Message Dialog Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Document serial number: 000004892117

Document serial number: 000004892117

Table of Contents

Message Dialog Call

sceMsgDialogParamInit

Macro for call parameter initialization

Definition

```
#include <message_dialog.h>
static inline
void sceMsgDialogParamInit(SceMsgDialogParam *param)
{
        sceClibMemset( param, 0x0, sizeof(SceMsgDialogParam) );
        _sceCommonDialogSetMagicNumber( &param->commonParam );
        param->sdkVersion = SCE_PSP2_SDK_VERSION;
}
```

Arguments

param Call parameter

Return Values

None

Description

This is a macro function for initializing the Message Dialog call parameters.

Before performing the various call parameter settings, be sure to use this macro to execute structure initialization. The appropriate SDK version is set at the same time.

Examples

```
SceMsgDialogParam param;
SceMsgDialogSystemMessageParam sysMsgParam;

sceMsgDialogParamInit( &param );
sceClibMemset( &sysMsgParam, 0, sizeof(sysMsgParam) );
param.mode = SCE_MSG_DIALOG_MODE_SYSTEM_MSG;
param.sysMsgParam = &sysMsgParam;
param.sysMsgParam->sysMsgType = SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT;

if ( sceMsgDialogInit( &param ) < 0 ) {
        // error
}
```

See Also

SceMsgDialogParam

sceMsgDialogInit

Call various functions of Message Dialog

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogInit(
        const SceMsgDialogParam *param,
)
```

Arguments

param Call parameter

Return Values

Returns SCE_OK(0) as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
SCE_COMMON_DIALOG_ERROR_BUSY	0x80020401	The Common Dialog feature is already being called (details below)
SCE_COMMON_DIALOG_ERROR_NULL	0x80020402	NULL was specified for the argument *param*
SCE_COMMON_DIALOG_ERROR_ UNEXPECTED_FATAL	0x8002047F	Internal error

Description

This function calls the Message Dialog feature.

This function can be called only when other Common Dialog features are not called (including the Message Dialog feature proper). If this function is called at times other than the above, SCE_COMMON_DIALOG_ERROR_BUSY is returned. When calling this function is successful, the operation status immediately changes to SCE_COMMON_DIALOG_STATUS_RUNNING. For details on the operation statuses, refer to the sceMsgDialogGetStatus() section.

In *param*, specify the call parameter structure for which the operation mode, the display character string, etc. was set.

Be sure to set the various values after performing initialization of *param* with the sceMsgDialogParamInit() macro first.

The *param* instance need not be allocated after this function is called, but some of the pointer reference parameters must be held until calling of the Message Dialog feature is ended by calling sceMsgDialogTerm().

If the content of *param* is invalid, a parameter error will occur, and SCE_MSG_DIALOG_ERROR_PARAM will return to SceMsgDialogResult.*result*, which is obtained with sceMsgDialogGetResult(). As an exception, only when *param*.*commonParam*.*bgColor* (background color) is not set to NULL, the API immediately returns SCE_MSG_DIALOG_ERROR_PARAM, resulting in a failure of the call.

This function is multithread safe.

Examples

```
SceMsgDialogParam param;
SceMsgDialogUserMessageParam userMsgParam;
static const SceChar8 msg_hello[] = "Hello world!";

sceMsgDialogParamInit( &param );
sceClibMemset( &userMsgParam, 0, sizeof(userMsgParam) );
param.mode = SCE_MSG_DIALOG_MODE_USER_MSG;
param.userMsgParam = &userMsgParam;
param.userMsgParam->msg = msg_hello;
param.userMsgParam->buttonType = SCE_MSG_DIALOG_BUTTON_TYPE_OK;

if ( sceMsgDialogInit( &param ) < 0 ) {
        // error
}
```

See Also

SceMsgDialogParam, sceMsgDialogParamInit(), sceMsgDialogGetStatus(), sceMsgDialogGetResult()

SceMsgDialogParam

Structure for calling Message Dialog

Definition

```
#include <message_dialog.h>
typedef struct SceMsgDialogParam {
        SceUInt32 sdkVersion;
        SceCommonDialogParam commonParam;
        SceMsgDialogMode mode;
        SceMsgDialogUserMessageParam *userMsgParam;
        SceMsgDialogSystemMessageParam *sysMsgParam;
        SceMsgDialogErrorCodeParam *errorCodeParam;
        SceMsgDialogProgressBarParam *progBarParam;
        SceMsgDialogEnvFlag flag;
        SceChar8 reserved[32];
} SceMsgDialogParam;
```

Members

sdkVersion	SDK version
commonParam	Common parameters
mode	Operation mode (details below)
userMsgParam	User specified message display parameter
sysMsgParam	System defined message display parameter
errorCodeParam	Error code display parameter
progBarParam	Progress bar display parameter
flag	Environmental setting flag (details below)
reserved	Reserved area (fill with all 0s)

Description

This is a structure passed to `sceMsgDialogInit()` to display Message Dialog. Use `sceMsgDialogParamInit()` to initialize it.

Specify the SDK version in *sdkVersion*. An appropriate value is input when the structure is initialized with `sceMsgDialogParamInit()`.

The display state of info bar and dimmer color can be specified with *commonParam*; however, the background color cannot be specified. For details, refer to the "Common Dialog Reference" document.

Specify the operation mode in *mode*. One of the following values is input.

Value	(Number)	Description
SCE_MSG_DIALOG_MODE_USER_MSG	1	Displays a user specified message
SCE_MSG_DIALOG_MODE_SYSTEM_MSG	2	Displays a system defined message
SCE_MSG_DIALOG_MODE_ERROR_CODE	3	Displays an error code
SCE_MSG_DIALOG_MODE_PROGRESS_BAR	4	Displays a progress bar

The structure that stores the user specified message display settings is passed to *userMsgParam*. For details, refer to the SceMsgDialogUserMessageParam section. If an operation mode other than SCE_MSG_DIALOG_MODE_USER_MSG was specified, NULL must be set for this value.

The structure that stores the system defined message display settings is passed to *sysMsgParam*. For details, refer to the SceMsgDialogSystemMessageParam section. If an operation mode other than SCE_MSG_DIALOG_MODE_SYSTEM_MSG was specified, NULL must be set for this value.

The structure that stores the error code display settings is passed to *errorCodeParam*. For details, refer to the SceMsgDialogErrorCodeParam section. If an operation mode other than SCE_MSG_DIALOG_MODE_ERROR_CODE was specified, NULL must be set for this value.

The structure that stores the progress bar display settings is passed to *progBarParam*. For details, refer to the SceMsgDialogProgressBarParam section. If an operation mode other than SCE_MSG_DIALOG_MODE_PROGRESS_BAR was specified, NULL must be set for this value.

The operating environment of Message Dialog is specified in *flag*. The following option can be specified in this SDK.

Value	(Number)	Description
SCE_MSG_DIALOG_ENV_FLAG_DEFAULT	0	Default setting

reserved is a reserved area for future function expansion. It must be filled with all 0s.

See Also

sceMsgDialogParamInit(), SceMsgDialogUserMessageParam, SceMsgDialogSystemMessageParam, SceMsgDialogErrorCodeParam, SceMsgDialogProgressBarParam

SceMsgDialogUserMessageParam

Structure for user specified message display

Definition

