



**Autodesk[®]
Maya[®]**

2009

10th Anniversary Edition

Autodesk[®]

What's New

Copyright Notice

© 2008 Autodesk, Inc. All rights reserved. Except as otherwise permitted by Autodesk, Inc., this publication, or parts thereof, may not be reproduced in any form, by any method, for any purpose. Certain materials included in this publication are reprinted with the permission of the copyright holder. Portions relating to Graph Layout Toolkit © Copyright 1992-2003 Tom Sawyer Software, Berkeley, California. All rights reserved. Portions relating to TIFF © Copyright 1988-1997 Sam Leffler. © Copyright 1991-1997 Silicon Graphics, Inc. All rights reserved. Permissions to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Portions relating to OpenEXR v 1.2.2 Copyright © 2002, Industrial Light & Magic, a division of Lucas Digital Ltd. LLC All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Industrial Light & Magic nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions relating to JPEG © Copyright 1991-1998 Thomas G. Lane. All rights reserved. This software is based in part on the work of the Independent JPEG Group.

Portions relating to GCC are Copyright © 2008 Free Software Foundation, Inc.

Portions relating to Animation Repurposing Technology, Copyright 1999-2003 House of Moves Motion Capture Studios, LLC.

Portions relating to Expat XML Copyright © 1998, 1999, 2000 Thai Open Source Software Center Ltd. and Clarke Cooper. Copyright © 2001, 2002 Expat maintainers. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Portions relating to XHTML Copyright © 1998, 1999, 2000 Thai Open Source Software Center Ltd. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Portions relating to OpenVRML are Copyright © 2008 Free Software Foundation, Inc.

Portions relating to pysqlite Copyright © 2005-2007 Gerhard Häring, gh@ghaering.de.

Portions relating to Flex 2.5.4 are developed by the University of California, Berkeley and its contributors.

Portions relating to Doc ++ are copyright © 1996 Roland Wunderling, Malte Zoeckler copyright © 1999-2001 Dragos Acostachioaie.

Portions relating to the implementation of the Edge Detection and Image Segmentation (EDISON) System are provided "AS IS". More information may be found at <http://www.caip.rutgers.edu/riul/research/code.html>.

Portions of the subdivision surface implementation technology are protected by U.S. patents 6,037,949, 6,222,553, 6,300,960, and 6,489,960 and used under license from Pixar.

Portions Copyrighted mental images GmbH 1989-2007.

Portions relating to Python 2.3.3 Copyright © 2001, 2002, 2003 Python Software Foundation; All Rights Reserved.

Portions relating to Imconvert Copyright 1999-2007 ImageMagick Studio LLC, a non-profit organization dedicated to making software imaging solutions freely available.

The following are registered trademarks or trademarks of Autodesk, Inc., in the USA and other countries: 3DEC (design/logo), 3December, 3December.com, 3ds Max, ADI, Alias, Alias (swirl design/logo), AliasStudio, Alias|Wavefront (design/logo), ATC, AUGI, AutoCAD, AutoCAD Learning Assistance, AutoCAD LT, AutoCAD Simulator, AutoCAD SQL Extension, AutoCAD SQL Interface, Autodesk, Autodesk Envision, Autodesk Insight, Autodesk Intent, Autodesk Inventor, Autodesk Map, Autodesk MapGuide, Autodesk Streamline, AutoLISP, AutoSnap, AutoSketch, AutoTrack, Backdraft, Built with ObjectARX (logo), Burn, Buzzsaw, CAICE, Can You Imagine, Character Studio, Cinestream, Civil 3D, Cleaner, Cleaner Central, ClearScale, Colour Warper, Combustion, Communication Specification, Constructware, Content Explorer, Create>what's>Next> (design/logo), Dancing Baby (image), DesignCenter, Design Doctor, Designer's Toolkit, DesignKids, DesignProf, DesignServer, DesignStudio,

DesignStudio (design/logo), Design Web Format, DWF, DWG, DWG (logo), DWG TrueConvert, DWG TrueView, DXF, Ecotect, Exposure, Extending the Design Team, FBX, Filmbox, FMDesktop, Freewheel, GDX Driver, Gmax, Green Building Studio, Heads-up Design, Heidi, HumanIK, IDEA Server, i-drop, ImageModeler, iMOUT, Incinerator, Inventor, Inventor LT, Kaydara, Kaydara (design/logo), Kynapse, Kynogon, LocationLogic, Lustre, Matchmover, Maya, Mechanical Desktop, MotionBuilder, Movimento, Mudbox, NavisWorks, ObjectARX, ObjectDBX, Open Reality, Opticore, Opticore Opus, PolarSnap, PortfolioWall, Powered with Autodesk Technology, Productstream, ProjectPoint, ProMaterials, Reactor, RealDWG, Real-time Roto, REALVIZ, Recognize, Render Queue, Retimer,Reveal, Revit, Showcase, ShowMotion, SketchBook, SteeringWheels, Stitcher, StudioTools, Topobase, Toxik, ViewCube, Visual, Visual Construction, Visual Drainage, Visual Landscape, Visual Survey, Visual Toolbox, Visual LISP, Voice Reality, Volo, Vtour, Wiretap, and WiretapCentral.

The following are registered trademarks or trademarks of Autodesk Canada Co. in the USA and/or Canada and other countries: Backburner, Discreet, Fire, Flame, Flint, Frost, Inferno, Multi-Master Editing, River, Smoke, Sparks, Stone, and Wire.

mental ray is a registered trademark of mental images GmbH licensed for use by Autodesk, Inc. Adobe, Illustrator and Photoshop are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. The Ravix logo is a trademark of Electric Rain, Inc. "Python" and the Python logo are trademarks or registered trademarks of the Python Software Foundation. All other brand names, product names or trademarks belong to their respective holders.

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

Chapter 1	Overview of What's New	1
	What's New in Autodesk Maya	1
Chapter 2	What's New in General	5
Chapter 3	What's New in Performance	9
Chapter 4	What's New in Modeling	11
Chapter 5	What's New in Animation	15
Chapter 6	What's New in Rigging	17
Chapter 7	What's New in Dynamics	21
Chapter 8	What's New in Fluid Effects	23
Chapter 9	What's New in nCloth	25

Chapter 10	What's New in nParticles	27
Chapter 11	What's New in Rendering and Render Setup	31
Chapter 12	What's New In Documentation	39
Chapter 13	What's New in API	41

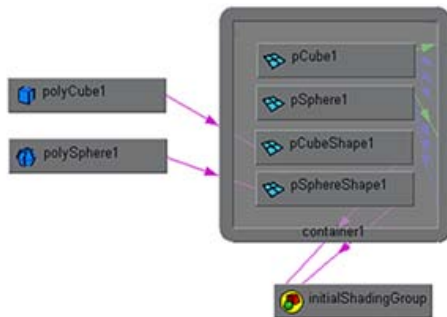
Overview of What's New



What's New in Autodesk Maya

Welcome to What's New in Autodesk® Maya® 2009!

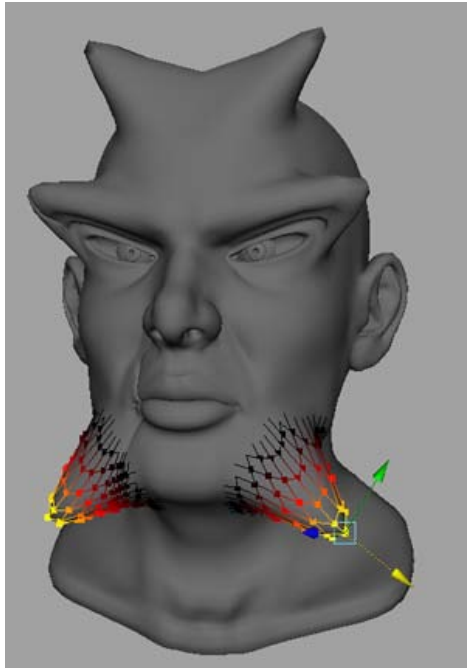
This release is full of features that respond to your needs in games, film, TV, and design. Maya 2009 delivers a host of new features and enhancements that maximize productivity, optimize workflows, and offer new creative possibilities.



