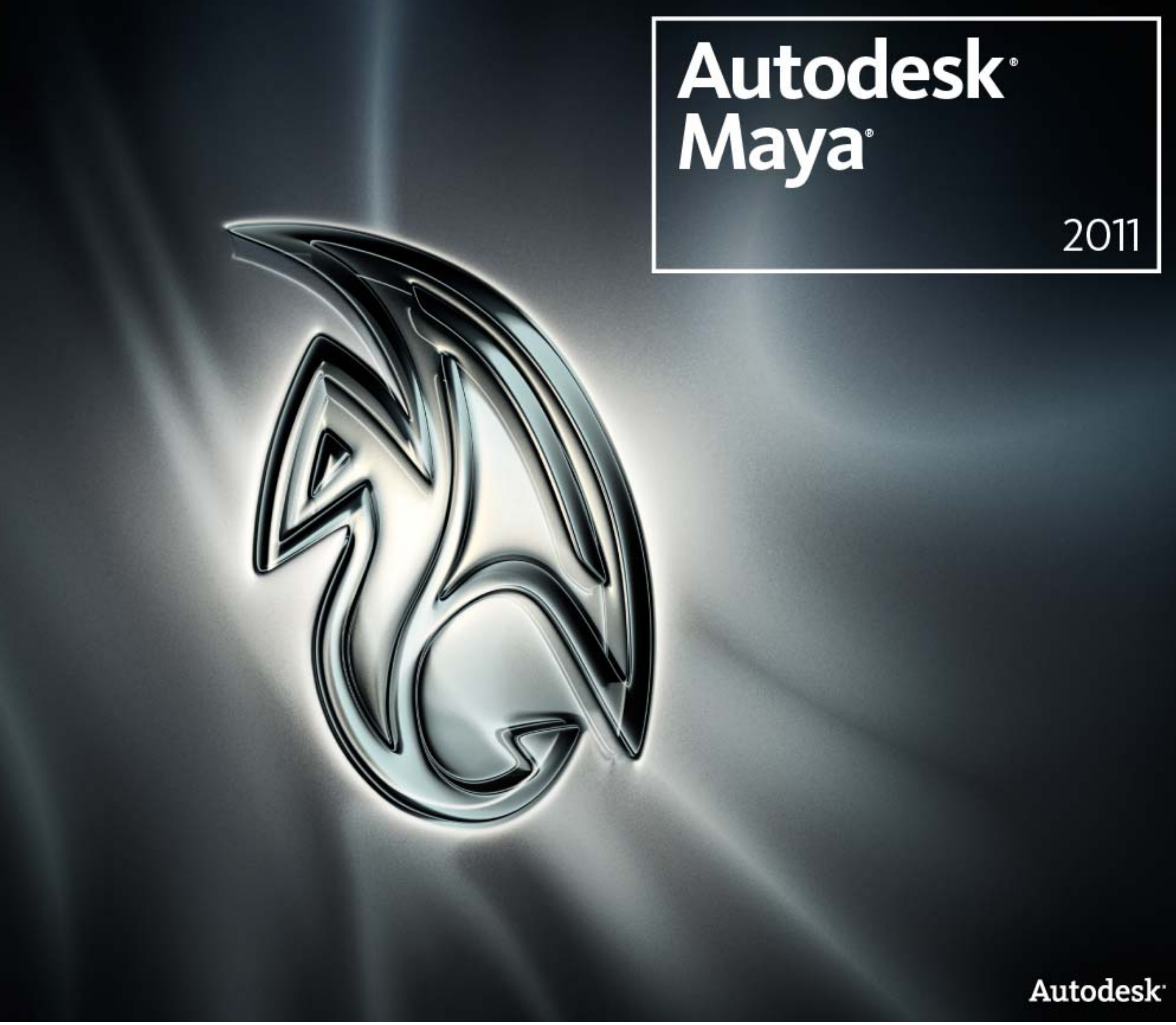


Autodesk® Maya® 2011

Games

Copyright Notice

Disclaimer

THIS PUBLICATION AND THE INFORMATION CONTAINED HEREIN IS MADE AVAILABLE BY AUTODESK, INC. "AS IS." AUTODESK, INC. DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE REGARDING THESE MATERIALS.

Contents

Introduction

1

Introduction

This documentation provides information on how to use the Maya game translators.

IMPORTANT Since Maya 4.0, we encourage you to use Maya's API directly. Several new functions have been added to API since the MDT layer was created for Maya 1.0 which means the need to use the MDT layer has reduced.

Maya game translators are provided for licensed Autodesk customers. The source code for the translators as well as the MDtApi is provided so that the supplied translators can be modified for site-specific customizations.

The game translators, VRML2, RTG and GE2, are written with an API interface layer (MDtApi). The MDtApi functions are implemented using the Maya API. MDtApi does not replace the Maya API in any form. The Dt functions are meant to ease the re-use of source code between the translators and different platforms and to provide example API source code.

The game translators are File Translator plug-ins accessed from the File > Export All or File > Export Selection menu items.

Translator processing has two basic stages internally

- Creating an internal database of information that allows for relatively easy access to the Maya scene information. This database can be considered as a grouping mechanism.
- Running the translator using the information from the data base.

This process divides into separate options that you need to define. The first set of options deals with the setup of the internal database, and the second with the actual options for the translator that you are going to run.

Common options for the internal data base

Animation options including frame ranges, the level of requested animation

Export Options including hierarchy information, tessellation parameters, verbose activity reporting

Texture Options including shader evaluation parameters

Translator-specific options are described for each translator.

Run a games translator

The Maya games translators include VRML2 RTG and GE2.

Running a translator involves two processes:

- Loading the translator from the Window > Settings/Preferences > Plug-in Manager window.
- Starting the translation from either the File > Export All or File > Export Selection menu items or from within a MEL script, using the commands:

```
file -rename output_filename
file -exportAll
```

or

```
file -exportSelected
```

Customize and rebuild translators

The following shows you how to recompile and/or rebuild translators.

