

Mali™ GPU Binary Asset Exporter

Version: 2.2

User Guide



Mali GPU Binary Asset Exporter

User Guide

Copyright © 2009 ARM. All rights reserved.

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
14 October 2009	A	Non-Confidential	First release for v2.2

Proprietary Notice

Words and logos marked with or are registered trademarks or trademarks of ARM in the EU and other countries, except as otherwise stated below in this proprietary notice. Other brands and names mentioned herein may be the trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

Where the term ARM is used it means “ARM or any of its subsidiaries as appropriate”.

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Unrestricted Access is an ARM internal classification.

Product Status

The information in this document is final, that is for a developed product.

Web Address

<http://www.arm.com>

Contents

Mali GPU Binary Asset Exporter User Guide

	Preface	
	About this book	x
	Feedback	xiii
Chapter 1	Introduction	
	1.1 About the Mali GPU Binary Asset Exporter	1-2
Chapter 2	Installing the Mali Binary Asset Exporter	
	2.1 Mali Binary Asset Exporter content and installation requirements	2-2
	2.2 Installation procedure for the Mali Binary Asset Exporter	2-3
Chapter 3	Using the Mali Binary Asset Exporter	
	3.1 About the Mali Binary Asset Exporter	3-2
	3.2 Using the Mali Binary Asset Exporter	3-3
	Glossary	

List of Tables

Mali GPU Binary Asset Exporter User Guide

Change history ii

List of Figures

Mali GPU Binary Asset Exporter User Guide

Figure 2-1	Directory structure	2-3
Figure 3-1	Mali Binary Asset Exporter user interface	3-4
Figure 3-2	Viewing nodes	3-6
Figure 3-3	Applying quality settings	3-7
Figure 3-4	Adding a new shader	3-9
Figure 3-5	Viewing COLLADA file materials	3-11

Preface

This preface introduces the *Mali GPU Binary Asset Exporter User Guide*. It contains the following sections:

- *About this book* on page x
- *Feedback* on page xiii.

About this book

This is the *Mali GPU Binary Asset Exporter User Guide*. It provides guidelines for using the Mali Binary Asset Exporter to assist in the development of 3D graphics applications that are targeted to run on a Mali *Graphics Processing Unit* (GPU). This book is part of a suite belonging to the Mali Developer Tools.

Intended audience

This guide is written for software developers who are writing OpenGL ES 2.0 applications that are targeted to run on a Mali GPU.

Using this book

This book is organized into the following chapters:

Chapter 1 *Introduction*

Read this for an introduction to the Mali Binary Asset Exporter.

Chapter 2 *Installing the Mali Binary Asset Exporter*

Read this for information about how to install the Mali Binary Asset Exporter.

Chapter 3 *Using the Mali Binary Asset Exporter*

Read this for information about how to start and use the Mali Binary Asset Exporter.

Glossary Read this for definitions of terms used in this book.

Typographical Conventions

The typographical conventions are:

<i>italic</i>	Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.
monospace	Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

<code>monospace</code>	Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
<code>monospace italic</code>	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
<code>monospace bold</code>	Denotes language keywords when used outside example code.
<code>< and ></code>	Enclose replaceable terms for assembler syntax where they appear in code or code fragments. For example: MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter, <http://infocenter.arm.com>, for access to ARM documentation.

ARM publications

This guide contains information that is specific to the Mali Developer Tools. See the following documents for other relevant information:

- *Mali GPU Developer Tools Technical Overview* (ARM DUI 501)
- *Mali GPU Performance Analysis Tool User Guide* (ARM DUI 0502)
- *Mali GPU Texture Compression Tool User Guide* (ARM DUI 0503)
- *Mali GPU Shader Developer Studio User Guide* (ARM DUI 0504)
- *Mali GPU Demo Engine User Guide* (ARM DUI 0505)
- *OpenGL ES 1.1 Emulator User Guide* (ARM DUI 0506)
- *Mali GPU Shader Library User Guide* (ARM DUI 0510)
- *OpenGL ES 2.0 Emulator User Guide* (ARM DUI 0511)
- *Mali GPU Offline Shader Compiler User Guide* (ARM DUI 0513).

Other publications

This section lists relevant documents published by third parties:

- *OpenGL ES 1.1 Specification* at <http://www.khronos.org>.
- *OpenGL ES 2.0 Specification* at <http://www.khronos.org>.
- *OpenGL ES Shading Language Specification* at <http://www.khronos.org>.
- *OpenVG 1.1 Specification* at <http://www.khronos.org>.

