Cross-Controller Dialog Overview

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

1 Library Overview

Scope of This Document

This document explains the Cross-Controller Dialog library, which supports device coordination.

Purpose and Features

The Cross-Controller Dialog library is one of the Common Dialog libraries. It supports device coordination. When creating an application that runs in coordination both on PlayStation®3 and PlayStation®Vita, the Cross-Controller Dialog library can be used to enable direct network communication between PlayStation®3 and PlayStation®Vita that are connected to a common local network.

Main Feature

The main feature offered by Cross-Controller Dialog is as follows:

- Feature to detect PlayStation®3 connected to a common local network

Embedding into a Program

Include cross_controller_dialog.h in the source program. (Additionally, a number of header files are automatically included.)

The PRX module need not be loaded.

When building programs, link libSceCommonDialog_stub.a.

Sample Program

The following files are provided as sample programs that use the Cross-Controller Dialog library for reference purposes.

sample_code/system/api_cross_controller_dialog/fixed_basic/

This sample shows basic uses of the Cross-Controller Dialog library.

Reference Materials

For the common limitations, specifications, etc., of the Common Dialog library, refer to the "Common Dialog Overview" document.

Regarding the coordination between PlayStation®3 and PlayStation®Vita, refer to the "Cross-Controller Utility Overview" and "Cross-Controller Utility Reference" documents included in the PlayStation®3 SDK.

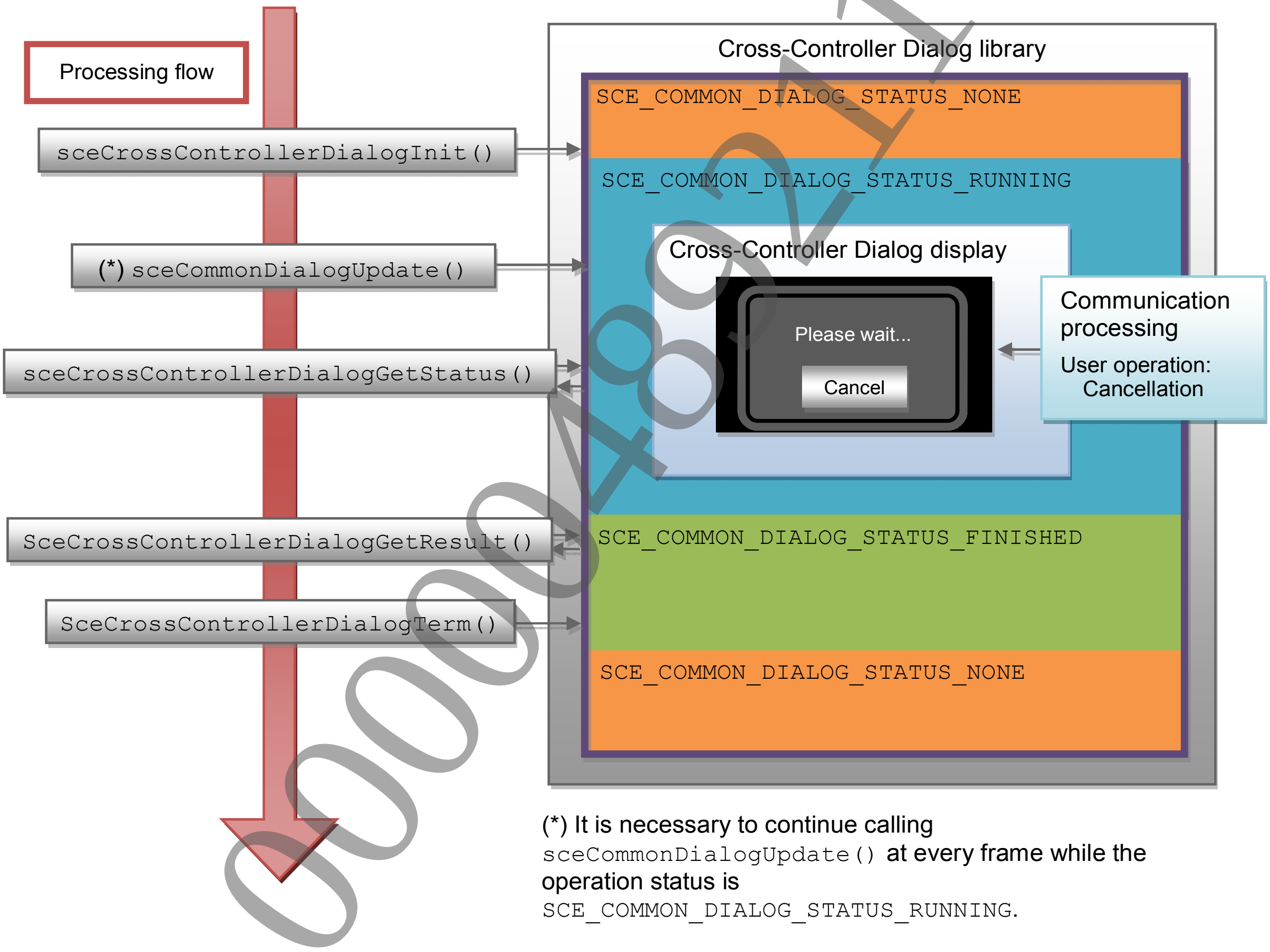
2 Usage Procedure

Basic Usage Procedure

The basic procedure to use the Cross-Controller Dialog library is described below. The processing flow is outlined below.

