NP WordFilter Library Overview

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

1 Library Overview

Purpose and Features

The NP WordFilter library is a library for checking whether a given text contains inappropriate comments. Character strings where inappropriate characters are replaced with "*" can also be obtained.

Main Functions

The main functions provided by the NP WordFilter library are as follows.

- Function to check whether inappropriate comments are included
- Function for obtaining character strings where inappropriate characters are replaced with "*"

Used Resources

The NP WordFilter library uses the following system resources.

Resource	Description
Footprint	Approximately 45KiB when PRX is loaded
Work memory	(The application is not required to provide memory.)
Thread	When `sceNpWordFilterInit()` is called with `SCE_TRUE` set in *`usesAsync`*, a thread for asynchronous functions is generated. The above thread is stopped with `sceNpWordFilterTerm()`. If `SCE_FALSE` is set in *`usesAsync`*, the thread will not be generated. Thread priority and CPU affinity mask are specified from the application with the argument of `sceNpWordFilterInit()`.
Processor time	Can be ignored

Embedding into a Program

Include np.h in the source program. Various header files will be automatically included as well.

Load also the PRX module in the program as follows.

```
if ( sceSysmoduleLoadModule(SCE_SYSMODULE_NP_UTILITY) != SCE_OK ) {
        // Error processing
}
```

Upon building the program, link libSceNpUtility_stub.a.

Sample Programs

The following program is provided as a NP WordFilter sample program for reference purposes.

sample_code/network/api_np/np_word_filter/

This sample is for checking whether inappropriate comments are included and obtaining character strings where inappropriate characters are replaced with "*".

Reference Materials

Refer to the following document for an overview of the PSN℠ functionalities.

- PSN℠ Overview

Refer to the following documents regarding the NP library, which is commonly required when using the PSN℠ functionalities.

- NP Library Overview
- NP Library Reference

Refer to the following document regarding Network Check Dialog for switching service states of the NP library.

- Network Overview

Refer to the following document regarding the system software related to the NP WordFilter library.

- System Software Overview

2 Using the Library

Initialization

To use the NP WordFilter library, first initialize the NP library and then initialize the NP WordFilter library.

(1) Load PRX

Call `sceSysmoduleLoadModule()` with `SCE_SYSMODULE_NP_UTILITY` specified as the module ID to load the NP PRX.

(2) Initialize the library

Load NET, HTTP, and NP modules, and perform initialization. For details, refer to the documents of the respective libraries.

Initialize the NP WordFilter library by `sceNpWordFilterInit()`.

```
ret = sceNpWordFilterInit(
               SCE_TRUE,
               SCE_KERNEL_DEFAULT_PRIORITY_USER,
               SCE_KERNEL_THREAD_CPU_AFFINITY_MASK_DEFAULT,
               NULL);
if (ret < 0) {
        // Error handling
}
```

When `SCE_TRUE` is set in the first argument *`usesAsync`*, a thread for asynchronous functions is generated internally. If `SCE_FALSE` is set in *`usesAsync`*, the thread will not be generated. If `SCE_FALSE` is set in *`usesAsync`*, the asynchronous functions `sceNpWordFilterCensorCommentAsync()` and `sceNpWordFilterSanitizeCommentAsync()` cannot be used.

(3) Create an NP WordFilter title context

Call `sceNpWordFilterCreateTitleCtx()` to create an NP WordFilter title context. As arguments, specify the

NP Communication ID (`SceNpCommunicationId`) for identifying the application, and the NP ID

(`SceNpId`) of the login user.

When NULL is passed to the NP communication ID of this function, the NP communication ID set with the `sceNpInit()` function of the NP library is used. Regarding `sceNpInit()`, refer to the "NP Library Overview" and "NP Library Reference" documents.

```
int ret, titleCtxId;
SceNpCommunicationId communicationId;
SceNpId selfNpId;
// communicationId assumed to store an appropriate value

ret = sceNpManagerGetNpId(&selfNpId);
if (ret < 0) {
        // Error handling
}
ret = sceNpWordFilterCreateTitleCtx(&communicationId, &selfNpId);
if (ret < 0) {
        // Error handling
}
titleCtxId = ret;
```

Communication

The NP WordFilter library provides a synchronous API and an asynchronous API for performing communication with the server and checking whether there are inappropriate comments. Follow the procedure below when using the synchronous API.