- *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 2* (5th Edition, 2005), Addison-Wesley Professional. ISBN 0-321-33573-2.
- *OpenGL Shading Language* (2nd Edition, 2006), Addison-Wesley Professional. ISBN 0-321-33489-2.

Feedback

ARM welcomes feedback on this product and its documentation.

Feedback on this product

If you have any comments or suggestions about this product then contact malidevelopers@arm.com and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to errata@arm.com. Give:

- the title
- the number, ARM DUI 0507A
- the page numbers to which your comments apply
- a concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Chapter 1

Introduction

This chapter provides information about the Mali GPU Binary Asset Exporter. It contains the following section:

- *About the Mali GPU Binary Asset Exporter* on page 1-2

1.1 About the Mali GPU Binary Asset Exporter

The Mali GPU Binary Asset Exporter is a program that runs on your desktop computer and converts graphics from the COLLADA format to the Mali Binary Asset format. These graphics assets include such elements as:

- geometry data
- textures
- lighting
- movements for animation.

The Mali Binary Asset Exporter has a GUI that enables you to modify and verify the data before you generate the Mali Binary Asset output files. After the asset files in COLLADA format have been converted to the more compact Mali Binary Asset format, they can be used with the Mali GPU Demo Engine. Mali Binary Asset files can be read in a graphics application by calling the appropriate functions in the Mali GPU Demo Engine.

The assets can be exported in COLLADA format from commercial 3D modelling tools such as Google SketchUpPro and Autodesk 3ds Max.

Chapter 2

Installing the Mali Binary Asset Exporter

This chapter describes how to install the Mali Binary Asset Exporter. It contains the following sections:

- *Mali Binary Asset Exporter content and installation requirements* on page 2-2
- *Installation procedure for the Mali Binary Asset Exporter* on page 2-3.

2.1 Mali Binary Asset Exporter content and installation requirements

This section describes the installation requirements for the Mali Binary Asset Exporter and the contents of the Mali Binary Asset Exporter package.

- *Mali Binary Asset Exporter content*
- *Installation requirements for the Mali Binary Asset Exporter.*

———— **Note** —————

The Mali Binary Asset Exporter has been tested successfully on a 32-bit computer.

2.1.1 Mali Binary Asset Exporter content

The Mali Binary Asset Exporter package contains a binary executable for Microsoft Windows and some example COLLADA files. For more information see the *Mali Binary Asset Exporter Release Note*.

2.1.2 Installation requirements for the Mali Binary Asset Exporter

To install the Mali Binary Asset Exporter on Microsoft Windows, you require:

- Microsoft Windows XP Professional version 2002, service pack 2
- Microsoft .NET framework installed
- a minimum of 200KB disk space for the application
- a minimum of 400KB disk space for the example COLLADA files.

2.2 Installation procedure for the Mali Binary Asset Exporter

The procedure to install the Mali Binary Asset Exporter on Microsoft Windows is:

1. Go to the Mali Developer Center website at:
<http://www.malideveloper.com>
2. Download the Mali Binary Asset Exporter package.
3. Run the file:
`Mali_GPU_Mali_Binary_Asset_Exporter_WinXP_vm.n.exe`
 where:
m identifies the major version
n identifies the minor version.
4. Select the required installation options and then click **Finish** to complete the installation.

By default, the Mali GPU Mali Binary Asset Exporter is installed in:

`C:\Program Files\ARM\Mali Developer Tools\Mali GPU Binary Asset Exporter vm.n\bin`

Some example COLLADA files are installed in:

`C:\Program Files\ARM\Mali Developer Tools\Mali GPU Binary Asset Exporter vm.n\collada`

Figure 2-1 shows the directory structure that is created at the location specified when you installed the Mali GPU Binary Asset Exporter:

```

Mali GPU Binary Asset Exporter vm.n/
├── bin/
│   └── mbaexporter.exe
└── collada/
    ├── lightshow.DAE
    ├── superpot.DAE
    └── teapot.DAE
  
```

Figure 2-1 Directory structure

Chapter 3

Using the Mali Binary Asset Exporter

This chapter describes how to use and customize the Mali Binary Asset Exporter. It contains the following sections:

- *About the Mali Binary Asset Exporter* on page 3-2
- *Using the Mali Binary Asset Exporter* on page 3-3.

