NP Matching 2 Library Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Document serial number: 000004892117

Document serial number: 000004892117

Table of Contents

Initialization and Termination Functions

sceNpMatching2Init

Initialize the library

Definition

```
#include <np.h>

int
sceNpMatching2Init(
        const SceSize poolSize,
        const SceInt32 threadPriority,
        const SceInt32 cpuAffinityMask,
        const SceSize threadStackSize
        );
```

Calling Conditions

Not multithread safe.

Arguments

poolSize	Memory pool size for the library in bytes
threadPriority	Priority of internal thread
cpuAffinityMask	CPU affinity of internal thread
threadStackSize	Stack size of internal thread in bytes

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_ALREADY_INITIALIZED	0x80550c02	Already initialized sceNpMatching2Init() may have already been called. Check the calling order.

For a list of the NP error codes, refer to each reference document.

Description

This function initializes the NP Matching 2 library. An internal thread of the NP Matching 2 library is created upon initialization. Specify the stack size and priority of this thread to *threadStackSize* and *threadPriority*, respectively. Specify the memory pool size to be used for matching for *poolsize*. Specify the CPU affinity of the thread to *cpuAffinityMask*.

When 0 is specified for *threadPriority* and *threadStackSize*, the default values will be internally used. The default values will be the following macros, respectively.

- SCE_NP_MATCHING2_THREAD_PRIORITY_DEFAULT
- SCE_NP_MATCHING2_THREAD_STACK_SIZE_DEFAULT

See Also

sceNpMatching2Term()

sceNpMatching2Term

Terminate the library

Definition

```
#include <np.h>

int
sceNpMatching2Term(
        void
        );
```

Calling Conditions

Not multithread safe.

Arguments

None

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized `sceNpMatching2Init()` may not have been called yet. Check the calling order.

For a list of the NP error codes, refer to each reference document.

Description

This function terminates the NP Matching 2 library.

See Also

`sceNpMatching2Init()`

Context Operation Functions

sceNpMatching2CreateContext

Create a context

Definition

```
#include <np.h>

int
sceNpMatching2CreateContext(
        const SceNpId *npId,
        const SceNpCommunicationId *commId,
        const SceNpCommunicationPassphrase *passPhrase,
        SceNpMatching2ContextId *ctxId,
        );
```

Calling Conditions

Multithread safe.

Arguments

npId	NP ID of the user creating the context
commId	NP Communication ID allocated to the application
passPhrase	NP communication passphrase allocated to the application
ctxId	Pointer to buffer to store the context ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_MAX	0x80550c04	No more contexts can be created Re-execute this function after deleting an existing context.
SCE_NP_MATCHING2_ERROR_CONTEXT_ALREADY_EXISTS	0x80550c05	Context already exists A context created by the same NP ID and NP Communication ID already exists.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.

For a list of the NP error codes, refer to each reference document.

Description

This function creates an NP Matching 2 library context. In order to use the NP Matching 2 library, it is necessary to create a context after initializing the library. To *commId* and *passPhrase*, specify the NP Communication ID and NP communication passphrase obtained upon user registration to NP. At this time, if NULL is specified to *commId* and *passPhrase*, values of NP Communication ID and NP communication passphrase specified in `sceNpInit()` will be used.

When a context is successfully created, the context ID will be stored to the buffer indicated by *ctxId*. From this point onwards, this context ID will be used when calling the NP Matching 2 library function.

Notes

Behavior cannot be guaranteed when this function is executed from multiple threads at the same time using the same combination of NP ID and NP Communication ID. This function is multithread safe for executing on different combinations of the NP ID and the NP Communication ID.

Examples

```
int ret;

//E Parameters required for creating a context
//  Assuming that appropriate values are stored
SceNpId npId;
SceNpCommunicationId commId;
SceNpCommunicationPassphrase passPhrase;

//E Variable to store the ID of the new context
SceNpMatching2ContextId ctxId;

//E Create a context
ret = sceNpMatching2CreateContext(
    &npId, &commId, &passPhrase, &ctxId);
if (ret < 0) {
    //E Error handling
}
```

See Also

`sceNpMatching2DestroyContext(),sceNpInit()`

sceNpMatching2DestroyContext

Destroy a context

Definition

```
#include <np.h>

int
sceNpMatching2DestroyContext(
        const SceNpMatching2ContextId ctxId
        );
```

Calling Conditions

Multithread safe. (Refer to the Notes.)

Arguments

ctxId Context ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.

For a list of the NP error codes, refer to each reference document.

Description

This function destroys an NP Matching 2 library context.

When this function is executed, the joined-in session and any request being executed will be deleted. Note that there will be no events occurring to notify these deletions.

Notes

Behavior cannot be guaranteed when this function is executed from multiple threads at the same time using the same context ID. This function is multithread safe for executing on different context IDs. Be sure not to call this function from a callback to be registered in the library.

Examples

```
int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;

//E Delete the context
ret = sceNpMatching2DestroyContext(ctxId);
if (ret < 0) {
    //E Error handling
}
```

See Also

sceNpMatching2CreateContext()

sceNpMatching2ContextStart

Start a context

Definition

```
#include <np.h>

int
sceNpMatching2ContextStart(
        const SceNpMatching2ContextId ctxId,
        const SceUInt64 timeout
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
timeout	Time-out time

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_CONTEXT_ALREADY_STARTED	0x80550c07	Start process already executed on context This function may have already been executed on the context specified in *ctxId*. Check the calling order.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_TIMEDOUT	0x80550c3c	Timed out Timed out because processing did not complete after a certain period of time (approximately 30 seconds).
SCE_NP_COMMUNITY_ERROR_ABORTED	0x80550707	Processing to start context aborted This error returns when a network disconnection occurs, for example, during the processing to start a context and the processing is aborted.

For a list of the NP error codes, refer to each reference document.

Description

This function starts an NP Matching 2 library context. Always carry out start operation after creating a context using this function.

Because this is a non-blocking function, the completion of, or error in, this processing by this function is notified to the context callback function. Make sure the context callback function to be registered before executing this function. The notified event is `SCE_NP_MATCHING2_CONTEXT_EVENT_STARTED`, and the event cause is `SCE_NP_MATCHING2_EVENT_CAUSE_CONTEXT_ACTION`. The successful completion of the processing is reported with a 0 in the *`errorCode`* argument of the context callback function. This argument will be a negative value for an error.

Specify the value of 10*1000*1000 (10 seconds) or greater for *`timeout`*. If 0 is specified the default time-out time (20 seconds) will be set.

To abort this function's processing, execute `sceNpMatching2AbortContextStart()`.

Notes

Behavior cannot be guaranteed when this function is executed from multiple threads at the same time using the same context ID. This function is multithread safe for executing on different context IDs.

Examples

```
int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;

//E Start the context
ret = sceNpMatching2ContextStart(ctxId);
if (ret < 0) {
    //E Error handling
}
```

See Also

`sceNpMatching2AbortContextStart()`, `sceNpMatching2ContextStop()`, `SceNpMatching2ContextCallback`

sceNpMatching2AbortContextStart

Abort processing to start a context

Definition

```
#include <np.h>

int
sceNpMatching2AbortContextStart(
        const SceNpMatching2ContextId ctxId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId Context ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_COMMUNITY_ERROR_INVALID_ID	0x8055070e	Abort-target processing to start a context does not exist Either the processing to start a context has not been executed or the processing has already completed, and there is no processing to abort.

For a list of the NP error codes, refer to each reference document.

Description

This function aborts the processing to start a context of the NP Matching 2 library.

Examples

```
int ret;

//E Assuming that an appropriate value is stored
SceNpMatching2ContextId ctxId;

//E Abort processing to start a context
ret = sceNpMatching2AbortContextStart(ctxId);
if (ret < 0) {
    //E Error handling
}
```

See Also

sceNpMatching2ContextStart()

sceNpMatching2ContextStop

Stop a context

Definition

```
#include <np.h>

int
sceNpMatching2ContextStop(
        const SceNpMatching2ContextId ctxId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId Context ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_STARTED	0x80550c08	Start process not yet executed on context The context specified in *ctxId* may not have been started yet. Check the calling order.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function stops an NP Matching 2 library context.

The completion of, or an error in, the processing to stop a context will be notified to the context callback function. The notified event is SCE_NP_MATCHING2_CONTEXT_EVENT_STOPPED, and the event cause is SCE_NP_MATCHING2_EVENT_CAUSE_CONTEXT_ACTION. The successful completion of the processing is reported with a 0 in the *errorCode* argument of the context callback function. This argument will be a negative value for an error.

When this function is called while joined-into a session, the session will be deleted and the session deletion event (SCE_NP_MATCHING2_ROOM_EVENT_ROOM_DESTROYED/SCE_NP_MATCHING2_LOBBY_EVENT_LOBBY_DESTROYED) will be notified to the room event callback and the lobby event callback. In this case,

the cause of the event to be included in the event data corresponding to the session deletion event will be SCE_NP_MATCHING2_EVENT_CAUSE_CONTEXT_ERROR.

Moreover, when this function is called while the execution of a request is being processed, the executed request processing will be aborted, and the request event corresponding to the request will be notified together with the SCE_NP_MATCHING2_ERROR_CONTEXT_STOPPED error code.

Notes

Behavior cannot be guaranteed when this function is executed from multiple threads at the same time using the same context ID. This function is multithread safe for executing on different context IDs.

Examples

```
int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;

//E Stop the context
ret = sceNpMatching2ContextStop(ctxId);
if (ret < 0) {
    //E Error handling
}
```

See Also

sceNpMatching2ContextStart()

Library Configuration Functions

sceNpMatching2SetDefaultRequestOptParam

Set the default request option parameters

Definition

```
#include <np.h>

int
sceNpMatching2SetDefaultRequestOptParam(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2RequestOptParam *optParam
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
optParam	Request option parameters

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not be specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function sets the default request option parameters. It registers the default request callback function and timeout settings, and uses the *cbFunc*, *cbFuncArg*, *timeout* members of the SceNpMatching2RequestOptParam structure.

The default request option parameters set with this function will be applied to all request functions issued from the same context. However, the request option parameters specified upon calling each request function, if any, will be given precedence and applied instead of the default setting.

Examples

```
int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RequestOptParam optParam;

ret = sceNpMatching2SetDefaultRequestOptParam(
        ctxId, &optParam);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2RequestOptParam

sceNpMatching2RegisterRoomEventCallback

Register a room event callback

Definition

```
#include <np.h>

int
sceNpMatching2RegisterRoomEventCallback(
        const SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomEventCallback cbFunc,
        void *cbFuncArg
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
cbFunc	Room event callback function
cbFuncArg	Pointer to data to be passed to the argument *arg* of the room event callback function

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function registers a room event callback function. When this function is executed, all room events associated with a context will be notified to the room event callback function specified in *cbFunc*.

Examples

```
//E Room event callback function
static void roomEventCb(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2Event event,
        const void *data,
        void *arg)
{
    //E Callback handling
}

int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;
void *arg;

ret = sceNpMatching2RegisterRoomEventCallback(
        ctxId, roomEventCb, arg);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2RoomEventCallback

sceNpMatching2RegisterRoomMessageCallback

Register a room message callback

Definition

```
#include <np.h>

int
sceNpMatching2RegisterRoomMessageCallback(
        const SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomMessageCallback cbFunc,
        void *cbFuncArg
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
cbFunc	Room message callback function
cbFuncArg	Pointer to data to be passed to the argument *arg* of the room message callback function

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function registers a room message callback function. When this function is executed, all room message events associated with a context will be notified to the room message callback function specified in *cbFunc*.

Examples

```
//E Room message callback function
static void roomMessageCb(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2RoomMemberId srcMemberId,
        SceNpMatching2Event event,
        const void *data,
        void *arg)
{
    //E Callback handling
}

int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;
void *arg;

ret = sceNpMatching2RegisterRoomMessageCallback(
        ctxId, roomMessageCb, arg);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2RoomMessageCallback

sceNpMatching2RegisterLobbyEventCallback

Register lobby event callback

Definition

```
#include <np.h>

int
sceNpMatching2RegisterLobbyEventCallback(
        const SceNpMatching2ContextId ctxId,
        SceNpMatching2LobbyEventCallback cbFunc,
        void *cbFuncArg
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
cbFunc	Lobby event callback function
cbFuncArg	Pointer to arbitrary data that is passed in *arg* argument of lobby event callback function

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function registers a lobby event callback function. When this function is executed, all lobby events associated with the same context are reported to the lobby event callback function specified by *cbFunc*.

Examples

```
//E Lobby event callback function
static void lobbyEventCb(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2LobbyId lobbyId,
        SceNpMatching2Event event,
        const void *data,
        void *arg)
{
    //E Callback function processing
}

int ret;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
void *arg;

ret = sceNpMatching2RegisterLobbyEventCallback(
        ctxId, lobbyEventCb, arg);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2LobbyEventCallback

sceNpMatching2RegisterLobbyMessageCallback

Register lobby message callback

Definition

```
#include <np.h>

int
sceNpMatching2RegisterLobbyMessageCallback(
        const SceNpMatching2ContextId ctxId,
        SceNpMatching2LobbyMessageCallback cbFunc,
        void *cbFuncArg
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
cbFunc	Lobby message callback function
cbFuncArg	Pointer to arbitrary data that is passed in *arg* argument of lobby message callback function

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function registers a lobby message callback function. When this function is executed, all lobby message events associated with the same context are reported to the lobby message callback function specified by *cbFunc*.

Examples

```
//E Lobby message callback function
static void lobbyMessageCb(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2LobbyId lobbyId,
        SceNpMatching2LobbyMemberId srcMemberId,
        SceNpMatching2Event event,
        const void *data,
        void *arg)
{
    //E Callback function processing
}

int ret;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
void *arg;

ret = sceNpMatching2RegisterLobbyMessageCallback(
        ctxId, lobbyMessageCb, arg);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2LobbyMessageCallback

sceNpMatching2RegisterSignalingCallback

Register signaling callback

Definition

```
#include <np.h>

int
sceNpMatching2RegisterSignalingCallback(
        const SceNpMatching2ContextId ctxId,
        SceNpMatching2SignalingCallback cbFunc,
        void *cbFuncArg
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
cbFunc	Signaling callback function
cbFuncArg	Pointer to arbitrary data that is passed in *arg* argument of signaling callback function

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function registers a signaling callback function. When this function is executed, all signaling events associated with the same context are reported to the signaling callback function specified by *cbFunc*.

Examples

```
//E Signaling callback function
static void signalingCb(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2RoomMemberId peerMemberId,
        SceNpMatching2Event event,
        int errorCode,
        void *arg)
{
    //E Callback function processing
}

int ret;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
void *arg;

ret = sceNpMatching2RegisterSignalingCallback(
        ctxId, signalingCb, arg);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SignalingCallback

sceNpMatching2RegisterContextCallback

Register context callback

Definition

```
#include <np.h>

int
sceNpMatching2RegisterContextCallback(
        SceNpMatching2ContextCallback cbFunc,
        void *cbFuncArg
        );
```

Calling Conditions

Multithread safe.

Arguments

cbFunc	Context callback function
cbFuncArg	Pointer to arbitrary data that is passed in *arg* argument of context callback function

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.

For a list of the NP error codes, refer to each reference document.

Description

This function registers a context callback function. When this function is executed, all context events are reported to the context callback function specified by *cbFunc*.

Examples

```
//E Context callback function
static void contextCb(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2Event event,
        SceNpMatching2EventCause eventCause,
        int errorCode,
        void *arg)
{
    //E Callback function processing
}

int ret;

//E Assume that an appropriate value is stored
void *arg;

ret = sceNpMatching2RegisterContextCallback(
        contextCb, arg);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2ContextCallback

Library Functions

sceNpMatching2GetMemoryInfo

Get memory information (for development)

Definition

```
#include <np.h>

int
sceNpMatching2GetMemoryInfo(
        SceNpMatching2MemoryInfo *memInfo
        );
```

Calling Conditions

Multithread safe.

Arguments

memInfo Pointer to area storing the memory information

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.

For a list of the NP error codes, refer to each reference document.

Description

This function obtains memory information regarding the heap area to be used by the NP Matching 2 library.

When this function is executed, the size of the heap area, the current memory usage volume, and the maximum memory usage in the past, can be obtained. Use this function in application development and check the memory size required by your application.

Examples

```
int ret;
SceNpMatching2MemoryInfo memInfo;

Memset(&memInfo, 0, sizeof(memInfo));
ret = sceNpMatching2GetMemoryInfo(&memInfo);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2MemoryInfo

Callback Functions

SceNpMatching2RequestCallback

Request callback function

Definition

```
#include <np.h>

typedef void (*SceNpMatching2RequestCallback)(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RequestId reqId,
        SceNpMatching2Event event,
        int errorCode,
        const void *data,
        void *arg
        );
```

Arguments

ctxId	Context ID
reqId	Request ID allocated when the function was executed
event	Request event
errorCode	Error code A negative value indicates that an error occurred while processing the request function. 0 indicates a success.
data	Event data NULL indicates that there is no event data.
arg	Pointer to application-specified data

Description

This callback function is used for notification of request events corresponding to request function.

Either register a default request callback function with `sceNpMatching2SetDefaultRequestOptParam()`, or specify a request callback function per request when executing request function.

See Also

`sceNpMatching2SetDefaultRequestOptParam()`,`SCE_NP_MATCHING2_REQUEST_EVENT_*`

SceNpMatching2RoomEventCallback

Room event callback function

Definition

```
#include <np.h>

typedef void (*SceNpMatching2RoomEventCallback)(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2Event event,
        const void *data,
        void *arg
        );
```

Arguments

ctxId	Context ID
roomId	ID of room where event occurred
event	Room event
data	Event data NULL indicates that there is no event data.
arg	Pointer to application-specified data

Description

This callback function is used for notification of room events that occur while in a room.

Before joining a room, register a room event callback function with `sceNpMatching2RegisterRoomEventCallback()`.

See Also

`sceNpMatching2RegisterRoomEventCallback()`,`SCE_NP_MATCHING2_ROOM_EVENT_*`

SceNpMatching2RoomMessageCallback

Room message callback function

Definition

```
#include <np.h>

typedef void (*SceNpMatching2RoomMessageCallback)(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2RoomMemberId srcMemberId,
        SceNpMatching2Event event,
        const void *data,
        void *arg
        );
```

Arguments

ctxId	Context ID
roomId	ID of room where event occurred
srcMemberId	ID of the room member who sent the message
event	Room message event
data	Event data NULL indicates that there is no event data.
arg	Pointer to application-specified data

Description

This callback function is used for notification of events indicating that a room chat message or a room message was received.

Before sending or receiving room chat messages and room messages, register a room message callback function with `sceNpMatching2RegisterRoomMessageCallback()`.

See Also

`sceNpMatching2RegisterRoomMessageCallback()`,
`SCE_NP_MATCHING2_ROOM_MSG_EVENT_*`

SceNpMatching2LobbyEventCallback

Lobby event callback function

Definition

```
#include <np.h>

typedef void (*SceNpMatching2LobbyEventCallback)(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2LobbyId lobbyId,
        SceNpMatching2Event event,
        const void *data,
        void *arg
        );
```

Arguments

ctxId	Context ID
lobbyId	Lobby ID of lobby for which event occurred
event	Room event
data	Event data NULL indicates that there is no event data.
arg	Pointer to arbitrary data specified by the application.

Description

This is the callback function to which a lobby event that occurs while a lobby is joined is reported.

Use `sceNpMatching2RegisterLobbyEventCallback()` to register a lobby event callback function before joining a lobby.

See Also

`sceNpMatching2RegisterLobbyEventCallback(),SCE_NP_MATCHING2_LOBBY_EVENT_*`

SceNpMatching2LobbyMessageCallback

Lobby message callback function

Definition

```
#include <np.h>

typedef void (*SceNpMatching2LobbyMessageCallback)(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2LobbyId lobbyId,
        SceNpMatching2LobbyMemberId srcMemberId,
        SceNpMatching2Event event,
        const void *data,
        void *arg
        );
```

Arguments

ctxId	Context ID
lobbyId	Lobby ID of lobby for which event occurred
srcMemberId	Lobby member ID of user who sent message
Event	Lobby message event
data	Event data NULL indicates that there is no event data.
arg	Pointer to arbitrary data specified by the application.

Description

This is the callback function to which an event that indicates a lobby chat message was received is reported.

Use `sceNpMatching2RegisterLobbyMessageCallback()` to register a lobby message callback function before sending or receiving a lobby chat message.

See Also

`sceNpMatching2RegisterLobbyMessageCallback()`,
`SCE_NP_MATCHING2_LOBBY_MSG_EVENT_*`

SceNpMatching2SignalingCallback

Signaling callback function

Definition

```
#include <np.h>

typedef void (*SceNpMatching2SignalingCallback)(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2RoomMemberId peerMemberId,
        SceNpMatching2Event event,
        int errorCode,
        void *arg
        );
```

Arguments

ctxId	Context ID
roomed	Room ID of room for which event occurred
peerMemberId	Room member ID of user who performed processing for establishing P2P connections
event	Signaling event
errorCode	Error code If this is a negative value, it indicates that an error occurred in P2P connection establishment processing. If it is 0, it indicates that processing was terminated normally (it does not indicate that P2P connections were established).
arg	Pointer to arbitrary data specified by the application.

Description

This is the callback function to which an event that indicates P2P connections were established or disconnected is reported.

Use `sceNpMatching2RegisterSignalingCallback()` to register a signaling callback function before a room is created or the signaling option parameter is set.

See Also

`sceNpMatching2RegisterSignalingCallback()`
`SCE_NP_MATCHING2_SIGNALING_EVENT_*`

SceNpMatching2ContextCallback

Context callback function

Definition

```
#include <np.h>

typedef void (*SceNpMatching2ContextCallback)(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2Event event,
        SceNpMatching2EventCause eventCause,
        int errorCode,
        void *arg
        );
```

Arguments

ctxId	Context ID
event	Context event
eventCause	Cause of context event
errorCode	Error code
arg	Pointer to arbitrary data specified by the application.

Description

This is the callback function to which an event that indicates a state in which the context cannot continue to be used is reported.

Use `sceNpMatching2RegisterContextCallback()` to register a context callback function immediately after an NP Matching 2 library context is created.

See Also

`sceNpMatching2RegisterContextCallback()`
`SCE_NP_MATCHING2_CONTEXT_EVENT_*`

Local Matching Functions

sceNpMatching2GetServerLocal

Get server information

Definition

```
#include <np.h>

int
sceNpMatching2GetServerLocal(
        const SceNpMatching2ContextId ctxId,
        SceNpMatching2Server *server
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
server	Pointer to buffer to store the server information

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function gets the server information.

Examples

```
int ret;
SceNpMatching2Server server;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;

memset(&server, 0, sizeof(server));

ret = sceNpMatching2GetServerLocal(
        ctxId, &server);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2Server

sceNpMatching2GetRoomMemberIdListLocal

Get list of room member IDs

Definition

```
#include <np.h>

int
sceNpMatching2GetRoomMemberIdListLocal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2RoomId roomId,
        const int sortMethod,
        SceNpMatching2RoomMemberId *memberId,
        const SceUInt32 memberIdNum
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
roomId	ID of room to obtain list of room member IDs
sortMethod	Sort method of list of room member IDs
memberId	Pointer to buffer to store room member IDs
memberIdNum	Number of room member IDs to obtain

Return Values

Returns the number of members currently in the room for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute `sceNpMatching2Init()` and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_FOUND	0x80550c09	Server could not be found The server information stored in the library could not be found. The *roomId* may be invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_SORT_METHOD	0x80550c13	Invalid sort method The value specified for *sortMethod* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function gets the list of room members currently in the room. It does not access the servers, but creates a list of room member IDs from information inside the NP Matching 2 library. The list of room member IDs is obtained as an array of `SceNpMatching2RoomMemberId` elements.

To *`memberId`*, specify the buffer prepared by the application. The size of this buffer must be at least `sizeof(SceNpMatching2RoomMemberId) * memberIdNum`. Up to *`memberIdNum`* room member IDs will be copied to the buffer *`memberId`*.

The list of room member IDs copied to the *`memberId`* buffer will be sorted in the method specified in *`sortMethod`*. For information of the sort methods that can be specified, refer to the constant `SCE_NP_MATCHING2_SORT_METHOD_*`.

When this function terminates normally, the current number of room members will be the return value.

Examples

```
int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId memberId[SCE_NP_MATCHING2_ROOM_MAX_SLOT];

//E Get list of room member IDs sorted by the join date/time
ret = sceNpMatching2GetRoomMemberIdListLocal(
        ctxId, roomId, SCE_NP_MATCHING2_SORT_METHOD_JOIN_DATE,
        memberId, SCE_NP_MATCHING2_ROOM_MAX_SLOT);
if (ret < 0) {
    //E Error handling
}
```

See Also

`SceNpMatching2RoomMemberId`, `SCE_NP_MATCHING2_SORT_METHOD_*`

sceNpMatching2GetRoomMemberDataInternalLocal

Get internal room member data (data available for room members)

Definition

```
#include <np.h>

int
sceNpMatching2GetRoomMemberDataInternalLocal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2RoomId roomId,
        const SceNpMatching2RoomMemberId memberId,
        const SceNpMatching2AttributeId *attrId,
        SceUInt32 attrIdNum,
        SceNpMatching2RoomMemberDataInternal *member,
        char *buf,
        SceSize bufLen
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
roomId	ID of current room
memberId	ID of room member to obtain internal room member data
attrId	Pointer to array of IDs of the internal room member attributes
attrIdNum	Number of elements in array of IDs of the internal room member attributes
member	Pointer to buffer to store internal room member data
buf	Pointer to buffer to store data associated with internal room member data structure
bufLen	Size of buffer to store data associated with internal room member data structure

Return Values

Returns the size of data associated with the internal room member data structure for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_FOUND	0x80550c09	Server could not be found The server information stored in the library could not be found. The *roomId* may be invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ATTRIBUTE_ID	0x80550c11	Invalid attribute ID An attribute ID other than the one applicable may have been specified. Check the value of the attribute ID included in the array indicated by *attrId*.
SCE_NP_MATCHING2_ERROR_INSUFFICIENT_BUFFER	0x80550c1b	Size of buffer is not enough The buffer size specified for *bufLen* is smaller than the buffer size required to copy an internal room member data. Specify a sufficient buffer size.

For a list of the NP error codes, refer to each reference document.

Description

This function gets internal room member data of the room member indicated by *memberId*. It does not access the servers, but returns the room member information inside the NP Matching 2 library.

Getting the internal room member attributes is optional. To obtain the internal room member attributes, specify an array of attribute IDs to *attrId*.

When this function is executed, the value of the internal room member data structure is copied to the buffer indicated by *member*, and the data associated with the internal room member data structure (the data pointed to by *member*) is copied to the buffer indicated by *buf*.

The return value of this function is the size of the data associated with the internal room member data structure. To allocate a buffer dynamically, specify NULL to *buf* and execute the function to find the size required for the buffer. Then call the function again with this buffer to obtain the internal room member data.

Examples

This example obtains internal room member data as follows.

- Obtain the value of binary attribute ID 1 of the internal room member data

```
int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId memberId;
//E Internal room member attribute to obtain
SceNpMatching2AttributeId attrId[1];
//E Buffer to copy the internal room member data
SceNpMatching2RoomMemberDataInternal member;
char *buf = NULL;
SceSize bufLen = 0;

//E Internal room attribute to obtain
//  Obtain value of internal room binary attribute ID 1
attrId[0] = SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_1_ID;

//E Obtain the required buffer size
```

```
ret = sceNpMatching2GetRoomMemberDataInternalLocal(
        ctxId, roomId, memberId,
        attrId, 1,
        NULL, NULL, 0);
if (ret < 0) {
    //E Error handling
}

//E Return value is the required buffer size
bufLen = ret;
//E Allocate a buffer
buf = (char *)malloc(bufLen);
if (buf == NULL) {
    //E Error handling
}
memset(buf, 0, bufLen);

//E Obtain internal room member data
ret = sceNpMatching2GetRoomMemberDataInternalLocal(
        ctxId, roomId, memberId,
        attrId, 1,
        &member, buf, bufLen);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2RoomMemberDataInternal

sceNpMatching2GetRoomPasswordLocal

Get room password

Definition

```
#include <np.h>

int
sceNpMatching2GetRoomPasswordLocal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2RoomId roomId,
        bool *withPassword,
        SceNpMatching2SessionPassword *roomPassword
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
roomId	ID of current room
withPassword	Pointer to the buffer storing the flag which indicates whether a room password has been set or not
roomPassword	Pointer to the buffer storing the room password set to the room

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_FOUND	0x80550c09	Server could not be found The server information stored in the library could not be found. The *roomId* may be invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_NOT_ALLOWED	0x80550c2e	User does not have right for executing requested processing Check for room ownership.

For a list of the NP error codes, refer to each reference document.

Description

This function obtains the room password set to the room specified with *roomId*. This function returns the password held internally by the NP Matching 2 library; there is no server access.

This function can only be executed by the room owner.

Examples

```
int ret;

//E Assuming that appropriate values are stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RoomId roomId;

//E Buffer to store the data
bool withPassword;
SceNpMatching2SessionPassword roomPassword;

memset(&roomPassword, 0, sizeof(roomPassword));

//E Get room password
ret = sceNpMatching2GetRoomPasswordLocal(
        ctxId, roomId, &withPassword, &roomPassword);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SessionPassword

sceNpMatching2GetLobbyMemberIdListLocal

Get lobby member ID list (not implemented)

Definition

```
#include <np.h>

int
sceNpMatching2GetLobbyMemberIdListLocal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2LobbyId lobbyId,
        SceNpMatching2LobbyMemberId *memberId,
        SceUInt32 memberIdNum,
        SceNpMatching2LobbyMemberId *me
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
lobbyId	Lobby ID of lobby for which lobby member ID list is to be obtained
memberId	Pointer to buffer where lobby member IDs are stored
memberIdNum	Number of lobby member IDs to be obtained
me	Pointer to buffer in which your own lobby member ID is stored

Return Values

Returns the number of members currently in the lobby for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute `sceNpMatching2Init()` and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_FOUND	0x80550c09	Server could not be found The server information stored in the library could not be found. The *roomId* may be invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID The value specified for *lobbyId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function is not implemented.

This function gets a list of lobby member IDs for the lobby that the user has currently joined. This function creates the lobby member ID list from information that is obtained internally by the NP Matching 2 library. It does not send an inquiry to the server. The lobby member ID list that is obtained will become the `SceNpMatching2LobbyMemberId` array.

For *`memberId`*, specify the buffer that the application prepared in advance. Prepare a buffer with a buffer size of at least `sizeof(SceNpMatching2LobbyMemberId) * memberIdNum`. Lobby member IDs are copied to the *`memberId`* buffer until the number of lobby member IDs is at most *`memberIdNum`*.

If this function is normally terminated, the number of lobby members that are currently joined becomes the return value.

Examples

```
int ret;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
SceNpMatching2LobbyId lobbyId;
SceNpMatching2LobbyMemberId memberId[SCE_NP_MATCHING2_LOBBY_MAX_SLOT];
SceNpMatching2LobbyMemberId me;

//E Get lobby member ID list
ret = sceNpMatching2GetLobbyMemberIdListLocal(
        ctxId, lobbyId, memberId,
        SCE_NP_MATCHING2_LOBBY_MAX_SLOT, &me);
if (ret < 0) {
    //E Error handling
}
```

See Also

`SceNpMatching2LobbyMemberId`

sceNpMatching2GetSignalingOptParamLocal

Get signaling option parameter

Definition

```
#include <np.h>

int
sceNpMatching2GetSignalingOptParamLocal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2RoomId roomId,
        SceNpMatching2SignalingOptParam *signalingOptParam
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
roomId	Room ID of room for which signaling option parameter is to be obtained
signalingOptParam	Pointer to buffer for storing signaling option parameter

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_FOUND	0x80550c09	Server could not be found The server information stored in the library could not be found. The *roomId* may be invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* is invalid. Check its value.

