Admin Web Service API Reference

iAnywhere® Mobile Office 5.7

SYBASE® iAnywhere

Document ID: DC01166-01-0570-01
LAST REVISED: *September 2009*

Copyright and Trademarks

Disclaimer

This documentation, as well as the software described in it, is furnished under a license agreement. The content of this documentation is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by iAnywhere Solutions, Inc. iAnywhere Solutions, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation. Any changes in the programs will be incorporated in a future edition of this publication.

Technical Support

For product-specific technical information, visit the iAnywhere Technical Support site at http://frontline.sybase.com/support/.

NOTE: Register at our technical support site for the latest information for your product. This site is available only to customers with a valid maintenance contract.

United States
+1 800 235 7576, menu options 2, 1
+1 208 322 7575, menu options 2, 1
6:00 a.m. to 6:00 p.m. Mountain Time (GMT -7)

United Kingdom
+44 (0) 117 333 9032
8:00 a.m. to 6:00 p.m. (GMT+1)

Germany
+49 (0) 7032 798-555
8:00 a.m. to 6:00 p.m. (GMT+1)

France
+33 (0) 825 826 835
8:00 a.m. to 6:00 p.m. (GMT+1)

Benelux
+31 (0) 302 478 455
8:00 a.m. to 6:00 p.m. (GMT+1)

Table of Contents

About Admin Web Service

The admin web service API enables developers to maintain Mobile Office server from remote web service clients. The functions cover device management, template management, license management, logging and reporting.

WSDL File

To obtain the WSDL file of the admin web service, install the Mobile Office server, and set up the admin web service virtual directory in IIS. Open a web browser and point to the following URL. The server name needs to be changed based on the actual installation.

http://YourWebServer/MobileOffice/Admin.asmx?WSDL

Configuring the Mobile Office Admin Web Service

The installation of the iAnywhere Mobile Office Admin web service and Mobile Office Sample Admin Web site currently requires the Administrator to manually create virtual directories in IIS. For the current version, the IIS server hosting the Mobile Office admin web service must run on the same machine as the Mobile Office server. ASP.Net 2.0 is required to run the Mobile Office admin web service. ASP.Net 2.0 and ASP.Net AJAX Extensions 1.0 or above are required to run the admin sample web site. The actual path may be different depending on the Mobile Office installation configuration. To configure the Admin Web Service, implement the following four steps.

1) Update the Path System Variable

Open the **Control** Panel and double-click on the **System** icon.

Click the **Advanced** tab and click the **Environment Variables** button.

In the **System variables** window, double-click the **Path** item and add the following path in the **Variable value** field: C:\Program Files\iAnywhere Mobile Office\Bin

Click the **OK** buttons.

2) Create a Virtual Directory for Admin Web Service

1. From the **Start** menu, click **Administrative Tools** and select **Internet Information Services (IIS) Manager**.

Expand the name of the local computer and then expand **Web Sites**.

Right-click on **Default Web Site**, click **New**, and then click **Virtual Directory**.

When the Welcome to the Virtual Directory Creation Wizard launches, click Next.

On the **Virtual Directory Alias** page, type **MobileOffice** for the virtual directory in the Alias field and then click **Next**.

On the **Web Site Content Directory** page, browse to the location of C:\Program Files\ iAnywhere Mobile Office \Bin\WebServices.

Select the **WebServices** directory, click **OK**, and then click **Next**.

On the **Virtual Directory Access Permission** page, click the **Read** and the **Run Scripts (such as ASP)** checkboxes and then click **Next**.

On the final screen of the wizard, click **Finish**.

Right-click on the newly created **MobileOffice** directory and click **Properties**.

Select the **ASP.NET** tab and verify that the ASP.NET version is 2.0. If not, change it to 2.0 and click **OK**.

Assign administrator privilege to the user account running the Admin web service application. The Admin web service requires administrator privilege, which is not provided by the default ASPNET account. There are many ways to do so depending on the IIS and web site configuration. Two easy options are provided here for reference:

Option 1:

- Open C:\WINDOWS\Microsoft.NET \ Framework \ v2.0.50727 \ Config\Machine.config
- Replace <processModel autoConfig="true"/> with <processModel userName="SYSTEM" password="AutoGenerate" />

(To apply this option on Windows Server 2003, the process model of IIS 6 must be IIS 5.0 isolation mode.)

Option 2:

- Open C:\Program Files\ iAnywhere Mobile Office \Bin\WebServices\Web.config
- Uncomment line <!--identity impersonate="true" userName="accountname" password="password"/-->, and replace “accountname” and “password” with a user credential having administrator privilege.

3) Create a Virtual Directory for Sample Admin Web Site

1. In the **Start** menu, click **Administrative Tools** and select **Internet Information Services (IIS) Manager**.

Expand the name of the local computer and then expand **Web Sites**.

Right-click on **Default Web Site**, click **New**, and then click **Virtual Directory**.

When the **Welcome to the Virtual Directory Creation Wizard** launches, click **Next**.

On the **Virtual Directory Alias** page, type **AdminWebConsole** for the virtual directory in the Alias field and then click **Next**.

On the **Web Site Content Directory** page, browse to the location of C:\Program Files\ iAnywhere Mobile Office\Tools\WebServcies Samples\WebConsole.

Select the **WebConsole** directory, click **OK**, and then click **Next**.

On the **Virtual Directory Access Permission** page, click the **Read** and the **Run scripts (such a ASP)** checkboxes and then click **Next**.

On the final screen of the wizard, click **Finish**.

Right-click on the newly created **AdminWebConsole** virtual directory, and click **Properties**.

Select **ASP.Net** tab and verify the ASP.NET version is 2.0. If not, change it to 2.0.

4) Restart the Computer and Verify the Configuration

1. Open Internet Explorer and go to http://localhost/MobileOffice/Admin.asmx

   The page should show the available web methods in admin web service.

