MCloud

Generated by Doxygen 1.8.6

Tue Jul 15 2014 09:26:09

Contents

Chapter 1

Deprecated List

Global mcloudAddFlowDescription (MCloud ∗cloudP, const char ∗password, int logging, const char ∗language, const char ∗name, const char ∗description)

This function has to be called after an MCloud object has been created and before connecting to the MCloud. A client can add more than one flow being just translations of the same descriptions. Therefore, the password and logging has to be the same over all flows. This call is deprecated and will be removed in future versions of the API

Global mcloudPacketAddAudio (MCloud ∗cloudP, MCloudPacket ∗p, const char ∗startTime, const char ∗stopTime, const char ∗fingerPrint, const short ∗sampleA, int sampleN, int isFinal)

This function can be used only for PCM s16le 16KHz audio both as input and as packet content!

Global mcloudPacketInitFromAudio (MCloud ∗cloudP, const char ∗startTime, const char ∗stopTime, const char ∗fingerPrint, const short ∗sampleA, int sampleN, int isFinal)

This function can be used only for PCM s16le 16KHz audio both as input and as packet content!

Global mcloudSetAttr (MCloud ∗cloudP, MCloudAttribute attr, const void ∗value)

Set the value of an attribute.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Data Structure Documentation

4.1 MCloudPacket_S Struct Reference

`#include <MCloud.h>`

Data Fields

- MCloudType packetType
  Type of the packet.
- MCloudType dataType
  Type of the data included if packet is of type MCloudData.
- char * sessionID
  The current sessionID which the packet belongs to.
- char * streamID
  The current streamID which the packet belongs to.
- char * fingerPrint
  Fingerprint of the packet.
- char * creator
  Name of the creator of the packet.
- char * start
  Human readable time stamp identifying the start time of the packet "dd/MM/YY-hh:mm:ss.mss".
- char * stop
  Human readable time stamp identifying the end time of the packet "dd/MM/YY-hh:mm:ss.mss".
- unsigned int startOffset
  Start time offset in ms relative to the beginning of the stream.
- unsigned int stopOffset
  Stop time offset in ms relative to the beginning of the stream.
- char * statusDescription
  Optional detailed status description in case of status messages.
- char * userID
- char * cmType
  The current userID.
- char * revision
  The CM-type of the packet.
- char * xmlString
  The current revision of the packet.
- xmlDoc * doc
  Reference to the whole XML document (libXML2 xmlDoc)

4.1.1 Field Documentation

4.1.1.1 char* MCloudPacket_S::cmType

The current userID.

4.1.1.2 char* MCloudPacket_S::creator

Name of the creator of the packet.

4.1.1.3 MCloudType MCloudPacket_S::dataType

Type of the data included if packet is of type MCloudData.

4.1.1.4 xmlDoc* MCloudPacket_S::doc

Reference to the whole XML document (libXML2 xmlDoc)

4.1.1.5 char* MCloudPacket_S::fingerPrint

Fingerprint of the packet.

4.1.1.6 MCloudType MCloudPacket_S::packetType

Type of the packet.

4.1.1.7 char* MCloudPacket_S::revision

The CM-type of the packet.

4.1.1.8 char* MCloudPacket_S::sessionID

The current sessionID which the packet belongs to.

4.1.1.9 char* MCloudPacket_S::start

Human readable time stamp identifying the start time of the packet "dd/MM/YY-hh:mm:ss.mss".

4.1.1.10 unsigned int MCloudPacket_S::startOffset

Start time offset in ms relative to the beginning of the stream.

4.1.1.11 char* MCloudPacket_S::statusDescription

Optional detailed status description in case of status messages.

4.1.1.12 char* MCloudPacket_S::stop

Human readable time stamp identifying the end time of the packet "dd/MM/YY-hh:mm:ss.mss".

4.1.1.13 unsigned int MCloudPacket_S::stopOffset

Stop time offset in ms relative to the beginning of the stream.

4.1.1.14 char* MCloudPacket_S::streamID

The current streamID which the packet belongs to.

4.1.1.15 char* MCloudPacket_S::userID

4.1.1.16 char* MCloudPacket_S::xmlString

The current revision of the packet.

Raw XML string

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

- MCloud.h

4.2 MCloudWordToken_S Struct Reference

```
#include <MCloud.h>
```

Data Fields

- int index
  The token index.
- char * internal
  The internal form of the token.
- char * written
  The written form of the token (can be NULL)
- char * spoken
  The spoken form of the token (optional)
- float confidence
  The confidence value in the interval [0,1].
- unsigned int startTime
  The start time [ms] relative to the start of the stream.
- unsigned int stopTime
  The end time [ms] relative to the start of the stream.
- int isFiller
  This value is set to 1, if the token is a filler token and not a regular word.

4.2.1 Field Documentation

4.2.1.1 float MCloudWordToken_S::confidence

The confidence value in the interval [0,1].

4.2.1.2 int MCloudWordToken_S::index

The token index.

4.2.1.3 char* MCloudWordToken_S::internal

The internal form of the token.

4.2.1.4 int MCloudWordToken_S::isFiller

This value is set to 1, if the token is a filler token and not a regular word.

4.2.1.5 char* MCloudWordToken_S::spoken

The spoken form of the token (optional)

4.2.1.6 unsigned int MCloudWordToken_S::startTime

The start time [ms] relative to the start of the stream.

4.2.1.7 unsigned int MCloudWordToken_S::stopTime

The end time [ms] relative to the start of the stream.

4.2.1.8 char* MCloudWordToken_S::written

The written form of the token (can be NULL)

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

- MCloud.h

Chapter 5

File Documentation

5.1 MCloud.h File Reference

```
#include <libxml/tree.h>
```

Data Structures

- struct MCloudPacket_S
- struct MCloudWordToken_S

Macros

- #define SHAREDDLL
- #define mcloudERROR(...) mcloudMsgHandler(__FILE__, __LINE__, 1, __VA_ARGS__)
- #define mcloudWARN(...) mcloudMsgHandler(__FILE__, __LINE__, 2, __VA_ARGS__)
- #define mcloudINFO(...) mcloudMsgHandler(__FILE__, __LINE__, 3, __VA_ARGS__)

Typedefs

- typedef struct MCloudPacket_S MCloudPacket
- typedef struct MCloudWordToken_S MCloudWordToken
- typedef struct MCloud_S MCloud
- typedef int MCloudCallbackFct (MCloud *cloudP, void *userData)

  MCloud general callback function type used for finalize, break, and error.
- typedef int MCloudPacketCallbackFct (MCloud *cloudP, MCloudPacket *p, void *userData)

  MCloud packet callback function type used for data and init.

