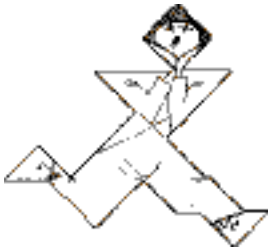


Hay River Software LLC

ForXs2gApps™ User Management agent for Google Apps for Education®

Hay River Software LLC N14397 380th St, Ridgeland WI 54763 **T** 715 455-1652 george@hayriv.com http://hayriv.com

Legal Notice

Hay River Software LLC N14397 380th St, Ridgeland WI 54763 **T** 715 455-1652 **george@hayriv.com** **http://hayriv.com**

ForXs2gApps

User Management agent for Google Apps for Education.

Introduction

Important Note: This documentation is for the beta version of ForrXs2gApps. It has been adapted from the documentation for the original ForXs agent for Macintosh OS X server. There may be errors in this document, missing information or statements that don't apply to this beta agent. If any discrepancies are found please contact george@hayriv.com.

About ForXs2gApps agent

ForXs2gApps is tool to automate user creation in Google Apps for Education or Google Apps for Enterprise. In a school district (K-12) that has implemented SIF (School Interoperability Framework) to exchange information between management systems the source of information will be from the SIF Zone server. In all other situations the agent will get information about users by periodically querying a SQL table with staff and/or student information for additions, changes or deletions.

As a SQL based management agent

ForXs2gApps queries a SQL database periodically to get at least first name, last name and a unique number, usually a student number or staff ID . It uses this information to create and manage user accounts in Google Apps for Education. Configuration settings allow the agent to accept usernames and passwords in the queries or it can create unique usernames and passwords as needed. The frequency of the queries can be configured anywhere from every second to daily.

As a SIF user management agent

ForXs2gApps registers with a SIF Zone(s) and accepts information sent to the agent from the zone(s) about staff and students. It uses this information to create and manage user accounts in Google Apps for Education. Configuration settings allow the agent to accept usernames and passwords sent to it from the Zone or it can create unique usernames and passwords as needed.

Up to the minute addition of student and staff user accounts

As new students are enrolled or existing student's information changed that information is passed on through the zone to ForXs2gApps. ForXs2gApps adds or changes the user accounts as it receives the information. The process can take a bit more than a minute but less time than it takes a new student to walk from the office to the computer lab.

SIF status

ForXs2gApps was designed to work within a school district using the School Interoperability Framework. It works with SIF version 1.5r1 and 2.0. It was built using the Java Agent Development Kit from Edusturctures LLC. At this time the agent is NOT "SIF Certified" and Hay River Software LLC is NOT a member of SIFA, School Interoperability Framework Association.

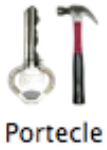
Planning and Configuration

Requires a computer with Java 1.5 or newer installed.

Installing ForXs2gApps and helper applications

ForXs2gApps Package (required)

ForXs2gApps and the helper applications can be downloaded from http://www.hayriv.com/ForXs2gApps/downloads.html. The files are in .zip format. They should expand automatically if you are running Macintosh System 10.4 or newer. By default ForXs2gApps is installed in the Applications folder in a folder named ForXs2gApps2 X.XX. Where the Xs are replaced with the version number of ForXs2gApps.



portecle Package (optional)

Portecle is an Open Source Java application that deals with certificates. It is needed for ForXs2gApps when a Zone requires a signed certificate. If a certificate is supplied by the Zone, portecle is used to place the certificate in a secure key store file. If a certificate is needed to be supplied to the Zone, portecle can be used to create the certificate. If the Zone uses the http protocol and does not require a certificate portecle need not be install. Portecle is a separate application and may be installed on any 10.3 or 10.4 computer. Information and source code is available at http://portecle.sourceforge.net/

Lingon (optional)

"Lingon is a graphical interface for creating launchd configuration files and controlling them through launchctl for Mac OS X Tiger." It is used as an easy way to set up ForXs2gApps to run as a daemon. It is available for download at http://lingon.sourceforge.net/
See the section on "Running ForXs2gApps as a Daemon" at the end of this document for instructions for using Lingon with ForXs2gApps.

Activating, Running, and Deploying ForXs2gApps

Minimal required setup.

The following steps need to be completed before users can be created using ForXs2gApps.

License Key

An Application License Key is required to use the ForXs2gApps application. The file name is ForXs.key. The key is an encrypted file that unlocks the application and will work only for your computer. This key file needs to be located in the same folder as the ForXs2gApps application. If not found there the application will ask for its location and move it for you. Check out the ForXs2gApps License page for more information and instructions for obtaining a key. http://hayriv.com/ForXs2gApps/license.html

Setting Preferences for ForXs2gApps

All preferences are set by editing a XML file with the default name of config.xml. Instructions for editing the file follow on this page or in later sections of this document.

Important Note

Test this software with

```
<property name="gmailAllowCreation" value="false"/>.
```

This will allow the populating of the internal database before accounts are actually created in Google Apps for Education. For security reasons a username once deleted in Google Apps can not be recreated for 5 days.

Do not test this software on a server that is being used for active production purposes. This agent will reset the password for an existing user if that username already exists which would make the user unable to log in to their account.

Prerequisites

1. A functioning SIF Zone and a SIF enabled Student Information System
or a SQL table with staff or student information.
2. The ForXs2gApps software, available for download at http://hayriv.com/ForXs2gApps/downloads.html
3. A license key for the ForXs2gApps software, available at http://hayriv.com/ForXs2gApps/license.html
4. Java version 5 installed (1.5) On a Apple OS X 10.4 or higher this should be installed by default if the normal Apple Software Updates are up to date.

Partially configured files

A basic configuration file is provided for these student information systems in the configuration folder of ForXs2gApps;
Infinite Campus
Power School
SASI

If you are not using SIF then choose the SQL configuration file.

Replace the config.xml file in the ForXs2gApps/config folder with the file associated with your system.

Beginning of the configuration file.

```
1<?xml version="1.0" encoding="UTF-8"?>
2<agent id="ForXs_Polk" sifVersion="1.5r1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3<zone id="Polk" template="Default" url="http://example.org:80/sif/interchange/sif/Polk"/>
4
```

Zone connection configuration.

In line 2 the agents id needs to be set. Change "ForXs_Polk" to something that will make sense when viewed in your SIF Zone software. This can be anything but including ForXs2gApps in the name identifies the agent being an Google Apps agent.

Line 3 has the minimal information needed to connect to your SIF environment. the id= is the name of the SIF Zone which this agent will connect to. It is case sensitive so "polk" or "POLK" would not work.
The url= can normally be found and copied from the zone administration software. An example from PowerSchool is

```
<zone id="district" template="Default" url="http://10.0.25.200:7080/district"/>
```

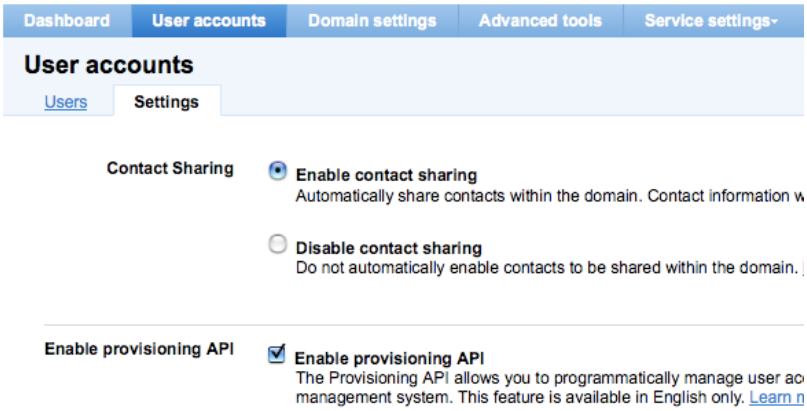
Google Apps for Education connection configuration.

Google config.xml file settings here;

```
<property name="gmailAdminEmail" value="admin@myschooldomain.k12.mn.us"/>

<property name="gmailAdminPassword" value="theAminPassword"/>

<property name="gmailDomain" value="myschooldomain.k12.mn.us"/>

<property name="gmailSetChangePasswordAtNextLogin" value="true"/>

<property name="gmailAllowCreation" value="true"/>
```

The gmailAdminEmail address is the full email address of an administrtor including the @domain..
gmailAdminPassword is the admin's password.
gmailDomain is the portion of the email address after the @ for your domain.
gmailSetChangePasswordAtNextLogin when set to true will force users to change their password on the next log in. This is a best practice.
gmailAllowCreation will need to be set to true to allow users to be created but this should be set to false when first testing the agent. This will allow you to verify that usernames and passwords are being generated in a manner that you want before creating many users that will need to be deleted.

The provisioning API for your Google Apps for Education also needs to be turned on before this agent can communicate and create users. To turn this on log into your Google domain's management web page. Select the "User accounts" tab and then the "Settings" sub tab. Click the "Enable provisioning API" checkbox and save your changes. The picture below shows what this page looks like.



Internal database configuration.