(1) Create a request

Create a request ID to be used for aborting and deleting a request. A request ID must be created per request, and deleted after the request ends.

```
int ret, reqId, titleCtxId;
SceNpId npId;
SceNpOnlineId onlineId;

// titleCtxId assumed to store an appropriate value

ret = sceNpWordFilterCreateRequest (titleCtxId);
if (ret < 0) {
        // Error handling
}
reqId = ret;
```

(2) Request

Call the appropriate function from below according to the information you wish to obtain.

- Check whether inappropriate comments are included: `sceNpWordFilterCensorComment()`
- Obtain character strings where inappropriate characters are replaced with "*": `sceNpWordFilterSanitizeComment()`

These functions block other processing until the applicable information is obtained from the server.

```
// Request
strncpy(censorText, "text", SCE_NP_WORD_FILTER_CENSOR_COMMENT_MAXLEN + 1);
ret = sceNpWordFilterCensorComment (
        reqId,       // Request ID
        censorText,  // Character string to be checked for inappropriate
comments
        NULL);
if (ret == SCE_NP_COMMUNITY_SERVER_ERROR_CENSORED) {
        // Processing if inappropriate comments were included
} else if (ret < 0) {
        // Error handling
}
....
```

(3) Destroy request

When the request ends, destroy the request ID.

```
// Destroy the request
sceNpWordFilterDeleteRequest (reqId);
```

Request Using an Asynchronous API

The functions below are non-blocking. They return immediately (without waiting for information to be obtained from the server) after starting the request.

- Check whether inappropriate comments are included: `sceNpWordFilterCensorCommentAsync()`
- Obtain character string where inappropriate characters are replaced with "*": `sceNpWordFilterSanitizeCommentAsync()`

To confirm the termination of the request started by these functions, call sceNpWordFilterWaitAsync() or sceNpWordFilterPollAsync(). sceNpWordFilterWaitAsync() waits for the request to complete if it hasn't already done so. sceNpWordFilterPollAsync() immediately returns the value 1 if the request has not completed. If one of these APIs returns 0 and the application receives the result of the communication, the request will complete.

Requests Which Do Not Allow Parallel Execution

Asynchronous APIs are processed with an internal thread generated with sceNpWordFilterInit(). Since only one thread is generated, even if multiple asynchronous functions are called simultaneously, internally they are processed one at a time, without allowing parallel execution. If you wish to perform parallel execution, generate multiple threads by the application, and execute synchronous API in each thread. Also, as the number of requests that can exist simultaneously is 32, parallel execution of a larger number is not possible.

Termination

(1) Destroy NP WordFilter title context

When the title context is no longer required, call sceNpWordFilterDeleteTitleCtx() to destroy the context.

```
int ret;
uint32_t titleCtxId;

// titleCtxId assumed to store an appropriate value

ret = sceNpWordFilterDeleteTitleCtx (titleCtxId);
if (ret < 0) {
        // Error handling
}
```

(2) Terminate the NP WordFilter library

Call sceNpWordFilterTerm() to terminate the NP WordFilter library

Note
sceNpWordFilterTerm() is not multithread safe.
When sceNpWordFilterTerm() is called, the created title context and request will be deleted automatically, however, it is recommended that this function be called after the title contexts and requests are explicitly deleted from the application side.

```
// If a thread for asynchronous functions had been generated, it will be stopped.
sceNpWordFilterTerm();
```

(3) Other termination processing

Perform termination processing of NET, HTTP and NP modules. For details, refer to the documents of the respective libraries.

Then, unload PRX by calling sceSysmoduleUnloadModule() with SCE_SYSMODULE_NP_UTILITY specified for the module ID.

3 Server Settings

Obtain an NP Communication ID

The NP WordFilter library requires an NP Communication ID to manage the title.

NP communication IDs are issued for each application by applying to the PlayStation®Vita Developer Network (https://psvita.scedev.net/).

4 Notes

Character strings requiring check for inappropriateness

It is not necessary to check strings used in chatting, for example. Use this function to check strings that will be visible to all players and will remain visible over a long time.

Rules for words judged to be inappropriate

Note that the rules regarding inappropriate words are subject to change without notice.