Change the URL to http://localhost/AdminWebConsole/Default.aspx

The page should show the Admin web console application's default page.

Note: For the current implementation, before registering devices through admin web service methods, at least one template needs to be configured on the server side by the Mobile Office Admin Win32 application. The template used by web methods must contain the valid value for the following settings:

Connection:
- Company ID
- Server Name
- Server Port

User Registration:
- Email From

Admin Web Service - Custom Types

AdminUser

Information about administrator user account

Syntax

```
public struct AdminUser
{
   public int DeviceID;
   public string UserName;
   public int DeviceStatus;
}
```

Members

DeviceID

Unique identifier of a device

UserName

Administrator account user name

DeviceStatus

The current status of a device. The possible values are:

1	**Registered**
2	**Online**
3	**Offline**
4	**Expired**
5	**Disabled**

BackendUser

Containing the backend user information.

Syntax

```
public struct BackendUser
  {
     public string FullName;
     public string ShortName;
     public string GroupList;
     public int ExistingRegistrationCount;
  }
```

Members

FullName
The full user name in the backend server.

ShortName
The short name in the backend server.

GroupList
The list of group names the user belongs to.

ExistingRegistrationCount
The count of the total device registration for this user

CloneRequest

Containing request information to clone a new device registration from an existing device registration.

Syntax

```
public struct CloneRequest
{
   public string UserName;
   public string BackendFullName;
   public int ExistingDeviceID;
   public string EmailCC;
   public string EmailBody;
   public string EmailFrom;
   public string EmailSubject;
   public string ActivationCode;
   public int ExpirationHours;
   public bool SendSMS;
   public string SMSAddress;
   public string SMSUserInstruction;
   public string SMSActivationInformation;
}
```

Members

UserName

The user name for the new registration. This member is optional.

BackendFullName

The backend full name for the new registration. This member is optional.
The BackendFullName property is used to identify the user linked to the registration. It is also used when sending technical notifications like errors or registration e-mails. The following is the list of supported Groupware platforms and the value to be used for the BackendFullName property:

Exchange = Unique Mailbox Alias (e.g. JSMITH)
Domino = FullName (e.g. JOHN SMITH/DOMINO_DOMAIN)
IMAP/POP3 = E-Mail Address (e.g. JSMITH@DOMAIN.COM)

In order register the user in no groupware mode, set this member to an empty string (“”);

If this member has its default value (NULL), or is explicitely set to NULL, then the BackendFullName in the existing device registration (specified by ExistingDeviceID) will be used into the new cloned registration.

ExistingDeviceID

The device id of the existing registration to be cloned.

EmailCC
The cc address for sending the activation email. This member is optional.

EmailBody
The registration email body. This member is optional.
If the email body contains any of the following place holders, the place holders will be replaced by the actual server settings when sending the activation email to the devices.

%CLIENT_DOWNLOAD_ADDRESS%	Client software download address
%SERVER_NAME%	Server name
%SERVER_PORT%	Server port
%COMPANY_ID%	Company ID
%USER_NAME%	User name
%ACTIVATION_CODE%	Activation code
%ACTIVATION_EXPIRATION%	Activation code expiration date

EmailFrom
The from address for sending the activation email. This member is optional.

EmailSubject
The subject of the activation email. This member is optional.

ActivationCode
The activation code for the new device registration. This member is optional.

ExpirationHours
The expiration datetime of the new device registration. This member is optional.

SendSMS
Flag to indicate whether to send a SMS activation message to the device.

SMSAddress
The SMS Address to send the activation SMS message. This member is ignored if SendSMS is false. If SendSMS is true, then a valid SMS address should be provided either in this field or in existing device regisration. . This member is optional.

SMSUserInstruction
Set the content of user instruction SMS message. This member is optional.

SMSActivationInformation;
Set the content of activation SMS message. This member is optional.

Remarks
If any optional members are missing when sending the registration request, the corresponding setting from the existing device registration (the device registration to be cloned) will be used. If the settings are also

missed in the existing device registration, then the default property value will be used. The default property value can be obtained from GetPropertyDefinitions method.

See Also
GetPropertyDefinitions

DeliveryReport

Represent a delivery record in reporting database.

Syntax

```
public struct DeliveryReport
{
   public string UserName;
   public string DeviceName;
   public int ItemType;.
   public bool ServerToClient;
   public int InsertedItems;
   public int UpdatedItems;
   public int DeletedItems;
}
```

Members

UserName
The user name of the reporing entry.

DeviceName
The device name for the reporting entry, if the device name is empty, then the delivery count is for all the devices belonging to the specified user.

ItemType
The type of the delivery report, as indicated by the following table

0	Contacts
1	Calendar
2	Tasks
3	Notepad
4	Inbox
5	Outbox
6	SendItems
7	Drafts
8	DeletedItems
9	Unknown

ServerToClient
Indicate whether the delivery items are sending from server to client, or from client to server.

InsertedItems
The count of inserted items

UpdatedItems
The count of updated items

DeletedItems
The count of deleted items.

LicenseInfo

The server's license information.

Syntax

```
public struct LicenseInfo
{
   public bool IsRegistered;
   public int AvailableLicense;
   public int TotalLicense;
   public string SerialNumber;
   public DateTime ExpireDate;
}
```

Members

IsRegistered
The flag to indicate whether the server is registered or not. The other fields are valid only if the server is registered.

AvailableLicense
The remaining license number that can be used.

TotalLicense
The total license number assigned to the server.

SerialNumber
The serial number registered for the server.

ExpireDate
The license expiration date.

LicenseRequestResponse

The result of a license request.

Syntax

```
public struct LicenseRequestResponse
{
   public int ErrorCode;
   public string ErrorMessage;
   public string ServerName;
   public string Version;
   public string RSAKey;
   public string PublicKey;
}
```

Members

ErrorCode
The error code of the request.