By default the agents uses a built in HSQLDB database that holds information as it comes in from the SIF environment. There is no need to change these settings if you will be using SIF.

```
    <property name="hibernate.connection.driver_class" value="org.hsqldb.jdbcDriver"/>

    <property name="hibernate.connection.url" value="jdbc:hsqldb:work/ForXsdata"/>

    <property name="hibernate.connection.username" value="sa"/>

    <property name="hibernate.connection.password" value=""/>

    <property name="hibernate.connection.dialect" value="org.hibernate.dialect.HSQLDialect"/>
```

If you will not be using SIF to get information and will be writing SQL queries on your own database of users then you will need to change the ForXs database and configure it to use the same type database with a user that has rights to read your database. An example of connecting to a MS-SQL database is listed below. The ForXs data and your user data can reside in different databases as long as those databases can be accessed with a join in your SQL query.

```
    <property name="hibernate.connection.driver_class" value="net.sourceforge.jtds.jdbc.Driver"/>

    <property name="hibernate.connection.url" value="jdbc:jtds:sqlserver://yourDatabaseIPorURL:
1433/databasename"/>

    <property name="hibernate.connection.username" value="yourUsername"/>

    <property name="hibernate.connection.password" value="yourPassword"/>

    <property name="hibernate.connection.dialect" value="org.hibernate.dialect.SQLServerDialect"/
>
```

Save config.xml

Save the configuration file as config.xml in the config folder is the same folder as ForXs2gApps.jar file.

Further configuration after initial testing.

There are many more options that can be changed to make the agent create users various settings and to control the way the agent interacts with the Zone. The default settings in this file should work for initial testing purposes. After the

initial testing is successful read through all the other sections in this document before creating all your user accounts. These sections go into greater detail for options available.

Testing ForXs2gApps

Prerequisites

ForXs2gApps software has been downloaded and configured per the instruction on the previous page.

License file.

Start up the agent. After starting up it will ask for the license file. Browse the hard drive for the file and select it.
Quit the agent. Restart the agent.

Check Connections

Check the Status window to see if connections to the Google server, database and SIF Zone all worked. Messages similar to the following should appear.

Internal database
Successful connection to the internal database.
"Connecting to internal database"
followed by no error messages.
Zone connection
Successful connection to the Zone.
"Connecting to zones...
- Connecting to zone "Polk" at http://example.org:80/sif/interchange/sif/Polk **not** followed by
" Could not establish a connection to the ZIS (http://example.org:80/sif/interchange/sif/Polk): java.net.ConnectException: Operation timed out"
or some other error.
If the agent is connecting to multiple zones each zone will have its own line above.

Google Apps connection

A successful connection to the Google server will produce the following message in the ForXs.log file and in the Status window.
"Successfully authenticated to Google Apps feed services." An error message will appear in it's place if the settings are not correct or if the Google provisioning API has not been enabled.

Failed connections?

If any of the three connections were not successful take note of the error messages, quit the agent and make adjustments to the config.xml file.
If the agent didn't show any windows or you need more information on the errors, open and check the end of the console.log and the ForXs.log in the ForXs2gApps/work/ folder for more details.

3 successful connections

If all connections were made the agent will need to be authorized by the Zone. The agent sends a registration request to the zone when it starts up. This gives the zones some basic information on what type of information it will be requesting, how it should be sent, how large any one packet can be and the SIF version being used.
This information needs to be approved by the Zone administrator. Once this is done move on to the next step.

Test the connection to the Zone.

The SIF enabled student information system will normally send out SIF objects or packets of information as they are created or changed. If you are connected to an active, production environment these might start coming through.
These can provide valuable information on the format of the XML messages sent and if the agent is interpreting them correctly.
The agent can also query the zone requesting existing information.
The simple way to start is to ask for all the information on the connected schools.

First zone test

SchoolInfo

Startup the ForXs2gApps agent. To get the SchoolInfo do the following;
Choose the Window >> SIF Query Window menu item.

Create and send a SIF query to the SIF Zone by
Choosing "Query all zones from the SIF Zone drop down list.
Choosing "SchoolInfo" from the SIF Object drop down list.
Leave the SIF Element at "Select a field"
Leave the SIF Element Value text box empty.
Click on the "Submit Query" button.

In the Status window a message requesting the information will appear first followed a minute or so later by a list of connected schools.

This is a sample of the information;
SchoolInfo SIF_Request sent to zone "Polk"
Requesting SchoolInfo from the Polk zone.
Processing SchoolInfo Response to Query
Receiving SchoolInfo Event
Connected to these schools:
SchoolInfo HashMap is
{LocalId=17, PhoneNumber=, StatePrId=0498, SchoolName=Polk Middle School, NCESId=498, RefId=000009A10000BDF19346479400000011}
RefId is 000009A10000BDF19346479400000011
Existing school record found in DB
Polk Middle School

If at least one school appeared the SIF SchoolInfo object connection and the basic SIF Zone connection is working. If not use the error messages to determine what needs to be changed.

Second zone test

StaffPersonal

The next test will be to create users on the Google Apps server for a few or all staff members.

Startup the ForXs2gApps agent. To get the StaffPersonal information do the following;
Choose the Window >> SIF Query Window menu item.
Create and send a SIF query to the SIF Zone by
Choosing "Query all zones from the SIF Zone drop down list. (Or choose a single zone by name)
Choose "StaffPersonal" from the SIF Object drop down list.
Leave the SIF Element at "Select a field" (If your zone has many staff members select "Name/LastName")
Leave the SIF Element Value text box empty or to limit the number of staff members imported, type in the last name of a staff member or last name that has a few by that name.
Click on the "Submit Query" button.

Staff should show up in the status window and users for them should be created on Google Apps domain.
The default settings are to create usernames and passwords for each user and not to ask the zone for this information.
The properties in the config.xml file control the format of usernames and how passwords are created. The agent can also be set to receive usernames and passwords from the SIF Zone if there is an agent that can provide them.

Internal database testing and troubleshooting

The instructions on the following pages will allow you to view the information saved about the staff in the internal database. If no users were created or some are missing, viewing the contents of the database might help determine why they were not created. If a staff member is missing an "otherid", a password or username the agent will not attempt to create a user.
http://hayriv.com/ForXs2gApps/passwordDB.html
http://hayriv.com/ForXs2gApps/internaldb.html

Third Zone test

Student information testing

Getting all the needed information to create a student is more complicated than it is for staff members. Students are normally assigned to an email list based on their school enrollment and can have their accounts disabled when they are

no longer enrolled. This information is not stored in the StudentPersonal object but is in the StudentSchoolEnrollment object. The agent can be configured to automatically create a SIF request for missing information. By default this is turned off but will need to be turned on before enough data can be collected to create a student user on your Google Apps domain. The following property will need to be changed to true;

```
<property name="autoQueryAsNeeded" value="false"/>
```

Query for a limited set of students

Startup the ForXs2gApps agent. To get the StudentPersonal information do the following;

1. Chose the Window >> SIF Query Window menu item.
2. Create and send a SIF query to the SIF Zone
3. Choose a single zone by name
4. Choose "StudentPersonal" from the SIF Object drop down list.
5. Choose "Name/LastName" from the SIF Element drop down list.
6. Type in the last name of a student known to be currently enrolled in the zone being queried. Using a common last name should result in receiving a record for each student with that last name.
7. Click on the "Submit Query" button.
   Watch for results

The following should happen.
The agent should report that it is receiving a StudentPersonal event. After a 60 second delay the agent should create a SIF request for StudentSchoolEnrollment and note that in the Status window.
Following another delay the StudentSchoolEnrollment should be received and users should start to be created just as the staff were created.

Synchronizing Current users with SIF

Minimal setup required.

The agent should be fully set up and tested per the directions in deployment and testing .

Google Apps preparation.

TODO

ForXs2gApps internal database prep.

The following query can be executed to delete all the data from previous testing.

```
DELETE FROM userinfo
```

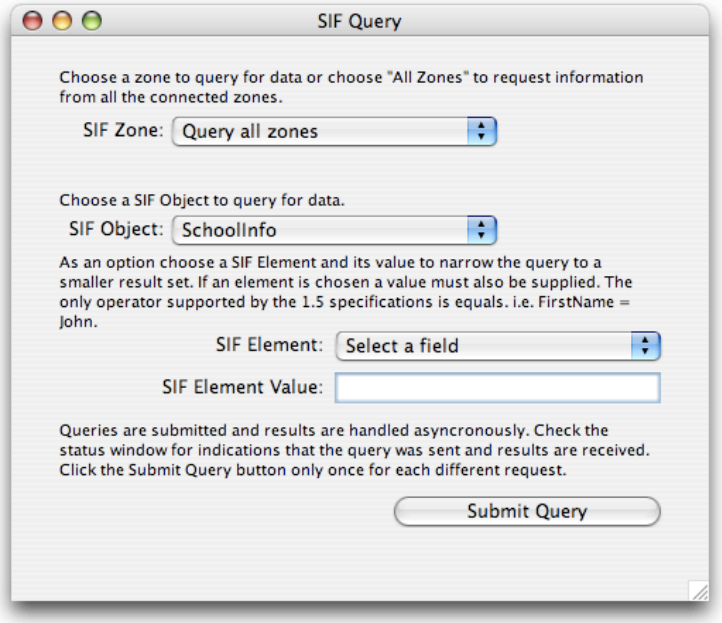
The SQL command above can be typed into the “Search PW DB” window in the ForXs2gApps application. Open this window by choosing the Window >> Search PW DB... menu item.

Synchronization order and timing.

