It is possible to configure the server's system properties.

Refer to **ApplinX Server, System Parameters** for further details.

ApplinX Server System Parameters

When running ApplinX server as a batch file, these parameters can be configured in the *startup.bat* batch file/*GXApplinXService.ini*/startup.sh.

Parameter	Description
com.sabratec.gxhome	The location of the applinx_home directory.
com.sabratec.license	The directory of the license file.
com.sabratec.useicon	Put the server icon in the icon tray.
com.sabratec.conf	ApplinX configuration file name - under the config folder of the installation.
com.sabratec.prpfile	The location of gxstartup.prp.
com.sabratec.logger	Servlet logger.
com.softwareag.applinx.ndt.endians_switch	When using Natural UNIX natural data transfer, if the download or upload gets stuck, this may be due to Endian issues and setting this parameter to true may fix this (by default this parameter is false).

The following parameters provide you with the ability to determine the server's system properties. Define these properties in the *\config\gxstartup.prp* file. These parameters can be overridden by parameters defined in the *startup.bat* batch file/*GXApplinXService.ini*/startup.sh files.

Parameter	Description
com.sabratec.license	The directory of the license file.
com.sabratec.useicon	Put the server icon in the icon tray.
com.sabratec.conf	ApplinX configuration file name - under the config folder of the installation.
com.sabratec.logger	Servlet logger.

Stopping ApplinX Server

Windows Environment

To stop the ApplinX server, when started as a Windows system service

1. Select **Start>Settings>Control Panel>Administrative Tools>Services**.
2. Stop the Software AG ApplinX Server service.

To stop the ApplinX server using a batch command file

1. Access the ApplinX installation folder.
2. Double-click *shutdown.bat.*

UNIX Environment

To stop the ApplinX server

- In the ApplinX installation directory, you will find the *shutdown.sh* shell command.

Connecting to the Server

To connect to the server

1. In the ApplinX Designer, click **Connect to Server** on the toolbar, or right-click on the relevant server and select **Connect**.
2. When connecting for the first time, ensure that the server address (IPv4 and IPv6 address formats are supported) and port that are displayed in the pop-up are correct.
3. Click **OK**.
4. Type the **User name** and **Password** and click **OK**.

To disconnect from the server

- In the ApplinX Designer, click **Disconnect from Server** on the toolbar, or right-click on the relevant server and select **Connect**.

  The communication with the server terminates, and the user's details disappear.

Configuring the Server

To update a Server's configuration, open ApplinX Designer, and either right-click on the server (in the ApplinX Explorer) and select **Properties**, or select **Properties** from the **Server** menu.

The *Server Properties* dialog box is displayed.

Edit the dialog box as follows:

- CentraSite
- General
- License
- Log
- Outgoing SMTP Server
- WS-Stack

CentraSite

Enable CentraSite

Selecting this option enables connecting to CentraSite. This is available only when you have a relevant license and have the installed the required Software AG common files. An error message will indicate when one of these are missing. If the common files are not installed, run the Software AG Installer and within "Infrastructure>Libraries" select to install "Shared Libraries" and "CentraSite Libraries".

Host

The name of the host where CentraSite is installed.

Port

The port number used to connect to CentraSite

User

Current user name.

Password

Password of current user.

Test Connection

Clicking on this button will test that a connection with CentraSite has been established.

General

Non secured port

Does not require user name and password authentication.

Secured port

The **Secured port** check box determines whether SSL is used to securely transfer the data between the client and ApplinX server.

To change the HTTP port, edit the server.xml and gxconfig.xml files.

Load ApplinX archive applications (gxar)

This check box determines whether to automatically update the application with new/updated gxar files. Once selected, ApplinX searches the host-applications folder looking for new/updated gxar files. When one is found, it is loaded automatically as a new/updated application. You must determine how often to search the folder (by default this is set as one minute).

Enable encryption of recorded sessions

Selecting this checkbox, enables encrypting recorded sessions.

Encryption key file

The encryption key enables ApplinX to encrypt and decrypt the recorded sessions. A key is required for each ApplinX server.

Note: Refer to Recording Sessions for further details regarding encrypting recorded sessions.

License

This page provides information regarding the ApplinX Server license. It includes the number of users, the number of licenses for each type of license, the expiry date, the platforms and the special license terms (such as whether the license includes Web Enablement and/or SOA and/or BPD (Screen Process Extractor)). Refer to the License Key Concept.

Log

The server log can be accessed from ApplinX Administrator or from the ApplinX Designer. Refer to **Viewing Server Logs** for further details.

Level

The contents of the log file are as detailed as this property defines, where every level includes the levels above it. For example, the Debug level also logs Normal, Warnings and Errors Only levels. Available values: "Errors only", "Warnings", "Normal" and "Debug" (by default Normal is selected).

Log File Name

The log is written to this file.

Open log folder
Enables you to see the list of existing log files.

Max. file size
Starts a new log file after the current file has been filled to the maximum file size.

Save History
Selecting this check box determines whether backups of old log files will be saved, after restarting the server.

The radio button options determine the number of backups saved before overwriting the old log files. For example: 10 means "save the last 10 log files, in addition to the current one, then start to overwrite". When selecting **All Files**, old log files are never deleted. Default value is "10".

Outgoing SMTP Server

Server Address
The IP address of the SMTP server.

From address
Mail sent as part of a Procedure, requires using a From address, which is defined in the procedure. When the From field in the procedure is left empty, the address defined here, in the **Default From address**, will be used.

Requires authentication
For security reasons, you may want to require authentication. Enter the **Account name** and **Password**.

Note: Only an Administrator is able to change the server's configuration.

WS-Stack

Enables selecting whether to work in embedded (default) or in external mode. Use the Embedded mode when WS-Stack uses the same Tomcat as ApplinX and also when working with Web Services created in previous ApplinX versions (Administrative Web Services and Procedure Group generated Web Services). Use the External mode when not using the same Tomcat as ApplinX, for example when working with one WS-Stack Web application for all SAG products or when working with a WS-Stack Web application which is on a different machine.

When connecting to a WS-Stack installation on a different machine, you are required to enter the following:

Host/IP
The host name or IP Address (IPv4 and IPv6 address formats are supported).

Port
The Tomcat port where WS-Stack Web application is deployed.

Servlet name

The URL of the servlet that the WS-Stack uses for deployment tasks.

User

The user name used by WS-Stack.

Password

The password used by WS-Stack.

Click **Test Connection**, to test the connection to the remote machine.