For a list of the NP error codes, refer to each reference document.

Description

This function gets the signaling option parameter that has been set for the room. This function returns the signaling option parameter that is being maintained internally by the NP Matching 2 library. It does not send and inquiry to the server.

Examples

```
int ret;
SceNpMatching2SignalingOptParam sigOptParam;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RoomId roomId;

memset(&sigOptParam, 0, sizeof(sigOptParam));

ret = sceNpMatching2GetSignalingOptParamLocal(
        ctxId, roomId, &sigOptParam);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SignalingOptParam

Request Functions

sceNpMatching2GetWorldInfoList

Get list of world data

Definition

```
#include <np.h>

int
sceNpMatching2GetWorldInfoList(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetWorldInfoListRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_SERVER_ID	0x80550c0c	Invalid server ID The value specified for *serverId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function gets the list of world data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_GET_WORLD_INFO_LIST event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

```
int ret;

SceNpMatching2ContextId ctxId;
SceNpMatching2GetWorldInfoListRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;
```

To *serverId* of the request parameters, specify the ID of the server for which to obtain the list of world data.

```
//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.serverId = serverId;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2GetWorldInfoList(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GetWorldInfoListRequest, SceNpMatching2GetWorldInfoListResponse, SCE_NP_MATCHING2_REQUEST_EVENT_GET_WORLD_INFO_LIST

sceNpMatching2SetUserInfo

Set user information

Definition

```
#include <np.h>

int
sceNpMatching2SetUserInfo(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SetUserInfoRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_SERVER_ID	0x80550c0c	Invalid server ID The value specified for *serverId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sets user information.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

The following user information can be set by this function.

- User binary attribute value

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_SET_USER_INFO event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

There is no event data corresponding to a request due to this function. Therefore, the *data* value that is passed to the request callback function always is NULL.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

The example shown below sets user information under the following conditions.

- Set user binary attribute ID 1

```
int ret;
SceNpMatching2BinAttr userBinAttr[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2SetUserInfoRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E User binary attribute
memset(userBinAttr, 0, sizeof(userBinAttr));
userBinAttr[0].id = SCE_NP_MATCHING2_USER_BIN_ATTR_1_ID;
userBinAttr[0].ptr = "USER_BIN_ATTR";
userBinAttr[0].size = strlen("USER_BIN_ATTR");

//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.serverId = serverId; //E Specify server ID of server that sets user
                              information
reqParam.userBinAttr = userBinAttr;
reqParam.userBinAttrNum = 1;

//E Assume that appropriate values are stored in ctxId and optParam

ret = sceNpMatching2SetUserInfo(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SetUserInfoRequest,
SCE_NP_MATCHING2_REQUEST_EVENT_SET_USER_INFO

sceNpMatching2GetUserInfoList

Get user information list

Definition

```
#include <np.h>

int
sceNpMatching2GetUserInfoList(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetUserInfoListRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_SERVER_ID	0x80550c0c	Invalid server ID The value specified for *serverId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ATTRIBUTE_ID	0x80550c11	Invalid attribute ID An attribute ID other than the one applicable may have been specified. Check the value of the attribute ID included in the array indicated by *attrId* of the request parameters.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function gets a user information list.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

This function can get user information of multiple users at one time. Set the NP IDs of users for which you want to get user information in an array and specify it for the request parameter *npId*. The maximum number of NP IDs that can be specified is SCE_NP_MATCHING2_GET_USER_INFO_LIST_NPID_NUM_MAX.

The acquisition of user attributes is optional. If you want to get user attributes, specify attribute IDs of user attributes that you want to obtain as an array in the request parameter *attrId*.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_GET_USER_INFO_LIST event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

The example shown below gets user information under the following conditions.

- Get attribute value of user binary attribute ID 1

```
int ret;
SceNpId npId[2];
SceNpMatching2AttributeId attrId[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2GetUserInfoListRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E User attribute to be obtained
//E Get attribute value of user binary attribute ID 1
attrId[0] = SCE_NP_MATCHING2_USER_BIN_ATTR_1_ID;

//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.serverId = serverId; //E Server ID of server for which user information
                              is to be obtained
reqParam.npId = npId; //E NP ID array of users for which information is to be
                      obtained
```

```
reqParam.npIdNum = 2;
reqParam.attrId = attrId;
reqParam.attrIdNum = 1;

//E Assume that appropriate values are stored in ctxId and optParam

ret = sceNpMatching2GetUserInfoList(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GetUserInfoListRequest, SceNpMatching2GetUserInfoListResponse, SCE_NP_MATCHING2_REQUEST_EVENT_GET_USER_INFO_LIST

sceNpMatching2CreateJoinRoom

Create and join a room

Definition

```
#include <np.h>

int
sceNpMatching2CreateJoinRoom(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2CreateJoinRoomRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MAX_SLOT	0x80550c14	Invalid total number of slots The value specified for *maxSlot* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function creates and joins a room.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_CREATE_JOIN_ROOM event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

This example creates a room as follows.

- Total number of slots: 8
- Create groups in room: No
- Room password: Set to slot numbers 5 - 8
- Set the total number of slots to the external room search integer attribute ID 1
- No affiliated lobby

```
int ret;
SceNpMatching2WorldId worldId;
SceNpMatching2SessionPassword sessionPassword;
SceNpMatching2RoomPasswordSlotMask roomPasswordSlotMask;
SceNpMatching2IntAttr  roomSearchableIntAttrExternal;

SceNpMatching2ContextId ctxId;
SceNpMatching2CreateJoinRoomRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Total number of slots
int maxSlot = 8;

//E Room password
memset(&sessionPassword, 0, sizeof(sessionPassword));
memcpy(sessionPassword.data, "PASSWORD" strlen("PASSWORD"));

//E Room password slot mask
memset(&roomPasswordSlotMask, 0, sizeof(roomPasswordSlotMask));
for (int i = 0; i < 4; i++) {
        SCE_NP_MATCHING2_ADD_SLOTNUM_TO_ROOM_PASSWORD_SLOT_MASK(
        roomPasswordSlotMask, (maxSlot - i));
}

//E External room search integer attribute
memset(roomSearchableIntAttrExternal, 0, sizeof(SceNpMatching2IntAttr));
roomSearchableIntAttrExternal.id =
        SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_1_ID;
roomSearchableIntAttrExternal.num = maxSlot;
```

```
//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.worldId = worldId; //E Specify the world to which the new room belongs
reqParam.lobbyId = 0;  //E Specify 0 if the room does not belong to a lobby
//E Parameters for creating a room
reqParam.maxSlot = maxSlot;
reqParam.groupConfig = NULL;
reqParam.groupConfigNum = 0;
reqParam.roomPassword = &sessionPassword;
reqParam.passwordSlotMask = &roomPasswordSlotMask;
reqParam.flagAttr = 0;
reqParam.roomSearchableIntAttrExternal = &roomSearchableIntAttrExternal;
reqParam.roomSearchableIntAttrExternalNum = 1;
reqParam.roomBinAttrInternal = NULL;
reqParam.roomBinAttrInternalNum = 0;
reqParam.roomSearchableBinAttrExternal = NULL;
reqParam.roomSearchableBinAttrExternalNum = 0;
reqParam.roomBinAttrExternal = NULL;
reqParam.roomBinAttrExternalNum = 0;
reqParam.allowedUser = NULL;
reqParam.allowedUserNum = 0;
reqParam.blockedUser = NULL;
reqParam.blockedUserNum = 0;
//E Parameters for joining a room
reqParam.roomMemberBinAttrInternal = NULL;
reqParam.roomMemberBinAttrInternalNum = 0;
reqParam.teamId = 0;
reqParam.joinRoomGroupLabel = NULL;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2CreateJoinRoom(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2CreateJoinRoomRequest

SceNpMatching2CreateJoinRoomResponse

SCE_NP_MATCHING2_REQUEST_EVENT_CREATE_JOIN_ROOM

sceNpMatching2SearchRoom

Search for rooms

Definition

```
#include <np.h>

int
sceNpMatching2SearchRoom(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SearchRoomRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MATCHING_SPACE	0x80550c16	Invalid search space The world ID or lobby ID specified as the search target is invalid. Check the *worldId* or *lobbyId* value of the request parameters.
SCE_NP_MATCHING2_ERROR_RANGE_FILTER_MAX	0x80550c1a	Larger than the maximum number of elements obtainable specified in the range filter Check the value specified for the maximum number of obtainable elements.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function searches for rooms.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

This function searches for a room that matches the search condition specified in the SceNpMatching2SearchRoomRequest request parameter, and from the resulting list of rooms, obtains the list of rooms that falls within the range also specified in the request parameter.

When the SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_RANDOM is specified, a room selected at random from among the list of rooms matching the search condition can be obtained.

In addition to the list of rooms (the result of executing the room search), it is also possible to obtain information of the room owner. (Refer to the information of the external room data structure SceNpMatching2RoomDataExternal.) Specify the user information and NP ID to obtain to the room search options in the request parameters.

It is also possible to obtain the values of the following attributes.

- External room search integer attribute
- External room search binary attribute
- External room binary attribute

Specify an array of attribute IDs to *attrId* in the request parameters.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_SEARCH_ROOM event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

This example searches for a room as follows. This is the room created in Examples of sceNpMatching2CreateJoinRoom().

- Total number of slots: 8 (Value of external room search integer attribute ID 1 is 8)

Search parameters are set as follows.

- Search for a room belonging to a world (but not specifying the lobby)
- Get the NP ID of the room owner
- Get the value of the external room search integer attribute ID 1
- Range: obtain maximum number of rooms that can be obtained (20) starting from position 0

```
int ret;
```

```
SceNpMatching2IntSearchFilter intFilter[1];
SceNpMatching2AttributeId attrId[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2SearchRoomRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Search conditions
// Search for room whose total number of slots is 8 (Value of external room search
//  value attribute ID 1 is 8)
memset(intFilter, 0, sizeof(intFilter));
intFilter[0].searchOperator = SCE_NP_MATCHING2_OPERATOR_EQ;
intFilter[0].attr.id =
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_1_ID;
intFilter[0].attr.num = 8;

//E Room data to obtain
// Get the value of external room search integer attribute ID 1
attrId[0] = SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_1_ID;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.worldId = worldId; //E Specify the world to search
reqParam.lobbyId = 0;  //E Specify 0 not to specify the lobby
//E Range of rooms to obtain is the maximum number of rooms that can be obtained
// (20), starting from position 0
reqParam.rangeFilter.startIndex =
        SCE_NP_MATCHING2_RANGE_FILTER_START_INDEX_MIN;
reqParam.rangeFilter.max = SCE_NP_MATCHING2_RANGE_FILTER_MAX;
reqParam.flagFilter = 0;
reqParam.flagAttr = 0;
reqParam.intFilter = intFilter;
reqParam.intFilterNum = 1;
reqParam.binFilter = NULL;
reqParam.binFilterNum = 0;
reqParam.attrId = attrId;
reqParam.attrIdNum = 1;
//E Get only the NP ID of the room owner
reqParam.option = SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_WITH_NPID;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2SearchRoom(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SearchRoomRequest

SceNpMatching2SearchRoomResponse

SCE_NP_MATCHING2_REQUEST_EVENT_SEARCH_ROOM

SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_*

sceNpMatching2JoinRoom

Join a room

Definition

```
#include <np.h>

int
sceNpMatching2JoinRoom(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2JoinRoomRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function joins a room.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

If *roomMemberBinAttrInternal*, *teamId* in the request parameters are set, these will be the initial values of the internal room member binary attribute and team ID upon joining a room.

If *optData* in the request parameters is set, the optional data will be added to the notification to other room members upon joining a room.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_JOIN_ROOM event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

This example joins a room as follows.

- Join a room not organized into groups
- Join a reserved slot accessible with a room password
- Set the initial team ID to 1
- Set optional presence data

```
int ret;
SceNpMatching2RoomId roomId;
SceNpMatching2SessionPassword sessionPassword;

SceNpMatching2ContextId ctxId;
SceNpMatching2JoinRoomRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Room password
memset(&sessionPassword, 0, sizeof(sessionPassword));
memcpy(sessionPassword.data, "PASSWORD" strlen("PASSWORD"));

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E ID of room to join
reqParam.roomPassword = &sessionPassword; //E Room password of reserved slot
reqParam.joinRoomGroupLabel = NULL;
reqParam.teamId = 1; //E Initial team ID
reqParam.roomMemberBinAttrInternal = NULL;
reqParam.roomMemberBinAttrInternalNum = 0;
//E Optional presence data
memcpy(&reqParam.optData.data, "HELLO", strlen("HELLO"));
reqParam.optData.len = strlen("HELLO") + 1;

//E Assuming that appropriate values are stored to ctxId, optParam
```

```
ret = sceNpMatching2JoinRoom(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2JoinRoomRequest

SceNpMatching2JoinRoomResponse

SCE_NP_MATCHING2_REQUEST_EVENT_JOIN_ROOM

sceNpMatching2LeaveRoom

Leave a room

Definition

```
#include <np.h>

int
sceNpMatching2LeaveRoom(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2LeaveRoomRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function leaves a room.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

If optional presence data (*optData* in the request parameters) is set, the optional data will be added to the notification to other room members upon leaving a room.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_LEAVE_ROOM event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

There is no event data associated with the request of this function. For this reason, the value of *data* passed to the request callback function is always NULL.

Examples

```
int ret;
SceNpMatching2RoomId roomId;

SceNpMatching2ContextId ctxId;
SceNpMatching2LeaveRoomRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E ID of room to leave
//E Optional presence data
memcpy(&reqParam.optData.data, "BYE", strlen("BYE"));
reqParam.optData.len = strlen("BYE") + 1;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2LeaveRoom(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2LeaveRoomRequest

SCE_NP_MATCHING2_REQUEST_EVENT_LEAVE_ROOM

sceNpMatching2GetRoomDataInternal

Get internal room data

Definition

```
#include <np.h>

int
sceNpMatching2GetRoomDataInternal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetRoomDataInternalRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function gets internal room data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

This function also provides the option of getting the internal room attributes. To obtain the internal room attributes, specify the IDs of the applicable attributes in an array to *attrId* in the request parameters.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_DATA_INTERNAL event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Examples

This example gets internal room data as follows.

- Get the value of internal room binary attribute ID 1

```
int ret;
SceNpMatching2AttributeId attrId[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2GetRoomDataInternalRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Internal room attribute to get
//  Get value of internal room binary attribute ID 1
attrId[0] = SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_1_ID;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Specify ID of current room
reqParam.attrId = attrId;
reqParam.attrIdNum = 1;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2GetRoomDataInternal(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0){
    //E Error handling
}
```

See Also

SceNpMatching2GetRoomDataInternalRequest

SceNpMatching2GetRoomDataInternalResponse

SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_DATA_INTERNAL

sceNpMatching2SetRoomDataInternal

Set internal room data

Definition

```
#include <np.h>

int
sceNpMatching2SetRoomDataInternal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SetRoomDataInternalRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ATTRIBUTE_ID	0x80550c11	Invalid attribute ID An attribute ID other than of the applicable attribute may have been specified. Check the value of the attribute ID included in the array indicated by *attrId* of the request parameters.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sets internal room data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

The internal room data that can be set with this function differs depending on whether the user is that room's owner or not, as follows.

Internal Room Data	Room Owner	Room Member
Room flag attributes	Yes	No
Internal room binary attributes	Yes	Yes
Room password per group for a group room	Yes	No
Room password per slot for a room that is not a group room	Yes	No
Priority order of the room members when automatically transferring room ownership	Yes	No

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_DATA_INTERNAL event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

There is no event data associated with the request of this function. For this reason, the value of *data* passed to the request callback function is always NULL.

Examples

This example sets internal room data as follows.

- Set the HIDDEN flag attribute (Hide room from searches)
- Set the CLOSED flag attribute (Set room to CLOSED)

```
int ret;

SceNpMatching2ContextId ctxId;
SceNpMatching2SetRoomDataInternalRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Specify ID of current room
reqParam.flagFilter = (SCE_NP_MATCHING2_ROOM_FLAG_ATTR_HIDDEN |
                       SCE_NP_MATCHING2_ROOM_FLAG_ATTR_CLOSED);
reqParam.flagAttr = (SCE_NP_MATCHING2_ROOM_FLAG_ATTR_HIDDEN |
                     SCE_NP_MATCHING2_ROOM_FLAG_ATTR_CLOSED);
reqParam.roomBinAttrInternal = NULL;
reqParam.roomBinAttrInternalNum = 0;
reqParam.passwordConfig = NULL;
reqParam.passwordConfigNum = 0;
reqParam.passwordSlotMask = NULL;
reqParam.ownerPrivilegeRank = NULL;
```

```
reqParam.ownerPrivilegeRankNum = 0

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2SetRoomDataInternal(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SetRoomDataInternalRequest
SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_DATA_INTERNAL

sceNpMatching2GetRoomDataExternalList

Get external room data list

Definition

```
#include <np.h>

int
sceNpMatching2GetRoomDataExternalList(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetRoomDataExternalListRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function gets a list of external room data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

This function can obtain the external room data of multiple rooms at once. Specify the IDs of the applicable rooms in an array to *roomId* in the request parameters.

This function also provides the option of getting the external room attributes. To obtain the external room attributes, specify the IDs of the applicable attributes in an array to *attrId* in the request parameters.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_DATA_EXTERNAL_LIST event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

This example gets external room data as follows.

- Get the value of external room binary attribute ID 1
- Get the value of external room search integer attribute ID 1

```
int ret;
SceNpMatching2RoomId roomId[2];
SceNpMatching2AttributeId attrId[2];

SceNpMatching2ContextId ctxId;
SceNpMatching2GetRoomDataExternalListRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Target external room attributes
//  Get the value of external room binary attribute ID 1
attrId[0] = SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_1_ID;
//  Get the value of external room search integer attribute ID 1
attrId[1] = SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_1_ID;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Array of target room IDs
reqParam.roomIdNum = 2;
reqParam.attrId = attrId;
reqParam.attrIdNum = 2;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2GetRoomDataExternalList(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
```

```
        //E Error handling
    }
```

See Also

SceNpMatching2GetRoomDataExternalListRequest
SceNpMatching2GetRoomDataExternalListResponse
SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_DATA_EXTERNAL_LIST

sceNpMatching2SetRoomDataExternal

Set external room data

Definition

```
#include <np.h>

int
sceNpMatching2SetRoomDataExternal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SetRoomDataExternalRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sets external room data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

The external room data that can be set with this function differs depending on whether the user is that room's owner or not, as follows.

External Room Data	Room Owner	Room Member
Value of external room search integer attribute	Yes	No
Value of external room search binary attribute	Yes	No
Value of external room binary attribute	Yes	No

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_DATA_EXTERNAL event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

There is no event data associated with the request of this function. For this reason, the value of *data* passed to the request callback function is always NULL.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

This example sets external room data as follows.

- Set the external room search integer attribute ID 1

```
int ret;
SceNpMatching2IntAttr roomSearchableIntAttrExternal[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2SetRoomDataExternalRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Set the external room search integer attribute ID 1
roomSearchableIntAttrExternal[0].id =
        SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_1_ID;
roomSearchableIntAttrExternal[0].num = 1;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Specify ID of target room
reqParam.roomSearchableIntAttrExternal = roomSearchableIntAttrExternal;
reqParam.roomSearchableIntAttrExternalNum = 1;
reqParam.roomSearchableBinAttrExternal = NULL;
reqParam.roomSearchableBinAttrExternalNum = 0;
reqParam.roomBinAttrExternal = NULL;
reqParam.roomBinAttrExternalNum = 0;

//E Assuming that appropriate values are stored to ctxId, optParam
```

```
ret = sceNpMatching2SetRoomDataExternal(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SetRoomDataExternalRequest

SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_DATA_EXTERNAL

sceNpMatching2GetRoomMemberDataInternal

Get internal room member data

Definition

```
#include <np.h>

int
sceNpMatching2GetRoomMemberDataInternal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetRoomMemberDataInternalRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function gets internal room member data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

This function also provides the option of getting the internal room member attributes. To obtain the internal room member attributes, specify the IDs of the applicable attributes in an array to *attrId* in the request parameters.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_MEMBER_DATA_INTERNAL event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Examples

This example gets internal room member data as follows.

- Get the value of internal room member binary attribute ID 1

```
int ret;
SceNpMatching2AttributeId attrId[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2GetRoomMemberDataInternalRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Internal room member attribute to get
//  Get value of internal room member binary attribute ID 1
attrId[0] = SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_1_ID;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Specify ID of current room
reqParam.memberId = memberId; //E Specify ID of target room member
reqParam.attrId = attrId;
reqParam.attrIdNum = 1;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2GetRoomMemberDataInternal(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GetRoomMemberDataInternalRequest

SceNpMatching2GetRoomMemberDataInternalResponse

SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_MEMBER_DATA_INTERNAL

sceNpMatching2SetRoomMemberDataInternal

Set internal room member data

Definition

```
#include <np.h>

int
sceNpMatching2SetRoomMemberDataInternal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SetRoomMemberDataInternalRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ATTRIBUTE_ID	0x80550c11	Invalid attribute ID An attribute ID other than that of the applicable attribute may have been specified. Check the value of the attribute ID included in the array indicated by *attrId* of the request parameters.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sets internal room member data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

This function can set the following internal room member data.

- Team ID
- Value of internal room member binary attribute

Currently, there are no internal room member flag attributes that can be set by this function, so *flagFilter*, *flagAttr* in the request parameters are not used.

When *teamId* in the request parameters is 0, this indicates that the team ID will not be set. When *memberId* is 0, this indicates that the target is oneself.

Only the room owner can set the internal room member data of other room members.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_MEMBER_DATA_INTERNAL event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

There is no event data associated with the request of this function. For this reason, the value of *data* passed to the request callback function is always NULL.

Examples

This example sets internal room member data as follows.

- Set the internal room member binary attribute ID 1
- No team ID
- Target is one's own internal room member data

```
int ret;
SceNpMatching2BinAttr roomMemberBinAttrInternal[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2SetRoomMemberDataInternalRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Internal room member binary attribute
memset(roomMemberBinAttrInternal, 0, sizeof(roomMemberBinAttrInternal));
roomMemberBinAttrInternal[0].id =
        SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_1_ID;
roomMemberBinAttrInternal[0].ptr = "ROOM_MEMBER_BIN_ATTR";
roomMemberBinAttrInternal[0].size = strlen("ROOM_MEMBER_BIN_ATTR");

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId;    //E Specify ID of current room
```

```
reqParam.memberId = 0;       //E Target is oneself
reqParam.teamId = 0;         //E Not set
reqParam.flagFilter = 0;     //E Unused
reqParam.flagAttr = 0;       //E Unused
reqParam.roomMemberBinAttrInternal = roomMemberBinAttrInternal;
reqParam.roomMemberBinAttrInternalNum = 1;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2SetRoomMemberDataInternal(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SetRoomMemberDataInternalRequest

SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_MEMBER_DATA_INTERNAL

sceNpMatching2GetRoomMemberDataExternalList

Get list of external room member data

Definition

```
#include <np.h>

int
sceNpMatching2GetRoomMemberDataExternalList(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetRoomMemberDataExternalListRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_SORT_METHOD	0x80550c13	Invalid sort method The value specified for *sortMethod* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function gets a list of external room member data.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_MEMBER_DATA_EXTERNAL_LIST event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

```
int ret;

SceNpMatching2ContextId ctxId;
SceNpMatching2GetRoomMemberDataExternalListRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Specify ID of target room

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2GetRoomMemberDataExternalList(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GetRoomMemberDataExternalListRequest

SceNpMatching2GetRoomMemberDataExternalListResponse

SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_MEMBER_DATA_EXTERNAL_LIST

sceNpMatching2KickoutRoomMember

Kick out a room member

Definition

```
#include <np.h>

int
sceNpMatching2KickoutRoomMember(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2KickoutRoomMemberRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for the *memberId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_BLOCK_KICK_FLAG	0x80550c18	Invalid setting for rejoin after kickout The value specified for *blockKickFlag* of the request parameters is invalid. Check its value.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function kicks a room member out of a room.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

Only the room owner is able to execute this function.

Set the room member's status with regards to rejoining the room to *blockKickFlag* in the request parameters. If SCE_NP_MATCHING2_BLOCKKICKFLAG_NG is specified, the room member who was kicked out will not be allowed to rejoin the room.

If optional presence data (*optData* in the request parameters) is set, the optional data will be added to the notification to other room members when the user leaves the room.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_KICKOUT_ROOM_MEMBER event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

There is no event data associated with the request of this function. For this reason, the value of *data* passed to the request callback function is always NULL.

Examples

This example kicks out a room member as follows.

- Forbid the room member from rejoining

```
int ret;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId memberId;

SceNpMatching2ContextId ctxId;
SceNpMatching2KickoutRoomMemberRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E ID of current room
reqParam.target = memberId; //E ID of room member to kick out
reqParam.blockKickFlag = SCE_NP_MATCHING2_BLOCKKICKFLAG_NG; //E Not allowed to

//rejoin
//E Optional presence data
memcpy(&reqParam.optData.data, "BYE", strlen("BYE"));
reqParam.optData.len = strlen("BYE") + 1;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2KickoutRoomMember(
    ctxId, &reqParam, &optParam, &assignedReqId);
```

```
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2KickoutRoomMemberRequest

SCE_NP_MATCHING2_REQUEST_EVENT_KICKOUT_ROOM_MEMBER

sceNpMatching2GrantRoomOwner

Transfer room ownership

Definition

```
#include <np.h>

int
sceNpMatching2GrantRoomOwner(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GrantRoomOwnerRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function transfers room ownership to another room member.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

Only the room owner is able to execute this function. When the request is executed successfully, the room owner will become a regular room member without room ownership.

If optional presence data (*optData* in the request parameters) is set, the optional data will be added to the notification to other room members upon the change in room ownership.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_GRANT_ROOM_OWNER event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

There is no event data associated with the request of this function. For this reason, the value of *data* passed to the request callback function is always NULL.

Examples

```
int ret;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId memberId;

SceNpMatching2ContextId ctxId;
SceNpMatching2GrantRoomOwnerRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E ID of current room
reqParam.newOwner = memberId; //E ID of room member to be given room ownership
//E Optional presence data
memcpy(&reqParam.optData.data, "GIVE YOU", strlen("GIVE YOU"));
reqParam.optData.len = strlen("GIVE YOU") + 1;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2GrantRoomOwner(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GrantRoomOwnerRequest

SCE_NP_MATCHING2_REQUEST_EVENT_GRANT_ROOM_OWNER

sceNpMatching2SendRoomMessage

Send a room message

Definition

```
#include <np.h>

int
sceNpMatching2SendRoomMessage(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SendRoomMessageRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CASTTYPE	0x80550c12	Invalid message cast type The value specified for *castType* of the request parameters is invalid. Check its value.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_INVALID_MESSAGE_TARGET	0x80550c19	Invalid message target Multicast is specified for the message cast type but the member IDs to whom the message is to be sent may not be specified.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sends a room message.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

Specify one of the following message cast types to *castType* in the request parameters.