The query for SchoolInfo needs to be done first so that appropriate email lists for each school are created before users are created. The rest of order listed below is just a suggestion and is not required by the agent. They are listed in the order from smallest to the largest expected return information. Only query for those groups you want as users in the server. These queries can generate large number of XML messages. This can potentially bog down the network, the zone server or the student information system. To avoid these potential problems it is wise to do the queries one at a time and during off-peak hours.

Query for SchoolInfo

Use the SIF Query window to request SchoolInfo. Leave the SIF Element set at "Select a field" so all schools are requested.



Query for StaffPersonal

Use the SIF Query window to request StaffPersonal. Leave the SIF Element set at "Select a field" so all staff members are requested.

Query for EmployeePersonal

Use the SIF Query window to request EmployeePersonal. Leave the SIF Element set at "Select a field" so all employees are requested. This SIF object would only be used if the StaffPersonal was not available or if the data provided was more accurate than the data available from StaffPersonal.

Query for StudentPersonal

Use the SIF Query window to request StudentPersonal. Leave the SIF Element set at "Select a field" so all students are requested. If the property autoQueryAsNeeded is set to true the following query for StudentSchoolEnrollment need not be run.

Query for StudentSchoolEnrollment

Use the SIF Query window to request StudentSchoolEnrollment. Leave the SIF Element set at "Select a field" so all student's enrollments are requested. If the property autoQueryAsNeeded is set to true and the StudentPersonal object has been requested there should be no need to do this query.

Authentication (username and passwords)

This query only needs to be run if autoQueryAsNeeded is set to false and passwords are being requested from the zone.

Password Lookup

Google Apps setup

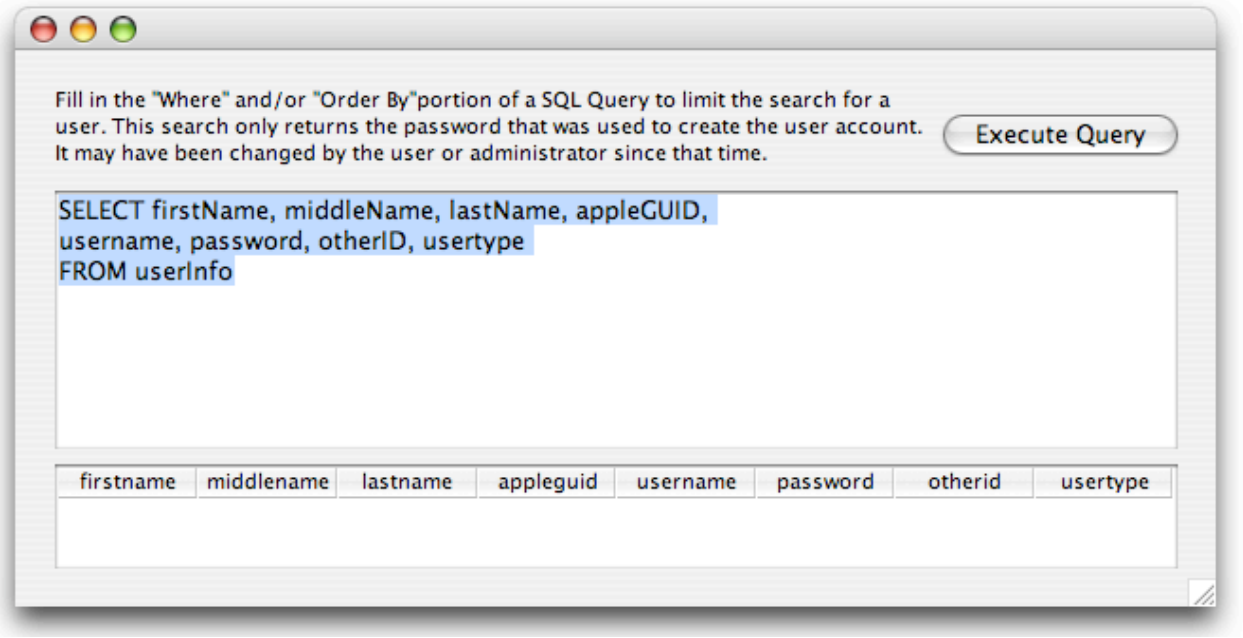
Google Apps for Education domains are set up with a password scheme that is very secure. Once a password is set or reset there is no way to get that password back in a human readable form. If your preferences in ForXs2gApps are set up to "Create Random passwords" it would be very hard to ever figure out what the passwords are.

ForXs2gApps agent data tracking

ForXs2gApps maintains a HSQL database of user information. This information is collected as various SIF messages come in. The information is checked and when all elements are recorded a user account is created. The passwords in this database are not encrypted.

Accessing the user information

When ForXs2gApps is running select the menu item "Search PW DB" available under the Window menu.



SQL Queries

The window opens with the following query.

```
SELECT firstName, middleName, lastName, grade, appleGUID,
username, password, otherID, usertype
FROM userInfo
```

This will return a limited amount of information about all users in the database. The SQL query can be altered to find a small subset of users or to display all information collected.

Example query to a password for a particular person.

```
SELECT password From userinfo WHERE firstName = 'George' and lastName = 'Adams'
```

Example query to find all fields and all records

```
SELECT * FROM userinfo
```

Example query to change the value of a single field in a record

```
UPDATE userinfo SET username='georgea003' WHERE otherID = '123456'
```

Example query to delete a record

```
DELETE FROM userinfo WHERE otherID = '123456'
```

Depending on the setup of your preferences deleting a user in this database and not in the server could possibly allow two user records to be created for a single user. Users deleted in this database should also be deleted from the server using Google Apps for Education administration web pages.

Example query to delete all records

```
DELETE FROM userinfo
```

If all users are deleted from the Google Apps at the beginning of the school year you may also want to delete all the records in this database so all information is requested fresh from the zone integration server. Depending on the username creation scheme deletng all the users in the internal database could create usernames with different digits at the end, sfranks02 instead of sfranks01.

Exporting Information

Exporting this information is not built into the agent but the data returned from a SQL query can be selected and copied using the shortcut keys Open Apple/Command and “C”. The result can then be pasted into a text document.

Subscribing to and Publishing SIF Objects

Introduction

In most cases the agent will be receiving information about students and creating user accounts on the Google server. The agent can also create accounts for staff and employees. Depending on the configuration, the agent can either get username and passwords from SIF or create it's own usernames and passwords. In this agent the SIF objects are referred to as topics in the configuration file. Subscription to these topics can be turned on or off depending on your needs.

Subscribing to student information

To receive information about students and create user accounts for them set the following topic properties;

```
<topic id="StudentPersonal">
<property name="subscribe" value="true"/>
<property name="publish" value="false"/>
<property name="matchOn" value="RefId"/>
<property name="IdPrefix" value=""/>
<property name="IdSuffix" value=""/>
</topic>
<topic id="StudentSchoolEnrollment">
<property name="subscribe" value="true"/>
<property name="publish" value="false"/>
<property name="matchOn" value="RefId"/>
</topic>
```

These are the default settings supplied in the sample config.xml files. Both StudentPersonal and StudentSchoolEnrollment need to have subscribe set to true for the agent to receive the information needed to create student accounts. The publish value is ignored.

Subscribing to staff information

The StaffPersonal topic needs to have it's subscribe value set to true to receive information from the SIF zone. Setting it to false will keep the agent from creating accounts for staff.

```
<topic id="StaffPersonal">
<property name="subscribe" value="true"/>
<property name="publish" value="false"/>
<property name="matchOn" value="RefId"/>
<property name="IdPrefix" value="9"/>
<property name="IdSuffix" value=""/>
</topic>
```

Subscribing to employee information

The EmployeePersonal topic needs to have it's subscribe value set to true to receive information from the SIF zone. Setting it to false will keep the agent from creating accounts for employees.

```
<topic id="EmployeePersonal">
<property name="subscribe" value="true"/>
<property name="publish" value="false"/>
<property name="matchOn" value="RefId"/>
<property name="IdPrefix" value="9"/>
<property name="IdSuffix" value=""/>
</topic>
```

Subscribing to school information

In very simple configurations SchoolInfo is not needed but can be a help in checking if an agent is set up correctly. In the more complex configurations SchoolInfo is needed to properly map home directories and to swap school numbers or school names into comments or keywords of the user account. The default is to subscribe to SchoolInfo.

```
<topic id="SchoolInfo">
<property name="subscribe" value="true"/>
<property name="publish" value="false"/>
<property name="matchOn" value="RefId"/>
</topic>
```

To publish username and password to other SIF agents

In the config.xml file

```
<topic id="Authentication">
<property name="subscribe" value="false"/>
<property name="publish" value="true"/>
<property name="matchOn" value="RefId"/>
</topic>
```

The agent can either subscribe or publish Authentication but not both.
Note that the usernames and passwords published are those created by the agent when they are added to the server. Changes to passwords on the server are not reflected in the ForXs2gApps internal database and are not published. To get usernames and passwords from another SIF agent
In the config.xml file

```
<topic id="Authentication">
<property name="subscribe" value="true"/>
<property name="publish" value="false"/>
<property name="matchOn" value="RefId"/>
</topic>
```

Authentication system name and system type properties.

In a SIF environment there can be multiple usernames and passwords for different systems. To identify and make sure the correct username and passwords are being used the agent can set the following two properties. When subscribing to the Authentication object the agent ignores any usernames and passwords that don't match these properties. When publishing these properties they are used to identify the name and type of the published username and password.
<property name="authSystemName" value="ForXs2gApps"/>
<property name="authSystemType" value="Application"/>
The value for authSystemType will be either Application or Network. The value for authSystemName can be anything but will normally be ForXs2gApps if publishing username and password. When subscribing authSystemName will be set by the publishing agent. It can be determined by viewing an XML message sent by the publishing agent.

