CTR-SDK
Texture Packager Library

Texture Packager Tool Backgrounder

Version 1.4

PROVISIONAL TRANSLATION

CTR-06-0028-001-A
Released: Review

Table of Contents

Code

Tables

Figures

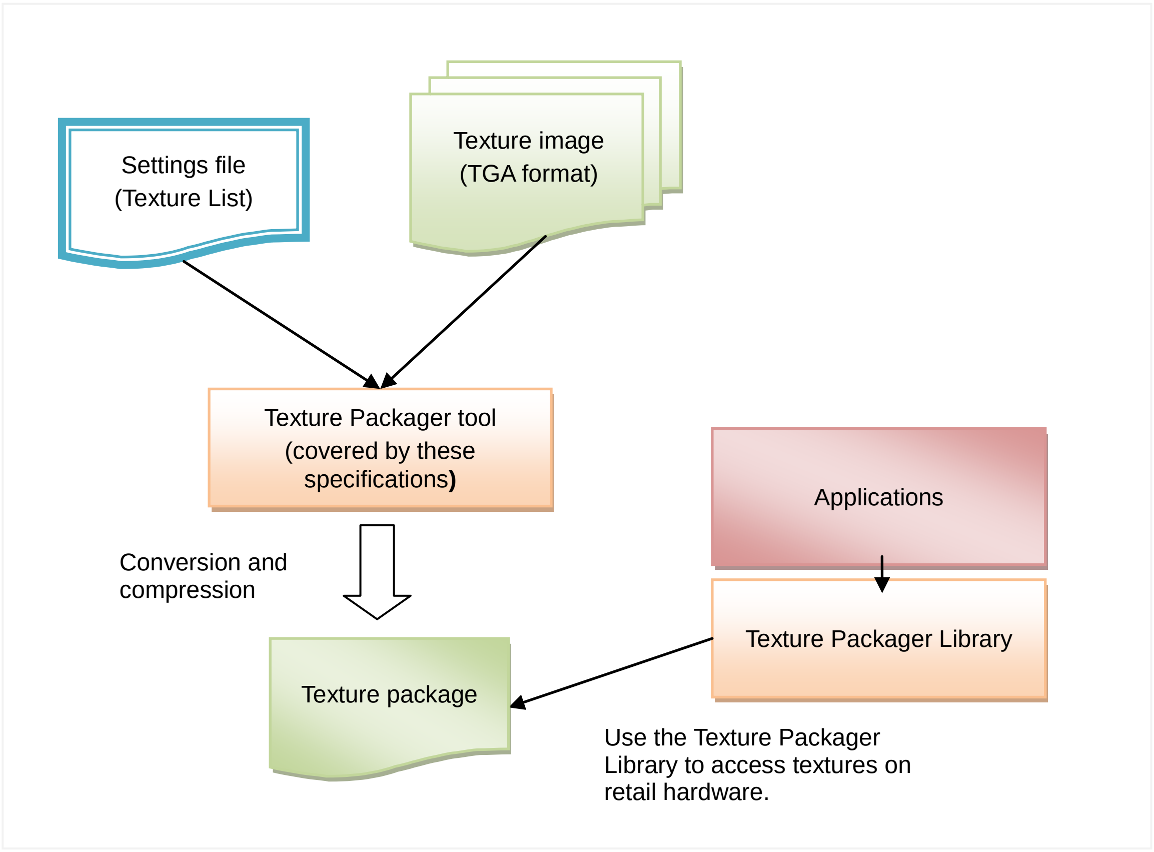
1 Overview of Features

The Texture Packager tool (`ctr_TexturePackager.exe`) is a command-line tool that reads image files in TGA format based on a settings file (a list of textures), converts these to formats that the CTR system can use (including compressed formats), and creates a package file containing multiple textures.

The Texture Packager tool has the following features.