To recompile the translators:

1 Before you begin, you need to have a C++ compiler, such as:

(Linux®) SGI C++ 7.3.1

(Windows®) Microsoft® VC++ 6.0

(Mac® OS X) Metrowerks® CodeWarrior 7.1

2 You can recompile the translator by creating a local work area and populate it with the following directory tree:

MDt/MDtApi

MDt/include

MDt/lib

MDt/vrml2Export

MDt/rtgExport

MDt/ge2Export

3 Copy the respective directories from $MAYA_LOCATION/devkit/obsolete/games.

Maya contains Makefiles for CodeWarrior®, SGI®, and MS IDE project files that can be used to rebuild the translators.

NOTE You may have to edit the Makefiles and customize the project files for specific Maya installation areas.

4 Build the MDtApi library first. This creates a static library used for the other translators. It is possible to create this as a LIB (Mac OS X), DSO (Linux), or DLL (Windows) file, but for the moment it is easier to create a static library for updates and changes to the translators.

(Linux) The libMDtApi.a will be copied to the lib directory and referenced from there.

(Windows) The libMDtApi.lib is found in the Release subdirectory and is referenced from there.

(Mac OS X) The libMDtApi.lib is found in the CFMSupport folder and is referenced from there.

Build the necessary translators

NOTE For Mac OS X, you do not need to build any translators, since the libraries are already provided.

1 Copy the translator plug-in file to the user's Maya work area into the plug-ins directory. There should be a work area with a directory structure such as:

$HOME/maya/prefs

$HOME/maya/projects

$HOME/maya/plug-ins

$HOME/maya/scripts

2. Unload the current version of the plug-in from Window > Settings/Preferences > Plug-in Manager.
3. Compile the new version of the plug-in.
4. Place the new translator plug into the plug-ins directory, and the MEL option script into the scripts directory.
5. Reload the plug-in from Window > Settings/Preferences > Plug-in Manager.
6. If the MEL option script has been modified, source it again so that the changes are updated within Maya. Else the changes will not be seen.
7. Run the translator.

Installing on different platforms

The direct translators for games are installed as a standard part of the Maya software under devkit/games.

The game translator directory structure is described in the following depending on which platform Maya is running.

NOTE In the following directory locations it is assumed that for Windows, Maya is installed on drive C:. If this is not the case, change the “c:” references to the drive where it is installed.

The translator binary image .lib (Mac OS X), .so (Linux) or .mll (Windows) files are in:

- (Linux)`/usr/autodesk/maya2011/bin/plug-ins/`
- (Windows) `C:\Program Files\Autodesk\Maya2011\bin\plug-ins`
- (Mac OS X) `/Applications/Autodesk/maya2011/Maya.app/Contents/plug-ins`

The MEL script files used for user options are in:

- (Linux) `/usr/autodesk/maya2011/scripts/others/`

- (Windows) `C:\Program Files\Autodesk\Maya2011\scripts`
- (Mac OS X) `/Applications/Autodesk/maya2011/Maya.app/Contents/scripts`

To run the translators, the binary plug-in and the MEL option files should be all that is needed. If the source code is necessary, it can be found under the devkit directory tree:

- (Linux) `/usr/autodesk/maya2011/devkit/obsolete/games/`
- (Windows) `C:\Program Files\Autodesk\Maya2011\devkit\obsolete\games`
- (Mac OS X) `/Applications/Autodesk/maya2011/devkit/obsolete/games`

These include the following directories:

- MDtApi
- include
- vrml2Export
- rtgExport
- ge2Export

NOTE You must create the lib directory yourself. The lib is not included by default.

Load translators in Maya

After you add the games plug-ins to the list of plug-ins, you still need to load the plug-ins to use them.

1. Select Window > Settings/Preferences > Plug-in Manager to open the Plug-in Manager window and display the list of all known plug-ins.
2. Find the plug-in you need and click the loaded check box to load the plug-in. If you only turn on the auto load, you must restart Maya to load the plug-in.

VRML2 Translator

2

Overview of VRML2

The VRML2 translator lets you export Maya scene files into VRML2 format. NURBS surfaces are tessellated to polygons and then output.

NOTE The VRML2 translator does not output all of the Maya feature set.

VRML2 Features

Outputs

- polygon geometry
- NURBS surfaces - tessellated into either triangles or quads
- hierarchy structure if wanted
- VRML2 primitives (if tagged)
- shader parameters
- textures - SGI RGBA image format
- rigid TRS animation
- vertex animation
- Camera information
- Light information

Commands

There is a MEL command included in the VRML2 translator. This command, vrml2Tags, let you define dynamic attributes onto objects that the translator queries and uses to define the output file.

Attributes

The following dynamic attributes can be defined and used with this translator.

- userAnimated boolean: On = force vertex animation processing
- VRML2Primitive: long, cube/plane
- VRML2Billboard: long, camera facing/Y facing/Z facing

Files required to use this plug-in

The files required for use of this plug-in are:

- vrml2Export.lib (Mac OS X), vrml2Export.so (Linux), or vrml2Export.mll (Windows)
- vrml2TranslatorOpts.mel

Source code can be found in:

```
(Linux) /devkit/obsolete/games/vrml2Export
```

```
(Windows) Program Files\Autodesk\Maya2011\devkit\obsolete
\games\vrml2Export
```

```
(Mac OS X) Applications/maya2011/devkit/obsolete/games/vrml2Export
```

Limitations

- On Mac OS X, the VRML2 Exporter does not export textures.
- The range for the light intensity in VRML2 is [0 to 1.0]. Maya doesn't have a limit. The translator assumes a scaling of [0-100] and divides down the Maya setting.
- No checks are made for negative scales. VRML2 defines scale to be greater than 0.
- Only SGI RGB/RGBA file textures are output. Microsoft's VRML2 viewer does not support this format. Cosmoplayer does read the texture files on both Linux and Windows systems. You may have to set up the file

associations on the Windows platform to run Netscape instead of Explore if the auto launch feature is used on Windows systems.

