libheap Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

Structures

SceHeapOptParam

Heap additional data

Definition

```
#include <libheap.h>
typedef struct SceHeapOptParam {
        unsigned int size;
} SceHeapOptParam;
```

Members

size	Size of this structure (`sizeof(SceHeapOptParam)`)

Description

This structure is used for storing additional data related to the heap. It is provided when `sceHeapCreateHeap()` is used to create the heap. Currently, it is not used yet.

See Also

`sceHeapCreateHeap()`

SceHeapAllocOptParam

Heap allocation additional data

Definition

```
#include <libheap.h>
typedef struct SceHeapAllocOptParam {
        unsigned int size;
        unsigned int alignment;
} SceHeapAllocOptParam;
```

Members

size	Size of this structure (`sizeof(SceHeapAllocOptParam)`)
alignment	Alignment size

Description

This structure is used for storing additional data related to heap allocation. It is provided when `sceHeapAllocHeapMemoryWithOption()` is used to allocate a memory block from the heap.

See Also

`sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemoryWithOption()`

SceHeapMallinfo

Heap usage status information

Definition

```
#include <libheap.h>
typedef struct SceHeapMallinfo {
        int arena;          /*E Total memry size allocated from kernel     */
        int ordblks;        /*E Number of allocated blocks            */
        int smblks;         /*E Unused -- always zero                */
        int hblks;          /*E Number of held blocks                */
        int hblkhd;         /*E Total bytes of held blocks (arena) */
        int usmblks;        /*E Unused -- always zero                */
        int fsmblks;        /*E Unused -- always zero                */
        int uordblks;       /*E Total allocated memory size               */
        int fordblks;       /*E Total memory size that can be allocated      */
        int keepcost;       /*E Unused -- always zero                */
} SceHeapMallinfo;
```

Members

arena	Total memory allocated from the kernel (bytes)
ordblks	Number of allocated blocks
smblks	Unused (always zero)
hblks	Number of times memory blocks were allocated from the kernel
hblkhd	Total memory allocated from the kernel (same as *arena*)
usmblks	Unused (always zero)
fsmblks	Unused (always zero)
uordblks	Total allocated memory
fordblks	Total memory that can be allocated
keepcost	Unused (always zero)

Description

This structure is used for storing data that is obtained when `sceHeapGetMallinfo()` is used to get the heap memory usage status.

See Also

`sceHeapGetMallinfo()`

Heap Memory Management Functions

sceHeapAllocHeapMemory

Allocate memory from heap memory

Definition

```
#include <libheap.h>
void * sceHeapAllocHeapMemory (
        void *heap,
        unsigned int nbytes
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.
nbytes	Size of memory to be allocated in bytes.

Return Values

Value	Description
Not NULL	Starting address of allocated memory
NULL	Allocation failed

Description

This function allocates memory from the argument *heap*.

See Also

`sceHeapCreateHeap(),sceHeapFreeHeapMemory()`

sceHeapAllocHeapMemoryWithOption

Allocate memory from heap memory

Definition

```
#include <libheap.h>
void * sceHeapAllocHeapMemoryWithOption (
        void *heap,
        unsigned int nbytes,
        const SceHeapAllocOptParam *optParam
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.
nbytes	Size of memory to be allocated in bytes.
optParam	Pointer to an `SceHeapAllocOptParam` structure that contains additional information for use when memory is allocated.

Return Values

Value	Description
Not NULL	Starting address of allocated memory
NULL	Allocation failed

Description

This function allocates memory from the argument *heap*. The alignment size of the memory to be allocated can be specified by using the `SceHeapAllocOptParam` structure.

See Also

`sceHeapCreateHeap()`, `sceHeapFreeHeapMemory()`, `SceHeapAllocOptParam`

sceHeapCreateHeap

Create heap memory

Definition

```
#include <libheap.h>
void * sceHeapCreateHeap (
        const char *name,
        int heapblocksize,
        int flags,
        const SceHeapOptParam *optParam
);
```

Calling Conditions

Multithread safe.

