

メモリアロケータ

メモリの確保と解放の統一インターフェース

Ver 1.0.1

任天堂株式会社発行

このドキュメントの内容は、機密情報であるため、**厳重な取り扱い、管理を行ってください。**

目次

1	はじめに	4
2	メモリアロケータの初期化	4
3	メモリアロケータの関数	4
4	独自のメモリ管理ライブラリとメモリアロケータを関連付ける場合	5

表

表 2-1	メモリアロケータを初期化する関数	4
表 3-1	メモリブロックの確保と解放を行う関数	5

改訂履歷

版	改訂日	改 訂 内 容	承認者	担当者
1.0.1	2005-08-26	関数名が間違っていたのを修正。 (NNS_FndInitAllocatorForOSHeap NNS_FndInitAllocatorForSDKHeap)		喜多村
1.0.0	2005-01-06	バージョン表記を修正。		西田泰
0.3.0	2004-08-02	初版。		喜多村

1 はじめに

NITRO-System ライブラリの中には、ライブラリ自身の中で必要なメモリの確保を行うものがあります。このようなライブラリは、ライブラリ内で動的なメモリの確保は必要とするものの、メモリリソースの管理方法は問いません。従って、特定のメモリ管理ライブラリに依存してしまうのは好ましくありません。

そこで、メモリアロケータという仕組みを導入しました。メモリアロケータは特定のメモリ管理ライブラリと組み合わせて使用します。メモリアロケータはメモリの確保と解放の機能だけを提供しますが、実際のメモリの確保と解放は組み合わせているメモリ管理ライブラリの機能呼び出すことで提供しています。そして、ライブラリの方はこのメモリアロケータを受け取り、アロケータを通してメモリの確保と解放を行います。このためライブラリ側は特定のメモリ管理ライブラリに依存しなくて済みます。

2 メモリアロケータの初期化

メモリアロケータは使用する前に初期化を行う必要があります。メモリアロケータは必要な情報を `NNSFndAllocator` という構造体に保持するようになっており、初期化関数でこの構造体へのポインタを渡して初期化します。初期化関数は 4 種類あり、それぞれ NITRO-System の 3 つヒープと NITRO-SDK のヒープに対応しています。次に初期化関数を示します。

表 2-1 メモリアロケータを初期化する関数

関数	機能
<code>NNS_FndInitAllocatorForExpHeap()</code>	拡張ヒープからメモリの確保と解放を行うようにアロケータを初期化します。
<code>NNS_FndInitAllocatorForFrmHeap()</code>	フレームヒープからメモリの確保と解放を行うようにアロケータを初期化します。
<code>NNS_FndInitAllocatorForUnitHeap()</code>	ユニットヒープからメモリの確保と解放を行うようにアロケータを初期化します。
<code>NNS_FndInitAllocatorForSDKHeap()</code>	NITRO-SDK のヒープからメモリの確保と解放を行うようにアロケータを初期化します。

いずれの初期化関数の場合でも、関数を呼ぶ前に使用するヒープをあらかじめ作成しておく必要があります。

3 メモリアロケータの関数

メモリアロケータの関数はメモリの確保を行う関数とメモリの解放を行う関数の 2 つだけです。また、実際の挙動は関連付けられているメモリ管理ライブラリの種類に依存します。次にメモリブロックの確保と解放を行う関数を示します。

表 3-1 メモリブロックの確保と解放を行う関数

関数	機能
NNS_FndAllocFromAllocator()	アロケータからメモリブロックを確保します。
NNS_FndFreeToAllocator()	アロケータにメモリブロックを返却してメモリブロックを解放します。

4 独自のメモリ管理ライブラリとメモリアロケータを関連付ける場合

独自のメモリ管理ライブラリをメモリアロケータで利用したい場合や、標準で用意されているメモリアロケータの振る舞いを変更したい場合があると思います。その場合は、メモリアロケータを独自に実装することにより可能になります。メモリアロケータを独自に実装する方法はメモリアロケータのソースを参考にしてください。

© 2004, 2005 Nintendo

任天堂株式会社の許諾を得ることなく、本書に記載されている内容の一部あるいは全部を無断で複製・複写・転写・頒布・貸与することを禁じます。