Enumerations

- enum S2S_Result { S2S_Success = 0, S2S_Error = 1 }
- enum MCloudType {
  MCloudModeWorker = 1, MCloudModeClient, MCloudData, MCloudDone,
  MCloudError, MCloudReset, MCloudFlush, MCloudAudio,
  MCloudText, MCloudImage, MCloudMixed, MCloudBinary,
  MCloudSendingQueue, MCloudProcessingQueue, MCloudCustomization }

- enum MCloudAttribute {
  MCloudA_sAudioCodec, MCloudA_iSampleRate, MCloudA_iSampleSize, MCloudA_iChannelN,
  MCloudA_iBitRate }
- enum MCloudCodec {
  MCloudAC_UNK = 0, MCloudAC_PCM, MCloudAC_FLAC, MCloudAC_SPEEX,
  MCloudAC_OPUS }

Functions

- SHAREDDLL void mcloudMsgHandler (const char *file, int line, int type, const char *format,...)

  Message handler that should be only used with the macros defined above.
- SHAREDDLL MCloudWordToken * mcloudWordTokenArrayCreate (int n)

  Creates an array of MCloudWordTokens.
- SHAREDDLL void mcloudWordTokenArrayFree (MCloudWordToken *tokenA, int n)

  Free an array of MCloudWordTokens.
- SHAREDDLL MCloudPacket * mcloudPacketInitFromText (MCloud *cloudP, const char *startTime, const char *stopTime, unsigned int startOffset, unsigned int stopOffset, const char *fingerPrint, const char *text)

  Initialize a new packet from text for sending.
- SHAREDDLL MCloudPacket * mcloudPacketInitFromWordTokenA (MCloud *cloudP, const char *startTime, const char *stopTime, unsigned int startOffset, unsigned int stopOffset, const char *fingerPrint, MCloudWordToken *tokenA, int tokenN)

  Initialize a new packet from a MCloudWordToken array for sending.
- SHAREDDLL MCloudPacket * mcloudPacketInitFromAudio (MCloud *cloudP, const char *startTime, const char *stopTime, const char *fingerPrint, const short *sampleA, int sampleN, int isFinal)

  Initialize a new packet from audio for sending.
- SHAREDDLL MCloudPacket * mcloudPacketInitFromBinary (MCloud *cP, const char *startTime, const char *stopTime, const char *fingerPrint, const char *filename, const char *mimetype, const uint8_t *bytes, int bytesN, int last)

  Initialize a new packet from binary for sending.
- MCloudPacket * mcloudPacketInitFromImage (MCloud *cloudP, const char *startTime, const char *stopTime, const char *fingerPrint, int width, int height, const char *format, const char *buffer, int bufferN)

  Initialize a new packet from image for sending.
- SHAREDDLL MCloudPacket * mcloudPacketAddAudio (MCloud *cloudP, MCloudPacket *p, const char *startTime, const char *stopTime, const char *fingerPrint, const short *sampleA, int sampleN, int isFinal)

  Add audio to an existing package for sending. Packet type will be changed to "mixed".
- SHAREDDLL void mcloudPacketDeinit (MCloudPacket *p)

  Free a packet.
- SHAREDDLL S2S_Result mcloudPacketGetText (MCloud *cloudP, MCloudPacket *p, char **text)

  Convenient function for extracting the string embedded in <text></text>.
- SHAREDDLL S2S_Result mcloudPacketReplaceText (MCloud *cloudP, MCloudPacket *p, char *text)

  Convenient function for replacing the string embedded in <text></text>.
- SHAREDDLL S2S_Result mcloudPacketGetWordTokenA (MCloud *cloudP, MCloudPacket *p, MCloudWordToken **tokenA, int *tokenN)

  Convenient function for extracting a word token array embedded in <wordtokens></wordtokens>.
- SHAREDDLL S2S_Result mcloudPacketGetAudio (MCloud *cloudP, MCloudPacket *p, short **sampleA, int *sampleN)

  Convenient function for extracting the data embedded in <audio></audio>.
- SHAREDDLL S2S_Result mcloudPacketGetBinary (MCloud *cP, MCloudPacket *p, uint8_t **bA, int *bN, char **filename, char **mimetype, int *last)

  Convenient function for extracting the data embedded in <audio></audio>
- SHAREDDLL MCloud * mcloudCreate (const char *name, int mode)

  Create an MCloud object with a given name and mode.

- SHAREDDLL void mcloudFree (MCloud ∗cloudP)

  Free an MCloud object.
- SHAREDDLL S2S_Result mcloudGetAttr (MCloud ∗cloudP, MCloudAttribute attr, void ∗value)

  Get the value of an attribute.
- SHAREDDLL S2S_Result mcloudSetAttr (MCloud ∗cloudP, MCloudAttribute attr, const void ∗value)

  Set the value of an attribute.
- SHAREDDLL S2S_Result mcloudSetAudioEncoder (MCloud ∗cp, MCloudCodec codec, int sampleRate, int bitRate, int channels)

  Set the audio codec.
- SHAREDDLL S2S_Result mcloudSetAudioEncoder2 (MCloud ∗cp, char ∗codec, int sampleRate, int bitRate, int channels)

  Set the audio codec.
- SHAREDDLL S2S_Result mcloudConnect (MCloud ∗cloudP, const char ∗host, int port)

  Connect to the MCloud server running on the host at port given.
- SHAREDDLL S2S_Result mcloudDisconnect (MCloud ∗cloudP)