Arguments

name	Name of the heap. This is only used by the operator for visual identification during debugging, so no specific check for uniqueness is performed.
heapblocksize	Size of heap memory to be allocated in bytes.
flags	0 or a flag.
optParam	Reserved for future extensions. NULL should be specified.

Return Values

Value	Description
Not NULL	Starting address of allocated memory
NULL	Allocation failed

Description

This function allocates heap memory with a size equal to the number of bytes specified by the *heapblocksize* argument rounded up to a multiple of 4 or more.

The following flags can be specified individually for *flags*.

Flag	Description
SCE_HEAP_AUTO_EXTEND	When there is insufficient heap memory, the heap is automatically extended.

See Also

`sceHeapDeleteHeap()`

sceHeapDeleteHeap

Delete heap memory

Definition

```
#include <libheap.h>
int sceHeapDeleteHeap (
        void *heap
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory to be released

Return Values

If an error occurs, a negative value is returned.

Value	Description
`SCE_OK`	Normal completion
`SCE_HEAP_ERROR_INVALID_ID`	*heap* is invalid

Description

This function releases the heap memory specified by the argument *heap* and returns it to the kernel. Note that even if memory that was previously allocated with a function such as `sceHeapAllocHeapMemory()` was left in heap memory, the entire heap memory will be deleted, regardless.

See Also

`sceHeapCreateHeap()`

sceHeapFreeHeapMemory

Release memory back to heap memory

Definition

```
#include <libheap.h>
int sceHeapFreeHeapMemory (
        void *heap,
        void *ptr
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.
ptr	Starting address of the memory to be released.

Return Values

If an error occurs, a negative value is returned.

Value	Description
`SCE_OK`	Normal completion
`SCE_HEAP_ERROR_INVALID_ID`	*heap* is invalid
`SCE_HEAP_ERROR_INVALID_POINTER`	*ptr* is invalid

Description

This function releases memory blocks that were allocated with `sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemory()`, or `sceHeapReallocHeapMemoryWithOption()` from the heap indicated by *heap* back to heap memory. The *heap* argument must specify the same heap as was specified when the memory was allocated.

When NULL is specified for the argument *ptr*, nothing is done.

See Also

`sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemory()`, `sceHeapReallocHeapMemoryWithOption()`

sceHeapGetTotalFreeSize

Get size of empty heap memory

Definition

```
#include <libheap.h>
int sceHeapGetTotalFreeSize (
        void *heap
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.

Return Values

If an error occurs, a negative value is returned.

Value	Description
`>=0`	Size of empty heap memory
`SCE_HEAP_ERROR_INVALID_ID`	*heap* is invalid

Description

This function gets the current amount of empty heap memory.

When there are discontiguous empty memory areas within heap memory, this function returns the sum of the sizes of all empty memory areas. Note that the size that is returned by this function does not match the total size of memory that can be allocated by a function such as `sceHeapAllocHeapMemory()` because of libheap management overhead.

If it is a problem that discontiguous empty memory areas exist within heap memory, this situation can be avoided by using the `sceHeapCreateHeap()` function to create multiple independent heap memories.

This function is assumed to be used essentially for debugging.

sceHeapGetMallinfo

Get heap memory usage status

Definition

```
#include <libheap.h>
int sceHeapGetMallinfo (
        void *heap,
        SceHeapMallinfo *pInfo
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.
pInfo	Pointer to an `SceHeapMallinfo` structure for obtaining heap memory information.

Return Values

If an error occurs, a negative value is returned.

Value	Description
SCE_OK	Normal completion
SCE_HEAP_ERROR_INVALID_ID	*heap* is invalid

Description

This function gets the current usage status of the specified heap memory.

This function is assumed to be used essentially for debugging.