General and Performance

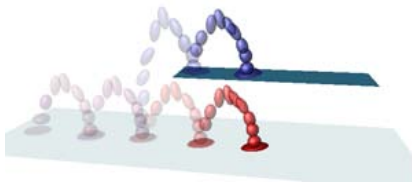
There are many enhancements in this release that help make you more productive, including Maya Assets, Pre-Selection highlighting, Heat Map Display, Transfer Attribute Values and a host of enhancements to the Maya User Interface.

In addition, you'll find significant performance improvements in numerous areas including more multi-threading work and algorithmic speed-ups to boost interactive draw, simulation, and rendering performance for even the heaviest scenes.



Modeling

Polygon modelers and texture artists can be much more efficient, thanks to a wide range of new features and workflow enhancements in Maya 2009, including true soft selection, a tweak mode for rapid modifications, new UV layout and unfolding options, and Merge Vertex.



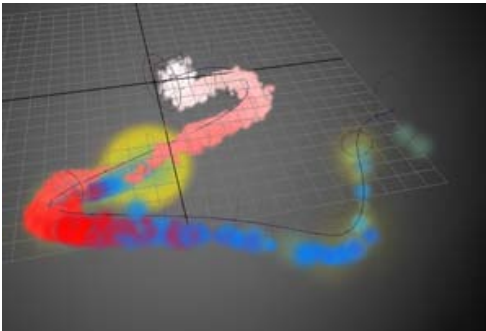
Animation

In Maya 2009, you'll find a powerful new animation layering paradigm built on technology from Autodesk® MotionBuilder® software. This feature allows artists to create multiple layers of animation non-destructively. The flexible architecture works with any attribute; animation layers can be blended, merged, grouped, and re-ordered, and can override or add to preceding layers.



Rigging

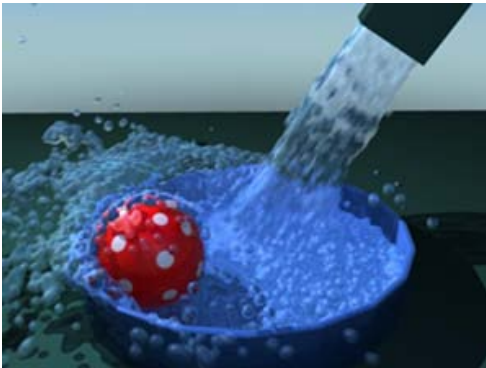
A fully updated Maya Muscle toolset provides realistic skin and muscle behavior with secondary motion, collisions, wrinkles, sliding and stickiness all built-in in Maya 2009.



Dynamics and Effects

In Maya 2009, you'll find new Dynamics features like Volume Axis Curve and Volume Trapping.

This release you can take advantage of nCaching with Fluid Effects, and enhanced smoothing for fluid to polygon mesh conversions.



nDynamics

The Maya Nucleus unified simulation framework gains an innovative new nParticles module, featuring particle to particle collisions, bi-directional interaction with Maya nCloth, and a special method for liquid simulation.

Also, in Maya 2009, you'll find new features for nCloth including Force Field Generation attributes and Stickiness. Find all Maya Nucleus help in the new **nDynamics** guide.



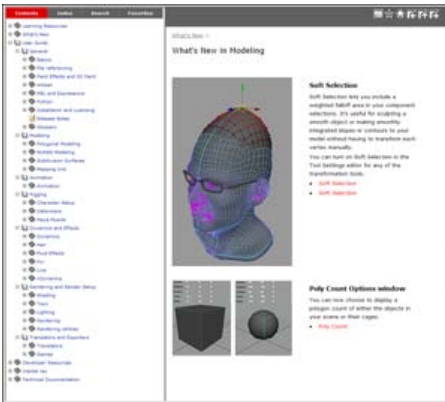
Rendering

In Maya 2009, we've added a completely updated Render Pass feature set that optimizes integration with compositing packages such as Autodesk® Toxik™ software and supports production-level pipelines. And to help studios capitalize on the trend for stereo 3D films, there is a flexible new stereoscopic camera rig complete with in-view-portal stereo viewing.

In addition, you'll find many enhancements to IPR, and new features for mental ray for Maya.

Documentation

The Maya Help is now faster and more easily searchable than ever before. Also, many new tutorials for beginners and advanced users have been added this release.

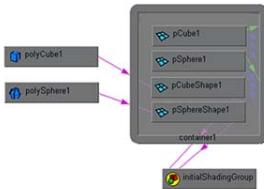


API

The Maya API now includes new classes for Render Passes and User Defined Manipulators, improved documentation and much more.

What's New in General

2



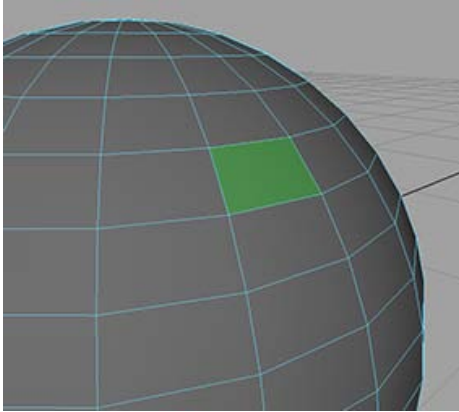
Assets

Assets are a new feature based on the existing container functionality. With assets, you can encapsulate a set of nodes in a container node and publish attributes from these internal nodes to the container. You can flag containers as a certain type and then use those types to limit what assets a specific artist can see using Views.

Assets promote encapsulation of data and black box design.

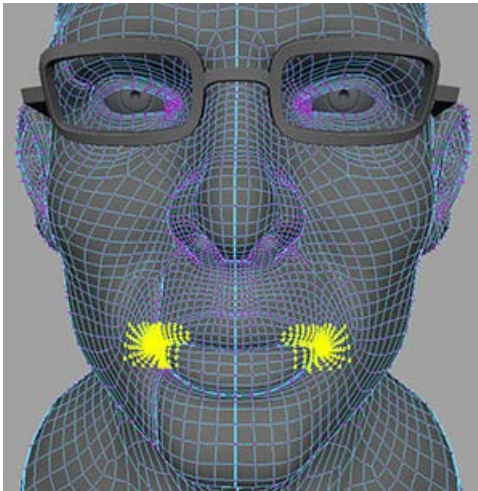
Once you have set up a container, you can save an XML template of its published attributes. Using these container templates, you can pre-plan all assets and their required attributes for your production.

Within a template, you can define multiple views that set up the user interface for the container. Artists can then use an assigned view as they create assets and bind their published attributes to the template.



Preselection Highlight

This feature provides a visual cue about the component that your mouse cursor is pointing to by highlighting the component in a different color. You can turn Preselection Highlight on and off by selecting Window > Settings/Preferences > Preferences and then Preselection Highlight in the Selection section.



Enhanced reflection

Reflection is now supported for the select tool. Additionally, reflection for all tools has been enhanced with additional options to give you more control of the seam.



Heat Map Display

You can now color nodes in the Hypergraph according to their performance in various metrics. This allows you to quickly identify computationally expensive nodes just by looking at the Hypergraph.

Transfer Attribute Values

You can now copy or transfer commonly named attributes between two objects using the Transfer Attribute Values option. You can also use this feature to copy incoming connections and outgoing connections which is useful for copying animation from one object to another.

Panel toolbar

You can now readily access many of the frequently used items that exist within the panel menus by using the panel toolbar. The panel toolbar rests below the panel menu in each view panel. You can toggle view the panel toolbar by pressing Ctrl + Shift + M.

Multi-component selection mode

You can now select faces, vertices and edges without having to change between selection modes. By selecting the new Multi selection mode, you can select components based on the mouse cursor's proximity to them.

Camera-based selection

Camera based selection limits the components you can select to those which are unobstructed from the viewpoint of the current camera. This allows you to streamline selection so that you don't accidentally choose components that are hidden from your view, especially when viewing a scene in shaded mode.

Drag select

Drag select provides you with a new way to select groups components quickly and easily. Like the Paint Select Tool, it allows you to select components by moving mouse along a surface.

