

EPOS

DSEA Call Control Library

V 1.19.0.5159

Fri Sep 4 2026

1 Class Index	1
1.1 Class List	1
2 File Index	1
2.1 File List	1
3 Class Documentation	2
3.1 dsea::callcontrol::sdk::DeviceInfo Struct Reference	2
3.1.1 Detailed Description	2
3.1.2 Member Data Documentation	2
3.2 dsea::callcontrol::sdk::ICallControlInterface Class Reference	3
3.2.1 Detailed Description	3
3.2.2 Constructor & Destructor Documentation	3
3.2.3 Member Function Documentation	4
3.3 dsea::callcontrol::sdk::ICallControlListener Class Reference	4
3.3.1 Detailed Description	5
3.3.2 Member Function Documentation	6
3.4 dsea::callcontrol::sdk::ILogger Class Reference	11
3.4.1 Detailed Description	11
3.4.2 Member Function Documentation	11
3.5 dsea::callcontrol::sdk::ISoftphoneEventHandler Class Reference	12
3.5.1 Detailed Description	13
3.5.2 Member Function Documentation	13
3.6 dsea::callcontrol::sdk::SoftphoneInfoSt Struct Reference	17
3.6.1 Detailed Description	18
4 File Documentation	18
4.1 DeviceInfo.h File Reference	18
4.1.1 Detailed Description	19
4.1.2 Typedef Documentation	19
4.2 DeviceInfo.h	19
4.3 ICallControlInterface.h File Reference	20
4.3.1 Detailed Description	21
4.3.2 Function Documentation	21
4.4 ICallControlInterface.h	21
4.5 ICallControlListener.h File Reference	22
4.5.1 Detailed Description	23
4.5.2 Enumeration Type Documentation	23
4.6 ICallControlListener.h	23
4.7 ILogger.h File Reference	24
4.7.1 Detailed Description	25
4.7.2 Enumeration Type Documentation	25
4.8 ILogger.h	25

4.9 ISoftphoneEventHandler.h File Reference	25
4.9.1 Detailed Description	26
4.10 ISoftphoneEventHandler.h	26
Index	27

1 Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

dsea::callcontrol::sdk::DeviceInfo	2
dsea::callcontrol::sdk::ICallControlInterface Interface for CallControlInterface. The call control interface provides the handler to send soft- phone events to sdk	3
dsea::callcontrol::sdk::ICallControlListener Interface for CallControlListener. This is a listener interface to be implemented by the softphone and is used for notifying events from call control sdk to softphone	4
dsea::callcontrol::sdk::ILogger Interface for logger	11
dsea::callcontrol::sdk::ISoftphoneEventHandler Interface for SoftphoneEventHandler. This interface provides the methods to send events to SDK from softphone	12
dsea::callcontrol::sdk::SoftphoneInfoSt Structure containing softphone information. Used by call control for logging and identification purposes	17

2 File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

DeviceInfo.h This file contains the structure for device information	18
ICallControlInterface.h This file provides declarations for the call control interface	20
ICallControlListener.h Contains the declarations for the ICallControlListener interface	22
ILogger.h This is interface for softphone logger which will be used to log call control events	24

ISoftphoneEventHandler.h

Contains the declaration for the ISoftphoneEventHandler interface

25

3 Class Documentation

3.1 dsea::callcontrol::sdk::DeviceInfo Struct Reference

```
#include <DeviceInfo.h>
```

Public Member Functions

- **DeviceInfo** (const std::string &f_strInstanceld, const std::string &f_strDeviceName, const std::string &f_strDeviceFriendlyName, const std::string &f_strSerialNo, HeadsetRole f_eDeviceRole, int m_iPID, int m_iVID)
Constructor.

Public Attributes

- std::string [m_strInstanceld](#)
- std::string [m_strDeviceName](#)
- std::string [m_strDeviceFriendlyName](#)
- std::string [m_strSerialNo](#)
- HeadsetRole [m_eHeadsetRole](#)
- int [m_iProductId](#)
- int [m_iVendorId](#)

3.1.1 Detailed Description

Structure defines the headset information.

3.1.2 Member Data Documentation

3.1.2.1 [m_eHeadsetRole](#) `HeadsetRole dsea::callcontrol::sdk::DeviceInfo::m_eHeadsetRole`

Role of the headset

3.1.2.2 [m_iProductId](#) `int dsea::callcontrol::sdk::DeviceInfo::m_iProductId`

Product ID of the headset

3.1.2.3 [m_iVendorId](#) `int dsea::callcontrol::sdk::DeviceInfo::m_iVendorId`

Vendor ID of the headset

3.1.2.4 m_strDeviceFriendlyName `std::string dsea::callcontrol::sdk::DeviceInfo::m_strDevice↔
FriendlyName`

Friendly name of the headset

3.1.2.5 m_strDeviceName `std::string dsea::callcontrol::sdk::DeviceInfo::m_strDeviceName`

Name of the headset

3.1.2.6 m_strInstanceId `std::string dsea::callcontrol::sdk::DeviceInfo::m_strInstanceId`

Instance Id of the headset. This uniquely identifies each connected headset instance.

3.1.2.7 m_strSerialNo `std::string dsea::callcontrol::sdk::DeviceInfo::m_strSerialNo`

Serial number of the headset

The documentation for this struct was generated from the following file:

- [DeviceInfo.h](#)

3.2 dsea::callcontrol::sdk::ICallControlInterface Class Reference

Interface for CallControlInterface. The call control interface provides the handler to send softphone events to sdk.

```
#include <ICallControlInterface.h>
```

Public Member Functions

- virtual [~ICallControlInterface](#) ()=default
Destructor.
- virtual std::shared_ptr< [ISoftphoneEventHandler](#) > [GetSoftphoneEventHandler](#) ()=0
Provides the pointer to Softphone Event Handler. It is used to send call events to SDK from softphone.
- virtual std::string [GetVersion](#) () const =0
Version of the SDK.

3.2.1 Detailed Description

Interface for CallControlInterface. The call control interface provides the handler to send softphone events to sdk.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 ~ICallControlInterface() `virtual dsea::callcontrol::sdk::ICallControlInterface::~~ICallControlInterface ( ) [virtual], [default]`

Destructor.