- Pixel sorting
- Format conversion (including compressed formats)
- Mipmap creation
- Data creation complying with the CTR system's VRAM texture data alignment (128-byte)

Figure 1-1 Texture Packager Library Structure



2 Individual Features

2.1.1 Settings File (Texture List)

To process multiple texture files, the tool uses a text-based settings file that denotes the paths and other information for each texture file to process in XML format. Specify the `-i` option when running `ctr_TexturePackager.exe` to use a settings file.

Within the settings file, the root node is the `<ctr_texturePackager>` node, and child nodes are `<texture>` nodes. Each `<texture>` node specifies a texture's type and related information. Each `<texture>` element may also have `<imagefile>` child elements. Each `<imagefile>` element corresponds to one texture file to load. The `<imagefile>` src attribute specifies the path to the texture file. These paths are relative to the texture list file, or relative to the path from the base directory specified by the `-b` command line option.

Any other element tags are ignored by `ctr_TexturePackager.exe`. The file must still start with the `<?xml>` tag declaration at the top to be valid XML.

Code 2-1

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ctr_texturepackager>
   <!-- The paths to textures included in this package are specified below in node
attributes. (This line is a comment.) -->
   <texture type="2D">
      <imagefile src="data/images001.tga" />
   </texture>
   <texture type="2D">
      < imagefile src ="data/images002.tga" />
   </texture>
   <texture type="2D">
      < imagefile src ="data/images003.tga" />
   </texture>
</ ctr_texturepackager >
```

2.1.2 <texture> Node

A `<texture>` node describes a texture's attributes. The following strings are valid attribute names.

2.1.2.1 type

Specifies the texture type. This attribute is required. The values that can be specified are listed below.

- `2D`: Standard 2D texture.
- `1D`: 1D texture, called a "lookup table."
- `cube`: Cube-map texture. This is a group of six images.

2.1.2.2 nocomp

If specified, this texture cannot be compressed.

2.1.2.3 miplimit

This restricts the texture mipmap level. The attribute value states the number of images in a mipmap, including the top-level surface.

- A value of 1 indicates no mipmapping.
- Do not use 0, as this indicates no value specified.

2.1.2.4 format

This explicitly specifies the output format for this texture. The attribute value is a string specifying the format. The formats that can be specified are as follows.

Table 2-1 Specifiable Formats

Format	DMPGL Format	DMPGL Type	Number of Bytes
RGBA4444	RGBA_NATIVE_DMP	UNSIGNED_SHORT_4_4_4_4	2
RGBA5551	RGBA_NATIVE_DMP	UNSIGNED_SHORT_5_5_5_1	2
RGBA8888	RGBA_NATIVE_DMP	UNSIGNED_BYTE	4
RGB565	RGB_NATIVE_DMP	UNSIGNED_SHORT_5_6_5	2
RGB888	RGB_NATIVE_DMP	UNSIGNED_BYTE	3
ETC1	ETC1_RGB8_NATIVE_DMP		
ETC1_A4	ETC1_ALPHA_RGB8_A4_NATIVE_DMP		
A8	ALPHA_NATIVE_DMP	UNSIGNED_BYTE	1
A4	ALPHA_NATIVE_DMP	UNSIGNED_4BITS_DMP	0.5
L8	LUMINANCE_NATIVE_DMP	UNSIGNED_BYTE	1
L4	LUMINANCE_NATIVE_DMP	UNSIGNED_4BITS_DMP	0.5
