

# TWL-System BuildNENR Manual

## Using BuildNENR

2008/05/30

**The content of this document is highly confidential  
and should be handled accordingly.**

**Confidential**

These coded instructions, statements, and computer programs contain proprietary information of Nintendo and/or its licensed developers and are protected by national and international copyright laws. They may not be disclosed to third parties or copied or duplicated in any form, in whole or in part, without the prior written consent of Nintendo.

## Table of Contents

---

1	Introduction .....	5
2	Use.....	6
3	Definition Text File Format .....	7
4	Example of Entity Application .....	8
4.1	Benefits of Using Entities .....	8
4.2	Sample Application of Entity Information .....	8

## Tables

---

Table 4-1 Sample Animation File .....	8
---------------------------------------	---

## Revision History

Revision Date	Description
2008/05/30	Made revisions in line with the NITRO-System name change (from NITRO-System to TWL-System).
2008/04/08	Changed the format of the Revision History. Made corrections throughout the entire document.
2005/01/31	Initial version. (Transferred contents of <code>BuildNENR_SimpleManual.txt</code> .)

# 1 Introduction

This document explains how to use `BuildNENR.exe`, which is a Windows application that creates NENR files (G2D entity run-time binaries) based on definition text files.

## 2 Use

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

[filename]	Required	Entity definition text filename. Specify the path + filename + extension to be converted.
[-o/]	Optional	Specify the data output directory. Include the path name after the -o/ without a space. Be aware that there will be no output if the specified directory does not exist.

Example:

```
BuildNENR.exe c:/data/text.txt -o/d:/data
```

### 3 Definition Text File Format

Entity definition block:

```
<[type],[labelName], Sequence Number, Sequence Number, ... , Sequence Number >
```

[type]: Character that specifies the Entity type: C indicates cell, and M indicates multicell.

[labelName]: Label name (required). Avoid using characters that cannot be used in C variables.  
(Be aware of the difference between sequence numbers and cell numbers.)

Items are stored in the Entity Bank in the order they are written in the definition block.

Examples:

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

Data that is defined by the four entities is created.

## 4 Example of Entity Application

An entity is defined as a concept to manage and store information close to the upper layer (user program) above data structures such as cell animation and multicell animation.

This entity should be used as information to define the basic portions of game characters. For further details, refer to the Entity Overview in the API reference.

### 4.1 Benefits of Using Entities

Benefits of using the entities are listed below.

- Even if multiple game character animations are included in a single NANR (animation file), they can be selected and used separately
- Sequences in an NANR (animation file) can be reordered and used
- Editing (changing and revising) the definition file is easy because it is in text format

### 4.2 Sample Application of Entity Information

The following is an example of applying entity information.

Assume that a NANR file has multiple game character animations stored.

**Table 4-1 Sample Animation File**

Sequence Number	Meaning of Animation
0	Enemy A: Wait
1	Enemy A: Move
2	Enemy A: Attack
3	Enemy B: Wait
4	Enemy B: Move
5	Enemy B: Attack
6	Enemy B: Attack (Separate pattern)

Implement a game engine where, as a rule for the game project, entity animation 0 is determined to be a wait animation, 1 is a move animation, 2 is an attack animation, and so on.

Define the entity information as follows.

- Entity A references 0, 1, and 2.
- Entity B references 3, 4, and 5.

The actual definition script is as follows.



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

You can change the attack pattern for Entity B simply by changing the Entity B definition file to 3, 4, 6 without changing the game engine itself. The definition script file after the changes is shown below.

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

Because it is possible for the definition file to be automatically generated from a script, more flexibility can be anticipated than by actually replacing and modifying data on NITRO-CHARACTER.

Windows is a registered trademark or trademark of Microsoft Corporation (USA) in the U.S. and other countries.

All company and product names in this document are the trademarks or registered trademarks of the respective companies.

© 2005-2009 Nintendo

The contents of this document cannot be duplicated, copied, reprinted, transferred, distributed, or loaned in whole or in part without the prior approval of Nintendo.