Returns

None

3.2.3 Member Function Documentation

3.2.3.1 GetSoftphoneEventHandler() `virtual std::shared_ptr< ISoftphoneEventHandler > dsea::callcontrol::sdk::ICallControlInterface::GetSoftphoneEventHandler ( ) [pure virtual]`

Provides the pointer to Softphone Event Handler. It is used to send call events to SDK from softphone.

Returns

Return Softphone Event Handler

3.2.3.2 GetVersion() `virtual std::string dsea::callcontrol::sdk::ICallControlInterface::GetVersion ( ) const [pure virtual]`

Version of the SDK.

Returns

Returns the SDK Version as string

The documentation for this class was generated from the following file:

- [ICallControlInterface.h](#)

3.3 dsea::callcontrol::sdk::ICallControlListener Class Reference

Interface for CallControlListener. This is a listener interface to be implemented by the softphone and is used for notifying events from call control sdk to softphone.

```
#include <ICallControlListener.h>
```

Public Member Functions

- virtual `~ICallControlListener` ()=default
Destructor.
- virtual void `LoginStatusChanged` (`LoginStatusValue` f_bLoginStatus)=0
Callback to inform the softphone on the login status.
- virtual void `HeadsetAvailable` (const `StDeviceInfo` &f_deviceInfo)=0
Callback to inform the softphone about new headset connected event.
- virtual void `HeadsetRemoved` (const `StDeviceInfo` &f_deviceInfo)=0
Callback to inform the softphone about headset disconnected event.
- virtual void `HeadsetConnected` (const `StDeviceInfo` &f_deviceInfo)=0
Callback to inform the softphone when a headset connected is made active.
- virtual void `HeadsetDisconnected` (const `StDeviceInfo` &f_deviceInfo)=0
Callback to inform the softphone when an active headset is disconnected.
- virtual void `AcceptCall` (const std::string &f_strCallId)=0
Callback to inform the softphone about accept call event from the headset. The softphone can choose to ignore this event or accept the call. If call is accepted the softphone should invoke `ISoftphoneEventHandler::SendActiveCallEvent()` with the same call id to acknowledge the operation.
- virtual void `HoldCall` (const std::string &f_strCallId)=0
Callback to inform the softphone about hold call event from the headset. The softphone can choose to ignore this event or hold the call. If call is put on hold the softphone should invoke `ISoftphoneEventHandler::SendHoldCallEvent()` with the same call id to acknowledge the operation.
- virtual void `ResumeCall` (const std::string &f_strCallId)=0
Callback to inform the softphone about resume call event from the headset. The softphone can choose to ignore this event or resume the call. If call is resumed the softphone should invoke `ISoftphoneEventHandler::SendActiveCallEvent()` with the same call id to acknowledge the operation.
- virtual void `EndCall` (const std::string &f_strCallId)=0
Callback to inform the softphone about end call event from the headset. The softphone can choose to ignore this event or end the call. If call is ended the softphone should invoke `ISoftphoneEventHandler::SendEndCallEvent()` with the same call id to acknowledge the operation.
- virtual void `RejectCall` (const std::string &f_strCallId)=0
Callback to inform the softphone to reject an incoming call specified by the call id when it is rejected from headset. The softphone can choose to ignore this event or to reject the call. If call is rejected the softphone should invoke `ISoftphoneEventHandler::SendEndCallEvent()` with the same call id to acknowledge the operation.
- virtual void `RequestMuteStatus` ()=0
This callback is to request the softphone to update the microphone mute status. The softphone should call the `ISoftphoneEventHandler::SendMuteStatus()` with the mute status.
- virtual void `SetMuteStatus` (bool f_bMuteStatus)=0
Callback to inform the softphone to mute/unmute event from headset. If there is a change in mute status the softphone should call the `ISoftphoneEventHandler::SendMuteStatus()` with the updated status value.
- virtual void `OffHook` (const std::string &f_strInstanceId)=0
Callback to inform about the OffHook event on the headset when idle. It is upto the softphone to interpret the event and perform the corresponding action.
- virtual void `Redial` (const std::string &f_strInstanceId)=0
Callback to inform the softphone to redial the last outgoing call when appropriate button is pressed on headset. If the outgoing call is created the softphone should invoke `ISoftphoneEventHandler::SendOutgoingCallEvent()`
- virtual void `ResponseSystemInformation` (const std::string &f_strProductName, const std::string &f_strVersion)=0
Callback to inform the softphone about Call Control System Information.

3.3.1 Detailed Description

Interface for CallControlListener. This is a listener interface to be implemented by the softphone and is used for notifying events from call control sdk to softphone.

3.3.2 Member Function Documentation

3.3.2.1 AcceptCall() `virtual void dsea::callcontrol::sdk::ICallControlListener::AcceptCall ( const std::string & f_strCallId ) [pure virtual]`

Callback to inform the softphone about accept call event from the headset. The softphone can choose to ignore this event or accept the call. If call is accepted the softphone should invoke [ISoftphoneEventHandler::SendActiveCallEvent\(\)](#) with the same call id to acknowledge the operation.

Parameters

<code>f_str↵ CallId</code>	: Id of the call to be accepted
--------------------------------	---------------------------------

Returns

None

3.3.2.2 EndCall() `virtual void dsea::callcontrol::sdk::ICallControlListener::EndCall ( const std::string & f_strCallId ) [pure virtual]`

Callback to inform the softphone about end call event from the headset. The softphone can choose to ignore this event or end the call. If call is ended the softphone should invoke [ISoftphoneEventHandler::SendEndCallEvent\(\)](#) with the same call id to acknowledge the operation.

Parameters

<code>f_str↵ CallId</code>	: Id of the call to be ended
--------------------------------	------------------------------

Returns

None

3.3.2.3 HeadsetAvailable() `virtual void dsea::callcontrol::sdk::ICallControlListener::Headset↵
Available (const StDeviceInfo & f_deviceInfo) [pure virtual]`

Callback to inform the softphone about new headset connected event.

Parameters

<code>f_deviceInfo</code>	: Device information of the newly connected headset. The softphone should call the ISoftphoneEventHandler::SendActiveDeviceChanged to set the device as active
---------------------------	--

Returns

None

3.3.2.4 HeadsetConnected() `virtual void dsea::callcontrol::sdk::ICallControlListener::HeadsetConnected ( const StDeviceInfo & f_deviceInfo ) [pure virtual]`

Callback to inform the softphone when a headset connected is made active.

