

Connected Backup

Software Version 9.0

Account Management Web Services Development



Document Release Date: May 2019
Software Release Date: May 2019

Legal notices

Copyright notice

© Copyright 2017-2019 Micro Focus or one of its affiliates.

The only warranties for products and services of Micro Focus and its affiliates and licensors (“Micro Focus”) are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

You can check for more recent versions of a document through the [MySupport portal](#). Many areas of the portal, including the one for documentation, require you to sign in with a Software Passport. If you need a Passport, you can create one when prompted to sign in.

Additionally, if you subscribe to the appropriate product support service, you will receive new or updated editions of documentation. Contact your Micro Focus sales representative for details.

Support

Visit the [MySupport portal](#) to access contact information and details about the products, services, and support that Micro Focus offers.

This portal also provides customer self-solve capabilities. It gives you a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the MySupport portal to:

- Search for knowledge documents of interest
- Access product documentation
- View software vulnerability alerts
- Enter into discussions with other software customers
- Download software patches
- Manage software licenses, downloads, and support contracts
- Submit and track service requests
- Contact customer support
- View information about all services that Support offers

Many areas of the portal require you to sign in with a Software Passport. If you need a Passport, you can create one when prompted to sign in. To learn about the different access levels the portal uses, see the [Access Levels descriptions](#).

Contents

- Chapter 1: Concepts 9
 - Account Management Website overview 9
 - Website purpose 9
 - Standard Account Management Website 9
 - Custom Account Management applications 10
 - Application interaction with the Data Center 10
 - Communication flow 10
 - Website settings in Support Center 11
 - Access user's account online link 12
 - Recommended content 12
 - General content 12
 - System requirement content 13
 - Backup plan content 13
 - Recommended functions 13
 - Introduction 13
 - Sign in 14
 - Registration and download 14
 - Backup plan and billing 14
 - Account recovery 14
 - Change password 15
 - Change profile 15
 - Order media 15

- Chapter 2: Development environment 16
 - Interfaces 16
 - Account Management Web Services 16
 - Authenticated methods 16
 - Unauthenticated methods 16
 - Simple Object Access Protocol 16
 - Find the Web Services API WSDL file 17
 - Available code examples 17
 - Description 17
 - Use the sample code 17
 - Use the standard online help files 18
 - Method overview 18
 - Sequence of calling methods 18
 - Authenticated methods 18
 - Variable types 19

- Chapter 3: Application navigation 20
 - Types of navigation 20
 - Top navigation 20
 - Left navigation 20
 - Bottom navigation 21
 - Examples 21
 - Sample top navigation bar 21
 - Sample left navigation bar 22
 - Sample bottom navigation bar 23

- Chapter 4: User welcome 24
 - About the Welcome page 24
 - Purpose 24
 - Recommended content 24
 - Sample Welcome page 24

- Chapter 5: Application sign in 26
 - Sign-in operation 26
 - Description 26
 - Sequence 26
 - Methods 27
 - Code examples 27
 - Recommendations 31
 - Recommended content for the Sign In page 31
 - Recommendation for navigation bars 31
 - Recommendations for signing out 32
 - Sample Sign In page 32

- Chapter 6: Registration and download 33
 - Registration operation 33
 - Description 33
 - Sequence 33
 - Methods 34
 - Registration code example 34
 - Description 34
 - Code example 35
 - Download operation 38
 - Description 38
 - Sequence 38
 - Methods 38
 - Download code example 39
 - Description 39

Code example	40
Recommendations	43
Recommended content	43
Sample Registration page	44
Sample Registration Confirmation page	44
Sample Download page	45
Chapter 7: Account management	46
Order media operation	46
Description	46
Sequence	46
Methods	47
Code example	47
Sample Application pages	51
Display operations	52
Description	52
Sequence	52
Methods	53
Code example	53
Change setting operations	55
Description	55
Sequence	55
Methods	56
Code example	56
Chapter 8: Retrieval	57
Retrieve operation	57
Description	57
Sequence	57
Methods	58
Code example	58
Data types and errors	59
Overview	60
About the data types	60
About error codes and events	60
AccountProfileField	60
AccountWebSiteSettings	61
AccountWebSiteSettings2	63
AccountWebsiteSettings3	66
AgentDownloadInfo	69

RegistrationInfo	69
RetrieveDownloadInfo	70
RetrieveFileInfo	70
RetrieveInfo	71
RetrieveInfo2	71
SSWSAccountHistory	72
SSWSAccountInfo	73
SSWSAccountSummaryInfo	74
SSWSAddressInfo	75
SSWSBackupDateInfo	75
SSWSBackupDateInfoWithoutSizes	76
SSWSBrandingInfo	77
SSWSBrandingInfo2	77
SSWSContactInfo	78
SSWSFileRevision	79
SSWSFileRevision2	79
SSWSFileRevision3	80
SSWSMediaType	81
SSWSNameInfo	82
SSWSProfileInfo	82
SSWSTicket	83
Returned errors	84
Event log errors	84
Error codes 2 through 8	84
Error codes 9 through 16	85
Error codes 17 through 24	86
Error codes 25 through 35	86
 Chapter 10: Registration operations	 88
Overview	88
GetCommunityType	88
GetRegistrationInfo	89
IsLicenseAvailable	90
IsLDAPCommunity	90
VerifyLDAPAccountRegistration	91
Return Value	92

VerifyAccountRegistrationCode	92
CanRegisterToCommunity	93
RegisterAccount	94
VerifyAccountReservationCode	96
VerifyAccountReservationCodeEx	97
Chapter 11: Download operations	98
Overview	98
GenerateAgentSetup	98
GenerateAgentSetupLocal	98
FetchAgentSetupBytes	100
Chapter 12: Sign-in operations	101
Overview	101
GetAccounts	101
GetFedAuthAccountsFromCommunity	102
GetFedAuthAccountsFromPartnerID	103
GetIDPURLForCommunity	103
GetIDPURLForPartnerId	104
SetLoginInformation	105
VerifyTechAccountAccess	106
Chapter 13: Management operations	107
Overview	107
GetAccountInfo	107
GetMediaTypes	108
SubmitMediaOrder	108
SetConfiguration	109
SetAccountPassword	110
SetAccountProfile	111
Chapter 14: Display operations	112
Overview	112
GetAppVersion	112
GetAccountSummaryInfo	113

GetAccountHistory	113
GetAccountWebSiteSettings	114
GetAccountWebSiteSettings2	115
GetAccountWebSiteSettings3	116
GetBrandingInfo	117
GetBackupDates	118
GetBackupDatesWithoutSizes	118
Chapter 15: Retrieve operations	120
Overview	120
FetchRetrieveBytes	120
FetchSingleFile	121
FindFiles	122
FindFiles2	123
GetFileList	124
GetFileList2	125
GetFilesCountForFolder	126
GetFoldersForTdate	127
GetFoldersForTdate2	128
GetMyRoamState	128
GetRetrieveSize	129
GetRetrieveSize2	130
IsMyRoamLicensed	130
RetrieveSelectedFiles	131
RetrieveSelectedFilesEx	132
Index	134
Send documentation feedback	137

Chapter 1: Concepts

This chapter describes basic concepts that you must understand before you build a custom Account Management application.

- [Account Management Website overview, below](#)
- [Recommended functions, on page 13](#)
- [Recommended content , on page 12](#)
- [Recommended functions, on page 13](#)

Account Management Website overview

Website purpose

The standard Account Management Website allows users to register and manage their backup accounts without intervention from system administrators. The standard Account Management Website is useful in the following situations:

- **Service provider model.** If you provide a backup service to external customers, you can use Support Center to create configurations for each service plan or customer type. When customers purchase your service, you can direct them to the Account Management Website where they register their account, download the software, and install the Agent, without assistance.
- **Enterprise model.** In a corporate environment, you can use the Account Management Website as a self-service tool for employees who need to use the Agent to back up their data. Use Support Center to create configurations for specific organizations or types of users. Employees access the Website, register their account, and install the software without a system administrator's assistance.

Standard Account Management Website

You can install the standard Account Management Website when you install the Data Center software. You can change the branding for this Website to match your corporate standards. The standard Website lets you perform the following functions:

- Register and download Agent software
- View a summary of backup statistics
- View details of backups and retrievals
- Order copies of backed-up files on CDs or DVDs

- Reinstall the Agent software
- Brand the Website to match corporate standards
- Edit or view profile information

Custom Account Management applications

If the standard Website does not meet your needs, you can use the Account Management Web Services Application Programming Interface (API) to create a custom Web-based application. The AMWS API does not support mobile applications.

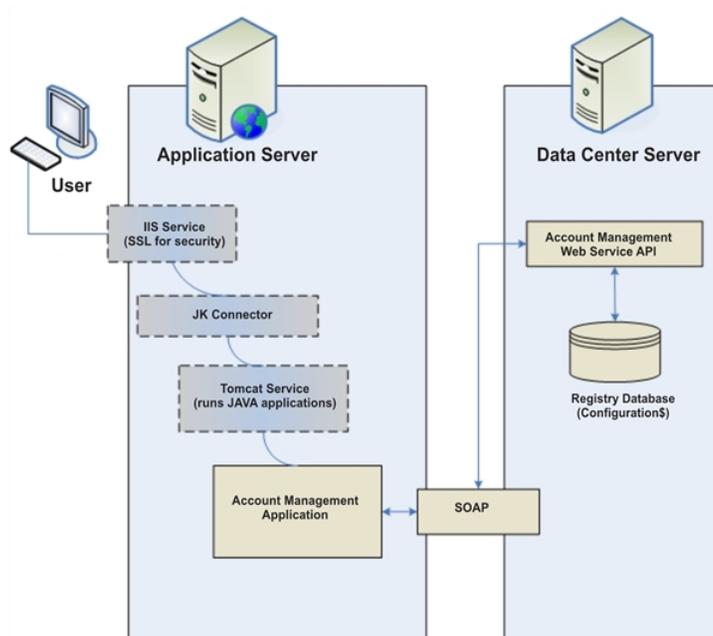
A custom application allows you to include features that are not part of the standard Website. For example, if you host a service for external users, you can add billing and transaction processing functionality. This type of functionality is not part of the standard Website and is not provided in the Account Management Web Services API. However, you can use your own API to build this functionality, and then call the Account Management Website API.

The Account Management Web Services API includes a method to retrieve branding information from the Data Center, so you can use the branding feature in Support Center to brand your custom application.

Application interaction with the Data Center

Communication flow

The following diagram shows an example of how you can design the communication flow for a custom Web-based Account Management application. Your implementation may require a different approach.



In this illustration, the following process occurs:

1. The Microsoft IIS Service enables you to use SSL for secure connections. If you do not want to create a Web-based application, you do not need this component.
2. The JK Connector forwards requests to the URL for the Web application server. In this example, the Jakarta Tomcat service runs the Account Management application. If you do not want to create a Web-based application, you do not need these components.
3. The Account Management application uses SOAP to communicate with the Account Management Web Service API that resides on the Data Center server.
4. The API communicates with the Registry Database on the Data Center server to store the account information, and then uses the SOAP interface to return data to the Account Management application.

Website settings in Support Center

The Account Management Web Services API includes methods that allow your application to retrieve Website Settings information from the Data Center. Therefore, you can use the Website Settings in Support Center to control the following options in your custom application:

- The information displayed in the registration and profile pages
- Whether users can order backed-up files on media
- Whether users can access the MyRoam application to retrieve files
- The Support contact information displayed in the Website

If your application is not designed to retrieve any or all of this information from the Data Center, you do not need to define these settings in Support Center.

Access user's account online link

When you install the Account Management Website, the Account Summary page in Support Center contains a link to the standard Account Management Website. It does not link to your custom Account Management application. If you have a license to use the MyRoam™ application, you can use this Website to retrieve files on behalf of the user.

However, you can use the Account Management Web Services API to link an Agent configuration to your custom application.

Recommended content

General content

If you create a custom Account Management application to use instead of the standard Website, you might want to include the following content in your application:

- Graphical overviews of the product.

Many users do not understand how the files from their computers are backed up and protected on a remote Data Center. A simple graphical diagram can help people understand the workflow for backups. Also include terminology and concepts, such as the difference between a first backup and an incremental backup. You also might want to include a product simulation or video that describes how to use the Agent.

- Support options.

Include a page that describes your support options, and let potential customers view this information before they select or purchase the backup service.

- About Us information

Provide an About Us section to give users information about your company or organization. If you provide a service to external customers, this information helps build a sense of confidence with users.

- Security feature description

Include information that describes the product's security features. On this page, emphasize the benefits of offsite storage in case of fire, flood, or hardware problems. You might want to provide answers for the following questions:

- How can I be sure that my data is safer with this service than it is in my home office?
- Who sees my data?
- What are my rights in case of an audit?
- What are my legal rights if my data is lost or temporarily unavailable?

- Frequently Asked Questions (FAQs)

- On this page, include information that helps users select the appropriate backup plan (if you offer different plans) or how to use the application.

System requirement content

If you plan to implement your application as a service to customers, include information about the following Agent system requirements:

- Processor speed
- Operating system
- RAM
- Disk space
- Recommended connection speeds for each backup plan that you offer
- Web browser version to access the Account Management Website

This information helps users make educated selections when they choose a backup plan.

Backup plan content

If you plan to implement your application as a service to customers, include the following information:

- Backup size estimation.

If you offer different backup plans with varying backup storage limits, provide instructions to help users estimate the amount of space they need. For example, you can include an explanation about how to use Windows Explorer to determine file and folder sizes.

- Chart of backup plans.

If you offer multiple backup plans, display them in a chart so users can compare the features of each plan.

Recommended functions

Introduction

The following sections describe recommended functions to include in your custom application. Depending on the functionality that you include in the application (registration, download, and account management to an internal enterprise or a backup service for external customers), you might also want to include components specific to your environment and needs.

Use the descriptions in the following sections as a guideline to build a useful application.

Sign in

To provide secure access to users' account information, include a logon component that requires users to enter their credentials.

By default, Account Management Website requires users with native or enterprise directory credentials to enter the e-mail address and password that they provided during registration. Users with single sign-on credentials must enter the credentials required by the third-party identity provider that authenticates their account. You can use this model or create one of your own.

Registration and download

This component directs users to enter profile information, specifically the credentials for the account. The type of credentials depend on the account type. Native and enterprise directory accounts require an e-mail address and password. Single sign-on accounts require the credentials specified by the third-party identity provider that authenticates the account.

After the registration completes, the application transmits this information to the Data Center, which creates the account and builds the custom Agent Setup file. The Agent Setup file includes the account number and community ID where the Agent is registered.

TIP:

Before you allow a user to register or download software, you might want to include an End User License Agreement window that users must accept.

Backup plan and billing

If you plan to offer different types of backup plans, for example, different plans based on storage limits, provide a component that allows users to select a plan. If necessary, add a transaction processing component to allow online payment for specific selections.

The Account Management Web Service does not include methods for transaction processing or billing. You must use a third-party package or create your own software to include this functionality in the application.

Account recovery

The Agent Setup file that the Data Center creates for a user is also the file that the user needs to recover an accounts. Include a component that allows users to download the same Agent Setup file that they downloaded when they originally registered their accounts.

Because the account number and community ID where the Agent originally registered is built in to the Agent Setup file, users only need to be able to access their account recovery link to download the correct file.

Change password

This component allows users with native Micro Focus Connected Backup accounts to change their own passwords, as needed, without a system administrator's intervention. Connected Backup does not store passwords for accounts with enterprise directory or single sign-on credentials. Therefore, this component does not support those types of accounts.

Change profile

This component allows users with native Connected Backup accounts to update their registration information at any time, without a system administrator's intervention. Connected Backup does not let users with enterprise directory or single sign-on account credentials update their profile. Therefore, this component does not support those types of accounts.

Order media

You can include a component in your application that can send a request to the DataBundler application. This functionality allows users to order a specific set of backed-up files on a CD or DVD. Users can then retrieve files to any location using the Media Agent. You might want to include a choice of delivery options, such as overnight, or two-day shipping.

NOTE:

Only users with accounts that run on a Microsoft Windows operating system can order media and retrieve files through the Media Agent.

Chapter 2: Development environment

This chapter contains information about the development environment.

- [Interfaces, below](#)
- [Available code examples, on the next page](#)
- [Method overview, on page 18](#)
- [Variable types, on page 19](#)

Interfaces

Account Management Web Services

The Account Management Web Services API is a stateless interface that allows a Website or application to communicate with a Data Center server. Stateless interfaces enable you to issue calls in any order without changing the state of the system or affecting other calls. The Account Management Web Services interface includes a Web Services Description Language (WSDL) file that defines the available methods, their parameters, and the data that they return.

This interface includes authenticated and unauthenticated methods.

Authenticated methods

These methods require you to provide authentication values whenever they are called. The authentication values include the following values:

- Community ID
- Account number
- Account password

Unauthenticated methods

These functions do not require any type of validation.

Simple Object Access Protocol

The Simple Object Access Protocol (SOAP) allows you to use HTTP and XML to pass information from one application to another. The Account Management Web Services uses SOAP to communicate with the Data Center server.

Find the Web Services API WSDL file

A WSDL file that describes the Web Services API is created on the server when you install the Account Management Website on a server. To locate the WSDL file, use the following URL on the server where you installed the Account Management Website software:

```
http|https://localhost/SSWSAPI/SSWSAPI.dll?Handler=GenWSDL
```

You can run a stub generator on the WSDL file to generate the appropriate stub files for the programming language you plan to use.

NOTE:

The WSDL file contains methods that are not described in this document. These methods are for use by Micro Focus. Do not use these methods in your custom applications.

Available code examples

Description

The Account Management Website software includes a set of code examples that you can use as a starting point for your application. The code examples are written in Java.

You also can use the source code for the standard Account Management Website as a starting point for your custom application.

Use the sample code

The code examples in this manual are written in Java and assume the use of Apache Axis to generate stub files from the WSDL file.

To compile and run the code examples

1. Download the sample code ZIP archive file from the Account Management Website installation folder:

```
C:\inetpub\wwwroot\SSWSAPI\ssws_examples.zip
```

2. Extract the files to a convenient folder.
3. Define system environment variables for the location of the Java software and add this variable to your PATH variable. For example:

```
set JAVA_HOME=C:\Program Files\Java\jdk1.5.0_06
```

```
set PATH=%JAVA_HOME%\bin;%PATH%
```

4. To compile and run each code example, enter commands similar to the following commands:

```
C:\sswsapisamples> setclasspath.cmd
```

```
C:\sswsapisamples> javac.exe example.java  
  
C:\sswsapisamples> java.exe example.java  
http://server/sswsapi/sswsapi.dll?Handler=Default
```

Use the standard online help files

You can use the existing online Help files as a starting point for your Help system. The Help files for the standard Account Management Website are in WebHelp format.

You can copy the existing HTML files from the following folder on the server where you installed the Data Center software:

```
DataCenter\apache-tomcat-7.0.42\webapps\ssws\help
```

Method overview

Sequence of calling methods

Because the Account Management Web Services API is stateless, you can call most methods in any order except for the following operations:

- Registration operations
- Logon operations
- Order media operations

Authenticated methods

Although the Account Management Web Service API is stateless, most methods require authentication. These methods must include the values in the `SSWS ticket` data type. This data type includes information about a user's account, including the account number, the community where the account is registered, and the password associated with the account. For a description of this data type, see [SSWSAddressInfo, on page 75](#).

The Account Management Web Service API includes the following methods:

- `GenerateAgentSetupLocal`. Sends a request to the Data Center server to create an Agent Setup file.
- `FetchAgentSetupBytes`. Downloads a specified number of bytes in the Agent Setup file.
- `VerifyTechAccountAccess`. Verifies that a specified technician can access a specific community.
- `GetAccountInfo`. Retrieves information about a specific account.
- `GetBackupDates`. Retrieves the available backup dates for a specific account.