3.1 About the Mali Binary Asset Exporter

The Mali Binary Asset Exporter is a converter tool for Windows that converts XML-based COLLADA documents to the Mali Binary Asset format. If you do not have your own optimized deployment format, you can use the Mali Binary Asset Exporter to convert graphics assets that you have created yourself to Mali Binary Asset format for use on a Mali GPU. As an OpenGL ES applications developer, you can use the Mali Binary Asset files with the Mali Demo Engine to develop applications that can be run and tested on the OpenGL ES 2.0 Emulator. These can then easily be ported to the Mali hardware.

The Mali Binary Asset format is faster to parse, uses less memory, and is smaller in size than COLLADA XML.

———— **Note** —————

The Mali Binary Asset Exporter accepts COLLADA version 1.4.1 documents, and exports relevant information as Mali Binary Asset format structures.

After importing a COLLADA file into the Mali Binary Asset Exporter, you can alter properties such as textures, light and shader object assigned to various nodes in the scene graph, and then export the result as an Mali Binary Asset file, for use with the Mali Demo Engine Library.

The Mali Binary Asset file can then be loaded in the Mali Demo Engine and used on target devices.

3.1.1 COLLADA format

COLLABorate Design Activity (COLLADA) is an industry-standard format for storing 3D assets based on XML. The main benefit of COLLADA is that it provides an intermediate format that enables 3D assets to be moved between various applications.

3.1.2 Mali Binary Asset format

COLLADA is not intended as a final format for loading onto devices, because of this the Mali Binary Asset file format is included with the Mali GPU package available from ARM. The Mali Binary Asset format is a compact binary representation of a 3D scene. It contains all the assets, such as data structures, necessary to draw a scene using the Mali Demo Engine Library, with the option to embed or to reference external assets. The Mali Binary Asset format is intended to be the format loaded on the final devices. It is not intended as an intermediate file format, it is better to use a common industry format such as COLLADA for this function.

3.2 Using the Mali Binary Asset Exporter

This section describes how to use the Mali Binary Asset Exporter. In addition to importing and exporting files, you can use the Mali Binary Asset Exporter to perform other subsidiary tasks.

The general steps for using the Mali Binary Asset Exporter are:

1. Set import options.
2. Import graphics assets.
3. Optionally, adjust asset geometry quality.
4. Optionally, add image data, as bitmaps, for export.
5. Optionally, add textures for export.
6. Optionally, add shaders for export.
7. Set export options.
8. Export Mali Binary Asset to required location.

3.2.1 Starting the Mali Binary Asset Exporter

To start the Mali Binary Asset Exporter, select:

Start → All Programs → ARM → Mali Developer Tools → Mali GPU Mali Binary Asset Exporter *vm.n*

where:

- m* identifies the major version
- n* identifies the minor version.

3.2.2 The Mali Binary Asset Exporter user interface

This section describes the basic operation of the Mali Binary Asset Exporter.

Figure 3-1 on page 3-4 shows the Mali Binary Asset Exporter user interface.