- You need to turn on Sample for Texture Options to get proper rotation of textures.
- There may be memory leaks in the translator. This is because not all of the intermediate internal structures are deleted (for instance, animating NURBS surfaces will create a noticeable memory leak with Maya 1.0 due to the tesselation).
- Vertex animation is based on the connection of the inMesh attribute, so this will cause more geometry to be considered animated that may be. For final results, you may want to clean up the history of non-animated tweaks. A check for dynamic attribute called userAnimated to force usage has been added. Use the Add Attribute command to add userAnimated as a boolean flag and then set it ON for those objects that you want to force the vertex animation to recognize and use. This will mostly be for those objects that don’t have history and an inMesh connection.
- Instanced geometry is output multiple times but doesn’t get reused.

Install the VRML2 translator

You install the VRML2 translator plug-in using the Plug-in Manager.

1. Select Window > Settings/Preferences > Plug-in Manager to display the list of all known plug-ins.
2. Click the loaded check box beside vrml2Export.so (Linux), or vrml2Export.mll (Windows), or vrml2Export.lib (Mac OS X) to load the plug-in.

   If you only turn on the auto load, you must restart Maya to load the plug-in.

Set VRML2 properties in Maya

VRML2 Plug-ins

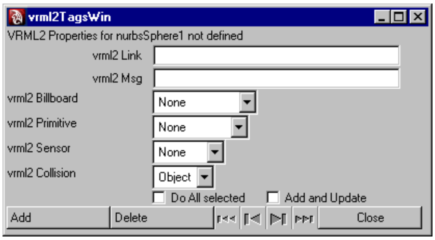
The vrml2Tags command lets you attach tags to objects that have no effect in Maya as dynamic Attributes, but will be added to the VRML2 file.

The vrml2Tags command is a MEL script file vrml2Tags.mel which is found in the scripts/other directory.

To set VRML2 properties:

1 Select the objects you want to create with VRML2 attributes.

2 Type vrml2Tags in the Command Line to run the vrml2Tags command. The following window displays.



vrml2TagsWin

VRML2 Properties for nurbsSphere1 not defined

vrml2 Link:

vrml2 Msg:

vrml2 Billboard: None

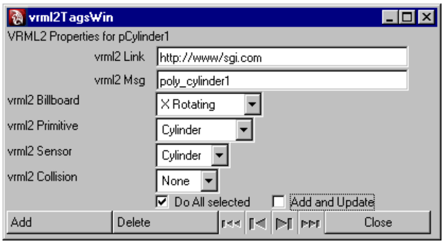
vrml2 Primitive: None

vrml2 Sensor: None

vrml2 Collision: Object

☐ Do All selected ☐ Add and Update

3 Change the settings of the vrml2 tags in this window and click the Add button.



vrml2TagsWin

VRML2 Properties for pCylinder1

vrml2 Link: http://www/sgi.com

vrml2 Msg: poly_cylinder1

vrml2 Billboard: X Rotating

vrml2 Primitive: Cylinder

vrml2 Sensor: Cylinder

vrml2 Collision: None

☑ Do All selected ☐ Add and Update

4 These attributes are now available for further changes in the Extra Attributes section of the object’s Attribute Editor.

See the following for attribute descriptions and workflows.

Link Objects to Web Addresses with VRML2Link

The Link and Message attributes lets you create, verify, or remove hypertext links.

To link objects to web addresses

1 Select the object or objects you want to link.

2 Execute the vrml2Tags MEL command to display the current VRML2 tags information. Type vrml2Tags in the Command Line.

3 Press the Add button at the bottom of the window.

The attributes are now accessible from the selected object's Attribute Editor's Extra Attributes section.

4 In the vrml2 Link box, enter a Universal Resource Locator (URL). For example:

```
http://www.alias.com
```

You can also add a description of the destination using the form:

```
URL*DESCRIPTION
```

Where URL is the destination, the * (asterisk) is a separator, and DESCRIPTION is a description that will be displayed in the VRML browser when you point at the link.

For example:

```
http://www.alias.com*Alias Web Page
```

5 When you output the scene using File > Export All, the objects become active VRML links.

Create preset VRML viewpoints

Viewpoints are predefined camera positions and views in a VRML world. Viewpoints help viewers navigate your VRML worlds (for example, allowing them to return to an entrance point, or view close-ups of objects).

To create VRML viewpoints

1 Create a new camera.

2 Name the camera. Use a name that contains only valid VRML2 characters. This means do not use dashes or [] {}. For example, a valid VRML2 camera name would be `EntranceView`.

3 On the URL in the vrml2 Link field, type the name of the viewpoint this object will trigger, preceded by #. For example:

```
#EntranceView
```

vrml2TagsWin attribute descriptions

The following describes the attributes in the vrml2TagsWin. Whatever you set in the vrmlTags window displays and becomes editable in the Extra Attributes section of the object's Attribute Editor.

vrml2 Billboard menu

The VRML2 billBoard options include the standard VRML2 options, None, X-Rotating, Y-Rotating, and Screen Aligned.

vrml2 primitive menu

The VRML2 primitive options include the standard VRML2 options, None, Box, Cone, Cylinder, Sphere, and Elevation Grid.

vrml2 Sensor menu

The VRML2 sensor options include the standard VRML2 options, None, Cylinder, Sphere, Plane, Proximity, Touch, and Visibility.

vrml2 Collision menu

The VRML2 collision options include the standard VRML2 options, None and Object.

Do All selected checkBox

This checkbox performs the Add/Update operation on the entire current selection list. If it is not turned on, the current item in the selection list as shown at the top of the vrml2TagsWin window is used.

This is only useful if multiple objects are currently selected.

Add and Update checkBox

This checkbox combines the add and update functions at the same time. When turned off, the Add operation creates the dynamic attributes on the objects and uses the default settings. When turned on, the currently displayed settings are used to assign to the attributes.

Add button

Click this button to create default dynamic attributes on the objects in the current selection list. See Do All selected checkBox on page 12 and Add and Update checkBox on page 12 options for details.

