

TwISDK
TWL-SDK アプリ開発ガイド
TWL への対応

2009-01-07

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	5
2	アプリケーションの種類	5
2.1	ROMタイプ	5
2.1.1	NITRO ROM	5
2.1.2	HYBRID ROM	5
2.1.3	LIMITED ROM	5
2.2	メディア	6
2.2.1	カードアプリ	6
2.2.2	NANDアプリ	6
2.2.3	機能の制限	6
3	HYBRID ROM	7
3.1	概要	7
3.2	作成	7
3.3	動作	7
3.4	開発	7
3.4.1	ニンテンドーDS上でのメインメモリサイズ	7
3.4.2	TWLモード専用APIの使用	8
3.5	デバッグ	8
4	CODECモード	9
4.1	CODEC	9
4.2	CODECのモード	9
4.3	CODEC-TWLモード	10
4.4	モードの決定	10
5	アプリの起動制限	11
5.1	リージョン	11
5.2	ペアレンタルコントロール	11
6	ROMの認証	11
6.1	完全性検査	11
6.2	開発用本体	11
7	デバッグ目的でのSDカードの使用	12
7.1	概要	12
7.2	使用できる条件	13
7.3	使用法	13
7.3.1	ファイルの読み書き	13
7.3.2	SDカードの挿抜	13

8	その他.....	13
8.1.1	libsyscall.a, libsyscall.twl.a について	13

コード

コード 3-1	twl/ltmain_begin.h, twl/ltmain_end.hの使用法	8
コード 3-2	TWLモード専用APIの実装の 1 つの例.....	8

表

表 6-1	開発版アプリとデバッガ/本体の動作する組み合わせ	12
表 6-2	製品版アプリとデバッガ/本体の動作する組み合わせ	12
表 7-1	デバッグ目的でSDカードが使用できる条件.....	13

図

図 4-1	ニンテンドーDSとTWLのA/D D/A変換モジュール	9
-------	-----------------------------------	---

改訂履歴

改訂日	改訂内容
2009-03-12	7.3.1 ファイルの読み書きで、SD カードを使用する際の注意を追記
2009-01-07	WramMapping で指定する値を変更。
2008-12-15	8 章に libsyscall.a について追記。
2008-12-05	デバッグ目的での SDCard 利用について追記。 ferret コンポーネントについて追記。 NAND アプリのクローンブートの記述を修正
2008-12-04	inline を使う上での注意を追加
2008-11-06	完全性検査についての関数リファレンスへの参照を修正。
2008-10-14	CAMERA ライブラリと CODEC モードの関係について修正。 完全性検査について関数リファレンスへの参照を追加。
2008-09-16	初版。

1 はじめに

このドキュメントでは TWL-SDK でのニンテンドーDS / TWL 向けアプリケーション開発において、SDK が TWL に対応したことにより新たに追加された話題について取り扱います。

2 アプリケーションの種類

2.1 ROMタイプ

NITRO-SDK ではニンテンドーDS 向けのアプリケーションは、特に区別されることなく 1 種類のみが存在しました。この従来のタイプのアプリケーションを TWL-SDK では「ニンテンドーDS 専用ソフト」と呼びます。TWL-SDK ではこれに加えて下記の 2 種類が追加されます。

- TWL 対応ソフト
- TWL 専用ソフト

これら 3 種類のアプリケーションはそれぞれ NITRO / HYBRID / LIMITED と呼ばれる異なるタイプの ROM として開発されます。

2.1.1 NITRO ROM

NITRO ROM は、ニンテンドーDS 専用ソフトを開発するための ROM タイプです。ニンテンドーDS 専用ソフトはニンテンドーDS でも TWL でも動作することが可能ですが、TWL で動作する場合であっても TWL で追加されたデバイスや追加のメモリ領域を使用することはできません。

NITRO ROM を作成するにはリンクするライブラリを、TWL の文字列が含まれないものにした上で、コンポーネントに NITRO 版の mongoose (\$TwlSDK/components/mongoose/ARM7-TS[.thumb]) または ichneumon を使用します。