LA88	LUMINANCE_ALPHA_NATIVE_DMP	UNSIGNED_BYTE	2
LA44	LUMINANCE_ALPHA_NATIVE_DMP	UNSIGNED_BYTE_4_4_DMP	1
HL8	HILO8_DMP_NATIVE_DMP	UNSIGNED_BYTE	1
REF	LUMINANCEF_DMP	FLOAT	4

Note: `LUMINANCEF_DMP` is a 1D texture called a "lookup table." For details, see the DMPGL specifications.

2.1.2.5 etcmethod

When using either the ETC1 or ETC1 A4 formats, you must explicitly specify ETC1 compression. The attribute value is a string specifying the ETC1 compression method. The table below shows valid values for this attribute.

Table 2-2 Valid ETC1 Compression Methods

ETC1 Compression Methods
Fast
Medium
Slow
FastPerceptual
MediumPerceptual
SlowPerceptual

Fast, *medium*, and *slow* refer to the speed of compression. Slower compression can produce better picture quality.

The *perceptual* values refer to compression methods that reduce errors in the green component, which human eyes are particularly sensitive to. We recommend non-perceptual compression for textures for normal maps.

The data size after compression is not affected by the compression method used.

2.1.3 <imagefile> Node

The *src* attribute is required for `<imagefile>` nodes. (Any backslashes “¥” used in the *src* value are converted to forward slashes “/”. Be aware of this when reading this value at runtime.) The *dir* attribute is also required if the parent `<texture>` node specifies **cube** for its *type* attribute. The *dir* attribute specifies the direction in which images are applied to the cube map. The following values can be specified for the *dir* attribute.

- **+x**: X-axis positive direction
- **-x**: X-axis negative direction
- **+y**: Y-axis positive direction
- **-y**: Y-axis negative direction
- **+z**: Z-axis positive direction
- **-z**: Z-axis negative direction

Code 2-2

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- The image file nodes are set below. (This line is a comment.) -->
<ctr_texturepackager>
   <!-- The paths to textures included in this package are set below in node
attributes. (This line is a comment.) -->
   <texture type="2D" miplimit="2" format="RGBA8888">
      <imagefile src="data/images001.tga" />
   </texture>
   <texture type="2D" nocomp="" miplimit="5" format="RGB888">
      <imagefile src="data/images002.tga" />
```

```
    </texture>
    <texture type="cube" miplimit="2" format="RGB888">
        <imagefile dir="+x" src="data/cube_images01.tga" />
        <imagefile dir="-x" src="data/cube_images02.tga" />
        <imagefile dir="+y" src="data/cube_images03.tga" />
        <imagefile dir="-y" src="data/cube_images04.tga" />
        <imagefile dir="+z" src="data/cube_images05.tga" />
        <imagefile dir="-z" src="data/cube_images06.tga" />
    </texture>
    <texture type="2D" format="RGBA4444">
        <imagefile src="data/images003.tga" />
    </texture>
    <texture type="2D" format="A8">
        <imagefile src="data/alpha001.tga" />
    </texture>
    <texture type="2D" format="ETC1" etcmethod=”FastPerceptual”>
        <imagefile src="data/etc1001.tga" />
    </texture>
    <texture type="1D" format="REF">
        <imagefile src="data/gas001.tga" />
    </texture>
</ctr_texturepackager>
```

