mas_asm_connect_ports

Name
`mas_asm_connect_ports` - Connect a source port to a sink without a DC.

Synopsis
```
#include “mas/mas.h”

int32 mas_asm_connect_ports( mas_port_t source,
mas_port_t sink );
```

Description
Connects *`source`* and *`sink`*, using the configured data characteristic (DC) from either port. This function can be used to connect ports that are each in different servers.

Return value
Returns

- 0 on success
- MERR_NOTDEF if either port wasn’t defined
- MERR_INVALID if a port was already connected or if neither port has a DC
- MERR_INVALID if the characteristic matrices were incompatible

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_asm_connect_source_sink

Name

`mas_asm_connect_source_sink` - Connect a source port to a sink port.

Synopsis

```
#include “mas/mas.h”

int32 mas_asm_connect_source_sink( mas_port_t source,
mas_port_t sink, struct  mas_data_characteristic* dc );
```

Description

Connects *`source`* and *`sink`* if their characteristic matrices are compatible. This transmits the `mas_asm_connect_source_sink` event and blocks on a response from the server.

Return value

Returns

- •0 on success
- •must look into

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_asm_get_dc

Name

`mas_asm_get_dc` - Retrieve the data characteristic of a port

Synopsis

```
#include “mas/mas.h”

int32 mas_asm_get_dc( mas_port_t port,
struct mas_data_characteristic** retval_dc );
```

Description

Retrieves the configured data characteristic for *`port`*, placing the result in *`*retval_dc`*. Memory will be allocated to hold *`*retval_dc`*.

Return value

Returns

- 0 on success
- MERR_COMM on communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_asm_get_device_by_name

Name

`mas_asm_get_device_by_name` - Given a device name, return a handle for it.

Synopsis

```
#include “mas/mas.h”

int32 mas_asm_get_device_by_name( char* name,  mas_device_t*
device );
```

Description

Locates an instantiated device by name on the display-side server. On return, *`device`* is a handle to the instantiated device.

Return value

Returns

- •0 on success
- •MERR_COMM on communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_asm_get_device_by_name_on_channel

Name

`mas_asm_get_device_by_name_on_channel` - Given a device name and a channel, return a handle for it.

Synopsis

```
#include “mas/mas.h”

int32 mas_asm_get_device_by_name_on_channel( char* name,
mas_device_t* device, mas_channel_t channel );
```

Description

Locates an instantiated device by name on the server channel specified in *`channel`*. On return, *`device`* is a handle to the instantiated device.

Return value

Returns

- 0 on success
- MERR_COMM on communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_asm_get_port_by_name

Name

`mas_asm_get_port_by_name` - Get a handle to a port given its name.

Synopsis

```
#include “mas/mas.h”

int32 mas_asm_get_port_by_name( mas_device_t device, char*
name, mas_port_t* port );
```

Description

Returns a handle *`port`* to the named port on device *`device`*. Use this function to get port handles prior to connecting them. Port names are unique for each device. Certain port names may be unique to a server. To return a handle to the first match of a port name on all devices in the display-side server, use 0 (zero) for the device handle.

Return value

Returns

- 0 on success
- MERR_COMM if communications error
- MERR_MEMORY if there isn’t enough memory

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_asm_instantiate_device

Name

`mas_asm_instantiate_device` - Instantiate a device in the display-side server.

Synopsis

```
#include “mas/mas.h”

int32 mas_asm_instantiate_device( char* name,
void* prediate, int32 predicate_len, mas_device_t* device );
```

Description

Instantiates a device on the display-side server, given its name. *`predicate`* and *`predicate_len`* are optional, but will form the predicate to the standard mas_dev_init_instance action. On return, *`device`* will be a handle to the instantiated device.

Return value

Returns

- 0 on success
- MERR_COMM on communcations error
-

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_asm_instantiate_device_on_channel

Name

`mas_asm_instantiate_device_on_channel` - Instantiate a device on the server specified in *`channel`*.