Password Encryption

Unlike the rest of the information sent in SIF XML messages, passwords are never sent in human readable form. All passwords are encrypted in one of several levels of encryption. The sending agent and receiving agent need to use the same encryption scheme and in most cases need to exchange encryption keys so that a third party can not decrypt the passwords. The settings for encryption are specific to each zone connected to the agent. The following is an example of exchanging passwords using base64;

```
<zone id="Polk" template="Default" url="http://example.com:80/sif/interchange/sif/Polk">
<property name="adk.encryption.algorithm" value="base64"/>
<property name="adk.encryption.keys" value="" />
</zone>
```

The adk.encryption.keys value is ignored for algorithms base64, SHA1, and MD5. base64 needs no key to decrypt the password. SHA1 and MD5 are one way encryption and will not return a clear text password. They will return the hashed password as it was sent.
DES, TripleDES, and RC2 do require a key which is not sent as part of the XML message. The key is exchanged with the sending agent and added to the config file prior to receiving messages. Only the name of the key is sent in the XML message.
The keys and their values are set for the entire agent or for each zone. There can be multiple name value pairs in the configuration file.

```
<property name="adk.encryption.keys.keyNameSentInXMLMessage" value="dW7SKzwdn0Q=" />
<property name="adk.encryption.keys.anotherKeyName" value="hekIheYgsT" />
```

Note that RSA and DSA encryption schemes are not supported by this agent at this time.

Updating of Existing Users

When changes to student or staff information occur in a SIF environment the changes get passed on as SIF change event. These events can report on changes to names, grade, username, password, address, phone number and many more. Often times a user account gets modified after it is created by ForXs2gApps. ForXs2gApps can be set to ignore these SIF change events if updates tend to wipe out your modifications. The property ignoreChangesForExistingUsers can be set to true to avoid changing any users once they have been created.

```
<property name="ignoreChangesForExistingUsers" value="true"/>
```

The default value for is false and the agent will try to make the appropriate updates to a user without wiping out all changes made to the user.

```
<property name="ignoreChangesForExistingUsers" value="false"/>
```

When a user already exists changes received through SIF are made to the user and follow these rules.
Changes to the username (short name), otherId (User ID) will trigger a deletion of the existing user and recreation of the same user based on the configuration. This is done because these items can not be changed for an existing user.
In all other cases only the specific attributes that are changed will be changed in an existing user account
Changes to first name or last name will modify the first and last names.
If the first letter of the first last or middle name are changed and First1, Last1 or Middle1 are used as swapWords those attributes of the user will be replaced.
If any of these swapWords are in an attribute, modeluser(the current users short name), GradeLevel, GradYear, OtherId, BirthDate, StreetNumber, PhoneNumber that attribute will be replaced.
If the SchoolInfoRefId for a student changes (change of school) all attributes that use swapWords from School Info will be replaced. These are LocalId (school number), StatePrId, SchoolName, NCESId, IdentificationInfo1 (sometimes a school number),IdentificationInfo2, IdentificationInfo3
.

Configuration of User Passwords in the ForXs2gApps agent.

Simple static passwords

Choosing static passwords will create the same password for every user. There are few if any situations where this is a good idea. The dynamic static passwords below is generally a better solution but much of the configuration is the same. In the following set of properties PwUseStatic should be set to true and the other two to false.

```
<property name="PwUseStatic" value="true"/>
<property name="PwCreateRandom" value="false"/>
<property name="PwUseSIF" value="false"/>
```

Change the value of the "noStaticPasswordSet" to be your static password;

```
<property name="PwStatic" value="noStaticPasswordSet"/>
```

When using simple static passwords the model user should have the password policy set to require a password change on next log in otherwise any student could log in as any other user.

Complex static passwords

Complex static passwords are a step up from the plain static passwords above. The property settings are the same as those above but the PwStatic property value is replaced with a combination of words that will be replaced with information from the students data.
The following "swap" words will be replaced in the "static" password.
GradYear, GradeLevel, OtherId, First1, Middle1, Last1, BirthDate, StreetNumber, PhoneNumber.

GradYear will be replaced with the 4 digit year the student is expected to graduate.
GradeLevel will be replaced with the student's current grade.
First1 will be replaced with the first letter of the first name.
Middle1 will be replaced with the first letter of the middle name.
Last1 will be replaced with the first letter of the last name.
OtherId will be replaced with the student number used as the uid for the user account.
BirthDate will be replaced by the students or staff members birthdate in YYYYMMDD format.
StreetNumber will be replaced by the number portion of the address.
PhoneNumber will be replaced by the phone number as formatted in the SIF XML message.

Note that these swap words are case sensitive and will not be replaced if they don't match. Last1 will be replaced but last1 will not be replaced.
GradYear and GradeLevel are replaced with "staff" for staff members and "employee" for employees.
OtherId for staff members will be their staff id numbers proceeded by the number 9 unless the default settings are changed.

If no data was received for a student for one of the swap words the words will be replaced with an empty space. For this reason it would be wise to have some characters in the "static" password other than just the key words. "*BirthDate*" for a static password would create a password of "**" if no BirthDate was recorded for a student but would look like "*19921205* if they did have a BirthDate that was passed by SIF.
Note that Google requires passwords to be at least 6 characters in length.

Random passwords (default setting)

In the following set of properties PwCreateRandom should be set to true and the other two to false.

```
<property name="PwUseStatic" value="false"/>
<property name="PwCreateRandom" value="true"/>
<property name="PwUseSIF" value="false"/>
```

The following properties value will be used as a source of the characters used to create passwords.

```
<property name="PwOkCharacters" value="abcdefghjkmnpqrstuvwxyz23456789"/>
```

The default value above was chosen because the letters are not easily confused with other. The letters i, I, l, L and the number 1 have been removed from the list as have o, O and zero. To make the passwords more complicated the following letters could be added. Upper case letters, !@#$%^&*()_+<>?~{}[]\. Letters could also be added in multiple times to make them more likely to appear as they do in normal words, eeeaaassstttt.

The PwLength property determines how many "random" characters will be in the password.

```
<property name="PwLength" value="7"/>
```

In order to figure out what the passwords are someone will need to query the internal database. Two properties will allow this to happen and should be set as follows;

```
<property name="PwEraseAfterUserCreated" value="false"/>
<property name="dbHidePasswords" value="false"/>
```

Once the passwords are known dbHidePasswords can be set back to true.

Password supplied by SIF Zone

In the following set of properties PwUseSIF should be set to true and the other two to false.

```
<property name="PwUseStatic" value="false"/>
<property name="PwCreateRandom" value="false"/>
<property name="PwUseSIF" value="true"/>
```

The topic property under Authentication subscriber's value should be set to true. The default is false.

```
<topic id="Authentication">
<property name="subscribe" value="true"/>
```

In the SIF environment a person can have multiple sets of usernames and passwords. The following two properties identify which set will be used by this agent. Any other will be ignored.

```
<property name="authSystemName" value="Infinite Campus"/>
<property name="authSystemType" value="Application"/>
```

authSystemName is the name of the SIF system that is providing usernames and passwords.
authSystemType will be either Network or Application.

If the values of these properties are not known a query for Authentication can be requested from the zone. The resulting XML message in the ForXs2gApps log file can be examined to determine the appropriate values.

The last two properties, authSystemName and authSystemType are used both by username and password generation to match with the information provided in SIF.

Password Encryption

Unlike the rest of the information sent in SIF XML messages, passwords are never sent in human readable form. All passwords are encrypted in one of several levels of encryption. The sending agent and receiving agent need to use the same encryption scheme and in most cases need to exchange encryption keys so that a third party can not decrypt the passwords. The settings for encryption are specific to each zone connected to the agent. The following is an example of exchanging passwords using base64;

```
<zone id="Polk" template="Default" url="http://example.com:80/sif/interchange/sif/Polk">
<property name="adk.encryption.algorithm" value="base64"/>
<property name="adk.encryption.keys" value="" />

</zone>
```

The adk.encryption.keys value is ignored for algorithms base64, SHA1, and MD5. base64 needs no key to decrypt the password. SHA1 and MD5 are one way encryption and will not return a clear text password. They will return the hashed password as it was sent.

DES, TripleDES, and RC2 do require a key which is not sent as part of the XML message. The key is exchanged with the sending agent and added to the config file prior to receiving messages. Only the name of the key is sent in the XML message.
The keys and their values are set for the entire agent or for each zone. There can be multiple name value pairs in the configuration file.

```
<property name="adk.encryption.keys.keyNameSentInXMLMessage" value="dW7SKzwdn0Q=" />
<property name="adk.encryption.keys.anotherKeyName" value="hekIheYgsT" />
```

Note that RSA and DSA encryption schemes are not supported by this agent at this time.

Other Password settings

PwEraseAfterUserCreated will delete the password from the internal ForXs2gApps database after the password has been set in Google Apps. This option should **not** be used when creating random passwords as there would be no way to know what the passwords for users are. This option should be set to true when passwords are provided by another SIF agent.

```
<property name="PwEraseAfterUserCreated" value="false"/>
```

dbHidePasswords will display passwords as **** when set to true. This would make the passwords slightly more secure as the casual user would not be able to read a password.
<property name="dbHidePasswords" value="false"/>

Configuration of Usernames in the ForXs2gApps agent.

