Sametime
Version 9.0

Sametime 9.0 Software Development Kit

Client Telephony API Guide

IBM

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

IBM® Sametime® Connect 7.5 was the first release of new real-time collaboration technology built on the Eclipse™-based Lotus Expeditor rich client platform. Sametime Connect 7.5 and later releases leverage the Eclipse plug-in framework to provide developers, like you, with extensibility features that go far beyond those available in previous Sametime Connect releases. In addition, the new Sametime Connect rich client is available as both a stand-alone application and an embedded version for any Lotus Expeditor application, including IBM Notes® 8.0 and later releases.

Sametime Connect 7.5 and 7.5.1 introduced new audio and video capabilities, plus access to telephony services provided by third-party telephony partners. Sametime Connect accesses telephony and audio/video services using a combination of client and server programming interfaces:

- **Client Telephony API:** A set of client APIs (application programming interfaces) that provide client applications with access to telephony and audio/video services, and SPIs (service provider interfaces) that enable service providers to provide implementations of those services.

- **TCSPI (Telephony Conferencing Service Provider Interface):** A server SPI that enables both IBM® Sametime and telephony partners to provide telephony and audio/video services to Sametime Connect and Sametime online meetings.

This document provides an overview of the Sametime Connect 9.0 client telephony API. To find additional information that you'll need to develop Sametime Connect client plug-ins or TCSPI adapters (server applications), see the *References* section at the end of this document. Unless otherwise noted, the information in this document applies to both the standalone and embedded versions of the Sametime Connect rich client.

Changes for Sametime 8.5.2 , 8.5.2 IFR 1 and 9.0

Sametime 8.5 introduced a number of new features and API changes of interest to telephony and audio/video application developers and service providers, for example:

- Client audio and video services are now provided by Sametime server components, rather than being peer-to-peer services.
- Telephony partners now provide both telephony and audio/video services via the TCSPI, rather than using creating client plug-ins to do so. Note that Sametime audio/video conferencing services are also provided via the TCSPI.
- The client includes new meeting plug-ins that provide rich client access to Sametime online meetings, including telephony and audio/video features.

Release 8.5.2 added some significant new telephony and audio/video features:

- The client now supports multiple third-party TCSPI adapters deployed in the same community. The user can select one service provider for audio calls and a different provider for video calls.

- Users who attend online meetings from a web browser can now participate in audio/video calls, along with rich client users. Note that the client telephony API is only applicable to the rich client, not the web.
- IBM® Sametime Unified Telephony Lite (SUT Lite) is a new offering that enables the Sametime Computer softphone to make and receive telephone calls via a SIP trunk. This feature is also included in the full Sametime Unified Telephony offering for 8.5.2.

Release 8.5.2 also includes a number of noteworthy API changes:

- The call service provider interfaces, which we started phasing out in release 8.5.1, have been deprecated and are no longer documented here. Since third-party telephony and audio/video services are now provided via the TCSPI, the call service provider interfaces are no longer needed or supported for external use.
- The audio/video and phone device controller interfaces have been deprecated and are no longer documented here. These interfaces were used by few (if any) customers and partners, so they will be maintained for internal use only. Note that the device interfaces used for headset integration were not deprecated and are still supported for external use.
- A new telephone number service has been added to allow customers to control retrieval and formatting of telephone numbers. See *Chapter 5. Using the Telephone Number Service* for more information.
- New service provider interfaces have been added to allow partners to create client plug-ins that evaluate audio/video media quality. See the next chapter for more information.

Release 8.5.2 IFR 1 includes some additional changes:

- The **callAction** extension point, which enables you to add actions to the call window, has been enhanced. You can now add menu items to the transfer call action in the call window, and to the Accept Options and Decline Options menus in the incoming call window.
- There's a new **participantDisplayInformation** extension point that allows you to update the participant image and display name in the call window.
- There are new sample plug-ins for both the callAction and participantDisplayInformation extension points.

Release 9.0 includes some additional features such as :

- Users can now access the list of video layouts allowed using the Call**.getAllowedVideoLayouts** API . They can then set the desired layout as per user requirement using the Call.**setVideoLayout()** API .
- Users can also retrieve additional Participant object properties using new API provided in 9.0 such as mutelock/unlock status.
- New properties have been added for MediaEvent and CallEvent for the new features provided in Sametime 9.0 such as Lecture Style meetings and Multiple-Moderator features which are detailed in Chapter 2. Client Telephony API Overview.

The client section of the Telephony Toolkit provides a complete list of API changes for this release. For more information on API changes and sample plug-ins, see the *References* section at the end of this document.

Notes:

1. Some screen shots in this document might not have been updated to reflect user interface changes for this release.

Chapter 2. Client Telephony API Overview

The IBM® Sametime Connect client telephony API provides both application programming interfaces (APIs) and service provider interfaces (SPIs) for the Sametime Connect telephony and audio/video features.

The APIs provide applications with access to the Sametime Connect telephony and audio/video features. The APIs are used by the Sametime Connect user interface, and can be used by third-party plug-ins for the Sametime Connect rich client.

The SPIs are used to provide the implementation of Sametime Connect telephony and audio/video services. Sametime Connect includes client plug-ins that implement these SPIs and provide access to the following services:

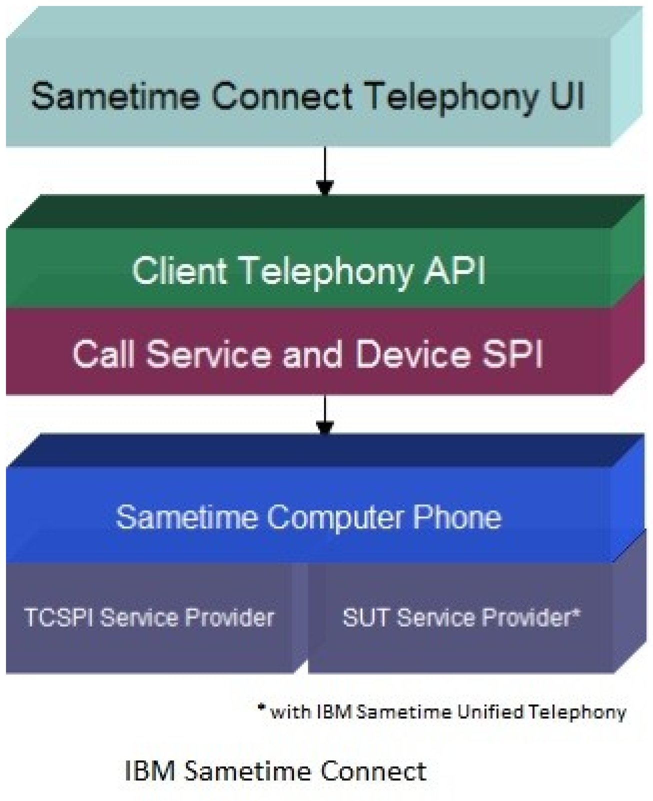
- Telephony and audio/video services provided by TCSPI (Telephony Conferencing Service Provider Interface) adapters, which are server applications that implement the TCSPI. This includes both third-party telephony and audio/video services and Sametime Audio/Video Conferencing.
- Sametime Unified Telephony (SUT) services, which are not currently implemented using the TCSPI for technical reasons. Note that these services are not used for SUT Lite, which uses the Sametime Audio/Video Conferencing TCSPI service.
- Sametime Computer Phone services, which are used for audio/video calls and softphone telephone calls, as well as audio/video device management. Note that Sametime softphone telephone calls are only possible with SUT or SUT Lite.

