Quick Start Guide

TWL-SDK

Version 2008/11/04

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

Revision History

Version	Description
2008/11/04	Added a detailed list of supported build platforms and a note regarding build environments to 3 Setting the Environment Variables.
2008/09/30	Removed the "Updating the IPL" section.
2008/09/10	Initial version.

1 Introduction

This document explains how to install TWL-SDK and create a simple application.

This document contains the following sections.

1. Introduction

2. Preparing the Development Tools

3. Setting the Environment Variables

4. Building the TWL-SDK Tree

5. Trying the Samples

6. Writing a Simple Program

7. Using Build Switches

Released: February 19, 2010

2 Preparing the Development Tools

The build for the current TWL-SDK has been confirmed for Microsoft Windows XP Professional Service Pack 2.

The following tools are necessary to build (compile, etc.) the TWL-SDK.

- CodeWarrior for NINTENDO DSi

- Cygwin

In addition, one of the following tools must be used for debugging.

- IS-TWL-DEBUGGER (software/hardware)
  Supported platforms: LIMITED ROM
                              HYBRID ROM (Runs on Nintendo DSi)
                              HYBRID ROM (Runs on Nintendo DS and Nintendo DS Lite)
                              NITRO ROM

- IS-NITRO-DEBUGGER / IS-NITRO-EMULATOR
  Supported platforms: HYBRID ROM (Runs on Nintendo DS and Nintendo DS Lite)
                              NITRO ROM

When a program has been built in an environment where only IS-NITRO-DEBUGGER was installed, that program's code will be able to display strings when run on IS-NITRO-DEBUGGER, but will not be able to display strings when run on IS-TWL-DEBUGGER software and hardware. If you want to also display strings when the program is run on IS-TWL-DEBUGGER software and hardware, you must build the program in an environment where the IS-TWL-DEBUGGER software is installed.

CodeWarrior for NINTENDO DSi, IS-TWL-DEBUGGER (software), and IS-NITRO-DEBUGGER are available from www.warioworld.com. For all other tools, please contact the distributors.

For information on installing Cygwin, see `CygwinPackageList.rtf` in the `$TwlSDK/docs/SDKTools` directory after unzipping the SDK.

For installation procedures, see the documentation for the individual tools.

3 Setting the Environment Variables

The two environment variables TWLSDK_ROOT and TWLSDK_PLATFORM must be set for TWL-SDK.

Set TWLSDK_ROOT to the absolute path of the expanded TWL-SDK directory. If nothing is set, the default value is C:\TwlSDK. This directory is referred to below as $TwlSDK.

TWLSDK_ROOT can also be configured with the $TwlSDK/setup script once the TWL-SDK has been extracted. The configuration procedure using the setup script is shown below.

```
% cd $TwlSDK
% source setup
```

Set the platform for the target build in the environment variable TWLSDK_PLATFORM. This environment variable specifies the code that must be generated for the platform. If nothing is set, the build will end with an error. For more information on TWLSDK_PLATFORM and other build options, see $TwlSDK/docs/SDKRules/Rule-Defines.html after extracting the SDK.

Released: February 19, 2010

4 Building the TWL-SDK Tree

The unzipped TWL-SDK contains pre-built libraries and tools. Normally, there is no need to build everything from the top of `$TwlSDK`.

The following example shows a build of a 3D_Pol_LightColor sample demo.

Start a shell (for example `bash)` with Cygwin, go to `$TwlSDK/build/demos/gx/UnitTours/3D_Pol_LightColor`, and enter `% make` to start the build.

If an error occurs during the process, it is possible that a mistake was made in the settings or that a bug may exist in the package. Review the settings before contacting the distributor.

In Chapter 3 Setting the Environment Variables, the value that was specified for the environment variable `TWLSDK_PLATFORM` will determine which platform code will be generated for.