  Disconnect from the MCloud server.
- SHAREDDLL S2S_Result mcloudAddService (MCloud ∗cloudP, const char ∗name, const char ∗service, const char ∗inputFingerPrint, const char ∗inputType, const char ∗outputFingerPrint, const char ∗outputType, const char ∗specifier)

  Add a service description of a worker to an MCloud object.
- SHAREDDLL S2S_Result mcloudAddFlowDescription2 (MCloud ∗cloudP, const char ∗username, const char ∗password, int logging, const char ∗language, const char ∗name, const char ∗description)

  Add a flow description of a client to an MCloud object.
- SHAREDDLL S2S_Result mcloudAddFlowDescription (MCloud ∗cloudP, const char ∗password, int logging, const char ∗language, const char ∗name, const char ∗description)

  Add a flow description of a client to an MCloud object.
- SHAREDDLL S2S_Result mcloudAnnounceOutputStream (MCloud ∗cloudP, const char ∗type, const char ∗fingerPrint, const char ∗streamID, const char ∗specifier)

  Announce an output stream of a client to an MCloud object.
- SHAREDDLL S2S_Result mcloudRequestInputStream (MCloud ∗cloudP, const char ∗type, const char ∗fingerPrint, const char ∗streamID, char ∗info, int infoN)

  Request an input stream of a client from an MCloud object.
- SHAREDDLL S2S_Result mcloudRequestForDisplay (MCloud ∗cloudP)

  Request the display of an output stream.
- SHAREDDLL S2S_Result mcloudWaitForClient (MCloud ∗cloudP, char ∗∗streamID)

  Wait for a service request to process.
- SHAREDDLL MCloudPacket ∗ mcloudGetNextPacket (MCloud ∗cloudP)

  Wait for the next data package.
- SHAREDDLL S2S_Result mcloudSendPacket (MCloud ∗cloudP, MCloudPacket ∗p)

  Send a packet.
- SHAREDDLL S2S_Result mcloudSendPacketAsync (MCloud ∗cloudP, MCloudPacket ∗p, void ∗userData)

  Send a packet asynchronously.
- SHAREDDLL S2S_Result mcloudSendBinaryFile (MCloud ∗cP, FILE ∗f, int chunkSize, char ∗filename, char ∗mimeType, char ∗fingerPrint)

  Convenience function for sending the content of a whole file.
- SHAREDDLL S2S_Result mcloudSendBinaryFileAsync (MCloud ∗cP, FILE ∗f, int chunkSize, char ∗filename, char ∗mimeType, char ∗fingerPrint, void ∗userData)

  Convenience function for sending the content of a whole file asyncronously.
- SHAREDDLL S2S_Result mcloudSendDone (MCloud ∗cloudP)

  Inform a client or a worker that there is no more data to receive.
- SHAREDDLL S2S_Result mcloudSendError (MCloud ∗cloudP, const char ∗description)

  Inform a client or a worker that an error occurred during processing.

- SHAREDDLL S2S_Result mcloudSendFlush (MCloud *cloudP)

  Inform subsequent worker to flush their output buffers.
- SHAREDDLL S2S_Result mcloudProcessDataAsync (MCloud *cloudP, MCloudPacket *p, void *userData)

  Process received packages asynchronously.
- SHAREDDLL int mcloudPending (MCloud *cloudP, MCloudType queueType)

  Return number of pending packages in queue.
- SHAREDDLL S2S_Result mcloudWaitFinish (MCloud *cloudP, MCloudType queueType, int done)

  Wait until all pending packages have been processed/ sent.
- SHAREDDLL S2S_Result mcloudBreak (MCloud *cloudP, MCloudType queueType)

  Stop processing, sending pending packages immediately, and reset queue.
- SHAREDDLL void mcloudSetInitCallback (MCloud *cloudP, MCloudPacketCallbackFct *callback, void *user-Data)

  Set an init callback function.
- SHAREDDLL void mcloudSetFinalizeCallback (MCloud *cloudP, MCloudCallbackFct *callback, void *user-Data)

  Set a finalize callback function.
- SHAREDDLL void mcloudSetBreakCallback (MCloud *cloudP, MCloudType queueType, MCloudCallbackFct *callback, void *userData)

  Set a break callback function.
- SHAREDDLL void mcloudSetErrorCallback (MCloud *cloudP, MCloudType queueType, MCloudCallbackFct *callback, void *userData)

  Set an error callback function.
- SHAREDDLL void mcloudSetDataCallback (MCloud *cloudP, MCloudPacketCallbackFct *callback)

  Set a data callback function.
- SHAREDDLL void mcloudSetCustomizationCallback (MCloud *cP, MCloudPacketCallbackFct *callback, void *userData)

  Set a customization callback function.
- SHAREDDLL MCloudPacket * mcloudPacketInitFromCmGet (MCloud *cP, const char *sessionId, const char *userId, const char *fingerPrint, const char *type, const char *revision)

  Initialize a new packet from a CmGet for sending.
- SHAREDDLL char * base64_encode (const char *data, size_t input_length, size_t *output_length)

  Convenient function for encoding the data in base64.
- SHAREDDLL char * base64_decode (const char *data, size_t input_length, size_t *output_length)

  Convenient function for decoding the data from base64.
- SHAREDDLL char * url_encode (const char *str)

  Convenient function that returns a url-encoded version of a string.
- SHAREDDLL char * url_decode (const char *str)

  Convenient function that returns a url-decoded version of a string.

5.1.1 Macro Definition Documentation

5.1.1.1 #define mcloudERROR(...) mcloudMsgHandler(__FILE__, __LINE__, 1, __VA_ARGS__)

5.1.1.2 #define mcloudINFO(...) mcloudMsgHandler(__FILE__, __LINE__, 3, __VA_ARGS__)

5.1.1.3 #define mcloudWARN(...) mcloudMsgHandler(__FILE__, __LINE__, 2, __VA_ARGS__)

5.1.1.4 #define SHAREDDLL

5.1.2 Typedef Documentation

5.1.2.1 typedef struct MCloud_S MCloud

MCloud object.

5.1.2.2 typedef int MCloudCallbackFct(MCloud *cloudP, void *userData)

MCloud general callback function type used for finalize, break, and error.