ErrorMessage
The message to describe the request result.

ServerName
The server name of the license request.

Version
The server software version.

RSAKey
The server's RSA key.

PublicKey
The server's public key.

LogEntry

Represent a log record in log database.

Syntax

```
public struct LogEntry
{
   public string DeviceName;
   public string UserName;
   public string Category;
   public string Status;
   public DateTime StartTime;
   public DateTime EndTime;
   public DateTime LastModified;
   public string ComputerName;
   public string Summary;
}
```

Members

DeviceName
The device name for the log entry.

UserName
The username for the log entry

Category
The category of the log entry

Status
The status of the log entry

StartTime
The start time of the log entry

EndTime
The end time of the log entry

LastModified
The last modifed time of the log entry

ComputerName
The computer name of the log entry. Only useful in load balance environment.

Summary
The summary of the log entry.

Remarks

If an operation has not yet finished, or if an operation gets interrupted abruptly in the middle without submitting its ending time, then the returned log entry for this operation will not have a valid endtime. In this case, the EndTime field will be set to the default value of (1/1/0001 12:00:00 AM).

PropertyDefinition

Property definition that can be used for managing template and device settings.

Syntax

```
public struct PropertyDefinition
{
   public int ID;
   public string Name;
   public int Type;
   public string Category;
   public string Description;
   public object DefaultValue;
   public int Platform;
   public bool ReadOnly;
   public bool SyncToDevice;
}
```

Members

ID

The unique property id.

Name

The name of the property, the property name is not unique.

Type

The type of the property value. The possible values include

1	Bool
2	Int
3	string

Category

The category of the property.

Description

The description of the property.

DefaultValue

The default value of the property. Some properties may not have default values.

Platform
The platforms which support this property. The value is a bitwise mask of all the supported platform as indicated by the following values

1	MS Pocket PC
2	MS Smartphone
4	Symbian
8	MS Windows

ReadOnly
Indicate whether the property value is a constant and can not be changed.

SyncToDevice
Indicate whether the property can be synced to device.

PropertyItem

Define a single property's data.

Syntax

```
public struct PropertyItem
{
   public int PropertyID;
   public int Mask;
   public object PropertyValue;
   public int DisplayMode ;
}
```

Members

ID
The unique property id.

Mask
When setting or getting a property's data, this mask field indicates which fields contain the valid data for the request, for example, in order to change an property's display mode to DisplayOnly (true) without changing the property's value, the mask field can be set as 2 when calling the SetDeviceSettings method, so that the PropertyValue field in the structure will be ignored.
The mask field can contain the following values:

0	All fields contain valid values
0x1	PropertyValue field contains valid value
0x2	DisplayMode field contains valid value

PropertyValue
The value for the property.

DisplayMode
The display mode of the property on device side, the valid values are as follows:

0x0	Property value can be updated on device

0x1	Property value is for display only on device

Remarks

If a DisplayMode is set to true, then the property can be viewed from the device, but it can not updated by the device user.

RegistrationRequest

The request to register a device.

Syntax

```
public struct RegistrationRequest
{
  public string UserName;
  public string BackendFullName;
  public string EmailCC;
  public string EmailBody;
  public string EmailFrom;
  public string EmailSubject;
  public string ActivationCode;
  public int ExpirationHours;
  public bool SendSMS;
  public string SMSAddress;
  public string SMSUserinstruction;
  public string SMSActivationInformation;
}
```

Members

UserName

The user name for the new registration.

BackendFullName

The backend full name for the new registration.

The BackendFullName property is used to identify the user linked to the registration. It is also used when sending technical notifications like errors or registration e-mails. The following is the list of supported Groupware platforms and the value to be used for the BackendFullName property:

Exchange = Unique Mailbox Alias (e.g. JSMITH)
Domino = FullName (e.g. JOHN SMITH/DOMINO_DOMAIN)
IMAP/POP3 = E-Mail Address (e.g. JSMITH@DOMAIN.COM)

In order to register the user in no groupware mode, set this member to an empty string (“”);

EmailCC

The cc address for sending the activation email. This member is optional.

EmailBody
The registration email body. This member is optional.
If the email body contains any of the following place holders, the place holders will be replaced by the actual server side settings when server sends the activation emails.

%CLIENT_DOWNLOAD_ADDRESS%	Client software download address
%SERVER_NAME%	Server name
%SERVER_PORT%	Server port
%COMPANY_ID%	Company ID
%USER_NAME%	User name
%ACTIVATION_CODE%	Activation code
%ACTIVATION_EXPIRATION%	Activation code expiration date

EmailFrom
The from address for sending the activation email. This member is optional.

EmailSubject
The subject of the activation email. This member is optional.

ActivationCode
The activation code for the new device registration. This member is optional.

ExpirationHours
The expiration datetime of the new device registration. This member is optional.

SendSMS
Flag to indicate whether to send a SMS activation message to the device.

SMSAddress
The SMS Address to send the activation SMS message. This member is ignored if SendSMS is false. If SendSMS is true, then a valid SMS address should be provided in this field. This member is optional if SendSMS is false.

SMSUserInstruction
Set the content of user instruction SMS message. This member is optional.

SMSActivationInformation;
Set the content of activation SMS message. This member is optional.

Remarks
If any optional members are missing when sending the registration request, if a template name is provided with the request, the missing setting will be replaced by the template settings. If the template name is not provided or if the property value is not defined in the template, then the default property value will be used. The default property value can be returned from GetPropertyDefinitions method.

See Also
GetPropertyDefinitions
RegisterUsersBySettings
RegisterUsersByTemplate

RegistrationResponse

Containing the device registration response information.

Syntax

Public struct RegistrationResponse

```
{
    public string UserName;
    public string BackendFullName;
     public int DeviceID,
    public int  ErrorCode,
    public string  ErrorMessage
}
```

Members

UserName
The user name for the new registration.