```
#include <message_dialog.h>
typedef struct SceMsgDialogUserMessageParam {
        SceMsgDialogButtonType buttonType;
        const SceChar8 *msg;
        SceMsgDialogButtonsParam *buttonParam;
        SceChar8 reserved[28];
} SceMsgDialogUserMessageParam;
```

Members

buttonType	Specifies the type of button displayed on the dialog (details below)
msg	Arbitrary character string displayed on the screen (NULL termination, UTF-8)
buttonParam	Specifies the button's contents (details below)
reserved	Reserved area (fill with all 0s)

Description

This structure is used to perform the setting for the user specified message display. It is used when calling Message Dialog in the SCE_MSG_DIALOG_MODE_USER_MSG mode. Fill it with all 0s during initialization.

One of the following values is input in *buttonType*.

Value	(Number)	Description
SCE_MSG_DIALOG_BUTTON_TYPE_OK	0	Displays 1 button, the **OK** button
SCE_MSG_DIALOG_BUTTON_TYPE_YESNO	1	Displays 2 buttons, the **Yes** button and the **No** button
SCE_MSG_DIALOG_BUTTON_TYPE_NONE	2	Does not display buttons
SCE_MSG_DIALOG_BUTTON_TYPE_OK_CANCEL	3	Displays 2 buttons, the **OK** button and the **Cancel** button
SCE_MSG_DIALOG_BUTTON_TYPE_CANCEL	4	Displays 1 button, the **Cancel** button (Busy indicator is automatically displayed.)
SCE_MSG_DIALOG_BUTTON_TYPE_3BUTTONS	5	Displays 3 buttons with user-specified character strings

Message Dialog for which SCE_MSG_DIALOG_BUTTON_TYPE_NONE has been specified for *buttonType* cannot be closed by user operation. It must be closed from the caller side by using sceMsgDialogClose() at an arbitrary timing.

In *msg*, specify the character string to be displayed on the screen with UTF-8. The character string termination must be NULL. Up to six line breaks can be inserted and the maximum size is SCE_MSG_DIALOG_USER_MSG_SIZE. When a string exceeding the maximum size is specified, the exceeding section will be cut off.

In *buttonParam*, you can specify the contents of each button when specifying SCE_MSG_DIALOG_BUTTON_TYPE_3BUTTONS in *buttonType*. If you have specified anything other than SCE_MSG_DIALOG_BUTTON_TYPE_3BUTTONS, it will need to be NULL.

reserved is a reserved area for future function expansion. It must be filled with all 0s.

See Also

`sceMsgDialogInit()`, `sceMsgDialogClose()`
`SceMsgDialogParam`, `SceMsgDialogButtonsParam`

SceMsgDialogSystemMessageParam

Structure for system defined message display

Definition