In Sametime Connect releases prior to 8.5, the only way to provide third-party audio/video services to the client was to create client plug-ins that implement the client telephony service provider interfaces. Starting with Sametime 8.5, the TCSPI became the only supported way to add third-party audio/video and telephony services to Sametime Connect. However, service providers that wish to customize or extend the Sametime Connect user interface still need to create client plug-ins to do so.

The TCSPI is described in the detail in the Telephony Toolkit. For more information, see the *References* section at the end of this document.

Client Telephony API Architecture

The following figure depicts the relationship between the IBM® Sametime Connect telephony user interface, client telephony API, and client telephony service plug-ins.:



Here's an explanation of this relationship in more detail, starting at the top of the figure:

- The Sametime Connect user interface layer accesses telephony and audio/video services using the client telephony API application interfaces. The UI calls client telephony API application methods to request services, and receives telephony API events that signal incoming calls and status changes for existing calls and call participants.
- The client telephony API application layer accesses telephony and audio/video services using the telephony API service provider interfaces, which are implemented by client plug-ins that provide these services. The service provider layer includes SPIs for both calling services and audio/video device management.
- The Sametime Computer Phone client plug-in provides an implementation of the call service and device management SPIs. The Sametime Computer Phone implementation delegates to the appropriate call service implementation, either the TCSPI or SUT call service provider, depending on the type of call.

Client Telephony API Package Summary

The IBM® Sametime Software Developers Kit (SDK) includes Javadoc that provides details on all the interfaces and classes in the Sametime Connect client telephony API. In order to use the application or service provider interfaces, you should understand how the Java code is organized.

The client telephony API consists of the following Java packages:

Table listing Java packages contained in the client telephony API.

Java package	Description
com.ibm.collaboration.realtime.telephony	General application interfaces.
com.ibm.collaboration.realtime.telephony.device	Application interfaces for telephony device controllers. *Note: Many of these interfaces have been deprecated since Release 8.5.2.*
com.ibm.collaboration.realtime.telephony.endpoint	Application interfaces for telephony endpoint management.
com.ibm.collaboration.realtime.telephony.event	Application interfaces for call events.
com.ibm.collaboration.realtime.telephony.exception	Exception classes.
com.ibm.collaboration.realtime.telephony.media	Media quality service provider interfaces for third-party audio/video media quality evaluation implementations.
com.ibm.collaboration.realtime.telephony.number	Telephone number service interface and abstract base class for telephone number service implementations.
com.ibm.collaboration.realtime.telephony.service	General service provider interfaces and abstract base classes for telephony service implementations. *Note: All of these interfaces have been deprecated since Release 8.5.2.*
com.ibm.collaboration.realtime.telephony.service.device	Abstract base classes for device controller implementations. *Note: All of these interfaces have been deprecated since Release 8.5.2.*
com.ibm.collaboration.realtime.telephony.ui.extension	Base class used by action classes for the call window toolbar.
com.ibm.collaboration.realtime.telephony.ui.menu.extension	Interface used by action classes for call options menu items.

Key Classes and Interfaces

This section describes some of the key client telephony API classes and interfaces used by both client applications and service providers.

com.ibm.collaboration.realtime.telephony package:

- **CallManager** provides methods that the application uses to start and join calls and to access telephony services.
- **CallFactory** creates instances of the client telephony interfaces in this package.

- **Call** represents a telephone, audio, or video call.
- **CallListener** listens for events that signal incoming calls and call status changes, and **AbstractCallListener** is the class extended by CallListener implementations.
- **Participant** represents a call participant.
- **Address** identifies a participant to be invited to a call.
- **Capability** defines the set of telephony capabilities that might be supported by a service provider, and is used by the application to determine which capabilities are available. **CapabilityManager** provides services to access capabilities for a specific call.
- **EndpointDescriptor** represents a telephony endpoint that can be selected as a preferred device in the user interface. A telephony endpoint can be a telephone or computer device, and call service provider implementations can define their own endpoints that will appear in the UI.
- **IncomingCallHandler** is a handler for incoming calls, which can be used by client plug-ins to override the default incoming call handling.
- **TelephonyUtil** provides general utility methods that aren't related to specific calls.
- **PropertyNames** defines the names of telephony API properties that are passed to or from the service provider. The PROPERTY_TCSPI_xxx properties correspond to Sametime Telephony Conferencing SPI configuration options. (New properties have been added to this class with 9.0)

com.ibm.collaboration.realtime.telephony.device package:

- **DeviceEvent** is used to signal status changes for devices, much like CallEvent (see below) signals status changes for calls.
- **DeviceFactory** creates instances of the device interfaces in this package.
- **DeviceListener** listens for device events, and **AbstractDeviceListener** is the class extended by DeviceListener implementations.
- **DeviceNotifier**, which is extended by DeviceController, provides methods for registering DeviceListener implementations, and **AbstractDeviceNotifier** is the class extended by DeviceNotifier implementations. *(Note: AbstractDeviceNotifier was moved to this package from com.ibm.collaboration.realtime.telephony.service since 8.5.2.)*
- **HeadsetNotifier** is a DeviceNotifier designed for client plug-ins that monitor a headset device. The HeadsetNotifier implementation generates headset device events when the user operates headset controls, for example, to adjust volume or mute the microphone; the IBM® Sametime Connect user interface responds to these events as if the user performed the same operation in the UI.
- **MediaPlayer** provides methods to play audio and video files, either during audio/video calls or separately.

com.ibm.collaboration.realtime.telephony.endpoint package:

- **EndpointManager** provides methods to add, remove, set and list preferred telephony endpoints.

