IBM Sametime
Version 9.0

IBM Sametime 9.0
Software Development Kit
Integrating Sametime with Helper Objects

Edition Notice

Note: Before using this information and the product it supports, read the information in "Notices."

This edition applies to version 9.0 of IBM Sametime (program number 5725–M36) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. Introduction

The IBM® Sametime® Helper Toolkit is an API that provides an external interface to basic functionality of the IBM Sametime Client. The Sametime Helper Toolkit is not intended to directly extend the capabilities of the Sametime Client. This toolkit differs from other Sametime client toolkits by providing an external interface to basic functionality exposed in the locally running desktop IBM Sametime Client application. Applications that integrate the Sametime Helper API are essentially able to proxy the functionality of the locally running Sametime Client (managing contacts, starting chats, Presence status).

The Sametime Helper Toolkit provides a Microsoft® Windows® native and Java object oriented API that facilitates inter-process communication between the desktop and the Sametime Client. A functionality example is the invocation of the Sametime Helper API to initiate a chat with an online contact, and results in the Sametime Client chat window opening and displaying on the desktop. Examples of integrating applications are third-party business productivity applications and Internet browsers.

Note that the Windows API ISametimeHelper included in the 8.5.2 version of Sametime has added a small number of new methods, primarily to provide new functionality for logging users in and out of Sametime. These functions were not yet included in the Java version of the API, but a similar set of additions will be available in an upcoming release.

Additional information about Sametime is available at http://www.ibm.com/sametime.

Requirements

To work with the samples described in this guide and to create your own Sametime Helper enabled applications, you will need the following:

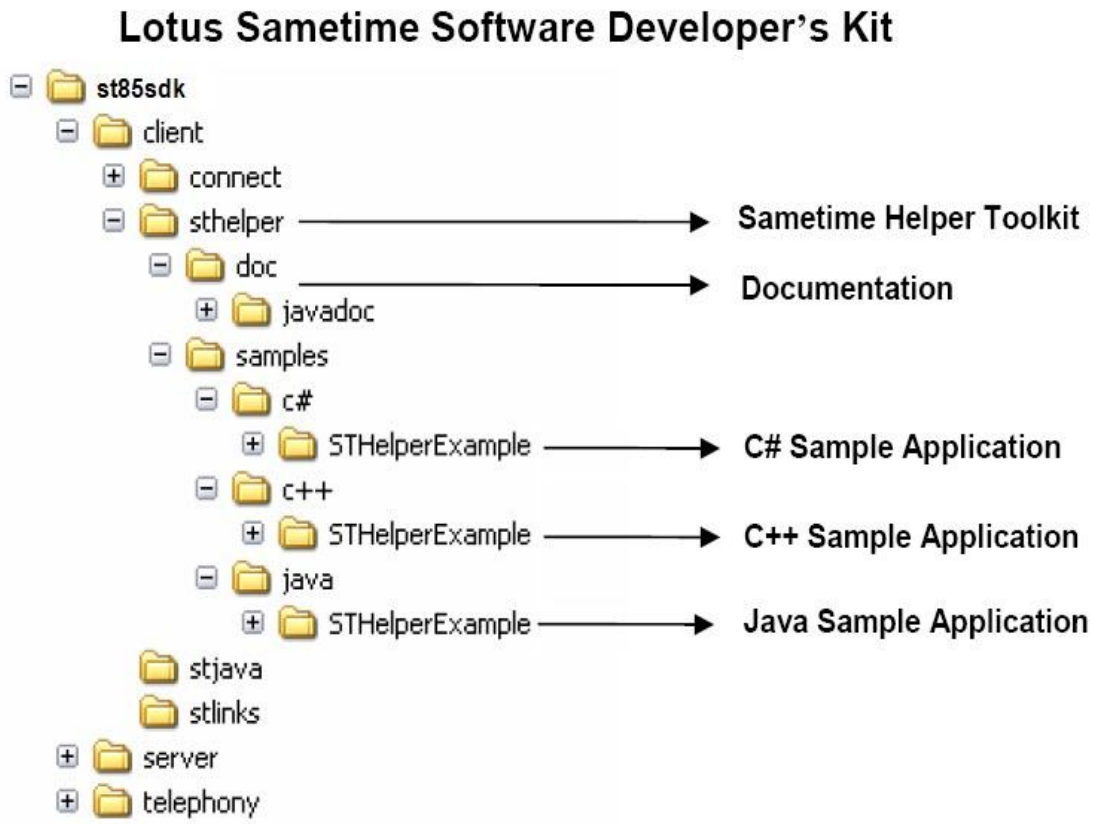
- IBM Sametime 9.0 installed and Microsoft Office Integration feature enabled in the target development environment.
- IBM Sametime 9.0 Software Development Kit
- Microsoft® Windows® operating system supported by IBM Sametime Connect 9.0. See the release notes on your IBM Sametime 9.0 server for a list of supported Windows client operating systems.

  Note that the samples documented in this guide were only tested with Microsoft Windows XP.
- Microsoft Visual Studio 2005 for native Windows platform application development (SP1 version for C# and .NET work).
- **Note** The C++ application developer audience is expected to have a thorough understanding and substantial experience with ATL and COM.
- Java Development Kit 1.4.x or greater for Java application development. Note that the samples documented in this guide were only tested with the Eclipse 3.2 Java Development Environment.

Chapter 2. Getting Started

Sametime Helper documentation overview

The following diagram outlines the Sametime Helper Toolkit contents:



This document contains references to ***STHELPER_SDK_HOME***, which is intended to reflect the toplevel directory where the Sametime Helper Toolkit is installed; for example:

```
C:\st90sdk\client\sthelper
```

This document contains references to ***SAMETIME_HOME***, which is intended to reflect the top-level directory where the IBM Sametime 9.0 Connect Client is installed on the development machine; for example:

```
C:\Program Files\IBM\Sametime Connect
```

Enabling the Sametime Helper API

To spare system resources, the Sametime Helper API is disabled by default. It is necessary to enable it before it can be used successfully. There are two different ways to enable the Sametime Helper API; the following paragraphs describe them.

Method 1: Using the EnableBroker() method of the STHelper object

This method involves calling the EnableBroker method of the STHelper object. That method will programmatically perform the steps described in the manual method below. This is the preferred way of enabling the Sametime Helper API.

See “Error: Reference source not found” of “ISametimeHelper interface” in Chapter 3 for more information on calling this method.

Note that the client must be restarted after the method has been called for the Sametime Helper API to be active.

Method 2: Manual enablement of the Sametime Helper API

This method involves opening the Sametime client plugin_customization.ini file to manually set the flag that enables the API.

1. Stop the Sametime client
2. Locate the plugin_customization.ini file in the ST_INSTALL_DIRECTORY/rcp directory
    3. Back up this file
4. Open the file with a text editor
5. Locate the entry named “com.ibm.collaboration.realtime.brokerbridge/startBroker”
6. Set this value to “true”
7. Save the file and restart the UIM client

Important: Modifying the plugin_customization.ini file in an unintended way may result in the Sametime client or some of its features not functioning properly.

Chapter 3. Sametime Helper API

Windows Native API

The Sametime Helper Toolkit provides documenation for the Sametime Helper Windows Native COM API. The COM API is exposed as Type Libraries available for integration in various Microsoft Windows programming lanaguages (such as C++, C#, and Visual Basic).

The following Sametime Helper interfaces are registered by the Sametime Client with the **Universal Unique Identifer (UUID)** `B5C34442-EE1D-4368-B861-6DC78F8EF1D7`, and are available to COM supportive programming languages as the SametimeHelper 1.0 Type Library .

Note The C++ application developer audience is expected to have a thorough understanding of, and substantial experience with, ATL and COM.

ISametimeHelper interface

The **ISametimeHelper** interface exposes a set of methods that facilitate interaction and functionality of the Sametime Connect client.

Method Summary

Table listing methods for ISametimeHelper interface

Name	Description
AddNewContact	Tells the **ISametimeHelper** to add a new Sametime contact to the accumulating set that will be sent by SendNewContacts
Chat	Tells the **ISametimeHelper** to open a chat window with the specified partner.
CreateSametimeGroup	Tells the **ISametimeHelper** to create the new specified Sametime group name.
DirectoryResolve	Tells the **ISametimeHelper** to resolve the specified directory item.
EnableBroker	Enables the Sametime client MicroBroker which is required by most STHelper functionality.
FindContact	Returns TRUE if the given contact is already included in the private groups of the Sametime client buddy list.
GetContacts	Tells the **ISametimeHelper** to retrieve the Sametime contacts for a specified group name.
GetContactsAsync	Tells the **ISametimeHelper** to retrieve the Sametime contacts for a specified group name asynchronously.
GetLocalUserID	Tells the **ISametimeHelper** to retrieve the local user id.
GetPreferredPhoneNumbers	Tells the **ISametimeHelper** to retrieve the list of preferred telephony endpoints.
GetSametimeGroups	Tells the **ISametimeHelper** to retrieve the Sametime groups for the specified group type.
GetSametimeGroupsAsync	Tells the **ISametimeHelper** to retrieve the Sametime groups for the specified group type asynchronously.
InstantShare	Tells the **ISametimeHelper** to open an InstantShare session for the designated window.
IsSametimeRunning	Returns TRUE if Sametime client is running.
LiveNameResolve	Tells the **ISametimeHelper** to resolve a list of email addresses or similar unique keys.
NWayChat	Tells the **ISametimeHelper** to open an nway chat window with the specified users.

NWayRichInstantMeeting	Tells the **ISametimeHelper** to open an instant meeting room for the specified users.
NWayVideoChat	Tells the **ISametimeHelper** to open an nway video chat window with the specified users.
NWayVoiceChat	Tells the **ISametimeHelper** to initiate an nway voice chat with the specified users.
OpenCallHistory	Tells the **ISametimeHelper** to open the call history window.
OpenCallInvitation	Tells the **ISametimeHelper** to open the call invitation window.
OpenContactList	Tells the **ISametimeHelper** to open and gives focus to the main Sametime client window ("Contact List").
OpenDialPad	Tells the **ISametimeHelper** to open the dial pad window
OpenPhoneBook	Tells the **ISametimeHelper** to open the phonebook window
OpenPreferredNumbers	Tells the **ISametimeHelper** to open the preferred numbers window.
PhoneUsingNumbers	Tells the **ISametimeHelper** to initiate a call using specified phone number
QuickFind	Tells the **ISametimeHelper** to open a generic directory resolution service dialog.
QuickView	Tells the **ISametimeHelper** to open a quick view window populated with the specified users.
RefreshWatch	Tells the **ISametimeHelper** to refresh the watch on the specified partner.
RemoveWatch	Tells the **ISametimeHelper** to remove a watch for the specified partner.
RichInstantMeeting	Tells the ISametimeHelper to open an instant meeting room invitation for the specified user.
SametimeLogin	Performs a login with a given user name and password.
SametimeLogout	Performs a logout of the current user
SendNewContacts	Tells the **ISametimeHelper** to send the set of contacts accumulated by AddNewContact to the Contact List
SetPreferredPhoneNumber	Tells the **ISametimeHelper** to select specified telephony endpoint for routing calls

SetWatch	Tells the **ISametimeHelper** to add a watch event for the specified partner.
StartSametime	Starts the Sametime client if not already running
UpdateLocalUserStatus	Tells the **ISametimeHelper** to update the local Sametime user with the specified status code.
VideoChat	Tells the **ISametimeHelper** to open a video chat window with the specified partner.
VoiceChat	Tells the **ISametimeHelper** to open a voice chat window with the specified partner.