Delete button

Click this button to delete the VRML2 dynamic attributes on the objects in the current selection list. See Do All selected checkBox on page 12 and Add and Update checkBox on page 12 options for details.

Selection stepping controls

The four stepping control buttons let you select which item to use for the individual object of interest. This is only useful if multiple objects are currently selected.

The buttons from left to right are first object, previous object, next object, and last object.

The current object of interest is listed in the first line of the vrml2Tags window.

Close button

Click this button to close the VRML2 tags window.

Import a VRML2 file using wrl2ma

The wrl2ma stand-alone program lets you convert VRML2 files to Maya ASCII.

The vrml2 importers include:

(Linux and Mac OS X) wrl2ma

(Windows) wrl2ma.exe

The conversion includes:

- geometry
- normals
- texture coordinates
- color per vertex
- shader parameters (Lambert and Phong)
- texture file references (in-line textures are not supported)

NOTE Animation is not supported with this version.

To use wrl2ma

➤ Specify the input .wrl file and the output .ma file, and also specify the extensions you want.

For example, type something similar to the following in a system shell or terminal:

```
  wrl2ma -i Porsc911.wrl -o Porsc911.ma
```

This example converts the VRML file Porsc911.wrl to a Maya 2011 ASCII file named Porsc911.ma. The command provides the following options:

Options	What it does
-h	Prints the help file
-i <inputFile>	Specifies the input file to convert
-o <outputFile>	Specifies the output file to save to
-m	Assume input file was output by Maya
-v	Prints verbose messages
-d	Uses a debugging aid to see how the lib-vrml97 scene parser parses the original VRML file.

Menus

File

File > Export All > VRML2

Animation Options

Range Control options

Loop Enables the VRML2 option to loop the animation on playback.

Time Slider Uses the start/end range from the time slider instead of from the settings in this window.

Enabled Click to quickly disable any animation without resetting the ranges.

Start/End The Start value specifies the frame to start the output from (integer values). The End value specifies the frame to end the output.

Step This value defines the number of animated frames by which the animation is sampled between exported frames.

Frames per sec This value specifies the VRML2 playback frame rate.

Animate

Click to turn on the following options. A check mark displays if the option is on and the option you select will animate.

Vertices Animates vertex animation at each frame (can be slow for NURBS geometry).This option is off by default.

Transf Animates transformation animation at each frame. This option is on by default.

Materials Animates shader material parameter animation. This option is on by default.

Lights Animates light parameter animation. This option is on by default.

Cameras Controls camera process information.

Keyframe using Anim Curves

Output by Keyframes on Animation curves. This option is off by default. (**Note**: This is a future option and is not currently implemented.)

Export Options

These options determine what elements you want to export.

Hierarchy

World A World hierarchy tells the VRML 2 translator not to export hierarchy. This means that all vertices use world space.

Flat A Flat hierarchy means one level of hierarchy. Selecting Flat means that transformations use world space and vertices use local space. There is no hierarchy/parenting information. This is the default Hierarchy selection.

Full A Full hierarchy exports the full DAG tree hierarchy. This means that transformations and vertices use local space. Full hierarchy/parenting information is available.

Joints Allows NULL chains to be output for hierarchy information, mostly for skeletons/joints.

Export

All Exports all scene information (everything visible in the scene). This is the default Export selection.

Picked Exports everything picked in the scene (items lower in the DAG tree from the DAG nodes of picked objects are not included).

Active Exports everything picked in the scene as well as any objects that are below them in the DAG tree.

Tesselation

Tri Tessellates NURBS surfaces into triangles. This is the default Tessellation selection.

Quad Tessellates NURBS surfaces into quads.

Include

Cameras Allows Camera information to be collected for the database. This option is on by default.

Lights Allows Light information to be collected for the database. This option is on by default.

Debug Info

Geo/Mat Outputs a lot of information during the export process. Shape/material information is output here.

Cameras Controls camera process information.

Lights Controls light process information.

Texture Options

Evaluate Uses convertSolidTx to create image files for both file and procedural textures. If turned off, Maya reads in the file textures as they are, and only converts the procedural textures.

Sample Uses the 2D texture placement transformation information to generate the file image for output. For instance, the checker texture image consists of a checker board of 8x8 checkers. If Sample is off, the UV repeats are set to 1 internally, UV Rotate is set to 0.0 internally, and then the texture is processed. The result is that the checker image will be 2x2 (assuming the default 4 repeat).

Original For file texture images, turn on to use a reference to the original file texture as defined in the texture node.

Resolution settings

Set the following X and Y texture resolution swatch sizes by entering a value and pressing Enter (Linux and Windows) or Return (Mac OS X), or by dragging the slider.

X Tex Res/ Y Tex Res Specifies the X or Y swatch size used for procedural textures.

Max X Tex Res/ Max Y Tex Res Specifies the Max X or Y swatch size used for all textures. File textures larger than the size specified here are rescaled to the maximum size when the texture is computed with the convertSolidTx command.

Texture Search path

This is a list of directories to search for file textures if not found in the default list of directories obtained with workspace -q -rd .

VRML2 options

Navigation

This is the VRML2 navigation list. These settings control the initial set-up of the VRML browser (or viewer) when it opens the .wrl file. Click to turn on the following items (a check mark displays) to suit your needs.

Walk Walk navigation is used to explore a virtual world on foot or in a vehicle that rests or hovers above the ground. It is strongly recommended that Walk navigation defines the up vector in the Y direction and provides some form of terrain and gravity to produce a walking or driving experience. If on, the browser supports collision detection. This option is off by default.

Examine Examine navigation is used to view individual objects and often includes (but does not require) the ability to spin around an object and move the viewer closer or further away. This option is on by default.

Fly Fly navigation is similar to Walk navigation except that terrain and gravity can be disabled or ignored. This option is off by default.

Any If the Any navigation mode is selected, the browser chooses the navigation mode that best suits the content and provides a user interface that lets you change the navigation mode dynamically. If off, you cannot change the navigation mode and the browser only uses the modes specified in the list. This option is on by default.