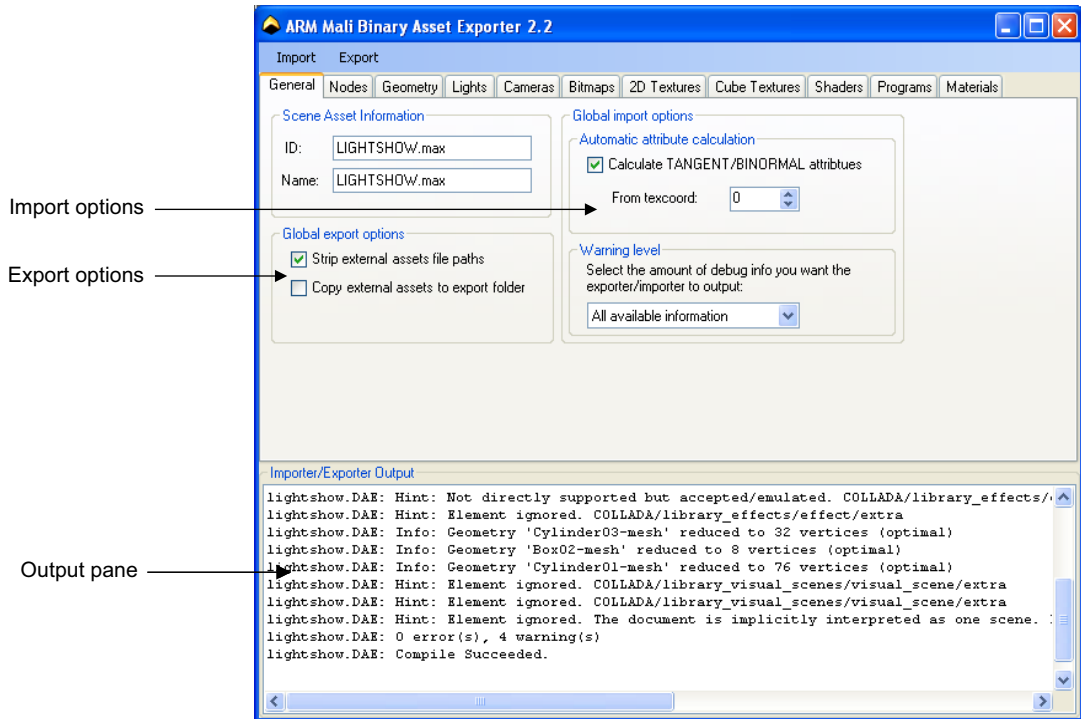


Figure 3-1 Mali Binary Asset Exporter user interface

The **General** tab of the Mali Binary Asset Exporter contains general options that relate to the Mali Binary Asset file. Each of the other tabs corresponds to an asset type. The **General** tab contains the following information:

Import and Export options

Use these options to control how COLLADA files are imported and how Mali Binary Asset files are exported. See *Setting import and export options* on page 3-5.

Output pane

The **Output** pane is available from all tabs and shows a log of import and export activity. During import, the Mali Binary Asset Exporter attempts to convert as many COLLADA structures as possible to Mali Binary Asset format. However, because the Mali Binary Asset format represents only a subset of the COLLADA feature set, some constructs are ignored. The log contains information about these ignored structures.

3.2.3 Setting import and export options

Use the following import options as required:

Strip external assets file paths

Use this option to instruct the Mali Binary Asset Exporter to strip away the file paths of all external assets that are referenced by the resulting Mali Binary Asset file. This means that the Mali Binary Asset Loader in the Mali Demo Engine looks for these asset files in the same directory as the Mali Binary Asset file.

Copy external assets to export folder

Use this option to instruct the application to copy all externally referenced asset files to the directory the Mali Binary Asset file is placed in.

Use the following export options as required:

Calculate TANGENT/BINORMAL

Use this option to ensure the application calculates the tangents and binormals for geometries that are imported from COLLADA files.

From texcoord

Use this control to specify the texture coordinates to use when calculating the tangents and binormal. This feature uses the texture coordinates and normal vector for each vertex, to generate tangent vectors and binormal vectors for the vertex.

3.2.4 Importing a COLLADA file

Before importing or exporting a file, select the required import options as *Setting import and export options* describes.

To import a COLLADA file, select **COLLADA** from the **Import** menu.

To export an Mali Binary Asset file, select **MBA** from the **Export** menu.

3.2.5 Viewing nodes

After importing a COLLADA file, click on the Nodes tab to view information about nodes in the scene. Figure 3-2 on page 3-6 shows an example **Nodes** tab.