Synopsis

```
#include “mas/mas.h”

int32 mas_asm_instantiate_device_on_channel( char* name,
void* prediate, int32 predicate_len, mas_device_t* device,
mas_channel_t channel );
```

Description

Instantiates a device on the server specified in *`channel`*, given its name. *`predicate`* and *`predicate_len`* are optional, but will form the predicate to the standard mas_dev_init_instance action. On return, *`device`* will be a handle to the instantiated device in the specified server.

Return value

Returns

- 0 on success
- MERR_COMM on communcations error
-

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_dev_show_state

Name

`mas_dev_show_state` - (debugging) log the state of a device to the server log.

Synopsis

```
#include "mas/mas.h"

int32 mas_dev_show_state( mas_device_t device );
```

Description

`This is a standard device action.  If implemented by` *`device`*`, the device will log its state to the server log.`

Return value

Returns

- 0 none

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_get

Name

`mas_get` - Standard interface for retrieving information from the server

Synopsis

```
#include “mas/mas.h”

int32 mas_get( mas_device_t device, char* key,
struct mas_package* arg, struct mas_package** r_package );
```

Description

`mas_get` is a MAS standard interface for retrieving information from a MAS server. Its functionality is supported by the core set of MAS devices, the assembler, and the scheduler. It is complementary to the `mas_set` function that allows clients to configure information in the MAS server.

`mas_get` triggers a `mas_get` action in the device identified by the *`device`* handle. The action is parameterized by a string *`key`* and a package *`arg`*. On return, *`*r_package`* contains the results of the query. The assembler and scheduler are addressed using their device handles; these are retrieved using the `mas_get_asm_device` and `mas_get_sch_device` functions, respectively.

The contents of the string key are used to select one of a number of primary queries that the target device supports. If *`key`* is “list”, with a null *`arg`*, the target device will report its supported primary queries.

`arg`’s interpretation varies depending on the value of *`key`* and the *`device`* queried. Often, no *`arg`* is required, and it is set to zero (0) by the caller. When *`arg`* is required, it is the caller’s responsibility to correctly construct the *`arg`* package.

Memory will be allocated by mas_get for *`*r_package`*.

Return value

Returns

- •0 on success

Examples

Retrieve the primary queries for the assembler:

```
    mas_get_asm_device( &asmb );
    mas_get( asmb, "list", 0, &r_package );
```

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

```
  Contents of r_package:
         0: "" (string) "list"
         1: "" (string) "ports"
         2: "" (string) "devices"
         3: "" (string) "actions"
         4: "" (string) "action_wcstat"
```

Retrieve the timing statistics for action 6 of device number 16:

```
    masc_make_package( &arg, 0 );
    masc_pushk_int32( arg, “device_instance”, 16 );
    masc_pushk_uint8( arg, “action”, 6 );
    masc_finalize_package( arg );
    mas_get( asmb, "action_wcstat", arg, &r_package );

  Contents of r_package:
         0: "count" (uint32) 238914
         1: "mean" (double) 0.000000
         2: "min" (double) 9.000000
         3: "max" (double) 126.000000
```

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_get_asm_device

Name

`mas_get_asm_device` - Retrieve a handle for the assembler device on the display-side server.

Synopsis

```
#include “mas/mas.h”

int32 mas_get_asm_device( mas_device_t* device );
```

Description

Retrieve a device handle to the assembler on the display-side server. On return, ***device*** will have the handle.

Return value

Returns

- 0 on success
- MERR_MEMORY if there isn’t enough memory

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_get_asm_device_on_channel

Name

`mas_get_asm_device_on_channel` - Retrieve a handle for the assembler device on the specified control channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_get_asm_device_on_channel( mas_device_t* device,
mas_channel_t channel );
```

Description

Retrieve a device handle to the assembler on the control channel *`channel`*. On return, *`*device`* will have the handle.

Return value

Returns

- •0 on success
- •MERR_MEMORY if there isn’t enough memory

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_get_sch_device

Name

`mas_get_sch_device` - Retrieve a handle for the scheduler device on the display-side server.

Synopsis