TWL-SDK の make ビルドシステムで NITRO ROM を作成するには TWLSDK_PLATFORM 環境変数に NITRO を指定します。

2.1.2 HYBRID ROM

HYBRID ROM は、TWL 対応ソフトを開発するための ROM タイプです。TWL 対応ソフトはニンテンドーDS でも TWL でも動作することが可能であり、また TWL で動作する場合には TWL で追加されたデバイスや追加のメモリ領域を使用できます。

HYBRID ROMの開発について詳しくは「3 HYBRID ROM」を参照してください。

2.1.3 LIMITED ROM

LIMITED ROM は、TWL 専用ソフトを開発するための ROM タイプです。TWL 専用ソフトは TWL でのみ動作し、ニンテンドーDS では動作しません。TWL で追加されたデバイスや追加のメモリを使用することができます。

LIMITED ROMを作成するにはリンクするライブラリを末尾が、TWL.LTD.aで終わるものにした上で、コンポーネントに racoonまたはferretを使用し、RSFのWramMappingプロパティにMAP2_TS_LTDを指定します。racoonとferret

の詳細に関しては[AboutComponents.pdf](#)を参照してください。

TWL-SDK の make ビルドシステムで LIMITED ROM を作成するには TWLSDK_PLATFORM 環境変数に TWL を指定した上で、Makefile 中で TWL_ARCHGEN に LIMITED を指定します。

2.2 メディア

ニンテンドーDS ではアプリケーションが格納されるメディアは DS カードに限定されていましたが、TWL では TWL の本体保存メモリにアプリケーションを格納することができます。

2.2.1 カードアプリ

カードアプリは DS カードまたは TWL カードに書き込まれて実行されるアプリケーションです。NITRO-SDK では全てのアプリケーションはカードアプリでした。

HYBRID / LIMITED ROM のカードアプリを作成するには makerom.TWL に与える RSF で Media プロパティを GameCard と指定します。なお Media プロパティのデフォルト値は GameCard ですので Media プロパティを指定しない場合もカードアプリとなります。

NITRO ROM ではカードアプリしか作成できません。NITRO ROM を作成する時に RSF に Media プロパティがあるとエラーになりますのでご注意ください。

2.2.2 NANDアプリ

NAND アプリは TWL の本体保存メモリに書き込まれて実行されるアプリケーションです。NAND アプリは TWL でしか動作しないため、基本的には LIMITED ROM として作成します。HYBRID ROM でも作成可能ですが、これはクローンブートのサポートを目的としています。NAND アプリのクローンブートは TWL-SDK 5.1 PR から対応しています。NAND アプリで HYBRID ROM を作成する場合は弊社窓口までご相談ください。

NAND アプリを作成するには makerom.TWL に与える RSF で Media プロパティを NAND と指定します。

NANDアプリの開発について、詳しくは[NandAppManual.pdf](#)を参照してください。

2.2.3 機能の制限

カードアプリか NAND アプリかによって以下のように利用できる機能に制限があります。

カードアプリで使用できない機能

- サブパナー
- 本体内蔵フォント
- NAND アプリセーブデータへのアクセス
- 写真データベースへのアクセス

NAND アプリで使用できない機能

- カード内 ROM およびバックアップへのアクセス

3 HYBRID ROM

3.1 概要

HYBRID ROM は TWL-SDK で追加された ROM タイプの 1 つです。

HYBRID ROM はニンテンドーDS 上で動作することができ、この場合ニンテンドーDS の機能のみを用いて動作します。また、HYBRID ROM は TWL 上でも動作することができ、この場合 TWL で追加されたデバイスやメモリ空間を使用することができます。

HYBRID ROM はニンテンドーDS 向けのアプリケーションでありながら、TWL で実行すると追加の機能が使用できるようなアプリケーションを実現するために用意されました。例えば、TWL 上で実行すると DS ワイヤレスプレイにおいてカメラで撮影した画像をエンブレムとして使用できる、などです。

3.2 作成

HYBRID ROM を作成するにはリンクするライブラリを末尾が `TWL.HYB.a` で終わるものにした上で、コンポーネントに HYBRID 版の `mongoose` (`$TwlSDK/components/mongoose/ARM7-TS.HYB[.thumb]/`)を使用し、RSF の `WramMapping` プロパティに `MAP2_TS_HYB`を指定します。

TWL-SDK の `make` ビルドシステムで HYBRID ROM を作成するには `TWLSDK_PLATFORM` 環境変数に TWL を指定した上で、`Makefile` 中で `TWL_ARCHGEN` に HYBRID を指定します。

