libsysmodule Overview

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

1 Library Overview

Purpose and Characteristics

libsysmodule is a library to be used for libraries implemented in PRX format.

For PlayStation®Vita, there are two ways of implementing runtime libraries. A PRX-format library has code on the storage managed by system software that is loaded onto memory when necessary (according to instructions by the application) and then unloaded when it is no longer required. libsysmodule provides the API for loading and unloading these modules.

Files

The files required for using libsysmodule are as follows.

Filename	Description
libsysmodule.h	Header file
libSceSysmodule_stub.a	Stub library file

Sample Programs

There are no sample programs provided specifically for libsysmodule. Refer to the sample programs for PRX-format libraries.

2 Using the Library

Basic Procedure

(1) Load PRX

In order to use a PRX-format library, its PRX must first be loaded. To load a library's PRX, call `sceSysmoduleLoadModule()` with the module ID specified as the argument. When loading is successful, `SCE_OK` will be returned.

(2) Use the PRX Library

When the PRX loads successfully, the API of that library will become callable.

(3) Unload PRX

After the library is used, its PRX can be unloaded to free memory. To unload a PRX, call `sceSysmoduleUnloadModule()` with the module ID specified as the argument. When this is successful, `SCE_OK` will be returned.

3 Library Operation

In addition to the static linking format, there are runtime libraries implemented in PRX format.

PRX is a format that has been in use since PSP™ (PlayStation®Portable) and provides function services as a resident library. By linking the stub libraries and loading the service modules, the function services will become available for use.

Static Linking Format

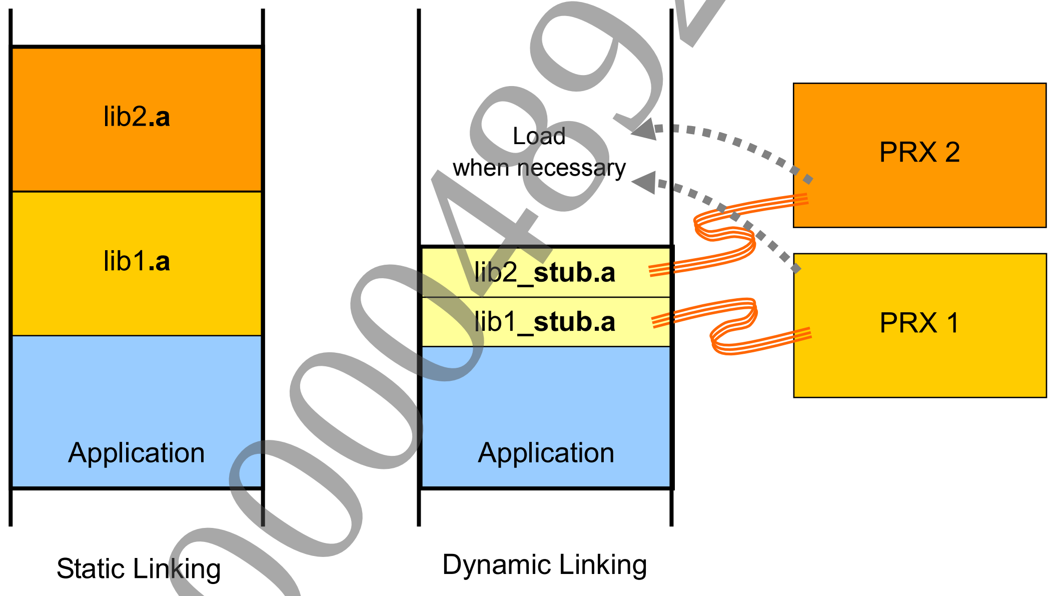
With statically linked libraries, linking is carried out during building. The library code is linked statically to the application code, and it occupies memory space during the whole time the application is running.

Dynamic Linking Format with PRX

With PRX-format libraries, linking is carried out during execution. The library code is in the PlayStation®Vita system; when a module will be used, it is loaded onto memory and linked; when it is unnecessary, it can be unloaded to free memory space.

This loading must be carried out by the application before the library is used. It is also necessary to statically link the stub libraries for relaying the function calls.

Figure 1 Static Linking and Dynamic Linking Using PRX

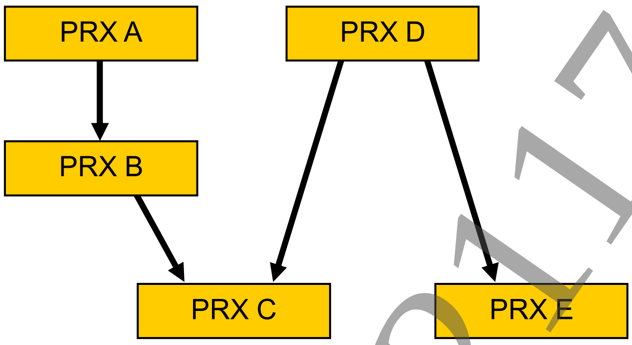


Resolving Module Dependency of libsysmodule

libsysmodule resolves module dependencies of each PRX and automatically loads the required PRX by a single call for a certain functionality. It also manages loading so that two or more of the same PRX are not loaded. Similarly upon unloading PRX, libsysmodule automatically unloads the depended PRX, however, this will not be carried out if there is another PRX dependent on it.

This scheme frees the programmer from having to worry about on which functionalities the functionality he/she wants to call is dependent.

For example, suppose there are modules with the following configuration.



Load of PRX A will entail:

(1) Load of PRX C

(2) Load of PRX B

(3) Load of PRX A

If PRX D is to be loaded next, this will entail:

(1) Load of PRX C -> because it is already loaded, actual load will not be performed

(2) Load of PRX E

(3) Load of PRX D

On the other hand, unload will entail:

(1) Unload of PRX A

(2) Unload of PRX B

(3) Unload of PRX C -> because PRX D requires PRX C, it will not be unloaded

If PRX D is to be unloaded, this will entail

(1) Unload of PRX D

(2) Unload of PRX E

(3) Unload of PRX C

4 Package Installation

Package Installation of Runtime Libraries

For part of the runtime libraries in PRX format, PRXs are distributed in the SDK. Since these runtime libraries are not contained in the PlayStation®Vita system, like EBOOT.BIN they need to be included in the package to be submitted to SCE.

The runtime libraries for distribution in the SDK are as follows:

Library Name	File Name
C and C++ Standard Libraries	%SCE_PSP2_SDK_DIR%\target\sdk\target\sce_module\libc.suprx
libfios2	%SCE_PSP2_SDK_DIR%\target\sdk\target\sce_module\libfios2.suprx
libult	%SCE_PSP2_SDK_DIR%\target\sdk\target\sce_module\libult.suprx
libface	%SCE_PSP2_SDK_DIR%\target\sdk\target\sce_module\libface.suprx
libsmart	%SCE_PSP2_SDK_DIR%\target\sdk\target\sce_module\libsmart.suprx

As with other runtime libraries, PRXs are loaded with `sceSysmoduleLoadModule()` by specifying the module ID.

Note
C and C++ Standard Libraries and libfios2 do not need to be loaded explicitly because they are loaded automatically by the system when the application is started up.

During application development, PRX files (.suprx) must be placed in advance below the app0:sce_module directory. Refer to the SDK sample programs for the placement method.

Refer to the "Package Generator User's Guide" included in the Publishing Tools for the packaging method of system libraries.

5 Notes

Restrictions

When a module that is already loaded/unloaded is specified for loading/unloading, the module will not be loaded/unloaded again, and `SCE_OK` will always be returned (an error will not return). Whether or not a module is loaded can be checked with `sceSysmoduleIsLoaded()`.