Google Developers

Google Tal X 搜索

melochale@gmail.com
退出帐户

产品 Google Talk for Developers

SessionClient Class

This is a pure virtual base class that translates between XMPP (for network transmission) and **XmppSessionMessage** and **SessionDescription** objects (used internally by libjingle). You should override this class with logic specific to your session type. In particular, you should override the methods declared by this class.

You should also define a string namespace to use as the session name when calling **SessionManager::CreateSession**. This should be a unique identifier for the **SessionClass** type. libjingle defines two values: NS_TUNNEL and NS_PHONE (both #defined values).

SessionClient translates session message stanzas between XMPP and **SessionDescription** objects. Session description information is customized for different types of sessions (for example, codecs for voice chat and file names for file transfer applications), so subclasses must be specifically designed to be able to read or create these messages. See Extending Session Client for more information about extending this object to support new session types.

The application instantiates this object. The methods shown are not thread-safe, and can be called on any thread, but you should call them from the signaling thread.

Syntax

```
class SessionClient
```

Methods

The following table lists public and protected (shaded rows) member methods for **SessionClient**.

Name	Description
virtual SessionDescription* **CreateSessionDescription**(const buzz::XmlElement **element*)=0	Converts an XMPP element into a **SessionDescription** object for internal libjingle use.
virtual void **OnSessionCreate**(Session * *session*, bool *received_initiate*) = 0	Called by the **SessionManager** object when a **Session** object is created, either explicitly as part of an outgoing connection request, or implicitly as part of an incoming connection request. Your implementation should determine whether the session represents an incoming or an outgoing connection request. If this is incoming, *received_initiate* will be True, and your application should connect to the **Session**'s signals and perform any other session-specific tasks, such as adding it to a list of active sessions and instantiating helper objects or resources. The voice chat example creates a new **Call** object when this method is received. It should also send a notification to the user about the incoming call, to enable them to call **Session::Accept** or **Session::Reject** to accept or reject the connection request.
virtual void	Called when a **Session** object is destroyed. You should use this to remove any

OnSessionDestroy(Session * *session*) = 0	pointers you hold to the object, and release any resources you associated with the session. This includes destroying the **Transport** object associated with the Session by calling **Session::DestroySocket**.
SessionClient(SessionManager **psm*)	Constructor for the object.
~SessionClient	Destructor for the object.
virtual buzz::XmlElement* **TranslateSessionDescription**(const SessionDescription **description*)=0	Converts a **SessionDescription** object into an XML stanza to send out. Subclasses must implement this in order to parse out custom information required for each type of session (for example, audio codecs for a voice chat, file names for a file transfer).

Attributes: public

Declaration file: talk/p2p/base/sessionclient.h

Last updated 三月 23, 2012.