(1) Setting the parameters
(2) Initializing the library (call Cross-Controller Dialog)
(3) Waiting for the response from the dialog
(4) Obtaining the call result
(5) Terminating the processing

Figure 1 Basic Processing Procedure



Processing flow
sceCrossControllerDialogInit()
(*) sceCommonDialogUpdate()
sceCrossControllerDialogGetStatus()
SceCrossControllerDialogGetResult()
SceCrossControllerDialogTerm()
Cross-Controller Dialog library
SCE_COMMON_DIALOG_STATUS_NONE
SCE_COMMON_DIALOG_STATUS_RUNNING
Cross-Controller Dialog display
Please wait...
Cancel
Communication processing
User operation: Cancellation
SCE_COMMON_DIALOG_STATUS_FINISHED
SCE_COMMON_DIALOG_STATUS_NONE

(*) It is necessary to continue calling `sceCommonDialogUpdate()` at every frame while the operation status is `SCE_COMMON_DIALOG_STATUS_RUNNING`.

Cross-Controller Dialog Call Procedure

(1) Setting the parameters

Prepare the `SceCrossControllerDialogParam` type structure and following initialization with `sceCrossControllerDialogParamInit()`, set the title ID of the title expected to be running on the search target PlayStation®3 to *`ps3TitleId`* and the appropriate minimum title version to *`ps3AppMinVer`* of the initialized structure.

(2) Initializing the library

Call Cross-Controller Dialog with `sceCrossControllerDialogInit()`.

Specify the `SceCrossControllerDialogParam` type structure set in (1) as the argument. If dialog call succeeds, a busy screen will be displayed and a search for PlayStation®3 on the local network will start. The operation status will transition to `SCE_COMMON_DIALOG_STATUS_RUNNING`.

(3) Waiting for the response from the dialog

Call `sceCrossControllerDialogGetStatus()` to poll the operation status of Cross-Controller Dialog at each frame.

Note
`sceCommonDialogUpdate()` must be called at every frame while the operation status is `SCE_COMMON_DIALOG_STATUS_RUNNING`. For details, refer to the "Common Dialog Overview" document.

The operation status will transition to `SCE_COMMON_DIALOG_STATUS_FINISHED` when a PlayStation®3 on the local network is found, when the user cancels the operation, and when `sceCrossControllerDialogAbort()` is called.

(4) Obtaining the call result

Obtain the result with `sceCrossControllerDialogGetResult()`. This function obtains the IP address of the detected PlayStation®3 or the result of user cancellation.

(5) Terminating the processing

Call `sceCrossControllerDialogTerm()` to terminate the processing. As a result, the resources allocated upon calling the Cross-Controller Dialog function are released, and the operation status becomes `SCE_COMMON_DIALOG_STATUS_NONE`.

Aborting the Processing

When aborting the display of Cross-Controller Dialog from the application side (when terminating the application, for example), call `sceCrossControllerDialogAbort()`. Processing is started to terminate the busy screen display and the operation status will change to `SCE_COMMON_DIALOG_STATUS_FINISHED` after the completion of the termination processing. In this case, too, the call result will be obtained with `sceCrossControllerDialogGetResult()`. `SCE_COMMON_DIALOG_RESULT_ABORTED` is obtained as the obtained result.

Main APIs Used for Basic Processing

API	Description
`SceCrossControllerDialogParam`	Cross-Controller Dialog parameter structure
`sceCrossControllerDialogParamInit()`	Macro function for call parameter initialization
`sceCrossControllerDialogInit()`	Calls Cross-Controller Dialog
`sceCrossControllerDialogGetStatus()`	Obtains operation status of the Cross-Controller Dialog library
`SceCrossControllerDialogResult`	Structure for storing obtained result of a Cross-Controller Dialog call
`sceCrossControllerDialogGetResult()`	Obtains result of a Cross-Controller Dialog call
`sceCrossControllerDialogTerm()`	Terminates Cross-Controller Dialog
`sceCrossControllerDialogAbort()`	Aborts a Cross-Controller Dialog call

3 Reference Information

Conditions for Use

A network-connected environment is required.

4 Precautions

Limitations

Common Dialog limitations apply.