ForXs2gApps created usernames

The ForXs2gApps agent can create unique usernames, or short names based on the first, middle and last names of your students and staff. This would be the option chosen if there was no SIF enabled application that can supply a unique username and password or if you wanted a different set of usernames and passwords for Google Apps. Having a different combination of username and password would allow you to manage other password systems by being able to automate emailing lost passwords to an email account. If all usernames and passwords are unified including email there is no easy way to get forgotten usernames and passwords out to users.

Created username configuration

To have ForXs2gApps create usernames UnUseSif should be set to false

```
<property name="UnUseSif" value="false"/>
```

UsernameFormat sets the alpha portion of the username. Before creating usernames the First, Last and Middle names provided are made all lower case and all characters other than a-z are removed. Fs,Ms and Ls in the format string will be replaced with letters from the First, Middle or Last name. Any F,M or Ls in excess of the length of the users name are ignored. Letters other than f, m and l will be left in place. Don't use spaces.
Examples
Given the name Laura Jane Smith-Jones
format fffffffffffffl the alpha portion of the username would be "lauras".
format lllllllll_f the alpha portion of the username would be "smithjon"_j
Given the name Kay Ray Fay
format ffffffff-mmmmmm-llllll_student the alpha portion of the username would be "kay-ray-fay_student" .

```
<property name="UsernameFormat" value="ffffffffffffl"/>
```

Things to consider when deciding on a username format;

A short username is easier and faster to type but will be less informative when used as an email address. It can also cause more of the alpha portions of the username to match so more trailing digits may need to be added to make it unique.
It might be easier to locate if the last name proceeds the first name in the UsernameFormat.

Digits added to usernames

Usernames are made unique by adding digits to the end of alpha portion of the username. How the digits are added is determined by three properties UnDigits, UnPadWithZeros, UnRandom.

UnDigits sets the range of digits that will be added. 1 uses numbers 1 through 9 and should only be used in schools with very small populations. 2 uses numbers 1 through 99 and 3 uses numbers 1 through 999. Unless very short usernames are used and you have a very large population number above 3 would not be needed.

```
<property name="UnDigits" value="3"/>
```

UnPadWithZeros can be set to true or false. When set to true leading zeros are added to all numbers to make them the same length as UnDigits above. Set to true a username would look like "myang002" while if set to false it would be "myang2".

```
<property name="UnPadWithZeros" value="true"/>
```

UnRandom can be set to true or false. When set to true each new username is given a trailing number picked randomly from the range of possible numbers as determined by UnDigits above. If it is set to false numbering starts at one and increments up. The false setting makes for a large number 01 or 001s. It might be a problem as many students could be easily singled out by figuring out the username scheme and adding a 001 to the end. When creating a random number the agent checks both the internal database and the existing user accounts to make sure it doesn't create accounts with duplicated usernames.

```
<property name="UnRandom" value="true"/>
```

Username supplied by SIF Zone

To have ForXs2gApps use usernames supplied to it by SIF UnUseSif should be set to true

```
<property name="UnUseSif" value="true"/>
```

The topic property under Authentication subscriber's value should be set to true. The default is false.

```
<topic id="Authentication">
<property name="subscribe" value="true"/>
```

In the SIF environment a person can have multiple sets of usernames and passwords. The following two properties identify which set will be used by this agent. Any other will be ignored.

```
<property name="authSystemName" value="Infinite Campus "/>
<property name="authSystemType" value="Application"/>
```

authSystemName is the name of the SIF system that is providing usernames and passwords.
authSystemType will be either Network or Application.

If the values of these properties are not known a query for Authentication can be requested from the zone. The resulting XML message in the ForXs2gApps log file can be examined to determine the appropriate values.

The last two properties, authSystemName and authSystemType are used both by username and password generation to match with the information provided in SIF.

The property convertSifUsernamesToLowercase when set to true will convert usernames with uppercase letters to lowercase. Email usernames are generally lowercase. When logging in usernames are not case sensitive but passwords are. Setting convertSifUsernamesToLowercase to false will leave usernames as they are received by SIF.

```
<property name="convertSifUsernamesToLowercase" value="true"/>
```

When using SIF for usernames the following properties are ignored; UsernameFormat, UnDigits, UnPadWithZeros, UnRandom

Configuration of SIF Traffic Management in the ForXs2gApps agent.

Properties explained

```
<property name="adk.messaging.pullFrequency" value="30000"/>
<property name="adk.messaging.maxBufferSize" value="320000"/>
<property name="shellTimerFrequency" value="5"/>
<property name="setPasswordWhenUserCreated" value="false"/>
<property name="queryRequestTimerFrequency" value="10"/>
<property name="maxNumToQuery" value="50"/>
<property name="limitQueriesToOffHours" value="false"/>
<property name="autoQueryAsNeeded" value="true"/>
<property name="queriesOkBeforeHour" value="6"/>
<property name="queriesOkBeforeMinute" value="0"/>
<property name="queriesOkAfterHour" value="4"/>
<property name="queriesOkAfterMinute" value="15"/>
```

Overview

Some school districts have concerns about the amount of network traffic created by SIF XML messages, the added load on the student information system. The properties above can be set to control the actions of the agent to ensure the agent has a minimal impact. There are three threads or separate sets of actions. The timing of each thread can be controlled.

User Creation based on changes to the internal database

```
<property name="shellTimerFrequency" value="5"/>
<property name="setPasswordWhenUserCreated" value="false"/>
```

The default setup for ForXs2gApps is to create the user as soon as all information is available. It sets the passwordStatus field in the database to "created". The shellTimer thread, working at its own pace, queries the internal database looing for a passwordstatus or "ready". If the internal database is updated by setting the passwordstatus to "ready" this thread will create the users.
The shellTimerFrequency property determines the delay in seconds between queries to the database to see if any new passwords are ready to be created.

```
<property name="shellTimerFrequency" value="5"/>
```

The value is in seconds and can be adjusted as needed.

Returning SIF Queries

```
<property name="queryRequestTimerFrequency" value="10"/>
<property name="maxNumToQuery" value="50"/>
<property name="limitQueriesToOffHours" value="false"/>
<property name="autoQueryAsNeeded" value="true"/>
<property name="queriesOkBeforeHour" value="6"/>
<property name="queriesOkBeforeMinute" value="0"/>
<property name="queriesOkAfterHour" value="16"/>
<property name="queriesOkAfterMinute" value="15"/>
```

The normal day to day traffic created by SIF should be a manageable amount of data but a simple query from the agent can unleash a flood of XML traffic. When first populating Google Apps for Education one would normally request all the students from the zone. This is a rather small XML message asking for StudentPersonal information. The data returned will be large packets of information about multiple students. This by itself is a large amount of information. To add to this

the information in a StudentPersonal object isn't enough to create a user so the agent will send a query back to the zone to ask for the missing information. XML is human readable but is very wordy. The amount of bytes passed over the network can pile up. In a district with 7000 students, requesting all StudentPersonal information might return 100 maximum size XML messages. For each student received the agent could generate a message for StudentSchoolEnrollment and Authentication for a total of 14,000 XML message followed by 14,000 return messages. These messages are small but large in number. They also require a fair amount of processing time on both ends of the SIF environment. To cut down on this traffic the queryTimer was created. Instead of sending a separate query for each piece of information needed the agent groups the queries together and sends them off. The grouped query is larger than a single query but much smaller than the sum total size of the individual query. The returning data can also be grouped together and cuts down on traffic. The traffic can be further controlled by delaying the queries until after peak hours or turned off completely forcing the user to send queries by hand to complete the needed information.

`autoQueryAsNeeded` is set to true by default but if set to false the agent will not create any queries for needed information.

`queryRequestTimerFrequency` determines the delay in seconds between queries.

`maxNumToQuery` sets the maximum number of users to include when creating the query. Some SIF agents do not understand the combined queries and in those cases this should be set to the number 1.

`limitQueriesToOffHours` is false by default but if set to true will allow queries to pile up waiting for off hours to send them.

`queriesOkBeforeHour, queriesOkBeforeMinute, queriesOkAfterHour, queriesOkAfterMinute` combined determine when it is OK to send queries. The hours are military time 0 - 23, minutes are 0-59

Incoming SIF messages

```
<property name="adk.messaging.pullFrequency" value="30000"/>
<property name="adk.messaging.maxBufferSize" value="320000"/>
```

`adk.messaging.pullFrequency` is the how often the agent sends a XML message to check if any messages are waiting for it. The value is in milliseconds so 30000 is once every 30 seconds (30 * 1000). When testing this value can be dropped so there is less delay in getting messages back from the zone. In normal operation it could be increased to every 5 minutes or more depending on your situation.

`adk.messaging.maxBufferSize` determines the maximum size in bytes of any one XML message. If the data is too big to fit in a single message it will be divided up into packages of the maximum size. The default is 32,000 bytes.

Configuration of Graphical User Interface in the ForXs2gApps agent.