- Broadcast (SCE_NP_MATCHING2_CASTTYPE_BROADCAST): Message is sent to all room members including the sender
- Unicast (SCE_NP_MATCHING2_CASTTYPE_UNICAST): Message is sent to a room member with the specified ID
- Multicast (SCE_NP_MATCHING2_CASTTYPE_MULTICAST): Message is sent to multiple room members specified by their IDs
- Multicast to a team (SCE_NP_MATCHING2_CASTTYPE_MULTICAST_TEAM): Message is sent to all room members with the specified team ID in the internal room member data

The request parameter *dst* is a union that indicates the intended recipients of the message. Set appropriate destinations according to *castType*. If *castType* is set to broadcast, it is not necessary to set this request parameter.

To the request parameter *option*, specify the send options. By specifying send options, the sender's NP ID can be sent along with the message data. In turn, the *srcMember* member in the SceNpMatching2RoomMessageInfo structure of the recipient will not be NULL, and will instead have a value to the SceNpId structure buffer.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_SEND_ROOM_MESSAGE event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

When the request is executed successfully, this indicates only that the message reached the matching server and the matching server finished sending the message to the specified room members. It does not guarantee the arrival of the message at the destinations.

There is no event data associated with the request of this function. For this reason, the value of *data* passed to the request callback function is always NULL.

The maximum size of a room message that can be sent at one time using this function is 1024 bytes. The macro definition representing the maximum size of a room message is SCE_NP_MATCHING2_BIN_MSG_MAX_SIZE.

Message flow limits are applied by the server to room message transmissions according to this function. Make sure that room messages sent by each client do not exceed 512 bytes/sec. If room messages continue to be sent at fixed intervals with a frequency that exceeds this limit, a request overflow error (SCE_NP_MATCHING2_SERVER_ERROR_REQUEST_OVERFLOW) will be returned to the request callback, room messages will not be able to be sent, and a subsequent 30-second busy state will continue.

Note that the following overhead is applied to the message flow limit of room messages.

- Header: 45 bytes
- When not SCE_NP_MATCHING2_CASTTYPE_BROADCAST: +20 bytes
- When SCE_NP_MATCHING2_SEND_MSG_OPTION_WITH_NPID is specified: +28 bytes

Examples

This example sends a room message as follows.

- Send a multicast
- Send the sender's NP ID

```
int ret;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId memberId[2];

SceNpMatching2ContextId ctxId;
SceNpMatching2SendRoomMessageRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E ID of current room
reqParam.castType = SCE_NP_MATCHING2_CASTTYPE_MULTICAST;
reqParam.dst.multicastTarget.memberId = memberId; //E Target recipients
reqParam.dst.multicastTarget.memberIdNum = 2; //E Specify two recipients
reqParam.msg = "ROOM MESSAGE"; //E Message data
reqParam.msgLen = strlen("ROOM MESSAGE");
//E Send the sender's NP ID
reqParam.option = SCE_NP_MATCHING2_SEND_MSG_OPTION_WITH_NPID;

//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2SendRoomMessage(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SendRoomMessageRequest

SCE_NP_MATCHING2_REQUEST_EVENT_SEND_ROOM_MESSAGE

SCE_NP_MATCHING2_SEND_MSG_OPTION_*

sceNpMatching2SendRoomChatMessage

Send a room chat message

Definition

```
#include <np.h>

int
sceNpMatching2SendRoomChatMessage(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SendRoomChatMessageRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer to store the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CASTTYPE	0x80550c12	Invalid message cast type The value specified for *castType* of the request parameters is invalid. Check its value.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_INVALID_MESSAGE_TARGET	0x80550c19	Invalid message target Multicast is specified for the message cast type, but the member IDs to whom the message is to be sent may not be specified.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sends a room chat message.

When this function terminates normally, the request ID is stored to the area specified with *assignedReqId*.

For information regarding the use of this function and specifying the recipients, refer to the section on sceNpMatching2SendRoomMessage(). Below, the differences from sceNpMatching2SendRoomMessage() are described.

The result of executing this request is notified to the request callback function along with an SCE_NP_MATCHING2_REQUEST_EVENT_SEND_ROOM_CHAT_MESSAGE event. If *errorCode* passed to the request callback function is 0, this indicates that the request was successful. A negative value indicates that an error occurred while processing the request.

Only UTF-8 is supported as text data to be sent with this function.

The maximum size of the room chat message that can be sent at one time using this function is 1024 bytes. The macro definition representing the maximum size of a room chat message is SCE_NP_MATCHING2_CHAT_MSG_MAX_SIZE.

The *filtered* member in the event data indicates that the server's vulgarity filter deleted inappropriate terms found in the message data of the room chat message.

Examples

This example sends a room chat message as follows.

- Send a broadcast
- Do not send sender information

```
int ret;
SceNpMatching2RoomId roomId;

SceNpMatching2ContextId ctxId;
SceNpMatching2SendRoomChatMessageRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E ID of current room
reqParam.castType = SCE_NP_MATCHING2_CASTTYPE_BROADCAST;
reqParam.msg = "ROOM CHAT MESSAGE"; //E Message data
reqParam.msgLen = strlen("ROOM CHAT MESSAGE");
//E Do not send sender information
reqParam.option = 0;
```

```
//E Assuming that appropriate values are stored to ctxId, optParam

ret = sceNpMatching2SendRoomChatMessage(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SendRoomChatMessageRequest
SceNpMatching2SendRoomChatMessageResponse
SCE_NP_MATCHING2_REQUEST_EVENT_SEND_ROOM_CHAT_MESSAGE
SCE_NP_MATCHING2_SEND_MSG_OPTION_*

sceNpMatching2SetSignalingOptParam

Set signaling option parameter

Definition

```
#include <np.h>

int
sceNpMatching2SetSignalingOptParam(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SetSignalingOptParamRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sets the signaling option parameter.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

This function can only be executed for the room that has been joined while the room is joined. The signaling option parameter can only be set by the room owner.

If the signaling option parameter is set by using this function, processing for establishing P2P connections is started immediately according to the setting. P2P connection establishment or disconnection events are reported to the signaling callback function.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_SET_SIGNALING_OPT_PARAM event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

There is no event data corresponding to a request due to this function. Therefore, the *data* value that is passed to the request callback function always is NULL.

Examples

The example shown below sets the signaling option parameter under the following conditions.

- The P2P connection topology is star type
- The room owner becomes the hub

```
int ret;

SceNpMatching2ContextId ctxId;
SceNpMatching2SetSignalingOptParamRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Specify the room ID of the joined room
reqParam.sigOptParam.type = SCE_NP_MATCHING2_SIGNALING_TYPE_STAR;
//E 0 indicates the room owner becomes the hub
reqParam. sigOptParam. sigOptParam.hubMemberId = 0;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2SetSignalingOptParam(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SetSignalingOptParamRequest

SCE_NP_MATCHING2_REQUEST_EVENT_SET_SIGNALING_OPT_PARAM

SCE_NP_MATCHING2_SIGNALING_TYPE_*

sceNpMatching2GetLobbyInfoList

Get lobby information list

Definition

```
#include <np.h>

int
sceNpMatching2GetLobbyInfoList(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetLobbyInfoListRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MATCHING_SPACE	0x80550c16	Invalid search space The world ID specified as the search space may be invalid. Check the value specified for *worldId* of the request parameters.
SCE_NP_MATCHING2_ERROR_RANGE_FILTER_MAX	0x80550c1a	Larger than the maximum number of elements obtainable specified in the range filter Check the value specified for the maximum number of obtainable elements.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function gets a lobby information list.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_INFO_LIST event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

```
int ret;

SceNpMatching2ContextId ctxId;
SceNpMatching2GetLobbyInfoListRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;
```

For the request parameter *worldId*, specify the world ID of the world for which you want to get the lobby information list.

```
//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.worldId = worldId;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2GetLobbyInfoList(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GetLobbyInfoListRequest

SceNpMatching2GetLobbyInfoListResponse

SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_INFO_LIST

sceNpMatching2JoinLobby

Join lobby

Definition

```
#include <np.h>

int
sceNpMatching2JoinLobby(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2JoinLobbyRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID The value specified for *lobbyId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function joins a lobby.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

By specifying the request parameters *joinedSessionInfo* and *lobbyMemberBinAttrInternal*, you can set initial values of joined session information and lobby-internal lobby member binary attribute values when the lobby is joined.

Also, if presence option data (request parameter *optData*) is set, that option data is added to lobby join notices sent to other lobby members.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_JOIN_LOBBY event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

Notes

The execution order of this function and the execution order on the server are not guaranteed to be the same. Design your application so that it is not dependent on execution orders. For details, refer to the description on the execution order of request functions and of the server in the document "NP Matching 2 Library Overview".

Examples

The example shown below joins a lobby under the following conditions.

- Set presence option data

```
int ret;
SceNpMatching2LobbyId lobbyId;

SceNpMatching2ContextId ctxId;
SceNpMatching2JoinLobbyRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.lobbyId = lobbyId; //E Lobby ID of lobby that is to be joined
reqParam.joinedSessionInfo = NULL;
reqParam.joinedSessionInfoNum = 0;
reqParam.lobbyMemberBinAttrInternal = NULL;
reqParam.lobbyMemberBinAttrInternalNum = 0;
//E Presence option data
memcpy(&reqParam.optData.data, "HELLO", strlen("HELLO"));
reqParam.optData.len = strlen("HELLO") + 1;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2JoinLobby(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2JoinLobbyRequest
SceNpMatching2JoinLobbyResponse
SCE_NP_MATCHING2_REQUEST_EVENT_JOIN_LOBBY

sceNpMatching2LeaveLobby

Leave lobby

Definition

```
#include <np.h>

int
sceNpMatching2LeaveLobby(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2LeaveLobbyRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID The value specified for *lobbyId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function leaves a lobby.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

If presence option data (request parameter *optData*) is set, that option data is added to lobby leave notices sent to other lobby members.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_LEAVE_LOBBY event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

There is no event data corresponding to a request due to this function. Therefore, the *data* value that is passed to the request callback function always is NULL.

Examples

```
int ret;
SceNpMatching2LobbyId lobbyId;

SceNpMatching2ContextId ctxId;
SceNpMatching2LeaveLobbyRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.lobbyId = lobbyId; //E Lobby ID of lobby that is to be left
//E Presence option data
memcpy(&reqParam.optData.data, "BYE", strlen("BYE"));
reqParam.optData.len = strlen("BYE") + 1;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2LeaveLobby(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2LeaveLobbyRequest

SCE_NP_MATCHING2_REQUEST_EVENT_LEAVE_LOBBY

sceNpMatching2GetLobbyMemberDataInternal

Get lobby-internal lobby member information (not implemented)

Definition

```
#include <np.h>

int
sceNpMatching2GetLobbyMemberDataInternal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetLobbyMemberDataInternalRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID The value specified for *lobbyId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function is not implemented.

This function gets lobby-internal lobby member information.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

The acquisition of lobby member internal attributes is optional. If you want to get lobby member internal attributes, specify attribute IDs of lobby member internal attributes that you want to obtain as an array in the request parameter *attrId*.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_MEMBER_DATA_INTERNAL event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

Examples

The example shown below gets lobby-internal lobby member information under the following conditions.

- Get attribute values of lobby member internal binary attribute ID 1

```
int ret;
SceNpMatching2AttributeId attrId[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2GetLobbyMemberDataInternalRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Lobby member internal attributes to be obtained
//E Get attribute values of lobby member internal binary attribute ID 1
attrId[0] = SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_1_ID;

// Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.lobbyId = lobbyId; //E Specify lobby ID of joined lobby
reqParam.memberId = memberId; //E Specify lobby member ID of lobby member for
                              which information is to be obtained
reqParam.attrId = attrId;
reqParam.attrIdNum = 1;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2GetLobbyMemberDataInternal(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GetLobbyMemberDataInternalRequest

SceNpMatching2GetLobbyMemberDataInternalResponse

SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_MEMBER_DATA_INTERNAL

sceNpMatching2GetLobbyMemberDataInternalList

Get list of lobby-internal lobby member information (not implemented)

Definition

```
#include <np.h>

int
sceNpMatching2GetLobbyMemberDataInternalList(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2GetLobbyMemberDataInternalListRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to buffer for storing the request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID The value specified for *lobbyId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function is not implemented.

This function gets a list of lobby-internal lobby member information.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

The acquisition of lobby member internal attributes is optional. If you want to get lobby member internal attributes, specify attribute IDs of lobby member internal attributes that you want to obtain as an array in the request parameter *attrId*.

To obtain information of the user joined-in session, and the internal lobby member attributes, set true to the extension data obtainment flag (*extendedData*) for the list of lobby member information. When you require the extension data, the maximum number of member information you can obtain is SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_EXTENDED_DATA_LIST_MAX. If you do not require the extension data, the maximum number of member information you can obtain is SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_LIST_MAX.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_MEMBER_DATA_INTERNAL_LIST event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

Example

The example shown below obtains a list of lobby-internal lobby member information under the following condition.

- Get attribute values of lobby member internal binary attribute ID 1

```
int ret;
SceNpMatching2AttributeId attrId[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2GetLobbyMemberDataInternalListRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

// E Lobby member internal attributes to be obtain:
//  Get attribute values of lobby member internal binary attribute ID 1
attrId[0] = SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_1_ID;

//E Array of lobby member IDs to obtain
#define LOBBY_MEMBER_DATA_LIST_NUM 10
SceNpMatching2LobbyMemberId memberId[LOBBY_MEMBER_DATA_LIST_NUM];
ret = sceNpMatching2GetLobbyMemberIdListLocal(
    ctxId, lobbyId, memberId, LOBBY_MEMBER_DATA_LIST_NUM, NULL);
if (ret < 0) {
    //E Error handling
}

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.lobbyId = lobbyId; //E Specify lobby ID of joined-in lobby
reqParam.memberId = memberId; //E Lobby member ID array of target lobby members
reqParam.memberIdNum = LOBBY_MEMBER_DATA_LIST_NUM;
reqParam.attrId = attrId;
```

```
reqParam.attrIdNum = 1;
reqParam.extendedData = true;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2GetLobbyMemberDataInternalList(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2GetLobbyMemberDataInternalListRequest
SceNpMatching2GetLobbyMemberDataInternalListResponse
SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_MEMBER_DATA_INTERNAL_LIST
SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_LIST_MAX
SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_EXTENDED_DATA_LIST_MAX

sceNpMatching2SetLobbyMemberDataInternal

Set lobby-internal lobby member information (not implemented)

Definition

```
#include <np.h>

int
sceNpMatching2SetLobbyMemberDataInternal(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SetLobbyMemberDataInternalRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID The value specified for *lobbyId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ATTRIBUTE_ID	0x80550c11	Invalid attribute ID An attribute ID other than that of the applicable attribute may have been specified. Check the attribute ID included in the array indicated by *attrId* of the request parameters.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function is not implemented.

This function sets lobby-internal lobby member information.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

The following lobby-internal lobby member information can be set by this function.

- Joined session information
- Lobby member internal binary attribute values

Currently, the request parameters *flagFilter* and *flagAttr* are not used because no lobby member internal flag attributes can be set by this function.

When the request parameter *memberId* is 0, it means that lobby-internal lobby member information is to be set for yourself.

Only the lobby owner can set lobby-internal lobby member information for a lobby member other than him/herself.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_SET_LOBBY_MEMBER_DATA_INTERNAL event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

There is no event data corresponding to a request due to this function. Therefore, the *data* value that is passed to the request callback function always is NULL.

Examples

The example shown below sets lobby-internal lobby member information under the following conditions.

- Set lobby member internal binary attribute ID 1
- Do not set joined session information
- Set lobby member information for yourself

```
int ret;
SceNpMatching2BinAttr lobbyMemberBinAttrInternal[1];

SceNpMatching2ContextId ctxId;
SceNpMatching2SetLobbyMemberDataInternalRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Lobby member internal binary attributes
memset(lobbyMemberBinAttrInternal, 0, sizeof(lobbyMemberBinAttrInternal));
lobbyMemberBinAttrInternal[0].id =
        SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_1_ID;
lobbyMemberBinAttrInternal[0].ptr = "LOBBY_MEMBER_BIN_ATTR";
```

```
lobbyMemberBinAttrInternal[0].size = strlen("LOBBY_MEMBER_BIN_ATTR");

//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.lobbyId = lobbyId; //E Specify lobby ID of joined lobby
reqParam.memberId = 0;      //E Set information for yourself
reqParam.flagFilter = 0;    //E Unused
reqParam.flagAttr = 0;      //E Unused
reqParam.joinedSessionInfo = NULL;
reqParam.joinedSessionInfo = 0;
reqParam.lobbyMemberBinAttrInternal = lobbyMemberBinAttrInternal;
reqParam.lobbyMemberBinAttrInternalNum = 1;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2SetLobbyMemberDataInternal(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SetLobbyMemberDataInternalRequest
SCE_NP_MATCHING2_REQUEST_EVENT_SET_LOBBY_MEMBER_DATA_INTERNAL

sceNpMatching2SendLobbyChatMessage

Send lobby chat message

Definition

```
#include <np.h>

int
sceNpMatching2SendLobbyChatMessage(
        const SceNpMatching2ContextId ctxId,
        const SceNpMatching2SendLobbyChatMessageRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameter
optParam	Request option parameter
assignedReqId	Pointer to buffer for storing request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID The value specified for *lobbyId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID The value specified for *memberId* of the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_CASTTYPE	0x80550c12	Invalid message cast type The value specified for *castType* of the request parameters is invalid. Check its value.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_INVALID_MESSAGE_TARGET	0x80550c19	Invalid message target Multicast is specified for the message cast type, but members IDs to whom the message is to be sent may not have been specified.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function sends a lobby chat message.

If this function is normally terminated, the request ID is stored in the area specified by *assignedReqId*.

For the request parameter *castType*, specify any of the following message transmission types.

- Broadcast (SCE_NP_MATCHING2_CASTTYPE_BROADCAST): Send to all lobby members including the sender.
- Unicast (SCE_NP_MATCHING2_CASTTYPE_UNICAST): Send to a specific lobby member specified by a lobby member ID.
- Multicast (SCE_NP_MATCHING2_CASTTYPE_MULTICAST): Send to multiple lobby members specified by multiple lobby member IDs.

The request member *dst* is a member union that indicates the message transmission destination. Set an appropriate destination according to *castType*. When *castType* is broadcast, no destination need be set.

For the request parameter *option*, specify a message transmission option. If a message transmission option is specified, the NP ID of the message sender can be sent together with message data. If the message transmission option is specified when the message is sent, the *srcMember* member of the SceNpMatching2LobbyMessageInfo structure at the receiving side will not be NULL but will have the value in the SceNpId structure buffer.

The request execution result due to this function is reported together with the SCE_NP_MATCHING2_REQUEST_EVENT_SEND_LOBBY_CHAT_MESSAGE event to the request callback function. If the *errorCode* value that is passed to the request callback function is 0, it indicates that the request was successful. If it is a negative value, it indicates that an error occurred during request processing.

If the request due to this function is completed successfully, it indicates that the message arrived at the matching server, and the matching server finished sending the message to the specified destination. Arrival at the destination lobby member is not guaranteed.

Only UTF-8 is supported for the text data to be sent with this function.

The maximum size of a lobby chat message that can be sent at one time using this function is 1024 bytes. The macro definition representing the maximum size of a lobby chat message is SCE_NP_MATCHING2_CHAT_MSG_MAX_SIZE.

The event data *filtered* member indicates that the message data of the lobby chat message that was sent contained inappropriate words and the inappropriate parts were altered by the server's vulgarity filter before the message was sent.

Examples

The example shown below sends a lobby chat message under the following conditions.

- Send a broadcast transmission
- Do not send the sender's user information

```
int ret;
SceNpMatching2LobbyId lobbyId;

SceNpMatching2ContextId ctxId;
SceNpMatching2SendLobbyChatMessageRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameter
memset(&reqParam, 0, sizeof(reqParam));
reqParam.lobbyId = lobbyId; // Lobby ID of joined lobby
reqParam.castType = SCE_NP_MATCHING2_CASTTYPE_BROADCAST;
reqParam.msg = "LOBBY CHAT MESSAGE"; //E Message data
reqParam.msgLen = strlen("LOBBY CHAT MESSAGE");
//E Do not send sender's user information
reqParam.option = 0;

//E Assume that appropriate values are stored for ctxId and optParam

ret = sceNpMatching2SendLobbyChatMessage(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SendLobbyChatMessageRequest

SceNpMatching2SendLobbyChatMessageResponse

SCE_NP_MATCHING2_REQUEST_EVENT_SEND_LOBBY_CHAT_MESSAGE

SCE_NP_MATCHING2_SEND_MSG_OPTION_*

Signaling Request Function

sceNpMatching2SignalingGetPingInfo

Get room QoS information (Ping information)

Definition

```
#include <np.h>

int
sceNpMatching2SignalingGetPingInfo(
        SceNpMatching2ContextId ctxId,
        const SceNpMatching2SignalingGetPingInfoRequest *reqParam,
        const SceNpMatching2RequestOptParam *optParam,
        SceNpMatching2RequestId *assignedReqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqParam	Request parameters
optParam	Request option parameters
assignedReqId	Pointer to where request ID is to be stored

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized Execute sceNpMatching2Init() and initialize the NP matching 2 utility.
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument An argument required for executing the function may not have been specified. Check the argument values.
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID The value specified for *ctxId* is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID The value specified for *roomId* within the request parameters is invalid. Check its value.
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.

For a list of the NP error codes, refer to each reference document.

Description

This function obtains the QoS information (Ping information) of a room.

When this function terminates normally, the request ID will be stored in the area specified by *assignedReqId*.

This function obtains the RTT between the room owner of the room specified by the *roomId* within the request parameters and the client executing this function. Execute this function on a room included in the room list obtained by executing `sceNpMatching2SearchRoom()` and obtain the RTT with the room owner, to use as a criteria for selecting a room with a good QoS.

The execution result of this function's request will be notified to the request callback function together with the `SCE_NP_MATCHING2_REQUEST_EVENT_SIGNALING_GET_PING_INFO` event. If the *errorCode* passed to the request callback function is 0, this means the request was successful; if the value is a negative value, this means an error occurred while processing the request.

Examples

```
int ret;
SceNpMatching2RoomId roomId;

SceNpMatching2ContextId ctxId;
SceNpMatching2SignalingGetPingInfoRequest reqParam;
SceNpMatching2RequestOptParam optParam;
SceNpMatching2RequestId assignedReqId;

//E Request parameters
memset(&reqParam, 0, sizeof(reqParam));
reqParam.roomId = roomId; //E Room ID of the room to get Ping information

//E Assuming appropriate values are stored in ctxId and optParam:

ret = sceNpMatching2SignalingGetPingInfo(
    ctxId, &reqParam, &optParam, &assignedReqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

`SceNpMatching2SignalingGetPingInfoRequest`
`SceNpMatching2SignalingGetPingInfoResponse`
`SCE_NP_MATCHING2_REQUEST_EVENT_SIGNALING_GET_PING_INFO`

sceNpMatching2SignalingGetPeerNetInfo

Get network information of peer

Definition

```
#include <np.h>

int
sceNpMatching2SignalingGetPeerNetInfo(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2RoomMemberId roomMemberId,
        SceNpMatching2SignalingRequestId *reqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
roomId	Room ID
roomMemberId	Member ID of communication peer
reqId	Destination where request ID is to be stored

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_NOT_INITIALIZED	0x80550e01	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_SIGNALING_ERROR_CTX_NOT_FOUND	0x80550e05	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_SIGNALING_ERROR_INVALID_ARGUMENT	0x80550e15	Invalid argument NULL was specified for *reqId*. Specify the pointer storing the request ID.

For a list of the NP error codes, refer to each reference document.

Description

This function gets network information for the peer specified by NP ID.

When this function, which only issues a request for obtaining information, terminates normally, the request ID is returned in *reqId*. Whether or not the actual network information is successfully obtained will be reported as a signaling event (SCE_NP_MATCHING2_SIGNALING_EVENT_NETINFO_RESULT) through the callback function. When success is reported, obtain the result by specifying the request ID in sceNpMatching2SignalingGetPeerNetInfoResult().

sceNpMatching2SignalingCancelPeerNetInfo() must be executed for a request of which sceNpMatching2SignalingGetPeerNetInfoResult() does not receive the result.

Notes

This function does not time out until the request result returns. Also, note that after the request is issued, there will be no error event occurrence. Make sure to cancel requests that are no longer required using sceNpMatching2SignalingCancelPeerNetInfo().

Examples

```
SceNpMatching2SignalingRequestId reqId;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId roomMemberId;

ret = sceNpMatching2SignalingGetPeerNetInfo(ctxId, roomId, roomMemberId,
&reqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

sceNpMatching2SignalingGetPeerNetInfoResult()

sceNpMatching2SignalingCancelPeerNetInfo()

SceNpMatching2SignalingRequestId

Local Signaling Functions

sceNpMatching2SignalingGetConnectionStatus

Get connection status

Definition

```
#include <np.h>

int
sceNpMatching2SignalingGetConnectionStatus(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2RoomMemberId memberId,
        int *connStatus,
        SceNetInAddr *peerAddr,
        SceUShort16 *peerPort
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
roomId	Joined room ID
memberId	Room member ID of room member for which connection status is to be obtained
connStatus	Buffer for storing connection status that was obtained
peerAddr	Buffer for storing peer IP address that was obtained
peerPort	Buffer for storing peer port number (network byte order) that was obtained

One of the following values is stored in *connStatus*.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_CONN_STATUS_INACTIVE	0	Connection status is INACTIVE
SCE_NP_MATCHING2_SIGNALING_CONN_STATUS_PENDING	1	Connection status is PENDING
SCE_NP_MATCHING2_SIGNALING_CONN_STATUS_ACTIVE	2	Connection status is ACTIVE

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_NOT_INITIALIZED	0x80550e01	Not initialized Execute `sceNpMatching2Init()` and initialize the NP Matching 2 library.

For a list of the NP error codes, refer to each reference document.

Description

This function gets the connection status of the room member specified by *memberId*.

peerAddr and *peerPort* are valid only when SCE_NP_MATCHING2_SIGNALING_CONN_STATUS_ACTIVE is returned in *connStatus*.

Examples

```
int ret;
int connStatus;
SceNetInAddr peerAddr;
SceUShort16 peerPort;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId memberId;

ret = sceNpMatching2SignalingGetConnectionStatus(
        ctxId, roomId, memberId, &connStatus, &peerAddr, &peerPort);
if (ret < 0) {
    //E Error handling
}
```

sceNpMatching2SignalingGetConnectionInfo

Get connection information

Definition

```
#include <np.h>

int
sceNpMatching2SignalingGetConnectionInfo(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2RoomId roomId,
        SceNpMatching2RoomMemberId memberId,
        int code,
        SceNpMatching2SignalingConnectionInfo *connInfo
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
roomId	Joined room ID
memberId	Room member ID of room member for which connection information is to be obtained
code	Information code of information to be obtained
connInfo	Buffer for storing connection information that was obtained

For *code*, specify the code of the connection information to be obtained. The following connection information codes can be specified.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_RTT	1	Round trip time (microseconds)
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_BANDWIDTH	2	Estimated bandwidth (bytes/sec)
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_PEER_NP_ID	3	Peer NP ID
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_PEER_ADDRESS	4	Peer IP address and port number
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_MAPPED_ADDRESS	5	Your own IP address and port number seen from peer
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_PACKET_LOSS	6	Packet loss rate (percent)

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_NOT_INITIALIZED	0x80550e01	Not initialized Execute `sceNpMatching2Init()` and initialize the NP Matching 2 library.

For a list of the NP error codes, refer to each reference document.

Description

This function gets connection information of the room member specified by *memberId*.

Examples

```
int ret;
SceNpMatching2SignalingConnectionInfo connInfo;

//E Assume that an appropriate value is stored
SceNpMatching2ContextId ctxId;
SceNpMatching2RoomId roomId;
SceNpMatching2RoomMemberId memberId;

memset (&connInfo, 0, sizeof (connInfo));

ret = sceNpMatching2SignalingGetConnectionInfo(
        ctxId, roomId, memberId,
        SCE_NP_MATCHING2_SIGNALING_CONN_INFO_PEER_ADDRESS,
        &connInfo);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SignalingConnectionInfo

sceNpMatching2SignalingGetLocalNetInfo

Get local network information

Definition

```
#include <np.h>

int
sceNpMatching2SignalingGetLocalNetInfo (
        SceNpMatching2SignalingNetInfo *netinfo,
        );
```

Calling Conditions

Multithread safe.

Arguments

netinfo Storage destination of network information that was obtained

Return Values

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_NOT_INITIALIZED	0x80550e01	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_SIGNALING_ERROR_INVALID_ARGUMENT	0x80550e15	Invalid argument NULL was specified for *netinfo* or the structure size specified for the *size* member of *netinfo* is invalid. Check the arguments to be specified.

For a list of the NP error codes, refer to each reference document.

Description

This function gets information of the network to which the user is connected.