IDL of the ISametimeHelper Interface:

```
interface ISametimeHelper : IDispatch{
	[id(12), helpstring("Starts a N-way InstantShare session")] HRESULT
NWayInstantShare([in] BSTR title, [in] VARIANT* pIdArray);
	[id(13), helpstring("Logs in the given user with optional return result")]
HRESULT SametimeLogin([in] BSTR username, [in] BSTR password, [out,retval] VARIANT*
result);
	[id(14), helpstring("Logs out the currently logged in user")] HRESULT
SametimeLogout(void);
	[id(15), helpstring("Starts Sametime if it is not already running.")] HRESULT
StartSametime([optional] VARIANT username/*optional*/,[optional] VARIANT
password/*optional*/);
	[id(16), helpstring("Starts a chat session")] HRESULT Chat([in]BSTR
partnerId);
	[id(17), helpstring("Resolves Sametime names by key")] HRESULT
LiveNameResolve([in]VARIANT* keys);
	[id(18), helpstring("Tells Sametime to watch for changes on a given user")]
HRESULT SetWatch([in]BSTR partnerId);
	[id(19), helpstring("Tells Sametime to remove a watch on a given user")]
HRESULT RemoveWatch([in]BSTR partnerId);
	[id(20), helpstring("Resolves Sametime names by key")] HRESULT
DirectoryResolve([in]BSTR key);
	[id(21), helpstring("Tells Sametime to refresh the watch on a given user")]
HRESULT RefreshWatch([in]BSTR partnerId);
	[id(22), helpstring("Starts a voice chat session")] HRESULT
VoiceChat([in]BSTR partnerId);
	[id(24), helpstring("Updates the logged in user's presence")] HRESULT
UpdateLocalUserStatus([in]LONG statusCode);
	[id(25), helpstring("returns VARIANT_TRUE if Sametime client is running.
Otherwise it returns VARIANT_FALSE")] HRESULT IsSametimeRunning([out,retval]
VARIANT_BOOL* bValue);
	[id(26), helpstring("Lauches a quickfind operation")] HRESULT
QuickFind([in]BSTR preferredAction);
	[id(27), helpstring("Returns the id of the logged in user")] HRESULT
GetLocalUserID(BSTR* userId);
	[id(28), helpstring("Gets the logged in user's community groups")] HRESULT
GetSametimeGroups([in] BSTR groupType, [out] VARIANT* groups);
```

```
      [id(29), helpstring("Returns the logged in user's contacts")] HRESULT
GetContacts([in] BSTR groupName, [out] VARIANT* contacts);
      [id(30), helpstring("Adds a Sametime contact")] HRESULT AddNewContact([in]
BSTR displayName, [in] BSTR id);
      [id(31), helpstring("method SendNewContacts")] HRESULT SendNewContacts([in]
BSTR targetGroup);
      [id(32), helpstring("Opens the Sametime contacts list")] HRESULT
OpenContactList(void);
      [id(33), helpstring("Creates a new Sametime contacts group")] HRESULT
CreateSametimeGroup([in] BSTR groupName);
      [id(34), helpstring("Launches a N-way chat")] HRESULT NWayChat([in] BSTR
topic, [in] VARIANT* pIdArray);
      [id(35), helpstring("Launches a N-way voice chat")] HRESULT
NWayVoiceChat([in] VARIANT * pIdArray);
      [id(37), helpstring("Launches avideo chat")] HRESULT VideoChat([in]BSTR
partnerId);
      [id(38), helpstring("Launches a N-way video chat")] HRESULT
NWayVideoChat([in]VARIANT* pIdArray);
      [id(39), helpstring("Starts an InstantShare session")] HRESULT
InstantShare([in]BSTR windowTitle,[optional]VARIANT userId/*optional*/);
      [id(42), helpstring("Returns TRUE if the username is found in the contacts,
otherwise FALSE")] HRESULT FindContact([in] BSTR username, [out,retval]
VARIANT_BOOL* contactFound);
      [id(43), helpstring("Starts a phone session using a specified number")]
HRESULT PhoneUsingNumber([in] BSTR partnerId, [in] BSTR partnerIdState, [in] BSTR
partnerName, [in] BSTR telephoneNumber);
      [id(44), helpstring("Opens the Dial pad")] HRESULT OpenDialPad(void);
      [id(45), helpstring("Opens user's phone book")] HRESULT OpenPhoneBook(void);
      [id(46), helpstring("Opens user's call history")] HRESULT
OpenCallHistory(void);
      [id(47), helpstring("Sets current preferred number for incoming calls")]
HRESULT SetPreferredPhoneNumber([in]BSTR endpointKey, [optional]VARIANT
altEndpointKey);
      [id(48), helpstring("Requests the list of preferred numbers for incoming
calls and ID of currently selected number")] HRESULT
GetPreferredPhoneNumbers(void);
      [id(49), helpstring("Opens a QuickView for persons specified")] HRESULT
QuickView([in] VARIANT pMapSet);
      [id(50), helpstring("Opens the Preferred Phone Numbers preference page")]
HRESULT OpenPreferredNumbers(void);
      [id(51), helpstring("Opens call invitation dialog")] HRESULT
OpenCallInvitation(void);
      [id(53), helpstring("Starts a rich Instant Meeting")] HRESULT
RichInstantMeeting([in]BSTR partnerId);
      [id(54), helpstring("Launches an N-way rich Instant Meeting session")]
HRESULT NWayRichInstantMeeting([in]VARIANT* pIdArray);
      [id(55), helpstring("Gets the logged in user's contacts (asynchronously)")]
HRESULT GetContactsAsync([in] BSTR groupName);
      [id(56), helpstring("Gets the logged in user's community groups
(asynchronously)")] HRESULT GetSametimeGroupsAsync([in] BSTR groupType);
      [id(57), helpstring("enables the broker element")] HRESULT EnableBroker();};
```

AddNewContact Method

Tells the **ISametimeHelper** to add a new Sametime contact to the accumulating set of contacts which will be sent by a subsequent SendNewContacts call.

Syntax

```
HRESULT AddNewContact (
    BSTR displayName,
```

```
    BSTR id );
```

Parameters

displayName The display name of the user to add.

id A resolvable id of the user to add -- for example this might be an e-mail address, or a Sametime ID.

Remarks

The **AddNewContact** method returns **S_OK** if successful, **S_FALSE** otherwise.

Chat Method

Tells the **ISametimeHelper** to open a chat window with the specified partner.

Syntax

```
HRESULT Chat (
    BSTR partnerId );
```

Parameters

partnerId The unique Sametime id of the target.

Remarks

The **Chat** method returns **S_OK** if successful, **S_FALSE** otherwise.

CreateSametimeGroup Method

Tells the **ISametimeHelper** to create the new specified Sametime group name.

Syntax

```
HRESULT CreateSametimeGroup (
    BSTR groupName );
```

Parameters

groupName The name of the group to create.

Remarks

The **CreateSametimeGroup** method returns **S_OK** if successful, **S_FALSE** otherwise.

Supports creation of nested groups by using the token $NESTED$

DirectoryResolve Method

Tells the **ISametimeHelper** to resolve the specified directory item.

Resolves a single search term which may have multiple matches. DirectoryResolve will only resolve the name, and will not put a watch on any resolved user. If multiple results are found based on the resolve, then all of the results will be returned in the callback.

Syntax

```
HRESULT DirectoryResolve (
    BSTR key );
```

Parameters

key The search id or name used for look up.

Remarks

The **DirectoryResolve** method returns **S_OK** if successful, **S_FALSE** otherwise.

EnableBroker Method

Tells the **ISametimeHelper** to enable the Sametime client MicroBroker, which is required by most features of the STHelper toolkit.

Syntax

```
HRESULT EnableBroker ( );
```

Parameters

NONE

Remarks

The **EnableBroker** method returns **S_OK** if successful, **S_FALSE** otherwise. The MicroBroker will be available next time the Sametime client starts.

FindContact Method

Tells the **ISametimeHelper** to query whether the specified name is present in the current private groups of the Sametime client's buddy list.

Syntax

```
HRESULT FindContact (
    BSTR username,
```

```
    VARIANT_BOOL  *contactFound  );
```
Parameters

username The username to be checked in the buddy list

contactFound The query result: TRUE if username matches a display name in one of the current private groups of the buddy list

Remarks

Public groups (e.g. those defined in a Domino directory) are not considered. Note that public group display is determined by which members of the group are currently logged in and so is constantly changing.

GetContacts Method

Tells the **ISametimeHelper** to retrieve the Sametime contacts for a specified group.

Syntax

```
HRESULT GetContacts (
    BSTR groupName,
    VARIANT * contacts );
```

Parameters

groupName The name of the group contacts.

contacts An out-param, a pointer to the resulting array of contacts.

Remarks

The **GetContacts** method returns **S_OK** if successful, **S_FALSE**

otherwise.

GetLocalUserID Method

Tells the **ISametimeHelper** to retrieve the local user id.

Syntax

```
HRESULT GetLocalUserID (
    VARIANT * userId );
```

Parameters

userId out-param, The Sametime id of the local user (the user currently logged in at the Sametime client.

Remarks

The **GetLocalUserID** method returns **S_OK** if successful, **S_FALSE** otherwise.

GetPreferredPhoneNumbers Method

Tells the **ISametimeHelper** to retrieve the list of preferred telephony endpoints.

Syntax

```
HRESULT GetPreferredPhoneNumbers ( );
```

Parameters

NONE

Remarks

The **GetPreferredPhoneNumbers** method returns **S_OK** if successful, **S_FALSE** otherwise.

GetSametimeGroups Method

Tells the **ISametimeHelper** to add to retrieve the Sametime groups for the specified group type. Returns the array of group names used in the local client buddy list.

Syntax

```
HRESULT GetSametimeGroups (
    BSTR groupType,
    VARIANT * groups );
```

Parameters

groupType The type of Sametime Contact group ie. “all”, “public”, “private”

groups The array of group names.

Remarks

The **GetSametimeGroups** method returns **S_OK** if successful, **S_FALSE** otherwise.

InstantShare Method

Requires the presence of Sametime Advanced 9.0 on the computer. Opens a Sametime Advanced Instant Share session with the application specified by windowTitle.

Syntax

```
HRESULT InstantShare (
```

```
    BSTR windowTitle);
```

Parameters

windowTitle The window title of the application to be shared

userId RESERVED, has no function in Sametime Advanced 9.0

Remarks

The **InstantShare** method returns **S_OK** if successful, **S_FALSE** otherwise.

The 9.0 version of Sametime Advanced does not support specification of a target user or users from ISametimeHelper – the userID parameter and the related function NWayInstantShare are reserved for future capability – will have no effect in clients that have Sametime Advanced 9.0 installed.

IsSametimeRunning Method

Asks the **ISametimeHelper** whether the Sametime client is already running, returns VARIANT_TRUE if it is running, VARIANT_FALSE otherwise..