Parameters

in	*cloudP*	reference to an MCloud object
in	*userData*	user data

Returns

0 if success

5.1.2.3 typedef struct MCloudPacket_S MCloudPacket

5.1.2.4 typedef int MCloudPacketCallbackFct(MCloud *cloudP, MCloudPacket *p, void *userData)

MCloud packet callback function type used for data and init.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	packet containing the data to process
in	*userData*	user data

Returns

0 if success

5.1.2.5 typedef struct MCloudWordToken_S MCloudWordToken

5.1.3 Enumeration Type Documentation

5.1.3.1 enum MCloudAttribute

Enumerator

MCloudA_sAudioCodec Audio codec type `RPCM (raw PCM), SPEEX (Speex), OPUS (Opus), FLAC (Flac)` (default: RPCM)

MCloudA_iSampleRate input/ output sample rate in Hz (get/set) (default: 16000)

MCloudA_iSampleSize input/ output sample size in bits (get/set) (default: 16)

MCloudA_iChannelN input/ output number of channels (get/set) (default: 1)

MCloudA_iBitRate input / output bit rate in bits/sec

5.1.3.2 enum MCloudCodec

Defines the supported encoding

Enumerator

MCloudAC_UNK

MCloudAC_PCM

MCloudAC_FLAC

MCloudAC_SPEEX

MCloudAC_OPUS

5.1.3.3 enum MCloudType

Enumerator

MCloudModeWorker worker mode

MCloudModeClient client mode

MCloudData message of type data

MCloudDone status message of type done

MCloudError status message of type error

MCloudReset status message of type reset

MCloudFlush status message of type flush

MCloudAudio data packet of type audio

MCloudText data packet of type text

MCloudImage data packet of type image

MCloudMixed data packet of type mixed

MCloudBinary data packet of type binary

MCloudSendingQueue sending queue

MCloudProcessingQueue receiving/ processing queue

MCloudCustomization message with customizations

5.1.3.4 enum S2S_Result

Enumerator

S2S_Success success

S2S_Error error

5.1.4 Function Documentation

5.1.4.1 SHAREDDLL char* base64_decode (const char * *data*, size_t *input_length*, size_t * *output_length*)

Convenient function for decoding the data from base64.

Parameters

in	*data*	data to be decoded
in	*input_length*	length of input data stream
in	*output_length*	length of output data stream

Returns

decoded data

5.1.4.2 SHAREDDLL char* base64_encode (const char * *data*, size_t *input_length*, size_t * *output_length*)

Convenient function for encoding the data in base64.

Parameters

in	*data*	data to be encoded
in	*input_length*	length of input data stream
in	*output_length*	length of output data stream

Returns

encoded data

5.1.4.3 SHAREDDLL S2S_Result mcloudAddFlowDescription (MCloud * *cloudP*, const char * *password*, int *logging*, const char * *language*, const char * *name*, const char * *description*)

Add a flow description of a client to an MCloud object.

Deprecated This function has to be called after an MCloud object has been created and before connecting to the MCloud. A client can add more than one flow being just translations of the same descriptions. Therefore, the password and logging has to be the same over all flows. This call is deprecated and will be removed in future versions of the API

Parameters

in	*cloudP*	reference to an MCloud object
in	*password*	an optional password which has to be used in order to subscribe to this flow (display server), NULL for no password
in	*logging*	if set to 0, the flow will not be logged in the database
in	*language*	descriptive language identifier of the flow, e.g. English
in	*name*	name of the flow, e.g. title of a talk
in	*description*	additional description of the flow, e.g. abstract

Returns

S2S_Success if no error occurs

5.1.4.4 SHAREDDLL S2S_Result mcloudAddFlowDescription2 (MCloud * *cloudP*, const char * *username*, const char * *password*, int *logging*, const char * *language*, const char * *name*, const char * *description*)

Add a flow description of a client to an MCloud object.

This function has to be called after an MCloud object has been created and before connecting to the MCloud. A client can add more than one flow being just translations of the same descriptions. Therefore, the password and logging has to be the same over all flows.

Parameters

in	*cloudP*	reference to an MCloud object
in	*username*	an optional username which has to be used in order to subscribe to this flow (display server), NULL for no username
in	*password*	an optional password which has to be used in order to subscribe to this flow (display server), NULL for no password
in	*logging*	if set to 0, the flow will not be logged in the database
in	*language*	descriptive language identifier of the flow, e.g. English

in	*name*	name of the flow, e.g. title of a talk
in	*description*	additional description of the flow, e.g. abstract

Returns

S2S_Success if no error occurs

5.1.4.5 SHAREDDLL S2S_Result mcloudAddService (MCloud * *cloudP,* const char * *name,* const char * *service,* const char * *inputFingerPrint,* const char * *inputType,* const char * *outputFingerPrint,* const char * *outputType,* const char * *specifier*)

Add a service description of a worker to an MCloud object.

This function has to be called after an MCloud object has been created and before connecting to the MCloud.

Parameters

in	*cloudP*	reference to an MCloud object
in	*name*	name of the worker
in	*service*	name of the service (asr, smt, tts, ...)
in	*inputFingerPrint*	service input finger print
in	*inputType*	data input type (audio, text)
in	*outputFinger-Print*	service output finger print
in	*outputType*	data output type (audio, text)
in	*specifier*	an additional specifier, i.e. a speaker identifier

Returns

S2S_Success if no error occurs

5.1.4.6 SHAREDDLL S2S_Result mcloudAnnounceOutputStream (MCloud * *cloudP,* const char * *type,* const char * *fingerPrint,* const char * *streamID,* const char * *specifier*)

Announce an output stream of a client to an MCloud object.

This function has to be called after the client has been connected to the MCloud.

Parameters

in	*cloudP*	reference to an MCloud object
in	*type*	data type (audio, text, image)
in	*fingerPrint*	finger print of the data stream
in	*streamID*	unique stream identifier
in	*specifier*	an additional specifier, i.e. a speaker identifier

Returns

S2S_Success if no error occurs

5.1.4.7 SHAREDDLL S2S_Result mcloudBreak (MCloud * *cloudP,* MCloudType *queueType*)

Stop processing, sending pending packages immediately, and reset queue.