Refer to Integration between ApplinX and WS-Stack

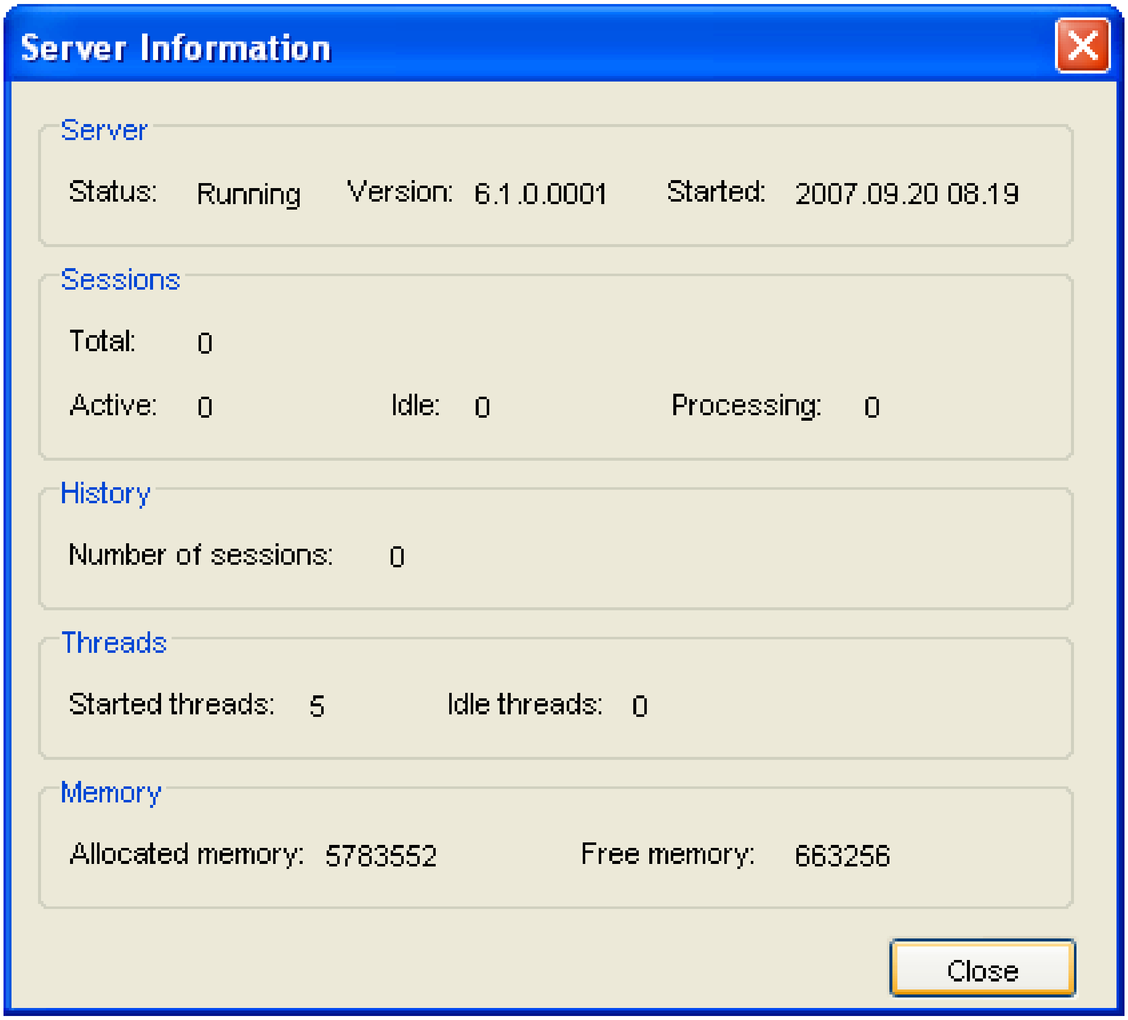
Accessing Server Information

This feature provides important information required for monitoring the server's status and activity history.

To access the Server's status and activity history

1 In the ApplinX Administrator, click **Server Information** on the toolbar, or from the **Tools** menu, select **Server Information**.

2 Click **Close** to exit the *Server Information* dialog box.



Status: Running Version: 6.1.0.0001 Started: 2007.09.20 08.19

Total: 0

Active: 0 Idle: 0 Processing: 0

Number of sessions: 0

Started threads: 5 Idle threads: 0

Allocated memory: 5783552 Free memory: 663256

Server

Server Status

"Running" when the server is connected.

Version

The ApplinX server's version.

Started

The date and time the server was last initialized.

Sessions

Total

The total number of sessions connected to the server presently.

Active

The number of currently active sessions.

Idle

The number of sessions currently idle (connected, but detached).

Processing

The number of sessions currently processing XML requests.

History

Number of Sessions

The total number of sessions that have connected to the server since it was last initialized.

Threads

Started Threads

The number of threads that were activated.

Idle Threads

How many threads were activated, but are currently not used.

Memory

Allocated Memory

The total amount of memory (in bytes) that is currently allocated to ApplinX server on the Java Virtual Machine.

Free Memory

The amount of memory (in bytes) currently available for ApplinX to use.

This information can be accessed using the API detailed in Retrieving Server Information.

Viewing Server Logs

The Server Log includes information as to the Server's activities and problems. The contents of the server log file are defined according to the settings configured in the Server Parameter>Log node. The Server Log can be accessed either from ApplinX Administrator, from ApplinX Designer or via an external browser: ApplinX Administrator can be used by administrators and/or developers. Administrators who do not have ApplinX Designer can access the Server Log from the Designer. Administrators who do not have ApplinX Designer or ApplinX Administrator can access the Server Log via an external browser.

▶ To View the contents of server log files from within ApplinX Administrator

1 In the ApplinX Administrator, click **Server Log** on the toolbar to view the current log file’s contents. To view previous log file history: click the **Server Logs** node under the Management node. The paths of these log files; according to the Server definitions (see Server Configuration) appear in the Main view.

2 Double-click a file's name. A dialog box with the file's contents appears.

3 Click **Refresh**, or press the **F5** key on the keyboard to renew information on the log file manually. Check **Auto refresh** to automatically refresh the log file.

4 Click **OK** to exit the *Log File* window.

To View the contents of server log files from within ApplinX Designer

1 In the ApplinX Designer, either right-click on a server and select **Show Server Log**, or select **Show Server Log** in the **Server** Menu.

2 To view previous log file history select the relevant file from the list of files in the File name field.

3 Double-click a file's name. A dialog box with the file's contents appears.

4 Click **Refresh**, or press the **F5** key on the keyboard to renew information on the log file manually.

5 Click **Restart Log** to restart the log. The previous data is saved to a file, and a new file is created.

2 Current Activity

Managing Sessions

Viewing Sessions

When you connect to the server, you are able to see in the Administrator details of all the sessions currently running on the server. To view the sessions on the server, open ApplinX Administrator and expand the **Management>Current Activity>Sessions** node. The Main view pane on the right will display the session's details:

Session ID
The ID of the session.

Example Value
U0000001

Description
The session's description, for example, User ID on the host or IP address.

Application
The name of the application the session relates to.

Example Value
CompositeDemo

Device Name
Workstation ID/LU Name, available only in certain protocols.

Duration
The amount of time the session has been in its current state.

Example Value
14:30

State
Current status of the service. Can be "Not Started", "Initializing", "Active", "In standby", "Suspended", "Stopping" or "Stopped".

Example Value
Active

Connection Pool
The name of the connection pool.

Example Value
/<connection pool name>

Type
The type of session currently running: Web Enablement, SOA, or Development session (Designer)

The status bar shows the total number of sessions currently running, and in parenthesis, the number of active sessions.

Viewing a Session's Properties

To view a specific session's properties, open ApplinX Administrator, right-click the specific session in the Main view and select **Open**. The *Session* dialog box appears including the following information:

Session

ID
The session’s ID on the ApplinX server.

Address
The IP address from where the session is connected to ApplinX.

Description
The session’s description. For example, this may be the session’s computer address.

Host

Address
The IP address of the host.

Device name
Workstation ID/LU name, available only in certain protocols.

Application
The name of the application on ApplinX server to which this session is connected.

Replay
The GCT file name that is to work with this session and the screen number in the GCT file.

Trace
The name of the trace file that is tracing the current session.

Bytes sent
The number of bytes sent to the host.

Bytes received
The number of bytes received from the host.

Status

Idle time

The time period in which a session has not performed a communication activity with the host.

Number of calls

The number of communication activities the session has made with the host.

State

The current communication status between ApplinX server and the host. Can be either: "Idle" (connected, not attached), "Initializing", "Processing" (executing an action), "Active" (attached), or "Disconnecting".

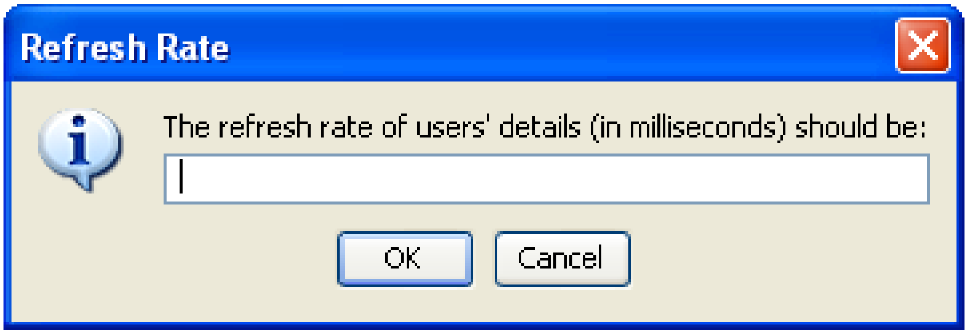
Setting the Refresh Rate

By default, the session list is sampled every second in order to keep it updated. When many users connect to the server, this refresh rate can significantly slow down the list's update. You can change the system's refresh rate to suit your needs.