None None navigation disables and removes all browser-specific navigation user interface. This means that you can only navigate using mechanisms provided in the scene, such as Anchor nodes or scripts that include loadURL().This option is off by default.

Options: Headlight

This is the VRML2 headlight control option. When turned on, the viewer casts light on the scene. This option is on by default.

Navigation speed

The value in this box is the VRML2 navigation speed setting. Enter a new value and press Enter (Linux and Windows)/Return (Mac OS X) or drag the slider if you want to change the initial speed of the viewer.

Float precision

This is the number of digits used after the decimal point.

Export

Click to turn on the following options. A check mark displays if the option is on and the Export option you select will export to VRML2.

Normals Exports normals. This option is off by default.

Opposite Flips the direction of generated normals (not normally needed).This option is off by default.

Textures Exports textures.This option is on by default.

Long Lines Allows long lines in output file.

Turn off to break the VRML text file into short lines (each value on a separate line). For instance, turn this off if you want to edit the output file with a text editor that does not handle lines longer than approximately 80 characters well (such as vi). This option is off by default.

Verbose Acquires feedback during the translation. This option is off by default.

Launch viewer If turned on, on completion of the translator Maya launches Netscape (Linux) or an associated program for files with the .wrl extension (Windows).This option is off by default, and not available on Mac OSX.

Compressed Runs the gzip program to compress the output file. The file extension will still be .wrl. On Windows, the gzip.exe program should be located in the execution path. This option is off by default.

Reversed Reverses the winding for NURBS surfaces that are single- sided and have the Opposite turned on. This option is on by default.

ColorPerVertex Enables the output of color per vertex if it is available for objects being output. This option is off by default.

Texture path

The textures are saved in the directory you specify in this box.

Run script when done

Click the Browse button to select a script or enter the name of the script to run after the translation is complete. A system() call is made so the executable command should be found on the PATH statement.

Append file name to script

This option determines whether or not the output file name is appended to the end of the command line submitted. This option is on by default.

RTG Utility and File Format

3

Overview of RTG translator

The RTG translator provides a method to output the Maya scene information generically. The source code is included so that site-specific changes or customizations can be done.

RTG Features

Outputs

- polygon geometry
- NURBS surfaces - tessellated into either triangles or quads
- hierarchy structure if wanted
- shader parameters
- textures - SGI RGBA image format
- rigid TRS animation

Commands

There are no MEL commands included with the RTG translator.

Attributes

There are no specific Maya attributes queried in the RTG translator.

Files required to use this plug-in

The files required for use of this plug-in are:

- rtgExport.lib (Mac OS X), rtgExport.so (Linux), or rtgExport.mll (Windows)
- rtgTranslatorOpts.mel

Source code can be found in: devkit/obsolete/games/rtgExport

Install the RTG translator

You install the RTG translator plug-in using the Plug-in Manager.

1. Select Window > Settings/Preferences > Plug-in Manager to display the list of all known plug-ins.
2. To load the plug-in, click the loaded check box beside rtgExport.so, rtgExport.mll, or rtgExport.lib (depending on your operating system).

   If you only turn on the auto load, you must restart Maya to load the plug-in.

Menus

File

File > Export All > RTG

RTG translator options

RTG Options

Base options

Click to turn on the following options. A check mark displays if the option is on and the option you select will be enabled during output.

Verbose Acquires feedback during the translation. This option is off by default.

Reversed Reverses the winding for NURBS surfaces that are single- sided and have the Opposite turned on. This option should be turned on by default.

Image Format

Choose either SGI RGB or Alias PIX from the list provided.

Format

Linux/DOS Select one of these options to determine whether the output ASCII files are written in Linux <lf> or DOS <cr><lf> format. Click to turn on the following options. A check mark displays if the option is on and the format option you select will output to rtg.

V1.8 Compatible Enables some of the PA RTG 1.8 output formats.

V Normals Outputs Vertex Normals.

P Normals Outputs Polygon Normals.

V Colors Outputs Vertex Colors.

UV Coords Outputs UV coordinates.

indexCnt Outputs index counters at the beginning of each list entry.

Degrees Outputs the transformation angle in degrees instead of radians.

Materials Outputs shading group Material properties.

MultiTexture Outputs separate texture images for those material properties that are texture mapped, including: diffuse, bump, translucence, specular color, cosinePower etc.

MDecomp Decomposes the transformation matrix into components of T/R/S.

Pivots Outputs pivot information.

Transforms Outputs transformation information.

Local Xform Outputs the local transformation matrix [4x4].

Animation Outputs T/R/S animation.

All Nodes Forces all nodes to output animation.

Animation Options

Range Control options

Time Slider Uses the start/end range from the time slider instead of the settings in this window.

Enabled Click to quickly disable any animation without resetting the ranges.

Start/End The Start value specifies the frame to start the output from (integer values). The End value specifies the frame to end the output.

Step This value defines the number of animated frames by which the animation is sampled between exported frames.

Export Options

These options determine what elements you want to export.

Hierarchy

World A World hierarchy tells the RTG translator not to export hierarchy. This means that all vertices use world space.

Flat A Flat hierarchy means one level of hierarchy. Selecting Flat means that transformations use world space and vertices use local space. There is no hierarchy/parenting information. This is the default Hierarchy selection.

Full A Full hierarchy exports the full DAG tree hierarchy. This means that transformations and vertices use local space. Full hierarchy/parenting information is also available.

Joints

Allows NULL chains to be output for hierarchy information, mostly for skeletons/joints.

Export

All Exports all scene information (everything visible in the scene). This is the default Export selection.

Picked Exports everything picked in the scene (items lower in the DAG tree from the DAG nodes of picked objects are not included).

Active Exports everything picked in the scene as well as any objects that are below them in the DAG tree.

Tessellation

Tri Tessellates NURBS surfaces into triangles. This is the default Tessellation selection.

Quad Tessellates NURBS surfaces into quads.

Include

Cameras Allows Camera information to be collected for the database. This option is on by default.

Lights Allows Light information to be collected for the database. This option is on by default.