Examples

```
int ret;
SceNpMatching2SignalingNetInfo netinfo;
netinfo.size = sizeof(netinfo);

ret = sceNpMatching2SignalingGetLocalNetInfo(&netinfo);
if (ret < 0) {
    //E Error handling
}
```

See Also

SceNpMatching2SignalingNetInfo

sceNpMatching2SignalingGetPeerNetInfoResult

Get network information of peer

Definition

```
#include <np.h>

int
sceNpMatching2SignalingGetPeerNetInfoResult(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2SignalingRequestId reqId,
        SceNpMatching2SignalingNetInfo *netinfo
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqId	Request ID
netinfo	Storage destination of network information that was obtained

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_NOT_INITIALIZED	0x80550e01	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_SIGNALING_ERROR_CTX_NOT_FOUND	0x80550e05	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_SIGNALING_ERROR_REQ_NOT_FOUND	0x80550e07	Request could not be found The request specified in *reqId* could not be found. Check the value specified for *reqId*.
SCE_NP_MATCHING2_SIGNALING_ERROR_INVALID_ARGUMENT	0x80550e15	Invalid argument NULL was specified for *netinfo* or the structure size specified for the *size* member of *netinfo* is invalid. Check the arguments to be specified.

For a list of the NP error codes, refer to each reference document.

Description

This function gets the network information of the peer.

For *reqId*, specify the request ID that is stored in the *reqId* argument upon executing sceNpMatching2SignalingGetPeerNetInfo().

Examples

```
int ret;
SceNpMatching2SignalingNetInfo netinfo

//E Assume that an appropriate value is stored
SceNpMatching2SignalingRequestId reqId;
SceNpMatching2ContextId ctxId;

ret = sceNpMatching2SignalingGetPeerNetInfoResult(ctxId, reqId, &netinfo);
if (ret < 0) {
    //E Error handling
}
```

See Also

sceNpMatching2SignalingGetPeerNetInfo()

SceNpMatching2SignalingRequestId

sceNpMatching2SignalingCancelPeerNetInfo

Cancel peer network information acquisition request

Definition

```
#include <np.h>

int
sceNpMatching2SignalingCancelPeerNetInfo(
        SceNpMatching2ContextId ctxId,
        SceNpMatching2SignalingRequestId reqId
        );
```

Calling Conditions

Multithread safe.

Arguments

ctxId	Context ID
reqId	Request ID

Return Values

Returns 0 for normal termination.

Returns a negative value for errors. The main error codes are shown below. Note, however, that the application must not malfunction even if other error codes are returned.

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_NOT_INITIALIZED	0x80550e01	Not initialized Execute sceNpMatching2Init() and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_SIGNALING_ERROR_CTX_NOT_FOUND	0x80550e05	Context could not be found The context specified in *ctxId* could not be found. Check the value specified for *ctxId*.
SCE_NP_MATCHING2_SIGNALING_ERROR_REQ_NOT_FOUND	0x80550e07	Request could not be found The request specified in *reqId* could not be found. Check the value specified for *reqId*.

For a list of the NP error codes, refer to each reference document.

Description

This function cancels the peer network information acquisition request.

For *reqId*, specify the request ID that is stored in the *reqId* argument upon executing sceNpMatching2SignalingGetPeerNetInfo().

Examples

```
int ret;

//E Assume that an appropriate value is stored
SceNpMatching2SignalingRequestId reqId;
SceNpMatching2ContextId ctxId;

ret = sceNpMatching2SignalingCancelPeerNetInfo(ctxId, reqId);
if (ret < 0) {
    //E Error handling
}
```

See Also

sceNpMatching2SignalingGetPeerNetInfo()
SceNpMatching2SignalingRequestId

Constants

SCE_NP_MATCHING2_POOLSIZE_DEFAULT

Default value of memory pool size for the library

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_POOLSIZE_DEFAULT  (128 * 1024)
```

Description

This constant represents the default value of memory pool size for the library.

See Also

`sceNpMatching2Init()`

SCE_NP_MATCHING2_THREAD_PRIORITY_DEFAULT

Default value of priority of internal thread

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_THREAD_PRIORITY_DEFAULT
        SCE_KERNEL_DEFAULT_PRIORITY_USER
```

Description

This constant represents the default value of priority of internal thread.

See Also

`sceNpMatching2Init()`

SCE_NP_MATCHING2_THREAD_STACK_SIZE_DEFAULT

Default value of stack size of internal thread

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_THREAD_STACK_SIZE_DEFAULT        (16 * 1024)
```

Description

This constant represents the default value of stack size of internal thread

See Also

`sceNpMatching2Init()`

SCE_NP_MATCHING2_RANGE_FILTER_START_INDEX_MIN

Minimum value of the range filter

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_RANGE_FILTER_START_INDEX_MIN    1
```

Description

This constant represents the minimum value of the range filter, which indicates the start of the information to be obtained.

See Also

SceNpMatching2RangeFilter

SCE_NP_MATCHING2_RANGE_FILTER_MAX

Maximum value of the range filter

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_RANGE_FILTER_MAX  20
```

Description

This constant represents the maximum value of the range filter, which indicates the maximum number of data items to obtain.

See Also

SceNpMatching2RangeFilter

SCE_NP_MATCHING2_LOBBY_MAX_SLOT

Maximum number of members that can join a lobby

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_LOBBY_MAX_SLOT    256
```

Description

This function represents the maximum number of members that can join a lobby.

SCE_NP_MATCHING2_ROOM_MAX_SLOT

Maximum number of members in a room

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_MAX_SLOT      64
```

Description

This constant represents the maximum number of members allowed in a room.

See Also

SceNpMatching2CreateJoinRoomRequest

SCE_NP_MATCHING2_ROOM_GROUP_ID_MAX

Maximum value of room group ID

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_GROUP_ID_MAX 15
```

Description

This constant represents the maximum value of the room group ID.

See Also

SceNpMatching2RoomGroupId

SCE_NP_MATCHING2_ROOM_ALLOWED_USER_MAX

Maximum number of users allowed to join a room

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_ALLOWED_USER_MAX      (100)
```

Description

This constant represents the maximum number of users that can be registered to join a room without a password.

See Also

SceNpMatching2CreateJoinRoomRequest

SCE_NP_MATCHING2_ROOM_BLOCKED_USER_MAX

Maximum number of users blocked from joining room

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_BLOCKED_USER_MAX      (100)
```

Description

This constant represents the maximum number of users that can be registered to be blocked from joining a room.

See Also

SceNpMatching2CreateJoinRoomRequest
SceNpMatching2JoinRoomRequest

SCE_NP_MATCHING2_CHAT_MSG_MAX_SIZE

Maximum size of a chat message

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_CHAT_MSG_MAX_SIZE  1024
```

Description

This constant represents the maximum size of a chat message.

See Also

SceNpMatching2SendRoomChatMessageRequest
SceNpMatching2SendLobbyChatMessageRequest

SCE_NP_MATCHING2_BIN_MSG_MAX_SIZE

Maximum size of a binary message

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_BIN_MSG_MAX_SIZE   1024
```

Description

This constant represents the maximum size of a binary message.

See Also

SceNpMatching2SendRoomMessageRequest

SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_LIST_MAX

Maximum number of lobby-internal lobby member information (without extension data)

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_LIST_MAX    256
```

Description

This constant represents the maximum number of lobby-internal lobby member information (without extension data) to obtain in a list.

See Also

SceNpMatching2GetLobbyMemberDataInternalListRequest

SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_EXTENDED_DATA_LIST_MAX

Maximum number of lobby-internal lobby member information (with extension data)

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_LOBBY_MEMBER_DATA_INTERNAL_EXTENDED_DATA_LIST_MAX 64
```

Description

This constant represents the maximum number of lobby-internal lobby member information (with extension data) to obtain in a list.

See Also

SceNpMatching2GetLobbyMemberDataInternalListRequest

SCE_NP_MATCHING2_GET_USER_INFO_LIST_NPID_NUM_MAX

Maximum number of users for which information can be obtained in the user information list

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_GET_USER_INFO_LIST_NPID_NUM_MAX 25
```

Description

This constant represents the maximum number of NP IDs that can be specified upon obtaining the user information list using `sceNpMatching2GetUserInfoList()`.

See Also

`SceNpMatching2GetUserInfoListRequest`

Constants (Parameters)

SCE_NP_MATCHING2_OPERATOR_*

Comparison operator specified as the search condition

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_OPERATOR_EQ	1	Equal to (==)
SCE_NP_MATCHING2_OPERATOR_NE	2	Not equal to (!=)
SCE_NP_MATCHING2_OPERATOR_LT	3	Less than (<)
SCE_NP_MATCHING2_OPERATOR_LE	4	Less than or equal to (<=)
SCE_NP_MATCHING2_OPERATOR_GT	5	Greater than (>)
SCE_NP_MATCHING2_OPERATOR_GE	6	Greater than or equal to (>=)

Description

These constants represent the comparison operator specified in the search conditions when searching for a room.

See Also

SceNpMatching2Operator
SceNpMatching2IntSearchFilter
SceNpMatching2BinSearchFilter

SCE_NP_MATCHING2_CASTTYPE_*

Message cast type

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_CASTTYPE_BROADCAST	1	Broadcast
SCE_NP_MATCHING2_CASTTYPE_UNICAST	2	Unicast
SCE_NP_MATCHING2_CASTTYPE_MULTICAST	3	Multicast
SCE_NP_MATCHING2_CASTTYPE_MULTICAST_TEAM	4	Multicast to users on a team with the specified team ID

Description

These constants represent the cast type when sending a message.

See Also

SceNpMatching2CastType
SceNpMatching2SendRoomChatMessageRequest
SceNpMatching2SendRoomMessageRequest
sceNpMatching2SendRoomChatMessage()
sceNpMatching2SendRoomMessage()

SCE_NP_MATCHING2_SESSION_TYPE_*

Session type

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SESSION_TYPE_LOBBY	1	Lobby
SCE_NP_MATCHING2_SESSION_TYPE_ROOM	2	Room

Description

These constants represent the session type.

See Also

SceNpMatching2SessionType

SCE_NP_MATCHING2_SIGNALING_TYPE_*

Signaling type

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_TYPE_NONE	0	No specification
SCE_NP_MATCHING2_SIGNALING_TYPE_MESH	1	Full mesh
SCE_NP_MATCHING2_SIGNALING_TYPE_STAR	2	Star

Description

This constant represents the session type (P2P connection topology).

See Also

SceNpMatching2SignalingType

SCE_NP_MATCHING2_SIGNALING_FLAG_*

Signaling flag

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_FLAG_MANUAL	0x01	Do not automatically establish a signaling connection

Description

This constant represents the signaling operation mode, etc.

See Also

SceNpMatching2SignalingFlag

SCE_NP_MATCHING2_EVENT_CAUSE_*

Event cause

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_EVENT_CAUSE_LEAVE_ACTION	1	A member explicitly left the room
SCE_NP_MATCHING2_EVENT_CAUSE_KICKOUT_ACTION	2	A member explicitly kicked another member out
SCE_NP_MATCHING2_EVENT_CAUSE_GRANT_OWNER_ACTION	3	A member explicitly transferred room ownership
SCE_NP_MATCHING2_EVENT_CAUSE_SERVER_OPERATION	4	Event was caused by server operation
SCE_NP_MATCHING2_EVENT_CAUSE_MEMBER_DISAPPEARED	5	A member disappeared
SCE_NP_MATCHING2_EVENT_CAUSE_SERVER_INTERNAL	6	Event was caused by an internal server error
SCE_NP_MATCHING2_EVENT_CAUSE_CONNECTION_ERROR	7	Connection was severed
SCE_NP_MATCHING2_EVENT_CAUSE_NP_SIGNED_OUT	8	NP signed out
SCE_NP_MATCHING2_EVENT_CAUSE_SYSTEM_ERROR	9	System error occurred
SCE_NP_MATCHING2_EVENT_CAUSE_CONTEXT_ERROR	10	Context error occurred
SCE_NP_MATCHING2_EVENT_CAUSE_CONTEXT_ACTION	11	Event was caused by a context operation

Description

These constants represent the cause of an event.

When a room event, lobby event, or context event occurs, the cause of the event is indicated with one of these constants.

See Also

SceNpMatching2EventCause
SceNpMatching2RoomMemberUpdateInfo
SceNpMatching2RoomOwnerUpdateInfo
SceNpMatching2RoomUpdateInfo
SceNpMatching2ContextCallback

SCE_NP_MATCHING2_SERVER_STATUS_*

Server status

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SERVER_STATUS_AVAILABLE	1	Usable
SCE_NP_MATCHING2_SERVER_STATUS_UNAVAILABLE	2	Not usable
SCE_NP_MATCHING2_SERVER_STATUS_BUSY	3	Busy
SCE_NP_MATCHING2_SERVER_STATUS_MAINTENANCE	4	Under maintenance

Description

These constants represent the server status.

See Also

SceNpMatching2ServerStatus

SCE_NP_MATCHING2_ROLE_*

Session member role

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROLE_MEMBER	1	Regular session member
SCE_NP_MATCHING2_ROLE_OWNER	2	Session owner

Description

These constants represent the role of a member in a session.

See Also

SceNpMatching2Role

SCE_NP_MATCHING2_BLOCKKICKFLAG_*

Status of kicked-out member with regards to rejoining

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_BLOCKKICKFLAG_OK	0	Allow member to rejoin
SCE_NP_MATCHING2_BLOCKKICKFLAG_NG	1	Forbid member from rejoining

Description

These constants represent the setting that allows or forbids a kicked-out member rejoining the session.

See Also

SceNpMatching2BlockKickFlag

SCE_NP_MATCHING2_SORT_METHOD_*

Sort method

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SORT_METHOD_JOIN_DATE	0	In order of join date/time
SCE_NP_MATCHING2_SORT_METHOD_SLOT_NUMBER	1	In order of slot number

Description

These constants specify the sort order of elements, when a list (like the list of member s) is obtained.

See Also

sceNpMatching2GetRoomMemberIdListLocal()

Constants (Optional)

SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_*

Room search options

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_WITH_NPID	0x01	Get the NP ID of the room owner
SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_NAT_TYPE_FILTER	0x08	Rooms for which P2P connections cannot be established appropriately with room members after joining the room are excluded from the room search results according to the NAT type of the user who executed the room search
SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_RANDOM	0x10	Obtain list of randomly selected rooms from those matching the search condition

Description

These constants represent the room search options, which are specified when searching for a room (when `sceNpMatching2SearchRoom()` is executed).

See Also

`sceNpMatching2SearchRoom()`

SCE_NP_MATCHING2_SEND_MSG_OPTION_*

Send options

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SEND_MSG_OPTION_WITH_NPID	0x01	Send the NP ID

Description

These constants represent the send options, which are specified when sending a message.

See Also

sceNpMatching2SendRoomChatMessage()
sceNpMatching2SendRoomMessage()

Constants (Attributes and Attribute IDs)

SCE_NP_MATCHING2_LOBBY_FLAG_ATTR_*

Flag-type lobby attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_LOBBY_FLAG_ATTR_MEMBER_NOTIFICATION	0x20000000	Room notification flag. Indicates whether or not to notify lobby members upon joining or leaving a lobby.

Description

These constants represent the flag-type attributes of a lobby.

See Also

SceNpMatching2FlagAttr

SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID

Attribute ID of lobby member internal binary attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_1_ID	0x0039	Lobby member internal binary attribute 1

Description

This constant represents the attribute ID of a lobby member's binary-type internal reference attribute.

See Also

SceNpMatching2BinAttr
SceNpMatching2AttributeId

SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_*_ID

Attribute ID of internal lobby binary attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_1_ID	0x0037	Internal lobby binary attribute 1
SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_2_ID	0x0038	Internal lobby binary attribute 2

Description

These constants represent the attribute ID of a lobby's internal binary-type reference attribute.

See Also

SceNpMatching2BinAttr
SceNpMatching2AttributeId

SCE_NP_MATCHING2_ROOM_FLAG_ATTR_*

Flag-type room attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOM_FLAG_ATTR_OWNER_AUTO_GRANT	0x80000000	Automatic ownership grant flag. Indicates whether or not the room server automatically grants room ownership to a member when the room owner leaves the room.
SCE_NP_MATCHING2_ROOM_FLAG_ATTR_CLOSED	0x40000000	CLOSED flag. Indicates the status of the room with regards to joining. If the room is set to CLOSED, users will be unable to join.
SCE_NP_MATCHING2_ROOM_FLAG_ATTR_FULL	0x20000000	FULL flag. Indicates whether or not the room is full. When the room becomes full, the FULL flag will be set automatically, and users will be unable to join.
SCE_NP_MATCHING2_ROOM_FLAG_ATTR_HIDDEN	0x10000000	HIDDEN flag. Indicates whether or not the room is to be included in room searches. If the room is hidden, it will not turn up in the results even if it matches the search conditions.
SCE_NP_MATCHING2_ROOM_FLAG_ATTR_NAT_TYPE_RESTRICTION	0x04000000	NAT type room entry limitation flag. If this flag is enabled, a user will no longer be able to join a room when that user's NAT type cannot establish a P2P connection between room members according to the topology specified by the signaling option parameter.
SCE_NP_MATCHING2_ROOM_FLAG_ATTR_PROHIBITIVE_MODE	0x02000000	Prohibitive mode flag. If this flag is enabled, a user joining the room can add NP IDs to the room's block list when joining the room.

Description

These constants represent the flag-type attributes of a room.

See Also

SceNpMatching2FlagAttr

SCE_NP_MATCHING2_ROOMMEMBER_FLAG_ATTR_*

Flag-type room member attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOMMEMBER_FLAG_ATTR_OWNER	0x80000000	Room owner flag. Indicates whether or not the room member is the room owner.

Description

These constants represent the flag-type attributes of a room member.

See Also

SceNpMatching2FlagAttr

SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID

ID of external room search integer attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_1_ID	0x004c	External room search integer attribute 1
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_2_ID	0x004d	External room search integer attribute 2
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_3_ID	0x004e	External room search integer attribute 3
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_4_ID	0x004f	External room search integer attribute 4
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_5_ID	0x0050	External room search integer attribute 5
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_6_ID	0x0051	External room search integer attribute 6
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_7_ID	0x0052	External room search integer attribute 7
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_8_ID	0x0053	External room search integer attribute 8

Description

These constants represent the IDs of a room's external integer-type attributes, which can be specified as search conditions.

See Also

SceNpMatching2IntAttr

SceNpMatching2AttributeId

SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID

ID of external room search binary attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_1_ID	0x0054	External room search binary attribute 1

Description

This constant represents the ID of a room's external binary-type attribute, which can be specified as a search condition.

See Also

SceNpMatching2BinAttr
SceNpMatching2AttributeId

SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID

ID of external room binary attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_1_ID	0x0055	External room binary attribute 1
SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_2_ID	0x0056	External room binary attribute 2

Description

These constants represent the IDs of a room's external binary-type attributes.

See Also

SceNpMatching2BinAttr
SceNpMatching2AttributeId

SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID

ID of internal room binary attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_1_ID	0x0057	Internal room binary attribute 1
SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_2_ID	0x0058	Internal room binary attribute 2

Description

These constants represent the IDs of a room's internal binary-type attributes.

See Also

SceNpMatching2BinAttr
SceNpMatching2AttributeId

SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID

ID of internal room member binary attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_1_ID	0x0059	Internal room member binary attribute 1

Description

This constant represents the ID of a room member's internal binary-type attribute.

See Also

SceNpMatching2BinAttr
SceNpMatching2AttributeId

SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID

Attribute ID of user binary attribute

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_USER_BIN_ATTR_1_ID	0x005f	User binary attribute 1

Description

This constant represents the attribute ID of a user's binary-type attribute.

See Also

SceNpMatching2BinAttr
SceNpMatching2AttributeId

Constants (Attribute Values and Attribute Sizes)

SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_NUM

Number of internal lobby binary attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_NUM  2
```

Description

This constant represents the number of internal lobby binary attributes.

See Also

SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_MAX_SIZE

Maximum size of an internal lobby binary attribute

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_MAX_SIZE  256
```

Description

This constant represents the maximum size of an internal lobby binary attribute.

See Also

SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_NUM

Number of internal lobby member binary attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_NUM  1
```

Description

This constant represents the number of internal lobby member binary attributes.

See Also

SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_MAX_SIZE

Maximum size of an internal lobby member binary attribute

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_MAX_SIZE  64
```

Description

This constant represents the maximum size of an internal lobby member binary attribute.

See Also

SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_NUM

Number of external room search integer attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_NUM  8
```

Description

This constant represents the number of external room search integer attributes.

See Also

SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID

SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_NUM

Number of external room search binary attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_NUM  1
```

Description

This constant represents the number of external room search binary attributes.

See Also

SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID

SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_MAX_SIZE

Maximum size of an external room search binary attribute

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_MAX_SIZE  64
```

Description

This constant represents the maximum size of an external room search binary attribute.

See Also

SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID

SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_NUM

Number of external room binary attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_NUM  2
```

Description

This constant represents the number of external room binary attributes.

See Also

SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID

SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_MAX_SIZE

Maximum size of an external room binary attribute

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_MAX_SIZE  256
```

Description

This constant represents the maximum size of an external room binary attribute.

See Also

SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID

SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_NUM

Number of internal room binary attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_NUM  2
```

Description

This constant represents the number of internal room binary attributes.

See Also

SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_MAX_SIZE

Maximum size of an internal room binary attribute

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_MAX_SIZE  256
```

Description

This constant represents the maximum size of an internal room binary attribute.

See Also

SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_NUM

Number of internal room member binary attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_NUM  1
```

Description

This constant represents the number of internal room member binary attributes.

See Also

SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_MAX_SIZE

Maximum size of an internal room member binary attribute

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_MAX_SIZE  64
```

Description

This constant represents the maximum size of an internal room member binary attribute.

See Also

SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID

SCE_NP_MATCHING2_USER_BIN_ATTR_NUM

Number of user binary attributes

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_USER_BIN_ATTR_NUM  1
```

Description

This constant represents the number of user binary attributes.

See Also

SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID

SCE_NP_MATCHING2_USER_BIN_ATTR_MAX_SIZE

Maximum size of a user binary attribute

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_USER_BIN_ATTR_MAX_SIZE  128
```

Description

This constant represents the maximum size of a user binary attribute.

See Also

SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID

Constants (Signaling)

SCE_NP_MATCHING2_SIGNALING_CONN_STATUS *

Current connection status

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_CONN_STATUS_INACTIVE	0	Available
SCE_NP_MATCHING2_SIGNALING_CONN_STATUS_PENDING	1	Available
SCE_NP_MATCHING2_SIGNALING_CONN_STATUS_ACTIVE	2	Unavailable

Description

These constants represent the connection status.

See Also

sceNpMatching2SignalingGetConnectionStatus()

SCE_NP_MATCHING2_SIGNALING_CONN_INFO_*

Connection information to be obtained

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_RTT	1	Round Trip Time (micro seconds)
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_BANDWIDTH	2	Bandwidth (bytes/sec)
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_PEER_NP_ID	3	Peer's NP ID
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_PEER_ADDRESS	4	Peer's IP address and port number
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_MAPPED_ADDRESS	5	Your own IP address and port number seen from peer
SCE_NP_MATCHING2_SIGNALING_CONN_INFO_PACKET_LOSS	6	Packet loss rate (percent)

Description

These constants represent the connection information to be obtained.

See Also

sceNpMatching2SignalingGetConnectionInfo()

SCE_NP_MATCHING2_SIGNALING_NETINFO_NAT_STATUS_*

NAT status type

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_NETINFO_NAT_STATUS_UNKNOWN	0	Unknown
SCE_NP_MATCHING2_SIGNALING_NETINFO_NAT_STATUS_TYPE1	1	Type 1
SCE_NP_MATCHING2_SIGNALING_NETINFO_NAT_STATUS_TYPE2	2	Type 2
SCE_NP_MATCHING2_SIGNALING_NETINFO_NAT_STATUS_TYPE3	3	Type 3

Description

These constants represent NAT status type in the network information.

See Also

SceNpMatching2SignalingNetInfo

Constants (Events)

SCE_NP_MATCHING2_REQUEST_EVENT_*

Event of request functions

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_REQUEST_EVENT_GET_WORLD_INFO_LIST	0x0002	sceNpMatching2GetWorldInfoList() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_MEMBER_DATA_EXTERNAL_LIST	0x0003	sceNpMatching2GetRoomMemberDataExternalList() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_DATA_EXTERNAL	0x0004	sceNpMatching2SetRoomDataExternal() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_DATA_EXTERNAL_LIST	0x0005	sceNpMatching2GetRoomDataExternalList() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_INFO_LIST	0x0006	sceNpMatching2GetLobbyInfoList() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SET_USER_INFO	0x0007	sceNpMatching2SetUserInfo() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_USER_INFO_LIST	0x0008	sceNpMatching2GetUserInfoList() completed
SCE_NP_MATCHING2_REQUEST_EVENT_CREATE_JOIN_ROOM	0x0101	sceNpMatching2CreateJoinRoom() completed
SCE_NP_MATCHING2_REQUEST_EVENT_JOIN_ROOM	0x0102	sceNpMatching2JoinRoom() completed
SCE_NP_MATCHING2_REQUEST_EVENT_LEAVE_ROOM	0x0103	sceNpMatching2LeaveRoom() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GRANT_ROOM_OWNER	0x0104	sceNpMatching2GrantRoomOwner() completed
SCE_NP_MATCHING2_REQUEST_EVENT_KICKOUT_ROOM_MEMBER	0x0105	sceNpMatching2KickoutRoomMember() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SEARCH_ROOM	0x0106	sceNpMatching2SearchRoom() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SEND_ROOM_CHAT_MESSAGE	0x0107	sceNpMatching2SendRoomChatMessage() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SEND_ROOM_MESSAGE	0x0108	sceNpMatching2SendRoomMessage() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_DATA_INTERNAL	0x0109	sceNpMatching2SetRoomDataInternal() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_DATA_INTERNAL	0x010a	sceNpMatching2GetRoomDataInternal() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SET_ROOM_MEMBER_DATA_INTERNAL	0x010b	sceNpMatching2SetRoomMemberDataInternal() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_ROOM_MEMBER_DATA_INTERNAL	0x010c	sceNpMatching2GetRoomMemberDataInternal() completed

Value	(Number)	Description
SCE_NP_MATCHING2_REQUEST_EVENT_SET_SIGNALING_OPT_PARAM	0x010d	sceNpMatching2SetSignalingOptParam() completed
SCE_NP_MATCHING2_REQUEST_EVENT_JOIN_LOBBY	0x0201	sceNpMatching2JoinLobby() completed
SCE_NP_MATCHING2_REQUEST_EVENT_LEAVE_LOBBY	0x0202	sceNpMatching2LeaveLobby() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SEND_LOBBY_CHAT_MESSAGE	0x0203	sceNpMatching2SendLobbyChatMessage() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SET_LOBBY_MEMBER_DATA_INTERNAL	0x0205	sceNpMatching2SetLobbyMemberDataInternal() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_MEMBER_DATA_INTERNAL	0x0206	sceNpMatching2GetLobbyMemberDataInternal() completed
SCE_NP_MATCHING2_REQUEST_EVENT_GET_LOBBY_MEMBER_DATA_INTERNAL_LIST	0x0207	sceNpMatching2GetLobbyMemberDataInternalList() completed
SCE_NP_MATCHING2_REQUEST_EVENT_SIGNALING_GET_PING_INFO	0x0e01	sceNpMatching2SignalingGetPingInfo() completed

Description

These constants represent request events, which correspond to specific request functions. Request events are notified to the request callback.

See Also

SceNpMatching2RequestCallback

SCE_NP_MATCHING2_ROOM_EVENT_*

Room event

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOM_EVENT_MEMBER_JOINED	0x1101	A new room member joined
SCE_NP_MATCHING2_ROOM_EVENT_MEMBER_LEFT	0x1102	A room member left
SCE_NP_MATCHING2_ROOM_EVENT_KICKEDOUT	0x1103	Kicked out of room
SCE_NP_MATCHING2_ROOM_EVENT_ROOM_DESTROYED	0x1104	Room was deleted
SCE_NP_MATCHING2_ROOM_EVENT_ROOM_OWNER_CHANGED	0x1105	Room owner changed
SCE_NP_MATCHING2_ROOM_EVENT_UPDATED_ROOM_DATA_INTERNALL	0x1106	Internal attributes and basic data of room were updated (Data in SceNpMatching2RoomDataInternal structure was updated)
SCE_NP_MATCHING2_ROOM_EVENT_UPDATED_ROOM_MEMBER_DATA_INTERNAL	0x1107	Internal attributes and basic data of room member were updated (Data in SceNpMatching2RoomMemberDataInternal structure was updated)
SCE_NP_MATCHING2_ROOM_EVENT_UPDATED_SIGNALING_OPT_PARAM	0x1108	Signaling option parameter was updated

Description

These constants represent room events. Room events are notified to the room event callback.

See Also

SceNpMatching2RoomEventCallback

SCE_NP_MATCHING2_ROOM_MSG_EVENT_*

Room message event

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_ROOM_MSG_EVENT_CHAT_MESSAGE	0x2101	Received a room chat message
SCE_NP_MATCHING2_ROOM_MSG_EVENT_MESSAGE	0x2102	Received a room message

Description

These constants represent room message events. Room message events are notified to the room message callback.

See Also

SceNpMatching2RoomMessageCallback

SCE_NP_MATCHING2_LOBBY_EVENT_*

Lobby event

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_LOBBY_EVENT_MEMBER_JOINED	0x3201	A new lobby member joined
SCE_NP_MATCHING2_LOBBY_EVENT_MEMBER_LEFT	0x3202	A lobby member left
SCE_NP_MATCHING2_LOBBY_EVENT_LOBBY_DESTROYED	0x3203	Lobby was destroyed
SCE_NP_MATCHING2_LOBBY_EVENT_UPDATED_LOBBY_MEMBER_DATA_INTERNAL	0x3204	Lobby member's internal attribute or basic information was updated (data included in SceNpMatching2LobbyMemberDataInternal structure was updated)

Description

This constant represents a lobby event. A lobby event is reported to the lobby event callback function.

See Also

SceNpMatching2LobbyEventCallback

SCE_NP_MATCHING2_LOBBY_MSG_EVENT_*

Lobby message event

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_LOBBY_MSG_EVENT_CHAT_MESSAGE	0x4201	Lobby chat message was received

Description

This constant represents a lobby message event. A lobby message event is reported to the lobby message callback function.

See Also

SceNpMatching2LobbyMessageCallback

SCE_NP_MATCHING2_SIGNALING_EVENT_*

Signaling event

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_EVENT_DEAD	0x5101	P2P connection was disconnected
SCE_NP_MATCHING2_SIGNALING_EVENT_ESTABLISHED	0x5102	P2P connection was established
SCE_NP_MATCHING2_SIGNALING_EVENT_NETINFO_RESULT	0x5103	It becomes possible to obtain the result of a request for obtaining network information through sceNpMatching2SignalingGetPeerNetInfo()

Description

This constant represents a signaling event. A signaling event is reported to the signaling callback function.

See Also

SceNpMatching2SignalingCallback

SCE_NP_MATCHING2_CONTEXT_EVENT_*

Context event

Definition