Syntax

```
HRESULT IsSametimeRunning (
    VARIANT_BOOL bValue );
```

Parameters

bValue The status result, TRUE if running, FALSE otherwise

LiveNameResolve Method

Tells the **ISametimeHelper** to attempt to resolve the list of phrases via the LiveNames search service.

Syntax

```
HRESULT LiveNameResolve (
    VARIANT * keys );
```

Parameters

Keys A collection of phrases on which resolution will be attempted. Because the LiveNames service is used, the phrases must be uniquely resolvable to a single user in order to produce a match – email addresses are suitable.

Remarks

The **LiveNameResolve** method returns **S_OK** if successful, **S_FALSE** otherwise.

Results will be delivered on the onPersonUpdate.

NWayChat Method

Tells the **ISametimeHelper** to open an nway chat window with the target users.

Syntax

```
HRESULT NWayChat (
    BSTR topic,
    VARIANT * pIdArray );
```

Parameters

topic The topic to start the nway chat.

pIdArray The unique Sametime ids of the targets.

Remarks

The **NWayChat** method returns **S_OK** if successful, **S_FALSE** otherwise.

NWayRichInstantMeeting Method

Tells the **ISametimeHelper** to open an instant meeting room for the target users.

Syntax

```
HRESULT NWayRichInstantMeeting (
    VARIANT * pIdArray );
```

Parameters

pIdArray The unique Sametime ids of the targets.

Remarks

The **NWayRichInstantMeeting** method returns **S_OK** if successful, **S_FALSE** otherwise.

NWayVideoChat Method

Tells the **ISametimeHelper** to initiate a nway voice chat with the target users.

Syntax

```
HRESULT NWayVideoChat (
    VARIANT * pIdArray );
```

Parameters

pIdArray The unique Sametime ids of the targets

Remarks

The **NWayVideoChat** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user video capabilities and availability of video services.

NWayVoiceChat Method

Tells the **ISametimeHelper** to initiate an nway voice chat with the target users.

Syntax

```
HRESULT NWayVoiceChat (
    VARIANT * pIdArray );
```

Parameters

pIdArray The unique Sametime ids of the targets

Remarks

The **NWayVoiceChat** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user voice capabilities and availability of voice services. Voice call will be automatically routed according to the user voice preferences and capabilities.

OpenCallHistory Method

Tells the **ISametimeHelper** to open the call history window.

Syntax

```
HRESULT OpenCallHistory ( );
```

Parameters

NONE

Remarks

The **OpenCallHistory** method returns **S_OK** if successful, **S_FALSE** otherwise.

OpenCallInvitation Method

Tells the **ISametimeHelper** to open the call invitation window.

Syntax

```
HRESULT OpenCallInvitation ( );
```

Parameters

NONE

Remarks

The **OpenCallInvitation** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user voice capabilities and availability of voice services.

OpenContactList Method

Tells the **ISametimeHelper** to open and gives focus to the main Sametime client window ("Contact List").

Syntax

```
HRESULT OpenContactList ( );
```

Parameters

NONE

Remarks

The **OpenContactList** method returns **S_OK** if successful, **S_FALSE** otherwise. The Sametime client must be running (perhaps minimized or under other windows); this is not capable of launching the client application if it is not running.

OpenDialPad Method

Tells the **ISametimeHelper** to open the dial pad window.

Syntax

```
HRESULT OpenDialPad ( );
```

Parameters

NONE

Remarks

The **OpenDialPad** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user voice capabilities and availability of voice services.

OpenPhoneBook Method

Tells the **ISametimeHelper** to open the phonebook window.

Syntax

```
HRESULT OpenPhoneBook ( );
```

Parameters

NONE

Remarks

The **OpenPhoneBook** method returns **S_OK** if successful, **S_FALSE** otherwise.

OpenPreferredNumbers Method

Tells the **ISametimeHelper** to open the preferred phone numbers window.

Syntax

```
HRESULT OpenPreferredNumbers ( );
```

Parameters

NONE

Remarks

The **OpenPreferredNumbers** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user voice capabilities and availability of voice services.

PhoneUsingNumber Method

Tells the **ISametimeHelper** to initiate a call using specified phone number.

Syntax

```
HRESULT PhoneUsingNumber (

  BSTR partnerId,
```

```
      BSTR partnerIdState,
      BSTR partnerName,
      BSTR telephoneNumber
);
```

Parameter

partnerId The unique Sametime id if the target is a Sametime user or any other identifier otherwise

partnerIdState “Sametime” if the target is a Sametime user or “NotSametime” otherwise

partnerName The target user display name

telephoneNumber The telephone number to be dialled

Remarks

The **PhoneUsingNumber** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user voice capabilities and availability of voice services.

QuickFind Method

Tells the **ISametimeHelper** to open a generic directory resolution service dialog.

Syntax

```
HRESULT QuickFind (
    BSTR preferredAction );
```

Parameters

preferredAction RESERVED: no effect in 9.0

Remarks

The **QuickFind** method returns **S_OK** if successful, **S_FALSE** otherwise.

QuickView Method

Tells the **ISametimeHelper** to open a quick view window populated with the specified users.

Syntax

```
HRESULT QuickView (
    VARIANT pMapSet );
```

Parameters

pMapSet The pointer to the map containing the maps representing the target users

Remarks

The **QuickView** method returns **S_OK** if successful, **S_FALSE** otherwise.

pMapSet is ISametimeMapsPtr pointer to ISametimeMaps. ISametimeMaps items are ISametimeMapPtr pointers to ISametimeMap objects. Use AddMap method to add ISametimeMap to ISametimeMaps:

```
HRESULT AddMap (
        IDispatch *pMap );
```

ISametimeMap items store the following values:

groupName The name of the group to visually organise users in "Quick View" window (it is not necessarily a Sametime group, it can be any arbitrary name)
displayName The display name of the user
lookupName The name that can be used for resolution by Sametime when Sametime user ID is unknown
partnerID (optional) The unique Sametime ID of the user

Use AddItem method to add values to ISametimeMap

```
HRESULT AddItem (
    VARIANT key,
    VARIANT value
 );
```

RefreshWatch Method

Tells the **ISametimeHelper** to refresh a watch event for the specified partner.

Refreshes a watch on a user, indicating that the caller is still interested in observing presence information on the user.

The Sametime client can be configured to enforce a maximum cap on the number of contacts watched, and will perform evictions based on Least-Recently-Used tracking; the refresh watch updates the LRU timestamp to help avoid evictions. In enterprises that never cap number of contacts per client, this function can be ignored.

Syntax

```
HRESULT RefreshWatch (
    BSTR partnerId );
```

Parameters

partnerId The target to refresh the watch

Remarks

The **RefreshWatch** method returns **S_OK** if successful, **S_FALSE** otherwise.

RemoveWatch Method

Tells the **ISametimeHelper** to remove a watch event for the specified partner.

Removes a watch on a target user if the caller is no longer interested in status changes on this target. Removal of unneeded watches will reduce server and network loads, and so is always appropriate. In enterprises that use maximum contact caps, it also frees space from the contact list.

Syntax

```
HRESULT RemoveWatch (
    BSTR partnerId );
```

Parameters

partnerId The id of the target no longer to be watched.

Remarks

The **RemoveWatch** method returns **S_OK** if successful, **S_FALSE** otherwise.

RichInstantMeeting Method

Tells the **ISametimeHelper** to open an instant meeting room for the target user.

Syntax

```
HRESULT RichInstantMeeting (
    BSTR partnerId );
```

Parameters

partnerId The target user ID

Remarks

The **RichInstantMeeting** method returns **S_OK** if successful, **S_FALSE** otherwise.

SametimeLogin Method

Tells the **ISametimeHelper** to log in a user using the specified username and password data.

Syntax

```
HRESULT SametimeLogin (
    BSTR username,
    BSTR password,
    VARIANT *result );
```

Parameters

username The username to be used for the login

password The password to be used for the login

result A result object that implements an IErrorResult interface – this has two methods:

ResultString(BSTR* pVal) a result string returned by the Sametime client on the login attempt

ResultCode(SCODE* pVal) a numeric code indicating success or failure as follows:

1 = Success

2 = Server failure (for example, invalid credentials)

-1 = STHelper channel error (login could not be attempted) -

2 = Not attempted because already logged in.

Remarks

SametimeLogin can only be used with the default community, and can only succeed if the client is not already logged in on that community.

An OnSametimeLogin event will also be delivered if the login is processed successfully.

SametimeLogout Method

Tells the **ISametimeHelper** to log off all user accounts currently using the Sametime client

Syntax

```
HRESULT SametimeLogout ( );
```

Parameters

NONE

Remarks

SametimeLogout will typically be used before a SametimeLogin call for those ISametimeHelper using applications that want to switch users without a separate user interaction with the Sametime client.

This method forces logout from all communities. An OnSametimeLogout will be delivered for each community that logs out. The SametimeLogout returns S_OK if logout was attempted or FALSE to indicate there was no login in effect.

SendNewContacts Method

Tells the **ISametimeHelper** to send the set of contacts accumulated via the AddNewContact function over to the Contact List and request they be added to the specified group.

Syntax

```
HRESULT SendNewContacts (
    BSTR targetGroup );
```

Parameters

targetGroup The group to which the accumulated contacts should be added.

Remarks

The **SendNewContacts** method returns **S_OK** if successful, **S_FALSE** otherwise.

Each contact accumulated by AddNewContact is a name-id pair. For each arriving name-id pair, the Sametime client will attempt a LiveName resolution on both the name and the id. In the case of a match, the user is added to the group, and the name value used as the display name. If LiveNames cannot find a match for either name or id, then behavior depends on whether RTC Gateway is in use: if it is, the id is taken as an external address and the contact is added; if it is not, the contact cannot be added and a note displayed to the user.

SetPreferredPhoneNumber Method

Tells the **ISametimeHelper** to select specified telephony endpoint for routing calls
.

Syntax

```
HRESULT SetPreferredPhoneNumber(

      BSTR endpointKey,

      VARIANT altEndpointKey);
```

Parameters

endpointKey telephony endpoint ID

altEndpointKey is not used

Remarks

The **SetPreferredPhoneNumber** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user voice capabilities and availability of voice services.

SetWatch Method

Tells the **ISametimeHelper** to add a watch event for the specified partner.

Sets a watch on a user, which establishes this client as one interested in all future status changes for that user. Typically used after a user selection from DirectoryResolve results.

Syntax

```
HRESULT SetWatch (
    BSTR partnerId );
```

Parameters

partnerId The target to be watched.

Remarks

The **SetWatch** method returns **S_OK** if successful, **S_FALSE** otherwise.

StartSametime Method

Tells the **ISametimeHelper** to start the Sametime client, optionally with a login using the given user name and password.

Syntax

```
HRESULT StartSametime (
    VARIANT username,
    VARIANT password );
```

Parameters

username Optional, a username to use for initial login

password Optional, a password to use for initial login

Remarks

If Sametime is already running the function has no effect on the client and returns S_FALSE.

If user name and password are not specified, the client starts but is not logged in; if they are specified, they are passed as process arguments (there is no connection to accept failure information – if success/fail must be tracked, use a subsequent SametimeLogin function instead)

Behavior of splash screen and login dialog can be tailored by client properties.