3.3 動作

ニンテンドーDS ではアプリケーションは ARM9 実行バイナリと ARM7 実行バイナリの 2 つのバイナリとオーバーレイから構成されていました。ニンテンドーDS の IPL はこの 2 つのバイナリをそれぞれのプロセッサ用にメモリに読み込み、処理を開始させます。

HYBRID ROM ではこの 2 つのバイナリに ARM9 TWL 専用実行バイナリと ARM7 TWL 専用実行バイナリが加わり、計 4 つのバイナリとオーバーレイ/再配置モジュールから構成されます。ニンテンドーDS で HYBRID ROM が実行された場合、ニンテンドーDS の IPL は ARM9 実行バイナリと ARM7 実行バイナリのみをメモリに読み込み、処理を開始させます。TWL で HYBRID ROM が実行された場合、TWL の IPL は 4 つのバイナリ全てを読み込み、処理を開始させます。この仕組みによりニンテンドーDS と TWL の両方で動作が可能であり、かつ TWL での追加の機能が使用可能となっています。

3.4 開発

HYBRID ROM の開発で気をつけるべき点は主に次の 2 つです。

- ニンテンドーDS 上でのメインメモリサイズ
- TWL モード専用 API の使用

3.4.1 ニンテンドーDS上でのメインメモリサイズ

HYBRID ROM が TWL 上で実行される場合は 16MB のメインメモリ空間が使用できますが、ニンテンドーDS 上で動作する場合、使用できるメインメモリ空間は 4MB しかありません。このためニンテンドーDS 上で動作する場合には極力 TWL 上でのみ必要なコードをロードしないようにする必要があります。

簡単な対応策としてはコードを ARM9 TWL 専用実行バイナリに格納されるようにする方法があります。アプリケーションのコードはデフォルトでは ARM9 実行バイナリに格納されます。これを ARM9 TWL 専用実行バイナリに格納されるようにするためには `twl/ltdmain_begin.h` ヘッダと `twl/ltdmain_end.h` ヘッダのペアを使用します。この 2 つのヘッダをソースコードで `include` すると、2 つに挟まれた領域で定義されるコードとデータが ARM9 TWL 専用実行バイナリに格納されるようになります。

コード 3-1 twl/ltdmain_begin.h, twl/ltdmain_end.h の使用法

```
// ARM9 実行バイナリに含まれるコード
#include <twl/ltdmain_begin.h>
// ARM9 TWL 専用実行バイナリに含まれるコード
#include <twl/ltdmain_end.h>
// ARM9 実行バイナリに含まれるコード
```

但しこの対応策を使用した場合で `inline` 関数を使用する際は注意が必要です。ARM9 実行バイナリと ARM9 TWL 専用実行バイナリの両方から参照されている `inline` 関数が `inline` 化されなかった場合、コードが必ず ARM9 実行バイナリに保存される保障がありません。もし ARM9 実行バイナリに保存されなかった場合、NITRO 互換モードで動いた時に参照ができなくなります。この場合は `inline` 化されるまでコードを小さくするか、通常関数を使用して下さい。

柔軟性が高い対応策はオーバーレイもしくは再配置モジュールを使用することです。TWL 上でのみ必要なコードをオーバーレイ/再配置モジュールにまとめておき、TWL 上で実行されている場合にのみ読み込むようにします

3.4.2 TWLモード専用APIの使用

TWL-SDK で追加された TWL モード専用の API 群はニンテンドーDS では動作しません。このため HYBRID ROM がニンテンドーDS 上で動作する場合にはこれらの API を使用しないように実装する必要があります。

なお、TWL モード専用 API 群の関数実体は ARM9 TWL 専用実行バイナリに含まれるよう定義されており、ニンテンドーDS 上ではメインメモリにロードされないようになっています。ただし、これらの API は TWL 上で動作しているかどうかを判別する `inline` 関数で包まれており、ニンテンドーDS 上で動作している場合に呼び出しても問題は発生しません。ニンテンドーDS 上で使用すると、その旨を知らせる、または失敗を示すエラーコードが返されるか、単に何も返しません。