In the absence of other specified build options, setting `TWLSDK_PLATFORM` to `TWL` will generate a Release build of a HYBRID ROM for ARM by default, and setting `TWLSDK_PLATFORM` to `NITRO` will generate a Release build of a NITRO ROM for ARM by default. For more information on ROM types and how to generate them, after unzipping the TWL-SDK, see `$TwlSDK/docs/TechnicalNotes/AboutTwlApplication.pdf`.

5 Trying the Samples

Run the samples to verify whether the build has completed successfully.

1. Start IS-TWL-DEBUGGER (software).

(For NITRO code, start `IS-NITRO-DEBUGGER` instead.)

2. On the **File** menu, click **Open**.

3. Specify an SRL file in the **Open** dialog box.

In this example, we specify the file shown below, which was created in the build we just completed in the previous section.

`$TwlSDK/build/demos/gx/UnitTours/3D_Pol_LightColor/bin/ARM9-TS.HYB/Release/main.srl`

(For Nitro code, use `$TwlSDK/build/demos/gx/UnitTours/3D_Pol_LightColor/bin/ARM9-TS/Release/main.srl`.)

4. On the **Debug** menu, click **Run**.

(Alternatively, you can click the **Run** button located on the toolbar of the IS-TWL-DEBUGGER (software), as shown in the red circle below.)
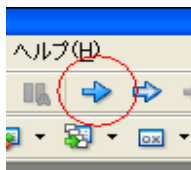
5. If everything is working properly, the following two cubes are displayed on the upper screen of the IS-TWL-DEBUGGER (software). The color of the cube on the right changes constantly.
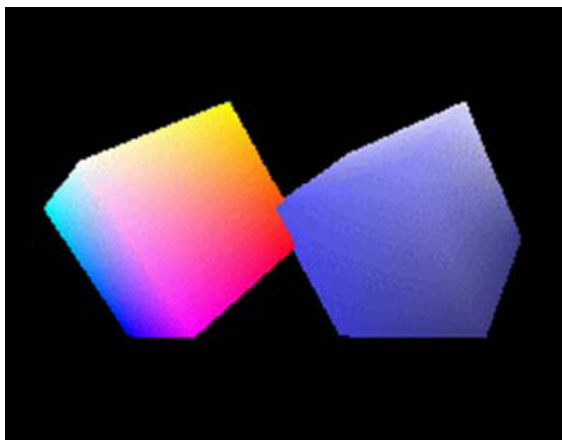
6. Click the **Pause** button (as shown in the red circle below) to stop execution of the file.
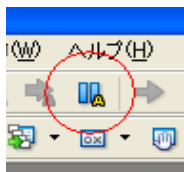
6 Writing a Simple Program

This section describes how to write a simple program.

1. Make an appropriate work directory and copy the following files into it.

   ```
   $TwlSDK/build/demos/_template/src/main.c
   $TwlSDK/build/demos/_template/makefile
   ```

2. Make the following changes to `main.c`.

Lines numbered 16 and later

```
#include <nitro.h>
void NitroMain(void)
{
    OS_Init( );
    OS_Printf( "Hello World of TWLid. \n" );          ← Add this line
    while (1){}
}
```

3. Run `make`. If an error or warning is displayed, review step 2 to check for errors.

4. Start IS-TWL-DEBUGGER. On the **File** menu, click **Open,** as you did in the previous section, then specify `bin/ARM9-TS.HYB/Release/main.srl` in the work directory. (For NITRO code, specify `bin/ARM9-TS/Release/main.srl` instead.)

5. Click the **Run** button. The debug output window displays the following.

   ```
   Hello World of TWLid.
   ```

6. Click the **Pause** button to stop the `.bin` file emulation.

7 Using Build Switches

TWL-SDK has several build settings. The `Release` versions of the libraries will be linked by default, but the `Debug` and `ROM` versions can be built by changing macro build-time settings using *build switches*.

See `$TwlSDK/docs/SDKRules/Rules-Defines.html` for more information on build switches.

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