```
#include <message_dialog.h>
typedef struct SceMsgDialogSystemMessageParam {
        SceMsgDialogSystemMessageType sysMsgType;
        SceInt32 value;
        SceChar8 reserved[32];
} SceMsgDialogSystemMessageParam;
```

Members

sysMsgType	Specifies the type of message to be displayed (details below)
value	Specifies a value during a message call requiring an additional value
reserved	Reserved area (fill with all 0s)

Description

This structure is used to perform the setting for the system defined message display. It is used when calling Message Dialog in the SCE_MSG_DIALOG_MODE_SYSTEM_MSG mode. Fill it with all 0s during initialization.

One of the following values is input in *sysMsgType*.

Value	(Number)	Description
SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT	1	Displays "Please wait."
SCE_MSG_DIALOG_SYSMSG_TYPE_NOSPACE	2	Displays "There is not enough free space on the memory card. To continue using the application, you must create at least XXXXKB of free space. Press the PS button to pause this application, and then delete other applications or content."
SCE_MSG_DIALOG_SYSMSG_TYPE_MAGNETIC_CALIBRATION	3	Displays "Move away from the source of interference, or adjust the compass by moving your PS Vita system as shown below."
SCE_MSG_DIALOG_SYSMSG_TYPE_MIC_DISABLED	4	Disabled (Be sure to use SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_MIC_DISABLED from SDK 1.000)
SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT_SMALL	5	Displays "Please wait." in a small-sized dialog box.
SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT_CANCEL	6	Displays "Please wait..." with a **Cancel** button
SCE_MSG_DIALOG_SYSMSG_TYPE_NOSPACE_CONTINUABLE	9	Displays "There is not enough free space on the memory card. To save your progress in the application, you must create at least XXXXKB of free space. To create the free space, press the PS button to pause this application, and then delete other applications or content."

Value	(Number)	Description
SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_OBTAINING	10	Displays "Obtaining location data... Please wait."
SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE	11	Displays appropriate message indicating the failure to obtain location information according to the situation (see details below)
SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE_RETRY	12	Displays appropriate message (with a **Retry** button) indicating the failure to obtain location information according to the situation (see details below)
SCE_MSG_DIALOG_SYSMSG_TYPE_PATCH_FOUND	13	Displays a notification of the availability of application update files (see details below)
SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_MIC_DISABLED	100	Displays either "Enable the microphone. Press and hold the PS button until the menu is displayed, and then verify that [Disable Microphone] is off. Also, stop using the microphone feature in any other applications." or "No microphone connected."
SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_WIFI_REQUIRED_OPERATION	101	Displays "You must use Wi-Fi to do this."
SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_WIFI_REQUIRED_APPLICATION	102	Displays "You must use Wi-Fi to use this application."
SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_EMPTY_STORE	103	Displays "No content is available yet."
SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_PSN_AGE_RESTRICTION	104	Displays "You cannot use PSN℠ features in this application due to age restrictions."
SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_PSN_CHAT_RESTRICTION	105	Displays "Use of this application's chat and messaging features is not allowed for your account."
SCE_MSG_DIALOG_SYSMSG_TYPE_TRC_MIC_DISABLED_CONTINUABLE	106	Displays either "The microphone is disabled. To use the microphone, press and hold the PS button until the menu is displayed, and then verify that [Disable Microphone] is off. Also, stop using the microphone feature in any other applications." or "No microphone connected."

System defined messages will be displayed automatically in multiple languages according to the language setting of the PlayStation®Vita.

Message Dialog for which `SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT` and `SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT_SMALL` has been specified for *`sysMsgType`* cannot be closed by user operation. It must be closed from the caller side by using `sceMsgDialogClose()` at an arbitrary timing.

If `SCE_MSG_DIALOG_SYSMSG_TYPE_NOSPACE` and `SCE_MSG_DIALOG_SYSMSG_TYPE_NOSPACE_CONTINUABLE` are specified in *`sysMsgType`*, the insufficient space requested by the last insufficient file system free space error occurring will be formatted into appropriate units and displayed in the message. The application cannot specify the insufficient space displayed. If the application's save data drive (savedata0:) is mounted to the PlayStation®Vita card, an error notification ("An error has occurred. (C2-13322-1, 0x80100aa0)") will be displayed instead of a message.

File system free space will run out when saving save data with the application utility library, or when saving a database with libSceSqlite; therefore, display this message when file system free space runs out with these libraries.

When specifying either SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE or SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE_RETRY to *sysMsgType*, the message that is automatically displayed will differ according to the state of the GPS device or Wi-Fi device.

SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE displays the following messages.

GPS/Wi-Fi State	Message
GPS device exists	Could not obtain location data. You must connect the PS Vita system to the Internet or go outdoors where interference will be minimal. It may take several minutes to obtain location data without an Internet connection.
GPS device does not exist, Wi-Fi connection is enabled	Could not obtain location data. You must connect the PS Vita system to the Internet. Location data cannot be obtained in areas without Wi-Fi location services.
GPS device does not exist, Wi-Fi connection is disabled	To obtain location data, you must go to the home screen and select [Settings] > [Network] > [Wi-Fi Settings] and then turn [Wi-Fi] on.

SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE_RETRY displays the following messages.

GPS/Wi-Fi State	Message
GPS device exists	Could not obtain location data. Connect the PS Vita system to the Internet or go outdoors where interference will be minimal, and then select [Retry]. It may take several minutes to obtain location data without an Internet connection.
GPS device does not exist, Wi-Fi connection is enabled	Could not obtain location data. Connect the PS Vita system to the Internet, and then select [Retry]. Location data cannot be obtained in areas without Wi-Fi location services.
GPS device does not exist, Wi-Fi connection is disabled	To obtain location data, go to the home screen, select [Settings] > [Network] > [Wi-Fi Settings], turn on [Wi-Fi] and then select [Retry].

When SCE_MSG_DIALOG_SYSMSG_TYPE_PATCH_FOUND is specified in *sysMsgType*, the message displayed will change automatically depending on the state of the download of the update files that can be applied to the relevant application. Also refer to the "GameUpdate Library Overview" document concerning this method of using messages.

State of the Download of Update Files	Message
Not yet downloaded /downloaded	A new version of the application is available. You must go back to the LiveArea™ screen and then update the application. You can update the application using the update icon on the LiveArea™ screen.
Download in progress	Downloading the application update file... After the download is complete, you must go back to the LiveArea™ screen and then update the application. You can update the application using the update icon on the LiveArea™ screen.

For the cases where `SCE_MSG_DIALOG_SYSMSG_TYPE_MAGNETIC_CALIBRATION` is specified for *`sysMsgType`*, refer to the "Implementation Guidelines for Magnetometer Sensor Calibration Message Dialog" chapter in the "Programming Startup Guide" document.

Specify 0 in *`value`*.

`reserved` is a reserved area for future function expansion. It must be filled with all 0s.

See Also

`sceMsgDialogInit()`, `sceMsgDialogClose()`

`SceMsgDialogParam`, `SceMsgDialogProgressBarParam`

SceMsgDialogErrorCodeParam

Structure for error code display

Definition

```
#include <message_dialog.h>
typedef struct SceMsgDialogErrorCodeParam {
        SceInt32 errorCode;
        SceChar8 reserved[32];
} SceMsgDialogErrorCodeParam;
```

Members

errorCode	Specifies the error code to be displayed
reserved	Reserved area (fill with all 0s)

Description

This structure is used to perform the setting for the error code display. It is used when calling Message Dialog in the SCE_MSG_DIALOG_MODE_ERROR_CODE mode. Fill it with all 0s during initialization.

Specify the error values returned by the various library APIs in *errorCode*. The error values are converted into the appropriate expression and displayed. The displayed dialog has one button, the **OK** button.

reserved is a reserved area for future function expansion. It must be filled with all 0s.

See Also

sceMsgDialogInit(),SceMsgDialogParam

SceMsgDialogProgressBarParam

Structure for progress bar display

Definition