BackendFullName
The backend full name for the new registration.

DeviceID
The unique device identifier generated for the new registration.

ErrorCode
The returned error code for the request. The possible values are:

0	Success
1	Database operation error
2	Email operation error
3	Invalid argument
4	Unknown error
5	Invalid application ID
6	Invalid template setting

ErrorMessage
The detailed message returned from server for the reqeust.

RequestResponse

Containing the request response information.

Syntax

```
Public struct RequestResponse
{
  public int DeviceID,
  public int  ErrorCode,
  public string  ErrorMessage
}
```

Members

DeviceID

The unique device identifier for the original request.

ErrorCode

The returned error code for the request. The possible values are:

0	Success
1	Database operation error
2	Email operation error
3	Invalid argument
4	Unknown error
5	Invalid application ID
6	Invalid template setting

ErrorMessage

The detailed message returned from server for the reqeust.

ReregistrationRequest

The request to reregister a device.

Syntax

```
public struct ReregistrationRequest
{
   public int ExistingDeviceID;
   public string EmailCC;
   public string EmailBody;
   public string EmailFrom;
   public string EmailSubject;
   public string ActivationCode;
   public int ExpirationHours;
   public bool SendSMS;
   public string SMSAddress;
   public string SMSUserInstruction;
   public string SMSActivationInformation;
}
```

Members

ExistingDeviceID

The existing device id to be reregistered. The device id will be deleted after the reregistration.

EmailCC

The cc address for sending the activation email. This member is optional.

EmailBody

The registration email body. This member is optional.

If the email body contains any of the following place holders, the place holders will be replaced by the actual server side settings when server sends the activation emails.

%CLIENT_DOWNLOAD_ADDRESS%	Client software download address
%SERVER_NAME%	Server name
%SERVER_PORT%	Server port
%COMPANY_ID%	Company ID
%USER_NAME%	User name
%ACTIVATION_CODE%	Activation code
%ACTIVATION_EXPIRATION%	Activation code expiration date

EmailFrom

The from address for sending the activation email. This member is optional.

EmailSubject

The subject of the activation email. This member is optional.

ActivationCode
The activation code for the new device registration. This member is optional.

ExpirationHours
The expiration datetime of the new device registration. This member is optional.

SendSMS
Flag to indicate whether to send a SMS activation message to the device.

SMSAddress
The SMS Address to send the activation SMS message. This member is ignored if SendSMS is false. If SendSMS is true, then a valid SMS address should be provided either in this field or in existing device regisration. . This member is optional.

SMSUserInstruction
Set the content of user instruction SMS message. This member is optional.

SMSActivationInformation;
Set the content of activation SMS message. This member is optional.

Remarks
If any optional members are missing when sending the reregistration request, the missing setting will be replaced by the settings from the old device ID's registration. If the property value is not defined in the old device's registration, then the default property value will be used. The default property value can be returned from GetPropertyDefinitions method.

See Also
GetPropertyDefinitions
ReregisterDevices

ReregistrationResponse

Containing the reregistration result.

Syntax

```
public struct ReregistrationResponse
{
   public int OldDeviceID;
   public int DeviceID;
   public int ErrorCode;
   public string ErrorMessage;
}
```

Members

OldDeviceID

The device ID to be reregistered. This device id will be deleted after the reregistration.

DeviceID

The new device ID generated on the reregisration.

ErrorCode

The returned error code for the request. The possible values are:

0	Success
1	Database operation error
2	Email operation error
3	Invalid argument
4	Unknown error
5	Invalid application ID
6	Invalid template setting

ErrorMessage

The detailed message returned from server for the reqeust.

SubfolderSettings

Information about a single subfolder

Syntax

```
public struct SubfolderSettings
{
   public string Path;
   public bool Enabled;
   public int PreviewSize;
   public int PastDays;
}
```

Members

Path

The subfolder path. The subfolder path is the unique identifier of subfolders. If a subfolder has a parent, then the path also includes the parent subfolder's name. Parent and child subfolder names are separated with '\r' character.

Enabled

Boolean flag that enables or disables syncing of subfolder items between the device and the server

PreviewSize

The maximum number of characters (KB) in an email that are downloaded from the server to the device

PastDays

Determines the number of days the device will retain previously downloaded emails. Applies to subfolders enabled in the future

TemplateInfo

Information about a template defined on server.

Syntax

```
public struct TemplateInfo
{
   public string Name;
   public int ID;
   public string Description;
}
```

Members

Name
The template name.

ID
The template ID.

Description
The brief desription of the specified template

UserDevice

Containing the information of a registered mobile device.

Syntax

```
public struct UserDevice
{
   public string UserName;
   public string BackendFullName;
   public string DeviceType;
   public string DeviceModel;
   public int DeviceID;
   public string DeviceName;
   public int DeviceStatus;
   public int PendingItems;
   public DateTime LastDeliveryDate;
   public DateTime ActivationCodeExpireDate;
   public string ClientVersion;
}
```

Members

UserName

The Mobile Office user name.

BackendFullName

The user's fullname or unique identifier in backend server.

The BackendFullName property is used to identify the user linked to the registration. It is also used when sending technical notifications like errors or registration e-mails. The following is the list of supported Groupware platforms and the value to be used for the BackendFullName property:

Exchange = Unique Mailbox Alias (e.g. JSMITH)
Domino = FullName (e.g. JOHN SMITH/DOMINO_DOMAIN)
IMAP/POP3 = E-Mail Address (e.g. JSMITH@DOMAIN.COM)
No Groupware = empty string or null

DeviceType

The type of the mobile device. The device must connect to server at least once to have this infomation. The following table lists the supported device type strings:

Device Type String	Description
ce	For Windows Mobile Professional (WM6) or Pocket PC (WM5) device
smartphone	For Windows Mobile Standard (WM6) or Smartphone (WM5) device
epoc	For Symbian device
iphone	For iPhone

DeviceModel
The model of the mobile device. The device must connect to server at least once to have this infomation.