Note: Any backslashes “¥” used in the *src* value are converted to forward slashes “/”. Be aware of this when reading texture files at runtime with regard to path separators in file paths.

2.2 Excluding Static Textures

If you have a texture used by multiple scenes that you would like to include in a separate texture package as a static texture, use the `-e` command-line option specifying the static texture list file to create a texture package consisting of the static textures taken from the list of texture files specified for the `-i` command-line option.

At runtime, the static textures and the texture package files for each scene are loaded, and both file types are searched for textures. If a texture is included in multiple texture packages, VRAM memory is wasted when all the packages are loaded. Use the `-e` option to resolve this.

The format of a static texture list file is the same as for the settings file (texture list).

2.3 File Format and Texture Format

The `ctr_TexturePackager.exe` program can only load image files in .tga format. If the *format* attribute is not set in the settings file, the `ctr_TexturePackager.exe` program chooses the output format based on the input image format.

A table of supported formats is shown below. There is no support for formats not in this table.

Table 2-3 Supported Formats

Input Format	Output Format	Comment
Index color	`ETC1_RGB8_NATIVE_DMP / RGB_NATIVE_DMP(RGB888)`	Index color converted to 24-bit RGB. ETC1 format when compression is enabled. RGB888 format when compression is disabled.
Grayscale	`LUMINANCE_NATIVE_DMP(L8)`	
24-bit color (RGB888)	`ETC1_RGB8_NATIVE_DMP / RGB_NATIVE_DMP(RGB888)`	ETC1 format when compression is enabled. RGB888 format when compression is disabled.
32-bit color (ARGB8888)	`ETC1_ALPHA_RGB8_A4_NATIVE_DMP / RGBA_NATIVE_DMP(RGBA8888)`	ETC1 format with alpha values when compression is enabled. RGBA8888 format when compression is disabled.

2.4 Automatic Mipmapping

Automatically creates mipmapped images when texture files are loaded.

2.5 Difference Conversion

The number of textures used in a large scene increases, leading to longer times required for conversion and especially compression. The `ctr_TexturePackager.exe` program compares the texture packages converted previously with a list of newly input texture files and only converts those textures that are new or different. This behavior is the default. Use the `-a` command line option if you wish to reconvert all textures.

Add the *nocomp*, *miplimit*, *format*, and *etcmethod* attributes to the texture file list for difference conversion to update only those textures that have changed.

2.6 Icon Creation

Use the `-dsl` command-line option to specify icon creation, in which one texture package file is created from one image file. Operation in icon creation mode is different from normal package file creation as follows.

- Texture information is not output to the texture package file. (Only the texture data itself is output.)
- No mipmaps are generated.
- The width and height of the loaded image must be multiples of 8. (Dimensions must be powers of 2 in standard mode.)
- The format after conversion is `RGB_NATIVE_DMP` (`UNSIGNED_SHORT_5_6_5`).

3 Command Reference

Specify the settings file (*texture_list*) and texture package file (*texture_package*) arguments when running the program.

Each option is explained below.

Code 3-1

```
ctrTexturePackager : compress and package textures for CTR.
Usage.
ctr_TexturePackager.exe [options] -i <texture_list> -o <texture_package>

Output.
   -o <texture_package>: Texture package file. (Can be omitted.)

Input.
      -i <texture_list>: Texture list file.
       -l <src_texture>: A single texture file.

Options.
                    -a: Reconvert all textures.
      -e <exclude_list>: List of textures to exclude.
                   -nc: No compression. (Applies to all textures)
    -b <input_base_dir>: Base directory for relative paths to image files to load.
                  -dsl: Icon creation mode. (Image width and height must be multiples
of 8, and not just powers of 2)
                  –nw4c: Use to load a tga file output by the nw4c_tga Photoshop
plugin and create an icon.
```

3.1 -o

Specifies the texture package file to output. If omitted, the program will use the filename specified for the `-i` argument, replacing the filename extension with `.ctpk`.

3.2 -i

Specifies the texture settings file to input. Refer to section 2.1.1 Settings File (Texture List).

3.3 -l

Specifies a single texture file (`.tga`) to convert. If specified, the `-i` option is ignored. (Can be omitted.)

If the `–o` option (output texture package file specification) is omitted, the program will use the filename specified for this argument, replacing the filename extension with `.ctpk`. In addition to the texture file, the `<src_texture>` section may also include the attribute values explained in section 2.1.1 Settings File (Texture List).

Example 1:

```
ctr_TexturePackager.exe -l "image1.tga,format(RGBA8888)"
```

Example 2:

```
ctr_TexturePackager.exe -l
"image2.tga,miplimit(2),format(ETC1),etcmethod(MediumPerceptual)"
```

3.4 -a

Disables difference conversion and converts all textures specified in the texture file list.

3.5 -e

Specifies a text file (in XML format) listing the texture files to exclude from the package. The format is identical to that for the settings file (`texture_list`). If no filename is given after the `–e` option, `textureExclude.xml` is used. If the option is omitted, no textures are excluded.