```
#include <message_dialog.h>
typedef struct SceMsgDialogProgressBarParam {
        SceMsgDialogProgressBarType barType;
        SceMsgDialogSystemMessageParam sysMsgParam;
        const SceChar8 *msg;
        SceChar8 reserved[32];
} SceMsgDialogProgressBarParam;
```

Members

barType	Specifies the type of progress bar
sysMsgParam	Specifies the type of message to be displayed (details below)
msg	Arbitrary character string to be displayed on the screen (NULL termination, UTF-8)
reserved	Reserved area (fill with all 0s)

Description

This structure is used to perform the setting for the progress bar display. It is used when calling Message Dialog in the SCE_MSG_DIALOG_MODE_PROGRESS_BAR mode. Fill it with all 0s during initialization.

Message Dialog for which SCE_MSG_DIALOG_MODE_PROGRESS_BAR has been specified cannot be closed by user operation. It must be closed from the caller side by using sceMsgDialogClose() at an arbitrary timing.

The following value is input in *barType*.

Value	(Number)	Description
SCE_MSG_DIALOG_PROGRESSBAR_TYPE_PERCENTAGE	0	Displays a progress bar expressed as a percentage.

Specify *sysMsgParam* to display a system defined message at the same time as the progress bar. For details on the structure itself, refer to the SceMsgDialogSystemMessageParam section.

The following value can be specified in *sysMsgParam.sysMsgType*.

Value	(Number)	Description
SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT	1	Displays " Please wait."
SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT_SMALL	5	Displays " Please wait."
SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT_CANCEL	6	Displays " Please wait." with a **Cancel** button

Specify *msg* to display a user specified character string at the same time as the progress bar. Specify the character string to be displayed on the screen with UTF-8. The character string termination must be NULL. Up to six line breaks can be inserted and the maximum size is SCE_MSG_DIALOG_USER_MSG_SIZE. When a string exceeding the maximum size is specified, the exceeding section will be cut off.

**sysMsgParam* and *msg* cannot be specified simultaneously. If a value is specified in *sysMsgParam.sysMsgType*, NULL must be specified in *msg*. Also, if no value is set in *sysMsgParam.sysMsgType*, a parameter error occurs even if NULL was set to *msg*.

reserved is a reserved area for future function expansion. It must be filled with all 0s.

See Also

`sceMsgDialogInit(),sceMsgDialogClose()`
`SceMsgDialogParam`

SceMsgDialogButtonsParam

Structure for setting displayed contents of user-specified character string buttons

Definition

```
#include <message_dialog.h>
typedef struct SceMsgDialogButtonsParam {
        const SceChar8 *msg1;
        SceMsgDialogFontSize fontSize1;
        const SceChar8 *msg2;
        SceMsgDialogFontSize fontSize2;
        const SceChar8 *msg3;
        SceMsgDialogFontSize fontSize3;
        SceChar8 reserved[32];
} SceMsgDialogButtonsParam;
```

Members

msg1	Arbitrary character string displayed on **Button 1** (NULL termination, UTF-8)
fontSize1	**Button 1** font size (details below)
msg2	Arbitrary character string displayed on **Button 2** (NULL termination, UTF-8)
fontSize2	**Button 2** font size (details below)
msg3	Arbitrary character string displayed on **Button 3** (NULL termination, UTF-8)
fontSize3	**Button 3** font size (details below)
reserved	Reserved area (fill with all 0s)

Description

This is a structure for setting the contents displayed on user-specified character string buttons. Use it when specifying SCE_MSG_DIALOG_BUTTON_TYPE_3BUTTONS in SCE_MSG_DIALOG_MODE_USER_MSG mode. Fill it with all 0s during initialization.

In *msg1* to *3*, specify the character strings to be displayed on each button in UTF-8. The termination of the character strings must be NULL. Starting a new line is not possible. Maximum length is SCE_MSG_DIALOG_BUTTON_MSG_SIZE. If you specify a character string exceeding the maximum size, the part in excess of the maximum size will be removed. Also, character strings that do not exceed the maximum size but cannot be displayed completely on a button will be abbreviated with "...".

Specify each button's font size in *fontSize1* to *3*. Input one of the following values.

Value	(Number)	Description
SCE_MSG_DIALOG_FONT_SIZE_DEFAULT	0	Default font size
SCE_MSG_DIALOG_FONT_SIZE_SMALL	1	Smaller font size

Normally, specify SCE_MSG_DIALOG_FONT_SIZE_DEFAULT for better visibility. Only try SCE_MSG_DIALOG_FONT_SIZE_SMALL when specifying a character string too long to fit on the button.

reserved is a reserved area for future function expansion. It must be filled with all 0s.

See Also

SceMsgDialogUserMessageParam

Obtaining Operation Status

sceMsgDialogGetStatus

Get operation status of Message Dialog

Definition

```
#include <message_dialog.h>
SceCommonDialogStatus sceMsgDialogGetStatus()
```

Arguments

None

Return Values

Returns one of the following operation statuses as the value of the function.

Value	(Number)	Description
SCE_COMMON_DIALOG_STATUS_NONE	0x0	Message Dialog is not running
SCE_COMMON_DIALOG_STATUS_RUNNING	0x1	Message Dialog is running
SCE_COMMON_DIALOG_STATUS_FINISHED	0x2	Message Dialog operation has finished

Description

This function gets the operation status of Message Dialog.

The initial operation status is SCE_COMMON_DIALOG_STATUS_NONE.

When calling sceMsgDialogInit() is successful, the operation status immediately changes to SCE_COMMON_DIALOG_STATUS_RUNNING.

When Message Dialog is closed either through user operation or by calling sceMsgDialogClose() or sceMsgDialogAbort(), the operation status changes to SCE_COMMON_DIALOG_STATUS_FINISHED after a while. sceMsgDialogGetResult() and sceMsgDialogTerm() can be called only while the operation status is SCE_COMMON_DIALOG_STATUS_FINISHED.

When sceMsgDialogTerm() is called, the operation status immediately changes to SCE_COMMON_DIALOG_STATUS_NONE.

This function is multithread safe.

Examples

```
SceCommonDialogStatus stat = sceMsgDialogGetStatus();
```

See Also

sceMsgDialogInit(), sceMsgDialogClose(), sceMsgDialogAbort(), sceMsgDialogGetResult(), sceMsgDialogTerm()

Updating Display Information

sceMsgDialogProgressBarInc

Increase Progress Bar Rate

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogProgressBarInc(
        SceMsgDialogProgressBarTarget target,
        SceUInt32 delta
)
```

Arguments

target	Target to be increased
delta	Increase value

Return Values

Returns SCE_OK(0) as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
SCE_COMMON_DIALOG_ERROR_NOT_RUNNING	0x80020404	Called during other than appropriate time (details below)
SCE_COMMON_DIALOG_ERROR_NOT_SUPPORTED	0x80020405	Not support the current operation mode (details below)
SCE_MSG_DIALOG_ERROR_PARAM	0x80100a01	Parameter error (details below)
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

Description

This function increases the progress bar rate of Message Dialog displaying the progress bar. The label indicating the progress rate is updated according to the increasing progress rate, and the progress bar extends smoothly with an animated image.