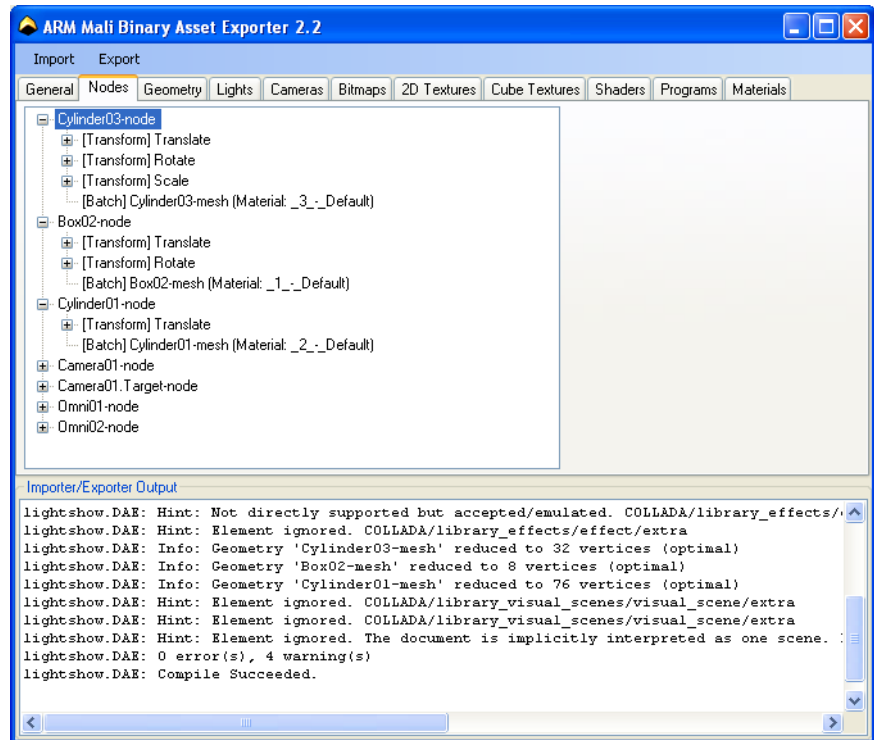


Figure 3-2 Viewing nodes

3.2.6 Controlling quality

Figure 3-3 on page 3-7 shows the **Geometry** tab. Use this tab to view the geometry meshes that are available in the loaded scene, the number of vertices and indices the geometries consist of, and the attributes available for each vertex.

Use the **Attribute Settings** group as follows to change the precision and set a remap method for attributes.

1. Select a geometry from the tree structure in the left pane.
2. Select the attribute to change from the **Attributes** list.
3. Select the quality and remap method to apply.

Use the **Quality** setting to select the precision to use for the attribute stream. The default is 32-bit floating point precision. The other quality settings can decrease the size of the generated Mali Binary Asset file and reduce loading time, but are also likely to reduce precision.

Use the **Remap** setting as follows:

- **Normalize** adjusts the input data to the range [0,1] inclusive.
 - **Scale and bias** normalizes to the range of the data type provided in the **Quality** setting, and also saves a scale value and a bias value to return the value back to its original value.
4. Click **Apply to selected** to apply the selected settings to the selected geometry. Alternatively, click **Apply to all** to apply the selected settings to all loaded geometries.

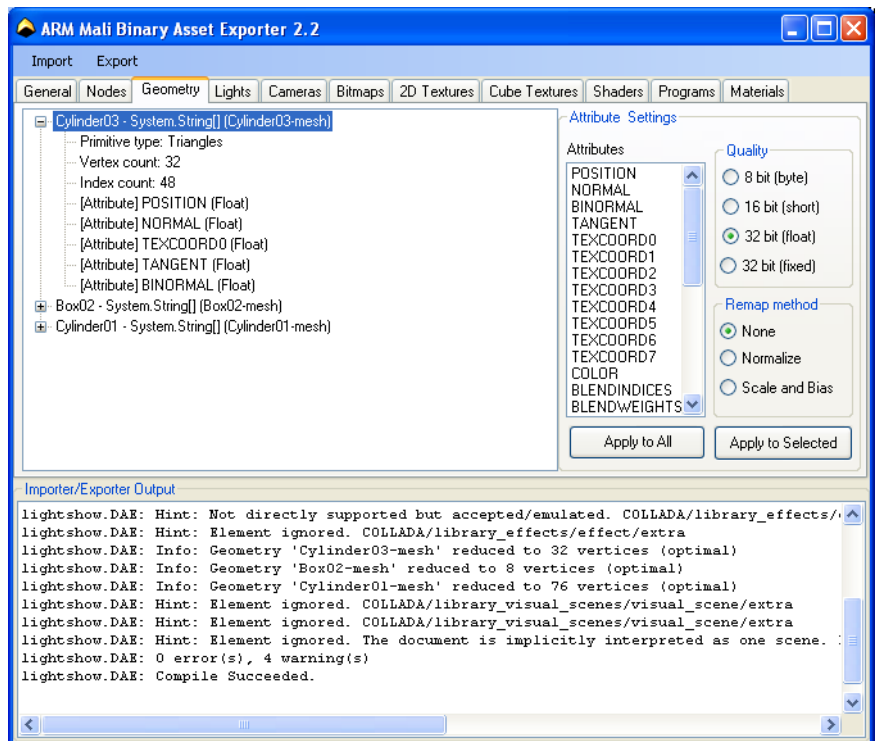


Figure 3-3 Applying quality settings

3.2.7 Adding 2D textures or cube textures

You can add 2D textures or cube textures using pre-loaded bitmaps. Mali Binary Asset Exporter supports the following bitmap formats:

- JPEG
- PNG

To load a bitmap into the Mali Binary Asset Exporter, follow these steps:

1. Click on the **Bitmaps** tab.
2. Click **Add bitmap**.
3. Navigate to the required file and click **Open**. The file appears in the left pane.