UpdateLocalUserStatus Method

Tells the **ISametimeHelper** to update the local Sametime user with the specified status code.

Syntax

```
HRESULT UpdateLocalUserStatus (

    long statusCode );
```

Parameters

statusCode The status of the local user to update.

Sametime status codes:

0 = offline

1 = available

2 = away

3 = do not disturb

4 = not using

5 = in a meeting

Remarks

The **UpdateLocalUserStatus** method returns **S_OK** if successful, **S_FALSE** otherwise.

Values 2 and 4 are often treated the same, as Away – the Sametime client uses 4 to indicate that the user's status is changed to Away on account of no mouse or keyboard activity.

VideoChat Method

Tells the **ISametimeHelper** to open a video chat window with the specified partner.

Syntax

```
HRESULT VideoChat (
    BSTR partnerId );
```

Parameters

partnerId The unique Sametime id of the target

Remarks

The **VideoChat** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user video capabilities and availability of video services.

VoiceChat Method

Tells the **ISametimeHelper** to open a voice chat window with the specified partner.

Syntax

```
HRESULT VoiceChat (
    BSTR partnerId );
```

Parameters

partnerId The unique Sametime id of the target

Remarks

The **VoiceChat** method returns **S_OK** if successful, **S_FALSE** otherwise. Success depends on the user voice capabilities and availability of voice services. Voice call will be automatically routed according to the user voice preferences and capabilities.

_ISametimeHelperEvents Interface

The **_ISametimeHelperEvents** interface exposes a set of callback function prototypes to facilitate events that were encountered from the interaction with the **ISametimeHelper** interface.

Method Summary

Table listing methods for _ISametimeHelperEvents interface

Name	Description
OnSametimeLogin	The **ISametimeHelperEvent** callback method that is invoked when a user has logged in at the Sametime client.
OnSametimeLogout	The **ISametimeHelperEvent** callback method that is invoked when a user has logged out at the Sametime client.
OnPersonUpdate	The **_ISametimeHelperEvent** callback method that is invoked as a result of a successful LiveName resolution or a status change for a watched contact.
OnCapabilityEvent	The **_ISametimeHelperEvent** callback method that is invoked as a result of arriving capability information from the server.
OnDirectoryResolve	The **_ISametimeHelperEvent** callback method that is invoked as a result of an executed DirectoryResolve.
OnEvictWatch	The **_ISametimeHelperEvent** callback method that is invoked as a result of an evicition because of maximum contact limits.
OnSametimeUnavailable	The **_ISametimeHelperEvent** callback method that is invoked as a result of the STHelper discovering the Sametime client has been closed.
OnPreferredPhoneNumbersResponse	The **_ISametimeHelperEvent** callback method that is invoked as a result of arriving preferred phone numbers information from the server.
OnPreferredPhoneNumberChange	The **_ISametimeHelperEvent** callback method that is invoked as a result of changing current preferred device originating from the Sametime client.

IDL of the_ISametimeHelperEvents Interface:

```
dispinterface _ISametimeHelperEvents
{   properties:   methods:
  [id(1)] void OnPersonUpdate([in]BSTR key, [in]BSTR partnerId,
                              [in]BSTR alias, [in]long statusCode,
                              [in]BSTR statusText, [in]BSTR location);
  [id(3)] void OnDirectoryResolve([in]BSTR key, [in]VARIANT maps);
  [id(4)] void OnSametimeUnavailable([in] LONG status);
  [id(5), HRESULT OnSametimeLogin([in] BSTR communityId);
  [id(6), HRESULT OnSametimeLogout([in] BSTR communityId);
  [id(7)] void OnCapabilityEvent([in]BSTR partnerID,
```

```
                                    [in]BSTR capabilityName,
                                    [in]BSTR capabilityValue);
  [id(8)] void OnEvictWatch([in] BSTR partnerID);
  [id(13)] void OnPreferredPhoneNumbersResponse([in]VARIANT phoneMap);
  [id(14)] void OnPreferredPhoneNumberChange([in]BSTR endpointKey, [in]BSTR
altEndpointKey);

};
```

OnSametimeLogin Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of the STHelper detecting that the Sametime client has performed a login to a community.

Syntax

```
HRESULT OnSametimeLogin (
    BSTR * communityId );
```

Parameters

communityId The communityId of the community that was joined – can be ignored in enterprises using only a single community.

Remarks

The **OnSametimeLogin** method will fire for any login, including non-default communities in a multi-community environment. In general, ISametimeHelper is intended for use in single-community environments.

The function should return **S_OK** if successful, **S_FALSE** otherwise.

OnSametimeLogout Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of the STHelper detecting that the Sametime client has performed a logout to a community.

Syntax

```
HRESULT OnSametimeLogout (
    BSTR * communityId );
```

Parameters

communityId The communityId of the community that was logged out – can be ignored in enterprises using only a single community.

Remarks

The **OnSametimeLogout** method will fire for any login, including non-default communities in a multi-community environment. In general, ISametimeHelper is intended for use in single-community environments.

The function should return **S_OK** if successful, **S_FALSE** otherwise.

OnPersonUpdate Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of a LiveNameResolve method invocation or when a presence status change has been received from the Sametime server.

Syntax

```
    HRESULT OnPersonUpdate (
        BSTR key,
        BSTR partnerId,
        BSTR alias,
long statusCode,
BSTR statusText,
        BSTR location );
```

Parameters

key The "lookupName" that was sent in the initial request, if the current call is the direct result of a lookup. If left null, the call is the result of a status change being received from the Sametime server.

partnerId The Sametime ID of the contact **alias** The display name of the contact

statusCode The numeric code for the contact's current presence status, see UpdateLocalUserStatus

statusText The text message from the contact's presence status if available.

location The location text for the contact if available.

Remarks

The **OnPersonUpdate** method should return **S_OK** if successful, **S_FALSE**

otherwise. Some fields may not always be present, based on client version and settings of the user being reported.

OnPersonUpdate will be called for all status changes that the local Sametime client is receiving – including Persons in the client buddylist or being watched because of other STHelper-based applications. The application must assume it will receive events for users that it had never itself requested.

OnCapabilityEvent Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of..

Syntax

```
    HRESULT OnCapabilityEvent (
        BSTR partnerId,
        BSTR capabilityName,
        BSTR capabilityValue );
```

Parameters **partnerId** The Sametime ID for which the capability is being reported.

capabilityName The name of the capability.

"voiceChat"
"telephony"
“com.ibm.st.capVideoChat”
“instantmeeting"

capabilityValue The value for that capability, for example “true” or “false”

Remarks

The **OnCapabilityEvent** method should return **S_OK** if successful, **S_FALSE**

otherwise.

OnDirectoryResolve Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of a DirectoryResolve method invocation.

Syntax

```
HRESULT OnDirectoryResolve (
    BSTR key,
const _variant_t & maps );
```

Parameters

key The lookup name used in the DirectoryResolve call for which these results are reported.

maps An array of Map objects, which in turn hold name-value pairs of data including data with the the following names: partnerAlias (user’s display name), partnerID, statusText, statusCode, location

Remarks

The **OnDirectoryResolve** method should return **S_OK** if successful, **S_FALSE**

otherwise.

The collection of map entries can vary, based on client versions and settings of users in the result set.

OnEvictWatch Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of an eviction from the watch list on the Sametime client.

Syntax

```
HRESULT OnEvictWatch (
    BSTR partnerId );
```

Parameters **partnerId** The Sametime

ID of the contact no longer being watched.

Remarks

The OnEvictWatch will only be called in the case where the Sametime client has been configured to limit the maximum number of simultaneous contact watches. For such a client, Least-Recently-Used tracking is applied if a new watch is requested when the maximum has already been reached: the entry with the oldest LRU timestamp will have its watch removed, and the OnEvictWatch will be distributed to let the STHelper user know that presence information for that user is no longer available.

The arrival of an OnEvictWatch means presence info is not tracked, but it does not invalidate the Sametime ID or display name. The STHelper user can again request a setWatch if the presence information is needed.

The **OnEvictWatch** method should return **S_OK** if successful, **S_FALSE** otherwise.

OnSametimeUnavailable Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of the STHelper detecting that the Sametime client has been exited.

Syntax

```
    HRESULT OnSametimeUnavailable
(         long * status );
```

Parameters

status One of the values

0 = "client disconnected"

1 = "client connected"

Remarks

The **OnSametimeUnavailable** method should return **S_OK** if successful, **S_FALSE**

otherwise.

OnPreferredPhoneNumbersResponse Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of arriving preferred phone numbers information from the server.

Syntax

```
    HRESULT OnPreferredPhoneNumbersResponse (
        VARIANT phoneMap );
```

Parameters

phoneMap The pointer to the map of telephony endpoints

Remarks

The **OnPreferredPhoneNumbersResponse** method will fire as a result of GetPreferredPhoneNumbers request. phoneMap is ISametimeMapsPtr pointer to ISametimeMaps.

ISametimeMaps items are ISametimeMapPtr pointers to ISametimeMap objects. Use AddMap method to add ISametimeMap to ISametimeMaps:

```
HRESULT AddMap (
    IDispatch *pMap );
```

ISametimeMap items store the following values:

key endpoint ID

label display name of the endpoint ("Office", "Home" etc.)

displayValue phone number or name of the device ("Computer", “+1 234 5678”)

preferred true when the endpoint is selected as preferred false or omitted otherwise

isComputer is true when the endpoint is a computer false or omitted otherwise

isTelephone is true when the endpoint is a phone set false or omitted otherwise

isUserDefined is true when the endpoint is defined by user false or omitted otherwise

isVideoDevice is true when the endpoint has video capabilities false or omitted otherwise

Use AddItem method to add values to ISametimeMap

```
HRESULT AddItem (
    VARIANT key,
    VARIANT value
 );
```

OnPreferredPhoneNumberChange Method

The **_ISametimeHelperEvent** callback method that is invoked as a result of arriving preferred phone numbers information from the server.

Syntax

```
HRESULT OnPreferredPhoneNumberChange (
   BSTR endpointKey,
   BSTR altEndpointKey );
```

Parameters

endpointKey telephony endpoint ID

altEndpointKey is not used

Remarks

The **OnPreferredPhoneNumberChange** method will fire as a result of changing current preferred device originating from the Sametime client. It is not triggered by invoking SetPreferredPhoneNumber method.

Java API

JSTHelper

The **JSTHelper** object exposes a set of methods that facilitate interaction and functionality of the Sametime Connect client.

Please refer to the JavaDoc of the **JSTHelper** object, which is located in the "***STHELPER_SDK_HOME***\doc\javadoc" directory of the Sametime Helper Toolkit.

The provides a detailed description of the publicly accessible methods

JSTCallBack

The **JSTCallBack** interface exposes a set of methods to facilitate events that were encountered from the interaction with the **STHelper** object.

Please refer to the JavaDoc of the **JSTCallBack** interface, which is located in the "***STHELPER_SDK_HOME***\doc\javadoc" directory of the Sametime Helper Toolkit.

The provides a detailed description of the publicly accessible methods

Chapter 4. Sametime Helper Toolkit by Example

The Sametime Helper Toolkit contains a set of sample applications. The sample applications are intended to maintain consistent behavior and usage of the Sametime Helper Toolkit. The following sections demonstrate the detail of using the example applications for each of the development languages.