3.6 -nc

Disables texture compression.

3.7 -b

Specifies the base directory for relative paths to image files to load. If omitted, the program will use the path to the parent directory of the filename specified for the `-i` argument.

3.8 -dsl

Specifies icon creation mode to create icons, sized 48x48 or 24x24 pixels. When this option is specified, one texture file must be specified (using the `-l` option).

When this option is specified, texture information is not output to the texture package file. Only the texture data itself is output.

3.9 –nw4c

Use this option to create an icon file by importing a TGA file created with the Photoshop plugin (`NW4C_tga`). When this option is specified, one texture file must be specified (using the `-l` option).

Note: Format conversion is not included with this option.

Note: Only the `RGB565` format can be used to save as a TGA file.

4 Loading Texture Packages and Getting Textures

To load a texture package file created with the `ctr_TexturePackager.exe` program, the application can load the texture package file into RAM from ROM (or from an SD Card) and call the Texture Packager Library's `GetTexture` function, passing a pointer to the beginning of the texture package as an argument. This function returns pointers to data and the texture information (the image) included in the texture package file.

To get a texture, first call the `GetTextureIndex` function, passing the texture file path specified in the settings file as an argument. The function returns the index of the texture in the package. Then call the `GetTexture` function, passing the index as an argument. This function returns information about the image and a pointer to the image data.

The `ctr_TexturePackager.exe` program creates a table within the texture package file. This table makes it possible to obtain the texture data offset from the filename of the pre-conversion image. The program also has the following two features to speed up texture searches.

- Filenames are sorted alphabetically and a binary search is used.
- Searches are further sped up by using a CRC code generated from the filename as a key.

Note: For details, see the Texture Packager Library runtime specifications**.**

Note: Texture package files for icons created with the `–dsl` option specified do not include texture information. Consequently, you cannot use `TexturepackageLibrary` functions.

5 Restrictions

- Both the width and height of the images to load must be an integer power of 2 between 8 and 1024. (For icons, the dimensions must be a multiple of 8 between 8 and 1024.)
- When using ETC1 compression, both the width and height of the images to load must be integer powers of 2 between 16 and 1024.
- For a cubemap texture, the width, height, mipmap count, and format must be the same for all six images.

Revision History

Version	Revision Date	Category	Description
1.4	2010-10-22	Change	• In section 2.6 Icon Creation, added text in line with feature changes. • In section 3.8 –dsl, added text in line with feature changes. • In Chapter 4 Loading Texture Packages and Getting Textures, added text in line with feature changes. • In Chapter 5 Restrictions, deleted text about warnings that are no longer output.
1.3	2010/09/14	Change	• In section 2.1.3 <imagefile> Node, added an explanation that backslashes "¥" in *src* values are converted to forward slashes "/". • In section 3.9 –nw4c, added a description of the new `–nw4c` feature.
1.2	2010/09/02	Change	• In section 2.1.2.5 etcmethod, added description of new features. • In section 2.5 Difference Conversion, added description of new features. • In section 3.3 -l, added example including expanded description of specifiable options.
1.1	2010/07/27	Change	Corrected the function names in Chapter 4 Loading Texture Packages and Getting Textures.
1.0	2010/05/25	Change	• Official release • In sections 2.6 Icon Creation and 3.8 -dsl, added descriptions in conjunction with new functionality.
0.9.4	2010/05/20	Change	In section Figure 1-1 Texture Packager Library Structure, grouped objects in image.
0.9.3	2010/05/19	Change	Used Word template to rework format.
0.9.2	2010/04/27	Change	In Table 2-1 Specifiable Formats, deleted mention of SHADOW and GAS from output formats because these are created by RenderTexture or another program.
0.9.1	2010/04/23	Change	Updated Table 2-1 Specifiable Formats to reflect the increase in DMPGL internal formats in SDK 0.9.
0.9.0	2009/03/25	-	Rough draft.