DeviceID
The unique identifier for a registered device generated and maintained by Mobile Office server.

DeviceName
The device name reported by a device. The device must connect to server at least once to have this infomation.

DeviceStatus
The current status of a device. The possible values are:

1	Registered
2	Online
3	Offline
4	Expired
5	Disabled

PendingItems
The pending items on the server side that needs to be sent to the device. The device must connect to server at least once to have this infomation.

LastDeliveryDate
The datetime of last item delivered from server to device. The device must connect to server at least once to have this infomation.

ActivationCodeExpireDate
The datetime of the expiration date of the device registraton.

ClientVersion
The version number of the Mobile Office client. The device must connect to server at least once to have this information.

Remarks
The DeviceType, DeviceModel, and DeviceName fields in the UserDevice structure will contain an empty string before the first time the device connects to the server. The LastDeliveryDate field will contain a default datetime value of “1/1/0001 12:00:00AM” before the first time the device connects to server. These default values will be replaced by the real values after the device successfully connects to the server at least once.

Admin Web Service - Operations

Device Management

AddAdminUser

Adds a new administrator user account

Syntax

```
public RequestResponse AddAdminUser
      (
        string strUserName,
        string strActivationCode,
        int iExpirationHours
      )
```

Request Parameters

strUserName
The unique name to identify a new administrator user account.

strActivationCode
Activation code for the admin user account

iExpirationHours
Number of hours before the account expires if not activated

Response
RequestResponse object to indicate the result of this operation

See Also
RequestResponse

CloneDeviceRegistration

Register a device by cloning an existing device registration.

Syntax

```
public ReregistrationResponse [] CloneDeviceRegistration
       (
          CloneRequest[] cloneRequests
       )
```

Request Parameters

cloneRequests

The array of clone request objects

Response

Array of RegistrationResponse objects.

Remarks

This method creates a new device registration by copying settings from an existing device registration.

See Also

RequestResponse

DeleteAdminUser

Deletes an existing administrator user account

Syntax

```
public RequestResponse DeleteAdminUser
      (
        int iDeviceID
      )
```

Request Parameters

iDeviceID

Integer unique identifier for an administrator user account

Response

RequestResponse object to indicate the result of this operation

See Also

RequestResponse

DeleteDevices

Delete registered devices from server.

Syntax

```
RequestResponse[] DeleteDevices( int[] iDeviceIDs )
```

Request Parameters

iDeviceIDs

The array of device IDs that will be deleted from server

Response

Array of RequestResponse objects.

See Also

RequestResponse

DeleteUser

Delete all registered devices from the server for the specified user.

Syntax

```
public RequestResponse[] DeleteUser( string strUserName )
```

Request Parameters

strUserName

The user name for the deleting operation

Response

Array of RequestResponse objects.

See Also

RequestResponse

GetAdminUserList

Returns a list of administrator user accounts

Syntax

```
public AdminUser[] GetAdminUserList ( )
```

Request Parameters

none

Response

Array of `AdminUser` objects.

See Also

`AdminUser`

GetDeviceList

Return all registered devices from server.

Syntax

```
public UserDevice[] GetDeviceList()
```

Request Parameters

None

Response

Array of UseDevice objects.

Remarks

This method returns all the registered devices on server. Use SearchDeviceList method to limit the records returned from the server.

See Also

UserDevice

GetDeviceListForSingleUser

Return all registered devices for a single user.

Syntax

```
public UserDevice[] GetDeviceListForSingleUser( string strUserName )
```

Request Parameters

strUserName

The user name to get the device list

Response

Array of UseDevice objects.

Remarks

This method returns all the registered devices belonging to a single user.

See Also

UserDevice

GetDeviceSettings

Return all property settings for a registered device.

Syntax

```
public PropertyItem[] GetDeviceSettings( int iDeviceID )
```

Request Parameters

iDeviceID

The device ID

Response

Array of PropertyItem objects.

Remarks

This method returns all the properties of a registered device.

See Also

PropertyItem

GetGroupwareType

Returns groupware type configured for this server instance

Syntax

```
public string GetGroupwareType ( )
```

Request Parameters

None.

Response

String representation of groupware type that the current server instance is configured with.

Groupware type can be one of the following values:

Groupware Type	Description
Email	Server is configured to work in POP/SMTP mode.
Domino	Server is configured for Lotus Domino enterprise backend
Exchange	Server is configured for Microsoft Exchange enterprise backend
None	No Groupware server installtion

GetSubfolderList

Returns a list of subfolders available for a specific device

Syntax

```
public string[] GetSubfolderList
      (
        int iDeviceID
      )
```

Request Parameters

iDeviceID

Unique device identifier for the request.

Response

String array listing all available subfolders for a device.

Remarks

The subfolder path is the unique identifier of subfolders. If a subfolder has a parent, then the path also includes the parent subfolder's name. Parent and child subfolder names are separated with '\r' character.

GetSubfolderSettings

Returns subfolder settings for a given device

Syntax

```
public SubfolderSettings[] GetSubfolderSettings
      (
        int iDeviceID
      )
```

Request Parameters

iDeviceID

Unique device identifier for the request.

Response

SubfolderSettings array listing all available subfolders for a device.

See Also

SubfolderSettings

KillDevices

Sends a remote request to specified clients to initiate deletion of client-side data. For more information about kill-pill functionality, refer to iAnywhere Mobile Office Administrator Guide, “Issuing the Kill Pill” section.

Syntax

```
public RequestResponse[] KillDevices
      (
        string[] strDeviceNames
      )
```

Request Parameters

strDeviceNames

String array listing device names

Response

RequestResponse array to indicate the result of this operation

See Also

RequestResponse

RegisterUsersBySettings

Register new device registration by specifying detailed settings. When using this method to register users, no template is required.