コード 3-2 TWL モード専用 API の実装の 1 つの例

```
SDK_INLINE CAMERAResult CAMERA_Init(void)
{
    if (OS_IsRunOnTwl() == TRUE)
    {
        CAMERA_InitCore();
        return CAMERA_I2CInitCore(CAMERA_SELECT_BOTH);
    }
    return CAMERA_RESULT_FATAL_ERROR;
}
```

3.5 デバッグ

HYBRID ROM を NITRO 互換モードで動作させるにはニンテンドーDS 上で動作させます。TWL モードで動作させるには開発用 TWL 本体で動作させます。

IS-TWL-DEBUGGER を使用すると 1 台で NITRO 互換モードと TWL モードを切り替えて動作させることができます。

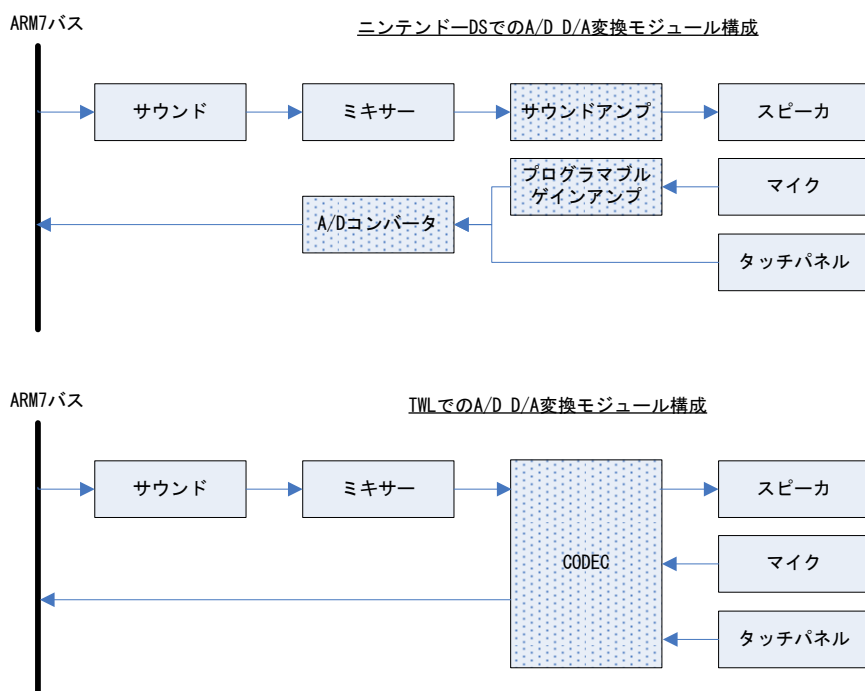
4 CODECモード

4.1 CODEC

TWL では D/A、A/D 変換を行うハードウェアモジュールが CODEC と呼ばれるモジュールに変更されています。

以下にニンテンドーDS と TWL のシステムブロック図から該当箇所を抜き出したものを掲載します。なお、この図では本質的でない部分を省いています。この図からわかるように CODEC はサウンドの出力、マイク / タッチパネルの入力を処理します。

図 4-1 ニンテンドーDS と TWL の A/D D/A 変換モジュール



4.2 CODECのモード

CODEC には CODEC-DS モードと CODEC-TWL モードと呼ばれる 2 つのモードがあります。

CODEC-DS モードはニンテンドーDS との互換性を最大限に確保するよう用意されたモードです。CODEC-DS モードでは CODEC は DS の時に個々に用意されていたモジュールの動作をエミュレーションし、同じ挙動を再現します。

CODEC-TWL モードは CODEC の性能を最大限発揮するように用意されたモードです。CODEC-TWL モードではニンテンドーDS の時には使用できなかったいくつかの拡張機能を使用することができます。

CODEC モードは RSF の CodecMode プロパティで指定され、この値に基づいてアプリケーション起動前に CODEC モードの変更が行われます。CODEC モードをアプリケーションの実行中に切り替えることはできません。なお NITRO ROM では CODEC モードは CODEC-DS モードで固定となり選択できません。

TWL-SDK のビルドシステムでは CODEC モードのデフォルトは CODEC-TWL モードとなっています。

4.3 CODEC-TWLモード

CODEC-TWL モードで追加される機能は以下のとおりです。

- 47.61KHz でのサウンド出力、マイクサンプリング
- 16bit 精度でのマイクサンプリング
- 120 段階のマイクアンプ
- マイク入力とサウンド出力への IIR フィルタ
- 音声の強制出力
- CODEC によるマイク/タッチパネル自動サンプリング