- **SetAccountPassword.** Sets the password for the account. This method does not affect password values stored in an enterprise directory.
- **SetAccountProfile.** Changes the values stored in the profile.
- **GetMediaTypes.** Determines the media types available for a specific account when requesting a backup image.
- **SubmitMediaOrder.** Sends a request for a backup account image to the DataBundler application.

Variable types

The Account Management Web Services API uses the following variable types:

Simple type	Description
wsdl_base64Binary	Base64-encoded binary data. The API uses base64-encoded binary data to encode strings such as file paths and file names to protect them from corruption. The encoding ensure that file paths and file names that are not valid XML or SOAP strings do not result in errors.
wsdl_bool	A boolean value. For example: <ul style="list-style-type: none">• 1 or 0• True or false
wsdl_int	An integer.
wsdl_long	A long integer.
wsdl_string	A text string.
wsdl_unsignedLong	A nonnegative long integer.
wsdl_utc_time_in_msec	A variable that evaluates to a wsdl_long type. It is a 10-digit number that represents the number of milliseconds between January 1, 1970 12:00:00 AM UTC and a specified time.

Chapter 3: Application navigation

This chapter contains recommendations for the types of navigation to consider for your application.

- [Types of navigation, below](#)
- [Examples, on the next page](#)

Types of navigation

When you design your Web application, you must design a way for your users to navigate the different options in the application.

Top navigation

Add a top navigation bar to your application to display and provide access to the following information:

- The user's name
A message such as **Welcome** *User* where *User* is the name of the user.
- Account number
The account number associated with the user.
- Computer name
The name of the computer associated with an account.
- Sign Out link
A link that allows users to end a session and protect access to their account information.
- Help link
An online Help system to describe how to use the functions in your application. The Help system can be any type of HTML-based help system.

Left navigation

You can use a left navigation bar as the primary way to navigate to functions of your application. You might want to include links to the following options:

- Summary
- View History
- Retrieve Files with the MyRoam application

- Order CDs or DVDs
- Reinstall Agent
- Edit Profile
- Change Backup Plan
- View Accounts
- Support

You might want to hide the left navigation bar on all pages. For example, you could exclude the left navigation from the following pages:

- Registration
- Support contact
- Change password

Bottom navigation

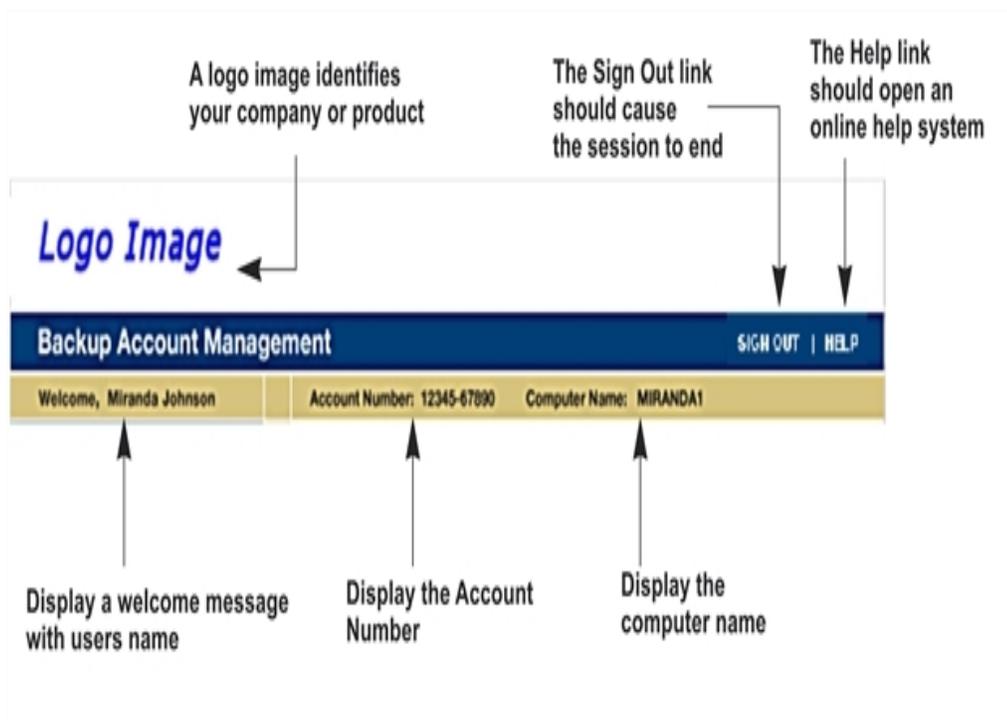
You can use a navigation bar at the bottom of your pages to provide links to your Terms of Condition, Privacy, and Security statements.

- On the Terms of Condition page, include information about the legal use of the application.
- On the Privacy Statement page, include information about privacy policies that your company uses to protect customer information.

Examples

Sample top navigation bar

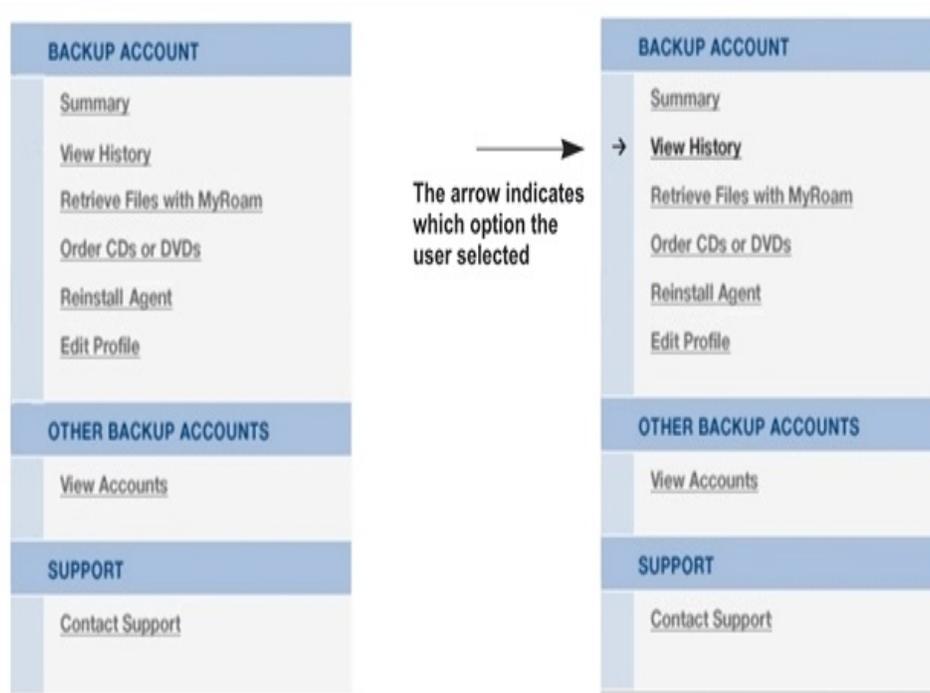
The following figure is an example of a top navigation bar:



You can use the [GetAccountInfo, on page 107](#) method to obtain information for the top navigation bar. This method returns the user name, the account number, and the computer name.

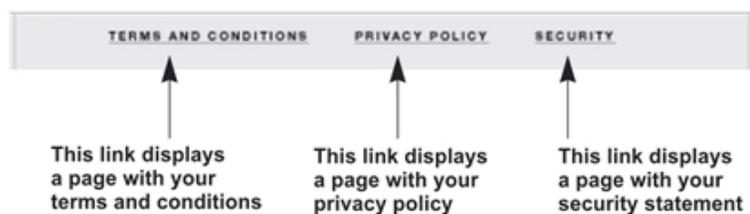
Sample left navigation bar

In the left navigation bar, include a visual cue that indicates the current selection. The following figure is example of a navigation bar before and after a user selects an option:



Sample bottom navigation bar

The following figures is an example of the bottom navigation bar:



Create pages that contain the appropriate content for your site, and use the bottom navigation bar to link to these pages.

Chapter 4: User welcome

This chapter contains recommendations for the first page in your application that users see.

- [About the Welcome page, below](#)

About the Welcome page

Purpose

The **Welcome** page provides basic information and links to help a users register and download an Agent Setup file.

There are no specific methods in the Account Management Web Services API for this purpose.

Recommended content

On the Welcome page, provide an introduction to your Account Management application and the backup software or service. For example, you might want to highlight the key features that the Agent provides. Also provide a link to the **Registration** page.

You also might want to include a link to your Sign In page for users that already have a registered backup account.

Sample Welcome page

The following figure is an example of a **Welcome** page:

Account Management SUPPORT | HELP

[HOME](#) [REGISTER](#) [SIGN IN](#)

Welcome

This Website gives you access to backup software and tools for managing your backup account. By using this software, you can securely protect your data from loss or damage.

Backup

The backup software that runs on your computer, called the Agent, automatically backs up your computer's information. It encrypts your data and transmits it safely over the Internet to a secure Data Center.

Retrieval

All your backed-up information is available to you whenever you need it.

Download Backup Software

To use this software, you must register a backup account and download the Agent.

[Register and Download](#)

Already registered? [Sign In](#) to view information about your backup account.

© Copyright 1996 - 2018 Micro Focus or one of its affiliates.

Chapter 5: Application sign in

This chapter contains recommendations for the sign in and sign out functions in your application.

- [Sign-in operation, below](#)
- [Recommendations, on page 31](#)

Sign-in operation

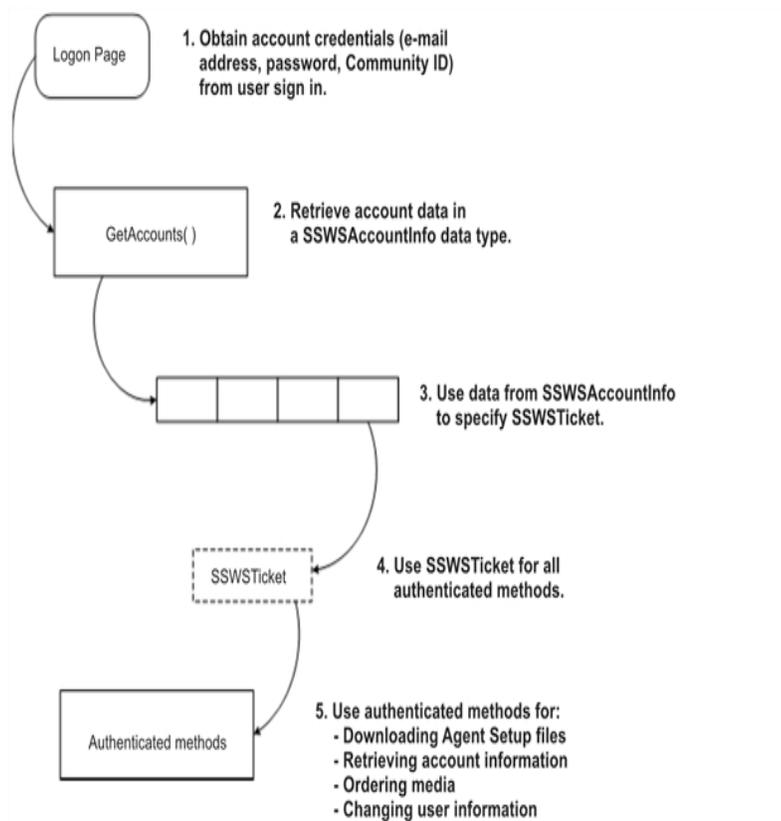
Description

The e-mail addressees associated with accounts are not unique. To allow users to perform various functions such as download the Agent Setup files or change their profile information after they sign in, the Account Management application must be able to authenticate users.

By passing the users' account credentials (e-mail address, password, and community ID) to the `GetAccounts` method, you can obtain the information required to create an authentication ticket (the `SSWSTicket` data type). The authentication ticket allows users to access and manage their accounts.

Sequence

The following figure is example of the sequence to obtain the information for user or technician authentication from an account with native or enterprise directory Connected Backup credentials.



Methods

For information about methods that you can use to configure sign in, see [Application sign in, on the previous page](#).

Code examples

This example uses the `GetAccounts` and the `GetAccountInfo` methods to obtain information to allow a user to sign in to a Web-based Account Management application.

Lines of code in **bold** text highlight data types or methods from the API.

```
import java.net.URL;  
  
import javax.xml.rpc.ServiceException;  
  
import org.apache.axis.AxisFault;  
  
import com.ironmountain.sswsapi.SSWSAPIServiceLocator;  
import com.ironmountain.sswsapi.SSWSAPIServiceSoap_PortType;  
import com.ironmountain.sswsapi.SSWSAccountInfo;  
import com.ironmountain.sswsapi.SSWSTicket;
```

```
/**
 * Example sign in to the SSWS API
 */

public final class LoginSSWSAPI {

    /**
     * Get the SOAP Service stub from the given URL.
     *
     * Notice that we need to maintain the session and the combination of Apache
     * Axis with a Web Service hosted by IIS as an ATL Web Service, requires
     * that the "sendMultiRefs" flag be sent as false.
     *
     * @param sswsapiUrl URL to connect to
     * @return Implementation of SOAP Service stub
     * @throws ServiceException
     */
    private static SSWSAPIServiceSoap_PortType getSoapService(final URL sswsapiUrl)
throws ServiceException {
        SSWSAPIServiceLocator locator = new SSWSAPIServiceLocator();
        locator.setMaintainSession(true);
        locator.getEngine().setOption("sendMultiRefs", Boolean.FALSE);
        final SSWSAPIServiceSoap_PortType soapService = locator.getSSWSAPIServiceSoap
(sswsapiUrl);
        return soapService;
    }

    /**
     * We have gotten an SOAP exception returned from AXIS. Extract the SSWS
     * return code from the exception if possible.
     * @param e AxisFault
     * @return SSWS API error code
     */
    private static int getApiReturnCode(AxisFault e) throws AxisFault {
        int soapErrorCode = -1;
        // Retrieve the error code if available
        if (e.getFaultDetails().length == 1) {
            String soapErrorDetailText = e.getFaultDetails()[0].getFirstChild
().getNodeValue();
            System.out.println("SSWS API Error Code: " + soapErrorDetailText + "
returned.");
            soapErrorCode = Integer.parseInt(soapErrorDetailText);
        } else {
            System.out.println("Unexpected error format. Incorrect number
of fields in the SOAP Fault" + " <detail> tag means it is not a valid SSWSAPI
return code.");
        }

        // Rethrow because we don't know what to do with the exception.
        throw e;
    }

    // Retrieve additional info. (NOTE: this is not translated, so
```

```
        // should be used for debugging only)
        String faultReasonText = e.getFaultReason();
        if (faultReasonText != null) {
            System.out.println("Additional info returned: " + faultReasonText);
        }
        return soapErrorCode;
    }

    /**
     * @param args - [url] [communityID] [emailAddress] [password]
     */
    public static void main(String[] args) {
        try {
            final URL sswsapiUrl;
            final int communityID;
            final String emailAddress;
            final String password;

            // 1. Read in command line arguments
            if (args.length == 0) {
                System.out.println("Attempting to log in with bad account
information");
                sswsapiUrl = new URL
("http://localhost:80/sswsapi/sswsapi.dll?Handler=Default");
                communityID = 1;
                emailAddress = "bogusEmail";
                password = "arbitraryPassword";
            } else if (args.length == 4) {
                sswsapiUrl = new URL(args[0]);
                communityID = Integer.parseInt(args[1]);
                emailAddress = args[2];
                password = args[3];
            } else {
                sswsapiUrl = null;
                communityID = 1;
                emailAddress = null;
                password = null;
                System.out.println("Usage: LoginSSWSAPI [url] [communityID]
[emailAddress] [password]");
                System.exit(2);
            }

            // 2. Instantiate a SOAP service stub using Apache Axis.
            final SSWSAPIServiceSoap_PortType soapService =
getSoapService(sswsapiUrl);

            // 3. Get the list of accounts which match the email / password
            // combination, and the root community. This is expected to fail.
            int soapErrorCode = 0;
            SSWSAccountInfo[] accountInfoArray = null;
            try {
                System.out.println("Retrieving all accounts for email address: " + emailAddress);
                accountInfoArray = soapService.getAccounts(communityID, emailAddress, password);
            }
        }
    }
}
```

```
    catch (AxisFault e) {
    soapErrorCode = getApiReturnCode(e);
    }

        if (soapErrorCode == 3 /* INVALID_LOGIN */) {
            System.out.println("An invalid login attempt has been made.");
            System.exit(0);
        }
        System.out.println(accountInfoArray.length + " accounts returned.");

// 4. Try to get account info for the first account returned.
if (accountInfoArray.length > 0) {
String accountNumber = accountInfoArray[0].getAccountNumber();
System.out.println("Trying to get account status information for account: "
+ accountNumber);

SSWSTicket ticket = new SSWSTicket(
0 /* Customer authentication type */,
communityID,
accountNumber,
"" /* internal use field */,
password,
0 /* internal use field */,
0 /* internal use field */
);

SSWSAccountInfo accountInfo = null;
try {
accountInfo = soapService.getAccountInfo(ticket);
        } catch (AxisFault e) {
            soapErrorCode = getApiReturnCode(e);
        }

        if (soapErrorCode != 0 /* SUCCESS */) {
            System.out.println("Failure to get account info
for account: " + accountNumber);
            System.exit(0);
        }

String statusCode = accountInfo.getBsAccountStatus();
if (statusCode.compareToIgnoreCase("A") == 0) {
    System.out.println("The account is active.");
} else if (statusCode.compareToIgnoreCase("H") == 0) {
    System.out.println("The account is on hold.");
} else if (statusCode.compareToIgnoreCase("C") == 0) {
    System.out.println("The account has been cancelled.");
} else {
    System.out.println("Unknown account status.");
}

if (accountInfo.isVersion80rHigher() == false) {
    System.out.println("Account agent version < 8.0.
```

```
Some SSWS API activity may be unavailable.");
    }

    if (accountInfo.isLDAP() == true) {
        System.out.println("Account is LDAP.
Profile data will be read-only.");
    }
}

    System.exit(0);
} catch (Exception e) {
    e.printStackTrace(System.out);
    System.exit(1);
}
}
}
```

Recommendations

Recommended content for the Sign In page

The items you display on your Sign In page depend on the type of user account:

- For an account that uses native or enterprise directory credentials:
 - **Email address text box.** Users enter the e-mail address stored in their profiles.
 - **Password text box.** Users enter the password stored in their profiles.
 - **Link to instructions to request a forgotten password.** Typically, a technician needs to reset the password for the user.
 - **Link to your Support Contact page.** Users might need assistance to sign in to the application.
- For an account that uses single sign-on (SSO) credentials:
 - **SSO Sign In page.** Display the SSO Sign In page from the third-party identity provider (IdP) that supports the community in which the user account resides. Users enter the IdP-specific account credentials.
 - **Link to your Support Contact page.** Users might need assistance to sign in to the application.

Recommendation for navigation bars

You might not want to display the left or bottom navigation bars until after the user signs in to the application. Until your application authenticates the user's account credentials, you do not want the user to view account information.