▶ To set the Refresh Rate

1 Open ApplinX Administrator.

Right-click on the Management>Current Activity>Sessions node and select Refresh Rate. The Refresh Rate dialog box appears.



Refresh Rate

The refresh rate of users' details (in milliseconds) should be:

2 Type the new **Refresh Rate** in milliseconds, click **OK**.

The Refresh Rate is changed, and you may read all updated data from the server in the desired frequency.

Accessing the Session Viewer

This feature allows you to view the current host session while you are in the Administrator. The benefits of this feature are that it allows you to view the current host screen at the same time as other users are viewing the same host screen. If you are experiencing problems in code, the *Session Viewer* allows you to view the current host screen.

To access the *Session Viewer*, open ApplinX Administrator and select the relevant session in the Main View, then either right-click on the session and select **Show Session** or select **Show Session** in the **Connection** menu.

Canceling a Session

Using the Administrator, you can cancel a session from the server. This is useful when:

- A session is "stuck" and is unable to perform any activity.
- An unauthorized session has logged on.
- A session remains idle for a long time, and a "non-activity" timeout has not been defined.

To cancel a session on the server

1. Open ApplinX Administrator.
2. Select the relevant session in the Main View, then either right-click on the session and select **Cancel Session** or select **Cancel Session** in the **Connection** menu.
3. Click **Yes** in the *Cancel Session* dialog box appears.

Managing Connection Pools

Note: Refer to the Administrative Web Services API to access connection pool information using the API.

Viewing Application Connection Pools

To view a list of all Pools whose applications are loaded open ApplinX Administrator and select the **Management>Current Activity>Connection Pools** node.

Application/Connection Pool

The name of the application followed by the full name of the Connection Pool.

Example Value

Demo:/folderA /ConnectionPoolName

Status

Current status of the connection pool:

"Not Started"- The connection pool was not initialized yet. If a user requests a connection the connection pool will return an immediate error.

"Initializing" - The connection pool is trying to reach Active status, but does not have any ready connection yet. If a user requests a connection the connection pool will return an immediate error.

"Active" - The connection pool is working, and managed to create at least one connection to the host.

"In standby" - The connection pool had several consecutive errors trying to create new connections to the host. The connection pool will continue to try to connect if user requests arrive, but will not initiate new connections otherwise. If a new connection is successfully created, the status will automatically change to Active.

"Suspended" - The connection pool is blocked for new users, and does not maintain its connections. If a user requests a connection the connection pool will return an immediate error.

"Stopping" - The connection pool is trying to reach Stopped status, but still has connections in different stages of termination. When all connections are down, the status will automatically change to Stopped. If a user requests a connection the connection pool will return an immediate error.

"Stopped" - The connection pool does not have connections or maintenance. If a user requests a connection the connection pool will return an immediate error.

Active

The number of connections in use (with users attached).

Example Value

10

Ready

The number of available connections.

Example Value

5

Process

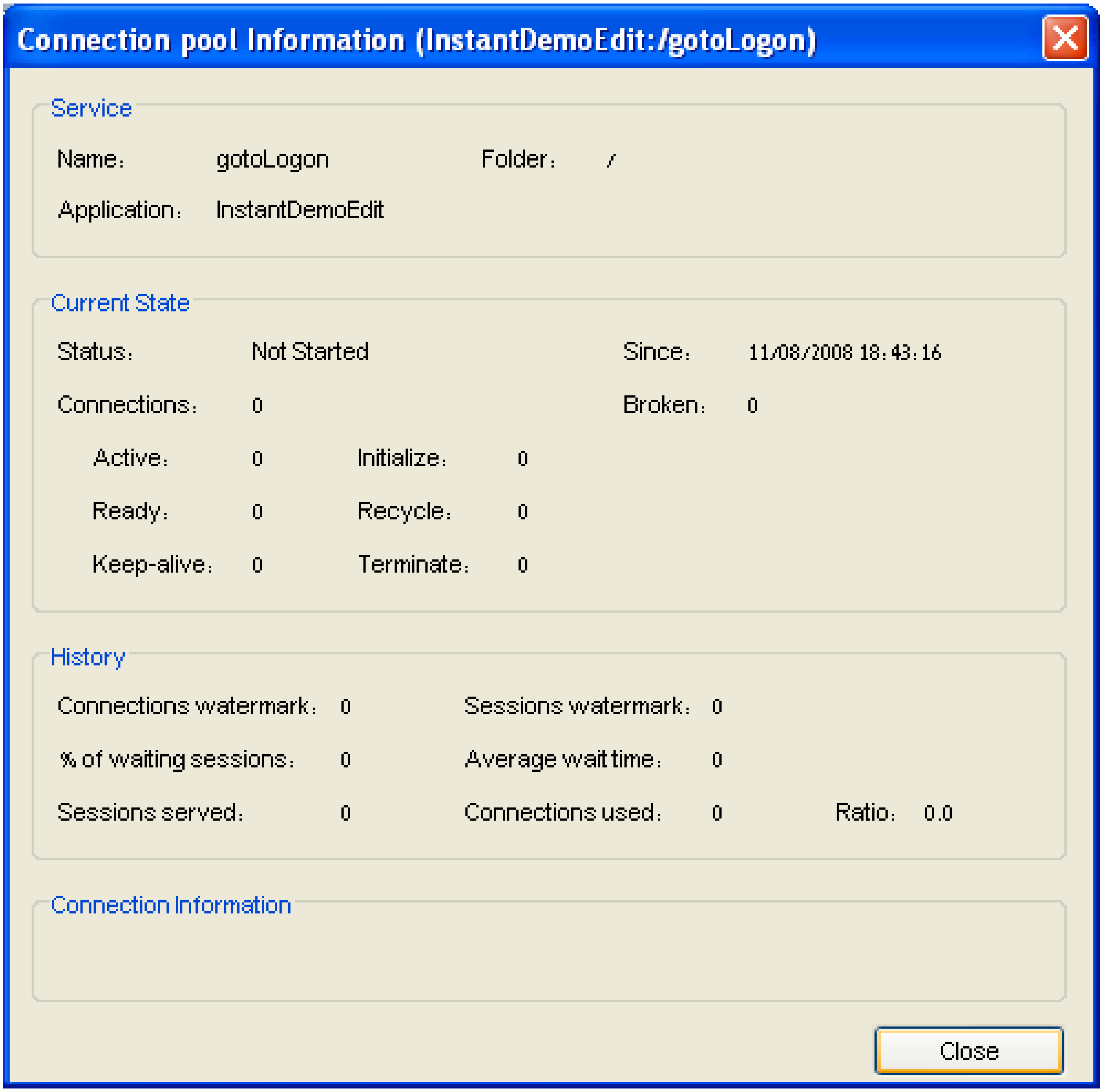
The number of connections currently in one of the following states: initializing, recycling or keep-alive.

Example Value

1

Viewing Connection Pool Monitoring Information

To view the run time information about a specific connection pool, open ApplinX Administrator and select the **Management>Current Activity>Connection Pools** node and then either double-click the required connection pool in the Main view area or right-click the required connection pool either in the ApplinX Explorer or in the Main view section and select **Show Monitor**.



Connection pool Information (InstantDemoEdit:/gotoLogon)

Service

Name: gotoLogon   Folder: /

Application: InstantDemoEdit

Current State

Status: Not Started   Since: 11/08/2008 18:43:16

Connections: 0   Broken: 0

Active: 0   Initialize: 0

Ready: 0   Recycle: 0

Keep-alive: 0   Terminate: 0

History

Connections watermark: 0   Sessions watermark: 0

% of waiting sessions: 0   Average wait time: 0

Sessions served: 0   Connections used: 0   Ratio: 0.0

Connection Information

Service

This panel displays identifying information about the displayed connection pool.

Name
The name of the displayed connection pool.

Folder
The folder in which the displayed host connection service is placed.

Application
The application to which the connection pool belongs.