This function can be used to stop further processing or sending packages in the queue specified.

Parameters

in	*cloudP*	reference to an MCloud object
in	*queueType*	type of queue MCloudSendingQueue, or MCloudProcessingQueue

Returns

S2S_Success if no error occurs

5.1.4.8 SHAREDDLL S2S_Result mcloudConnect (MCloud * *cloudP,* const char * *host,* int *port*)

Connect to the MCloud server running on the host at port given.

This function has to be called after an MCloud object has been created and before waiting for a client or worker.

Parameters

in	*cloudP*	reference to an MCloud object
in	*host*	host name
in	*port*	port number

Returns

S2S_Success if no error occurs

5.1.4.9 SHAREDDLL MCloud* mcloudCreate (const char * *name,* int *mode*)

Create an MCloud object with a given name and mode.

Parameters

in	*name*	descriptive name of the worker or client (used as 'creator' in XML messages)
in	*mode*	working mode, i.e. MCloudModeWorker, or MCloudModeClient

Returns

reference to an MCloud object or NULL if failed

5.1.4.10 SHAREDDLL S2S_Result mcloudDisconnect (MCloud * *cloudP*)

Disconnect from the MCloud server.

Parameters

in	*cloudP*	reference to an MCloud object

Returns

S2S_Success if no error occurs

5.1.4.11 SHAREDDLL void mcloudFree (MCloud * *cloudP*)

Free an MCloud object.

Parameters

in	*cloudP*	reference to an MCloud object

5.1.4.12 SHAREDDLL S2S_Result mcloudGetAttr (MCloud * *cloudP,* MCloudAttribute *attr,* void * *value*)

Get the value of an attribute.

Parameters

in	*cloudP*	reference to an MCloud object
in	*attr*	an MCloud attribute
out	*value*	reference to the returned value

Returns

S2S_Success if no error occurs

5.1.4.13 SHAREDDLL MCloudPacket* mcloudGetNextPacket (MCloud * *cloudP*)

Wait for the next data package.

This function has to be called in order to wait for the next data packet. The function times out after an internally specified amount of time.

Parameters

in	*cloudP*	reference to an MCloud object

Returns

S2S_Success if no error occurs

5.1.4.14 SHAREDDLL void mcloudMsgHandler (const char * *file,* int *line,* int *type,* const char * *format,* ...)

Message handler that should be only used with the macros defined above.

Parameters

in	*file*	source code file name **FILE**
in	*line*	source code line **LINE**
in	*type*	message type
in	*format*	printf format
in	...	variable list of additional arguments

5.1.4.15 SHAREDDLL MCloudPacket* mcloudPacketAddAudio (MCloud * *cloudP,* MCloudPacket * *p,* const char * *startTime,* const char * *stopTime,* const char * *fingerPrint,* const short * *sampleA,* int *sampleN,* int *isFinal*)

Add audio to an existing package for sending. Packet type will be changed to "mixed".

Deprecated This function can be used only for PCM s16le 16KHz audio both as input and as packet content!

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	MCloudPacket to which the audio should be added
in	*startTime*	human readable starting time stamp
in	*stopTime*	human readable ending time stamp
in	*fingerPrint*	finger print of the audio
in	*sampleA*	reference to an array of audio samples
in	*sampleN*	number of samples in array
in	*isFinal*	indicates whether the sample array given is the final one

Returns

reference to the created package or NULL if failed

5.1.4.16 SHAREDDLL void mcloudPacketDeinit (MCloudPacket * *p*)

Free a packet.

Parameters

in	*p*	reference to an MCloud package

5.1.4.17 SHAREDDLL S2S_Result mcloudPacketGetAudio (MCloud * *cloudP*, MCloudPacket * *p*, short ** *sampleA*, int * *sampleN*)

Convenient function for extracting the data embedded in <audio></audio>.

Don't forget to free the memory allocated for sampleA.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	reference to an MCloud package
out	*sampleA*	returned array of samples
out	*sampleN*	returned number of samples in array

Returns

S2S_Success if no error occurred

5.1.4.18 SHAREDDLL S2S_Result mcloudPacketGetBinary (MCloud * *cP*, MCloudPacket * *p*, uint8_t ** *bA*, int * *bN*, char ** *filename*, char ** *mimetype*, int * *last*)

Convenient function for extracting the data embedded in <audio></audio>

Don't forget to free the memory allocated for sampleA.

Parameters

in	*cP*	reference to an MCloud object
in	*p*	reference to an MCloud package
out	*bA*	returned array of binary samples

out	*bN*	returned number of binary samples in array
out	*filename*	return the declared name for the file or NULL if not specified
out	*mimetype*	return the declared mimetype for the file or NULL if not specified
out	*lsat*	1 if this is the last packet for the file, 0 if more packet are incoming, -1 if not specified

Returns

S2S_Success if no error occurred

5.1.4.19 SHAREDDLL S2S_Result mcloudPacketGetText (MCloud * *cloudP,* MCloudPacket * *p,* char ** *text*)

Convenient function for extracting the string embedded in <text></text>.

Don't forget to free the memory allocated for text.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	reference to an MCloud package
out	*text*	returned text

Returns

S2S_Success if no error occurred

5.1.4.20 SHAREDDLL S2S_Result mcloudPacketGetWordTokenA (MCloud * *cloudP,* MCloudPacket * *p,* MCloudWordToken ** *tokenA,* int * *tokenN*)

Convenient function for extracting a word token array embedded in <wordtokens></wordtokens>.

Don't forget to free the memory allocated for tokenA.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	reference to an MCloud package
out	*tokenA*	returned array of word tokens
out	*tokenN*	number of word tokens returned

Returns

S2S_Success if no error occurred

5.1.4.21 SHAREDDLL MCloudPacket* mcloudPacketInitFromAudio (MCloud * *cloudP,* const char * *startTime,* const char * *stopTime,* const char * *fingerPrint,* const short * *sampleA,* int *sampleN,* int *isFinal*)

Initialize a new packet from audio for sending.