After adding a bitmap, you can use it as a 2D texture as follows:

1. Click on the **2D Textures** tab.
2. Click **Add Texture2D**. The **New Texture2D** window appears.
3. Type a name for the new texture in the **Name** field.
4. From the Target pane, select the bitmap to use in the texture.
5. Click **OK**.

To add a bitmap to a cube texture, follow the same procedure from the **Cube Textures** tab.

3.2.8 Embedding and referencing shaders

The exporter also supports shaders, both embedded and referenced. To embed or reference a new shader, follow these steps:

1. Select your required options from the General tab, as *The Mali Binary Asset Exporter user interface* on page 3-3 and *Setting import and export options* on page 3-5 describes.
2. Click on the **Shaders** pane.
3. Click **Add shader**. The **New shader** window is displayed.

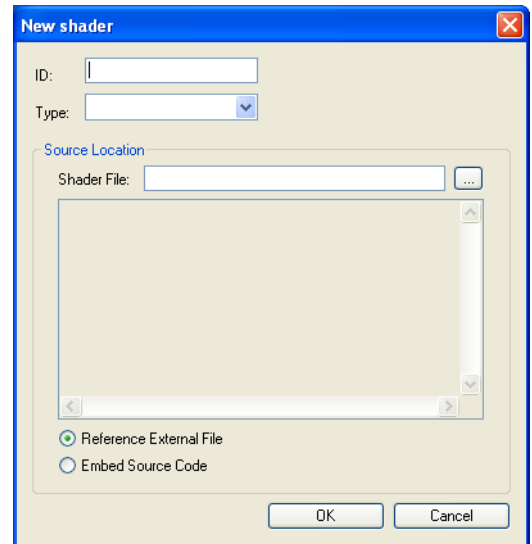


Figure 3-4 Adding a new shader

4. Specify a name for the new shader in the **ID** field.
5. Select either **Vertex Shader** or **Fragment Shader** from the **Type** drop-down menu.
6. Click the **Shader file** selector, browse to the shader source code location, and click **Open**.
7. Select either **Reference External File** or **Embed Source Code**.
8. Click **OK**.

If a shader is referenced, the exporter saves the path to the shader in the Mali Binary Asset file. If you select the option **Strip external assets file paths** on the **General** tab, the path to the file is not written to the Mali Binary Asset file. Only the file name is written. This means that if you load the Mali Binary Asset file at a later stage, using the Mali Demo Engine, it looks for the shader source code file with the given name in the same folder as the Mali Binary Asset file.

———— **Note** —————

Shaders must have unique names, even if they are of different types and belong to the same program.

3.2.9 Viewing COLLADA file materials

Geometric objects can have many parameters that describe their material properties. These material properties are the parameters for the rendering computations that produce the visual appearance of the object in the final output.

The specific set of material parameters depend on the graphics rendering system employed. When developing graphics applications for programmable graphics pipelines, you define the set of material parameters. These parameters satisfy the rendering algorithm defined in the vertex and fragment shaders.

The **Materials** tab displays the materials that have been loaded from the COLLADA file. You can also link programs from the **Programs** tab to the materials, as follows:

1. Ensure you have added at least one program to the **Programs** tab.
2. Click on the **Materials** tab.
3. From the left pane, select the material that you want to link to a program.
4. Click the **Link to program** button. The **Link Material to Program** window is displayed.
5. Select the required program from the list and click **OK**.

