

NP IN-GAME Commerce 2 Reference

© 2014 Sony Computer Entertainment Inc.
All Rights Reserved.
SCE Confidential

Table of Contents

Datatypes.....	4
SceNpCommerce2CommonData	5
SceNpCommerce2Range	6
Initialization/Termination API	7
sceNpCommerce2Init.....	8
sceNpCommerce2Term	9
Context API	10
sceNpCommerce2CreateCtx	11
sceNpCommerce2DestroyCtx	12
sceNpCommerce2GetShortfallOfLibhttpPool	13
sceNpCommerce2GetShortfallOfLibsslPool	14
Empty Store Check API.....	15
sceNpCommerce2StartEmptyStoreCheck	16
sceNpCommerce2StopEmptyStoreCheck	17
PlayStation®Store Icon Display API.....	18
sceNpCommerce2ShowPsStoreIcon	19
sceNpCommerce2HidePsStoreIcon	20
Session API	21
sceNpCommerce2CreateSessionCreateReq	22
sceNpCommerce2CreateSessionStart	23
sceNpCommerce2CreateSessionGetResult	24
SceNpCommerce2SessionInfo	25
sceNpCommerce2GetSessionInfo	27
Category Content API	28
sceNpCommerce2GetCategoryContentsCreateReq	29
sceNpCommerce2GetCategoryContentsStart	30
sceNpCommerce2GetCategoryContentsGetResult	31
sceNpCommerce2InitGetCategoryContentsResult	33
SceNpCommerce2GetCategoryContentsResult	34
sceNpCommerce2GetCategoryInfo	35
SceNpCommerce2CategoryInfo	36
sceNpCommerce2GetContentInfo	38
SceNpCommerce2ContentInfo	39
sceNpCommerce2GetCategoryInfoFromContentInfo	40
sceNpCommerce2GetGameProductInfoFromContentInfo	41
sceNpCommerce2DestroyGetCategoryContentsResult	42
Product Information API	43
sceNpCommerce2GetProductInfoCreateReq	44
sceNpCommerce2GetProductInfoStart	45
sceNpCommerce2GetProductInfoGetResult	46
sceNpCommerce2InitGetProductInfoResult	48
SceNpCommerce2GetProductInfoResult	49
sceNpCommerce2GetGameProductInfo	50

SceNpCommerce2GameProductInfo	51
sceNpCommerce2DestroyGetProductInfoResult	53
Product Info List API	54
sceNpCommerce2GetProductInfoListCreateReq	55
sceNpCommerce2GetProductInfoListStart	56
sceNpCommerce2GetProductInfoListGetResult	57
sceNpCommerce2InitGetProductInfoListResult	59
SceNpCommerce2GetProductInfoListResult	60
sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult	61
sceNpCommerce2DestroyGetProductInfoListResult	62
Rating Information API	63
SceNpCommerce2ContentRatingInfo	64
SceNpCommerce2ContentRatingDescriptor	65
sceNpCommerce2GetContentRatingInfoFromGameProductInfo	66
sceNpCommerce2GetContentRatingInfoFromCategoryInfo	67
sceNpCommerce2GetContentRatingDescriptor	68
SKU Information API	69
SceNpCommerce2GameSkulInfo	70
sceNpCommerce2GetGameSkulInfoFromGameProductInfo	73
sceNpCommerce2GetPrice	74
Request API	75
sceNpCommerce2AbortReq	76
sceNpCommerce2DestroyReq	78
Constants	79
SCE_NP_COMMERCE2_RECV_BUF_SIZE	80
SCE_NP_COMMERCE2_GETCAT_MAX_COUNT	81
SCE_NP_COMMERCE2_GETPRODLIST_MAX_COUNT	82
String Lengths	83
Return Codes	84

Datatypes

000004892117

SCE CONFIDENTIAL

SceNpCommerce2CommonData

Common structure used when receiving data

Definition

```
#include <np/np_commerce2.h>
typedef struct SceNpCommerce2CommonData_
{
    SceUInt32 version;
    const SceUChar8 *buf_head;
    const SceUChar8 *buf_size;
    const SceUChar8 *data;
    SceSize data_size;
    const SceUChar8 *data2;
    unsigned int ov;
    SceUInt32 reserved[16];
} SceNpCommerce2CommonData;
```

Members

<i>version</i>	Version
<i>buf_head</i>	Starting address of data
<i>buf_size</i>	Size of data
<i>data</i>	Internal data
<i>data_size</i>	Size of internal data
<i>data2</i>	Internal data 2
<i>ov</i>	Internal data 3
<i>reserved</i>	(Unused)

Description

This structure is used for data obtained with the NP IN-GAME Commerce 2 library.
Do not access the members of this structure directly.

See Also

SceNpCommerce2GetCategoryContentsResult, SceNpCommerce2CategoryInfo,
SceNpCommerce2ContentInfo, SceNpCommerce2GetProductInfoResult,
SceNpCommerce2GameProductInfo, SceNpCommerce2ContentRatingInfo,
SceNpCommerce2ContentRatingDescriptor, SceNpCommerce2GameSkuInfo

SCE CONFIDENTIAL

SceNpCommerce2Range

Structure indicating the range of results obtained

Definition

```
#include <np/np_commerce2.h>
typedef struct SceNpCommerce2Range_
{
    SceUInt32 startPosition;
    SceUInt32 count;
    SceUInt32 totalCountOfResults;
    SceUInt32 reserved[8];
} SceNpCommerce2Range;
```

Members

<i>startPosition</i>	Position of starting element
<i>count</i>	Number of elements
<i>totalCountOfResults</i>	Total number of elements that can be obtained from the server
<i>reserved</i>	(Unused)

Description

When multiple elements are obtained in a list, this structure indicates the range of information obtained.

See Also

SceNpCommerce2GetCategoryContentsResult

Initialization/Termination API

SCE CONFIDENTIAL

sceNpCommerce2Init

Initialize the NP IN-GAME Commerce 2 library

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2Init(
    void
);
```

Calling Conditions

Not multithread safe.

Arguments

None

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function initializes the NP IN-GAME Commerce 2 library.

See Also

sceNpCommerce2Term()

SCE CONFIDENTIAL

sceNpCommerce2Term

Terminate the NP IN-GAME Commerce 2 library

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2Term (
    void
);
```

Calling Conditions

Not multithread safe.

Arguments

None

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function terminates the NP IN-GAME Commerce 2 library.

See Also

sceNpCommerce2Init()

Context API

000004892117

SCE CONFIDENTIAL

sceNpCommerce2CreateCtx

Create a commerce 2 context

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2CreateCtx(
    SceUInt32 version,
    SceUInt32 *ctxId
);
```

Calling Conditions

Multithread safe.

Arguments

<i>version</i>	Version [IN]
<i>ctxId</i>	Pointer to the area to store the context ID [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function creates a commerce 2 context. A commerce 2 context is required when calling a function that issues a request to the server of PSN[®]. This means that a commerce 2 context is required when creating sessions, obtaining category content information or product information, and executing checkouts.

When this function ends normally, the context ID is returned to **ctxId*.

Notes

In the current specifications, the number of contexts that can be created at one time is one (SCE_NP_COMMERCE2_CTX_MAX).

See Also

`sceNpCommerce2DestroyCtx()`

SCE CONFIDENTIAL

sceNpCommerce2DestroyCtx

Destroy a commerce 2 context

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2DestroyCtx(
    SceUInt32 ctxId
);
```

Calling Conditions

Multithread safe.

Arguments

ctxId Context ID [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function destroys a commerce 2 context.

See Also

sceNpCommerce2CreateCtx()

SCE CONFIDENTIAL

sceNpCommerce2GetShortfallOfLibhttpPool

Obtain lacking size of the libhttp memory pool during the previous error

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetShortfallOfLibhttpPool(
    SceUInt32 ctxId,
    SceInt32 *shortfall
);
```

Calling Conditions

Multithread safe.

Arguments

<i>ctxId</i>	Context ID [IN]
<i>shortfall</i>	Size lacking in the libhttp memory pool during the previous error [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function obtains the size that was lacking in the specified commerce 2 context when the last SCE_NP_COMMERCE2_ERROR_HTTP_POOL_TOO_SHORT error occurred.

SCE CONFIDENTIAL

sceNpCommerce2GetShortfallOfLibsslPool

Obtain lacking size of the libssl memory pool during the previous error

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetShortfallOfLibsslPool(
    SceUInt32 ctxId,
    SceInt32 *shortfall
);
```

Calling Conditions

Multithread safe.

Arguments

<i>ctxId</i>	Context ID [IN]
<i>shortfall</i>	Size lacking in the libssl memory pool during the previous error [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function obtains the size that was lacking in the specified commerce 2 context when the last `SCE_NP_COMMERCE2_ERROR_SSL_POOL_TOO_SHORT` error occurred.

Empty Store Check API

SCE CONFIDENTIAL

sceNpCommerce2StartEmptyStoreCheck

Start empty store check

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2StartEmptyStoreCheck (
    int storeCheckType,
    const char *targetId,
    void *buf,
    SceSize bufLen,
    int *isEmpty
);
```

Calling Conditions

Multithread safe.

Arguments

<i>storeCheckType</i>	Detection target type [IN] SCE_NP_COMMERCE2_STORE_CHECK_TYPE_CATEGORY: category
<i>targetId</i>	Detection target ID (category ID) corresponding to the type specified in <i>storeCheckType</i> [IN]
<i>buf</i>	Buffer area to use internally [IN]
<i>bufLen</i>	Size of area pointed to by <i>buf</i> [IN] Specify SCE_NP_COMMERCE2_RECV_BUF_SIZE.
<i>isEmpty</i>	Pointer to area to return the detection result [OUT] SCE_NP_COMMERCE2_STORE_IS_EMPTY: target category is empty (no distributed items) SCE_NP_COMMERCE2_STORE_IS_NOT_EMPTY: target category is not empty (distributed items exist)

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function starts the detection of whether distributed items exist (or whether the store is empty) by specifying a category.

This function is blocking. It issues a request to the server of PSN™ and does not return until a response is received or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2StopEmptyStoreCheck()`.

See Also

`sceNpCommerce2StopEmptyStoreCheck()`

SCE CONFIDENTIAL

sceNpCommerce2StopEmptyStoreCheck

Abort empty store check

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2StopEmptyStoreCheck (
    void
);
```

Calling Conditions

Multithread safe.