Debug Info

Geo/Mat Outputs a lot of information during the export process. Shape/material information is output here.

Cameras Controls camera process information.

Lights Controls light process information.

Texture Options

Texture control

Evaluate Uses convertSolidTx to create image files for both file and procedural textures. If turned off, Maya reads in the file textures as they are, and only converts the procedural textures.

Sample Uses the 2D texture placement transformation information to generate the file image for output. For instance, the checker texture image consists of a checker board of 8x8 checkers. If Sample is off, the UV repeats are set to 1 internally, UV Rotate is set to 0.0 internally, and then the texture is processed. The result is that the checker image will be 2x2 (assuming the default 4 repeat).

Original For file texture images, turn on to use a reference to the original file texture as defined in the texture node.

Resolution settings

Set the following X and Y texture resolution swatch sizes by entering a value and pressing Enter (Linux and Windows)/Return (Mac OS X) or dragging the slider.

X Tex Res/Y Tex Res Specifies the X or Y swatch size used for procedural textures.

Max X Tex Res/Max Y Tex Res Specifies the Max X or Y swatch size used for all textures. If the size of file texture images is larger than the maximum specified here, then the file textures will be scaled down to this maximum setting when the convertSolidTx command is processed.

Texture Search path

This is a list of directories to search for file textures if not found in the default list of directories obtained with workspace -q -rd .

Game Exchange 2 Translator

4

Overview of GE2 translator

The GE2 translator lets you export Maya scene files into GE2 format.

GE2 Features

The following lists GE2 Outputs on page 27 and Files required to use this plug-in on page 28.

Outputs

- polygon geometry
- NURBS surfaces - tessellated into either triangles or quads
- hierarchy structure if wanted
- shader parameters
- textures - possible formats are: SGI RGBA, Alias wirefile, bmp, png, ppm, and tif

  NOTE GE2 export does not properly produce .bmp format image files for converted textures. We suggest you specify .sgi or .tif format instead, and convert the textures to .bmp.

- TRS animation
- vertex animation

Commands

Use the ge2tags script to assign ge2 properties to Maya shaders. This command adds a set of extra attributes to all Maya shaders, and directs the Attribute Editor to display the new attributes in a tabbed, easy-to-manage format.

Attributes

The ge2tags script assigns extra attributes to shaders to correspond with ge2 Material attributes. These can be edited and queried just as any other attributes.

Files required to use this plug-in

The files required for use of this plug-in are:

- ge2Export.lib (Mac OSX), ge2Export.so (Linux), or ge2Export.mll (Windows)
- ge2TranslatorOpts.mel
- ge2tags.mel
- ge2MatAttr.mel
- AElambertGe2Revision.mel
- Source code can be found in: devkit/obsolete/games/ge2Export

Install the GE2 translator

You install the vrml translator plug-in using the Plug-in Manager.

To load the plug-ins

1. Select Window > Settings/Preferences > Plug-in Manager to display the list of all known plug-ins.
2. Click the loaded or auto load check box to load one of the ge2Export.mll, or ge2Export.so, or ge2Export.lib plug-ins.

Menus

File

File > Export All > GE2

Domain Options

Click to turn on the options. A check mark displays if the option is on and the option you select will be enabled during output. The checked domains will be filled with the material attributes.

Format data for domain: Options include: GL/DirectX, SonyPSX, N64, Custom

Animation Options

Enable Animation

Click to quickly disable any animation without resetting the ranges.

Start/End The Start value specifies the frame to start the output from (integer values). The End value specifies the frame to end the output.

Step The Step option is only available when the Keyframe using: Anim Curves option is turned off. This value defines the number of animated frames by which the animation is sampled between exported frames.

Animate options

Click to turn the following options on or off.

Vertices If on, Vertices exports vertex animation.

Transforms If on, Transforms exports TRS animation.

Lights If on, Lights animates the lights in the scene.

Camera If on, Camera animates the camera in the scene.

Keyframe using options

Click to turn the following options on or off.

Anim Curves If on, Anim Curves extracts keyframes from the AnimCurves on the shapes. If this option is selected, Step is set to 1 and disabled to ensure keyframes will fall on exported frames. If off, Step is enabled and you can change the default value.

Sampling When on, Maya adaptively samples the animation. If this option is selected, Sample By and Tolerance are enabled. Frames are sampled at the given rate and exported as keyframes if geometry does not meet tolerance requirements.

Sample By You must select Sampling in the Keyframe Using section to enable this option. This value represents the sample rate for adaptive sampling.

Tolerance You must select Sampling in the Keyframe Using section to enable this option. This value determines whether sampled frames are exported as keys.

Animation Displacement options

These options are only enabled if you select Vertices in the Animate section.

- Relative to Current Frame specifies relative vertex displacements.
- Absolute generates absolute vertex displacement.

Export Options

These options determine what elements you want to export and how to export them.

Hierarchy

World A World hierarchy tells the GE2 translator not to export hierarchy. This means that all vertices use world space.

Flat A Flat hierarchy means one level of hierarchy. Selecting Flat means that transformations use world space and vertices use local space. There is no hierarchy/parenting information. This is the default Hierarchy selection.

Full A Full hierarchy exports the full DAG tree hierarchy. This means that transformations and vertices use local space. Full hierarchy/parenting information is exported.

Export

All Exports all scene information (everything visible in the scene). This is the default Export selection.

Selected Exports everything picked in the scene (items lower in the DAG tree from the DAG nodes of picked objects are not included).

Active Exports everything picked in the scene as well as any objects that are below them in the DAG tree.

Tessellation

Tri Tessellates NURBS surfaces into triangles. This is the default Tessellation selection.

Quad Tessellates NURBS surfaces into quads.

Export

If these options are on, (by default, all the Export options are on), Maya exports information for lights, cameras, textures, geometry, and normals.

Export options

The following options are off by default.

Reverse winding If on, Maya enables reverse winding on polygons. Default winding is counter-clockwise.

Flip Normals If on, Maya reverses the direction of normals. This option is off by default.