Properties

```
<property name="screenLogLevel" value="all"/>
<property name="adk.log.level" value="-1"/>
<property name="dbHidePasswords" value="false"/>
<property name="ShowProperties" value="false"/>
<property name="runHeadless" value="false"/>
```

screenLogLevel

This property controls how much information is displayed in the Status Window. Valid options are none, info, warnings, debug, all. By default it is set to debug but should be set to info or none once testing and synchronization are complete.
<property name="screenLogLevel" value="all"/>

adk.log.level

This property determines how much data is written into the log file. Some of the SIF specific and XML messaging information is only written to this log and not to the Status Window. Valid options are the numbers listed below. The numbers can be added together to combine options.
ADK.DBG_TRANSPORT is 4
ADK.DBG_MESSAGING is 8
ADK.DBG_MESSAGING_PULL is 64
ADK.DBG_MESSAGING_DETAILED is 128
ADK.DBG_MESSAGE_CONTENT is 256
ADK.DBG_PROVISIONING is 512
ADK.DBG_LIFECYCLE is 268435456
ADK.DBG_PROPERTIES is 1073741824
The following are combinations of the above
ADK.DBG_ALL is -1
ADK.DBG_NONE is 0
ADK.DBG_MINIMAL is 536871424
ADK.DBG_MODERATE is 805306888
ADK.DBG_MODERATE_WITH_PULL is 805306952
ADK.DBG_DETAILED is 805307084
ADK.DBG_VERY_DETAILED is 1879048956

```
<property name="adk.log.level" value="-1"/>
```

dbHidePasswords

This property will display passwords as **** when set to true. This would make the passwords slightly more secure as the casual user would not be able to read a password. The default value is true. Valid options are true or false.

```
<property name="dbHidePasswords" value="false"/>
```

ShowProperties

This property will display properties as they are read in by the agent. This can help debug problems where a property has been added to a file twice and only one value is being read in. The default value is false.

```
<property name="ShowProperties" value="false"/>
```

runHeadless

This property will cause the agent to start up and not display any windows or menus. The default value is false. It can be set to true when all configuration is complete and the agent is set up to run as a daemon. If the agent encounters a problem with license file or config file when started it will temporarily set this property to true and display the windows and menus.

```
<property name="runHeadless" value="false"/>
```

Configuration of Student Information System related properties in the ForXs2gApps agent.

Objects and Fields

```
<object object="Authentication">
<object object="EmployeePersonal">
<object object="StaffPersonal">
<object object="SchoolInfo">
<object object="StudentPersonal">
<object object="StudentSchoolEnrollment">
```

These objects and the fields that are associated with them map the SIF elements to the ForXs2gApps data elements. Each student system can provide data in slightly different formats. Start off ForXs2gApps using the provided config.xml file for your student system. If your system is not listed try one from another student system and we can assist you in making any changes needed.

Properties

```
<property name="convertSifUsernamesToLowercase" value="true"/>
<property name="useSifOtherID" value="true"/>
<property name="ignoreSifEnrollmentDeletes" value="true"/>
<property name="ignoreSifEnrollmentExitDates" value="false"/>
<property name="calcGradYearFromGrade" value="false"/>
<property name="currentGradYear" value="2007"/>
<property name="studentSchoolEnrollmentTimeFrame" value=""/>
```

convertSifUsernamesToLowercase

The property convertSifUsernamesToLowercase when set to true will convert usernames with uppercase letters to lowercase. Apple's Workgroup Manager creates all usernames as lowercase but they seem to work OK if they are mixed or uppercase. When logging in usernames are not case sensitive but passwords are. Setting convertSifUsernamesToLowercase to false will leave usernames as they are received by SIF.

```
<property name="convertSifUsernamesToLowercase" value="true"/>
```

useSifOtherID

This property defaults to true and should be left that way if at all possible. When true the ID in Workgroup Manager will be the student or staff number. When set to false a new user will get the next highest available ID that is above 1000.

```
<property name="useSifOtherID" value="true"/>
```

ignoreSifEnrollmentDeletes

This property determines what the agent does when it receives a StudentSchoolEnrollment Delete event. When set to true the event is ignored, when set to false the agent will disable **but not delete** the user account.

```
<property name="ignoreSifEnrollmentDeletes" value="true"/>
```

ignoreSifEnrollmentExitDates

This property determines what the agent does when it receives a StudentSchoolEnrollment which includes an ExitDate. When set to true the exit date is not set as the date the user account expires. When set to true the exit date is used to set the date that an account will expire or become disabled. If you would like students to have access to accounts after the end of the school year set this to true.

```
<property name="ignoreSifEnrollmentExitDates" value="false"/>
```

calcGradYearFromGrade

For those student systems that don't supply the GradYear element this and the next property can be used to calculate a graduation year. The default is false which will allow the StudentPersonal GradYear element fill the GradYear.

```
<property name="calcGradYearFromGrade" value="false"/>
```

currentGradYear

This property is used when calcGradYearFromGrade is set to true. The default is 2007 but should increase by one each school year. This should be 2007 for the 06-07 school year. The calculation used is
currentGradYear + 12 - GradeLevel = GradYear

```
<property name="currentGradYear" value="2007"/>
```

studentSchoolEnrollmentTimeFrame

In order to handle the historical StudentSchoolEnrollments sent by the SASIxp agent changes were made to the SIF_Query code to allow for multiple criteria when requesting information. Specifically TimeFrame="Current" so that Historical and Future enrollments were not sent. The SASIxp agent appears to not understand the compound SIF_Query so the ForXs2gApps agent can be configured to ignore all but "Current" enrollment objects.

```
<property name="studentSchoolEnrollmentTimeFrame" value="Current"/>
```

The default is as follows and will create queries as they were prior to adding this feature.

```
<property name="studentSchoolEnrollmentTimeFrame" value=""/>
```

Working with the internal HSQL database of users

How it works

SIF sends out chunks of information in SIF objects. No single object has all the information needed to create a user so ForXs2gApps holds what it does get in an integrated HSQL database until it has all the pieces then it creates the user and shortly after that creates the password and password policies.

Troubleshooting

The initial set up of ForXs2gApps and its connection to a SIF Zone involves a number of options. The database can be used to see if the information you expected to be receiving has actually arrived and been recorded by ForXs2gApps. Use the sample SQL queries below to display data from the database.

Exporting from ForXs2gApps

There is no built in feature to export data from the database but you can run a SQL query, select all the data, copy it and then paste it into a text document. It produces a tab delimited file which can be read by other programs. Note that the passwords in the database are those that ForXs2gApps used to create the initial password for the user and if the user changes their password it will not be reflected in this database,

SQL Queries

This is the default SQL query that shows up when first opening the PW DB window.

```
SELECT firstName, middleName, lastName, grade, appleGUID,
username, password, otherID, usertype
FROM userInfo
```

Display every user and all columns in the database

```
SELECT * FROM userinfo
```

Display all usernames and passwords for students only

```
SELECT username, password FROM userinfo WHERE usertype like 'student%'
```

Display the status of SIF queries back to the zone to get more information.

```
SELECT queryAuth, queryStuPersonal, queryStuEnrollment, firstName, lastName
FROM userinfo
```

Halt pending queries to the zone.

To stop automatically sending queries to the zone for users that have pending Authentication queries. Be careful with update queries.

```
UPDATE userinfo
SET queryAuth = 'received'
WHERE queryAuth = 'yes'
-- For StudentPersonal queries replace both instances of queryAuth above with queryStuPersonal
-- For StudentSchoolEnrollment queries replace both instances of queryAuth above with queryStuEnrollment
```

To send SIF queries for existing users reset the query flag.

This example asks for StudentSchoolEnrollment for all students in grade 08 and with a last name of Johnson. Be careful with update queries. When the queries are received ForXs2gApps will update/create these users if all other information is complete.

```
UPDATE userinfo
SET queryStuEnrollment = 'yes'
WHERE grade ='08' AND lastName ='Johnson'
```

Delete all the records without deleting the database.

```
DELETE FROM userinfo
```

If you are doing this to start a new year fresh you might want to do the same in Workgroup Manager by deleting all the user accounts other than the admin accounts and model user accounts.

Delete all the passwords in the database.

```
UPDATE userinfo SET password = ''
```

Reset a password for a particular user and force the agent to change the password in Google Apps.

```
UPDATE userinfo SET password = 'apple' , passwordstatus='ready' WHERE username='fredh001'
```

Check if GradYear is coming from SIF or being calculated based on Grade

```
SELECT gradYear, grade, firstName, lastName
FROM userinfo
ORDER BY gradYear
```

Find a particular student by student number

(replace the 123456 with the number you are looking for)

```
SELECT firstName, middleName, lastName, grade, appleGUID, username, password, otherID, usertype
FROM userInfo
WHERE otherID ='123456'
```

Find all staff users sorted by last name

```
SELECT firstName, middleName, lastName, grade, appleGUID, username, password, otherID, usertype
FROM userInfo WHERE usertype LIKE 'staff%'
ORDER BY lastName, firstName
```

Count the number of user records in the database.

```
SELECT count(*) FROM userinfo
```

Count the number of student user records in the database.

```
SELECT count(*) FROM userinfo WHERE usertype LIKE 'student%'
```

Count the number of staff user records in the database.

```
SELECT count(*) FROM userinfo WHERE usertype LIKE 'staff%'
```

School Info Table

This table will be filled or updated any time the zone is queried for SchoolInfo

To display all school records

```
SELECT * FROM schoolinfo
```

To delete all school records

```
DELETE FROM schoolinfo
```

To delete some school records

```
DELETE FROM schoolinfo WHERE schoolname LIKE '%elem%'
```

This would find and delete all the elementary schools that actually had Elementary in their name
It might also delete a High School if its name was Celement High so be careful.