一方 CODEC-TWL モードで追加される制限は以下のとおりです。

- 利用可能なマイクサンプリング周波数が CODEC 処理周波数に応じて 4 種類に制限される
- 8bit、12bit 精度でのマイクサンプリングが使用不可

CODEC-TWL モードでは CODEC の処理周波数を 32.73KHz と 47.61KHz とから選択することができます。CODEC の処理周波数はマイクサンプリングとサウンド出力に同時に影響を与えるため、32.73KHz でマイクサンプリングを行いながら、47.61KHz でサウンド出力といったことはできません。切り替えながら使用することは可能です。

音声の強制出力はユーザが指定しているボリュームの値や、ヘッドホンがヘッドホンジャックに挿されているかどうかによらず、TWL の本体スピーカから任意の音量での音声出力を可能にします。この機能はカメラ撮影時のシャッター音の再生に使用します。

CODEC によるマイク/タッチパネル自動サンプリングは、従来 ARM7 でソフトウェア的に行われていたこれらの自動サンプリング処理を CODEC ハードウェアで行うものです。特にマイクサンプリング時の ARM7 処理負荷が軽減され、マイクと無線などの ARM7 側処理を同時に使用した場合の処理落ちを軽減することができます。

4.4 モードの決定

CODEC モードはアプリケーションごとに自由に選択することができますが、アプリケーションの実行中に切り替えることができないため、注意して決定する必要があります。以下に決定に影響を与える要素を列挙します。

- カメラを使用するかどうか
CAMERA ライブラリは CODEC が CODEC-TWL モードでないと使用できないようになっています。よってカメラを使用する場合は、必ず CODEC-TWL モードを使用する必要があります。
- HYBRID ROM かどうか
HYBRID ROM はニンテンドーDS と TWL の両方で動作する必要があります。CODEC-TWL モードで追加される機能が不要なのであれば、CODEC-DS モードを使用することでニンテンドーDS 上と TWL 上でハードウェアの挙動をあわせることができるため開発が容易になります。

5 アプリの起動制限

5.1 リージョン

ニンテンドーDS では中国向けに作られたアプリケーション以外は全てのニンテンドーDS で動作させることができました。一方 TWL 上で動作する HYBRID ROM、LIMITED ROM にはリージョンの概念があります。

TWL では日本、北米、欧州、豪州の 4 つのリージョンがあります。アプリケーションのリージョンはカードアプリ / NAND アプリによらず RSF の CardRegion プロパティで指定します。アプリケーションごとに適切なリージョンを設定してください。

5.2 ペアレンタルコントロール

TWL 上で動作する HYBRID ROM、LIMITED ROM にはレーティングに基づくペアレンタルコントロールが適用されます。

アプリケーションにはリージョンごとのレーティング機関によって与えられたレーティング情報を付加しなければなりません。レーティング情報の付加は MasterEditor によって行います。

6 ROMの認証

6.1 完全性検査

TWL 上で動作する HYBRID ROM、LIMITED ROM では ROM の読み取りアクセス時に ROM の内容に対する完全性検査が行われます。

カードアプリであればCARDライブラリを使用したROMへのアクセスとFSライブラリでのromアーカイブへのアクセス時に、NANDアプリであればFSライブラリでのromアーカイブへのアクセス時に、読み取ったデータに対して内部でハッシュの算出とアプリ起動時に認証されたハッシュテーブルとの比較が行われます。詳細は関数リファレンスの[「FS」→「概要」→「ROMアーカイブ」](#)を参照してください。

特に HYBRID ROM ではニンテンドーDS 上で動作する場合に比べて TWL 上で動作する場合には ROM へのアクセス速度が遅くなることに注意する必要があります。

完全性検査に失敗した場合、OS_TPanic で停止します。

6.2 開発用本体

TWL-SDK で作成した TWL 用アプリを動作させるには開発用 TWL 本体が必要です。

NITRO ROM や HYBRID ROM の NITRO 互換モードは開発においても市販のニンテンドーDS 本体を使用することができます。一方すべての ROM タイプにおいて開発中の SRL を市販の TWL 本体で動作させることはできません。HYBRID ROM、LIMITED ROM の開発には開発用 TWL 本体を使用する必要があります。