```
#include “mas/mas.h”

int32 mas_get_sch_device( mas_device_t* device );
```

Description

Retrieve a device handle to the scheduler on the display-side server. On return, ***device*** will have the handle.

Return value

Returns

- 0 on success
- MERR_MEMORY if there isn’t enough memory

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_get_sch_device_on_channel

Name

`mas_get_sch_device_on_channel` - Retrieve a handle for the scheduler device on the specified control channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_get_sch_device_on_channel( mas_device_t* device,
mas_channel_t channel );
```

Description

Retrieve a device handle to the scheduler on the control channel *`channel`*. On return, *`*device`* will have the handle.

Return value

Returns

- •0 on success
- •MERR_MEMORY if there isn’t enough memory

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_get_display_control_channel

Name

`mas_get_display_control_channel` - Retrieve a handle for the control channel to the display-side server.

Synopsis

```
#include “mas/mas.h”

int32 mas_get_display_control_channel( mas_channel_t*
channel );
```

Description

Retrieve a channel handle to the display-side server. On return, *`channel`* will have the handle.

Return value

Returns

- 0 on success
- MERR_MEMORY if there isn’t enough memory

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_get_local_control_channel

Name

`mas_get_local_control_channel` - Retrieve a handle for the control channel to the local MAS.

Synopsis

```
#include “mas/mas.h”

int32 mas_get_local_control_channel( mas_channel_t*
channel );
```

Description

Retrieve a channel handle *`channel`* to the local server.

NOTE *Use this with caution! Specifically addressing the local server breaks network transparency!*

Return value

Returns

- 0 on success
- MERR_MEMORY if there isn’t enough memory

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_init

Name

`mas_init` - Initialize MAS for this client.

Synopsis

```
#include “mas/mas.h”

int32 mas_init( void );
```

Description

Creates connections between the client and the local Media Application Server and, if required, connections between the local server and a remote display-side server. This function must be called prior to any other interaction with MAS.

Return value

Returns

- 0 on success
- MERR_MEMORY if there isn’t enough memory
- MERR_COMM if there was a communications problem.

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_make_data_channel

Name

`mas_make_data_channel` - Create a bi-directional data connection to the display-side server.

Synopsis

```
#include “mas/mas.h”

int32 mas_make_data_channel( char* name, mas_channel_t*
data_channel, mas_port_t* remote_source, mas_port_t*
remote_sink );
```

Description

Opens a bi-directional communication pathway between the client and the display-side MAS. The channel will be labelled with the text string in *`name`*. On return, *`data_channel`* will be a handle to the new channel, *`remote_source`* will be the source port for the channel in the display-side server, and *`remote_sink`* will be the sink port for the channel in the display-side server. This communication pathway is used to send non-control information from the client to devices in the server, or to receive non-control information from devices in the server.

Return value

Returns

- •0 on success
- •MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_recv_package

Name

`mas_recv_package` - Receive a MAS package from the specified channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_recv_package( mas_channel_t channel, struct
mas_package* package );
```

Description

Receive a package from the channel *`channel`*. On return, *`package`* contains the package. Use this function on data or control channels. This function may block.

Return value

Returns

- •0 on success
- •MERR_INVALID if the channel is invalid
- •MERR_MEMORY if there isn’t enough memory
- •MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_recv_package_from_display

Name

`mas_recv_package_from_display` - Receive a MAS package from the display-side server control channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_recv_package_from_display( struct mas_package*
package );
```

Description

Receive a package from the display-side control channel. On return, *`package`* contains the package. This function may block.

Return value

Returns

- 0 on success
- MERR_MEMORY if there isn’t enough memory
- MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_recv_package_from_local

Name

`mas_recv_package_from_local` - Receive a MAS package from the local server control channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_recv_package_from_local( struct mas_package* pack-
age );
```

Description

Receive a package from the local MAS server control channel. On return, *`package`* contains the package. This function may block.

NOTE Use this with caution! Specifically addressing the local server breaks network transparency!

Return value

Returns

- 0 on success
- MERR_INVALID if the channel is invalid
- MERR_MEMORY if there isn’t enough memory
- MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_send

Name

`mas_send` - Send data to the specified channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_send( mas_channel_t channel, struct mas_data*
data);
```

Description

Transforms *`data`* to an RTP packet and sends it over the specified channel.

