ATF Tools
User Guide

Contents

1. Introduction

Adobe Texture Format, or ATF is the recommended file type for fixed texture assets that are used with the Flash runtime's Stage3D API.

This guide is a reference for the following tools, which Adobe provides to create and manage ATF files (Adobe Texture Files):

- png2atf
- pvr2atf
- dds2atf
- ATFViewer
- atfinfo

1.1 ATF basics

ATF files are primarily a file container to store lossy texture data. It achieves its lossy compression through the use of two common techniques: JPEG-XR[1] compression and block based compression. JPEG-XR compression provides a competitive method to save storage space and network bandwidth. Block based compression provides a way to reduce texture memory usage on the client, at a fixed ratio of 1:8 compared to RGBA textures. ATF supports four types of block based compression: DXT1[2]/DXT5[3], ETC1[4], ETC2[5], and PVRTC[6].

In ATF files compression is performed on two levels: first an optional block based compression and on top of that standard loss less or lossy JPEG-XR compression. Other features in ATF files include:

- Embedding of mip maps
- Optionally embeds a complete cube map (sky map)
- Optionally supports selection of internal color space (4:4:4, 4:2:2 and 4:2:0).

1.2 ATF limitations

ATF files have various limitations which are the direct result of existing hardware capabilities on various mobile devices. The specific limitations include:

- Texture sizes are limited to a maximum of 4096x4096 pixels.
- Texture sizes are limited to power of two numbers on each side, that is, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 102, 2048, and 4096.
- At the resolution of 2048x2048 you are required to provide at least one regular mip map level because certain devices only allow texture sizes of up to 1024x1024. The Flash runtime will automatically switch to 1024x1024 if it sees a need for it.
- ATF Version 3 is available from AIR version 18 (for mobile) and Flash Player version 18.

---

1 http://www.itu.int/rec/T-REC-T.832
2 http://msdn.microsoft.com/en-us/library/bb147243%28v=VS.85%29.aspx
3 http://msdn.microsoft.com/en-us/library/bb147243%28v=VS.85%29.aspx
4 http://www.khronos.org/registry/gles/extensions/OES/OES_compressed_ETC1_RGB8_texture.txt
5 https://www.opengl.org/registry/specs/ARB/ES3_compatibility.txt
6 http://www.imgtec.com/powervr/insider/powervr-pvrtexlib.asp

2. png2atf

png2atf is a command line utility which converts PNG files to ATF files. The resulting ATF files can then be directly used with the uploadCompressedTextureFromByteArray() ActionScript 3 API. ***png2atf*** takes any valid PNG file and by default converts it to either a RGB or RGBA ATF file, depending on if the PNG file has transparency. It can optionally also create a block based compression texture.

2.1 Invocation

To convert a PNG file to a RGB or RGBA ATF file run the command as such:

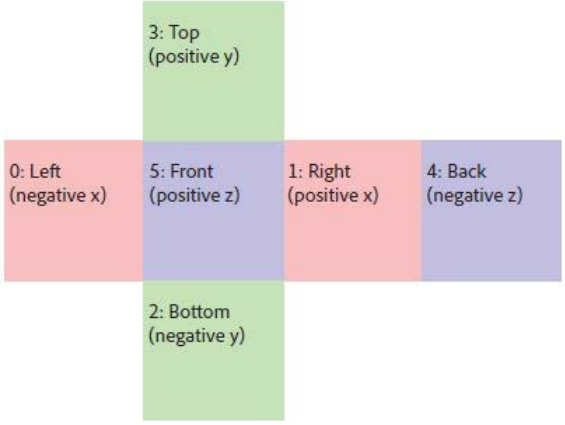
```
> png2atf -i test.png -o test.atf
.
[In 1365KB][Out 61KB][Ratio 4.49851%][LZMA:0KB JPEG-XR:61KB]
```

To create a block based compression texture file run the command as such:

```
> png2atf -c -i test.png -o test.atf
.....................................................
[In 682KB][Out 682KB][Ratio 99.9954%][LZMA:937KB JPEG-XR:682KB]
```