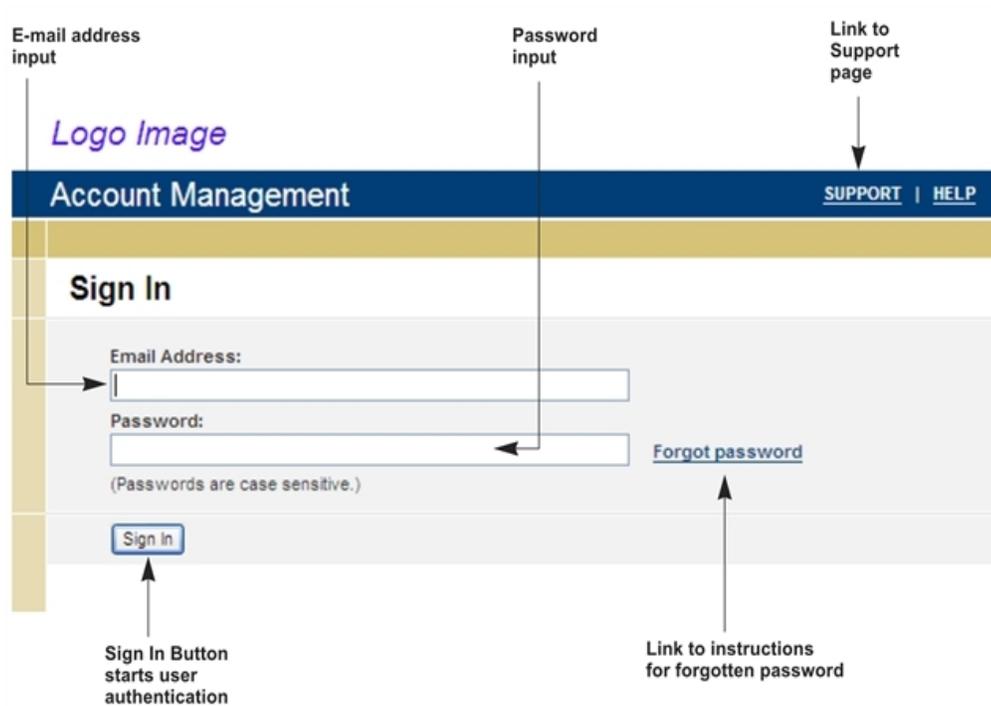
To ensure that users end their sessions properly, you might want to include a **Sign Out** link on one of your navigation bars.

Recommendations for signing out

When users click a **Sign Out** link, you might want to display a page that confirms that the session ended successfully. Or, you might want to redisplay the **Sign In** page to allow users to sign in again.

Sample Sign In page

The following illustration shows an example of a **Sign In** page:



Chapter 6: Registration and download

This chapter contains recommendations for registration and Agent Setup file download operations.

- [Registration operation, below](#)
- [Registration code example, on the next page](#)
- [Download operation, on page 38](#)
- [Download code example, on page 39](#)
- [Recommendations, on page 43](#)

Registration operation

Description

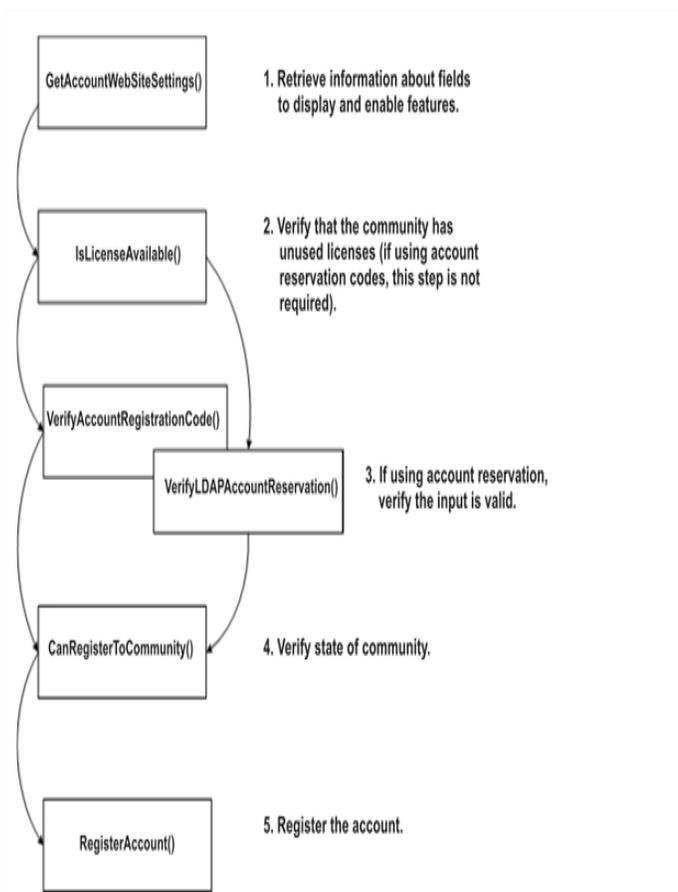
Registration allows users to register an account in a community in the Data Center. After a user registers an account, the user can download the Agent Setup file, install the Agent, and back up files.

You can use the following types of registration:

- **Open registration.** A user with either native or enterprise directory credentials enters their e-mail address and password during registration.
- **SSO registration.** A user with single sign-on (SSO) credentials enters their credentials into the third-party identity provider (IdP) SSO Sign In page that the Account Management WebSite (AMWS) displays. The IdP that authenticates the account returns an OAuth token that AMWS uses to create the account.
- **Native account reservation.** A technician creates one or more account reservation codes in Support Center. The user enters the account reservation code during reservation.
- **Enterprise directory reservation.** If the community where the account is registered is mapped to an enterprise directory, the user enters the enterprise directory user name and password during registration.
- **SSO reservation.** If the community where the account is registered is configured to use single sign-on, the user enters their SSO account credentials. After the IdP authenticates the credentials, the reserved account is associated with the SSO user.

Sequence

The following figure shows a sample design a sequence for registration:



Methods

For information about the methods that you can use to configure registration, see [Registration and download, on the previous page](#)

Registration code example

Description

The following example shows how you can use the Account Management Web Service API to allow a user to register a new account. This example shows how to create the datatypes that contain the registration information that a user supplies, and how to use the RegisterAccount method to register the account.

Code example

```
import java.net.URL;

import javax.xml.rpc.ServiceException;

import org.apache.axis.AxisFault;

import com.ironmountain.sswsapi.SWSAPIServiceLocator;
import com.ironmountain.sswsapi.SWSAPIServiceSoap_PortType;
import com.ironmountain.sswsapi.SWSAddressInfo;
import com.ironmountain.sswsapi.SWSContactInfo;
import com.ironmountain.sswsapi.SWSNameInfo;
import com.ironmountain.sswsapi.SWSProfileInfo;

public class RegisterAccountSSWSAPI {

    /**
     * Get the SOAP Service stub from the given URL.
     *
     * Notice that we need to maintain the session and the combination of Apache
     * Axis with a Web Service hosted by IIS as an ATL Web Service, requires
     * that the "sendMultiRefs" flag be sent as false.
     *
     * @param sswsapiUrl
     *         URL to connect to
     * @return Implementation of SOAP Service stub
     * @throws ServiceException
     */
    private static SWSAPIServiceSoap_PortType getSoapService(final URL sswsapiUrl)
throws ServiceException {
        SWSAPIServiceLocator locator = new SWSAPIServiceLocator();
        locator.setMaintainSession(true);
        locator.getEngine().setOption("sendMultiRefs", Boolean.FALSE);
        final SWSAPIServiceSoap_PortType soapService = locator.getSSWSAPIServiceSoap
(sswsapiUrl);
        return soapService;
    }

    /**
     * We have gotten an SOAP exception returned from AXIS. Extract the SSWS
     * return code from the exception if possible.
     *
     * @param e -
     *         AxisFault
     * @return - SSWS API error code
     */
    private static int getApiReturnCode(AxisFault e) throws AxisFault {
        int soapErrorCode = -1;
        // Retrieve the error code if available
        if (e.getFaultDetails().length == 1) {
```

```
String soapErrorDetailText = e.getFaultDetails()[0].getFirstChild()
.getNodeValue();
System.out.println("SSWS API Error Code: " + soapErrorDetailText + "
returned.");
soapErrorCode = Integer.parseInt(soapErrorDetailText);
} else {
System.out.println("Unexpected error format. Incorrect number of
fields in the SOAP Fault"
+ " <detail> tag means it is not a valid SSWSAPI return code.");

// Rethrow because we don't know what to do with the exception.
throw e;
}

// Retrieve additional info. (NOTE: this is not translated, so
// should be used for debugging only)
String faultReasonText = e.getFaultReason();
if (faultReasonText != null) {
System.out.println("Additional info returned: " + faultReasonText);
}

return soapErrorCode;
}

/**
 * New account registration example.
 *
 * @param args - [url] [communityID] [configID] [emailAddress] [password]
 */
public static void main(String[] args) {
    try {
        final URL sswsapiUrl;
        final int communityID;
        final int configID;
        final String emailAddress;
        final String password;

        // 1. Read in command line arguments
        if (args.length == 0) {
            System.out.println("Attempting to register with default
information");

            sswsapiUrl = new URL
("http://localhost:80/sswsapi/sswsapi.dll?Handler=Default");
            communityID = 1;
            configID = 5;
            emailAddress = "bogusEmail";
            password = "youllNeverGuessMe";
        } else if (args.length == 5) {
            sswsapiUrl = new URL(args[0]);
            communityID = Integer.parseInt(args[1]);
            configID = Integer.parseInt(args[2]);
            emailAddress = args[3];
            password = args[4];
        }
    }
}
```

```
        } else {
            sswsapiUrl = null;
            communityID = 1;
            configID = 5;
            emailAddress = null;
            password = null;
            System.out.println("Usage: RegisterAccountSSWSAPI [url]
[communityID] [configID] [emailAddress] [password]");
            System.exit(2);
        }

        // 2. Instantiate a SOAP service stub using Apache Axis.
        final SSWSAPIServiceSoap_PortType soapService =
getSoapService(sswsapiUrl);

        // 3. Construct example registration information
        final SSWSAddressInfo addressInfo = new SSWSAddressInfo("address1", "address2",
"address3", "city", "state", "zip", "country");

        final SSWSContactInfo contactInfo = new SSWSContactInfo("telephone", "extension",
emailAddress, "location", "mailStop", addressInfo);

        final SSWSNameInfo nameInfo = new SSWSNameInfo("firstName", "middleName",
"lastName");

        final SSWSProfileInfo profileInfo = new SSWSProfileInfo("companyName",
"department",
"costCenter", "employeeID", contactInfo, "customFieldValue", nameInfo);

        // 4. Register an account with this registration info
        int soapErrorCode = 0;
        int accountNumber = -1;
        try {
            System.out.println("Registering account");
            accountNumber = soapService.registerAccount(communityID,
configID,
emailAddress, /* Username */
password,
profileInfo);
        } catch (AxisFault e) {
            soapErrorCode = getApiReturnCode(e);
        }

        // 5. Check return code to ensure nothing bad happened
        if (soapErrorCode != 0 /* SUCCESS */) {
            System.out.println("Registration has failed.");
            System.exit(0);
        }

        System.out.println("Registration has succeeded. Account " +
accountNumber + " has been created.");
        System.exit(0);
    } catch (Exception e) {
```

```
        e.printStackTrace(System.out);  
        System.exit(1);  
    }  
}  
}
```

Download operation

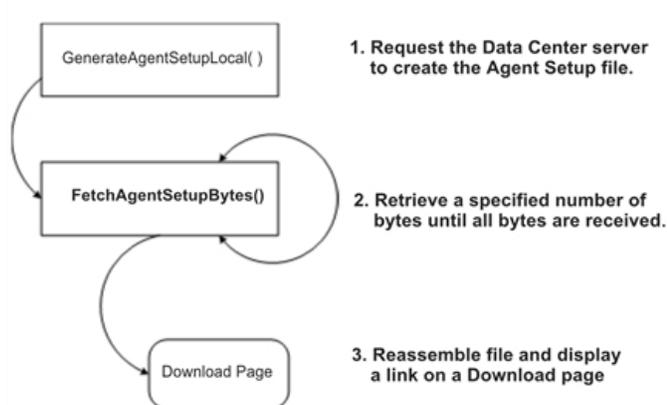
Description

The download operations enable your application to request a customized Agent Setup file for the registered account. The Agent Setup file includes an account number and the community ID.

To control the performance of the download, you can call the `FetchAgentSetupBytes` method multiple times and then reassemble the file after you download the bytes.

Sequence

The following figure shows an example of the sequence that you can use to download the Agent Setup file:



Methods

For information about the methods that you can use to configure download operations, see [Registration and download, on page 33](#).

Download code example

Description

The following example shows how you can use the Account Management Web Service API to allow a user to sign in to the Account Management Service and download an Agent Setup file for an account recovery.

This example shows how to create a `SSWSTicket` data type, use the `GenerateSetup` method to generate an Agent Setup file, and use the `FetchAgentSetupBytes` method to download the file.

Code example

```
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.net.URL;

import javax.xml.rpc.ServiceException;

import org.apache.axis.AxisFault;

import com.ironmountain.sswsapi.AgentDownloadInfo;
import com.ironmountain.sswsapi.SWSAPIServiceLocator;
import com.ironmountain.sswsapi.SWSAPIServiceSoap_PortType;
import com.ironmountain.sswsapi.SSWSTicket;

/**
 * Example sign in to the Account Management WebSite and download an Agent Setup file
 */

public final class AgentDownloadSSWSAPI {

    /**
     * Get the SOAP Service stub from the given URL.
     *
     * Notice that we need to maintain the session and the combination of Apache
     * Axis with a Web Service hosted by IIS as an ATL Web Service, requires
     * that the "sendMultiRefs" flag be sent as false.
     *
     * @param sswsapiUrl
     *         URL to connect to
     * @return Implementation of SOAP Service stub
     * @throws ServiceException
     */
    private static SWSAPIServiceSoap_PortType getSoapService(final URL sswsapiUrl) throws
        ServiceException {
        SWSAPIServiceLocator locator = new SWSAPIServiceLocator();
        locator.setMaintainSession(true);
        locator.getEngine().setOption("sendMultiRefs", Boolean.FALSE);
        final SWSAPIServiceSoap_PortType soapService = locator.getSSWSAPIServiceSoap(sswsapiUrl);
        return soapService;
    }

    /**
     * We have gotten an SOAP exception returned from AXIS. Extract the SSWS
     * return code from the exception if possible.
     *
     * @param e -
     *         AxisFault
     * @return - SSWS API error code
     */
}
```

```
private static int getApiReturnCode(AxisFault e) throws AxisFault {
    int soapErrorCode = -1;
    // Retrieve the error code if available
    if (e.getFaultDetails().length == 1) {
        String soapErrorDetailText = e.getFaultDetails()[0].getFirstChild()
        .getNodeValue();
        System.out.println("SSWS API Error Code: " + soapErrorDetailText + " returned.");
        soapErrorCode = Integer.parseInt(soapErrorDetailText);
    } else {
        System.out.println("Unexpected error format. Incorrect number of
fields in the SOAP Fault" + " <detail> tag means it is not a valid SSWSAPI return code.");
        // Rethrow because we don't know what to do with the exception.
        throw e;
    }

    // Retrieve additional info. (NOTE: this is not translated, so
// should be used for debugging only)
    String faultReasonText = e.getFaultReason();
    if (faultReasonText != null) {
        System.out.println("Additional info returned: " + faultReasonText);
    }

    return soapErrorCode;
}

/**
 * @param args
 */
public static void main(String[] args) {
    try {
        final URL sswsapiUrl;
        final int communityID;
        final String password;
        final String accountNumber;
        final String outputDirectory;

        // 1. Read in command line arguments

        if (args.length == 0) {
            System.out.println("Attempting to sign in with bad account
information");

            sswsapiUrl = new URL("http://localhost:80/sswsapi/sswsapi.dll?Handler=Default");
            communityID = 1;
            password = "groggie";
            accountNumber = "101000001";
            outputDirectory = "C://Temp/";
        } else if (args.length == 5) {
            sswsapiUrl = new URL(args[0]);
            communityID = Integer.parseInt(args[1]);
            password = args[2];
            accountNumber = args[3];
            outputDirectory = args[4];
        } else {
            sswsapiUrl = null;
        }
    }
}
```

```
        communityID = 1;
        password = null;
        accountNumber = null;
        outputDirectory = null;
        System.out.println("Usage: AgentDownloadSSWSAPI [url]
[communityID] [password] [accountNumber] [msiOutputDir]");
        System.exit(2);
    }

    // 2. Instantiate a SOAP service stub using Apache Axis.

    final SSWSAPIServiceSoap_PortType soapService =
getSoapService(sswsapiUrl);

    // 3. Build a sign in ticket object to store the authentication
information

SSWSTicket ticket = new SSWSTicket(0, communityID, accountNumber, "", password, 0,
0);

    // 4. Generate the agent setup on the datacenter

    AgentDownloadInfo downloadInfo = null;
    int soapErrorCode = 0;
    try {
        System.out.println("Generating agent setup");
downloadInfo = soapService.generateAgentSetup(ticket, false);
    } catch (AxisFault e) {
        soapErrorCode = getApiReturnCode(e);
    }

    if (soapErrorCode == 3 /* INVALID_LOGIN */) {
        System.out.println("Invalid sign in information was supplied.");
        System.exit(0);
    }

    // 5. Transfer the agent setup to the SOAP server

    final long size = downloadInfo.getSize().longValue();
    int start = 0;
    try {
        final File fObj = new File(outputDirectory + "tmp.msi");
        if (!fObj.exists()) {
            fObj.createNewFile();
        }
        BufferedOutputStream f = new BufferedOutputStream(
new FileOutputStream(outputDirectory + "tmp.msi"));
        int count = 1024 * 1024;
        while (start < size) {
            final int start1 = start;
            final int count1 = count;
            try {
                byte fileBuffer[] = soapService.
```

```
fetchAgentSetupBytes(ticket, downloadInfo,
                    start1, count1);
Integer.toString(fileBuffer.length)
Integer.toString(start));
                    f.write(fileBuffer);
                    } catch (AxisFault e) {
                    soapErrorCode = getApiReturnCode(e);
                    }
                    start += count;
                }
                f.flush();
                f.close();
                System.out.println("Agent download retrieve complete: "
+ fObj.toString());
                } catch (IOException e) {
                System.out.println("RegistrationCompletePage::getSetupBuffer:
Failed to write fetched bytes to the setup file. " + e.getMessage());
                }
                } catch (Exception e) {
                e.printStackTrace(System.out);
                System.exit(1);
                }
            }
        }
```

Recommendations

Recommended content

When you design your registration and download pages, consider the following recommendations:

- Include the fields that you want a new user to complete before the user is able to download an account. You can get the Website Settings from Support Center (using the `GetAccountWebSiteSettings` method) to determine which standard fields to display and their state (read-only, editable, or required), or you can create your own profile fields. For an example of a Registration page, see [Sample Registration page, on the next page](#).
- After registration is complete, display a Confirmation page with the account information. This helps users know the state of the registration and whether they are ready to go to the next step in the process. For an example of a Registration Confirmation page, see [Sample Registration Confirmation page, on the next page](#).
- After registration is complete, your application must contact the Data Center with a request to generate an Agent Setup file for the registered user. Use the `GenerateAgentSetupLocal` method. While the file generation is in progress, display a progress page so that the user is aware that work is in progress.

- Include a procedure that describes how to download the Agent Setup file. Design these instructions for your target audience. For example, a less-experienced audience needs more detailed instructions than a more technical or experienced audience. For an example of a Download page, see [Sample Download page, on the next page.](#)

Sample Registration page

The following illustration shows an example of a registration page:

The screenshot shows a registration page for 'Account Management'. The page has a blue header with 'Account Management' and 'SUPPORT | HELP' links. Below the header, there are two tabs: '1. REGISTER' (active) and '2. DOWNLOAD and INSTALL'. The main heading is 'Registration - Enter Registration Information'. A red asterisk indicates required fields. The form is divided into three sections: 'Name' with fields for First name, Middle name, and Last name; 'Sign-In Information' with fields for Email address, Password, and Confirm password, accompanied by two warning icons and messages: 'You need an email address to access your account online.' and 'Passwords are case sensitive and must contain at least 6 characters.'; and 'Contact Information' with fields for Company, Country (dropdown menu showing 'United States'), Address line 1, 2, and 3, City, State (dropdown menu showing '-- Select One --'), Zip Code, and Phone number. At the bottom, there are 'Continue' and 'Cancel' buttons, and a copyright notice: '© Copyright 1996 - 2018 Micro Focus or one of its affiliates.'