Moveable marquee select

You can now reposition a marquee select without redrawing the marquee by holding the Alt key while dragging the marquee with the left mouse button.

Channel Box Enhancements

The Channels menu has now been divided into the Channels and Edit menus. In addition, new filtering options are now available in the Show menu to help you manage the attributes and objects that appear in the Channel Box. You can now filter by attribute types, or create your own filter types.

Transform tool marking menus

You can now access some transform tool specific features from a marking menu by holding Ctrl + Shift + the right mouse button.

Merge connections

You can now condense multiple connections from one node to another in the Hypergraph into a single bolded connection between the two nodes. This can greatly simplify the display.

Archive scene

You can now archive your scene and all its dependencies into a single zip file with the Archive Scene command. This is useful to ensure scenes work correctly when transferred between artists.

Customer Involvement Program

When you start Maya 2009 for the first time, the Customer Involvement Program (CIP) dialog box appears. The CIP collects information to help Autodesk learn how you use Maya. Your participation in the CIP helps design new features and helps improve existing features. You can choose to participate or stop participating at any time by opening the Customer Involvement Program dialog box in the Help menu.

What's New in Performance

3

The following sections describe some performance improvements in Autodesk Maya 2009.

Display and User Interface Components

- Improved display performance for geometry due to additional multithreading.
- Faster tumbling of textured and trimmed surfaces.
- Faster textured display for a large number of objects.
- Faster textured display for a large single object with multiple UV sets.
- Faster tumbling of lit polygons with a single UV set.
- Faster container node creation when the hypergraph window opens.
- Opening the hypergraph window in the presence of a container with a large number of nodes is much faster.

Animation

- Faster ghosting for scenes with rigs.
- Substituting geometry is faster when you are replacing a low resolution model with a high resolution model.

Deformers

- The lattice deformer is now multithreaded.
- Optimized blend shapes.

- Optimized cluster deformers.
- The wire deformer solver is now multithreaded.

Dynamics and Effects

- Improved rewind speed for scenes with multiple nCloth objects.
- Optimized fluids emitter.
- The Fluids surface emitter is now multithreaded.
- Much faster performance of degree 1 NURBS curves in Paint Effects.
- Hair collisions with constraints are faster.
- Rewind performance for scenes with large numbers of nCloth objects has been improved.

Rigging

- Muscle sliding constraints are faster.

Modeling

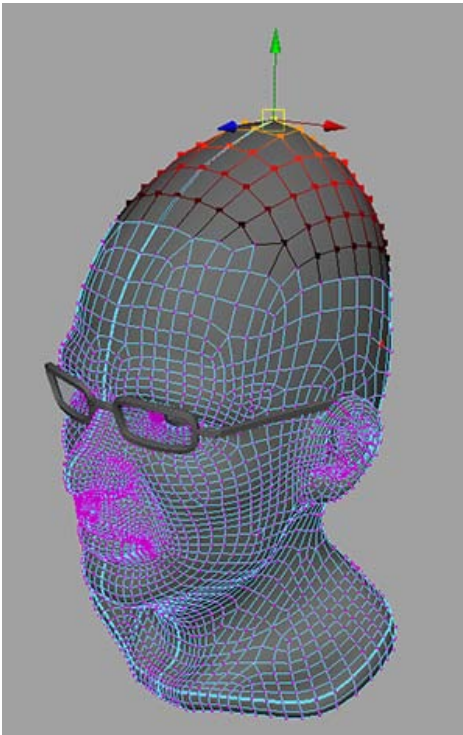
- Polygon UV smoothing is now better multithreaded to improve performance on systems with 4 or more cores.
- Editing operations such as sculpting, soft modification, and Soft Select are now faster when you are using the Smooth Mesh Preview mode.
- Querying your scene for blind data using the polyQueryBlindData command is now much faster, particularly if you are working with large meshes.

General

- Files that contain a dynGlobals particle node load faster.
- Improved performance of large scenes on 64-bit Linux operating system.
- Improved performance in several areas of math-intensive code on 64-bit Windows operating system.
- Saving a file is faster when referencing a file that references a large number of files.

What's New in Modeling

4



Soft Selection

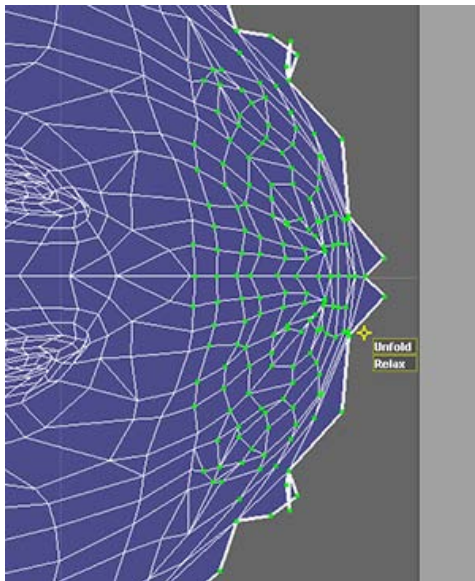
Soft Selection lets you include a weighted falloff area in your component selections. It's useful for sculpting a smooth object or making smoothly integrated slopes or contours to your model without having to transform each vertex manually. You can turn on Soft Selection in the Tool Settings editor for any of the transformation tools.



Merge Vertex Tool

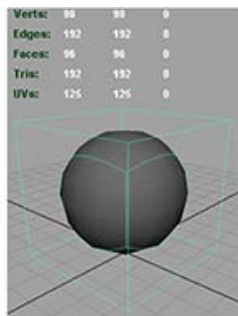
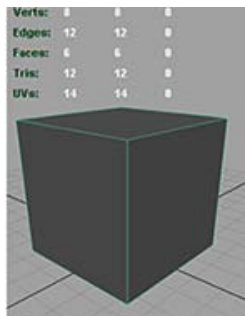
You can now merge vertices quickly with the Merge Vertex Tool by dragging from one vertex to another.

You can access the Merge Vertex Tool by selecting Edit Mesh > Merge Vertex Tool.



Smooth UV Tool

You can now interactively relax or unfold a selection of UVs in the UV Texture Editor with the Smooth UV tool. This allows you to perform these operations by dragging the mouse and offer instant visual feedback so you can judge exactly how much you want to relax or unfold.



Poly Count Options window

You can now choose to display a polygon count of either the objects in your scene or their cages.

Tweak mode

Tweak mode lets you quickly move components under the mouse cursor regardless of whether you are currently using the Select Tool, Move Tool, Rotate Tool or Scale Tool.

You can activate Tweak mode by holding ' on the keyboard and simultaneously dragging a component with the left mouse button.

Enhanced loop selection

You can now select edge loops by double-clicking a single edge or select face and vertex loops by selecting a face or vertex and double-clicking a face or vertex next to it.


Convert polygon edges to curve

You can now convert a path of polygon edges to a NURBS curve by selecting Modify > Convert > Polygon Edges to Curve.

Normal Average component movement

This option lets you select multiple components or objects and move them all along the average of their combined normals.

UV Texture Editor snapping improvements

You can now use snapping in the UV Texture Editor to lock your transformations to existing objects in the scene. You can also access a number of options related to pixel snapping by selecting Image > Pixel Snap >  in the UV Texture Editor.

You can also use Discrete Rotate and Discrete Scale to rotate and scale UVs in specific increments.

UV Layout enhanced options

A number of new options have been added to the UV Layout Options window, which allow you to adjust multiple UV shells in a single UV space more easily. These options also allow you to offset your UV shells and scale them in the UV space.

Preserve textures during transformation

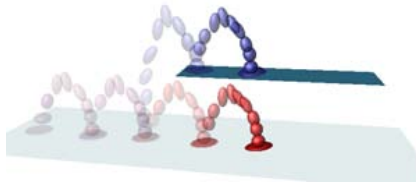
You can now transform components of a mesh without warping its texture. By turning on the Preserve UVs option in the Tool Settings editor for any of the transformation tools, Maya will automatically perform the appropriate transformations to the UV shell to maintain the overall look of the texture.

Bevel inside curves

You can now bevel within the bounds of a curve by turning on the Bevel inside curves option in the Bevel Plus Options window. Beveling inside a curve gives you more precise control over the overall size of a bevelled surface.