Arguments

None

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function aborts the empty store check (detection processing of a store without distributed items) started with `sceNpCommerce2StartEmptyStoreCheck()`.

See Also

`sceNpCommerce2StartEmptyStoreCheck()`

PlayStation®Store Icon Display API

SCE CONFIDENTIAL

sceNpCommerce2ShowPsStoreIcon

Show PlayStation®Store Icon

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2ShowPsStoreIcon(
    int iconDisp
);
```

Calling Conditions

Multithread safe.

Arguments

<i>iconDisp</i>	Position where PlayStation®Store Icon is to be displayed [IN] SCE_NP_COMMERCE2_ICON_DISP_LEFT: Bottom left of the screen SCE_NP_COMMERCE2_ICON_DISP_CENTER: Bottom center of the screen SCE_NP_COMMERCE2_ICON_DISP_RIGHT: Bottom right of the screen
-----------------	---

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function performs the overlay display of PlayStation®Store Icon on the screen.

For *iconDisp*, specify the position where PlayStation®Store Icon is to be located.

Use `sceNpCommerce2HidePsStoreIcon()` to hide PlayStation®Store Icon displayed with this function.

See Also

`sceNpCommerce2HidePsStoreIcon()`

SCE CONFIDENTIAL

sceNpCommerce2HidePsStoreIcon

Hide PlayStation®Store Icon

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2HidePsStoreIcon(
    void
);
```

Calling Conditions

Multithread safe.

Arguments

None

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function terminates the overlay display of PlayStation®Store Icon performed through `sceNpCommerce2ShowPsStoreIcon()`.

See Also

`sceNpCommerce2ShowPsStoreIcon()`

Session API

000004892117

SCE CONFIDENTIAL

sceNpCommerce2CreateSessionCreateReq

Create a request to obtain a commerce 2 session

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2CreateSessionCreateReq(
    SceUInt32 ctxId,
    SceUInt32 *reqId
);
```

Calling Conditions

Multithread safe.

Arguments

ctxId Context ID [IN]
reqId Request ID [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function creates a commerce 2 request for creating a commerce 2 session.

Upon normal termination, the request ID will be stored in **reqId*. Specify this request ID as the argument of `sceNpCommerce2CreateSessionStart()`.

Notes

In the current implementation, the number of commerce 2 requests that can be created at one time for one commerce 2 context is one.

See Also

`sceNpCommerce2CreateSessionStart()`, `sceNpCommerce2CreateSessionGetResult()`,
`sceNpCommerce2AbortReq()`, `sceNpCommerce2DestroyReq()`

sceNpCommerce2CreateSessionStart

Start creating a commerce 2 session

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2CreateSessionStart(
    SceUInt32 reqId
);
```

Calling Conditions

Multithread safe.

Arguments

reqId Request ID [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function starts the creation of a commerce 2 session. Before a request to the server of PSNSM can be issued to obtain category information or product information, a commerce 2 session must be created with this function.

For *reqId*, specify the request ID created using `sceNpCommerce2CreateSessionCreateReq()`.

This function is blocking. It issues a request to the server of PSNSM and does not return until a response is received or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2AbortReq()`.

See Also

`sceNpCommerce2CreateSessionCreateReq()`,
`sceNpCommerce2CreateSessionGetResult()`, `sceNpCommerce2AbortReq()`

SCE CONFIDENTIAL

sceNpCommerce2CreateSessionGetResult

Obtain resulting data of a commerce 2 session creation

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2CreateSessionGetResult (
    SceUInt32 reqId,
    void *buf,
    SceSize bufLen,
    SceSize *fillSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>reqId</i>	Request ID [IN]
<i>buf</i>	Buffer area for storing the obtained data [IN]
<i>bufLen</i>	Size of area pointed to by <i>buf</i> [IN]
<i>fillSize</i>	Size of obtained data [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function obtains the resulting data regarding the creation of a commerce 2 session.

For *reqId*, specify the request ID specified in `sceNpCommerce2CreateSessionStart()`.

Prepare a memory area of `SCE_NP_COMMERCE2_RECV_BUF_SIZE` bytes. Specify the beginning address of this area to *buf*, and its size to *bufLen*. Upon normal termination, the session data will be stored in this memory area and the size of the data will be stored in **fillSize*.

See Also

`sceNpCommerce2CreateSessionCreateReq()`, `sceNpCommerce2CreateSessionStart()`,
`sceNpCommerce2AbortReq()`

SCE CONFIDENTIAL

SceNpCommerce2SessionInfo

Structure for session information

Definition

```
#include <np/np_commerce2.h>
typedef struct SceNpCommerce2SessionInfo_
{
    char currencyCode[SCE_NP_COMMERCE2_CURRENCY_CODE_LEN + 1];
    SceUInt32 decimals;
    char currencySymbol[SCE_NP_COMMERCE2_CURRENCY_SYMBOL_LEN + 1];
    SceUInt32 symbolPosition;
    SceBool symbolWithSpace;
    SceUChar8 padding1[3];
    char thousandSeparator[SCE_NP_COMMERCE2_THOUSAND_SEPARATOR_LEN + 1];
    char decimalLetter[SCE_NP_COMMERCE2_DECIMAL_LETTER_LEN + 1];
    SceUChar8 padding2[2];
    SceUInt32 reserved[16];
} SceNpCommerce2SessionInfo;
```

Members

<i>currencyCode</i>	Currency code with terminating NULL character (USD, JPY, etc.)
<i>decimals</i>	Number of digits past the decimal point
<i>currencySymbol</i>	Currency symbol with terminating NULL character (\$, ¥, etc.)
<i>symbolPosition</i>	Flag indicating the position of the currency symbol SCE_NP_COMMERCE2_SYM_POS_PRE: places the currency symbol before the number SCE_NP_COMMERCE2_SYM_POS_POST: places the currency symbol after the number
<i>symbolWithSpace</i>	Flag indicating whether to include a space between the currency symbol and the number
<i>padding1</i>	Specifying true inserts a space. Padding
<i>thousandSeparator</i>	Separator to use per 3 digits in the number (with terminating NULL character)
<i>decimalLetter</i>	Character to use as the decimal (with terminating NULL character)
<i>padding2</i>	Padding
<i>reserved</i>	(Unused)

Description

This structure represents session information.

For a list of currency codes organized by country/region, refer to the document "PSN™ Commerce Service Overview".

Notes

Under normal circumstances, it is not necessary for the application to access the members of this structure directly, because the function `sceNpCommerce2GetPrice()` is provided for formatting the price properly for display. However, `sceNpCommerce2GetPrice()` always outputs the price with the currency code. If it is necessary to display the price using the currency symbol, use the information in this structure.

SCE CONFIDENTIAL

See Also

sceNpCommerce2GetSessionInfo()

000004892117

SCE CONFIDENTIAL

sceNpCommerce2GetSessionInfo

Obtain commerce 2 session information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetSessionInfo(
    SceNpCommerce2SessionInfo *sessionInfo,
    void *data,
    SceSize dataSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>sessionInfo</i>	Pointer to the session information structure [OUT]
<i>data</i>	Data obtained with <code>sceNpCommerce2CreateSessionGetResult()</code> [IN]
<i>dataSize</i>	Size of the data obtained with <code>sceNpCommerce2CreateSessionGetResult()</code> [IN]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function obtains session information from the resulting data of the session creation process obtained using `sceNpCommerce2CreateSessionGetResult()`.

For *data*, specify the beginning address of the resulting data of the session creation process obtained using `sceNpCommerce2CreateSessionGetResult()`. For *dataSize*, specify the size of that data [in other words, the size returned to **fillSize* in `sceNpCommerce2CreateSessionGetResult()`].

When this function terminates normally, session information will be stored in the structure pointed to by *sessionInfo*.