```
#include <np.h>
```

Value	(Number)	Description
SCE_NP_MATCHING2_CONTEXT_EVENT_START_OVER	0x6f01	Entered a state in which context cannot continue to be used
SCE_NP_MATCHING2_CONTEXT_EVENT_STARTED	0x6f02	Processing to start context completed
SCE_NP_MATCHING2_CONTEXT_EVENT_STOPPED	0x6f03	Processing to stop context completed

Description

This constant represents a context event. A context event is reported to the context callback function.

See Also

SceNpMatching2ContextCallback

Constants (Event Data Sizes)

SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_*

Maximum sizes of event data

Definition

```
#include <np.h>
```

Request Events

Value	Constant
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_WORLD_INFO_LIST	3848
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_ROOM_MEMBER_DATA_EXTERNAL_LIST	15624
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_ROOM_DATA_EXTERNAL_LIST	25768
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_LOBBY_INFO_LIST	1296
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_USER_INFO_LIST	17604
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_CREATE_JOIN_ROOM	25224
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_JOIN_ROOM	25224
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_SEARCH_ROOM	25776
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_SEND_ROOM_CHAT_MESSAGE	1
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_ROOM_DATA_INTERNAL	25224
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_ROOM_MEMBER_DATA_INTERNAL	372
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_JOIN_LOBBY	1124
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_SEND_LOBBY_CHAT_MESSAGE	1
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_LOBBY_MEMBER_DATA_INTERNAL	672
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_GET_LOBBY_MEMBER_DATA_INTERNAL_LIST	42760

Room Events

Value	Constant
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_ROOM_MEMBER_UPDATE_INFO	396
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_ROOM_UPDATE_INFO	28
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_ROOM_OWNER_UPDATE_INFO	40
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_ROOM_DATA_INTERNAL_UPDATE_INFO	25404
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_ROOM_MEMBER_DATA_INTERNAL_UPDATE_INFO	493
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_SIGNALING_OPT_PARAM_UPDATE_INFO	8

Room Message Events

Value	Constant
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_ROOM_MESSAGE_INFO	1407

Lobby Events

Value	Constant
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_LOBBY_MEMBER_UPDATE_INFO	696
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_LOBBY_UPDATE_INFO	8
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_LOBBY_MEMBER_DATA_INTERNAL_UPDATE_INFO	472

Lobby Message Events

Value	Constant
SCE_NP_MATCHING2_EVENT_DATA_MAX_SIZE_LOBBY_MESSAGE_INFO	1790

Description

These constants represent the maximum values of event data sizes.

Macros

SCE_NP_MATCHING2_GET_WORLD_NUMBER

Get the world number

Definition

```
#include <np.h>
#define SCE_NP_MATCHING2_GET_WORLD_NUMBER(worldId) \
        ( (SceNpMatching2WorldNumber)((worldId) & 0x0000ffffU) )
```

Arguments

worldId World ID

Description

This macro gets the world number from a world ID.

Examples

```
SceNpMatching2WorldId worldId;
SceNpMatching2WorldNumber worldNumber;

worldNumber = SCE_NP_MATCHING2_GET_WORLD_NUMBER(worldId);
```

See Also

SceNpMatching2WorldId

SceNpMatching2WorldNumber

SCE_NP_MATCHING2_GET_LOBBY_NUMBER

Get the lobby number

Definition

```
#include <np.h>
#define SCE_NP_MATCHING2_GET_LOBBY_NUMBER(lobbyId) \
     ( (SceNpMatching2LobbyNumber)(((lobbyId) & 0x00000000ffff0000ULL)>>16) )
```

Arguments

lobbyId	Lobby ID

Description

This macro gets the lobby number from a lobby ID.

Examples

```
SceNpMatching2LobbyId lobbyId;
SceNpMatching2LobbyNumber lobbyNumber;

lobbyNumber = SCE_NP_MATCHING2_GET_LOBBY_NUMBER(lobbyId);
```

See Also

SceNpMatching2LobbyId
SceNpMatching2LobbyNumber

SCE_NP_MATCHING2_GET_ROOM_NUMBER

Get the room number

Definition

```
#include <np.h>
#define SCE_NP_MATCHING2_GET_ROOM_NUMBER(roomId) \
        ( (SceNpMatching2RoomNumber)((roomId) & 0x000000000000ffffULL) )
```

Arguments

roomId Room ID

Description

This macro gets the room number from a room ID.

Examples

```
SceNpMatching2RoomId roomId;
SceNpMatching2RoomNumber roomNumber;

roomNumber = SCE_NP_MATCHING2_GET_ROOM_NUMBER(roomId);
```

See Also

SceNpMatching2RoomId

SceNpMatching2RoomNumber

SCE_NP_MATCHING2_ADD_SLOTNUM_TO_ROOM_PASSWORD_SLOT_MASK

Add slot numbers to room password slot mask

Definition

```
#include <np.h>
#define SCE_NP_MATCHING2_ADD_SLOTNUM_TO_ROOM_PASSWORD_SLOT_MASK(mask,
slotNumber) \
        ( (mask) |= (SceUInt64)(0x1ULL<<(64-(slotNumber))) )
```

Arguments

mask	Room password slot mask
slotNumber	Slot number

Description

This macro sets the bits of *mask* corresponding to the slot numbers specified in *slotNumber* and generates a room password slot mask. This macro is used to enable room passwords for specific slots when creating a room without setting groups in the room.

Examples

This example enables room passwords for slot numbers 3, 4.

```
SceNpMatching2RoomPasswordSlotMask mask;

memset(&mask, 0, sizeof(mask));
SCE_NP_MATCHING2_ADD_SLOTNUM_TO_ROOM_PASSWORD_SLOT_MASK(mask, 3);
SCE_NP_MATCHING2_ADD_SLOTNUM_TO_ROOM_PASSWORD_SLOT_MASK(mask, 4);
```

See Also

`sceNpMatching2CreateJoinRoom()`

`SceNpMatching2RoomPasswordSlotMask`

Typedefs

SceNpMatching2ServerId

Server ID

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2ServerId;
```

Description

This type represents a server ID.

SceNpMatching2WorldId

World ID

Definition

```
#include <np.h>

typedef SceUInt32 SceNpMatching2WorldId;
```

Description

This type represents a world ID.

SceNpMatching2WorldNumber

World number

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2WorldNumber;
```

Description

This type represents a world number.

SceNpMatching2LobbyId

Lobby ID

Definition

```
#include <np.h>

typedef SceUInt64 SceNpMatching2LobbyId;
```

Description

This type represents a lobby ID.

SceNpMatching2LobbyNumber

Lobby number

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2LobbyNumber;
```

Description

This type represents a lobby number.

SceNpMatching2LobbyMemberId

Lobby member ID

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2LobbyMemberId;
```

Description

This type represents a lobby member ID.

SceNpMatching2RoomId

Room ID

Definition

```
#include <np.h>

typedef SceUInt64 SceNpMatching2RoomId;
```

Description

This type represents a room ID.

SceNpMatching2RoomNumber

Room number

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2RoomNumber;
```

Description

This type represents a room number.

SceNpMatching2RoomMemberId

Room member ID

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2RoomMemberId;
```

Description

This type represents a room member ID.

SceNpMatching2RoomGroupId

Group ID

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2RoomGroupId;
```

Description

This type represents a group ID.

SceNpMatching2TeamId

Team ID

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2TeamId;
```

Description

This type represents a team ID.

SceNpMatching2ContextId

Context ID

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2ContextId;
```

Description

This type represents a context ID.

SceNpMatching2RequestId

Request ID

Definition

```
#include <np.h>

typedef SceUInt32 SceNpMatching2RequestId;
```

Description

This type represents a request ID, which is allocated when a request function is executed.

SceNpMatching2SignalingRequestId

Signaling request ID

Definition

```
#include <np.h>

typedef SceUInt32 SceNpMatching2SignalingRequestId;
```

Description

This type represents the request ID of an asynchronous signaling API.

See Also

`sceNpMatching2SignalingGetPeerNetInfo()`

SceNpMatching2AttributeId

Attribute ID

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2AttributeId;
```

Description

This type represents an attribute ID.

See Also

```
SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID
SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID
SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID
SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID
SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID
```

SceNpMatching2FlagAttr

Flag-type attribute

Definition

```
#include <np.h>

typedef SceUInt32 SceNpMatching2FlagAttr;
```

Description

This type represents a flag-type attribute.

See Also

```
SCE_NP_MATCHING2_LOBBY_FLAG_ATTR_*
SCE_NP_MATCHING2_ROOM_FLAG_ATTR_*
SCE_NP_MATCHING2_ROOMMEMBER_FLAG_ATTR_*
```

SceNpMatching2NatType

NAT type

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2NatType;
```

Description

This type represents a NAT type.

SceNpMatching2Operator

Comparison operator

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2Operator;
```

Description

This type represents a comparison operator, which is specified when searching for a room.

See Also

SCE_NP_MATCHING2_OPERATOR_*

SceNpMatching2CastType

Message cast type

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2CastType;
```

Description

This type represents a cast type, which is specified when sending messages.

See Also

SCE_NP_MATCHING2_CASTTYPE_*

SceNpMatching2SessionType

Session type

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2SessionType;
```

Description

This type represents a session type.

See Also

SCE_NP_MATCHING2_SESSION_TYPE_*

SceNpMatching2SignalingType

Signaling type

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2SignalingType;
```

Description

This type represents the signaling type.

See Also

SCE_NP_MATCHING2_SIGNALING_TYPE_*

SceNpMatching2SignalingFlag

Signaling flag

Definition

```
#include <np.h>

typedef uint8_t SceNpMatching2SignalingFlag;
```

Description

This type represents the signaling flag.

See Also

SCE_NP_MATCHING2_SIGNALING_FLAG_*

SceNpMatching2EventCause

Event cause

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2EventCause;
```

Description

This type represents an event cause.

See Also

SCE_NP_MATCHING2_EVENT_CAUSE_*

SceNpMatching2ServerStatus

Server status

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2ServerStatus;
```

Description

This type represents a server status.

See Also

SCE_NP_MATCHING2_SERVER_STATUS_*

SceNpMatching2Role

Session member role

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2Role;
```

Description

This type represents a session member role.

See Also

SCE_NP_MATCHING2_ROLE_*

SceNpMatching2BlockKickFlag

Setting for member after being kicked out

Definition

```
#include <np.h>

typedef SceUChar8 SceNpMatching2BlockKickFlag;
```

Description

This type represents a setting for a member who has been kicked out regarding rejoining privileges.

See Also

SCE_NP_MATCHING2_BLOCKKICKFLAG_*

SceNpMatching2RoomPasswordSlotMask

Room password slot mask

Definition

```
#include <np.h>

typedef SceUInt64 SceNpMatching2RoomPasswordSlotMask;
```

Description

This type represents a room password slot mask.

It is a bit string representing the room password settings of the room slots. The least significant bit corresponds to slot number 1. If the room password of a slot is enabled, the corresponding bit value is 1.

See Also

SCE_NP_MATCHING2_ADD_SLOTNUM_TO_ROOM_PASSWORD_SLOT_MASK

SceNpMatching2RoomJoinedSlotMask

Joined room's slot mask

Definition

```
#include <np.h>

typedef SceUInt64 SceNpMatching2RoomJoinedSlotMask;
```

Description

This type represents a joined room's slot mask.

It is a bit string representing the occupancy (or lack of) of each slot for 1 room. The least significant bit corresponds to slot number 1. If the slot is occupied, the corresponding bit value will be 1.

SceNpMatching2Event

Event

Definition

```
#include <np.h>

typedef SceUShort16 SceNpMatching2Event;
```

Description

This type represents an event.

See Also

```
SCE_NP_MATCHING2_REQUEST_EVENT_*
SCE_NP_MATCHING2_ROOM_EVENT_*
SCE_NP_MATCHING2_ROOM_MSG_EVENT_*
SCE_NP_MATCHING2_LOBBY_EVENT_*
SCE_NP_MATCHING2_LOBBY_MSG_EVENT_*
SCE_NP_MATCHING2_CONTEXT_EVENT_*
```

Structures

SceNpMatching2SessionPassword

Session password

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_SESSION_PASSWORD_SIZE 8
typedef struct SceNpMatching2SessionPassword {
        SceUChar8 data[SCE_NP_MATCHING2_SESSION_PASSWORD_SIZE];
} SceNpMatching2SessionPassword;
```

Members

data Session password data

Description

This structure represents the session password.
It is specified when creating and joining a session.

See Also

sceNpMatching2CreateJoinRoom()
sceNpMatching2JoinRoom()
SceNpMatching2RoomOwnerUpdateInfo

SceNpMatching2PresenceOptionData

Optional presence data

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_PRESENCE_OPTION_DATA_SIZE 16
typedef struct SceNpMatching2PresenceOptionData {
        SceUChar8 data[SCE_NP_MATCHING2_PRESENCE_OPTION_DATA_SIZE];
        SceSize len;
} SceNpMatching2PresenceOptionData;
```

Members

data	Optional presence data
len	Length of optional presence data

Description

This structure represents optional presence data.

It is optional data that can be specified upon joining a room, leaving a room, granting room ownership, or kicking out a room member.

See Also

sceNpMatching2CreateJoinRoom()
sceNpMatching2JoinRoom()
SceNpMatching2RoomOwnerUpdateInfo

SceNpMatching2IntAttr

Integer-type attribute

Definition

```
#include <np.h>

typedef struct SceNpMatching2IntAttr {
        SceNpMatching2AttributeId id;
        SceUChar8 padding[2];
        SceUInt32 num;
} SceNpMatching2IntAttr;
```

Members

id	Attribute ID
padding	Padding
num	Value of an unsigned integer-type attribute

Description

This structure represents an integer-type attribute .

See Also

SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID

SceNpMatching2BinAttr

Binary-type attribute

Definition

```
#include <np.h>

typedef struct SceNpMatching2BinAttr {
        SceNpMatching2AttributeId id;
        SceUChar8 padding[2];
        const void *ptr;
        SceSize size;
} SceNpMatching2BinAttr;
```

Members

id	Attribute ID
padding	Padding
ptr	Pointer to binary-type attribute value
size	Size of binary-type attribute value

Description

This structure represents a binary-type attribute.

See Also

SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID
SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID
SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID
SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID

SceNpMatching2RangeFilter

Range filter

Definition

```
#include <np.h>

typedef struct SceNpMatching2RangeFilter {
        SceUInt32 startIndex;
        SceUInt32 max;
} SceNpMatching2RangeFilter;
```

Members

startIndex	Position to start obtaining elements
max	Maximum number of elements to obtain

Description

This structure represents the range of elements to obtain.

It is used when searching for rooms to specify the range of the room list to obtain.

The maximum value that can be specified in *max* is SCE_NP_MATCHING2_RANGE_FILTER_MAX.

When SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_RANDOM is specified as a room search option, the value of *startIndex* will be ignored.

See Also

sceNpMatching2SearchRoom()
SCE_NP_MATCHING2_RANGE_FILTER_START_INDEX_MIN
SCE_NP_MATCHING2_RANGE_FILTER_MAX

SceNpMatching2IntSearchFilter

Integer-type search condition

Definition

```
#include <np.h>

typedef struct SceNpMatching2IntSearchFilter {
        SceNpMatching2Operator searchOperator;
        SceUChar8 padding[3];
        SceNpMatching2IntAttr attr;
} SceNpMatching2IntSearchFilter;
```

Members

searchOperator	Comparison operator
padding	Padding
attr	Attribute for comparison

Description

This structure represents an integer-type search condition.

It is used when searching for rooms to specify an integer-type attribute to search for.

Specify SCE_NP_MATCHING2_OPERATOR_* to the *searchOperator* member of this structure.

Specify SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID to the *id* member of the SceNpMatching2IntAttr structure that the *attr* member of this structure indicates.

See Also

sceNpMatching2SearchRoom()
SCE_NP_MATCHING2_OPERATOR_*

SceNpMatching2BinSearchFilter

Binary-type search condition

Definition

```
#include <np.h>

typedef struct SceNpMatching2BinSearchFilter {
        SceNpMatching2Operator searchOperator;
        SceUChar8 padding[3];
        SceNpMatching2BinAttr attr;
} SceNpMatching2BinSearchFilter;
```

Members

searchOperator	Comparison operator
padding	Padding
attr	Attribute for comparison

Description

This structure represents a binary-type search condition.

It is used when searching for rooms to specify a binary-type attribute to search for.

Specify SCE_NP_MATCHING2_OPERATOR_EQ or SCE_NP_MATCHING2_OPERATOR_NE to the *searchOperator* member of this structure.

Specify SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID to the *id* member of the SceNpMatching2BinAttr structure that the *attr* member of this structure indicates.

See Also

sceNpMatching2SearchRoom()
SCE_NP_MATCHING2_OPERATOR_*

SceNpMatching2Range

Range of result

Definition

```
#include <np.h>

typedef struct SceNpMatching2Range {
        SceUInt32 startIndex;
        SceUInt32 total;
        SceUInt32 resultCount;
} SceNpMatching2Range;
```

Members

startIndex	Position to start obtaining elements
total	Total number of elements matching the search conditions
resultCount	Number of elements actually obtained

Description

This structure represents the range of the search result.

It indicates the range of the room list actually obtained as the result of the room search.

When specifying SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_RANDOM as the room search option, the value of *startIndex* will always be 0.

See Also

sceNpMatching2SearchRoom()

SceNpMatching2JoinedSessionInfo

Session information about a session joined by the user

Definition

```
#include <np.h>

typedef struct SceNpMatching2JoinedSessionInfo {
        SceNpMatching2SessionType sessionType;
        SceUChar8 padding1[1];
        SceNpMatching2ServerId serverId;
        SceNpMatching2WorldId worldId;
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2RoomId roomId;
        SceRtcTick joinDate;
} SceNpMatching2JoinedSessionInfo;
```

Members

sessionType	Session type of joined session
padding1	Padding
serverId	Server ID of server to which joined session belongs
worldId	World ID of world to which joined session belongs
lobbyId	Lobby ID of lobby to which joined session belongs
roomId	Room ID of room to which joined session belongs
joinDate	Date and time user joined session

Description

When a user joins a session, this structure represents information of the session that is joined.

This structure represents one joined session. If multiple sessions are joined, multiple instances of this structure are used.

See Also

SceNpMatching2UserInfo

SceNpMatching2UserInfo

User information

Definition

```
#include <np.h>

typedef struct SceNpMatching2UserInfo {
        struct SceNpMatching2UserInfo *next;
        SceNpId npId;
        SceNpMatching2BinAttr *userBinAttr;
        SceUInt32 userBinAttrNum;
        SceNpMatching2JoinedSessionInfo *joinedSessionInfo;
        SceUInt32 joinedSessionInfoNum;
} SceNpMatching2UserInfo;
```

Members

next	Pointer to next element of the list
npId	User information
userBinAttr	Pointer to user binary attribute array Specify SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID for the *id* member of SceNpMatching2BinAttr to be specified here
userBinAttrNum	Number of elements in user binary attribute array
joinedSessionInfo	Pointer to joined session information array
joinedSessionInfoNum	Number of elements in joined session information array

Description

This structure represents user information.

The array indicated by the *joinedSessionInfo* member indicates one or more sessions that the user has joined.

See Also

sceNpMatching2GetUserInfoList()

SceNpMatching2Server

Server

Definition

```
#include <np.h>

typedef struct SceNpMatching2Server {
        SceNpMatching2ServerId serverId;
        SceNpMatching2ServerStatus status;
        SceUChar8 padding[1];
} SceNpMatching2Server;
```

Members

serverId	Server ID
status	Server status SCE_NP_MATCHING2_SERVER_STATUS_* will be stored
padding	Padding

Description

This structure represents a server.

See Also

sceNpMatching2GetServerLocal()
SCE_NP_MATCHING2_SERVER_STATUS_*

SceNpMatching2World

World

Definition

```
#include <np.h>

typedef struct SceNpMatching2World {
        SceNpMatching2WorldId worldId;
        SceUInt32 numOfLobby;
        SceUInt32 maxNumOfTotalLobbyMember;
        SceUInt32 curNumOfTotalLobbyMember;
        SceUInt32 curNumOfRoom;
        SceUInt32 curNumOfTotalRoomMember;
        bool withEntitlementId;
        SceNpEntitlementId entitlementId;
        SceUChar8 padding[3];
} SceNpMatching2World;
```

Members

worldId	World ID
numOfLobby	Number of lobbies belonging to the world
maxNumOfTotalLobbyMember	Maximum number of lobby members belonging to the world
curNumOfTotalLobbyMember	Number of lobby members currently belonging to the world
curNumOfRoom	Number of rooms currently belonging to the world
curNumOfTotalRoomMember	Number of room members currently belonging to the world
withEntitlementId	Flag indicating whether or not an entitlement ID is included. If an entitlement ID is included, the entitlement ID is required in order to use this world. (unused)
entitlementId	Entitlement ID (unused)
padding	Padding

Description

This structure represents a world.

SceNpMatching2LobbyMemberBinAttrInternal

Lobby member internal binary attribute

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyMemberBinAttrInternal {
        SceRtcTick updateDate;
        SceNpMatching2BinAttr data;
        SceUChar8 padding[4];
} SceNpMatching2LobbyMemberBinAttrInternal;
```

Members

updateDate	Last update date and time
data	Lobby member internal binary attribute data `SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID` will be stored in the *id* of `SceNpMatching2BinAttr` to be specified here
padding	Padding

Description

This structure represents the lobby member's internal binary attribute.

See Also

`SceNpMatching2LobbyMemberDataInternal`
`SceNpMatching2LobbyMemberDataInternalUpdateInfo`

SceNpMatching2LobbyMemberDataInternal

Lobby-internal lobby member information

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyMemberDataInternal {
        struct SceNpMatching2LobbyMemberDataInternal *next;
        SceNpId npId;

        SceRtcTick joinDate;
        SceNpMatching2LobbyMemberId memberId;
        SceUChar8 padding[2];

        SceNpMatching2FlagAttr flagAttr;

        SceNpMatching2JoinedSessionInfo *joinedSessionInfo;
        SceUInt32 joinedSessionInfoNum;
        SceNpMatching2LobbyMemberBinAttrInternal
*lobbyMemberBinAttrInternal;
        SceUInt32 lobbyMemberBinAttrInternalNum;
} SceNpMatching2LobbyMemberDataInternal;
```

Members

next	Pointer to next element of the list
npId	Lobby member's user information
joinDate	Date and time user joined lobby
memberId	Lobby member ID
padding	Padding
flagAttr	Lobby member flag attribute (unused)
joinedSessionInfo	Pointer to joined session information array
joinedSessionInfoNum	Number of elements in joined session information array
lobbyMemberBinAttrInternal	Pointer to lobby member internal binary attribute array
lobbyMemberBinAttrInternalNum	Number of elements in lobby member internal binary attribute array

Description

This structure represents lobby member information that can be obtained from within a lobby.

See Also

SceNpMatching2LobbyMemberUpdateInfo

sceNpMatching2GetLobbyMemberDataInternal()

SceNpMatching2LobbyMemberIdList

Lobby member ID list

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyMemberIdList {
        SceNpMatching2LobbyMemberId *memberId;
        SceUInt32 memberIdNum;
        SceNpMatching2LobbyMemberId me;
        SceUChar8 padding[6];
} SceNpMatching2LobbyMemberIdList;
```

Members

memberId	Pointer to lobby member ID array
memberIdNum	Number of elements in lobby member ID array
me	User's own lobby member ID
padding	Padding

Description

This structure represents a lobby member ID list.

See Also

SceNpMatching2LobbyDataInternal

SceNpMatching2LobbyBinAttrInternal

Lobby-internal binary attribute

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyBinAttrInternal {
        SceRtcTick updateDate;
        SceNpMatching2LobbyMemberId updateMemberId;
        SceUChar8 padding[2];
        SceNpMatching2BinAttr data;
} SceNpMatching2LobbyBinAttrInternal;
```

Members

updateDate	Last update date and time
updateMemberId	Lobby member ID of lobby member that was updated last
padding	Padding
data	Lobby-internal binary attribute data `SCE_NP_MATCHING2_LOBBY_BIN_ATTR_INTERNAL_*_ID` will be stored in the *id* of `SceNpMatching2BinAttr` to be set here

Description

This structure represents a lobby-internal binary attribute.

See Also

`SceNpMatching2LobbyDataInternal`

SceNpMatching2LobbyDataExternal

Lobby-external lobby information

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyDataExternal {
        struct SceNpMatching2LobbyDataExternal *next;
        SceNpMatching2ServerId serverId;
        SceUChar8 padding1[2];
        SceNpMatching2WorldId worldId;
        SceUChar8 padding2[4];
        SceNpMatching2LobbyId lobbyId;
        SceUInt32 maxSlot;
        SceUInt32 curMemberNum;
        SceNpMatching2FlagAttr flagAttr;
        SceNpMatching2IntAttr *lobbySearchableIntAttrExternal;
        SceUInt32 lobbySearchableIntAttrExternalNum;
        SceNpMatching2BinAttr *lobbySearchableBinAttrExternal;
        SceUInt32 lobbySearchableBinAttrExternalNum;
        SceNpMatching2BinAttr *lobbyBinAttrExternal;
        SceUInt32 lobbyBinAttrExternalNum;
        SceUChar8 padding3[4];
} SceNpMatching2LobbyDataExternal;
```

Members

next	Pointer to next element of the list
serverId	Server ID of server to which lobby belongs
padding1	Padding
worldId	World ID of word to which lobby belongs
padding2	Padding
lobbyId	Lobby ID of lobby
maxSlot	Total number of lobby slots
curMemberNum	Current number of lobby members
flagAttr	Lobby flag attribute
lobbySearchableIntAttrExternal	Lobby-external search numeric attribute array (unused)
lobbySearchableIntAttrExternalNum	Number of lobby-external search numeric attribute array elements (unused)
lobbySearchableBinAttrExternal	Lobby-external search binary attribute array (unused)
lobbySearchableBinAttrExternalNum	Number of lobby-external search binary attribute array elements (unused)
lobbyBinAttrExternal	Lobby-external binary attribute array (unused)
lobbyBinAttrExternalNum	Number of lobby-external binary attribute array elements (unused)
padding3	Padding

Description

This structure represents lobby information that can be obtained from outside a lobby.

