

# BuildNENR マニュアル

## BuildNEAR の使用方法

2008-05-30

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	使用方法 .....	4
3	定義テキストファイルの書式 .....	4
4	エンティティ応用例 .....	5
4.1	エンティティの利点 .....	5
4.2	エンティティ情報の応用例 .....	5

## 改訂履歴

改訂日	改訂内容
2008-05-30	NITRO-System の名称変更による修正 (NITRO-System を TWL-System に変更)。
2008-04-08	・改訂履歴の書式変更。 ・全体の校正。
2005-01-31	初版作成(BuildNENR_SimpleManual.txt の内容を転記)

# 1 はじめに

BuildNENR.exe は 定義テキストファイルをもとに NENR(G2D エンティティ ランタイムバイナリ)ファイルを生成する Windows アプリケーションです。本文章は BuildNENR.exe の使用方法を説明するものです。

## 2 使用方法

BuildNENR.exe [filename] [-o/...]

[filename]	必須	Entity 定義テキストファイル名。 変換したいパス+ファイル名+拡張子を指定します。
[-o/]	任意	データ出力ディレクトリを指定します。-o/の後にスペースを空けずにパスを記述します。存在しないディレクトリが指定された場合は出力が行われませんのでご注意ください。

例: > BuildNENR.exe c:/data/test.txt -o/d:/data

## 3 定義テキストファイルの書式

Entity 定義ブロック

<[type],[labelName], シーケンス番号, シーケンス番号, ... , シーケンス番号 >

[type] : Entity の種類を指定する文字。C はセルを表し、M はマルチセルを表します。

[labelName] : ラベル名(必須)。注意:C 言語の変数に使用できない文字の使用は避けてください。

(注意:シーケンス番号と Cell 番号の違いに注意してください。)

定義ブロックを記述した順に Entity Bank に格納されます。

例文:

```
< C, ent1, 0, 1, 3, 4 >
< C, ent2, 2 >
< C, ent3, 5, 6, 7, 8 >
< M, ent4, 5, 6, 7, 8 >
```

4 つの Entity が定義されたデータが生成されます。

## 4 エンティティ応用例

エンティティはセルアニメーション、マルチセルアニメーションといったデータ構造より上位レイヤ(ユーザプログラム)よりの情報を管理、格納する概念として定義されました。エンティティはゲームキャラクタの基本部分を定義する情報として利用されることを想定しています。詳細は API リファレンスのエンティティモジュール概要の項を参照してください。

### 4.1 エンティティの利点

エンティティを利用する利点を以下に示します。

- 単一の NANR(アニメーションファイル)内に複数のゲームキャラクタのアニメーションが含まれていても、それらを選択し、区別して使用することが可能となります。
- NANR(アニメーションファイル)内のシーケンスを任意の順番に再整列して使用することが可能になります。
- 定義ファイルがテキストフォーマットなので、編集(変更や修正)が容易です。

### 4.2 エンティティ情報の応用例

具体的なエンティティ情報の応用例を説明します。

複数のゲームキャラクタのアニメーションが格納された以下のような NANR ファイルが存在するとします。

表 4-1 サンプル・アニメーションファイル

シーケンス番号	アニメーションの意味
0	敵 A : 待機
1	敵 A : 移動
2	敵 A : 攻撃
3	敵 B : 待機
4	敵 B : 移動
5	敵 B : 攻撃
6	敵 B : 攻撃 (別パターン)

ここで、ゲームプロジェクトの固有ルールとして、エンティティアニメーションの 0 番は待機アニメーション、1 番は移動アニメーション、2 番は攻撃アニメーションなどと決定しておき、そのようにゲームエンジンを実装します。

- エンティティ A は 0, 1, 2 を参照する。
- エンティティ B は 3, 4, 5 を参照する。

上記のようにエンティティ情報を定義します。実際の定義スクリプトは下記ようになります。

```
< C, A, 0, 1, 2 >  
< C, B, 3, 4, 5 >
```

ここで、エンティティ B の攻撃のパターンを変更してみたいと考えたときに、エンティティ B の定義ファイルを 3, 4, 6 と変更するだけで、ゲームエンジンを変更することなく攻撃アニメーションの変更が可能になります。変更後の定義スクリプトを以下に示します。

```
< C, A, 0, 1, 2 >  
< C, B, 3, 4, 6 >
```

定義ファイルはスクリプトなどによって自動生成することも可能ですので、NITRO-CHARACTER 上で実際にデータを入れ替えたり、変更したりするよりも、より柔軟な対応が可能になることが期待できます。

Microsoft Windows、は米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

その他、記載されている会社名、製品名等は、各社の登録商標または商標です。

© 2004-2008 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。