See Also

`sceNpCommerce2CreateSessionGetResult()`, `SceNpCommerce2SessionInfo`

Category Content API

SCE CONFIDENTIAL

sceNpCommerce2GetCategoryContentsCreateReq

Create a request to obtain category content data

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetCategoryContentsCreateReq(
    SceUInt32 ctxId,
    SceUInt32 *reqId
);
```

Calling Conditions

Multithread safe.

Arguments

ctxId Context ID [IN]
reqId Request ID [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function creates a commerce 2 request for obtaining category content data.

When this function ends normally, the request ID is stored to **reqId*. Specify this request ID as an argument of `sceNpCommerce2GetCategoryContentsStart()`.

Notes

In the current implementation, the number of commerce 2 requests that can be created at one time for one commerce 2 context is one.

See Also

`sceNpCommerce2GetCategoryContentsStart()`,
`sceNpCommerce2GetCategoryContentsGetResult()`, `sceNpCommerce2AbortReq()`,
`sceNpCommerce2DestroyReq()`

SCE CONFIDENTIAL

sceNpCommerce2GetCategoryContentsStart

Start obtaining category content data

Definition

```
#include <np/np_commerce2.h>
#define SCE_NP_COMMERCE2_GETCAT_MAX_COUNT 60
int sceNpCommerce2GetCategoryContentsStart(
    SceUInt32 reqId,
    const char *categoryId,
    SceUInt32 startPosition,
    SceUInt32 maxCountOfResults
);
```

Calling Conditions

Multithread safe.

Arguments

<i>reqId</i>	Request ID [IN]
<i>categoryId</i>	ID of the category to obtain the content of [IN]
<i>startPosition</i>	Index indicating the content to start from [IN]
<i>maxCountOfResults</i>	Maximum number of contents to obtain [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function starts obtaining category content data.

To *reqId*, specify the request ID returned by `sceNpCommerce2GetCategoryContentsCreateReq()`.

To *categoryId*, specify the ID of the target category.

To *startPosition* and *maxCountOfResults*, specify the position of the first content to obtain and the number of contents. The value of *maxCountOfResults* must be no greater than `SCE_NP_COMMERCE2_GETCAT_MAX_COUNT`.

This function is blocking. It issues a request to the server of PSNSM and does not return until a response is received or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2AbortReq()`.

See Also

`sceNpCommerce2GetCategoryContentsCreateReq()`,
`sceNpCommerce2GetCategoryContentsGetResult()`, `sceNpCommerce2AbortReq()`

SCE CONFIDENTIAL

sceNpCommerce2GetCategoryContentsGetResult

Obtain category content data

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetCategoryContentsGetResult (
    SceUInt32 reqId,
    void *buf,
    SceSize bufLen,
    SceSize *fillSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>reqId</i>	Request ID [IN]
<i>buf</i>	Buffer area to store the data obtained [IN]
<i>bufLen</i>	Size of area pointed to by <i>buf</i> [IN]
<i>fillSize</i>	Size of data obtained [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function obtains category content data.

To *reqId*, specify the request ID specified in `sceNpCommerce2GetCategoryContentsStart()`.

Create a memory area of `SCE_NP_COMMERCE2_RECV_BUF_SIZE` bytes, and specify its starting address to *buf*, and its size to *bufLen*.

This function is blocking. It does not return until it receives category content data from the server of PSN™ or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2AbortReq()`.

When this function ends normally, the size of the category content data obtained is stored to **fillSize*. The category content data can then be taken out with a function such as `sceNpCommerce2GetCategoryInfo()`, but only after it is initialized with `sceNpCommerce2InitGetCategoryContentsResult()`.

Warning

An account with a role that allows accesses with the NP IN-GAME Commerce 2 library (such as TitleAdmin or TitleDev) is required during development in the development environment. It is also necessary to specify the source IP addresses. Without this setup, this function will return an `SCE_NP_COMMERCE2_SERVER_ERROR_ACCESS_PERMISSION_DENIED` or `SCE_NP_COMMERCE2_SERVER_ERROR_UNKNOWN_ERROR` error.

SCE CONFIDENTIAL

Notes

The data obtained with this function is of just the category content defined for the country/region to which the NP account belongs.

The SKUs that are unavailable to the user according to eligibility rules will not be included in the category content data obtained. For example, an SKU that is available only for a single purchase will not be included in the category content data if it has already been purchased, but unless there is such an eligibility rule, information of all SKUs can be obtained whether or not they have been purchased. If an SKU has already been purchased, the member *purchasabilityFlag* in the *SceNpCommerce2GameSkuInfo* structure obtained will be set to *SCE_NP_COMMERCE2_SKU_PURCHASABILITY_FLAG_OFF*.

In principle, do not cache the category content data obtained (and the category content information taken out) locally, and instead obtain/take out the data when needed.

See Also

`sceNpCommerce2GetCategoryContentsStart()`, `sceNpCommerce2AbortReq()`

SCE CONFIDENTIAL

sceNpCommerce2InitGetCategoryContentsResult

Initialize category content data

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2InitGetCategoryContentsResult(
    SceNpCommerce2GetCategoryContentsResult *result,
    void *data,
    SceSize dataSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>result</i>	Pointer to initialized structure with category content data [OUT]
<i>data</i>	Starting address of category content data [IN]
<i>dataSize</i>	Size of category content data [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function initializes the category content data obtained with `sceNpCommerce2GetCategoryContentsGetResult()`, and enables category content information to be taken out with a function such as `sceNpCommerce2GetCategoryInfo()`.

To *data*, specify the starting address of the category content data obtained with `sceNpCommerce2GetCategoryContentsGetResult()`. To *dataSize*, specify the size of the data as returned to **fillSize* with `sceNpCommerce2GetCategoryContentsGetResult()`.

When this function ends normally, the initialized category content data is stored to the structure pointed to by *result*. Specify the address of this structure as an argument when calling a function such as `sceNpCommerce2GetCategoryInfo()` to take out category content information. However, the actual category content information is stored in the data area specified with *data*, so do not free this area or write other data until category content information has been taken out and `sceNpCommerce2DestroyGetCategoryContentsResult()` has been called.

See Also

`sceNpCommerce2GetCategoryContentsGetResult()`,
`SceNpCommerce2GetCategoryContentsResult`,
`sceNpCommerce2DestroyGetCategoryContentsResult()`

SceNpCommerce2GetCategoryContentsResult

Structure for initialized category content data

Definition

```
#include <np/np_commerce2.h>
typedef struct SceNpCommerce2GetCategoryContentsResult_
{
    SceNpCommerce2CommonData commonData;
    SceNpCommerce2Range rangeOfContents;
    SceUInt32 reserved[8];
} SceNpCommerce2GetCategoryContentsResult;
```

Members

<i>commonData</i>	Information common to data received
<i>rangeOfContents</i>	Range of content included
<i>reserved</i>	(Unused)

Description

This structure represents the initialized category content data.

First, put the category content data obtained with `sceNpCommerce2GetCategoryContentsGetResult()` into this structure by executing `sceNpCommerce2InitGetCategoryContentsResult()`. Then use a function such as `sceNpCommerce2GetCategoryInfo()` to take out the necessary information.

See Also

`sceNpCommerce2InitGetCategoryContentsResult()`

SCE CONFIDENTIAL

sceNpCommerce2GetCategoryInfo

Take out category information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetCategoryInfo(
    const SceNpCommerce2GetCategoryContentsResult *result,
    SceNpCommerce2CategoryInfo *categoryInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>result</i>	Pointer to initialized structure with category content data [IN]
<i>categoryInfo</i>	Pointer to category information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out the category name, the number of products in the category, and other category information from the initialized category content data.

When this function ends normally, category information is stored to **categoryInfo*. This category information has `SCE_NP_COMMERCE2_CAT_DATA_TYPE_NORMAL` set to *dataType*, and includes all category information.

See Also

`SceNpCommerce2CategoryInfo`, `sceNpCommerce2DestroyGetCategoryContentsResult()`

SCE CONFIDENTIAL

SceNpCommerce2CategoryInfo

Structure for category information

Definition

```
#include <np/np_commerce2.h>

enum SceNpCommerce2CategoryDataType {
    SCE_NP_COMMERCE2_CAT_DATA_TYPE_THIN = 0,
    SCE_NP_COMMERCE2_CAT_DATA_TYPE_NORMAL,
    SCE_NP_COMMERCE2_CAT_DATA_TYPE_MAX
};

typedef struct SceNpCommerce2CategoryInfo_
{
    SceNpCommerce2CommonData commonData;
    enum SceNpCommerce2CategoryDataType dataType;
    const char *categoryId;
    SceUInt32 padding;
    SceRtcTick releaseDate;
    const char *categoryName;
    const char *categoryDescription;
    const char *imageUrl;
    const char *spName;
    SceUInt32 countOfSubCategory;
    SceUInt32 countOfProduct;
    SceUInt32 reserved[16];
} SceNpCommerce2CategoryInfo;
```

Members

<i>commonData</i>	Information common to data received
<i>dataType</i>	Type of category information SCE_NP_COMMERCE2_CAT_DATA_TYPE_THIN: category information that excludes some information SCE_NP_COMMERCE2_CAT_DATA_TYPE_NORMAL: category information with all the information
<i>categoryId</i>	ID of category
<i>padding</i>	Padding
<i>releaseDate</i>	(Unused)
<i>categoryName</i>	Name of category
<i>categoryDescription</i>	Detailed information of category
<i>imageUrl</i>	Image URL of category
<i>spName</i>	(Unused)
<i>countOfSubCategory</i>	Number of subcategories in category (valid only when <i>dataType</i> is SCE_NP_COMMERCE2_CAT_DATA_TYPE_NORMAL)
<i>countOfProduct</i>	Number of products in category (valid only when <i>dataType</i> is SCE_NP_COMMERCE2_CAT_DATA_TYPE_NORMAL)
<i>reserved</i>	(Unused)

Description

This structure represents category information.

It includes information of the category itself (such as the category name and image URL) and information of content in the category.

Of the category content data obtained with `sceNpCommerce2GetCategoryContentsGetResult()`, information of the target category can be taken out with `sceNpCommerce2GetCategoryInfo()`.

This category information structure has *dataType* set to

`SCE_NP_COMMERCE2_CAT_DATA_TYPE_NORMAL`.

Information of the subcategories in the target category can be taken out with

`sceNpCommerce2GetCategoryInfoFromContentInfo()`. This category information structure has *dataType* set to `SCE_NP_COMMERCE2_CAT_DATA_TYPE_THIN`.

See Also

`sceNpCommerce2GetCategoryInfo()`,

`sceNpCommerce2GetCategoryInfoFromContentInfo()`

SCE CONFIDENTIAL

sceNpCommerce2GetContentInfo

Take out information of content in the category

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetContentInfo (
    const SceNpCommerce2GetCategoryContentsResult *result,
    unsigned int index,
    SceNpCommerce2ContentInfo *contentInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>result</i>	Pointer to initialized structure with category content data [IN]
<i>index</i>	Index number of content to take out [IN]
<i>contentInfo</i>	Pointer to content information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out information of one content in the category (one subcategory or product) from the initialized category content data.

To *index*, specify the index number of the content to take out. To take out the first content, specify 0.

When this function ends normally, the specified content information is stored to **contentInfo*.

The value of *contentType* of the content information structure indicates whether the target content is a subcategory or a product. Information of subcategories can be taken out with `sceNpCommerce2GetCategoryInfoFromContentInfo()`. (This includes only partial category information.)

Information of products can be taken out with

`sceNpCommerce2GetGameProductInfoFromContentInfo()`. (This includes only partial product information.)

Notes

The number of contents included in the category can be obtained in *count* from the *rangeOfContents* member in the initialized `SceNpCommerce2GetCategoryContentsResult` structure.

See Also

`SceNpCommerce2ContentInfo`, `sceNpCommerce2DestroyGetCategoryContentsResult()`

SceNpCommerce2ContentInfo

Structure for content information within the category

Definition