Syntax

```
public RegistrationResponse[] RegisterUsersBySettings
      (
        RegistrationRequest[] requests,
        PropertyItem[] settings
      )
```

Request Parameters

requests

Array of registration request objects. This parameter specifies the per user registration settings.

settings

The property settings used to register the devices. This parameter specifies the common settings that apply to the current registrations for all users.

ReadOnly properties cannot be included in the settings parameter, otherwise an excpetion will be thrown. Refer to PropertyDefinition for details of the device readonly properties.

The ServerName (PropertyID: 1) must be specified in the settings parameter, and it is the only property required for this parameter. There is no default value for ServerName property. All other properties are optional.

Response

Array of RegisrationResponse objects.

Remarks

This method registers multiple devices using the detailed property settings. If any optional property is not set in the registration request, the value will be replaced by the default property value.

If a property is not specified in any of the parameters, then the default property value will be used in the registrations. Be sure the default values (for example, the port number, CompanyID, etc.) match the server configuration.

The value of the following properties can be specified in both **RegistratonRequest[] requests** and **PropertyItem[] settings** parameters. If they are specified in both parameters, then the value specified in the requests parameter (i.e. the per user registration settings) will apply. If the value specified in the request parameter is null (for string type) or 0 (for integer type), then the value specified in the settings parameter will apply. If a property is not specified in any of the parameters, then the default property values will apply.

Property ID	Description	RegistrationRequest field
901	Registration expiration	ExpirationHours

902	Activation email body	EmailBody
903	Registration email CC list	EmailCC
905	Activation email from	EmailFrom
906	Activation email subject	EmailSubject
907	SMS activation user instruction	SMSUserInstruction
908	SMS activation information	SMSActivationInformation
909	SMS activation addresss	SMSAddress

See Also
RegistrationRequest
PropertyItem
RegistrationResponse

RegisterUsersByTemplate

Register new devices by a template setting.

Syntax

```
public RegistrationResponse[] RegisterUsersByTemplate
        (
          string strTemplateName,
          RegistrationRequest[] requests
         )
```

Request Parameters

strTemplateName
The template name for registrating the devices.

requests
Array of registration request objects.

Response
Array of RegisrationResponse objects.

Remarks
This method registers multiple devices using a template definition. If any optional property is not set in the registration request, the value will be replaced by the template settings. If the missing property is not defined in the template, then the default property value will be used for the device registration.
The specified template must have the valid value defined for the following properties:
server name, port, company id, and activation email's from address.

See Also
RegistrationRequest
RegistrationResponse

ReregisterDevices

Reregister a device.

Syntax

```
public ReregistrationResponse[] ReregisterDevices
      (
         ReregistrationRequest[] reregistrationRequests
      )
```

Request Parameters

reregistrationRequests

The information for the new device registration.

Response

Array of ReregisrationResponse objects.

Remarks

This method reregisters multiple devices. If any optional property is not set in the reregistration request, the value will be replaced by settings from the old device registration. If the old device registration does not have the property defined either, then the default property value will be used for the registration.
The reregistration will generate a new device ID, and the old device ID will be deleted.

See Also

ReregistrationRequest
ReregistrationResponse

RequestTraceFilesFromDevices

Sends a remote request to specified clients to initiate transfer of client-side trace files

Syntax

```
public RequestResponse[] RequestTraceFilesFromDevices
      (
        int[] iDeviceIDs
      )
```

Request Parameters

iDeviceIDs

Integer array listing device IDs

Response

RequestResponse array to indicate the result of this operation

Remarks

When the request is processed, the trace files will be saved into "ClientTrace" folder in “Data” folder as specified during the installation of MobileOffice.

Trace file names have the following file name structure:

yyyyMMddHHmmss_UserName_DeviceName_FileName

You can manage the debug trace level and the size of this diagnostic trace log. These settings can be obtained/modified using GetDeviceSettings and SetDeviceSettings webservice calls.

Debug Trace Level property item sets the level of detail (from 1 to 5). A higher value increases the level of detail, but also increases the time to upload the device log.

Debug Trace Size property item sets the size of the debug trace log on the device. The larger the size, the more time it takes to upload the device log.

See Also

RequestResponse

SearchBackendUsersByGroup

Search backend users based on group name. This method is only supported by Notes backend.

Syntax

```
public BackendUser[] SearchBackendUsersByGroup
      (
        string strGroupNameFilter
      )
```

Request Parameters

strGroupNameFilter

All users belong to the matched group filter will be regturned. This parameter can not be null or empty. The match is case insensitive.

The group name filter is a pattern match starting from the beginning of each group's name, for example, search filter of "Test" will include users in group of "test", "TestGroup", but not include the users in "GroupTest".

Response

Array of BackendUser objects.

Remarks

This method return all backend users who belong to the group specified by the group name filter.

See Also

BackendUser

SearchBackendUsersByName

Search backend users based on user name. This method is only supported by Notes and Exchange backend.

Syntax

```
public BackendUser[] SearchBackendUsersByName
      (
        string strUserNameFilter
      )
```

Request Parameters

strUserNameFilter

The user name to be searched on. This parameter can not be null or empty. The match is case insensitive.

The user name filter is a pattern match starting from the beginning of earch user's first name and last name. For example, search filter "first" will include both users of "FirstJoe LastJoe" and "LastJoe FirstJoe", but not include "JoeFirst JoeLast".

Response

Array of BackendUser objects.

Remarks

This method returns all backend users whose names match the search filter.

See Also

BackendUser

SearchDeviceList

Search registered device list based on input condition

Syntax

```
public UserDevice[] SearchDeviceList(UserDevice searchCondition)
```

Request Parameters

searchCondition

The search filter.