This function can be called only when the operation mode is SCE_MSG_DIALOG_MODE_PROGRESS_BAR.

This function can be called only when the operation status is SCE_COMMON_DIALOG_STATUS_RUNNING.

Specify the target of which the progress rate is to be increased to *target*. The following value is set.

Value	(Number)	Description
SCE_MSG_DIALOG_PROGRESSBAR_TARGET_BAR_DEFAULT	0	Progress bar expressed as a percentage

SCE_MSG_DIALOG_ERROR_PARAM will be returned as the value of the function when values other than the above are specified.

Specify the increase value of the progress rate to *delta*. The initial value of the progress bar rate, which is acquired by specifying SCE_MSG_DIALOG_PROGRESSBAR_TYPE_PERCENTAGE to SceMsgDialogProgressBarParam.barType, is 0%. The percentage is increased by the value specified to delta, and the maximum value is 100%. Even if it is attempted to increase the progress rate after reaching 100%, the rate will not be changed.

This function is multithread safe.

Examples

```
// a thread different from the thread that has called sceMsgDialogInit()

for( SceUInt32 rate=0; rate<=100; rate+10 ) {
        sceMsgDialogProgressBarInc(
              SCE_MSG_DIALOG_PROGRESSBAR_TARGET_BAR_DEFAULT,
              10 );
}

sceMsgDialogClose();
```

See Also

sceMsgDialogInit(),sceMsgDialogClose()

sceMsgDialogProgressBarSetValue

Specify Progress Bar Rate

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogProgressBarSetValue(
        SceMsgDialogProgressBarTarget target,
        SceUInt32 rate
)
```

Arguments

target	Target to be specified
rate	Value to be specified

Return Values

Returns SCE_OK(0) as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
SCE_COMMON_DIALOG_ERROR_NOT_RUNNING	0x80020404	Called during other than appropriate time (details below)
SCE_COMMON_DIALOG_ERROR_NOT_SUPPORTED	0x80020405	Not support the current operation mode (details below)
SCE_MSG_DIALOG_ERROR_PARAM	0x80100a01	Parameter error (details below)
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

Description

This function specifies the progress bar rate of Message Dialog displaying the progress bar. The label indicating the progress rate is updated according to the specified progress rate, and the length of the progress bar immediately changes to a suitable length without an animated image.

This function can be called only when the operation mode is SCE_MSG_DIALOG_MODE_PROGRESS_BAR.

This function can be called only when the operation status is SCE_COMMON_DIALOG_STATUS_RUNNING.

Specify the target of which the progress rate is to be specified to *target*. The following value is input.

Value	(Number)	Description
SCE_MSG_DIALOG_PROGRESSBAR_TARGET_BAR_DEFAULT	0	Progress bar expressed as a percentage

SCE_MSG_DIALOG_ERROR_PARAM will be returned as the value of the function when values other than the above are specified.

Specify the progress rate to be specified to *rate*. The maximum value of the progress bar rate, which is acquired by specifying SCE_MSG_DIALOG_PROGRESSBAR_TYPE_PERCENTAGE to SceMsgDialogProgressBarParam.barType, is 100%. If a value larger than this is specified, the value is automatically rounded to 100%.

This function is multithread safe.

Examples

```
SceMsgDialogParam param;
SceMsgDialogProgressBarParam progBarParam;

sceMsgDialogParamInit( &param );
sceClibMemset( &progBarParam, 0, sizeof(progBarParam) );
param.mode = SCE_MSG_DIALOG_MODE_PROGRESS_BAR;
param.progBarParam = &progBarParam;
param.progBarParam->sysMsgParam.sysMsgType = SCE_MSG_DIALOG_SYSMSG_TYPE_WAIT;

if ( sceMsgDialogInit( &param ) < 0 ) {
        // error
}

sceMsgDialogProgressBarSetValue(
              SCE_MSG_DIALOG_PROGRESSBAR_TARGET_BAR_DEFAULT,
              50 );  // Start from 50%
}
```

See Also

sceMsgDialogInit()

sceMsgDialogProgressBarSetMsg

Specify Progress Bar Character String

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogProgressBarSetMsg(
        SceMsgDialogProgressBarTarget target,
        const SceChar8 *msg
)
```

Arguments

target	Target to be specified
msg	Specified character string

Return Values

Returns SCE_OK(0) as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
SCE_COMMON_DIALOG_ERROR_NOT_RUNNING	0x80020404	Called during other than appropriate time (details below)
SCE_COMMON_DIALOG_ERROR_NOT_SUPPORTED	0x80020405	Not support the current operation mode (details below)
SCE_MSG_DIALOG_ERROR_PARAM	0x80100a01	Parameter error (details below)
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

Description

This function specifies the character string shown on the progress bar screen of Message Dialog displaying the progress bar. The character string on the progress bar screen is updated according to the specified character string.

This function can be called only when the operation mode is SCE_MSG_DIALOG_MODE_PROGRESS_BAR.

This function can be called only when the operation status is SCE_COMMON_DIALOG_STATUS_RUNNING.

Specify the target to which the character string is to be specified to *target*. The following value is set.

Value	(Number)	Description
SCE_MSG_DIALOG_PROGRESSBAR_TARGET_BAR_DEFAULT	0	Progress bar expressed as a percentage

SCE_MSG_DIALOG_ERROR_PARAM will be returned as the value of the function when values other than the above are specified.

In *msg*, specify the character string to be displayed on the screen with UTF-8. The character string termination must be NULL. Up to six line breaks can be inserted and the maximum size is SCE_MSG_DIALOG_USER_MSG_SIZE. When a string exceeding the maximum size is specified, the exceeding section will be cut off.

The character string can be updated multiple times while the progress bar is displaying. The update of the character string, however, is performed asynchronously with the function call; thus, if the update is performed quickly and continuously, it is not guaranteed that each updated character string is always displayed.

This function is multithread safe.