What's New in Animation

5



Animation layers

There is now a third editor in the Channel Box that lets you work with layers of animation. The Animation Layer Editor gives you all of the tools and options you need to create and manipulate animation layers. Animation layers let you create and blend multiple levels of animation in a scene. You can create layers to organize new keyframe animation, or to keyframe on top of existing animation without overwriting the original curves.

Bake Simulation options

New options for animation layers have been added to the Bake Simulation Options. You can use these options, such as Smart Bake, to specify how you want to bake animation layers.

What's New in Rigging

6



Image by Alan Wilson

Maya Muscle

Maya now includes the powerful Maya Muscle skin deformer, which provides you with a complete set of tools for muscle rigging and deformation effects.

The following sections summarize the new features provided by Maya Muscle:

Muscle creation

Maya Muscle provides you with streamlined muscle creation workflows using the Muscle Builder or Muscle Creator. These tools let you quickly and intuitively create, sculpt, and orient muscles in order to create convincing character rigs. Using the Muscle Creator window, you can create muscles by growing them out to a surface, mirror muscles from one side of a character to the other, and control the muscle shape at different phases of contraction.

Skin deformation

The main Muscle deformer lets you set up muscle and skin deformation, but it can also be used as a standalone deformer. To use the Muscle deformer as a skin solution for Sticky or Sliding weights, you can quickly apply default weights, then paint

weights on the mesh when you need a higher level of detail. You can also mirror weights from one side of a symmetrical mesh to the other.

Re-use an existing skin set up

If you already have a rig set up with skin clusters, you can use Maya Muscle to add detail without re-creating your existing skin to joint weighting. Muscle's Relative Sticky mode lets you use your existing skin cluster setup and focus your efforts only on areas that benefit from additional muscle and skin detail.

Other deformation effects


Along with the main muscle/skin deformation tools, Maya Muscle provides several standalone deformers that can be used for any number of deformation effects. For example:

- Simulate effects based on natural forces like wind and gravity using the Force deformer.
- Create high quality, detailed displacement and sliding effects based on curves or a 2D image map using the displacement options.
- Set up jiggle effects directly on each point of your skin mesh.
- Apply Relax and Wrinkle effects to a mesh.

Skin Collision

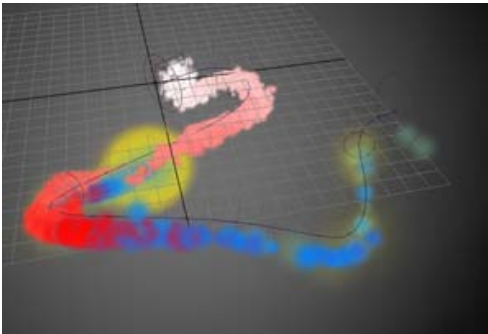
Several skin collision features let you simulate realistic skin collision effects. Smart Collision, Self Collision, and Multi-Collision give you options for controlling muscle and skin shapes when collisions occur.

Wrap deformer:Auto Weight Threshold option

A new wrap deformer option (Create Deformers > Wrap > ) automatically sets the optimal weight of the wrap influence objects' shapes.

What's New in Dynamics

7



Volume Axis Curve

With Fields, you can now define an axis field region around a NURBS curve by creating a volume axis curve. Volume axis curves let you move particles and nParticles in various directions along the curve. When working with volume axis curves, you can now create a ramp to scale curve section radius along the curve's axis.

Volume Trapping

Fields now include volume trapping attributes, which allow objects to be trapped inside volume fields. Volume trapping works with all volume shapes including cubes, spheres, cylinders, and volume axis curves.

Axial Magnitude

You can now create a ramp to scale field Magnitude along the axis of a volume shape field or volume axis curves.

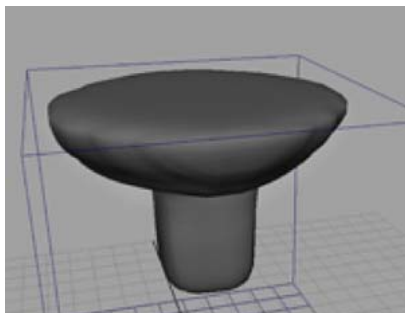
What's New in Fluid Effects

8

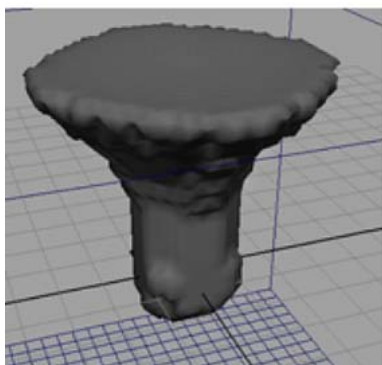


nCaching fluids

Fluid caching is now done through nCaching, the Maya caching system used to cache nCloth simulation data. nCaching fluids improves caching performance, as well as uses a more efficient storage method for your cached files. As with the previous fluids caching, you can specify which fluid properties are saved in the cache file.



Fluid to Polygon Conversion
with Output Mesh set



Fluid to Polygon Conversion
without Output Mesh set

Improved fluid to polygon mesh conversions

The quality and speed of the fluid output mesh (that results from fluid to polygon conversion), as well as the interactive display of surface fluids have been greatly improved. New Output Mesh attributes allow you can control the resolution, smoothness, and speed of your fluid to polygon mesh conversions.

What's New in nCloth

9



nCloth force fields

nCloth now includes Force Field Generation attributes. Force fields generated by nCloth can repel and attract nParticle objects and other nCloth objects. Force Fields are defined by the magnitude of the force field, as well as by the distance from the field generating object that the field is active. Passive collision objects can also generate force fields.

Stickiness

nCloth now includes a Stickiness attribute. Using Stickiness, you can adjust the tendency of your nCloth object to stick to other Nucleus objects (nCloth, nParticles, and passive objects) when they collide. Passive collision and nParticle objects also have Stickiness.

Convert nCloth output space

You can now convert the space of your nCloth output mesh to either world space or local space, depending on whether your

nCloth was created for Local Space Output
or World Space Output.

What's New in nParticles

10

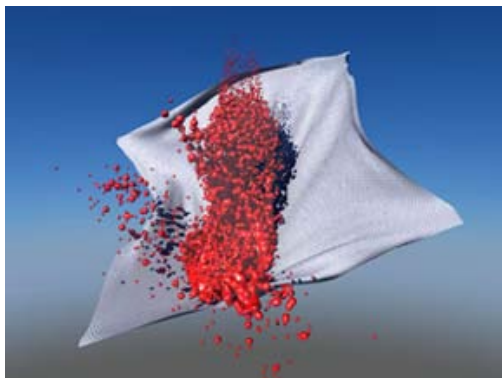


Image by Dan Pressman

New particle generation system

There is a new particle generation system called Maya® nParticles in Maya 2009. nParticles use Maya® Nucleus™, which is the same dynamic simulation framework that generates nCloth simulations. As part of the Nucleus system, nParticles interact and collide with nCloth and passive collision objects, as well as with other nParticle objects. You can use nParticles with Nucleus-based nConstraints to create particle effects and dynamic simulations that cannot be achieved with Maya classic particles. Like other Nucleus objects, nParticle objects are assigned to a Nucleus solver which calculates the nParticle simulation in an iterative manner.

In addition to Nucleus-based dynamics, nParticles can be used in place of Maya classic particles for particle goals, geometry instancing, and sprite effects. nParticles can also be manipulated by external non-Nucleus forces, including gravity and wind.



nParticle collisions

When nParticles objects are created, they are automatically capable of colliding with nCloth objects, passive objects, and other nParticle objects. With collisions between nParticles and nCloth, you can use the force of emitted nParticles to drive nCloth deformations. For example, you can create an nParticle rain effect with droplets that fall onto and deform the material of an nCloth umbrella. nParticles can self-collide, meaning that nParticles generated from the same nParticle object can collide and interact with each other. nParticle collisions can only occur between nParticles and nCloth or passive objects that are members of the same Maya Nucleus system.

nParticle constraints

You can use nConstraints to control the behavior of individual or emitted nParticles by restricting the particle's movement or by attaching the nParticle object to other Nucleus objects. For example, you can constrain nParticles to an animated nCloth or passive object so that particle splatter, fire, or smoke effects move with the object. nParticles can be constrained using **Transform**, **Component to Component**, **Point to Surface**, **Slide on Surface**, and **Force Field** constraints.