See Also

SceNpMatching2GetLobbyInfoListResponse

SceNpMatching2LobbyDataInternal

Lobby-internal lobby information

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyDataInternal {
        SceNpMatching2ServerId serverId;
        SceUChar8 padding1[2];
        SceNpMatching2WorldId worldId;
        SceNpMatching2LobbyId lobbyId;

        SceUInt32 maxSlot;
        SceNpMatching2LobbyMemberIdList memberIdList;
        SceNpMatching2FlagAttr flagAttr;

        SceNpMatching2LobbyBinAttrInternal *lobbyBinAttrInternal;
        SceUInt32 lobbyBinAttrInternalNum;
} SceNpMatching2LobbyDataInternal;
```

Members

serverId	Server ID of server to which lobby belongs
padding1	Padding
worldId	World ID of world to which lobby belongs
lobbyId	Lobby ID of lobby
maxSlot	Total number of lobby slots
memberIdList	Lobby member ID list
flagAttr	Lobby flag attribute The OR of SCE_NP_MATCHING2_LOBBY_FLAG_ATTR_* will be stored
lobbyBinAttrInternal	Lobby-internal binary attribute array (unused)
lobbyBinAttrInternalNum	Number of lobby-internal binary attribute array elements (unused)

Description

This structure represents lobby information that can be obtained from within a lobby.

See Also

SceNpMatching2JoinLobbyResponse

SceNpMatching2LobbyMessageDestination

Lobby message transmission destination

Definition

```
#include <np.h>

typedef union SceNpMatching2LobbyMessageDestination {
        SceNpMatching2LobbyMemberId unicastTarget;
        struct {
              const SceNpMatching2LobbyMemberId *memberId;
              SceUInt32 memberIdNum;
        } multicastTarget;
} SceNpMatching2LobbyMessageDestination;
```

Members

unicastTarget	Lobby member ID of member that is transmission destination when unicasting
multicastTarget	Transmission destination when multicasting
memberId	Lobby member ID array of members that are transmission destinations when multicasting
memberIdNum	Number of lobby member ID array elements for members that are transmission destinations when multicasting

Description

This union represents the transmission destination of lobby chat messages.

Determine the member to access depending on the type of message to send (SCE_NP_MATCHING2_CASTTYPE_*).

Message Type	Member
SCE_NP_MATCHING2_CASTTYPE_BROADCAST	None
SCE_NP_MATCHING2_CASTTYPE_UNICAST	*unicastTarget*
SCE_NP_MATCHING2_CASTTYPE_MULTICAST	*multicastTarget*
SCE_NP_MATCHING2_CASTTYPE_MULTICAST_TEAM	None

See Also

SceNpMatching2SendLobbyChatMessageRequest

SceNpMatching2LobbyMessageInfo

SceNpMatching2GroupLabel

Group label

Definition

```
#include <np.h>

#define SCE_NP_MATCHING2_GROUP_LABEL_SIZE 8
typedef struct SceNpMatching2GroupLabel {
        SceUChar8 data[SCE_NP_MATCHING2_GROUP_LABEL_SIZE];
} SceNpMatching2GroupLabel;
```

Members

data Group label data

Description

This structure represents a group label.
It is used when creating a room with groups or when joining a group in a room.

See Also

SceNpMatching2RoomGroupConfig
SceNpMatching2RoomGroup
SceNpMatching2CreateJoinRoomRequest
SceNpMatching2JoinRoomRequest

SceNpMatching2RoomGroupConfig

Set groups in a room

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomGroupConfig {
        SceUInt32 slotNum;
        bool withLabel;
        SceNpMatching2GroupLabel label;
        bool withPassword;
        SceUChar8 padding[2];
} SceNpMatching2RoomGroupConfig;
```

Members

slotNum	Number of slots in the group
withLabel	Flag indicating whether or not to set group labels If false, the group will have no group label set to it
label	Group label to set to group
withPassword	Flag indicating whether or not to enable a room password for the group
padding	Padding

Description

This structure is used when creating groups in a room.

One `SceNpMatching2RoomGroupConfig` structure has the settings of one group. To set multiple groups, prepare as many structures as necessary in an array. The elements with smaller array indices correspond to the groups with smaller room slot numbers.

Examples

This example creates a room with groups as follows.

- Total number of slots: 8
- Number of groups: 2
- Configuration of groups: Group 1 (Slot numbers 1-4), Group 2 (Slot numbers 5-8)
- Set group labels
- Enable room passwords

```
SceNpMatching2RoomGroupConfig groupConfig[2];

groupConfig[0].slotNum = 4;
groupConfig[0].withLabel = true;
memcpy(groupConfig[0].label.data, "LABEL01", 7);
groupConfig[0].withPassword = true;

groupConfig[1].slotNum = 4;
groupConfig[1].withLabel = true;
memcpy(groupConfig[1].label.data, "LABEL02", 7);
groupConfig[1].withPassword = true;
```

See Also

`SceNpMatching2CreateJoinRoomRequest`
`sceNpMatching2CreateJoinRoom()`

SceNpMatching2RoomGroupPasswordConfig

Set group password

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomGroupPasswordConfig {
        SceNpMatching2RoomGroupId groupId;
        bool withPassword;
        SceUChar8 padding[1];
} SceNpMatching2RoomGroupPasswordConfig;
```

Members

groupId	ID of group to set password
withPassword	Flag indicating whether or not to enable room passwords for the group
padding	Padding

Description

This structure is used to enable or disable the room password for a room group.

To set multiple groups, prepare as many structures as necessary in an array.

Examples

This example enables the room password of the group with group ID 1.

```
SceNpMatching2RoomGroupPasswordConfig groupPasswordConfig;

groupPasswordConfig.groupId = 1;
groupPasswordConfig.withPassword = true;
```

See Also

SceNpMatching2SetRoomDataInternalRequest

sceNpMatching2SetRoomDataInternal()

SceNpMatching2RoomGroup

Group (of slots in a room)

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomGroup {
        SceNpMatching2RoomGroupId groupId;
        bool withPassword;
        bool withLabel;
        SceUChar8 padding[1];
        SceNpMatching2GroupLabel label;
        SceUInt32 slotNum;
        SceUInt32 curGroupMemberNum;
} SceNpMatching2RoomGroup;
```

Members

groupId	ID of group
withPassword	Flag indicating whether or not room passwords are enabled
withLabel	Flag indicating whether or not group labels are set
padding	Padding
label	Label
slotNum	Number of slots in the group
curGroupMemberNum	Number of room members currently belonging to the group

Description

This structure represents a group.

See Also

SceNpMatching2RoomMemberDataInternal
SceNpMatching2RoomDataExternal
SceNpMatching2RoomDataInternal
SceNpMatching2RoomDataInternalUpdateInfo

SceNpMatching2RoomMemberBinAttrInternal

Internal room member binary attribute

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomMemberBinAttrInternal {
        SceRtcTick updateDate;
        SceNpMatching2BinAttr data;
        SceUChar8 padding[4];
} SceNpMatching2RoomMemberBinAttrInternal;
```

Members

updateDate	Last update date/time
data	Data of internal room member binary attribute `SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID` will be stored in the *id* of `SceNpMatching2BinAttr` to be set here
padding	Padding

Description

This structure represents an internal room member binary attribute.

See Also

`SceNpMatching2RoomMemberDataInternal`
`SceNpMatching2RoomMemberDataInternalUpdateInfo`

SceNpMatching2RoomMemberDataExternal

External room member data

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomMemberDataExternal {
        struct SceNpMatching2RoomMemberDataExternal *next;
        SceNpId npId;
        SceRtcTick joinDate;
        SceNpMatching2Role role;
        SceUChar8 padding[7];
} SceNpMatching2RoomMemberDataExternal;
```

Members

next	Pointer to the next element in the list
npId	User information of room member
joinDate	Date/time room was joined
role	Role in the room SCE_NP_MATCHING2_ROLE_* will be stored
padding	Padding

Description

This structure represents the room member data available to users outside the room.

See Also

SceNpMatching2GetRoomMemberDataExternalListResponse

SceNpMatching2RoomMemberDataInternal

Internal room member data

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomMemberDataInternal {
        struct SceNpMatching2RoomMemberDataInternal *next;
        SceNpId npId;

        SceRtcTick joinDate;
        SceNpMatching2RoomMemberId memberId;
        SceNpMatching2TeamId teamId;
        SceUChar8 padding1[1];

        SceNpMatching2RoomGroup *roomGroup;

        SceNpMatching2NatType natType;
        SceUChar8 padding2[3];
        SceNpMatching2FlagAttr flagAttr;
        SceNpMatching2RoomMemberBinAttrInternal *roomMemberBinAttrInternal;
        SceUInt32 roomMemberBinAttrInternalNum;
} SceNpMatching2RoomMemberDataInternal;
```

Members

next	Pointer to the next element in the list
npId	User information of room member
joinDate	Date/time room was joined
memberId	Room member ID
teamId	Team ID
padding1	Padding
roomGroup	Group to which room member belongs
natType	NAT type
padding2	Padding
flagAttr	Room member flag attributes The OR of `SCE_NP_MATCHING2_ROOMMEMBER_FLAG_ATTR_*` will be stored
roomMemberBinAttrInternal	Pointer to array of internal room member binary attributes
roomMemberBinAttrInternalNum	Number of elements in array of internal room member binary attributes

Description

This structure represents the room member data available to users inside the room.

See Also

`SceNpMatching2RoomMemberDataInternalList`
`SceNpMatching2RoomMemberUpdateInfo`
`SceNpMatching2RoomMemberDataInternalUpdateInfo`
`sceNpMatching2GetRoomMemberDataInternalLocal()`

SceNpMatching2RoomMemberDataInternalList

Internal room member data list

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomMemberDataInternalList {
        SceNpMatching2RoomMemberDataInternal *members;
        SceUInt32 membersNum;
        SceNpMatching2RoomMemberDataInternal *me;
        SceNpMatching2RoomMemberDataInternal *owner;
} SceNpMatching2RoomMemberDataInternalList;
```

Members

members	Pointer to top of room member list
membersNum	Number of elements in room member list
me	Pointer to element representing oneself in the member list pointed to by *members*
owner	Pointer to element representing the room owner in the member list pointed to by *members*

Description

This structure represents a list of internal room member data.

See Also

SceNpMatching2RoomDataInternal

SceNpMatching2RoomBinAttrInternal

Internal room binary attribute

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomBinAttrInternal {
        SceRtcTick updateDate;
        SceNpMatching2RoomMemberId updateMemberId;
        SceUChar8 padding[2];
        SceNpMatching2BinAttr data;
} SceNpMatching2RoomBinAttrInternal;
```

Members

updateDate	Last update date/time
updateMemberId	ID of room member last updated
padding	Padding
data	Data of internal room binary attribute `SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID` will be stored in the *id* of `SceNpMatching2BinAttr` to be set here

Description

This structure represents an internal room binary attribute.

See Also

`SceNpMatching2RoomDataInternal`
`SceNpMatching2RoomDataInternalUpdateInfo`

SceNpMatching2RoomDataExternal

External room data

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomDataExternal {
        struct SceNpMatching2RoomDataExternal *next;
        SceUShort16 maxSlot;
        SceUShort16 curMemberNum;
        SceNpMatching2ServerId serverId;
        SceUChar8 padding[2];
        SceNpMatching2WorldId worldId;
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2RoomId roomId;
        SceNpMatching2RoomPasswordSlotMask passwordSlotMask;
        SceNpMatching2RoomJoinedSlotMask joinedSlotMask;
        SceUShort16 publicSlotNum;
        SceUShort16 privateSlotNum;
        SceUShort16 openPublicSlotNum;
        SceUShort16 openPrivateSlotNum;
        SceNpId *owner;
        SceNpMatching2FlagAttr flagAttr;
        SceNpMatching2RoomGroup *roomGroup;
        SceUInt32 roomGroupNum;
        SceNpMatching2IntAttr *roomSearchableIntAttrExternal;
        SceUInt32 roomSearchableIntAttrExternalNum;
        SceNpMatching2BinAttr *roomSearchableBinAttrExternal;
        SceUInt32 roomSearchableBinAttrExternalNum;
        SceNpMatching2BinAttr *roomBinAttrExternal;
        SceUInt32 roomBinAttrExternalNum;
} SceNpMatching2RoomDataExternal;
```

Members

next	Pointer to the next element in the list
maxSlot	Total number of slots in room
curMemberNum	Current number of room members
serverId	ID of server to which the room belongs
padding	Padding
worldId	ID of world to which the room belongs
lobbyId	ID of lobby to which the room belongs
roomId	Room ID
passwordSlotMask	Password slot mask. Indicates the password of slots if the room is not organized into groups.
joinedSlotMask	Joined room's slot mask
publicSlotNum	Number of public slots
privateSlotNum	Number of reserved slots
openPublicSlotNum	Number of unused public slots
openPrivateSlotNum	Number of unused reserved slots
owner	User information of room owner
flagAttr	Room flag attributes The OR of SCE_NP_MATCHING2_ROOM_FLAG_ATTR_* will be stored

roomGroup	Array of group structures indicating the configuration of groups in the room. NULL when the room is not organized into groups.
roomGroupNum	Number of elements in array of group structures
roomSearchableIntAttrExternal	Array of external room search integer attributes `SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID` will be stored in the *id* member of `SceNpMatching2IntAttr` to be set here
roomSearchableIntAttrExternalNum	Number of elements in array of external room search integer attributes
roomSearchableBinAttrExternal	Array of external room search binary attributes `SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID` will be stored in the *id* member of `SceNpMatching2BinAttr` to be set here
roomSearchableBinAttrExternalNum	Number of elements in array of external room search binary attributes
roomBinAttrExternal	Array of external room binary attributes `SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID` will be stored in the *id* member of `SceNpMatching2BinAttr` to be set here
roomBinAttrExternalNum	Number of elements in array of external room binary attributes

Description

This structure represents the room data available to users outside the room.

See Also

`SceNpMatching2GetRoomDataExternalListResponse`
`SceNpMatching2SearchRoomResponse`

SceNpMatching2RoomDataInternal

Internal room data

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomDataInternal {
        SceUShort16 maxSlot;
        SceNpMatching2ServerId serverId;
        SceNpMatching2WorldId worldId;
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2RoomId roomId;
        SceNpMatching2RoomPasswordSlotMask passwordSlotMask;
        SceNpMatching2RoomJoinedSlotMask joinedSlotMask;
        SceUShort16 publicSlotNum;
        SceUShort16 privateSlotNum;
        SceUShort16 openPublicSlotNum;
        SceUShort16 openPrivateSlotNum;
        SceNpMatching2RoomMemberDataInternalList memberList;
        SceNpMatching2RoomGroup *roomGroup;
        SceUInt32 roomGroupNum;
        SceNpMatching2FlagAttr flagAttr;
        SceUChar8 padding[4];
        SceNpMatching2RoomBinAttrInternal *roomBinAttrInternal;
        SceUInt32 roomBinAttrInternalNum;
} SceNpMatching2RoomDataInternal;
```

Members

maxSlot	Total number of slots in room
serverId	ID of server to which the room belongs
worldId	ID of world to which the room belongs
lobbyId	ID of lobby to which the room belongs
roomId	Room ID
passwordSlotMask	Password slot mask. Indicates the password of slots if the room is not organized into groups.
joinedSlotMask	Joined room's slot mask
publicSlotNum	Number of public slots
privateSlotNum	Number of reserved slots
openPublicSlotNum	Number of unused public slots
openPrivateSlotNum	Number of unused reserved slots
memberList	Room member list
roomGroup	Array of group structures indicating the configuration of groups in the room. NULL when the room is not organized into groups.
roomGroupNum	Number of elements in array of group structures
flagAttr	Room flag attributes The OR of `SCE_NP_MATCHING2_ROOM_FLAG_ATTR_*` will be stored
padding	Padding
roomBinAttrInternal	Array of internal room binary attributes `SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID` will be stored in the *id* member of `SceNpMatching2BinAttr` to be set here
roomBinAttrInternalNum	Number of elements in array of internal room binary attributes

Description

This structure represents the room data available to users inside the room.

See Also

`SceNpMatching2CreateJoinRoomResponse`
`SceNpMatching2JoinRoomResponse`
`SceNpMatching2GetRoomDataInternalResponse`

SceNpMatching2RoomMessageDestination

Room message recipient

Definition

```
#include <np.h>

typedef union SceNpMatching2RoomMessageDestination {
        SceNpMatching2RoomMemberId unicastTarget;
        struct {
              const SceNpMatching2RoomMemberId *memberId;
              SceUInt32 memberIdNum;
        } multicastTarget;
        SceNpMatching2TeamId multicastTargetTeamId;
} SceNpMatching2RoomMessageDestination;
```

Members

unicastTarget	ID of room member targeted in a unicast
multicastTarget	Targets of a multicast
memberId	Array of room member IDs targeted in a multicast
memberIdNum	Number of elements in array of room member IDs targeted in a multicast
multicastTargetTeamId	ID of team targeted in a multicast

Description

This union represents the recipient of a room chat message or a room message.

Determine the member to access depending on the type of message to send (SCE_NP_MATCHING2_CASTTYPE_*).

Message Type	Member
SCE_NP_MATCHING2_CASTTYPE_BROADCAST	None
SCE_NP_MATCHING2_CASTTYPE_UNICAST	*unicastTarget*
SCE_NP_MATCHING2_CASTTYPE_MULTICAST	*multicastTarget*
SCE_NP_MATCHING2_CASTTYPE_MULTICAST_TEAM	*multicastTeamId*

See Also

SceNpMatching2SendRoomMessageRequest

SceNpMatching2SendRoomChatMessageRequest

SceNpMatching2RoomMessageInfo

SceNpMatching2SignalingOptParam

Signaling option parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2SignalingOptParam {
        SceNpMatching2SignalingType type;
        SceNpMatching2SignalingFlag flag;
        SceNpMatching2RoomMemberId hubMemberId;
        SceUChar8 reserved2[4];
} SceNpMatching2SignalingOptParam;
```

Members

type	Signaling type (P2P connection topology) Specify SCE_NP_MATCHING2_SIGNALING_TYPE_*
flag	Signaling flag Specify SCE_NP_MATCHING2_SIGNALING_FLAG_*
hubMemberId	Room member ID of room member that becomes the hub when the signaling type (P2P connection topology) is star type
reserved2	Reserved area

Description

This structure represents the signaling option parameter for P2P connection establishment processing.

See Also

SceNpMatching2CreateJoinRoomRequest
SceNpMatching2SetSignalingOptParamRequest
SCE_NP_MATCHING2_SIGNALING_TYPE_*

SceNpMatching2RequestOptParam

Option parameters for requests

Definition

```
#include <np.h>

typedef struct SceNpMatching2RequestOptParam {
        SceNpMatching2RequestCallback cbFunc;
        void *cbFuncArg;
        SceUInt32 timeout;
        SceUShort16 appReqId;
        SceUChar8 padding[2];
} SceNpMatching2RequestOptParam;
```

Members

cbFunc	Request callback function
cbFuncArg	Pointer to data to pass to the request callback function
timeout	Time before request of request function times out (microseconds)
appReqId	Upper 16 bits of the request ID
padding	Padding

Description

This structure represents the optional parameters common to all request functions.

It is passed to `sceNpMatching2SetDefaultRequestOptParam()` to set the default for all requests, or to an individual request function upon executing the function, to set the settings for that particular request.

See Also

`sceNpMatching2SetDefaultRequestOptParam()`

All the request functions

Request and Response Structures

SceNpMatching2GetWorldInfoListRequest

World data list request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetWorldInfoListRequest {
        SceNpMatching2ServerId serverId;
} SceNpMatching2GetWorldInfoListRequest;
```

Members

serverId ID of the target server to obtain list of world data

Description

This structure represents the parameters of a request for a world data list.
It is specified as an argument of `sceNpMatching2GetWorldInfoList()`.

See Also

`sceNpMatching2GetWorldInfoList()`
`SceNpMatching2GetWorldInfoListResponse`

SceNpMatching2GetWorldInfoListResponse

World data list request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetWorldInfoListResponse {
        SceNpMatching2World *world;
        SceUInt32 worldNum;
} SceNpMatching2GetWorldInfoListResponse;
```

Members

world	Pointer to array of world data
worldNum	Number of elements in array of world data

Description

This structure represents the response data to a request for a world data list.

See Also

sceNpMatching2GetWorldInfoList()
SceNpMatching2GetWorldInfoListRequest

SceNpMatching2SetUserInfoRequest

User information setting request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2SetUserInfoRequest {
        SceNpMatching2ServerId serverId;
        SceUChar8 padding[2];
        const SceNpMatching2BinAttr *userBinAttr;
        SceUInt32 userBinAttrNum;
} SceNpMatching2SetUserInfoRequest;
```

Members

serverId	Server ID of server for which user information is to be set
padding	Padding
userBinAttr	Pointer to user binary attribute array Specify SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID
userBinAttrNum	Number of user binary attribute array elements

Description

This structure represents the user information setting request parameter.

It is specified for an argument of sceNpMatching2SetUserInfo().

See Also

sceNpMatching2SetUserInfo()

SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID

SceNpMatching2GetUserInfoListRequest

User information list acquisition request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetUserInfoListRequest {
        SceNpMatching2ServerId serverId;
        SceUChar8 padding[2];
        const SceNpId *npId;
        SceUInt32 npIdNum;
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
        int option;
} SceNpMatching2GetUserInfoListRequest;
```

Members

serverId	Server ID of server for which user information list is to be obtained
padding	Padding
npId	Pointer to NP ID array of user for which user information is to be obtained
npIdNum	Number of elements of NP ID array of user for which user information is to be obtained
attrId	Pointer to attribute ID array of user binary attributes to be obtained Specify SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID.
attrIdNum	Number of elements of attribute ID array of user binary attributes to be obtained
option	Unused

Description

This structure represents the user information list setting request parameter.

It is specified for an argument of sceNpMatching2GetUserInfoList().

See Also

sceNpMatching2GetUserInfoList()

SceNpMatching2GetUserInfoListResponse

SCE_NP_MATCHING2_USER_BIN_ATTR_*_ID

SceNpMatching2GetUserInfoListResponse

User information list acquisition response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetUserInfoListResponse {
        SceNpMatching2UserInfo *userInfo;
        SceUInt32 userInfoNum;
} SceNpMatching2GetUserInfoListResponse;
```

Members

userInfo	Pointer to array of user information that was obtained
userInfoNum	Number of elements in array of user information that was obtained

Description

This structure represents response data for a user information list setting request.

See Also

sceNpMatching2GetUserInfoList()
SceNpMatching2GetUserInfoListRequest

SceNpMatching2GetRoomMemberDataExternalList Request

External room member data list request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomMemberDataExternalListRequest {
        SceNpMatching2RoomId roomId;
} SceNpMatching2GetRoomMemberDataExternalListRequest;
```

Members

roomId ID of room to obtain external room member data list

Description

This structure represents the parameters of a request for an external room member data list.
It is specified as an argument of `sceNpMatching2GetRoomMemberDataExternalList()`.

See Also

`sceNpMatching2GetRoomMemberDataExternalList()`
`SceNpMatching2GetRoomMemberDataExternalListResponse`

SceNpMatching2GetRoomMemberDataExternalList Response

External room member data list request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomMemberDataExternalListResponse {
        SceNpMatching2RoomMemberDataExternal *roomMemberDataExternal;
        SceUInt32 roomMemberDataExternalNum;
} SceNpMatching2GetRoomMemberDataExternalListResponse;
```

Members

roomMemberDataExternal	Pointer to start of external room member data list
roomMemberDataExternalNum	Number of elements in external room member data list

Description

This structure represents the response data to a request for an external room member data list.

See Also

sceNpMatching2GetRoomMemberDataExternalList()
SceNpMatching2GetRoomMemberDataExternalListRequest

SceNpMatching2SetRoomDataExternalRequest

External room data configuration request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2SetRoomDataExternalRequest {
        SceNpMatching2RoomId roomId;
        const SceNpMatching2IntAttr *roomSearchableIntAttrExternal;
        SceUInt32 roomSearchableIntAttrExternalNum;
        const SceNpMatching2BinAttr *roomSearchableBinAttrExternal;
        SceUInt32 roomSearchableBinAttrExternalNum;
        const SceNpMatching2BinAttr *roomBinAttrExternal;
        SceUInt32 roomBinAttrExternalNum;
} SceNpMatching2SetRoomDataExternalRequest;
```

Members

roomId	ID of the room to set external room data
roomSearchableIntAttrExternal	Pointer to array of external room search integer attributes Specify SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID for the *id* member of SceNpMatching2IntAttr to be specified here
roomSearchableIntAttrExternalNum	Number of elements in array of external room search integer attributes
roomSearchableBinAttrExternal	Pointer to array of external room search binary attributes Specify SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID for the *id* member of SceNpMatching2BinAttr to be specified here
roomSearchableBinAttrExternalNum	Number of elements in array of external room search binary attributes
roomBinAttrExternal	Pointer to array of external room binary attributes Specify SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID for the *id* member of SceNpMatching2BinAttr to be specified here
roomBinAttrExternalNum	Number of elements in array of external room binary attributes

Description

This structure represents the parameters of a request to configure external room data.

It is specified as an argument of sceNpMatching2SetRoomDataExternal().

See Also

sceNpMatching2SetRoomDataExternal()

SceNpMatching2GetRoomDataExternalListRequest

External room data list request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomDataExternalListRequest {
        const SceNpMatching2RoomId *roomId;
        SceUInt32 roomIdNum;
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
} SceNpMatching2GetRoomDataExternalListRequest;
```

Members

roomId	Pointer to array of room IDs to obtain external room data
roomIdNum	Number of elements in array of room IDs to obtain external room data
attrId	Pointer to array of attribute IDs to obtain optional attribute data For the attribute IDs, specify SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID, SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID, SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID.
attrIdNum	Number of elements in array of attribute IDs to obtain optional attribute data

Description

This structure represents the parameters of a request for an external room data list.

It is specified as an argument of `sceNpMatching2GetRoomDataExternalList()`.

See Also

`sceNpMatching2GetRoomDataExternalList()`
`SceNpMatching2GetRoomDataExternalListResponse`
`SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID`
`SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID`
`SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID`

SceNpMatching2GetRoomDataExternalListResponse

External room data list request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomDataExternalListResponse {
        SceNpMatching2RoomDataExternal *roomDataExternal;
        SceUInt32 roomDataExternalNum;
} SceNpMatching2GetRoomDataExternalListResponse;
```

Members

roomDataExternal	Pointer to start of external room data list
roomDataExternalNum	Number of elements in external room data list

Description

This structure represents the response data to a request for an external room data list.

See Also

sceNpMatching2GetRoomDataExternalList()
SceNpMatching2GetRoomDataExternalListRequest

SceNpMatching2CreateJoinRoomRequest

Create-and-join room request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2CreateJoinRoomRequest {
        SceNpMatching2WorldId worldId;
        SceUChar8 padding1[4];
        SceNpMatching2LobbyId lobbyId;

        SceUInt32 maxSlot;
        SceNpMatching2FlagAttr flagAttr;
        const SceNpMatching2BinAttr *roomBinAttrInternal;
        SceUInt32 roomBinAttrInternalNum;
        const SceNpMatching2IntAttr *roomSearchableIntAttrExternal;
        SceUInt32 roomSearchableIntAttrExternalNum;
        const SceNpMatching2BinAttr *roomSearchableBinAttrExternal;
        SceUInt32 roomSearchableBinAttrExternalNum;
        const SceNpMatching2BinAttr *roomBinAttrExternal;
        SceUInt32 roomBinAttrExternalNum;
        const SceNpMatching2SessionPassword *roomPassword;
        const SceNpMatching2RoomGroupConfig *groupConfig;
        SceUInt32 groupConfigNum;
        const SceNpMatching2RoomPasswordSlotMask *passwordSlotMask;
        const SceNpId *allowedUser;
        SceUInt32 allowedUserNum;
        const SceNpId *blockedUser;
        SceUInt32 blockedUserNum;

        const SceNpMatching2GroupLabel *joinRoomGroupLabel;
        const SceNpMatching2BinAttr *roomMemberBinAttrInternal;
        SceUInt32 roomMemberBinAttrInternalNum;
        SceNpMatching2TeamId teamId;
        SceUChar8 padding2[3];

        const SceNpMatching2SignalingOptParam *sigOptParam;
        SceUChar8 padding3[4];
} SceNpMatching2CreateJoinRoomRequest;
```

Members

worldId	ID of world to create a room
padding1	Padding
lobbyId	ID of lobby to create a room
maxSlot	Total number of slots in room
flagAttr	Initial values of room flag attributes Specify the OR of `SCE_NP_MATCHING2_ROOM_FLAG_ATTR_*`.
roomBinAttrInternal	Pointer to array of internal room binary attributes Initial values of the internal room binary attributes. Specify `SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID` for the *id* member of `SceNpMatching2BinAttr` to be specified here.

roomBinAttrInternalNum	Number of elements in array of internal room binary attributes
roomSearchableIntAttrExternal	Pointer to array of external room search integer attributes Initial values of the external room search integer attributes. Specify `SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID` for the *id* member of `SceNpMatching2IntAttr` to be specified here.
roomSearchableIntAttrExternalNum	Number of elements in array of external room search integer attributes
roomSearchableBinAttrExternal	Pointer to array of external room search binary attributes Initial values of the external room search binary attributes. Specify `SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID` for the *id* member of `SceNpMatching2BinAttr` to be specified here.
roomSearchableBinAttrExternalNum	Number of elements in array of external room search binary attributes
roomBinAttrExternal	Pointer to array of external room binary attributes Initial values of the external room binary attributes. Specify `SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID` for the *id* member of `SceNpMatching2BinAttr` to be specified here.
roomBinAttrExternalNum	Number of elements in array of external room binary attributes
roomPassword	Room password. Specify NULL if a room password will not be set.
groupConfig	Pointer to array of group settings. Specify NULL if groups will not be set in the room.
groupConfigNum	Number of elements in array of group settings.
passwordSlotMask	Room password slot mask. Specify NULL if the room is organized into groups or if a password will not be set.
allowedUser	Pointer to array of users who can join the room without a password
allowedUserNum	Number of elements in array of users who can join the room without a password
blockedUser	Pointer to array of users not allowed to join the room
blockedUserNum	Number of elements in array of users not allowed to join the room
joinRoomGroupLabel	Group label required to join a group. If the room is organized into groups, specify the group label set for the group including slot number 1.
roomMemberBinAttrInternal	Pointer to array of internal room member binary attributes Initial values of the internal room member binary attributes. Specify `SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID` for the *id* member of `SceNpMatching2BinAttr` to be specified here.
roomMemberBinAttrInternalNum	Number of elements in array of internal room member binary attributes

teamId	Initial team ID upon joining a room
padding2	Padding
sigOptParam	Signaling option parameter
padding3	Padding

Description

This structure represents the parameters of a request to create and join a room.

It is specified as an argument of `sceNpMatching2CreateJoinRoom()`.

See Also

`sceNpMatching2CreateJoinRoom()`

`SceNpMatching2CreateJoinRoomResponse`

SceNpMatching2CreateJoinRoomResponse

Create-and-join room request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2CreateJoinRoomResponse {
        SceNpMatching2RoomDataInternal *roomDataInternal;
} SceNpMatching2CreateJoinRoomResponse;
```

Members

roomDataInternal Pointer to internal room data of the new room

Description

This structure represents the response data to a request to create and join a room.