com.ibm.collaboration.realtime.telephony.event package:

- **CallEvent** and its subclasses define call events that signal incoming calls and status changes for existing calls and call participants. Applications register event listeners to handle call events. The service provider client implementation is responsible for generating the appropriate events.
- **CallEventFactory** creates instances of call events interfaces.

com.ibm.collaboration.realtime.telephony.exception package:

- **TelephonyException**, and its subclass **TelephonyServiceException**, are the base classes for most exception classes in this package. TelephonyException extends java.lang.Exception.
- **TelephonyRuntimeException**, which extends java.lang.RuntimeException, is the base class for classes in this package that represent runtime (unchecked) exceptions.

com.ibm.collaboration.realtime.telephony.media package:

- **MediaQualityServiceProvider** is the interface implemented by third-party client plug-ins that provide audio/video media quality evaluation services, and **AbstractMediaQualityServiceProvider** is the class extended by MediaQualityServiceProvider implementations.
- **MediaEvent** is used to signal audio/video media events that occur on a call.
- **MediaEventListener** listens for media events related to a call, and **MediaEventAdapter** is the class extended by MediaEventListener implementations.
- **LocalMediaEventListener** listens for media events related to the local microphone and speaker devices, and **LocalMediaEventAdapter** is the class extended by LocalMediaEventListener implementations.
- **MediaEventManager** is used to register MediaEventListener and LocalMediaEventListener classes.

com.ibm.collaboration.realtime.telephony.number package:

- **TelephoneNumberService** is the interface implemented by customers wishing to override the retrieval or formatting of telephone numbers, and **AbstractTelephoneNumberService** is the class extended by TelephoneNumberService implementations.

List of Properties Added for Sametime 9.0

1. **PropertyNames :** Following properties have been newly added in com.ibm.collaboration.realtime.telephony.PropertyNames:

Property Name	***Summary***
PROPERTY_TCSPI_SERVICE_LOCATION	Holder for selected TCSPI service location
PROPERTY_TCSPI_MUTE_LOCK_SUPPORTED	TCSPI property indicating whether the service provider supports the ability to mute with lock all participants who join a call.
PROPERTY_TCSPI_PARTICIPANT_DISPLAY_NAME	Participant's new display name. Used for TCSPI rename operation. TCSPI service fires a property change on this property once rename is executed.
PROPERTY_TCSPI_PARTICIPANT_ASSOCIATED_ID	Id of the external participant who is associated with a Sametime participant. Used for TCSPI associate operation. TCSPI service fires a property change of this property on the Sametime participant once associate is executed.
PROPERTY_IS_MODERATOR	Property to hold the isModerator value for a participant to support Multi Moderator
PROPERTY_IS_ROOM_OWNER	Property to hold the isModerator (only if it is a room owner too) value for a participant to support Multi Moderator
PROPERTY_IS_PRESENTER	Property to hold the Presenter value for a participant to support

Property Name	***Summary***
	Multi Moderator
PROPERTY_CONFERENCE_URI	Property to hold the conference URL by which an external participant can also join
PROPERTY_TCSPI_TELEPHONY_ENABLED_INTERNAL	Property to hold actual value of telephonyEnabled value in case of internal service provider if it is overridden when SUTLite is enabled.
PROPERTY_TCSPI_AUTO_MUTE_SUPPORTED	TCSPI property indicating whether the service provider supports the ability to automatically mute all participants who join a call.

2. New events have been added to **com.ibm.collaboration.realtime.telephony.event.CallEvent** which represents a call-related telephony event , callbacks for which are needed to perform the necessary processing logic for each event.

Event Name	***Summary***
EVENT_PARTICIPANT_UNMUTE_REQUESTED	Generated when a participant requests to be unmuted
EVENT_PARTICIPANT_ASSOCIATED_ID	Generated when a Sametime participant is associated with an external participant.
EVENT_PARTICIPANT_MUTE_LOCKED	Generated when a Sametime participant joins a lecture style meeting
EVENT_PARTICIPANT_MUTE_UNLOCKED	Generated when a Sametime participant joins the lecture style meeting
EVENT_PARTICIPANT_UNMUTE_REQUEST_CANCELLED	Generated when a participant requests to cancel unmute in a lecture style meeting

3. New property codes have been added to the **com.ibm.collaboration.realtime.telephony.media.MediaEvent** which defines Telephony Media Engine notifications. These are generated when the processing of video packets has overflowed depending on the related scenario:

Property Codes	***Value***
SIP_REGISTRATION_FAILURE	6
ICE_ALLOCATION_FAILURE	7
TURN_AUTHENTICATION_CONF_FAILURE	8
TURN_UNAVILABLE_HTTP_FAILURE	9
TURN_UNAVILABLE_FAILURE	10
TURN_SERVER_SETTING_FAILURE	11
TURN_FAILED_DOWN_FAILURE	12
TLS_CERT_AVAILABLE_FAILURE	13
TLS_SELF_SIGNED_CERT_FAILURE	14
TLS_HOSTNAME_FAILURE	15
INVALID_PROPERTIES_FAILURE	16
CAMERA_NOT_SUPPORTED	17

Property Codes	***Value***
INVALID_CAMERA_DEVICE	18
INVALID_MICROPHONE_DEVICE	19
INVALID_SPEAKER_DEVICE	20
AUDIO_DEVICE_FAILURE	21
START_VIDEO_FAILURE	22
SIP_CALL_FAILURE	23

Chapter 3. Using the Client Telephony API

This chapter provides some guidance for using the IBM® Sametime Connect client telephony API.

Creating Object Instances

Most of the client telephony APIs are interfaces rather than classes. Interfaces are used wherever possible, because they separate the API from its implementation, and they make it easier to evolve the API without breaking existing applications.

Because interfaces are used, API objects cannot be instantiated directly. Some API object instances can be created using API factory classes; for example, applications create a CallManager object to access calls, and service providers create CallEvent objects to let applications know about call status changes. Other API objects are created by internal IBM implementation code, and can only be obtained using API methods; for example, Call objects are created by the implementation of CallManager methods.