Examples

```
// a thread different from the thread that has called sceMsgDialogInit()

char msg[SCE_MSG_DIALOG_USER_MSG_SIZE];

for( SceUInt32 rate=0; rate<=100; rate+=10 ) {
        sceMsgDialogProgressBarSetMsg(
              SCE_MSG_DIALOG_PROGRESSBAR_TARGET_BAR_DEFAULT,
              10 );

        snprintf( msg, sizeof(msg), "Downloading file%d", rate );
        sceMsgDialogProgressBarSetValue(
              SCE_MSG_DIALOG_PROGRESSBAR_TARGET_BAR_DEFAULT,
              msg );
}

sceMsgDialogClose();
```

See Also

sceMsgDialogInit()

Closing Message Dialog

sceMsgDialogClose

Close Message Dialog

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogClose()
```

Arguments

None

Return Values

Returns SCE_OK(0) as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
SCE_COMMON_DIALOG_ERROR_NOT_RUNNING	0x80020404	Called during other than appropriate time (details below)
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

Description

This function closes Message Dialog.

This function can be called only while the operation status of Message Dialog is SCE_COMMON_DIALOG_STATUS_RUNNING. If this function is called at times other than the above, SCE_COMMON_DIALOG_ERROR_NOT_RUNNING is returned.

If calling this function is successful, the operation status changes to SCE_COMMON_DIALOG_STATUS_FINISHED after the finish processing.

For details on the operation statuses, refer to the sceMsgDialogGetStatus() section.

When Message Dialog is closed with this function, calling sceMsgDialogGetResult() returns the following as the call result.

```
SceMsgDialogResult.result    : 0
SceMsgDialogResult.buttonId  : SCE_MSG_DIALOG_BUTTON_ID_INVALID
```

Since a user specified Message Dialog for which SCE_MSG_DIALOG_BUTTON_TYPE_NONE was specified as the button type, or some of system defined message dialogs without the buttons, cannot be closed through user operation, they must be closed with this function at an arbitrary timing. This function can be used also for other Message Dialogs that can be closed through user operation.

This function is multithread safe.

Examples

```
SceCommonDialogStatus stat;

while(1) {
        stat = sceMsgDialogGetStatus();
        if( stat == SCE_COMMON_DIALOG_STATUS_RUNNING ) {
              if( need_close ) {
                    sceMsgDialogClose();
                    break;
              }
        }
        else if( stat == SCE_COMMON_DIALOG_STATUS_FINISHED ) {
              sceMsgDialogTerm();
              break;
        }
}
```

See Also

sceMsgDialogGetStatus(),sceMsgDialogGetResult()

Aborting Message Dialog

sceMsgDialogAbort

Abort call of Message Dialog

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogAbort()
```

Arguments

None

Return Values

Returns SCE_OK(0) as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
SCE_COMMON_DIALOG_ERROR_NOT_IN_USE	0x80020411	sceMsgDialogInit() is not called
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

Description

This function aborts calling of Message Dialog.

It can be called at any time between when sceMsgDialogInit() is called and sceMsgDialogTerm() is called. If it is called at times other than the above, SCE_COMMON_DIALOG_ERROR_NOT_IN_USE is returned.

When calling is successful, processing is started to terminate the Message Dialog display being executed, and the operation status will change to SCE_COMMON_DIALOG_STATUS_FINISHED after the completion of the termination processing.

For details on the operation statuses, refer to the sceMsgDialogGetStatus() section.

When Message Dialog is closed with this function, calling sceMsgDialogGetResult() returns the following.

```
SceMsgDialogResult.result   : SCE_COMMON_DIALOG_RESULT_ABORTED
SceMsgDialogResult.buttonId : SCE_MSG_DIALOG_BUTTON_ID_INVALID
```

sceMsgDialogAbort() is used to promptly abort Message Dialog display, for example when an urgent interrupt must be processed. Since the finish processing may be skipped, use sceMsgDialogClose() when wishing to close Message Dialog normally from the caller side.

This function is multithread safe.

Examples

```
SceCommonDialogStatus stat;

while(1) {
        stat = sceMsgDialogGetStatus();
        if( stat == SCE_COMMON_DIALOG_STATUS_RUNNING ) {
              if( need_abort ) {
                    sceMsgDialogAbort();
                    break;
              }
        }
        else if( stat == SCE_COMMON_DIALOG_STATUS_FINISHED ) {
              sceMsgDialogTerm();
              break;
        }
}
```

See Also

sceMsgDialogClose(),sceMsgDialogGetResult()

Obtaining Message Dialog Call Result

sceMsgDialogGetResult

Get call result of Message Dialog

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogGetResult(
        SceMsgDialogResult *result
)
```

Arguments

result Stores the call result

Return Values

Returns SCE_OK(0) as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
SCE_COMMON_DIALOG_ERROR_NULL	0x80020402	NULL was specified in the argument *result*
SCE_COMMON_DIALOG_ERROR_NOT_FINISHED	0x80020410	Called during other than the appropriate operation status (details below)
SCE_MSG_DIALOG_ERROR_PARAM	0x80100a01	Parameter error (details below)
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

Description

This function obtains the call result of Message Dialog.

This function can be called only while the operation status of Message Dialog is SCE_COMMON_DIALOG_STATUS_FINISHED. If it is called at times other than the above, SCE_COMMON_DIALOG_ERROR_NOT_FINISHED is returned. For details on the operation statuses, refer to the sceMsgDialogGetStatus() section.

The call result of Message Dialog is stored in *result*. For details on the call results, refer to the SceMsgDialogResult section.

Be sure to initialize the argument *result* before passing it to this function. If SceMsgDialogResult.*reserved* is not filled with 0s, SCE_MSG_DIALOG_ERROR_PARAM is returned as the value of the function.

This function is multithread safe.

Examples

```
SceCommonDialogStatus stat;

while(1) {
        stat = sceMsgDialogGetStatus();
        else if( stat == SCE_COMMON_DIALOG_STATUS_FINISHED ) {
              SceMsgDialogResult result;
              sceClibMemset( &result, 0, sizeof(result) );
              if( 0 > sceMsgDialogGetResult( &result ) ) {
                    // error
              }
              sceMsgDialogTerm();
              break;
        }
}
```

See Also

SceMsgDialogResult,sceMsgDialogGetStatus()

SceMsgDialogResult

Structure for obtaining Message Dialog call result

Definition