```
#include <np/np_commerce2.h>
typedef struct SceNpCommerce2ContentInfo_
{
    SceNpCommerce2CommonData commonData;
    SceUInt32 contentType;
    SceUInt32 reserved[8];
} SceNpCommerce2ContentInfo;
```

Members

<i>commonData</i>	Information common to data received
<i>contentType</i>	Type of content SCE_NP_COMMERCE2_CONTENT_TYPE_CATEGORY: subcategory SCE_NP_COMMERCE2_CONTENT_TYPE_PRODUCT: product
<i>reserved</i>	(Unused)

Description

This structure represents one content in the category content taken out with `sceNpCommerce2GetContentInfo()`.

If the content type is a subcategory, category information can be taken out with `sceNpCommerce2GetCategoryInfoFromContentInfo()`.

If the content type is a product, product information can be taken out with `sceNpCommerce2GetGameProductInfoFromContentInfo()`.

See Also

```
sceNpCommerce2GetContentInfo(),
sceNpCommerce2GetCategoryInfoFromContentInfo(),
sceNpCommerce2GetGameProductInfoFromContentInfo()
```

SCE CONFIDENTIAL

sceNpCommerce2GetCategoryInfoFromContentInfo

Take out subcategory information from content information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetCategoryInfoFromContentInfo(
    const SceNpCommerce2ContentInfo *contentInfo,
    SceNpCommerce2CategoryInfo *categoryInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>contentInfo</i>	Pointer to content information structure [IN]
<i>categoryInfo</i>	Pointer to category information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out subcategory information from the content information taken out with `sceNpCommerce2GetContentInfo()`. This function is enabled only when the content is a subcategory (*contentType* of *contentInfo* is `SCE_NP_COMMERCE2_CONTENT_TYPE_CATEGORY`).

When this function ends normally, the category information is stored to **categoryInfo*. This category information has *dataType* set to `SCE_NP_COMMERCE2_CAT_DATA_TYPE_THIN` and does not include some information.

See Also

`sceNpCommerce2GetContentInfo()`, `SceNpCommerce2CategoryInfo`

SCE CONFIDENTIAL

sceNpCommerce2GetGameProductInfoFromContentInfo

Take out product information from content information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetGameProductInfoFromContentInfo (
    const SceNpCommerce2ContentInfo *contentInfo,
    SceNpCommerce2GameProductInfo *gameProductInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>contentInfo</i>	Pointer to content information structure [IN]
<i>gameProductInfo</i>	Pointer to product information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out product information from the content information taken out with `sceNpCommerce2GetContentInfo()`. This function is enabled only when the content is a product (*contentType* of *contentInfo* is `SCE_NP_COMMERCE2_CONTENT_TYPE_PRODUCT`).

When this function ends normally, the product information is stored to **gameProductInfo*. This product information has *dataType* set to `SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_THIN` and does not include some information.

See Also

`sceNpCommerce2GetContentInfo()`, `SceNpCommerce2GameProductInfo`

sceNpCommerce2DestroyGetCategoryContentsResult

Destroy category content data

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2DestroyGetCategoryContentsResult (
    SceNpCommerce2GetCategoryContentsResult *result
);
```

Calling Conditions

Multithread safe.

Arguments

result Pointer to initialized structure with category content data [IN]

Return Values

Value	Description
0	Normal termination

Description

This function destroys the category content data initialized with `sceNpCommerce2InitGetCategoryContentsResult()`.

After this function returns, the area pointed to by *result*, and the area specified with *data* in `sceNpCommerce2InitGetCategoryContentsResult()` can be freed.

Notes

In the current implementation, processing that requires explicit freeing inside `sceNpCommerce2InitGetCategoryContentsResult()` is not performed, so this function is an empty function that always returns a success.

See Also

`sceNpCommerce2InitGetCategoryContentsResult()`

Product Information API

SCE CONFIDENTIAL

sceNpCommerce2GetProductInfoCreateReq

Create a request to obtain product data

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetProductInfoCreateReq(
    SceUInt32 ctxId,
    SceUInt32 *reqId
);
```

Calling Conditions

Multithread safe.

Arguments

ctxId Context ID [IN]
reqId Request ID [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function creates a commerce 2 request for obtaining product data.

When this function ends normally, the request ID is stored to **reqId*. Specify this request ID as an argument of `sceNpCommerce2GetProductInfoStart()`.

Notes

In the current implementation, the number of commerce 2 requests that can be created at one time for one commerce 2 context is one.

See Also

`sceNpCommerce2GetProductInfoStart()`, `sceNpCommerce2GetProductInfoGetResult()`,
`sceNpCommerce2AbortReq()`, `sceNpCommerce2DestroyReq()`

SCE CONFIDENTIAL

sceNpCommerce2GetProductInfoStart

Start obtaining product data

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetProductInfoStart (
    SceUInt32 reqId,
    const char *categoryId,
    const char *productId
);
```

Calling Conditions

Multithread safe.

Arguments

<i>reqId</i>	Request ID [IN]
<i>categoryId</i>	ID of category where the target product is stored [IN]
<i>productId</i>	ID of target product [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function starts obtaining product data.

To *reqId*, specify the request ID returned by `sceNpCommerce2GetProductInfoCreateReq()`.

To *categoryId*, specify the ID of the category to which the target product belongs. NULL can be specified if the category is unknown, but to alleviate the server load, always specify the category ID if known.

This function is blocking. It issues a request to the server of PSNSM and does not return until a response is received or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2AbortReq()`.

See Also

`sceNpCommerce2GetProductInfoCreateReq()`,
`sceNpCommerce2GetProductInfoGetResult()`, `sceNpCommerce2AbortReq()`

SCE CONFIDENTIAL

sceNpCommerce2GetProductInfoGetResult

Obtain product data

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetProductInfoGetResult (
    SceUInt32 reqId,
    void *buf,
    SceSize bufLen,
    SceSize *fillSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>reqId</i>	Request ID [IN]
<i>buf</i>	Buffer area to store the data obtained [IN]
<i>bufLen</i>	Size of area pointed to by <i>buf</i> [IN]
<i>fillSize</i>	Size of data obtained [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function obtains product data.

To *reqId*, specify the request ID specified in `sceNpCommerce2GetProductInfoStart()`.

Create a memory area of `SCE_NP_COMMERCE2_RECV_BUF_SIZE` bytes, and specify its starting address to *buf*, and its size to *bufLen*.

This function is blocking. It does not return until it receives product data from the server of PSN™ or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2AbortReq()`.

When this function ends normally, the size of the product data obtained is stored to **fillSize*. The product data can then be taken out with a function such as

`sceNpCommerce2GetGameProductInfo()`, but only after it is initialized with `sceNpCommerce2InitGetProductInfoResult()`.

Warning

An account with a role that allows accesses with the NP IN-GAME Commerce 2 library (such as TitleAdmin or TitleDev) is required during development in the development environment. It is also necessary to specify the source IP addresses. Without this setup, this function will return an

`SCE_NP_COMMERCE2_SERVER_ERROR_ACCESS_PERMISSION_DENIED` or `SCE_NP_COMMERCE2_SERVER_ERROR_UNKNOWN_ERROR` error.

SCE CONFIDENTIAL

Notes

The product data obtained with this function is of just the products defined for the country/region to which the NP account belongs.

The SKUs that are unavailable to the user according to eligibility rules will not be included in the data obtained. For example, an SKU that is available only for a single purchase will not be included if it has already been purchased, but unless there is such an eligibility rule, information of all SKUs can be obtained whether or not they have been purchased. If an SKU has already been purchased, the member *purchasabilityFlag* in the *SceNpCommerce2GameSkuInfo* structure obtained will be set to *SCE_NP_COMMERCE2_SKU_PURCHASABILITY_FLAG_OFF*.

In principle, do not cache the product data obtained (and the product information taken out) locally, and instead obtain/take out the data when needed.

See Also

`sceNpCommerce2GetProductInfoStart()`, `sceNpCommerce2InitGetProductInfoResult()`,
`sceNpCommerce2AbortReq()`

SCEI CONFIDENTIAL

sceNpCommerce2InitGetProductInfoResult

Initialize the product data obtained

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2InitGetProductInfoResult(
    SceNpCommerce2GetProductInfoResult *result,
    void *data,
    SceSize dataSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>result</i>	Pointer to initialized product data structure [OUT]
<i>data</i>	Starting address of product data [IN]
<i>dataSize</i>	Size of product data [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function initializes the product data obtained with `sceNpCommerce2GetProductInfoGetResult()`, and enables product information to be taken out with a function such as `sceNpCommerce2GetGameProductInfo()`.

To *data*, specify the starting address of the product data obtained with `sceNpCommerce2GetProductInfoGetResult()`. To *dataSize*, specify the size of the data as returned to **fillSize* with `sceNpCommerce2GetProductInfoGetResult()`.