Sample Registration Confirmation page

The following illustration shows an example of a registration confirmation page.

The screenshot shows a web page titled "Account Management" with a dark blue header. In the top right corner of the header are links for "SUPPORT" and "HELP". Below the header is a yellow navigation bar with "1. REGISTER" and "2. DOWNLOAD and INSTALL". The main content area has a white background with a large heading "Registration Complete". Below this heading is a message: "Your backup account is registered. You can download the Agent software and install it on your computer now." This is followed by a section titled "Backup Account Information" containing the following details: "Account number: 10101-34554", "Name: Miranda Johnson", and "Email address: mjohnson@mycompany.com". Below the information is a link "Print this information for your records." and two buttons: "Print" and "Download Software".

Sample Download page

The following illustration shows an example of a download page.

The screenshot shows a web page titled "Account Management" with a dark blue header. In the top right corner of the header are links for "SUPPORT" and "HELP". Below the header is a yellow navigation bar with "1. REGISTER" and "2. DOWNLOAD and INSTALL". The main content area has a white background with a large heading "Download Instructions". Below this heading is a section titled "Downloading with Internet Explorer" with the text: "If you are using Internet Explorer, do the following to download and install the Agent:" followed by a numbered list: "1. To start the download, click **Begin Download**.", "2. Click **Save** in the File Download box to save the Agent Installer to your Desktop.", and "3. When the download completes, click **Open** in the Download Complete dialog to run the installer." Below this is another section titled "Downloading with Other Browsers" with the text: "If you are using any browser other than Internet Explorer, save the Agent Installer to an easy-to-find location, such as your Desktop. Double-click the file to start the installation." At the bottom of the page is a button labeled "Begin Download".

Chapter 7: Account management

This chapter contains recommendations for account management operations.

- [Order media operation, below](#)
- [Display operations, on page 52](#)
- [Change setting operations, on page 55](#)

Order media operation

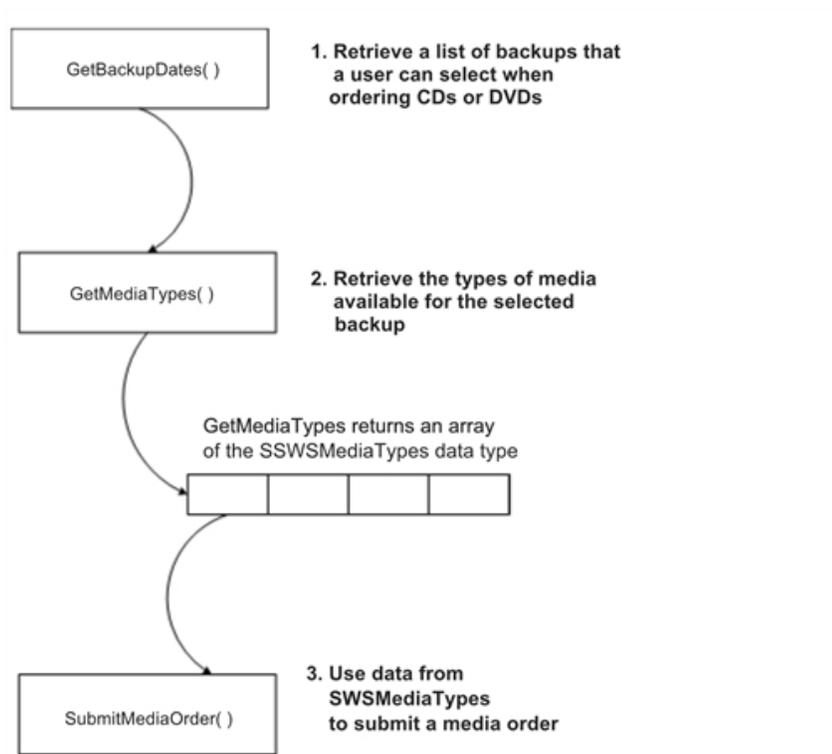
Description

To order media for a selected backup, you need a sequence that requires communication with the Data Center server.

After your application submits an order, the Data Center sends these requests to the Data Bundler application. After the DataBundler application creates the image, you can burn it to a CD or DVD. Users can access the Agent user interface on the media and retrieve the files in the backup image.

Sequence

The following figure shows a sequence that you can use to order media for a backup account image:



Methods

For information about the methods that you can use to order media, see [Management operations, on page 107](#).

Code example

The following code example shows how to use the `GetAccounts`, `GetBackupDates`, `GetMediaTypes`, and `SubmitMediaOrder` methods to order a specific set of backed-up files on CDs.

```
import java.net.URL;

import javax.xml.rpc.ServiceException;

import org.apache.axis.AxisFault;

import com.ironmountain.sswsapi.SWSAPIServiceLocator;
import com.ironmountain.sswsapi.SWSAPIServiceSoap_PortType;
import com.ironmountain.sswsapi.SWSAccountInfo;
import com.ironmountain.sswsapi.SWSBackupDateInfo;
import com.ironmountain.sswsapi.SWSMediaTypes;
import com.ironmountain.sswsapi.SWSTicket;

/**
 * Example sign in to the Account Management WebSite
 */
```

```
public final class OrderMediaSSWSAPI {

    /**
    * Get the SOAP Service stub from the given URL.
    *
    * Notice that we need to maintain the session and the combination of Apache
    * Axis with a Web Service hosted by IIS as an ATL Web Service, requires
    * that the "sendMultiRefs" flag be sent as false.
    *
    * @param sswsapiUrl URL to connect to
    * @return Implementation of SOAP Service stub
    * @throws ServiceException
    */
    private static SSWSAPIServiceSoap_PortType getSoapService(final URL sswsapiUrl) throws
    ServiceException {
        SSWSAPIServiceLocator locator = new SSWSAPIServiceLocator();
        locator.setMaintainSession(true);
        locator.getEngine().setOption("sendMultiRefs", Boolean.FALSE);
        final SSWSAPIServiceSoap_PortType soapService = locator.getSSWSAPIServiceSoap(sswsapiUrl);
        return soapService;
    }

    /**
    * We have gotten an SOAP exception returned from AXIS. Extract the SSWS
    * return code from the exception if possible.
    * @param e - AxisFault
    * @return - SSWS API error code
    */
    private static int getApiReturnCode(AxisFault e) throws AxisFault {
        int soapErrorCode = -1;
        // Retrieve the error code if available
        if (e.getFaultDetails().length == 1) {
            String soapErrorDetailText = e.getFaultDetails()[0].getFirstChild().getNodeValue();
            System.out.println("SSWS API Error Code: " + soapErrorDetailText + " returned.");
            soapErrorCode = Integer.parseInt(soapErrorDetailText);
        } else {
            System.out.println("Unexpected error format. Incorrect number of fields in the SOAP Fault"
            + " <detail> tag means it is not a valid SSWSAPI return code.");
            // Rethrow because we don't know what to do with the exception.
            throw e;
        }

        // Retrieve additional info. (NOTE: this is not translated, so
        // should be used for debugging only)
        String faultReasonText = e.getFaultReason();
        if (faultReasonText != null) {
            System.out.println("Additional info returned: " + faultReasonText);
        }

        return soapErrorCode;
    }

    /**
```

```
 * Entry point for OrderMediaSSWSAPI example  
 * @param args - [url] [communityID] [emailAddress] [password]  
 */  
  
public static void main(String[] args) {  
    try {  
        final URL sswsapiUrl;  
        final int communityID;  
        final String emailAddress;  
        final String password;  
        final String shippingLabel = "My Company\n555 Somewhere St\nCity, State  
PostalCode";  
  
        // 1. Read in command line arguments  
  
        if (args.length == 0) {  
            System.out.println("Attempting to sign in with bad account information");  
            sswsapiUrl = new URL  
("http://localhost:80/sswsapi/sswsapi.dll?Handler=Default");  
            communityID = 1;  
            emailAddress = "bogusEmail";  
            password = "arbitraryPassword";  
        } else if (args.length == 4) {  
            sswsapiUrl = new URL(args[0]);  
            communityID = Integer.parseInt(args[1]);  
            emailAddress = args[2];  
            password = args[3];  
        } else {  
            sswsapiUrl = null;  
            communityID = 1;  
            emailAddress = null;  
            password = null;  
            System.out.println  
("Usage: LoginSSWSAPI [url] [communityID] [emailAddress] [password]");  
            System.exit(2);  
        }  
  
        // 2. Instantiate a SOAP service stub using Apache Axis.  
  
        final SSWSAPIServiceSoap_PortType soapService = getSoapService(sswsapiUrl);  
  
        // 3. Get the list of accounts which match the email / password  
        // combination, and the root community. This is expected to fail.  
  
        int soapErrorCode = 0;  
        SSWSAccountInfo[] accountInfoArray = null;  
        try {  
            System.out.println("Retrieving all accounts for email address: " +  
            emailAddress);  
            accountInfoArray = soapService.getAccounts(communityID, emailAddress,  
            password);  
        }
```

```
    } catch (AxisFault e) {
        soapErrorCode = getApiReturnCode(e);
    }

    if (soapErrorCode == 3 /* INVALID_LOGIN */) {
        System.out.println("An invalid sign in attempt has been made.");
        System.exit(0);
    }

    System.out.println(accountInfoArray.length + " accounts returned.");

    if (accountInfoArray.length > 0) {

        // 4. Try to get backup dates for the first account returned.

        String accountNumber = accountInfoArray[0].getAccountNumber();
        System.out.println("Trying to get backup dates for account: " + accountNumber);
        SSWSTicket ticket = new SSWSTicket(0 /* Customer authentication type */,
            communityID,
            accountNumber,
            "" /* Leave technician ID field null for user authentication. */,
            password,
            0 /* field only valid for technician password hashes */,
            0 /* field only valid for technician password hashes */
        );

        SWSBackupDateInfo[] backupDates = null;
        try {
            backupDates = soapService.getBackupDates(ticket);
        } catch (AxisFault e) {
            soapErrorCode = getApiReturnCode(e);
        }

        if (soapErrorCode == 8 /* NO_DATA */) {
            System.out.println("No backup dates for account: " + accountNumber);
            System.exit(0);
        } else if (soapErrorCode != 0 /* SUCCESS */) {
            System.out.println("Failure to get backup dates for account: " + accountNumber);
            System.exit(0);
        }

        System.out.println(backupDates.length + " dates returned.");

        // 5. Get the available media types we can order

        SWSMediaType[] mediaTypes = null;
        try {
            mediaTypes = soapService.getMediaTypes(ticket,
                backupDates[0].getTBackupTimeInMSec());
        } catch (AxisFault e) {
            soapErrorCode = getApiReturnCode(e);
        }

        //
    }
}
```

6. Try to order media from the first backup date

```
        try {
            soapService.submitMediaOrder(ticket,
                mediaTypes[0].getIMediaId(),
                shippingLabel,
                1, /* Use fixed shipping priority */
                password,
                backupDates[0].getTBackupTimeInMSec());
        } catch (AxisFault e) {
            soapErrorCode = getApiReturnCode(e);
        }

        if (soapErrorCode != 0 /* SUCCESS */) {
            System.out.println("Failure to submit media order.");
            System.exit(0);
        }

        System.out.println("Media order submitted.");
    }

    System.exit(0);
} catch (Exception e) {
    e.printStackTrace(System.out);
    System.exit(1);
}
}
```

Sample Application pages

The following illustration shows an example of the pages in an Order Media wizard that uses the Account Management Web Service API:

Order CDs and DVDs

Order a CD or DVD set that contains a copy of each file in your backup set as of a selected backup date. The status of a backup is indicated in the following list.

If you select a backup configured for full backup, you can order a CD or DVD set.

*** Indicates required fields.**

Step 1: Choose Backup Date

Available backup dates:

- Wed, 2/20/13, 8:12:00 AM
- Tue, 2/19/13, 8:12:00 AM
- Fri, 2/15/13, 8:12:00 AM
- Wed, 2/13/13, 8:12:00 AM
- Tue, 2/12/13, 8:12:00 AM
- Mon, 2/11/13, 8:12:00 AM
- Sun, 2/10/13, 5:12:00 AM
- Fri, 2/8/13, 8:12:00 AM
- Thu, 2/7/13, 8:12:00 AM
- Wed, 2/6/13, 8:12:00 AM

Hide incomplete backup dates

Step 2: Choose Media Type

DVD (Estimated number of discs in order:1)

CD (Estimated number of discs in order:3)

NOTE: The Media creation process can produce a number of discs that differs from the displayed estimate. The number of discs in the shipped media order is the correct number of discs for the order.

Step 3: Set Media Password

Save this password in a secure location. **After you click Submit Order, you cannot recover the password.**

*Media Password:

*Confirm Media Password:

Step 4: Provide a Shipping Label

* Shipping Label

Ship to this address:

Miranda Johnson
123 Main Street
Anytown, MA 01234

Ship to new address:

Full Name:

Address Line 1:

Address Line 2:

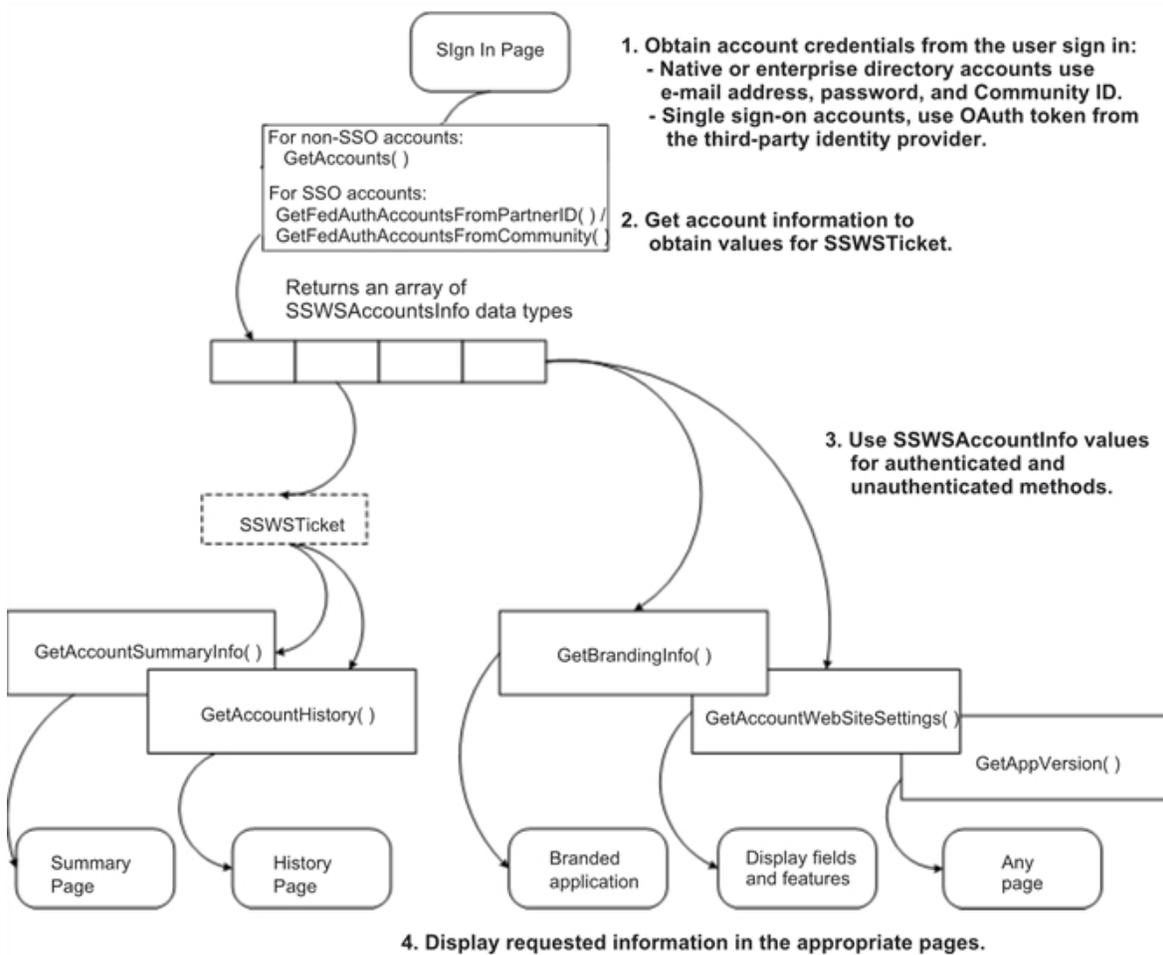
Display operations

Description

You can use the Account Management Web Service API to change the appearance of your application, to change the fields and features in your application, or to display specific types of status and history information.

Sequence

The following figure shows an example of how to use the Account Management Web Service API to change the appearance of your application.



Methods

For information about the methods that you can use to display information in your application, see [Display operations, on page 112](#).

Code example

The following example shows how to use the `GetAppVersion` method to connect to a Data Center server and obtain the software version of the installed software. This example shows how to perform the following tasks:

Determine the URL to use to connect to the Data Center server.

Instantiate a SOAP service stub using Apache Axis.

Call a display method to display the version of the software installed on the Data Center server.

```
import com.ironmountain.sswsapi.SSWSAPIServiceLocator;  
import com.ironmountain.sswsapi.SSWSAPIServiceSoap_PortType;  
import java.net.URL;
```

```
import javax.xml.rpc.ServiceException;

/**
 * Connect to SSWSAPI service and report the service version.
 *
 * We presume Apache Axis has been used to generate a stub for the
 * sswsapi.wsdl file supplied with the Data Center. In this case,
 * the package of the generated files is com.ironmountain.sswsapi.
 *
 * This example calls the getAppVersion() call which returns the
 * API service version string.
 */

public final class PingSSWSAPI {

    /**
     * Get the SOAP Service stub from the given URL.
     *
     * Notice that we need to maintain the session and the combination of Apache
     * Axis with a Web Service hosted by IIS as an ATL Web Service, requires that
     * the "sendMultiRefs" flag be sent as false.
     *
     * @param sswsapiUrl URL to connect to
     * @return Implementation of SOAP Service stub
     * @throws ServiceException
     */
    private static SSWSAPIServiceSoap_PortType getSoapService(final URL sswsapiUrl) throws
ServiceException {
        SSWSAPIServiceLocator locator = new SSWSAPIServiceLocator();
        locator.setMaintainSession(true);
        locator.getEngine().setOption("sendMultiRefs", Boolean.FALSE);
        final SSWSAPIServiceSoap_PortType soapService = locator.getSSWSAPIServiceSoap(sswsapiUrl);
        return soapService;
    }

    /**
     * Talk to the SSWS SOAP Service and print out the version.
     *
     * @param args URL to connect to
     */
    public static void main(final String[] args) {
        try {

// 1. Determine the URL to connect to. By default connect to the localhost
// service implementation.

            final URL sswsapiUrl;
            if (args.length == 0) {
                sswsapiUrl = new URL
("http://localhost:80/sswsapi/sswsapi.dll?Handler=Default");
            } else if (args.length == 1) {
                sswsapiUrl = new URL(args[0]);
            } else {
```

```
        sswsapiUrl = null;
        System.out.println("Usage: PingSSWSAPI [url]");
        System.exit(2);
    }

    // 2. Instantiate a SOAP service stub using Apache Axis.

    System.out.println("Connecting to SSWSAPI: " + sswsapiUrl);
    final SSWSAPIServiceSoap_PortType soapService =
getSoapService(sswsapiUrl);

    // 3. Call a method on the remote service. In this simple case we show
    // the current version string for the service.

    System.out.println("SSWSAPI Service Version " +
soapService.getAppVersion());
    System.exit(0);
} catch (Exception e) {
    e.printStackTrace(System.out);
    System.exit(1);
}
}
}
```

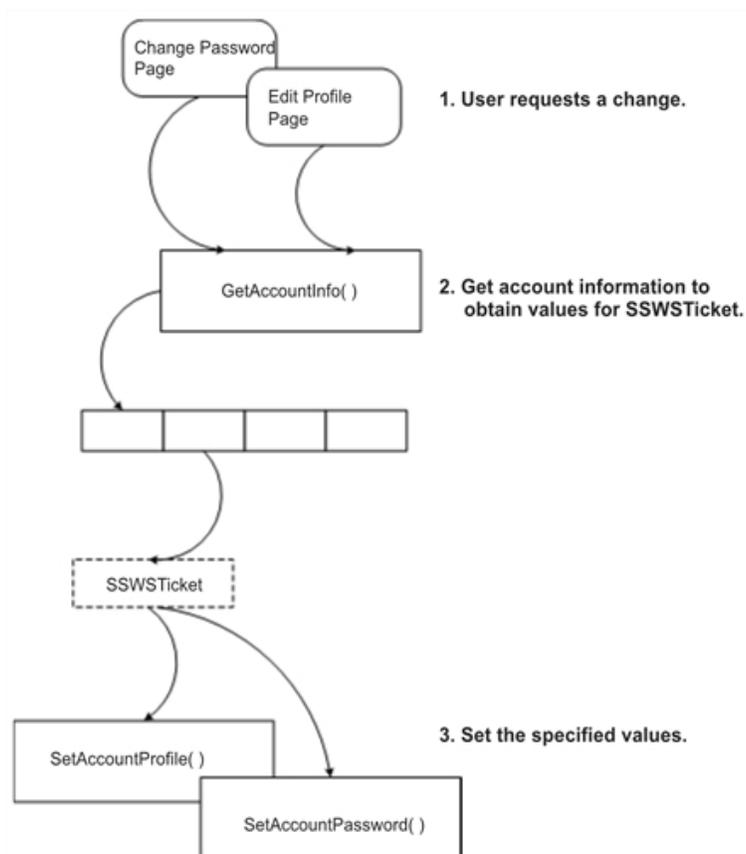
Change setting operations

Description

You can use the Account Management Web Service API to change the values stored in a user's profile and to reset the user's password.