```
#include <message_dialog.h>
typedef struct SceMsgDialogResult {
        SceMsgDialogMode mode;
        SceInt32 result;
        SceMsgDialogButtonId buttonId;
        SceChar8 reserved[32];
} SceMsgDialogResult;
```

Members

mode	Stores *mode* at the time of call
result	Stores the call result (details below)
buttonId	Stores the ID of the button selected by the user (details below)
reserved	Reserved area (fill with all 0s)

Description

This structure receives the Message Dialog call result. It is passed to `sceMsgDialogGetResult()`. Fill it with all 0s during initialization.

The value of `SceMsgDialogParam.`*mode* specified with `sceMsgDialogInit()` is stored in *mode*. It can be used to determine which function's calling result it is.

The call result of Message Dialog is stored in *result*. In the case of normal termination, one of the following positive values is stored.

Value	(Number)	Description
SCE_COMMON_DIALOG_RESULT_OK	0x0	User selected a button. Closed with `sceMsgDialogClose()`
SCE_COMMON_DIALOG_RESULT_USER_CANCELED	0x1	User performed cancel operation.
SCE_COMMON_DIALOG_RESULT_ABORTED	0x2	Aborted with `sceMsgDialogAbort()`

In the case of an error, one of the following error codes (negative value) is stored.

Value	(Number)	Description
SCE_MSG_DIALOG_ERROR_PARAM	0x80100a01	Parameter error
SCE_MSG_DIALOG_ERROR_MODULE	0x80100a02	The required module is not loaded
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

SCE_MSG_DIALOG_ERROR_MODULE returns when the liblocation library is not loaded or initialized when SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE or SCE_MSG_DIALOG_SYSMSG_TYPE_LOCATION_DATA_FAILURE_RETRY is specified upon displaying a system defined message.

One of the following values is input in *buttonId*.

Value	(Number)	Description
SCE_MSG_DIALOG_BUTTON_ID_INVALID	0	No button was selected
SCE_MSG_DIALOG_BUTTON_ID_OK	1	The **OK** button was selected
SCE_MSG_DIALOG_BUTTON_ID_YES	1	The **Yes** button was selected
SCE_MSG_DIALOG_BUTTON_ID_NO	2	The **No** button was selected
SCE_MSG_DIALOG_BUTTON_ID_RETRY	3	The **Retry** button was selected
SCE_MSG_DIALOG_BUTTON_ID_BUTTON1	1	**Button 1** was selected
SCE_MSG_DIALOG_BUTTON_ID_BUTTON2	2	**Button 2** was selected
SCE_MSG_DIALOG_BUTTON_ID_BUTTON3	3	**Button 3** was selected

reserved is a reserved area for future function expansion. It must be filled with all 0s.

The following are typical call result examples.

User selected the OK button

result : 0
buttonId : SCE_MSG_DIALOG_BUTTON_ID_OK

User selected the Yes or the No button

result : 0
buttonId : SCE_MSG_DIALOG_BUTTON_ID_YES/SCE_MSG_DIALOG_BUTTON_ID_NO

User selected the Button 1, the Button 2 or the Button 3 button

result : 0
buttonId : SCE_MSG_DIALOG_BUTTON_ID_BUTTON1 /
SCE_MSG_DIALOG_BUTTON_ID_BUTTON2 /
SCE_MSG_DIALOG_BUTTON_ID_BUTTON3

User selected the Cancel button

result : SCE_COMMON_DIALOG_RESULT_USER_CANCELED
buttonId : SCE_MSG_DIALOG_BUTTON_ID_INVALID

User selected the Retry button

result : 0
buttonId : SCE_MSG_DIALOG_BUTTON_ID_RETRY

Message Dialog was closed with `sceMsgDialogClose()`

result : 0
buttonId : SCE_MSG_DIALOG_BUTTON_ID_INVALID

Message Dialog was closed with `sceMsgDialogAbort()`

result : SCE_COMMON_DIALOG_RESULT_ABORTED
buttonId : SCE_MSG_DIALOG_BUTTON_ID_INVALID

See Also

sceMsgDialogInit(), sceMsgDialogClose(), sceMsgDialogAbort(),
sceMsgDialogGetResult()

Ending of Message Dialog

sceMsgDialogTerm

End call of Message Dialog

Definition

```
#include <message_dialog.h>
SceInt32 sceMsgDialogTerm()
```

Arguments

None

Return Values

Returns `SCE_OK(0)` as the value of the function for success.

Returns one of the following error codes (negative value) for errors.

Value	(Number)	Description
`SCE_COMMON_DIALOG_ERROR_NOT_FINISHED`	0x80020410	Called during other than the appropriate operation status (details below)
`SCE_COMMON_DIALOG_ERROR_NOT_IN_USE`	0x80020411	`sceMsgDialogInit()` is not called
`SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL`	0x8002047F	Internal error

Description

This function ends calling of Message Dialog. Calling must be ended with this function after Message Dialog has been called with `sceMsgDialogInit()`.

This function can be called only while the operation status of Message Dialog is `SCE_COMMON_DIALOG_STATUS_FINISHED`. If it is called during times other than the above, `SCE_COMMON_DIALOG_ERROR_NOT_FINISHED` is returned.
`SCE_COMMON_DIALOG_ERROR_NOT_IN_USE` will be returned if the Message Dialog function is not called.

If calling this function is successful, the operation status changes immediately to `SCE_COMMON_DIALOG_STATUS_NONE`. For details on the operation statuses, refer to the `sceMsgDialogGetStatus()` section.

This function is multithread safe.

Examples