Texture Options

Texture Control

Uses convertSolidTx to create image files for both file and procedural textures. If turned off, Maya reads in the file textures as they are, and only converts the procedural textures.

Use 2d and 3d texture placement Uses the 2D texture placement transformation information to generate the file image for output. For instance, the checker texture image consists of a checker board of 8x8 checkers. If off, the UV repeats are set to 1 internally, UV Rotate is set to 0.0 internally, and then the texture is processed. The result is that the checker image will be 2x2 (assuming the default 4 repeat).

Use convertSolidTx on file textures This is a future option and is not currently implemented.

Use original file textures If on, Maya reads in the file textures as they are, and only converts the procedural textures. If off, both procedural and file textures are converted.

Resolution settings

Set the following X and Y texture resolution swatch sizes by entering a value and pressing Enter (Linux and Windows)/Return (Mac OS X) or dragging the slider.

X Tex Res/Y Tex Res Specifies the X or Y swatch size used for procedural textures.

Max X Tex Res/Max Y Tex Res Specifies the Max X or Y swatch size used for all textures. If the size of file texture images is larger than the maximum specified here, then the file textures will be scaled down to this maximum setting when the convertSolidTx command is processed.

Texture path

This is a list of directories to search for file textures if not found in the default list of directories obtained with workspace -q -rd.

Other Options

Float precision

Select a menu item to print out floats with the number of decimal places you choose.

Format Output

Tabs If on, output is tabbed for readability. This option is on by default.

Comments If on, output is commented (numbered indices, etc.). This option is off by default.

Debug Output

The following options are off by default.

Geometry If on, Maya outputs information pertaining to geometry and anything attached to geometry (including materials) during the export process. A lot of information is generated during this process.

Cameras If on, Maya outputs information about cameras during the export process.

Lights If on, Maya outputs information about lights is during the export process.

Run script when done

Specify a script (and command line options) to be run after export to convert ge2 files to another file format or otherwise post-process the files.

Append file name to script

If on, the appropriate file (.grp or .gaf file) will be appended to the script you specify in Run script when done before invoking.

MDt API

5

Overview of Games export toolkit

IMPORTANT We encourage you to use Maya's API directly. Several new functions have been added to API since the MDT layer was created for Maya 1.0 which means the need to use the MDT layer has reduced. The source code has also been moved to: devkit/obsolete/games.

The MDt API is an implementation of the GAME EXPORT toolkit for Maya. It is a wrapper layer around the Maya API. Its main purposes are:

- To enable reuse of common code in multiple translators.
- Ease conversion of PA game translators over to Maya.
- Provide a level of grouping information obtained from the Maya API.
- Provide an additional source of Maya API examples.

NOTE The MDt Api DOES NOT replace the Maya API. MDt Api functions work by using the Maya API.

Incorporate functions/ideas

The MDt Api compiles into a static library which the translators can be linked with. The files that make up the MDt Api library are:

MDtShape.cpp	geometry grouping (vertices, normals, colors, tessellation)

MDtMaterial.cpp	shaders, generating texture images, material properties
MDtLayer.cpp	miscellaneous control functions/startup/cleanup
MDtLight.cpp	light parameters
MDtCamera.cpp	camera parameters
MDtCnetwork.cpp	not used currently
iffreader.cpp	class to read iff files
iffwriter.cpp	class to write iff files
MDt.h	main header file for MDtApi
MDtExt.h	extension functions for MDtApi, access to original Maya objects so that the Maya API can be used with them.

Use the MDt library

The MDt library works by walking through the DAG structure of the scene and creating internal data tables that are then referenced by the MDt functions. This process can take a bit of time for large files or NURBS surfaces. If it takes too much time to generate the internal data tables, using the Maya API directly may better suit your needs.

Once the data tables are created, the translator is then able to access the tables and output the data in whatever format it needs to. References to the original Maya objects are stored (temporarily) so they can be referenced to get additional data without having to walk the DAG tree again.

Start with MDt functions and then use Maya API to obtain information that is not cached in the MDt data tables.

TIP It is not recommended that you start with the Maya API and add in the MDt functions. The MDt functions can be used to quickstart or prototype a translator, and then use the knowledge gained to refine/customize the translator.

Find API function documentation

The documentation for the MDt API is integrated into the source files themselves where possible.

Use the MDt API

The MDt API is an example of how to use the Maya API. It is not the only method of obtaining Maya scene information from Maya. It is a combination of Maya API functions and executing MEL commands to get access to the data.

MDt API is not meant to replace Maya API in any way. It is grown and functions are added when deemed that something is of common interest to multiple uses. For example, not all of the Shader attributes are cached in the database. Those that are static and found to be used repeatedly are cached, but the API allows for the original Maya Object to be referenced to obtain additional data that is needed. It can be thought of as a higher level grouping function of common routines.

The two-stage process of generating the internal database and then running a translator with that data is a very general implementation. The scanning of the DAG tree and populating the internal database takes time. It is possible to write faster translators using the Maya API functions directly. The approach used for the VRML2 and other MDt based translators was to reduce the amount of work that had to be redone for each translator, and to reuse what was possible from pre-existing translators.

The VRML2, RTG, and GE2 game translators have been implemented as "file translators". They could also be implemented as normal Maya MPxCommand plug-ins.

Structure of the internal database

The database that is generated by the walking of the DAG tree is controlled by user options that are set/defined in the option box of the file translators.

The information gathered for the database consists of:

Shapes, Groups, Materials, Textures, Lights, and Cameras

Controls

These options are described in the sections for each of the games translators using the MDt API.

There is a lot of flexibility in the settings that can be set. The source code is also supplied so that additional options can be added for local customizations.

The options are implemented in the file MDtLayer.cpp and the header file MDtExt.h. It is possible to add additional control flags to the source code and also modify the MEL option scripts to set/reset the options.

This would also then modify the MDt files that implement the particular area of interest for the option. The translator Option MEL script file would then be modified to pass the values of the options to the translator itself.