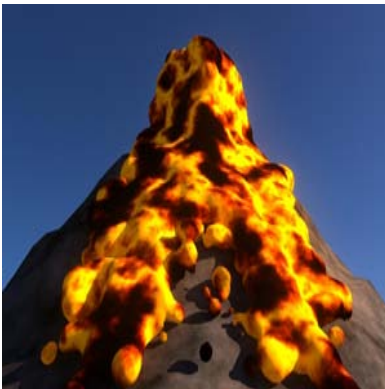
Internal per particle attribute ramps

nParticles objects have internal per particle attribute ramps created on the nParticle-Shape node. The internal ramps simplify the workflow of setting per particle radius, mass, color, opacity, and incandescence. nParticle internal ramps work the same way as Maya texture ramps, but provide additional control for setting ranges of input values and randomized multipliers for output values.

The internal ramp's per particle attributes can be deleted from the nParticle object when you want to set per particle attributes using expressions.

Generate force fields with nParticles

nParticles provide Force Field Generation attributes that allow you to generate force fields that repel and attract nCloth objects and other nParticle objects. nParticle force fields are defined by the magnitude of the force field as well as distance, which is the distance from the field generating object that the field is active.



Liquid simulation with nParticles

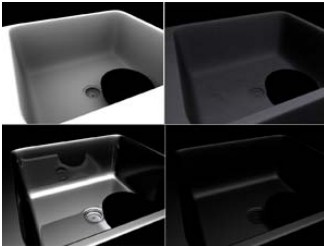
nParticles provide Liquid Simulation attributes that allow you to create nParticle objects that look and behave like liquids. Control over liquid properties, such as viscosity, allow you to create effects that range from fast moving waterfalls to slow rolling lava flows.



nParticle to polygon mesh conversion

Static and emitted Bloby Surface nParticles can be converted to polygon meshes. nParticle output meshes can be edited and manipulated like any other polygon. Before the conversion, you can use the Output Mesh attributes to set the quality of resulting polygon mesh. nParticle output meshes are useful for creating smooth flowing surfaces for realistic liquid simulation effects.

What's New in Rendering and Render Setup



Multi-render passes for mental ray for Maya

The new multi-render pass feature provides an easy workflow for configuring render passes. You can render an unlimited number of render passes, and you can group them into render pass sets. You can also select a subset of the objects or lights in your scene to contribute to each render pass. This subset is called a render pass contribution map.

Using the multi-render pass feature, you can reduce the need to use render layers, thus reducing compute times for scene translation and actual rendering. If you work with complex multi-layered compositions, rendering may also be several times faster. Multi-render passes also allow you perform scene segmentation at render time.


Setting up your scene to use multi-render passes is simple.

Follow these steps:

- Use the Create Render Passes window to create your render passes.
- Use the Passes tab under the Render Settings window to create and manage the render passes for each layer.
- Use the renderPass Attribute Editor to set the options for each render pass.

For advanced users, you can also:

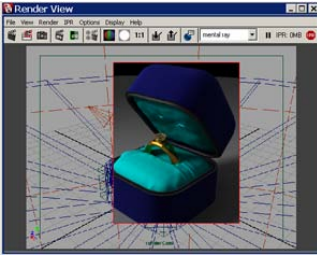
- Use either the Render Layer Editor or the Passes tab to create your render pass contribution maps, then use the Passes tab to manage the render passes for each map.
- Group your render passes into render pass sets. Use the Passes tab to create a render pass set and the Relationship Editor (select Window >

Relationship Editors or click  in the Passes tab) to manage its membership.



mental ray Render Settings re-organization

The mental ray tab in the Render Settings window is now divided into 5 mental ray tabs, including: Passes, Features, Quality, Indirect Lighting, and Options, providing better organization for the mental ray controls. For example, if you need to tweak your final gather settings, you can simply go to the Indirect Lighting tab.



IPR improvements for mental ray for Maya

This section lists the IPR improvements for mental ray for Maya:

Light manipulation

Light manipulation improvements for mental ray include:

- Area light changes: switching from a Maya to mental ray area light, shape type changes, changing of High Samples and Low Samples
- Deletion of lights
- Changes to light photon attributes such as energy and color
- Changes to photon attributes for the shaders and lights when caustics, global illumination or final gather is used
- Changes to shadow map settings, including switching from raytraced shadows to shadow maps and back, as well as shadow map format and file changes
- Duplicating a light or object by copying or duplicating a light or object as an instance
- Hiding and unhiding of light source instances
- Light linking and shadow linking
- Rotation of the IBL node.

Object manipulation

Object manipulation improvements for mental ray include:

- Interactive creation of objects
- Duplicating a light or object by copying or duplicating a light or object as an instance

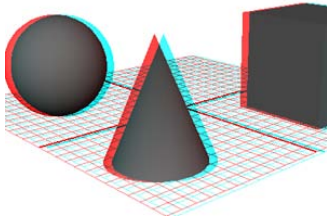
- Adding or removing a displacement shader and tuning of the displacement map shading network
- Adding and deleting or topology changes to geometry from the scene
- Changes to the Render Stats and mental ray render stats (under the object's shape node). Previously, only changes to the instance render stats (under the object's transform node) were supported.

Options

Options improvements for mental ray include:

- Adding or deleting and tuning of a mental ray approximation node
- Creation of a camera and render camera change.
- Changing camera settings such as the angle of view
- Enabling and tuning the Depth of Field option for the camera node

You can also set IPR specific options such as the verbosity level for error messages by selecting Render > IPR Render Current Frame > .



Stereoscopic camera

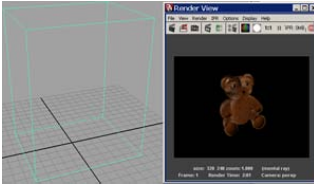
You can now add a stereoscopic camera to your scene. This allows you to create depth of field with the illusion of a three dimensional interface. Choose between various stereo modes such as anaglyph, checkerboard and horizontal interlace. You can render from the stereo camera and view the render output in any of these stereo modes.

For advanced users, you can also customize your stereoscopic camera by adding a custom stereo rig to your scene.



Elliptical filtering

Use elliptical filtering to perform high quality texture filtering and anti-aliasing when rendering with mental ray for Maya. Instead of using point sampling, elliptical filtering uses an area (an ellipse) to perform an image lookup. This ellipse contains many pixels, and the pixels are averaged and the average color is returned as a result. Elliptical filtering in Maya is very simple to use. You can access its controls through the Attribute Editor of your file node.



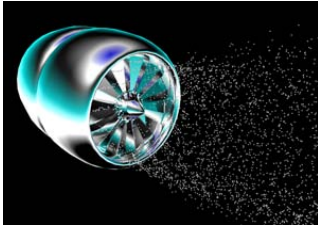
Render proxies

Use render proxies with mental ray rendering to manage large scenes with complex geometry. Export your complex object as a mental ray assembly file, then replace it in your scene with a placeholder object that references this file. When you render, the exported object is loaded into memory and rendered with the rest of your scene. Translation time and memory usage are cut down, allowing mental ray for Maya to render large scenes.



Smooth mesh render

Previously, you could preview a smooth polygon mesh in the scene view by pressing 3. Now you can also render the smooth preview in mental ray for Maya. You can choose to use the same level of smoothing for the 3D viewport preview as for your render. Alternatively, you can also use a different division level for the two.



Particle rendering in mental ray for Maya

All attributes in the ParticleInfoSampler node are now supported when rendering particles in mental ray for Maya.

new mental ray Render Settings attributes

new mental ray 3.6 and 3.7 attributes have been added to the Render Settings, including: Importons, Irradiance Particles, Ambient Occlusion and Merge Distance for photons, caustics, and global illumination.

A Render Mode attribute has been added that allows you to render only final gather, or shadow maps, or light maps. In addition, flags that enable you to globally disable light maps and lens shaders have been added. The new BSP2 acceleration method is also available.