Password options involving keywords swapped for user information.

Swap words

There are a number of "swap words" that can be added to complex "static passwords". These specific words will be replaced in a new user's password when the user is created. There are two categories of swap words. The first is user information and the other is school information.

Places where words are swapped

In the password, when using a static password for all users.

User Information Swap words

The following user swap words replace the word with information specific to that user.

- **GradYear** is replaced with the graduation year supplied in the StudentPersonal SIF object.
- **GradeLevel** is replaced with the current grade level supplied in the StudentSchoolEnrollment SIF object.
- **OtherId** is replaced with the student number in the StudentPersonal object or by the staff number for Staff members
- **First1** is replaced with the first letter of the first name of the student, staff or employees name.
- **Middle1** is replaced with the first letter of the middle name of the student, staff or employees name.
- **Last1** is replaced with the first letter of the last name of the student, staff or employees name.
- **BirthDate** will be replaced by the students or staff members birthdate in YYYYMMDD format.
- **StreetNumber** is replaced with the number portion of the user's street address. *
- **PhoneNumber** is replaced with the phone number of the user as supplied by SIF. It is replaced in the same format as supplied.

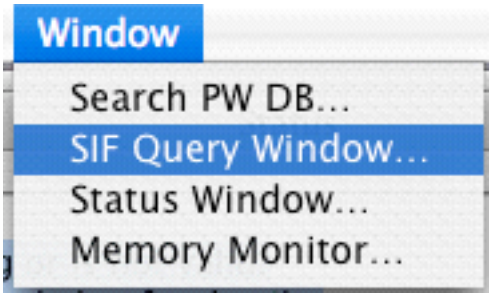
Note that these words are case sensitive and will not be replaced if they don't match. Last1 will be replaced but last1 will not be replaced.
GradeLevel and **GradYear** for staff is always replaced with "staff".
GradeLevel and **GradYear** for employees is always replaced with "employee".
If street number comes from the StudentAddress/Address/Street/Line1 SIF element instead of the StreetNumber SIF element it will have extra digits in those cases where the street name has digits. 19283 W 4th Street would have a StreetNumber of 192834 instead of the expected 19283.

School Information swap words

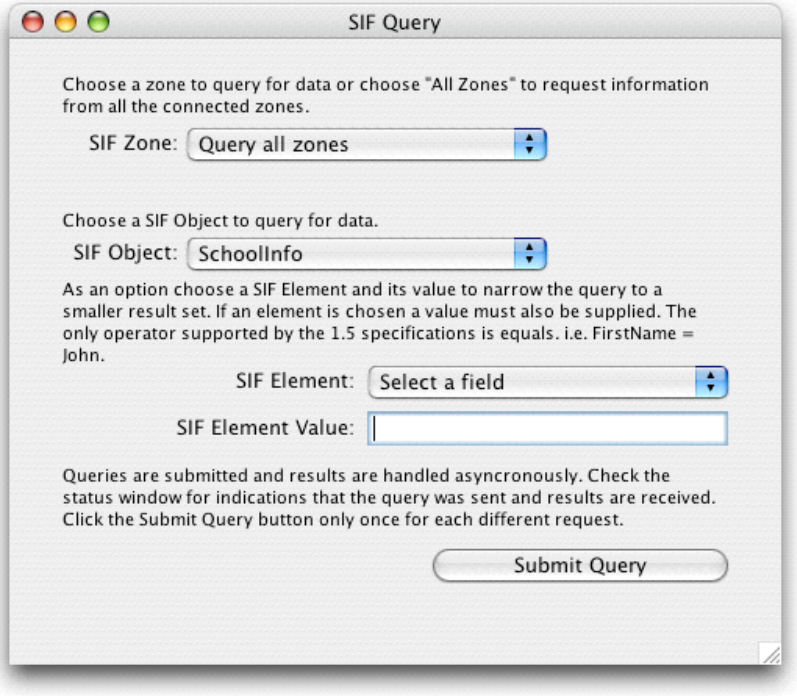
Using the school information swap words is useful when the agent is attached to a zone or zones that have multiple schools attached to it. The swap words can be used to identify which school a student is enrolled
LocalId is replaced with the school number supplied by SIF in LocalId
SchoolName is replaced with the school name supplied by SIF
StatePrId is replaced with the state's school number supplied by SIF
NCESId is replaced with the NCES number supplied by SIF
IdentificationInfo1 is replaced with text supplied by SIF for IdentificationInfo1
IdentificationInfo2 is replaced with text supplied by SIF for IdentificationInfo2
IdentificationInfo3 is replaced with text supplied by SIF for IdentificationInfo3

For these school information swap words to work the agent needs to have queried the SIF zones and received information about the connected schools. To get this information follow these steps.

Start up the ForXs2gApps agent.
Choose the Window >> SIF Query Window menu item.



Create and send a SIF query to the SIF Zone by
Choosing "Query all zones from the SIF Zone drop down list.
Choosing "SchoolInfo" from the SIF Object drop down list.



Leave the SIF Element at "Select a field"
Leave the SIF Element Value text box empty.
Click on the "submit Query" button.

Click on the OK in the dialog box.
Bring the Status window to the front and wait for the school info to appear.
You should see at least a list of connected school names or a bunch of info about the schools depending on the log level set in the agent.
Once all the schools appear in the window choose the Window >>Search PW DB item.

Replace the SQL query with the following text;

```
SELECT *
FROM schoolInfo
Order by schoolname
```

The bottom half of the window should fill with text. The school name, LocalId and other information that appears will be used when swapping school information into a user's record. If the information for a swap word is missing it will replace the swap word with a blank.

ForXs2gApps SOFTWARE LICENSE AGREEMENT

In consideration for your use of the software and any updates, customizations and/or enhancements, entitled ForXs2gApps Agent provided by Hay River Software LLC ("Licensor"), you ("User") agree to the following terms and conditions. If you do not agree to these terms, you may not install the software and you must return the package to your point of purchase immediately for a refund.

1. License. Licensor hereby grants the User a non-exclusive, non-transferable license to use the Software for personal use on one computer by User only. Licensor reserves the right at any time, without liability or prior notice, to change the features or characteristics of the Software, this Agreement, or the Software's documentation and related materials.

2. License Restrictions.
a. User acknowledges that the Software and its structure, organization, and source code constitute valuable trade secrets of Licensor. Accordingly, User agrees not to (i) copy, perform, distribute, modify, adapt, alter, translate, or create derivative works from the Software; (ii) merge the Software with other software; (iii) sublicense, lease, rent, or loan the Software to any third party; (iv) reverse engineer, decompile, disassemble, or otherwise attempt to derive the source code for the Software; or (v) otherwise use the Software except as expressly allowed in this Agreement.
b. User shall comply with all applicable export and import control laws and regulations in its use of the Software and, in particular, User shall not export or re-export the Software without all required United States and foreign government licenses. User understands that access and use of the Software from outside the United States may constitute export of technology and technical data which may implicate export regulations and/or require export license.
c. Licensor retains exclusive ownership of all worldwide copyrights, trade marks, service marks, trade secrets, patent rights, moral rights, property rights and all other industrial rights in the Software and documentation, including any derivative works, modification, updates, or enhancements. All rights in and to the Software not expressly granted to User in this Agreement are reserved by Licensor. Nothing in this Agreement shall be deemed to grant, by implication, estoppel or otherwise, a license under any of Licensor's existing or future patents.
d. If User is an employee, contractor or agent of the United States Government, the following provision applies. The Software and documentation are comprised of "commercial computer software" and "commercial computer software documentation" as such terms are used in 48 C.F.R. 12.212 (SEPT 1995) and are provided to the Government (i) for acquisition by or on behalf of civilian agencies, consistent with the policy set forth in 48 C.F.R. 12.212; or (ii) for acquisition by or on behalf of units of the Department of Defense, consistent with the policies set forth in 48 C.F.R. 227.7202-1 (JUN 1995) and 227.7202-3 (JUN 1995). Unpublished rights reserved under the copyright laws of the United States.
e. User shall not use the Software in any way that violates any local, state, federal or law of other nations, including but not limited to the posting of information that may violate third party rights, that may defame a third party, that may be obscene or pornographic, that may harass or assault others, that may violate hacking or other computer crime regulations, etc. Licensor does not monitor or edit any transmissions, postings, routings or other materials which User may send, post, route, transmit or otherwise move through or with the Software.

3. WARRANTY DISCLAIMER. THE SOFTWARE IS PROVIDED "AS IS" WITHOUT ANY WARRANTY WHATSOEVER, INCLUDING BUT NOT LIMITED TO ANY FUNCTIONALITY OR ITS BEING VIRUS FREE. USER RECOGNIZES THAT THE AS IS CLAUSE OF THIS AGREEMENT IS AN IMPORTANT PART OF THE BASIS OF THIS AGREEMENT, WITHOUT WHICH LICENSOR WOULD NOT HAVE AGREED TO ENTER THIS AGREEMENT. LICENSOR AND THIRD PARTIES DISCLAIM ALL WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE SOFTWARE, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NONINFRINGEMENT. NO REPRESENTATION OR OTHER AFFIRMATION OF FACT REGARDING THE SOFTWARE SHALL BE DEEMED A WARRANTY FOR ANY PURPOSE OR GIVE RISE TO ANY LIABILITY OF THIRD PARTIES WHATSOEVER. USER ACKNOWLEDGES THAT HE OR SHE HAS RELIED ON NO WARRANTIES OR STATEMENTS OTHER THAN AS MAY BE SET FORTH HEREIN.