Current State

This panel displays run time information about the displayed connection pool.

Status
The connection pool's status: "Not Started, Initializing, Active, In standby, Suspended, Stopping "or "Stopped".

Since
The connection pool is in its current status since this time.

Connections
The total number of connections in the connection pool, ignoring broken connections.

Broken
The number of connections that failed initialization recently. When more than 0, this number is shown in red.

Active
The number of connections currently held by a session (user).

Ready
The number of connections ready for use.

Keep-Alive
The number of connections currently performing keep-alive activity.

Initialize
The number of connections currently performing initialization activity.

Recycle
The number of connections currently performing recycling activity.

Terminate
The number of connections currently performing termination activity.

History

This panel displays accumulative information about the displayed connection pool. The information in this panel is reset when the connection pool is stopped.

Connections watermark
The maximum number of concurrent connections in this connection pool.

Sessions watermark
The maximum number of sessions that used this connection pool concurrently.

% of waiting sessions
The percent of sessions that did not immediately get a connection when trying to connect to ApplinX.

Average wait time

The average time (in milliseconds) sessions waited for a READY connection (calculated only among those sessions that waited) multiplied by the percentage of waiting sessions. For example: if 8% sessions had to wait, and in average each of those waited 1000 milliseconds, the overall average wait time was: 0.08 * 1000 = 80 milliseconds.

Sessions served

The total number of sessions that connected to the host service since the service started.

Connections used

The total number of different host connections created by this connection pool.

Ratio

Sessions served divided by **Connections used**. This parameter can give a general indication of how much connection recycling is effective. A large ratio means that a relatively small number of connections served a large number of sessions.

Connection Information

This panel displays the name of the **Information Set** that the displayed host service uses. If this panel is empty, no Information set is used.

Information set

The name of the information set used by this connection pool.

Viewing Connection Information

To view a list of all the connections of a particular connection pool, open ApplinX Administrator and select the **Management>Current Activity>Connection Pools** node and in the ApplinX Explorer area select the required connection pool.

Connection ID

The connection identification number (unique in the contexts of this connection pool)

Example Value

6

Status

Current status of the connection. Can be "Active", "Ready", "Initializing", "Keep-alive", "Recycle" or "Terminating".

Example Value

Active

Session ID

This column is relevant only for active connections. The Session ID of the user that holds this connection.

Example Value

U0000014

Time

The time that passed since the connection is in its current status.

Example Value

00:03:24

Error Message

Relevant only for broken connections. A message that may imply on the reason this connection became corrupt.

Canceling a Connection

To cancel a connection, open ApplinX Administrator and expand the **Management>Current Activity>Connection Pools** node. Right-click on a connection and select **Cancel**.

Note: Active connections can also be cancelled by canceling their user through the Sessions node.

Changing the Status of a Connection Pool

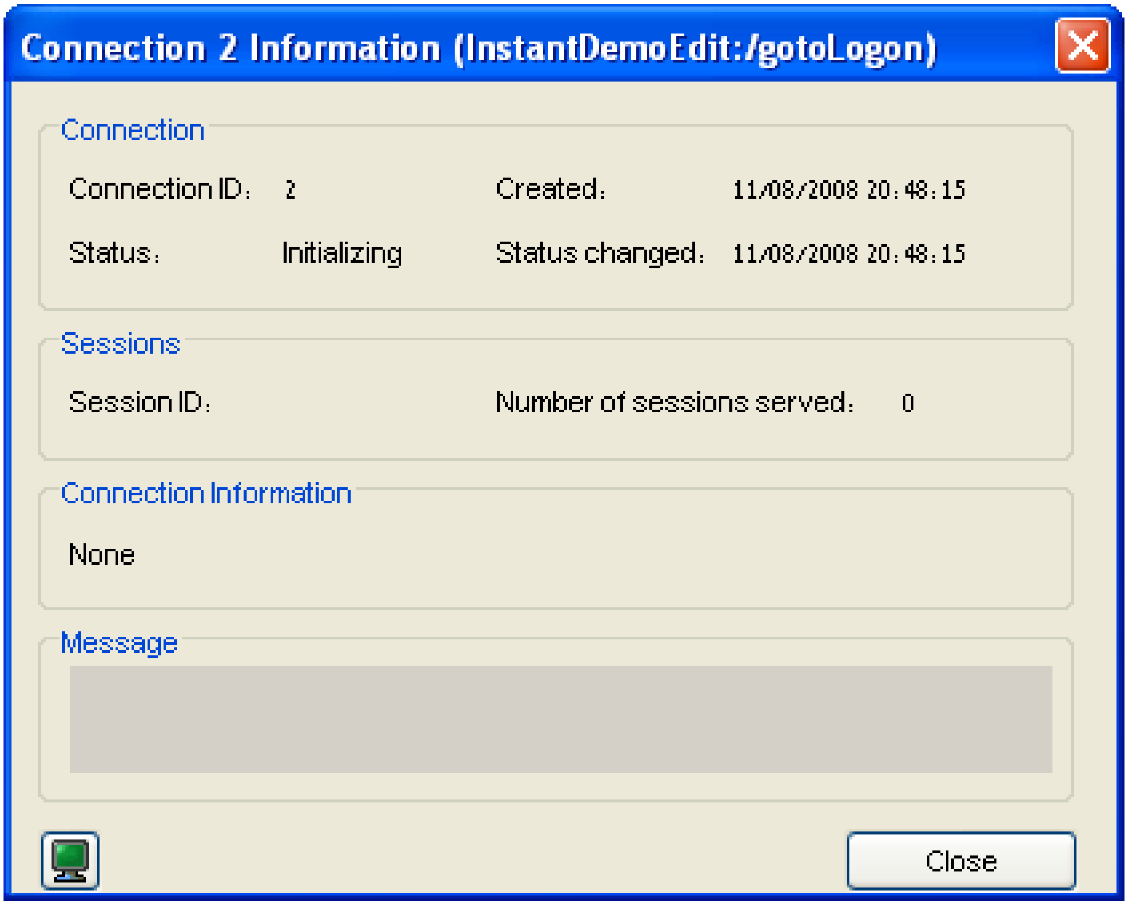
To modify the activity status of a connection pool, open ApplinX Administrator and expand the **Management>Current Activity>Connection Pools** node. Select one or more connections you want to control, right-click and select one of the options detailed below:

Note: The available options will vary according to the current status of the selected connection pool(s).

Name	Description
Start Connection Pool	A connection pool that has not been started or a stopped connection pool will reload its configuration, initialize a new pool and start the connection pool operation. During the activation of the pool, until at least one connection is ready, the connection pool is in Initializing status. When the Start connection pool option is used on a suspended connection pool, it will simply resume operation.
Suspend Connection Pool	Relevant only for active connection pools. Will stop creating new connections and connection pools for new requests, but will not close the existing connections in the pool.
Resume Connection Pool	Resumes the possibility to use a suspended host service of the application
Stop Connection Pool	Stop providing connection pool and close all connections in the pool. During this process, the connection pool enters Stopping status, until all connections are closed. Then the connection pool becomes Stopped. Reactivation will reload the connection pool configuration.
Restart Connection Pool	Stop the connection pool. After it reaches Stopped status, start the connection pool again.

Viewing Connection Monitoring Information

To view a form that displays the run time information about a specific connection of a host service, open ApplinX Administrator and click on the **Management>Current Activity>Connection Pools** node and select a connection pool. Right-click the required connection in the Main view section and select **Show Monitor**. The *Connection Information* dialog box which displays run time information about a specific connection of a host service appears.



Connection 2 Information (InstantDemoEdit:/gotoLogon)

Connection

Connection ID: 2

Created: 11/08/2008 20:48:15

Status: Initializing

Status changed: 11/08/2008 20:48:15

Sessions

Session ID:

Number of sessions served: 0

Connection Information

None

Message

Connection

Connection ID
The Identifier of the displayed connection.

Created
Creation time of this connection.

Status
The current status of the connection. One of:" Active/ Ready/ Initializing/ Recycling/ Keep-alive/ Terminating/ Broken".

Status changed
The time of the last change of the connection’s status.