To create objects using a factory, use the factory class in the same package that includes the interfaces. Use the factory `getInstance()` method to get a factory instance, then use the appropriate `create…` method to create an instance of the desired interface. For example:

```
import com.ibm.collaboration.realtime.telephony.CallFactory;
import com.ibm.collaboration.realtime.telephony.CallManager;
import com.ibm.collaboration.realtime.telephony.event.CallEvent;
import com.ibm.collaboration.realtime.telephony.event.CallEventFactory;

// Create a CallManager instance.
CallManager callManager = CallFactory.getInstance().createCallManager();

// Create a CallEvent instance.
CallEvent event = CallEventFactory.getInstance().createCallEvent();
```

Starting a Call

Client application plug-ins can start a telephone or audio/video call in one of two ways:

1. Using the `CallManager.startCall` method. This provides the application with complete programmatic control over the call and the associated call events, but the IBM® Sametime Connect user interface is unaware of the call and will not display a call status window. This method is appropriate for applications that provide their own UI for the call (including Sametime Connect), and is intended primarily for internal use.
2. Using one of the `CallManager.startUiCall` methods. These methods trigger the Sametime Connect UI to start a call as if the user had selected an action to do so, displaying the call status window. This gives control of the call to the user, rather than the application that started the call. This method is recommended for third-party applications.

To obtain a CallManager instance, use CallFactory as shown in the previous section.

When using `CallManager.startCall`, you need to provide both of the following:

- A list of Address objects each call participant -- use CallFactory to create Address objects. The list should include an Address for the caller as the first entry, so that the service provider knows how to connect the caller.
- A CallOptions object (created using CallFactory) specifying options and properties needed to set up the call. See the CallOptions API Javadoc in the Telephony Toolkit for details on which properties are required; for more information, see the *References* section at the end of this document.

When using `CallManager.startUiCall`, you only need to provide a telephone number string, or the following parameters:

- A list of participants specified as either an array of Person objects or an array of participant ID strings (it is not necessary to include the caller). The participant ID is the Sametime user ID. When logged in to multiple communities, it's better to use Person objects, because unlike participant IDs, they include the community ID. For details on using Person objects, see the Sametime Connect API Javadoc and other Sametime Connect Toolkit documentation; for more information, see the *References* section at the end of this document.
- An optional call type, one of the these values defined in CallManager: `CALL_TYPE_TELEPHONE`, `CALL_TYPE_AUDIO`, or `CALL_TYPE_VIDEO`. By default, the call type is determined by the user's preferred device and service provider preferences, so you should only specify a call type when necessary.
- An optional TCSPI provider string parameter, which gets passed to the TCSPI implementation via the com.ibm.telephony.conferencing.property.ProviderData property in the TCSPI conference document. This parameter is for third-party use and is ignored by Sametime Connect. The call type must be specified in order to use this parameter.

Note that the user's ability to make telephone, audio, or video calls is determined by the availability of the service and policy settings, as well as the Sametime status and policy settings for other call participants. Regardless of the method used to start a call, your application must allow for the possibility that the call might fail for any number of reasons.

Working with Call and Participant Objects

Once a call has been started, the Call and Participant objects are used to perform operations on the call and participants and to monitor their status. Note that you cannot access Call or Participant objects for calls started using `CallManager.startUiCall`.

The CallManager methods `startCall`, `acceptIncomingCall`, and `connectIncomingCall` return a Call object when starting or joining a call. You can also use `createCall` to create a Call object prior to starting or joining a call, and use that object to start or join the call. The Call object provides access to information about call participants and call status, as well as methods to add or remove participants and perform call operations such as muting, pausing, or ending the call.

Starting with Sametime 9.0, following new API are available for the Call object:

- Users can now access the list of video layouts allowed using the Call.getAllowedVideoLayouts API . They can then set the desired layout as per user requirement using the Call.setVideoLayout() API .
- Property "PROPERTY_STATUS" has been added which is used to fire a java.beans.PropertyChangeEvent to indicate that the call status has changed.
- The new API isMissedCall() determines if the call is a missed call. A call is considered missed under the following conditions:
  1. An EVENT_INCOMING_CALL_INVALID event is received for this call.
  2. An EVENT_CALL_TERMINATED event is received for this call and the call is not already connected.
- API dockUndockCallWindow(Canvas) and aboutToDockUndockCallWindow() have been added which are used to provide the internal logic to attach and detach the video streams while docking and undocking the video window in meetings.

Similarly, the Participant object provides access to information and status for a call participant, as well as methods to perform participant operations such as muting and volume adjustment. To get Participant objects for all call participants, use `Call.getParticipants()`. To get a Participant object for a specific participant, use `Call.getParticipant(String id)`.

With Sametime 9.0, a few new API have been added for the **Participant** object which include:

- requestUnmute(boolean): Sends a request to the call moderator to unmute this participant, if supported by the service provider. This alerts the moderator that the participant wishes to be unmuted, but does not unmute the participant.
- isMuteLocked(): Determines if this participant is mute locked, meaning his/her audio is muted such that he/she cannot unmute himself/herself.
- muteLocked(): Mutes this participant's audio and prevents the participant from unmuting himself, if supported by the service provider. The participant remains muted until one of the unmute methods is used to unmute him.
- muteUnLocked() : unlocks the Mutelock on this participant's audio if supported by the service provider.
- silenceAudio(): used to mute audio only for a moderator in meetings
- unSilenceAudio() :used to unmute audio only for a moderator in meetings
- isUnmuteRequested() : used by participants in lecture style meetings when they request to speak

The Capability Interface

The Capability interface allows service providers to declare which capabilities they support. Applications can query these capabilities in order to enable supported features and disable unsupported features.

The Capability interface consists of a `CAPABILITY_xxx` string constant for each capability, and an `isCapabilitySupported(String capability)` method that returns `true` if the specified capability is supported. The Capability interface is implemented by each of the following interfaces: Call, CallManager, CapabilityManager, and Participant.

To determine which capabilities a service provider supports in general, use `CallManager.isCapabilitySupported`. To determine which capabilities are available for a given call, use either `Call.isCapabilitySupported` or `Participant.isCapabilitySupported` (these are equivalent). The capabilities for a call should be the either the same as, or a subset of, the general service provider capabilities. Bear in mind that when multiple service providers are available, `CallManager.isCapabilitySupported` determines if *any* service provider supports the specified capability.

Starting with 8.5.2, you can also monitor capability changes for a given call using a CapabilityManager, which you get using `Call.getCapabilityManager,` or by listening for CapabilityEvent, a new type of CallEvent. CapabilityManager allows you to register property change listeners for capability changes, which is useful for user interface code that enables or disables controls based on capabilities. In addition, CapabilityManager allows you to query all currently supported capabilities, either as a list of capabilities or a series of CapabilityEvents.

Applications should assume that capabilities can change dynamically, based on user preferences or runtime conditions. This is true for both general service provider capabilities and call-specific capabilities. Therefore, applications should not save the result of an `isCapabilitySupported` method call for later use, since the result can change at any time.