See Also

sceNpMatching2CreateJoinRoom()
SceNpMatching2CreateJoinRoomRequest

SceNpMatching2JoinRoomRequest

Join room request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2JoinRoomRequest {
        SceNpMatching2RoomId roomId;
        const SceNpMatching2SessionPassword *roomPassword;
        const SceNpMatching2GroupLabel *joinRoomGroupLabel;
        const SceNpMatching2BinAttr *roomMemberBinAttrInternal;
        SceUInt32 roomMemberBinAttrInternalNum;
        SceNpMatching2PresenceOptionData optData;
        SceNpMatching2TeamId teamId;
        SceUChar8 padding[3];
        const SceNpId *blockedUser;
        SceUInt32 blockedUserNum;
} SceNpMatching2JoinRoomRequest;
```

Members

roomId	ID of room to join
roomPassword	Room password required for joining. Specify NULL if a password is not necessary.
joinRoomGroupLabel	Group label required to join a group
roomMemberBinAttrInternal	Pointer to array of internal room member binary attributes Initial values of internal room member binary attributes. Specify `SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID` for the *id* member of `SceNpMatching2BinAttr` to be specified here.
roomMemberBinAttrInternalNum	Number of elements in array of internal room member binary attributes
optData	Optional data. Appended to notification to other room members upon joining the room
teamId	Initial team ID upon joining a room
padding	Padding
blockedUser	Users to be added to the room's block list Applied only when `SCE_NP_MATCHING2_ROOM_FLAG_ATTR_PROHIBITIVE_MODE` is valid
blockedUserNum	Number of users to be added to the room's block list Applied only when `SCE_NP_MATCHING2_ROOM_FLAG_ATTR_PROHIBITIVE_MODE` is valid

Description

This structure represents the parameters of a request to join a room.

It is specified as an argument of `sceNpMatching2JoinRoom()`.

See Also

`sceNpMatching2JoinRoom()`
`SceNpMatching2JoinRoomResponse`

SceNpMatching2JoinRoomResponse

Join room request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2JoinRoomResponse {
        SceNpMatching2RoomDataInternal *roomDataInternal;
} SceNpMatching2JoinRoomResponse;
```

Members

roomDataInternal Pointer to internal room data of room joined

Description

This structure represents the response data to a request to join a room.

See Also

sceNpMatching2JoinRoom()
SceNpMatching2JoinRoomRequest

SceNpMatching2LeaveRoomRequest

Leave room request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2LeaveRoomRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2PresenceOptionData optData;
        SceUChar8 padding[4];
} SceNpMatching2LeaveRoomRequest;
```

Members

roomId	ID of room to leave
optData	Optional data. Appended to notification to other room members upon leaving the room
padding	Padding

Description

This structure represents the parameters of a request to leave a room.

It is specified as an argument of `sceNpMatching2LeaveRoom()`.

See Also

`sceNpMatching2LeaveRoom()`

SceNpMatching2GrantRoomOwnerRequest

Room ownership grant request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2GrantRoomOwnerRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2RoomMemberId newOwner;
        SceUChar8 padding[2];
        SceNpMatching2PresenceOptionData optData;
} SceNpMatching2GrantRoomOwnerRequest;
```

Members

roomId	ID of room with room ownership
newOwner	ID of room member to grant room ownership
padding	Padding
optData	Optional data. Appended to notification to other room members upon the change in room ownership

Description

This structure represents the parameters of a request to grant room ownership.

It is specified as an argument of `sceNpMatching2GrantRoomOwner()`.

See Also

`sceNpMatching2GrantRoomOwner()`

SceNpMatching2KickoutRoomMemberRequest

Kickout request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2KickoutRoomMemberRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2RoomMemberId target;
        SceNpMatching2BlockKickFlag blockKickFlag;
        SceUChar8 padding[1];
        SceNpMatching2PresenceOptionData optData;
} SceNpMatching2KickoutRoomMemberRequest;
```

Members

roomId	ID of current room
target	ID of room member to kick out
blockKickFlag	Setting regarding rejoining
padding	Padding
optData	Optional data. Appended to notification to other room members upon leaving the room

Description

This structure represents the parameters of a request to kick out a room member.

It is specified as an argument of `sceNpMatching2KickoutRoomMember()`.

See Also

`sceNpMatching2KickoutRoomMember()`

SceNpMatching2SearchRoomRequest

Room search parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2SearchRoomRequest {
        int option;
        SceNpMatching2WorldId worldId;
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2RangeFilter rangeFilter;
        SceNpMatching2FlagAttr flagFilter;
        SceNpMatching2FlagAttr flagAttr;
        const SceNpMatching2IntSearchFilter *intFilter;
        SceUInt32 intFilterNum;
        const SceNpMatching2BinSearchFilter *binFilter;
        SceUInt32 binFilterNum;
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
} SceNpMatching2SearchRoomRequest;
```

Members

option	Room search options Specify SCE_NP_MATCHING2_SEARCH_ROOM_OPTION_*.
worldId	ID of world to search for room
lobbyId	ID of lobby to search for room
rangeFilter	Room list range filter
flagFilter	Flag attributes to use as search conditions Specify SCE_NP_MATCHING2_ROOM_FLAG_ATTR_*.
flagAttr	Flag attribute values to use as search conditions When making it a condition to have a flag set, specify the target SCE_NP_MATCHING2_ROOM_FLAG_ATTR_*.
intFilter	Pointer to array of integer attribute search conditions Specify SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID to *intFilter*->*attr.id*.
intFilterNum	Number of elements in array of integer attribute search conditions
binFilter	Pointer to array of binary attribute search conditions
binFilterNum	Number of elements in array of binary attribute search conditions
attrId	Pointer to array of room attribute IDs to obtain with the list of rooms
attrIdNum	Number of elements in array of room attribute IDs to obtain with the list of rooms For the attribute IDs, specify SCE_NP_MATCHING2_ROOM_SEARCHABLE_INT_ATTR_EXTERNAL_*_ID, SCE_NP_MATCHING2_ROOM_SEARCHABLE_BIN_ATTR_EXTERNAL_*_ID, SCE_NP_MATCHING2_ROOM_BIN_ATTR_EXTERNAL_*_ID.

Description

This structure represents the parameters of a room search request.

It is specified as an argument of sceNpMatching2SearchRoom().

See Also

`sceNpMatching2SearchRoom()`
`SceNpMatching2SearchRoomResponse`

SceNpMatching2SearchRoomResponse

Room search response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2SearchRoomResponse {
        SceNpMatching2Range range;
        SceNpMatching2RoomDataExternal *roomDataExternal;
} SceNpMatching2SearchRoomResponse;
```

Members

range	Range obtained in the room search
roomDataExternal	Pointer to start of the room list obtained

Description

This structure represents the response data to a room search request.

See Also

sceNpMatching2SearchRoom()
SceNpMatching2SearchRoomRequest

SceNpMatching2SendRoomMessageRequest

Room message send request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2SendRoomMessageRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2CastType castType;
        SceUChar8 padding[3];
        SceNpMatching2RoomMessageDestination dst;
        const void *msg;
        SceSize msgLen;
        int option;
} SceNpMatching2SendRoomMessageRequest;
```

Members

roomId	ID of current room
castType	Message cast type Specify SCE_NP_MATCHING2_CASTTYPE_*.
padding	Padding
dst	Recipients
msg	Message data
msgLen	Length of message data
option	Send options Specify SCE_NP_MATCHING2_SEND_MSG_OPTION_*.

Description

This structure represents the parameters of a request to send a room message.

It is specified as an argument of sceNpMatching2SendRoomMessage().

See Also

sceNpMatching2SendRoomMessage()

SceNpMatching2SendRoomChatMessageRequest

Room chat message send request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2SendRoomChatMessageRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2CastType castType;
        SceUChar8 padding[3];
        SceNpMatching2RoomMessageDestination dst;
        const void *msg;
        SceSize msgLen;
        int option;
} SceNpMatching2SendRoomChatMessageRequest;
```

Members

roomId	ID of current room
castType	Message cast type Specify SCE_NP_MATCHING2_CASTTYPE_*.
padding	Padding
dst	Recipients
msg	Message data (UTF-8)
msgLen	Length of message data
option	Send options Specify SCE_NP_MATCHING2_SEND_MSG_OPTION_*.

Description

This structure represents the parameters of a request to send a room chat message.

It is specified as an argument of sceNpMatching2SendRoomChatMessage().

See Also

sceNpMatching2SendRoomChatMessage()

SceNpMatching2SendRoomChatMessageResponse

SceNpMatching2SendRoomChatMessageResponse

Room chat message send request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2SendRoomChatMessageResponse {
        bool filtered;
} SceNpMatching2SendRoomChatMessageResponse;
```

Members

filtered Flag indicating whether or not the message had content requiring the vulgarity filter

Description

This structure represents the response data to a request to send a room chat message.

See Also

sceNpMatching2SendRoomChatMessage()
SceNpMatching2SendRoomChatMessageRequest

SceNpMatching2SetRoomDataInternalRequest

Internal room data configuration request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2SetRoomDataInternalRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2FlagAttr flagFilter;
        SceNpMatching2FlagAttr flagAttr;
        const SceNpMatching2BinAttr *roomBinAttrInternal;
        SceUInt32 roomBinAttrInternalNum;
        const SceNpMatching2RoomGroupPasswordConfig *passwordConfig;
        SceUInt32 passwordConfigNum;
        const SceNpMatching2RoomPasswordSlotMask *passwordSlotMask;
        const SceNpMatching2RoomMemberId *ownerPrivilegeRank;
        SceUInt32 ownerPrivilegeRankNum;
        SceUChar8 padding[4];
} SceNpMatching2SetRoomDataInternalRequest;
```

Members

roomId	ID of current room
flagFilter	Specify the room flag attribute SCE_NP_MATCHING2_ROOM_FLAG_ATTR_* to which value is to be set. When setting values simultaneously to multiple room flag attributes, specify by OR.
flagAttr	Value to set to the room flag attribute specified in *flagFilter*. Specify SCE_NP_MATCHING2_ROOM_FLAG_ATTR_*, which represents the room flag attribute for the flag to be enabled. When not enabling a flag, make no specification. When simultaneously enabling multiple room flag attributes, specify by OR.
roomBinAttrInternal	Pointer to array of target internal room binary attributes Specify SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID for the *id* member of SceNpMatching2BinAttr to be specified.
roomBinAttrInternalNum	Number of elements in array of target internal room binary attributes
passwordConfig	Pointer to array of group passwords. Specified when setting passwords to groups.
passwordConfigNum	Number of elements in array of group passwords
passwordSlotMask	Room password slot mask. Specified for setting passwords to rooms not organized into groups.
ownerPrivilegeRank	Array indicating priority of room members in receiving automatic room ownership. Room member IDs are sorted in order of priority (from highest to lowest).
ownerPrivilegeRankNum	Number of elements in array indicating priority of room members in receiving automatic room ownership
padding	Padding

Description

This structure represents the parameters of a request to configure internal room data.
It is specified as an argument of `sceNpMatching2SetRoomDataInternal()`.

See Also

`sceNpMatching2SetRoomDataInternal()`

SceNpMatching2GetRoomDataInternalRequest

Internal room data request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomDataInternalRequest {
        SceNpMatching2RoomId roomId;
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
} SceNpMatching2GetRoomDataInternalRequest;
```

Members

roomId	ID of current room
attrId	Pointer to array of IDs of internal room attributes to get For the attribute ID, specify SCE_NP_MATCHING2_ROOM_BIN_ATTR_INTERNAL_*_ID.
attrIdNum	Number of elements in array of IDs of internal room attributes to get

Description

This structure represents the parameters of a request for internal room data.

It is specified as an argument of sceNpMatching2GetRoomDataInternal().

See Also

sceNpMatching2GetRoomDataInternal()

SceNpMatching2GetRoomDataInternalResponse

SceNpMatching2GetRoomDataInternalResponse

Internal room data request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomDataInternalResponse {
        SceNpMatching2RoomDataInternal *roomDataInternal;
} SceNpMatching2GetRoomDataInternalResponse;
```

Members

roomDataInternal Pointer to internal room data

Description

This structure represents the response data to a request for internal room data.

See Also

sceNpMatching2GetRoomDataInternal()
SceNpMatching2GetRoomDataInternalRequest

SceNpMatching2SetRoomMemberDataInternalRequest

Internal room member data configuration request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2SetRoomMemberDataInternalRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2RoomMemberId memberId;
        SceNpMatching2TeamId teamId;
        SceUChar8 padding[5];
        SceNpMatching2FlagAttr flagFilter;
        SceNpMatching2FlagAttr flagAttr;
        const SceNpMatching2BinAttr *roomMemberBinAttrInternal;
        SceUInt32 roomMemberBinAttrInternalNum;
} SceNpMatching2SetRoomMemberDataInternalRequest;
```

Members

roomId	ID of current room
memberId	ID of room member to set the internal room member data
teamId	Value of team ID. Specify 0 if a team ID will not be set.
padding	Padding
flagFilter	Room member flag attributes to set (unused)
flagAttr	Values of room member flag attributes to set (unused)
roomMemberBinAttrInternal	Pointer to array of internal room member binary attributes to set Specify SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID for the *id* member of SceNpMatching2BinAttr to be specified here.
roomMemberBinAttrInternalNum	Number of elements in array of internal room member binary attributes to set

Description

This structure represents the parameters of a request to configure internal room member data.

It is specified as an argument of sceNpMatching2SetRoomMemberDataInternal().

See Also

sceNpMatching2SetRoomMemberDataInternal()

SceNpMatching2GetRoomMemberDataInternalRequest

Internal room member data request parameters

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomMemberDataInternalRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2RoomMemberId memberId;
        SceUChar8 padding[6];
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
} SceNpMatching2GetRoomMemberDataInternalRequest;
```

Members

roomId	ID of current room
memberId	ID of room member to get the internal room member data
padding	Padding
attrId	Pointer to array of IDs of internal room member attributes to get Specify `SCE_NP_MATCHING2_ROOMMEMBER_BIN_ATTR_INTERNAL_*_ID` for the attribute ID.
attrIdNum	Number of elements in array of IDs of internal room member attributes to get

Description

This structure represents the parameters of a request for internal room member data.

It is specified as an argument of `sceNpMatching2GetRoomMemberDataInternal()`.

See Also

`sceNpMatching2GetRoomMemberDataInternal()`

`SceNpMatching2GetRoomMemberDataInternalResponse`

SceNpMatching2GetRoomMemberDataInternalResponse

Internal room member data request response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetRoomMemberDataInternalResponse {
        SceNpMatching2RoomMemberDataInternal *roomMemberDataInternal;
} SceNpMatching2GetRoomMemberDataInternalResponse;
```

Members

roomMemberDataInternal Pointer to internal room member data

Description

This structure represents the response data to a request for internal room member data.

See Also

sceNpMatching2GetRoomMemberDataInternal()
SceNpMatching2GetRoomMemberDataInternalRequest

SceNpMatching2SetSignalingOptParamRequest

Signaling option parameter setting request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2SetSignalingOptParamRequest {
        SceNpMatching2RoomId roomId;
        SceNpMatching2SignalingOptParam sigOptParam;
} SceNpMatching2SetSignalingOptParamRequest;
```

Members

roomId	Room ID of room for which the signaling option parameter is to be set
sigOptParam	Signaling option parameter

Description

This structure represents the signaling option parameter setting request parameter.

It is specified for an argument of `sceNpMatching2SetSignalingOptParam()`.

See Also

`sceNpMatching2SetSignalingOptParam()`

SceNpMatching2GetLobbyInfoListRequest

Lobby information list acquisition request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetLobbyInfoListRequest {
        SceNpMatching2WorldId worldId;
        SceNpMatching2RangeFilter rangeFilter;
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
} SceNpMatching2GetLobbyInfoListRequest;
```

Members

worldId	World ID of world for which lobby information list is to be obtained
rangeFilter	Lobby information list acquisition range filter
attrId	Pointer to attribute ID array of lobby attributes to be obtained (unused)
attrIdNum	Number of elements in attribute ID array of lobby attributes to be obtained (unused)

Description

This structure represents the lobby information list acquisition request parameter.

It is specified for an argument of `sceNpMatching2GetLobbyInfoList()`.

See Also

`sceNpMatching2GetLobbyInfoList()`
`SceNpMatching2GetLobbyInfoListResponse`

SceNpMatching2GetLobbyInfoListResponse

Lobby information list acquisition response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetLobbyInfoListResponse {
        SceNpMatching2Range range;
        SceNpMatching2LobbyDataExternal *lobbyDataExternal;
} SceNpMatching2GetLobbyInfoListResponse;
```

Members

range	Range for which lobby information was obtained
lobbyDataExternal	Pointer to the beginning of the lobby information list that was obtained

Description

This structure represents response data for the lobby information list acquisition request.

See Also

sceNpMatching2GetLobbyInfoList()
SceNpMatching2GetLobbyInfoListRequest

SceNpMatching2JoinLobbyRequest

Lobby joining request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2JoinLobbyRequest {
        SceNpMatching2LobbyId lobbyId;
        const SceNpMatching2JoinedSessionInfo *joinedSessionInfo;
        SceUInt32 joinedSessionInfoNum;
        const SceNpMatching2BinAttr *lobbyMemberBinAttrInternal;
        SceUInt32 lobbyMemberBinAttrInternalNum;
        SceNpMatching2PresenceOptionData optData;
        SceUChar8 padding[4];
} SceNpMatching2JoinLobbyRequest;
```

Members

lobbyId	Lobby ID of lobby to be joined
joinedSessionInfo	Pointer to joined session information array
joinedSessionInfoNum	Number of elements in joined session information array
lobbyMemberBinAttrInternal	Pointer to lobby member internal binary attribute array Initial values of the internal lobby member binary attributes. Specify `SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID` for the *id* member of `SceNpMatching2BinAttr` to be specified here.
lobbyMemberBinAttrInternalNum	Number of elements in lobby member internal binary attribute array
optData	Option data This is added to join notification sent to other lobby members.
padding	Padding

Description

This structure represents a lobby joining request parameter.

It is specified for an argument of `sceNpMatching2JoinLobby()`.

See Also

`sceNpMatching2JoinLobby()`

`SceNpMatching2JoinLobbyResponse`

SceNpMatching2JoinLobbyResponse

Lobby joining response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2JoinLobbyResponse {
        SceNpMatching2LobbyDataInternal *lobbyDataInternal;
} SceNpMatching2JoinLobbyResponse;
```

Members

lobbyDataInternal Pointer to lobby-internal lobby information of joined lobby

Description

This structure represents response data for a lobby joining request.

See Also

sceNpMatching2JoinLobby()
SceNpMatching2JoinLobbyRequest

SceNpMatching2LeaveLobbyRequest

Lobby leaving request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2LeaveLobbyRequest {
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2PresenceOptionData optData;
        SceUChar8 padding[4];
} SceNpMatching2LeaveLobbyRequest;
```

Members

lobbyId	Lobby ID of lobby to be left
optData	Option data This is added to leave notification sent to other lobby members.
padding	Padding

Description

This structure represents a lobby leaving request parameter.

It is specified for an argument of `sceNpMatching2LeaveLobby()`.

See Also

`sceNpMatching2LeaveLobby()`

SceNpMatching2SendLobbyChatMessageRequest

Lobby chat message sending request parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2SendLobbyChatMessageRequest {
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2CastType castType;
        SceUChar8 padding[3];
        SceNpMatching2LobbyMessageDestination dst;
        const void *msg;
        SceSize msgLen;
        int option;
} SceNpMatching2SendLobbyChatMessageRequest;
```

Members

lobbyId	Lobby ID of joined lobby
castType	Message transmission type Specify SCE_NP_MATCHING2_CASTTYPE_*. SCE_NP_MATCHING2_CASTTYPE_MULTICAST_TEAM cannot be specified.
padding	Padding
dst	Message transmission destination
msg	Message data (UTF-8)
msgLen	Message data length
option	Message sending option Specify SCE_NP_MATCHING2_SEND_MSG_OPTION_*.

Description

This structure represents a lobby chat message sending request parameter.

It is specified for an argument of sceNpMatching2SendLobbyChatMessage().

See Also

sceNpMatching2SendLobbyChatMessage()

SceNpMatching2SendLobbyChatMessageResponse

SceNpMatching2SendLobbyChatMessageResponse

Lobby chat message sending response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2SendLobbyChatMessageResponse {
        bool filtered;
} SceNpMatching2SendLobbyChatMessageResponse;
```

Members

filtered Flag indicating whether message was filtered by vulgarity filter

Description

This structure represents response data for a lobby chat message sending request.

See Also

sceNpMatching2SendLobbyChatMessage()
SceNpMatching2SendLobbyChatMessageRequest

SceNpMatching2SetLobbyMemberDataInternalRequest

Lobby-internal lobby member information setting request parameter (not implemented)

Definition

```
#include <np.h>

typedef struct SceNpMatching2SetLobbyMemberDataInternalRequest {
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2LobbyMemberId memberId;
        SceUChar8 padding1[2];
        SceNpMatching2FlagAttr flagFilter;
        SceNpMatching2FlagAttr flagAttr;
        const SceNpMatching2JoinedSessionInfo *joinedSessionInfo;
        SceUInt32 joinedSessionInfoNum;
        const SceNpMatching2BinAttr *lobbyMemberBinAttrInternal;
        SceUInt32 lobbyMemberBinAttrInternalNum;
        SceUChar8 padding2[4];
} SceNpMatching2SetLobbyMemberDataInternalRequest;
```

Members

lobbyId	Lobby ID of joined lobby
memberId	Lobby member ID of member for which lobby-internal lobby member information is to be set
padding1	Padding
flagFilter	Lobby member flag attribute to be set (unused)
flagAttr	Value of lobby member flag attribute to be set (unused)
joinedSessionInfo	Pointer to joined session information array to be set
joinedSessionInfoNum	Number of elements in joined session information array to be set
lobbyMemberBinAttrInternal	Pointer to lobby member internal binary attribute array to be set Specify `SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID` for the *id* member of `SceNpMatching2BinAttr` to be specified here.
lobbyMemberBinAttrInternalNum	Number of elements in lobby member internal binary attribute array to be set
padding2	Padding

Description

This structure is not implemented.

This structure represents a lobby-internal lobby member information setting request parameter.

It is specified for an argument of `sceNpMatching2SetLobbyMemberDataInternal()`.

See Also

`sceNpMatching2SetLobbyMemberDataInternal()`

SceNpMatching2GetLobbyMemberDataInternalRequest

Lobby-internal lobby member information acquisition request parameter (not implemented)

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetLobbyMemberDataInternalRequest {
        SceNpMatching2LobbyId lobbyId;
        const SceNpMatching2LobbyMemberId memberId;
        SceUChar8 padding[6];
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
} SceNpMatching2GetLobbyMemberDataInternalRequest;
```

Members

lobbyId	Lobby ID of joined lobby
memberId	Lobby member ID of member for which lobby-internal lobby member information is to be obtained
padding	Padding
attrId	Pointer to attribute ID array of lobby member internal attributes to be obtained Specify `SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID` for the attribute ID to be specified here.
attrIdNum	Number of elements in attribute ID array of lobby member internal attributes to be obtained

Description

This structure is not implemented.

This structure represents a lobby-internal lobby member information acquisition request parameter.

It is specified for an argument of `sceNpMatching2GetLobbyMemberDataInternal()`.

See Also

`sceNpMatching2GetLobbyMemberDataInternal()`

`SceNpMatching2GetLobbyMemberDataInternalResponse`

SceNpMatching2GetLobbyMemberDataInternalResponse

Lobby-internal lobby member information acquisition response data

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetLobbyMemberDataInternalResponse {
        SceNpMatching2LobbyMemberDataInternal *lobbyMemberDataInternal;
} SceNpMatching2GetLobbyMemberDataInternalResponse;
```

Members

lobbyMemberDataInternal Pointer to lobby-internal lobby member information

Description

This structure represents response data for a lobby-internal lobby member information acquisition request.

See Also

sceNpMatching2GetLobbyMemberDataInternal()
SceNpMatching2GetLobbyMemberDataInternalRequest

SceNpMatching2GetLobbyMemberDataInternalListRequest

Request parameters for obtaining a list of lobby-internal lobby member information (not implemented)

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetLobbyMemberDataInternalListRequest {
        SceNpMatching2LobbyId lobbyId;
        SceNpMatching2LobbyMemberId *memberId;
        SceUInt32 memberIdNum;
        const SceNpMatching2AttributeId *attrId;
        SceUInt32 attrIdNum;
        bool extendedData;
        SceUChar8 padding[7];
} SceNpMatching2GetLobbyMemberDataInternalListRequest;
```

Members

lobbyId	Lobby ID of the joined-in lobby
memberId	Array of lobby member IDs for which lobby-internal lobby member information are to be obtained in a list
memberIdNum	Number of members for which lobby-internal lobby member information are to be obtained in a list
attrId	Pointer to the array of attribute IDs corresponding to the internal lobby member attributes to obtain Specify `SCE_NP_MATCHING2_LOBBYMEMBER_BIN_ATTR_INTERNAL_*_ID` for the attribute IDs.
attrIdNum	Number of attribute IDs for which corresponding internal lobby member attributes are to be obtained in the array
extendedData	Flag indicating whether or not to obtain extension data
padding	Padding

Description

This structure is not implemented.

This structure represents request parameters for obtaining a list of lobby-internal lobby member information.

It is specified as an argument of `sceNpMatching2GetLobbyMemberDataInternalList()`.

See Also

`sceNpMatching2GetLobbyMemberDataInternalList()`

`SceNpMatching2GetLobbyMemberDataInternalListResponse`

SceNpMatching2GetLobbyMemberDataInternalListResponse

Response data for obtaining a list of lobby-internal lobby member information

Definition

```
#include <np.h>

typedef struct SceNpMatching2GetLobbyMemberDataInternalListResponse {
        SceNpMatching2LobbyMemberDataInternal *lobbyMemberDataInternal;
        SceUInt32 lobbyMemberDataInternalNum;
} SceNpMatching2GetLobbyMemberDataInternalListResponse;
```

Members

lobbyMemberDataInternal	Pointer to lobby-internal lobby member information
lobbyMemberDataInternalNum	Number of lobby-internal lobby member information

Description

This structure represents response data for a request to obtain a list of lobby-internal lobby member information.

See Also

sceNpMatching2GetLobbyMemberDataInternalList()
SceNpMatching2GetLobbyMemberDataInternalListRequest

SceNpMatching2SignalingGetPingInfoRequest

Request parameters for obtaining Ping information

Definition

```
#include <np.h>

typedef struct SceNpMatching2SignalingGetPingInfoRequest {
        SceNpMatching2RoomId roomId;
        SceUChar8 reserved[16];
} SceNpMatching2SignalingGetPingInfoRequest;
```

Members

roomId	Room ID of the room to obtain Ping information
reserved	Reserved area

Description

This structure represents request parameters for obtaining Ping information.

Specify this structure as an argument in `sceNpMatching2SignalingGetPingInfo()`.

See Also

`sceNpMatching2SignalingGetPingInfo()`
`SceNpMatching2SignalingGetPingInfoResponse`

SceNpMatching2SignalingGetPingInfoResponse

Response data for obtaining Ping information

Definition

```
#include <np.h>

typedef struct SceNpMatching2SignalingGetPingInfoResponse {
        SceNpMatching2ServerId serverId;
        SceUChar8 padding1[2];
        SceNpMatching2WorldId worldId;
        SceNpMatching2RoomId roomId;
        SceUInt32 rtt;
        SceUChar8 reserved[20];
} SceNpMatching2SignalingGetPingInfoResponse;
```

Members

serverId	Server ID of the server where the room from which Ping information was obtained belongs
padding1	Padding
worldId	World ID of the world where the room from which Ping information was obtained belongs
roomId	Room ID of the room from which Ping information was obtained
rtt	Obtained Ping information Indicates the RTT (in microseconds) between the room owner and the client.
reserved	Reserved area

Description

This structure represents response data for a Ping information obtainment request.

See Also

sceNpMatching2SignalingGetPingInfo()
SceNpMatching2SignalingGetPingInfoRequest

Session Event and Message Structures

SceNpMatching2RoomMemberUpdateInfo

Room member update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomMemberUpdateInfo {
        SceNpMatching2RoomMemberDataInternal *roomMemberDataInternal;
        SceNpMatching2EventCause eventCause;
        SceUChar8 padding[3];
        SceNpMatching2PresenceOptionData optData;
} SceNpMatching2RoomMemberUpdateInfo;
```

Members

roomMemberDataInternal	Internal room member data of room member who joined or left
eventCause	Event cause SCE_NP_MATCHING2_EVENT_CAUSE_* will be stored.
padding	Padding
optData	Optional data

Description

This structure represents the update information notified to room members when a room member joins or leaves the room.