2.2 Command line options

-i <file>	Specifies the input file name
-o <file>	Specifies the output file name
-c	Creates a block based compression texture. Three types of block based compression will be used and embedded into the same ATF file: DXT1, ETC1, ETC2 and PVRTC 4bpp. To create ATF with multiple block based texture you can specify e ,e2,d or p arguments in any combinations to create ATF with ETC1,ETC2 ,DXT or/and PVRTC. Specifying only –c argument will create ATF with all four block based compressions. When you load this type of texture into the Flash Player, the Flash Player will pick the appropriate format for the device. If the PNG file has transparency a RGBA texture will be generated instead. You can use the ***atfinfo*** tool later to find out what format was chosen.
-m	Creates a cube map texture. Input files need to be named consecutively in this form: XXXn.png where XXX is the base file name and n=0-5. Example: ‘cube0.png’, ‘cube1. png’, ‘cube2.png’, ‘cube3.png’, ‘cube4.png’, ‘cube5.png’. The input file name in this case should be ‘cube0.png’.



-s Silences any output the tool might have during compression. This can be useful for batch processing.

-4 Instructs the JPEG-XR encoder to use a 4:4:4 color space internally. This is the default for block based compression. In some cases it is desirable to use this color space for RGB/RGBA textures in case you see color bleeding artifacts around red or blue shapes or for normal maps.

-2 Instructs the JPEG-XR encoder to use a 4:2:2 color space internally. It is not recommended to use this color space for block based compression as this can cause severe artifacts.

-0 Instructs the JPEG-XR encoder to use a 4:2:0 color space internally. This is the default for RGB/RGBA textures. It is not recommended to use this color space for block based compression as this can cause severe artifacts; though it might be worth experimenting with this option if for instance the image data is monochromatic.

-q <0-180> Selects the quantization level, that is, how much the image data should be compressed. A value of 0 means loss less compression which is the default for block based compression textures. It is not recommended to use lossy compression for block based compression; though it is worth to experiment with this value from a case to case basis.

The default value for RGB and RGBA textures is 30. Higher values will create more artifacts, smaller values will reduce artifacts. Note that you should not confuse this with the standard JPEG compression factor. It's similar in concept but applies in a different non-linear range.

-f <0-15> Selects how many flex bits should be trimmed during JPEG-XR compression. This option in not related to the quantization level but selects how much noise should be retained across the image. Like the quantization level higher values create more artifact. The default value is always 0.

-n <start,end> Embeds a specific range of textures for texture streaming. Range starts from 0 for the main texture and 1 and above for mip levels. The size information of the main level is retained when the .atf file is created even when a sub set of ranges are selected.

Example for texture streaming - creating 3 levels for a 512x512 texture:

```
> png2atf -n 1,1 -i i.png -o ohigh.atf
> png2atf -n 2,2 -i i.png -o omed.atf
> png2atf -n 3,9 -i i.png -o olow.atf
```

To stream these 3 levels, create a 512x512 texture in ActionScript and then upload olow.atf, omed.atf and ohigh.atf in sequence.

-x Reads mip map images from input files instead of auto-creating them. Input files need to be named <filename><ll>.png, where ll=00-12. For cubemaps, the format is <filename><ll><n>.png.

-e Embeds empty (black) mip maps.

-r Compresses block compressed textures using JPEG-XR+LZMA to reduce file size.

3. pvr2atf

pvr2atf is a command line utility which converts PVR texture files to ATF files which can then be used with the uploadCompressedTextureFromByteArray() ActionScript 3 API. The tool works similarly to ***png2atf*** except that you have to provide input files in the PVR texture file format. For block based compression you can either provide a single compressed file or a set of 3 to embed all three supported formats to target multiple GPU architectures.