Final gathering now accepts multiple final gather map files. New rasterizer controls include: Rasterizer Transparency, Rasterizer Pixel Samples, and Rasterizer use Opacity. New motion blur controls include: Displace Motion Factor and Force Motion Vector Computation.

Render Settings window: Common tab re-organization

The Common tab in the Render Settings window has been re-organized so that there is a clear distinction between the frame and file attributes.

The Frame Buffer Naming and Custom Naming String attributes have been added so that if you are using the OpenEXR file format with the multi-render pass feature, you can customize the naming of the channels in your OpenEXR file.

Texture baking using multiple threads

You can use the Bake Optimization feature to select between baking using one thread per object or baking using multiple threads per object. The former performs better for baking multiple objects and the latter for baking a single complex piece of geometry.

What's New In Documentation

12

New Advanced Tutorials

In Maya 2009 we've introduced Advanced Techniques. These tutorials are designed for intermediate and advanced users and concentrate on specific, complex topics and techniques.

New Getting Started Unlimited Tutorials

This release you'll find new nParticle tutorials to introduce you to the latest module built on the Maya® Nucleus™ unified simulation framework. nParticles can be used to simulate a wide range of effects including liquids, clouds, smoke, spray, dust.

Improved Maya Help system

The Maya Help is now faster and more easily searchable than ever before. New features include several new search methods, search highlighting, and the ability to save favorite pages for easy future reference.

Improved Access to FBX Help

In Maya 2009 you can now open the Autodesk® FBX® Help from the Maya Help menu. Click Help > Autodesk FBX Help to open the FBX documentation.

Maya Release Notes now online

You can now find the Maya Release Notes for this release online. For more information, see:

- <http://www.autodesk.com/maya-release-notes>

What's New in API

I 3

The changes to the API are described in the following sections:

- [General Updates](#) on page 41
- [New classes](#) on page 42
- [New methods](#) on page 44
- [Modified methods](#) on page 51
- [New enums](#) on page 53
- [Modified enums](#) on page 54
- [Changes to examples](#) on page 56
- [Bug fixes](#) on page 57

General Updates

Compiler information

- Linux 64: gcc 4.1.2
- Mac OS C Intel: Xcode 2.4.1
- Windows 32|64: VC8 with SP1 and Template hotfix

Const changes

A number of methods which do not modify their objects have been changed to be const.

A number of method parameters which are not modified by the method have been changed to be const.

Reference documentation

API reference documentation for Maya now has improved styles and new functions to provide more useful and readable documentation, including:

- A module list with a collaboration diagram for each set of classes
- An enhanced class index that includes
 - An alphabetic listing of classes
 - A class list with brief descriptions
 - A class hierarchy
 - A graphical class hierarchy to illustrate inheritance
 - A class members list with hyperlinks to corresponding methods
- A list of deprecated classes and methods
- A list of examples with cross-references within the source code

New classes

We have added a number of new classes to this release.

MAtomic

The `MAtomic` class provides cross-platform atomic operations, such as incrementing and decrementing counters, which are useful when writing multi-threaded code.

MContainerMessage

The `MContainerMessage` class provides the ability to register callback functions for changes in the published attributes of a container.

MFnContainerNode

The `MFnContainerNode` class is a function set for creating, querying and editing containers.

MFnRenderPass

The `MFnRenderPass` class is a function set for working with render passes.

MFnUInt64SingleIndexedComponent

The `MFnUInt64SingleIndexedComponent` class is a function set for working with components which are indexed by an `MUInt64` (64-bit unsigned integer), such as subdivision surface vertex ids.

MMeshSmoothOptions

The `MMeshSmoothOptions` class is used to set and retrieve the options to be used with the `MFnMesh::generateSmoothMesh` method.

MNurbsIntersector

The `MNurbsIntersector` class provides methods for efficiently finding the closest point to a NURBS surface.

MPlane

The `MPlane` class is a utility class which describes a mathematical plane.

MPxCacheFormat

The `MPxCacheFormat` is used to implement new cache file formats in Maya.

MPxManipulatorNode

The `MPxManipulatorNode` class is used to create user-defined manipulators which can be used standalone or with `MPxManipContainer`.

MPxRenderPassImpl

The `MPxRenderPassImpl` class is used to create custom render passes.

MRenderPassDef

The `MRenderPassDef` class provides the ability to create, query and edit a render pass.

MRenderPassRegistry

The `MRenderPassRegistry` class is used to register render pass definitions with Maya and to retrieve already-registered render pass definitions.

MRichSelection

The `MRichSelection` class provides a selection list which retains soft selection and symmetry information.

MThreadUtils

The `MThreadUtils` class provides utility methods which are useful when writing threaded plug-ins.

MWeight

The `MWeight` class is used to access soft select and symmetric selection weights.

New methods

We have added the following new methods to this release:

M3dView

```
void getScreenPosition( int& x, int& y, MStatus* ReturnStatus = NULL ) const
```

Returns the position of the view's upper-left corner, in screen coordinates.

```
MStatus colorMask( bool& r, bool& g, bool& b, bool& a )
```

Retrieves the current color mask.

```
MStatus setColorMask( bool r, bool g, bool b, bool a )
```

Sets the color mask.

MDGMessage

```
static MCallbackId addPreConnectionCallback( MMessage::MPlugFunction func,  
void* clientData = NULL, MStatus* ReturnStatus = NULL )
```

Adds a callback which is called just prior to a connection being made or break.

MDrawInfo

```
bool inUserInteraction() const
```

Returns true if the current refresh is the result of the user interacting with the scene.

```
bool userChangingViewContext() const
```

Returns true if the current refresh is the result of the user manipulating view context tools such as tumble, dolly, track, etc. This is useful for changing drawing mode to something simpler to speed up interactive redraws.

MFileIO

```
static MPxFileTranslator* beforeOpenUserFileTranslator( MStatus* ReturnStatus  
= NULL )
```

Can be used within `MSceneMessage::kBeforeOpen` and `MSceneMessage::kBeforeOpenCheck` callbacks to return the plug-in file translator being used, if any.

```
static MPxFileTranslator* beforeImportUserFileTranslator( MStatus* ReturnStatus  
= NULL )
```

Can be used within `MSceneMessage::kBeforeImport` and `MSceneMessage::kBeforeImportCheck` callbacks to return the plug-in file translator being used, if any.

```
static MPxFileTranslator* beforeSaveUserFileTranslator( MStatus* ReturnStatus  
= NULL )
```

Can be used within `MSceneMessage::kBeforeSave` and `MSceneMessage::kBeforeSaveCheck` callbacks to return the plug-in file translator being used, if any.

```
static MPxFileTranslator* beforeExportUserFileTranslator( MStatus* ReturnStatus  
= NULL )
```

Can be used within `MSceneMessage::kBeforeExport` and `MSceneMessage::kBeforeExportCheck` callbacks to return the plug-in file translator being used, if any.

```
static MPxFileTranslator* beforeReferenceUserFileTranslator( MStatus* Return  
Status = NULL )
```

Can be used within `MSceneMessage::kBeforeReference` and `MSceneMessage::kBeforeReferenceCheck` callbacks to return the plug-in file translator being used, if any.

MFileObject

```
void setResolveMethod( MFileResolveMethod method )
```

Sets how file path resolution should be done.

```
MFileResolveMethod resolveMethod() const
```

Retrieves the method being used to resolve file paths.

MFncamera

```
MStatus getViewParameters( double windowAspect, double& apertureX, double& apertureY, double& offsetX, double& offsetY, bool applyOverScan = false, bool applySqueeze = false ) const
```

Returns the raw viewing parameters which MFncamera::getViewingFrustum and MFncamera::getRenderingFrustum use internally in their calculations.

```
MStatus setDisplayGateMask( bool displayGateMask )
```

If the film or resolution gate is active, this method can be used to set whether the masked portion outside the gate is shaded to make it more visible.

MFncomponent

```
bool hasWeights() const
```

Returns true if the component has soft selection or symmetry weight data attached.

```
MWeight weight( int index, MStatus* ReturnStatus = NULL ) const
```

Returns the soft selection or symmetry weight data for a given element within the component.