Sessions

Session ID
Active connections only: displays the ID of the session holding the displayed connection.

Number of sessions served
The number of sessions this specific connection served so far.

Connection Information

Using connection information ID
If a Connection Information set is used by this connection pool, this is the ID of the specific record used by the displayed connection.

Message Panel

Displays error messages for broken connections.

3 Users: Roles and Permissions

The Security node in ApplinX allows managing users, groups and their permissions. It is possible to define certain permissions for a group, and then associate users with this group, giving the user the permissions defined for this group. For example, a specific application that has a list of users who can develop the application and a list of users who can only view the application will have two groups with relevant permissions, and users will be associated to the relevant group. A change in the group permissions will take affect on all users belonging to this group. Users will inherit the permissions from all the groups to which they have been associated. Specific permissions given to a user, will override group permissions. For example, if a user inherits edit permission for 'CompositeDemo' but also has view permission, he will have view permission only.

Users with Administrator or Supervisor permissions, can access the Security node and manage users and groups.

Multiple Developers Working on the same Application

ApplinX applications are typically developed by more than one user. This can sometimes cause conflicts on the ApplinX server. Working methodologically and investing time and effort in planning the development and design of the application can help prevent such conflicts:

- Divide responsibilities between the developers (such as developers working on specific entity types, or workflows).
- Provide each user with a unique user name and determine permissions according to user names.
- Work with folders: Permissions can be given to specific folders or users. These permissions can be defined for specific entities/processes.

Typical conflicting scenarios and outcomes:

- More than one developer editing the Application Properties: ApplinX will save the changes of the first developer who saves the changes.
- More than one developer editing an entity:

  When more than one developer edits the same entity, and one of the developers saves the entity, the other developers receive a message indicating that this entity has been saved by another developer. You are required to determine whether you would like to work on the newly saved entity (and update your editor to reflect the newly saved entity) or to continue working on the outdated editor.

  If you choose to continue working on the outdated editor then when trying to save the entity, you will be informed of the name of the user who made the changes and you will be able to decide whether to either:

  - Overwrite the changes that the other developer has made.

- Save the entity with a different name. Note that references that pointed to the original entity will not point to this entity and need to be added manually. References that this entity referred to will be maintained.
- Discard the changes that you made.

New User Properties

The *New User* dialog box is used to define new users, their permissions and passwords. Access this dialog box by selecting **Management>Security>Users** in the ApplinX Explorer and then clicking on the **New** icon in the Toolbar. The *New User* dialog box is displayed. Fill in the **Name**, **Full Name** and **Description** and define associated groups and permissions.

Note: If you do not associate a group to a user, the user will, by default, be associated with Everyone.

Name
The unique identifier of the user. Can contain only digits, English letters (upper or lower case), underscore and spaces. (Obligatory field)

Full Name
The full name of the user.

Description
A suitable description of the user.

Associated groups
The user belongs to these groups. If you do not associate a user with a group, the user will, by default, be associated with Everyone.

Add
Allows you to add one or more groups to the list of groups associated with the user.

Remove
Allows you to remove a group from the list of associated groups by first selecting the group name and then clicking on the Remove button.

Permissions
Displays a dialog box where the folders the user can view and/or edit are defined. Refer to **Defining User Permissions**.

Password
User password, required when accessing ApplinX. Refer to **Defining Passwords**.

Account is disabled
Determines if the account will be disabled.

System Administrator
When checked, provides the user/group with System Administrator permissions.

Defining User Permissions

To view or edit a user's permissions

1 Select Management>Security>Users in the ApplinX Explorer and select the relevant user. The User dialog box is displayed.

2 Click Permissions. The User Permissions dialog box is displayed.

3 To add a permission, click the Add button. The Select Folder dialog box appears. Select from the list of applications or folders in order to define the user's permissions for this application or folder and click OK.

ApplinX
Top-level permission for all ApplinX features and operations.

Management
Permission for runtime monitoring and managing.

Applications (previously Composer)
Development permission for all applications.

<Application Name>
Per application permission.

Note: The Administrator's permissions cannot be changed.

4 Check the Edit or View check boxes to change the selected permissions level.

5 To remove a permission, select a permission and click Remove.

Defining Passwords

To change a user's password

1 Select Management>Security>Users node in the ApplinX Explorer.

2 Double-click on the relevant user or define a new user. The User dialog box is displayed.

Note: It is highly recommended to changing the Administrator's password often.

3 In the User dialog box click on the Password button. The User Permissions dialog box is displayed.

4 In the New password field, enter the new password.

5 In the Confirm new password field, enter the new password again and click OK.

Disabling a User's Account

To disable a user's account

1 Select Management>Security>Users node in the ApplinX Explorer.

2 Double-click on the relevant user. The Information dialog box is displayed.

3 Click the Account is disabled check box to disable a user account.

Adding a New Group

To add a new group

1 Select Management>Security>Groups node.

2 Either click the New button on the toolbar or right-click the Groups node and select New. The New Group dialog box appears.

3 Fill in the Name (can contain only digits, English letters (upper or lower case), underscore and spaces) and Description and define the users registered in this group.

4 Click the Add or Remove buttons to add or remove users to or from this group. There are four built in users.

Administrator
Built-in account for administering the ApplinX Server.

sysDeveloper
Built-in account for configuring and developing the ApplinX Server.

sysOperator
Built-in account for monitoring and managing the ApplinX Server.

sysUser
Built-in account with all administrator rights except managing groups and users.

Note: System (pre-defined) groups and users cannot be deleted.

There are a number of predefined groups:

Everyone
System group that includes all ApplinX users.

Developers
System group with full access to all the applications on the ApplinX Server.

Supervisors
System group with complete and unrestricted access to the ApplinX Server.

Note: System (pre-defined) groups and users cannot be deleted.

4 Administrative Web Services API

ApplinX Administrative Web Services API provides the developer the capability to retrieve data and perform actions based on information received from ApplinX server in runtime, without accessing the ApplinX Designer or Administrator but rather using standard Web services.

Using these Web services you can:

Note: The user name required in this method refers to a user who has ApplinX server Administrator permissions.

Retrieving Server Information

This Web service retrieves data regarding server information.

URL: **http://localhost:2380/wsstack/services/ServerManager?wsdl**

Note: This address is relevant when working with the WS-Stack in local mode. When working in remote mode, you need to update the server address to reflect the WS-Stack server address.

Method: getServerInformation

Returns a `ServerInformationResponse` object which contains the ServerInfomation object.

Format: `ServerInformationResponse getServerInformation(ServerInformationRequest request)`

ServerInformationRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

ServerInformationResponse parameters:

String version
The ApplinX server’s version.

String startedTime
The date and time the server was last initialized.

int activeSessions
The number of currently active sessions.

int idleSessions
The number of sessions currently idle (connected, but detached).

int processingSessions
The number of sessions currently performing actions against the host.

int totalSessions
The total number of sessions opened since the server was started.

int startedThreads
The number of threads that were activated.

int idleThreads
The number of threads that were activated, but are currently not being used.

long allocatedMemory
The total amount of memory (in bytes) that is currently allocated to ApplinX server on the Java Virtual Machine.

long freeMemory
The amount of memory (in bytes) currently available for ApplinX to use.

Method: getSessionsCounters

Returns a `sessionsCountersResponse` object which contains the ConnectionsPerLicense object.

Format: `sessionsCountersResponse getSessionsCounters(sessionsCountersRequest request)`

sessionsCountersRequest parameters:

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

sessionsCountersReponse parameters:

Each of the parameters includes the number of connections currently being used, the maximum number of connections ever connected at one time and the date and time that this occurred. Notice that each of the parameters relates to a specific type of connection.

numberOfWebEnablementDisplay
Relates to Web enablement connections (excluding printer sessions).

numberOfWebEnablementPrinter
Relates to the printer sessions. The printer sessions are counted as Web Enablement connections.

numberOfSOAEnablement
Relates to the SOA enablement connections (excluding Web integration and RPC connections).

numberOfWebIntegration
Relates to Web Integration connections. The Web Integration connections are counted as SOA connections.

numberOfRPCConnections
Relates to RPC connections. The RPC connections are counted as SOA connections.

numberOfUnassignedPoolConnections
Relates to the unassigned connections in the connection pool.