Return value

Returns

- 0 on success
- MERR_INVALID if the specified channel was invalid
- MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_send_package

Name

`mas_send_package` - Send a package to the specified channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_send_package( mas_channel_t channel, struct
mas_package* package );
```

Description

Sends a package to the channel *`channel`*. Use this function on data or control channels. This function may block.

Return value

Returns

- •0 on success
- •MERR_INVALID if the specified channel was invalid
- •MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_send_package_to_display

Name

`mas_send`_package_to_display - Send a package to the display-side control channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_send_package_to_display( struct mas_package*
package );
```

Description

Sends a package to the display-side server control channel. This function may block.

Return value

Returns

- •0 on success
- •MERR_INVALID if the specified channel was invalid
- •MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_send_package_to_local

Name

`mas_send_package_to_local` - Send a package to the specified channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_send_package_to_local( struct mas_package*
package);
```

Description

Sends a package to the local MAS server control channel. This function may block.

NOTE Use this with caution! Specifically addressing the local server breaks network transparency!

Return value

Returns

- 0 on success
- MERR_INVALID if the specified channel was invalid
- MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_send_to_display

Name

`mas_send_to_display` - Send data to the display-side server control channel.

Synopsis

```
#include "mas/mas.h"

int32 mas_send_to_display( struct mas_data* data);
```

Description

Transforms *`data`* to an RTP packet and sends it over the display-side server control channel.

Return value

Returns

- 0 on success
- MERR_INVALID if the specified channel was invalid
- MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_send_to_local

Name

`mas_send_to_local` - Send data to the local server control channel.

Synopsis

```
#include “mas/mas.h”

int32 mas_send_to_local( struct mas_data* data);
```

Description

Transforms *`data`* to an RTP packet and sends it over the local server control channel.

Return value

Returns

- 0 on success
- MERR_INVALID if the specified channel was invalid
- MERR_COMM if there was a communication error

Examples

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.

mas_set

Name

`mas_set` - Standard interface for configuring information in the server

Synopsis

```
#include “mas/mas.h”

int32 mas_set( mas_device_t device, char* key,
struct mas_package* arg );
```

Description

`mas_set` is a MAS standard interface for configuring or adjusting information information in a MAS server. Its functionality is supported by the core set of MAS devices, the assembler, and the scheduler. It is complementary to the `mas_get` function that allows clients to retrieve information from the MAS server.

`mas_set` triggers a `mas_set` action in the device identified by the *`device`* handle. The action is parameterized by a string *`key`* and a package *`arg`*. The assembler and scheduler are addressed using their device handles; these are retrieved using the `mas_get_asm_device` and `mas_get_sch_device` functions, respectively.

The contents of the string key are used to select one of a number of parameters that the target device supports. Typically, although not always, the parameters are a subset of the queries supported by mas_get on the same device. If they are, a call to `mas_get` with the *`key`* “list” will report its supported primary queries.

`arg`’s interpretation varies depending on the value of *`key`* and the target *`device`*. The parameter to set may be simple, requiring only one member in the *`arg`* package, or it may be a complex, or compound parameter, requiring many members of the *`arg`* package. It is the caller’s responsibility to correctly construct the *`arg`* package.

Return value

Returns 0 on success

Examples

Set the gain of the main output on the anx device:

```
    mas_asm_get_device_by_name( "anx", &anx );
    masc_make_package( &arg, 0 );
    masc_push_uint8( arg, “channel”, 0 );
    masc_push_int16( arg, “left”, -60 );
    masc_push_int16( arg, “right”, -60 );
    err = mas_set( anx, "gain_db", arg );
```

See Also

Copyright (c) 2003 Shiman Associates Inc. All Rights Reserved.