MFncDependencyNode

```
MStatus dgTimerOn()
```

Turns the node's dependency graph timer on. DG timers are used to collect timing information for performance analysis.

```
MStatus dgTimerOff()
```

Turns the node's dependency graph timer off. DG timers are used to collect timing information for performance analysis.

```
MdgTimerState dgTimerQueryState( MStatus* ReturnStatus = NULL )
```

Returns the current state of the node's dependency graph timer. DG timers are used to collect timing information for performance analysis.

```
MStatus dgTimerReset()
```

Resets the node's dependency graph timer and counters to zero. DG timers are used to collect timing information for performance analysis.

```
double dgTimer( const MdgTimerMetric timerMetric, const MdgTimerType timerType, MStatus* ReturnStatus = NULL ) const
```

Returns the specified timer value for the node. DG timers are used to collect timing information for performance analysis.

```
MStatus dgCallbacks( const MdgTimerType type, MStringArray& callbackName,  
MDoubleArray& value )
```

Returns how much of the time logged against a node's DG timer was spent in each type of callback. DG timers are used to collect timing information for performance analysis.

```
MStatus dgCallbackIds( const MdgTimerType type, const MString& callbackName,  
MCallbackIdArray& callbackId, MDoubleArray& value )
```

Returns how much of the time logged against a node's DG timer was spent in each callback method for a given type of callback. DG timers are used to collect timing information for performance analysis.

```
MString pluginName( MStatus* ReturnStatus = NULL ) const
```

If the node is from a plug-in, this method returns the plug-ins pathname.

MFnMesh

```
MStatus getSmoothMeshDisplayOptions( MMeshSmoothOptions& options ) const
```

Retrieves the current display smoothing options for the mesh.

```
MStatus setSmoothMeshDisplayOptions( const MMeshSmoothOptions& options )
```

Sets the current display smoothing options for the mesh.

```
MStatus booleanOp( BoolOperation op, MFnMesh& mesh1, MFnMesh& mesh2 )
```

Computes a boolean operation between two meshes.

```
const float* getRawPoints( MStatus* ) const
```

Returns a pointer to the internal vertex list for this mesh.

```
const float* getRawNormals( MStatus* ) const
```

Returns a pointer to the internal normal list for this mesh.

```
MStatus getVertexNormal( int vertexId, bool angleWeighted, MVector& normal,  
MSpace::Space space = MSpace::kObject ) const
```

This variant of `getVertexNormal` allows the caller to turn off the weighted averaging Maya normally uses when calculating normals in favor of a simple average which is much faster.

```
MStatus getVertexNormals( bool angleWeighted, MFloatVectorArray& normals,  
MSpace::Space space ) const
```

This variant of `getVertexNormals` allows the caller to turn off the weighted averaging Maya normally uses when calculating normals in favor of a simple average which is much faster.

```
MStatus getColorSetFamilyNames( MStringArray& familyNames ) const
```

Get the names of all of the color set families on this object.

```
MStatus getColorSetsInFamily( const MString& familyName, MStringArray& setNames  
    ) const
```

Get the names of the color sets that belong to a color set family.

```
bool isColorSetPerInstance( const MString& name, MStatus* ReturnStatus ) const
```

Returns true if the specified color set is per-instance and false if it is shared across all instances.

```
MStatus getAssociatedColorSetInstances( const MString& setName, MIntArray&  
    instances ) const
```

Returns the instance numbers of all instances which use a given color set.

MFnNurbsSurface

```
MObject getDataObject() const
```

Returns an `MObject` if the class has been constructed with an `MFn::kNurbsSurfaceData` entity, otherwise `MObject::kNullObj` is returned.

MFnPlugin

```
MStatus registerCacheFormat( const MString& cacheFormatName, MCreatorFunction  
    creatorFunction )
```

Register a custom cache format with Maya.

```
MStatus deregisterCacheFormat( const MString& cacheFormatName )
```

De-register a custom cache format with Maya.

```
MStatus registerRenderPassImpl( const MString& passImplId, MRenderPassDef*  
    passDef, MCreatorFunction creatorFunction, bool overload )
```

Register a custom render pass implementation with Maya.

```
MStatus deregisterRenderPassImpl( const MString& passImplId )
```

De-register a custom render pass implementation with Maya.

MFnRenderLayer

```
MStatus externalRenderPasses( MObjectArray& renderPassArray ) const
```

Returns an array of all the render pass nodes in a render layer.

```
bool passHasObject( const MObject& renderPass, const MDagPath& objectPath,  
MStatus* ReturnStatus ) const
```

Returns true if the specified object instance contributes to the given render pass.

```
bool passHasLight( const MObject& renderPass, const MObject& light, MStatus*  
ReturnStatus ) const
```

Returns true if the specified light contributes to the given render pass.

MGlobal

```
static MStatus getRichSelection( MRichSelection& dest, bool defaultToActiveSelec  
tion )
```

Returns the active selection with any soft selection and symmetry applied.

```
static SelectionMethod selectionMethod( MStatus* ReturnStatus )
```

Returns the selection method that should be used in the currently active viewport.

MItGeometry

```
MWeight weight( MStatus* ReturnStatus ) const
```

Returns the weight of the current component.

```
int exactCount( MStatus* ReturnStatus )
```

Returns the exact number of items in this iteration.

```
MStatus allPositions( MPointArray& points, MSpace::Space space ) const
```

Returns the position of all the points/CVs/vertices. This operation is faster than using the iterator to get values one by one.

```
MStatus setAllPositions( const MPointArray& points, MSpace::Space space )
```

Sets the position of all the points/CVs/vertices at once. This operation is faster than using the iterator to set values one by one.

MItSelectionList

```
MStatus getPlug( MPlug& plug )
```

Retrieves the current selection item as a plug.

MObjectHandle

```
unsigned int hashCode() const
```

Returns a hash code for the internal Maya object referenced by the `MObject` within this `MObjectHandle`. This provides a way for safely using internal Maya objects as keys in hash-based lookups, such as Python dictionaries.

MPxGIBuffer

```
virtual void beginBufferNotify()
```

This method is called when the GL buffer is being setup by the viewport renderer.

```
virtual void endBufferNotify()
```

This method is called when the GL buffer is being shutdown by the viewport renderer.

MPxManipContainer

```
MStatus addMPxManipulatorNode( const MString& manipTypeName, const MString& manipName, MPxManipulatorNode*& proxyManip )
```

This method creates a custom `MPxManipulatorNode` and adds it to the manip container.

MPxModelEditorCommand

```
MStatus setResult( const MDoubleArray& result )
```

This method should be called when the result of the panel command is a double array.

```
MStatus setResult( const MIntArray& result )
```

This method should be called when the result of the panel command is an integer array.

```
virtual MString editorCommandName() const
```

Returns the name of the model editor command.

```
virtual MPx3dModelView* makeModelView( MStatus* ReturnStatus )
```

This method is called when the model editor is being created and it can be overridden to allow the user to handle the allocation of the model view directly or alter the creation according to command flags.

MRenderUtil

```
static MString mainBeautyPassName()
```

Returns the name of the main beauty pass for use by renderers for token substitution.

MSceneMessage

```
static MCallbackId addCallback( Message, void (*func)( const MStringArray&,
void* ), void* clientData, MStatus* ReturnStatus )
```

This version of the method takes a callback function which takes an array of strings. This is provided to support the new plugin load/unload messages.

Modified methods

Changes were made to a number of existing methods. In those cases where the change has made the new calling sequence incompatible with the old, the old calling sequence has been retained as well for backward compatibility, but is marked as deprecated in the documentation.

For brevity, those methods that only had changes in their constness or the constness of their parameters are not listed.

MColor

A number of methods have been inlined to improve performance.

MFileIO

```
MStatus open( const MString& fileName, const char* type = NULL, bool force =
false, ReferenceMode refMode = kLoadDefault, bool ignoreVersion = false )
```

The optional `ignoreVersion` parameter has been added to indicate whether Maya should ignore the version number when opening the file.