Response

Array of UserDeivce objects.

Remarks

This method returns all matched devices' registration.

In the searchCondition parameter, for members of string type, the search is case insensitive. String search uses a wildcard pattern match and all records beginning with the search criteria will be returned. In order to return the records that contain the search criteria in the middle of the data, wildcard symbol (*) should be used at both the beginning and the end of the search filter. For example, if the user name search filter is "jo", it will match "Jonathan" but not "Maryjo". If the user name search filter is "*jo*", it will match both "Jonathan" and "Maryjo".

For members of integer type, exact match "=" will be used.

If the string search filter's value is null (empty), or an integer search filter's value is -1, the search filter will be ignored in the search.

Searching by DateTime fields are not supported. The value specified in LastDeliveryDate and ActivationCodeExpireDate fields will be ignored..

See Also

UserDevice

SendUpgradeToDevices

Sends a retry request to specified devices to perform upgrade of client applications. The specified devices must have Client Auto Upgrade property enabled.

Syntax

```
public RequestResponse[] SendUpgradeToDevices
      (
        int[] iDeviceIDs
      )
```

Request Parameters

iDeviceIDs

Integer array listing device IDs

Response

RequestResponse array to indicate the result of this operation

Remark

If the Client Auto Upgrade property (Property ID: 1405) is set to true in the device settings, when the administrator upgrades to a new version of the iAnywhere Mobile Office, the new version of the client software will be automatically pushed to all the devices with Client Auto Upgrade enabled. So, normally, it is not required to call this method to upgrade the device side software.

However, if a device has Client Auto Upgrade enabled, but fails to update its software to the server side version, then calling this method will send to a retry command to the device to trigger a software upgrade operation.

If a device has Client Auto Upgrade property disabled, then calling this method on that device has no effect, the retry command will be ignored by the device.

See Also

RequestResponse

SetDeviceSettings

Update property settings for devices.

Syntax

```
public RequestResponse[] SetDeviceSettings
      (
        int[] iDeviceIDs,
        PropertyItem[] settings
      )
```

Request Parameters

iDeviceIDs
The device IDs to set the settings.

settings
The settings to update for the specified deivces

Response
Array of RequestResponse objects.

Remarks
The identical settings will be applied to all the devices specified in iDevicesIDs parameter.

See Also
PropertyItem
RequestResponse

SetSubfolderSettings

Set subfolder settings for a registered device

Syntax

```
public RequestResponse SetSubfolderSettings
      (
        int iDeviceID,
        SubfolderSettings settings,
        bool bApplySettingsToAllChildFolders
      )
```

Request Parameters

iDeviceID
Unique device identifier for the request.

settings
New settings that should be applied to a specified subfolder.

bApplySettingsToAllChildFolders
The flag to indicate whether to apply the settings to all child subfolders, if they exist.

Response
RequestResponse indicating the result of the operation.

Remarks
In order to set common settings to all available subfolders, set Path in SubfolderSettings to an empty string and bApplySettingsToAllChildFolders flag to true.

All parameters in SubfolderSettings are required. If only one parameter needs to be updated, call GetSubfolderSettings first, modify the parameter, followed by SetSubfolderSettings call.

See Also
SubfolderSettings
RequestResponse

Template Management

AddTemplate

Add a template definition to be used for device registration.

Syntax

```
public void AddTemplate
      (
        string strTemplateName,
        string strTemplateDescription,
        PropertyItem[] settings
      )
```

Request Parameters

strTemplateName
The unique name to identify the new template.

strTemplateDescription
A brief text to describe the template

settings
The settings to specified for this template

Response
None. An exception will be thrown if the operation fails.

Remarks
For properties not specified in settings parameter, the default property value will be used to create the template

See Also
PropertyItem

DeleteTemplate

Delete an existing template defined on server.

Syntax

```
public void DeleteTemplate( string strTemplateName)
```

Request Parameters

strTemplateName

The name of the template to be deleted.

Response

None. An exception will be thrown if the operation fails.

GetPropertyDefinitions

Get all the available properties defined on the server for the current backend, including property id, name, category, description, type and default value.

Syntax

```
public PropertyDefinition[] GetPropertyDefinition()
```

Request Parameters

None

Response

Array of PropertyDefinition objects.

Remarks

This method returns all the properties that can be used to register a device or create a template.

See Also

PropertyDefinition

GetTemplateList

Get a list of templates defined on server.

Syntax

```
public TemplateInfo[] GetTemplateList()
```

Request Parameters

none.

Response

A templateInfo object array.

See Also

TemplateInfo

GetTemplateSettings

Get the settings of a template.

Syntax

```
public PropertyItem[] GetTemplateSettings( string strTemplateName )
```

Request Parameters

strTempateName

The template name to get the settings.

Response

A PropertItem object array.

See Also

PropertyItem

SetTemplateSettings

Set the settings of a template.

Syntax

```
public void SetTemplateSettings
      (
        string strTemplateName,
        PropertyItem[] settings
      )
```

Request Parameters

strTempateName

The template name to get the settings.

settings

The settings to be changed for the template

Response

none

See Also

PropertyItem

License Management

GetLicenseInfo

Get the license information of the server.

Syntax

```
public LicenseInfo GetLicenseInfo()
```

Request Parameters

None

Response

A license information object.

See Also

LicenseInfo

RequestLicense

Submit the serial number of the server to request the license.

Syntax

```
public LicenseRequestResponse RequestLicense( string strSerialNumber )
```

Request Parameters

strSerialNumber

The serial number for the server to request the license

Response

A license request response object.

Remarks

If the server is connected to internet through a proxy server, use RequestLicenseWithProxy method to request a license.

See Also

LicenseRequestResponse

RequestLicenseWithProxy

Submit the serial number to request the license for server through a proxy server.