Retrieving Session Information

This Web service retrieves data regarding session information.

URL: **http://localhost:2380/wsstack/services/SessionManager?wsdl**

Note: This address is relevant when working with the WS-Stack in local mode. When working in remote mode, you need to update the server address to reflect the WS-Stack server address.

Method: getAllSessions

Returns a object which contains a list of all the sessions that are open on the server.

Format: `GetAllSessionsResponse getAllSessions(GetAllSessionsRequest request) getServerInformation(ServerInformationRequest request).`

GetAllSessionsRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

GetAllSessionsResponse Parameters

Session [ ] sessions
An array of **Session** objects which contain the session information.

Method: getApplicationSessions

Returns a `GetApplicationSessionsResponse` object which contains a list of sessions that are connected to a specific application.

Format: `GetApplicationSessionsResponse getApplicationSessions(GetApplicationSessionsRequest request)`

GetApplicationSessionsRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

Session Information Response Parameters

Session [ ] sessions
An array of **Session** objects which contain the session information.

Method: getServiceSessions

Returns a `GetServiceSessionsResponse` object which contains a list of sessions that are connected to a specific Connection pool.

Format: `GetServiceSessionsResponse getServiceSessions(GetServiceSessionsRequest request)`

GetServiceSessionsRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String serviceName
The name of the Connection Pool

GetServiceSessionsResponse Parameters

Session [ ] sessions
An array of **Session** objects which contain the session information.

Method: cancelSession

Returns a `CancelSessionResponse` object which contains a boolean that indicates that the session has been canceled.

Format: `CancelSessionResponse cancelSession(CancelSessionRequest request)`

CancelSessionRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String sessionId
The ID of the session that you would like to cancel.

CancelSessionResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Session Object Attributes

String id
The session’s ID on the ApplinX server.

String description
The session’s description. For example, this may be the session’s computer address.

String type
The type of session: Display session, Printer session, RPC session, Web Integration session or Development session.

String application
The name of the application on ApplinX server to which this session is connected.

String deviceName
Workstation ID, available only in certain protocols.

String state
The current communication status between ApplinX server and the host. Can be either: Idle (connected, not attached), Initializing, Processing (executing an action), Active (attached), or Disconnecting.

String idleTime
The time period a session has not performed a communication activity with the host.

String currentScreen
The name of the current screen. When the screen is not identified this will be UNKNOWN.

String userAddress
The IP address from where the session is connected to ApplinX.

String hostAddress
The IP address of the host.

String replayFile
The GCT file name that is working with this session and the screen number in the GCT file.

String traceFile
The name of the trace file that is tracing the current session.

int bytesSent
The number of bytes sent to the host.

int bytesReceived
The number of bytes received from the host

String serviceName
The name of the connection pool used by the current session. Will return an empty value when no connection pool is used by the session.

Managing Connection Pools

This Web service retrieves runtime data regarding existing connection pools and enables starting and stopping these services.

URL: **http://localhost:2380/wsstack/services/ServiceManager?wsdl**

Note: This address is relevant when working with the WS-Stack in local mode. When working in remote mode, you need to update the server address to reflect the WS-Stack server address.

Method: getApplicationServices

Returns a `GetApplicationServicesResponse` object which contains a list of connection pools and their status, for a specific application.

Format: `GetApplicationServicesResponse getApplicationServices(GetApplicationServicesRequest request)`

GetApplicationServicesRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

GetApplicationServicesResponse Parameters

Service [ ] services
An array of **Service** objects which contain the session information.

Method: getFolderServices

Returns a `GetFolderServicesResponse` object which contains a list of names and status of connection pools that are in a specific folder of an application.

Format: `GetFolderServicesResponse getFolderServices(GetFolderServicesRequest request)`

GetFolderServicesRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String folder
The name of the folder where the connection pools are located.

GetFolderServicesResponse Parameters

Service [ ] services

An array of **Service** objects which contain the session information.

Service Object Attributes

Service attributes:

String name

The name of the connection pool.

String folder

The folder in which the connection pool is placed.

String status

The connection pool's status: Not Started, Initializing, Active, In standby, Suspended, Stopping or Stopped.

int activeConnections

The number of connections currently held by a session (user).

int readyConnections

The number of connections ready for use.

int processingConnections

The number of connections currently in the Processing state.

long averageWaitTime

The average time (in milliseconds) sessions waited for a READY connection (calculated only among those sessions that waited) multiplied by the percentage of waiting sessions. For example: if 8% sessions had to wait, and in average each of those waited 1000 milliseconds, the overall average wait time was: 0.08 * 1000 = 80 milliseconds.

int percentOfWaiting

The percent of sessions that did not immediately get a connection when trying to connect to ApplinX.

int connectionCount

The total number of connections in the connection pool (since the last time the connection pool was started), ignoring broken connections.

int maxConnections

The maximum number of connections that were connected concurrently since the connection pool started.

int sessionCount

The total number of sessions that connected to the host service since the connection pool started.

int maxConcurrentSessions

The maximum number of sessions that were connected concurrently since the connection pool started.

String connectionInfoName
The name of the information set used by this connection pool.

int numberOfCurrentlyWaiting
The number of users currently waiting for a connection.

int numberOfTimeouts
the number of users who received a timeout after a connection was not assigned to them.

int numberOfWaitedUsers
the total number of users who waited for a connection since the connection pool was last started.

long maxWaitTime
Maximum time, since the session started, that a user waited for a connection.

CancelConnectionResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: cancelConnection

Returns a `CancelConnectionResponse` object which contains a boolean that indicates that the connection has been canceled.

Format: `CancelConnectionResponse cancelConnection(CancelConnectionRequest request)`

CancelConnectionRequest parameters

username
The name of the user who has the relevant permissions to access ApplinX server.

password
The user's password

appName
The name of the application on ApplinX server

serviceName
The name of the connection pool

folder
The name of the folder where the connection pools are located.

connectionId
The ID of the connection which you would like to cancel.

CancelConnectionResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: getServiceConnections

Returns a `GetServiceConnectionsRequest` object which contains a list of connections for a specific Connection Pool.

Format: `GetServiceConnectionsRequest getServiceConnections(GetServiceConnectionsRequest request)`

GetServiceConnectionsRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String serviceName
The name of the connection pool

GetServiceConnectionsResponse parameters

ServiceConnection [ ] connections
An array of **ServiceConnection** objects which contain the session information.

ServiceConnection object attributes:

String ConnectionId
The identifier of the connection.

String status
The current status of the connection. Possible values: Active, Ready, Initializing, Recycling, Keep-alive, Terminating or Broken.

String sessionId
Active connections only—displays the ID of the session holding the displayed connection.

String elapsedTime
The amount of time that has elapsed since the session status last changed.

String errorMessage
Displays error messages for broken connections.

String createdTime
The time the connection was created.

String statusChangedTime
The time the status was last changed.

int sessionServed
The number of sessions this specific connection served so far.

String connectionInformation
The connection information set row number used by the connection.

String currentScreen
The name of the current screen.

Method: startService

Starts the connection pool specified in the request and returns a response object with a boolean indicating success or failure.

Format: `StartServiceResponse startService(StartServiceRequest request)`

StartServiceRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String serviceName
The name of the connection pool

String folderName
The name of the folder where the connection pool is located. By default, this is the root folder.

StartServiceResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: stopService

Stops the connection pool specified in the request and returns a `StopServiceResponse` object with a boolean indicating success or failure.

Format: `StopServiceResponse stopService(StopServiceRequest request)`

StopServiceRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String serviceName
The name of the connection pool

String folderName
The name of the folder where the connection pool is located. By default, this is the root folder.

StopServiceResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: stopAllServices

Stops all the connection pools of the application and returns a `StopAllServicesResponse` object with a boolean indicating success or failure.

Format: `StopAllServicesResponse stopAllServices(StopAllServicesRequest request)`

StopAllServicesRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

StopAllServicesResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: suspendService

Suspends the available connection pool of the application and returns a `SuspendServiceResponse` object with a boolean indicating success or failure.