Sequence

The following figure shows an example of how to use Account Management Web Service methods to change passwords and profile settings:



Methods

For information about the methods that you can use to change the settings that are stored in your application, see [Management operations, on page 107](#).

Code example

The following example shows how to use `setAccountProfile` and `setAccountPassword` to change the values in a user's profile:

```
SSWSAccountInfo accountInfo = service.getAccountInfo(ticket);  
SSWSProfileInfo profileInfo = accountInfo.getProfileInfo();  
profileInfo.setDepartment("Finance");  
profileInfo.setCostCenter("Finance-Cost-Center");  
profileInfo.getContactInfo().setTelephone("555-5555");  
service.setAccountProfile(ticket, profileInfo);  
service.setAccountPassword(ticket, "paSSw0rd");
```

Chapter 8: Retrieval

This chapter contains recommendations for the retrieval operations.

- [Retrieve operation, below](#)

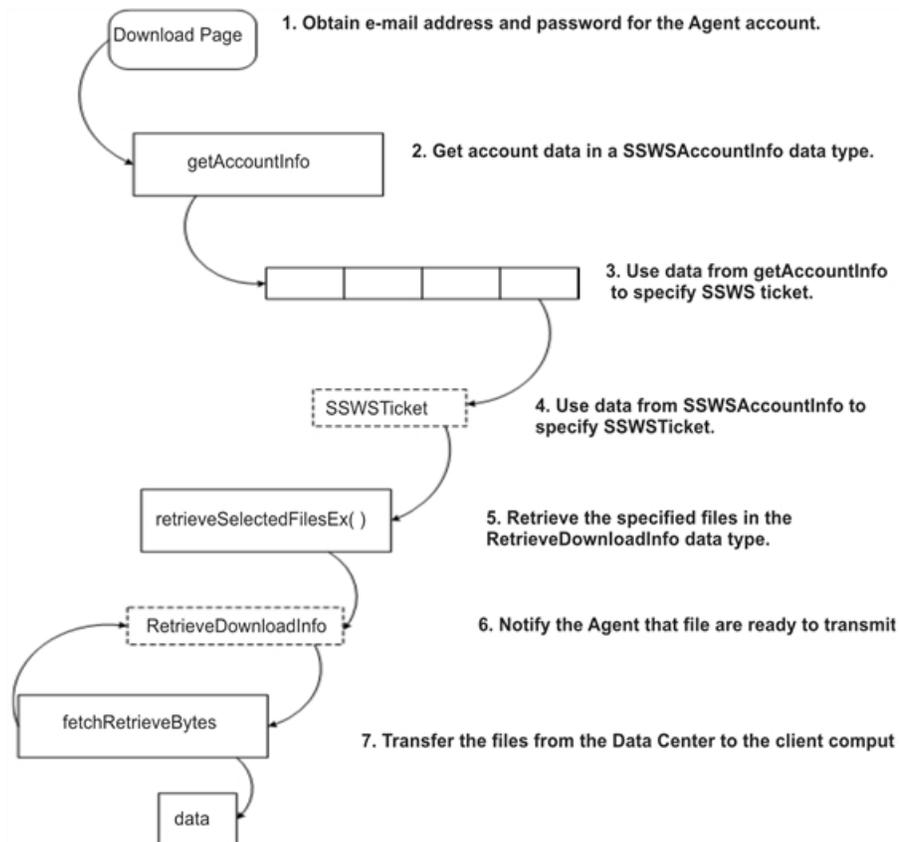
Retrieve operation

Description

Retrieval allows users to retrieve files that the Agent has backed up to the Data Center.

Sequence

The following figure is example of the sequence to retrieve backed up files from the Data Center.



Methods

For information about methods that you can use to retrieve backed up files from the Data Center, see [Retrieve operations, on page 120](#).

Code example

This example uses the `retrieveSelectedFileEx` and the `fetchRetrieveBytes` methods to retrieve backed up files from the Data Center.

```
RetrieveDownloadInfo downloadInfo = (RetrieveDownloadInfo)
service.retrieveSelectedFilesEx(ticket,
fileRevisionsToRetrieve,
new
org.apache.axis.types.UnsignedInt(downloadType),
getLocale());

// setup the file where the downloaded data will be written

BufferedOutputStream f = new BufferedOutputStream(
new FileOutputStream(destinationFileName));
long size = downloadInfo.getSize().longValue();
int start = 0;
final int count = 1024 * 1024;

// read the data in chunks

while (start < size) {
byte[] buffer = service.fetchRetrieveBytes(ticket,downloadInfo, start, count);
f.write(buffer);
start += count;
}f.flush();
f.close();
```

Data types and errors

This chapter describes the data types that the Account Management Web Services API uses.

- [Overview](#), on the next page
- [AccountProfileField](#), on the next page
- [AccountWebSiteSettings](#), on page 61
- [AccountWebSiteSettings2](#), on page 63
- [AccountWebsiteSettings3](#), on page 66
- [AgentDownloadInfo](#), on page 69
- [RegistrationInfo](#), on page 69
- [RetrieveDownloadInfo](#), on page 70
- [RetrieveFileInfo](#), on page 70
- [RetrieveInfo](#), on page 71
- [RetrieveInfo2](#), on page 71
- [SSWSAccountHistory](#), on page 72
- [SSWSAccountInfo](#), on page 73
- [SSWSAccountSummaryInfo](#), on page 74
- [SSWSAddressInfo](#), on page 75
- [SSWSBackupDateInfo](#), on page 75
- [SSWSBackupDateInfoWithoutSizes](#), on page 76
- [SSWSBrandingInfo](#), on page 77
- [SSWSBrandingInfo2](#), on page 77
- [SSWSContactInfo](#), on page 78
- [SSWSFileRevision](#), on page 79
- [SSWSFileRevision2](#), on page 79
- [SSWSFileRevision3](#), on page 80
- [SSWSMediaType](#), on page 81
- [SSWSNameInfo](#), on page 82
- [SSWSProfileInfo](#), on page 82
- [SSWSTicket](#), on page 83
- [Returned errors](#), on page 84

Overview

About the data types

The types element in the Web Services Description Language (WSDL) file encloses data type definitions that the Account Management Web Services API methods use to exchange messages.

All of the data types in the Account Management Web Services API are complexTypes.

About error codes and events

For a description of the errors that the methods in the Account Management Web Services API can return, see [Returned errors, on page 84](#).

Additionally, you can view errors in the system event log on the server where you installed the standard Account Management Website. You must install the standard Website to be able to access the Account Management Web Services API.

AccountProfileField

Description

This complexType contains the attributes of the fields in the profile. This data type specifies the length, default value, and attributes (whether the field is visible, editable, or required) for each value stored in the profile. These values determine how fields are displayed in an Account Management application.

Elements

The following table lists the elements in this data type:

Element	Description	Type
fieldName	The name of the field in the profile.	wsdl_string
defaultValue	The default value to display for the field.	wsdl_string
fieldLength	The length of the field.	wsdl_int
visible	Whether the field is visible to users.	wsdl_bool
editable	Whether users can edit the default values.	wsdl_bool
required	Whether users are required to supply a value for this field.	wsdl_bool

AccountWebSiteSettings

Description

This data type contains the settings that apply to the Account Management Website Profile page for the Agent configuration.

For more recent versions of this data type, see [AccountWebSiteSettings2, on page 63](#) and [AccountWebsiteSettings3, on page 66](#).

Elements

The following table lists the elements in this data type:

Element	Description	Type
firstName	The first name of the user associated with the account.	AccountProfileField, on the previous page
middleName	The middle name of the user associated with the account.	AccountProfileField, on the previous page
lastName	The last name of the user associated with the account.	AccountProfileField, on the previous page
email	The e-mail address associated with the account. Typically, users need an e-mail address to sign in to the Account Management Website	AccountProfileField, on the previous page
country	The country portion of the address associated with the account.	AccountProfileField, on the previous page
addressLineOne	The first line of the street address associated with the account.	AccountProfileField, on the previous page
addressLineTwo	The second line of the street address associated with the account.	AccountProfileField, on the previous page
addressLineThree	The third line of the street address associated with the account.	AccountProfileField, on the previous page
city	The city associated with the user's address.	AccountProfileField, on the previous page
state	The state associated with the	AccountProfileField,

Element	Description	Type
	user's address.	on the previous page
postalCode	The ZIP code or postal code associated with the user's address.	AccountProfileField, on page 60
company	The name of the company associated with the user.	AccountProfileField, on page 60
department	The name of the department associated with the user.	AccountProfileField, on page 60
location	The location of the company associated with the user.	AccountProfileField, on page 60
mailStop	The mail stop address associated with the user.	AccountProfileField, on page 60
costCenter	The cost center identifier associated with the user.	AccountProfileField, on page 60
employeeId	The employee identifier associated with the user.	AccountProfileField, on page 60
phoneNumber	The telephone number associated with the user.	AccountProfileField, on page 60
extension	The telephone extension number associated with the user.	AccountProfileField, on page 60
customFieldLabel	The name of a custom field.	AccountProfileField, on page 60
customField	The value for the custom field.	AccountProfileField, on page 60
supportInfo	The text for the Support contact information.	wsdl_string
permitAsianCharacters	Whether to display Asian characters when users enter them in the profile.	wsdl_bool
showEula	Whether an End User License Agreement is displayed when users register their accounts.	wsdl_bool
promptForAccountReservationCode	Whether the system prompts users to enter an account reservation code when registering their accounts.	wsdl_bool
allowUserToOrderMedia	Whether users can order backed-	wsdl_bool

Element	Description	Type
	up files on media, such as CDs or DVDs.	
allowRetrieveUsingMyRoam	Whether the account includes permission to use MyRoam in the Account Management Website to retrieve files using a Web browser.	wsdl_bool
allowManageAccountLink	Whether there is a link in the Agent Tools menu to the Account Management application.	wsdl_bool
healAllowed	<p>DEPRECATED: The Heal feature has been deprecated.</p> <p>To ensure API methods that use this data type function correctly, set this element to False.</p>	wsdl_bool

AccountWebSiteSettings2

Description

This data type contains the settings that apply to the Account Management Website Profile page for the Agent configuration.

This data type replaces [AccountWebSiteSettings, on page 61](#). For a more recent version of this data type, see [AccountWebsiteSettings3, on page 66](#).

Elements

The following table lists the elements in this data type:

Element	Description	Type
firstName	The first name of the user associated with the account.	AccountProfileField, on page 60
middleName	The middle name of the user associated with the account.	AccountProfileField, on page 60
lastName	The last name of the user associated with the account.	AccountProfileField, on page 60

Element	Description	Type
email	The e-mail address associated with the account. Typically, users need an e-mail address to sign in to the Account Management Website	AccountProfileField, on page 60
country	The country portion of the address associated with the account.	AccountProfileField, on page 60
addressLineOne	The first line of the street address associated with the account.	AccountProfileField, on page 60
addressLineTwo	The second line of the street address associated with the account.	AccountProfileField, on page 60
addressLineThree	The third line of the street address associated with the account.	AccountProfileField, on page 60
city	The city associated with the user's address.	AccountProfileField, on page 60
state	The state associated with the user's address.	AccountProfileField, on page 60
postalCode	The ZIP code or postal code associated with the user's address.	AccountProfileField, on page 60
company	The name of the company associated with the user.	AccountProfileField, on page 60
department	The name of the department associated with the user.	AccountProfileField, on page 60
location	The location of the company associated with the user.	AccountProfileField, on page 60
mailStop	The mail stop address associated with the user.	AccountProfileField, on page 60
costCenter	The cost center identifier associated with the user.	AccountProfileField, on page 60
employeeId	The employee identifier associated with the user.	AccountProfileField, on page 60
phoneNumber	The telephone number associated with the user.	AccountProfileField, on page 60
extension	The telephone extension number associated with the user.	AccountProfileField, on page 60

Element	Description	Type
customFieldLabel	The name of a custom field.	AccountProfileField, on page 60
customField	The value for the custom filed.	AccountProfileField, on page 60
supportInfo	The text for the Support contact information.	wsdl_string
permitAsianCharacters	Whether Asian characters are displayed when users enter them in the profile.	wsdl_bool
showEula	Whether an End User License Agreement is displayed when users register their accounts.	wsdl_bool
promptForAccountReservationCode	Whether the system prompts users to enter an account reservation code when registering their accounts.	wsdl_bool
allowUserToOrderMedia	Whether users can order backed-up files on media, such as CDs or DVDs.	wsdl_bool
allowRetrieveUsingMyRoam	Whether the account includes permission to use MyRoam in the Account Management Website to retrieve files using a Web browser.	wsdl_bool
allowManageAccountLink	Whether there is a link in the Agent Tools menu to the Account Management application.	wsdl_bool
healAllowed	<p>DEPRECATED: The Heal feature has been deprecated.</p> <p>To ensure API methods that use this data type function correctly, set this element to <code>False</code>.</p>	wsdl_bool
productCode	The types of products in the configuration. This element has the following values: <ul style="list-style-type: none"> • 0. PC account • 2. Mac account 	wsdl_int

Element	Description	Type
isLegacy	Whether the configuration contains legacy accounts. Legacy accounts are versions earlier than 8.x.	wsdl_bool
agentType	The types of Agents in the configuration. This element has the following values: <ul style="list-style-type: none">• -1. Invalid• 0. PC or Mac	wsdl_int

AccountWebsiteSettings3

Description

This data type contains the settings that apply to the Account Management Website Profile page for the Agent configuration.

This data type replaces [AccountWebSiteSettings2, on page 63](#).

Elements

The following table lists the elements in this data type:

Element	Description	Type
firstName	The first name of the user associated with the account.	AccountProfileField, on page 60
middleName	The middle name of the user associated with the account.	AccountProfileField, on page 60
lastName	The last name of the user associated with the account.	AccountProfileField, on page 60
email	The e-mail address associated with the account. Typically, users need an e-mail address to sign in to the Account Management Website	AccountProfileField, on page 60
country	The country portion of the address associated with the account.	AccountProfileField, on page 60
addressLineOne	The first line of the street address	AccountProfileField,

Element	Description	Type
	associated with the account.	on page 60
addressLineTwo	The second line of the street address associated with the account.	AccountProfileField, on page 60
addressLineThree	The third line of the street address associated with the account.	AccountProfileField, on page 60
city	The city associated with the user's address.	AccountProfileField, on page 60
state	The state associated with the user's address.	AccountProfileField, on page 60
postalCode	The ZIP code or postal code associated with the user's address.	AccountProfileField, on page 60
company	The name of the company associated with the user.	AccountProfileField, on page 60
department	The name of the department associated with the user.	AccountProfileField, on page 60
location	The location of the company associated with the user.	AccountProfileField, on page 60
mailStop	The mail stop address associated with the user.	AccountProfileField, on page 60
costCenter	The cost center identifier associated with the user.	AccountProfileField, on page 60
employeeId	The employee identifier associated with the user.	AccountProfileField, on page 60
phoneNumber	The telephone number associated with the user.	AccountProfileField, on page 60
extension	The telephone extension number associated with the user.	AccountProfileField, on page 60
customFieldLabel	The name of a custom field.	AccountProfileField, on page 60
customField	The value for the custom field.	AccountProfileField, on page 60
supportInfo	The text for the Support contact information.	wsdl_string
permitAsianCharacters	Whether to display Asian	wsdl_bool

Element	Description	Type
	characters when users enter them in the profile.	
showEula	Displays an End User License Agreement when users register their accounts.	wsdl_bool
promptForAccountReservationCode	Prompts users to enter an account reservation code when registering their accounts.	wsdl_bool
allowUserToOrderMedia	Allows users to order backed-up files on media, such as CDs or DVDs.	wsdl_bool
allowRetrieveUsingMyRoam	Allows users to use the MyRoam in the Account Management Website to retrieve files using a Web browser.	wsdl_bool
allowManageAccountLink	Displays a link to your Account Management application in the Agent Tools menu.	wsdl_bool
healAllowed	<p>DEPRECATED: The Heal feature has been deprecated.</p> <p>To ensure API methods that use this data type function correctly, set this element to <code>False</code>.</p>	wsdl_bool
productCode	<p>The types of products in the configuration. This element has the following values:</p> <ul style="list-style-type: none"> • 0. PC account • 2. Mac account 	wsdl_int
isLegacy	Indicates whether the configuration includes legacy Agents. Legacy Agents are version 7.x and earlier.	wsdl_bool
agentType	<p>Uses the following values to indicate the types of Agents in the configuration:</p> <ul style="list-style-type: none"> • -1. Invalid 	wsdl_int

Element	Description	Type
	<ul style="list-style-type: none">• 0. PC or Mac	
prohibitMSIDownload	Indicates to the configuration grants users permission to allowed to download the AgentSetup.msi file. If the configuration does not grant users permission to download the AgentSetup.msi file, the methods for downloading the MSI file fail.	wSDL_bool

AgentDownloadInfo

Description

This complexType includes information about the size of the Agent Setup file that the Data Center generated for a registered user.

Elements

The following table lists the elements in this data type:

Element	Description	Type
size	The size of the Agent Setup file.	wSDL_ulong
code	Unique identifier for the Agent Setup file created for download.	wSDL_int

RegistrationInfo

Description

This complexType contains information about the community and configuration to which a user registers.

Elements

The following table lists the elements in this data type:

Element	Description	Type
community_id	The community to which the user registers	wSDL_int
configuration_id	The configuration to which the user registers	wSDL_int

RetrieveDownloadInfo

Description

This complexType contains information about files specified for retrieval.