4. LIMITATION OF LIABILITY. LICENSOR SHALL NOT BE LIABLE TO USER OR ANY THIRD PARTY FOR ANY INCIDENTAL, INDIRECT, EXEMPLARY, SPECIAL OR CONSEQUENTIAL DAMAGES, UNDER ANY CIRCUMSTANCES, INCLUDING, BUT NOT LIMITED TO, LOST PROFITS, REVENUE OR SAVINGS, LOSS OF GOODWILL, OR THE LOSS OF USE OF ANY DATA, EVEN IF LICENSOR HAD BEEN ADVISED OF, KNEW, OR SHOULD HAVE KNOWN, OF THE POSSIBILITY THEREOF. UNDER NO CIRCUMSTANCES SHALL LICENSOR'S AGGREGATE CUMULATIVE LIABILITY HEREUNDER, WHETHER IN CONTRACT, TORT, OR OTHERWISE, EXCEED THE TOTAL AMOUNT OF FEES ACTUALLY

PAID BY USER UNDER THIS AGREEMENT. USER ACKNOWLEDGES THAT THE FEES PAID BY HIM OR HER REFLECT THE ALLOCATION OF RISK SET FORTH IN THIS AGREEMENT AND THAT LICENSOR WOULD NOT ENTER INTO THIS AGREEMENT WITHOUT THESE LIMITATIONS ON ITS LIABILITY.

5. Indemnification. User shall defend, indemnify and hold harmless Licensor, its officers, directors contractors, agents and employees, from any and all claims or causes of action arising out of use of or related to the Software, and pay any and all damages and expenses (including but not limited to attorneys fees incurred by Licensor and/or third parties) in connection therewith. Licensor reserves the right, at it own expense, to assume the exclusive defense and control of any matter otherwise subject to indemnification by User, in which event User shall cooperate with the Licensor in asserting any available defenses.

6. Termination. This Agreement is effective unless terminated by Licensor at any time for any breach of this Agreement. User may terminate this Agreement at any time by destroying all copies of the Software in User's possession and deleting the Software from User's computer system and other storage media, or by returning all such copies to Licensor. This Agreement and User's right to use this Software automatically terminate if User breaches this Agreement.

7. Legal Compliance. Licensor may suspend or terminate use of Software and this Agreement immediately upon receipt of any notice which alleges that User has used the Software for any purpose that violates any local, state, federal or law of other nations, including but not limited to the posting of information that may violate third party rights, that may defame a third party, that may be obscene or pornographic, that may harass or assault others, that may violate hacking or other criminal regulations, etc. of its agents, officers, directors, contractors or employees. In such event, Licensor may disclose the User's identity and contact information, if requested by a government or law enforcement body, or as a result of a subpoena or other legal action, and Licensor shall not be liable for damages or results thereof and User agrees not to bring any action or claim against this Licensor for such disclosure.

8. Miscellaneous. Either party may assign this Agreement to any successor in interest who purchases or through change in control owns greater than fifty percent of the assets or equity of such entity and agrees in writing to be bound by the terms and conditions herein; any other assignment shall be void. This Agreement and any dispute arising hereunder shall be construed in accordance with the laws of the State of Wisconsin without regard to principles of conflict of laws. For the purpose of this Agreement, User consents to the personal jurisdiction and venue of the state and federal courts located in Wisconsin. If any provision of this Agreement is prohibited by law or held to be unenforceable, the remaining provisions hereof shall not be affected, and this Agreement shall continue in full force and effect as if such unenforceable provision had never constituted a part hereof, and the unenforceable provision shall be automatically amended to so as to best accomplish the objectives of such unenforceable provision within the limits of applicable law. This Agreement may be executed in counterparts, each of which shall be deemed an original but all of which together shall constitute the same instrument. Any waiver of a provision of this Agreement must be in writing and signed by the party to be charged. A valid waiver hereunder shall not be interpreted to be a waiver of that obligation in the future or any other obligation under this Agreement. This Agreement constitutes the entire agreement between the parties related to the subject matter hereof, supersedes any prior or contemporaneous agreement between the parties relating to the Software and shall not be changed except by written agreement signed by an officer of Licensor.

Running ForXs2gApps as a Daemon on Mac OS X

Introduction

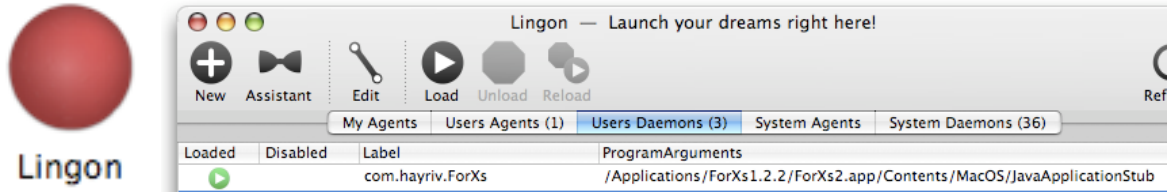
When run as an application ForXs2gApps a user needs to be logged in to the computer. This is acceptable when setting up the agent and when synchronization with the zone is being done but it should not be the normal practice. Passwords could be easily read from the ForXs2gApps internal database if the computer is left unattended. Once the agent is fully configured and tested it can be run in headless mode. In this case no windows appear but the icon for ForXs2gApps remains in the dock and a user still needs to be logged in. A better alternative is to run ForXs2gApps in the background as a user daemon, To set up ForXs2gApps to run as a daemon do the following.

ForXs2gApps prerequisites

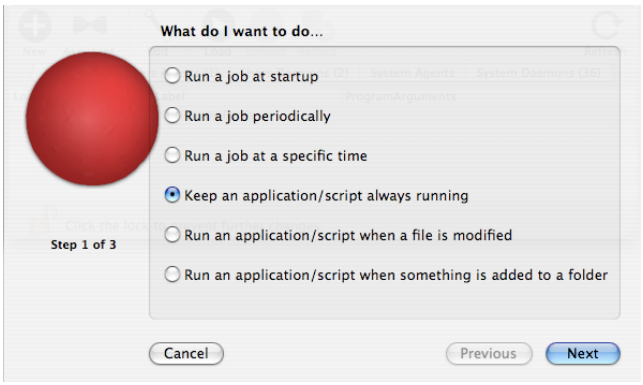
1. Configure ForXs2gApps per the direction in this document.
2. Test the agent's connection to the zone server, to Google Apps and create some users using SIF
3. Synchronize your server with the zone by querying the zone for all students and staff.
4. Change the runHeadless property in the config.xml file to true.

Lingon

"Lingon is a graphical interface for creating launchd configuration files and controlling them through launchctl for Mac OS X Tiger." It is used as an easy way to set up ForXs2gApps to run as a daemon. It is available for download at http://lingon.sourceforge.net/
After downloading and installing Lingon start it up.
Click on the Assistant icon to launch the assistant.

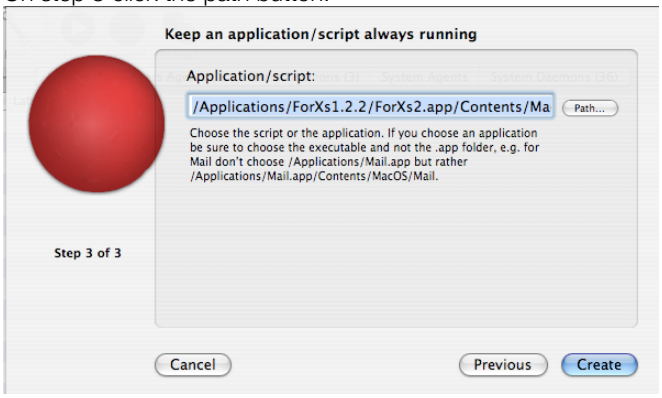


On step one select "Keep an appication/script always running"

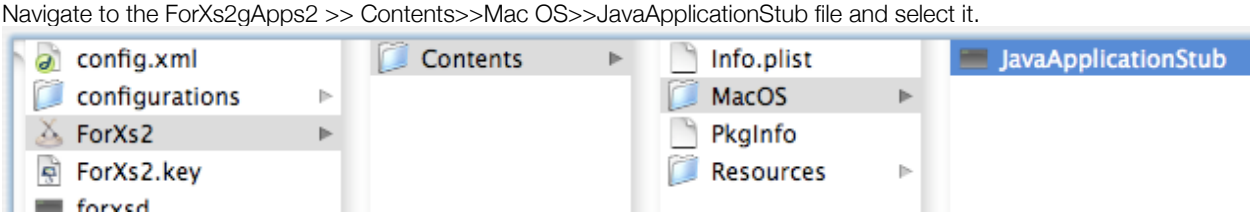


On step two enter com.hayriv.ForXs2gApps as the label and select "Must run as root"

On step 3 click the path button.



Navigate to the ForXs2gApps2 >> Contents>>Mac OS>>JavaApplicationStub file and select it.



Click the Create button.

The agent should start up and run without displaying any windows. It can be stopped by selecting it in Lingon and clicking the Unload icon.