Format: `SuspendServiceResponse suspendService(SuspendServiceRequest request)`

SuspendServiceRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String serviceName
The name of the connection pool

String folderName
The name of the folder where the connection pool is located. By default, this is the root folder.

SuspendServiceResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: StandbyService

Suspends the available connection pool of the application and returns a `standbyServiceResponse` object with a boolean indicating success or failure.

Format: `StandbyServiceResponse standbyService(standbyServiceRequest request)`

standbyServiceRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String serviceName
The name of the connection pool

String folderName
The name of the folder where the connection pool is located. By default, this is the root folder.

StandbyServiceResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: resumeService

Resumes the possibility to use a suspended connection pool of the application and returns a `ResumeServiceResponse` object with a boolean indicating success or failure.

Format: `ResumeServiceResponse resumeService(ResumeServiceRequest request)`

ResumeServiceRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String serviceName
The name of the connection pool

String folderName
The name of the folder where the connection pool is located. By default, this is the root folder.

ResumeServiceResponse Parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Managing Connection Pools Connection Information Sets

This Web service updates the connection information set cell values.

URL: http://localhost:2380/wsstack/services/ConnectionInfoManager?wsdl

Note: This address is relevant when working with the WS-Stack in local mode. When working in remote mode, you need to update the server address to reflect the WS-Stack server address.

Method: getConnectionInfoSet

Returns a `GetConnectionInfoSetResponse` object which contains a `ConnInfoSet` object.

Format: `GetConnectionInfoSetResponse getConnectionInfoSet (GetConnectionInfoSetRequest request)`

GetConnectionInfoSetRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String connectionInfoName
The name of the connection information set

GetConnectionInfoSetResponse parameters

ConnInfoSet connectionInfoSet

Objects which contain the connection information set parameters

ConnInfoSet parameters

ConnInfoColumn[ ] columns

Array of the connection information columns, not including the ID and Repeat columns.

ConnInfoRow[ ] rows

Array of the connection information set rows.

ConnInfoColumn parameters

Boolean hidden

Indicates whether the column is defined as a password column.

int type

Indicates the column type: Variables (0), Application fields (1) and Application and connection parameters (2).

String name

Column name.

ConnInfoRow parameters

ConnInfoCell[] cells

Array of the cells in a specific row in the connection information set not including the ID and Repeat cells.

int Repeat

The value of the repeat cell.

int id

The value of the ID cell.

ConnInfoCell parameters

String value

The cells's value.

String columnName

The cell's column name.

int columnType

Indicates the cell's column type: Variables (0), Application fields (1) and Application and connection parameters (2).

Note: The user name required in this method refers to a user who has ApplinX server Administrator permissions.

Method: updateConnectionInfoRow

Returns a `UpdateConnectionInfoRowResponse` object which contains a boolean indicating whether the update operation succeeded.

Format: `UpdateConnectionInfoRowResponse updateConnectionInfoRow(UpdateConnectionInfoRowRequest request)`

UpdateConnectionInfoRowRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String connectionInfoName
The name of the connection information set

ConnInfoRow row
The row object that contains the information that we wish to update in the connection information set. The following parameters must be set:

- The ID of the row as it appears in the server.
- The repeat number: when either updating the repeat number or preserving the number currently set on the server (when this number is not zero).
- The value of each cell in the ConnInfoCell[]. Caution: Null values will replace existing values on the server.

UpdateConnectionInfoRowResponse parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: addConnectionInfoRow

Format: `AddConnectionInfoRowResponse addConnectionInfoRow(AddConnectionInfoRowRequest request)`

AddConnectionInfoRowRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

String connectionInfoName
The name of the connection information set

ConnInfoRow row
The row object that contains the information that we wish to add to the connection information set. See the following guidelines for setting the object parameters:

- The ID of the row as it appears in the server.
- The repeat number: when either updating the repeat number or preserving the number currently set on the server (when this number is not zero).
- The value of each cell in the ConnInfoCell[]. Caution: Null values will replace existing values on the server.

AddConnectionInfoRowResponse parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Method: removeConnectionInfoRow

Format: `RemoveConnectionInfoRowResponse removeConnectionInfoRow(RemoveConnectionInfoRowRequest request)`

RemoveConnectionInfoRowRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String application
The name of the application on ApplinX server

String connectionInfoName
The name of the connection information set

int rowed
The row ID

RemoveConnectionInfoRowResponse parameters

boolean isSuccessful
Returns true or false, to indicate the success or failure of the action.

Managing RPC Connection Pools

This Web service retrieves runtime data regarding existing RPC connection pools and enables resetting the pool.

URL:http://localhost:2380/wsstack/services/ProgramPoolManager?wsdl

Note: This address is relevant when working with the WS-Stack in local mode. When working in remote mode, you need to update the server address to reflect the WS-Stack server address.

Method: getPoolInformation

Returns a `PoolInformationResponse` object which contains runtime information regarding the pool manager. `resetPool(ProgramPoolRequest request)` resets the pool.

Format: `PoolInformationResponse getPoolInformation(ProgramPoolRequest request)`

ProgramPoolRequest parameters

String username
The name of the user who has the relevant permissions to access ApplinX server.

String password
The user's password

String appName
The name of the application on ApplinX server

PoolInformationResponse parameters

int availableConnectionsCount
The number of available connections in the pool.

int maxPoolSize
The maximum number of connections that can exist in the pool concurrently.

int excludedCount
The number of resources that were invalidated but are still in use

int minPoolSize
The minimum number of connections that must exist in a pool.

int poolSize
The number of connections that currently exist in the pool.

int awaitingCheckinCount
The number of used resources (including the excluded resources).

Method: resetPool

This method restarts the pool.

Format: `resetPool( )`

5 Batch Automation Utilities

You may be in a situation where a GUI interface is not available. This may be a common problem when a Unix station is used. In other cases, it may be more convenient to double-click a batch file to activate the Mapping Utility via the Administrator. The batch files can be found in the utilities directory in ApplinX installation directory.

Migration Batch File

ApplinX applications from previous ApplinX versions are able to run on the current ApplinX version once you have performed a number of migration activities. Migration can be performed when installing the new ApplinX server version or via batch files after the new server has been installed. You can:

- Migrate all ApplinX applications, when installing/upgrading ApplinX.
- Migrate all ApplinX applications via batch files, after ApplinX has been installed.
- Migrate a specific ApplinX application, after ApplinX has been installed.

To migrate all ApplinX applications after installing/upgrading ApplinX you must run migrate_ApplinX_server.bat/sh file, enter the installation path of your previous installation, and then enter the new path of the current installation.

For example: `migrate_ApplinX_server.bat "c:\ApplinX52" "c:\SoftwareAG\ApplinX"`

Refer to Migrating Applications from Previous ApplinX Versions

Convert Utility Batch File

This utility is used to convert gxz files which include the application configuration from previous ApplinX versions, to gxar files.

To activate the Convert Utility batch file using the command prompt window:

1. Open a command prompt window.
2. Change the current directory to the relevant directory.
3. Type convert_gxz_to_gxar followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.). For example: -u Administrator -f "C:\Documents and Settings\sagsis.gxz"

Parameter	Description	Default
-s	Server address	127.0.0.1
-p	Server port	2323
-u	User name	
-w	Password	
-f	File name	

Importing Screens using a Batch File

Standard maps, such as Natural, BMS and MFS, are used in host applications and include the screen data such as static data and dynamic fields. ApplinX enables importing these application maps, saving time and effort spent on manually identifying screens and simplifying the update process when changes are made in the host. When importing application maps, a screen is automatically created from each map, minimizing errors that may occur when creating the screens manually, one by one. The ApplinX screen created includes identifiers (based on the static data) and fields (based on the dynamic data). ApplinX supports a number of different types of maps:

- Natural: Natural map support (from Systrans file).
- BMS: CICS basic map support.
- MFS: IMS message format service.
- SDFX: ApplinX generic map format, used for other standard maps. To create SDFX files refer to SDFX File Format Definition.
- SDF: Compatible with Software AG's JIS product.