There are a few key capabilities worth noting here:

- `CAPABILITY_TELEPHONY`: This capability indicates that one or more service providers supports telephone calls, meaning calls that can include telephone number endpoints. This capability is supported only when telephony services are provided by IBM® Sametime Unified Telephony (SUT), SUT Lite, or a third-party service provider.
- `CAPABILITY_AUDIO`: This capability indicates that one or more service provider supports computer audio calls (a.k.a. voice chat). Sametime includes built-in support for computer audio (and video) calls via the Sametime Audio/Video Conferencing service; third-party service providers might also support computer audio, typically in addition to telephony services.
- `CAPABILITY_VIDEO`: This capability indicates that one or more service provider supports video calls. Note that SUT, SUT Lite, and some third-party service providers support video in conjunction with telephone calls.
- With 9.0, a new capability `CAPABILITY_ANSWER_CALL` has been added which determines whether an Active call can support answering the call with the current endpoint.

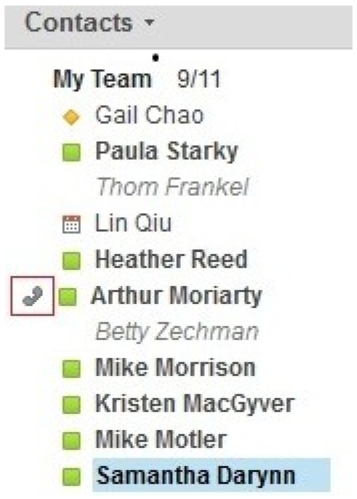
Chapter 4. Extending the IBM Sametime Connect User Interface

The Eclipse platform provides a number of standard ways to extend an application's user interface, for example, by adding items to menus, actions to toolbars, and preference pages. As with any Eclipse application, these standard extensions can be used to extend the IBM® Sametime Connect user interface.

The Sametime Connect Toolkit includes documentation that explains in detail how to use Eclipse extension points to extend Sametime Connect; see the *References* section at the end of this document for more information. This chapter highlights some telephony-related UI extensions that are not documented elsewhere.

Telephony Status

IBM® Sametime Connect includes extended status and telephony status plug-ins that can be enabled to display telephony status for Sametime contacts, as in the contact list snippet below (the telephone icon indicates the contact is on the phone):



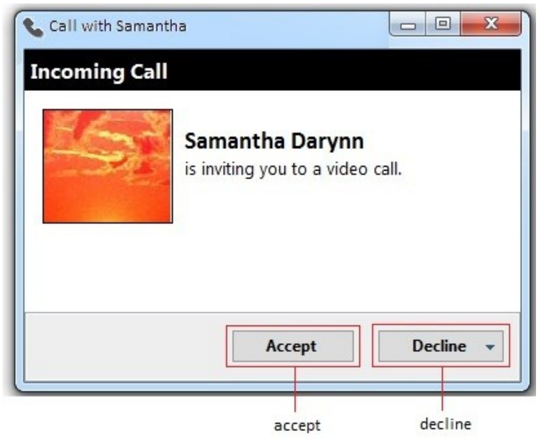
The Sametime server obtains telephony status from the Sametime Telephony Presence Adapter and publishes status changes via user attributes, which the client plug-ins respond to. The Telephony Presence Adapter is used to integrate with Sametime Unified Telephony (SUT), as well as third-party telephony presence solutions.

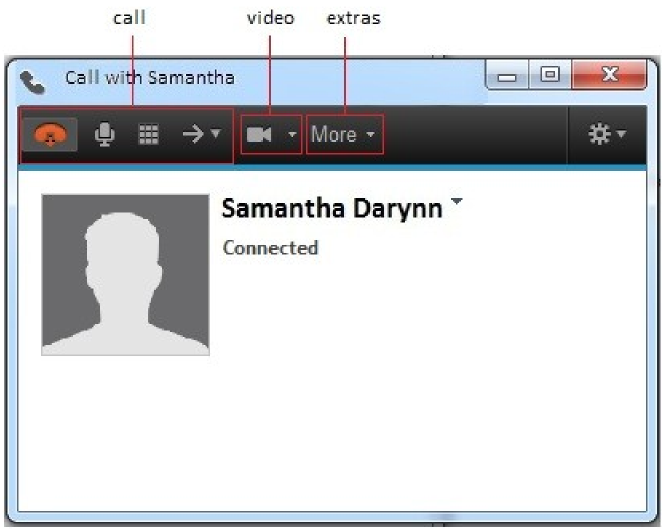
The client plug-ins are disabled by default, unless Sametime Unified Telephony (SUT) features have been enabled. However, administrators can enable these plug-ins for environments where a mix of SUT and non-SUT clients is used, to allow non-SUT users to see telephony status for SUT users, or for environments where a third-party telephony presence solution is used.

The telephony status plug-in is described in more detail in the Sametime Connect Toolkit in the SDK, and the Telephony Presence Adapter is described in the Community Server Toolkit. The process for enabling the telephony status plug-ins is documented in the Sametime Information Center. For more information, see the *References* section at the end of this document.

Adding Actions to the Call Window

You can add your own actions to the call window using the **callAction** extension point. The extensible areas of the call window are shown below:





The extension point ID is com.ibm.collaboration.realtime.telephony.ui.callAction, with the following attributes:

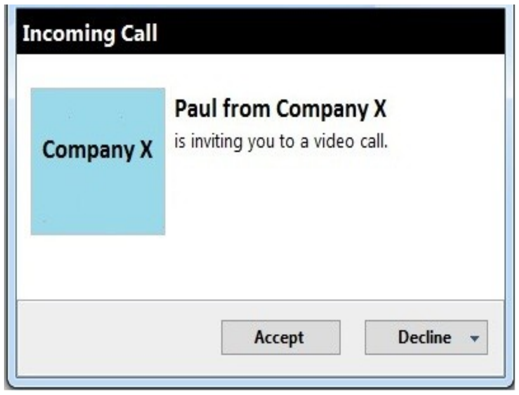
Table detailing the com.ibm.collaboration.realtime.telephony.ui.callAction attributes.