Syntax

```
public LicenseRequestResponse RequestLicenseWithPrxoy
      (
        string strSerialNumber,
        string strProxy,
        int iProxyPort,
        string strProxyUserName,
        string strProxyPassword
      )
```

Request Parameters

strSerialNumber
The serial number for the server to request the license

strProxy
Proxy server name or ip address

iProxyPort
Proxy server port

strProxyUserName
The user name, if the proxy server requires to authenticate users.

strProxyPassword
The password, if the proxy server requires to authenticate users.

Response

A license request response object.

Remarks

Only use this method if the server is connected to internet through a proxy server.

See Also

LicenseRequestResponse

Logging and Reporting

Reporting API is not supported on No Groupware servers.

DeleteLogData

Delete the log data for the specified user or device.

Syntax

```
public int DeleteLogData
(
  string strUserName,
  string strDeviceName,
  DateTime startTime,
  DateTime endTime
 )
```

Request Parameters

strUserName

The user name for the deletion. If the username is null or empty string, it matches all users. Case insensitive match.

strDeviceName

The device name for the deletion. If the device name is null or empty string, it matches all devices for the specified user. Case insensitive match.

startTime

The start time of the log to be deleted. Only those log entries whoes start time is larger than the specified **startTime**, or whose start time is null will be deleted.

endTime

The end time of the log to be deleted. Only those log entries whoes end time is smaller than the specified **endTime**, or whose end time is null will be deleted.

Response

The actual entries have been deleted from the log database.

DeleteDeliveryReportByDevice

Delete the delivery report for the specified device.

Syntax

```
public int DeleteDeliveryReportByDevice
(
  string strDeviceName,
  DateTime startTime,
  DateTime endTime
)
```

Request Parameters

strDeviceName

The device name for the deletion. If the device name is null or empty string, it matches all devices. Case sensitive match.

startTime

The start time of the delivery report to be deleted. Only those reports whose start time is larger than the specified **startTime** will be deleted.

endTime

The end time of the delivery report to be deleted. Only those report records whose end time is smaller than the specified **endTime** will be deleted.

Response

The actual record number that have been deleted from the reporting database.

Remarks

Since delivery report table stores the delivery information on hourly step, the minute and second information specified in starttime and endtime parameter will be ignored, and truncated to 0 when searching the database.

DeleteDeliveryReportByUserName

Delete the delivery report for the specified user.

Syntax

```
public int DeleteDeliveryReportByUserName
(
  string strUserName,
  DateTime startTime,
  DateTime endTime
)
```

Request Parameters

strUserName

The user name for the deletion. If the user name is null or empty string, it matches all devices. Case insenstive match.

startTime

The start time of the delivery report to be deleted. Only those reports whose start time is larger than the specified **startTime** will be deleted.

endTime

The end time of the delivery report to be deleted. Only those report records whose end time is smaller than the specified **endTime** will be deleted.

Response

The actual record number that have been deleted from the reporting database.

Remarks

Since delivery report table stores the delivery information on hourly step, the minute and second information specified in starttime and endtime parameter will be ignored, and truncated to 0 when searching the database.

GetDeliveryReportByDevice

Get the delivery report for a specified device.

Syntax

```
public DeliveryReport [] GetDeliveryReportByDevice
(
  string strDeviceName,
  DateTime startTime,
  DateTime endTime
 )
```

Request Parameters

strDeviceName

The device name to get the delivery report. If the device name is null or empty string, it matches all device. Case senstive match.

startTime

The start time of the delivery records to be returned. Only those reports whose start time is larger than the specified **startTime** will be returned.

endTime

The end time of the delivery records to be returned. Only those reports whose end time is smaller than the specified **endTime** will be returned.

Response

An array of DeliveryReport objects.

Remarks

Since delivery report table stores the delivery information on hourly step, the minute and second information specified in starttime and endtime parameter will be ignored, and truncated to 0 when searching the database.

See Also

DeliveryReport

GetDeliveryReportByUserName

Get the delivery report for the specified user.

Syntax

```
public DeliveryReport [] GetDeliveryReportByUserName
(
   string strUserName,
   DateTime startTime,
   DateTime endTime
)
```

Request Parameters

strUserName

The user name to get the delivery report. If the user name is null or empty string, it matches all device. Case insenstive match.

startTime

The start time of the delivery records to be returned. Only those reports whose start time is larger than the specified **startTime** will be returned.

endTime

The end time of the delivery records to be returned. Only those reports whose end time is smaller than the specified **endTime** will be returned.

Response

An array of DeliveryReport objects.

Remarks

Since delivery report table stores the delivery information on hourly step, the minute and second information specified in starttime and endtime parameter will be ignored, and truncated to 0 when searching the database.

See Also

DeliveryReport

GetLogData

Get the log data for the specified user or device.

Syntax

```
public LogEntry [] GetLogData
(
   string strUserName,
   string strDeviceName,
   DateTime startTime,
   DateTime endTime,
   int iMaxEntries,
   string strFilter,
   out int iTotalRecordsAvailable
)
```

Request Parameters

strUserName

The user name to get the log data. If the username is null or empty string, it matches all users. Case insenstive match.

strDeviceName

The device name to get the log data. If the device name is null or empty string, it matches all device for the specified user. Case insensitive match.

startTime

The start time of the log to be returned. Only those log entries whoes start time is larger than the specified **startTime**, or whose starttime is null will be returned.

endTime

The end time of the log to be returned. Only those log entries whoes end time is smaller than the specified **endTime**, or whose end time is null will be returned.

iMaxEntries

The maximal records to returned.

strFilter

The search filter to look up within the log entry's summary field. Case insensitive match.

iTotalRecordsAvailable

Output parameter to indicate the total records available on the server side based on the current search conditions. Comparing the returned value of this parameter to iMaxEntries to find out whether there are more records available on the server side..

Response
An array of LogEntry objects.

See Also
LogEntry