When this function ends normally, the initialized product data is stored to the structure pointed to by *result*. Specify the address of this structure as an argument when calling a function such as `sceNpCommerce2GetGameProductInfo()` to take out product information. However, the actual product information is stored in the data area specified with *data*, so do not free this area or write other data until product information has been taken out and `sceNpCommerce2DestroyGetProductInfoResult()` has been called.

See Also

`sceNpCommerce2GetProductInfoGetResult()`, `SceNpCommerce2GetProductInfoResult`, `sceNpCommerce2DestroyGetProductInfoResult()`

SceNpCommerce2GetProductInfoResult

Structure for initialized product data

Definition

```
#include <np/np_commerce2.h>
typedef struct SceNpCommerce2GetProductInfoResult_
{
    SceNpCommerce2CommonData commonData;
    SceUInt32 reserved[8];
} SceNpCommerce2GetProductInfoResult;
```

Members

<i>commonData</i>	Information common to data received
<i>reserved</i>	(Unused)

Description

This structure represents the initialized product data.

First, put the product data obtained with `sceNpCommerce2GetProductInfoGetResult()` into this structure by executing `sceNpCommerce2InitGetProductInfoResult()`. Then use a function such as `sceNpCommerce2GetGameProductInfo()` to take out the necessary information.

See Also

`sceNpCommerce2InitGetProductInfoResult()`

SCE CONFIDENTIAL

sceNpCommerce2GetGameProductInfo

Take out game product information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetGameProductInfo (
    const SceNpCommerce2GetProductInfoResult *result,
    SceNpCommerce2GameProductInfo *gameProductInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>result</i>	Pointer to initialized product data structure [IN]
<i>gameProductInfo</i>	Pointer to game product information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out the product name, the number of SKUs, and other game product information from the initialized product data.

When this function ends normally, game product information is stored to **gameProductInfo*. This game product information has `SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_NORMAL` set to *dataType*, and includes all game product information.

See Also

`SceNpCommerce2GameProductInfo`, `sceNpCommerce2DestroyGetProductInfoResult()`

SCE CONFIDENTIAL

SceNpCommerce2GameProductInfo

Structure for game product information

Definition

```
#include <np/np_commerce2.h>

enum SceNpCommerce2GameProductDataType {
    SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_THIN = 0,
    SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_NORMAL,
    SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_MAX
};

typedef struct SceNpCommerce2GameProductInfo_
{
    SceNpCommerce2CommonData commonData;
    enum SceNpCommerce2GameProductDataType dataType;
    const char *productId;
    SceUInt32 countOfSku;
    SceRtcTick releaseDate;
    const char *productName;
    const char *productShortDescription;
    const char *imageUrl;
    const char *spName;
    const char *productLongDescription;
    const char *legalDescription;
    SceUInt32 productAttr;
    SceUInt32 reserved[20];
} SceNpCommerce2GameProductInfo;
```

Members

<i>commonData</i>	Information common to data received
<i>dataType</i>	Type of product information
	SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_THIN: product information that excludes some information
	SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_NORMAL: product information with all the information
<i>productId</i>	ID of product
<i>countOfSku</i>	Number of SKUs included in the product
<i>releaseDate</i>	Date to start distribution
<i>productName</i>	Name of product
<i>productShortDescription</i>	Detailed information (short description) of product
<i>imageUrl</i>	Image URL of product
<i>spName</i>	Name of licensee
<i>productLongDescription</i>	Detailed information (long description) of product (valid only when <i>dataType</i> is SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_NORMAL)
<i>legalDescription</i>	Legal text
<i>productAttr</i>	(Unused)
<i>reserved</i>	(Unused)

Description

This structure represents game product information.

Game product information can be taken out with `sceNpCommerce2GetGameProductInfo()` or `sceNpCommerce2GetGameProductInfoFromContentInfo()`. In the former situation, all the product information is included, and *dataType* is set to `SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_NORMAL`. In the latter, the product information does not include some information, and *dataType* is set to `SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_THIN`.

See Also

```
sceNpCommerce2GetGameProductInfo(),  
sceNpCommerce2GetGameProductInfoFromContentInfo(),  
sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()
```

sceNpCommerce2DestroyGetProductInfoResult

Destroy product information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2DestroyGetProductInfoResult (
    SceNpCommerce2GetProductInfoResult *result
);
```

Calling Conditions

Multithread safe.

Arguments

result Pointer to initialized product data structure [IN]

Return Values

Value	Description
0	Normal termination

Description

This function destroys the product data initialized with `sceNpCommerce2InitGetProductInfoResult()`.

After this function returns, the area pointed to by *result*, and the area specified with *data* in `sceNpCommerce2InitGetProductInfoResult()` can be freed.

Notes

In the current implementation, processing that requires explicit freeing inside `sceNpCommerce2InitGetProductInfoResult()` is not performed, so this function is an empty function that always returns a success.

See Also

`sceNpCommerce2InitGetProductInfoResult()`

Product Info List API

000004892117

sceNpCommerce2GetProductInfoListCreateReq

Create a request to obtain a product info list

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetProductInfoListCreateReq(
    SceUInt32 ctxId,
    SceUInt32 *reqId
);
```

Calling Conditions

Multithread safe.

Arguments

ctxId Context ID [IN]
reqId Request ID [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function creates a commerce 2 request for obtaining a product info list.

When this function ends normally, the request ID is stored to **reqId*. Specify this request ID as an argument of `sceNpCommerce2GetProductInfoListStart()`.

Notes

In the current implementation, the number of commerce 2 requests that can be created at one time for one commerce 2 context is one.

See Also

`sceNpCommerce2GetProductInfoListStart()`,
`sceNpCommerce2GetProductInfoListGetResult()`, `sceNpCommerce2AbortReq()`,
`sceNpCommerce2DestroyReq()`

SCE CONFIDENTIAL

sceNpCommerce2GetProductInfoListStart

Start obtaining a product info list

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetProductInfoListStart (
    SceUInt32 reqId,
    const char *productIds[],
    SceUInt32 productNum
);
```

Calling Conditions

Multithread safe.

Arguments

<i>reqId</i>	Request ID [IN]
<i>productIds</i>	Array of target product IDs [IN]
<i>productNum</i>	Number of product IDs in <i>productIds</i> [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function starts obtaining a product info list.

To *reqId*, specify the request ID returned by `sceNpCommerce2GetProductInfoListCreateReq()`.

To *productIds* and *productNum*, specify the product IDs for which to obtain product information, and the number of IDs, respectively. The maximum number of product IDs that can be specified is `SCE_NP_COMMERCE2_GETPRODLIST_MAX_COUNT` (60).

This function is blocking. It issues a request to the server of PSNSM and does not return until a response is received or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2AbortReq()`.

See Also

`sceNpCommerce2GetProductInfoListCreateReq()`,
`sceNpCommerce2GetProductInfoListGetResult()`, `sceNpCommerce2AbortReq()`

SCE CONFIDENTIAL

sceNpCommerce2GetProductInfoListGetResult

Obtain a product info list

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetProductInfoListGetResult (
    SceUInt32 reqId,
    void *buf,
    SceSize bufLen,
    SceSize *fillSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>reqId</i>	Request ID [IN]
<i>buf</i>	Buffer area to store the data obtained [IN]
<i>bufLen</i>	Size of area pointed to by <i>buf</i> [IN]
<i>fillSize</i>	Size of data obtained [OUT]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function obtains a product info list.

To *reqId*, specify the request ID specified in `sceNpCommerce2GetProductInfoListStart()`.

Create a memory area of `SCE_NP_COMMERCE2_RECV_BUF_SIZE` bytes, and specify its starting address to *buf*, and its size to *bufLen*.

This function is blocking. It does not return until it receives the product info list from the server of PSNSM or an error (timeout, for example) occurs within the internally-used libhttp. To abort this function's processing, use `sceNpCommerce2AbortReq()`.

When this function ends normally, the size of the product info list obtained is stored to **fillSize*.

The product info list can then be taken out with

`sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()`, but only after it is initialized with `sceNpCommerce2InitGetProductInfoListResult()`.

Warning

An account with a role that allows accesses with the NP IN-GAME Commerce 2 library (such as TitleAdmin or TitleDev) is required during development in the development environment. It is also necessary to specify the source IP addresses. Without this setup, this function will return an

`SCE_NP_COMMERCE2_SERVER_ERROR_ACCESS_PERMISSION_DENIED` or `SCE_NP_COMMERCE2_SERVER_ERROR_UNKNOWN_ERROR` error.

SCE CONFIDENTIAL

Notes

The product info list obtained with this function contains just the products defined for the country/region to which the NP account belongs.

The SKUs that are unavailable to the user according to eligibility rules will not be included in the data obtained. For example, an SKU that is available only for a single purchase will not be included in the product info list if it has already been purchased, but unless there is such an eligibility rule, information of all SKUs can be obtained whether or not they have been purchased. If an SKU has already been purchased, the member *purchasabilityFlag* in the *SceNpCommerce2GameSkuInfo* structure obtained will be set to *SCE_NP_COMMERCE2_SKU_PURCHASABILITY_FLAG_OFF*.

In principle, do not cache the product info list obtained (and the product/SKU information taken out) locally, and instead obtain/take out the data when needed.

See Also

`sceNpCommerce2GetProductInfoListStart()`, `sceNpCommerce2AbortReq()`,
`sceNpCommerce2InitGetProductInfoListResult()`

SCE CONFIDENTIAL

sceNpCommerce2InitGetProductInfoListResult

Initialize the product info list obtained

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2InitGetProductInfoListResult(
    SceNpCommerce2GetProductInfoListResult *result,
    void *data,
    SceSize dataSize
);
```

Calling Conditions

Multithread safe.