Elements

The following table lists the elements in this data type:

Element	Description	Type
size	The size of the retrieved data	wSDL_ulong
code	The unique identifier of the retrieval operation	wSDL_int
numfileserror	Whether the retrieval was successful	wSDL_int

RetrieveFileInfo

Description

This complexType contains information about a retrieved file.

Elements

The following table lists the elements in this data type:

Element	Description	Type
Size	The size, in bytes, of the retrieved file	wSDL_ulong
Code	The unique identifier of the retrieval operation	wSDL_int
FileName	The name of the retrieved file	wSDL_string

RetrieveInfo

Description

This complexType contains information about the size, in bytes, of retrieved data.

Elements

The following table lists the elements in this data type:

Element	Description	Type
size	The size of the data retrieved	wSDL_ulong
numfiles	The number of files retrieved	wSDL_int

RetrieveInfo2

Description

This complexType contains information about the size, in bytes, of retrieved data.

Elements

The following table lists the elements in this data type:

Element	Description	Type
size	The size of retrieved data	wSDL_ulong
numFiles	The number of files retrieved	wSDL_int
numBundles	The number of bundles retrieved	wSDL_int
numFilesInBundles	The number of files contained in the bundles retrieved	wSDL_int

SSWSAccountHistory

Description

This complexType contains information retrieved from the Agent History.

Elements

The following table lists the elements in this data type:

Element	Description	Type
ActivationTimeInMSec	The time when the account became active.	wsdl_utc_time_in_msec
StorageLimitInBytes	The storage limit, in bytes, for the user's backup account.	wsdl_long
LastBackupTimeInMSec	The time of the last backup.	wsdl_utc_time_in_msec
BackupSetSizeInBytes	The number of bytes saved in the last backup.	wsdl_long
TotalBackupsInMonth	The number of backups in the last 30 days.	wsdl_int
NumFilesProtected	The total number of files in the last backup.	wsdl_int
MyRoamRetrieveTimeInMSec	The date of the last MyRoam retrieval. If MyRoam has never been used to retrieve files for the account, the value is 0.	wsdl_utc_time_in_msec
NumMyRoamFiles	The number of files fetched by the last MyRoam retrieval.	wsdl_int
MyRoamRestoreRetrieveInBytes	The size, in bytes, of the last MyRoam retrieval.	wsdl_long
OrderMediaTimeInMSec	The time of the last media order.	wsdl_utc_time_in_msec
OrderMediaSizeInBytes	The size, in bytes, of the last media order.	wsdl_long
OrderMediaQuantity	Number of CDs or DVDs required for the last media order.	wsdl_int
OrderMediaType	An integer enumerator representing	wsdl_int

Element	Description	Type
	the media type: 0. CD 1. DVD	

SSWSAccountInfo

Description

This complexType contains information about the account, where the account is registered, the Agent software version, whether the community is mapped to an enterprise directory, the profile information, and the account status.

You can use this data type to retrieve the information needed to construct the [SSWSAddressInfo](#), on [page 75](#) data type.

Elements

The following table lists the elements in this data type:

Element	Description	Type
AccountNumber	The account number.	wsdl_string
CommunityId	The numeric identifier associated with the community where the account will register. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_int
ConfigurationId	The numeric identifier of the Agent configuration being used. If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	wsdl_int
ProductCode	The types of products in the configuration. This element has the following values:	wsdl_int

Element	Description	Type
	<ul style="list-style-type: none"> 0. PC account 2. Mac account 	
Version8OrHigher	Whether the Agent is version 8.0 or higher.	wsdl_bool
LDAP	Whether the account is registered in a community that is mapped to an enterprise directory.	wsdl_bool
FedAuth	Whether the account is registered in a community that is configured for single sign-on (SSO).	wsdl_bool
ComputerName	The name of the computer associated with the account.	wsdl_string
ProfileInfo	The information stored in the user's profile.	SSWSProfileInfo, on page 82
bsAccountStatus	The current status of the account (canceled, on hold, or active).	wsdl_string
FedAuthUserID	The SSO user ID.	wsdl_string

SSWSAccountSummaryInfo

Description

This complexType contains information about the time of the last backup and the time when the account registered to a community.

Elements

The following table lists the elements in this data type:

Element	Description	Type
LastBackupTime	The time of the last backup.	wsdl_utc_time_in_msec
AccountRegistration	The time when the account registered and became active.	wsdl_utc_time_in_msec

SSWSAddressInfo

Description

This complexType contains the address information stored in the user's profile. Typically, a user supplies this information during registration.

This data type is contained in the [SSWSContactInfo, on page 78](#) data type. The SSWSContactInfo data type is contained in the [SSWSAccountInfo, on page 73](#) data type.

Elements

The following table lists the elements in this data type:

Element	Description	Type
Address1	The first line in the address.	wSDL_string
Address2	The second line in the address.	wSDL_string
Address3	The third line in the address.	wSDL_string
City	The name of the city.	wSDL_string
Zip	The postal or ZIP code.	wSDL_string
Country	The name of the country.	wSDL_string

SSWSBackupDateInfo

Description

This complexType contains information about the date, status, and size of a specific backup.

Elements

This data type contains the following elements:

Element	Description	Type
tDate	The date of the backup. Enter the value 0 to request all versions of backups.	wSDL_utc_time_in_msec

Element	Description	Type
iStatus	A status enumeration for the backup date: <ul style="list-style-type: none"> • 0. EVENTSTATUS_CANCELLED • 1. EVENTSTATUS_COMPLETED • 2. EVENTSTATUS_COMPLETEDWITH WARINGS • 3. EVENTSTATUS_COMPLETEDWITHERRORS • 4. EVENTSTATUS_INCOMPLETE 	wsdl_int
bHealable	DEPRECATED: The Heal feature has been deprecated. To ensure API methods that use this data type function correctly, set this element to <code>False</code> .	wsdl_bool
IMediaSizeInBytes	The size, in bytes, of the specified backup.	wsdl_long

SSWSBackupDateInfoWithoutSizes

Description

This complexType contains information about the date and status of a specific backup.

Elements

This data type contains the following elements:

Element	Description	Type
Date	The date of the backup. Enter the value 0 to request all versions of backups.	wsdl_utc_time_in_msec
Status	A status enumeration for the backup date: <ul style="list-style-type: none"> • 0. EVENTSTATUS_CANCELLED • 1. EVENTSTATUS_COMPLETED • 2. EVENTSTATUS_COMPLETEDWITH WARINGS • 3. EVENTSTATUS_COMPLETEDWITHERRORS 	wsdl_int

Element	Description	Type
	<ul style="list-style-type: none"> 4. EVENTSTATUS_INCOMPLETE 	
bHealable	<p>DEPRECATED: The Heal feature has been deprecated.</p> <p>To ensure API methods that use this data type function correctly, set this element to <code>False</code>.</p>	wsdl_bool

SSWSBrandingInfo

Description

This complexType contains information about the product name and the images used to rebrand Support Center, the Agent, or the Account Management application.

Elements

The following table lists the elements in this data type:

Element	Description	Type
ProductName	The product name to use in place of the default product name.	wsdl_string
blobImage	A binary large object (BLOB) that contains the images to use in place of the default images.	wsdl_base64Binary

SSWSBrandingInfo2

Description

This complexType contains information about the product name and the images used to rebrand Support Center, the Agent, and the Account Management application.

This data type replaces [SSWSBrandingInfo](#), above.

Elements

The following table lists the elements in this data type:

Element	Description	Type
ProductName	The product name to use in place of the default product name.	wSDL_string
blobImage	A binary large object (BLOB) that contains the images to use in place of the default images.	wSDL_base64Binary
PoweredBy	Whether the application displays the powered by Micro Focus logo in the application header	wSDL_bool
IsImageAvailable	Whether a custom image is available for the application header	wSDL_bool

SSWSContactInfo

Description

This complexType includes contact information that is displayed in a page that provides instructions about contacting Support.

The [SSWSAccountInfo, on page 73](#) data type includes this data type.

Elements

The following table lists the elements in this data type:

Element	Description	Type
Telephone	The telephone number to use for contacting Support.	wSDL_string
Extension	The telephone extension number associated with the telephone number.	wSDL_string
Email	The email address for the Support organization.	wSDL_string
Location	The location of the Support organization.	wSDL_string
MailStop	The mail stop associated with the location	wSDL_string

Element	Description	Type
	of the Support organization.	
AddressInfo	The street address for the Support organization.	SSWSAddressInfo, on page 75

SSWSFileRevision

Description

This complexType identifies the version of the backup from which you want to retrieve data.

Elements

The following table lists the elements in this data type:

Element	Description	Type
tdate	The date of the backup set. Enter the value 0 to request all versions of backups.	wSDL_utc_time_in_msec
mdate	The date that the file was modified.	wSDL_utc_time_in_msec
path	The location of the file.	wSDL_string
filename	The name of the file.	wSDL_string
size	The file size in bytes.	wSDL_ulong
availableForRestore	Whether the Agent can retrieve the file. For example, the Agent cannot retrieve PST files.	wSDL_bool

SSWSFileRevision2

Description

This complexType identifies the version of the backup from which you want to retrieve data.

Elements

The following table lists the elements in this data type:

Element	Description	Type
tdate	The date of the backup set. Enter the value 0 to request all versions of backups.	wsdl_utc_time_in_msec
mdate	The date that the file was modified.	wsdl_utc_time_in_msec
path	The location of the file.	wsdl_string
filename	The name of the file.	wsdl_string
size	The file size in bytes.	wsdl_ulong
availableForRestore	Whether the Agent can retrieve the file. For example, the Agent cannot retrieve PST files.	wsdl_bool
hidden	Whether the file is a hidden file.	wsdl_bool
extensionHidden	Whether the file extension is hidden.	wsdl_bool
package	Whether the file is a package folder. This element applies only to Mac.	wsdl_bool
inPackage	Whether the file is contained in a package. This element applies only to Mac.	wsdl_bool

SSWSFileRevision3

Description

This complexType identifies the version of the backup from which you want to retrieve data.

Elements

The following table lists the elements in this data type:

Element	Description	Type
tdate	The date of the backup.	wsdl_utc_time_in_msec

Element	Description	Type
	Enter the value 0 to request all versions of backups.	
mdate	The date that the file was modified.	wsdl_utc_time_in_msec
path	The location of the file.	ALTSOAP_BLOB
filename	The name of the file.	ALTSOAP_BLOB
size	The file size in bytes.	wsdl_ulong
availableForRestore	Whether the Agent can retrieve the file. For example, the Agent cannot retrieve PST files.	wsdl_bool
hidden	Whether the file is a hidden file.	wsdl_bool
extensionHidden	Whether the file extension is hidden.	wsdl_bool
package	Whether the file is a package folder. This element applies only to Mac.	wsdl_bool
inPackage	Whether the file is contained in a package. This element applies only to Mac.	wsdl_bool

SSWSMediaType

Description

This complexType contains information about the type of media request for backed-up files and the number of disks required for the requested files.

Elements

The following table lists the elements in this data type:

Element	Description	Type
MediaId	An integer enumerator representing the media type: <ul style="list-style-type: none"> • 0. CD • 1. DVD 	wsdl_int
NumMediaRequired	The number of disks (CDs or DVDs) required to store the selected backup image.	wsdl_int

SSWSNameInfo

Description

This complexType contains information about the name of a user associated with an account. Typically, this information is included in data structure that uses the [SSWSAccountInfo, on page 73](#) data type.

Elements

This data type contains the following elements:

Element	Description	Type
FirstName	The first name of the user associated with the account.	wsdl_string
MiddleName	The middle name of the user associated with the account.	wsdl_string
LastName	The last name of the user associated with the account.	wsdl_string

SSWSProfileInfo

Description

This complexType contains the editable information stored in a user's profile.

Elements

This data type contains the following elements

Element	Description	Type
CompanyName	The name of a company associated with the user of the account.	wsdl_string
Department	The department name associated with the user.	wsdl_string
CostCenter	A cost center identifier associated	wsdl_string

Element	Description	Type
	with the user.	
EmployeeID	An employee identifier associated with the user.	wsdl_string
ContactInfo	A structure that contains the contact information.	SSWSContaktInfo, on page 78
CustomerFieldValue	The value for a custom field.	wsdl_string
NameInfo	A structure that contains the name associated with the account.	SSWSNameInfo, on the previous page

SSWSTicket

Description

This complexType contains information about a specific type of user, technician, or end-user. In most cases, use this data type for end-user authentication. Technician authentication is typically required to gain access to the standard Account Management Website from Support Center.

Elements

The following table lists the elements in this data type:

Element	Description	Type
TicketType	This enumeration identifies the type of user: <ul style="list-style-type: none"> • 0.User. This value is reserved for use by Micro Focus. • 1. Technician. 	wsdl_int
CommunityID	The numeric identifier associated with a community. Use a value of -1 for the default community ID, otherwise use the actual number for the community. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_int
AccountNumber	The 10-digit account number. Account numbers have a format of: nnnnn-nnnnn	wsdl_string
TechID	A technician ID.	wsdl_

Element	Description	Type
	If you are using the <code>SSWSTicket</code> data type for user authentication, this value is ignored.	string
Password	<p>Either:</p> <ul style="list-style-type: none"> A user or technician password, if the user account or technician ID has native or enterprise directory credentials. OAuth token, if the user account or technician ID has single sign-on credentials. <p>If you are using the <code>SSWSTicket</code> data type for technician authentication, this value is ignored.</p>	wsdl_string
PasswordHash1	<p>A hashed password. This parameter is reserved for use by Micro Focus.</p> <p>If you are using the <code>SSWSTicket</code> data type for user authentication, this value is ignored.</p>	wsdl_int
PasswordHash2	<p>A hashed password. This parameter is reserved for use by Micro Focus.</p> <p>If you are using the <code>SSWSTicket</code> data type for user authentication, this value is ignored.</p>	wsdl_int

Returned errors

Event log errors

In addition to the error codes that the Account Management Web Service methods return, you can also obtain additional information by checking the event log on the server where the Account Management Web Service API is installed.

Error codes 2 through 8

The following table describes the error codes from 2 through 8 that a method can return if it is not successful:

Code	Error	Description
2	NO_CONNECTION	The method could not establish a connection to the Data Center server.
3	INVALID_LOGIN	The credentials used to sign in to another application are invalid.

Code	Error	Description
4	LOCKED	The account is locked after three unsuccessful sign in attempts.
5	FAILURE	A request failed.
6	IN_PROGRESS	A request is in progress.
7	INVALID_ARGUMENTS	An invalid argument was specified in the method. The error returns a string indicating which argument is invalid.
8	NO_DATA_FOUND	A request for data resulted in no data matching the specified criteria being found.

Error codes 9 through 16

The following table describes the error codes from 9 through 16 that a method can return if it is not successful:

Code	Error	Description
9	INVALID_COMMUNITY	The specified community does not exist or was specified incorrectly.
10	INVALID_CONFIGURATION	The specified Agent configuration does not exist in the specified community.
11	REG_TICKET_INVALID	The specified account reservation code does not exist or was specified incorrectly.
12	REG_TICKET_INUSE	The specified account reservation code is currently in use; a new account cannot register using this code.
13	COMMUNITY_ISNOT_LDAP	The specified community is not mapped to an enterprise directory.
14	COMMUNITY_ISNOT_TICKETED	The specified community does not use account reservation codes.
15	LDAPSERVER_UNREACHABLE	The enterprise directory server was not available.
16	CANT_INITIALIZE_LDAP_ACCOUNT	The method could not set the initial settings for an account registered in a community that is mapped to an enterprise direction. It could not create the account object. Typically, the cause of the error is the method could not connect to the database on the Data Center server or it could not read the master encryption key.

Error codes 17 through 24

The following table describes the error codes from 17 through 24 that a method can return if it is not successful:

Code	Error	Description
17	INVALID_LDAP_USER	The specified user name is not a valid enterprise directory user name.
18	NO_LICENSE_TO_REGISTER	The specified community does not have any available for licenses, new accounts cannot register in this community.
19	REGISTRATION_DISABLED	Registration for the specified community is disabled, new accounts cannot register in this community. Accounts using existing account reservation codes CAN register in this community.
20	INVALID_PASSWORD	The specified password does not match the password associated with the account.
21	REG_TICKET_INVALID_LDAP	The community where the account will register requires an account reservation code but the community does not have a ticket that matches the enterprise directory user name.
22	CHANGE_LOGIN_PASSWORD	The sign in was successful, however, the account password has expired. The user must reset the password.
23	ACCOUNT_LOCKED	The account is currently locked because a user entered incorrect account credentials multiple times.
24	ACCOUNT_PRE80	The specified account is a Legacy PC account.

Error codes 25 through 35

The following table describes the error codes from 25 through 35 that a method can return if it is not successful:

Code	Error	Description
25	NO_ORDER_MEDIA_PERMISSION	The technician does not have permission to request backed up files on media in Support Center.
26	NO_ORDER_MEDIA_ALLOWED	The order media feature is not enabled for an account.

Code	Error	Description
27	AMBIGUOUS_LOGIN_RESULT	For accounts registered in a community that is mapped to an enterprise directory, the account is locked or the sign in is invalid. The method is unable to determine the actual issue.
28	INVALID_TICKET_TYPE	One of the values in the SSWSTicket data type is invalid. Often, this error indicates an invalid value for the iTicketType element.
29	CANCELLED_ACCOUNT	The account is canceled.
30	NO_MODIFY_ACCOUNT_AGENTCONFIGURATION_PERMISSION	The operation failed because the option to change the account configuration is disabled for the technician.
31	NO_ACCESS_ACCOUNT_DATA_PERMISSION	The technician does not have permission to access the user's account.
32	NO_CAPACITY	The Data Transfer API / AMWS API is busy performing other functions and cannot service the request at this time.
33	INVALID_PARTNER_ID	The SSO provider ID is invalid.
34	INVALID_FEDAUTH_ACCOUNT	The SSO account is invalid. Unable to retrieve the user ID from the OAuth token.
35	NO_ACCOUNT_FOUND_FOR_FEDAUTHUSER	Unable to find the SSO account for the specified SSO user ID.

Chapter 10: Registration operations

This chapter describes the methods for account registration operations.

- [Overview](#), below
- [GetCommunityType](#), below
- [GetRegistrationInfo](#), on the next page
- [IsLicenseAvailable](#), on page 90
- [IsLDAPCommunity](#), on page 90
- [VerifyLDAPAccountRegistration](#), on page 91
- [VerifyAccountRegistrationCode](#), on page 92
- [CanRegisterToCommunity](#), on page 93
- [RegisterAccount](#), on page 94
- [VerifyAccountReservationCode](#), on page 96
- [VerifyAccountReservationCodeEx](#), on page 97

Overview

The registration operations control the tasks required to register an account when a user accesses your Account Management application. For information about the registration operation, see [Registration and download](#), on page 33.

GetCommunityType

Description

This method determines the authentication mechanism for a given community. This method is unauthenticated because it is part of the registration process.

Parameters

You can specify the following input parameters:

Parameter	Description	Type
CommunityID	Specifies the ID of the community for which you want to determine the authentication mechanism.	wsdl_int

Return Value

This method returns an integer enumeration that indicates the community type:

- **0.** Indicates the community not mapped to an enterprise directory or configured for single sign-on (SSO).
- **1.** Indicates the community is mapped to an enterprise directory.
- **2.** Indicates the community is configured for SSO.

For a list of possible errors, see [Returned errors, on page 84](#).

GetRegistrationInfo

Description

This method calls the registration session from an encrypted registration URL.

Parameters

You can specify the following input parameters:

Parameter	Description	Type
RegistrationURL	Contains encrypted information about the configuration in which the user registers.	wsdl_string

Return Value

This method returns information in the [RegistrationInfo, on page 69](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

IsLicenseAvailable

Description

This method determines the license availability for a given community. This method is an unauthenticated method.