Parameters

<code>f_deviceInfo</code>	: Device information of the active headset. Call Control will work on the headset only if it is the active headset
---------------------------	--

Returns

None

3.3.2.5 HeadsetDisconnected() `virtual void dsea::callcontrol::sdk::ICallControlListener::HeadsetDisconnected ( const StDeviceInfo & f_deviceInfo ) [pure virtual]`

Callback to inform the softphone when an active headset is disconnected.

Parameters

<code>f_deviceInfo</code>	: Device information of the active headset that is now disconnected
---------------------------	---

Returns

None

3.3.2.6 HeadsetRemoved() `virtual void dsea::callcontrol::sdk::ICallControlListener::HeadsetRemoved ( const StDeviceInfo & f_deviceInfo ) [pure virtual]`

Callback to inform the softphone about headset disconnected event.

Parameters

<code>f_deviceInfo</code>	: Device information of the disconnected headset
---------------------------	--

Returns

None

3.3.2.7 HoldCall() `virtual void dsea::callcontrol::sdk::ICallControlListener::HoldCall ( const std::string & f_strCallId ) [pure virtual]`

Callback to inform the softphone about hold call event from the headset. The softphone can choose to ignore this event or hold the call. If call is put on hold the softphone should invoke [ISoftphoneEventHandler::SendHoldCallEvent\(\)](#) with the same call id to acknowledge the operation.

Parameters

<code>f_strCallId</code>	: Id of the call to be put on hold
--------------------------	------------------------------------

Returns

None

3.3.2.8 LoginStatusChanged() `virtual void dsea::callcontrol::sdk::ICallControlListener::LoginStatusChanged ( LoginStatusValue f_bLoginStatus ) [pure virtual]`

Callback to inform the softphone on the login status.

Parameters

<code>f_bLoginStatus</code>	: Status of the login operation. This will be called in response to the ISoftphoneEventHandler::SendLoginEvent() or ISoftphoneEventHandler::SendLogoutEvent() API calls
-----------------------------	---

Returns

None

3.3.2.9 OffHook() `virtual void dsea::callcontrol::sdk::ICallControlListener::OffHook ( const std::string & f_strInstanceId ) [pure virtual]`

Callback to inform about the OffHook event on the headset when idle. It is upto the softphone to interpret the event and perform the corresponding action.

Parameters

<i>f_str↔ InstanceId</i>	Instance ID of Headset
------------------------------	------------------------

Returns

None

3.3.2.10 Redial() `virtual void dsea::callcontrol::sdk::ICallControlListener::Redial ( const std::string & f_strInstanceId ) [pure virtual]`

Callback to inform the softphone to redial the last outgoing call when appropriate button is pressed on headset. If the outgoing call is created the softphone should invoke [ISoftphoneEventHandler::SendOutgoingCallEvent\(\)](#)

Parameters

<i>f_str↔ InstanceId</i>	Instance ID of Headset
------------------------------	------------------------

Returns

None

3.3.2.11 RejectCall() `virtual void dsea::callcontrol::sdk::ICallControlListener::RejectCall ( const std::string & f_strCallId ) [pure virtual]`

Callback to inform the softphone to reject an incoming call specified by the call id when it is rejected from headset. The softphone can choose to ignore this event or to reject the call. If call is rejected the softphone should invoke [ISoftphoneEventHandler::SendEndCallEvent\(\)](#) with the same call id to acknowledge the operation.

Parameters

<i>f_str↔ CallId</i>	: Id of the call to be rejected or ended.
--------------------------	---

Returns

None

3.3.2.12 RequestMuteStatus() `virtual void dsea::callcontrol::sdk::ICallControlListener::↔RequestMuteStatus ( ) [pure virtual]`

This callback is to request the softphone to update the microphone mute status. The softphone should call the [ISoftphoneEventHandler::SendMuteStatus\(\)](#) with the mute status.

Returns

None

3.3.2.13 ResponseSystemInformation() `virtual void dsea::callcontrol::sdk::ICallControlListener::ResponseSystemInformation (`
`const std::string & f_strProductName,`
`const std::string & f_strVersion ) [pure virtual]`

Callback to inform the softphone about Call Control System Information.

Parameters

<i>f_strProductName</i>	: Product Name of the call control software
<i>f_strVersion</i>	: Version of call control

Returns

None

3.3.2.14 ResumeCall() `virtual void dsea::callcontrol::sdk::ICallControlListener::ResumeCall (`
`const std::string & f_strCallId ) [pure virtual]`

Callback to inform the softphone about resume call event from the headset. The softphone can choose to ignore this event or resume the call. If call is resumed the softphone should invoke [ISoftphoneEventHandler::SendActiveCallEvent\(\)](#) with the same call id to acknowledge the operation.

Parameters

<i>f_str↔ CallId</i>	: Id of the call to be put on resumed
--------------------------	---------------------------------------

Returns

None

3.3.2.15 SetMuteStatus() `virtual void dsea::callcontrol::sdk::ICallControlListener::SetMute↔
Status (`
`bool f_bMuteStatus ) [pure virtual]`

Callback to inform the softphone to mute/unmute event from headset. If there is a change in mute status the softphone should call the [ISoftphoneEventHandler::SendMuteStatus\(\)](#) with the updated status value.

Parameters

<code>f_bMuteStatus</code>	: Mute status - true if muted, false otherwise
----------------------------	--

Returns

None

The documentation for this class was generated from the following file:

- [ICallControlListener.h](#)

3.4 dsea::callcontrol::sdk::ILogger Class Reference

Interface for logger.

```
#include <ILogger.h>
```

Public Member Functions

- virtual `~ILogger()`=default
Destructor.
- virtual void `WriteLog(LogLevel f_ecLogLevel, const std::string &f_strLogMsg)=0`
Write the log message to a console or log file. Log levels supported - warning, info, debug or error messages.

3.4.1 Detailed Description

Interface for logger.

3.4.2 Member Function Documentation

3.4.2.1 WriteLog() virtual void dsea::callcontrol::sdk::ILogger::WriteLog (
`LogLevel f_ecLogLevel,`
`const std::string & f_strLogMsg ) [pure virtual]`

Write the log message to a console or log file. Log levels supported - warning, info, debug or error messages.