Deprecated This function can be used only for PCM s16le 16KHz audio both as input and as packet content!

Parameters

in	*cloudP*	reference to an MCloud object
in	*startTime*	human readable starting time stamp
in	*stopTime*	human readable ending time stamp
in	*fingerPrint*	finger print of the audio
in	*sampleA*	reference to an array of audio samples
in	*sampleN*	number of samples in array
in	*isFinal*	indicates whether the sample array given is the final one

Returns

reference to the created package or NULL if failed

5.1.4.22 SHAREDDLL MCloudPacket* mcloudPacketInitFromBinary (MCloud * *cP,* const char * *startTime,* const char * *stopTime,* const char * *fingerPrint,* const char * *filename,* const char * *mimetype,* const uint8_t * *bytes,* int *bytesN,* int *last*)

Initialize a new packet from binary for sending.

Parameters

in	*cloudP*	reference to an MCloud object
in	*startTime*	human readable starting time stamp (set to NULL if not relevant)
in	*stopTime*	human readable ending time stamp (set to NULL if not relevant)
in	*fingerPrint*	finger print of the binary data
in	*filename*	name of the file being sent (set to NULL if not relevant)
in	*mimetype*	MIME type of the file being sent (set to NULL if not relevant)
in	*bytes*	reference to a buffer of bytes
in	*bytesN*	number of bytes
in	*last*	whether this packet is the last for the current file (i.e. following packets with the same filename should be treated as a different file)

Returns

reference to the created package or NULL if failed

5.1.4.23 SHAREDDLL MCloudPacket* mcloudPacketInitFromCmGet (MCloud * *cP,* const char * *sessionId,* const char * *userId,* const char * *fingerPrint,* const char * *type,* const char * *revision*)

Initialize a new packet from a CmGet for sending.

Parameters

in	*cloudP*	reference to an MCloud object
in	*sessionId*	current sessionID which the packet belongs to
in	*userId*	current userID
in	*fingerPrint*	current finger print
in	*type*	current type
in	*revision*	current revision

Returns

reference to the created package or NULL if failed

5.1.4.24 **MCloudPacket*** mcloudPacketInitFromImage (**MCloud** * *cloudP,* const char * *startTime,* const char * *stopTime,* const char * *fingerPrint,* int *width,* int *height,* const char * *format,* const char * *buffer,* int *bufferN*)

Initialize a new packet from image for sending.

Parameters

in	*cloudP*	reference to an MCloud object
in	*startTime*	human readable starting time stamp
in	*stopTime*	human readable ending time stamp
in	*fingerPrint*	language finger print of the image (slide text language)
in	*width*	image width
in	*height*	image height
in	*format*	image format, i.e. PNG
in	*buffer*	reference to a buffer that keeps the image
in	*bufferN*	length of buffer in bytes

Returns

reference to the created package or NULL if failed

5.1.4.25 SHAREDDLL MCloudPacket* mcloudPacketInitFromText (MCloud * *cloudP,* const char * *startTime,* const char * *stopTime,* unsigned int *startOffset,* unsigned int *stopOffset,* const char * *fingerPrint,* const char * *text*)

Initialize a new packet from text for sending.

startOffset and stopOffset do not necessarily correspond with startTime and stopTime. While startTime and stopTime define the absolute time of the packet, startOffset and stopOffset define that start and stop time stamp of e.g. the speech transcription within the packet (still relative to the beginning of the stream. For example: Packet from startTime=14:01:00.000 to stopTime=14:02:00.000 in a stream that started at startTime=14:00:00.000. However, speech within this packet starts 10 seconds ahead from the beginning of the packet because there is some silence to first 10 seconds and ends 1 second before the end of the packet. This means we need to define a startOffset=70000 (1 min since start of packet + 10 seconds of silence) and a stopOffset=119000.

Parameters

in	*cloudP*	reference to an MCloud object
in	*startTime*	human readable starting time stamp
in	*stopTime*	human readable ending time stamp
in	*startOffset*	start time offset in ms relative to the beginning of the stream
in	*stopOffset*	stop time offset in ms relative to the beginning of the stream
in	*fingerPrint*	finger print of the text
in	*text*	the text string

Returns

reference to the created package or NULL if failed

5.1.4.26 SHAREDDLL MCloudPacket* mcloudPacketInitFromWordTokenA (MCloud * *cloudP,* const char * *startTime,* const char * *stopTime,* unsigned int *startOffset,* unsigned int *stopOffset,* const char * *fingerPrint,* MCloudWordToken * *tokenA,* int *tokenN*)

Initialize a new packet from a MCloudWordToken array for sending.

Parameters

in	*cloudP*	reference to an MCloud object
in	*startTime*	human readable starting time stamp

in	*stopTime*	human readable ending time stamp
in	*startOffset*	start time offset in ms relative to the beginning of the stream
in	*stopOffset*	stop time offset in ms relative to the beginning of the stream
in	*fingerPrint*	finger print of the text
in	*tokenA*	reference to an MCloudWordToken array
in	*tokenN*	number of word tokens in array

Returns

reference to the created package or NULL if failed

5.1.4.27 SHAREDDLL S2S_Result mcloudPacketReplaceText (MCloud * *cloudP,* MCloudPacket * *p,* char * *text*)

Convenient function for replacing the string embedded in <text></text>.

Don't forget to free the memory allocated for text.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	reference to an MCloud package
out	*text*	returned text

Returns

S2S_Success if no error occurred

5.1.4.28 SHAREDDLL int mcloudPending (MCloud * *cloudP,* MCloudType *queueType*)

Return number of pending packages in queue.

Parameters

in	*cloudP*	reference to an MCloud object
in	*queueType*	type of queue MCloudSendingQueue, or MCloudProcessingQueue

Returns

number of pending packages

5.1.4.29 SHAREDDLL S2S_Result mcloudProcessDataAsync (MCloud * *cloudP,* MCloudPacket * *p,* void * *userData*)

Process received packages asynchronously.