See Also

SCE_NP_MATCHING2_ROOM_EVENT_MEMBER_JOINED
SCE_NP_MATCHING2_ROOM_EVENT_MEMBER_LEFT

SceNpMatching2RoomOwnerUpdateInfo

Room owner update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomOwnerUpdateInfo {
        SceNpMatching2RoomMemberId prevOwner;
        SceNpMatching2RoomMemberId newOwner;
        SceNpMatching2EventCause eventCause;
        SceUChar8 padding[3];
        SceNpMatching2SessionPassword *roomPassword;
        SceNpMatching2PresenceOptionData optData;
} SceNpMatching2RoomOwnerUpdateInfo;
```

Members

prevOwner	Room member ID of old owner
newOwner	Room member ID of new owner
eventCause	Event cause SCE_NP_MATCHING2_EVENT_CAUSE_* will be stored.
padding	Padding
roomPassword	Room password set to the room. NULL if there is no room password.
optData	Optional data

Description

This structure represents the update information notified to room members when the room owner changes.

See Also

SCE_NP_MATCHING2_ROOM_EVENT_ROOM_OWNER_CHANGED

SceNpMatching2RoomUpdateInfo

Room update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomUpdateInfo {
        SceNpMatching2EventCause eventCause;
        SceUChar8 padding[3];
        int errorCode;
        SceNpMatching2PresenceOptionData optData;
} SceNpMatching2RoomUpdateInfo;
```

Members

eventCause	Event cause SCE_NP_MATCHING2_EVENT_CAUSE_* will be stored.
padding	Padding
errorCode	Error code
optData	Optional data

Description

This structure represents the room update information notified to room members when a member is kicked out of the room or the room is deleted.

See Also

SCE_NP_MATCHING2_ROOM_EVENT_KICKEDOUT
SCE_NP_MATCHING2_ROOM_EVENT_ROOM_DESTROYED

SceNpMatching2RoomDataInternalUpdateInfo

Internal room data update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomDataInternalUpdateInfo {
        SceNpMatching2RoomDataInternal *newRoomDataInternal;
        SceNpMatching2FlagAttr *newFlagAttr;
        SceNpMatching2FlagAttr *prevFlagAttr;
        SceNpMatching2RoomPasswordSlotMask *newRoomPasswordSlotMask;
        SceNpMatching2RoomPasswordSlotMask *prevRoomPasswordSlotMask;
        SceNpMatching2RoomGroup **newRoomGroup;
        SceUInt32 newRoomGroupNum;
        SceNpMatching2RoomBinAttrInternal **newRoomBinAttrInternal;
        SceUInt32 newRoomBinAttrInternalNum;
} SceNpMatching2RoomDataInternalUpdateInfo;
```

Members

newRoomDataInternal	Internal room data after the update
newFlagAttr	Pointer to *newRoomDataInternal*->*flagAttr* NULL if there is no update.
prevFlagAttr	Flag attributes before the update. NULL if there is no update.
newRoomPasswordSlotMask	Pointer to *newRoomDataInternal*->*passwordSlotMask*. NULL if there is no update.
prevRoomPasswordSlotMask	Room password slot mask before the update. NULL if there is no update.
newRoomGroup	Array of pointers to updated groups
newRoomGroupNum	Number of elements in array of pointers to updated groups
newRoomBinAttrInternal	Array of pointers to updated internal room binary attributes
newRoomBinAttrInternalNum	Number of elements in array of pointers to updated internal room binary attributes

Description

This structure represents the update information of internal room data.

See Also

SCE_NP_MATCHING2_ROOM_EVENT_UPDATED_ROOM_DATA_INTERNALL

SceNpMatching2RoomMemberDataInternalUpdateInfo

Internal room member data update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomMemberDataInternalUpdateInfo {
        SceNpMatching2RoomMemberDataInternal *newRoomMemberDataInternal;
        SceNpMatching2FlagAttr *newFlagAttr;
        SceNpMatching2FlagAttr *prevFlagAttr;
        SceNpMatching2TeamId *newTeamId;
        SceNpMatching2RoomMemberBinAttrInternal

**newRoomMemberBinAttrInternal;
        SceUInt32 newRoomMemberBinAttrInternalNum;
} SceNpMatching2RoomMemberDataInternalUpdateInfo;
```

Members

newRoomMemberDataInternal	Internal room member data after the update
newFlagAttr	Pointer to *newRoomMemberDataInternal*->*flagAttr* NULL if there is no update.
prevFlagAttr	Flag attributes before the update. NULL if there is no update.
newTeamId	Pointer to *newRoomMemberDataInternal*->*teamId* NULL if there is no update.
newRoomMemberBinAttrInternal	Pointer to array of updated internal room member binary attributes
newRoomMemberBinAttrInternalNum	Number of elements in array of updated internal room member binary attributes

Description

This structure represents the update information of internal room member data.

See Also

SCE_NP_MATCHING2_ROOM_EVENT_UPDATED_ROOM_MEMBER_DATA_INTERNAL

SceNpMatching2RoomMessageInfo

Room message information

Definition

```
#include <np.h>

typedef struct SceNpMatching2RoomMessageInfo {
        bool filtered;
        SceNpMatching2CastType castType;
        SceUChar8 padding[2];
        SceNpMatching2RoomMessageDestination *dst;
        SceNpId *srcMember;
        const void *msg;
        SceSize msgLen;
} SceNpMatching2RoomMessageInfo;
```

Members

filtered	Flag indicating whether or not the message had content requiring the vulgarity filter
castType	Message cast type SCE_NP_MATCHING2_CASTTYPE_* will be stored.
padding	Padding
dst	Recipient(s) Access the appropriate member depending on the value of *castType*.
srcMember	User information of sender
msg	Message data
msgLen	Length of message data

Description

This structure represents the room message information notified to a room member when a room chat message or room message is received.

See Also

SCE_NP_MATCHING2_ROOM_MSG_EVENT_CHAT_MESSAGE
SCE_NP_MATCHING2_ROOM_MSG_EVENT_MESSAGE

SceNpMatching2LobbyMemberUpdateInfo

Lobby member update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyMemberUpdateInfo {
        SceNpMatching2LobbyMemberDataInternal *lobbyMemberDataInternal;
        SceNpMatching2EventCause eventCause;
        SceUChar8 padding[3];
        SceNpMatching2PresenceOptionData optData;
} SceNpMatching2LobbyMemberUpdateInfo;
```

Members

lobbyMemberDataInternal	Lobby-internal lobby member information of lobby member that joined or left
eventCause	Cause of event SCE_NP_MATCHING2_EVENT_CAUSE_* will be stored.
padding	Padding
optData	Option data

Description

This array represents lobby member update information that is reported when a lobby member joined or left a lobby.

See Also

SCE_NP_MATCHING2_LOBBY_EVENT_MEMBER_JOINED
SCE_NP_MATCHING2_LOBBY_EVENT_MEMBER_LEFT

SceNpMatching2LobbyUpdateInfo

Lobby update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyUpdateInfo {
        SceNpMatching2EventCause eventCause;
        SceUChar8 padding[3];
        int errorCode;
} SceNpMatching2LobbyUpdateInfo;
```

Members

eventCause	Cause of event SCE_NP_MATCHING2_EVENT_CAUSE_* will be stored.
padding	Padding
errorCode	Error Codes

Description

This structure represents lobby update information that is reported when a lobby is destroyed.

See Also

SCE_NP_MATCHING2_LOBBY_EVENT_LOBBY_DESTROYED

SceNpMatching2LobbyMemberDataInternalUpdateInfo

Lobby-internal lobby member information update information

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyMemberDataInternalUpdateInfo {
        SceNpMatching2LobbyMemberId memberId;
        SceUChar8 padding[2];
        SceNpId npId;
        SceNpMatching2FlagAttr flagFilter;
        SceNpMatching2FlagAttr newFlagAttr;
        SceNpMatching2JoinedSessionInfo *newJoinedSessionInfo;
        SceUInt32 newJoinedSessionInfoNum;
        SceNpMatching2LobbyMemberBinAttrInternal

*newLobbyMemberBinAttrInternal;
        SceUInt32 newLobbyMemberBinAttrInternalNum;
} SceNpMatching2LobbyMemberDataInternalUpdateInfo;
```

Members

memberId	Lobby member ID of lobby member for which lobby-internal lobby member information was updated
padding	Padding
npId	NP ID of lobby member for which lobby-internal lobby member information was updated
flagFilter	Filter representing the flags that have been updated SCE_NP_MATCHING2_LOBBY_FLAG_ATTR_* representing the flags that have been updated will be stored. When multiple flags have been updated, an OR value will be stored.
newFlagAttr	Flag attributes after information was updated Represented by an OR of SCE_NP_MATCHING2_LOBBY_FLAG_ATTR_*.
newJoinedSessionInfo	Pointer to updated joined session information array
newJoinedSessionInfoNum	Number of elements in updated joined session information array
newLobbyMemberBinAttrInternal	Pointer to updated lobby member internal binary attribute array
newLobbyMemberBinAttrInternalNum	Number of elements in updated lobby member internal binary attribute array

Description

Structure representing lobby-internal lobby member information.

See Also

SCE_NP_MATCHING2_LOBBY_EVENT_UPDATED_LOBBY_MEMBER_DATA_INTERNAL

SceNpMatching2LobbyMessageInfo

Lobby message information

Definition

```
#include <np.h>

typedef struct SceNpMatching2LobbyMessageInfo {
        bool filtered;
        SceNpMatching2CastType castType;
        SceUChar8 padding[2];
        SceNpMatching2LobbyMessageDestination *dst;
        SceNpId *srcMember;
        const void *msg;
        SceSize msgLen;
} SceNpMatching2LobbyMessageInfo;
```

Members

filtered	Flag indicating whether message was filtered by vulgarity filter
castType	Message transmission type SCE_NP_MATCHING2_CASTTYPE_* will be stored.
padding	Padding
dst	Message transmission destination Access the appropriate member depending on the value of *castType*.
srcMember	User information of message sender
msg	Message data
msgLen	Message data length

Description

This structure represents lobby message information that is reported when a lobby chat message is received.

See Also

SCE_NP_MATCHING2_LOBBY_MSG_EVENT_CHAT_MESSAGE

SceNpMatching2SignalingOptParamUpdateInfo

Update information of the signaling option parameter

Definition

```
#include <np.h>

typedef struct SceNpMatching2SignalingOptParamUpdateInfo {
        SceNpMatching2SignalingOptParam newSignalingOptParam;
} SceNpMatching2SignalingOptParamUpdateInfo;
```

Members

newSignalingOptParam	Signaling option parameter after update

Description

This structure represents the update information of the signaling option parameter notified when the parameter is updated.

See Also

SCE_NP_MATCHING2_ROOM_EVENT_UPDATED_SIGNALING_OPT_PARAM

Signaling Structures

SceNpMatching2SignalingConnectionInfo

Connection data union

Definition

```
#include <np.h>

typedef union SceNpMatching2SignalingConnectionInfo {
        SceUInt32 rtt;
        SceUInt32 bandwidth;
        SceNpId npId;
        struct{
              SceNetInAddr addr;
              SceUShort16 port;
              SceUChar8 padding[2];
        } address;
        SceUInt32 packetLoss;
} SceNpMatching2SignalingConnectionInfo;
```

Members

rtt	Round Trip Time
bandwidth	Bandwidth (bytes/sec)
npId	NP ID
address	IP address and port number
packetLoss	Packet loss rate

Description

This data type represents connection data.

When data is obtained by using `sceNpMatching2SignalingGetConnectionInfo()`, the data is received by using a variable of this type.

See Also

`sceNpMatching2SignalingGetConnectionInfo()`

SceNpMatching2SignalingNetInfo

Network data structure

Definition

```
#include <np.h>

typedef union SceNpMatching2SignalingNetInfo {
        SceSize size;
        SceNetInAddr localAddr;
        SceNetInAddr mappedAddr;
        int natStatus;
} SceNpMatching2SignalingNetInfo;
```

Members

size	Structure size
localAddr	Local IP address
mappedAddr	External IP address
natStatus	NAT status type

Description

This data type represents network data.

When data is obtained by using `sceNpMatching2SignalingGetLocalNetInfo()` or `sceNpMatching2SignalingGetPeerNetInfoResult()`, the data is received by using a variable of this type.

For *size*, specify the size of this structure

See Also

`sceNpMatching2SignalingGetLocalNetInfo()`,
`sceNpMatching2SignalingGetPeerNetInfoResult()`

Library Structures

SceNpMatching2MemoryInfo

Memory information

Definition

```
#include <np.h>

typedef struct SceNpMatching2MemoryInfo {
        SceSize totalMemSize;
        SceSize curMemUsage;
        SceSize maxMemUsage;
        SceUChar8 reserved[12];
} SceNpMatching2MemoryInfo;
```

Members

totalMemSize	Size of the memory area (bytes)
curMemUsage	Currently used memory size (bytes)
maxMemUsage	Past maximum memory usage (bytes)
reserved	Reserved area

Description

This structure represents the memory information of the heap area for the NP Matching 2 library.
This structure is specified as an argument of `sceNpMatching2GetMemoryInfo()`.

See Also

`sceNpMatching2GetMemoryInfo()`

Error Codes

List of Error Codes

List of error codes returned by the NP Matching 2 library

Definition

Client Errors

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_OUT_OF_MEMORY	0x80550c01	Insufficient memory
SCE_NP_MATCHING2_ERROR_ALREADY_INITIALIZED	0x80550c02	Already initialized
SCE_NP_MATCHING2_ERROR_NOT_INITIALIZED	0x80550c03	Not initialized
SCE_NP_MATCHING2_ERROR_CONTEXT_MAX	0x80550c04	No more contexts can be created
SCE_NP_MATCHING2_ERROR_CONTEXT_ALREADY_EXISTS	0x80550c05	Context already exists
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_FOUND	0x80550c06	Context cannot be found
SCE_NP_MATCHING2_ERROR_CONTEXT_ALREADY_STARTED	0x80550c07	Context has already been started
SCE_NP_MATCHING2_ERROR_CONTEXT_NOT_STARTED	0x80550c08	Context has not yet been started
SCE_NP_MATCHING2_ERROR_SERVER_NOT_FOUND	0x80550c09	Server cannot be found
SCE_NP_MATCHING2_ERROR_INVALID_ARGUMENT	0x80550c0a	Invalid argument
SCE_NP_MATCHING2_ERROR_INVALID_CONTEXT_ID	0x80550c0b	Invalid context ID
SCE_NP_MATCHING2_ERROR_INVALID_SERVER_ID	0x80550c0c	Invalid server ID
SCE_NP_MATCHING2_ERROR_INVALID_WORLD_ID	0x80550c0d	Invalid world ID
SCE_NP_MATCHING2_ERROR_INVALID_LOBBY_ID	0x80550c0e	Invalid lobby ID
SCE_NP_MATCHING2_ERROR_INVALID_ROOM_ID	0x80550c0f	Invalid room ID
SCE_NP_MATCHING2_ERROR_INVALID_MEMBER_ID	0x80550c10	Invalid member ID
SCE_NP_MATCHING2_ERROR_INVALID_ATTRIBUTE_ID	0x80550c11	Invalid attribute ID
SCE_NP_MATCHING2_ERROR_INVALID_CASTTYPE	0x80550c12	Invalid message cast type
SCE_NP_MATCHING2_ERROR_INVALID_SORT_METHOD	0x80550c13	Invalid sort method
SCE_NP_MATCHING2_ERROR_INVALID_MAX_SLOT	0x80550c14	Invalid number of slots (total)
SCE_NP_MATCHING2_ERROR_INVALID_MATCHING_SPACE	0x80550c16	Invalid search space
SCE_NP_MATCHING2_ERROR_INVALID_BLOCK_KICK_FLAG	0x80550c18	Invalid setting for rejoin after kickout
SCE_NP_MATCHING2_ERROR_INVALID_MESSAGE_TARGET	0x80550c19	Invalid message target
SCE_NP_MATCHING2_ERROR_RANGE_FILTER_MAX	0x80550c1a	Larger than the maximum number of obtainable elements specified in the range filter

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_INSUFFICIENT_BUFFER	0x80550c1b	Buffer size is insufficient
SCE_NP_MATCHING2_ERROR_DESTINATION_DISAPPEARED	0x80550c1c	Destination of request disappeared
SCE_NP_MATCHING2_ERROR_REQUEST_TIMEOUT	0x80550c1d	Request timed out
SCE_NP_MATCHING2_ERROR_INVALID_ALIGNMENT	0x80550c1e	Invalid alignment
SCE_NP_MATCHING2_ERROR_CONNECTION_CLOSED_BY_SERVER	0x80550c22	Server closed the connection
SCE_NP_MATCHING2_ERROR_SSL_VERIFY_FAILED	0x80550c23	SSL verification failed
SCE_NP_MATCHING2_ERROR_SSL_HANDSHAKE	0x80550c24	SSL handshake error
SCE_NP_MATCHING2_ERROR_SSL_SEND	0x80550c25	SSL send error
SCE_NP_MATCHING2_ERROR_SSL_RECV	0x80550c26	SSL receive error
SCE_NP_MATCHING2_ERROR_JOINED_SESSION_MAX	0x80550c27	Exceeded the maximum number of sessions that can be joined
SCE_NP_MATCHING2_ERROR_ALREADY_JOINED	0x80550c28	Already joined session for which join request was issued
SCE_NP_MATCHING2_ERROR_INVALID_SESSION_TYPE	0x80550c29	Invalid session type
SCE_NP_MATCHING2_ERROR_NP_SIGNED_OUT	0x80550c2b	Signed out of NP
SCE_NP_MATCHING2_ERROR_BUSY	0x80550c2c	Processing cannot be accepted
SCE_NP_MATCHING2_ERROR_SERVER_NOT_AVAILABLE	0x80550c2d	The server status of the server to which the request was sent is unavailable Use another available server.
SCE_NP_MATCHING2_ERROR_NOT_ALLOWED	0x80550c2e	User does not have the right for executing requested processing
SCE_NP_MATCHING2_ERROR_ABORTED	0x80550c2f	Processing was aborted
SCE_NP_MATCHING2_ERROR_REQUEST_NOT_FOUND	0x80550c30	Request cannot be found
SCE_NP_MATCHING2_ERROR_SESSION_DESTROYED	0x80550c31	Session disappeared This error code is returned for a request if while processing the request, the target session disappeared.
SCE_NP_MATCHING2_ERROR_CONTEXT_STOPPED	0x80550c32	Context has been stopped Processing could not be continued because the context being used has been stopped. This error occurs, for example, when `sceNpMatching2ContextStop()` is executed while a request is being executed.
SCE_NP_MATCHING2_ERROR_INVALID_REQUEST_PARAMETER	0x80550c33	Invalid request parameter The values of the request parameter specified in the request function may be invalid. Check the specified values.

Value	(Number)	Description
SCE_NP_MATCHING2_ERROR_NOT_NP_SIGN_IN	0x80550c34	Not signed into the NP You must be signed into the NP in order to use the NP Matching 2 library. This error occurs, for example, when you call sceNpMatching2CreateContext() without signing into the NP first. Check whether or not you are signed into the NP.
SCE_NP_MATCHING2_ERROR_ROOM_NOT_FOUND	0x80550c35	Room cannot be found Check the room ID specified as a request parameter.
SCE_NP_MATCHING2_ERROR_ROOM_MEMBER_NOT_FOUND	0x80550c36	Room member cannot be found Check the room member ID specified as a request parameter.
SCE_NP_MATCHING2_ERROR_LOBBY_NOT_FOUND	0x80550c37	Lobby cannot be found Check the lobby ID specified as a request parameter.
SCE_NP_MATCHING2_ERROR_LOBBY_MEMBER_NOT_FOUND	0x80550c38	Lobby member cannot be found Check the lobby member ID specified as a request parameter.
SCE_NP_MATCHING2_ERROR_KEEPALIVE_TIMEOUT	0x80550c3a	Keep-alive timeout was detected. Keep-alive packet was not received from the server for a certain interval.
SCE_NP_MATCHING2_ERROR_TIMEOUT_TOO_SHORT	0x80550c3b	Time-out time value is small. Specified time-out time value is too small. Check the time-out time value.
SCE_NP_MATCHING2_ERROR_TIMEDOUT	0x80550c3c	Timed out
SCE_NP_MATCHING2_ERROR_INVALID_SLOTGROUP	0x80550c3d	Invalid group slot was specified.
SCE_NP_MATCHING2_ERROR_INVALID_ATTRIBUTE_SIZE	0x80550c3e	Size of the specified attribute is invalid
SCE_NP_MATCHING2_ERROR_CANNOT_ABORT	0x80550c3f	Cannot abort the request
SCE_NP_MATCHING2_ERROR_SESSION_NOT_FOUND	0x80550c40	Session cannot be found

Server Errors

Value	(Number)	Description
SCE_NP_MATCHING2_SERVER_ERROR_BAD_REQUEST	0x80550d01	Invalid request
SCE_NP_MATCHING2_SERVER_ERROR_SERVICE_UNAVAILABLE	0x80550d02	Services cannot be used due to maintenance
SCE_NP_MATCHING2_SERVER_ERROR_BUSY	0x80550d03	Busy due to server overload
SCE_NP_MATCHING2_SERVER_ERROR_END_OF_SERVICE	0x80550d04	Service already ended
SCE_NP_MATCHING2_SERVER_ERROR_INTERNAL_SERVER_ERROR	0x80550d05	Error in the server
SCE_NP_MATCHING2_SERVER_ERROR_PLAYER_BANNED	0x80550d06	Banned user
SCE_NP_MATCHING2_SERVER_ERROR_FORBIDDEN	0x80550d07	Forbidden operation
SCE_NP_MATCHING2_SERVER_ERROR_BLOCKED	0x80550d08	Blocked user
SCE_NP_MATCHING2_SERVER_ERROR_UNSUPPORTED_NP_ENV	0x80550d09	Unsupported NP environment
SCE_NP_MATCHING2_SERVER_ERROR_INVALID_TICKET	0x80550d0a	Invalid ticket
SCE_NP_MATCHING2_SERVER_ERROR_INVALID_SIGNATURE	0x80550d0b	Invalid signature

Value	(Number)	Description
SCE_NP_MATCHING2_SERVER_ERROR_EXPIRED_TICKET	0x80550d0c	Ticket has expired
SCE_NP_MATCHING2_SERVER_ERROR_ENTITLEMENT_REQUIRED	0x80550d0d	Entitlement ID is required for using the world
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_CONTEXT	0x80550d0e	Server context does not exist
SCE_NP_MATCHING2_SERVER_ERROR_CLOSED	0x80550d0f	Session was closed
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_TITLE	0x80550d10	Specified NP Communication ID does not exist
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_WORLD	0x80550d11	Specified world does not exist
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_LOBBY	0x80550d12	Specified lobby does not exist
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_ROOM	0x80550d13	Specified room does not exist
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_LOBBY_INSTANCE	0x80550d14	Specified lobby instance does not exist
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_ROOM_INSTANCE	0x80550d15	Specified room instance does not exist
SCE_NP_MATCHING2_SERVER_ERROR_PASSWORD_MISMATCH	0x80550d17	Session password does not match
SCE_NP_MATCHING2_SERVER_ERROR_LOBBY_FULL	0x80550d18	Lobby is full
SCE_NP_MATCHING2_SERVER_ERROR_ROOM_FULL	0x80550d19	Room is full
SCE_NP_MATCHING2_SERVER_ERROR_GROUP_FULL	0x80550d1b	Room group is full
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_USER	0x80550d1c	Specified user does not exist
SCE_NP_MATCHING2_SERVER_ERROR_TITLE_PASSPHRASE_MISMATCH	0x80550d1e	Title passphrase does not match
SCE_NP_MATCHING2_SERVER_ERROR_LOBBY_ALREADY_EXIST	0x80550d25	Specified lobby already exists
SCE_NP_MATCHING2_SERVER_ERROR_ROOM_ALREADY_EXIST	0x80550d26	Specified room already exists
SCE_NP_MATCHING2_SERVER_ERROR_NO_ROOMGROUP	0x80550d29	Join group label does not exist
SCE_NP_MATCHING2_SERVER_ERROR_NO_SUCH_GROUP	0x80550d2a	Specified room group does not exist
SCE_NP_MATCHING2_SERVER_ERROR_NO_PASSWORD	0x80550d2b	Room password slot mask is specified but there is no room password
SCE_NP_MATCHING2_SERVER_ERROR_INVALID_GROUP_SLOT_NUM	0x80550d2c	Specified number of slots in group exceeds the total number of slots
SCE_NP_MATCHING2_SERVER_ERROR_INVALID_PASSWORD_SLOT_MASK	0x80550d2d	Specified room password slot mask exceeds the total number of slots
SCE_NP_MATCHING2_SERVER_ERROR_DUPLICATE_GROUP_LABEL	0x80550d2e	Same group label is set multiple times
SCE_NP_MATCHING2_SERVER_ERROR_REQUEST_OVERFLOW	0x80550d2f	Room message flow sent from one client exceeded the limit Make sure that room messages sent by one client do not exceed 512 bytes/sec.
SCE_NP_MATCHING2_SERVER_ERROR_ALREADY_JOINED	0x80550d30	Same user already joined

Value	(Number)	Description
SCE_NP_MATCHING2_SERVER_ERROR_NAT_TYPE_MISMATCH	0x80550d31	Joining by users of the specified NAT type is restricted The NAT type room entry limitation flag has been set to the room flag attributes and a user with the applicable NAT type attempted to join.
SCE_NP_MATCHING2_SERVER_ERROR_ROOM_INCONSISTENCY	0x80550d32	Room information is inconsistent. Although extremely unlikely, it is possible for room information to be temporarily inconsistent, depending on when the room information is synchronized. If room information is obtained while in this state, this error can occur. This state of inconsistency usually exists for an extremely short time and recovery is quick.
SCE_NP_MATCHING2_SERVER_ERROR_BLOCKED_USER_IN_ROOM	0x80550d33	At least one of the members specified in the blocked user list is already a member of the room.

Signaling Errors

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_NOT_INITIALIZED	0x80550e01	Not initialized Execute `sceNpMatching2Init()` and initialize the NP Matching 2 library.
SCE_NP_MATCHING2_SIGNALING_ERROR_ALREADY_INITIALIZED	0x80550e02	Already initialized `sceNpMatching2Init()` may have already been called. Check the calling order.
SCE_NP_MATCHING2_SIGNALING_ERROR_OUT_OF_MEMORY	0x80550e03	Insufficient memory
SCE_NP_MATCHING2_SIGNALING_ERROR_CTXID_NOT_AVAILABLE	0x80550e04	Context ID could not be obtained
SCE_NP_MATCHING2_SIGNALING_ERROR_CTX_NOT_FOUND	0x80550e05	Specified context does not exist
SCE_NP_MATCHING2_SIGNALING_ERROR_REQID_NOT_AVAILABLE	0x80550e06	Request ID could not be obtained
SCE_NP_MATCHING2_SIGNALING_ERROR_REQ_NOT_FOUND	0x80550e07	Specified request ID does not exist
SCE_NP_MATCHING2_SIGNALING_ERROR_PARSER_CREATE_FAILED	0x80550e08	Failed to create protocol message
SCE_NP_MATCHING2_SIGNALING_ERROR_PARSER_FAILED	0x80550e09	Failed to parse protocol message
SCE_NP_MATCHING2_SIGNALING_ERROR_INVALID_NAMESPACE	0x80550e0a	Name space of protocol message is invalid
SCE_NP_MATCHING2_SIGNALING_ERROR_NETINFO_NOT_AVAILABLE	0x80550e0b	Network connection information could not be used
SCE_NP_MATCHING2_SIGNALING_ERROR_PEER_NOT_RESPONDING	0x80550e0c	No response from peer
SCE_NP_MATCHING2_SIGNALING_ERROR_CONNID_NOT_AVAILABLE	0x80550e0d	Connection ID could not be obtained
SCE_NP_MATCHING2_SIGNALING_ERROR_CONN_NOT_FOUND	0x80550e0e	Specified connection does not exist
SCE_NP_MATCHING2_SIGNALING_ERROR_PEER_UNREACHABLE	0x80550e0f	Peer could not be reached
SCE_NP_MATCHING2_SIGNALING_ERROR_TERMINATED_BY_PEER	0x80550e10	Became INACTIVE due to request from peer

Value	(Number)	Description
SCE_NP_MATCHING2_SIGNALING_ERROR_TIMEOUT	0x80550e11	Request timed out
SCE_NP_MATCHING2_SIGNALING_ERROR_CTX_MAX	0x80550e12	Exceeded maximum number of contexts that can be created
SCE_NP_MATCHING2_SIGNALING_ERROR_RESULT_NOT_FOUND	0x80550e13	Specified result does not exist
SCE_NP_MATCHING2_SIGNALING_ERROR_CONN_IN_PROGRESS	0x80550e14	Specified connection is not ACTIVE
SCE_NP_MATCHING2_SIGNALING_ERROR_INVALID_ARGUMENT	0x80550e15	Invalid argument was specified
SCE_NP_MATCHING2_SIGNALING_ERROR_OWN_NP_ID	0x80550e16	Specified own NP ID
SCE_NP_MATCHING2_SIGNALING_ERROR_TOO_MANY_CONN	0x80550e17	Number of connections has exceeded 64
SCE_NP_MATCHING2_SIGNALING_ERROR_TERMINATED_BY_MYSELF	0x80550e18	Became INACTIVE due to own request
SCE_NP_MATCHING2_SIGNALING_ERROR_MATCHING2_PEER_NOT_FOUND	0x80550e19	Specified peer does not exist