Parameters

<code>f_ecLogLevel</code>	Severity of the log message.
<code>f_strLogMsg</code>	The log message as string to be logged

Returns

None

The documentation for this class was generated from the following file:

- [ILogger.h](#)

3.5 dsea::callcontrol::sdk::ISoftphoneEventHandler Class Reference

Interface for SoftphoneEventHandler. This interface provides the methods to send events to SDK from softphone.

```
#include <ISoftphoneEventHandler.h>
```

Public Member Functions

- virtual **~ISoftphoneEventHandler** ()=default
Destructor.
- virtual void **ResetSoftphoneHandler** ()=0
Releases all the softphone handler resources.
- virtual void **SendLoginEvent** ()=0
Send the Login event to the SDK. The login status will be notified by the [ICallControlListener::LoginStatusChanged\(\)](#) callback.
- virtual void **SendLogoutEvent** ()=0
Send the Logout event to the SDK. The login status will be notified by the [ICallControlListener::LoginStatusChanged\(\)](#) callback.
- virtual void **SendOutgoingCallEvent** (const std::string &f_strCallId)=0
Send the new outgoing call event to SDK.
- virtual void **SendIncomingCallEvent** (const std::string &f_strCallId)=0
Send the new incoming call event to SDK.
- virtual void **SendActiveCallEvent** (const std::string &f_strCallId)=0
Send Active call event to SDK.
- virtual void **SendHoldCallEvent** (const std::string &f_strCallId)=0
Send hold call event to SDK.
- virtual void **SendEndCallEvent** (const std::string &f_strCallId)=0
Send end call event to SDK. The call will cease to exist in the SDK after this.
- virtual void **SendActiveDeviceChanged** (bool f_isSupportedDevice, const std::string &f_strDeviceInstance↵
Id)=0
Send active device changed event to SDK. This is used to select the active headset by the softphone. At a time only one headset can be active and call control can be done only using the active headset. The [ICallControlListener::HeadsetConnected\(\)](#) callback will report the new active headset.
- virtual void **SendMuteStatus** (bool f_bMuteStatus)=0
Send mute status to the SDK.
- virtual void **RequestSystemInformation** ()=0
*Request for System Information. The requested information will be provided by [ICallControlListener::Response↵
SystemInformation\(\)](#)*
- virtual void **SetPresentingContentStatus** (bool f_bStatus)=0
Set the presenting content status of the softphone. This is used for displaying the corresponding status on the busylight.
- virtual void **SetIMNotificationEvent** ()=0
Indicate an incoming Instant Message (IM) notification event. This is used to display the corresponding notification on the busylight device.
- virtual void **SetPresenceStatus** (const std::string &f_strPresence)=0
Set the softphone presence status. This is used to indicate the presence status on the busylight device.
- virtual void **SetDNDStatus** (bool f_bStatus)=0
Set the DND (Do not disturb) of the softphone.

3.5.1 Detailed Description

Interface for SoftphoneEventHandler. This interface provides the methods to send events to SDK from softphone.

3.5.2 Member Function Documentation

3.5.2.1 RequestSystemInformation() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::RequestSystemInformation ( ) [pure virtual]`

Request for System Information. The requested information will be provided by `ICallControlListener::ResponseSystemInformation()`

Returns

None

3.5.2.2 SendActiveCallEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendActiveCallEvent ( const std::string & f_strCallId ) [pure virtual]`

Send Active call event to SDK.

Parameters

<code>f_strCallId</code>	Call id of the active/connected/resumed call
--------------------------	--

Returns

None

3.5.2.3 SendActiveDeviceChanged() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendActiveDeviceChanged ( bool f_isSupportedDevice, const std::string & f_strDeviceInstanceId ) [pure virtual]`

Send active device changed event to SDK. This is used to select the active headset by the softphone. At a time only one headset can be active and call control can be done only using the active headset. The [ICallControlListener::HeadsetConnected\(\)](#) callback will report the new active headset.

Parameters

<code>f_isSupportedDevice</code>	: Set this to true if the device is EPOS/DSEA, false otherwise
<code>f_strDeviceInstanceId</code>	: Instance Id of the headset to be set as active

Returns

None

3.5.2.4 SendEndCallEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendEndCallEvent (`
`const std::string & f_strCallId ) [pure virtual]`

Send end call event to SDK. The call will cease to exist in the SDK after this.

Parameters

<code>f_strCallId</code>	: Call Id of the call to be ended
--------------------------	-----------------------------------

Returns

None

3.5.2.5 SendHoldCallEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendHoldCallEvent (`
`const std::string & f_strCallId ) [pure virtual]`

Send hold call event to SDK.

Parameters

<code>f_strCallId</code>	: Call Id of the held call
--------------------------	----------------------------

Returns

None

3.5.2.6 SendIncomingCallEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendIncomingCallEvent (`
`const std::string & f_strCallId ) [pure virtual]`

Send the new incoming call event to SDK.

Parameters

<code>f_strCallId</code>	: Call id of the incoming call
--------------------------	--------------------------------

Returns

None

3.5.2.7 SendLoginEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendLoginEvent ( ) [pure virtual]`

Send the Login event to the SDK. The login status will be notified by the [ICallControlListener::LoginStatusChanged\(\)](#) callback.

Returns

None

3.5.2.8 SendLogoutEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendLogoutEvent ( ) [pure virtual]`

Send the Logout event to the SDK. The login status will be notified by the [ICallControlListener::LoginStatusChanged\(\)](#) callback.

Returns

None

3.5.2.9 SendMuteStatus() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendMuteStatus ( bool f_bMuteStatus ) [pure virtual]`

Send mute status to the SDK.

Parameters

<code>f_bMuteStatus</code>	True if the microphone should be muted, false otherwise
----------------------------	---

Returns

None

3.5.2.10 SendOutgoingCallEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::SendOutgoingCallEvent ( const std::string & f_strCallId ) [pure virtual]`

Send the new outgoing call event to SDK.

Parameters

<i>f_str↔ CallId</i>	: Call id of the outgoing call
--------------------------	--------------------------------

Returns

None

3.5.2.11 SetDNDStatus() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::Set↔
DNDStatus (`
 `bool f_bStatus ) [pure virtual]`

Set the DND (Do not disturb) of the softphone.

Parameters

<i>f_bStatus</i>	Set this to true if the DND is enabled in the softphone, false otherwise
------------------	--

Returns

None

3.5.2.12 SetIMNotificationEvent() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler↔
::SetIMNotificationEvent () [pure virtual]`

Indicate an incoming Instant Message (IM) notification event. This is used to display the corresponding notification on the busylight device.