3.1 Invocation

To convert a PVR file to a RGB or RGBA ATF file run the command as such:

```
> pvr2atf -r test.pvr -o test.atf

[In 4096KB][Out 410KB][Ratio 10.0241%][LZMA:0KB JPEG-XR:410KB]
```

To create a block based compression texture file run the command as such, where each input PVR file represents a block compressed file format:

```
> pvr2atf -d test_dxt1.pvr -e test_etc1.pvr -p text_pvrtc.pvr -o test.atf
.....................................................
[In 2048KB][Out 1704KB][Ratio 83.2007%][LZMA:937KB JPEG-XR:766KB]
```

A single block compressed file can also be provided, in which case the other formats will be left empty (and potentially creating runtime exceptions when uploadCompressedTextureFromByteArray() is used):

```
> pvr2atf -d test_dxt1.pvr -o test.atf
.....................................................
[In 2048KB][Out 1704KB][Ratio 83.2007%][LZMA:937KB JPEG-XR:766KB]
```

3.2 Accepted PVR texture types

pvr2atf accepts the following types of PVR files:

- RGB 888
- RGBA 8888
- DirectX 9 DXT 1 (D3D_DXT1, D3D_BC1)
- DirectX 9 DXT5 (D3D_DXT5, D3D_BC3)
  Note: This format is only supported in Flash Player 10.4 (SWF17) and newer.
- ETC2 RGB 888
  Note: This format is only supported in AIR for mobile (SWF29) and newer.
- ETC2 RGBA 888
  Note: This format is only supported in AIR for mobile (SWF29) and newer.
- ETC
- PVRTC 4BPP (opaque) (OGL_PVRTC4)
- PVRTC 4BPP (transparent) (OGL_PVRTC4)
  Note: This format is only supported in Flash Player 10.4 (SWF17) and newer.

SRGB textures are not supported in runtime, so do not create pvr file in SRGB format.

pvr2atf does not accept flipped textures. Make sure that you create PVR texture with the flipping option turned off. This can be achieved with the '-yflip0' option of the PVRTexTool command line tool or by unchecking the 'Flipped' checkbox in the PVRTexTool GUI tool.

3.3:Command line options

-r <file> Specifies the input RGB or RGBA PVR file. Note that you can't mix this with the -d/-e/- p options.

-d <file> Specifies the input DXT1 or DXT5 PVR file.

-e <file> Specifies the input ETC1 PVR file.

-e2<file> Specifies the input ETC2 PVR file (*supported only in SWF29 and newer).*

-p <file> Specifies the input PVRTC PVR file. ***pvr2atf*** will automatically detect if the file contains transparency and pick the right target ATF format.

-o <file> Specifies the output file name

-4 Instructs the JPEG-XR encoder to use a 4:4:4 color space internally. This is the default for block based compression. In some cases it is desirable to use this color space for RGB/RGBA textures in case you see color bleeding artifacts around red or blue shapes or for normal maps.

-2 Instructs the JPEG-XR encoder to use a 4:2:2 color space internally. It is not recommended to use this color space for block based compression as this can cause severe artifacts.

-0 Instructs the JPEG-XR encoder to use a 4:2:0 color space internally. This is the default for RGB/RGBA textures. It is not recommended to use this color space for block based compression as this can cause severe artifacts; though it might be worth experimenting with this option if for instance the image data is monochromatic.

-q <0-180> Selects the quantization level, i.e. how much the image data should be compressed. A value of 0 means loss less compression which is the default for block based compression textures. It is not recommended to use lossy compression for block based compression; though it is worth to experiment with this value from a case to case basis.

The default value for RGB and RGBA textures is 30. Higher values will create more artifacts, smaller values will reduce artifacts. Note that you should not confuse this with the standard JPEG compression factor. It's similar in concept but applies in a different non-linear range.

-f <0-15> Selects how many flex bits should be trimmed during JPEG-XR compression. This option is not related to the quantization level but selects how much noise should be retained across the image. Like the quantization level, higher values create more artifact. The default value is always 0.

-n <start,end> Embeds a specific range of textures for texture streaming. Range starts from 0 for the main texture and 1 and above for mip levels. The size information of the main level is retained when the .atf file is created even when a sub set of ranges are selected.
Example for texture streaming - creating 3 levels for a 512x512 texture: pvr2atf -n1,1 -r i.pvr -o ohigh.atf
pvr2atf -n2,2 -r i.pvr -o omed.atf
pvr2atf -n3,9 -r i.pvr -o olow.atf
To stream these 3 levels, create a 512x512 texture in ActionScript and then upload olow.atf, omed.atf and ohigh.atf in sequence. Texture streaming is only supported in Flash Player 10.4 or newer.