```
SceCommonDialogStatus stat;

while(1) {
        stat = sceMsgDialogGetStatus();
        if( stat == SCE_COMMON_DIALOG_STATUS_FINISHED ) {
              sceMsgDialogTerm();
              break;
        }
}
```

See Also

`sceMsgDialogInit(),sceMsgDialogGetStatus()`

Constants

Character String Size

Maximum size of character strings of Message Dialog

Definition

Value	(Number)	Description
SCE_MSG_DIALOG_USER_MSG_SIZE	512	Size of character string that can be specified to SceMsgDialogUserMessageParam.*msg* or SceMsgDialogProgressBarParam.*msg*
SCE_MSG_DIALOG_BUTTON_MSG_SIZE	64	Size of character string that can be specified to SceMsgDialogButtonsParam.*msg1*, SceMsgDialogButtonsParam.*msg2*, or SceMsgDialogButtonsParam.*msg3*

Description

Message Dialog has a number of features that can be displayed through specification of a user specified character string. The maximum size of the character string length that can be specified is fixed for each feature, so specify the character string within the above values. The value above is the size including NULL termination.

Specify all the character strings with UTF-8. Whether a new line can be started or not depends on the location of the display.

Return Codes

List of return codes returned by Message Dialog

Definition

Value	(Number)	Description
SCE_COMMON_DIALOG_RESULT_OK	0x0	User selected a button. Closed with sceMsgDialogClose()
SCE_COMMON_DIALOG_RESULT_USER_CANCELED	0x1	User performed cancel operation.
SCE_COMMON_DIALOG_RESULT_ABORTED	0x2	Aborted with sceMsgDialogAbort()
SCE_COMMON_DIALOG_ERROR_BUSY	0x80020401	Calling another Common Dialog feature
SCE_COMMON_DIALOG_ERROR_NULL	0x80020402	NULL was specified as the function's argument
SCE_COMMON_DIALOG_ERROR_NOT_RUNNING	0x80020404	Called during a period other than SCE_COMMON_DIALOG_STATUS_RUNNING
SCE_COMMON_DIALOG_ERROR_NOT_SUPPORTED	0x80020405	Called a function not supported by the current operation mode
SCE_COMMON_DIALOG_ERROR_NOT_FINISHED	0x80020410	Called during a period other than SCE_COMMON_DIALOG_STATUS_FINISHED
SCE_COMMON_DIALOG_ERROR_NOT_IN_USE	0x80020411	sceMsgDialogInit() is not called
SCE_MSG_DIALOG_ERROR_PARAM	0x80100a01	Parameter error
SCE_MSG_DIALOG_ERROR_MODULE	0x80100a02	The required module is not loaded
SCE_COMMON_DIALOG_ERROR_UNEXPECTED_FATAL	0x8002047F	Internal error

Parameter Errors

Codes output when values of structures specified to Message Dialog are invalid

Definition

Value	Description
1	`SceMsgDialogParam.`*`sdkVersion`* is invalid
2	`SceMsgDialogParam.`*`mode`* is invalid
3	`SceMsgDialogParam.`*`flag`* is invalid
4	`SceMsgDialogParam.`*`reserved`* is invalid
5	`SceMsgDialogUserMessageParam` is NULL
6	`SceMsgDialogUserMessageParam` is not NULL
7	`SceMsgDialogUserMessageParam.`*`buttonType`* is invalid
8	`SceMsgDialogUserMessageParam.`*`msg`* is NULL
9	`SceMsgDialogUserMessageParam.`*`msg`* format is invalid
10	`SceMsgDialogUserMessageParam.`*`reserved`* is invalid
11	`SceMsgDialogSystemMessageParam` is NULL
12	`SceMsgDialogSystemMessageParam` is not NULL
13	`SceMsgDialogSystemMessageParam.`*`sysMsgType`* is invalid
14	`SceMsgDialogSystemMessageParam.`*`value`* is invalid
15	`SceMsgDialogSystemMessageParam.`*`reserved`* is invalid
16	`SceMsgDialogErrorCodeParam` is NULL
17	`SceMsgDialogErrorCodeParam` is not NULL
18	`SceMsgDialogErrorCodeParam.`*`errorCode`* is invalid
19	`SceMsgDialogErrorCodeParam.`*`reserved`* is invalid
20	`SceMsgDialogProgressBarParam` is NULL
21	`SceMsgDialogProgressBarParam` is not NULL
22	`SceMsgDialogProgressBarParam.`*`barType`* is invalid
23	`SceMsgDialogProgressBarParam.sysMsgParam.`*`sysMsgType`* is invalid
24	`SceMsgDialogProgressBarParam.`*`msg`* is NULL
25	`SceMsgDialogProgressBarParam.`*`msg`* is not NULL
26	`SceMsgDialogProgressBarParam.`*`msg`* format is invalid
27	`SceMsgDialogProgressBarParam.`*`reserved`* is invalid
28	`SceMsgDialogButtonsParam` is NULL
29	`SceMsgDialogButtonsParam` is not NULL
30	`SceMsgDialogButtonsParam.`*`msg1`* is invalid
31	`SceMsgDialogButtonsParam.`*`fontSize1`* is invalid
32	`SceMsgDialogButtonsParam.`*`msg2`* is invalid
33	`SceMsgDialogButtonsParam.`*`fontSize2`* is invalid
34	`SceMsgDialogButtonsParam.`*`msg3`* is invalid
35	`SceMsgDialogButtonsParam.`*`fontSize3`* is invalid
36	`SceMsgDialogButtonsParam.`*`reserved`* is invalid
37	`SceMsgDialogButtonsParam.`*`msg1`* is NULL
38	`SceMsgDialogButtonsParam.`*`msg2`* is NULL
39	`SceMsgDialogButtonsParam.`*`msg3`* is NULL
100	`SceMsgDialogResult.`*`reserved`* is invalid
101	`SceMsgDialogProgressBarTarget` is invalid
102	`SceMsgDialogParam.`*`commonParam.bgColor`* is not NULL
103	*`msg`* specified with `sceMsgDialogProgressBarSetMsg()` is invalid

Description

If the contents of the structures specified for the various functions provided by Message Dialog are invalid, they are processed as parameter errors, and the operation status of Message Dialog changes immediately to `SCE_COMMON_DIALOG_STATUS_FINISHED`.

At this time, `SCE_MSG_DIALOG_ERROR_PARAM` returns to the return value of the called API or `SceMsgDialogResult.`*`result`*, which can be obtained with `sceMsgDialogGetResult()`.

Furthermore, the concrete parameter error occurrence locations are output to the console in the following format.

```
****** SceMsgDialog Parameter Error : XX ****** (XX is one of the above numbers)
```

Parameter errors are coding mistakes of the application and must absolutely be fixed before release.