Returns

None

3.5.2.13 SetPresenceStatus() `virtual void dsea::callcontrol::sdk::ISoftphoneEventHandler::Set↔
PresenceStatus (`
 `const std::string & f_strPresence ) [pure virtual]`

Set the softphone presence status. This is used to indicate the presence status on the busylight device.

Parameters

<i>f_strPresence</i>	string containing the default Presence then To set Presence use "<DefaultPresenceString>" where DefaultPresenceString = Default presence status like Available,Busy,DoNotDisturb and Away E.g. "Available" will result in solid green to be shown on the Busy light device OR string containing User Defined Custom Presence State on the Busylight device "<presenceName>,<presenceColor>,<presencePattern>" presenceName: Expected value is any string. This parameter has no validation it can also be left empty. presenceColor: Expected value is of the format "#RRGGBB" presencePattern: Expected value is {0,1,2,3,4}: 0 – Busylight will not Glow 1 – Busylight will continuously Glow 2 – Busylight will Flash 3 – Busylight will Pulse (Breathing Light effect) 4 – Busylight will glow with a Long Flash Pattern E.g. "BeRightBack,\$FF0000,3" will result in Red flash to be shown on the Busy light device
----------------------	---

Returns

None

3.5.2.14 SetPresentingContentStatus() `virtual void dsea::callcontrol::sdk::ISoftphoneEvent↔
Handler::SetPresentingContentStatus (`
`bool f_bStatus ) [pure virtual]`

Set the presenting content status of the softphone. This is used for displaying the corresponding status on the busylight.

Parameters

<i>f_bStatus</i>	Set this to true if the content is being presented on the softphone, false otherwise
------------------	--

Returns

None

The documentation for this class was generated from the following file:

- [ISoftphoneEventHandler.h](#)

3.6 dsea::callcontrol::sdk::SoftphoneInfoSt Struct Reference

Structure containing softphone information. Used by call control for logging and identification purposes.

```
#include <ICallControlInterface.h>
```

Public Member Functions

- **SoftphoneInfoSt** (const std::string &f_strSoftphoneName, const std::string &f_strSoftphoneBrand, const std::string &f_strSoftphoneVersion)
- **bool operator==** (const [SoftphoneInfoSt](#) &f_softphoneInfo) const

Public Attributes

- `std::string m_strSoftphoneName`
The name of the softphone.
- `std::string m_strSoftphoneBrand`
The brand name of the softphone.
- `std::string m_strSoftphoneVersion`
The version of the softphone.

3.6.1 Detailed Description

Structure containing softphone information. Used by call control for logging and identification purposes.

The documentation for this struct was generated from the following file:

- [ICallControlInterface.h](#)

4 File Documentation

4.1 DeviceInfo.h File Reference

This file contains the structure for device information.

```
#include <string>
#include <map>
```

Classes

- struct [dsea::callcontrol::sdk::DeviceInfo](#)

Typedefs

- typedef struct [dsea::callcontrol::sdk::DeviceInfo](#) [dsea::callcontrol::sdk::StDeviceInfo](#)

Enumerations

- enum class **HeadsetRole** {
 Unknown = 0 , **Headset** = 1 , **Speakerphone** = 2 , **Busylight** = 3 ,
 Camera = 4 , **Dongle** = 5 , **Microphone** = 6 , **DAC** = 7 ,
 ExtensionCable = 8 }

Functions

- bool **dsea::callcontrol::sdk::operator==** (const StDeviceInfo &lhs, const StDeviceInfo &rhs)

4.1.1 Detailed Description

This file contains the structure for device information.

Author

EPOS

Copyright

Copyright 2021 EPOS. All rights reserved.

Date

2021-03-04

4.1.2 Typedef Documentation

4.1.2.1 StDeviceInfo typedef struct dsea::callcontrol::sdk::DeviceInfo dsea::callcontrol::sdk::StDeviceInfo

Structure defines the headset information.

4.2 DeviceInfo.h

[Go to the documentation of this file.](#)

```
1
11 #ifndef DEVICEINFO_H
12 #define DEVICEINFO_H
13
14 #include <string>
15 #include <map>
16
17 namespace dsea::callcontrol::sdk
18 {
19     enum class HeadsetRole
20     {
21         Unknown = 0,
22         Headset = 1,
23         Speakerphone = 2,
24         Busylight = 3,
25         Camera = 4,
26         Dongle = 5,
27         Microphone = 6,
28         DAC = 7,
29         ExtensionCable = 8
30     };
31
32     typedef struct DeviceInfo
33     {
34     public:
35         DeviceInfo(const std::string& f_strInstanceId,
36                   const std::string& f_strDeviceName,
37                   const std::string& f_strDeviceFriendlyName,
38                   const std::string& f_strSerialNo,
39                   HeadsetRole f_eDeviceRole,
40                   int m_iPID,
41                   int m_iVID)
42             : m_strInstanceId(f_strInstanceId),
43               m_strDeviceName(f_strDeviceName),
44               m_strDeviceFriendlyName(f_strDeviceFriendlyName),
45               m_strSerialNo(f_strSerialNo),
46               m_eHeadsetRole(f_eDeviceRole),
```

```

53         m_iProductId(m_iPID),
54         m_iVendorId(m_iVID)
55     {
56     }
61     std::string m_strInstanceId;
62
66     std::string m_strDeviceName;
67
71     std::string m_strDeviceFriendlyName;
72
76     std::string m_strSerialNo;
77
81     HeadsetRole m_eHeadsetRole;
82
86     int m_iProductId;
87
91     int m_iVendorId;
92 } StDeviceInfo;
93
94 inline bool operator==(const StDeviceInfo& lhs, const StDeviceInfo& rhs)
95 {
96     if ((lhs.m_strInstanceId == rhs.m_strInstanceId) &&
97         (lhs.m_strDeviceName == rhs.m_strDeviceName) &&
98         (lhs.m_strDeviceFriendlyName == rhs.m_strDeviceFriendlyName) &&
99         (lhs.m_strSerialNo == rhs.m_strSerialNo) &&
100        (lhs.m_eHeadsetRole == rhs.m_eHeadsetRole) &&
101        (lhs.m_iProductId == rhs.m_iProductId) &&
102        (lhs.m_iVendorId == rhs.m_iVendorId))
103     {
104         return true;
105     }
106     else
107     {
108         return false;
109     }
110 }
111 }
112
113 #endif // DEVICEINFO_H

```

4.3 ICallControlInterface.h File Reference

This file provides declarations for the call control interface.

```

#include "ICallControlListener.h"
#include "ILogger.h"
#include <memory>
#include <string>

```

Classes

- struct [dsea::callcontrol::sdk::SoftphoneInfoSt](#)
Structure containing softphone information. Used by call control for logging and identification purposes.
- class [dsea::callcontrol::sdk::ICallControlInterface](#)
Interface for CallControlInterface. The call control interface provides the handler to send softphone events to sdk.