Input Parameters

This method uses the following parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community where the account will register. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wSDL_int

Return Values

This method returns a boolean result that indicates whether the specified community contains an available license:

- A result of true or 1 indicates the community contains an available license.
- A result of false or 0 indicates the community does not contain an available license.

Typically, you use this method during registration of an account.

For a list of possible errors, see [Returned errors, on page 84](#).

IsLDAPCommunity

Description

This method returns information that indicates whether the specified community is mapped to an enterprise directory. If a community is mapped to an enterprise directory, users must use their

enterprise directory user names to sign in to your Account Management application and access their account information.

This method is an unauthenticated method.

Parameters

This method uses the following parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community where the account will register If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_ int

Return Values

This method returns a boolean value that indicates whether the specified community is mapped to an enterprise directory:

- A result of true or 1 indicates the community is mapped to an enterprise directory.
- A result of false or 0 indicates the community is not mapped to an enterprise directory.

For a list of possible errors, see [Returned errors, on page 84](#).

VerifyLDAPAccountRegistration

Description

This method validates that the supplied user name is a member of the enterprise directory community. If the community is using account reservation, the enterprise directory user name is also compared to the reserved account IDs in the system.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
	The numeric identifier associated with the community where the account	wsdl_

Parameter	Description	Type
	will register. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	int
ConfigurationId	The numeric identifier associated with the configuration the account will use. If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	wsdl_ int
Username	LDAP user name to verify.	wsdl_ string
Password	LDAP password to verify.	wsdl_ string

Return Value

This method returns a boolean result indicating whether the specified users is a valid LDAP user name:

- A result of true or 1 indicates the user name is a valid LDAP user name.
- A result of false or 0 indicates the user name is not a valid LDAP user name.

Typically, you use this method during registration of an account.

For a list of possible errors, see [Returned errors, on page 84](#).

VerifyAccountRegistrationCode

Description

This method compares the supplied account reservation code to the reserved account codes in the system. The case for the account reservation code is ignored.

Parameters

You can specify the following input parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community where the account will register. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_int
ConfigurationID	The numeric identifier associated with the configuration the account will use. If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	wsdl_int
Ticket	The account reservation code to verify.	wsdl_string

Return Value

This method returns a boolean result indicating whether the specified account reservation code is valid:

- A result of true or 1 indicates the account reservation code is valid.
- A result of false or 0 indicates the account reservation code is not valid.

Typically, you use this method during registration of an account.

For a list of possible errors, see [Returned errors, on page 84](#).

CanRegisterToCommunity

Description

This method determines if registration to a specified community is possible. The ability to use the community for registration is based on licenses availability and, if the community is not using account reservation codes, the community's registration status.

This method is an unauthenticated method.

Parameters

Specify the following input parameters:

Parameter	Description	Type
CommunityID	<p>The numeric identifier associated with the community where the account will register.</p> <p>If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.</p>	wsdl_int
ConfigurationID	<p>The numeric identifier associated with the configuration the account will use.</p> <p>If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.</p>	wsdl_int

Return Value

This method returns a boolean result indicating whether account can register in the specified community:

- A result of true or 1 indicates that accounts can register in the specified community.
- A result of false or 0 indicates that accounts cannot register in the specified community.

Typically, you use this method during registration of an account.

For a list of possible errors, see [Returned errors, on page 84](#).

RegisterAccount

Description

This method registers a new account in a a specified community using a specified configuration. This method does the following:

- Verifies that an unused license is available in the community.
- If the community is mapped to an enterprise directory, verifies the user credentials.
- If the community is configured for single sign-on (SSO), verifies that the SSO Service Provider that services Connected Backup can use the OAuth token to retrieve the user ID from the identity provider. If successful, it uses the SSO user ID to register the account.
- If the community uses account reservation codes, verifies that the supplied code is not in use.

- Registers an account and sets the account credentials if the account is not registering in a community mapped to an enterprise directory.

Parameters

You can specify the following input parameters:

Parameter	Description	Type
CommunityID	<p>The numeric identifier associated with the community where the account will register.</p> <p>If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.</p>	wsdl_integer
ConfigurationID	<p>The numeric identifier associated with the configuration the account will use.</p> <p>If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.</p>	wsdl_integer
Username	<p>The user name for account reservation registration.</p> <p>If registering an account in a community that is mapped to an enterprise directory, this value is the enterprise directory user name.</p> <p>If registering an account in a community configured for SSO, the value of this parameter is ignored.</p>	wsdl_string
Password	<p>One of the following:</p> <ul style="list-style-type: none">• For accounts with native or enterprise directory credentials, the password associated with the account.• For accounts with SSO credentials, the OAuth token.	wsdl_string
AccountInfo	All of the account information in the <code>SSWSPProfileInfo</code> data type.	SSWSPProfileInfo, on page 82

Return Value

This method returns an integer that represents the account number assigned to the new account.

For a list of possible errors, see [Returned errors, on page 84](#).

VerifyAccountReservationCode

Description

This method compares a specified account reservation code to the codes for reserved accounts in the Data Center. For a more complete version of this method, see [VerifyAccountReservationCodeEx, on the next page](#). For best practice, use the most complete version.

Parameters

You can specify the following input parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community where the account will register. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_ integer
ConfigurationID	The numeric identifier associated with the configuration the account will use. If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	wsdl_ integer
Ticket	The reservation code that you want to verify.	wsdl_ string

Return Value

This method returns a boolean result that indicates whether the specified reservation code is valid:

- A result of true or 1 indicates the account reservation code is valid.
- A result of false or 0 indicates the account reservation code is not valid.

Typically, you use this method during registration of an account.

For a list of possible errors, see [Returned errors, on page 84](#).

VerifyAccountReservationCodeEx

Description

This method compares the specified account reservation code to the codes for reserved accounts in the Data Center. If a reservation code exists, this method retrieves information about the account associated with the reservation code.

Parameters

You can specify the following input parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community where the account will register. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_ integer
ConfigurationID	The numeric identifier associated with the configuration the account will use. If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	wsdl_ integer
Ticket	The reservation code that you want to verify.	wsdl_ string

Return Value

If there is a reservation code in the Data Center for an account, this method returns an array of [SSWSAccountInfo](#), on page 73 data types.

Typically, you use this method during registration of an account.

For a list of possible errors, see [Returned errors](#), on page 84.

Chapter 11: Download operations

This chapter describes the available methods for download operations.

- [Overview, below](#)
- [GenerateAgentSetup, below](#)
- [GenerateAgentSetupLocal, below](#)
- [FetchAgentSetupBytes, on page 100](#)

Overview

The download operations control the tasks required to download an Agent Setup file from your Account Management application. You can control the download by downloading a set number of bytes multiple times until you have downloaded all of the bytes.

GenerateAgentSetup

DEPRECATED:

The `GenerateAgentSetup` method has been deprecated.

GenerateAgentSetupLocal

Description

The method requests the Data Center server to generate an Agent Setup file in a location that you specify. To use this method, you must have valid authentication information in the form of an `SSWSTicket` object.

This method uses information from a `SSWSTicket` data type.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the	SSWSTicket, on

Parameter	Description	Type
	account and authenticate the user.	page 83
IsReinstall	Specifies whether this operation is a new installation or a reinstallation for an existing account.	wsdl_bool
destPath	Specifies where to put the AgentSetup.msi file. This value must be a shared network drive on a server that is accessible from both the server that host the AMWS API and the one that hosts your application. NOTE: Do not specify a location on the same server that hosts the AMWS API.	wsdl_string

Return Value

This method returns a result that indicates if the Agent Setup file creation succeeded.

For a list of possible errors, see [Returned errors, on page 84](#).

Code Example

```
/**  
 * Generates agent setup on disk at a specified location.  
 *  
 * @param AuthenticationInfo - [in] Account / User sign in info  
 * @param IsReinstall - [in] Whether or not to make a reinstall MSI or not  
 * @param destPath - [in] Where to put the agent setup MSI  
 * @return S_OK if operation succeeds, E_FAIL otherwise  
 */  
  
HRESULT GenerateAgentSetupLocal(/*in*/ SSWSTicket AuthenticationInfo,  
/*in*/ wsdl_bool IsReinstall,  
/*in*/ wsdl_string destPath);
```

Usage

This section provides a high-level overview of how to use the GenerateAgentSetupLocal method.

1. Create a shared network drive where the GenerateAgentSetupLocal method will place the AgentSetup.msi file that it creates.

The shared drive must be accessible by the following servers:

- The server that will host your custom application that calls the `GenerateAgentSetupLocal` method
- The server that hosts the AMWS API

NOTE:

To ensure that the `GenerateAgentSetupLocal` method works correctly, do not create the shared drive on the server that hosts the AMWS API.

2. Create an application that uses the `GenerateAgentSetupLocal` method.

When you define the method's parameter values, use the shared network drive you just created as the value of the `destPath` parameter. You can specify this value as either:

- a direct path to the shared network drive
For example: `\\myServer\DCGeneratedFiles`
- a mapped network drive, if you mapped a drive on the AMWS API server to it
For example: `G:\DCGeneratedFiles`

3. Run your application, which creates an `AgentSetup.msi` file in the shared folder.
4. Move the resulting `AgentSetup.msi` file from the shared network drive to another location.

NOTE:

The `GenerateAgentSetupLocal` method cannot create a new `AgentSetup.msi` file if one already exists in the location you specify for the `destPath` parameter.

Therefore, if you plan to generate additional `AgentSetup.msi` files in the same location, be sure to move the previously generated file before you generate the next one.

FetchAgentSetupBytes

DEPRECATED:

The `FetchAgentSetupBytes` method has been deprecated.

Chapter 12: Sign-in operations

This chapter describes the available methods for sign in operations.

- [Overview](#), below
- [GetAccounts](#), below
- [GetFedAuthAccountsFromCommunity](#), on the next page
- [GetFedAuthAccountsFromPartnerID](#), on page 103
- [GetIDPURLForCommunity](#), on page 103
- [GetIDPURLForPartnerId](#), on page 104
- [SetLoginInformation](#), on page 105
- [VerifyTechAccountAccess](#), on page 106

Overview

To ensure that users are able to access only their account information, require them to sign in to your application. You can use the account credentials to determine what information to display and which features to enable for specific users.

GetAccounts

Description

This method retrieves account information associated with specified account credentials.

This method is an unauthenticated method.

Parameters

This method uses the following parameters:

Parameter	Description	Type
CommunityID	Either: <ul style="list-style-type: none">• -99, to retrieve all accounts that the user owns, regardless of community.	wSDL_int

	<ul style="list-style-type: none">The numeric identifier for the community where the user's accounts are registered. <p>If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.</p>	
Email	The email address associated with the account.	wsdl_string
Password	The password associated with the account.	wsdl_string

Return Value

This method returns an array of [SSWSAccountInfo, on page 73](#) data types and the number of elements in the array.

For a list of possible errors, see [Returned errors, on page 84](#). Also see [Application sign in, on page 26](#).

GetFedAuthAccountsFromCommunity

Description

This method retrieves a list of accounts for a specific single sign-on user in a given SSO community. This method is an unauthenticated method.

Parameters

This method uses the following parameters:

Parameter	Description	Type
CommunityID	<p>The numeric identifier for the community where the accounts are registered.</p> <p>If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.</p>	wsdl_int
AuthToken	The OAuth token from the SSO Service Provider.	wsdl_string

Return Value

This method returns an array of [SSWSAccountInfo, on page 73](#) data types and the number of elements in the array.

For a list of possible errors, see [Returned errors, on page 84](#). Also see [Application sign in, on page 26](#).

GetFedAuthAccountsFromPartnerID

Description

This method retrieves a list of accounts for a specific single sign-on user that is associated with a given SSO provider ID. This method is an unauthenticated method.

Parameters

This method uses the following parameters:

Parameter	Description	Type
PartnerID	The SSO Provider ID.	wSDL_string
AuthToken	The OAuth token from the SSO Service Provider.	wSDL_string

Return Value

This method returns an array of [SSWSAccountInfo, on page 73](#) data types and the number of elements in the array.

For a list of possible errors, see [Returned errors, on page 84](#). Also see [Application sign in, on page 26](#).

GetIDPURLForCommunity

Description

This method retrieves the URL of the third-party identity provider's SSO Sign In page for a given community. This method is an unauthenticated method.

Parameters

This method uses the following parameters:

Parameter	Description	Type
CommunityId	The numeric identifier of the community configured for single sign-on (SSO).	wsdl_int

Return Value

This method returns a `wsdl_string` that contains the URL of the SSO Sign In page of the identity provider for the given community. For example:

```
https://myserver.com:9031/as/authorization.oauth2?PartnerIdpId=myPartnerID&client_id=CB_Web&redirect_uri=https://sswshostname/fed_auth_validation.jsp&response_type=token
```

Before you use this string as the URL to the SSO Sign In page, replace the substring `https://sswshostname/fed_auth_validation.jsp` with the URL to your Web application that will embed this page.

For a list of possible errors, see [Returned errors, on page 84](#). Also see [Application sign in, on page 26](#).

GetIDPURLForPartnerId

Description

This method retrieves the URL of the third-party identity provider's SSO Sign In page for a given SSO provider ID. This method is an unauthenticated method.

Parameters

This method uses the following parameters:

Parameter	Description	Type
PartnerID	The SSO provider ID.	wsdl_string

Return Value

This method returns a `wSDL_string` that contains the URL of the SSO Sign In page of the identity provider for the given community. For example:

```
https://myserver.com:9031/as/authorization.oauth2?PartnerIdpId=myPartnerID&client_id=CB_Web&redirect_uri=https://sswshostname/fed_auth_validation.jsp&response_type=token
```

Before you use this string as the URL to the SSO Sign In page, replace the substring `https://sswshostname/fed_auth_validation.jsp` with the URL to your Web application that will embed this page.

For a list of possible errors, see [Returned errors, on page 84](#). Also see [Application sign in, on page 26](#).

SetLoginInformation

Description

This method sets sign in information for a specified user.

Parameters

This method uses the following parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
EmailAddress	The email address associated with the account.	<code>wSDL_string</code>
Password	The password associated with the account.	<code>wSDL_string</code>

Return Value

None.

VerifyTechAccountAccess

Description

This method validates the supplied technician credentials and whether the technician has permission to manage a specific account. To access a user's account, the technician must have the **Access Users' Data** permission enabled in Support Center.

This method is an authenticated method. For an example how to use this method, [Application sign in, on page 26](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83

Return Value

This method returns a boolean result that indicates whether the technician account credentials are valid.

For a list of possible errors, see [Returned errors, on page 84](#).

Chapter 13: Management operations

This chapter describes methods to submit a request for backup sets on media and to change user information.

- [Overview, below](#)
- [GetAccountInfo, below](#)
- [GetMediaTypes, on the next page](#)
- [SubmitMediaOrder, on the next page](#)
- [SetConfiguration, on page 109](#)
- [SetAccountPassword, on page 110](#)
- [SetAccountProfile, on page 111](#)

Overview

The management operations allow you to perform the following tasks:

- Submit a request for an image of a specific backup to the Data Bundler™ application.
- Change information typically stored in a user's information. This information includes all of the data that a user enters during registration. The profile operations also allow users to change the passwords that they use when they sign in to your Account Management application.

GetAccountInfo

Description

This method retrieves profile information for a specified account.

This method is an authenticated method. For a code example that uses this method, see [Account management, on page 46](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83

Return Value

This method returns the [SSWSAccountInfo, on page 73](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetMediaTypes

Description

This method retrieves the media types available for the specified backup, CD or DVD.

This method is an authenticated method. For a code example that uses this method, see [Order media operation, on page 46](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
BackupDate	The date of the backup being requested.	wsdl_utc_time_in_msec

Return Value

This method returns an array of [SSWSMediaType, on page 81](#) data types and the number of elements in the array. This data type includes the type of media available for the account (CD or DVD) and the number of disks required for the data backed up at the specified backup date.

For a list of possible errors, see [Returned errors, on page 84](#).

SubmitMediaOrder

Description

This method submits a request for an image of a specific backup to the DataBundler application.

This method is an authenticated method. For a code example that uses this method, see [Order media operation, on page 46](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
MediaType	The type of media you want to order. 0=CD 1=DVD.	wSDL_int
ShippingLabel	The text to use on the shipping label.	wSDL_string
ShippingPriority	The shipping priority.	wSDL_int
MediaPassword	A password associated with the backup image on the media. Users must use this password to retrieve data from the media.	wSDL_string
BackupDate	The date of the requested backup.	wSDL_utc_time_in_msec

Return Value

This method does not return any values. If successful, it submits the order to the DataBundler application.

For a list of possible errors, see [Returned errors, on page 84](#).

SetConfiguration

Description

This method changes the Agent configuration for the specified account. This method is an authenticated method.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
ConfigurationId	Specifies the ID of the new configuration.	wsdl_int

Return Value

This method does not return any values. If successful, it changes the configuration for the specified account.

For a list of possible errors, see [Returned errors, on page 84](#).

SetAccountPassword

Description

This method enables you to change the password associated with a specific account. This method has no effect on passwords stored in enterprise directories. You must change those passwords in the enterprise directory.

This method is an authenticated method.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
NewPassword	The new password for the account.	wsdl_string

Return Value

This method does not return values. If the method is successful, it sets the password to the new value.

For a list of possible errors, see [Returned errors, on page 84](#).

SetAccountProfile

Description

The method supplies information that changes the values in a user's profile. This method has no effect on information stored in an enterprise directory, You must change those values in the enterprise directory.

This method is an authenticated method.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
ProfileData	Specifies the information you want to change in the profile.	SSWSBackupDateInfo, on page 75

Returns

This method does not return values. If the method is successful, it changes the profile data to the new values specified in this method.

For a list of possible errors, see [Returned errors, on page 84](#).

Chapter 14: Display operations

This chapter describes the methods that affect the display of your Account Management application.

- [Overview, below](#)
- [GetAppVersion, below](#)
- [GetAccountSummaryInfo, on the next page](#)
- [GetAccountHistory, on the next page](#)
- [GetAccountWebSiteSettings, on page 114](#)
- [GetAccountWebSiteSettings2, on page 115](#)
- [GetAccountWebSiteSettings3, on page 116](#)
- [GetBrandingInfo, on page 117](#)
- [GetBackupDates, on page 118](#)
- [GetBackupDatesWithoutSizes, on page 118](#)

Overview

The display operations affect the appearance of your Account Management application. The available methods allow you to retrieve and use the different settings configured for a community in Support Center. Depending on how you develop your application pages, you can choose to use or ignore the information you retrieve.

GetAppVersion

Description

This method retrieves the application version to display on an application page.

This method is an unauthenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

Parameters

This method does not use input parameters.

Return Value

This method returns the version of the Account Management Website in a format of x.x.x.x. It retrieves this information from the Registry database on the Data Center server.

For a list of possible errors, see [Returned errors, on page 84](#).

GetAccountSummaryInfo

Description

This method retrieves information that you can use to display information on an account Summary page.

This method is an authenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user	SSWSTicket, on page 83

Return Value

This method returns the information in a [SSWSAccountSummaryInfo, on page 74](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetAccountHistory

Description

This method retrieves the information about the backups associated with the account, when the account became active, the storage limits associated with the account, and orders for backup images on media.

This method is an authenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user	SSWSTicket, on page 83

Return Value

This method returns the information in a [SSWSAccountHistory, on page 72](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetAccountWebSiteSettings

Description

This method returns the Website Settings configured in Support Center for a specified community configuration. The Website Settings determine which fields to display and which features to enable in your Account Management application.