Arguments

<i>result</i>	Pointer to initialized product info list structure [OUT]
<i>data</i>	Starting address of product info list [IN]
<i>dataSize</i>	Size of product info list [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function initializes the product info list obtained with `sceNpCommerce2GetProductInfoListGetResult()`, and enables product information to be taken out with `sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()`. To *data*, specify the starting address of the product info list obtained with `sceNpCommerce2GetProductInfoListGetResult()`. To *dataSize*, specify the size of the data as returned to **fillSize* with `sceNpCommerce2GetProductInfoListGetResult()`.

When this function ends normally, the initialized product info list is stored to the structure pointed to by *result*. Specify the address of this structure as an argument when calling `sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()` to take out product information. However, the actual product information is stored in the data area specified with *data*, so do not free this area or write other data until product information has been taken out and `sceNpCommerce2DestroyGetProductInfoListResult()` has been called.

See Also

`sceNpCommerce2GetProductInfoListGetResult()`,
`SceNpCommerce2GetProductInfoListResult`,
`sceNpCommerce2DestroyGetProductInfoListResult()`

SCE CONFIDENTIAL

SceNpCommerce2GetProductInfoListResult

Structure for initialized product info list

Definition

```
#include <np/np_commerce2.h>
typedef struct SceNpCommerce2GetProductInfoListResult_
{
    SceNpCommerce2CommonData commonData;
    SceUInt32 reserved[8];
} SceNpCommerce2GetProductInfoListResult;
```

Members

<i>commonData</i>	Information common to data received
<i>reserved</i>	(Unused)

Description

This structure represents the initialized product data of multiple products.

First, put the information from the product info list obtained with `sceNpCommerce2GetProductInfoListGetResult()` into this structure by executing `sceNpCommerce2InitGetProductInfoListResult()`. Then use `sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()` to take out the necessary information.

See Also

`sceNpCommerce2InitGetProductInfoListResult()`

SCE CONFIDENTIAL

sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult

Take out game product information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult(
    const SceNpCommerce2GetProductInfoListResult *result,
    unsigned int index,
    SceNpCommerce2GameProductInfo *gameProductInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>result</i>	Pointer to initialized product info list structure [IN]
<i>index</i>	Index number of the product information to take out [IN]
<i>gameProductInfo</i>	Pointer to game product information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out the product name, the number of SKUs, and other game product information from the initialized product info list.

When this function ends normally, game product information is stored to **gameProductInfo*. This game product information has `SCE_NP_COMMERCE2_GAME_PRODUCT_DATA_TYPE_THIN` set to *dataType*, and only includes some of the available information.

See Also

`SceNpCommerce2GameProductInfo`,
`sceNpCommerce2DestroyGetProductInfoListResult()`

sceNpCommerce2DestroyGetProductInfoListResult

Destroy a product info list

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2DestroyGetProductInfoListResult (
    SceNpCommerce2GetProductInfoListResult *result
);
```

Calling Conditions

Multithread safe.

Arguments

result Pointer to initialized product info list structure [IN]

Return Values

Value	Description
0	Normal termination

Description

This function destroys the product info list initialized with `sceNpCommerce2InitGetProductInfoListResult()`.

After this function returns, the area pointed to by *result*, and the area specified with *data* in `sceNpCommerce2InitGetProductInfoListResult()` can be freed.

Notes

In the current implementation, processing that requires explicit freeing inside `sceNpCommerce2InitGetProductInfoListResult()` is not performed, so this function is an empty function that always returns a success.

See Also

`sceNpCommerce2InitGetProductInfoListResult()`

Rating Information API

000004892117

SCE CONFIDENTIAL

SceNpCommerce2ContentRatingInfo

Structure for rating information

Definition

```
#include <np/np_commerce2.h>

typedef struct SceNpCommerce2ContentRatingInfo_
{
    SceNpCommerce2CommonData commonData;
    const char *ratingSystemId;
    const char *imageUrl;
    SceUInt32 countOfContentRatingDescriptor;
    SceUInt32 reserved[8];
} SceNpCommerce2ContentRatingInfo;
```

Members

<i>commonData</i>	Information common to data received
<i>ratingSystemId</i>	ID of rating system (for example: PEGI, ESRB)
<i>imageUrl</i>	URL of rating icon
<i>countOfContentRatingDescriptor</i>	Number of rating descriptors for this content
<i>reserved</i>	(Unused)

Description

This structure represents rating information of one content.

Rating information has one main rating icon (*imageUrl*) with multiple rating descriptors stemming from it. To obtain these rating descriptors, use `sceNpCommerce2GetContentRatingDescriptor()`.

See Also

```
sceNpCommerce2GetContentRatingInfoFromGameProductInfo(),
sceNpCommerce2GetContentRatingInfoFromCategoryInfo(),
sceNpCommerce2GetContentRatingDescriptor()
```


SCE CONFIDENTIAL

SceNpCommerce2ContentRatingDescriptor

Structure for a rating descriptor

Definition

```
#include <np/np_commerce2.h>

#define SCE_NP_COMMERCE2_CONTENT_RATING_DESC_TYPE_ICON 1
#define SCE_NP_COMMERCE2_CONTENT_RATING_DESC_TYPE_TEXT 2

typedef struct SceNpCommerce2ContentRatingDescriptor_
{
    SceNpCommerce2CommonData commonData;
    SceUInt32 descriptorType;
    const char *imageUrl;
    const char *contentRatingDescription;
    SceUInt32 reserved[8];
} SceNpCommerce2ContentRatingDescriptor;
```

Members

<i>commonData</i>	Information common to data received
<i>descriptorType</i>	Type of rating descriptor SCE_NP_COMMERCE2_CONTENT_RATING_DESC_TYPE_ICON: icon SCE_NP_COMMERCE2_CONTENT_RATING_DESC_TYPE_TEXT: text
<i>imageUrl</i>	URL of icon (valid only when <i>descriptorType</i> is SCE_NP_COMMERCE2_CONTENT_RATING_DESC_TYPE_ICON)
<i>contentRatingDescription</i>	Descriptor text (valid only when <i>descriptorType</i> is SCE_NP_COMMERCE2_CONTENT_RATING_DESC_TYPE_TEXT)
<i>reserved</i>	(Unused)

Description

This structure represents one rating descriptor.

Rating descriptors can be either icons or text, and the type is indicated with *descriptorType*. For icons, *imageUrl* is the descriptor information. For text, *contentRatingDescription* is the descriptor information.

See Also

`sceNpCommerce2GetContentRatingDescriptor()`

SCE CONFIDENTIAL

sceNpCommerce2GetContentRatingInfoFromGameProductInfo

Take out rating information of a product

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetContentRatingInfoFromGameProductInfo (
    const SceNpCommerce2GameProductInfo *gameProductInfo,
    SceNpCommerce2ContentRatingInfo *contentRatingInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>gameProductInfo</i>	Pointer to game product information structure [IN]
<i>contentRatingInfo</i>	Pointer to rating information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out rating information from game product information.

To *gameProductInfo*, specify a pointer to the game product information taken out with *sceNpCommerce2GetGameProductInfo()*, *sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()*, or *sceNpCommerce2GetGameProductInfoFromContentInfo()*.

When this function ends normally, rating information is stored to the structure specified with *contentRatingInfo*.

See Also

sceNpCommerce2GetGameProductInfo(),
sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult(),
sceNpCommerce2GetGameProductInfoFromContentInfo(),
SceNpCommerce2ContentRatingInfo

SCE CONFIDENTIAL

sceNpCommerce2GetContentRatingInfoFromCategoryInfo

Take out rating information of a category

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetContentRatingInfoFromCategoryInfo (
    const SceNpCommerce2CategoryInfo *categoryInfo,
    SceNpCommerce2ContentRatingInfo *contentRatingInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>categoryInfo</i>	Pointer to category information structure [IN]
<i>contentRatingInfo</i>	Pointer to rating information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out rating information from category information.

To *categoryInfo*, specify a pointer to the category information taken out with `sceNpCommerce2GetCategoryInfo()` or `sceNpCommerce2GetCategoryInfoFromContentInfo()`.

When this function ends normally, rating information is stored to the structure specified with *contentRatingInfo*.

See Also

`sceNpCommerce2GetCategoryInfo()`,
`sceNpCommerce2GetCategoryInfoFromContentInfo()`,
`SceNpCommerce2ContentRatingInfo`

SCE CONFIDENTIAL

sceNpCommerce2GetContentRatingDescriptor

Take out a rating descriptor

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetContentRatingDescriptor(
    const SceNpCommerce2ContentRatingInfo *contentRatingInfo,
    unsigned int index,
    SceNpCommerce2ContentRatingDescriptor *contentRatingDescriptor
);
```

Calling Conditions

Multithread safe.

Arguments

<i>contentRatingInfo</i>	Rating information structure [IN]
<i>index</i>	Index number of rating descriptor to take out [IN]
<i>contentRatingDescriptor</i>	Rating descriptor structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out a rating descriptor from rating information.

To *contentRatingInfo*, specify a pointer to the rating information structure obtained with `sceNpCommerce2GetContentRatingInfoFromGameProductInfo()` or `sceNpCommerce2GetContentRatingInfoFromCategoryInfo()`.

To *index*, specify the index number indicating the rating descriptor to take out in the rating information structure. To take out the first rating descriptor, specify 0.

When this function ends normally, the rating descriptor is stored to the structure specified with *contentRatingDescriptor*.

Notes

The number of rating descriptors included in the rating information structure is indicated by the *countOfContentRatingDescriptor* member.