Functions

- `std::unique_ptr< dsea::callcontrol::sdk::ICallControlInterface > CreateCallControlInstance` (const [dsea::callcontrol::sdk::SoftphoneInfoSt](#) &f_softphoneInfo, [dsea::callcontrol::sdk::ICallControlListener](#) &f_callControllistener, [dsea::callcontrol::sdk::ILogger](#) &f_logger, bool f_bEnableVerboseLogging)
Create and return the call control interface.

4.3.1 Detailed Description

This file provides declarations for the call control interface.

Author

EPOS

Copyright

Copyright 2021 EPOS. All rights reserved.

Date

2021-02-18

4.3.2 Function Documentation

4.3.2.1 CreateCallControlInstance() `std::unique_ptr< dsea::callcontrol::sdk::ICallControlInterface
> CreateCallControlInstance (`
 `const dsea::callcontrol::sdk::SoftphoneInfoSt & f_softphoneInfo,`
 `dsea::callcontrol::sdk::ICallControlListener & f_callControllistener,`
 `dsea::callcontrol::sdk::ILogger & f_logger,`
 `bool f_bEnableVerboseLogging )`

Create and return the call control interface.

Parameters

<i>f_softphoneInfo</i>	: Softphone information
<i>f_callControllistener</i>	: Listener interface to receive events from call control to softphone
<i>f_logger</i>	: Interface to logger
<i>f_bEnableVerboseLogging</i>	: To enable verbose logging
<i>f_bForceDseaStandalone</i>	: To enable DSEA Standalone even when DSEA SDK is installed.

Returns

Return the unique pointer to call controller interface

4.4 ICallControlInterface.h

[Go to the documentation of this file.](#)

```
1
11 #ifndef ICALLCONTROLINTERFACE_H
12 #define ICALLCONTROLINTERFACE_H
13
14 #include "ICallControlListener.h"
15 #include "ILogger.h"
```

```

16
17 #include <memory>
18 #include <string>
19
20 namespace dsea::callcontrol::sdk
21 {
22     struct SoftphoneInfoSt
23     {
24         SoftphoneInfoSt(const std::string& f_strSoftphoneName, const std::string& f_strSoftphoneBrand,
25             const std::string& f_strSoftphoneVersion) :
26             m_strSoftphoneName(f_strSoftphoneName),
27             m_strSoftphoneBrand(f_strSoftphoneBrand),
28             m_strSoftphoneVersion(f_strSoftphoneVersion)
29         {
30         }
31     };
32
33     bool operator == (const SoftphoneInfoSt& f_softphoneInfo) const
34     {
35         return ((m_strSoftphoneName == f_softphoneInfo.m_strSoftphoneName) &&
36             (m_strSoftphoneBrand == f_softphoneInfo.m_strSoftphoneBrand) &&
37             (m_strSoftphoneVersion == f_softphoneInfo.m_strSoftphoneVersion));
38     }
39
40     std::string m_strSoftphoneName;
41     std::string m_strSoftphoneBrand;
42     std::string m_strSoftphoneVersion;
43 };
44
45 class ISoftphoneEventHandler;
46
47 class ICallControlInterface
48 {
49 public:
50     virtual ~ICallControlInterface() = default;
51
52     virtual std::shared_ptr<ISoftphoneEventHandler> GetSoftphoneEventHandler() = 0;
53
54     virtual std::string GetVersion() const = 0;
55 };
56
57 #ifdef _WIN32
58 __declspec(dllexport) std::unique_ptr<dsea::callcontrol::sdk::ICallControlInterface>
59     CreateCallControlInstance(const dsea::callcontrol::sdk::SoftphoneInfoSt& f_softphoneInfo,
60         dsea::callcontrol::sdk::ICallControlListener& f_callControlListener,
61         dsea::callcontrol::sdk::ILogger& f_logger,
62         bool f_bEnableVerboseLogging);
63 #else
64 std::unique_ptr<dsea::callcontrol::sdk::ICallControlInterface> CreateCallControlInstance(const
65     dsea::callcontrol::sdk::SoftphoneInfoSt& f_softphoneInfo,
66     dsea::callcontrol::sdk::ICallControlListener& f_callControlListener,
67     dsea::callcontrol::sdk::ILogger& f_logger,
68     bool f_bEnableVerboseLogging);
69 #endif
70 #endif

```

4.5 ICallControlListener.h File Reference

Contains the declarations for the ICallControlListener interface.

```

#include "DeviceInfo.h"
#include <vector>
#include <string>

```

Classes

- class `dsea::callcontrol::sdk::ICallControlListener`

Interface for CallControlListener. This is a listener interface to be implemented by the softphone and is used for notifying events from call control sdk to softphone.

Enumerations

- enum class `dsea::callcontrol::sdk::LoginStatusValue` {
`LoggedOut` , `LoggedInStandalone` , `LoggedInDSEASDK` , `ConnectingToDSEASDK` ,
`ErrorUnsupportedDSEAVersion` }

Enumeration defining the status of the login operation.

4.5.1 Detailed Description

Contains the declarations for the `ICallControlListener` interface.

Author

EPOS

Copyright

Copyright 2021 EPOS. All rights reserved.

Date

2021-02-23

4.5.2 Enumeration Type Documentation

4.5.2.1 LoginStatusValue `enum class dsea::callcontrol::sdk::LoginStatusValue` [strong]

Enumeration defining the status of the login operation.