4. dds2atf

dds2atf is a command line utility which converts DDS texture files to ATF files which can then be used with the uploadCompressedTextureFromByteArray() ActionScript 3 API. The tool works similarly to ***png2atf*** except that you have to provide input files in the DDS texture file format.

4.1 Invocation

To convert a DDS file to an ATF file run the command as such:

```
> dds2atf -i test.dds -o test.atf
```

4.2 Accepted DDS texture types

dds2atf accepts the following types of DDS files:

- DirectX 9 RGB 888 (D3DFMT_R8G8B8)
- DirectX 9 BGRA 8888 (D3DFMT_A8R8G8B8)
- DirectX 9 DXT 1 (D3DFMT_DXT1)
- DirectX 9 DXT5 (D3DFMT_DXT5)
  Note: This format is only supported in Flash Player 10.4 (SWF17) and newer.

4.3 Command line options

-i <file>	Specifies the input DDS file.
-o <file>	Specified the output file name
-4	Instructs the JPEG-XR encoder to use a 4:4:4 color space internally. This is the default for block based compression. In some cases it is desirable to use this color space for RGB/RGBA textures in case you see color bleeding artifacts around red or blue shapes or for normal maps.
-2	Instructs the JPEG-XR encoder to use a 4:2:2 color space internally. It is not recommended to use this color space for block based compression as this can cause severe artifacts.
-0	Instructs the JPEG-XR encoder to use a 4:2:0 color space internally. This is the default for RGB/RGBA textures. It is not recommended to use this color space for block based compression as this can cause severe artifacts; though it might be worth experimenting with this option if for instance the image data is monochromatic.
-q <0-180>	Selects the quantization level, i.e. how much the image data should be compressed. A value of 0 means loss less compression which is the default for block based compression textures. It is not recommended to use lossy compression for block based compression; though it is worth to experiment with this value from a case to case basis. The default value for RGB and RGBA textures is 30. Higher values will create more artifacts, smaller values will reduce artifacts. Note that you should not confuse this with the standard JPEG compression factor. It is similar in concept but applies in a different non-linear range.

-f <0-15> Selects how many flex bits should be trimmed during JPEG-XR compression. This option is not related to the quantization level but selects how much noise should be retained across the image. Like the quantization level higher values create more artifact. The default value is always 0.