逆に製品版の HYBRID ROM、LIMITED ROM は開発用 TWL 本体では動作しません。

それぞれの本体で動作するアプリは下記の表の通りになります。

表 6-1 開発版アプリとデバッガ/本体の動作する組み合わせ

	NITRO ROM	HYBRID ROM		LIMITED ROM
		NITRO 互換モード	TWL モード	
ニンテンドーDS 本体	○	○	×	×
IS-NITRO-EMULATOR	○※1	○	×	×
IS-NITRO-CAPTURE	○	○	×	×
IS-TWL-DEBUGGER	○※2	○※2	○※2	○※2
IS-TWL-CAPTURE	○	×	○	○
開発用 TWL 本体	○	×	○	○
市販 TWL 本体	×	×	×	×

表 6-2 製品版アプリとデバッガ/本体の動作する組み合わせ

	NITRO ROM	HYBRID ROM		LIMITED ROM
		NITRO 互換モード	TWL モード	
ニンテンドーDS 本体	○	○	×	×
IS-NITRO-EMULATOR	×	×	×	×
IS-NITRO-CAPTURE	○	×	×	×
IS-TWL-DEBUGGER	○※2	×	×	×
IS-TWL-CAPTURE	○	×	×	×
開発用 TWL 本体	○	×	×	×
市販 TWL 本体	○	×	○	○

※1 IS-NITRO-EMULATOR では DS/TWL カードから実行することができません。

※2 IS-TWL-DEBUGGER では特殊デバイスを使用した DS/TWL カードが動作しません。

7 デバッグ目的でのSDカードの使用

7.1 概要

最終製品での SD カードへのアクセスはガイドラインで禁止されますが、開発時に限りデバッグ目的で SD カードを使用することができます。

7.2 使用できる条件

デバッグ目的での SD カードの使用ができるのは、ROM タイプが HYBRID ROM または LIMITED ROM のどちらかで、かつ DEBUG ビルドまたは RELEASE ビルドの場合のみです。NITRO ROM の場合や FINALROM ビルドでは使用できません。なお、FINALROM ビルドでなければ ROM 出しできないことにご注意ください。

表 7-1 デバッグ目的で SD カードが使用できる条件

	NITRO ROM	HYBRID ROM	LIMITED ROM
DEBUG ビルド	×	○	○
RELEASE ビルド	×	○	○
FINALROM ビルド	×	×	×

7.3 使用法

7.3.1 ファイルの読み書き

SD カードは FS ライブラリに `sdmc` アーカイブとしてマウントされます。このため `sdmc` をアーカイブ名として FS ライブラリを使用することで、SD カード上のファイルを読み書きすることができます(「HYBRID ビルドのカードブートアプリケーション」で SD カードを利用する場合は、注意事項※1があります)。

例えば “`sdmc:/log.txt`” というパスであれば、SD カード上のルートディレクトリ直下の `log.txt` を表します。

※1 「HYBRID ビルドのカードブートアプリケーション」で SD カードを利用する場合は、FS ライブラリ初期化時に、`FS_Init()`に加えて `FS_InitFatDriver()`も実行する必要があります。

7.3.2 SDカードの挿抜

SDカードは本体保存メモリと異なり、任意のタイミングで抜き差しすることができます。SDカードが抜き差しされたことを知るには[FS_RegisterEventHook](#)関数を使用します。この関数を使用してコールバックを登録することでSDカードが挿抜されたタイミングでコールバックが呼び出されるようになります。

なお、mini SD カードを SD カードアダプタ経由で使用している場合に、SD カードアダプタを TWL 本体に挿したまま mini SD カードのみを抜き差ししても、これを検出することはできませんのでご注意ください。

8 その他

8.1.1 libsyscall.a, libsyscall.twl.a について

`libsyscall.a` と `rom_header.template.sbin` については弊社窓口からお受け取りください。HYBRID ROM、LIMITED ROM では `libsyscall.twl.a` と `rom_header.LTD.sbin` も追加が必要となりますが、これらは SDK 付属のものを使用してください。

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。

© 2009 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。