This function can be used to process packets asynchronously. The packages will be placed into an internal queue and processed in the background by calling mcloudDataCallback. Callback functions are used to forward status messages such as errors. Use mcloudBreak to stop processing pending packages. Use mcloudWaitFinish to wait until the last package has been processed. Packages are freed automatically after they have been sent. While processing, the data callback function is called for the next pending package. As soon as no more packages are pending and mcloudWaitFinish has been called, the finalize callback function is called. The error callback function may be called in case of errors, and the break callback function if mcloudBreak has been called.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	an MCloud packet
in	*userData*	additional user data, that is passed to the callback functions

Returns

S2S_Success if no error occurs

5.1.4.30 SHAREDDLL S2S_Result mcloudRequestForDisplay (MCloud * *cloudP*)

Request the display of an output stream.

By calling this function, the client requests the display of the output stream on the display server. For cancelling the request for display, the client needs to disconnect.

Parameters

in	*cloudP*	reference to an MCloud object

Returns

S2S_Success if no error occurs

5.1.4.31 SHAREDDLL S2S_Result mcloudRequestInputStream (MCloud * *cloudP,* const char * *type,* const char * *fingerPrint,* const char * *streamID,* char * *info,* int *infoN*)

Request an input stream of a client from an MCloud object.

This function has to be called in order to request a specific input stream from the MCloud such as ASR or MT results. Otherwise, the client will not receive any data. This function has to be called after the client has been connected to the MCloud.

Parameters

in	*cloudP*	reference to an MCloud object
in	*type*	data type (audio, text, image)
in	*fingerPrint*	finger print of the data stream
in	*streamID*	stream identifier of the (output) stream which the requested data stream belongs to
in,out	*info*	additional information received from the MCloud
in	*infoN*	length of info field in bytes

Returns

S2S_Success if no error occurs

5.1.4.32 SHAREDDLL S2S_Result mcloudSendBinaryFile (MCloud * *cP,* FILE * *f,* int *chunkSize,* char * *filename,* char * *mimeType,* char * *fingerPrint*)

Convenience function for sending the content of a whole file.

Parameters

in	*cloudP*	reference to an MCloud object
in	*f*	the FILE stream to send
in	*chunkSize*	bytes read from the file to be inserted in 1 packet
in	*filename*	name of the file being sent (set to NULL if not relevant)
in	*mimetype*	MIME type of the file being sent (set to NULL if not relevant)
in	*fingerPrint*	finger print of the file

Returns

S2S_Success if no error occurs

5.1.4.33 SHAREDDLL S2S_Result mcloudSendBinaryFileAsync (MCloud * *cP,* FILE * *f,* int *chunkSize,* char * *filename,* char * *mimeType,* char * *fingerPrint,* void * *userData*)

Convenience function for sending the content of a whole file asyncronously.

Parameters

in	*cloudP*	reference to an MCloud object
in	*f*	the FILE stream to send
in	*chunkSize*	bytes read from the file to be inserted in 1 packet
in	*filename*	name of the file being sent (set to NULL if not relevant)
in	*mimetype*	MIME type of the file being sent (set to NULL if not relevant)
in	*fingerPrint*	finger print of the file
in	*userData*	additional user data, that is passed to the callback functions for each packet

Returns

S2S_Success if no error occurs

5.1.4.34 SHAREDDLL S2S_Result mcloudSendDone (MCloud * *cloudP*)

Inform a client or a worker that there is no more data to receive.

This function should be called to inform a worker that there is no more data to receive from a client, or to inform a client, that the worker finished processing of the data received. As soon as both, worker and client have sent a done, the session is terminated.

Parameters

in	*cloudP*	reference to an MCloud object

Returns

S2S_Success if no error occurs

5.1.4.35 SHAREDDLL S2S_Result mcloudSendError (MCloud * *cloudP,* const char * *description*)

Inform a client or a worker that an error occurred during processing.

This function can be called if an error occurs during processing either in the worker or client in order to inform all other involved components. Typically this results in a termination of the current session.

Parameters

in	*cloudP*	reference to an MCloud object
in	*description*	error description

Returns

S2S_Success if no error occurs

5.1.4.36 SHAREDDLL S2S_Result mcloudSendFlush (MCloud * *cloudP*)

Inform subsequent worker to flush their output buffers.

This function should be called to inform subsequent workers finalize processing data stored in the queue and to flush their buffers.

Parameters

in	*cloudP*	reference to an MCloud object

Returns

S2S_Success if no error occurs

5.1.4.37 SHAREDDLL S2S_Result mcloudSendPacket (MCloud * *cloudP*, MCloudPacket * *p*)

Send a packet.

This function has to be called to send a data packet to the MCloud. The data packet has to be created in advance by using the packet handling functions and has to be freed after usage.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	an MCloud packet

Returns

S2S_Success if no error occurs

5.1.4.38 SHAREDDLL S2S_Result mcloudSendPacketAsync (MCloud * *cloudP*, MCloudPacket * *p*, void * *userData*)

Send a packet asynchronously.

This function can be used for sending data packets asynchronously. The packages will be placed into an internal queue and sent in the background. Callback functions are used to forward status messages such as errors. Use mcloudBreak to stop sending pending packages. Use mcloudWaitFinish to wait until the last package has been sent. Packages are freed automatically after they have been sent. While sending, the error callback function may be called in case of errors and the break callback function if mcloudBreak has been called. As soon as no more packages are pending and mcloudWaitFinish has been called, the finalize callback function is called.

Parameters

in	*cloudP*	reference to an MCloud object
in	*p*	an MCloud packet
in	*userData*	additional user data, that is passed to the callback functions

Returns

S2S_Success if no error occurs

5.1.4.39 SHAREDDLL S2S_Result mcloudSetAttr (MCloud * *cloudP,* MCloudAttribute *attr,* const void * *value*)

Set the value of an attribute.

Deprecated Set the value of an attribute.

Parameters

in	*cloudP*	reference to an MCloud object
in	*attr*	an MCloud attribute
in	*value*	reference to the value to be set

Returns

S2S_Success if no error occurs

5.1.4.40 SHAREDDLL S2S_Result mcloudSetAudioEncoder (MCloud * *cp,* MCloudCodec *codec,* int *sampleRate,* int *bitRate,* int *channels*)

Set the audio codec.