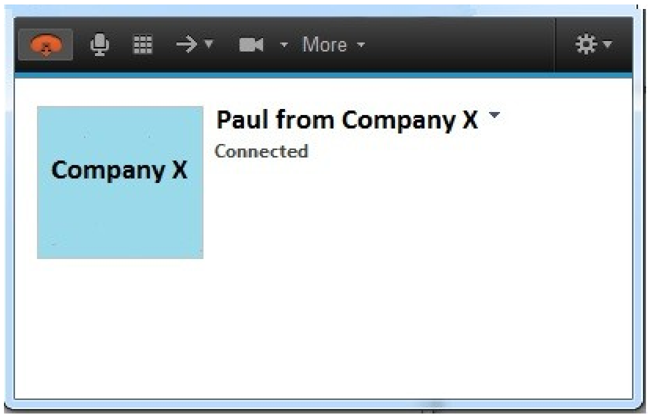
Attribute	**Description and values**
`id`	Unique identifier for this action.
`class`	Action class that extends the com.ibm.collaboration.realtime.telephony.ui.extension.CallAction class.
`displayName`	Action label text.
`image`	The image displayed for this action, used only when the path attribute is `call`. If no image is specified, the `displayName` is used.
`disabledImage`	The image displayed for this action when disabled. If you specify an image but no disabled image, Eclipse will generate a disabled version of your image for you, but it might be of lower quality than an image you supply.
`tooltipText`	Action tooltip text.
`hoverImage`	The image displayed when hovering over this action.
`path`	Toolbar or action path, one of the following: Incoming call window: ● `accept` puts the action on the Accept Options drop-down menu ● `decline` puts the action on the Decline Options drop-down menu Call window: ● `call` puts the action on the call window toolbar ● `forward` puts the action on the transfer call action drop-down menu ● `video` puts the action on the Video drop-down menu ● `extras` puts the action in the More drop-down menu (this is the default if no path is specified)

See the sample callAction plug-in in the client section of the Telephony Toolkit for more information.

Customizing the Call Window Participant Image and Display Name

In a call window, you can customize the participant image and display name using the **participantDisplayInformation** extension point, as shown below:





The extension point ID is com.ibm.collaboration.realtime.telephony.ui.participantDisplayInformation, with the following attributes:

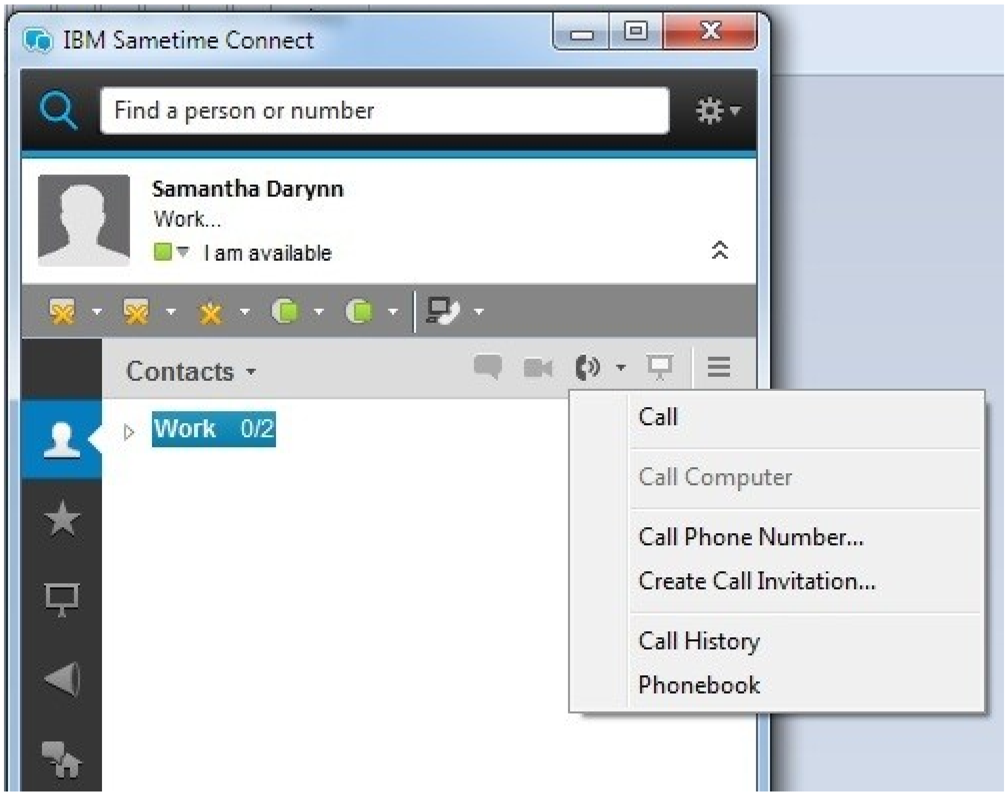
Table detailing the com.ibm.collaboration.realtime.telephony.ui.participantDisplayInformation attributes.

Attribute	Description and values
`id`	Unique identifier for this action.
`class`	Action class that implements the com.ibm.collaboration.realtime.telephony.ui.extension.ParticipantDisplayInformation interface.

See the sample participantDisplayInformation plug-in in the client section of the Telephony Toolkit for more information.

Adding Menu Items to the Call Options Menu

You can add your own menu items to the call options action icon shown below, using the **callMenuAction** extension point.



In the screenshot, you can see the IBM® Sametime Unified Telephony Call History and Sametime Phonebook menu items, which are added using this extension point.

The extension point ID is com.ibm.collaboration.realtime.telephony.ui.callMenuAction, with the following attributes:

Table detailing com.ibm.collaboration.realtime.telephony.ui.callMenuAction attributes.

Attribute	**Description and values**
`id`	Unique identifier for this action.
`class`	Action class that implements the com.ibm.collaboration.realtime.telephony.ui.menu.extension.CallMenuActionRunnable interface.
`image`	Optional: The image displayed for this action.
`displayName`	Menu item text.

Example extension:

```
<extension
  point="com.ibm.collaboration.realtime.telephony.ui.callMenuAction">
   <callMenuAction
```

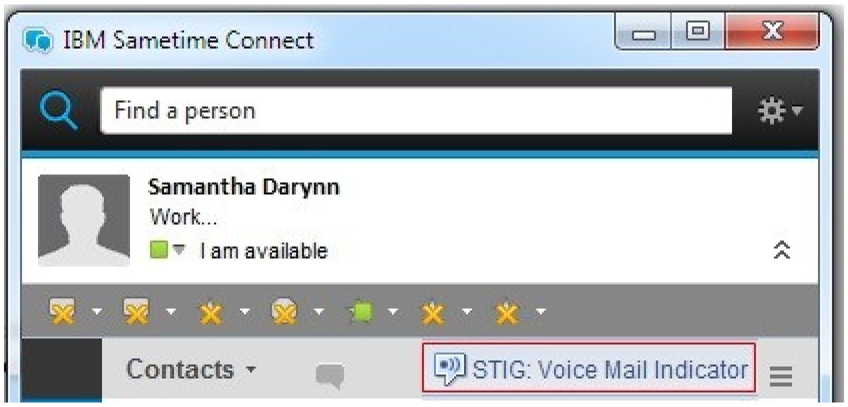
```
        class=" com.partner.custom.action.MyCallMenuAction"
        id="com.partner.custom.action.MyCallMenuAction"
        image="images/CallMenuActionPicture.png"
        displayName="%menu_label"/>
  </extension>
```