The import map feature can be used to import an application's maps for a new application or to maintain and update previously imported maps. When updating previously imported maps, screen identifiers will be deleted and replaced, existing fields will be updated with their new positions and their references to other entities will be preserved. Fields that were previously imported, but no longer exist on the host will be deleted.

Note: Invalid entity names, such as names which include invalid characters such as "#" or begin with a digit, will be automatically corrected by omitting the invalid characters.

Maps can be imported either using the Import Host Screen Maps wizard or using a batch file.

To import screens via a batch file (using the command prompt window):

1 Open a command prompt window.

2 Change the current directory to the <ApplinX home>/Utilities directory.

3 Type screen_import.bat/sh followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

Parameter	Description	Default
-s	Server address	127.0.0.1
-p	Server port	2323
-u	ApplinX user name (Required parameter)	
-w	ApplinX user password	Empty by default
-a	ApplinX application name (Required parameter)	
-f	File name, or directory name (when importing more than one file). Required parameter	
-x	The file extension. All files from the given directory that have this extension will be loaded (when not specified, the default extension for the map type is used)	
-af	ApplinX target folder within the application repository.	Root folder
-t	Map type. Possible values: "sdf", "sdfx", "natural", "bms", "mfs" (required parameter).	natural
-m	Indicates where the error line is located: "first", "last", "lastm1" (last minus 1), "lastm2", "lastm3", "lastm4" (Natural maps only)	last
-mf	Message line field name (Natural maps only)	MessageLine
-k	Don't skip map with write command. (Natural maps only)	true (skip)

The screens created appear in the directory you determined in the **Target folder** field. The names of the screens are identical to the map names.

The report is displayed in the Eclipse console and includes a list of the screens added as well as the fields and tables created/updated/deleted.

Export Batch File

To activate the Export batch file by using the command program prompt:

1 Open a command prompt window.

2 Change the current directory to the <ApplinX home>/Utilities directory.

3 Type exportapp followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

exportapp Parameters

Parameter	Description	Default Value
-s	Server address	127.0.0.1
-p	Server port	2323
-u	User name	
-w	Password	
-a	Application name	
-f	The target folder and/or file name. ■ When only the path is specified (<pathname>followed by "\"), the file name is the application name. ■ When only the file name is specified, the file is created in the current location.	<current location>\<application name>
Include one of the following parameters:		
-c	Export only the application configuration (gxar file)	
-e	Export only entities (gxz file)	
-l	Export the application configuration and the entities (gxar file)	

Examples for the -f parameter

- **Specify target folder and file name:** `exportapp.bat -s localhost -p 2323 -u administrator -a InstantDemo -f C:\SoftwareAG_821_GA\ApplinX\utilities\export\MyApplication -c`

  The file MyApplication.gxar is created in the C:\SoftwareAG_821_GA\ApplinX\utilities\export\ direrctory.

- **Specify target folder only:** `exportapp.bat -s localhost -p 2323 -u administrator -a InstantDemo -f C:\SoftwareAG_821_GA\ApplinX\utilities\export\ -c`

  The file InstantDemo.gxar is created in the C:\SoftwareAG_821_GA\ApplinX\utilities\export\ directory

- **Specify file name only:** `exportapp.bat -s localhost -p 2323 -u administrator -a InstantDemo -f MyApplication -c`

  The file MyApplication.gxar is created in the current local directory.

- **Default when the parameter is not specified:** `exportapp.bat -s localhost -p 2323 -u administrator -a InstantDemo -c`

  The file InstantDemo.gxar is created in the current local directory.

Examples for the -c, -e and -l parameters

- **Export application configuration only:** `exportapp.bat -s localhost -p 2323 -u administrator -a InstantDemo -f C:\SoftwareAG_821_GA\ApplinX\utilities\export\ -c`

  The file InstantDemo.gxar is created.
- **Export entities only:** `exportapp.bat -s localhost -p 2323 -u administrator -a InstantDemo -f C:\SoftwareAG_821_GA\ApplinX\utilities\export\ -e`

  The file InstantDemo.gxz is created.
- **Export application configuration and entities:** `exportapp.bat -s localhost -p 2323 -u administrator -a InstantDemo -f C:\SoftwareAG_821_GA\ApplinX\utilities\export\ -l`

  The file InstantDemo.gxar is created.

Import Batch File

To activate the Import batch file by using the command program prompt:

1. Open a command prompt window.
2. Change the current directory to the <ApplinX home>/Utilities directory.
3. Type importapp followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

```
importapp.bat/sh [-a [attribute]] [-s [attribute]] [-p [attribute]] [-o [attribute]]
[-u [attribute]] [-f [attribute]] [-w [attribute]]
```

Note: When wishing to importing a complete ApplinX application you will require a gxar (ApplinX application archive) file. This file includes the application configuration, ApplinX entities (as a read only <gxz> file) and a trace file. When importing only the application's entities, you require the gxz file only.

importapp Parameters

Parameter	Description	Default Value
-a	ApplinX application name [required].	
-s	ApplinX Server address.	localhost
-p	ApplinX Server port.	2323
-o	One of the following operations can be performed: ▪ x - Import entities from a gxz file, overriding conflicting entities.	

Parameter	Description	Default Value
	■ c - Import application from a gxar file, retaining the existing host configuration. The repository will be read-only. ■ h - Import application and host configuration from a gxar file (overriding existing host configuration). The repository will be read-only. ■ r - Import application and host configuration. When importing, retain the repository configuration (import the gxz within the gxar, to the currently configured repository). ■ hr - Perform both 'h' and 'r' operations. When not set, entities and/or configuration will be imported to an existing application when one exists, or to a new application when there is no existing one.	
-u	ApplinX user [required].	
-f	The path and name of the gxz/gxar file [required].	
-w	ApplinX user password.	

Note: The Session Data entity will be merged with the existing Session Data entity. When there is a conflict between the imported to the existing Session Data entity, your selection in this checkbox will determine how the Session Data entity will be.

Examples:

```
importapp.bat -u Administrator -a app2 -f c:\entities.gxz -o x
```

This command imports entities into the "app2" application (only if it exists), overriding any conflicting entities within the application.

```
importapp.bat -u Administrator -a app1 -f c:\app.gxar -o c
```

This command imports the host and application configuration, from the provided gxar file as a new application. The repository will be read-only.

```
importapp.bat -u Administrator -a app1 -f c:\entities.gxar -o h
```

This command retains the existing host configuration, and imports the "app1" application (entities and configuration) from the provided gxar file, overriding any conflicting entities within the application. The repository will be read-only.

```
importapp.bat -u Administrator -a app1 -f c:\entities.gxar -o r
```

This command imports the host configuration, and the "app1" application (just the configuration) from the provided gxar file, and imports the entities into the repository, using the existing repository configuration. The repository will not be read-only.

```
importapp.bat -u Administrator -a app1 -f c:\entities.gxar -o hr
```

This command imports the host configuration, retains the application configuration, and imports the "app1" application entities from the provided gxar file to the existing repository, overriding any conflicting entities within the application. The repository will not be read-only.

Extracting Activities from Trace Files

The Screen Process Extractor is used to analysis activities performed by the user and make business decisions based on this analysis. As part of the process, you will need analyze sessions/trace files. This is carried out either via a wizard in the Designer, typically used when developing, or via a batch file, typically used for the production environment. The following instructions detail how to extract activities from the trace files via a batch file (using the command prompt window)

1. Open a command prompt window.
2. Change the current directory to the <ApplinX home>/Utilities directory.
3. Type extract_activities.bat/sh followed by your required parameters. (The minus sign and letter should precede the value to distinguish between the parameters. The order of the parameters is not significant.)

Parameter	Description	Default
-s	Server address	127.0.0.1
-p	Server port	2323
-u	ApplinX user name (Required parameter)	
-w	ApplinX user password	Empty by default
-a	ApplinX application name (Required parameter)	
-x	The name of the XML file to use for the analysis process.	Default.xml
-d	A semi-colon of folders and gct files to analyze	The "TraceFilesLocations" specified in the xml file