See Also

`SceNpCommerce2ContentRatingInfo`, `SceNpCommerce2ContentRatingDescriptor`

SKU Information API

000004892117

SCE CONFIDENTIAL

SceNpCommerce2GameSkuInfo

Structure for SKU information

Definition

```
#include <np/np_commerce2.h>

enum SceNpCommerce2GameSkuDataType {
    SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_THIN = 0,
    SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_NORMAL,
    SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_MAX
};

typedef struct SceNpCommerce2GameSkuInfo_
{
    SceNpCommerce2CommonData commonData;
    enum SceNpCommerce2GameSkuDataType dataType;
    const char *skuId;
    SceUInt32 skuType;
    SceUInt32 countUntilExpiration;
    SceUInt32 timeUntilExpiration;
    SceUInt32 purchasabilityFlag;
    SceUInt32 annotation;
    SceBool downloadable;
    SceUInt32 price;
    const char *skuName;
    const char *productId;
    const char *contentLinkUrl;
    SceUInt32 countOfRewardInfo;
    SceUInt32 skuAttr;
    SceUInt32 salesType;
    SceUInt32 firstPlayExpiration;
    SceUInt32 countOfApplicableRewardInfo;
    SceUInt32 countOfGameSkuEntitlementInfo;
    SceUInt32 reserved1;
    SceRtcTick playableDate;
    SceRtcTick chargeDate;
    SceUInt32 chargeModel;
    const char *augmentedDescription;
    SceUInt32 reserved2[8];
} SceNpCommerce2GameSkuInfo;
```

Members

commonData
dataType

Information common to data received

Type of SKU information

SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_THIN: SKU information that excludes some information

SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_NORMAL: SKU information with all the information

skuId

ID of SKU

skuType

(Unused)

countUntilExpiration

Number of usable times left

(valid only when the number of entitlements in the SKU is one)

timeUntilExpiration

Amount of usable time (in hours)

(This is currently unsupported, and an undefined value will return.)

SCE CONFIDENTIAL

<i>purchasabilityFlag</i>	Flag indicating whether or not the SKU is available for purchase (details below)
<i>annotation</i>	SKU information (details below)
<i>downloadable</i>	Flag indicating whether or not the SKU includes DRM content. If true, at least one DRM content is included
<i>price</i>	Price (details below)
<i>skuName</i>	Name of SKU (SCE will set a specific string for the SKU name before Product distribution. If you want to set a unique SKU name for your title, contact SCE in advance.)
<i>productId</i>	ID of product to which the SKU belongs (valid only when <i>dataType</i> is SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_NORMAL)
<i>contentLinkUrl</i>	URL of ContentLink (valid only when <i>dataType</i> is SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_NORMAL)
<i>countOfRewardInfo</i>	(Unused)
<i>skuAttr</i>	(Unused)
<i>salesType</i>	(Unused)
<i>firstPlayExpiration</i>	(Unused)
<i>countOfApplicableRewardInfo</i>	(Unused)
<i>countOfGameSkuEntitlementInfo</i>	(Unused)
<i>reserved1</i>	(Unused)
<i>playableDate</i>	(Unused)
<i>chargeDate</i>	(Unused)
<i>chargeModel</i>	(Unused)
<i>augmentedDescription</i>	(Unused)
<i>reserved2</i>	(Unused)

Description

This structure represents SKU information.

To *purchasabilityFlag*, one of the following values is stored.

Value	(Number)	Description
SCE_NP_COMMERCE2_SKU_PURCHASABILITY_FLAG_ON	1	Can be purchased
SCE_NP_COMMERCE2_SKU_PURCHASABILITY_FLAG_OFF	0	Cannot be purchased (DRM content in this SKU has already been purchased, for example)

To *annotation*, one of the following values is stored.

Value	(Number)	Description
SCE_NP_COMMERCE2_SKU_ANN_PURCHASED_CANNOT_PURCHASE_AGAIN	0x80000000	Already purchased, and cannot be purchased again
SCE_NP_COMMERCE2_SKU_ANN_PURCHASED_CAN_PURCHASE_AGAIN	0x40000000	Already purchased, and can be purchased again
SCE_NP_COMMERCE2_SKU_ANN_IN_THE_CART	0x20000000	(Unused)
SCE_NP_COMMERCE2_SKU_ANN_CONTENTLINK_SKU	0x10000000	ContentLink type SKU
SCE_NP_COMMERCE2_SKU_ANN_CREDIT_CARD_REQUIRED	0x08000000	(Unused)

If the currency uses a decimal point as indicated by the positive value of *decimals* in the *SceNpCommerce2SessionInfo* structure, the value stored to *price* includes the numbers past the decimal point, and the application must move the decimal point as necessary. For example, in the US store, *decimals* is 2. If *price* is 399, this indicates \$3.99.

SCE CONFIDENTIAL

SKU information can be taken out from product information using
`sceNpCommerce2GetGameSkuInfoFromGameProductInfo()`.

SKU information taken out with `sceNpCommerce2GetGameProductInfo()` has *dataType* set to
`SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_NORMAL`.

However, when product information is taken out with
`sceNpCommerce2GetGameProductInfoFromContentInfo()` or
`sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()`, SKU information
is available only if there is one SKU in the product. This SKU information has *dataType* set to
`SCE_NP_COMMERCE2_GAME_SKU_DATA_TYPE_THIN`.

See Also

`sceNpCommerce2GetGameSkuInfoFromGameProductInfo()`

000004892117

SCE CONFIDENTIAL

sceNpCommerce2GetGameSkuInfoFromGameProductInfo

Take out SKU information

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetGameSkuInfoFromGameProductInfo (
    const SceNpCommerce2GameProductInfo *gameProductInfo,
    SceUInt32 index,
    SceNpCommerce2GameSkuInfo *gameSkuInfo
);
```

Calling Conditions

Multithread safe.

Arguments

<i>gameProductInfo</i>	Game product information structure [IN]
<i>index</i>	Index number of SKU to take out [IN]
<i>gameSkuInfo</i>	SKU information structure [OUT]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function takes out SKU information from game product information.

To *gameProductInfo*, specify a pointer to the game product information obtained with `sceNpCommerce2GetGameProductInfo()`, `sceNpCommerce2GetGameProductInfoFromGetProductInfoListResult()`, or `sceNpCommerce2GetGameProductInfoFromContentInfo()`.

To *index*, specify the index number indicating the SKU to take out in the game product information structure. To take out the first SKU information, specify 0.

When this function ends normally, SKU information is stored to the structure specified with *gameSkuInfo*.

Notes

The number of SKUs included in the game product information structure is indicated by the *countOfSku* member.

See Also

`SceNpCommerce2GameProductInfo`, `SceNpCommerce2GameSkuInfo`

SCE CONFIDENTIAL

sceNpCommerce2GetPrice

Obtain the price (string)

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2GetPrice(
    const SceNpCommerce2SessionInfo *s,
    char *buf,
    SceSize buflen,
    SceUInt32 price
);
```

Calling Conditions

Multithread safe.

Arguments

<i>s</i>	Structure for session information [IN]
<i>buf</i>	Buffer area to store the string obtained [IN]
<i>buflen</i>	Size of area pointed to by <i>buf</i> [IN]
<i>price</i>	Price information obtained from the SKU information structure [IN]

Return Values

Value	Description
0 or higher	Normal termination
Negative value	Error (See "Return Codes")

Description

This function formats the string indicated by the *price* member of the SKU information structure. The format depends on the country/region, such as whether the currency code is placed before or after the number and whether the decimal point is indicated by a comma or a period. This function returns the string in the appropriate format for the country/region, as determined by the territory information obtained for the account from the server when the session was created. The use of this function is recommended when displaying price information to the user.

Request API

000004892117

SCE CONFIDENTIAL

sceNpCommerce2AbortReq

Abort a commerce 2 request

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2AbortReq(
    SceUInt32 reqId
);
```

Calling Conditions

Multithread safe.

Arguments

reqId Request ID [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function aborts the following processes (communication processing for creating/obtaining commerce 2 session/category content information, product information, and product info lists).

- `sceNpCommerce2CreateSessionStart()`
- `sceNpCommerce2CreateSessionGetResult()`
- `sceNpCommerce2GetCategoryContentsStart()`
- `sceNpCommerce2GetCategoryContentsGetResult()`
- `sceNpCommerce2GetProductInfoStart()`
- `sceNpCommerce2GetProductInfoGetResult()`
- `sceNpCommerce2GetProductInfoListStart()`
- `sceNpCommerce2GetProductInfoListGetResult()`

To *reqId*, specify the request ID that was specified in the applicable function.

The aborted function returns the error `SCE_NP_COMMERCE2_ERROR_ABORTED`.

See Also

```
sceNpCommerce2CreateSessionCreateReq(), sceNpCommerce2CreateSessionStart(),  
sceNpCommerce2CreateSessionGetResult(),  
sceNpCommerce2GetCategoryContentsCreateReq(),  
sceNpCommerce2GetCategoryContentsStart(),  
sceNpCommerce2GetCategoryContentsGetResult(),  
sceNpCommerce2GetProductInfoCreateReq(), sceNpCommerce2GetProductInfoStart(),  
sceNpCommerce2GetProductInfoGetResult(),  
sceNpCommerce2GetProductInfoListCreateReq(),  
sceNpCommerce2GetProductInfoListStart(),  
sceNpCommerce2GetProductInfoListGetResult()
```

SCE CONFIDENTIAL

sceNpCommerce2DestroyReq

Delete a commerce 2 request

Definition