To unlink programs and or remove them from the materials that have been loaded from the COLLADA file, do as follows:

1. Click the **Unlink program** button.
2. Click the **Remove Selected** button, to remove the program.

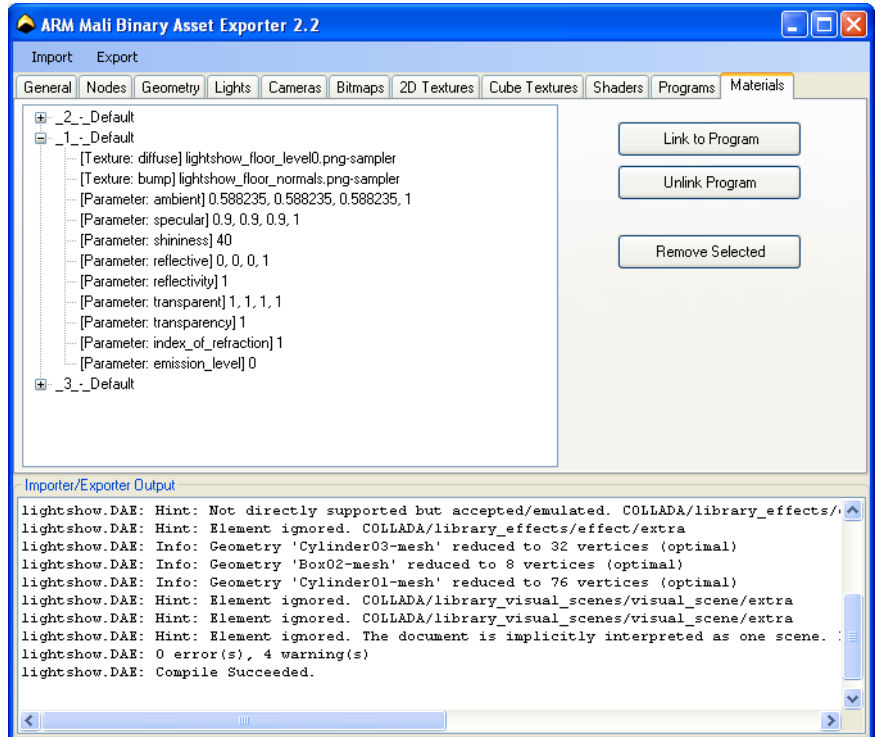


Figure 3-5 Viewing COLLADA file materials

Glossary

This glossary describes some of the terms used in ARM manuals. Where terms can have several meanings, the meaning presented here is intended.

COLLABorative Design Activity (COLLADA) files

Open-standard XML files that describe digital assets. COLLADA files are compatible with many graphics applications.

COLLADA

See COLLABorative Design Activity (COLLADA) files.

Fragment shader

A program running on the pixel processor that calculates the color and other characteristics of each fragment.

GPU

See Graphics Processor Unit.

Graphics application

A custom program that runs on the platform CPU and uses the Mali GPU to display graphics.

Graphics Processor Unit (GPU)

A hardware accelerator for graphics systems using OpenGL ES and OpenVG. The hardware accelerator comprises of an optional geometry processor and a pixel processor together with memory management. Mali programmable GPUs, such as the

Mali-200 and Mali-400 MP GPUs, consist of a geometry processor and at least one pixel processor. Mali fixed-function GPUs, such as the Mali-55 GPU consist of a pixel processor only.

Mali A name given to graphics software and hardware products from ARM that aid 2D and 3D acceleration.

Mali Binary Asset The Mali Binary Asset file format provides an optimized binary representation of a 3D scene for loading into the Mali Demo Engine Library.

Mali Binary Asset Exporter A converter tool for Windows that converts XML-based COLLADA documents to the *Mali Binary Asset* format for use with the Mali Demo Engine. The Mali Binary Asset Exporter is a component of the Mali Developer Tools.

Mali Demo Engine The Mali Demo Engine is a component of the Mali Developer Tools. The Mali Demo Engine library enables you to develop 3D graphics applications more easily than using OpenGL ES 2.0 alone.

Mali Demo Engine Library A C++ class framework for developing OpenGL ES 2.0 applications for the Mali GPU. The Mali Demo Engine Library is a component of the Mali Developer Tools.

Mali Developer Tools A set of development programs that enables software developers to create graphics applications.

Pixel A pixel is a discrete element that forms part of an image on a display. The word pixel is derived from the term Picture Element.

Shader A program, usually an application program, running on the GPU, that calculates some aspect of the graphical output. See fragment shader and vertex shader.

Shader Library A set of shader examples, tutorials, and other information, designed to assist with developing shader programs for the Mali GPU. The Shader Library is a component of the Mali Developer Tools.

Vertex shader A program running on the geometry processor, that calculates the position and other characteristics, such as color and texture coordinates, for each vertex.