C# Sample Application

The Sametime Helper Toolkit contains a C# sample application, and a Visual Studio C# Sample Project "STHelperExample.csproj" in the "***STHELPER_SDK_HOME***\samples\c#" directory of the Sametime Helper Toolkit.

To get started with the C# Sample Project:

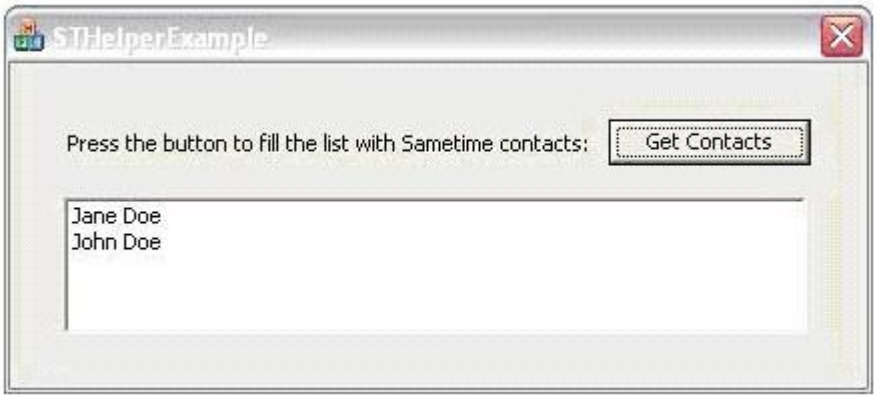
1. Start Microsoft Visual Studio and click File > Open > Project/Solution.

2. Navigate to the appropriate Sametime Helper Toolkit installation directory (***STHELPER_SDK_HOME***) and select the "STHelperExample.csproj" file; for example:

```
C:\st90sdk\client\sthelper\samples\c#\STHelperExample\STHelperExample.csproj
```

3. Build the sample application.

The running C# sample application should look similar to the following picture:



Details of the main initialization of the example application are found in:

STHELPER_SDK_HOME \c#\STHelperExample\src\STHelperExampleDlg.cs

The **STHelperExampleDlg** constructor creates an instance of the **STHelperExampleEventHandler** object. The **STHelperExampleEventHandler** object is an implementation of the **_ISametimeHelperEvents** interface, and processes the Sametime Helper event notification callbacks.

```
// The Sametime Helper event handler.
STHelperExampleEventHandler stHelperEventHandler = null;

public STHelperExampleDlg()
{
  InitializeComponent();
  // Create the STHelperExampleEventHandler interface to handle
  // the Sametime Helper related events.
stHelperEventHandler = new STHelperExampleEventHandler(); }
```

Details of Sametime Helper usage are located in the **getSametimeContacts()** method defined in:

STHELPER_SDK_HOME \c#\STHelperExample\src\STHelperExampleDlg.cs

The following method details how to use the **SametimeHelper** interface to obtain the list of Sametime group names, and the list of contacts located in each group.

```
//Method that uses the Sametime Helper Library to request
//the list of IBM Sametime Connect Client contacts, and populate
//the UI List control with the returned list of contacts.
private void getSametimeContacts() {   //Create the
Sametime Helper object.
  STHelperLib.SametimeHelper stHelper =
new STHelperLib.SametimeHelper();

  //Sametime Group type
  String strGroupType = "all";   //Define an object to
hold Sametime Contact Group names.
  Object objGroups = null;
```

```
  //Invoke the Sametime Helper to populate the group names object.
stHelper.GetSametimeGroups(strGroupType, out objGroups);

  if (objGroups != null)   {     //Assign the array that has the
Sametime Contact Group names.
    Array arrGroups = ((Array) objGroups);


    //Loop through the Sametime Contact Group names and retrieve
    //the Sametime Contact names for each group.
for (int i = 0; i < arrGroups.Length; i++)
{       //Assign the current group name.
      String strGroupName = (String)arrGroups.GetValue(i);

      //Define array to hold Sametime Contact Contact
      //names for the current group
      Object objContacts = null;

      //Invoke the Sametime Helper to populate the contact
      //names object for the current group.
stHelper.GetContacts(strGroupName, out objContacts);

      if (objContacts != null)       {         //Assign the array
that has the current group contact names.
        Array arrContacts = ((Array) objContacts);
        //Loop through the Sametime Contact names and add each name
        //to the List control.         for (int ct = 0;
ct < arrContacts.Length; ct++)         {
          //Add the contact name to the List control.
          listContacts.Items.Add(arrContacts.GetValue(ct));
        }
      }
    }
  }
}
```

Details of the Sametime Helper example **_ISametimeHelperEvents** interface implementation are found in:

STHELPER_SDK_HOME\c#\STHelperExample\src\STHelperExampleEventHandler.cs

The **STHelperExampleEventHandler** object is an implementation of the **_ISametimeHelperEvents** interface, which registers for notification and processes the Sametime Helper event notification callbacks.

```
class STHelperExampleEventHandler
{   //The Sametime Helper object.
  STHelperLib.SametimeHelper stHelper = null;

  public STHelperExampleEventHandler()
{
    //Create the Sametime Helper object.
stHelper = new STHelperLib.SametimeHelper();

    //Register the Sametime Helper event callbacks.
```

```
    //event _ISametimeHelperEvents_OnPersonUpdateEventHandler
stHelper.OnPersonUpdate += new STHelperLib.
      _ISametimeHelperEvents_OnPersonUpdateEventHandler(
        this.OnPersonUpdateEventHandler);     //event
_ISametimeHelperEvents_OnCapabilityEventEventHandler
stHelper.OnCapabilityEvent += new STHelperLib.

_ISametimeHelperEvents_OnCapabilityEventEventHandler(         this.OnCapabilityEvent
EventHandler);

    //event _ISametimeHelperEvents_OnDirectoryResolveEventHandler
stHelper.OnDirectoryResolve += new STHelperLib.

_ISametimeHelperEvents_OnDirectoryResolveEventHandler(         this.OnDirectoryResol
veEventHandler);

    //event _ISametimeHelperEvents_OnEvictWatchEventHandler
stHelper.OnEvictWatch += new STHelperLib.

_ISametimeHelperEvents_OnEvictWatchEventHandler(         this.OnEvictWatchEventHandl
er);

    //event _ISametimeHelperEvents_OnSametimeUnavailableEventHandler
stHelper.OnSametimeUnavailable += new STHelperLib.
      _ISametimeHelperEvents_OnSametimeUnavailableEventHandler(
        this.OnSametimeUnavailableEventHandler);
  }

  //ST Helper event callbacks
  //event _ISametimeHelperEvents_OnPersonUpdateEventHandler
public void OnPersonUpdateEventHandler(string key,
            string partnerId,
string alias,             int
statusCode,             string
statusText,             string
location)
  {
    MessageBox.Show("OnPersonUpdateEventHandler");
  }

  //event _ISametimeHelperEvents_OnCapabilityEventEventHandler
  public void OnCapabilityEventEventHandler(string
partnerId,             string capabilityName,
string capabilityValue)
  {
    MessageBox.Show("OnCapabilityEventEventHandler");
  }

  //event _ISametimeHelperEvents_OnDirectoryResolveEventHandler
public void OnDirectoryResolveEventHandler(string key, object maps)
  {
    MessageBox.Show("OnDirectoryResolveEventHandler");
  }

  //event _ISametimeHelperEvents_OnEvictWatchEventHandler
public void OnEvictWatchEventHandler(string partnerId)
  {
```

```
    MessageBox.Show("OnEvictWatchEventHandler");
  }



  //event _ISametimeHelperEvents_OnSametimeUnavailableEventHandler
public void OnSametimeUnavailableEventHandler(int status)
  {
    MessageBox.Show("OnSametimeUnavailableEventHandler");
  }
}
```

C# Application Development Reminders

Key points to remember when developing a Sametime Helper enabled C# application:

Ensure that IBM Sametime 9.0 Connect Client is installed on the development target machine as identified by ***SAMETIME_HOME***.

Add Sametime Helper Library to your C# application.

From Visual Studio Solution Explorer > References >Add Reference. Select the COM tab and scroll down to the list and select the COM SametimeHelper 1.0 Type Library.

Create an instance of the **`SametimeHelper`** interface.

```
//Create the Sametime Helper object.
STHelperLib.SametimeHelper stHelper =
new STHelperLib.SametimeHelper();
```

C++ Sample Application

The Sametime Helper Toolkit contains a C++ sample application, and a Visual Studio C++ sample project “STHelperExample.vcproj” in the “***`STHELPER_SDK_HOME`*** `\samples\c++`” directory of the Sametime Helper Toolkit.

Note The C++ application developer audience is expected to have a thorough understanding of, and substantial experience with, ATL and COM.

To get started with the C++ Sample Application:

1. Start Microsoft Visual Studio and click File > Open > Project/Solution.

2. Navigate to the appropriate Sametime Helper Toolkit installation directory (***STHELPER_SDK_HOME***) and select the “STHelperExample.vcproj” file; for example:

   ```
   C:\st90sdk\client\sthelper\samples\c++\STHelperExample\
   STHelperExample.vcproj
   ```

3. Build the sample application.

The running C++ sample application should look similar to the following picture:

Details of the main initialization of the example application are found in:

STHELPER_SDK_HOME \c++\STHelperExample\src\STHelperExampleDlg.cpp

The **STHelperExampleDlg** constructor creates an instance of the **ISTHelperExampleEventHandler** interface. The **ISTHelperExampleEventHandler** instance is an implementation of the **ISametimeHelperEvents** interface, and processes the Sametime Helper event notification callbacks.

```
BOOL CSTHelperExampleDlg::OnInitDialog()
{
  CDialog::OnInitDialog();

  // Set the icon for this dialog. The framework does
  // this automatically when the application's main
// window is not a dialog.
  SetIcon(m_hIcon, TRUE);           // Set big icon
  SetIcon(m_hIcon, FALSE);          // Set small icon

  // Declare the STHelperExampleEventHandler pointer.
  ISTHelperExampleEventHandler *pSTHelperExampleEventHandler = NULL;



  // Create the STHelperExampleEventHandler interface to handle
// the Sametime Helper related events.
  HRESULT hr =
                 CoCreateInstance( CLSID_STHelperExampleEventHand
                 ler,
                 NULL,
                 CLSCTX_INPROC_SERVER,
                 IID_ISTHelperExampleEventHandler,
                 (void**) &pSTHelperExampleEventHandler);

  // Return for successful initialization, false otherwise.
if (SUCCEEDED(hr))
  { return TRUE;
  }
else
  { return FALSE;
  }
}
```

Details of Sametime Helper usage are located in the **getSametimeContacts()** method defined in:

STHELPER_SDK_HOME \c++\STHelperExample\src\STHelperExampleDlg.cpp

The following method details how to use the **SametimeHelper** interface to obtain the list of Sametime group names, and the list of contacts located in each group.

```
//Method that uses the IBM Sametime Helper Library to request
//the list of IBM Sametime Connect Client contacts, and populate
//the UI List control with the returned list of contacts.
void CSTHelperExampleDlg::getSametimeContacts()
{
  //Sametime Helper object.
  STHelperLib::ISametimeHelperPtr pSTHelper;

  //Create the Sametime Helper object.
  HRESULT hr =
pSTHelper.CreateInstance(__uuidof(STHelperLib::SametimeHelper));

  if (hr == S_OK)
{     //Sametime Group
type.
    BSTR bstrGroupType = L"all";

    //Define array to hold Sametime Contact Group names.
    VARIANT vGroupsArray;
vGroupsArray.vt = VT_VARIANT;
    //Invoke the Sametime Helper to populate the group names array.
hr = pSTHelper->GetSametimeGroups(bstrGroupType, &vGroupsArray);

    if (hr == S_OK)     {       //Pointer for
accessible array of groups.
      BSTR HUGEP *pbstrGroups;

      //Assign the groups array pointer.
      SAFEARRAY *psaGroups = vGroupsArray.parray;
      //Get a pointer to the elements of the groups array.       hr =
SafeArrayAccessData(psaGroups, (void HUGEP**)&pbstrGroups);

      if (hr == S_OK)
      {
        //Loop through the Sametime Contact Group names and retrieve
        //the Sametime Contact names for each group.         for
(ULONG i = 0; i < psaGroups->rgsabound->cElements; i++)
{           //Assign the current group name.
          BSTR bstrGroupName = pbstrGroups[i];

          //Define array to hold Sametime Contact Contact
//names for the current group.
          VARIANT vContactsArray;
vContactsArray.vt = VT_VARIANT;

          //Invoke the Sametime Helper to populate the contact
          //names array for the current group.           hr =
pSTHelper->GetContacts(bstrGroupName, &vContactsArray);

          if (hr == S_OK)
{             //Pointer for accessible array of
groups.
```

```
            BSTR HUGEP *pbstrContacts;
            //Assign the array that has the current group
//contact names.
            SAFEARRAY *psaContacts = vContactsArray.parray;
            //Get a pointer to the elements of the contacts array.
            hr = SafeArrayAccessData(
                   psaContacts, (void HUGEP**)&pbstrContacts);
            if (hr == S_OK)
            {
              //Loop through the Sametime Contact names and add
              //each name to the List control.
for (ULONG j = 0;
                    j < psaContacts->rgsabound->cElements; j++)
{
                //Add the contact name to the List control.
m_ListContacts.AddString(pbstrContacts[j]);               }

              //Release the contacts array pointer.
              SafeArrayUnaccessData(psaContacts);
            }
          }
        }

        //Release the groups array pointer.
        SafeArrayUnaccessData(psaGroups);
      }
    }
  }
}
```

Details of the Sametime Helper example **_ISametimeHelperEvents** interface implementation ATL registration and deregistration are found in:

STHELPER_SDK_HOME\c++\STHelperExample\src\STHelperExampleEventHandler.h

The **STHelperExampleEventHandler** object is an implementation of the **ISametimeHelperEvents** interface, which registers for notification and processes the Sametime Helper event notification callbacks.

The following details the example **_ISametimeHelperEvents** interface implementation of the ATL event dispatch requirements:

```
// CSTHelperExampleEventHandler

//Define an ATL _ATL_FUNC_INFO for each of the _ISametimeHelperEvents
//dispinterface event functions.
_ATL_FUNC_INFO OnPersonUpdateInfo = { CC_STDCALL, VT_EMPTY, 6, { VT_BSTR, VT_BSTR,
VT_BSTR, VT_I4,  VT_BSTR, VT_BSTR} };
_ATL_FUNC_INFO OnCapabilityInfo = {CC_STDCALL, VT_EMPTY, 3, {VT_BSTR, VT_BSTR,
VT_BSTR}};
_ATL_FUNC_INFO OnDirectoryResolveInfo = { CC_STDCALL, VT_EMPTY, 2, {VT_BSTR,
VT_VARIANT} };
_ATL_FUNC_INFO OnEvictWatchInfo = {CC_STDCALL, VT_EMPTY, 1, {VT_BSTR}};
_ATL_FUNC_INFO OnSametimeUnavailableInfo = {CC_STDCALL, VT_EMPTY, 1, {VT_I4}};
```

```
//Define constants for each of the _ISametimeHelperEvents
//ATL dispinterface event function ids. static const long
STHELPER_EVENT_DISPID_LIVENAME_RESOLVE     = 1; static const long
STHELPER_EVENT_DISPID_CAPABILITY           = 7; static const long
STHELPER_EVENT_DISPID_DIRECTORY_RESOLVE    = 3; static const long
STHELPER_EVENT_DISPID_EVICT_WATCH          = 8; static const long
STHELPER_EVENT_DISPID_SAMETIME_UNAVAILABLE = 4;

//Define a constant for each of the _ISametimeHelperEvents events.
static const unsigned int STHELPER_EVENT_LIVENAME_RESOLVE     = 1000;
static const unsigned int STHELPER_EVENT_CAPABILITY_EVENT     =
  STHELPER_EVENT_LIVENAME_RESOLVE + 1;
static const unsigned int STHELPER_EVENT_DIRECTORY_RESOLVE    =
  STHELPER_EVENT_CAPABILITY_EVENT + 1;
static const unsigned int STHELPER_EVENT_EVICT_WATCH          =
  STHELPER_EVENT_DIRECTORY_RESOLVE + 1;
static const unsigned int STHELPER_EVENT_SAMETIME_UNAVAILABLE =
STHELPER_EVENT_EVICT_WATCH + 1;

class ATL_NO_VTABLE CSTHelperExampleEventHandler :
public CComObjectRootEx<CComSingleThreadModel>,
public CComCoClass<CSTHelperExampleEventHandler,
    &CLSID_STHelperExampleEventHandler>,
  public IDispatchImpl<ISTHelperExampleEventHandler,
    &IID_ISTHelperExampleEventHandler, &LIBID_STHelperExampleLib,
    /*wMajor =*/ 1, /*wMinor =*/ 0>,


 public IDispatchImpl<_ISametimeHelperEvents,
    &__uuidof(_ISametimeHelperEvents), &LIBID_STHelperLib,
    /* wMajor = */ 1>,

  //Add an ATL IDispEventSimpleImpl reference for each of the
  //_ISametimeHelperEvents events.   public
IDispEventSimpleImpl<STHELPER_EVENT_LIVENAME_RESOLVE,
    CSTHelperExampleEventHandler,
    &__uuidof(::STHelperLib::_ISametimeHelperEvents)>,
public IDispEventSimpleImpl<STHELPER_EVENT_CAPABILITY_EVENT,
    CSTHelperExampleEventHandler,
    &__uuidof(::STHelperLib::_ISametimeHelperEvents)>,   public
IDispEventSimpleImpl<STHELPER_EVENT_DIRECTORY_RESOLVE,
    CSTHelperExampleEventHandler,
    &__uuidof(::STHelperLib::_ISametimeHelperEvents)>,
public IDispEventSimpleImpl<STHELPER_EVENT_EVICT_WATCH,
    CSTHelperExampleEventHandler,
    &__uuidof(::STHelperLib::_ISametimeHelperEvents)>,
  public IDispEventSimpleImpl<STHELPER_EVENT_SAMETIME_UNAVAILABLE,
    CSTHelperExampleEventHandler,
    &__uuidof(::STHelperLib::_ISametimeHelperEvents)>
{ public
:
  CSTHelperExampleEventHandler()
  {
  }

DECLARE_REGISTRY_RESOURCEID(IDR_STHELPEREXAMPLEEVENTHANDLER)
```

```
  BEGIN_COM_MAP(CSTHelperExampleEventHandler)
   COM_INTERFACE_ENTRY(ISTHelperExampleEventHandler)
   COM_INTERFACE_ENTRY2(IDispatch, _ISametimeHelperEvents)
COM_INTERFACE_ENTRY(_ISametimeHelperEvents)
  END_COM_MAP()

  //Add the ATL SINK_MAP entries for the
//ISametimeHelperEvents dispinterface events.
  BEGIN_SINK_MAP(CSTHelperExampleEventHandler)
    SINK_ENTRY_INFO(STHELPER_EVENT_LIVENAME_RESOLVE,
      __uuidof(::STHelperLib::_ISametimeHelperEvents),
      /*dispid*/ STHELPER_EVENT_DISPID_LIVENAME_RESOLVE,
      OnPersonUpdate, &OnPersonUpdateInfo)

    SINK_ENTRY_INFO(STHELPER_EVENT_CAPABILITY_EVENT,
      __uuidof(::STHelperLib::_ISametimeHelperEvents),
      /*dispid*/ STHELPER_EVENT_DISPID_CAPABILITY,
      OnCapabilityEvent, &OnCapabilityInfo)

    SINK_ENTRY_INFO(STHELPER_EVENT_DIRECTORY_RESOLVE,
      __uuidof(::STHelperLib::_ISametimeHelperEvents),
      /*dispid*/ STHELPER_EVENT_DISPID_DIRECTORY_RESOLVE,
      OnDirectoryResolve, &OnDirectoryResolveInfo)

    SINK_ENTRY_INFO(STHELPER_EVENT_EVICT_WATCH,
      __uuidof(::STHelperLib::_ISametimeHelperEvents),
      /*dispid*/ STHELPER_EVENT_DISPID_EVICT_WATCH,
      OnEvictWatch, &OnEvictWatchInfo)
    SINK_ENTRY_INFO(STHELPER_EVENT_SAMETIME_UNAVAILABLE,
      __uuidof(::STHelperLib::_ISametimeHelperEvents),
      /*dispid*/ STHELPER_EVENT_DISPID_SAMETIME_UNAVAILABLE,
      OnSametimeUnavailable, &OnSametimeUnavailableInfo)
  END_SINK_MAP()

  DECLARE_PROTECT_FINAL_CONSTRUCT()
```

The following details the event dispatch registration of the example ATL **`_ISametimeHelperEvents`** interface implementation:

```
  HRESULT FinalConstruct()
  {     //Create the Sametime Helper
object.
    HRESULT hr =
pSTHelper.CreateInstance(__uuidof(STHelperLib::SametimeHelper));

    //Establish the ATL connections for the
    //ISametimeHelperEvents dispinterface events.     hr =
IDispEventSimpleImpl<STHELPER_EVENT_LIVENAME_RESOLVE,
CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
           DispEventAdvise(pSTHelper);
    hr = IDispEventSimpleImpl<STHELPER_EVENT_CAPABILITY_EVENT,
CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
           DispEventAdvise(pSTHelper);
```

```
    hr = IDispEventSimpleImpl<STHELPER_EVENT_DIRECTORY_RESOLVE,
            CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
            DispEventAdvise(pSTHelper);
    hr = IDispEventSimpleImpl<STHELPER_EVENT_EVICT_WATCH,
CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
          DispEventAdvise(pSTHelper);
    hr = IDispEventSimpleImpl<STHELPER_EVENT_SAMETIME_UNAVAILABLE,
CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
          DispEventAdvise(pSTHelper);
return S_OK;    }
```