See Also

`SceHeapMallinfo`

sceHeapIsAllocatedHeapMemory

Check whether or not memory was allocated from heap memory

Definition

```
#include <libheap.h>
int sceHeapIsAllocatedHeapMemory (
        void *heap,
        void *ptr
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.
ptr	Starting address of the memory to be checked.

Return Values

If an error occurs, a negative value is returned.

Value	Description
1	The memory was allocated from *heap*
0	The memory was not allocated from *heap*
`SCE_HEAP_ERROR_INVALID_ID`	*heap* is invalid
`SCE_HEAP_ERROR_INVALID_POINTER`	*ptr* is invalid

Description

This function checks whether or not the heap indicated by the *heap* argument is memory that was allocated by `sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemory()`, or `sceHeapReallocHeapMemoryWithOption()`.

This function is assumed to be used essentially for debugging.

See Also

`sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemory()`, `sceHeapReallocHeapMemoryWithOption()`

sceHeapReallocHeapMemory

Reallocate memory from heap memory

Definition

```
#include <libheap.h>
void * sceHeapReallocHeapMemory (
        void *heap,
        void *ptr,
        unsigned int nbytes
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.
ptr	Starting address of memory block to be reallocated.
nbytes	Size of new memory to be allocated in bytes.

Return Values

Value	Description
Not NULL	Starting address of reallocated memory
NULL	Reallocation failed

Description

This function reallocates memory from the argument *heap*. For the argument *ptr*, specify either NULL or the memory block that was previously allocated with `sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemory()`, or `sceHeapReallocHeapMemoryWithOption()`. If NULL is specified for *ptr*, this function will operate the same as `sceHeapAllocHeapMemory()`.

For *nbytes*, specify the updated memory block size that is desired. If 0 is specified for *nbytes*, this function operates the same as `sceHeapFreeHeapMemory()`. The return value at this time will be NULL.

If the newly allocated size is larger than the current size, the newly allocated memory block may be moved to a different location than the address specified by *ptr*. If a move occurs, the current memory block will be released by `sceHeapFreeHeapMemory()`.

See Also

`sceHeapCreateHeap()`, `sceHeapFreeHeapMemory()`, `sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemoryWithOption()`

sceHeapReallocHeapMemoryWithOption

Reallocate memory from heap memory

Definition

```
#include <libheap.h>
void * sceHeapReallocHeapMemoryWithOption (
        void *heap,
        void *ptr,
        unsigned int nbytes,
        const SceHeapAllocOptParam *optParam
);
```

Calling Conditions

Multithread safe.

Arguments

heap	Starting address of heap memory previously allocated with `sceHeapCreateHeap()`.
ptr	Starting address of memory block to be reallocated.
nbytes	Size of new memory to be allocated in bytes.
optParam	Pointer to an `SceHeapAllocOptParam` structure that contains additional information for use when memory is allocated.

Return Values

Value	Description
Not NULL	Starting address of reallocated memory
NULL	Reallocation failed

Description

This function reallocates memory from the argument *heap*. For the argument *ptr*, specify either NULL or the memory block that was previously allocated with `sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemory()`, or `sceHeapReallocHeapMemoryWithOption()`. If NULL is specified for *ptr*, this function will operate the same as `sceHeapAllocHeapMemoryWithOption()`.

For *nbytes*, specify the updated memory block size that is desired. If 0 is specified for *nbytes*, this function operates the same as `sceHeapFreeHeapMemory()`. The return value at this time will be NULL.

The alignment size of the memory to be allocated can be specified in the `SceHeapAllocOptParam` structure.

If the newly allocated size is larger than the current size or if the current pointer is incompatible with the alignment size specified in the `SceHeapAllocOptParam` structure, the newly allocated memory block may be moved to a different location than the address specified by *ptr*. If a move occurs, the current memory block will be released by `sceHeapFreeHeapMemory()`.

See Also

`sceHeapCreateHeap()`, `sceHeapFreeHeapMemory()`, `sceHeapAllocHeapMemory()`, `sceHeapAllocHeapMemoryWithOption()`, `sceHeapReallocHeapMemory()`, `SceHeapAllocOptParam`