Example action class:

```
public class MyCallMenuAction implements CallMenuActionRunnable {

   public String getLabel() {
      // This returns the menu label defined in plugin.properties.
       return Platform.getResourceString(MyPlugin.getDefault().getBundle(),
            "%menu_label");
   }

   public void setSelection(ISelection selection) {
      // This selection contains an array of Person objects for
      // selected contact list entries. Use this data if the action
      // depends on the selection.
   }

   public void run() {
      // Do something useful here.
   }
}
```

Displaying a Voicemail Indicator

The IBM® Sametime Connect Toolkit includes a sample plug-in that shows you how to display a voice mail "message waiting" indicator on the upper toolbar of the Sametime Connect main window:



The sample simulates the voicemail indicator, but can be used as a starting point to implement this feature for Sametime Connect. A real implementation would query an external voicemail system, and perhaps provide actions to display a list of voicemail messages.

For details, see **com.ibm.collaboration.realtime.sample.snippets** sample in the Sametime Connect Toolkit. The code that displays the voice mail indicator is SampleVoiceMailIndicatorAction in the

.actions Java subpackage, and the extension is defined in the sample's `plugin.xml` file. For more information about the Sametime Connect Toolkit, see the *Reference* section at the end of this document.

Chapter 5. Using the Telephone Number Service

The telephone number service is a new extension for IBM® Sametime Connect 8.5.2 that allows customers to control the retrieval and formatting of telephone numbers used to make telephone calls from the client. This extension was initially designed for Sametime Unified Telephony (SUT), but is useful in any telephony-enabled environment where phone numbers retrieved from the corporate directory, or entered by users, might require modification to meet dialing plan requirements.

Sametime Connect 8.5.2 includes a default implementation of the telephone number service, but you can deploy your own implementation to customize the default behavior as follows:

- Retrieve the user's phone number from an alternate location. The default implementation attempts to retrieve the phone number from the “telephone” attribute in the corporate directory, and then from the user's location profile.
- Remove extensions embedded in the phone number, for example, change “111-2222 ext 1234” to “111-2222”. The default implementation removes extensions using extension delimiter(s) defined in the plug-in preference **com.ibm.collaboration.realtime.telephony.ui/extensionDelimiters**, which specifies one or more comma-separated delimiter strings. The default extension delimiter is “ext”. Note that when using Sametime Computer softphone to dial a number, the extension is treated like a passcode and automatically dialed once the call is connected, if the default implementation is used.
- Remove national prefixes embedded in the phone number, for example, change “+111 (0) 12345678” to +11112345678”. The default implementation removes national prefixes defined in **com.ibm.collaboration.realtime.telephony.ui/nationalPrefixes**, which specifies one or more comma-separated prefixes. The default national prefix is “(0)”.
- Replace an international prefix at the beginning of the phone number with “+”, for example, change “00123456789” to “+123456789”. The default implementation replaces international prefixes defined in **com.ibm.collaboration.realtime.telephony.ui/internationalPrefixes**, which specifies one or more comma-separated prefixes. There is no default international prefix.

Because the behavior of the default implementation is controlled by plug-in preferences, which can be modified by the Sametime administrator as needed, most customers will find it unnecessary to create and deploy their own telephone number service implementation. However, you can integrate your own telephone number service implementation using the telephone number service extension point com.ibm.collaboration.realtime.telephony.core.telephoneNumberService, which has the following attributes:

Table detailing com.ibm.collaboration.realtime.telephony.core.telephoneNumberService attributes.

Attribute	**Description and values**
`class`	Telephone number service class that implements the com.ibm.collaboration.realtime.telephony.number.TelephoneNumberService interface.
`communityHost`	Optional: The name of the Sametime community for which this telephone number service implementation should be used. If omitted, the service will be used for all communities.

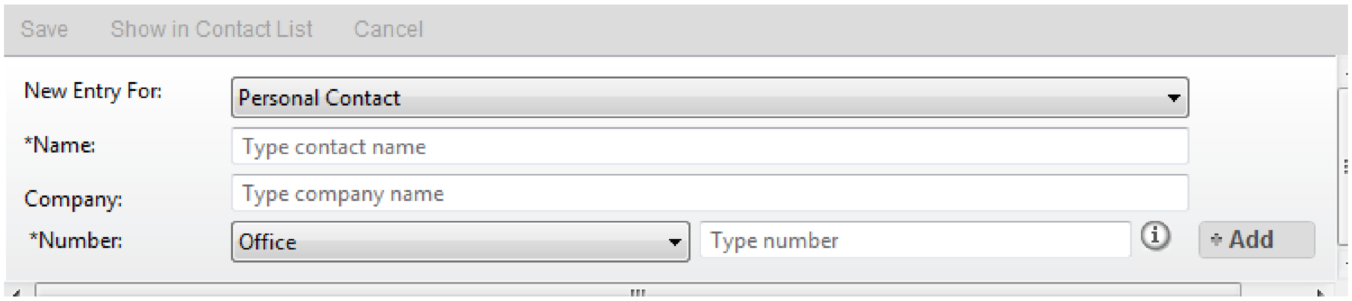
Your implementation of the telephone number service should extend com.ibm.collaboration.realtime.telephony.number.AbstractTelephoneNumberService, which provides the default implementation, and override only those methods that cannot be customized by changing the plug-in preferences described above. See the telephone number service sample plug-in in the client section of the Telephony Toolkit for more information.

Note that for Sametime 8.5.2 IFR 1, the following utility method has been added to allow plug-ins to access a directory telephone number using the telephone number service:

com.ibm.collaboration.realtime.telephony.TelephonyUtil.getDirectoryTelephoneNumber(Person)

Chapter 6. Using the Phone Book Service

The phonebook service is a new interface for IBM® Sametime Connect 9 that allows external developers to programmatically manipulate contacts with in the phonebook. This new interface allows below UI functionalities.



New Entry For:	Personal Contact
*Name:	Type contact name
Company:	Type company name
*Number:	Office / Type number

Note - “Show in Contact List” is not supported with this interface , but possible by adding the same contact using Buddy List API's. Refer to usage section below to see Buddy list API's, for more details please refer to Buddy List API's.
The API's allows following operations with in the phonebook.
1. Create Contact
2. Update Contact.
3. Read Contact.
4. Delete Contact.