The following details the event dispatch deregistration of the example ATL **_ISametimeHelperEvents** interface implementation:

```
  void FinalRelease()
  {
    //Break the ATL connections for the
//ISametimeHelperEvents dispinterface events.
    HRESULT hr =
      IDispEventSimpleImpl<STHELPER_EVENT_LIVENAME_RESOLVE,
                  CSTHelperExampleEventHandler,
                  &__uuidof(STHelperLib::_ISametimeHelperEvents)>::
                  DispEventUnadvise(pSTHelper);

    hr = IDispEventSimpleImpl<STHELPER_EVENT_CAPABILITY_EVENT,
CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
            DispEventUnadvise(pSTHelper);
    hr = IDispEventSimpleImpl<STHELPER_EVENT_DIRECTORY_RESOLVE,
CSTHelperExampleEventHandler,      &__uuidof(STHelperLib::_ISametimeHelperEvents)>::
            DispEventUnadvise(pSTHelper);

    hr = IDispEventSimpleImpl<STHELPER_EVENT_EVICT_WATCH,
CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
            DispEventUnadvise(pSTHelper);
    hr = IDispEventSimpleImpl<STHELPER_EVENT_SAMETIME_UNAVAILABLE,
CSTHelperExampleEventHandler,
&__uuidof(STHelperLib::_ISametimeHelperEvents)>::
            DispEventUnadvise(pSTHelper);
  }

private:
  //Sametime Helper object.
  STHelperLib::ISametimeHelperPtr pSTHelper;

  // _ISametimeHelperEvents Methods
public:
  STDMETHOD_(void, OnPersonUpdate)(BSTR key, BSTR partnerId,
    BSTR alias, long statusCode, BSTR statusText, BSTR location);
  STDMETHOD_(void, OnCapabilityEvent)(BSTR partnerId,
    BSTR capabilityName, BSTR capabilityValue);
  STDMETHOD_(void, OnDirectoryResolve)(BSTR key, VARIANT maps);
```

```
  STDMETHOD_(void, OnEvictWatch)(BSTR partnerId);
  STDMETHOD_(void, OnSametimeUnavailable)(long * status);
};
```

C++ Application Development Reminders

Key points to remember when developing a Sametime Helper enabled C++ application:

Ensure that IBM Sametime 9.0 Connect Client is installed on the development target machine as identified by ***SAMETIME_HOME***.

Add Sametime Helper Library to your C++ application.

```
      //Import the Sametime Helper Library
      #import "libid:B5C34442-EE1D-4368-B861-6DC78F8EF1D7"
      rename_namespace("STHelperLib") named_guids using
      namespace STHelperLib;

Initialize the COM library prior to using the Sametime Helper Library interface.
      //Initialize COM.
      CoInitialize(NULL);


Create an instance of the SametimeHelper interface.
        //Sametime Helper object.
        STHelperLib::ISametimeHelperPtr pSTHelper;
      //Create the Sametime Helper object.
        HRESULT hr =
            pSTHelper.CreateInstance( __uuidof(STH
            elperLib::SametimeHelper));

Uninitialize the COM library when finished using the Sametime Helper Library
interface.
      //Uninitialize COM.
      CoUninitialize();
```

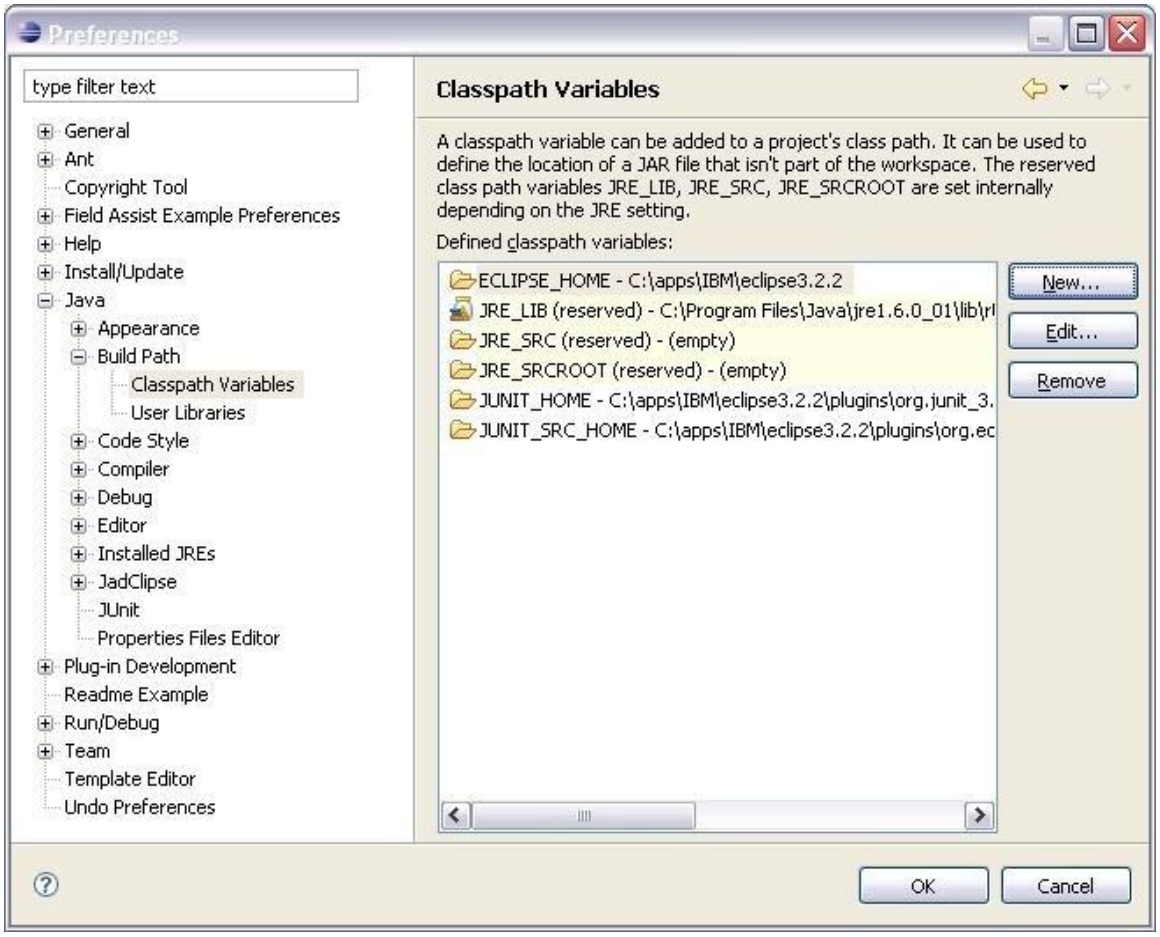
Java Sample Application

The Sametime Helper Toolkit contains a Java sample application, and an Eclipse Sample Project in the "***STHELPER_SDK_HOME*** \samples\java" directory of the Sametime Helper Toolkit.

To get started with the Java Sample Project, sStart Eclipse 3.2 with a development workspace.

Add SAMETIME_HOME as a Java Build Path > Classpath Variable representing the path where the Sametime Connect Client is installed as follows:

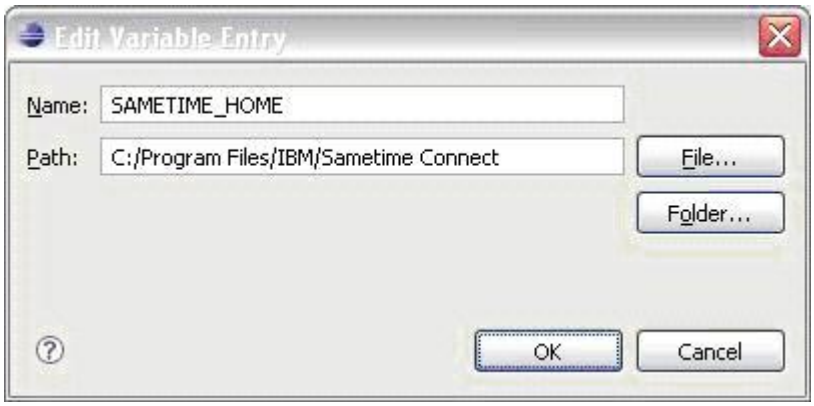
1. Click File > Preferences:



2. Click Java > Build Path > Classpath Variables.

3. Click **New** and add the SAMETIME_HOME variable pointing to the folder where the Sametime Client is installed; for example:

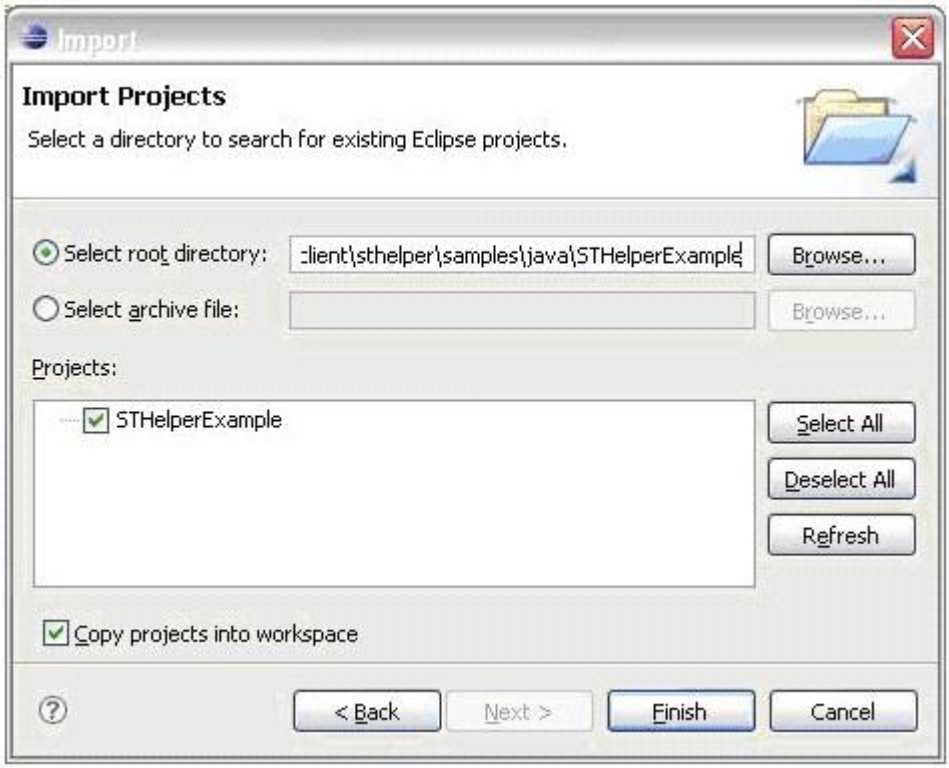
`C:\Program Files\IBM\Sametime Connect`:



Next, import the STHelperExample Java Project into the workspace as follows:

1. Click File > Import and select project type General > Existing Projects into Workspace:

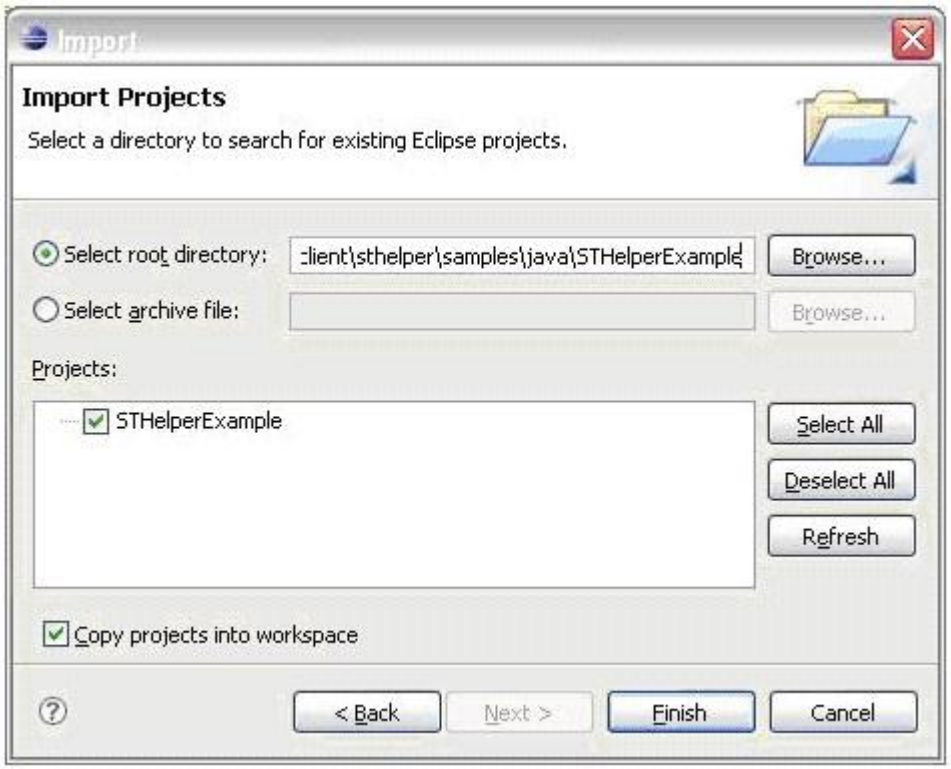
2. In the Import Projects panel, click "Select root directory" and then click **Browse**:



3. Navigate to the appropriate Sametime Helper Toolkit installation directory (STHELPER_SDK_HOME) and select the "STHelperExample" directory. For example:

`C:\st90sdk\client\sthelper\samples\java\STHelper`

4. At the bottom of the Import dialog box, select Copy Projects into workspace checkbox:



5. Finally, click **Finish**.

Eclipse now builds the STHelperExample project. The running Java sample application should look similar to the following picture:

Details of the main initialization of the example application are found in:
STHELPER_SDK_HOME`\java\STHelperExample\src\STHelperExampleDlg.java`

The **`STHelperExampleDlg`** initialize method creates instances of the Sametime Helper and **`STHelperExampleEventHandler`** objects. The **`STHelperEventHandler`** object implements the **`JSTCallback`** interface, and is registered with the Sametime Helper object to processes the event

notification callbacks. The Sametime Helper object's connect method is then invoked to establish the Sametime Helper functionality.

```
/**  * This method performs the
initialization.
 *
 * @return void
 */
private void initialize()
{   this.setSize(433, 297);
  this.setContentPane(getJContentPane());
this.setTitle("STHelperExample");
this.addWindowListener(this);
  this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
this.setVisible(true);

  // Create the Sametime Helper object.   stHelper =
new JSTHelper();   // Create the
STHelperExampleEventHandler to handle
  // the Sametime Helper related events.
stHelperEventHandler = new STHelperExampleEventHandler();
// Add the STHelperExampleEventHandler
  // to the Sametime Helper object.
  stHelper.addSametimeCallback(stHelperEventHandler);
  // Connect the Sametime Helper.
stHelper.connect(); }
```

The **`STHelperExampleDlg`** uninitialize method disconnects the instance the Sametime Helper and removes the **`STHelperExampleEventHandler`** reference, and finally removes the remaining references.

```
/**  * This method performs the
uninitialization.
 *
 * @return void
 */
private void unitialize() {
 // Disconnect the Sametime Helper.
stHelper.disconnect();   // Remove the
STHelperExampleEventHandler
  // from the Sametime Helper object.
stHelper.removeSametimeCallback(stHelperEventHandler);
stHelper = null;
  stHelperEventHandler =
null; }
```

Details of Sametime Helper usage are located in the **`getSametimeContacts()`** method defined in:

`STHELPER_SDK_HOME` `\java\STHelperExample\src\STHelperExampleDlg.java`

The following method details how to use the **`SametimeHelper`** interface to obtain the list of Sametime group names, and the list of contacts located in each group.

```
// Method that uses the IBM Sametime Helper Library to request
// the list of IBM Sametime Connect Client contacts, and populate
// the UI List control with the returned list of contacts.
private void getSametimeContacts() {
  // Sametime Group type
  String strGroupType = "all";
  // Invoke the Sametime Helper to obtain the group names.
```

```
  stHelper.getGroups(strGroupType);

  // Wait for the event data to return the group names.
  String eventData[] = stHelperEventHandler.waitForEventData();

  if (eventData != null) {     // Assign a Vector to
hold the list of contact names     // to be used as the
data for the UI List control.
    Vector contacts = new Vector();

    // Assign the array that has the Sametime Contact Group names.
    String strGroupNames[] = eventData;

    // Loop through the Sametime Contact Group names and retrieve
    // the Sametime Contact names for each group.
for (int i = 0; i < strGroupNames.length; i++) {
      // Assign the current group name.
      String strGroupName = strGroupNames[i];
      // Invoke the Sametime Helper to obtain the contact
      // names for the current group.
      stHelper.getContacts(strGroupName);

      // Wait for the event data to return the contact names.
      eventData = stHelperEventHandler.waitForEventData();

      if (eventData != null) {
        // Assign the current contact.
        String strContactNames[] = eventData;
        // Loop through the Sametime Contact names and add each name
        // to the Vector of contact names.   for (int ct =
      0; ct < strContactNames.length; ct++) {
          contacts.add(strContactNames[ct]);
        }
      }
    }
    // Set the UI List control data with the Vector
    // of contact names.
getJList().setListData(contacts);
  }
}
```

Details of the Sametime Helper example **`JSTCallback`** interface implementation are found in:

`STHELPER_SDK_HOME``\java\STHelperExample\src\STHelperExampleEventHandler.java`

The **`STHelperExampleEventHandler`** object is an implementation of the **`JSTCallback`** interface, which registers for notification and processes the Sametime Helper event notification callbacks.

```
/**
*  This class implements the JSTCallback interface in order  * to
   process Sametime Helper events.
 */
public class STHelperExampleEventHandler implements JSTCallback {
  //The current event related data.
private String eventData[] = null;

  /**
```

```
*  Convenience method which waits for up the specified max_time
   *  in milliseconds for an event to be received from one of the
   *  Sametime Helper JSTCallBack methods.
   *
*  @param max_time the number of milliseconds to wait for an event
*  @return the current event related data
   */
  public String[] waitForEventData(long max_time)
{     String data[] = null;

  try { synchronized (this)
      {   wait(max_time);   if
      (eventData != null)
      {     data = eventData;
      eventData = null;
        }
      }
    } catch (InterruptedException e) {
      e.printStackTrace();
    }
    return
data;   }

  /**
*            Result for one of the keys sent to a liveNameResolve request
   *
*            @param key
*            the key which resolved to the current results
*            @param partnerInfo
*            Map of name-value pairs, see getContacts
   */
  public void onPersonUpdate(String key, Map partnerInfo) {
   JOptionPane.showMessageDialog(null, "onPersonUpdate", null,
                        JOptionPane.PLAIN_MESSAGE);
  }

  /**
*            Update the information on a partner who has a watch established
   *
*            @param partnerInfo
*            contains name-value pairs as detailed in getContacts
   */
  public void onChange(Map partnerInfo) {
    JOptionPane.showMessageDialog(null, "onChange", null,
                        JOptionPane.PLAIN_MESSAGE);
}

  /**
*            Results for one request to directoryResolve
   *
*            @param key
*            the key which produced these results
*            @param results
*            an array of Maps, each in turn a set of
*            name-value pairs for
*            users who matched the search key
   */
```

```
  public void onDirectoryResolve(String key, Map[] results) {
    JOptionPane.showMessageDialog(null, "onDirectoryResolve", null,
                        JOptionPane.PLAIN_MESSAGE);
}

  /**
*             Results for one request to getContacts
   *
*             @param groupName
*             the name of the groups getting contacts for
*             @param results
*             an array of Strings, one for each contact
*                */
  public void onGetContacts(String groupName, String[] results) {
    synchronized (this)
{       eventData = results;
notify();
    }
  }

  /**
*             Results for one request to getGroups
   *
*             @param type
*             the type of group being requested
*             ("all", "public", "private")
*             @param results
*             an array of Strings, one for each map
   */
  public void onGetGroups(String type, String[] results) {
    synchronized (this)
{       eventData = results;
notify();
    }
  }
  /**
*             User logged out
   */
  public void onLogout() {
    JOptionPane.showMessageDialog(null, "onLogout", null,
                        JOptionPane.PLAIN_MESSAGE);
}

  /**
*             Call from STHelper indicating that Sametime functions
*             are available
   *
*             @param statusCode
*             indicates first connect versus restored availability
   */
  public void onSametimeAvailable(int code) {
    JOptionPane.showMessageDialog(null, "onSametimeAvailable", null,
                        JOptionPane.PLAIN_MESSAGE);
}

  /**
*             Call from STHelper indicating that Sametime functions
```

```
*             are unavailable
   */
  public void onSametimeUnavailable(int code) {
    JOptionPane.showMessageDialog(null, "onSametimeUnavailable", null,
                        JOptionPane.PLAIN_MESSAGE);
}

  /**
*             Call from STHelper for messages not fitting into one of the above
*             categories
   *
*             @param messageType
*             the message type
*             @param typeVersion
*             the type version
*             @param map
*             the message argument map
   */
  public void onMessage(String messageType,
String typeVersion,
                        Map map) {
    JOptionPane.showMessageDialog(null, "onMessage", null,
                        JOptionPane.PLAIN_MESSAGE);
  }
}
```

Java Application Development Reminders

Key points to remember when developing a Sametime Helper enabled Java application:

Ensure that IBM Sametime Connect Client is installed on the development target machine as identified by ***SAMETIME_HOME***.

Ensure that the following Sametime Helper libraries are in the classpath of development and runtime environments:

```
SAMETIME_HOME/shared/eclipse/plugins/
      com.ibm.collaboration.realtime.jsthelper_XXX.jar"
SAMETIME_HOME/rcp/eclipse/plugins/com.ibm.micro.utils_XXX.jar"
SAMETIME_HOME/rcp/eclipse/plugins/com.ibm.mqttclient_XXX.jar"
SAMETIME_HOME/rcp/eclipse/plugins/org.eclipse.osgi_3.4.3.XXX.jar"
```

Where **XXX** is the version of the library that is installed by the IBM IBM Sametime Connect Client.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any nonIBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
5 Technology Park Drive

Westford Technology Park
Westford, MA 01886

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only. All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp.
Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

These terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM

AIX

DB2

DB2 Universal Database Domino

Domino

Domino Designer

Domino Directory

i5/OS iSeries

Lotus

Notes

OS/400

Sametime

System i

WebSphere

AOL is a registered trademark of AOL LLC in the United States, other countries, or both.

AOL Instant Messenger is a trademark of AOL LLC in the United States, other countries, or both.

Google Talk is a trademark of Google, Inc, in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Microsoft, and Windows are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.