Shapes

Shapes are the basic transformation/joint/polygonal/NURBS geometry objects. Each "shape" has an associated number of parameters.

- Transformations
- Materials/Groups
- Parents

Each shape also has an associated list of Groups. If the shape is a pure transformation/joint node then the number of groups will be Zero (for instance, no shader assignments). This usually only occurs when the hierarchy is set to FULL, or all of the transformations are flattened out, and geometry is associated with all of them.

Some API functions that operate using the Shape index are:

- DtShapeGetCount() - returns the number of shape nodes found.
- DtShapeGetMatrix(idx, *matrix) - returns the transformation matrix for shape idx, in the variable matrix

In general it is faster to use the geometry from the ByShape based functions rather than those of the ByGroup family. There isn't a second copy of the data generated and passed by to the caller.

The Shape functions are found in the source file MDtShape.cpp.

Groups

Groups are subsets of its Shape, that are defined by multiple shader assignments to it. For a NURB object there would be at most 1 group. For polygonal objects there would be 1 group for each shader that is assigned to that object. If there are no shaders assigned to the polygonal object then the number of groups will be zero. In this case it is most likely to be a transformation node of some kind.

The ByGroup functions need to allocate memory to generate the lists that are returned. Since the API functions don't know when that memory is no longer used, it is up to you to free it when done with the returned lists.

The MDtShape.cpp file implements these functions.

Materials

Materials are the shaders found in the DAG tree walk.

For each shader that is found, a reference is stored for it, and can be accessed by either "name" or ID number. From the reference it is possible to use the Maya API directly to obtain the shader parameters of interest.

The MDtMaterial.cpp file implements these functions.

Textures

Textures are the textures mapped for the shaders. By default, only the color and transparency textures are evaluated and a combined rgba file image is generated in memory. It is possible to enable extra processing to output all or most of the textures mapped on a shader, this is part of the "MultiTexture" option.

The MDtMaterial.cpp file implements these functions

Lights

Lights information is gathered and made available. Similarly to Materials, a reference is temporarily saved to access the light parameters of interest.

The MDtLights.cpp file implements these functions

Camera

Camera information is gathered and made available. Similarly to the Materials, a reference is temporarily saved to access the camera parameter of interest.

The MDtCamera.cpp file implements these functions

Utility

There are some utility functions that allow for decomposing matrices, and other general functions. This are found in the MDtLayer.cpp file along with the control options for the DAG walk processing

Animation

Obtaining animation data out of the MDt functions is the same as getting the non-animation data out.

When the DAG tree is walked for generating the internal database, the references that are saved are used to obtain the geometry/material information that is needed.

Iterating over a range of animation, it is then accomplished with the DtFrameSet(frame) call which sets the current frame time. Maya automatically updates its dependency graph.

For each frame, DtFrameSet then updates the parameters that are wanted. Currently, this is mostly the geometry (vertices, normals, uvs) so the same function calls are reused to get the new frames geometry data.

The transformation data is obtained by referencing the transformation object. This data is not cached.

We recommend that you iterate over the animation range as few times as possible. See the following example:

```
for ( animation = start, to end )
 for ( shape = 0, to last Shape )
  get and save TRS data for shape
 next shape
next frame
```

Minimal Program example

The bare minimum plug-in to use the MDt functions consist of:

- initializing the internal database
- walking the Maya scene graph and populating the database
- freeing the internal database

The following code snippet shows the functions that would need to be called. This snippet doesn’t actually output anything. There would normally be additional calls to customize the method of populating the internal database and actual calls to retrieve and output the data itself. This is just a sample.

```
// Initialize the internal database with a scene “name”
DtExt_SceneInit( “basic” );
//Now we can setup the database from the wire file geometry
DtExt_dbInit();
// Clean house.
DtExt_CleanUp();
```

Basic Program example

The following is a basic Dt based plug-in that queries the selected objects and outputs the name of the shape (mesh) and any “color” texture file names that are mapped to the shader.

```
#include <maya/MSimple.h>
#include <MDt.h>
#include <MDtExt.h>
DeclareSimpleCommand( basic, "Alias - Example", "2.0");
void outputFileNames()
{
char *shapeName;
char *mtlName;
char *texName;
// 1st thing find out how many shapes (meshes) are in this scene
// currently being looked at.
int numShapes = DtShapeGetCount();
// For each shape, check out what is assigned as the color texture
// There may not be a texture associated with material, or the
texture
// may be a procedural texture and so have no file name.
for ( int shape=0; shape < numShapes; shape++ )
{
// Find the name of the current shape that we are looking at.
DtShapeGetName( shape, &shapeName );
// Find out the number of groups (materials) assigned to this shape
numGroups = DtGroupGetCount( shape );
// For each group (material) check out to see if there is a file
texture
// associated with the material and output its name.
// It is possible that the number of groups will be zero. In this
 case
// there would be no materials assigned:
// 1) Joint or other Transform node
// 2) user deleted all of the materials including the default
shading group
// 3) removed the connections between the mesh and the shading
groups.
// Usual reason is #1, a Joint or straight Transform node. (when
using
// full hierarchy mode
 for ( int group=0; groups < numGroups; group++ )
 {
 // For the current shape+group combination find out its material

 DtMtlGetName( shape, group, &mtlName );
 // For the current material, find out its texture file name if
any
```

```
 DtTextureGetFileName( mtlName, &texName );
 // output any information that is wanted about what we found
above.
 printf( "for shape %s(%d) group %s(%d), texture filename is %s\n",
 shapeName, shape, mtlName, group, texName ? texName : "(NULL)"
);
 }
 }
}
//
// main doIt function for the plugin command.
MStatus basic::doIt( const MArgList& args )
{
// Initialize the Dt database
DtExt_SceneInit( “basic” );
// using the default settings, as no initializer
// functions are called here
// Walk the dag and fill in the internal database
DtExt_initdb();
// walk thru shapes found and output the texture file names
outputFileNames();
// Clean up the allocated memory and internal storage
DtExt_CleanUp();
// return from the plugin command
return MS::kSuccess;
}
```

Index