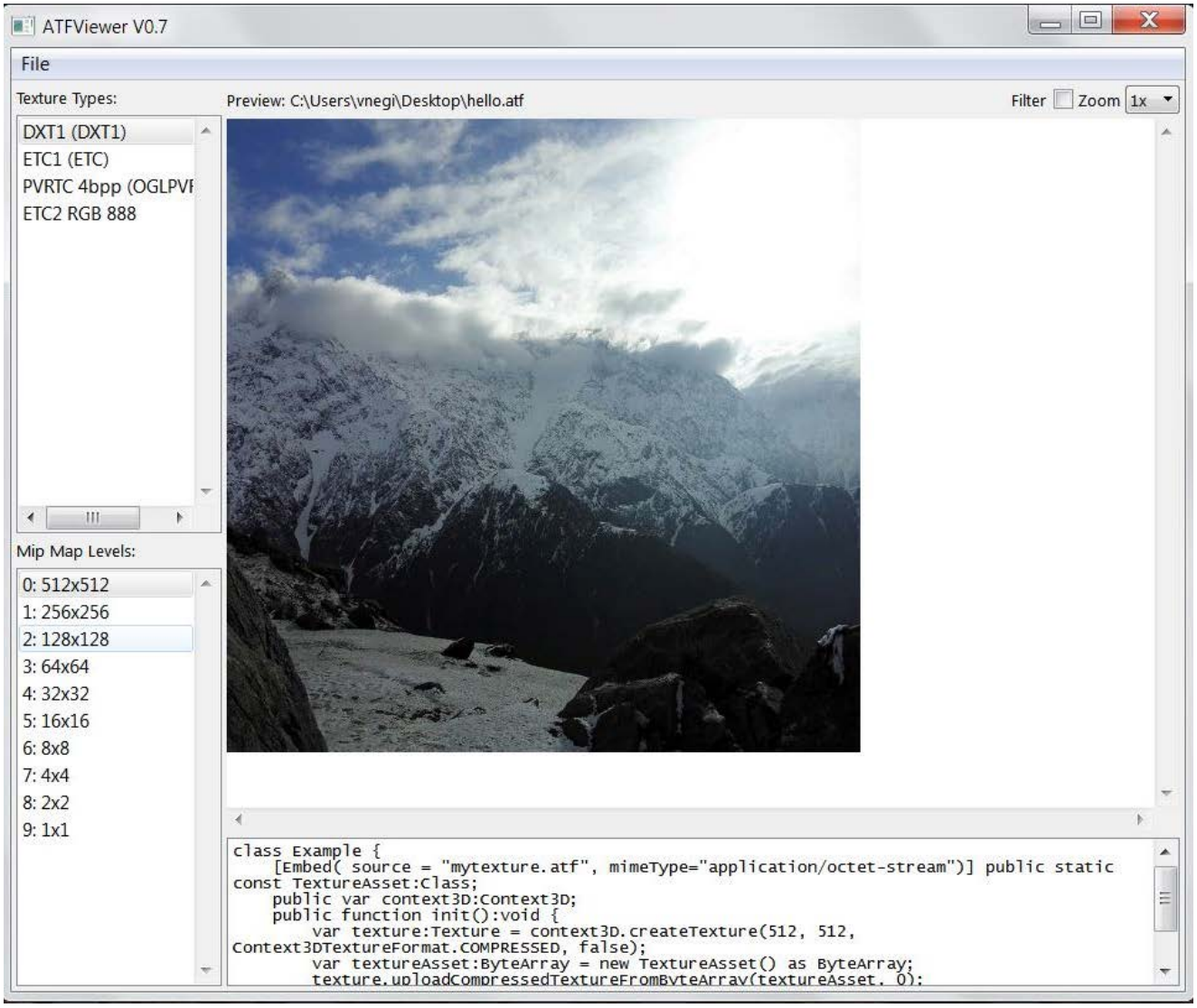
-n <start,end> Embeds a specific range of textures for texture streaming. Range starts from 0 for the main texture and 1 and above for mip levels. The size information of the main level is retained when the .atf file is created even when a sub set of ranges are selected. Example for texture streaming - creating 3 levels for a 512x512 texture:
dds2atf –n 1,1 –i i.dds –o ohigh.atf
dds2atf –n 2,2 –i i.dds –o omed.atf
dds2atf –n 3,9 –i i.dds –o olow.atf

To stream these 3 levels, create a 512x512 texture in ActionScript and then upload olow.atf, omed.atf, and ohigh.atf in sequence. Texture streaming is only supported in Flash Player 10.4 or newer.

5. ATFViewer

ATFViewer is a GUI tool which previews and inspects ATF files. The primary purpose is to audit DXT1, ETC1, ETC2, and PVRTC compression artifacts. You can open and view ATF files by either using the 'Open' menu item or by dragging a file from Explorer into the window.

The Snippet preview area (in the figure below) shows you an example of how to load a particular ATF file in ActionsScript 3 code.



6. atfinfo

atfinfo is command line utility which displays internal information about ATF files. It prints size, mip map count, texture type, texture format, the actual number of mipmaps, whether there are any empty mip- maps, and whether it is a cube map. It also shows which ActionScript 3 classes and format correspond to a particular format.

6.1 Invocation

The following is an example invocation of atfinfo and its output:

```
> atfinfo.exe -i hello.atf
File Name: hello.atf
ATF Version: 3
ATF File Type: RAW Compressed (DXT1+ETC1+PVRTV4bpp+ETC2 RGB)
Size: 512x512
Cube Map: no
Empty Mip maps: no
Embedded Levels: XXXXXXXXXX (512x512, 256x256, 128x128, 64x64, 32x32,
16x16, 8x8, 4x4, 2x2, 1x1)
AS3 Texture Class: Texture (flash.display3D.Texture)
AS3 Texture Format: Context3DTextureFormat.COMPRESSED
(flash.display3D.Context3DTextureFormat)
```