API Details

- **public void createContact(Contact contact) throws InvalidContactException**
  - This API adds the contact to existing contact list .
  - A contact is said to be same if it has same contactId & communityId. If the contact is same it will update the existing contact.
  - If the contact fails in validation , InvalidContactException is thrown.

- **public boolean removeContact(Contact contact)**
  - This API will remove the contact from the contact list.
  - The API will check for search for contact in storage and will remove it if contact is found.

- **public List<Contact> getAllContacts()**
  - Returns list of all contacts from phonebook

- **public List<Contact> getContactsByName(String name)**
  - Returns all contacts from phonebook find by name.

- **public Contact getContactForNumber(PhoneNumber pbn)**
  - Returns the first contact from phonebook matches with given phone number.
  - A phone number is matched based on its label, number & passcode.

- **public List<Contact> getContactsByNameAndCompany(String name, String company)**
  - Returns list of contacts from contacts with matching name and company name.

- **public boolean updateContact(Contact oldContact , Contact newContact) throws InvalidContactException**
  - updates existing contact by new contact

Validation details

Each contact object will undergo following validations.

1. Check for mandatory inputs like communityId , Name , contactId , PhoneNumber , PhoneType.
2. Validate for correct PhoneType (`Home, Office , Mobile , Computer (SIP URI) , A/V Conferencing Device (h323), Conf)`. All these phone types are defined as constants in PhoneBookExternal interface which developer can make use of to avoid any string mismatch.
3. If a conference contact is added , it validates for phone number count , the count should always be one. Personal contact can have many phone numbers , but each conference contact can have only one number.
4. Validate for contactId format , connect client has its own internal contactId format which always starts with `"$xn$"` to distinguish an external contact. `PhoneBookExternalUtil` class exposes a method to create a contactId in connect client format which developers can make use of.
5. Validates if the contact is external contact.
6. Validate the number , following dial strings are supported

SIP/Tel/H323	Dial String supported
	sip:+18884266840@xyz.com (backward compatibility)
	sip:18884266840@xyzcom (backward compatibility)
	18884266840@xyz.com
	+18884266840 (new string supported for all SIP, TEL, & H323)
	18884266840 (new string supported for all SIP, TEL, & H323)
	tel:+18884266840 (backward compatibility)
	h323:+18884266840 (backward compatibility)
	h323:18884266840 (backward compatibility)

Usage

1. Add a contact on Phonebook

```
//create  PhoneBookExternal instance.
PhoneBookExternal pb= new PhoneBookExternalImpl();
 //create a phone number first
PhoneNumber number= new PhoneNumber();
number.setLabel(PhoneBookExternal.TYPE_SIP);
number.setValue("+18884266840");
List<PhoneNumber> phoneNumberList= new ArrayList<PhoneNumber>();
phoneNumberList.add(number);

//create contact object
Contact contact = new Contact();
contact.setName("XYZ");
// add a connect client contactId format
contact.setContactId(PhoneBookExternalUtil.createExternalUserContactId());
contact.setCompany("ABC");
contact.setIsExternal(true);
```

```
        contact.setPhoneNumbers(phoneNumberList);
        contact.setCommunityId(CommunityUtil.getDefaultCommunityId());

        // create the contact
        pb.createContact(contact);
```

2. Update a contact on phonebook

```
         //create a phone number object to search with
         PhoneNumber number= new PhoneNumber();
         number.setLabel(PhoneBookExternal.TYPE_CONF);
         number.setValue("18884266840@ibm.com");

         // get the contact object which has to be updated
         Contact oldContact = pb.getContactForNumber(number);
         // copy the contact
         Contact newContact=PhoneBookExternalUtil.copyContact(oldContact);
         //update the contact with new name
         newContact.setName("xyz");
         // update the contact to the phone book
         pb.updateContact(oldContact, newContact);
```

3. Remove a contact on phonebook

```
       //search the contact list by specified name
       List<Contact> contactList = pb.getContactsByName("xyz");
       // remove the contact
       pb.removeContact(contactList.get(0));
```

4. Add a contact in Buddy list

```
// create a person object from contact object
person=PeopleService.getPerson(cont.getContactId(),cont.getCommunityId(),cont.isExternal()/
get the groups
com.ibm.collaboration.realtime.buddylist.BuddyList.getAllPrivateGroups();
// add the person to the buddy list
com.ibm.collaboration.realtime.people.addPerson(person);
```

References

The IBM® Sametime Software Development Kit (SDK) provides a wealth of information for anyone that develops solutions for the Sametime platform. Telephony and audio/video application developers and service providers will find the following SDK references useful:

- The client section of the Telephony Toolkit (in the `telephony\client` directory) provides documentation for the client telephony API. The `doc` directory contains this *Client Telephony API Guide* and a list of API changes for this release (`API_Changes.txt`), and the `javadoc` directory contains the API Javadoc. The `samples` directory contains a sample plug-ins and features -- to access the sample code, import the sample plug-in JAR files (with source code) into your Eclipse workspace.
- The TCSPI (Telephony Conferencing Service Provider Interface) section of the Telephony Toolkit (in the `telephony\tcspi` directory) provides documentation and sample code for the TCSPI. Start with the *Sametime Voice and Video Integration Guide* (in the `tcspi` directory).
- The Sametime Connect Toolkit (`client\connect` directory) provides documentation and samples for Sametime Connect client plug-in developers. See the *IBM Sametime Integration Guide* and the *Extending Sametime* redbook, both in the `doc` directory, and the Sametime Connect API Javadoc, in the `javadoc\connect` directory.
- The telephony status plug-in described in Chapter 2 is documented in *Appendix B. Livename Extended Status API* of the *IBM Sametime Integration Guide* in the Sametime Connect Toolkit.
- The Telephony Presence Adapter, also described in Chapter 2, is documented in *Chapter 13. Telephony Presence Adapter* of the *Community Server Toolkit Developer's Guide*, in the Community Server Toolkit (in the `server\commserver\doc` directory).

In addition to the SDK materials, see the following online resources:

- IBM Sametime wiki
  http://www-10.lotus.com/ldd/stwiki.nsf/xpViewCategories.xsp?lookupName=Product%20Documentation
- IBM Lotus Expeditor 6.2 Information Center
  http://publib.boulder.ibm.com/infocenter/ledoc/v6r2/index.jsp
- IBM developerWorks Sametime product page
  http://www.ibm.com/developerworks/lotus/products/instantmessaging
- IBM developerWorks technical library
  http://www.ibm.com/developerworks/views/lotus/libraryview.jsp

Notices

Trademarks