This method is an unauthenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

For a more complete version of this method, see [GetAccountWebSiteSettings3, on page 116](#). For best practice, use the most complete version.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_ int
ConfigurationID	The numeric identifier associated with the configuration.	wsdl_

Parameter	Description	Type
	If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	int

Return Value

This method returns information in the [AccountWebSiteSettings, on page 61](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetAccountWebSiteSettings2

Description

This method returns the Website Settings for a community and configuration. The Website Settings determine which fields to display and which features to enable in your Account Management application.

This method is an unauthenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

For a more complete version of this method, see [GetAccountWebSiteSettings3, on the next page](#). For best practice, use the most complete version.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_ int
ConfigurationID	The numeric identifier associated with the configuration. If you do not know the configuration ID, you can use Support Center to	wsdl_ int

Parameter	Description	Type
	determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	

Return Value

This method returns information in the [AccountWebSiteSettings2, on page 63](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetAccountWebSiteSettings3

Description

This method returns the Website Settings for a community and configuration. The Website Settings determine which fields to display and which features to enable in your Account Management application.

This method is an unauthenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wSDL_int
ConfigurationID	The numeric identifier associated with the configuration. If you do not know the configuration ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the configuration. Support Center displays the ID at the bottom of the Edit tab in the right pane.	wSDL_int

Return Value

This method returns information in the [AccountWebsiteSettings3, on page 66](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetBrandingInfo

Description

This method retrieves the product name and image required to rebrand the Account Management application with a different product name and logo.

This method is an unauthenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
CommunityID	The numeric identifier associated with the community where the account will register. If you do not know the community ID, you can use Support Center to determine it. In the left pane of Support Center, click the node that represents the community. Support Center displays the ID in the Community Status section of the Status tab in the right pane.	wsdl_ int
LastDownload	The time when branding information was downloaded previously.	wsdl_ utc_ time_ in_ msec

Return Value

This method returns information in the [SSWSBrandingInfo, on page 77](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetBackupDates

Description

This method retrieves the backup dates for a specific account. Use this method to display a list of backup dates associated with an account and the status of each backup that occurred at that date.

This method is an authenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83

Return Value

This method returns an array of [SSWSBackupDateInfo, on page 75](#) data types and the number of elements in the array.

For a list of possible errors, see [Returned errors, on page 84](#).

GetBackupDatesWithoutSizes

Description

This method retrieves the backup dates for a specific account. Use this method to display a list of backup dates associated with an account without information about the status of the backups.

This method is an authenticated method. For an example of how to use a display method, see [Display operations, on page 52](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket , on page 83

Return Value

This method returns an array of [SSWSBackupDateInfoWithoutSizes](#), on [page 76](#) data types and the number of elements in the array.

For a list of possible errors, see [Returned errors](#), on [page 84](#)

Chapter 15: Retrieve operations

This chapter describes the methods for retrieve operations.

- [Overview](#), below
- [FetchRetrieveBytes](#), below
- [FetchSingleFile](#), on the next page
- [FindFiles](#), on page 122
- [FindFiles2](#), on page 123
- [GetFileList](#), on page 124
- [GetFileList2](#), on page 125
- [GetFilesCountForFolder](#), on page 126
- [GetFoldersForTdate](#), on page 127
- [GetFoldersForTdate2](#), on page 128
- [GetMyRoamState](#), on page 128
- [GetRetrieveSize](#), on page 129
- [GetRetrieveSize2](#), on page 130
- [IsMyRoamLicensed](#), on page 130
- [RetrieveSelectedFiles](#), on page 131
- [RetrieveSelectedFilesEx](#), on page 132

Overview

The retrieve operations control the tasks required to retrieve backed up files from the Data Center.

FetchRetrieveBytes

Description

This method retrieves a group of files. To use this method, you must have valid authentication information in the form of an `SSWSTicket` object.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
DownloadInfo	Contains the information returned by the download.	RetrieveDownloadInfo, on page 70
ByteStart	Specifies the starting point at which to fetch bytes.	wsdl_int
ByteCount	Specifies the number of bytes to fetch.	wsdl_int

Return Value

This method returns an array of bytes.

For a list of possible errors, see [Returned errors, on page 84](#).

FetchSingleFile

Description

This method retrieves a single file from the Data Center without sending it to a compressed ZIP folder. To use this method, you must have valid authentication information in the form of an SSWSTicket object.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
RetrieveFileInfo	Specifies whether this operation is a new	RetrieveFileInfo,

Parameter	Description	Type
	installation or a reinstallation for an existing account.	on page 70
ByteStart	Specifies the starting point at which to fetch bytes.	wsdl_int
ByteCount	Specifies the number of bytes to fetch.	wsdl_int

Return Value

This method returns an array of bytes.

For a list of possible errors, see [Returned errors, on page 84](#).

FindFiles

Description

This method finds the list of files or folders from a specified path that match a specified pattern for a specified backup date. To use this method, you must have valid authentication information in the form of an SSWSTicket object.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

For a more complete version of this method, see [FindFiles2, on the next page](#). For best practice or to retrieve files from Mac accounts, use the most complete version.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
tdate	Specifies the date of the backup. Enter the value 0 to request all versions of backups.	wsdl_utc_time_in_msec
SearchString	Specifies the pattern that you want the files to match.	wsdl_string

Parameter	Description	Type
FolderPathName	Specifies the path to the folder in which you want to search.	wSDL_string

Return Value

This method returns an array of [SSWSFileRevision](#), on page 79 data types.

For a list of possible errors, see [Returned errors](#), on page 84.

FindFiles2

Description

This method finds the list of files or folders from a specified path that match a specified pattern for a specified backup date. To use this method, you must have valid authentication information in the form of an `SSWSTicket` object.

To retrieve files from Mac accounts, you must use this method instead of [FindFiles](#), on the previous page.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket , on page 83
tdate	Specifies the date of the backup. Enter the value 0 to request all versions of backups.	wSDL_utc_time_in_msec
SearchString	Specifies the pattern that you want the files to match.	wSDL_string
FolderPathName	Specifies the path to the folder in which you want to search.	wSDL_string

Return Value

This method returns the [SSWSFileRevision3](#), on page 80 data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetFileList

Description

This method returns a list of files for a specified folder and a specified backup date.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

For a more complete version of this method, see [GetFileList2, on the next page](#). For best practice or to retrieve files for a Mac account, use the most complete version.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Specifies the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
tdate	Specifies the date of the backup. Enter the value 0 to request all versions of backups.	wSDL_utc_time_in_msec
FolderPathName	Specifies the name and location of the folder from which you want to get files.	wSDL_string
Start	Specifies the starting point of a range of files to include in the list. For example, a Start of 1 and a Range of 5 instructs the method to return the list in batches of five files. A Start of 0 with a Range of 0 returns the entire list.	wSDL_int
Range	Specifies the ending point of a range of files to include in the list. For example, a Start of 1 and a Range of 5 instructs the method to return the list in batches of five files. A Start of 0 with a Range of 0 returns the entire list.	wSDL_int

Return Value

This method returns an array of [SSWSFileRevision](#), on page 79 data types.

GetFileList2

Description

This method gets a list of the files in a specified folder for a specified backup date. To use this method, you must have valid authentication information in the form of an SSWSTicket object.

To retrieve files from Mac accounts, you must use this method instead of [GetFileList](#), on the previous page.

This method is an authenticated method. For sample code that uses this method, see [Registration and download](#), on page 33.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket , on page 83
tdate	Specifies the date of the backup. Enter the value 0 to request all versions of backups.	wsdl_utc_time_in_msec
FolderPathName	Specifies the path to the folder in which you want to search.	wsdl_int
Start	Specifies the starting point of a range of files to include in the list. For example, a Start of 1 and a Range of 5 instructs the method to return the list in batches of five files. A Start of 0 with a Range of 0 returns the entire list.	wsdl_int
Range	Specifies the ending point of a range of files to include in the list. For example, a Start of 1 and a Range of 5 instructs the	wsdl_int

Parameter	Description	Type
	method to return the list in batches of five files. A Start of 0 with a Range of 0 returns the entire list.	

Return Value

This method returns the [SSWSFileRevision3, on page 80](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetFilesCountForFolder

Description

The method returns the total number of files and subfolders for the specified folder for the specified backup date. To use this method, you must have valid authentication information in the form of an SSWSTicket object.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
tdate	Specifies the date of the backup. Enter the value 0 to request all versions of backups.	wSDL_utc_time_in_msec
FolderPathName	Specifies the folder from which you want to get the file count.	wSDL_string

Return Value

This method returns an integer that represents the number of files in the specified folder

For a list of possible errors, see [Returned errors, on page 84](#).

GetFoldersForTdate

Description

This method gets the list of folder for the specified account for the specified backup date. To use this method, you must have valid authentication information in the form of an SSWSTicket object.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

For a more complete version of this method, see [GetFoldersForTdate2, on the next page](#). For best practice or to retrieve files for a Mac account, use the most complete version.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
tdate	Specifies the date of the backup. Enter the value 0 to request all versions of backups.	wsdl_utc_time_in_msec

Return Value

This method returns an array of [SSWSFileRevision, on page 79](#) data types and an integer that represents the number elements in the array.

For a list of possible errors, see [Returned errors, on page 84](#).

GetFoldersForTdate2

Description

This method gets the list of folder for the specified account for the specified backup date. To use this method, you must have valid authentication information in the form of an SSWSTicket object.

To retrieve files from Mac accounts, you must use this method instead of [GetFoldersForTdate](#), on the [previous page](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket , on page 83
tdate	Specifies the date of the backup. Enter the value 0 to request all versions of backups.	wsdl_utc_time_in_msec

Return Value

This method returns an array of [SSWSFileRevision3](#), on page 80 data types and an integer that represents the number of elements in the array.

For a list of possible errors, see [Returned errors](#), on page 84.

GetMyRoamState

Description

This method indicates whether the specified user has permissions to use MyRoam.

This method is an authenticated method. For sample code that uses this method, see [Registration and download](#), on page 33.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83

Return Value

This method returns the following boolean results:

- TRUE = MyRoam is allowed
- FALSE = MyRoam is not allowed

For a list of possible errors, see [Returned errors, on page 84](#).

GetRetrieveSize

Description

This method gets the total size and count of files that the user can retrieve with MyRoam.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

For a more complete version of this method, see [GetRetrieveSize2, on the next page](#). For best practice, use the most complete version.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
sizeIn	Specifies the number of files in the file list. This parameter does not pertain if you use Java. It pertains only if you use C++.	wsdl_int
FileList	Specifies the file list to verify.	SSWSFileRevision, on page 79

Return Value

This method returns the [RetrieveInfo, on page 71](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

GetRetrieveSize2

Description

This method gets the total size and count of files that the user can use MyRoam to retrieve.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
sizeIn	Specifies the number of files in the file list. This parameter does not pertain if you use Java. It pertains only if you use C++.	wsdl_int
FileList	Specifies the file list to verify.	SSWSFileRevision2, on page 79

Return Value

This method returns the [RetrieveInfo2, on page 71](#) data type.

For a list of possible errors, see [Returned errors, on page 84](#).

IsMyRoamLicensed

Description

This method verifies whether MyRoam is licensed for a specified user.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83

Return Value

This method returns the following boolean values:

- **TRUE**. Licensed
- **FALSE**. Not licensed

For a list of possible errors, see [Returned errors, on page 84](#).

RetrieveSelectedFiles

Description

This method retrieves a specified set of files and sends the files to a compressed ZIP folder.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

For a more complete version of this method, see [RetrieveSelectedFilesEx, on the next page](#). For best practice, use the most complete version.

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
size	Specifies the number of files in the file list. This parameter does not pertain if you use Java. It pertains only if you use C++.	wsdl_int

Parameter	Description	Type
FileList	Specifies the files to retrieve.	SSWSFileRevision3, on page 80

Return Value

This method returns an array of [RetrieveDownloadInfo, on page 70](#) data types.

For a list of possible errors, see [Returned errors, on page 84](#).

RetrieveSelectedFilesEx

Description

This method retrieves a specified set of files and sends the files to a compressed ZIP folder.

This method is an authenticated method. For sample code that uses this method, see [Registration and download, on page 33](#).

Parameters

This method uses the following input parameters:

Parameter	Description	Type
AuthenticationInfo	Contains the information needed to identify the account and authenticate the user.	SSWSTicket, on page 83
sizeIn	Specifies the number of files in the file list. This parameter does not pertain if you use Java. It pertains only if you use C++.	wsdl_int
FileList	Specifies the files to retrieve.	SSWSFileRevision3, on page 80
flags	Specifies the format in which to deliver the files. You can use the following flags: <ul style="list-style-type: none">• 0 = zip• 1 = self-extracting executable	wsdl_uint
locale	Specifies the language of the installer program. You can specify the standard locale string for any of the languages that Connected Backup supports (for example jp = Japanese). This parameter is ignored if you specify 0 for flags.	wsdl_string

Return Value

This method returns an array of [RetrieveDownloadInfo, on page 70](#) data types.

For a list of possible errors, see [Returned errors, on page 84](#).

Index

A

- Access User's Account Online link 12
- Account Management application
 - custom 10
 - recommended content 12
- Account Management Website
 - overview 9
 - purpose 9
 - standard 9
- AccountProfileField 60
- AccountWebSite Settings 61
- AccountWebSite Settings2 63
- AgentDownloadInfo 69

C

- CanRegisterToCommunity 93
- change setting
 - description 55
 - methods 56
 - sequence 55
- communication flow 10

D

- data types
 - about 60
 - AccountProfileField 60
 - AccountWebSiteSettings 61
 - AccountWebSiteSettings2 63
 - AgentDownloadInfo 69
 - SSWSAccountHistory 72
 - SSWSAccountInfo 73
 - SSWSAccountSummaryInfo 74
 - SSWSAddressInfo 75
 - SSWSBackupDateInfo 75
 - SSWSBrandingInfo 77
 - SSWSContaktInfo 78
 - SSWSMediaType 81
 - SSWSNameInfo 82
 - SSWSProfileInfo 82
 - SSWSTicket 83
- display
 - code example 53
 - description 52
 - GetAccountHistory 113
 - GetAccountSummaryInfo 113

- GetAccountWebSiteSettings 114
- GetAccountWebSiteSettings2 115
- GetAccountWebSiteSettings3 116
- GetBackupDates 118
- GetBackupDatesWithoutSizes 118
- GetBrandingInfo 117
 - methods 53
 - sequence 52
- download
 - code example 39
 - description 38
 - example 45
 - FetchAgentSetupBytes 100
 - GenerateAgentSetup 98
 - GenerateAgentSetupLocal 98
 - recommended content 43
 - sequence 38

E

- errors
 - 17 through 24 86
 - 2 through 8 84
 - 25 through 35 86
 - 9 through 16 85
 - about 60
- examples
 - description 17
 - using sample code 17

F

- FetchAgentSetupBytes 100
- FetchRetrieveBytes 120
- FetchSingleFile 121
- findFiles 122
- findFiles2 123

G

- GenerateAgentSetup 98
- GenerateAgentSetupLocal 98
- GetAccountHistory 113
- GetAccountInfo 107
- GetAccounts 101
- GetAccountSummaryInfo 113
- GetAccountWebSiteSettings 114
- GetAccountWebSiteSettings2 115
- GetAccountWebSiteSettings3 116
- GetAppVersion 112
- GetBackupDates 118
- GetBackupDatesWithoutSizes 118
- GetBrandingInfo 117
- GetFedAuthAccountsFromCommunity 102

GetFedAuthAccountsFromPartnerID 103
getFileList 124
getFileList2 125
GetFilesCountForFolder 126
GetFoldersForTdate 127
GetFoldersForTdate2 128
GetIDPURLForCommunity 103
GetIDPURLForPartnerId 104
GetMediaTypes 108
GetMyRoamState 128
GetRegistrationInfo 89
getRetrieveSize 129
getRetrieveSize2 130

I

interfaces
 Account Management Web Services 16
 SOAP 16
 WSDL file 17
IsLDAPCommunity 90
IsLicenseAvailable 90
IsMyRoamLicensed 130

L

logon
 code example 27, 58
 description 26
 GetAccounts 101
 GetAppVersion 112
 GetFedAuthAccountsFromCommunity 102
 GetFedAuthAccountsFromPartnerID 103
 GetIDPURLForCommunity 103
 GetIDPURLForPartnerId 104
 recommendations for logging off 32
 recommended navigation 31
 sequence 26, 57
 SetLoginInformation 105
 VerifyTechAccountAccess 106

M

management operations
 change setting 55
 display 52
 order media 46
methods
 authenticated description 18
 authenticated overview 16
 sequence description 18
 unauthenticated overview 16
MyRoam
 getFileList 124

N

navigation
 examples 21
 recommendations for bottom 21
 recommendations for left 20
 recommendations for top 20

O

order media
 description 46
 example 51
 GetMediaTypes 108
 sequence 46
 SubmitMediaOrder 108

P

profile
 GetAccountInfo 107
 SetAccountPassword 110
 SetAccountProfile 111
 SetConfiguration 109
purpose, Website 9

R

recommended content
 backup plan 13
 general 12
 navigation 20
 system requirements 13
recommended functions 13
 account recovery 14
 backup plan and billing 14
 change password 15
 change profile 15
 order media 15
 registration and download 14
 sign in 14
RegisterAccount 94
registration
 CanRegisterToCommunity 93
 description 33
 example 44
 GetCommunityType 88
 GetRegistrationInfo 89
 IsLDAPCommunity 90
 IsLicenseAvailable 90
 recommended content 43
 RegisterAccount 94
 sequence 33
 VerifyAccountRegistratinoCode 92

- verifyAccountReservationCode 96
- verifyAccountReservationCodeEx 97
- VerifyLDAPAccountRegistration 91
- retrieve
 - description 57
 - FetchRetrieveBytes 120
 - FetchSingleFile 121
 - findFiles 122
 - findFiles2 123
 - getFileList2 125
 - GetFilesCountForFolder 126
 - GetFoldersForTdate 127
 - GetFoldersForTdate2 128
 - GetMyRoamState 128
 - getRetrieveSize 129
 - getRetrieveSize2 130
 - IsMyRoamLicensed 130
 - retrieveSelectedFiles 131
 - retrieveSelectedFilesEx 132
- retrieveSelectedFiles 131
- retrieveSelectedFilesEx 132

S

- SetAccountPassword 110
- SetAccountProfile 111
- SetConfiguration 109
- SetLoginInformation 105
- sign in
 - example 32
 - recommended content 31
- SOAP 16
- SSO (single sign on)
 - registration 33
 - reservation 33
 - SSO Sign In page 31
- SSWSAccountHistory 72
- SSWSAccountInfo 73
- SSWSAccountSummaryInfo 74
- SSWSAddressInfo 75
- SSWSBackupDateInfo 75
- SSWSBrandingInfo 77
- SSWSContactInfo 78
- SSWSMediaType 81
- SSWSNameInfo 82
- SSWSProfileInfo 82
- SSWSTicket 83
- SubmitMediaOrder 108

V

- variable types 19
- VerifyAccountRegistrationCode 92

- VerifyAccountReservationCode 96
- VerifyAccountReservationCodeEx 97
- VerifyLDAPAccountRegistration 91
- VerifyTechAccountAccess 106

W

- Website Settings from Support Center 11
- Welcome page
 - example 24
 - purpose 24
 - recommendations 24
- WSDL file 17
- wsdl_base64Binary 19
- wsdl_bool 19
- wsdl_int 19
- wsdl_long 19
- wsdl_string 19
- wsdl_unsignedLong 19
- wsdl_utc_time_in_msec 19

Send documentation feedback

If you have comments about this document, you can [contact the documentation team](#) by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Account Management Web Services Development (Micro Focus Connected Backup 9.0)

Add your feedback to the email and click **Send**.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to swpdl.ConnectedBackup.DocFeedback@microfocus.com.

We appreciate your feedback!