MFnMesh

```
MStatus createColorSet( MString& colorSetName, MDGModifier* modifier, bool
clamped, MColorRepresentation rep, const MUIntArray* instances = NULL )
```

The optional `instances` parameter has been added to allow the color set to be only associated with specific instances of the node.

```
MString createColorSetWithName( const MString& colorSetName, MDGModifier*
modifier = NULL, const MUIntArray* instances = NULL, MStatus* ReturnStatus =
NULL )
```

The optional `instances` parameter has been added to allow the color set to be only associated with specific instances of the node.

```
MString currentColorSetName( int instance = kMFnMeshInstanceUnspecified,
MStatus* ReturnStatus = NULL ) const
```

The optional `instance` parameter has been added to return the instance-specific color set, if there is one.

```
MObject generateSmoothMesh( MObject parentOrOwner = MObject::kNullObj,
    MMeshSmoothOptions* options = NULL, MStatus* ReturnStatus = NULL )
```

The optional `options` parameter has been added to allow the caller to set the options used in generating the smooth mesh.

```
MStatus getCurrentColorSetName( MString& setName, int instance = kMFnMeshIn
    stanceUnspecified )
```

The optional `instance` parameter has been added to return the instance-specific color set, if there is one.

MGlobal

```
MStatus selectFromScreen( const short& x_pos, const short& y_pos, MGlobal::Lis
    tAdjustment listAdjustment = kAddToList, MGlobal::SelectionMethod selectMethod
    = kWireframeSelectMethod )
```

The optional `selectMethod` parameter has been added to specify whether objects should be selected as wireframes or as surfaces.

```
MStatus selectFromScreen( const short& start_x, const short& start_y, const
    short& end_x, const short& end_y, MGlobal::ListAdjustment listAdjustment =
    kAddToList, MGlobal::SelectionMethod selectMethod = kWireframeSelectMethod )
```

The optional `selectMethod` parameter has been added to specify whether objects should be selected as wireframes or as surfaces.

MProgressWindow

An optional `ReturnStatus` parameter has been added to the following methods:

- `static int progressMin(MStatus* ReturnStatus = NULL)`
- `static int progressMax(MStatus* ReturnStatus = NULL)`
- `static int progress(MStatus* ReturnStatus = NULL)`
- `static MString title(MStatus* ReturnStatus = NULL)`
- `static MString progressStatus(MStatus* ReturnStatus = NULL)`
- `static bool isInterruptable(MStatus* ReturnStatus = NULL)`
- `static bool isCancelled(MStatus* ReturnStatus = NULL)`

MPxConstraintCommand

```
virtual MStatus handleNewTargets( MDagPath& dagObject )
```

The `dagObject` parameter has been changed from an `MObject` to an `MDagPath` so that it can distinguish between different instances of the same node.

```
virtual MStatus connectTarget( MDagPath& targetPath, int index )
```

Previously the target was passed as an opaque pointer. This has been changed to an `MDagPath`, providing the developer with more flexibility when overriding this virtual method.

```
MStatus connectTargetAttribute( MDagPath& targetPath, int index, MObject&
tarAttr, MObject& constraintAttr, bool instanced = false )
```

Previously the target was passed as an opaque pointer. This has been changed to an `MDagPath`, providing the developer with more flexibility when overriding this virtual method.

An optional `instanced` parameter has been added to indicate if the target attribute is instanced.

New enums

We have added a number of new enums to existing classes in this release:

- `MFileObject::MFileResolveMethod` { `kNone`, `kExact`, `kDirMap`, `kReferenceMappings`, `kRelative`, `kBaseName`, `kInputFile`, `kInputReference`, `kStrict` }

This enum specifies how to resolve file paths.

- `MFnDependencyNode::MdgTimerState` { `kTimerOff`, `kTimerOn`, `kTimerUninitialized`, `kTimerInvalidState` }

Node timer states.

- `MFnDependencyNode::MdgTimerMetric` { `kTimerMetric_callback`, `kTimerMetric_compute`, `kTimerMetric_dirty`, `kTimerMetric_draw`, `kdgTimerMetric_fetch`, `kTimerMetric_callbackViaAPI`, `kTimerMetric_callbackNotViaAPI`, `kTimerMetric_computeDuringCallback`, `kTimerMetric_computeNotDuringCallback` }

The different node timer metrics which can be queried.

- `MFnDependencyNode::MdgTimerType` { `kTimerType_self`, `kTimerType_inclusive`, `kTimerType_count` }

Types of node timers available.

- `MFnMesh::BoolOperation` { `kIntersection`, `kDifference`, `kUnion` }

The types of boolean operations which can be performed on meshes.

- `MGlobal::SelectionMethod` { `kSurfaceSelectMethod`, `kWireframeSelectMethod` }

Selection methods available in the various selectFromScreen functions.

Modified enums

MFN::Type

- Added: kAnimLayer
- Added: kAsset
- Added: kBlendNodeAdditiveRotation
- Added: kBlendNodeAdditiveScale
- Added: kBlendNodeBase
- Added: kBlendNodeBoolean
- Added: kBlendNodeDouble
- Added: kBlendNodeDoubleAngle
- Added: kBlendNodeDoubleLinear
- Added: kBlendNodeEnum
- Added: kBlendNodeFloat
- Added: kBlendNodeFloatAngle
- Added: kBlendNodeFloatLinear
- Added: kBlendNodeInt16
- Added: kBlendNodeInt32
- Added: kFloatArrayData
- Added: kMembrane
- Added: kMergeVertsToolManip
- Added: kNBase
- Added: kNIdData
- Added: kPassContributionMap
- Added: kPluginManipulatorNode
- Added: kPolySelectEditFeedbackManip

- Added: kPolyToolFeedbackManip
- Added: kPrecompExport
- Added: kRenderPass
- Added: kRenderPassSet
- Added: kStereoCameraMaster
- Added: kTexSmoothManip
- Added: kUint64SingleIndexedComponent
- Added: kWriteToColorBuffer
- Added: kWriteToDepthBuffer
- Added: kWriteToFrameBuffer
- Added: kWriteToLabelBuffer
- Added: kWriteToVectorBuffer

MGlobal::ListAdjustment

- Added: kAddToHeadOfList

MPxManipContainer::baseType

- Added: kCustomManip

MPxNode::Type

- Added: kManipulatorNode

MSceneMessage::Message

- Added: kBeforePluginLoad
- Added: kAfterPluginLoad
- Added: kBeforePluginUnload
- Added: kAfterPluginUnload

MSelectionMask::SelectionType

- Added: kSelectNParticles

Changes to examples

New C++ Example Plug-ins

- closestPointOnNurbsSurfaceCmd
- intersectOnNurbsSurfaceCmd
- lineManip
- lineManipContainer
- listRichSelectionCmd
- manipOverride
- peltOverlapCmd
- richMoveCmd
- splatDeformer
- squareScaleManip
- squareScaleManipContext
- threadedBoundingBox
- threadingLockTests
- XmlGeometryCache

New Python Example Plug-ins

- filmMoveManip
- manipulatorMath

Newly Localized Plug-ins

- AshliShader

Bug fixes

The following is a list of bugs fixed in this release of Maya which are most likely to be of interest to users of the API:

- `MProgressWindow::reserve` crashes in custom translators in prompt/batch mode.
- `MFnMesh::create` fails in the `postConstructor` of a custom `MPxTransform` node.
- Custom imageplanes do not respect the `frameCache` attribute.
- `MGlobal::selectFromScreen` doesn't respect the `listAdjustment` flag.
- Resizing an `MStringArray` to be smaller causes a memory leak.
- Modifying a set generates unexpected calls to topology changed callbacks.
- `MFnMesh::createBlindDataType` does not work from Python.
- Using `setRotationOrder` in the `postConstructor` of an `MPxTransform` does not work.
- `MPxConstraintCommand` only works on the first instance of a target with multiple instances.
- `MFnAnimCurve::addKeys()` leaks memory.
- `MRampAttribute::getEntries()` crashes when it is passed a `NULL MStatus` pointer.
- "Export Selected" exports "requires" statements for plug-ins which are not needed by the selected objects.
- Maya looks in the wrong place for the `MAYA_INSTALL_LOCATION` registry key on Windows.
- `MPxModelEditorCommands` do not properly generate a `requires` statement on save.