Parameters

in	*cp*	reference to an MCloud object
in	*codec*	MCloud codec used to transmit/receive data to/from the Mediator
in	*sampleRate*	sample rate used to transmit/receive data to/from the Mediator
in	*bitRate*	bit rate used to transmit/receive data to/from the Mediator
in	*channels*	channels used to transmit/receive data to/from the Mediator

Returns

S2S_Success if no error occurs

5.1.4.41 SHAREDDLL S2S_Result mcloudSetAudioEncoder2 (MCloud * *cp,* char * *codec,* int *sampleRate,* int *bitRate,* int *channels*)

Set the audio codec.

Parameters

in	*cp*	reference to an MCloud object
in	*codec*	codec (string form) used to transmit/receive data to/from the Mediator

in	*sampleRate*	sample rate used to transmit/receive data to/from the Mediator
in	*bitRate*	bit rate used to transmit/receive data to/from the Mediator
in	*channels*	channels used to transmit/receive data to/from the Mediator

Returns

S2S_Success if no error occurs

5.1.4.42 SHAREDDLL void mcloudSetBreakCallback (MCloud * *cloudP,* MCloudType *queueType,* MCloudCallbackFct * *callback,* void * *userData*)

Set a break callback function.

This function is called when the worker should stop the processing as soon as possible. This callback is available for both the sending and the processing queue.

Parameters

in	*cloudP*	reference to an MCloud object
in	*queueType*	type of queue MCloudSendingQueue, or MCloudProcessingQueue
in	*callback*	reference to the function that has to be called
in	*userData*	reference to some data that is passed to the callback function

5.1.4.43 SHAREDDLL void mcloudSetCustomizationCallback (MCloud * *cP,* MCloudPacketCallbackFct * *callback,* void * *userData*)

Set a customization callback function.

Parameters

in	*cloudP*	reference to an MCloud object
in	*callback*	reference to the function that has to be called
in	*userData*	reference to some data that is passed to the callback function

5.1.4.44 SHAREDDLL void mcloudSetDataCallback (MCloud * *cloudP,* MCloudPacketCallbackFct * *callback*)

Set a data callback function.

This function is called for each incoming data package in a serial way, i.e. after a package has been processed it is called again if more packages are pending. Note that for this function no userData is given at the time of the set of the callback function. Instead, the userData is given per packet with mcloudProcessDataAsync or mcloudSendAsync. This callback is available for the processing queue only.

Parameters

in	*cloudP*	reference to an MCloud object
in	*callback*	reference to the function that has to be called

5.1.4.45 SHAREDDLL void mcloudSetErrorCallback (MCloud * *cloudP,* MCloudType *queueType,* MCloudCallbackFct * *callback,* void * *userData*)

Set an error callback function.

This function is called as soon as an error occurs in the asynchronous processing. This callback is available for both the sending and the processing queue.

Parameters

in	*cloudP*	reference to an MCloud object
in	*queueType*	type of queue MCloudSendingQueue, or MCloudProcessingQueue
in	*callback*	reference to the function that has to be called
in	*userData*	reference to some data that is passed to the callback function

5.1.4.46 SHAREDDLL void mcloudSetFinalizeCallback (MCloud * *cloudP,* MCloudCallbackFct * *callback,* void * *userData*)

Set a finalize callback function.

This function is called as soon as the processing of packets should be finalized, i.e. no more packets will follow and the worker should output the final results after all pending packets have been processed. This callback is only available for the processing queue.

Parameters

in	*cloudP*	reference to an MCloud object
in	*callback*	reference to the function that has to be called
in	*userData*	reference to some data that is passed to the callback function

5.1.4.47 SHAREDDLL void mcloudSetInitCallback (MCloud * *cloudP,* MCloudPacketCallbackFct * *callback,* void * *userData*)

Set an init callback function.

This function is called as soon as an incoming service request has been accepted by the worker, i.e. in mcloudWaitForClient. The packet containing the service description is passed to the init callback function as argument. This callback is only available for the processing queue.

Parameters

in	*cloudP*	reference to an MCloud object
in	*callback*	reference to the function that has to be called
in	*userData*	reference to some data that is passed to the callback function

5.1.4.48 SHAREDDLL S2S_Result mcloudWaitFinish (MCloud * *cloudP,* MCloudType *queueType,* int *done*)

Wait until all pending packages have been processed/ sent.

This function can be used to wait until all pending packages have been processed or sent in the queue specified.

Parameters

in	*cloudP*	reference to an MCloud object
in	*queueType*	type of queue MCloudSendingQueue, or MCloudProcessingQueue
in	*done*	if set to 1, indicates that processing of the request has been completed

Returns

S2S_Success if no error occurs

5.1.4.49 SHAREDDLL S2S_Result mcloudWaitForClient (MCloud * *cloudP,* char ** *streamID*)

Wait for a service request to process.

This function has to be called after the worker has been successfully connected to the MCloud in order to wait for an incoming service request to process.

Parameters

in	*cloudP*	reference to an MCloud object
out	*streamID*	ID of input stream

Returns

S2S_Success if no error occurs

5.1.4.50 SHAREDDLL MCloudWordToken* mcloudWordTokenArrayCreate (int *n*)

Creates an array of MCloudWordTokens.

Parameters

in	*n*	number of elements

Returns

reference to the created token array or NULL if failed

5.1.4.51 SHAREDDLL void mcloudWordTokenArrayFree (MCloudWordToken * *tokenA,* int *n*)

Free an array of MCloudWordTokens.

Parameters

in	*tokenA*	reference to an MCloudWorkToken array
in	*n*	number of elements

5.1.4.52 SHAREDDLL char* url_decode (const char * *str*)

Convenient function that returns a url-decoded version of a string.

Don't forget to free() the returned string after use

Parameters

in	*str*	string to be decoded

Returns

url-decoded version of str

5.1.4.53 SHAREDDLL char* url_encode (const char * *str*)

Convenient function that returns a url-encoded version of a string.

Don't forget to free() the returned string after use

Parameters

in	*str*	string to be encoded

Returns

url-encoded version of str

Index