Enumerator

<code>LoggedOut</code>	There is no call control in the logged out state, invoke <code>SoftphoneEventHandler::SendLoginEvent()</code> to initiate login
<code>LoggedInStandalone</code>	The call control is enabled and is standalone without DSEA SDK
<code>LoggedInDSEASDK</code>	The call control is enabled and is performed by DSEA SDK
<code>ConnectingToDSEASDK</code>	Detected DSEA SDK and trying to reestablish the connection. Call Control will be disabled till status changes to <code>LoggedInDSEASDK</code>
<code>ErrorUnsupportedDSEAVersion</code>	The installed DSEA SDK version is not supported and should be replaced by the latest version

4.6 ICallControlListener.h

[Go to the documentation of this file.](#)

1

```

11 #ifndef ICALLCONTROLLISTENER_H
12 #define ICALLCONTROLLISTENER_H
13
14 #include "DeviceInfo.h"
15
16 #include <vector>
17 #include <string>
18
19 namespace dsea::callcontrol::sdk
20 {
21     enum class LoginStatusValue
22     {
23         LoggedOut,
24         LoggedInStandalone,
25         LoggedInDSEASDK,
26         ConnectingToDSEASDK,
27         ErrorUnsupportedDSEAVersion
28     };
29
30     class ICallControlListener
31     {
32     public:
33         virtual ~ICallControlListener() = default;
34
35         virtual void LoginStatusChanged(LoginStatusValue f_bLoginStatus) = 0;
36
37         virtual void HeadsetAvailable(const StDeviceInfo& f_deviceInfo) = 0;
38
39         virtual void HeadsetRemoved(const StDeviceInfo& f_deviceInfo) = 0;
40
41         virtual void HeadsetConnected(const StDeviceInfo& f_deviceInfo) = 0;
42
43         virtual void HeadsetDisconnected(const StDeviceInfo& f_deviceInfo) = 0;
44
45         virtual void AcceptCall(const std::string& f_strCallId) = 0;
46
47         virtual void HoldCall(const std::string& f_strCallId) = 0;
48
49         virtual void ResumeCall(const std::string& f_strCallId) = 0;
50
51         virtual void EndCall(const std::string& f_strCallId) = 0;
52
53         virtual void RejectCall(const std::string& f_strCallId) = 0;
54
55         virtual void RequestMuteStatus() = 0;
56
57         virtual void SetMuteStatus(bool f_bMuteStatus) = 0;
58
59         virtual void OffHook(const std::string& f_strInstanceId) = 0;
60
61         virtual void Redial(const std::string& f_strInstanceId) = 0;
62
63         virtual void ResponseSystemInformation(const std::string& f_strProductName, const std::string&
64             f_strVersion) = 0;
65     };
66 }
67 #endif

```

4.7 ILogger.h File Reference

This is interface for softphone logger which will be used to log call control events.

```
#include <string>
```

Classes

- class `dsea::callcontrol::sdk::ILogger`
Interface for logger.

Enumerations

- enum class `dsea::callcontrol::sdk::LogLevel` { `LEVEL_WARNING` , `LEVEL_INFO` , `LEVEL_DEBUG` , `LEVEL_ERROR` }

4.7.1 Detailed Description

This is interface for softphone logger which will be used to log call control events.

Author

EPOS

Copyright

Copyright 2021 EPOS. All rights reserved.

Date

2021-02-23

4.7.2 Enumeration Type Documentation

4.7.2.1 LogLevel `enum class dsea::callcontrol::sdk::LogLevel [strong]`

Enumeration class representing the supported log levels

4.8 ILogger.h

[Go to the documentation of this file.](#)

```
1
12 #ifndef ILOGGER_H
13 #define ILOGGER_H
14
15 #include <string>
16
17 namespace dsea::callcontrol::sdk
18 {
19     enum class LogLevel { LEVEL_WARNING, LEVEL_INFO, LEVEL_DEBUG, LEVEL_ERROR };
20
21     class ILogger
22     {
23     public:
24         virtual ~ILogger() = default;
25
26         virtual void WriteLog(LogLevel f_ecLogLevel, const std::string& f_strLogMsg) = 0;
27     };
28 }
29 #endif
```

4.9 ISoftphoneEventHandler.h File Reference

Contains the declaration for the ISoftphoneEventHandler interface.

```
#include <string>
```

Classes

- class [dsea::callcontrol::sdk::ISoftphoneEventHandler](#)

Interface for SoftphoneEventHandler. This interface provides the methods to send events to SDK from softphone.

4.9.1 Detailed Description

Contains the declaration for the ISoftphoneEventHandler interface.