```
#include <np/np_commerce2.h>
int sceNpCommerce2DestroyReq(
    SceUInt32 reqId
);
```

Calling Conditions

Multithread safe.

Arguments

reqId Request ID [IN]

Return Values

Value	Description
0	Normal termination
Negative value	Error (See "Return Codes")

Description

This function deletes the commerce 2 request specified with *reqId*. Call this function when a request is no longer needed.

This function also aborts all communication processing of the specified request.

See Also

`sceNpCommerce2AbortReq()`

Constants

000004892117

SCE CONFIDENTIAL

SCE_NP_COMMERCE2_RECV_BUF_SIZE

Size of the receive buffer

Definition

Value	(Number)
SCE_NP_COMMERCE2_RECV_BUF_SIZE	256*1024

Description

This is the receive buffer size to be specified in the functions
`sceNpCommerce2StartEmptyStoreCheck()`, `sceNpCommerce2CreateSessionGetResult()`,
`sceNpCommerce2GetCategoryContentsGetResult()`,
`sceNpCommerce2GetProductInfoGetResult()`, and
`sceNpCommerce2GetProductInfoListGetResult()`.

SCE CONFIDENTIAL

SCE_NP_COMMERCE2_GETCAT_MAX_COUNT

Maximum number of category contents to get

Definition

Value	(Number)
SCE_NP_COMMERCE2_GETCAT_MAX_COUNT	60

Description

This is the largest value that can be specified to the argument *maxCountOfResults* of the function `sceNpCommerce2GetCategoryContentsStart()` for starting getting category contents.

SCE CONFIDENTIAL

SCE_NP_COMMERCE2_GETPRODLIST_MAX_COUNT

Maximum items to obtain in the product information list

Definition

Value	(Number)
SCE_NP_COMMERCE2_GETPRODLIST_MAX_COUNT	60

Description

This is the maximum value that can be specified to the *productNum* argument of the function for starting the obtainment of the product information list, `sceNpCommerce2GetProductInfoListStart()`.

SCE CONFIDENTIAL

String Lengths

Maximum lengths of strings

Definition

Value	(Number)	Description
SCE_NP_COMMERCE2_CURRENCY_CODE_LEN	3	Size of currency code
SCE_NP_COMMERCE2_CURRENCY_SYMBOL_LEN	3	Size of the currency symbol
SCE_NP_COMMERCE2_THOUSAND_SEPARATOR_LEN	4	Size of the character separating every 3 digits of the price
SCE_NP_COMMERCE2_DECIMAL_LETTER_LEN	4	Size of the character indicating the decimal point in the price
SCE_NP_COMMERCE2_SP_NAME_LEN	256	Size of licensee name
SCE_NP_COMMERCE2_CATEGORY_ID_LEN	56	Size of category ID
SCE_NP_COMMERCE2_CATEGORY_NAME_LEN	256	Size of category name
SCE_NP_COMMERCE2_CATEGORY_DESCRIPTION_LEN	1024	Size of category details
SCE_NP_COMMERCE2_PRODUCT_ID_LEN	48	Size of product ID
SCE_NP_COMMERCE2_PRODUCT_NAME_LEN	256	Size of product name
SCE_NP_COMMERCE2_PRODUCT_SHORT_DESCRIPTION_LEN	1024	Size of product details (short description)
SCE_NP_COMMERCE2_PRODUCT_LONG_DESCRIPTION_LEN	4000	Size of product details (long description)
SCE_NP_COMMERCE2_PRODUCT_LEGAL_DESCRIPTION_LEN	4000	Size of legal text
SCE_NP_COMMERCE2_SKU_ID_LEN	56	Size of SKU ID
SCE_NP_COMMERCE2_SKU_NAME_LEN	180	Size of SKU name
SCE_NP_COMMERCE2_URL_LEN	256	Size of URL
SCE_NP_COMMERCE2_RATING_SYSTEM_ID_LEN	16	Size of rating system ID
SCE_NP_COMMERCE2_RATING_DESCRIPTION_LEN	60	Size of rating descriptor text

Description

These are the maximum sizes of the strings that can be obtained by the NP IN-GAME Commerce 2 library.

The sizes indicated in the table do not include the terminating NULL character. All string data obtained by the NP IN-GAME Commerce 2 library is terminated with a NULL character.

SCE CONFIDENTIAL

Return Codes

Error codes returned by the NP IN-GAME Commerce 2 library

Definition

Value	(Number)	Description
SCE_NP_COMMERCE2_ERROR_NOT_INITIALIZED	0x80550f01	Not initialized
SCE_NP_COMMERCE2_ERROR_ALREADY_INITIALIZED	0x80550f02	Already initialized
SCE_NP_COMMERCE2_ERROR_INVALID_ARGUMENT	0x80550f03	Invalid argument
SCE_NP_COMMERCE2_ERROR_UNSUPPORTED_VERSION	0x80550f04	Unsupported version
SCE_NP_COMMERCE2_ERROR_CTX_MAX	0x80550f05	Maximum number of contexts that can be created was exceeded
SCE_NP_COMMERCE2_ERROR_INVALID_INDEX	0x80550f06	Specified index is invalid
SCE_NP_COMMERCE2_ERROR_INVALID_SKUID	0x80550f07	Specified SKU ID is invalid
SCE_NP_COMMERCE2_ERROR_INVALID_SKU_NUM	0x80550f08	Specified number of SKUs is invalid
SCE_NP_COMMERCE2_ERROR_OUT_OF_MEMORY	0x80550f0b	Memory could not be allocated
SCE_NP_COMMERCE2_ERROR_CTX_NOT_FOUND	0x80550f0c	Specified context or request does not exist
SCE_NP_COMMERCE2_ERROR_CTXID_NOT_AVAILABLE	0x80550f0d	Context ID could not be obtained
SCE_NP_COMMERCE2_ERROR_REQ_NOT_FOUND	0x80550f0e	Specified request does not exist
SCE_NP_COMMERCE2_ERROR_REQID_NOT_AVAILABLE	0x80550f0f	Request ID could not be obtained
SCE_NP_COMMERCE2_ERROR_ABORTED	0x80550f10	Request was aborted with <code>sceNpCommerce2AbortReq()</code>
SCE_NP_COMMERCE2_ERROR_RESPONSE_BUF_TOO_SMALL	0x80550f12	Specified buffer for obtaining result is not large enough
SCE_NP_COMMERCE2_ERROR_COULD_NOT_RECV_WHOLE_RESPONSE_DATA	0x80550f13	Not all data could be obtained from the server
SCE_NP_COMMERCE2_ERROR_INVALID_RESULT_DATA	0x80550f14	Data passed to the take out function does not match the data content
SCE_NP_COMMERCE2_ERROR_UNKNOWN	0x80550f15	Unknown error
SCE_NP_COMMERCE2_ERROR_SERVER_MAINTENANCE	0x80550f16	Server is under maintenance
SCE_NP_COMMERCE2_ERROR_SERVER_UNKNOWN	0x80550f17	Unexpected situation on the server
SCE_NP_COMMERCE2_ERROR_INSUFFICIENT_BUF_SIZE	0x80550f18	Specified buffer size is too small
SCE_NP_COMMERCE2_ERROR_REQ_MAX	0x80550f19	Tried to create a commerce 2 request to exceed the maximum number
SCE_NP_COMMERCE2_ERROR_HTTP_POOL_TOO_SHORT	0x80550f70	Not enough space left in the libhttp memory pool
SCE_NP_COMMERCE2_ERROR_SSL_POOL_TOO_SHORT	0x80550f71	Not enough space left in the libssl memory pool
SCE_NP_COMMERCE2_ERROR_NOT_SIGNIN	0x80550f72	Not signed in to PSN SM

SCE CONFIDENTIAL

Value	(Number)	Description
SCE_NP_COMMERCE2_ERROR_DATA_NOT_FOUND	0x80550f87	The target data does not exist in the passed buffer
SCE_NP_COMMERCE2_SERVER_ERROR_BAD_REQUEST	0x80551001	Invalid parameter sent to the server. The value specified for the argument may be invalid.
SCE_NP_COMMERCE2_SM_ERROR_BUF_TOO_SMALL	0x80550fc1	Buffer size specified in <code>sceNpCommerce2StartEmptyStoreCheck()</code> is too small
SCE_NP_COMMERCE2_SC_ERROR_INVALID_RESPONSE	0x80550fe1	Data received from the server with <code>sceNpCommerce2StartEmptyStoreCheck()</code> is invalid
SCE_NP_COMMERCE2_SERVER_ERROR_UNKNOWN_ERROR	0x80551002	Unknown error in the server
SCE_NP_COMMERCE2_SERVER_ERROR_SESSION_EXPIRED	0x80551005	Validity of created session expired
SCE_NP_COMMERCE2_SERVER_ERROR_ACCESS_PERMISSION_DENIED	0x80551007	Access rights error
SCE_NP_COMMERCE2_SERVER_ERROR_NO_SUCH_CATEGORY	0x80551010	Specified category does not exist
SCE_NP_COMMERCE2_SERVER_ERROR_NO_SUCH_PRODUCT	0x80551011	Specified product does not exist
SCE_NP_COMMERCE2_SERVER_ERROR_NOT_ELIGIBILITY	0x80551013	Eligibility rules error
SCE_NP_COMMERCE2_SERVER_ERROR_INVALID_SKU	0x8055101a	Specified SKU ID is invalid
SCE_NP_COMMERCE2_SERVER_ERROR_ACCOUNT_SUSPENDED1	0x8055101b	Account is suspended
SCE_NP_COMMERCE2_SERVER_ERROR_ACCOUNT_SUSPENDED2	0x8055101c	Account is suspended
SCE_NP_COMMERCE2_SERVER_ERROR_OVER_SPENDING_LIMIT	0x80551020	Spending limit has been exceeded
SCE_NP_COMMERCE2_SERVER_ERROR_INVALID_VOUCHER	0x8055102f	Incorrect code number was input for the promotion code input processing
SCE_NP_COMMERCE2_SERVER_ERROR_VOUCHER_ALREADY_CONSUMED	0x80551030	The input promotion code has already been used up
SCE_NP_COMMERCE2_SERVER_ERROR_EXCEEDS_AGE_LIMIT_IN_BROWSING	0x80551039	Access to this product catalog is denied due to age restrictions