Author

EPOS

Date

2021-02-23

4.10 ISoftphoneEventHandler.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef ISOFTPHONEEVENTHANDLER_H
11 #define ISOFTPHONEEVENTHANDLER_H
12
13 #include <string>
14
15 namespace dsea::callcontrol::sdk
16 {
17     class ISoftphoneEventHandler
18     {
19     public:
20         virtual ~ISoftphoneEventHandler() = default;
21
22         virtual void ResetSoftphoneHandler() = 0;
23
24         virtual void SendLoginEvent() = 0;
25
26         virtual void SendLogoutEvent() = 0;
27
28         virtual void SendOutgoingCallEvent(const std::string& f_strCallId) = 0;
29
30         virtual void SendIncomingCallEvent(const std::string& f_strCallId) = 0;
31
32         virtual void SendActiveCallEvent(const std::string& f_strCallId) = 0;
33
34         virtual void SendHoldCallEvent(const std::string& f_strCallId) = 0;
35
36         virtual void SendEndCallEvent(const std::string& f_strCallId) = 0;
37
38         virtual void SendActiveDeviceChanged(bool f_isSupportedDevice, const std::string&
39 f_strDeviceInstanceId) = 0;
40
41         virtual void SendMuteStatus(bool f_bMuteStatus) = 0;
42
43         virtual void RequestSystemInformation() = 0;
44
45         virtual void SetPresentingContentStatus(bool f_bStatus) = 0;
46
47         virtual void SetIMNotificationEvent() = 0;
48
49         virtual void SetPresenceStatus(const std::string& f_strPresence) = 0;
50
51         virtual void SetDNDStatus(bool f_bStatus) = 0;
52     };
53 }
54 #endif
```

Index

- `~ICallControlInterface`
 - `dsea::callcontrol::sdk::ICallControlInterface`, 3
- `AcceptCall`
 - `dsea::callcontrol::sdk::ICallControlListener`, 6
- `ConnectingToDSEASDK`
 - `ICallControlListener.h`, 23
- `CreateCallControlInstance`
 - `ICallControlInterface.h`, 21
- `DeviceInfo.h`, 18, 19
 - `StDeviceInfo`, 19
- `dsea::callcontrol::sdk::DeviceInfo`, 2
 - `m_eHeadsetRole`, 2
 - `m_iProductId`, 2
 - `m_iVendorId`, 2
 - `m_strDeviceFriendlyName`, 2
 - `m_strDeviceName`, 3
 - `m_strInstanceId`, 3
 - `m_strSerialNo`, 3
- `dsea::callcontrol::sdk::ICallControlInterface`, 3
 - `~ICallControlInterface`, 3
 - `GetSoftphoneEventHandler`, 4
 - `GetVersion`, 4
- `dsea::callcontrol::sdk::ICallControlListener`, 4
 - `AcceptCall`, 6
 - `EndCall`, 6
 - `HeadsetAvailable`, 6
 - `HeadsetConnected`, 7
 - `HeadsetDisconnected`, 7
 - `HeadsetRemoved`, 7
 - `HoldCall`, 8
 - `LoginStatusChanged`, 8
 - `OffHook`, 8
 - `Redial`, 9
 - `RejectCall`, 9
 - `RequestMuteStatus`, 9
 - `ResponseSystemInformation`, 10
 - `ResumeCall`, 10
 - `SetMuteStatus`, 10
- `dsea::callcontrol::sdk::ILogger`, 11
 - `WriteLog`, 11
- `dsea::callcontrol::sdk::ISoftphoneEventHandler`, 12
 - `RequestSystemInformation`, 13
 - `SendActiveCallEvent`, 13
 - `SendActiveDeviceChanged`, 13
 - `SendEndCallEvent`, 14
 - `SendHoldCallEvent`, 14
 - `SendIncomingCallEvent`, 14
 - `SendLoginEvent`, 15
 - `SendLogoutEvent`, 15
 - `SendMuteStatus`, 15
 - `SendOutgoingCallEvent`, 15
 - `SetDNDStatus`, 16
 - `SetIMNotificationEvent`, 16
 - `SetPresenceStatus`, 16
 - `SetPresentingContentStatus`, 17
- `dsea::callcontrol::sdk::SoftphoneInfoSt`, 17
- `EndCall`
 - `dsea::callcontrol::sdk::ICallControlListener`, 6
- `ErrorUnsupportedDSEAVersion`
 - `ICallControlListener.h`, 23
- `GetSoftphoneEventHandler`
 - `dsea::callcontrol::sdk::ICallControlInterface`, 4
- `GetVersion`
 - `dsea::callcontrol::sdk::ICallControlInterface`, 4
- `HeadsetAvailable`
 - `dsea::callcontrol::sdk::ICallControlListener`, 6
- `HeadsetConnected`
 - `dsea::callcontrol::sdk::ICallControlListener`, 7
- `HeadsetDisconnected`
 - `dsea::callcontrol::sdk::ICallControlListener`, 7
- `HeadsetRemoved`
 - `dsea::callcontrol::sdk::ICallControlListener`, 7
- `HoldCall`
 - `dsea::callcontrol::sdk::ICallControlListener`, 8
- `ICallControlInterface.h`, 20, 21
 - `CreateCallControlInstance`, 21
- `ICallControlListener.h`, 22, 23
 - `ConnectingToDSEASDK`, 23
 - `ErrorUnsupportedDSEAVersion`, 23
 - `LoggedInDSEASDK`, 23
 - `LoggedInStandalone`, 23
 - `LoggedOut`, 23
 - `LoginStatusValue`, 23
- `ILogger.h`, 24, 25
 - `LogLevel`, 25
- `ISoftphoneEventHandler.h`, 25, 26
- `LoggedInDSEASDK`
 - `ICallControlListener.h`, 23
- `LoggedInStandalone`
 - `ICallControlListener.h`, 23
- `LoggedOut`
 - `ICallControlListener.h`, 23
- `LoginStatusChanged`
 - `dsea::callcontrol::sdk::ICallControlListener`, 8
- `LoginStatusValue`
 - `ICallControlListener.h`, 23
- `LogLevel`
 - `ILogger.h`, 25
- `m_eHeadsetRole`
 - `dsea::callcontrol::sdk::DeviceInfo`, 2
- `m_iProductId`
 - `dsea::callcontrol::sdk::DeviceInfo`, 2
- `m_iVendorId`
 - `dsea::callcontrol::sdk::DeviceInfo`, 2

m_strDeviceFriendlyName
dsea::callcontrol::sdk::DeviceInfo, 2

m_strDeviceName
dsea::callcontrol::sdk::DeviceInfo, 3

m_strInstanceId
dsea::callcontrol::sdk::DeviceInfo, 3

m_strSerialNo
dsea::callcontrol::sdk::DeviceInfo, 3

OffHook
dsea::callcontrol::sdk::ICallControlListener, 8

Redial
dsea::callcontrol::sdk::ICallControlListener, 9

RejectCall
dsea::callcontrol::sdk::ICallControlListener, 9

RequestMuteStatus
dsea::callcontrol::sdk::ICallControlListener, 9

RequestSystemInformation
dsea::callcontrol::sdk::ISoftphoneEventHandler, 13

ResponseSystemInformation
dsea::callcontrol::sdk::ICallControlListener, 10

ResumeCall
dsea::callcontrol::sdk::ICallControlListener, 10

SendActiveCallEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 13

SendActiveDeviceChanged
dsea::callcontrol::sdk::ISoftphoneEventHandler, 13

SendEndCallEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 14

SendHoldCallEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 14

SendIncomingCallEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 14

SendLoginEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 15

SendLogoutEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 15

SendMuteStatus
dsea::callcontrol::sdk::ISoftphoneEventHandler, 15

SendOutgoingCallEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 15

SetDNDStatus
dsea::callcontrol::sdk::ISoftphoneEventHandler, 16

SetIMNotificationEvent
dsea::callcontrol::sdk::ISoftphoneEventHandler, 16

SetMuteStatus
dsea::callcontrol::sdk::ICallControlListener, 10

SetPresenceStatus
dsea::callcontrol::sdk::ISoftphoneEventHandler, 16

SetPresentingContentStatus
dsea::callcontrol::sdk::ISoftphoneEventHandler, 17

StDeviceInfo
DeviceInfo.h, 19

WriteLog
dsea::callcontrol::sdk::ILogger, 11